数据库相关操作

创建数据库：create database 数据库名；

重命名：alter database 数据库名 rename to 新数据库名；

删除数据库：drop database数据库名；

模式操作与其一样把database改成schema即可；

1. Select \* from tab;查看所有表; DML:数据操作语言 insert delect update select；
2. Set linesize 300; DDL：数据定义语言 create drop/truncate alter table;
3. Set pagesize 300; DCL：数据库控制 噶、grant、revoke等
4. Desc emp; 查看表结构；
5. Select empno as “编号”，ename “姓名”，job 工作 起别名的三种方式；
6. Select \*from emp where ename like ‘%\-%’ escape ‘\’; 特别字符转译；
7. 数字、日期——字符，可以隐式转换
8. 图：数字-----------字符--------------日期 用to\_char to\_date to\_number三个函数转化。

比如to\_char(‘￥123.456.7’,’L9999,999.9’)前边的要转的，后面的转化后的格式。yyyy-mm-dd

1. Select \*from emp where empno not in(10,20,30,null); × 因为not in 里不得有null 否则结果为空 not in 相当于（!=） is null , is not null √
2. Select empno，ename，sal from emp order by sal desc nulls last 系统默认null最大，如果想把null排后面，加上 nulls last语句；
3. Case 目标对象 when…..then……

when….then……

else……

end

1. Order by 所以语句后使用
2. Begin 批量增加

Insert into emp(字段1,字段2,字段3) values(123,’adasd’666666,’19-9月-88’)；

Insert into emp values(123,’adasd’666666,’19-9月-88’)； **增一行**

Insert into emp values(123,’adasd’666666,’19-9月-88’)；

end；

1. Update 表名 set ename=’x’,job=’y’ where empno >7900; 如果不加where条件 所有列的值都会变x，y； **改一行内容**
2. Delete from emp where nmpno=7900; 如果不加where条件 所有行都会被删除；**删一行**

**创建表**

1. Create table mytab6

(

id number,

name varchar2(10),

age number

);

1. Create table mybat1 as select \* from emp； 复制表 ， 插入同理
2. Create table mybat1 as select \* from emp where 1=0; 快速创建表结构
3. Insert into mytab1(empno,ename,sal) select empno , ename , sal from emp; 在旧表中插入已存在的表
4. Set timing on/off 开/关 所有时间；
5. alter table emp move; 整理碎片；
6. vacuum analyze emp;
7. vacuum full emp; postgreSQL的碎片整理

**修改表：**

1. 追加新列：alter table mytab1 add myother varchar2(10);
2. 修改列(长度)：alter table mytab1 modify myother varchar2(20)；
3. 修改列(类型)：alter table mytab1 modify myother number；
4. 删除列：alter table mytable1（表名） drop column myother（列名）；
5. 重命名列：alter table mytable1 rename column myother to newmyother;

**删除表**：

1. Delect from emp； 删除表中数据 可退回； 退回指令：rollback；
2. Truncate table emp；删除表中数据 不可退回；不记日志 DCL、DDL自动commit。
3. drop table mytable1； rollback 可恢复；
4. Drop table mytable1 purge； 彻底删除 不可恢复 相当于删除并同时清空回收站
5. Purge recyclebin；清空回收站；
6. Show recyclebin； 查看回收站；
7. 还原-------“闪回”技术；

**添加约束 :**

1. Unique 可以有null
2. Check (length(address>)>2)
3. not null
4. primary key 不能有null
5. foreign key
6. default
7. alter table student add constraint pk\_stuno primary key(stuno);
8. alter table 表二 add constraint fk\_student\_sub foreign key(subid) references 表2(sid);

alter table – add constraint fk\_student\_sub key(subid)

（在表外加的约束）只有4个：主键、外键 唯一 检查，表级约束

约束例子：studen表和sub表来举例子

Create table sub(

Sid number(3),

Sname varchar(30)

）

Create table student(

Sno number(3) constraint PK\_stuno primary key,

Sname varchar2(10) onstraint NN\_sname not null constraint UQ\_sname unique,

Saddress varchar2(20) default ‘陕西西安’ constraint CK\_saddress check(length(address)<2),

