基本 select 语句总结

--SQL:结构化查询语句

一.基本查询语法

```
select * from emp;
select empno,ename from emp;
--空值不参与运算
select empno,sal as mysal from emp; --为列重命名
select empno,sal,ename as "worker's name" from emp; --""表示特殊字符串引用
select empno,sal,deptno "部门编号" from emp;        --as 可以没有
--使用||将多个列合并

select '编号为'||empno||'的工资为'||sal||',10 年的收入为'||(sal*13)*10 from emp;
--""仅限于特殊引用，表示字符串用''
```

二.过滤和排序

```
select * from emp where deptno=20; --过滤出部门编号为 20 的员工信息
Select distinct JOB From emp where JOB = 'CLERK'; --distinct 过滤出不重复的数据
Select * from emp where SAL != 800; --过滤
--between and 可用于数字、字符、和日期类型数据. 包括 800 和 2500
Select * from emp where SAL Between 800 And 2500;
Select * from emp where empno In(7521,7900,7876); --In(某个集合)

select * from emp where ENAME Like '%A%N%'; --模糊查询，名字中有 A 和 N 的，A 在 N 之
前
select * from emp where ENAME Like '__M%'; --两跳，第三个字母是 M 的。

Select * from emp where MGR is null; --过滤出没有上司的，这里用 IS 不是 =。

Select * from emp where SAL > 2000 AND COMM Is null; --and
Select * from emp where SAL>2500 OR COMM Is not null; --or
```

--and,or 连用，注意默认情况下 and 的优先级比 or 高
Select * from emp where SAL>2000 AND MGR is null or job = 'ANALYST';

--排序，默认 ASC-升序，DESC 降序
Select * from emp Order by sal asc,Hiredate desc; --按 sal 升序，sal 相同按 hiredate 降序排列。
Select ename name , sal salary from emp Order by salary asc; --按别名排序

三.单行函数：对每行输入数据进行计算得到对应的结果，单行函数可嵌套使用

--分类：字符函数、数字函数、日期函数、转换函数、以及其它函数

A.字符类:lower,upper,initcap,concat,substr,LPAD,RPAD,trim,replace
select ename,lower(ename) from emp; --loewer
select ename,lower(ename),upper(lower(ename)) from emp; --upper
select ename,INITCAP(ename) from emp; --INITCAP 首字母大写，其余小写

--字符处理类
select CONCAT(empno,ename) from emp; --CONCAT:连接不同字段，只能连接两个字段。

--SUBSTR 截取子串
select ename,substr(ename,2) from emp; --从第二个位置截至末尾。（截掉第一个字符）
select ename,substr(ename,2,3) from emp; --从第二个位置开始截取三个字符

select ename,length(ename) from emp; --length:统计字符长度
select ename,instr(ename,'A') from emp; --INSTR 求 A 在 ename 中的索引
--LPAD,RPAD： 左右粘贴
select empno,ename,sal,LPAD(sal,10,'$') from emp; --左粘贴 ，显示 10 位，不够左粘贴'$'.
--trim 两边滤空 LTRIM,RTRIM
select TRIM('   a b   c   ') from dual; --字符串中间的空格不管
--replace:替换
Select ename ,Replace(ename,'A','*') from emp; --将 A 用*代替

B.数字转换类型:ROUND,TRUNC,MOD
Select ROUND(16.7777788888888,3) from dual; --保留三位小数，四舍五入
Select ROUND(178.36934,-1) from dual; --整数位减少一位显示
Select TRUNC(16.7777788888888,3) from dual; --直接截尾
Select MOD(89,10) from dual; --MOD：取余

