*--第一种方法查询所有员工信息*

select \* from scott.emp

*--第二种方法查询所有信息*

select empno,ename,job,mgr,hiredate,sal,comm,deptno from scott.emp

*--查询部分列*

select empno,ename,job,sal,deptno from scott.emp

*--转正的员工工资加20%*

select empno,ename,sal,sal\*1.2 from scott.emp

*--算一个员工的第一年工资，后半年转正工资加20%*

select empno,ename,(sal+sal\*1.2)\*6 from scott.emp

select comm from scott.emp

*--算一个员工一个月的奖金加工资*

select empno,ename,sal,comm,nvl(sal+comm,sal) from scott.emp

*--算一个员工一年的工资加奖金，nvl(expr1,expr2) 如exp1是空值，则返回exp2,否则返回expr1;*

select ename,sal,comm,nvl((sal+sal\*1.2+2\*comm)\*6,(sal+sal\*1.2)\*6) from scott.emp

*--算一个员工一年的工资加奖金，格式（xx一年的工资为xx）*

select ename ||'一年的总工资为'|| nvl((sal+sal\*1.2+2\*comm)\*6,(sal+sal\*1.2)\*6) from scott.emp

*--查找员工表中一共有哪几种岗位类型*

select distinct job from scott.emp

select count(distinct job) from scott.emp

*--查找员工表、部门表、薪资表的数据*

select \* from scott.dept

select \* from scott.salgrade

select \* from scott.emp

*--查找全班有多少个姓*

create table SCOTT.class3

(

sno char(10) primary key,

name char(10)

)

select \* from scott.class3

select count(distinct substr(Name,0,1)) from scott.class3

*-------------------------------上午-------------------------------------*

*--查询职位为‘salesman’的职员*

select empno,ename,hiredate from emp where job='SALESMAN'

select \* from emp

*--查询年龄在xx年龄以前的员工*

select ename,hiredate from emp where hiredate<'31-12月-85'

*--查询部门号不等于10 的员工*

select deptno,ename from emp where deptno<>10

*--查询出生年在82与85之间的员工*

select ename,hiredate from emp where substr(hiredate,8,9) between '82' and '85'

*--查询工资在30000-50000的员工*

select ename,sal from emp where sal between 3000 and 5000

*--查询部门号等于10或20 的员工*

select ename,deptno from emp where deptno in(10,20)

*--查询职位号为7902，7566，7788的员工*

select empno,ename from emp where empno in(7902,7566,7788)

*--查询W开头的员工号，*

select ename from emp where ename like 'W%'

*--查询倒数第二个字母为W的员工*

select ename from emp where ename like '%T\_'

*--查询奖金为null的员工*

select ename,comm from emp where comm is null

*--查询工资大于2000或者职位为salesman的员工*

select ename,job,sal from emp where sal>2000 or job='SALESMAN'

*--查询工资大于2000而且职位为salesman或manager的员工*

select ename,job,sal from emp where sal>2000 and job in('SALESMAN','MANAGER')

*--查询员工部门在10或20，而且工资为3000到5000的员工*

select ename,deptno,sal from emp where deptno in(10,20) and sal between 3000 and 5000

*--查询员雇佣年为81年，而且工资为3000到5000的员工*

select ename,hiredate,job from emp where substr(hiredate,8,9)='81' and job not like 'SALES%'

*--查询职位为salesman或manager的员工而且部门在10或20*

select ename,job,deptno from emp where job in('SALESMAN','MANAGER') and deptno in(10,20)

and ename like '%A%'

*--查询部门在20或30的员工，根据工资升序排序*

select deptno,ename from emp where deptno in(20,30)

order by sal

*--查询员工工资为2000到3000且部门不等于10的员工，根据部门升序排序，工资降序排序*

select ename,deptno,sal from emp where sal between 2000 and 3000 and deptno<>10

order by deptno , sal desc

*--查询员工在82，83年间的，而且职位名以SALES或MAN开头的*

select ename,hiredate,job from emp

where substr(hiredate,8,9) between '82' and '83'

and job like'SALES%' or job like'MAN%'

order by hiredate desc

*--查询入职时间在82-7-9后，而且工作不为SALESMAN*

select ename,hiredate,job from emp

where hiredate>'09-7月-82'

and job<>'SALESMAN'

*--查询第三个字母为A的员工*

select ename from emp where ename like '\_\_A%'

*--查询除了10，20部门外的员工*

select ename,deptno from emp where deptno not in(10,20)

*--查询部门为30的员工，并以工资升序排序，以名字降序排序*

select ename,deptno,sal from emp

where deptno=30

order by sal desc,ename

*--以部门降序排除，工资升序排序*

select deptno,empno,ename,sal from emp order by deptno desc,sal

*--查询经理级员工*

select ename from emp where mgr is null

*--查询工资大于等于4500而且部门在10或20*

select ename,sal,deptno from emp where sal>=4500

and deptno in(10,20)

*-----------------------------------下午----------------------------------*

*--首字母大写，开始字母为J，A，M*

select initcap(ename) as name,length(ename) as length

from emp

where substr(ename,1,1) in('J','A','M')

order by ename

*--查询ename中含有A和a的员工*

select ename from emp where lower(ename) like '%a%'

select ename from emp where instr(ename,'A')<>0 or instr(ename,'a')<>0

*--查询包含A，部门在10，20，雇佣日期在81年5月后，输出名字长度*

select deptno,hiredate,ename,length(ename) as length

from emp

where deptno in(10,20) and hiredate>'01-5月-81'

and instr(ename,'A')<>0

*--编号，10位左对齐，姓名10位左对齐，工资10位右对齐，不足用‘\*’填充*

select concat(concat(rpad(empno,10,'\*'),rpad(ename,10,'\*')),lpad(sal,10,'\*'))

from emp

*--round四舍五入*

select round(100.456,2)，round(100.456,1),round(100.456,0)

from dual

*--trunc截取*

select trunc(100.456,2),trunc(100.456,1),trunc(100.456,0)

from dual

*--每个员工的入职天数*

select empno,ename,round(sysdate-hiredate,0) as days from emp

select sysdate from dual

*---------01-1月-01-------------*

*--RR*

*--2015,2001*

*--1998,2001*

*--YY*

*--2015，2001*

*--1998，1901*

*---------01-1月-95------------*

*--RR*

*--2015,1995*

*--1998,1995*

*--YY*

*--2015,2095*

*--1998,1995*

*--查询服务器当前时间*

select sysdate from dual

*--查询部门10，20的员工截止到2000年1月1，工作了多少个月，入职的月份*

select ename,months\_between('01-1月-2000',hiredate) as months,

extract(month from hiredate)

from emp

where deptno in(10,20)

*--查询不是manager的员工入职、转正等时间*

select ename,hiredate 入职时间,add\_months(hiredate,6) 转正时间,

next\_day(hiredate,'星期一'） 入职后第一个星期一，

last\_day(hiredate) 入职当月最后一天

from emp

where job='MANAGER'

*--显示当前服务器系统时间*

select to\_char(sysdate,'yyyy-month-dd HH24:MI:ss ')

from dual

*--显示ename，hireman和雇员开始工作日是星期几（day）*

select ename,hiredate,to\_char(hiredate,'day')

from emp

*--查询员工姓名，工资，格式化的工资*

select ename,sal,to\_char(sal,'999,999,99') from emp

*--把字符串2015-3月18 13：13：13转换位日期格式*

select to\_date('2015-3月-18 13:13:13','yyyy/month/dd hh24:mi:ss'),

(sysdate-to\_date('2015-3月-18','yyyy/month/dd')) as dayfromnow

from dual

*--计算2000年1月1日到现在有多少月，多少周（四舍五入）。*

select months\_between(sysdate,to\_date('2000/1月/01','yyyy/month/dd')),

round(sysdate-to\_date('2000/1月/01','yyyy/month/dd'),0)

from dual

*--查询员工ENAME的第三个字母是A的员工的信息（使用2个函数）。*

select ename from emp

where instr(ename,'A',3,1)=3

select ename from emp

where substr(ename,3,1)='A'

*--使用trim函数将字符串'hello'、'Hello'、'bllb'、'hello,分别处理得到下列字符串ello、Hello、11、hello。*

select trim('h' from 'hello'),trim(' ' from 'Hello ') ,

trim(both 'b' from 'bllb'),trim(' ' from 'hello ')

from dual

*--将员工工资按如下格式显示：123,234.00RMB。*

select concat(to\_char(sal,'999,999.00'),'RMB') from emp

*--查询员工的姓名及其经理编号，要求对于没有经理的显示"No Manager"字符串。*

select ename,

(case mgr

when mgr then to\_char(mgr)

else 'No manager'

end

)mgr

from emp

*--将员工的参加工作日期按如下格式显示：月份/年份。*

select to\_char(hiredate,'yyyy/month') from emp

*--在员工表中查询出员工的工资，并计算应交税款：如果工资小于1000,*

*--税率为0,如果工资大于等于1000并小于2000,*

*--税率为10%,如果工资大于等于2000并小于3000,税率为*

*--15%,如果工资大于等于3000,税率为20%。*

select ename,sal,

(case

when sal<1000 then 0

when sal<2000 then 0.1\*sal

when sal<3000 then 0.15\*sal

else 0.2\*sal

end

)shui

from emp

*--创建一个查询显示所有雇员的ename和sal。格式化sal为15个字符长度，用$左填充，列标签SALARY。*

select ename,lpad(sal,15,'$') as salary

from emp

*--------------------------多表连接-------------------*

*--写一个查询，显示所有员工姓名，部门编号， 部门名称 部门名称*

select ename,emp.deptno,dname

from emp,dept

where emp.deptno=dept.deptno

*--写一个查询，显示所有工作在CHICAGO并且奖 金不为空的员工姓名工作地点奖金*

select \* from emp

select \* from dept

select \* from salgrade

select \* from bonus

select ename,loc,comm

from emp,dept

where emp.deptno=dept.deptno

*--写一个查询，显示所有姓名中含有A字符的员 工姓名工作地点 工姓名，工作地点*

select ename,loc

from emp,dept

where instr(ename,'A')<>0 and emp.deptno=dept.deptno

*--查询每个员工的编号，姓名，工资，工资等级，所在工 作城市，按照工资等级进行升序排序*

select empno,ename,sal,grade,loc

from emp,dept,salgrade

where emp.deptno=dept.deptno and

emp.sal between salgrade.losal and salgrade.hisal

order by grade

*--查询每个员工的姓名和直接上级姓名？*

select a.ename,b.ename

from emp a,emp b

where a.mgr=b.empno(+)

*--查询所有工作在NEW YORK和CHICAGO的员工姓 名，员工编号，*

*--以及他们的经理姓名，经理编号，并按照员工编号排序*

select a.ename,a.empno,b.ename,b.empno

from emp a,emp b,dept d

where a.mgr=b.empno(+)

and d.deptno=a.deptno and d.loc in('NEW YORK','CHICAGO')

order by a.empno

*--查询所有员工编号，姓名，部门名称，包括没 部有 部门的员工也要显示出来*

select empno,ename,dname

from emp e,dept d

where e.deptno=d.deptno(+)

*--交叉连接*

select emp.empno,emp.ename,emp.sal,emp.deptno,dept.loc

from emp cross join dept

*--自然连接*

select empno,ename,sal,deptno,loc

from emp

natural join dept

*--using*

select e.ename,e.ename,e.sal,deptno,d.loc

from emp e join dept d using(deptno)

where deptno=20;

*--on*

select e.empno,e.ename,e.deptno,d.loc

from emp e

join dept d

on e.deptno=d.deptno

*--左外连接*

select e.ename,e.deptno,d.loc

from emp e

left outer join dept d

on e.deptno=d.deptno

*--右外连接*

select e.ename,e.deptno,d.loc

from emp e

right outer join dept d

on e.deptno=d.deptno

*--全外连接*

select e.ename,e.deptno,d.loc

from emp e

full outer join dept d

on e.deptno=d.deptno

*--创建个员工表和部门表的交叉连接*

select ename,e.deptno,loc

from emp e cross join dept

*--使用自然连接，显示入职日期在80年5月1日之后的员工 姓名部门名称入职日期*

select ename,dname,hiredate

from emp natural join dept

where emp.hiredate>'01-5月-80'

