|  |  |
| --- | --- |
| 成绩评定 |  |
| 教师签名 |  |

**嘉应学院 计算机学院**

**数据库课程设计**

**实验报告**

|  |  |
| --- | --- |
| **课程名称：** | **数据库课程设计** |
| **开课学期：** | **2018-2019第一学期** |
| **班 级：** | **软件工程1602** |
| **指导老师：** | **陈旭东** |
| **实验题目：** | **实验11： 数据库完整性设计与触发器设计** |
| **学 号：** | **161060009** |
| **姓 名：** | **黄佳俊** |

实验11 数据库完整性设计与触发器设计

一、实验目的

通过本实验掌握数据库完整性设计的基本方法及触发器设计方法。

二、实验原理

数据库的完整性是指数据的正确性和相容性。为维护数据库的完整性，任何数据库管理系统必须 提供定义完整性约束条件的机制，提供完整性检查的方法

及违约处理。

数据库完整性包括 实体完整性，参照完整性及用户定义完整性。

触发器（Trigger）是用户定义在关系表上的一类由事件驱动的特殊过程，由服务器自动激活可以进行更为复杂的检查和操作，具有更精细和更强大的数据控制能力。

创建触发器语法格式

CREATE TRIGGER <触发器名>

{BEFORE | AFTER} <触发事件> ON <表名>

FOR EACH ROW

［WHEN <触发条件>］

<触发动作体>

触发时间：Before, AFTER

触发事件： insert,update, delete

删除触发器： drop trigger <触发器名>

显示创建触发器脚本

>show create trigger <触发器名>

>show triggers;

三、实验环境

操作系统： win7

开发环境：数据库管理服务器 MYSQL 5.5，mysql IDE 管理平台 MySQL Workbench 5.2.44 CE

四、实验内容方法

（一）完整性设计

在E盘建立子目录 db, 将老师提供的两个数据库脚本(trade\_script.txt，TradeDBSQL.txt)放置此目录中。

1.在msyql 上建立数据库 trade

在命令行窗口， 运行 mysqld --consol 开启服务器

在另一个命令行窗口，运行mysql -u root -p 登录系统。

在mysql>窗口下，分别运行

1） mysql>source e:\db\trade\_script.txt

将自动创建trade1 数据库，有完整性设计，没有数据。

2） mysql>source e:\db\tradeDBSQL.txt

将自动创建trade 数据库，无完整性设计，有数据。

3) 打开 MySQL Workbench.exe,通过反向工程方法，建立数据库trade1,trade的数据模型图（ER）图，比较两张图的异同

对照 trade\_script.txt 脚本生成的数据库trade1, 找出每张表的

1. 实体完整性，（主键不为空不为空唯一）
2. 参照完整性，（表间规则，外键）
3. 及用户定义完整性（例如no null, default 等）顾名思义

并且在trade数据库中 增加完整性约束：

方法1：用sql 命令

例如 在Suppliers 表中增加实体完整性（主键名为 spk ）约束

Alter table Suppliers add constraint spk Primary Key(SupplierID);

在Products表中增加参照完整性(外键fk\_supp\_casc)约束 并且实现级联更新

Alter table Products add constraint fk\_supp\_casc foreign key (SupplierID)

references Suppliers(SupplierID)

ON UPDATE CASCADE;

方法2: 用workbench 平台。

（二）触发器 -实现业务逻辑的完整性（一致性）

1 用触发器实现参照完整性级联约束

更新级联触发器 orderd\_upd

DELIMITER %% /\* 用%% 代替; 作为命令结束符 \*/

Create trigger orderd\_upd /\* 创建触发器 名为orderd\_upd \*/

after update on Orders /\* 在表orders 上 建立 更新后触发器 \*/

for each row /\* 对于每一行 \*/

begin

/\* 在表orders上更新后 自动更新OrderDetails 表 \*/

update OrderDetails set OrderID = new.OrderID where OrderID=old.OrderID;

end %%

DELIMITER;

特别要注意new,old 的区别

old 表示更新前的一行记录（for each row）

new 表示更新为新的数据的一行记录。

2. 用触发器实现用户定义完整性约束

Mysql 没有check 约束，只能通过触发器实现check 约束

例如 产品表Products 中，规定单价UnitPrice的范围是大于0，小于100，设计触发器prod\_ins实现该约束

delimiter %%

create trigger prod\_ins

before insert on Products

for each row

begin

if new.UnitPrice <=0 or new.UnitPrice>=100 then

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = '不能插入数据，单价只能是大于0小于100';

end if;

end;