C.日期类函数:MONTH_BETWEEN,ADD_MONTHS,NEXT_DAY,LAST_DAY,ROUND,TRUNC

--MONTH_BETWEEN:两个日期的月份之差
Select max(hiredate) from emp; --最晚工作的人
Select min(hiredate) from emp; --最早工作的人
select max(TO_CHAR(hiredate,'yyyy'))-min(TO_CHAR(hiredate,'yyyy')) from emp; --两者年份之差

```
Select ename,hiredate,MONTHS_BETWEEN(sysdate,hiredate) from emp;
--MONTHS_BETWEEN:hiredate 和 sysdate 之间月份之差
select max(MONTHS_BETWEEN(sysdate,hiredate)) from emp; --最早雇佣的人
select * from emp where MONTHS_BETWEEN(sysdate,hiredate)=(select
max(MONTHS_BETWEEN(sysdate,hiredate)) from emp);--最早工作人的全部信息
--ADD_MONTHS
select hiredate,ADD_MONTHS(hiredate,3) from emp;
--NEXT_DAY:以当前时间为基准，下一个目标日的日期
select NEXT_DAY(sysdate,'星期二') from dual;
--LAST_DAY:计算当前日期的最后一天
select LAST_DAY(sysdate) from dual;
select LAST_DAY(hiredate)-hiredate from emp;
--ROUND:对日期进行四舍五入
select ROUND(sysdate,'yyyy') from dual; --按年进行四舍五入
select round(sysdate) from dual; --按日进行舍入
--TRUNC:截取
select trunc(sysdate,'D') from dual; --截取至本周第一天
select trunc(sysdate,'MM') from dual; --截取至本月第一天
select trunc(sysdate,'DD') from dual; --截取至本日 0：00
select trunc(sysdate,'yyyy') from dual; --截取至本年第一天
Select TRUNC(sysdate-365) from dual; --sysdate-1:代表一天。TRUNC：截掉时分秒
Select Sysdate-1/24/60 from dual; --一分钟之前的时间显示。
```

D：转换类函数。类型转换只能发生在：数字与字符、日期与字符之间。
to_char,to_date,to_number

```
to_char:转换数字为字符，转换日期为字符
--tochar(date,'fmt')
select TO_CHAR(sal,'$999,999.00') from emp; --按千分位保留两位小数显示，L 本地货币符号显示
--数字类型格式控制符： 9 一代表一位数字，有显示，无不显示。 0 一强制显示该位，无以 0 显示
                --$,L:货币符号显示。，. ：千分位、小数位显示

select TO_CHAR(123.235,'L999.00') from dual; --需要显示的位数不能比原来少
select TO_CHAR(sysdate,'"今天是"yyyy-mm-dd DAY MON HH24:MI:SS PM') from dual;--p41
select TO_CHAR(sysdate,'"今天是"YYYY YEAR DAY ') from dual;
--日期类格式控制符：year:年的拼写。MONTH：月的拼写。DAY：星期的拼写。MON：月的缩写。DY：星期的缩写。
--其它时间格式控制符：AM：早上或下午。HH24：24 小时制。SP：数字的拼写。TH：数字的序列词。"":特殊字符。

TO_DATE:将字符串转换为日期格式，格式要匹配。
select TO_DATE('01-5 月-04') from dual; --隐式转换，因为字符串与默认日期格式完全匹配。默认日期格式：DD-MON-YY
select TO_DATE('2008 年-01-01','yyyy"年"-mm-dd') from dual; --转换格式要和原格式完
```

全匹配。
select TO_DATE('2008.8.4','yyyy.mm.dd') from dual; --格式匹配
Select TO_DATE('2008-01-01 星期二 10:36:31 上午','YYYY-MM-DD DAY HH:MI:SS PM') From DUAL

TO_NUMBER:将字符串转换为数字。
注意:格式必须要匹配.
Select TO_NUMBER('$1,600.01','$999,999.99') From DUAL

E:其它函数
--与空值相关的函数：NVL(exper1，exper2)，NVL2(exper1，exper2，exper3)，NULLIF(exper1，exper2)，COALESCE (exper1，……，expern)
select comm,NVL(comm,200) from emp; --每有的以 200 显示，NVL 的两个参数类新需匹配，否则出错
select SAL,comm,NVL2(comm,sal+comm,sal) from emp; --NVL2，comm 不为空则显示 sal+comm，否则显示 sal。
select ENAME,JOB,NULLIF(LENGTH(ename),5) from emp; --NULLIF，判断是否相等，相等返回 null 否则返回表达式 1。
select * from emp where NULLIF(LENGTH(ename),5) is NULL; --显示 emp 表中所有名字长为 5 的数据记录。
--coalesce:逐个判断，成立显示，否则向后

