MySQL关系型数据库

（二）数据操作、子句、聚合

作者：Daniel.Wang

主要内容

1. 修改、删除数据
2. 运算符
3. 查询子句
4. 聚合
5. 表结构调整

（一）修改、删除数据

1. 修改数据

1）语法格式

```
update 表名称
set    字段1 = 值1,
       字段2 = 值2,
       ……
where 条件表达式
```

2）示例

- 修改订单201801010001的状态为2

```
update orders set status = 2
where order_id = '201801010001';
```

- 修改订单201801010002的商品数量为2，订单总金额为400

```
update orders set products_num = 2,
                  amt = 400
where  order_id = '201801010002';
```

3）修改数据注意事项

- 修改的值要和字段类型匹配
- 字符串需要单引号引起来
- 若不使用where限定条件，则修改所有
- 限定条件时，只修改符合条件的；若没有则修改行数为0行

2. 删除数据

1）语法格式

```
delete from 表名称 where 条件表达式
```

2）示例：删除订单编号为201801010002的数据

```
delete from orders where order_id = '201801010002'
```

3）注意事项：

- 如果where后面不带条件，则删除所有数据，慎用！！！
- 真实项目中，删除之前一定要做好数据备份

（二）更多查询

1. 运算符

1）比较运算符：>, <, ==, <>, >=, <=

- 示例1：查询总金额大于200元的订单

```
select * from orders where amt > 200;
```

- 示例2：查询状态不等于2的订单

```
select * from orders where status <> 2;
```

2）逻辑运算符：and, or

- and：多个条件同时满足
- or： 满足多个条件中的一个
- 示例：
  - 示例1：查询总金额大于200且状态为1的订单

```
select * from orders where amt > 200 and status = 1;
```

  - 示例2：查询客户编号为C0002或C0003的订单

```
select * from orders where cust_id='C0002' or cust_id = 'C0003';
```

  - 示例3：and，or的组合使用,查询客户编号为C0003或C0002且状态为1的订单

```
select * from orders
where (cust_id='C0002' or cust_id = 'C0003)
and status = 1
```

3）范围比较

- between ... and ....: 在...与...之间
- in/not in: 在/不在某个集合内
- 示例：
  - 示例1：查询总金额在200~300之间的订单信息（包含200,300）

```
select * from orders where amt between 200 and 300;
```

  - 示例2：查询客户编号在C0002，C0003的订单信息

```
select * from orders where cust_id in('C0002', 'C0003');
```

  - 示例3：查询客户编号不在C0002，C0003的订单信息

```
select * from orders where cust_id not in('C0002', 'C0003');
```

4）模糊查询

- 格式：where 字段名称 like "通配字串"
- 通配符匹配规则

  下划线（_）：匹配单个字符

  百分号（%）：匹配任意个字符

模糊查询示例：

- 第一步：创建客户信息表

```
create table customer (
    cust_id varchar(32),
    cust_name varchar(32),
    tel_no varchar(32)
) default charset=utf8;
```

- 第二步：插入多笔数据

```
insert into customer values ('C0001', 'Jerry', '13511223344'),
('C0002', 'Dekie', '13844445555'), ('C0003', 'Dokas', '15822223333');
```

- 第三步：模糊查询，查询姓名以D开头的客户信息

```
select * from customer where cust_name like 'D%';
```

5）空、非空值判断

- 判断空： is null
- 判断非空：is not null
- 示例：

  示例1：从orders表中查询状态为空的订单信息

  ```
  select * from orders where status is null;
  ```

  示例2：从orders表中查询状态不为空的订单信息

  ```
  select * from orders where status is not null;
  ```

2. 查询子句

1）Order by子句

- 作用：对查询结果按照某个字段进行排序
- 格式： order by 排序字段 [ASC/DESC]

  ASC-升序，DESC-降序，不写默认为升序
- 示例：查询所有订单信息，按照订单金额降序排列

```
select * from orders order by amt desc;
```

2）limit子句

- 作用：限定显示查询结果的笔数
- 格式：

  limit n　　限定显示前n笔数据

  limit m,n　　从第m笔显示，总共显示n笔
- 示例：

  示例1：查询所有订单，显示前3笔

  ```
  select * from orders limit 3;
  ```

  示例2：查询所有订单，从第2笔开始显示，共显示3笔

  ```
  select * from orders limit 1, 3;
  ```

利用limit实现分页查询

- 分页：一批数据不全部显示，而是分页显示，每页显示固定笔数
- 分页方式：
  - 显示时分页：查询出所有满足条件的数据，分页显示
  - 查询时分页：只查询当前页要显示的数据
- 查询时分页原理：利用limit函数，带两个参数，控制从哪一笔数据开始显示、共显示多少笔。例如：每页3显示笔数据，利用limit查询分页关系如下：

页数	limit语句	说明
第1页	limit 0, 3	limit第一个参数为：(n-1)*m 第二个参数为m 其中，n为页数，m为每页显示笔数
第2页	limit 3, 3	
第3页	limit 6, 3	

利用limit实现分页查询（续）

- 计算方式：根据上一页表格中给出的规律，只需要知道当前第几页(n)、每页多少笔数据(m)，就能得出limit子句的两个参数，从而实现分页查询。
- 示例：当前第3页，每页显示3笔数据

  n = （3-1）*3 = 6

  m = 3

  查询语句为：

  select * from orders limit 6, 3;

- 思考：如何计算总体页数?

