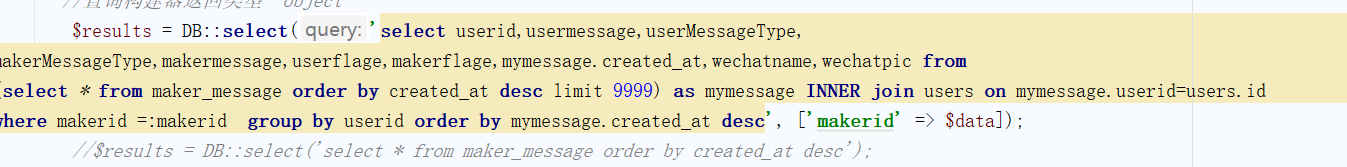
1. mysql5.7子查询中order by 要加limit
2. 
3. Access denied for user 'root'@'localhost' (using password: YES)

My.conf添加 skip-grant-tables跳过密码认证

mysql修改账号密码

update mysql.user set authentication\_string='123456' where user='root';

flush privileges;更新数据库

## 简单操作

### 数据库连接

mysql –u root -p

### 选择数据库

use databaseName(数据库名);

**例如**

use score;

### 查看数据库列表

show databases;

### 查看表列表

show tables;

### 显示表列

Book表

show columns from book;

## 创建、删除,选择数据库

create database 数据库名；

create database book;

drop database 数据库名；

drop database book;

use 数据库名；

use book；

## 创建删除数据表

crate table 数据表名；

drop table 数据表名；

CREATE TABLE IF NOT EXISTS `runoob\_tbl`(

`runoob\_id` INT UNSIGNED AUTO\_INCREMENT,

`runoob\_title` VARCHAR(100) NOT NULL,

`runoob\_author` VARCHAR(40) NOT NULL,

`submission\_date` DATE,

PRIMARY KEY ( `runoob\_id` )

)ENGINE=InnoDB DEFAULT CHARSET=utf8;

## 插入数据

Insert into 数据表名 （字段名,..,字段名）values（字段值,..,字段值）;

Insert into book (‘name’,’price’) values(‘php’,’120’);

## 更新数据

update 数据表名 set 字段名=字段值…字段名=字段值 where ..

update book set name=”java”,writer=”caa” where price>120;

## 删除数据

DELETE FROM table\_name [WHERE Clause]

delete from book where id=1;

---------------------------------------------查询语句

## 单表查询与多表查询见sqlServer的word

## MySQL ALTER命令

当我们需要修改数据表名或者修改数据表字段时，就需要使用到MySQL ALTER命令。

### 删除，添加或修改表字段

Alter table 数据表名 drop 字段名；

Alter table 数据表名 add 字段名 类型；

ALTER TABLE testalter\_tbl DROP i;

ALTER TABLE testalter\_tbl ADD i INT FIRST; 指定新增字段的位置

ALTER TABLE testalter\_tbl DROP i;

ALTER TABLE testalter\_tbl ADD i INT AFTER c; 指定新增字段在哪个字段之后

**改字段类型及名称**

Alter table 数据表名 modify 字段名 类型

Alter table 数据表名 change原字段名 新字段名 类型

ALTER TABLE testalter\_tbl MODIFY c CHAR(10);

ALTER TABLE testalter\_tbl CHANGE j j INT;