-条件表达式：case+字段名，when……then……else……end，decode（）
select JOB,case job when'CLERK'then'店员'when'SALESMAN'then'销售'when'MANAGER'then'经理'else'其他'end from emp;
select job,decode(job,'CLERK','店员','SALESMAN','销售','MANAGER','经理','其它')from emp;
按工作种类调整工资
select job,sal,decode(job,'CLERK',sal,'SALESMAN',sal*1.2,'MANAGER',sal*1.8,sal) from emp;

四.分组函数：是多行函数，对多行值进行计算，得到多行对应单个结果。

对每个表数据进行分组，得到多组结果，对每组记录进行计算，每组分别返回一个结果，最终产生多个组对应的数据集合
select job,max(sal),min(sal),avg(sal),sum(sal),count(distinct deptno) from emp group by job;
注意：a.所有包含在 select 列表不再组函数中的字段都必须在 GROUP BY 中。
b.group by 可以和 where 搭配，where 只能在 group by 前面，where 子句中不能包含任何组函数。
select job from emp where sal=(select max(sal) from emp) group by job; --此处 where 子句不能直接用组函数，只能使用子查询。
select job from emp where sal>2000 group by job;

使用 HAVING 子句对分组结果进行过滤
select max(sal),min(sal),deptno,sal from emp group by deptno,sal HAVING sal>2000;
组函数的嵌套最多嵌套两层，如果一个使用组函数查询没有 group by 子句，那么组函数不能嵌

套。

五.子查询

语句内部的子句,
Select * From EMP Where SAL>(Select SAL From EMP Where ENAME='ALLEN');

a.注意:
A.子查询总是先于主语句的运行
B.必须有(),表示是一个整体
C.习惯上,把子查询放在条件的右边
D.单行操作符只能接受一个值
单行:> < >= <=等

b.子查询的分类:
A.单行
B.多行

c.单行的语法
select 字段 from 表 where 字段 >[单行] (子查询)

d.多行
in ,all,any
in:表示在一个区间内
```
select * from emp where ename in (
     select ename from emp where sal>2000
)
```

any:
> any:从子查询中取一个最小的
< any:从子查询中取一个最大的

all
>all :从子查询中取一个最大的
<all:从子查询中取一个最小的

六.多表连接

数据来源于多个表，通过表之间的关系取出对应的行数据。查询通常建立在存在相互关系的父子表上
多表连接的分类：等值连接、非等值连接、外连接和自连接四种。从 Orcale 9i 开始增加了对

新的 SQL 标准-SQL1999 的支持。

A.等值连接：通常建立在存在主外键约束条件的多表上，两个字段通过等号建立等值关系。
select * from emp,dept; --产生笛卡尔集：两个表任意组合的结果。
select * from emp e,dept d where e.deptno=d.deptno;
select ename,job,dname,sal,e.deptno,d.deptno from emp e,dept d where e.deptno=d.deptno and sal>2000;
(1)列只出现一次不需要加前缀，明确定义可以提高效率，别名可以提高查找效率。
(2)不同的连接条件可以使用 and 或 or
(3)当然,你可以为表定义一个别名,但是不能使用 as(字段重命名可以使用)

B.不等值连接：一个表中的记录,在另一个表中能够找到匹配的记录即可
Select e.*,g.grade From emp e,salgrade g Where e.sal Between g.losal And g.hisal; --查询员工的工资级别

C.外连接：把不满足条件的数据显示出来
Select * From EMP E,DEPT D Where E.DEPTNO=D.DEPTNO(+)
外连接可以在左表,也可以在右表，但只能出现一次。
左连接 将产生以表达式右边的字段为依据的查询效果，即右边所有数据和左边和右边匹配的数据。

