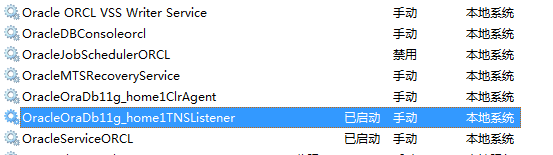
Oracle

Oracle 安装完成之后实际上会在系统中进行服务注册，在注册的服务中有以下两个服务必须启动，否则Oracle无法正常使用

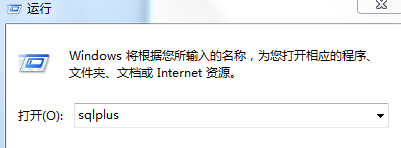


1. OracleOraDb10\_home1TNSListener：表示监听服务，如果客户端要想连接到数据库，此服务必须打开，在日后程序开发中此服务起作用
2. OracleServiceORCL表示数据库的主服务，命名规则：OracleService数据库名，此服务必须启动，否则Oracle无法使用

# Oracle数据库基本操作

Oracle只有一个数据库ocle，一个用户代表一个session

1. 进入数据库中： sqlplus



1. 用户类型及我的密码

普通用户： 用户名：scott 密码： tiger

普通管理员： system 用户的密码：manager

超级管理员： sys用户的密码：851596

1. 基本 select \* from emp;

设置页行：set linesize 300; set pagesize 20;

1. 以系统管理员的身份登录

Sql> conn sys/851596 as sysdba

Sql> conn system/manger as sysdba

再次发出 select \* from emp;会提示出表或试图不存在，因为emp表时属于scott用户下的表，而sys用户本身没有。如果现在不同用户下想访问emp该表，则必须加上用户名，即：表的完整名称：“scott.emp”

1. 关于表操作：

Show user;显示当前正在连接的用户是哪一个

查看用户下所有的表：select \* from tab;

查看表结构： desc 表名;

Create table t\_user(id int , varchar2(23),password varchar2(23));

复制表：Create table myemp as select \* from emp;

dual系统中的一张表，是一张虚拟表

select sysdate from dual;获取当前系统时间

删除表，drop table [表名];

插入：

Insert into <表名> [(<属性列1>……)] value(<常量1>……)

Insert into t\_users(username,password) values(‘tom’,’1234’);

Insert into t\_users values(1,’tom’,’12345’);

下面一种必须对应创建表的字段循序

### oracle的scott用户下默认表

|  |
| --- |
| EMP（雇员表）  NO 字段 类型 描述  1 EMPNO NUMBER(4) 雇员编号  2 ENAME VARCHAR2(10) 表示雇员姓名  3 JOB VARCHAR2(9) 表示工作职位  4 MGR NUMBER(4) 表示一个雇员的领导编号  5 HIREDATE DATE 表示雇佣日期  6 SAL NUMBER(7,2) 表示月薪，工资  7 COMM NUMBER(7,2) 表示奖金或佣金  8 DEPTNO NUMBER(2) 表示部门编号 |

部门表：表示一个部门的具体信息

|  |
| --- |
| DEPT（部门表）  NO 字段 类型 描述  1 DEPTNO NUMBER(2) 部门编号  2 DNAME VARCHAR2(14) 部门名称  3 LOC VARCHAR2(13) 部门位置 |

奖金表：表示一个雇员的工资及奖金。

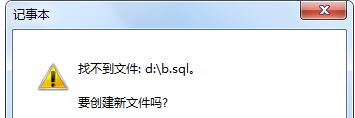
|  |
| --- |
| BONUS（奖金表）  NO 字段 类型 描述  1 ENAME VARCHAR2(10) 雇员姓名  2 JOB VARCHAR2(9) 雇员工作  3 SAL NUMBER 雇员工资  4 COMM NUMBER 雇员奖金 |

一个公司是有等级制度，用此表表示一个工资的等级

|  |
| --- |
| SALGRADE（工资等级表）  NO 字段 类型 描述  1 GRADE NUMBER 等级名称  2 LOSAL NUMBER 此等级的最低工资  3 HISAL NUMBER 此等级的最高工资 |

简单命令语句：

创建编辑文件：ed d:/a;关闭后执行



执行文件： @ d:/a.sql;