Subid number(3),

constraint PK\_stuno primary key(sno,sname), 加两个字段名叫复合主键

constraint UQ\_sname unique(sname,sid) 加两个字段 他俩都唯一的意思

constraint FK\_student\_sub foreign key(stuid) references sub(sid) on delete cascade/set null 在外键后加上级联删除/级联置空

)

**\*加约束：**建完表后再约束，分为两类

1．唯一 、检查、主键、外键

alter table 表名 add constraint 约束名 约束类型

Alter table student add constrain UQ\_saddress unique(saddress);

Alter table student add constrain FK\_sno primary key(sno);

Alter table student add constrain CK\_saddress check(length(address)<2);

Alter table student add constrain FK\_syudent\_sub foreigh key(subid) references sub(sid)

2. 非空 、默认

Alter table 表名 motify 字段名 constraint 约束名 约束类型

Alter table student modify stname constraint NN\_sname not null

Alter table student modify stname default ‘没有名字’ ； 注意默认约束比较特别 一般不写约束名，并且不写constraint （没有constraint当然不用写名字了）

**\*删除约束**

Alter table 表名 drop constraint 约束名（就是你起的那个）

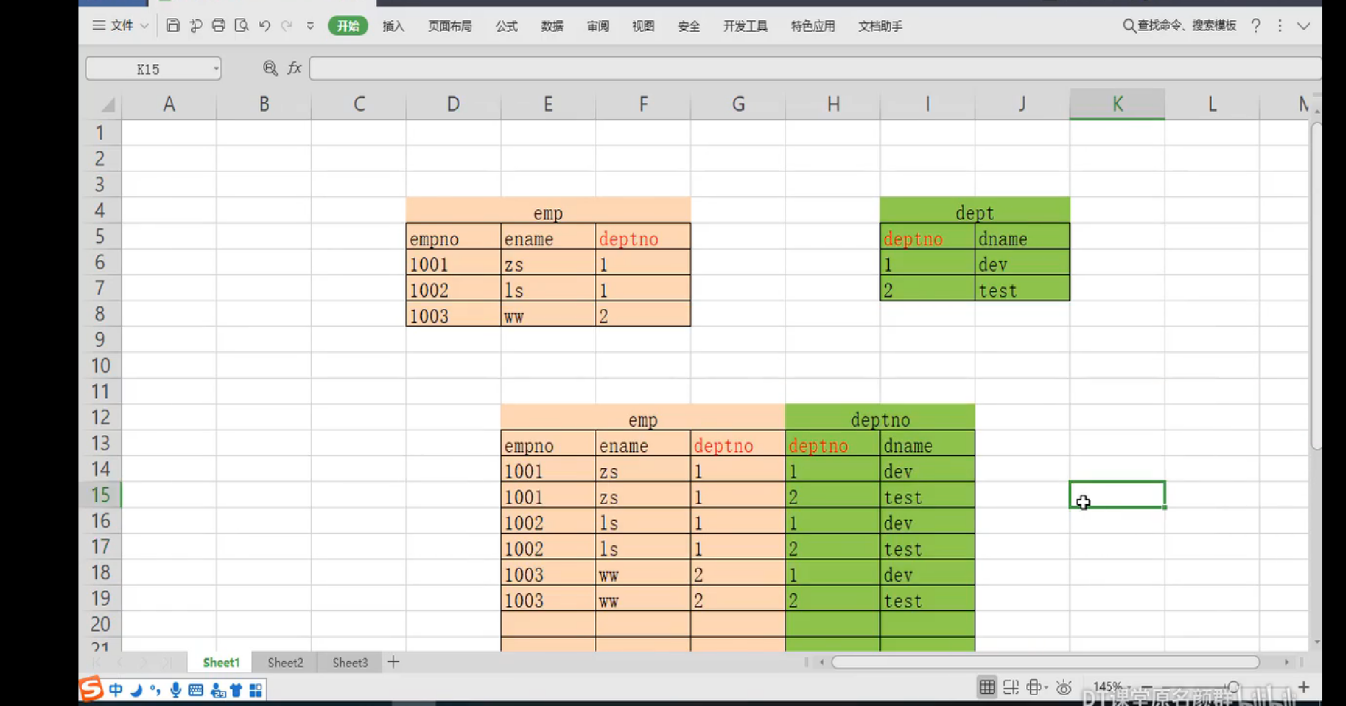
特殊：删除默认约束Alter table student modify stname default null；“跟加一样”

