MySQL 数据库学习笔记

郑华

2018 年 2 月 13 日

第一章　基础概念

1.1　术语

- **数据库**: 数据库是一些关联表的集合。.
- **数据表**: 表是数据的矩阵。在一个数据库中的表看起来像一个简单的电子表格。
- **列**: 一列 (数据元素) 包含了相同的数据, 例如邮政编码的数据。
- **行：**一行（= 元组，或记录）是一组相关的数据，例如一条用户订阅的数据。
- **冗余：**存储两倍数据，冗余降低了性能，但提高了数据的安全性。
- **主键：**主键是唯一的。一个数据表中只能包含一个主键。你可以使用主键来查询数据。
- **外键：**外键用于关联两个表。
- **复合键：**复合键（组合键）将多个列作为一个索引键，一般用于复合索引。
- **索引：**使用索引可快速访问数据库表中的特定信息。索引是对数据库表中一列或多列的值进行排序的一种结构。类似于书籍的目录。
- **参照完整性**: 参照的完整性要求关系中不允许引用不存在的实体。与实体完整性是关系模型必须满足的完整性约束条件，目的是保证数据的一致性。

第二章　数据库基本操作

2.1　SQL 分类

- 数据库查询：代表关键字 `select`
- 数据库操纵：代表关键字 `insert delete update`
- 数据库定义：代表关键字 `create drop　alter`
- 事务控制：代表关键字 `commit rollback`
- 数据控制：代表关键字 `grant, revoke`

2.2　常用命令

显示当前的数据库们　`show databases;`

使用某个数据库　`use databaseName;`

显示数据库中的表们　`show tables;`

查看表的创建语句　`show create table tableName;`

查看表的结构　`desc tableName;`

select version();　显示当前数据库管理系统的版本。

重名命结果　`as` , 如 `select lower(ename) as E from emp;`

创建数据库　`create database Name;`

终止一条语句　`\c`

退出管理系统　`exit Or quit Or ctrl+c`

2.3 查询语句

2.3.1 条件查询

表 2.1: 查询符号

运算符	功能说明
=	等于
!=	不等于
between ... and ..	等同于 >= ... and <= ...
is null	为null(is not null 不为空)
and	并且
or	或者
in	包含，相当于多个or,(not in 不在这个范围内)
not	取非
like	为模糊查询，支持% 或_ 匹配，其中% 匹配任意个字符，_ 只匹配一个字符

example ->

```
// 执行顺序
    select // 3
       xx, xx2, xx3
     from // 1
       XX
     where // 2
        xx = xx;

// in 示例 查找job是什么的，不是什么的
    select
       ename,job
    from
       emp
    where
       job in('MANAGER','SALESMAN');

    select
       ename,job
    from
       emp
    where
       job not in('MANAGER','SALESMAN');
```

```
// like 示例，查找以S 开头的名字
    select
        ename
    from
        emp
    where
        ename like 'S%'
```

2.3.2 排序

order by

```
// order 示例 默认升序，（desc 降序）
select
    ename,salary
from
    emp
order by
    salary

// 按照第几个字段排序
select
    ename,salary // 1,2 字段
from
    emp
order by
    2 // 第2个字段

// 多个字段排序，ename 升序，salary 降序，使用逗号分割
select
    ename,salary
from
    emp
order by
    salary desc, ename
```

2.3.3 数据处理函数（单行）

处理单行后结束

- lower：转换小写
- upper：转换大写
- substr：取子串（被截取的串，起始位置，截取长度）
- length：取长度

- trim：去空格
- round：四舍五入
- rand()：生成随机数
- ifnull(xx, num)：可以将null 值转换成一个具体值

2.3.4 分组函数、聚合函数（多行）

处理多行后结束，自动忽略空值
先分组，然后再执行分组函数，而 where 在分组函数之前执行，所以不能
where 中不能出现分组函数

- count ：取得记录数
- sum ：求和
- avg ：求平均
- max ：取最大值
- min ：取最小值

distinct 去重关键字 -> select distinct job from emp; 只能出现在所有**字段**的最前面
select count(distinct job) from emp;