%%

3.用触发器实现业务逻辑完整性

新订单业务逻辑

1）在订单orders中添加一条新纪录

2）在订单细节Orderdetails表中添加订购产品的相关记录

3）更新修改产品表中相关产品的库存量，减去订购的数量

重点在2),3)之间的业务逻辑的一致性！

在OrderDetails 表上设计一个插入触发器，自动实现3）的更新库存的操作。

DELIMITER %%

create trigger insertorder

before insert on OrderDetails

for each row

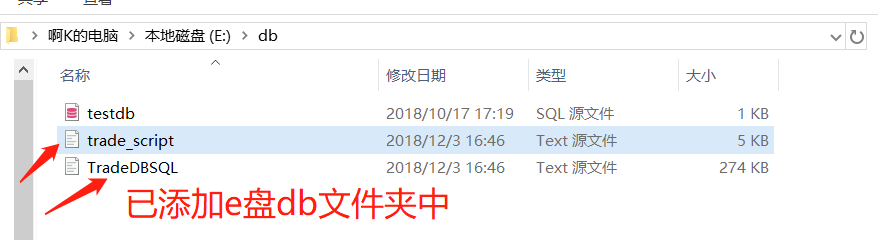
begin

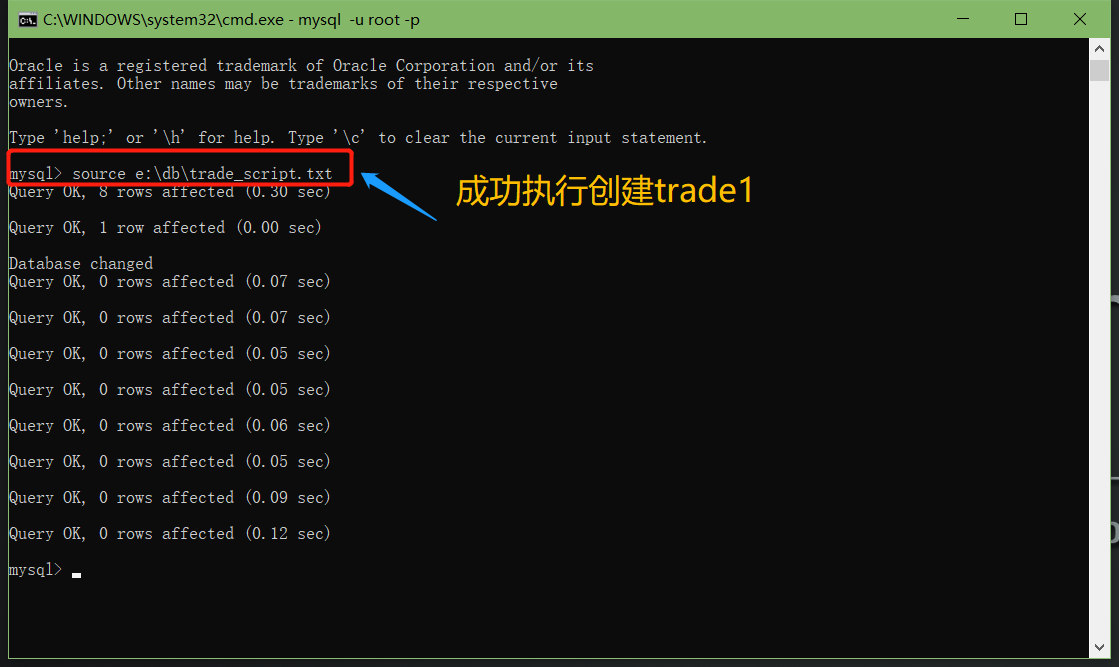
update Products set UnitsInStock = UnitsInStock - new.Quantity where ProductID = new.ProductID;

