SQL语句的分类

|  |
| --- |
| (1)DDL: Data Define Language 定义数据  CREATE / DROP / ALERT(修改) / TRUNCATE(截断)  (2)DML: Data Manipulate Language 操作数据  INSERT / DELETE / UPDATE  (3)DQL: Data Query Language 查询数据  SELECT  (4)DCL: Data Control Language 控制用户权限  GRANT(授权) / REVOKE(收权) |

**详解DQL**

**简单查询**

1.简单查询 —— 只查询特定的列

示例：查询员工的姓名，工资，生日

SELECT ename,salary,birthday FROM emp;

示例：查询员工的编号，姓名，性别，部门编号

SELECT eid,ename,sex,deptId FROM emp;

2.简单查询 —— 查询所有的列

SELECT \* FROM emp;

3.简单查询 —— 给列取别名

示例：查询员工的姓名，生日，收入，取别名

SELECT ename AS 姓名,birthday AS 生日,salary AS 收入 FROM emp;

SELECT ename 姓名,birthday 生日,salary 收入 FROM emp;

示例:查询员工的编号，姓名，性别，起英文别名

SELECT eid a,ename b,sex c FROM emp;

4.简单查询 —— 只显示不同的记录

示例：查询哪些部门编号下有员工

SELECT DISTINCT deptId FROM emp;

示例：查询公司有哪些性别的员工

SELECT DISTINCT sex FROM emp;

5.简单查询 —— 在查询时执行计算

示例：计算2+3\*5/10

SELECT 2+3\*5/10 结果;

示例：查询出所有员工的姓名和年薪

SELECT ename 姓名,salary\*12 年薪 FROM emp;

示例：假设每个员工的工资+500，年终每个人再给5000的年终奖，查询每个员工一年的收入

SELECT ename 姓名,(salary+500)\*12+5000 AS 年收入 FROM emp;

6.简单查询 —— 查询结果集的排序

示例：查询所有的员工，结果按照工资由低到高

SELECT \* FROM emp ORDER BY salary ASC; #**asc**endant升序

示例：查询所有的员工，结果按照工资由高到低

SELECT \* FROM emp ORDER BY salary DESC;#**desc**endant降序

示例：查询所有的员工，结果按照年龄从大到小排序

SELECT \* FROM emp ORDER BY birthday ASC;

示例：查询所有的员工，结果集按照姓名排序

SELECT \* FROM emp ORDER BY ename ASC;

示例：查询所有的员工，按照生日由小到大排序，如果生日相同，再按照姓名由小到大排序

SELECT \* FROM emp ORDER BY birthday ASC,ename ASC;

示例：查询所有的员工，按照工资由小到大，如果工资相同，女员工显示在男员工的前边。

SELECT \* FROM emp ORDER BY salary ASC,sex ASC;

说明：排序默认是按照升序排列。

7.简单查询 —— 条件查询

示例：查询出编号为5的员工所有信息

SELECT \* FROM emp WHERE eid=5;

练习：查询出姓名为KING的员工的编号、工资、生日

SELECT eid,salary,birthday FROM emp WHERE ename='king';

练习：查询出20号部门下所有员工的信息

SELECT \* FROM emp WHERE deptId=20;

练习：查询出所有的男员工信息

SELECT \* FROM emp WHERE sex=1;

示例：查询出工资大于等于5000的员工所有信息

SELECT \* FROM emp WHERE salary>=5000;

练习：查询出1991-1-1之后出生的员工所有信息

SELECT \* FROM emp WHERE birthday>'1991-1-1';

|  |
| --- |
| > >= < <= = != |

练习：查询出不在10号部门的所有员工信息

SELECT \* FROM emp WHERE deptId!=10;

练习：查询出暂时没有部门的员工信息

SELECT \* FROM emp WHERE deptId IS NULL;

|  |
| --- |
| 注意：NULL和任何值都不能进行相等/不等判定，可以使用IS NULL 和 IS NOT NULL |

练习：查询出有明确部门的员工信息

SELECT \*FROM emp WHERE deptId IS NOT NULL;

练习：查询出工资大于6000的女员工所有信息

SELECT \* FROM emp WHERE salary>6000 AND sex=0;

练习：查询出工资在5000~5999之间的员工信息

SELECT \* FROM emp WHERE salary>=5000 AND salary<=5999;

SELECT \* FROM emp WHERE salary BETWEEN 5000 AND 5999;

|  |
| --- |
| between 5000 and 5999; >=5000 and <=5999 |

练习：查询出工资小于4000和大于8000的员工信息

SELECT \* FROM emp WHERE salary<4000 OR salary>8000;

|  |
| --- |
| 多条件拼接：  AND 并且 两个都要满足  OR 或者 两个满足其一 |

练习：查询出在1991年出生的员工信息

SELECT \* FROM emp WHERE birthday>='1991-1-1' AND birthday<='1991-12-31';

SELECT \* FROM emp WHERE birthday BETWEEN '1991-1-1' AND '1991-12-31';

练习：查询出在1990年之前和1993之后出生的员工信息

SELECT \* FROM emp WHERE birthday<'1990-1-1' OR birthday>'1993-12-31';

练习：查询出在10和30号部门的员工信息

SELECT \* FROM emp WHERE deptId=10 OR deptId=30;

SELECT \* FROM emp WHERE deptId IN(10,30);

练习：查询出在10、30、50、80号部门的员工信息

SELECT \* FROM emp WHERE deptId=10 OR deptId=30 OR deptId=50 OR deptId=80;

SELECT \* FROM emp WHERE deptId IN(10,30,50,80);

练习：查询出不在10和30号部门的员工信息

SELECT \* FROM emp WHERE deptId NOT IN(10,30);