*--使用USING子句，显示工作在CHICAGO的员工姓名，部门 名称，工作地点*

select e.ename,d.dname,d.loc

from emp e join dept d using(deptno)

where loc='CHICAGO'

select \* from dept

*--使用ON子句，显示工作在CHICAGO的员工姓名，部门名 称，工作地点，薪资等*

select e.ename,d.dname,d.loc,sal

from emp e join dept d

on loc='CHICAGO' and e.deptno=d.deptno

*--使用左连接，查询每个员工的姓名，经理姓名，没有经 理的King也要显示出来*

select a.ename,b.ename

from emp a left outer join emp b

on a.mgr=b.empno

*--使用右连接，查询每个员工的姓名，经理姓名，没有经 理的King也要显示出来。*

select a.ename,b.ename

from emp b right outer join emp a

on a.mgr=b.empno

*--显示员工SMITH的姓名，部门名称，直接上级名*

*--等值连接*

select a.ename,b.ename,d.dname

from emp a,emp b,dept d

where a.mgr=b.empno(+) and a.deptno=d.deptno

and a.ename='SMITH'

*--显示员工姓名，部门名称，工资，工资级别，要求工资 级别大于4级*

select ename,dname,sal,grade

from emp e,dept d,salgrade s

where e.deptno=d.deptno and

e.sal between s.losal and s.hisal

and s.grade>4

*--显示员工KING和FORD管理的员工姓名及其经理姓名*

select a.ename,b.ename

from emp a,emp b

where a.mgr =b.empno and b.ename in('KING','FORD')

*--显示员工姓名，参加工作时间，经理名，参加工作时间,要求参加时间比经理早*

select a.ename,a.hiredate,b.ename

from emp a,emp b

where a.mgr=b.empno and a.hiredate<b.hiredate

*----------------------------多行函数------------------------*

*--查询部门20的员工，每个月的工资总和及平均 工资 工资。*

select sum(sal),avg(sal)

from emp

where deptno=20

*--查询工作在CHICAGO的员工人数，最高工资及 最低工资 最低工资*

select count(empno),max(sal),min(sal)

from emp,dept

where dept.loc='CHICAGO' and dept.deptno=emp.deptno

*--查询员工表中一共有几种岗位类型*

select count(distinct job) from emp

*--查询每个部门的平均工资*

select avg(sal)

from emp

group by deptno

*--查询每个部门每个岗位的工资总和*

select deptno,job,sum(sal)

from emp

group by deptno,job

*--查询每个部门的部门编号，部门名称，部门人 数最高工资最低工资工资总和平均工资*

select e.deptno,d.dname,count(e.deptno),

max(sal),min(sal),sum(sal),avg(sal)

from emp e,dept d

where e.deptno=d.deptno

group by e.deptno,d.dname

*--2.查询每个部门，每个岗位的部门编号，部门名 称，*

*--岗位名称，部门人数，最高工资，最低工资 工资总和平均工资 ，*

*--工资总和，平均工资。*

select d.deptno,dname,job,count(e.empno),

max(sal),min(sal),sum(sal),avg(sal)

from dept d,emp e

where d.deptno=e.deptno

group by d.deptno,dname,job

*--查询每个经理所管理的人数，经理编号，经理姓名要求包括没有经理的人员信息 姓名，要求包括没有经理的人员信息*

select count(a.empno),b.empno,b.ename

from emp a,emp b

where a.mgr=b.empno(+)

group by b.empno,b.ename

*--查询部门人数大于2的部门编号，部门名称， 部门人数 部门人数。*

select dept.deptno,dname,count(empno)

from dept,emp

where dept.deptno=emp.deptno

group by dept.deptno,dname

having count(empno)>2

*--查询部门平均工资大于2000，且人数大于2的 部门编号部门名称部门人数部门平均工资*

*--部门编号，部门名称，部门人数，部门平均工资 ，并按照部门人数升序排序*

select d.deptno,count(e.empno),dname,avg(e.sal)

from emp e, dept d

where e.deptno=d.deptno

group by d.deptno,dname

having count(e.empno)>2

order by count(e.empno)

*--查询部门平均工资在2500元以上的部门名称及平均工资*

select dname,round(avg(sal),0),d.deptno

from emp e,dept d

where e.deptno=d.deptno

group by d.deptno,dname

having avg(sal)>2500

*--查询员工岗位中不是以“SA”开头并且平均工资在2500*

*--查询员岗位中不是以开头并且平均资在 元以上的岗位及平均工资，并按平均工资降序排序。*

select job,round(avg(sal),0)

from emp

where job not like 'SA%'

group by job

having avg(sal)>2500

order by avg(sal) desc

*--.查询部门人数在2人以上的部门名称、最低工资、最高 工资并对求得的工资进行四舍五入*

*--到整数位 工资,并对求得的工资进行四舍五入到整数位。*

select dname,round(min(sal),0),round(max(sal),0)

from emp e, dept d

where e.deptno=d.deptno

group by dname

having count(e.empno)>2

*--查询岗位不为SALESMAN，工资和大于等于2500的岗位及 每种岗位的工资和*

select job,sal,sum(sal)

from emp

where job<>'SALESMAN'

group by job,sal

having sum(sal)>2500

*--显示经理号码和经理姓名，这个经理所管理员工的最低 工资，没有经理的KING也要显示，*

*--不包括最低工资小于 3000的，按最低工资由高到低排序。*

select b.empno,b.ename,min(a.sal)

from emp a,emp b

where a.mgr=b.empno(+)

group by b.empno,b.ename

having min(a.sal)>=3000

order by min(a.sal) desc

*--写一个查询，显示每个部门最高工资和最低工资的差额*

select max(sal),min(sal),max(sal)-min(sal)

from emp

group by deptno

*--查询入职日期最早的员工姓名，入职日期*

select ename,hiredate

from emp

where hiredate=(select min(hiredate)

from emp)

*--查询工资比SMITH工资高并且工作地点在 CHICAGO的员工姓名，工资，部门名*

select ename,sal,loc

from emp e, dept d

where e.deptno=d.deptno

and d.loc='CHICAGO'

and sal>(select sal

from emp

where ename='SMITH')

*--查询入职日期比10部门入职日期最早的员工还 要早的员工姓名，入职日期*

select ename,hiredate

from emp

where hiredate<(select min(hiredate)

from emp

where deptno=10)

*-- 查询部门人数大于所有部门平均人数的的部门 编号，部门名称，部门人数*

select d.deptno,dname,count(empno)

from dept d,emp e

where d.deptno=e.deptno

group by d.deptno,dname

having count(empno)>(select avg(count(empno))

from emp

group by deptno

)

*--查询入职日期比10部门任意一个员工晚的员工 姓名入职日期不包括10部门员*

select ename,hiredate

from emp

where deptno<>10

and hiredate>any(select hiredate

from emp

where deptno=10)

*--查询入职日期比10部门所有员工晚的员工姓名 入职日期不包括10部门员工 、*

*-- 入职日期，不包括10部门员工*

select ename,hiredate

from emp

where deptno<>10

and hiredate>all(select hiredate

from emp

where deptno=10)

*--查询职位和10部门任意一个员工职位相同的员 工姓名职位不包括10部门员工*

select ename,job

from emp

where deptno<>10

and job in(select job

from emp

where deptno=10)

*--查询职位及经理和10部门任意一个员工职位及*

*--经理相同的员工姓名职位不包括10部门员工*

select ename,job

from emp

where job in(select job

from emp

where deptno=10)

and mgr in(select mgr

from emp

where deptno=10)

and deptno<>10

*--查询职位及经理和10部门任意一个员工职位或*

*--经理相同的员工姓名职位不包括10部门员工*

select ename,job

from emp

where deptno<>10

and (job in(select job

from emp

where deptno=10)

or mgr in(select mgr

from emp

where deptno=10)

)

*--查询比自己职位平均工资高的员工姓名、职位 部门名称职位平均工资 ，*

*--部门名称，职位平均工资*

select ename,dname,avg(sal)

from emp a,dept d,

(select job,avg(sal) salary

from emp

group by job) b

where a.job=b.job and a.deptno=d.deptno and a.sal>b.salary

group by ename,dname

*--查询职位和经理同员工SCOTT或BLAKE完全相同 的员工姓名职位不包括SCOOT和BLAKE本人*

*--的员工姓名、职位，不包括SCOOT和BLAKE本人*

select a.ename, a.job,a.mgr

from emp a,(select job,mgr

from emp

where ename in('SCOTT','BLAKE')) b

where a.job=b.job and a.mgr=b.mgr

and a.ename not in('SCOTT','BLAKE')

*--查询不是经理的员工姓名*

select ename,a.job

from emp a,(select job from emp

where job='MANAGER') b

where a.job<>b.job

*--rownum,输出的顺序号*

SELECT rownum,empno, ename,job

from emp

order by empno desc

*--查询入职日期最早的前5名员工姓名，入职日期。*

select ename，hiredate

from emp

where rownum<=5

order by hiredate

*--查询工作在CHICAGO并且入职日期最早的前2名 员工姓名入职日期 员工姓名，入职日期*

select ename,hiredate

from emp e, dept d

where e.deptno=d.deptno and loc='CHICAGO' and rownum<=2

order by hiredate

*--按照每页显示5条记录，分别查询第1页，第2 页第3页信息要求显示员工姓名入职日期 页，*

*--第3页信息，要求显示员工姓名、入职日期 、部门名称。*

select ename,hiredate,b.dname

from (select rownum rn,dname,hiredate,ename

from emp e,dept d

where e.deptno=d.deptno

) b

where rn<=3\*5 and rn>2\*5

select ename,hiredate,b.dname

from (select rownum rn,dname,hiredate,ename

from emp e,dept d

where e.deptno=d.deptno

and rownum<=3.5\*5

) b

where rn>2\*5

*--按照每页显示5条记录，分别查询工资最高的 第1页第2页第3页信息要求*

*--显示员工姓名 第1页，第2页，第3页信息，*

*--要求显示员工姓名 、入职日期、部门名称、工资*

select ename,hiredate,dname,salary

from (select rownum rn,ename,hiredate,dname,max(sal) salary

from emp e,dept d

where e.deptno=d.deptno

group by rownum,ename,hiredate,dname

order by salary desc

)b

where rn<=3\*5 and rn>2\*5

*--查询工资高于编号为7782的员工工资，*

*--并且和7369号员 工从事相同工作的员工的编号、姓名及工资*

select empno,ename,sal

from emp

where job=(select job

from emp

where empno=7369)

and sal>(select sal

from emp

where empno=7782)

*--查询工资最高的员工姓名和工资*

select ename,sal

from (select ename,sal

from emp

order by sal desc

)

where rownum<=1

*--查询部门最低工资高于10号部门最低工资的部门的编号、*

*-- 名称及部门最低工资*

select e.deptno,dname,min(sal)

from emp e,dept d

where e.deptno=d.deptno

group by e.deptno,dname

having min(sal)<(select min(sal)

from emp

where deptno=10)

*--查询员工工资为其部门最低工资的员工的编号和姓名及工资*

select e.empno,e.ename,b.sal

from emp e,(select deptno,min(sal) sal

from emp

group by deptno

)b

where e.sal=b.sal and e.deptno=b.deptno

*--显示经理是KING的员工姓名，工资。*

select ename,sal

from emp a,(select empno

from emp

where ename='KING') b

where a.mgr=b.empno

*--显示比员工SMITH参加工作时间晚的员工姓名，工资，参加工作时间。*

select ename,sal,hiredate

from emp

where hiredate>(select hiredate

from emp

where ename='SMITH')

*--使用子查询的方式查询哪些职员在NEW YORK工作。*

select empno,ename,loc

from emp e,dept d

where e.deptno=d.deptno

and loc in(select loc

from emp

where loc='NEW YORK')

*--写个查询显示和员工SMITH工作在同个部门的员工 姓名，*

*--雇用日期，查询结果中排除SMITH*

select ename,hiredate

from emp

where deptno=(select deptno

from emp

where ename='SMITH')

and ename<>'SMITH'

*--写一个查询显示其工资比全体职员平均工资高的员工编 号、姓名。*

select empno,ename

from emp,(select avg(sal) asal

from emp)b

where sal>b.asal

*--显示所有工作在RESEARCH部门的员工姓名，职位*

select ename,job,dname

from emp e,dept d

where e.deptno=d.deptno

and dname='RESEARCH'

*--查询每个部门的部门编号、平均工资，*

*--要求部门的平 均工资高于部门20的平均工资*

select deptno,avg(sal)

from emp

having avg(sal)>(select avg(sal)

from emp

where deptno=20)

group by deptno

*--查询大于自己部门平均工资的员工姓名，工资，*

*--所在 部门平均工资，高于部门平均工资的额度。*

select ename,sal,b.avgsal,a.deptno

from emp a,(select deptno,round(avg(sal),0) avgsal

from emp

group by deptno)b

where a.deptno=b.deptno

and a.sal>b.avgsal

*--列出至少有一个雇员的所有部门*

select deptno,count(\*)

from emp

having count(\*)>1

group by deptno

*--列出薪金比"SMITH"多的所有雇*

select empno,ename,sal

from emp

where sal>(select sal

from emp

where ename='SMITH')

