

# 一、数据库三范式和练习准备

---

## 1、三范式

---

- 第一范式（1NF）是对关系模式的基本要求，不满足第一范式（1NF）的数据库就不是关系数据库，是指数据库表的每一列都是不可分割的基本数据项，同一列中不能有多值（一个列中只能存储一个值）；
- 第二范式（2NF）要求数据库表中的每个实例或行必须可以被惟一地区分（主键）；
- 第三范式（3NF）要求一个数据库表中不包含已在其它表中已包含的非主键段信息。

## 2、SQL 分类

---

### 2.1、数据查询语言（DQL）

其语句，也称为“数据检索语句”，用以从表中获得数据，确定数据怎样在应用程序给出。保留字 SELECT 是 DQL（也是所有 SQL）用得最多的动词，其他 DQL 常用的保留字有 WHERE，ORDER BY，GROUP BY 和 HAVING。这些 DQL 保留字常与其他类型的 SQL 语句一起使用。

### 2.2、数据定义语言（DDL）

其语句包括动词 CREATE 和 DROP。在数据库中创建新表或删除表（CREATE TABLE 或 DROP TABLE）；为表加入索引等。DDL 包括许多与人数据库目录中获得数据有关的保留字。它也是动作查询的一部分。

### 2.3、数据操作语言（DML）

其语句包括动词 INSERT，UPDATE 和 DELETE。它们分别用于添加，修改和删除表中的行，Insert / Update / Delete。也称为动作查询语言。

### 2.4、事务处理语言（TCL）

它的语句能确保被 DML 语句影响的表的所有行及时得以更新。TCL 语句包括 BEGIN TRANSACTION，COMMIT 和 ROLLBACK。

### 2.5、数据库控制语言（DCL）

它的语句通过 GRANT 或 REVOKE 获得许可，确定单个用户和用户组对数据库对象的访问。某些 RDBMS 可用 GRANT 或 REVOKE 控制对表单个列的访问。

### 2.6、指针控制语言（CCL）


它的语句，像 DECLARE CURSOR，FETCH INTO 和 UPDATE WHERE CURRENT 用于对一个或多个表单独行的操作。

## 3、练习的表结构

---

通过 navicat 导入 mysqlplus.sql 文件，查看表结构。


部门表: dept

名		类型	长度	小数点	不是 null	
DEPTNO	主键	bigint	2	0	<input checked="" type="checkbox"/>	 1
DNAME	名称	varchar	14	0	<input type="checkbox"/>	
LOC	位置	varchar	13	0	<input type="checkbox"/>	

员工表: emp

名		类型	长度	小数点	不是 null	
EMPNO	主键	bigint	4	0	<input checked="" type="checkbox"/>	 1
ENAME	名称	varchar	10	0	<input type="checkbox"/>	
JOB	岗位	varchar	9	0	<input type="checkbox"/>	
MGR	经理	bigint	4	0	<input type="checkbox"/>	
HIREDATE	入职时间	date	0	0	<input type="checkbox"/>	
SAL	薪水	double	7	2	<input type="checkbox"/>	
COMM	奖金	double	7	2	<input type="checkbox"/>	
DEPTNO	部门	bigint	2	0	<input type="checkbox"/>	

薪水等级表

名		类型	长度	小数点	不是 null	
GRADE	等级	bigint	11	0	<input checked="" type="checkbox"/>	 1
LOSAL	最低薪水	int	11	0	<input type="checkbox"/>	
HISAL	最高薪水	int	11	0	<input type="checkbox"/>	

## 二、单表查询

```
SELECT [DISTINCT] * | 字段 [别名] [, 字段 [别名]]
FROM 表名称 [别名]
[WHERE 条件(S)]
[ORDER BY 字段 [ASC|DESC] [, 字段 [ASC|DESC], ...]];
```