### Oracle: 数据类型：

#### 基本的数据类型：

**number(4)表示数字，长度为4**

**Varchar2（10）表示的是字符串，只能容纳10个长度**

**Date；表示日期**

**Number(7,2) 表示数字，其中小数位占2位，整数位占5位**

2. 数字函数   
ABS(abs) 取绝对值 (power) 乘方 ln 10为底数取0   
sqrt 平方根 EXP e的n次乘方 LOG(m,n) m为底数n取0   
数学运算函数:ACOS ATAN ATAN2 COS COSH SIGN SIN SINH TAN TANH   
CEIL 大于或等于取整数   
FLOOR 小于或等于取整数   
mod 取余数   
ROUND(n,m) 按m的位数取四舍五入值如果round(日期): 中午12以后将是明天的日期.       
round(sysdate,'Y')是年的第一天   
TRUNC(n,m) 按m的位数取小数点后的数值如果trunc(日期), 确省的是去掉时间

|  |
| --- |
| ORACLE中TRUNC是截取的函数  用在日期类型上，就是截取到日或时间  SELECT TRUNC(SYSDATE) FROM DUAL;  默认是截取系统日期到日，得到2012-12-19  比如值为"2012-12-19 14:30:50"的日期变量  TRUNC(SYSDATE)得到 2012-12-19 , 和TRUNC(SYSDATE,'DD')一样  TRUNC(SYSDATE,'YYYY')，得到2012-1-1  TRUNC(SYSDATE,'MM')，得到2012-12-1  TRUNC(SYSDATE,'DD')，得到2012-12-19  TRUNC(SYSDATE,'HH')或HH24，得到2012-12-19 14:00:00  TRUNC(SYSDATE,'MI')，得到2012-12-19 14:30:00  TRUNC(SYSDATE,'SS')，是会报错的，直接用SYSDATE不用TRUNC  另外说明一下，截取后得到的仍为date数据类型 |

#### 字符函数

CHR 按数据库的字符集由数字返回字符   
CONCAT(c1,c2) 把两个字符c1,c2组合成一个字符, 和 || 相同   
REPLACE(c,s,r) 把字符c里出现s的字符替换成r, 返回新字符   
SUBSTR(c,m,n) m大于0,字符c从前面m处开始取n位字符,m等于0和1一样,   
m小与0,字符c从后面m处开始取n位字符   
TRANSLATE(c,f1,t1) 字符c按f1到t1的规则转换成新的字符串   
INITCAP 字符首字母大写,其它字符小写   
LOWER 字符全部小写   
UPPER 字符全部大写   
LTRIM(c1,c2) 去掉字符c1左边出现的字符c2   
RTRIM(c1,c2)   
TRIM(c1,c2) 去掉字符c1左右两边的字符c2   
LPAD(c1,n,c2) 字符c1按制定的位数n显示不足的位数用c2字符串替换左边的空位   
RPAD(c1,n,c2)

#### 日期函数

ADD\_MONTHS(d,n) 日期值加n月   
LAST\_DAY  返回当月的最后一天的日期   
MONTHS\_BETWEEN(d1,d2) 两个日期值间的月份,d1<d2 返回负数   
NEXT\_DAY  返回日期值下一天的日期   
SYSDATE 当前的系统时间   
DUAL是SYS用户下一个空表，它只有一个字段dummy

#### 转换函数

**TO\_CHAR(date,'日期显示格式') to\_char(sysdate,’yyyy-mm-dd hh24:mi:ss’)**

**所有转换后是字符串，不在date格式的数据  
TO\_CHAR(number) 用于显示或报表的格式对齐**

**To\_char(123456789,’999,999,999’);  
TO\_DATE(char,'日期显示格式')**   
TO\_LOB 把long字段转换成lob字段   
TO\_NUMBER(char) 用于计算或者比较大小   
4. 转换函数(2)   
to\_date里日期显示格式 :

YYYY 年 YEAR YYY YY Y   
Q 季度   
MM 月 MONTH MON   
W 星期 (week of month) WW, IW (week of year) (说明：周计是按ISO标准,从1月1日的星期数到后面七天为一周,不一定是从周一到周日)   
DD 日 DAY DY   
HH24 小时 HH12 HH   
MI 分钟   
SS 秒   
如果想固定日期的显示格式可以在数据库的参数文件initorasid.ora里新写一行参数        
NLS\_DATE\_FORMAT=yyyy-mm-dd hh24:mi:ss可以在UNIX环境变量或者NT的注册表里的设置 NLS\_DATE\_FORMAT=yyyy-mm-dd

4. 转换函数(3)

如果想固定日期的显示格式可以用alter session命令改变 SQL> alter session set nls\_date\_format='yyyy-mm-dd hh24:mi:ss';   
它的作用顺序如下:   
initialization parameter   
Environment variable   
ALTER SESSION command   
4. 转换函数 (4)   
to\_char(number)里数字显示格式 9 数字位 0 数字前面补0 to\_char(-1200,'00000.00') . 小数点的位置 , 标记位置的逗号 用在数字显示格式的左边 L 根据数据库字符集加货币符号 to\_char(-1200,'L9999.99') B 把数字0显示为空格,用在数字显示格式的右边 MI 右边的负数标记 to\_char(-1200,'9999.99MI') PR 括起来的负数 to\_char(-1200,'9999.99PR') EEEE 用指数方式表示 to\_char(-1200,'9999.99EEEE')