*--列出入职日期早于其直接上级的所有雇员*

select empno,ename,hiredate,job

from emp e

where hiredate <any(select b.hiredate

from emp a,emp b

where a.mgr=b.empno

)

*--找员工姓名和直接上级的名字*

select a.ename,b.ename

from emp a,emp b

where a.mgr=b.empno

*--显示部门名称和人数*

select dname,count(empno)

from emp e, dept d

where e.deptno=d.deptno

group by e.deptno,dname

*--显示每个部门的最高工资的员工*

select empno,ename,b.sal,a.deptno

from emp a join (select max(sal) sal,deptno

from emp

group by deptno

)b

on a.deptno=b.deptno

where a.sal=b.sal

order by deptno

*--显示出和员工号7369部门相同的员工姓名，工资*

select ename,sal

from emp

where deptno=(select deptno

from emp

where empno=7369)

*--显示出和姓名中包含"W"的员工相同部门的员工姓名*

select ename

from emp

where deptno in(select deptno

from emp

where ename like '%W%')

*--显示出工资大于平均工资的员工姓名，工资*

select ename,sal

from emp

where sal>(select avg(sal)

from emp)

*--显示出工资大于本部门平均工资的员工姓名工资*

select ename,sal

from emp

where sal>any(select avg(sal)

from emp

group by deptno)

*--显示每位经理管理员工的最低工资，及最低工资者的姓名*

select empno,ename,a.sal,a.mgr

from emp a,(select min(a.sal) sal,b.empno bempno

from emp a,emp b

where a.mgr=b.empno

group by b.empno

)b

where a.sal=b.sal and a.mgr=b.bempno

*--显示比工资最高的员工参加工作时间晚的员工姓名，参加 工作时间*

select ename,hiredate,sal

from emp

where hiredate>(select hiredate

from emp

where sal=(select max(sal)

from emp)

)

*--显示出平均工资最高的的部门平均工资及部门名称*

select dname,max(b.s)

from (select deptno,round(avg(sal)) s

from emp

group by deptno)b ,

dept a

where a.deptno=b.deptno and rownum<2

group by dname

*--union,联合后去除重复行，按第一列排序*

*--查询编号为7839的员工当前工资、岗位 及历史工资、岗位？*

select empno, sal,job from emp where empno=7839

union select empno,sal,job from jobhistory where empno=7839

*--union all,不去重复行，不排序*

select empno, sal,job from emp where empno=7839

union all select empno,sal,job from jobhistory where empno=7839

*--分别使用联合运算及完全联合运算完成，按照 时间升序顺序查询员工7839的工作岗位列表*

*--时间升序顺序，查询员工7839的工作岗位列表。*

*--union*

select empno,job,hiredate from emp where empno=7839

union select empno,job,begindate

from jobhistory where empno=7839

order by hiredate

*--union all*

select empno,job,hiredate from emp where empno=7839

union all select empno,job,begindate

from jobhistory where empno=7839

order by hiredate

*--使用多表连接，查询每个部门的部门编号，部*

*--门人数没有人数的部门显示0 门人数*

SELECT d.deptno,COUNT (empno)

FROM emp e,dept d

WHERE e.deptno(+)=d.deptno

GROUP BY d.deptno

select \* from dept

select distinct deptno from emp

*--使用联合运算，查询每个部门的部门编号，部门人数没有人数的部门显示0*

SELECT d.deptno,COUNT (empno)

FROM emp e,dept d

WHERE e.deptno=d.deptno

GROUP BY d.deptno

UNION

SELECT deptno,0

FROM dept

WHERE deptno NOT IN(SELECT deptno FROM emp

where deptno is not null

)

*--使用联合运算，查询10号部门及20号部门的员工姓名部门编号*

select ename,deptno

from emp

where deptno=10

union

select ename,deptno

from emp

where deptno=20

*--练习一5*

select deptno,loc,null as ename,null as hiredate

from dept

union

select d.deptno,null,ename,hiredate

from emp e,dept d

where e.deptno=d.deptno

*--查询哪些员工做过岗位调动？*

select empno from emp

intersect

select empno from jobhistory

*--查询哪些员工没做过岗位调动*

select empno from emp

minus

select empno from jobhistory

*--用集合运算，列出不包含job为SALESMAN的部门 的部门号 的部门号*

select deptno

from dept

minus

select deptno

from emp

where job='SALESMAN'

*--.写一个联合查询，列出下面的信息： EMP表中所有雇员的名字和部门编号,*

*--不管他们是 表中所有雇员的名字和部门编号,不管他们是 否属于任何部门。*

*-- DEPT表中的所有部门编号和部门名称，不管他们是否有员工*

select ename,deptno,null

from emp

union

select null,deptno,dname

from dept

order by deptno,ename desc

*--用集合运算查询出职位为SALESMAN和部门编号为*

*-- 10的人员编号姓名职位不排除重复结果*

select deptno,ename,job

from emp

where deptno=10

union all

select deptno,ename,job

from emp

where job='SALESMAN'

*--用集合查询出部门为10和20的所有人员编号、*

*--姓 名所在部门名称*

select empno,ename,dname,d.deptno

from emp e,dept d

where e.deptno=10 and e.deptno=d.deptno

union

select empno,ename,dname,d.deptno

from emp e,dept d

where e.deptno=20 and e.deptno=d.deptno

order by deptno

*--查询比本部门平均薪水高的员工姓 名，薪水*

SELECT empno,ename,sal

FROM emp e ,(SELECT deptno,avg(sal) avgsal

FROM emp

GROUP BY deptno) d

WHERE e.deptno =d.deptno

AND e.sal >d.avgsal;

*--相关子查询，父查询中的行每被处理一次，子查询就执行一次*

select ename，sal

from emp e

where sal>(select avg(sal)

from emp

where deptno=e.deptno)

*--查询比所在职位平均工资高的员工姓名，职位*

select ename,job

from emp e

where sal>(select avg(sal)

from emp

where job=e.job)

*--查询工资为其部门最低工资的员工编号，姓名 ，工资*

select empno,ename,sal

from emp e

where sal=(select min(sal)

from emp

where deptno=e.deptno)

*--查询所有部门名称和人数*

select dname,(select count(empno)

from emp

where deptno=d.deptno)

from dept d

*--查询哪些员工是经理?*

select empno,ename,mgr

from emp d

where 0<(select count(empno)

from emp

where mgr=d.empno)

select e.empno,ename,e.job

from emp e

where 2<=(select count(\*)

from jobhistory

where empno=e.empno)

*--查询所有雇员编号，名字和部门名字*

select empno,ename,dname

from emp e,dept d

where e.deptno=(select deptno

from dept

where deptno=d.deptno)

*--查询哪些员工不是经理*

select empno,ename,job

from emp e

where (select count(empno)

from emp

where mgr=e.empno)=0

*--查询哪些员工是经理*

select empno,ename,job

from emp e

where (select count(empno)

from emp

where mgr=e.empno)>0

*--查询每个部门工资最低的两个员工编号，姓名， 工资。*

select deptno, ename, sal

from emp e

where empno in(

select empno

from (select \*

from emp

order by sal )b

where e.deptno=b.deptno

and rownum<3

)

order by deptno,sal

*--列出至少有一个雇员的所有部门名称*

select deptno,dname

from dept d

where exists (select empno

from emp

where deptno=d.deptno)

*--列出一个雇员都没有的所有部门名称*

select deptno,dname

from dept d

where not exists (select empno

from emp

where deptno=d.deptno)

*--查询薪水多于他所在部门平均薪水的雇员名字 ，部门号*

select ename,sal,deptno

from emp e

where sal>(select avg(sal)

from emp

where deptno=e.deptno)

*--查询员工姓名和直接上级的名字*

select a.ename,b.ename,a.job,b.job

from emp a,emp b

where a.mgr in(select empno

from emp

where a.mgr=b.empno)

*--查询每个部门工资最高的员工姓名，工资*

select ename,sal,deptno

from emp e

where sal=(select max(sal)

from emp

where deptno=e.deptno

group by deptno)

*--查询每个部门工资前两名高的员工姓名，工资*

*--1*

select ename,sal,deptno

from emp e

where empno in(select empno

from (select \*

from emp

order by sal desc)b

where b.deptno=e.deptno

and rownum<3)

order by deptno

*--2 子查询的表和嵌套表属性个数要一样*

select e.ename,e.sal,e.deptno

from (select sal,deptno,empno,ename

from emp) e

where empno in(select empno

from (select sal,deptno,empno,ename

from emp

order by sal desc)b

where b.deptno=e.deptno

and rownum<3)

order by e.deptno

*--遍历树，从底向上*

select empno,ename,job,mgr

from emp

start with empno=7876

connect by prior mgr=empno

*--从顶向下*

select empno,ename,job,mgr

from emp

start with empno=7876

connect by prior empno=mgr

*--从顶向下*

select ename||'reports to'||

prior ename "walk top down"

from emp

start with ename='KING'

connect by prior empno=mgr;

*--用LEVEL伪列将行分等级*

select level,empno,ename,job,mgr

from emp

start with ename='KING'

connect by prior empno=mgr

*--和LPAD生成分级报告*

select lpad(ename,length(ename)+(level\*2)-2,'\*')

as org\_chart

from emp

start with ename='KING'

connect by prior empno=mgr

*--.产生一个报告显示BLAKE的所有下级*

*--（包括直接和间接下 级）雇员的名字、薪水和部门号*

select level,ename||' reports to '||prior ename "name" ,

sal,deptno

from emp

start with ename='BLAKE'

connect by prior empno=mgr *--从顶向下*

*--创建一个报告显示对于雇员SMITH 经理的层次，包括级 别和姓名，首先显示他的直接经理*

select level,ename,prior ename

from emp

start with ename='SMITH'

connect by prior mgr=empno *--从底向上*

*--创建个缩进报告显示经理层次，从名字为KING的雇员 开始，显示雇员的名字、经理ID和部门ID。*

select lpad（ename,length(ename)+(level),'\*'),prior empno,deptno

from emp

start with ename='KING'

connect by prior empno=mgr *--从顶向下*

*--产生一个公司组织图表显示经理层次。从最顶级的人开始 ，*

*--排除所有job为CLERK的人，还要排除FORD和那些对FORD报告的雇员*

select prior ename,job,ename

from emp

start with ename='KING'

connect by prior empno=mgr

and job<>'CLERK' and prior ename<>'FORD'

*--向部门表新增一个部门，部门编号为50，部门 名称为HR，工作地点为SY*

insert into dept values(50,'HR','SY')

*--向部门表新增一个部门，部门编号为60，部门 名称为MARKET*

insert into dept values(60,'MARKET',null)

select \* from dept

*--向员工表中新增一个员工，员工编号为8888，姓 名为BOB，岗位为CLERK，*

*--经理为号7788，入职日期 为1985-03-03，薪资3000，奖金和部门为空。*

insert into emp values(8888,'BOB','CLERK',7788,

to\_date('3月 3,1985','mon DD,yyyy'),3000,null,null)

select TO\_DATE('2月 3,1997', 'MON DD, YYYY') from dual

select \* from emp

*--创建 emp\_back表*

CREATE TABLE emp\_back as

SELECT \* FROM EMP WHERE 1=0*--复制表结构*

*--把emp表中入职日期大于1982年1月1日之前的员 工信息复制到emp\_back表中。*

insert into emp\_back

select \*

from emp

where hiredate>'01-1月-1982'

select \* from emp\_back

*--修改部门20的员工信息，把82年之后入职的员 工入职日期向后调整10天*

update emp

set hiredate=hiredate+10

where deptno=20

and substr(hiredate,8,9)>82

select \* from emp

*--修改奖金为null的员工，奖金设置为0*

update emp

set comm=0

where is null

*--修改工作地点在NEW YORK或CHICAGO的员工工 资，工资增加500*

update emp

set sal=sal+500

where job in('NEW YORK','CHICAGO')

*--删除经理编号为7566的员工记录*

delete from emp where mgr=7566

*--删除工作在NEW YORK的员工记录*

delete from emp

where deptno=(select deptno

from dept

where loc='NEW YORK')

*--删除工资大于所在部门平均工资的员工记录*

delete from emp e

where sal>(select avg(sal)

from emp

where deptno=e.deptno)

select \* from emp

savepoint beforedelete

rollback to beforedelete

rollback

create table test(

id number,

name char(10))