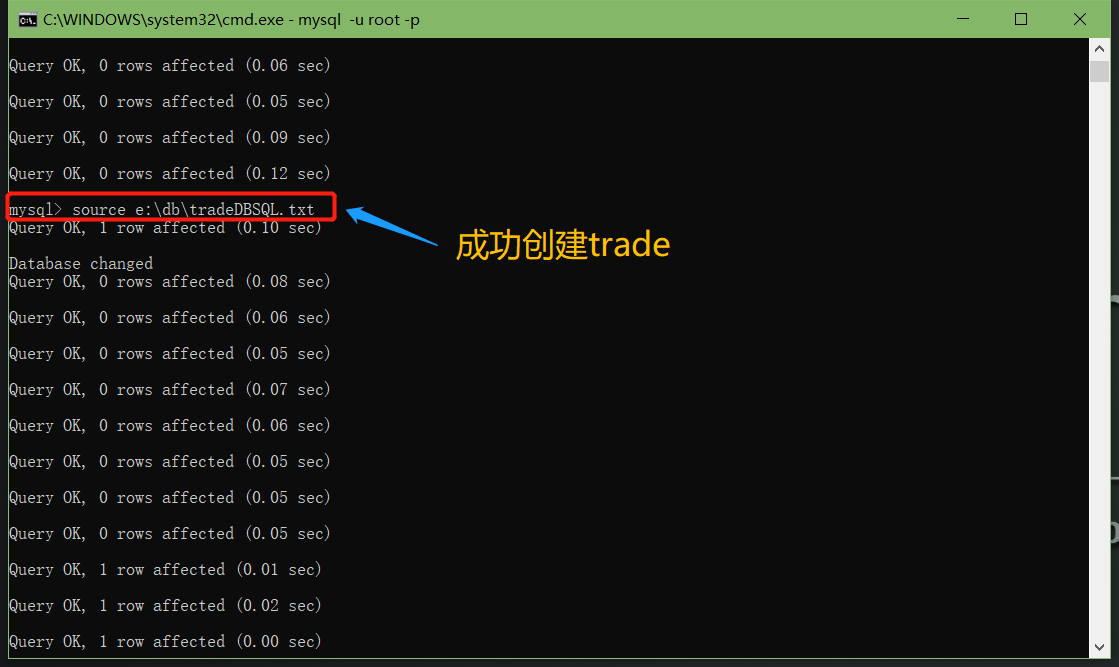
end

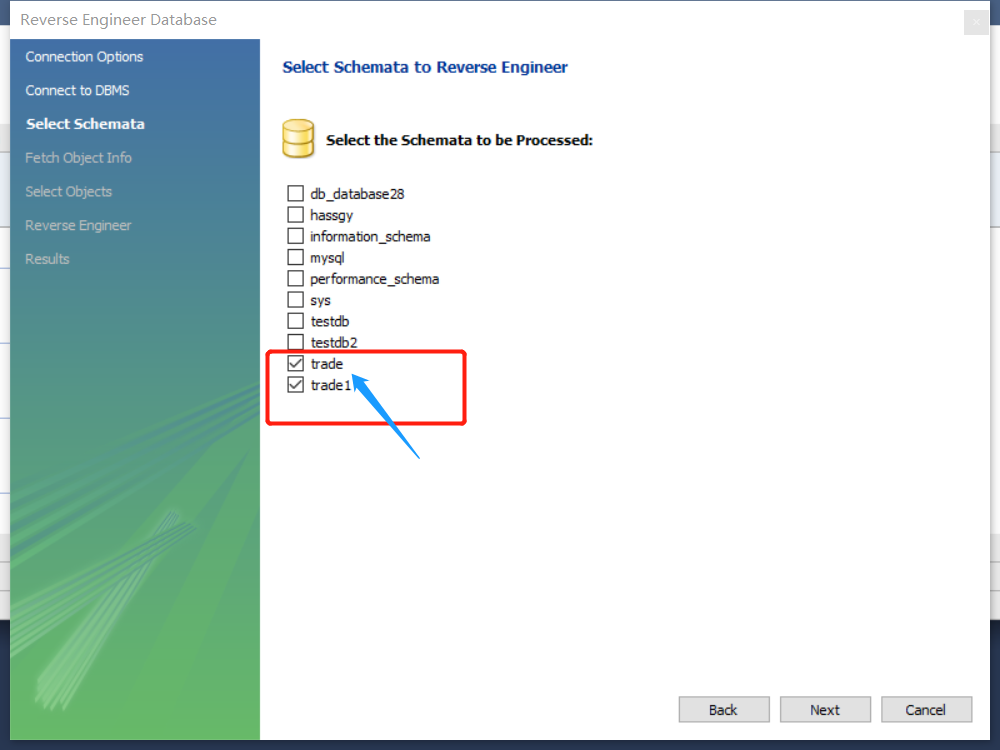
%%

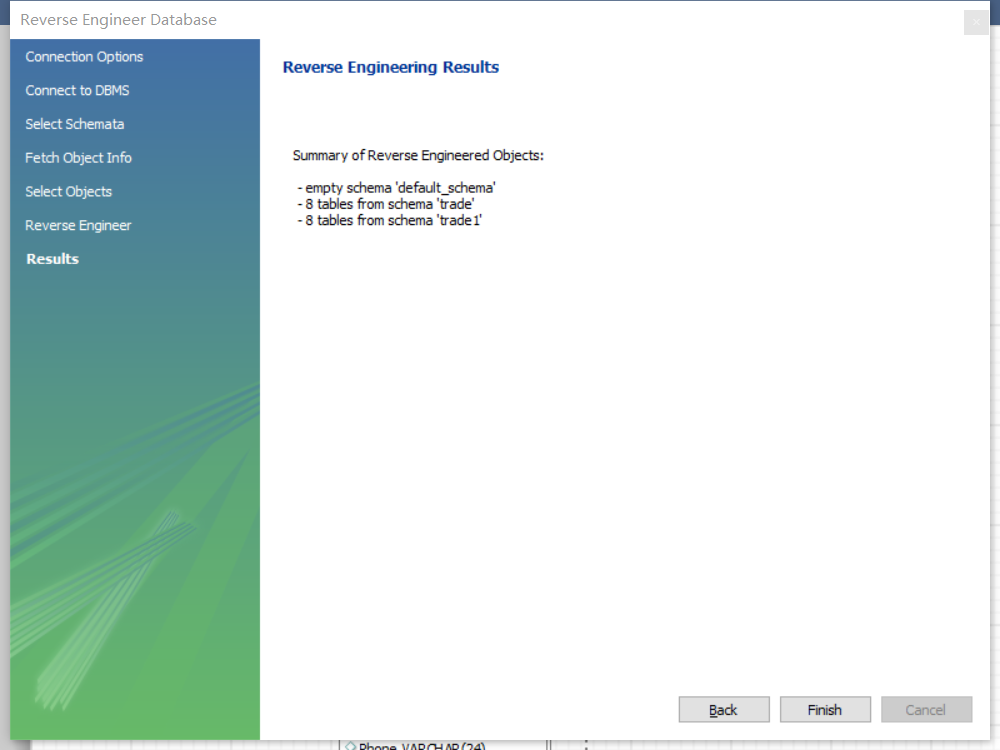
实验结果如下：



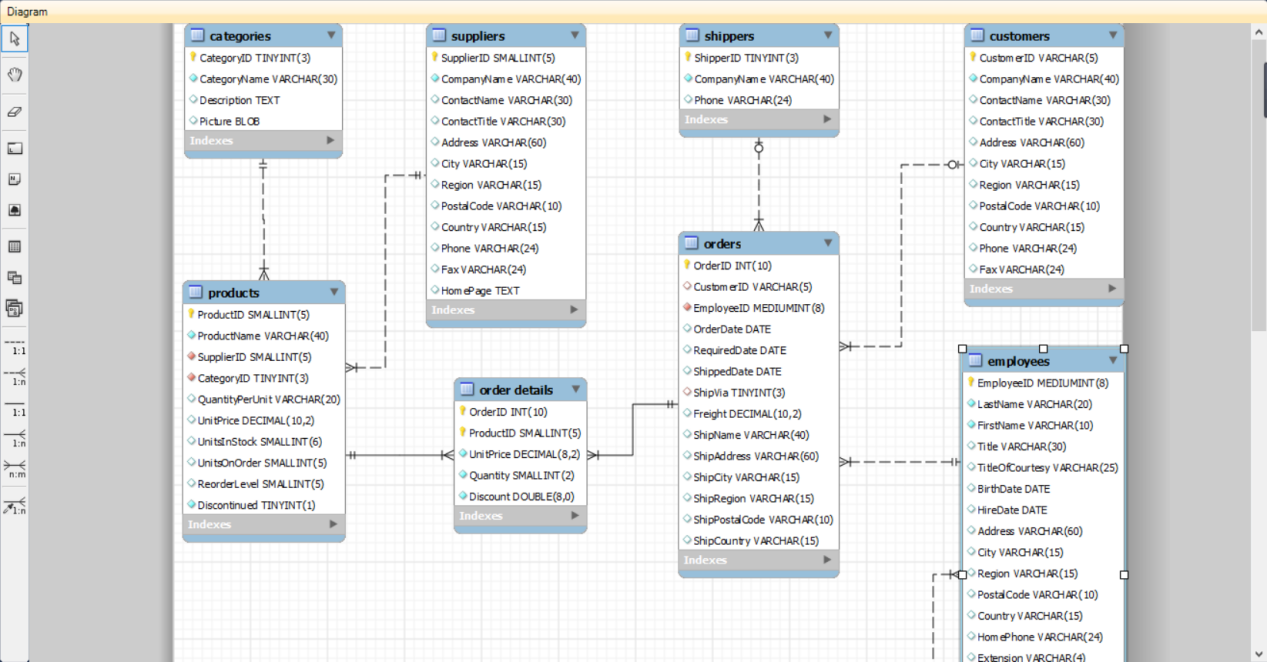






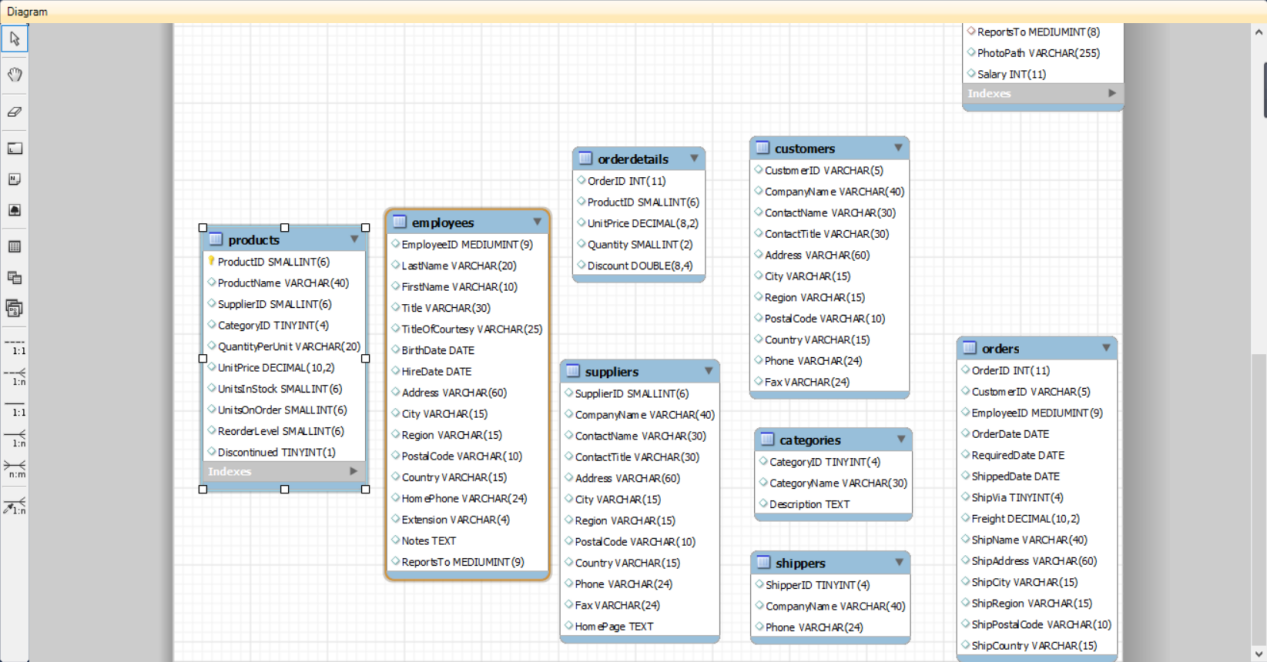


完整性约束的trade（ER图）



没有完整性的trade1（ER图）