输入字符,返回数字的函数 instr(c1,c2) 字符c2出现在c1的位置, 不出现, 返回0, 常用于模糊查询

**length  按数据库的字符集,算出字符c的长度,跟数据库的字符集有关, 一个汉字长度为1**

**有逻辑比较的函数NVL(EXPR1, EXPR2)函数**   
解释: IF EXPR1=NULL RETURN EXPR2   
ELSE RETURN EXPR1   
**DECODE(AA0V10R10V20R2....)函数**解释: IF AA=V1 THEN RETURN R1   
IF AA=V2 THEN RETURN R2   
..…   
ELSE   
RETURN NULL   
**举例: decode(id,1,'dept sale',2,'dept tech')**

# SQL高级程序

## SQL UNION 和 UNION ALL 操作符

**SQL UNION 操作符**

UNION 操作符用于合并两个或多个 SELECT 语句的结果集。

请注意，UNION 内部的 SELECT 语句必须拥有相同数量的列。列也必须拥有相似的数据类型。同时，每条 SELECT 语句中的列的顺序必须相同。

### SQL UNION 语法

SELECT column\_name(s) FROM table\_name1

UNION

SELECT column\_name(s) FROM table\_name2

**注释：**默认地，UNION 操作符选取不同的值。如果允许重复的值，请使用 UNION ALL。

### SQL UNION ALL 语法

SELECT column\_name(s) FROM table\_name1

UNION ALL

SELECT column\_name(s) FROM table\_name2

另外，UNION 结果集中的列名总是等于 UNION 中第一个 SELECT 语句中的列名。

### 下面的例子中使用的原始表：

Employees\_China:

|  |  |
| --- | --- |
| **E\_ID** | **E\_Name** |
| 01 | Zhang, Hua |
| 02 | Wang, Wei |
| 03 | Carter, Thomas |
| 04 | Yang, Ming |

Employees\_USA:

|  |  |
| --- | --- |
| **E\_ID** | **E\_Name** |
| 01 | Adams, John |
| 02 | Bush, George |
| 03 | Carter, Thomas |
| 04 | Gates, Bill |

使用 UNION 命令

**实例**

列出所有在中国和美国的不同的雇员名：

SELECT E\_Name FROM Employees\_China

UNION

SELECT E\_Name FROM Employees\_USA

**结果**

|  |
| --- |
| **E\_Name** |
| Zhang, Hua |
| Wang, Wei |
| Carter, Thomas |
| Yang, Ming |
| Adams, John |
| Bush, George |
| Gates, Bill |

**注释：**这个命令无法列出在中国和美国的所有雇员。在上面的例子中，我们有两个名字相同的雇员，他们当中只有一个人被列出来了。UNION 命令只会选取不同的值。

**UNION ALL**

UNION ALL 命令和 UNION 命令几乎是等效的，不过 UNION ALL 命令会列出所有的值。

SQL Statement 1

UNION ALL

SQL Statement 2

**使用 UNION ALL 命令**

**实例：**

列出在中国和美国的所有的雇员：

SELECT E\_Name FROM Employees\_China

UNION ALL

SELECT E\_Name FROM Employees\_USA

**结果**

|  |
| --- |
| **E\_Name** |
| Zhang, Hua |
| Wang, Wei |
| Carter, Thomas |
| Yang, Ming |
| Adams, John |
| Bush, George |
| Carter, Thomas |
| Gates, Bill |

## 查询：

select {distinct}\*|查询列名1 别名1，查询列名2 别名2，……

from 表名称1 别名1，表名称2 别名2，……

[where 条件]

[group by 分组条件 [having 分组后条件]]

[order by 排序字段1 asc|desc]

查询语句关键字：

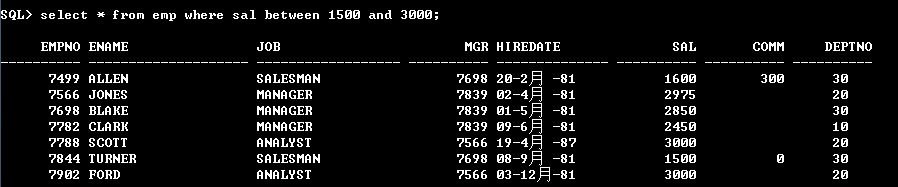
From \* from table;