D.自连接：一个表被虚拟成两个表，查看两个相关字段的信息。
select e.ename||' works for :'||e2.ename from emp e,emp e2 where e.mgr=e2.empno; --自查询，查询每个人老大的名称

E.SQL99 新标准下的多表连接
(1)CROSS JOIN：交叉连接
  select * from emp cross join dept; --产生笛卡尔集
(2)NATURAL JOIN:自然连接，默认将名称相同的列进行连接。
  select * from emp natural join dept;
  select * from emp e,dept d where e.deptno=d.deptno; --和自然连接的效果一样
(3)用 using 字句进行指定连接条件连接
  select * from emp Join dept using(deptno);
(4)使用 on 来指定连接具体条件
  Select * From EMP Join DEPT On emp.deptno=dept.deptno And EMP.DEPTNO=20
(5)多个表连接查询
  Select E.ENAME,E.SAL,D.DNAME,S.GRADE
  FROM EMP E Join DEPT D On E.DEPTNO=D.DEPTNO
  Join SALGRADE S On E.SAL Between S.LOSAL And S.HISAL
(6)外连接:
  与内连接的区别:
  内连接========查询条件相等的记录(能够匹配的)
  外连接========条件相等的+没匹配的
  select * from EMP E Left Outer join DEPT D on E.DEPTNO=D.DEPTNO; --LEFT OUTER JION
  select * from EMP E Right Outer join DEPT D on E.DEPTNO=D.DEPTNO; --LEFT OUTER JION
  select * from EMP E Full Outer join DEPT D on E.DEPTNO=D.DEPTNO; --LEFT OUTER JION
  LEFT OUTER JOIN……ON：左外连接，返回左边所有的值及右边匹配的值。(和上面的外连接

刚好相反）

七、数据处理

DML:DATA Market language
insert
update
delete
这三类的操作对数据会产生影响,牵涉到"事务"

1.写入数据:
语法: insert into 表(列1,列2) values(值1,值2)
Insert Into EMP(EMPNO,ENAME,JOB,SAL) Values(1234,'ZHANGSHAN','MANAGER',8000)
rollback
注意:
A.表后面()内表示的是字段名称
Values()内表示是字段所对应的值
B.值要与列所表示的数据类型对应
C.值的个数要和列的个数对应
D.注意顺序,对应
E.没有指定值的列默认为null
查询的时候,可以使用is null来提取
F.写入的数据只在当前窗口内是有效的,打开一个新的SQL Window就无效

如果需要持久性保存下来
需要进行事务处理

事务:所谓事务,就是一组操作单元,事务提交,再语句的结果全部生效,如果回滚,全部撤销

事务操作
A.提交commit
B.rollback
C.回滚到指定的点

--insert的值可以是一个固定的取值,比如sysdate,序列等
Insert Into EMP(EMPNO,ENAME,JOB,SAL,comm,hiredate)
Values(1238,'ZHANGSHAN','MANAGER',8000,Null,Sysdate)
Insert Into emp(empno,ename,hiredate)
Values(2222,'zhangshan',to_date('2008-08-08','yyyy-mm-dd'))

--使用Insert从其他表中拷贝数据
Insert Into emp(empno,ename) select deptno,dname from dept;
insert into temp_1(select * from emp where sal > 3000);

2.使用变量是执行 sql 语句
--在 sql 中,可以是"&变量名称"来定义表名,字段名,值等信息
Insert Into emp(empno,&filedname,hiredate) Values(&请输入编号,'&人员姓名',to_date('2008-08-08','yyyy-mm-dd'))

3.使用 insert 语句从其他表中拷贝数据
Insert Into EMP(EMPNO,ENAME) Select DEPTNO,DNAME From DEPT

注意:
A.字段类型要对应
B.字段个数要对应

4.更新操作
语法:update 表 set 字段=值 where 条件
Update emp Set ename='zhangshan',job='manager' Where empno=7369

注意:
A.不同的字段用逗号隔开
B.如果没有指定条件,默认更新所有的记录

5.在更新语句中使用子查询
```
Update emp Set sal=(
       Select sal From emp Where ename='SMITH'
),job=(
     Select job From emp Where ename='ALLEN'
) Where empno=7521
```