*--.test表为空表，分析如下语句操作后，最后 test表的状态。*

*--最后只有id=1的一条语句，不能回滚*

select \* from test

INSERT INTO test(id,name) values(1, 'a')

INSERT INTO test(id,name) values(2, 'b')

SAVEPOINT s1

INSERT INTO test(id,name) values(3, 'c')

INSERT INTO test(id,name) values(4, 'd')

DELETE FROM test WHERE id in (1,3);

ROLLBACK TO s1

DELETE FROM test WHERE id in (2,4)

COMMIT

ROLLBACk

create table student (*--学生表*

xh char(4),*--学号*

xm varchar2(10),*--姓名*

sex char(2),*--性别*

birthday date,*--出生日期*

sal number(7,2), *--奖学金*

studentcid number(2) *--学生班级号 )*

)

Create table class ( *--班级表*

classid number(2), *--班级编号*

cname varchar2(20),*--班级名称*

ccount number(3) *--班级人数 )*

)

*--（1）添加三个班级信息为：1，JAVA1班，null*

*--2，JAVA2班，null*

*--3，JAVA3班，null*

insert into class values(1,'JAVA1班',null)

insert into class values(2,'JAVA2班',null)

insert into class values(3,'JAVA3班',null)

*--添加学生信息如下：‘A001’,‘张三’,‘男’,‘01-5月05’,100,1*

insert into student values('Aoo1','张三','男','01-5月05',100,1)

*--添加学生信息如下：'A002','MIKE','男','1905-05-06',10*

insert into student values('A002','MIKE','男','05-6月05',10,null )

*--插入部分学生信息： 'A003','JOHN','女*

insert into student(xh,xm,sex) values( 'A003','JOHN','女')

*--将A001学生性别修改为'女‘*

update student

set sex='女'

where xh='A001'

*--将A001学生信息修改如下：性别为男，生日设置为1980-04-01*

update student

set sex='男' , birthday='80-4月-01'

where xh='A001'

*--将生日为空的学生班级修改为Java3班*

update student

set studentcid=(select classid

from class where cname='JAVA3班')

where birthday is null

*--请使用一条SQL语句，使用子查询，更新班级表中每个班级的人数字段*

update class c

set ccount=(select count(xh)

from student

where c.classid=studentcid

)

select \* from class

select \* from student

*--*

CREATE TABLE copy\_emp (

empno number(4),

ename varchar2(20),

hiredate date default sysdate ,

deptno number(2),

sal number(8,2))

*--在表copy\_emp中插入数据，要求sal字段插入空值，部门号50 ，*

*--参加工作时间为2000年1月1日，其他字段随意*

select \* from copy\_emp

insert into copy\_emp values(50,'COCO','01-1月-00','20',null)

*--在表copy\_emp中插入数据，要求把emp表中部门号为10号部门 的员工信息插入*

insert into copy\_emp

select empno , ename,hiredate , deptno ,sal

from emp

where deptno=10

*--修改copy\_emp表中数据，要求10号部门所有员工涨20%的工资*

update copy\_emp

set sal=sal\*1.2

where deptno=10

*--修改copy\_emp表中sal为空的记录，工资修改为平均工资*

update copy\_emp

set sal=(select avg(sal)

from emp）

where sal is null

*--通过子查询的方式创建一个表dept10,该表保 存10号部门的员工数据*

create table dept10

as

select \* from dept where deptno=10

select \* from dept10

*--在员工表中添加一个性别列，列名为gender， 类型为h(2)默认值为“男”*

select \* from emp

alter table emp add gender char(2) default '男'

*--修改员工表中性别列的数据类型为char(4)*

alter table emp modify (gender char(4))

*--修改员工表中性别列的默认值为“女*

alter table emp modify (gender default '女')

*--.删除员工表中的性别列*

alter table emp drop column gender

create table temp(

length char(2000),*--如存‘张三’，实际长度为4 --(1)最大2000个字节定长字符串*

name char(10),*--如存‘张三’，实际长度还是10 --(2)如果输入‘张三’ 后添空格6个*

gender char(5),

constraint checkgender check(gender='男' or gender='女'),*--(3)性别输入'男'或'女’*

length1 varchar(4000),*--(4)最大4000个字节变长字符串*

name1 varchar(10),*--(5)如果在数据库中输入'张三'则显示数据'张三’*

number1 number(10,2),*--(6)表示数字范围为10的125次方到10的126次方可表示整数和小数*

number2 number(4),*--(7)最大表示4位整数-9999到9999 (7)最大表示4位整数9999 到9999*

number3 number(5,2),*--(8)表示5位有效数字2位小数的一个小数-999.99 到999.99*

date1 date,*--包括年月日时分秒*

date2 timestamp,*--9i之后新增的，精度比DATE更高一些，可以精确到毫秒*

blob1 blob*--最大可存储4G二进制的数据，可以存放图片，声音，文件*

)

select \* from temp

drop table temp

*--.创建表date\_test,包含列d，类型为date型。试向 datetest表中插入两条记录，*

*--一条当前系统日期记录，一条记录为“1998-08-18”。*

create table date\_test(

d date

)

insert into date\_test values(sysdate)

select \* from date\_test

insert into date\_test values(to\_date('1998-8月-08','yyyy/month/dd'))

*--创建与dept表相同表结构的表dtest，将dept表中部门编 号在40之前的信息插入该表*

create table dtest as

select \* from dept where deptno=40

select \* from dtest

select \* from student

alter table student modify(sex default '女'）

alter table student modify(xm char(10))

*--通过子查询的方式创建一个表dept10,该表保 存10号部门的员工数据*

create table dept10

as

select \* from dept where deptno=10

select \* from dept10

*--在员工表中添加一个性别列，列名为gender， 类型为h(2)默认值为“男”*

select \* from emp

alter table emp add gender char(2) default '男'

*--修改员工表中性别列的数据类型为char(4)*

alter table emp modify (gender char(4))

*--修改员工表中性别列的默认值为“女*

alter table emp modify (gender default '女')

*--.删除员工表中的性别列*

alter table emp drop column gender

create table temp(

length char(2000),*--如存‘张三’，实际长度为4 --(1)最大2000个字节定长字符串*

name char(10),*--如存‘张三’，实际长度还是10 --(2)如果输入‘张三’ 后添空格6个*

gender char(5),

constraint checkgender check(gender='男' or gender='女'),*--(3)性别输入'男'或'女’*

length1 varchar(4000),*--(4)最大4000个字节变长字符串*

name1 varchar(10),*--(5)如果在数据库中输入'张三'则显示数据'张三’*

number1 number(10,2),*--(6)表示数字范围为10的125次方到10的126次方可表示整数和小数*

number2 number(4),*--(7)最大表示4位整数-9999到9999 (7)最大表示4位整数9999 到9999*

number3 number(5,2),*--(8)表示5位有效数字2位小数的一个小数-999.99 到999.99*

date1 date,*--包括年月日时分秒*

date2 timestamp,*--9i之后新增的，精度比DATE更高一些，可以精确到毫秒*

blob1 blob*--最大可存储4G二进制的数据，可以存放图片，声音，文件*

)

select \* from temp

drop table temp

*--.创建表date\_test,包含列d，类型为date型。试向 datetest表中插入两条记录，*

*--一条当前系统日期记录，一条记录为“1998-08-18”。*

create table date\_test(

d date

)

insert into date\_test values(sysdate)

select \* from date\_test

insert into date\_test values(to\_date('1998-8月-08','yyyy/month/dd'))

*--创建与dept表相同表结构的表dtest，将dept表中部门编 号在40之前的信息插入该表*

create table dtest as

select \* from dept where deptno=40

select \* from dtest

select \* from student

alter table student modify(sex default '女'）

alter table student modify(xm char(10))

*--1.学校有一个选课系统，其中包括如下关系模式：*

*--• 系(系编号： 主键，*

*--• 系名称: 唯一键，*

*--• 系主任： 非空约束，*

*--• 系所在校去：取值范围只能在南湖校区和浑南校区)*

create table course(

cid char(4) primary key,

cname char(10) constraint cname\_con unique,

cmg char(4) not null,

loc char(20) constraint loc\_con check(loc='南湖校区' or loc='浑南校区')

)

drop table course

*--创建copy\_emp，要求格式同emp表完全一样，不包含数据*

drop table copy\_emp

create table copy\_emp

as

select \* from emp where 1=0

*--创建copy\_dept，要求格式同dept表完全一样，不包含数据*

create table copy\_dept

as

select \* from dept where 1=0

*--设置copy\_emp 表中外键deptno，参照copy\_dept中deptno,语句能否成功,为什么？*

alter table copy\_emp

add constraint f\_con foreign key(deptno) references copy\_dept(deptno)

*--不能，因为copy\_dept表中的deptno不是主键或唯一值*

*--追加copy\_dept表中主键deptno*

alter table copy\_dept add constraint copy\_dept\_deptno\_pk primary key(deptno);

*--.创建一个视图，通过该视图可以查询到工资在 20005000内并且姓名中*

*--包含有A的员工编号姓名，工资*

create or replace view emp1

as

select empno,ename,sal

from scott.emp

where sal between 2000 and 5000

and ename like'%A%'

select \* from emp1

*--创建视图时，在子查询中使用列的别名*

CREATE VIEW salvu30

AS SELECT empno EMPLOYEE\_NUMBER, ename NAME,sal SALARY

FROM scott.emp

WHERE deptno= 30

select \* from salvu30

*--创建一个视图，通过该视图可以查询到工作在 NEWYORK和CHICAGO的*

*--员工编号姓名部门编号，入职日期。*

create view temp1

as

select empno,ename,e.deptno,hiredate

from scott.emp e,scott.dept d

where e.deptno=d.deptno

and loc in('NEW YORK','CHICAGO')

select \* from temp1

*-- 创建一个视图，通过该视图可以查询到每个部 门的部门名称及最低工资。*

create view temp2

as

select dname,min(sal) lowsal

from scott.emp e,scott.dept d

where e.deptno=d.deptno

group by dname

select \* from temp2

*--.创建视图v\_emp\_20，包含20号部门的员工编号，姓名，*

*--年薪列(年薪=12\*(工资+奖金）；*

create view v\_emp\_20

as

select empno,ename,nvl(12\*(sal+comm),12\*sal) as ysal

from scott.emp

where deptno=20

select \* from v\_emp\_20

*--从视图v\_emp\_20中查询年薪大于1万元员工的信*

select \* from v\_emp\_20

where ysal>10000

*--.请为工资大于2000的员工创建视图，*

*--要求显示员工的部 门信息，职位信息，工作地点；*

create or replace view sal2\_ve

as

select e.deptno,e.job,d.loc

from scott.emp e,scott.dept d

where e.deptno = d.deptno;

select \* from sal2\_ve;

*--针对以上视图执行insert,update,delete,语句能否成功，为什么？*

*-- 向v\_emp\_20 视图执行操作*

*-------------------权限不足---------------------*

insert into v\_emp\_20 values (1214,'aa',2000);*--no 不允许虚拟列*

update v\_emp\_20 set empno = 1214 where empno = 1234; *--ok*

delete v\_emp\_20;*--ok*

*-- 向sal2\_ve 视图执行操作*

insert into sal2\_ve values (12,'老板','北京');*--no ok*

*-- 1、当职位用相关子查询时，为不允许虚拟列*

*-- 2、当使用多表连接时，为无法通过连接视图修改多个基表*

update sal2\_ve set deptno = 20 where deptno = 10; *--ok*

delete sal2\_ve;*--ok*

select \* from emp;

select \* from dept;

select \* from sal2\_ve;

create or replace view v1

as

select e.ename,a.deptno,d.dname,salary

from (select deptno,max(sal) salary

from scott.emp

group by deptno) a,

scott.emp e,scott.dept d

where e.sal=a.salary and a.deptno=d.deptno and e.deptno=d.deptno;

select \* from v1

select \* from scott.emp

*----创建序列test\_seq，起始值为10，每次增长2最大值100最小值9循环序列每次缓存10*

CREATE SEQUENCE test\_seq START WITH 10 *--序列从10开始*

INCREMENT BY 2 *--序列每次增加2*

MAXVALUE 100 *--序列最大值100*

MINVALUE 9 *--序列最小值9 序列循环每次增加直到后回到从新开始*

CYCLE *--序列循环,每次增加2,一直到100后回到9从新开始*

CACHE 10； *--缓存中序列值个数为10*

*--–创建序列studentseq：*

CREATE SEQUENCE student\_seq

START WITH 10000

INCREMENT BY 1;