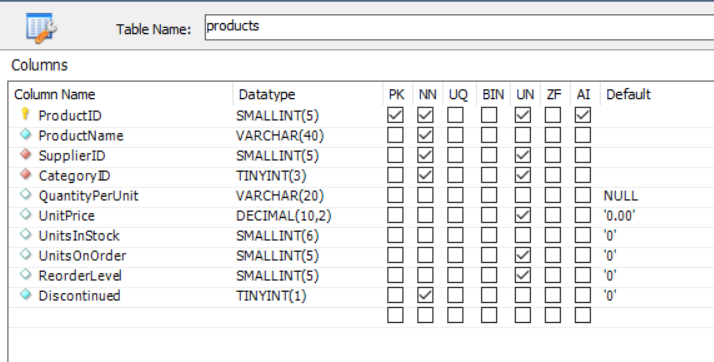
最大区别就是缺少键的问题以及不严谨，缺乏完整性约束条件。



对照 trade\_script.txt 脚本生成的数据库trade1,

**实体完整性**，参照完整性和用户自定义完整性约束。实体完整性规定表的每一行在表中是唯一的实体。  
**参照完整性**指两个表的主关键字和外关键字的数据一致，保证表之间的数据一致性，防止数据丢失或无意义的数据在数据库中扩散。  
**用户自定义完整性，**是不同数据库根据应用环境不同，用户定义的一些特殊约束条件。。