6.关于更新时发生的数据关联问题
如果一个字段的值的来源是另外一个表,比如 emp 表中的 deptno,其值不能随意指定,因为 deptno 的值引用自 dept 表中的 deptno 字段

Update emp Set deptno=55 Where empno=7499 ---error
因为 dept 表中如果没有 deptno 为 55 的记录,会出现数据完整性约束问题

Update emp Set deptno=10 Where empno=7499---ok

7.删除记录
delete [from] table where condition
如:
Delete From emp Where empno=1234

条件可以为多个
Delete From emp Where empno=7369 And ename='SMITH'

8.使用子查询执行 delete 操作
```
Delete EMP Where ENAME=(
Select ENAME From EMP Where SAL=(Select Max(SAL) From EMP)
)
```

9.关于删除的完整性约束:
如果一个字段被另外一个表的字段引用,那个这个值就不能删除

```
delete from dept where deptno=40
```
可以删除,因为在 emp 表中没有某个人在 40 部门下面

```
delete from dept where deptno=30
```
出现数据完整性约束条件错误,因为 30 号部门被引用

10.事务管理
事务:一组操作单元
同进同出
你可以设置一个保存点,必要的时候可以退到某个指定的位置
```
Insert Into EMP(EMPNO,ENAME,JOB,SAL) Values(1236,'ZHANGSHAN','MANAGER',8000)
Savepoint insert_done;
Insert Into EMP(EMPNO,ENAME,JOB,SAL) Values(1235,'ZHANGSHAN','MANAGER',8000)
Insert Into EMP(EMPNO,ENAME,JOB,SAL,comm,hiredate)
Values(1238,'ZHANGSHAN','MANAGER',8000,Null,Sysdate)

Rollback To insert_done
```
有些场合下,事务是自动来提交的
比如创建或修改表的时候

八 、创建和管理数据

数据库表主要分为几大类
A.系统表 B.自定义表 C.数据字典表
```
Select * From USER_TABLES
Select * From USER_SEQUENCES
```

1.命名规则
A.表名长度必须在 30 位以内 B.表名要避开关键词 C.表名要有意义

2.创建表
语法规则类似于:
```
Create Table MYTABLE(Id Number(4,2),Name Varchar2(20),AGE Number(3));
```
你可以在创建表的同时,指定默认值

```
create table DEMO2(
dname varchar2(20),
dage number(3) check (dage  between 10 and 150),
hiredate date default sysdate
);
```

3.查看其他用户的表
以 sys 身份可以看到 scott 用户下的表
```
select * from scott.emp
```
反之,scott 不能看 sys 用户下的表的信息

4.默认值的指定
```
Create Table A1(
Id Number(4),
BIRTHDAY Date Default TO_DATE('2008-08-08','YYYY-MM-DD')
)
```

5.字段类型的定义
char---定长字符串
varchar2----变长字符串

char(20)   abcd---------20
varchar(20) abcd--------4

NUMBER(N):表示整数位不能大于 n 个,小数位可以任意
number(m,n):表示总长度为 m 位,其中小数位为 n,整数为为 m-n

Date 类型:可以使用 to_date 把字符串类型转换为 date 类型
BLOB:大的二进值对象,可以用来存储图片,视频文件等信息

6.使用子查询来创建表
```
create table 表名 as 子查询

create table t3 as select * from emp where sal>2000;
```

7.更新表的结构
alter table
A.追加列
```
Alter Table userinfo Add (addr Varchar2(20) Default 'beijing')
```

B.修改列
```
Alter Table userinfo Modify (age Number(6))
```
使用 modify 可以修改数据类型,长度,默认值
请注意:对默认值的修改,只对后面的数据才有效

C.删除列
```
Alter Table userinfo Drop Column age
```

8.删除表的定义
drop table 表的名称
drop table emp;

