作者：jzh

时间：2022.2.27

基于《MySQL必知必会》

请关注增删查改的内容

1.MySQL的介绍

Structure Query Language(结构化查询语言)简称SQL，它被美国国家标准局(ANSI)确定为关系型数据库语言的美国标准，后被国际化标准组织(ISO)采纳为关系数据库语言的国际标准。数据库管理系统可以通过SQL管理数据库；定义和操作数据，维护数据的完整性和安全性。

MySQL 是最流行的关系型数据库管理系统，在 WEB 应用方面 MySQL 是最好的 RDBMS(Relational Database Management System：关系数据库管理系统)应用软件之一。

2.安装MySQL

2021MySql-8.0.26安装详细教程

具体请参照上面的博客内容

一些tips：

1. 请注意以管理员打开terminal去运行`net start mysql`，否则无法启动
2. 若发现端口3306被占用，请查看自己是否已经安装并开启了旧的MySQL
3. 若感觉MySQL下载速度太慢，可以使用迅雷下载器，或采用国内源（清华源，中科大源等）进行下载
4. 请记住务必务必密码，虽然可以在忘记密码后修改，但十分麻烦连接MySQL时出现：ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: YES)

当安装完成后，请进行测试，在terminal中输入`mysql -u 用户名 -p 密码`，若可以登录，即安装成功

若肯定安装成功，却发现无法使用，请进入管理员模式，重新输入`net start mysql`，并再次尝试

3.基础知识介绍

`数据库`：保存数据的容器，一般一个数据库有很多张表，可以当作一个Excel文件

`数据表`：保存数据的表格，可以当成Excel里的一个sheet

注意：一个数据库中不能有相同名字的表

`列`：表中的一个字段，也就是Excel中的一个列

`行`：一条数据，相当于Excel中的一行

`主键`：一列（或者多列），其中内容是不能为空，也不能重复

注意：尽量不要将主键设置为可能会更改的值，或者主动去更改主键的值

4.连接MySQL和基础使用

当完成了MySQL的安装后，便可开始连接数据库

1. 本地连接

在terminal中输入`mysql -u 用户名 -p 密码`

若看到

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.27 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

即连接成功

2. 远程连接

在terminal中输入`mysql -h 数据库公网ip地址 -p 端口号（默认3306） -u 用户名 -p 密码`

若看到和上面同样内容时，即代表连接成功。

注意：在现实中的开发中我们一般采用ODBC进行连接，而非采用电脑里的terminal进行连接，同时正常的工程中我们一般采用类似半自动化ORM或者全自动化ORM进行控制，类似mybatis

3. 基础使用

1. 展示数据库：`show databases;`
2. 选择数据库：`use 数据库名字;`
3. 展示数据表（请先选择应该数据库）：`show tables;`
4. 展现数据表里的内容（请先选择应该数据库）：`show columns from 数据表名;`
5. 展示服务器状态：`show status;`
6. 展示权限:`show grants;`
7. 展示错误：`show errors;`
8. 展示警告：`show warnings`

5.Navicat的安装与使用

Navicat介绍

“Navicat”是一套可创建多个连接的数据库管理工具，用以方便管理 MySQL、Oracle、PostgreSQL、SQLite、SQL Server、MariaDB 和/或 MongoDB 等不同类型的数据库，并支持管理某些云数据库，例如阿里云、腾讯云。Navicat 的功能足以符合专业开发人员的所有需求，但是对数据库服务器初学者来说又相当容易学习。Navicat 的用户界面 (GUI) 设计良好，让你以安全且简单的方法创建、组织、访问和共享信息。