### 1、全列和投影查询

一个是查询表中所有列，另一个是查询表中部分的列。

练习：

- 查询所有员工信息
- 查询每个员工的编号、姓名、职位
- 查询所有部门信息

### 2、消除重复

DISTINCT 关键字可以用于一列，也可以用于多列。比如：SELECT distinct job,deptno FROM emp; 只有当 job 和 deptno 相同，才认为是重复的数据。

练习：

- 查询所有有员工的部门编号
- 查询有员工的部门和职位

### 3、算术运算符

- 对 NUMBER 型数据可以使用算数操作符创建表达式 (+ - \* /) ;

- 对 DATE 型数据可以使用算数操作符创建表达式 (+ -)。

练习：

- 查询所有员工的年薪
- 查询所有员工的年薪（使用别名）

## 4、过滤查询

### 4.1、过滤查询值的注意事项

- 字符串和日期要用单引号扩起来；
- 数字类型直接书写；
- 字符串是大小写不敏感的，日期值是格式大小写敏感的；
- 字符串若要大小写敏感，需要添加 binary 关键字。（SELECT \* FROM dept WHERE BINARY DNAME = 'sales'）

### 4.2、空值

用 NULL 表示，注意：

- 空值是指不可用、未分配的值，也就是没有值；
- 空值不等于零或空格，也不表示空字符串；
- 任意类型都可以支持空值，也就是说任何类型的字段都可以允许空值作为值的存在；
- 包括空值的任何算术表达式都等于空。
- 使用函数 IFNULL(expr1, expr2)，若 expr1 不是 NULL，IFNULL() 返回 expr1，否则它返回 expr2

### 4.3、常用算术比较运算符

- =, !=, <>, >, >=, <, <=
- BETWEEN ... AND ...：在两值之间（包含开始和结尾，数学中的闭区间）
- IN：匹配列出的值；如：IN(1, 2, 3, 4)
- LIKE：匹配字符串模式，\_、%，LIKE 运算符必须使用通配符才有意义；
  - 匹配单个字符：\_ 表示 1 个；
    - 匹配任意多个字符：% 表示 0 个、1 个、多个。
- IS NULL：是否为空

练习：

- 查询所有员工的年薪((月薪 + 奖金) \* 12)
- 查询有奖金的员工信息
- 查询公司的老板
- 查询出基本工资高于 1500 的所有员工信息
- 查询名字叫 SCOTT 的员工所从事的工作
- 查询 1981 年入职的员工信息
- 查询年薪小于 3W 的员工
- 查询所有不是销售人员的员工信息
- 查询工资在 2000-3000 之间的员工信息
- 查询 1981 年入职的员工
- 查询工资为 800 或 1600 或 3000 的员工
- 查询出所有雇员姓名是以 A 开头的全部雇员信息。
- 查询出雇员姓名第二个字母是 M 的全部雇员信息。
- 查询出雇员姓名任意位置上包含字母 A 的全部雇员信息。

## 4.4、逻辑运算符

- AND: 如果组合的条件都是 true, 返回 true;
- OR: 如果组合的条件之一是 true, 返回 true;
- NOT: 如果下面的条件是 false, 返回 true。
- 优先级规则: 比较运算符 > NOT > AND > OR。

练习:

- 查询姓名中有 e 或者 a 的员工姓名
- 查询工资在 1500~3000 之间的全部员工信息
- 查询工资不在 2000-3000 之间的员工信息
- 查询工资不为 800 或 1600 或 3000 的员工
- 查询出职位是办事员 (CLERK) 或者是销售人员 (SALESMAN) 的全部信息, 且工资在 1000 以上

## 5、结果排序

- ASC: 升序, 缺省
- DESC: 降序
- ORDER BY 子句出现在 SELECT 语句后执行;
- ORDER BY 可以使用别名, **但不能使用加了引号的别名或列名来排序, 没有效果。**

练习:

- 查询所有员工信息, 按照工资排序
- 查询所有员工信息, 按照年薪降序排序
- 查询所有员工信息, 按照部门和年薪降序排序

# 三、多表查询概述

## 1、为什么使用多表查询

需要查询的数据分散在多张表中, 只有联合多张表才能查询出期望的数据。

比如: emp 表存储了员工的信息, dept 表存储了部门的信息, 而需求是查询出员工的编号、姓名以及对应部门的名称。

## 2、笛卡尔积

- 数学中: 假设集合  $A = \{a, b\}$ , 集合  $B = \{0, 1, 2\}$ , 则两个集合的笛卡尔积为  $\{(a, 0), (a, 1), (a, 2), (b, 0), (b, 1), (b, 2)\}$ 。
- MySQL 中: 多表查询会产生笛卡尔积, 比如: `SELECT * FROM emp, dept`, 实际运行环境下, 应避免使用全笛卡尔集。

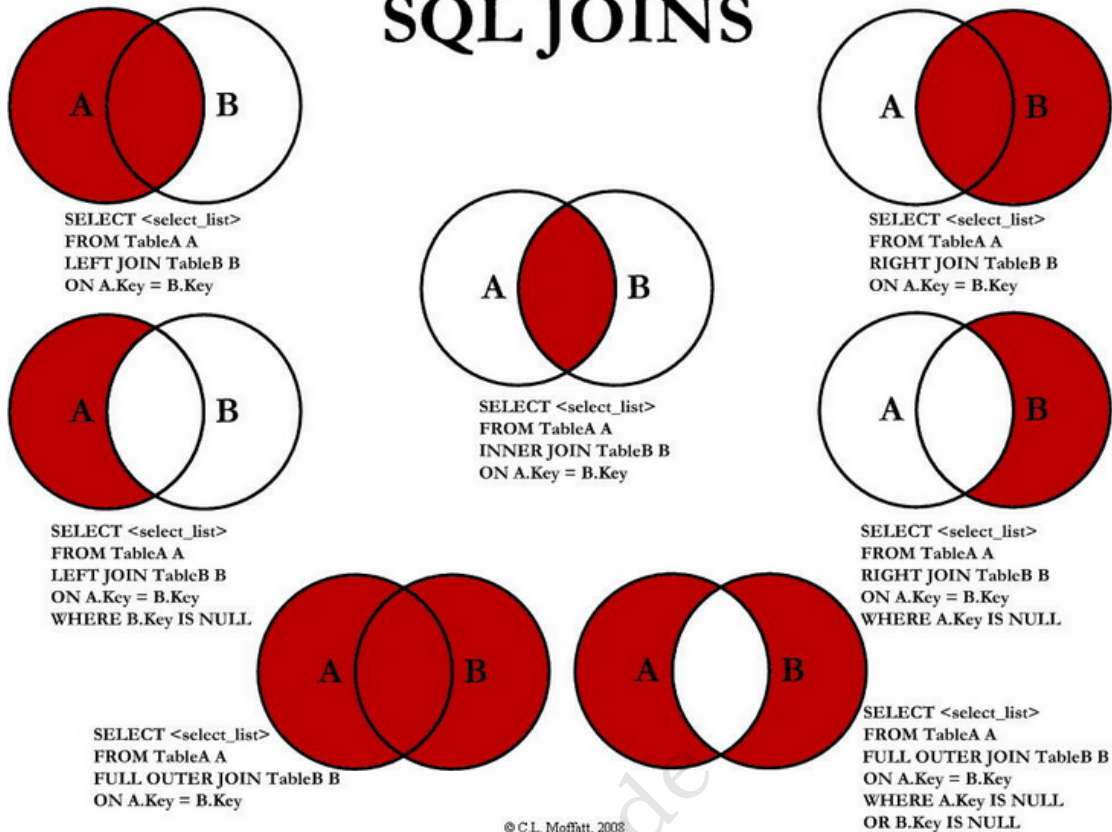
解决方案: 在 WHERE 语句中加入有效的连接条件 --> 一般是等值连接, 注意: 连接 n 张表, 至少需要 n-1 个连接条件。