9.对表进行重命名
使用 rename .... to
比如：RENAME A TO B
注意：这个表是你的，你是这个对象的所有者

提示：
在 PLSQL 中，添加记录的两种方式
A.可以使用 insert 语句　B.直接在工具里面直接编辑
前提：必须获得一个可更新的结果集
B1:SELECT * FROM EMP FOR UPDATE
B2：select t.*, t.rowid from userinfo t

10.删除表的内容
两种方式
A.delete 删除
B.truncate 清空表的内容

区别：
delete：从数据库的缓存区清除该数据
truncate：把数据删除了，然后清空所占用的空间

delete 可以撤销
truncate 不能撤销
truncate===delete+commit

TRUNCATE 语法
TRUNCATE TABLE EMP;

TRUNCATE 和 drop 区别
drop：删除表的定义，整个对象删掉，删除的是对象的本身，全部
truncate：删除表的内容，只是删除数据，表的结果会保留

11.注释
分两种
A.表　B.字段
表的注释
COMMENT ON TABLE EMP IS 'MY DEMO'

列的注释
COMMENT ON COLUMN EMP.ENAME IS '我的注释'

12.从 plsql 工具中导出表的信息

OLS---EXPORT TABLES---SQL INSERTS

九 、约束

约束:主要是用来限定内容　　作用对象:表
1.约束分类
五大类
A.非空
B.唯一
C.主键:非空+唯一
D.外键:值是外来的
E.检查:值是符合条件

2.注意事项
A.可以给约束起个名字,也可以是使用默认的名字
B.创建的两种时机
　创建表的同时/修改表的定义的时候
C.约束的定义位置
表级/列级
D.在字典表中可以查看到这些约束信息
表的定义:user_tables
约束的定义:user_constraints
使用 SELECT * FROM USER_CONSTRAINTS 语句可以查看到所有的约束信息

注意:以后如果出现:ORA-01400:无法将 null 写入用户.表.字段,表示该字段必须要赋一个值

3.not null 约束
用来修饰字段的　　表示该列的值不能为 null

定义位置:not null 只能在列级
列级:区别于表级
表级:定义的时候,与字段用逗号分开,和字段是同一语句级别
列级:定义的时候,紧跟在字段的后面

```
CREATE TABLE employees(
employee_id  NUMBER(6),
first_name   VARCHAR2(20),
         ...
job_id       VARCHAR2(10) NOT NULL,
CONSTRAINT emp_emp_id_pk PRIMARY KEY (EMPLOYEE_ID)
);
```

上面的例子中
not null=======列级
CONSTRAINT emp_emp_id_pk PRIMARY KEY (EMPLOYEE_ID)代表一个表级的定义

4.unique 唯一约束
是用来修饰一个字段的,内容的 unique 要求字段内容不能有重复

unique 定义位置
A.表级(自定义名称)　B.列级(使用系统默认名称)

基本语法:
列级:字段名 unique
表级:,constraints uni_名称 unique(字段名)

例子:
```
CREATE TABLE T4(
ID NUMBER(4) UNIQUE,--column
NAME VARCHAR2(20),
AGE NUMBER(3),
CONSTRAINTS UNI_NAME UNIQUE(NAME)--table
)
```

5.主键约束
也是用来约束字段的,要求其内容必须不能为 null,也不能重复
PRIMARY KEY:主键==not null+unique

定义位置:
A.表级　B.列级

定义方式
列级:字段名 primary key
表级:constraints pk_id primary key(id)
一般情况下,我们在创建表的同时,都会为表指定一个主键,用来唯一标识一条记录,以便在程序中实现修改,删除等业务逻辑操作时,根据该主键标识来准确定位到要操作的记录

联合主键:
把多个字段合起来作为一个主键,要求多个字段每一项都不能为 null,并且合起来的值不能重复.