1.确定范围： between and ，not between and

**|| 字符串连接符：**



**Between** : between 1500 and 300;(1500<=sal<=300)



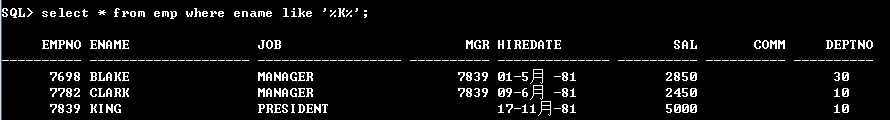
**相反的是： not between 1500 and 3000;**

**相同： sal>=1500 and sal<=3000;**

2.字符匹配：like ， not like；

通配符：用%代表任意长度（可以为0）的字符串，

用\_ 代表任意单个字符

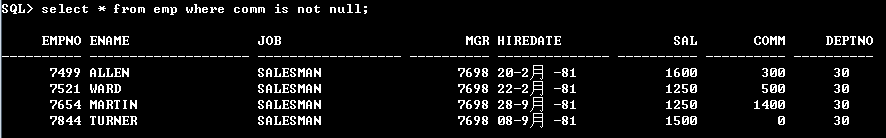


当查询t\_user类型，就不能用 like ‘\_\_%’，需要是\_是，用’\\_’来取消通配符，

2.比较大小： > < = <= >= != !< !>

4.空值：is null，is not null

当查询为是否null的时候，select \*from emp where conn is null;不能用”=”;等不到值



5.确定集合 in not in

1. 要求查询出每个雇员的姓名、工作、雇员的直接上级领导的姓名

|  |
| --- |
| select e.ename,e.job,m.ename  from emp e,emp m  where e.empno(+)=m.mgr; |

雇员的直接上级领导,可以了解到mgr字段是显示领导的empno，肯定要将雇员emp表和领导的emp表进行关联

其中由于KING是总裁，没有领导，所以当 e.empno=m.mgr是无法显示出KING字段，但是由于查询时要求或者显示出需要所有员工信息，就要用到左右连接，其中左连接是等号左边，**（+）是Oracle特殊表示左右连接的符号**

1. 要求进一步扩展之前的程序，将雇员所在的部门名称同时列出

|  |
| --- |
| # select e.ename,e.job,m.ename,d.dname  from emp e,emp m,dept d  where e.empno(+)=m.mgr and e.deptno=d.deptno(+); |

1. 现在要求查询出每个雇员的姓名，工资，部门名称，工资在公司等级（salgrade）以及其领导的姓名及工资所在等级

|  |
| --- |
| select e.ename,e.sal,d.dname,s.grade,m.ename,ms.grade  from emp e,emp m,dept d,salgrade s,salgrade ms  where e.empno=m.mgr and e.deptno=d.deptno and  e.sal between s.losal and s.hisal and  m.sal between ms.losal and ms.hisal; |

1. 现在按照要求显示工资等级

1 第一等

2 第二等

依次……

|  |
| --- |
| select e.ename,e.sal,d.dname,  **decode(s.grade,1,'第一等',2,'第二等',3,'第三等',4,'第四等',5,'第五等') 雇员工资等级,m.ename,**  decode(ms.grade,1,'第一等',2,'第二等',3,'第三等',4,'第四等',5,'第五等') 上级领导工资等级  from emp e,emp m,dept d,salgrade s,salgrade ms  where e.empno=m.mgr and e.deptno=d.deptno and  e.sal between s.losal and s.hisal and  m.sal between ms.losal and ms.hisal; |

Decode类似于if else判断语句

语法：decode(字段，值1，’需求1’ ，值2，’需求2’……)

### 组函数与分组统计

#### 组函数

在sql语句中常用的函数有以下几个

* COUNT();求出全部的记录数
* MAX（）求出一组中最大值
* MIN（）求出一组中最小值
* AVG（）求出平均值
* SUM() 求和

#### 分组统计（group by）

格式：

|  |
| --- |
| select {distinct}\*|查询列名1 别名1，查询列名2 别名2，……  from 表名称1 别名1，表名称2 别名2，……  [where 条件]  [group by 分组条件 [having 分组后条件]]  [order by 排序字段1 asc|desc] |

**范例：** 求出每个雇员数量，肯定应该按照部门编号划分，按deptno分组

|  |
| --- |
| select e.deptno,count(e.ename)  from emp e  group by e.deptno; |

**范例:** 求出每个部门的平均工资

|  |
| --- |
| select e.deptno,trunc(avg(sal),1)  from emp e  group by e.deptno; |

**注意：** 如果程序使用了分组函数，则有两种可能使用的情况

1. 如果程序使用了分组函数，则有两种可以使用的情况

A) 程序存在了group by，并指定了分组条件，这样分组条件和分组函数一起查询出

B) 不使用分组的话，只能单独使用

2. 在使用分组函数的时候，不能出现分组函数和分组条件之外的字段

错误示范：

|  |
| --- |
| select deptno count(sal)  from emp; #不使用分组的话，只能单独使用 |

|  |
| --- |
| select deptno,**ename,**sum(sal)  from emp  group by deptno; |

select deptno,ename,sum(sal)

\*

第 1 行出现错误:

ORA-00979: 不是 GROUP BY 表达式

**范例：**按部门分组，并显示部门的名称，及每个部门的员工数

|  |
| --- |
| select dname,count(ename)  from dept,emp  where dept.deptno = emp.deptno  group by dname |

