sql语句：DDL（定义）、DML（操作，DQL 查询）、DCL（控制）

## DQL语句

对复杂逻辑可以分步逐步查询以得到最后结果

### sql语句格式规范：

1. **系统保留字、内置函数、sql保留字大写，其它小写，字段值依实。**
2. **注释，单行 -- 多行 /\* \*/**
3. **where条件格式：a、每个条件占一行，语句另起一行；b、以保留字或连接符开始，连接符右对齐**
4. **SQL命令大小写不敏感、可写成一行或多行，不过关键字不能跨多行或者缩写**
5. **子句位于独立行——易维护和理解**

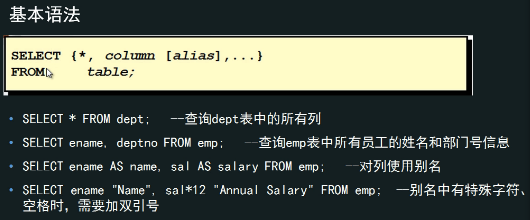
库表格中字段不分大小写

字段值用单引号表字符串。字段值分大小写,且是字符串时要用单引号，

**字符串用于字段值用'' ，用于字段别名用""。引号内分大小写**

## DQL单表查询

查询格式： select 字段 from 表名;



AS可省略，换用空格

别名查询： select 字段 字段别名 from emp; select 字段 AS 字段别名 from emp;

SELECT JOB 工作,EMPNO 员工编号,DEPTNO FROM EMP tabname;

条件查询where： select 字段 from 表名 where 条件;

select ename from emp where ename='JONES';

where条件支持:1、放在from句后，2、between..and..，column in (值,值,值)，like，is null

select \* from emp where empno>=7700 and empno<=7900;

（逻辑运算符，运算优先级：比较>NOT>AND>OR）



select \* from emp where empno between 7700 and 7900;（between and 包含边界值）

select \* from emp where deptno!=10；等价于select \* from emp where not deptno=10;

一个条件 一个’非条件’

非条件：not 字段

范围查询in；column in (字段值1,字段值2,字段值3)

~~注意，没有 column = 值1 or 值2的写法;括号内含null时返回为空~~

SELECT \* FROM emp WHERE empno in (7839,7844,7876,7900,7902) and deptno = 10;

单行子查询：用=、>、<、<>、等等连接子查询作为where子句

多行子查询：用in、all、any连接子查询作为where子句: >all、<all、>any、<any

空，无值 null：

select \* from emp where not comm is null; not 字段

select \* from emp where comm is not null; 也可以is not ，仅用于is

模糊查询：column LIKE ‘关键词%\_’;

LIKE ，通配符 %，表示所有字符 \_表一个字符 转义符： escape ‘\’

SELECT \* FROM emp WHERE ename LIKE ‘%S’;

SELECT \* FROM emp WHERE ename LIKE ‘\_\_\_\_’;

SELECT \* FROM emp WHERE ename LIKE ‘\_\\_\_’ ESCAPE ‘\’; 查询含x\_x的

如 '%E\_' 表示倒数第二个字符为大写E的值



函数：nvl(column,b)

null的算数表达式为null，。 表示字段值为空：column is null

select sal+comm from emp;



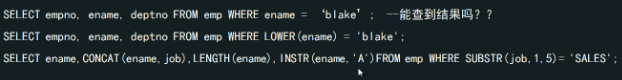


nvl(字段,b)替空值函数：把字段的null换成b显示，非空值不换：

select sal+nvl(comm,0) from emp;

字符函数：





其它常用统计查询函数：聚合函数

max(column) 最大值 min(column) 最小值 AVG(column) 平均值

sum(column) 总和 count(\*) 计数

select count(\*) from emp;

select min(ename) from emp;

select \* from emp where sal - avg(sal) >-500 and sal -avg(sal) <500; ——×

聚合函数不能作为where子句直接使用→用子查询嵌套 where sal -(select avg(sal) from emp) ....

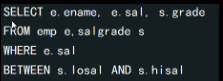
排序查询：ORDER BY column， ASC(升序ascending, 默认) , DESC(降序)

SELECT \* FROM emp ORDER BY sal ASC;

SELECT \* FROM emp ORDER BY sal ASC,deptno DESC ;优先按sal升序后按deptno降序

order by在查询语句后执行(根据查询结果,有别名)，区别于where子句(查询时执行，此时查询的字段的别名还未执行)。而from后的表名别名则先于wher，selecte执行

别名定义顺序：from别名化 →where别名使用→select别名化→order by别名使用

可运行

分组查询：GROUP BY column

SELECT count(\*),deptno FROM emp GROUP BY deptno;

分组条件查询，用having对分组后进行条件过滤。分组后用having对聚合函数进行限制，分组前用where

SELECT job,count(\*) FROM emp WHERE deptno > 10 GROUP BY job HAVING count(\*)>2;

注:having会在group by的结果上进行筛选，若不能在group by结果上进行筛选，则会报错，如：SELECT count(\*),deptno FROM emp WHERE deptno != 10 GROUP BY job；。按组分后，一个 组含多个deptno，无法同行显示。

SELECT count(\*),deptno FROM emp WHERE deptno != 10 GROUP BY deptno; 等价于

SELECT count(\*),deptno FROM emp GROUP BY deptno HAVING deptno !=10;

**使用注意点：**

**1、group by 必须结合聚合函数使用**

**2、查询的普通字段必须出现在group by 子句中**

**3、查询普通字段,聚合函数时必须group by**

**4、分组后过滤，对聚合函数的作限定，只能用having：having avg(sal)>2000。且用在group by 之后**

keep：6、列出各个部门的MANAGER（经理）的最低薪水。

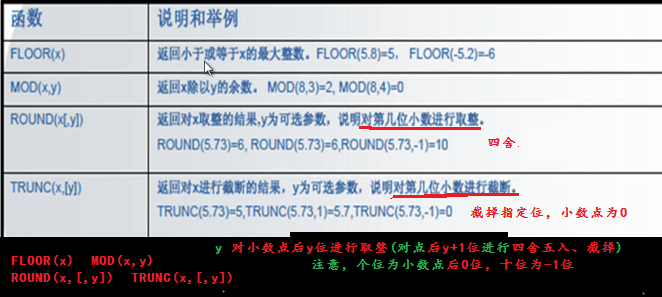
select min(SAL),deptno from emp where job = 'MANAGER' group by deptno;

——having 用以对聚合函数作限定

ROUND(x[,y]) TRUNC(x[,y])

**y 对小数点后y位进行取整(对点后y+1位进行四舍五入、截掉)**

**注意，个位为小数点后0位，十位为-1位**



rownum 隐藏字段，不可运算→解决→ 别名化

~~\*,rownum~~ × \*后只能跟from→ table.\* , rownum