*--使用序列student\_seq生成student表中sid列插入值*

select \* from scott.student

INSERT INTO scott.student

VALUES(student\_seq.NEXTVAL,'Scott','男','09-3月-03','120',2);

alter table scott.student modify(xh char(5))

*--查看studentseq序列当前值：*

SELECT student\_seq.CURRVAL FROM dual

*--创建一个序列，该序列起始值从1开始，无最 大值增量是1不循环*

create sequence emp\_sequence

start with 1

increment by 1

*--.查询序列的当前值及下一个值*

select emp\_sequence.nextval from dual

select emp\_sequence.currval from dual

*--使用第1题所建的序列，向部门表中插入两条 记录，部门编号使用序列值，*

*--部门名称分别为： EdiMk城市分别为DALLAS Education、Market*

insert into dept values(emp\_sequence.nextval,'EDUCATION','SALLAS')

select \* from dept

SELECT rowid,ename FROM emp;

*--创建测试环境表*

CREATE TABLE e1 AS SELECT \* FROM emp;

INSERT INTO e1 SELECT \* FROM e1 *--多次运行*

select \* from e1

UPDate e1 SET empno=ROWNUM*--更新所有记录的*

TRUNCATE TABLE e1;

rollback

alter table e1 modify empno number(20)

*--测试无索引检索时间*

SELECT ename,sal FROM e1 WHERE empno=200000;0.056

*--测试建索引后检索时间*

CREATE INDEX e1\_id ON e1(empno); *--创建索引*

SELECT ename,sal FROM e1 WHERE empno=210000; *--再次在有索引的情况下做测试*

*--0.014*

*--创建序列，起始位1，自增为1，最小值为1，最大值为9999*

CREATE SEQUENCE test\_sequence

START WITH 1 *--序列从1开始*

INCREMENT BY 1 *--序列每次增加1*

MAXVALUE 9999 *--序列最大值9999*

MINVALUE 1 *--序列最小值1 序列循环每次增加直到后回到从新开始*

create sequence test\_sequence1

start with 50

increment by 5

*--创建表，采用“create table copy\_emp\_indexas select \* from emp”*

*--生成500万条数据把其中的“员工号”字段修改为唯一；*

insert into copy\_emp\_index select \* from copy\_emp\_index

UPDate copy\_emp\_index SET empno=ROWNUM*--更新所有记录的*

*--查询表copy\_emp\_index表中员工号为200001的员工姓名，工资，记 录执行时间；*

select ename，sal from copy\_emp\_index where empno=200001*--0.139*

CREATE INDEX copy\_id ON copy\_emp\_index(empno);

*--在copy\_emp\_index表的empno字段上创建索引，再次执行第6题语句，*

*-- 记录执行时间并做对比*

select ename，sal from copy\_emp\_index where empno=200001*--0.046*

*--.建立新用户neu*

create user neu identified by 123

grant create session to neu

*--给用户neu授权使其能够登录到数据库，能够查询 2.给用户neu授权,使其能够登录到数据库，*

*--能够查询 scott下的emp表，能修改emp表的sal,ename两个字段*

grant select on scott.emp to neu

grant update( sal,ename) on scott.emp to neu

select \* from scott.emp

update scott.emp set sal=9000 where empno=7369

*--回收用户neu的登录权限*

revoke create session from neu

*--4.回收用户neu的所有对象权限*

revoke update on scott.emp from neu;

*--5.建立角色*

create role role\_neu;

*--6.给角色role\_neu授权,使其能够登录到数据库*

grant connect to role\_neu;

*--7.赋角色role\_neu给用户*

grant role\_neu to neu;

*--8.删除角色*

drop role role\_neu;

*--9.删除用户*

drop user neu;

*--写一个块，查询最大的部门编号，并在屏幕上 输出该部门编号 输出该部门编号*

declare

emp\_deptno emp.empno%type;

begin

select max(deptno) into emp\_deptno from emp;

dbms\_output.put\_line(emp\_deptno);

end;

select \* from emp

*--写一个块，用来向10号部门入职一名新员工， 员工编号为当前最大员工编号加1员工姓名为 员工编号为当前最大员工编号加1，*

*--员工姓名为 JAMES，岗位为CLERK，入职日期为当前日期，工 资为4000上级为SMITH奖金为ll*

declare

max\_empno emp.empno%type;

emp\_mgr emp.mgr%type;

begin

select max(empno)+1 into max\_empno from emp;

select empno into emp\_mgr from emp where ename='SMITH';

insert into emp values(max\_empno,'JAMES','CLERK',emp\_mgr,sysdate,4000,null,10);

dbms\_output.put\_line('插入成功');

end;

*--创建一个匿名块，查询emp表，显示雇员名是’SCOTT‘的薪水，通 过DBMS\_OUTPUT包来显示。*

declare

salary emp.sal%type;

begin

select sal into salary from emp where ename='SCOTT';

dbms\_output.put\_line(salary);

end;

*--创建一个匿名块，使用SQL\*Plus的替代变量emp\_num (雇员编号）,*

*-- 查询emp表，通过外部变量显示对应的雇员名*

declare

emp\_num emp.empno%type;

emp\_ename emp.ename%type;

begin

emp\_num :=7369;

select ename into emp\_ename from emp

where empno=emp\_num;

dbms\_output.put\_line(emp\_ename);

end;

*--创建和emp表结构样的test表不要求有数据*

create table test1

as select \* from emp where 1=0;

*--创建pl/sql块，将emp表中高薪水员工的信息插入到test表中*

declare

max\_sal emp.sal%type;

begin

select max(sal) into max\_sal from emp;

insert into test1

select \* from emp

where sal=max\_sal;

end;

select \* from test1

*--创建pl/sql块，将emp表中员工的平均薪水更新到test表中，并打印平均薪水*

declare

avg\_sal emp.sal%type;

begin

select avg(sal) into avg\_sal from emp;

update emp set sal=avg\_sal;

dbms\_output.put\_line(avg\_sal);

end;

*--.写SQL语句，向部门表中添加一个字段 b整型表示部门编制人数*

*-- maxnumber整型，表示部门编制人数。*

alter table dept add maxnumber number(8)

*--.把10号部门的编制更新为5人*

update dept set maxnumber=5 where deptno=10

select \* from dept

select \* from emp

*--写一个块，用来向10号部门入职一名新员工， 员工编号为当前最大员工编号加1，*

*--员工姓名为 TOM岗位为CLERK其它字段都为ll当10号 TOM，岗位为CLERK，其它字段都为null*

*--当部门的人数超过编制人数时提示入职失败*

declare

counting number;

max\_number dept.maxnumber%type;

begin

select maxnumber into max\_number from dept

where deptno=10;

select count(\*) into counting from emp

where deptno=10;

if counting<max\_number

then

insert into emp values(counting+1,'TOM','CLERK',null,null,null,null,10);

dbms\_output.put\_line('入职成功');

else

dbms\_output.put\_line('入职失败');

end if;

end;

*--NULL的处理 5!=null*

DECLARE

v\_x number(2):= 5;

v\_y number(2):= NULL;

BEGIN

IF v\_x != v\_y

THEN

dbms\_output.put\_line(100);

ELSE

dbms\_output.put\_line(200);

END IF;

END;

*--NULL的处理(续) null,不可知的值，所有比较都返回错误*

DECLARE

v\_x number(2):= NULL;

v\_y number(2):= NULL;

BEGIN IF

v\_x = v\_y

THEN

dbms\_output.put\_line(100);

ELSE

dbms\_output.put\_line(200);

END IF;

END;

*--NULL的处理*

DECLARE

v\_x number(2):= 5;

v\_y number(2):= NULL;

BEGIN IF

nvl(v\_x,0) != nvl(v\_y,0)

THEN

dbms\_output.put\_line(100);

ELSE

dbms\_output.put\_line(200);

END IF; END;

*--CASE 语句*

declare

v\_var varchar2(10);

v\_deptno emp.deptno%type;

begin

select distinct deptno into v\_deptno

from emp

where sal=(select max(sal) from emp);

v\_var:=

case v\_deptno

when 10 then '部门1'

when 20 then '部门2'

else '部门3'

end;

dbms\_output.put\_line(v\_var);

end;

*--使用简单循环，批量入职5名员工，员工编号 分别为当前最大编号加1部门为20号部门姓*

*-- 分别为当前最大编号加1，部门为20号部门，姓 名为zs1,zs2…，*

*--入职日期为当前日期，其它字段为null暂时不判断人数是否超编。*

declare

v\_empno emp.empno%type;

v\_name emp.ename%type;

c\_num number;

begin

v\_name:='zs1';

c\_num:=0;

loop

select max(empno)+1 into v\_empno from emp;

insert into emp (empno,ename,deptno,hiredate) values(v\_empno,v\_name,10,sysdate);

dbms\_output.put\_line(c\_num);

c\_num:=c\_num+1;

v\_name:=concat(substr(v\_name,1,2),(substr(v\_name,3,3)+1));

exit when c\_num=5;

end loop;

end;

select \* from emp

delete from emp where ename like 'z%'

*--使用for循环，遍历员工信息，依次输出每个员工的姓名及部门名称*

*--EXCEPTION的用法*

declare

c\_count number;

min\_count number;

max\_count number;

v\_ename emp.ename%type;

v\_dname dept.dname%type;

begin

select min(empno),max(empno) into min\_count,max\_count from emp;

c\_count:=min\_count;

for c\_count in min\_count..max\_count loop

begin*--EXCEPTION*

select ename,dname into v\_ename,v\_dname

from emp e,dept d

where e.deptno=d.deptno and empno=c\_count;

EXCEPTION *--EXCEPTION*

WHEN NO\_DATA\_FOUND THEN

v\_ename := NULL;

end;*--EXCEPTION必要的三步*

IF v\_ename IS NOT NULL

THEN

dbms\_output.put\_line(v\_ename||' '||v\_dname);

end if;

end loop;

end;

*--.使用while循环，遍历员工信息，依次输出每 个员工的姓名及部门名称 个员工的姓名及部门名称。*

declare

c\_count number;

min\_count number;

max\_count number;

v\_ename emp.ename%type;

v\_dname dept.dname%type;

begin

select min(empno),max(empno) into min\_count,max\_count from emp;

while min\_count<max\_count loop

begin*--EXCEPTION*

c\_count:=min\_count+1;

min\_count:=min\_count+1;

select ename,dname into v\_ename,v\_dname

from emp e,dept d

where e.deptno=d.deptno and empno=c\_count;

EXCEPTION *--EXCEPTION*

WHEN NO\_DATA\_FOUND THEN

v\_ename := NULL;

end;*--EXCEPTION必要的三步*

IF v\_ename IS NOT NULL

THEN

dbms\_output.put\_line(v\_ename||' '||v\_dname);

end if;

end loop;

end;

*--使用外部替代变量提供雇员的ID，传递该值到PL/SQL块，查询 emp表的薪水，*

*--如果薪水小于2000的，显示‘挣的不多，需努力 ’；*

*--如果薪水在2000到5000的显示‘收入还可以还需努 ；*

*--薪水大于5000的显示'挣的挺多了，歇歇吧‘*

select \* from emp

declare

v\_empno emp.empno%type;

v\_sal emp.sal%type;

v\_put varchar2(20);

begin

select sal into v\_sal from emp where empno=&v\_empno;

v\_put:=

case

when v\_sal<2000 then '挣的不多，需努力'

when v\_sal<5000 then '收入还可以还需努力'

else '挣的挺多了，歇歇吧'

end;

dbms\_output.put\_line(v\_put);

end;

*--屏幕上输出1到10（不包括6和8)*

declare

c\_num number:=1;

begin

for c\_num in 1..10 loop

if c\_num!=6 and c\_num!=8

then

dbms\_output.put\_line(c\_num);

end if;

end loop;

end;

*--使用FOR循环和while循环，分别实现练习2的批量员工入职功能。*

*--for loop*

declare

v\_empno emp.empno%type;

v\_name emp.ename%type;

c\_num number;

begin

v\_name:='zs1';

c\_num:=0;

for c\_num in 1..5 loop

select max(empno)+1 into v\_empno from emp;

insert into emp (empno,ename,deptno,hiredate) values(v\_empno,v\_name,10,sysdate);

dbms\_output.put\_line(c\_num);

v\_name:=concat(substr(v\_name,1,2),(substr(v\_name,3,3)+1));

end loop;

end;

select \* from emp

delete from emp where ename like 'z%'