Navicat安装和破解

MySQL和Navicat下载、安装及使用详细教程

下载文件请自取

Navicat的使用

点击左上角链接MySQL，然后按设置链接名：并输入密码

右键点击打开连接

即可在上面增删查改数据

Navicat使用快速入门教程

6.创建修改数据库与数据表

1. 基本数据类型

1. 整数型
    1. tinyint　　1个字节　　[0~255]或[-128~127]
    2. smallint　　2个字节　　[0~65535]或[-32768~32767]
    3. mediumint　　3个字节
    4. int(integer)　　4个字节
    5. bigint　　8个字节
2. 浮点数
    1. float　　4个字节
    2. double　　8个字节
    3. decimal　　M+2个字节(`DECIMAL[(M,D)]`)
3. 位类型
    1. bit
4. 字符串类型
    1. char
    2. varchar
    3. tinyblob
    4. tinytext
    5. blob
    6. text
    7. mediumblob
    8. mediumtext
    9. longblob
    10. longtext

blob一般用于二进制文件

text一般用于文本文件

char一般由于字符串

11. `ENUM`： 一个字符串对象，从值列表中选择，最多可包含 65,535 个不同的元素
12. `SET`： 字符串对象，可以有零个或多个值，最多可包含 64 个不同的成员

5. 时间类型
    1. DATE： 取值范围为1000-01-01~9999-12-31
    2. TIME： TIME 不是表示时分秒，而表示两个事件之间的时间间隔，所以 TIME 类型可以为负值
    3. DATETIME： 是 DATE 和 TIME 两个种数据类型的组合
    4. TIMESTAMP： 保存日期与时间的组合值，与时区相关，默认以 UTC(世界标准时间)格式存储
    5. YEAR： 写法为 YEAR 和 YEAR(4)，取值范围 1901~2155

2. 创建数据库

`create database 数据库名字;`

可以用`show databases;`查看

`create database 数据库名字 character set utf8;`

设置编码格式为utf8

3. 创建数据表

首先进入一个数据库，也就是`use 数据库名字;`

然后去创建数据表

`create table 表名(里面的数据类型);`

e.g.

```
create table Phone_table(pid INT, name CHAR(20), price INT);
```

可以用`show create table 表名;`来查看表的信息

4. 给表添加字段

`alter table 表名 add 数据类型;`

e.g.

```
alter table Phone_table add color CHAR(20);
```

5. 给表删除字段

`alter table 表名 drop 字段名;`

e.g.

```
alter table Phone_table drop price;
```

6. 修改表的字段类型

`alter table 表名 modify 字段名 数据类型;`

e.g.

`alter table Phone_table modify name VARCHAR(12);`

7. 修改字段的数据类型并且改名

`alter table 表名 change 原字段名 新字段名 数据类型;`

e.g.

`alter table Phone_table change name pname CHAR(18);`

~~不过实际上这些操作都可以在Navicat中点击实现~~

7.检索数据（查）

最基础的select语句

1. 搜索单个列

   `select 列名 from 表名;`

   e.g.

   `select prod_name from products;`

2. 搜索多个列

   `select 列名1, 列名2, 列名3 from 表名;`

   e.g.

   `select prod_name, prod_id, prod_price from products;`

3. 搜索所有列

   `select * from 表名;`

   e.g.

   `select * from products;`

4. 搜索不重复的行

   防止行重复导致阅读不方便

   `select distinct 列名 from 表名;`

   e.g.

   `select distinct prod_name from products;`

5. 限制结果

   `select prod_name from products limit 5;`

   最多出现5项

   `select prod_name from products limit 10,5;`

   从第10行开始出现5项

6. 限制表名（用的不多）

`select products.prod_name from crashcourse.products;`

排序检索出来的数据

当我们需要对检索出来的数据排序时，我们便需要使用以下代码

`select 列名 from 表名 order by 列名;`

e.g.

`select prod_name from products order by prod_name;`

按照多个列进行排序

`select 列名1,列名2,列名3 from 表名 order by 列名1,列名2;`

按照指定排序方向进行排序

降序排列：`select 列名 from 表名 order by 列名 desc;`

升序排列：`select 列名 from 表名 order by 列名 asc;`

8.过滤数据（查）

基础查询工作

当我们需要进行一些条件筛选时，我们会选择用where进行判断

`select 列名 from 表名 where 条件;`

e.g.