## 索引

### 普通索引



### 唯一索引



## 视图

视图是虚拟的表，是动态的查询数据，数据表中的数据改变视图中的数据也会改变

### 为什么使用视图

重用SQL语句

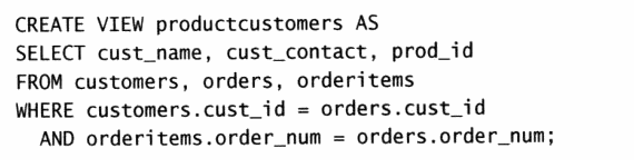
简化复杂的SQL操作，在编写查询后，可以方便的重用它而不必知道它的查询细节

使用表的部分而不是全部

保护数据。可以给用户授权特定部分的访问权限而不是全部表的权限

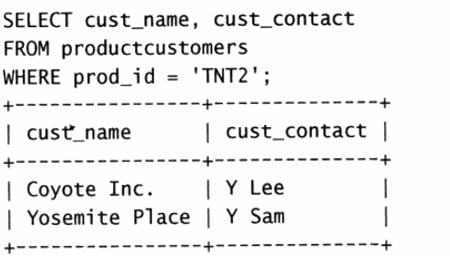
### 视图的创建

create view 视图名 as select语句；



### 检索视图

使用视图的时候就和表一样



### 删除视图

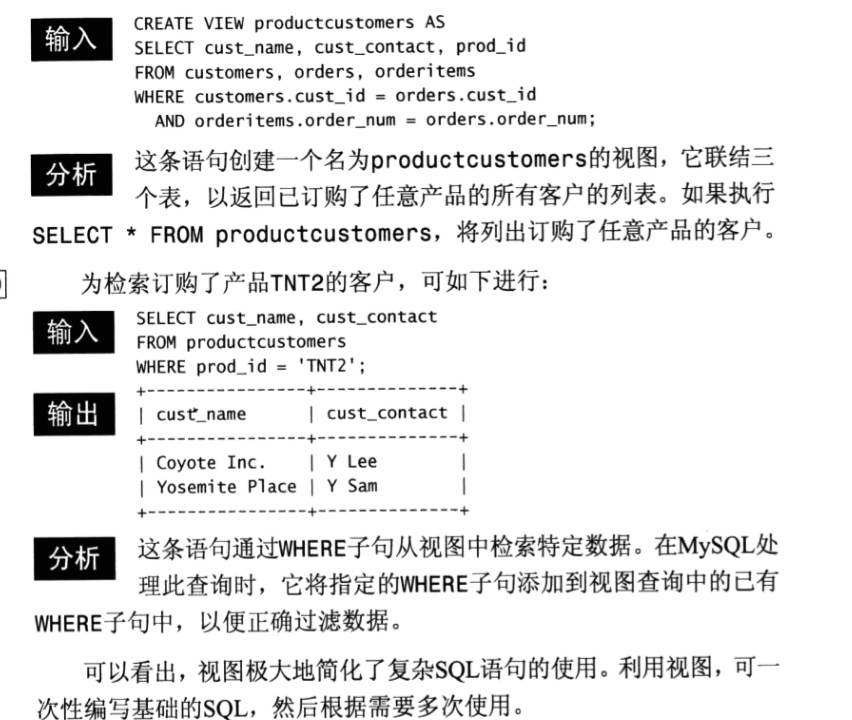
drop view 视图名；

### 查看创建视图语句

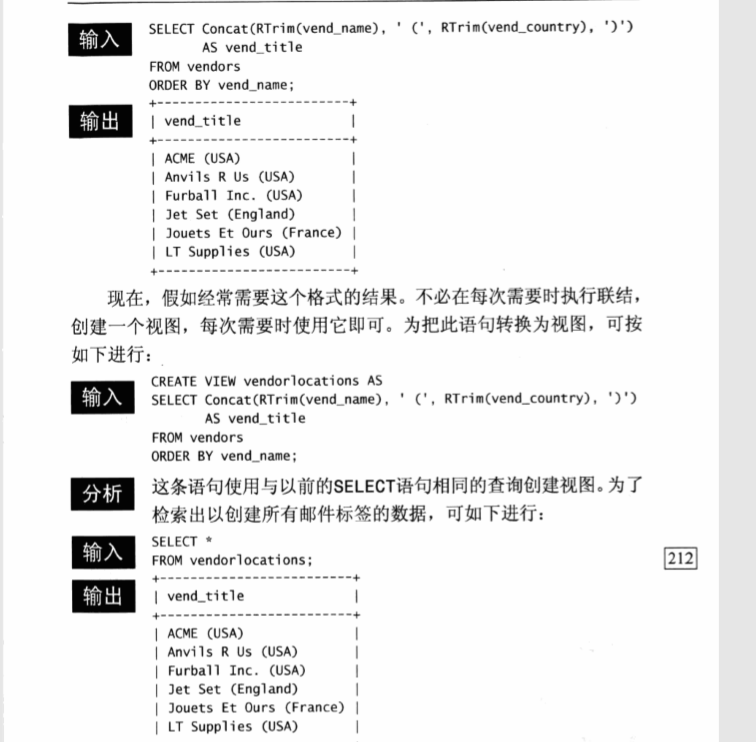
Show create view 视图名；

### 视图使用场景

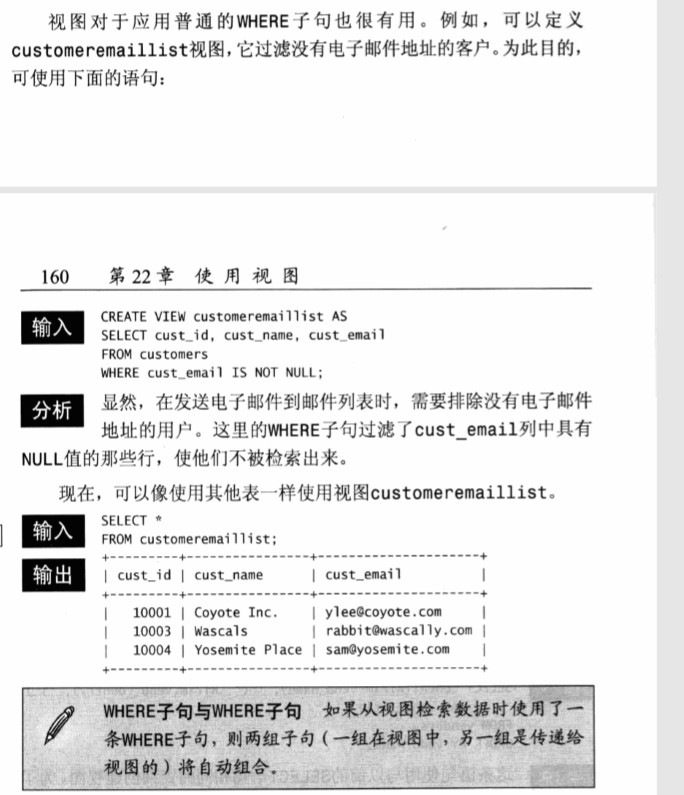