*--while loop*

declare

v\_empno emp.empno%type;

v\_name emp.ename%type;

c\_num number;

begin

v\_name:='zs1';

c\_num:=0;

while c\_num<5 loop

c\_num:=c\_num+1;

select max(empno)+1 into v\_empno from emp;

insert into emp (empno,ename,deptno,hiredate) values(v\_empno,v\_name,10,sysdate);

dbms\_output.put\_line(c\_num);

v\_name:=concat(substr(v\_name,1,2),(substr(v\_name,3,3)+1));

end loop;

end;

*-------------------隐式游标---------------------------*

*--从表EMP中删除指定部门的行，并返回所删除行的数量*

DECLARE

vdeptno NUMBER:=20;

v\_deptno NUMBER := 20;

v\_rows\_deleted VARCHAR2(30);

BEGIN

DELETE FROM emp

WHERE deptno = v\_deptno;

v\_rows\_deleted := SQL%ROWCOUNT ;

dbms\_output.put\_line(v\_rows\_deleted || ' rows deleted.');

END;

select \* from emp

rollback

*--假设10号部门有3名员工，20号部门有5名 员工，30号部门有6名员工。*

*--如下代码段执行后，程序输出为： 1*

declare

v\_count number(2);

begin

select count(empno) into v\_count

from emp where deptno=10;

if sql%rowcount>0 then

dbms\_output.put\_line('最近一条执行的SQL语句所影响到的记录行数');

else

dbms\_output.put\_line('没有改变行数');

end if;

dbms\_output.put\_line(sql%rowcount);

end;

*--假设部门表中没有部门编号为60的记录，执 行如下代码后，程序输出为：*

begin

insert into dept(deptno,dname,loc) values(80,'HR','SY');

if sql%rowcount>0 then

dbms\_output.put\_line('成功');

else

dbms\_output.put\_line('失败');

end if;

dbms\_output.put\_line(sql%rowcount);

end;

*--隐式游标集合*

begin

DELETE FROM emp where deptno= 100;

dbms\_output.put\_line('游标所影响的行数：'||SQL%ROWCOUNT);

IF SQL%FOUND THEN DBMS\_output.put\_line('Found为真');*-- sql%found>0,true*

ELSE DBMS\_output.put\_line('Found为假');

END IF;

IF SQL%NOTFOUND then DBMS\_output.put\_line('NotFound为真'); *-- sql%notfound<0,true,没改变行数*

ELSE DBMS\_output.put\_line('NotFound为假');

END IF;

IF SQL%ISOPEN THEN DBMS\_output.put\_line('isOpen为真');*--隐式游标永远为假*

ELSE DBMS\_output.put\_line('isOpen为假');

END IF;

END;

*-----------------------显式游标-----------------------------*

*--循环输出数据*

declare

v\_empno emp.empno%type;

v\_ename emp.ename%type;

i number:=1;

cursor emp\_cursor is

select empno,ename from emp;

begin

open emp\_cursor;

for i in 1..5 loop

fetch emp\_cursor into v\_empno,v\_ename;

dbms\_output.put\_line(v\_empno||' '||v\_ename);

end loop;

close emp\_cursor;

end;

*--创建一个游标，查询雇员的名字和薪水，当查询到某雇员的薪 水大于3000元时，*

*--退出循环，并显示其对应的名字和薪水。*

declare

v\_ename emp.ename%type;

v\_sal emp.sal%type;

i number;

cursor emp\_cursor is

select ename,sal from emp;

begin

open emp\_cursor;

select count(\*) into i from emp;

while i>0 loop

fetch emp\_cursor into v\_ename,v\_sal;

i:=i+1;

if v\_sal>3000 then

dbms\_output.put\_line(v\_ename||' '||v\_sal);

exit;

end if;

end loop;

close emp\_cursor;*--释放游标中select语句的查询结果所占用的系统资源*

end;

select \* from emp

*--使用游标，完成遍历所有员工姓名和部门名 称的操作 称的操作。*

declare

v\_ename emp.ename%type;

v\_dname dept.dname%type;

i number;

cursor read\_cursor is

select ename,dname

from emp e,dept d

where e.deptno=d.deptno;

begin

open read\_cursor;

select count(\*) into i from emp;

while i>0 loop

fetch read\_cursor into v\_ename,v\_dname;

dbms\_output.put\_line(v\_ename||' '||v\_dname);

i:=i-1;

end loop;

close read\_cursor;

end;

*--游标处理中的FOR循环*

*--依次检索所有员工*

declare

cursor emp\_cursor is

select empno,ename from emp;

begin

for emp\_record in emp\_cursor loop

dbms\_output.put\_line(emp\_record.empno);

END LOOP;

end;

*--创建一个游标，查询雇员名字和对应薪水，当遇到雇员名 是’SCOTT’的雇员时退出游标*

*--的FOR循环显示其对应的薪 是SCOTT的雇员时,退出游标的FOR循环,显示其对应的薪 水,*

*--并显示其在游标中的位置.*

declare

cursor emp\_cursor is

select ename,sal from emp;

begin

for emp\_record in emp\_cursor loop

if emp\_record.ename='SCOTT' then

dbms\_output.put\_line(emp\_record.sal);

dbms\_output.put\_line(emp\_cursor%rowcount);

exit;

end if;

end loop;

end;

*--使用游标和记录联合方式，完成遍历所有员工姓名和部门名称的操作*

declare

cursor emp\_cursor is

select ename,dname

from emp e,dept d

where e.deptno=d.deptno;

emp\_record emp\_cursor%rowtype;

i number;

begin

select count(\*) into i from emp;

open emp\_cursor;

loop

i:=i-1;

fetch emp\_cursor into emp\_record;

dbms\_output.put\_line(emp\_record.ename||' '||emp\_record.dname);

exit when i=0;

end loop;

close emp\_cursor;

end;

*--使用游标式的FOR循环方式，完成遍历所有员 工姓名和部门名称的操作*

declare

cursor emp\_cursor is

select ename,dname

from emp e,dept d

where e.deptno=d.deptno;

v\_ename emp.ename%type;

v\_dname dept.dname%type;

i number;

n number:=1;

begin

select count(\*) into i from emp;

open emp\_cursor;

for n in 1..i loop

fetch emp\_cursor into v\_ename,V\_dname;

dbms\_output.put\_line(v\_ename||' '||v\_dname);

end loop;

close emp\_cursor;

end;

*--使用不需声明的游标方式，完成遍历所有员工 姓名和部门名称的操作*

*--1*

begin

for emp\_record in (select ename,dname

from emp e,dept d

where e.deptno=d.deptno)

loop

dbms\_output.put\_line(emp\_record.ename||' '||emp\_record.dname);

end loop;

end;

*--2*

declare

cursor emp\_cursor is

select ename,dname

from emp e,dept d

where e.deptno=d.deptno;

begin

for emp\_record in emp\_cursor loop

dbms\_output.put\_line(emp\_record.ename||' '||emp\_record.dname);

end loop;

end;

*--.使用带参数的游标，分别遍历每个部门的员工姓名及入职日期*

*--1*

declare

cursor emp\_cursor(v\_deptno emp.deptno%type) is

select ename,hiredate

from emp

where deptno=v\_deptno;

c\_count number;

begin

select count(\*) into c\_count from emp where deptno=10;

dbms\_output.put\_line('------10号部门的员工列表------');

for emp\_record in emp\_cursor(10) loop

dbms\_output.put\_line('姓名'||emp\_record.ename||'，入职日期：'||emp\_record.hiredate);

end loop;

dbms\_output.put\_line('小计：10号部门员工数为：'||c\_count||'人');

select count(\*) into c\_count from emp where deptno=20;

dbms\_output.put\_line('------20号部门的员工列表------');

for emp\_record in emp\_cursor(20) loop

dbms\_output.put\_line('姓名'||emp\_record.ename||'，入职日期：'||emp\_record.hiredate);

end loop;

dbms\_output.put\_line('小计：20号部门员工数为：'||c\_count||'人');

select count(\*) into c\_count from emp where deptno=30;

dbms\_output.put\_line('------30号部门的员工列表------');

for emp\_record in emp\_cursor(30) loop

dbms\_output.put\_line('姓名'||emp\_record.ename||'，入职日期：'||emp\_record.hiredate);

end loop;

dbms\_output.put\_line('小计：30号部门员工数为：'||c\_count||'人');

end;

*--2*

declare

cursor emp\_cursor(v\_deptno emp.deptno%type) is

select ename,hiredate

from emp

where deptno=v\_deptno;

cursor dept\_cursor is

select distinct deptno

from dept;

c\_count number;

begin

for dept\_record in dept\_cursor loop

select count(\*) into c\_count from emp where deptno=dept\_record.deptno;

dbms\_output.put\_line('------'||dept\_record.deptno||'号部门的员工列表------');

for emp\_record in emp\_cursor(dept\_record.deptno) loop

dbms\_output.put\_line('姓名'||emp\_record.ename||'，入职日期：'||emp\_record.hiredate);

end loop;

dbms\_output.put\_line('小计：'||dept\_record.deptno||'号部门员工数为：'||c\_count||'人');

end loop;

end;

*--for update*

declare

cursor emp\_cursor is

select ename,sal

from emp

for update nowait;

begin

for emp\_record in emp\_cursor loop

if emp\_record.ename='SCOTT'

then

update emp set sal=emp\_record.sal+300

where ename='SCOTT';

exit;

end if;

end loop;

end;

select \* from emp

rollback

*--不使用WHERE CURRENT OF*

declare

cursor sal\_cursor is

select sal,empno

from emp

where deptno=30

for update of sal nowait;

begin

for emp\_record in sal\_cursor loop

update emp

set sal=emp\_record.sal\*1.1;

*--where current of sal\_cursor;*

dbms\_output.put\_line(emp\_record.sal);

end loop;

end;

*--执行三次后结果，指针一直停留在最后一个*

*/\**

*1149.5*

*1149.5*

*1149.5*

*1149.5*

*1149.5*

*1149.5*

*\*/*

*--使用where current of*

declare

cursor sal\_cursor is

select sal,empno

from emp

where deptno=30

for update of sal nowait;

begin

for emp\_record in sal\_cursor loop

update emp

set sal=emp\_record.sal\*1.1

where current of sal\_cursor;

dbms\_output.put\_line(emp\_record.sal);

end loop;

end;

*--执行三次后结果*

*/\*1936*

*1512.5*

*1512.5*

*3448.5*

*1815*

*1149.5*

*\*/*

*--写一个存储过程getAllDept,遍历所有部门。*

create or replace procedure getAllDept

as

v\_deptno dept.deptno%type;

begin

for dept\_record in (select deptno from dept) loop

dbms\_output.put\_line(dept\_record.deptno);

end loop;

end;

begin

getAllDept;

end;

*--统计表数据个数*

create or replace procedure prm\_example

is

n\_number number;

begin

select count(\*) into n\_number from emp;

dbms\_output.put\_line(n\_number);

end;

begin

prm\_example;

end;

*--.写一个存储过程addDept，实现添加一个部门功能，部 门编号为当前最大部门编号加1，其它信息通过in模式参*

*--门编号为当前最大部门编号加1，其它信息通过in模式参 数传入。*

create or replace procedure addDept(v\_dname in dept.dname%type,

v\_loc in dept.loc%type)

as

v\_deptno dept.deptno%type;

begin

select max(deptno)+1 into v\_deptno from dept;

insert into dept values(v\_deptno,v\_dname,v\_loc);

end;

declare

v\_dname dept.dname%type:='AD';

v\_loc dept.loc%type:='us';

begin

addDept(v\_dname,v\_loc);

end;

select \* from dept

*--.写一个存储过程getDeptByID，实现根据编号读取部门 信息功能。*

create or replace procedure getDeptByID(v\_deptno in dept.deptno%type)

as

begin

for dept\_record in(select \* from dept where deptno=v\_deptno) loop

dbms\_output.put\_line

(dept\_record.deptno||' '||dept\_record.dname||' '||dept\_record.loc);

end loop;

end;

declare

v\_deptno dept.deptno%type:=10;

begin

getDeptByID(v\_deptno);

end;