`select prod_name, prod_price from products where prod_price = 100;`

`select prod_name, prod_price from products where prod_name = 'jzh';`

判断条件

操作符	说明
=	等于
<>	不等于
!=	不等于
<	小于
>	大于
<=	小于等于
>=	大于等于
between	在两者之间

1. between使用方法

   `select prod_name, prod_price from products where prod_price between 100 and 200;`

2. 查找空值

```
select prod_name, prod_price from products where prod_price is null;
```

组合查询

1.组合子句

当我们需要组合子句时，我们可以使用`AND`,`OR`来实现

1. AND 同时满足

```
select prod_name, prod_price from products where prod_price = 100 and prod_name = 'jzh';
```

2. OR 只满足一个

```
select prod_name, prod_price from products where prod_price = 100 or prod_name = 'jzh';
```

3. 计算次序，and的优先级是高于or的，所以建议打括号

```
select prod_name, prod_price from products where prod_price = 100 and (prod_name = 'jzh' or prod_name = 'jzzh');
```

2. IN操作符

```
select prod_name, prod_price from products where vend_id in (1002,1003);
```

IN是用来筛选范围内的内容的

tips：能使用IN就用IN，因为IN的执行速度是比AND,OR快的

3. NOT操作符

```
select prod_name, prod_price from products where vend_id not in (1002,1003);
```

not是取反用的

通配符过滤

`通配符`：用来匹配一部分的特殊字符，例如%，_

`like`：使用通配符过滤需要用like这个关键词

1. %

```
select prod_id, prod_name from profuctys where product_name like 'jzh%';
```

意思就是可以接受以jzh开头的所有内容

```
select prod_id, prod_name from profuctys where product_name like '%jzh';
```

意思就是可以接受以jzh结尾的所有内容

```
select prod_id, prod_name from profuctys where product_name like '%jzh%';
```

意思就是可以接受存在jzh的所有内容

注意：%不能匹配null，请不要用来匹配null

mysql默认是不区分大小写的，所以`JZH`也是可以的

而想要区分大小写，请加上`COLLATE latin1_bin`，即

```
select prod_id, prod_name from profuctys where product_name COLLATE latin1_bin like 'jzh%';
```

所以说，%可以匹配0，1，无穷个字符

2. _

_与%类似，只不过它能且仅能匹配一个字符，多一个少一个都不行

```
select prod_id, prod_name from profuctys where product_name like 'jzh_';
```

意思就是可以接受以jzh开头且只有4位的内容

```
select prod_id, prod_name from profuctys where product_name like '_jzh';
```

意思就是可以接受以jzh结尾且只有4位的内容

```
select prod_id, prod_name from profuctys where product_name like '_jzh_';
```

`ajzhb`可以，`ajzh`,`jzhb`不可以

tips

1. 不要过度使用通配符
2. 尽量不要把通配符放到最前面，效率很低

正则表达式

请先自学正则表达式，这里就不再赘述了

基本字符匹配

`sdelect prod_name from products where prod_name regexp '正则表达式';` `ajzhb`可以，`ajzh`,`jzhb`不可以

regex和like的区别

regex是匹配列值的内部，而like是匹配整个列值

9.计算字段

`计算字段`：当字段并不存在时，但我们需要合并输出一些字段时，我们就需要生成一些计算字段进行处理（例如通过购买数目和价格求出总价）

`拼接字段`：不需要计算，但需要将他们拼接起来

1. 拼接字段

```
select concat(vend_name,'(', vend_contry,')') from vendor;
```

最终得到的就是Hangzhou(China)

tips:去掉右边空格，可以使用RTrim函数

e.g.`select concat(RTrim(vend_name),'(', vend_contry,')') from vendor;`

使用别名

```
select concat(vend_name,'(', vend_contry,')') as vend_title from vendor;
```

这样子生成的就和一个表列类似

2. 计算字段

```
select prod_id, quantity, item_price, quantity*item_price as expanded_price from orderitems where order_num = 20005
```