8.简单查询 —— 模糊条件查询

示例：查询出姓名中包含字母E的员工所有信息

SELECT \* FROM emp WHERE ename LIKE '%E%';

练习：查询出姓名以E结尾的员工所有信息

SELECT \* FROM emp WHERE ename LIKE '%E';

练习：查询出姓名中倒数第二个字符是E的员工所有信息

SELECT \* FROM emp WHERE ename LIKE '%E\_';

|  |
| --- |
| SQL中提供了两个模糊查询的匹配符  % 可以匹配任意多个字符 >= 0  \_ 可以匹配任意一个字符 =1  上述两个符号不能和=连用，只能使用LIKE连用。 |

9.简单查询 —— 分页查询

查询出所有的男员工，对查询的结果集，按照工资升序排序。

SELECT \* FROM emp WHERE sex=1 ORDER BY salary ASC;

分页显示：查询的结果集记录太多，一次显示不完，可以一页一页的显示。分页查询语句在不同的数据库中实现方法不一样，MySQL是最简单的。

SELECT \* FROM emp LIMIT start, count;

start: 从结果集中开始读取的值

count: 指定读取的最多行数

第1页： SELECT \* FROM emp LIMIT 0,5; #(页码-1)\*每页数量

第2页： SELECT \* FROM emp LIMIT 5,5;

第3页： SELECT \* FROM emp LIMIT 10,5;

...

第n页： SELECT \* FROM emp LIMIT (n-1)\*5,5

start = (n-1)\*count

**复杂查询**

1. SQL语句中复杂查询分为 —— 聚合查询/分组查询

**聚合查询**

示例：查询出所有员工的数量

SELECT COUNT(\*) FROM emp; #推荐写法

SELECT COUNT(ename) FROM emp;

SELECT COUNT(eid) FROM emp;

SELECT COUNT(deptId) FROM emp;

|  |
| --- |
| 函数： FUNCTION，具有特定的功能，需要接收若干个数据，最终到处特定的计算结果。—— 饺子机 |

MySQL中提供了5个聚合函数

COUNT(...)/SUM(...)/AVG(..)/MAX(...)/MIN(...)

练习：查询出10号部门员工的数量

SELECT COUNT(ename) FROM emp WHERE deptId=10;

练习：查询出10号部门所有工资的总和

SELECT SUM(salary) FROM emp WHERE deptId=10;

练习：查询出所有男员工的平均工资

SELECT SUM(salary)/COUNT(\*) FROM emp WHERE sex=1;

SELECT AVG(salary) FROM emp WHERE sex=1;

练习：查询出1990年出生的员工工资最大值和最小值

SELECT MAX(salary),MIN(salary) FROM emp WHERE birthday>='1990-1-1' AND birthday<='1990-12-31';

**分组查询**

练习：查询出每个部门的员工数量

SELECT deptId,COUNT(\*) FROM emp GROUP BY deptId;

|  |
| --- |
| 注意：分组查询中SELECT后面只能跟分组条件，或者其它列的聚合函数； |

练习：查询出每个部门的平均工资，最大工资，最小工资

SELECT deptId,AVG(salary),MAX(salary),MIN(salary) FROM emp GROUP BY deptId;

练习：查询出男员工和女员工的人数是多少

SELECT sex,COUNT(\*) FROM emp GROUP BY sex;

**子查询**

示例：查询出工资比tom高的员工的所有信息

步骤1：先查询出tom的工资是多少 —— 6000

SELECT salary FROM emp WHERE ename='tom';

步骤2：再查询出工资比6000高的员工信息

SELECT \* FROM emp WHERE salary>6000;

综合：

SELECT \* FROM emp WHERE salary>(SELECT salary FROM emp WHERE ename='tom');

练习：查询出“研发部”所有的员工信息

步骤1：查询出“研发部”的部门编号 ——10

SELECT did FROM dept WHERE dname='研发部';

步骤2：查询出10号部门所有的员工信息

SELECT \* FROM emp WHERE deptId=10;

总和：

SELECT \* FROM emp WHERE deptId=(SELECT did FROM dept WHERE dname='研发部');

**多表查询/跨表查询**

示例：查询出所有的员工姓名及其所在部门的名称

SELECT ename,dname FROM emp,dept WHERE deptId=did ;

笛卡尔积

|  |
| --- |
| 注意：为了防止出现笛卡尔积，在多表查询的时候，需要添加查询条件 |

问题：无法查询出没有部门的员工，无法查询出没有员工的部门。

**SQL99中的多表查询**

**(1)内连接 INNER JOIN ... ON** —— 和之前的作用一样

SELECT ename,dname FROM emp **INNER JOIN** dept **ON** deptId=did;

**(2)左外连接 LEFT OUTER JOIN** ... **ON** —— 显示左侧所有的记录

SELECT ename,dname FROM emp **LEFT OUTER JOIN** dept ON deptId=did;

**(3)右外连接 RIGHT OUTER JOIN...ON** 显示右侧所有的记录

SELECT ename,dname FROM emp **RIGHT OUTER JOIN** dept **ON** deptId=did;

说明：outer关键字可以省略的。

**(4)全连接 FULL JOIN**

MySQL不支持全支持

UNION / UNION ALL

**UNION / UNION ALL**

union all 两组数据出现了相同项**不合并。**

union 两组数据出现了相同项进行**合并**。

(SELECT ename,dname FROM emp LEFT JOIN dept ON deptId=did)

UNION ALL

(SELECT ename,dname FROM emp RIGHT JOIN dept ON deptId=did);

(SELECT ename,dname FROM emp LEFT JOIN dept ON deptId=did)

UNION

(SELECT ename,dname FROM emp RIGHT JOIN dept ON deptId=did);