**表products**



**实体完整性**：（主键不为空不重复）

ProductID；

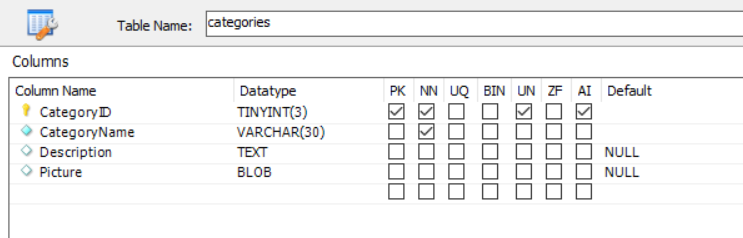
**参照完整性**：（表间规则，外键）

SupplierID；CategoryID；

**用户定义完整性**（例如not null, default 等）顾名思义：

图表中看出NN这一列向右到Default的就是属于用户定义。

**表categories**



**实体完整性**，（主键不为空不为空唯一）

CategoryID；

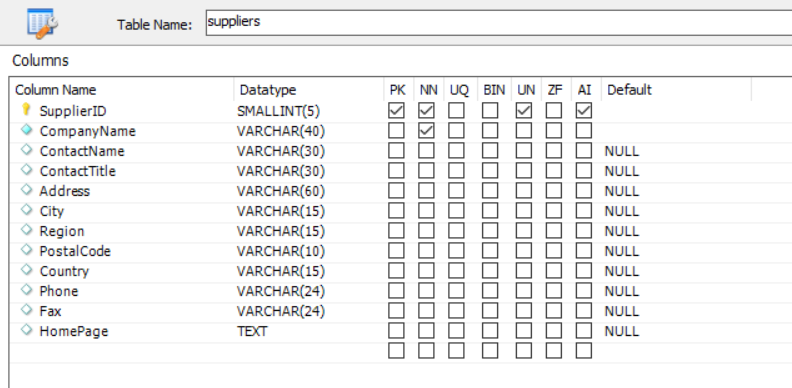
**参照完整性**，（表间规则，外键）

这里没有外键引入，没有。

**用户定义完整性**（例如no null, default 等）顾名思义：

图表中看出NN这一列向右到Default的就是属于用户定义。

**表suppliers**



**实体完整性**，（主键不为空不为空唯一）

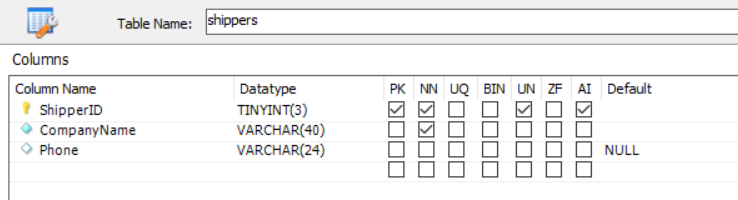
SuppliersID

**参照完整性**，（表间规则，外键）

这里没有引入外键，没有。

**用户定义完整性**（例如no null, default 等）顾名思义：  
SuppliersID；CompanyName，图表中看出NN这一列向右到Default的就是属于用户定义。

**表shippers**



**实体完整性**，（主键不为空不为空唯一）

ShipperID

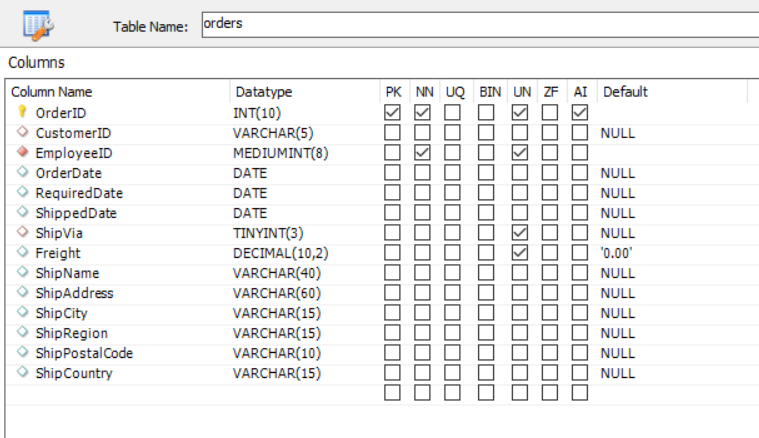
**参照完整性**，（表间规则，外键）

这里没有引入外键。