这样子就可以通过价格乘以数量进行计算总价

（加+，减-，乘*，除/）

10.函数

函数：mysql中用函数进行数据的处理

函数

1. 文本处理

函数	说明
left(str, num)	取左侧num个字符
length()	求长度
locate(substr，str)	找出一个串的子串
lower()	转换为小写
ltrim()	去掉左边空格
right()	取右边num个字符
rtrim()	去掉右边空格
soundex()	发音联想？返回soundex值
substring(str，num)	返回字串的字符（第一个是1）
upper()	转换为大写

2. 日期时间处理

函数	说明
adddate()	增加一个日期
addtime()	增加一个时间
curdate()	返回当前日期
curtime()	返回当前时间

函数	说明
date()	返回日期时间的日期部分
datediff()	计算两时间之差
date_add()	日期运算函数
date_format()	格式化日期或时间
day()	返回日期的天数
dayofweek()	返回星期几
hour()	返回小时
minute()	返回分钟
now()	返回现在日期+时间
second()	返回现在的秒
time()	返回现在的时间部分
year()	返回年份的部分
month()	返回月份的部分

3. 数值处理函数

函数	说明
abs()	返回一个数的绝对值
cos()	返回一个角度的余弦
exp()	返回一个数的指数
mod()	返回一个操作的余数
pi()	返回圆周率
rand()	返回随机数
sin()	返回sin值
sqrt()	返回平方根
tan()	返回tan值

4. 聚集函数

函数	说明
ave()	平均值
count()	行数

函数	说明
max()	最大值
min()	最小值
sum()	求和

e.g. `select ave(prod_price) as ave_price from products;`

`select count(*) as row_num from products;`

tips：当需要求不同值时，我们可以采用distinct参数

`select ave(distinct prod_price) as ave_price from products;`

组合聚集函数

`select ave(prod_price) as ave_price, max(prod_price) as max_price from products;`

11.分组数据

`分组`的作用，就是将同一个数据表中的内容进行分组，就可以实现分开使用聚集函数

1. 创建分组

使用group by 子句进行创建

`select vend_id, count(*) as num_prods from products group by vend_id;`

以`vend_id`进行分组，求出不同组的个数

tips:

1. select中每个列都必须再group by子句中给出
2. group by子句必须出现在where后，在order之前

2.过滤分组

使用`having`进行分组的过滤

`select cust_id, count(*) as orders from orders group by cust_id having count(*) >= 2;`

过滤出了统计数据大于等于2的商单来

where和having的区别

都是过滤，但是where是在数据过滤之前完成，而having是在数据过滤之后完成

学了的总结

`select id,count(*) as row_num from student where id > 100 group by id having row_num > 10 order by id limit 10,100`

12.子查询（查）

普通子查询

如果存在`关系表`，我们可以通过子查询进行查询

```
select order_num from orderitems where prod_id = 'TNT2';
```

得出20005和20007

```
select cust_id from orders where order_num in (20005,20007);
```

不如直接

```
select cust_id from orders where order_num in (select order_num from orderitems where prod_id = 'TNT2');
```

其实本质上就是嵌套的一种，但是不建议嵌套太多，速度太慢，阅读起来也不是很方便

计算字段使用子查询

~~有意思~~

```
select cust_name, cust_state,(select count(*) from orders where orders.cust_id = customers.cus_id) as orders from customers order by cust_name;
```

13.联结表与高级联结

MySQL最强的功能就是连结(join)。——Ben Forta

关系表

每个类别的物品占一行，存储产品信息，然而同一个供应商可能有多种产品，那么在哪放供应商信息呢？一般我们选择将供应商信息分开存储，这样子可以保证同一个供应商的信息是一样的，同时，可以保证修改时只需要修改一处即可，让报表一致（防止一样的数据出现多次重复）

所以在此，我们建立两个表格，一个存储物品，一个存储供应商信息。vendors表格包含供应商信息，每个供应商都有一个独立的id（主键），而product中存储产品信息，它只存储供应商的id，不存储供应商的信息，所以vendors的主键（id）就是products的外键，将vendors和products进行关联