## 3、多表查询分类

- 内连接查询
  - 隐式内连接查询
  - 显示内连接查询
- 外连接查询
  - 左外连接查询

- 右外连接查询
- 全外连接查询

# SQL JOINS



## 四、内连接查询

### 1、隐式内连接查询

```
SELECT [DISTINCT] * | 字段 [别名] [, 字段 [别名], ...]
FROM 表名称 [别名], [表名称 [别名], ...]
[WHERE 条件(S)/消除笛卡尔积连接]
[ORDER BY 排序字段 [ASC|DESC] [, 排序字段 [ASC|DESC], ...]];
```

- 在 WHERE 子句中写入连接条件;
- 当多个表中有重名列时, 必须在列的名字前加上表名作为前缀或者表的别名 (使用别名更简单, 性能更高);
- 等值连接是连接操作中最常见的一种, 通常是在存在主外键约束条件的多表上建立的, 连接条件中的两个字段通过等号建立等值关系。

练习:

- 查询员工编号, 员工名称, 员工所属部门的编号和名称
- 查询员工的姓名, 工资, 所在部门的名称, 以及工资的等级

### 2、显式内连接查询

```
SELECT table1.column, table2.column
FROM table1 [INNER] JOIN table2 ON table1.column1 = table2.column2
WHERE 条件
```

显示内连接查询：查询的结果和隐式内连接一模一样。区别在于：

- 显示内连接可以看到 [INNER] JOIN；
- 消除笛卡尔积条件使用写在 ON 子句。

## 五、外连接查询

### 1、需求

查询出员工的编号，名字，薪水和所在部门的名称（使用内连接查询），SQL 如下：

```
SELECT emp.EMPNO, emp.ENAME, emp.SAL, dept.DNAME FROM emp JOIN dept ON  
emp.DEPTNO = dept.DEPTNO
```

执行完会发现，没有部门的员工则查询不出来。

### 2、左外连接查询

```
SELECT table1.column, table2.column  
FROM table1 LEFT [OUTER] JOIN table2 ON table1.column1 = table2.column2  
WHERE 条件
```

```
SELECT emp.EMPNO, emp.ENAME, emp.SAL, dept.DNAME  
FROM emp LEFT JOIN dept ON emp.DEPTNO = dept.DEPTNO
```

查询出 JOIN 左边表的全部数据查询出来，JOIN 右边的表不匹配的数据使用 NULL 来填充数据。

### 3、右外连接查询

```
SELECT table1.column, table2.column  
FROM table1 RIGHT [OUTER] JOIN table2 ON table1.column1 = table2.column2  
WHERE 条件
```

```
SELECT emp.EMPNO, emp.ENAME, emp.SAL, dept.DNAME FROM dept RIGHT JOIN emp ON  
emp.DEPTNO = dept.DEPTNO
```

查询出 JOIN 右边表的全部数据查询出来，JOIN 左边的表不匹配的数据使用 NULL 来填充数据。

## 六、分组函数

### 1、函数分类

- 单行函数：将每条数据进行独立的计算，然后每条数据得到一条结果；
- 多行函数：多条数据同时计算，最终得到一条结果数据。也成为聚集函数、分组函数，主要用于完成一些统计功能等等。

### 2、多行函数

- COUNT(): 查询表中的数据记录；

- AVG(): 求出平均值;
- SUM(): 求和;
- MAX(): 求出最大值;
- MIN(): 求出最小值。

注意:

- 统计函数忽略空值, 可以使用 IFNULL, 因为是 NULL 不会影响汇总值, 但会影响汇总数量;
- 不能在 where 语句中使用分组函数。

练习:

- 查询所有员工每个月的平均工资及总工资
- 查询月薪在 2000 以上的员工总人数
- 查询员工最高工资和最低工资差距

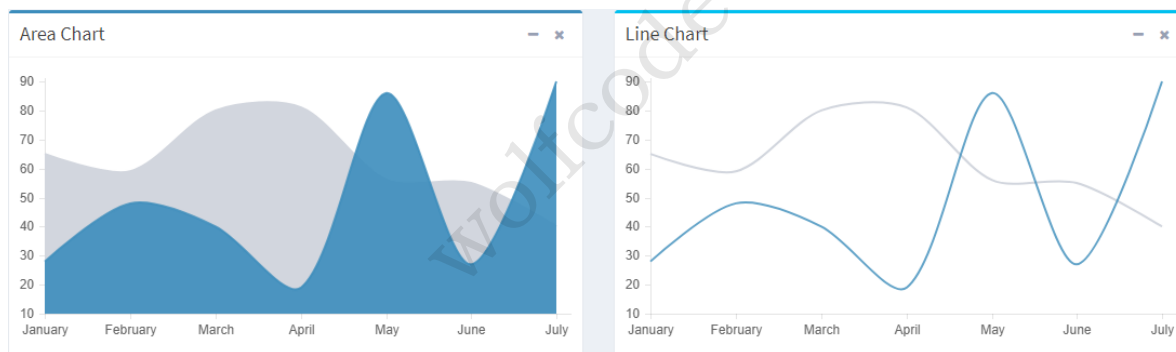
## 七、分组查询

### 1、什么是分组及其应用

分组情况1: 按照性别分组, 男生一组, 女生一组, 之后可以统计男生和女生的数量;

分组情况2: 按照年龄段分组, 80 后一组, 90 后一组;

分组情况3: 按照籍贯分组, 广东一组, 湖南一组, 江西一组。



### 2、分组语法

在 MySQL 使用 GROUP BY 来实现分组。语法如下:

```
SELECT [DISTINCT] * | 分组字段1 [别名] [, 分组字段2 [别名] ,...] | 统计函数
FROM 表名称 [别名], [表名称 [别名] , ...]
[WHERE 条件(s)]
[GROUP BY 分组字段1 [, 分组字段2 ,...]]
[ORDER BY 排序字段 ASC | DESC [, 排序字段 ASC | DESC]];
```

分组效果:

- 使用 GROUP BY 子句将表分成小组;
- 结果集隐式按升序排列, 如果需要改变排序方式可以使用 ORDER BY 子句。

练习:

- 按照职位分组, 求出每个职位的最高和最低工资
- 查询出每一个部门员工的平均奖金

### 3、使用分组注意

- SELECT 子句出现的字段，要不在统计函数中，要不出现在 GROUP BY 子句中，否则不合理（整体与个体）；
- 在 GROUP BY 子句中出现的字段，可以不出现在 SELECT 列表中；
- 统计函数可以单独使用，SQL 中可以没有 GROUP BY 子句；
- 在 GROUP BY 子句中，可以按单列进行分组，也可以在多列上进行分组，多列分组就是按照多个字段的组合进行分组，最终的结果也会按照分组字段进行排序显示。

练习：

- 查询出每一个部门员工的平均工资
- 查询各个部门和岗位的平均工资

## 4、分组限定

- 不能在 WHERE 子句中对分组限定，限制组须使用 HAVING 子句；
- 不能在 WHERE 子句中使用统计函数，而在 HAVING 子句可使用统计函数。

练习：

- 查询部门平均工资高于 2000 的部门及其平均工资
- 查询在 80, 81, 82 年各进公司多少人

```
SELECT YEAR(HIREDATE) y, COUNT(EMPNO)
FROM emp
GROUP BY YEAR(HIREDATE)
HAVING y BETWEEN '1980' AND '1982'
```

- 查询各个管理人员下员工的平均工资，其中最低工资不能低于 1300，不计算老板

```
SELECT MGR, AVG(SAL), MIN(SAL) AS minsal
FROM emp
GROUP BY MGR
HAVING MGR IS NOT NULL AND minsal >= 1300
```