**用户定义完整性**（例如no null, default 等）顾名思义

ShipperID；CompanyName；

**表orders**



**实体完整性**，（主键不为空不为空唯一）

OrderID；

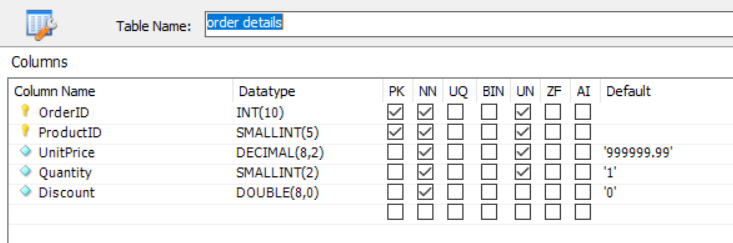
**参照完整性，**（表间规则，外键）

外键：EmployeeID

**用户定义完整性**（例如no null, default 等）顾名思义

NN，UN，AI，Default这四列打勾的就是属于此类约束。

**表order details**



**实体完整性**，（主键不为空不为空唯一）

OrderID；ProductsID；

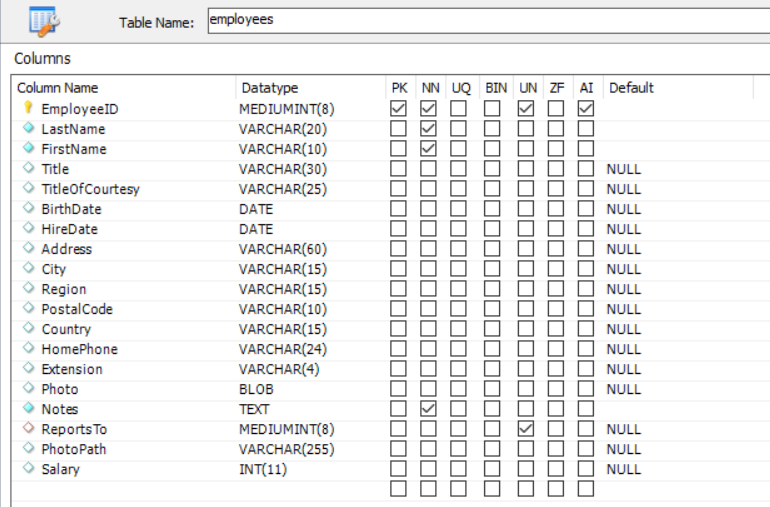
**参照完整性**，（表间规则，外键）

没有引入外表的外键，没有。

**用户定义完整性**（例如no null, default 等）顾名思义

NN，UN，Default看他们打勾的所属字段名就是。

**表employees**



**实体完整性**，（主键不为空不为空唯一）

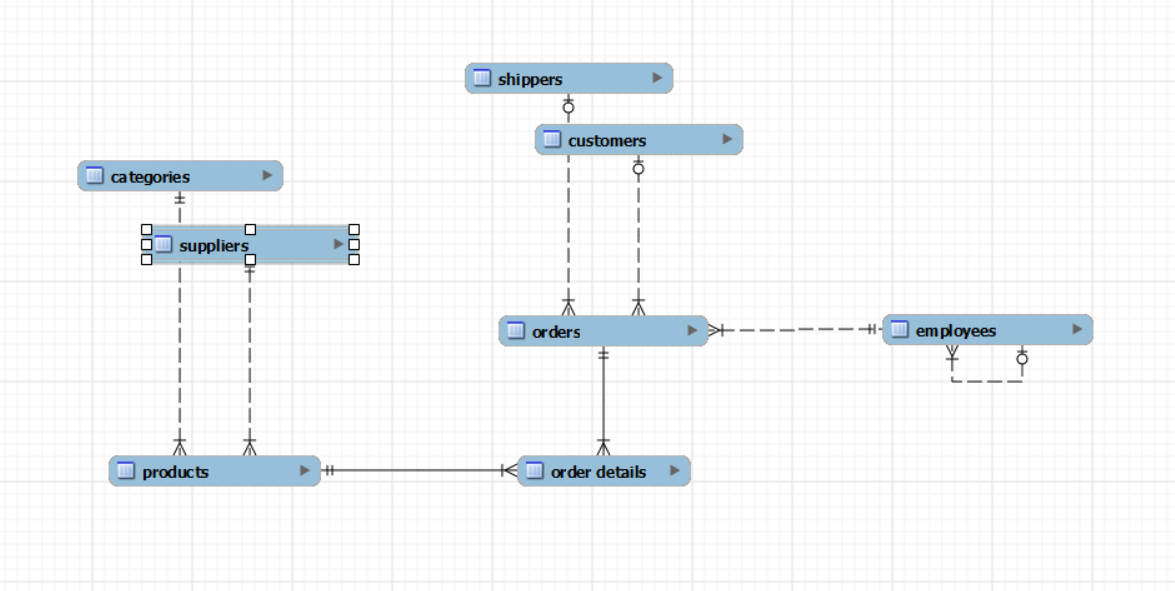
EmployeeID；

**参照完整性**，（表间规则，外键）

没有引入外键，没有

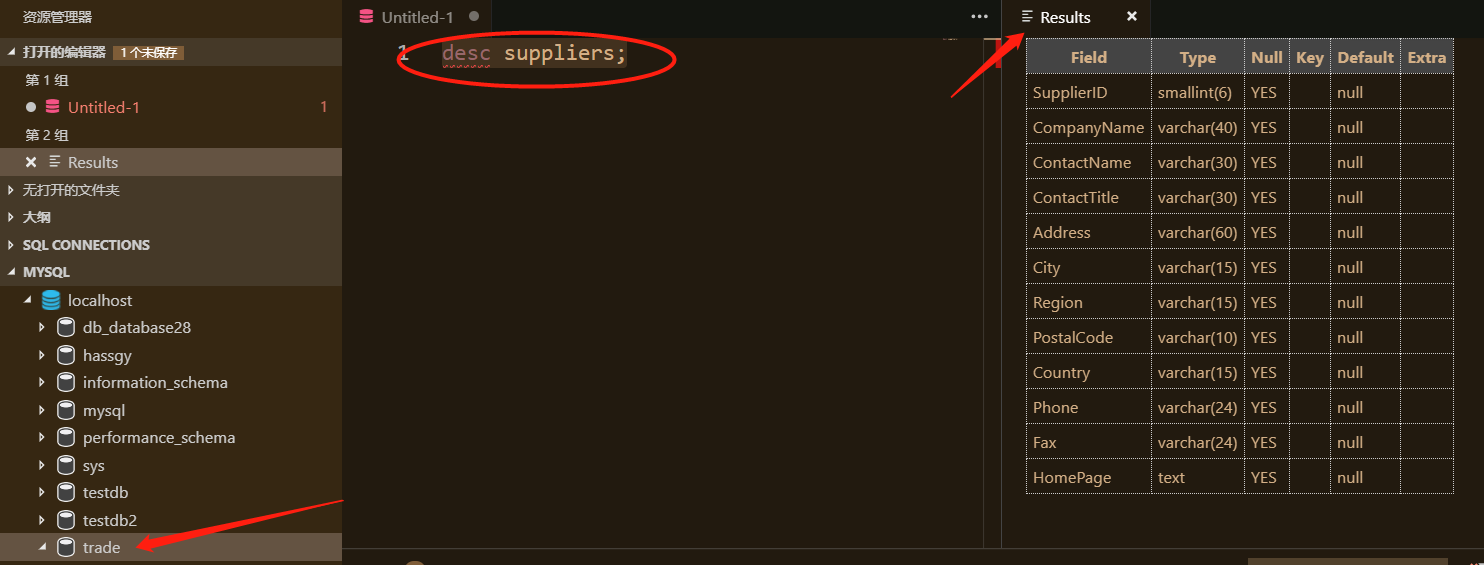
**用户定义完整性**（例如no null, default 等）顾名思义

NN，UN，AI，default列看图表。

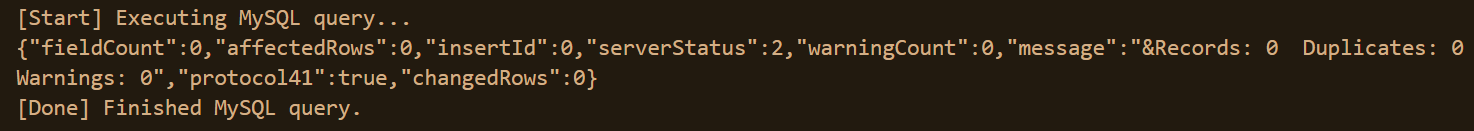
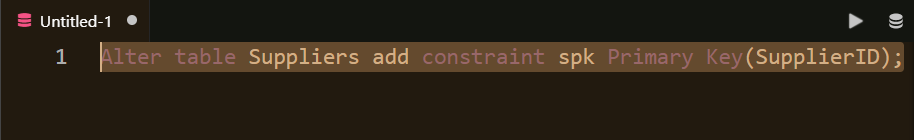