2.3.5 分组查询

group by ：通过哪个或哪些字段进行分组，使用后 select 后只能跟参与分组的字段和分组函数。

example-> 找出每个工作岗位的最高薪水【先按照工作岗位分组，使用 max 函数求每一组的最高工资】

```
// 先按照job 分组，然后对每一组使用max(salary) 求最大值。
   select    //3
      max(salary)
   from   //2
      emp;
   group by  //1
      job;

// 结合where 限定分组前条件，即分组前过滤
   select
      job, max(sal)
   from
      emp
```

```
    where
        job != 'MANAGER'
    group by
        job;
```

example-> 找出每个工作岗位的平均薪水，要求显示平均薪水大于 1500 where 处理不了

having 与 where 都是为了完成数据的过滤，where 和 having 后面都是添加过滤条件，where 是在 group by 之前执行，而 having 是在 group by 后执行。

```
//上例子解法
select
    job,avg(sal)
from
    emp
group by
    job
having
    avg(sal) > 1500;
```

2.3.6 查询语句总结

关键字顺序不能变 :

```
    select
        ...
    from
        ...
    where
        ...
    group by
        ...
    having
        ...
    order by
        ...
```

执行顺序 :

1. from 从某张表中检索数据
2. where 经过某条件进行过滤
3. group by 然后分组
4. having 分组之后不满意再过滤
5. select 查询出来

6. order by 排序输出

2.4 连接查询

查询的时候只从一张表检索数据称为单表查询

在实际的开发中，数据并不是存储在一张表中的，是同时存储在多张表中，这些表和表之间存在关系，我们在检索的时候通常需要将多长表联合起来取得有效数据，这种多表查询被称为连接查询或者叫做跨表查询。

连接查询根据连接方式可以分为如下方式：

- 内连接
  - 等值连接
  - 非等值连接
  - 自连接
- 外连接
  - 左外连接
  - 右外连接
- 全连接【几乎不用】

2.4.1 内连接

查找两张表匹配的数据。

A 表和 B 表能够完全匹配的记录查询出来，被称为内连接。

别名的使用，内连接的等值连接 在进行多表连接查询的时候，尽量给表起别名，这样效率高，可读性高

```
// 将表emp 用别名 e表示..

// 查询员工名与其对应的部门名
    select
        e.ename, d.dname
    from
        emp e, dept d;
    where
        e.depno = d.depno

// SQL99 语法,使得表连接独立出来了，结构更清晰
    select
        e.ename, d.dname
    from
```

```
    emp e
join // 内连接的inner 可以省略
    dept d
on
    e.depno = d.depno;
```

内连接的非等值连接 范围

```
// 找出员工名， 薪水，与其的薪水等级
    select
        e.name, e.sal, s.grade
    from
        emp e
    join
        salgrade s
    on e.sal >= s.lower and e.sal <= s.higher; // 可以使用between and 替代
```

内连接的自连接 自己与自己连接, 将自己视为两张表

```
// 找出每一个员工的上级领导，要求显示员工名以及对应的领导名
    表结构:
    empno ename mgr
    7369 SMITH 7123

// 要点：将自己视为两张表
    select
        a.ename empname, b.ename leaderName
    from
        emp a
    join
        emp b
    on
        a.mgr = b.empno;
```

2.4.2 外连接

A 表和 B 表能够匹配的记录查询出来之外，将其中一张表的记录完全无条件的完全查询出来，对方表没有匹配的记录，会自动模拟出 NULL 与之匹配。

外连接查询的结构条数 >= 内连接的查询结果数量

可以添加除了内连接外的其他数据。

example -> 找出每一个员工对应的部门名称，并且显示所有部门名称，注意部门可能没有员工。

左外连接 `select e.ename, d.dname from dept d left join emp e on e.deptno = d.deptno;`

右外连接 `select e.ename, d.dname from emp e right join dept d on e.deptno = d.deptno;`// outer 省略

总结 希望将哪边表的数据完全显示出来，join 的前边的修饰词 `right left` 可以恰好说明，如上，希望将 dept 表完全显示，那么先写`dept` 的话，那么就在`join` 的左边，就是 `left join`.

第三章　高级操作

3.1　主从模式-replication

3.2　集群