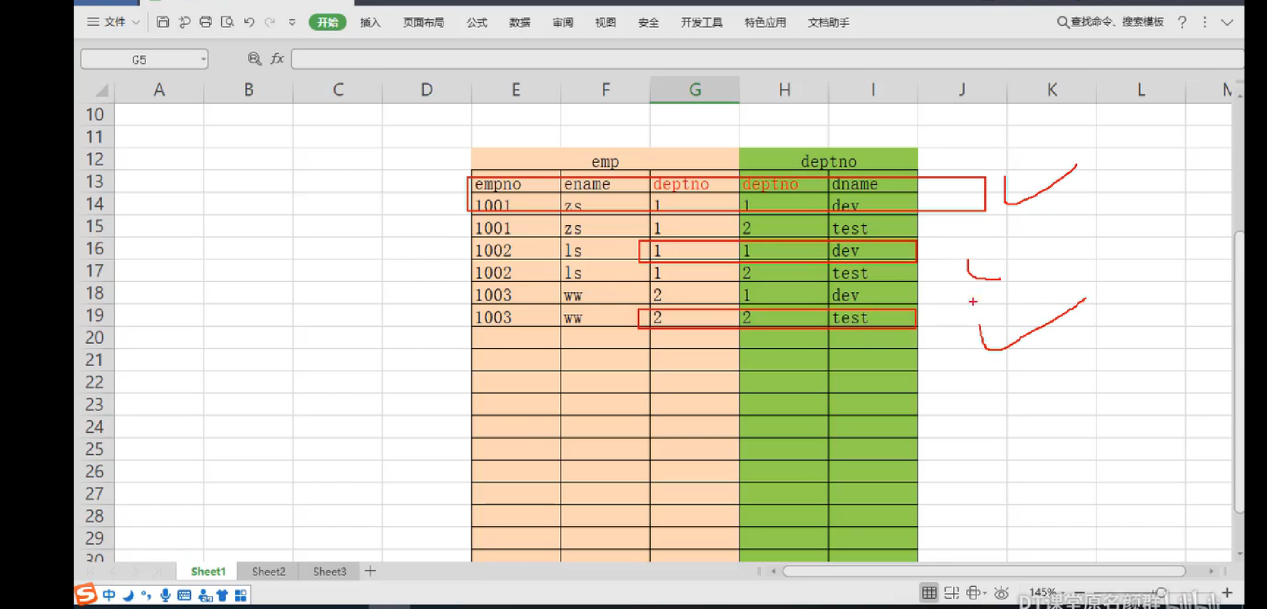
**创建保存点：**

**savepoint 保存点名；例如：savepoint initdata01;**

**rollback to savepoint 保存点名；rollback to savepoint initdata01;**

**多表连接查询：**

****

****

1.内连接：多张表通过相同字段匹配，只显示匹配成功的数据。不成功不显示

1. select \* from emp e,dept d where e.deptno = d.deptno;
2. Select \* from emp e inner join dept d on e.deptno =d.deptno;