3）distinct子句

- 作用：对某个字段去除重复后显示
- 格式：select distinct(字段名称) from 表名称
- 示例：查询所有订单状态并显示

```
select distinct(status) from orders;
```

3. 聚合函数

1）什么是聚合

有时候不需要返回表中具体数据，而是对数据进行总结，将结果返回

2）聚合函数

函数名称	作　　用
MAX	返回某列最大值
MIN	返回某列最小值
AVG	返回某列平均值
SUM	返回某列值得和
COUNT	返回记录笔数

3）聚合函数示例

- 示例1：查询订单金额最大值

```
select max(amt) from orders;
```

- 示例2：查找订单金额平均值

```
select avg(amt) from orders;
```

- 示例3：查询所有订单金额总和

```
select sum(amt) from orders;
```

- 示例4：统计订单笔数

```
select count(*) from orders;
```

4. Group by 分组子句

1）作用：对查询结果进行分组，通常和聚合函数配合使用

2）格式：group by 分组字段名称

3）示例

- 示例1：统计各种状态订单数量

```
select status, count(*) from orders group by status;
```

- 示例2：统计各种状态订单的最大金额

```
select status, max(amt) from orders group by status;
```

5. Having分组筛选子句

1）作用：对分组统计结果进行筛选，需要和group by子句配合使用

2）格式：group by 分组字段名称 having 过滤条件

3）示例

按照订单状态统计总笔数，不显示状态为NULL的数据

```
select status, count(*) from orders
group by status
having status is not null;
```

注意：group by分组聚合结果只能使用having进行过滤，而不能使用where

where只能针对表中真实存在的字段限定条件

4）SQL语句执行步骤（难点）

- 第一步：首先执行from子句，从表中找到源数据
- 第二步：执行where子句，选出所有满足条件的数据
- 第三步：group by子句进行分组
- 第四步：聚合操作
- 第五步：having子句对聚合结果进行过滤
- 第六步：order by子句对结果进行排序
- 第七步：limit限制显示笔数

（三）表结构调整

1. 添加字段

1）语法

- 添加到最后一个字段

  alter table 表名 add 字段名 类型

- 添加到第一个字段

  alter table 表名 add 字段名 类型 first

- 添加到指定位置

  alter table 表名 add 字段名 类型 after 字段名

2）添加字段示例

- 首先创建student表

```
create table student (
        stu_no varchar(32),
        stu_name varchar(128)
);
```

- 添加字段

```
alter table student add age int; -- 添加最后一个字段
alter table student add id int first; -- 添加第一个字段
alter table student add tel_no varchar(32) after stu_name; -- 在stu_name后面添加tel_no字段
```

2. 修改字段

1）语法

- 修改字段类型

  alter table 表名 modify 字段名 类型(宽度)

- 修改字段名称

  alter table 表名 change 旧字段名 新字段名 类型(宽度)

2）示例

```
alter table student modify stu_name varchar(64);  -- 修改学生名称为varchar(64)
alter table student change age stu_age int; -- 修改age为stu_age
```

3. 删除字段

1）语法

alter table 表名 drop 字段名称

2）示例

alter table student drop id; -- 删除id字段

（四）总结与回顾

1. 修改、删除数据

1）修改

```
update orders set status = 2
where order_id = '201801010001'
```

2）删除

```
delete from orders
where order_id = '201801010001'
```

2. 运算符

1）比较运算符：>, <, >=, <=, <>

2）逻辑运算符：and, or

3）范围比较：in， not in，between...and...

4）模糊查询：

下划线(_): 匹配一个字符

百分号(%): 匹配任意个字符

5）空、非空判断：is null, is not null

3. 子句

1）Order by：排序

```
select * from orders
order by amt asc;    -- desc
```

2）limit：限定显示行数

```
select * from orders limit 3
select * from orders limit 1,3
```

4. 聚合函数

1）聚合：max, min, avg, sum, count

2）group by: 分组，通常和聚合函数配合使用

3）having子句：对分组结果进行过滤

distinct: 去重

5. 表结构调整

1）添加字段

alter table 表名 add 字段名 类型

alter table 表名 add 字段名 类型 first

alter table 表名 add 字段名 类型 after 字段名

2）修改字段

alter table 表名 modify 字段名 类型(宽度)

alter table 表名 change 旧字段名 新字段名 类型(宽度)

3）删除字段

alter table 表名 drop 字段名称