#### 视图简化复杂的联结



#### 视图重新格式化索引的数据



#### 过滤不想要的数据



## 存储过程

### 使用场景

在实际工作中可能会碰到这样的需求，我经常会进行一些操作，这些操作是固定的，但是每次操作都要写几十条语句，这时候就要用到存储过程

### 什么使存储过程

组为了完成特定功能的SQL 语句集，它存储在数据库中，一次编译后永久有效，用户通过指定存储过程的名字并给出参数（如果该存储过程带有参数）来执行它。存储过程是数据库中的一个重要对象。

### 存储过程优点

通过封装复杂的SQL语句集来简化操作

由于不要求反复建立一系列处理步骤，这保证了数据的完整性

简化对变动的管理，如果表明、列明或业务逻辑有变化，只需要改变存储过程的代码。使用人员甚至都不需要知道这些变化

提高性能

### 创建存储过程

#### 不携带参数

Create produce name()

Begin

Select \* from book;

End;

#### 携带参数

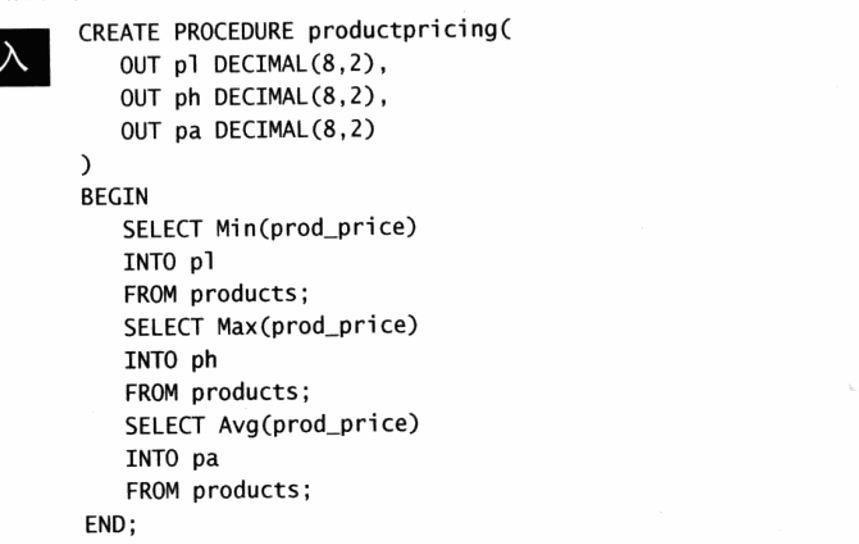
Create produce 存储过程名称（out 参数名 参数类型，…，out 参数名 参数类型）