2.外连接

（1）左外连接：以左表为标准（左表全显示），去匹配右表如果匹配成功 则全部显示，匹配不成功则部分显示。（无数据部分用null填充）

①select \* from emp e,dept d where e.deptno = d.deptno(+); ----oracle独有

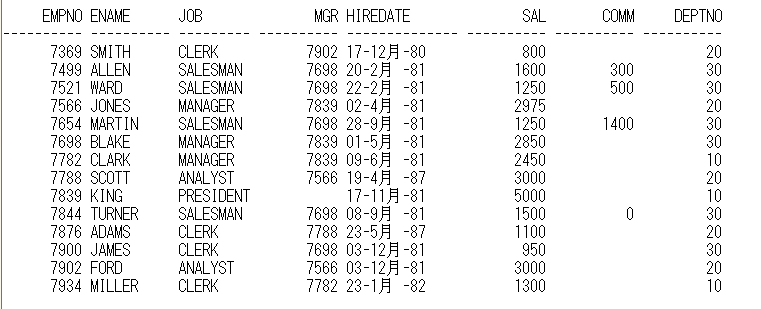
②Select \* from emp e left outer join dept d on e.deptno =d.deptno;

（2）右外连接：

①select \* from emp e,dept d where e.deptno(+) = d.deptno; ----oracle独有

②Select \* from emp e right outer join dept d on e.deptno =d.deptno;

1. 全外连接：左外+右外 - 重复
2. 自连接：将一张表 通过别名 “视为”不同的表 ------比较费性能

\*（1）查询 员工姓名，以及该员工的领导姓名。e.empno=b.mgr ✖

Select e.ename,b.ename from emp e,emp b where e.mgr= b.empno; ✔

员工表看 ename mgr 领导表看 empno ename

//员工表的领导编号mgr = 领导表的 员工编号empno.

（2）层次连接（了解） select level,empno,mgr from emp connect by prior empno = mgr start with mar is null order by level;

1. 子查询(不出现null，不排序)

①查询比SCOTT工资高的 员工信息；

Select \* from emp sal >(select sal from emp where ename =’CSOTT’);

注意：子查询可以写在 where, select,having,from 不能写在 group by 后面。

**\*②查询最低工资比10号部门的最低工资高的部门编号；**

Select deptno,min(sal) from emp

group by deptno

having min(sal) >(select min(sal) from emp where deptno=10);

**③把子查询放到from后**相当于一个新表：

select \* from(select empno,ename,sal\*12 from emp);

④查询销售部门的员工信息；

第一步：根据销售部所在表查询，销售部门编号30；

Select deptno from dept where dname=’SALES’;

第二步：根据部门编号30查询员工信息

Select \* from emp where deptno=(Select deptno from dept where dname=’SALES’)

any与all的应用：

Select \* from emp where sal>(select min(sal) from emp);

Select \* from emp where sal>any (select sal from emp);

Select \* from emp where sal>(select max(sal) from emp);

Select \* from emp where sal>all (select sal from emp);

查询不是领导的员工信息（子查询时排除null）：

第一步 判断是不是领导：即判断empno是否存在于mgr中，存在 是领导 否则不是。

Select \* from emp where empno not in (select mgr from emp where mgr is not null);

子查询取前多少：（前三名的工资）

令P = (select \* from emp order by sal desc)

select rownum,ename,sal from P where rownum<=3;

删除重复数据：一个一个删

Select rownum,s.\* from mystudent s;

ROWNUM STUNO STUNAME STUAGE

1 1 ZS 23

2 1 ZS 23

3 2 LS 24

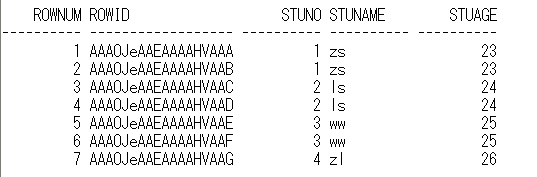
4 2 LS 24

5 3 WW 25

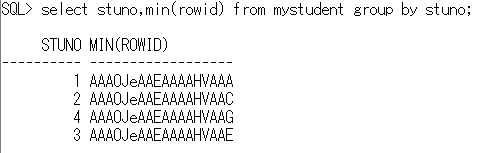
如果不加伪列 delete from mystudent where wtuno=1; 此时zs都删了

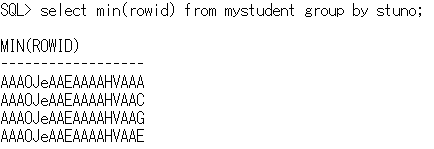
所以：delete from mystudent where rownum=1;