**范例：**要求显示出平均工资大于2000的部门编号和平均工资

|  |
| --- |
| select deptno,round(avg(sal))  from emp  where avg(sal)>2000  group by deptno; |

where avg(sal)>2000

\*

第 3 行出现错误:

ORA-00934: 此处不允许使用分组函数

#### Having

注意：**分组函数只能在分组中使用，不可以在where语句中出现，那么如果现在需要指定分组条件，则只能通过Having**

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

#### 多表查询：

多表查询需要消除笛卡尔积，

|  |
| --- |
|  |

### 查询练习：

1. 列出至少有一个员工的所有部门

|  |
| --- |
| select deptno,count(deptno)  from emp e  group by deptno  having count(deptno)>=5; |

1. 列出薪金比“SMITH”多的所有员工

|  |
| --- |
| select empno,ename,e.sal  from emp e  where sal>(  select d.sal  from emp d  where ename='SMITH'); |

1. 列出所有员工的姓名和直接上级的名字

|  |
| --- |
| select e.ename,m.ename  from emp e,emp m  where e.mgr=m.empno; |

1. 受雇日期早于其直接上级的所有员工的编号，姓名，部门名称

|  |
| --- |
| select e.empno,e.ename,d.dname  from emp e,emp m,dept d  where e.mgr=m.empno and e.hiredate<m.hiredate  and e.deptno=d.deptno; |

1. 列出部门名称和这写部门的员工信息，同时列出那些没有员工的部门

|  |
| --- |
| select d.deptno,d.dname,e.ename  from emp e,dept d  where e.deptno(+)=d.deptno; |

1. **列出'CLERK'的姓名及其部门名称，部门的人数**

|  |
| --- |
| **select e.ename e,d.dname d,ed.cou**  **from emp e,dept d,**  **(select deptno,count(deptno) cou from emp group by deptno) ed**  **where e.deptno(+)=d.deptno and e.job='CLERK' and ed.deptno=d.deptno;** |

1. 列出最低薪金大于1500的各种工作及此从事工作的全部雇员人数

|  |
| --- |
| select job,count(ename)  from emp  group by job  having min(sal)>1500; |

8.列出在部门‘SALES’工作的员工的姓名，假定不知道销售部的部门编号

select e.ename

from emp e

where e.deptno=(

select d.deptno

from dept d

where d.dname='SALES');

**9.列出薪金高于公司的平均薪金的所有员工，所在部门，上级领导，公司的工资等级**

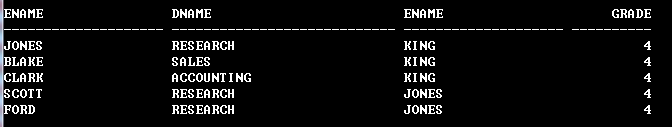
**select m.ename,d.dname,ms.ename,s.grade**

**from emp m,dept d,emp ms,salgrade s**

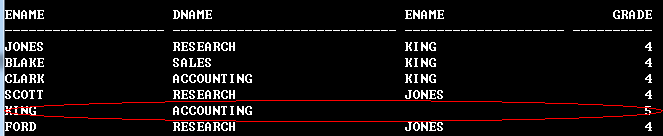
**where m.sal>(select avg(e.sal) from emp e)**

**and d.deptno=m.deptno and ms.empno(+)=m.mgr**

**and m.sal between s.losal and s.hisal;**



**注意，其中由于KING没有领导，所以在ms.empno=m.mgr时，会遗失KING，只有加上左连接才可以**



|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

## 子查询：

子查询可以在select，from，where内

有一面试题：

在Oracle数据库中有一张表test，其中记录如下

Col (1,2,3) col2(aa,bb,cc)

请用sql语句，将结果显示为：aa bb cc

第一种：

|  |
| --- |
| select (select col2 from test where col=1) ,  (select col2 from test where col=2) ,  (select col2 from test where col=3)  from dual; |

第二种：

|  |
| --- |
| Select a.col2,b.col2,c,col3  **from test a , (select col2 from test where col=1) b,**  **(select col2 from test where col=2) c ,**  from test a  where col=1; |

## 数据伪列：

数据伪列，只是了解到ROWNUM和ROWID，最重要的是ROWNUM在实际操作是可以

如：

面试题：

在oracle中，使用emp表，查询公司中工资最高的5个人的排名

|  |
| --- |
| select ename,sal  from (select ename,sal  from emp  order by sal desc)  **where rownum <6;** |

## 事务管理：

Rollback和commit

## 更新操作：

### 表的创建

文本数据，clob 4GB文本

二进制数据:4GB 图像 视频，不建议使用

在创建时可以使用默认值 content clob default ‘默认值，可以有4G文本’

其中插入值时，用date类时，用to\_date（’19990-10-29’,’yyyy-mm-dd’）

截断表：

Truncate table 表名；

删除表

当删除表时，会遗留一些字段，可以同select \* from tab；其中就是回收站的意思

### 回收站：

查看回收站、show recyclebn

恢复myemp表 flashback table myemp to before drop;