创建联结

```
select vendors.vend_name, products.prod_name, products.prod_price from vendors, products where vendors.vend_id = products.vend_id order by vend_name, prod_name;
```

使用where创建联结

如果不使用`where`进行联结的化，会输出笛卡尔积，这里就不是联结了

内部联结（等值联结）

```
select vend_name, prod_name,prod_price from vendoes inner join products on vendors.vend_id = products.vend_id;
```

等价于

```
select vendors.vend_name, products.prod_name, products.prod_price from vendors, products where vendors.vend_id = products.vend_id order by vend_name, prod_name;
```

联结多个表格

```
select prod_name, vend_name, prod_price, quantity from orderitems, products, vendors where products.vend_id = products.prod_id and order_num = 20005;
```

高级联结

1. 使用表的别名

当一句SQL语句中多次出现同一个表名时，我们可以取一个别名来减少SQL语句

E.G.

```
select cust_name, cust_contact from customers as c, orders as o, orderitems as oi where c.cust_id = o.cust_id and oi_order_num = o.order_num and prod_id = 'TNT2';
```

可以大大缩短联结的长度。

2. 不同的联结方法

1.自联结

自己和自己联结

法1：`select prod_id, prod_name from products where vend_id = (select vend_id from products where prod_id = 'DTNYR');`

法2：`select p1.prod_id, p1.prod_name from products as p1, prodycts as p2 where p1.vend_id = p2.vend_id and p2.prod_id = 'DTNTR';`

2. 自然联结

自然联结排除多测出现同样的列（而以上的所有联结都是自然连接，需要非自然联结请自行搜索）

3. 外部联结

```
select customers.cust_id,orders.order_num from customers inner join orders on customers,cust_id = orders.cust_id;
```

3. 带聚集函数的联结

```
select customers,cust_name, customers.cust_id, count(order.order_num) as num_ord from customers inner join orders on customers,cust_id = orders.cust_id group by customers.cust_id;
```

4. 使用联结和联结条件

一般使用内部联结，但外部联结也有效

请多次注意联结的语法

14.组合查询（查）

组合查询

MySQL支持多个select查询，并返回单个查询结果

创建组合查询

我们一般使用`union`来组合数条MySQL语句，并将结果组成单个结果集（同时显示，不是与，而是或）

e.g.

```
select vend_id, prod_id,prod_price from products where prod_price <= 5 union select end_id, prod_id, prod_price from products where vend_id in (1001,1002);
```

部分规则：

1. union需要多条select语句
2. union中必须有相同的列
3. 列的数据类型得兼容
4. union自动删除重复的列

union和or的区别

union可以在不同的表之间union，但or只能在同一个表内部

15.全文本搜索

全文本搜索曹勇`match()`和`against()`一起使用

e.g.

```
select note_text from productnotes where match(note_test) against('rabbit');
```

选出有`rabbit`内容的句子

这个可以保证次序是相同的，而like的次序不一定是相同的

其他搜索方法

1. 查询扩展

   使用`with quert expension`

2. 布尔文本搜索

   使用`in boolean mode`

```
select note_text from productnotes where match(note_test) against('rabbit' with quert expension);
```

16.插入数据（增）

1. 插入完整的行

```
insert into 表名 values(数据内容);
```

```
insert into 表名(列的内容) values(数据内容);
```

e.g.

```
insert into customers values(null,'jzh',19,'male');
```

`insert into customers(uuid,name,age,sex) values(null,'jzh',19,'male');`

第二个方法更安全，因为第二个方法你顺序可以换下，但第一个方法你顺序一定要锁定和设计是一样的

2. 插入多行

`insert into 表名 values(数据内容),(数据内容);`

`insert into 表名(列的内容) values(数据内容),(数据内容);`

e.g.