删除重复数据：一起删



delect from mystudent where rowid not in(select stuno,min(rowid) from mystudent group by stuno);





**三大范式:**

1. 每列保持原子性
2. 非主键列依赖于主键列

//a🡪b🡪c 不能b依赖于a ,c依赖于b从而间接依赖于a

1. 非主键列不传递依赖于主键列

**视图**：

授权grant create view to scott(用户名) 撤销revoke create view from scott(用户名)

**创建**

Create view myempview

as

select empno,ename,deptno from emp where deptno =20 with read only;

两个小约束**：**

with read only; 只查看

with check option; 操作时必须在原有where条件里操作

\*\*在执行更新操作中，如果针对的单张表，则可以批量更新：

update xx set xx = xx where……;

但如果是“多表连接的表（视图）”则必须明确，不能有歧义。

Update myview set ename = ‘hello’ where empno = 7934;

事物：单个逻辑工作单元执行的一系列操作

原子性：要么都成功，要么都失败

一致性：事物执行前后总量保持一致

隔离性：各个事物并发执行时，彼此独立。

持久性：持久化操作（开关机后数据不恢复）

**多个事物产生很多并发问题：**

1. 脏读：别人拿到数据已经要修改为2了只是没提交，你还看的1
2. 不可重复读：多次读取，结果不同
3. 幻读（虚读）：读取同一批数据，结果不同

**为解决这些问题**

**提出隔离级别：解决能力依次提高，但效率依次降低（线程同步与非同步）**

①读未提交

②读已提交

③可重复读

④序列化（串行

化）：相当于同步代码块

Oracle只支持②和④

切换隔离级别：set transaction isolation level serializable；从②切换到④

序列：本质，内存中的数组

创建序列：create sequence myseq1 删除序列drop sequence myseq1

修改序列：把create变成alter.(只影响修改后再用的地方)