删除回收站中表 purge table person

清空回收站 purge recyclebin

约束：

五大约束：

非空约束 not null 唯一 unique

可以指定约束名称，constraint UK\_email unique(email)

主键约束 primary key（主键）

检查约束： check

外键约束：foreign key id references table（id）

### 建表、约束、查询的练习

在某个学生运动会比赛信息的数据库，保存了如下的表：

·运动表 sport（运动编号sporterid name sex department（所属系号））

·项目item（itemid，itemname，location项目比赛地点）

·成绩grade（sporterid，itemid，mark）

功能要求

请用SQL语句完成如下功能：1. 建表，注意满足如下要求：b) 定义各个表的主码外码约束c) 运动员的姓名和所属系别不能为空值。d) 积分要么为空值，要么为6,4,2,0, 分别代表第一，二，三名和其他名次的积分，注意名次可以有并列名次，后面的排名不往前提升，例如，如果有两个并列第一，则没有第二名。2. 往表中插入数据：运动员（ 1001, 李明，男，计算机系 1002，张三，男，数学系 1003, 李四，男，计算机系 1004, 王二，男，物理系 1005, 李娜，女，心理系 1006, 孙丽，女，数学系）项目（ x001, 男子五千米，一操场 x002，男子标枪，一操场 x003, 男子跳远，二操场 x004, 女子跳高，二操场 x005, 女子三千米，三操场）积分（ 1001, x001, 6 1002, x001, 4 1003, x001, 2 1004, x001, 0 1001, x003, 4 1002, x003, 6 1004, x003, 2 1005, x004, 6 1006, x004, 4）3. 完成如下查询a） 求出目前总积分最高的系名，及其积分。b） 找出在一操场进行比赛的各项目名称及其冠军的姓名c） 找出参加了张三所参加的所有项目的其他同学的姓名4. 经查张三因为使用了违禁药品，其成绩都记0分，请在数据库中作出相应修改。5. 经组委会协商，需要删除女子跳高比赛项目

2. 往表中插入数据：

运动员（ 1001, 李明，男，计算机系

1002，张三，男，数学系

1003, 李四，男，计算机系

1004, 王二，男，物理系

1005, 李娜，女，心理系

1006, 孙丽，女，数学系）

insert into sport(sporterid,name,sex,departmant) values(1001, '李明'，'男'，'计算机系');

insert into sport(sporterid,name,sex,departmant) values(1002，'张三'，'男'，'数学系');

insert into sport(sporterid,name,sex,departmant) values(1003, '李四'，'男'，'计算机系');

insert into sport(sporterid,name,sex,departmant) values(1004, '王二'，'男'，'物理系');

项目（ x001, 男子五千米，一操场

x002，男子标枪，一操场

x003, 男子跳远，二操场

004, 女子跳高，二操场

x005, 女子三千米，三操场）

insert into item(itemid,itemname,location) values(x001, '男子五千米'，'一操场');

insert into item(itemid,itemname,location) values(x002，'男子标枪'，'一操场');

insert into item(itemid,itemname,location) values(x003, '男子跳远'，'二操场');

insert into item(itemid,itemname,location) values(x004, '女子跳高'，'二操场');

积分（ 1001, x001, 6

1002, x001, 4

1003, x001, 2

1004, x001, 0

1001, x003, 4

1002, x003, 6

1004, x003, 2

1005, x004, 6

1006, x004, 4）

insert into grade(sporterid,itemid,mark) values(1001, 'x001', 6);

insert into grade(sporterid,itemid,mark) values(1002, 'x001', 4 );

insert into grade(sporterid,itemid,mark) values(1003, 'x001', 2 );

insert into grade(sporterid,itemid,mark) values(1004, 'x001', 0);

insert into grade(sporterid,itemid,mark) values(1001, 'x003', 4 );

## 序列（重点）：

很多数据库都有一个自增长的列，如果现在想要Oracle中完成自动增长的功能，则只能依靠序列完成，所有自动增长操作，需要用户手工编写

|  |
| --- |
| 1.首先创建序列，Oracle序列的语法格式为： CREATE SEQUENCE sequence //创建序列名称  [INCREMENT BY n] //递增的序列值是n 如果n是正数就递增,如果是负数就递减 默认是1  [START WITH n] //开始的值,递增默认是minvalue 递减是maxvalue  [{MAXVALUE n | NOMAXVALUE}] //最大值  [{MINVALUE n | NOMINVALUE}] //最小值  [{CYCLE | NOCYCLE}] //循环/不循环  [{CACHE n | NOCACHE}];//分配并存入到内存中cache |
| NEXTVAL 返回序列中下一个有效的值并返回变化后的值  CURRVAL 中存放序列的当前值  **（NEXTVAL 应在 CURRVAL 之前指定 ，二者应同时有效,并且currval不能单独使用，而nextval可以单独，所以说nextval使用比较广泛）** |