另外，再把trade的数据库ER图添加完整性约束。

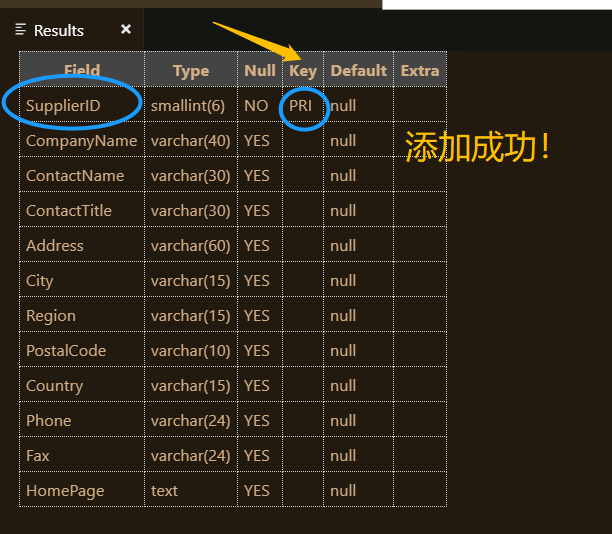
如图**suppliers**添加**主键**：



通过添加语句：

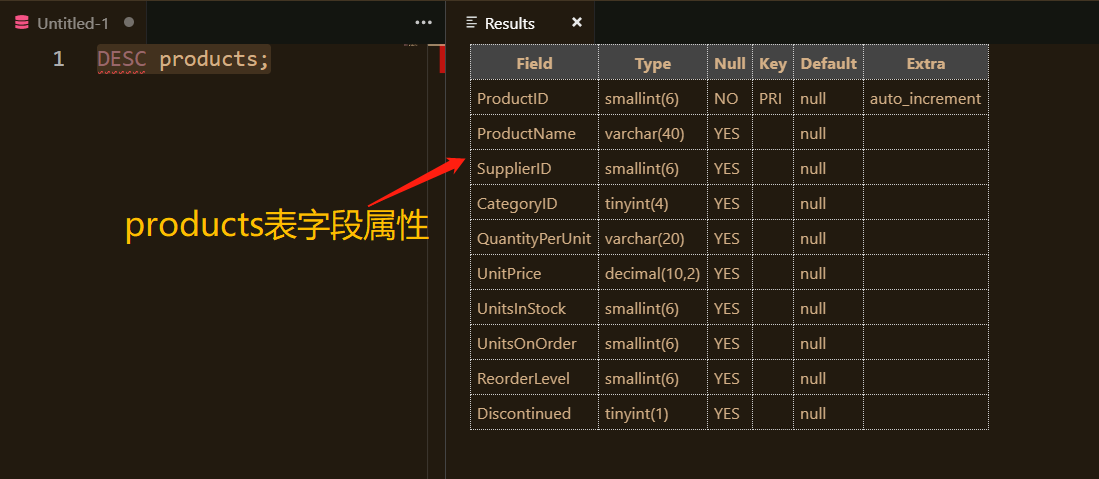


添加主键后的表属性：

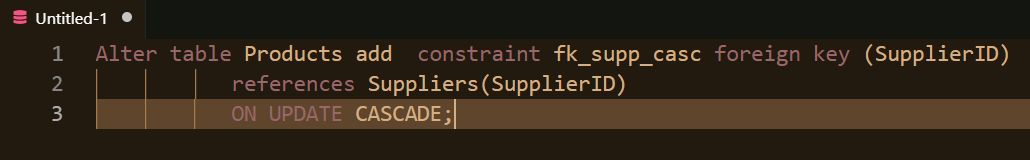


再有**products**表添加**外键**：

显示原来的状态字段名属性如下：



添加语句：



执行成功之后，查看属性。如下图：



再有**更新级联触发器 orderd\_upd**

**语句为：**

DELIMITER %% /\* 用%% 代替; 作为命令结束符 \*/

Create trigger orderd\_upd /\* 创建触发器 名为orderd\_upd \*/

after update on Orders /\* 在表orders 上 建立 更新后触发器 \*/

for each row /\* 对于每一行 \*/

begin

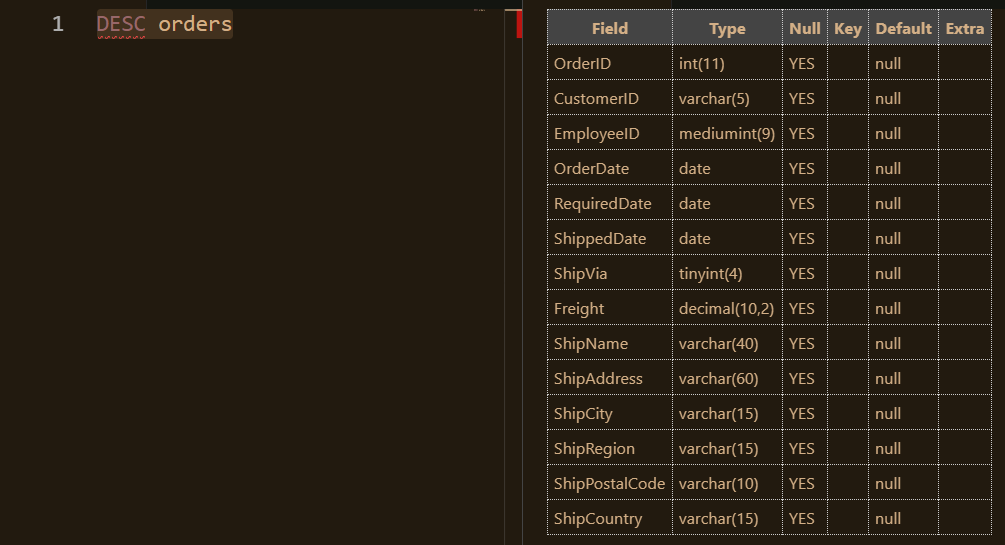
/\* 在表orders上更新后 自动更新OrderDetails 表 \*/

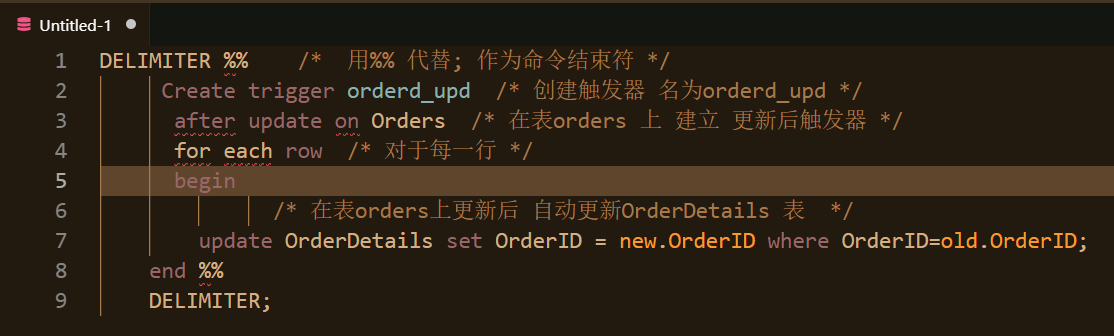
update OrderDetails set OrderID = new.OrderID where OrderID=old.OrderID;

end %%

DELIMITER;