## 八、单行函数

### 1、日期函数

- NOW(): 获取当前时间；
- DAY(date): 获取日期中的天数，范围是从 1 到 31；
- HOUR(time): 返回 time 对应的小时数，范围是从 0 到 23；
- MINUTE(time): 返回 time 对应的分钟数，范围是从 0 到 59；
- MONTH(date): 返回 date 对应的月份，范围是从 1 到 12；
- YEAR(date): 返回 date 对应的年份，范围是从 1000 到 9999；
- LAST\_DAY(date): 获取一个日期或日期时间值，返回该月最后一天对应的值。

### 2、日期转换为字符串

DATE\_FORMAT(date, format): 把日期转换为字符串。其中 format 中常见格式化符号如下（更多参见 MySQL 5.5 中文参考手册 549 页）：



%Y	年份, 数字形式,4位数
%m	月份, 数字形式 (00..12)
%d	该月日期, 数字形式 (00..31)
%H	小时(00..23)
%i	分钟,数字形式 (00..59)
%s	秒 (00..59)

```
SELECT DATE_FORMAT(NOW(), '%Y-%m-%d')
```

## 九、子查询

### 1、定义和作用

子查询指的就是在一个查询之中嵌套了其他的若干查询。

在使用 SELECT 语句查询数据时, 有时候会遇到这样的情况, 在 WHERE 查询条件中的限制条件不是一个确定的值, 而是一个来自于另一个查询的结果 (比如查询大于平均工资的员工) 。

```
SELECT select_list
FROM table
WHERE expr operator (SELECT select_list FROM table)
```

注意:

- 子查询一般出现在 FROM 和 WHERE 子句中;
- 子查询要使用圆括号括起来;
- 将子查询放在比较运算符的右边 (增强可读性);
- 子查询在主查询前执行一次, 主查询使用子查询的结果; 但不宜嵌套过多。

### 2、分类

按查询结果分类:

- 单行单列: 只包含一个字段的查询, 返回的查询结果也只包含一行数据;
- 多行单列: 只包含了一个字段, 但返回的查询结果可能多行或者零行;
- 多行多列: 包含多个字段的返回, 查询结果可能是单行或者多行, 好比是一张表。

### 3、单行单列

一般用于 WHERE 之后的子查询, 子查询结果是一行一列记录。

使用单行记录比较运算符: =、>、>=、<、<=、<>。

练习:

- 查询出工资比 MARTIN 还要高的全部雇员信息

```
SELECT * FROM emp WHERE sal > (SELECT sal FROM emp WHERE ename = 'MARTIN')
```

- 查询平均工资高于公司平均工资的部门信息

```
SELECT e.deptno, d.dname, AVG(sal)
FROM emp e JOIN dept d ON e.deptno = d.deptno
GROUP BY deptno HAVING AVG(sal) >= (SELECT AVG(sal) FROM emp)
```

## 4、多行单列

一般也用于 WHERE 子句中，子查询结果只有一列，但是有多行。使用多行比较运算符：

- **IN**：与列表中的任意一个值相等
- **ANY**：与子查询返回的任意一个值比较
- **=ANY**：此时和 IN 操作符相同
- **>ANY**：大于子查询中最小的数据
- **<ANY**：小于子查询中最大的数据
- **ALL**：与子查询返回的每一个值比较
- **>ALL**：大于子查询中最大的数据
- **<ALL**：小于子查询中最小的数据

练习：

- 查询工资等于部门经理（职位是 MANAGER）的员工信息。

## 5、多行多列

子查询的结果是多行多列，一般会把子查询返回的结果当成一个临时表，接着在临时表上继续查询或者连接查询；

注意：多行多列的子查询返回的结果必须要设置一个临时表名。

练习：

- 查询出每个部门的编号、名称、部门人数、平均工资

可以先把每一个部门的编号，总人数，平均工资先查询出来：

```
SELECT deptno dno, COUNT(empno) count, AVG(sal) avg FROM emp GROUP BY dno
```

再和 dept 表联合查询部门名称：

```
SELECT dept.deptno, temp.count, temp.avg
FROM dept JOIN (SELECT deptno dno, COUNT(empno) count, AVG(sal) avg FROM emp
GROUP BY deptno) temp ON dept.deptno = temp.dno
```