这nextval是插入数据最佳方法

1)INCREMENT BY用于定义序列的步长，如果省略，则默认为1，如果出现负值，则代表Oracle序列的值是按照此步长递减的。

2)START WITH 定义序列的初始值(即产生的第一个值)，默认为1。

3)MAXVALUE 定义序列生成器能产生的最大值。选项NOMAXVALUE是默认选项，代表没有最大值定义，这时对于递增Oracle序列，系统能够产生的最大值是10的27次方;对于递减序列，最大值是-1。

4)MINVALUE定义序列生成器能产生的最小值。选项NOMAXVALUE是默认选项，代表没有最小值定义，这时对于递减序列，系统能够产生的最小值是?10的26次方;对于递增序列，最小值是1。

5)CYCLE和NOCYCLE 表示当序列生成器的值达到限制值后是否循环。CYCLE代表循环，NOCYCLE代表不循环。如果循环，则当递增序列达到最大值时，循环到最小值;对于递减序列达到最小值时，循环到最大值。如果不循环，达到限制值后，继续产生新值就会发生错误。

6)CACHE(缓冲)定义存放序列的内存块的大小，默认为20。NOCACHE表示不对序列进行内存缓冲。对序列进行内存缓冲，可以改善序列的性能。

#### 创建序列

|  |
| --- |
| create sequence myseq; |

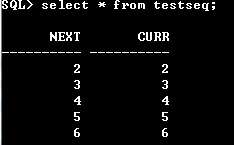
创建表，测试

|  |
| --- |
| create table testseq(next number,curr number); |

插入数据（序列名.nextval,或序列名.currval）

|  |
| --- |
| insert into testseq(next,curr) values(**myseq.nextval,myseq.currval**); |

查询出



**从中发现，next的内容始终进行自动增长的操作，而currval取出当然操作的序列结果**

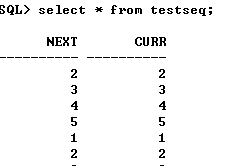
**修改序列的增长长度：**

·每次的增长幅度：increment 长度，start with n maxvalue n minvalue n cycle nocache

范例：

|  |
| --- |
| create sequence myseq **increment by 1 start with 1 maxvalue 5 cycle nocache;** |

插入数据后

显示：

 Alter Sequence修改序列

不能修改序列的初值    
序列的最小值不能大于当前值    
序列的最大值不能小于当前值

 Alter sequence示例：

|  |
| --- |
| alter sequence myseq minvalue 2; |

报错：ORA-04007: MINVALUE 不能大于当前值（开始值为0）

尽量还是重新drop在create序列

## 视图（重点）：

什么是视图：在sql中，视图是基于sql语句的结果集的可视化的表

详细介绍：视图是一个或多个数据库中导出的表。与基本表不同，数据库只存放视图的定义，而不存放视图对应的数据，这些数据存放在原来的基本表中，所以基本表中数据发生改变，从视图中查询的数据也就随之该变，从这个意义上讲，视图就是一个窗口，透过它可以看到数据中自己感兴趣数据及其变化

**视图一旦定义，就可以和基本表一样被查询、被删除。也可以在一个视图是在定义新的视图，但对视图的更新（增、删、改）操作有一点限制·**

### 视图的作用：

1. 视图能够简化用户的操作

视图机制使用户可以将注意力集中在所关心的数据上。用视图看起来结构简单、清晰，并且简化用户的数据查询

**例如：**那些定义了若干表连接的视图，就将表与表之间连接操作操作对用户隐藏起来了。换句话说，用户所做只是对一个虚表的简单查询，而这个虚表怎么来的，用户无需了解

1. 视图使用户能以多种角度看待同一数据

视图机制能使不同的用户以不同方式看待同意数据，当许多不同种类的用户共享同一数据库是，这中灵活性非常重要

**例如：**对于那些不负责处理工资单的雇员，只允许他们看见雇员表中的姓名列、办公室列、工作电话列和部门列，而不能看见任何包含工资信息或个人信息的列。

1. 视图对重构数据库提供一定程度的逻辑独立性
2. 视图能够对机密数据提供安全保护
3. 适当的利用视图可以更清晰的表带查询

### 定义视图：

#### [解决Oracle创建视图（VIEW）权限不足的方法](http://blog.csdn.net/yichangzyh/article/details/8232084)

**问题描述**：在scott用户下创建视图的时候，报错：权限不足。（其他用户以此类推）  
**解决方法**：

以dba用户登录

Conn  sys/密码 **as** sysdba

赋予scott用户创建VIEW的权限

**grant** **create** **view** **to** scott

conn scott/tiger

#### 创建视图

|  |
| --- |
| create view <视图名> [(<列名>[,<列名>]……]  as <子查询>  [with check option][with read only] |