`insert into customers values(null,'jzh',19,'male'),(null,'jzh2',19,'male');`

`insert into customers(uuid,name,age,sex) values(null,'jzh',19,'male'),(null,'jzh2',19,'male');`

当然你也可以一句句的插入

3. 插入查询结果

将查询出来的内容存到另外一个数据库中

`insert into customers(name,age,sex) select name,age,sex from students;`

`insert into customers(name,age,sex) select name,age,sex from students where age > 18;`

17.修改数据（改）

修改数据，使用`update`来进行更新

`update 表名 set 列名 = 更新值 where 条件;`

e.g.

`update students set age = 18 where name = 'jzh';`

`update students set age = 18,tel = '123456789' where name = 'jzh';`

18.删除数据（删）

删除数据使用`delete`进行是实现

`delete from 表名 where 条件;`

e,g,

`delete from customers where name = 'jzh';`

19.使用视图

`视图`：虚拟的表，是动态检索数据的查询

视图的优势：

1. 重用SQL语句
2. 简化操作
3. 保护原有数据

视图的限制：

1. 唯一命名
2. 要有高权限
3. order by 在视图中，若视图中select中有order by会被覆盖
4. 视图不能索引

视图的创建：

用`create view`语句创建，用`show create view viewname`查看，用`drop view viewname`来删除

e.g.

```
create vier productcustomers as select cust_name, cust_contact, prod_id from customers, orders, orderitems where customers, orders, orderitems where customers.cust_id = orders.cust_id and orderitems.order_num = orders.order_num;
```

这样就创建了一个视图了

```
select * from productcustomers;
```

当然我们也可用视图重新格式化检索出数据

```
create view vendorlocations as select concat(rtrim(vend_name),'(', rtrim(vend_country),')') as vend_title from vendors order by vend_name;
```

同样，我们可以使用视图过滤一些不需要的内容，使用视图与计算字段

更新视图：

视图没有数据，所以更新视图都是对基表的更新

20.使用存储过程

`存储过程`：使用和保存一条或者多条语句的集合，可以视为批处理文件

存储过程的优势：

1. 可以简化复杂操作
2. 增加代码复用性
3. 降低代码出错的可能性
4. 提高性能

使用存储过程：

```
call productpricing(@pricelow, @pricehigh, @priceaverage);
```

调用productpricing，返回最高，最低和平均值

创建存储过程：

```
crearte procedure productpricing()
begin
    select avg(prod_price) as priceaverage from products;
end;


call productpricing()
```

删除存储过程：

```
drop procedure productpricing;
```

使用参数：

```
crearte procedure productpricing(
    out pl decimal(8,2),
    out ph decimal(8,2),
    out pa decimal(8,2)
)
begin
    select min(prod_price) into pl from products;
    select max(prod_price) into pm from products;
    select avg(prod_price) into pa from products;
end;

call productpricing(@pricelow, @pricehigh, @priceaverage);



crearte procedure ordertotal(
    in onumber int
    out ototal decimal(8,2)
)
begin
    from orderitems where order_num = onumber into ototal;
end;

call ordertotal(20005, @total)
```

检查存储过程：

`show create procedure ordertotal;`

21.使用游标

`游标`：存储在MySQL上的数据库查询结果集

使用游标：

1. 先需要声明定义它
2. 一旦声明，必须使用游标
3. 结束时必须关闭游标

创建游标：

```
create procedure processorders()
begin
    declare ordernumbers cursor for select order_num from orders;
end;
```

就创建了一个名为`ordernumbers`的游标

打开关闭游标：

```
open ordernumbers;
close ordernumbers;
```

使用游标数据：

```
creare procedure processorders()
begin
    declare o int;
    declare ordernumbers cursor for select order_num from orders;
    open ordernumbers;
    fetch ordernumbers into o;
    close ordernumbers;
end;


creare procedure processorders()
begin
    declare done boolean default 0;
    declare o int;
    declare t decimal(8,2);
    declare ordernumbers cursor for select order_num from orders;
    declare continue handler for sqlstate '02000' set done=1;
    create table if not exists ordertotals(order_num int, totak decimal(8,2));
    open ordernumbers;
    repeat
        fetch ordernumbers into o;
        call ordertotal(o,l,t);
        insert into ordertotals(order_num, total) values(o,t);
    until done end repeat;
    close ordernumbers;
end;
```