设置属性：

Increment by 2 设置步长为2

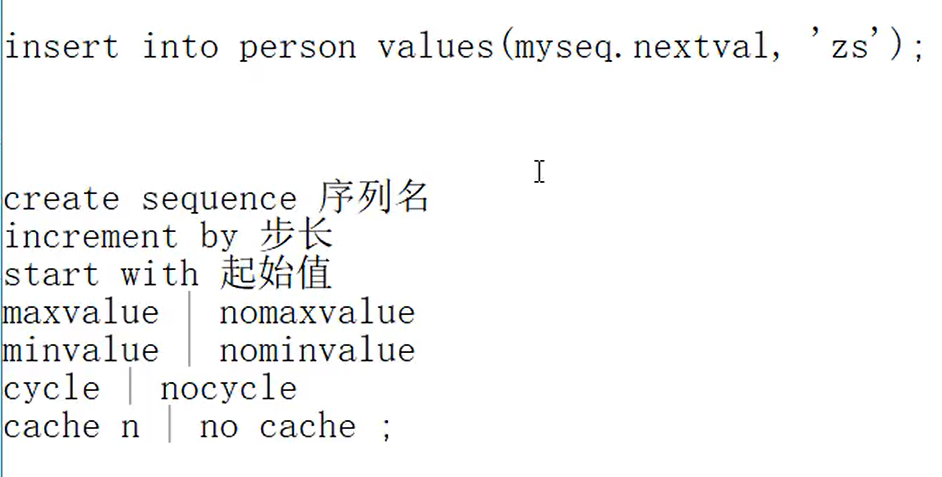
Start with 100 开始值为100

myseq1.nextval 起点在空 用的时候先myseq1.nextval

myseq1.currtval

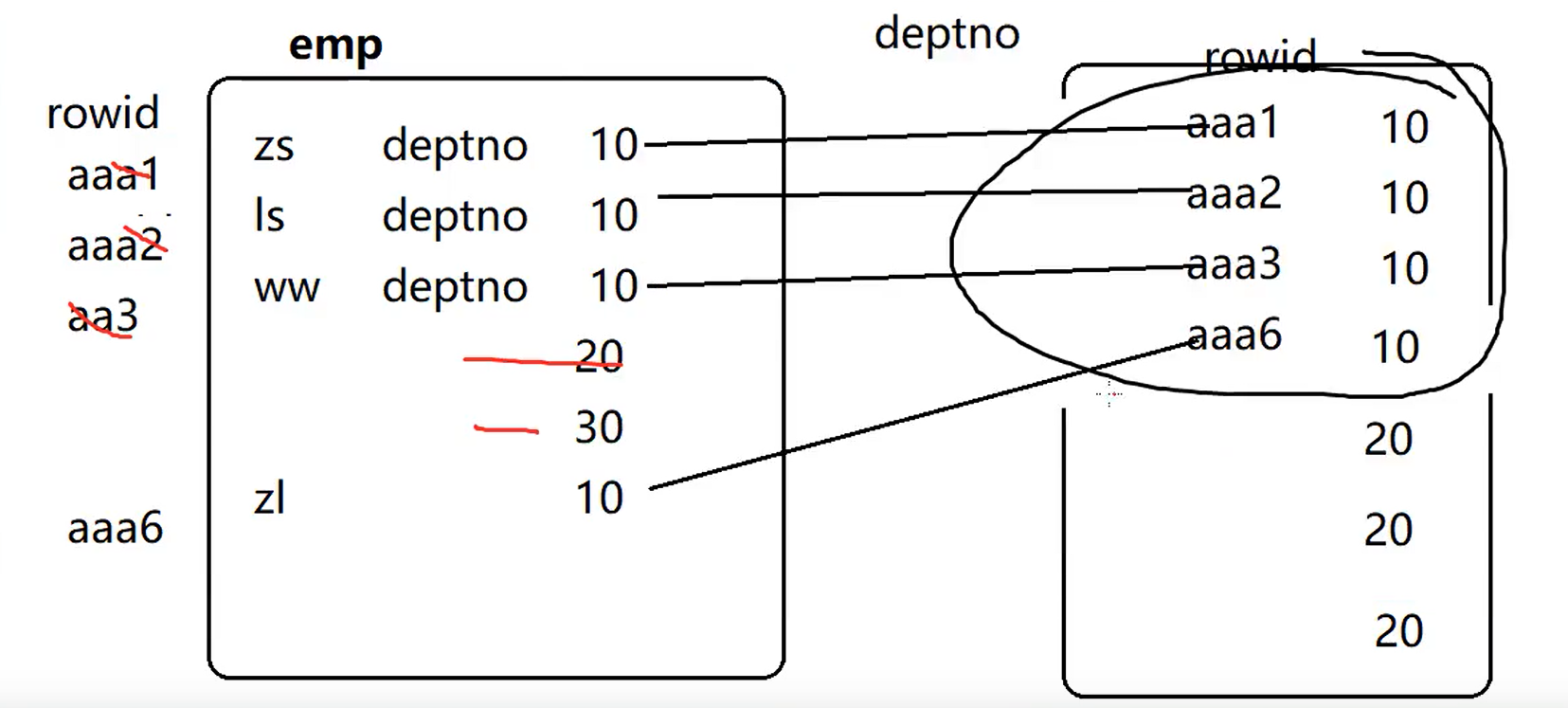
例：select myseq1.nextval,empno,ename from emp;