注意：其中子查询可以是任意复杂的select语句，但通常不能含有order by子句和distinct

短语，但实验后，找不到相应的错误，所以等以后看下

With check option 不允许更新子查询的条件

With read only 允许查询，就是只读

|  |
| --- |
| create view myview  as  select distinct deptno from emp ordery by deptno; |

可以创建视图，说明可以有orde by、distinct语句

**范例：**建立一个视图，此视图包含了全部的20部门的雇员信息（雇员编号，姓名，工作，雇佣日期）

|  |
| --- |
| create view emp20 as select empno,ename,job,hiredate from emp where deptno=20; |

视图创建后，可以像查找基本表一样来查询

范例：查询视图

|  |
| --- |
| Select \* from emp20； |

**当视图创建后，视图已经存在，就无法重名，此时可以先删除后在重新建立视图,这样需要修改视图时，比较麻烦，所以提供一个替换的命令，此时完整的视图创建语法：**

#### 重新创建已有视图并替换：

|  |
| --- |
| create or replace 视图名称 as 子查询; |

使用以上语法，在更改视图的时候就不用先删除在执行了，系统会为用户自动进行删除及重建的功能

|  |
| --- |
| create **or replace view** emp20 as  select empno,ename,job,hiredate from emp where deptno=20; |

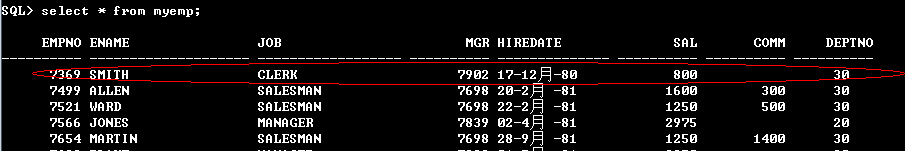
#### 更新：

**范例：**修改视图中的7369的部门编号

|  |
| --- |
| update emp20 set deptno=30 where empno=7369; |

此时提示视图正常进行了更新，重新查询视图之后，会发现在视图中已经没有7369这个雇员了，那么emp表呢？

|  |
| --- |
| Select \* from myemp; |



发现在emp表中的7369雇员的部门编号已经修改为30了，这样做明显不合适，因为创建视图时是有条件的，一旦修改之后，则此条件就破坏了，所以在创建视图是sql中提供了两个重要的参数

**·with check option； 不能更新视图的创建条件**

范例：在视图创建中使用此with check option

|  |
| --- |
| Create or replace view emp20  As select \* from myemp where deptno=20  With check option; |

再进行更新时，就报错

**第 1 行出现错误:**

**ORA-01402: 视图 WITH CHECK OPTION where 子句违规**

**范例：**其他字段还是可以更新的

|  |
| --- |
| **update emp20 set ename='wulaing' where empno=7566;** |

更新成功

但是，视图本身还是用来查询的，所以不应该允许更新，所以可以使用第二参数：

**·with read only;创建的视图只读**

|  |
| --- |
| create or replace view emp20 as  select \* from myemp where deptno=20  **with read only**; |

再次进行更新时，报错

**第 1 行出现错误:**

**ORA-42399: 无法对只读视图执行 DML 操作**

#### 删除视图语法：

|  |
| --- |
| 0drop view 视图名; |

注意：其中视图只是一个虚拟表，而其中删除了视图，不需要经过回收站，和级联其他的基本表

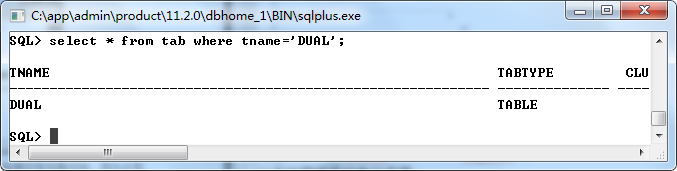
## 同义词dual来源

一直强调dual是一张虚拟表，类似于视图，在那定义的呢？

在scott的用户中，用select \* from tab；查询不到dual表

登录sys用户

|  |
| --- |
| Conn sys/851596 as sysdba; |



在sys用户下存在此表，此表在sys下，但是在scott用户下却可以直接通过表名称访问，那么正常情况下要访问不同用户的表需要使用“用户名.表名”

**那么此时，实际上就是使用了同义词的作用，同义词，可以让其他的用户通过一个名称方便的访问“用户名.表名称”**

#### 创建同义词的语法：

|  |
| --- |
| Create synonym 同义词名称 for 用户名表名称； |

范例：将scott,emp定义emp的同义词

|  |
| --- |
| 在sys用户下  create synonym myemp for scott.myemp; |

**类似于视图，可不和视图一样，不能删除**



能插入，修改，删除等操作，能级联到基本表上，所以要注意

|  |
| --- |
| insert into myemp(empno) values(7366); |

**结果：**