当然也可以通过游标来完成循环读数据，或者不断的读取下一个数据，这里就不再赘述了

22.使用触发器

`触发器`：在事件发生时自动执行（就是在任何情况下自动执行的一条MySQL语句）

创建触发器：

```
create trigger newproduct after insert on products for each row select 'product
added';
```

就是在每行执行插入成功后，输出product added的消息

每个表最多支持6个触发器（insert, update, delete之前和之后）

删除触发器：

```
drop trigger newproduct;
```

使用触发器：

1. **insert触发器**

   `create trigger newproduct after insert on products for each row select 'product added';`

2. **delete触发器**

   `create trigger newproduct after delete on products for each row select 'product deleted';`

3. **update触发器**

   `create trigger newproduct after update on products for each row select 'product updated';`

23.管理事务处理

`事务处理`：是保证成批的MySQL操作完全执行或完全不执行，不会出现执行一半的现象

`回退`：撤销指定SQL语句的过程

`提交`：将为存储的SQL语句结果写入数据表

`保留点`：事务处理中设置的临时占位符

控制事务处理：

1. 开始

   `start transaction;`

2. 使用ROLLBACK

   ```
   select * from ordertotals;
   start transaction;
   delect from ordertotals;
   select * from ordertotals;
   rollback;
   select * from ordertotals;
   ```

3. 使用commit

```
start tramsaction;
delete from orderitems where order_num = 20010;
delete from order where order_num = 20010;
commit;
```

4. 使用保留点

```
savepoint delete1;
rollback to delete1;
```

5. 更改默认的提交行为

```
set autocommit = 0;
--0的时候默认不提交
```

24.安全管理

应该对用户设有权限，尽量降低权限，防止得到不能达到的东西（尽量不要使用root，权限太高了）

管理用户：

```
use mysql;
select user from user;
```

创建用户账号：

```
create user ben identified by 'p@$$w0rd';
```

删除用户账号：

```
drop user bforta;
--删除bforta
```

设置访问权限：

```
show grants for bforta;

revoke select on 'crashcourse' .* from bforta
```

更改口令：

```
set password for bforta = password('123456');

set password = password('123456');
```

25. 数据库的维护

为了防止数据库出现问题，我们建议使用多重备份，防止数据丢失

数据库维护：

1. 检查键表是否正确

```
analyze table orders;
```

2. 对表进行检查

```
check table order, orderitems;
```

启动诊断问题：

`--help`，`--safe-mode`，`--verbose`，`--version`

查询日志文件：

1. 错误日志：`hostname.err`
2. 查询日志：`hostname.log`
3. 二进制日志：`hostname-bin`
4. 缓慢查询日志：`hostname-slow.log`

26.TIPS

1.全球化和本地化

1. 展示字符集

   `show character set;`

2. 查看支持校对的完整列表

   `show collation;`

3. 创建时选择不同的编码类型`character set`和`collate`

```
create table mytable
(
    column1 int,
    column2 varchar(10)
)default character set hebrew collate hebrew_general_ci;
```

4. 给不同的列设置不同的编码格式

```
create table mytable
(
    column1 int,
    column2 varchar(10),
    column3 varchar(10) character set latin1 collate latin1_general_ci
)default character set hebrew collate hebrew_general_ci;
```

5. 如果绝对需要，可以在字符间使用`cast()`和`convert()`函数

2.改善性能

1. 根据自己的电脑配置调整内存分配，缓冲区大小等内容
2. 如果执行卡住了，可以使用kill杀线程
3. 不要检索比你需求还多的数据

27.拓展知识

`Mybatis`： 半自动的映射持久层框架

`Hibernate` ： 全自动的映射持久层框架