*--.写一个存储过程updateDept，实现根据编号修改部门功能。*

create or replace procedure updateDept(v\_deptno in dept.deptno%type,

v\_dname in dept.dname%type,v\_loc in dept.loc%type)

as

begin

update dept set dname=v\_dname,loc=v\_loc where deptno=v\_deptno;

end;

declare

v\_dname dept.dname%type:='AD';

v\_loc dept.loc%type:='fs';

v\_deptno dept.deptno%type:=81;

begin

updateDept(v\_deptno,v\_dname,v\_loc);

end;

select \* from dept

*--写一个存储过程delDept，实现根据部门编号删除部门 功能，当部门中存在员工时，*

*--提示该部门不能删除*

create or replace procedure delDept(v\_deptno in dept.deptno%type)

as

emp\_exist exception;

v\_count number;

begin

delete from dept where deptno=v\_deptno;

select count(\*) into v\_count from emp where deptno=v\_deptno;

if v\_count>0 then

raise emp\_exist;

end if;

exception

when emp\_exist then

dbms\_output.put\_line('该部门不能删除');

rollback;

end;

declare

v\_deptno dept.deptno%type:=10;

begin

delDept(v\_deptno);

end;

*--写一个存储过程getAllEmpByID，实现根据部门编号查询部门所有员工信息功能,*

*--要求带分页功能*

create or replace procedure getAllEmpByID(v\_deptno in dept.deptno%type,i in number)

as

begin

for emp\_record in(select empno,ename,job,hiredate,sal,mgr,deptno from

(select rownum rn,empno,ename,job,hiredate,sal,mgr,deptno

from emp

where deptno=v\_deptno

)

where rn<=i\*2 and rn>(i-1)\*2) loop

dbms\_output.put\_line

(emp\_record.empno||' '||emp\_record.ename||' '||emp\_record.job||' '||

emp\_record.mgr||' '||emp\_record.hiredate||' '||emp\_record.sal );

end loop;

end;

declare

v\_deptno dept.deptno%type:=20;

i number:=1;

begin

getAllEmpByID(v\_deptno,i);

end;

select \* from jobhistory

*--写一个存储过程changeDept，实现部门调转功 能传入参数员工编号调入部门编号传出 能，*

*--传入参数：员工编号、调入部门编号；传出 参数：调转是否成功，成功返回ture，*

*--否则返回 fl实现该功能时需要判断传入的员工编 flase。实现该功能时，*

*--需要判断传入的员工编 号及部门编号是否是有效的；同时修改emp表的 dt字段并且在员工历*

*--史岗位表中插入条 deptno字段，并且在员工历史岗位表中插入一条 新记录*

create or replace procedure changeDept

(v\_empno in emp.empno%type,v\_deptno in dept.deptno%type)

as

a varchar(5);

v\_temp1 varchar(8);

v\_temp2 varchar(8);

v\_id jobhistory.id%type;

begin

select deptno into v\_temp1 from dept where deptno=v\_deptno;

select empno into v\_temp2 from emp where empno=v\_empno;

select max(id) into v\_id from jobhistory;

insert into jobhistory(id,empno) values(v\_id+1,v\_empno);

update emp

set deptno=v\_deptno

where empno=v\_empno;

a:='true';

dbms\_output.put\_line(a);

exception

when no\_data\_found then

dbms\_output.put\_line('没有该部门或员工');

a:='false';

dbms\_output.put\_line(a);

end;

declare

v\_empno emp.empno%type:=7369;

v\_deptno dept.deptno%type:=30;

begin

changeDept(v\_empno,v\_deptno);

end;

*/\*.创建一个员工离职表:dimission,包括如下字段：*

*–流水号:数值型*

*–员工编号:数值型 员编号数值型*

*–离职时所在部门编号：数值型*

*–离职日期:日期型*

*–离职原因:变长字符*

*–离职去向：变长字符\*/*

create table dimission(

id number(8) primary key,

empno number(8),

deptno number(8),

hiradate date,

reason varchar(20),

newloc varchar(20)

)

*--修改员工表emp结构，添加员工状态字段status,整型， 默认值为1,表示正常状态*

alter table emp add (status number default 1)

*--.写一个存储过程addDim，实现员工离职功能，办理成功返回ture，失败返回false，*

*--仔细分析传入及传出参数，办理离职时，修改员工表的员工状态字段status为2，*

*--并且向dimission表中插入1条记录。*

create or replace procedure addDim(v\_empno in emp.empno%type)

as

v\_deptno emp.deptno%type;

v\_id number;

begin

update emp set status=2 where empno=v\_empno;

dbms\_output.put\_line('true');

select nvl(max(id),0) into v\_id from dimission;

select deptno into v\_deptno from emp where empno=v\_empno;

insert into dimission(id,empno,deptno,hiradate) values(v\_id+1,v\_empno,v\_deptno,sysdate);

exception

when no\_data\_found then

dbms\_output.put\_line('false');

end;

declare

v\_empno emp.empno%type:=7369;

begin

addDim(v\_empno);

end;

*--写一个存储过程searchDim,实现遍历指定时间段办理入职的员工姓名。*

create or replace procedure searchDim(v\_hiredate in emp.hiredate%type)

as

v\_ename emp.ename%type;

begin

for emp\_record in (select ename from emp where hiredate=v\_hiredate) loop

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

end loop;

end;

declare

v\_hiredate emp.hiredate%type;

begin

v\_hiredate:=to\_date('1981-12-3','yyyy-mm-dd');

searchDim(v\_hiredate);

end;

select \* from emp

*--.编写一个函数，计算员工应交个人所得税， 1000元以下的员工不交税10002000元的按 1000元以下的员工，不交税，*

*--1000-2000元的按 照5%缴纳，2000以上的10%缴纳。*

create or replace function p\_count(v\_empno emp.empno%type)

return number

as

v\_sal number;

v\_money number;

begin

select sal into v\_sal from emp where empno=v\_empno;

v\_money:=

case

when v\_sal<1000 then 0

when v\_sal<2000 then v\_sal\*0.05

else v\_sal\*0.1

end;

return v\_money;

end;

select \* from emp

*-- 2.写一个块，调用上述函数*

declare

v\_empno emp.empno%type:=7499;

v\_money number(4,2);

begin

v\_money:=p\_count(v\_empno);

dbms\_output.put\_line('个人所得税'||v\_money);

end;

*--写一个SELECT语句，调用上述函数*

select empno, p\_count(empno)

from emp;

*--创建一个函数，根据输入的参数（部门代码）值 ，返回对应的部门员工的最高薪水*

create or replace function ename\_finction(v\_empno emp.empno%type)

return emp.ename%type

as

v\_ename emp.ename%type;

begin

select ename into v\_ename from emp where empno=v\_empno;

return v\_ename;

end;

select empno, ename\_finction(empno)

from emp;

*--创建一个存储过程，查询dept表的所有部门信息 ,*

*--调用该函数，当函数返回的薪水大于4000时， 产生异常，*

*--在异常部分显示'某某部门的薪水太高 了' 了'。*

create or replace function sal\_function(v\_deptno emp.deptno%type)

return emp.sal%type

as

v\_avg emp.sal%type;

begin

select nvl(round(avg(sal),0),0) into v\_avg from emp where deptno=v\_deptno

group by deptno;

return v\_avg;

exception

when no\_data\_found then

v\_avg:=0;

return v\_avg;

end;

create or replace procedure checksal

as

sal\_too\_much exception;

begin

for emp\_record in

(select deptno,dname,loc,nvl(sal\_function(deptno),0) from dept) loop

begin

if nvl(sal\_function(emp\_record.deptno),0)>4000 then

raise sal\_too\_much;

end if;

dbms\_output.put\_line(emp\_record.deptno||' '||emp\_record.dname||''||

emp\_record.loc||' '||nvl(sal\_function(emp\_record.deptno),0));

exception

when sal\_too\_much then

dbms\_output.put\_line(emp\_record.deptno||'部门的薪水太高了');

end;

end loop;

end;

begin

checksal;

end;

*--编写个块，实现员工入职功能：其中员工编号为7839，*

*--姓名为张三，工资为3000，入职日期为系统当前日期，职位 为CLERK*

*--，奖金为null，所在部门为SALES，上级经理为 为CLERK，奖金为null，*

*--所在部门为SALES ，上级经理为 JONES ，要求写出所有可能发生的异常处理程序。*

declare

v\_mgr emp.mgr%type;

v\_deptno emp.deptno%type;

begin

select empno into v\_mgr from emp where ename='JONES';

select deptno into v\_deptno from dept where dname='SALES';

insert into emp values

(7839,'张三'，'CLERK',v\_mgr,sysdate,3000,null,v\_deptno);

exception

when DUP\_VAL\_ON\_INDEX then

dbms\_output.put\_line('数据重复');

when no\_data\_found then

dbms\_output.put\_line('数据错误');

end;

select \* from dept

rollback

*--.编写一个匿名块，完成删除一个部门的功能，要有相应的 异常处理程序。*

DECLARE e\_emps\_remaining EXCEPTION;

PRAGMA EXCEPTION\_INIT(e\_emps\_remaining, -02292);

v\_deptno dept.deptno%TYPE := &p\_deptno;

BEGIN

DELETE FROM dept WHERE deptno = v\_deptno;

EXCEPTION WHEN e\_emps\_remaining THEN

DBMS\_OUTPUT.PUT\_LINE ('该部门不能被删除');

END;

*--捕获异常的函数*

*--编写一个匿名块，分别对部门做一个查询、插入、修改 、删除操作*

*--，验证这几个操作执行后的SQLCODE和 SQLERRM结果。*

declare

v\_depno emp.empno%type;

BEGIN

*--INSERT INTO emp*

*--values(7369,'SMITH','CLERK',7902,to\_date('1980-12-17','yyyy-mm-dd'),800,NULL,20);*

*--select empno into v\_depno from emp where deptno=11;*

*--update emp set empno='7369';*

delete emp where empno='ee' ;

EXCEPTION WHEN OTHERS THEN

dbms\_output.put\_line (sqlerrm);

dbms\_output.put\_line(sqlcode);

END;

select \* from emp

*--用户定义的异常*

declare

e\_too\_many exception;

begin

update emp

set sal=5000

where deptno=&p\_deptno;

if sql%rowcount>2 then

raise e\_too\_many;

end if;

exception

when e\_too\_many then

dbms\_output.put\_line('涨工资人数 '||SQL%ROWCOUNT||'人太多了');

rollback;

end;

*--修改员工入职的功能，当入职部门的人数超编时，提示 用户部门以超编，不能再录入员工。*

declare

e\_too\_many exception;

v\_empno emp.empno%type;

v\_name emp.ename%type;

c\_num number;

i number;

begin

v\_name:='zs1';

c\_num:=0;

for c\_num in 1..5 loop

begin

select max(empno)+1 into v\_empno from emp;

insert into emp (empno,ename,deptno,hiredate)

values(v\_empno,v\_name,10,sysdate);

dbms\_output.put\_line(c\_num);

v\_name:=concat(substr(v\_name,1,2),(substr(v\_name,3,3)+1));

select count(\*) into i from emp;

if i>20 then

raise e\_too\_many;

end if;

exception

when e\_too\_many then

dbms\_output.put\_line('太多人了');

rollback;

exit;

end;

end loop;

end;

select \* from emp;

*--获得每个部门的平均工资，如果平均工资大于15000，*

*--视为用户 定义的异常提示“该部门的平均工资过高”*

declare

e\_too\_many exception;

v\_sal emp.sal%type;

cursor dept\_cursor is

select deptno

from dept;

begin

for dept\_record in dept\_cursor loop

begin

select avg(sal) into v\_sal from emp where deptno=dept\_record.deptno;

if v\_sal>15000 then

raise e\_too\_many;

end if;

exception

when e\_too\_many then

dbms\_output.put\_line('该部门的平均工资过高'||'部门：'||dept\_record.deptno||'平均工资：'||v\_sal);

end;

end loop;

end;

select \* from emp

rollback

update emp set sal=16000 where deptno=10;

*--创建表message(resultvarchar2(100)),存放状态信息*

create table message(

result varchar2(100)

)

*--当emp表中没有该sal值时引发异常在异常部分将‘*

*--当emp表中没有该sal值时，引发异常，在异常部分将 没有雇员挣sal的薪水’信息插入到message表中，*

*--并显示该 信息； 信息； 当emp表中有多于一个sal值时，引发异常，*

*--在异常部分 将‘太多的雇员挣sal的薪水’信息插入到message表中，*

*--并 将太多的雇员挣sa的薪水信息插入到essage表中，并 显示该信息；*

*-- 当只有一个雇员具有该工资时，则输出雇员名和工资。*

declare

v\_ename emp.ename%type;

v\_sal emp.sal%type;

begin

select sal,ename into v\_sal,v\_ename from emp;

exception

when NO\_DATA\_FOUND then

insert into message values('没有雇员挣sal的薪水');

dbms\_output.put\_line('没有雇员挣sal的薪水');

when TOO\_MANY\_ROWS then

insert into message values('太多的雇员挣sal的薪水');

dbms\_output.put\_line('太多的雇员挣sal的薪水');

when others then

dbms\_output.put\_line(v\_ename||' '||v\_sal);

end;

select \* from emp

*--根据员工号，获得员工到目前为止参加工作年限（保留到整数） ，*

*--员工号不存在时提示“此员工号不存在”*

declare

v\_hiredate number;

begin

select round(months\_between(sysdate,hiredate)/12,0) into v\_hiredate

from emp

where empno=&v\_empno ;

exception

when no\_data\_found then

dbms\_output.put\_line('此员工号不存在');

end;