**如下图：**



裂缝问题：加入给a,b,c的序列是1，2，3 然后你发现弄个错了把abc又给删掉了，重新再来 序列从4开始，也就是说分配出去的序列不能再回来。

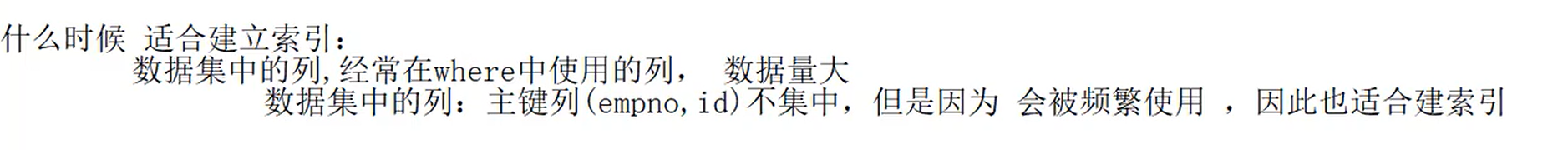
索引：相当于书的目录



创建索引：create index myindex on emp(deptno) 给emp表的deptno列创建索引。

create index myindex on emp(deptno , sal)二级索引。

主键默认索引。



游标：相当于集合，里面可以存放查询出内容的‘整体’rs1.1,rs1,2

语法：cursor cemp is select ename,sal from emp;

用的时候必须先打开：open cemp;

触发器：相当于JS里的事件onclick……….

# 查询补充

Group by a , b , c, 的用法：

先按照a分组，如果a相同再按照b分组，如果b相同再按照c分组，最终统计的是最小分组的信息。

记住：使用了group by 之后 select 中只能出现分组后的整体信息，不能出现组内的详细信息。

Select deptno , job , avg(sal) 平均工资 , count(\*) 部门人数 ， sum(sal)

From emp

Group by deptno , job

Order by depyno

把姓名不包含A的所有的员工按照部门编号分组，统计输出部门平均工资大于2000的部门的部门编号，部门的平均工资。

Select deptno , avg(sal) ,平均工资

from emp

where ename not like ‘%A%’

group by deptno

having avg (sal) > 2000

三表连接

Select e.ename , d.dname , s.grade

from emp e

join dept d

on e.deptno = d.deptno

join salgrade s

on e.sal >=s.losal and e.sal <= s.higsal

where e.sal >2000

注意：

Select \* from emp , dept

Where deptno = 10 ✖

原因：

Select \* from emp , dept 此时是两个表已经变成一个临时表了（没加条件是笛卡尔积表而已） 在这个表中有两列deptno（一个是emp表中的，一个是dept表中的）当执行where deptno = 10的时候不知选哪个deptno 所以要写成 :

Select \* from emp , dept

Where emp.deptno = 10

或者，

Select \* from emp , dept

Where dept. deptno = 10

内连接注意：

Select \* from emp , dept

Where dept.deptno = 10 -----过滤条件，不是连接条件

下面的写法是不等价的 是错误的

Select \*

from emp

join dept

on emp.deptno = dept.deptno ✖

having dept.deptno = 10 ✖

错误1：where后人家是过滤条件不是连接条件 你写on链接条件里干什么

错误2：having是写在有分组语句后面的过滤语法 这里没有分组 你用having干什么

正确写法：

Select \*

from emp

join dept

on 1=1 （当没有连接条件的时候随便写个恒等式就行）

where dept.deptno = 10

华宇

视图：

limit 1 输出几行

offset 1 输出（除limit）之后的多少行

常见函数用法总结

nullif(value1，value2)如果value1=value2返回null，否则返回value1

nvl(comm,0)如果comm为空就取0

nvl2(comm,comm,0)如果comm不为空就取comm，如果为空取0

coalesce(comm，value1，value2，……)如果comm为空就往后找，直到找到不为null的

String\_agg(字段名,“分割符合”)

Lower(ename)全小写

Upper(ename)全大写

Initcap(ename)首字母大字

Substr(ename,1,3)对ename的值进行切割 从1开始 一共切割3个凡是有数字的地方都可以用random()\*10或者\*100等待自己调整去替换数字，函数的结合使用。

Length(ename)求长度

Instr(ename,’A’)求ename值中‘A’的数目

Lpad(sal,10,’\*’)让sal值保持10位 不够从左边补\*

Lpad(sal,10,’#’)让sal值保持10位 不够从右边补#

Trim(ename)清空两边空格

Replace(ename,’A’,’a’)把ename中A替换为a

Round(45.943) 46 不带参数

Round(45.943,2) 小数点后两位 45.94

Round(45.943,0) 个位 46

Round(45.943,-1) 十位 50

Mod(sal,300)求余 工资除以300余数是多少

To\_date(‘1987-12-17 18:21:08’ , ’ YYYY-MM-DD HH24:MI:SS’);