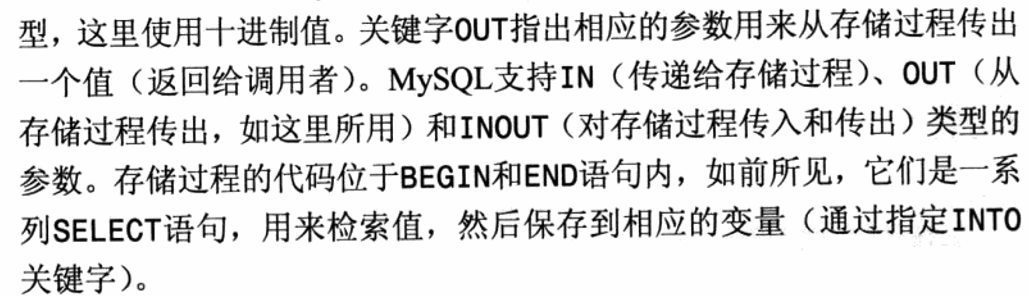
begin

select id into 参1 from A;

select name into 参数2from B;

end;



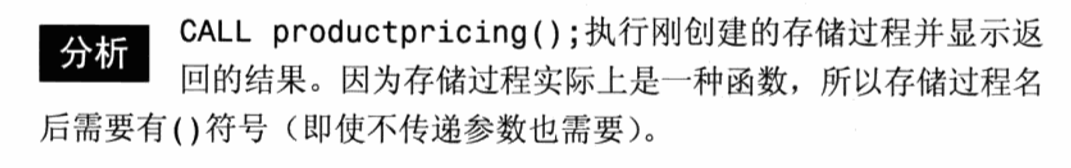


### 执行存储过程

#### 不携带参数

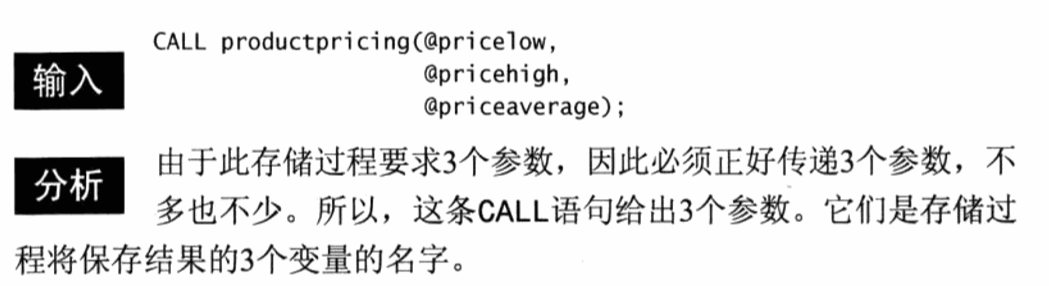
Call 存储过程名称；

Call name();