6.外键约束
外来的键值,如 EMP 表的 deptno 字段引用了 dept 表中的 deptno 字段
我们认为:emp 表中的 deptno 是一个外键,emp.deptno 不能任意指定,只能从 dept.deptno 的值中任意选一个

外键约束定义的位置:
A.表级　B.列级

语法定义格式:constraints 自定义名称 FOREIGN KEY (字段名) references 表名(字段名)

使用外键约束的条件：如果 A 表引用 B 表中的字段 id

条件有两个：B 表中的 id 必须有主键约束 或 唯一性约束

删除表的方式:A.drop table　B.工具中 drop

列级外键定义:直接在列名的后面 references dept(deptno)

7.关于外键的删除级联设置
联动的效果，我们可以在父项进行删除的时候,让子项联动

两种方式
A.on delete cascade:级联删除
B.on delete set null(不引用)

外键引用时的值的选择：A.要么为 null　B.要么从父表中取一个值

8.CHECK 约束
check:对某个字段的值的约束,要求该字段的每一行都必须满足的条件,它以一种很直观的方式,来限定你的行值

```
CREATE TABLE STAFF(
STAFFID NUMBER(5),
NAME VARCHAR2(20) CHECK(NAME IN ('A','B','C'))
)
```

定义位置:
可以表级也可以是列级

9,约束的添加
语法:alter table 表名 add primary key (字段名称)
实例:ALTER TABLE STAFF ADD PRIMARY KEY (NAME)

10.删除已存在的约束
ALTER TABLE 表名 drop constraints 约束名
比如:ALTER TABLE STAFF DROP CONSTRAINTS SYS_C003042

11.级联删除约束
在删除约束的同时,也删除跟该字段有关系的约束设置
ALTER TABLE DEPT DROP CONSTRAINT PK_DEPT CASCADE
该代码执行后:emp 表中的外键引用就不存在了

12.有效化和无效化约束
使用 disable 和 enable 关键词
ALTER TABLE EMP DISABLE CONSTRAINT PK_EMPNO
ALTER TABLE EMP ENABLE CONSTRAINT PK_EMPNO

13.从数据字典中查询
select * from user_constraint:查询对应的表

SELECT * FROM USER_CONS_COLUMNS:查询对应的列

十 、视图

10g：以 DBA 的身份登录（sys，Oracle），在 Command Window 中输入以下命令：grant create view to scott;

视图:逻辑上的数据集合
逻辑上:区别于物理上
物理上:表示真实的,现实存在
逻辑上:虚的数据,来源于真实的表

比如,我们可以为 emp 创建一个视图:
CREATE OR REPLACE VIEW EMP_VIEW AS SELECT EMPNO,ENAME,JOB,HIREDATE FROM EMP

以后如果想要 EMPNO,ENAME,JOB,HIREDATE 这些数据,就可以不必访问 emp 了,直接从视图中来查找
作用:a.控制数据访问 b.简化查询 c.数据独立性 d.避免重复访问相同的数据

1.创建格式:
create or replace view 视图名词 as 数据集合(通常是一个查询语句)

2.在创建视图的时候,可以使用别名
CREATE OR REPLACE VIEW EMP_VIEW AS
SELECT empno employee_number,ename employee_name,job type,hiredate birthday FROM EMP

3.修改视图:
create or replace view myview as select...

4.视图创建的规则
视图的内容可以是任意的集合
```
CREATE OR REPLACE VIEW EMP_VIEW AS
SELECT JOB,MAX(SAL) AS MAXSAL,MIN(SAL) AS MINSAL,
(MAX(SAL)-MIN(SAL)) AS SAL,COUNT(*) NUM,AVG(SAL) AVGSAL
FROM EMP GROUP BY JOB
HAVING JOB IS NOT NULL
```

5.区别:
A 语句:
```
CREATE OR REPLACE VIEW JOB_VIEW AS
      SELECT DISTINCT(JOB) FROM EMP WHERE JOB IS NOT NULL
```
B 语句:CREATE TABLE JOB_TEMP AS SELECT DISTINCT(JOB) FROM EMP WHERE JOB IS NOT NULL

主要区别:
共同点:都做到了:数据被独立出来

最大不同:JOB_TEMP 是一个物理的表,是要占用很多的数据库空间的,
相当于硬盘上的资料被拷贝了一份

JOB_VIEW:数据还是同一份,不会多占空间,JOB_VIEW 相当于是一个"快捷方式"