To\_char(sal,’L99,999.9999’)

结果￥1.600.0000

To\_char(hiredate,’YYYY-MM-DD HH:MI:SS’)

结果1987-12-17 12:00:00

To\_char(sysdate ,’YYYY-MM-DD HH24:MI:SS’)

结果1987-12-17 18:21:08

To \_number(‘$1,250.00’ , ’$9,999.99’)

nvl(comn,0) 如果comn的值为null就替换成0，一般在计算中用，因为null不能参与运算。

Select count(distinct deptno) from emp

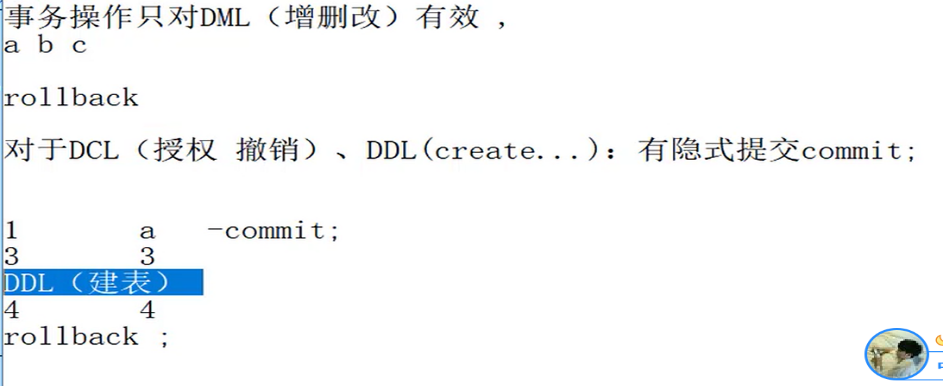
不重复的部门编号有多好个 distinct的嵌套使用

01：数据库操作时关键字大小写不敏感，数据大小写敏感。

02：对试图一般只进行查询，不推荐增删改操作。比如，某个试图有某个表两个字段，你往试图里增加数据，同时也会对应往表中增加数据，这时候试图中没有的 在表中默认为null，问题来了，如果包含主键，你把主键默认为null了就必然报错。在创建视图的时候加上:with read only，那么该视图就只能查询，不能增删改了。

03：事物操作（比如rollback）只对DML（增删改）有效。

对DCL（授权，撤销）、DDL（create,…..）有隐式提交commit。



04: delete属于DML Truncate属于DCL

对已经提交的数据进行恢复：闪回技术：undo\_sql

05:join….on…. 和 using()和 natural 区别

select \* from t1 inner join t2 on t1.c = t2.c 也可以使用 using(c)

但join on 会保留相同字段 using()只留一个

自然连接select \* from t1 natural join t2; 和using一样相同字段也只保留一个。

相当于隐式的using(c,x) 不同之处在于 自然连接不能写连接条件 写会报错。

06:

>any(10 , 30 , 5) 存在大于里面的就行 等价于>5

>all(10 , 30 , 5) 必须大于里面的每一个 等价于>30

07：unique 唯一约束不适用于null 可以存在多个null

08：创建表时 对列进行约束，default默认约束 必须放在第一个。否则报错。

09:对视图进行DML曾删改时原表也跟着改，同样，对原表进行DML时视图也改。

类似于C语言的指针，Java中的引用，原表和视图指向的同一片数据。

但drop视图原表不受影响，但drop原表也会drop视图。

10：drop…………….如果删不掉 说有东西在用 在后面加cascade，级联删除。

11:插入多行：Insert into emp(字段1,字段2,字段3) values

(121,’adasd’666666,’19-9月-88’)，

(122,’adasd’666666,’19-9月-88’)，

(123,’adasd’666666,’19-9月-88’)；

12：select \* from generate\_series(1,100,2);

12：select generate\_series(1,100);