#### 携带参数

Call存储过程名称（@参数，…，@参数）



### 删除存储过程

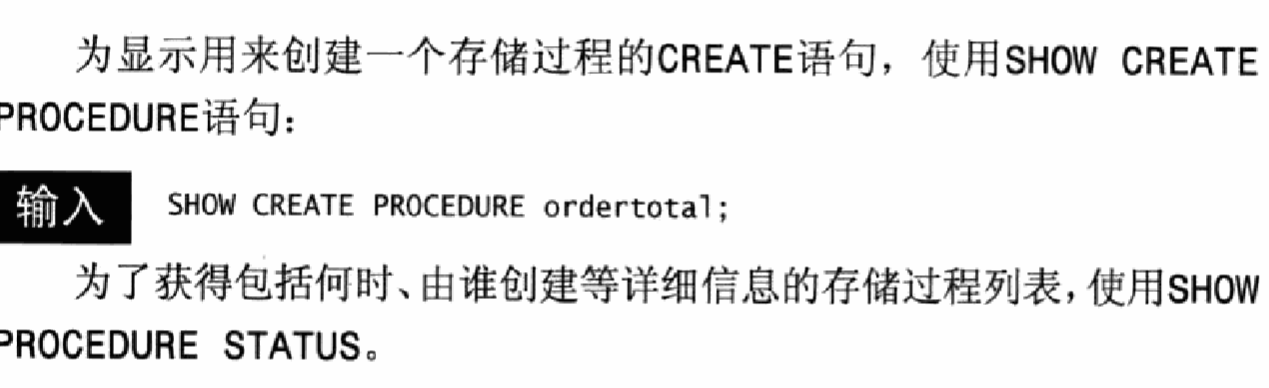
drop produce 存储过程名称；

drop produce name;

### 检查存储过程

为了显示存储过程的create语句

Show create produce 存储过程名称；



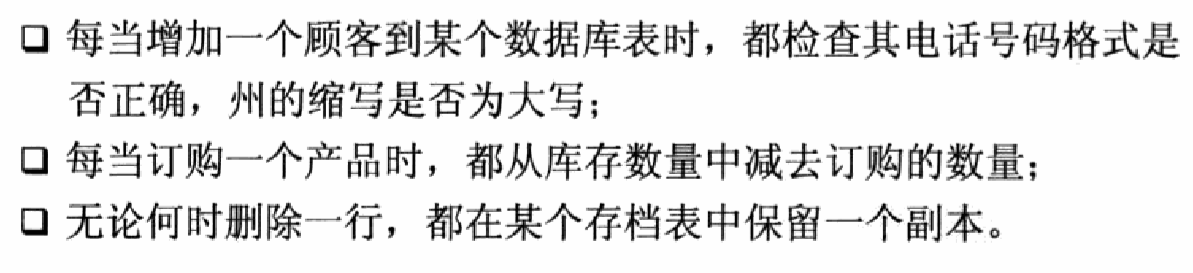
## 游标

以后补充

## 触发器

pass

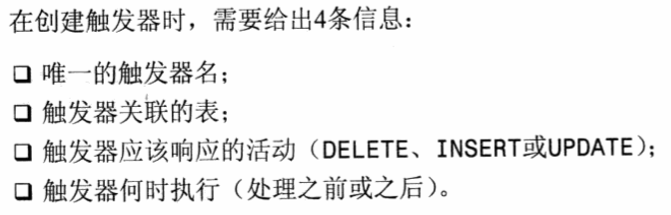
### 使用场景



### 什么是触发器

触发器是让想要的某些SQL语句自动执行的Sql语句；

### 创建触发器



Create trigger 触发器名称 after SQL语句 for SQL 语句

### 使用触发器

### 删除触发器

## Mysql事务

### 使用场景

MySQL 事务主要用于处理操作量大，复杂度高的数据。比如说，在人员管理系统中，你删除一个人员，你既需要删除人员的基本资料，也要删除和该人员相关的信息，如信箱，文章等等，这样，这些数据库操作语句就构成一个事务！

### 事务特性

**原子性：**一个事务（transaction）中的所有操作，要么全部完成，要么全部不完成，不会结束在中间某个环节。事务在执行过程中发生错误，会被回滚（Rollback）到事务开始前的状态，就像这个事务从来没有执行过一样。

**一致性：**在事务开始之前和事务结束以后，数据库的完整性没有被破坏。这表示写入的资料必须完全符合所有的预设规则，这包含资料的精确度、串联性以及后续数据库可以自发性地完成预定的工作。

**隔离性：**数据库允许多个并发事务同时对其数据进行读写和修改的能力，隔离性可以防止多个事务并发执行时由于交叉执行而导致数据的不一致。事务隔离分为不同级别，包括读未提交（Read uncommitted）、读提交（read committed）、可重复读（repeatable read）和串行化（Serializable）。

**持久性：**事务处理结束后，对数据的修改就是永久的，即便系统故障也不会丢失。

**BEGIN** 开始一个事务

**ROLLBACK** 事务回滚

**COMMIT** 事务确认