*--编写PL/SQL块，使用SELECT语句将管理者编号为空的员工的姓名*

*--及工作编号显示出来如果符合条件的员工多于人则返回字 及工作编号显示出来，*

*--如果符合条件的员工多于一人，则返回字 符串“最高管理者人员过多！”字符串，*

*--如果找到没有符合条件 的记录则返回字符串“没有最高管理者请指定”*

declare

e\_too\_many exception;

v\_count number;

begin

for emp\_record in

(select empno,ename from emp where mgr is null) loop

begin

select count(empno) into v\_count from emp where mgr is null;

dbms\_output.put\_line(emp\_record.ename||' '||emp\_record.empno);

if v\_count>1 then

raise e\_too\_many;

end if;

exception

when e\_too\_many then

dbms\_output.put\_line('最高管理者人员过多');

when no\_data\_found then

dbms\_output.put\_line('没有最高管理者,请指定');

end;

end loop;

end;

*--统计大于平均薪水的员工数量,如果数量大于5,触发 etoomany异常,*

*--显示’大于平均工资的人数不少’;如果数量小 e\_too\_many异常,*

*--显示大于平均工资的人数不少;如果数量小 于等于5,触发e\_too\_low异常,*

*--显示’大于平均工资的人数太少*

declare

e\_too\_many exception;

e\_too\_low exception;

v\_count number;

begin

select count(empno) into v\_count

from (select avg(sal) a from emp) b,emp

where b.a<emp.sal;

if v\_count>5 then

raise e\_too\_many;

elsif v\_count<5 then

raise e\_too\_low;

end if;

exception

when e\_too\_many then

dbms\_output.put\_line('大于平均工资的人数不少');

when e\_too\_low then

dbms\_output.put\_line('大于平均工资的人数太少');

end;

CREATE OR REPLACE PACKAGE comm\_package

IS

v\_deptno NUMBER := 20;

PROCEDURE dept\_comm;

PROCEDURE create\_comm(v\_deptno IN dept.deptno%type);

PROCEDURE deptno\_comm(v\_deptno IN dept.deptno%type);

PROCEDURE change\_comm(v\_deptno IN dept.deptno%type);

PROCEDURE emp\_comm(v\_deptno IN dept.deptno%type);

PROCEDURE delete\_comm(v\_deptno IN dept.deptno%type);

END comm\_package;

create or replace package body comm\_package

is

*------------------验证部门编号是否有效*

function deptno\_found(v\_deptno in dept.deptno%type)

return boolean

is

temp dept.deptno%type;

begin

select deptno into temp from dept where deptno=v\_deptno;

return true;

exception

when no\_data\_found then

return false;

end deptno\_found;

*------过程：查询当前所有部门信息*

procedure dept\_comm

as

begin

for dept\_record in (select \* from dept) loop

dbms\_output.put\_line(dept\_record.deptno||''||dept\_record.dname||''||

dept\_record.loc);

end loop;

end dept\_comm;

*--过程：新建一个部门，成功返回新建的部门编号，失 败返回-1*

procedure create\_comm(v\_deptno in dept.deptno%type)

as

temp dept.deptno%type;

begin

if deptno\_found(v\_deptno) then

dbms\_output.put\_line('该部门已存在');

else

insert into dept(deptno) values(v\_deptno);

dbms\_output.put\_line(v\_deptno);

end if;

end create\_comm;

*--过程：根据部门编号读取部门信息*

procedure deptno\_comm(v\_deptno in dept.deptno%type)

as

begin

for dept\_record in (select \* from dept where deptno=v\_deptno) loop

dbms\_output.put\_line(dept\_record.deptno||''||dept\_record.dname||''||

dept\_record.loc);

end loop;

end deptno\_comm;

*--：过程：根据部门编号修改部门信息*

procedure change\_comm(v\_deptno in dept.deptno%type)

as

begin

if deptno\_found(v\_deptno) then

update dept set dname='AA' where deptno=v\_deptno;

else

dbms\_output.put\_line('该部门不存在');

end if;

end change\_comm;

*--过程：根据部门编号查询部门所有员工信息*

procedure emp\_comm(v\_deptno in dept.deptno%type)

as

begin

if deptno\_found(v\_deptno) then

for emp\_record in (select \* from emp where deptno=v\_deptno) loop

dbms\_output.put\_line(emp\_record.deptno||''||emp\_record.ename||''||

emp\_record.empno);

end loop;

else

dbms\_output.put\_line('该部门不存在');

end if;

end emp\_comm;

*--过程：根据部门编号删除一个部门，之前要判断该部门下是否存在员工，*

*--如果存在员工，则提示用户“部门 下已经存在员工，不能删除该部门”。*

procedure delete\_comm(v\_deptno in dept.deptno%type)

as

v\_count emp.empno%type;

begin

select count(\*) into v\_count from emp where deptno=v\_deptno;

dbms\_output.put\_line('部门 下已经存在员工，不能删除该部门');

exception

when no\_data\_found then

delete from dept where deptno=v\_deptno;

end delete\_comm;

end comm\_package;

create or replace procedure test3

is

begin

comm\_package.dept\_comm;

comm\_package.create\_comm(10);

comm\_package.deptno\_comm(20);

comm\_package.change\_comm(20);

comm\_package.emp\_comm(40);

comm\_package.delete\_comm(50);

end;

begin

test3;

end;

rollback

DROP PACKAGE comm\_package1;

select \* from test

drop table test

create table test(

empno number(8) primary key,

ename varchar2(10),

sal number(7,2)

)

*--创建包的声明test\_pak,内容包含add\_user过程、del\_user过程*

*--(根据empno值删除用户）、add\_sal函数（根据empno值确定用户 ，*

*--增加员工工资，并返回该用户的工资）*

create or replace package test\_pak

as

procedure add\_user(v\_empno in test.empno%type);

procedure del\_user(v\_empno in test.empno%type);

procedure add\_sal(v\_empno in test.empno%type,v\_sal out test.sal%type);

end;

create or replace package body test\_pak

as

*------adduser*

procedure add\_user(v\_empno in test.empno%type)

as

begin

insert into test(empno,ename,sal) values(v\_empno,'new1',9000);

exception

when DUP\_VAL\_ON\_INDEX then

dbms\_output.put\_line('已存在该员工');

end;

*--------del\_user*

procedure del\_user(v\_empno in test.empno%type)

as

begin

delete from test where empno=v\_empno;

exception

when no\_data\_found then

dbms\_output.put\_line('该员工不存在');

end;

*-------*

procedure add\_sal(v\_empno in test.empno%type,v\_sal out test.sal%type)

as

begin

update test set sal=sal+500 where empno=v\_empno;

select sal into v\_sal from test where empno=v\_empno;

exception

when no\_data\_found then

dbms\_output.put\_line('该员工不存在');

end;

end;

create or replace procedure test3

is

v\_sal test.sal%type;

begin

test\_pak.add\_user(7370);

test\_pak.del\_user(7370);

test\_pak.add\_sal(7369,v\_sal);

dbms\_output.put\_line(v\_sal);

end;

begin

test3;

end;

rollback

select \* from test

insert into test select empno,deptno,sal from emp

*--创建一个记录类型emp\_record\_type,它包含雇员的名字、薪 水、工作岗位，*

*--声明个该类型的变量emp\_record，查询雇员编号是7788雇员的上述信息，*

declare

type emp\_record\_type is record(

ename varchar(20),

sal emp.sal%type,

job emp.job%type);

emp\_record emp\_record\_type;

begin

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

dbms\_output.put\_line(emp\_record.ename||' '||emp\_record.sal);

end;

*--.写一个块，遍历所有员工信息。要求使用记录 型变量 型变量。*

declare

type emp\_record\_type is record(

empno emp.empno%type,

ename emp.ename%type,

sal emp.sal%type,

job emp.job%type);

emp\_record emp\_record\_type;

begin

for emp\_record in (select \* from emp ) loop

dbms\_output.put\_line(emp\_record.ename||' '||emp\_record.sal

||' '||emp\_record.sal||' '||emp\_record.job);

end loop;

end;

*----. 写一个块，遍历所有员工信息，要求使用 %ROWTYPE %ROWTYPE。*

declare

emp\_record emp%rowtype;

i number;

cursor a is

select \* from emp;

begin

open a;

select count(\*) into i from emp;

while i>0 loop

i:=i-1;

fetch a into emp\_record;

dbms\_output.put\_line(emp\_record.ename||' '||emp\_record.sal

||' '||emp\_record.sal||' '||emp\_record.job);

end loop;

end;

*--创建一个视图，通过该试图可以查询每个部门工资最高的人员姓名，部门编号，部门名称，工 资*

create or replace view v1(ename,deptno,dname,sal)

as

select e.ename,a.deptno,d.dname,salary

from (select deptno,max(sal) salary

from scott.emp

group by deptno) a,

scott.emp e,scott.dept d

where e.sal=a.salary and a.deptno=d.deptno and e.deptno=d.deptno;

*-- 写一个块，遍历每个部门工资最高的人员姓 名，部门名称，工资。*

*--要求使用第2题中创建的视图。*

declare

emp\_record v1%rowtype;

i number;

cursor a is

select \* from v1;

begin

open a;

select count(\*) into i from v1;

while i>0 loop

i:=i-1;

fetch a into emp\_record;

dbms\_output.put\_line(emp\_record.ename||' '||emp\_record.sal||' '||emp\_record.dname);

end loop;

end;

*--.在EMP表上创建语句级别的触发器,*

*--当用户在8:00点至17:00点以外插入数据时，系统提示只 是在工作期间可以录入数据’;*

*--当用户在8:00点至17:00点以外修改数据时，系统提示‘只 是在工作期间可以修改数据’*

*--当用户在8:00点至17:00点以外删除数据时，系统提示‘只 是在工作期间可以删除数据’.*

create or replace trigger emp\_trigger

before insert or update or delete on emp

begin

if(to\_char(sysdate,'HH24:MI') not between '08:00' and '17:00')

then

if inserting then

raise\_application\_error(-20002, '只是在工作期间可以录入数据');

elsif updating then

raise\_application\_error(-20003, '只是在工作期间可以修改数据');

elsif deleting then

raise\_application\_error(-20001, '只是在工作期间可以删除数据');

end if;

end if;

end;

drop trigger emp\_trigger

update emp set sal=111

delete from emp where sal=1300

select \* from emp

*--创建DML行触发器*

CREATE OR REPLACE TRIGGER sal\_emp

BEFORE UPDATE OF sal ON emp

FOR EACH ROW

BEGIN

IF(:new.sal >:old.sal)

THEN raise\_application\_error (-20001,'薪水太高了');

END IF;

END;

UPDATE emp SET sal=5000;

*--在员工部门调转时，当修改员工表中的部门编号时，请在该 操作上建立触发器，*

*--同步实现员工历史岗位表中数据的操作。*

create or replace trigger emp\_change

after update of deptno on emp

for each row

declare v\_id number;

begin

select max(id) into v\_id from jobhistory;

insert into jobhistory values

(v\_id+1,:old.empno,:old.job,:old.hiredate,:old.sal);

end;

update emp set deptno=20 where empno=7839;

drop trigger emp\_change

select \* from jobhistory

rollback

*--创建视图*

CREATE VIEW my\_view

AS SELECT emp.empno,emp.ename,emp.job,emp.sal,emp.deptno, dept.dname

FROM emp,dept WHERE emp.deptno = dept.deptno;

*--ORA-01776: 无法通过联接视图修改多个基表*

INSERT INTO my\_view VALUES(1005,'ljs','clerk',2000,30,'sales');

*--INSTEAD OF触发器*

create or replace trigger sal\_emp1

instead of insert on my\_view

for each row

begin

INSERT INTO dept(deptno,dname) VALUES(90,'neu');

INSERT INTO emp (empno,ename,job,sal,deptno)

VALUES(1005,'ljs','clerk',2000,90);

END;

INSERT INTO my\_view VALUES (1005,'ljs','clerk',2000,30,'sales');

select \* from my\_view