视图数据的来源===基础表
对基础表的数据的改变将会影响到视图,视图本身并不保存数据,只是对相关数据集合的一个引用而已
简单视图操作的数据和基础表中的数据是"同一份"

复杂视图不能使用 DML:
原因:数据已经被整合了,如果被改变,无法和基础表的数据对应了

6.删除视图:
drop view 视图名称,删除视图是对基础表不会产生任何的影响

7.TOP-N 分析:
使用到一个关键的列名:rownum
rownum:行编号
伪列:虚拟的列,你并没有在表中人为的添加这个列,
但是当你使用 rownum 关键词,oracle 服务器能够解析

使用的方式
rownum<或<=
因为它总是从第一行开始

例外:
ROWNUM=1 不会为 null

分页查询的变通方式:查询第三行到第八行之间的数据
```
SELECT * FROM (SELECT e.*,ROWNUM AS RN FROM EMP e) WHERE RN BETWEEN 3 AND 8;
--方法二
select * from emp where rownum <= 8
minus
select * from emp where rownum <3;
```

十一 、其它数据库对象

数据库主要对象:表、视图、序列、索引、同义词
表:数据集合
视图:逻辑上数据集
序列:有规律数值(不为 null 且唯一的键值)
索引:定位记录

同义词:类似于别名

1.序列:
A.共享的　B.提供有规律的数值

提取方式:序列名.nextval
select seq.nextval from emp
INSERT INTO EMP(EMPNO) VALUES(MYSEQ.NEXTVAL)

2.查询已定义的序列
SELECT * FROM USER_SEQUENCES

3.使用序列的约束
nextval 下一个值 ， currval 表示当前值
currval 的值必须在 nextval 被指定后才有效

4.删除序列
drop sequence myseq

5.索引:
一种典型的数据库对象
作用:提交数据的查询效率,尤其对一些数据量很大的表
索引是用来为表服务的
索引是oracle服务器自动来使用(索引区的查找是oracle自动的)和维护(当记录发生变化的时候,索引区会自动随之更改)

6.定义
定义的两种方式
A.自动创建:
主键 或 唯一性约束

B.手动创建
CREATE INDEX ENAME_IDX ON emp(ename)

7.索引的创建时机问题:
两种情况下一般都要创建一个索引:
列经常被用来做条件查询的时候
表数据量很大的时候

下列情况不要创建索引:
表很小
列不经常作为连接条件或出现在 WHERE 子句中
查询的数据大于 2%到 4%
表经常更新

8.同义词
为对象起一个同义的名称

目的:为了缩短对名称长度的引用

创建方式:
synonym:CREATE SYNONYM 同义词名词 for 对象(scott.emp)

```
CREATE SYNONYM SN_EMP FOR  SCOTT.EMP
CREATE SYNONYM SN_DEPT FOR SCOTT.DEPT
SELECT * FROM SN_EMP E,SN_DEPT D WHERE E.DEPTNO=D.DEPTNO
```

9.集合操作符
三大类
合集:union/union all
交集:intersect
差集:minus

合并集合:
```
SELECT * FROM EMP WHERE SAL>2000
UNION
SELECT * FROM EMP WHERE JOB='MANAGER'
```

注意:
A.不同的表合并时,要注意字段个数要对象
B.类型要对应
C.显示的字段是第一个表的字段列名为新的集合的列名
使用 union all 保留重复的记录
intersect 交集
不保留重复的
minus

分页查询
两种方式
```
SELECT * FROM (
       SELECT ENAME,EMPNO,JOB,ROWNUM RN FROM EMP
) WHERE RN BETWEEN 3 AND 9
```
A.使用 rownum 限定
```
select * from emp where rownum <= 8
minus
select * from emp where rownum < 3
```
B.minus 集合操作符

10.关于 dual 表的使用
dual 是一个虚表
其表中只是返回一条记录
作用:A.计算数值
     B.提取系统的某些值,比如 user,sysdate
     C.转换

DUAL 和其他的表的区别
SELECT UPPER('smITh') FROM DUAL　　只返回一条
SELECT UPPER('smITh') FROM EMP　　返回 N 条