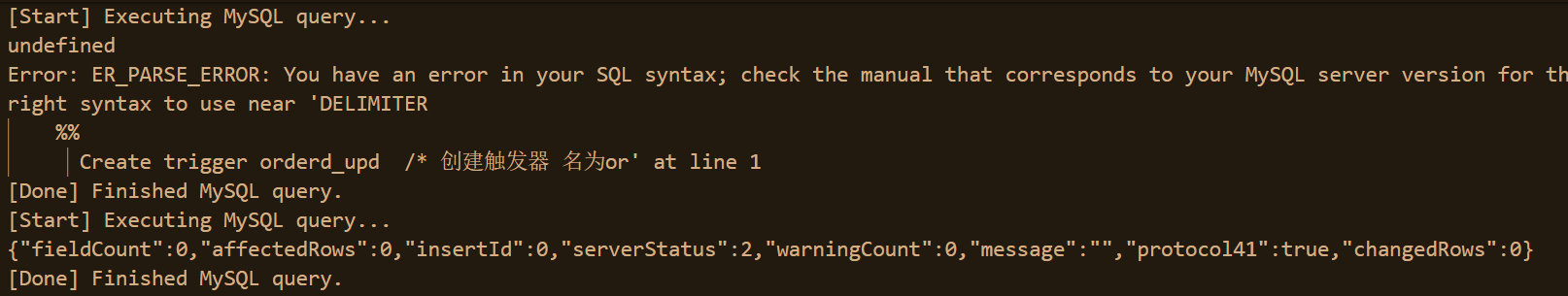
原来的**orders表**如下图：



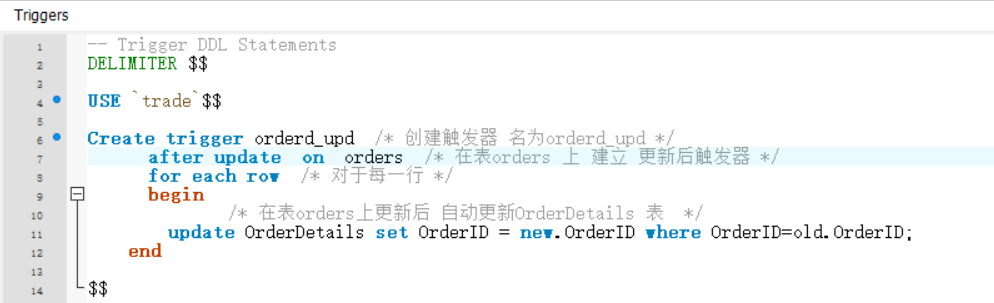


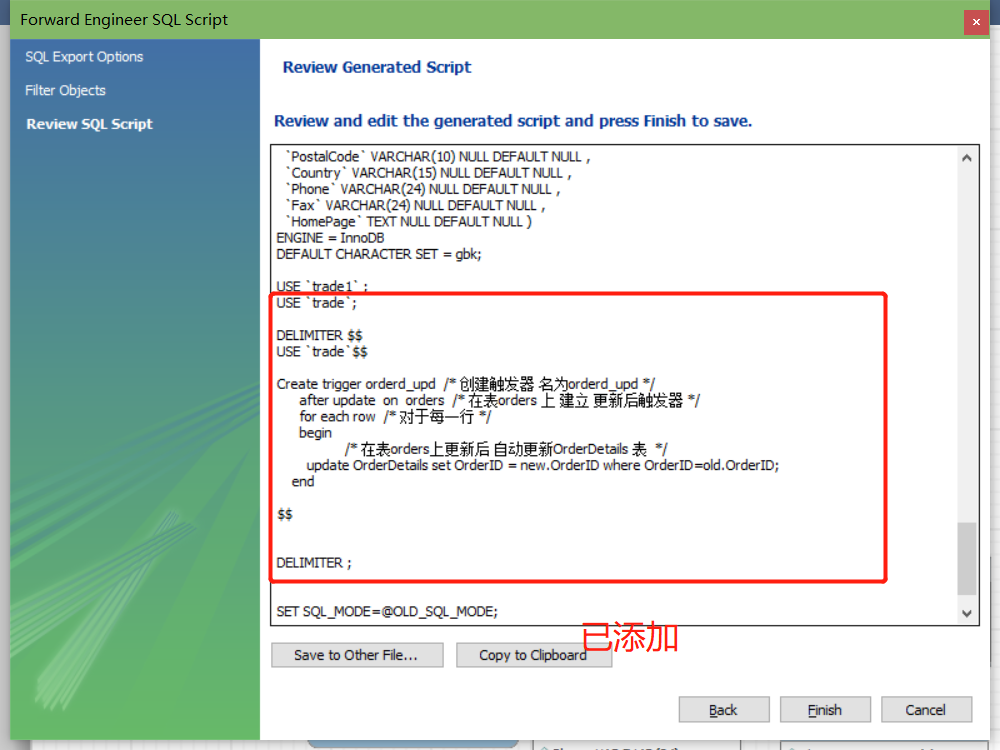
准备执行，添加级联触发器，如下图显示：

通过修改，终于添加成功。



或者再workbench执行也可以，如下图：

  
已添加：

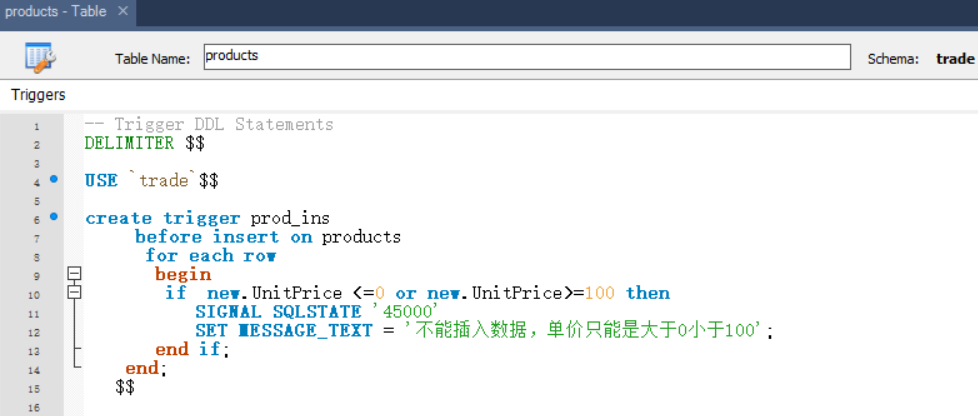


**用触发器实现用户定义完整性约束：**

Mysql 没有check 约束，只能通过触发器实现check 约束

产品表**Products** 中，规定单价UnitPrice的范围是大于0，小于100，设计触发器**prod\_ins**实现该约束

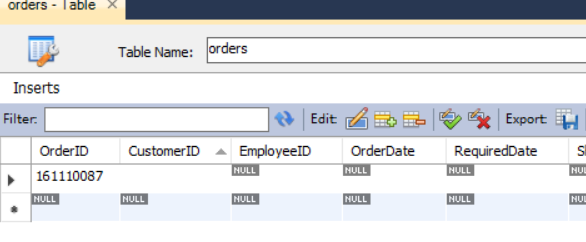
如下图语句添加表中：



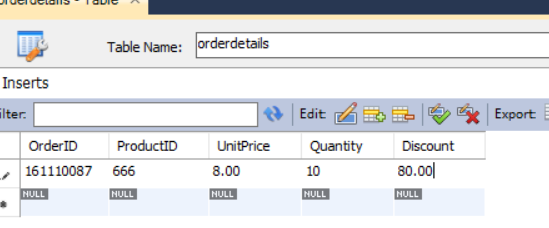
**用触发器实现业务逻辑完整性**

新订单业务逻辑

1. 在订单orders中添加一条新纪录



1. 在订单细节Orderdetails表中添加订购产品的相关记录

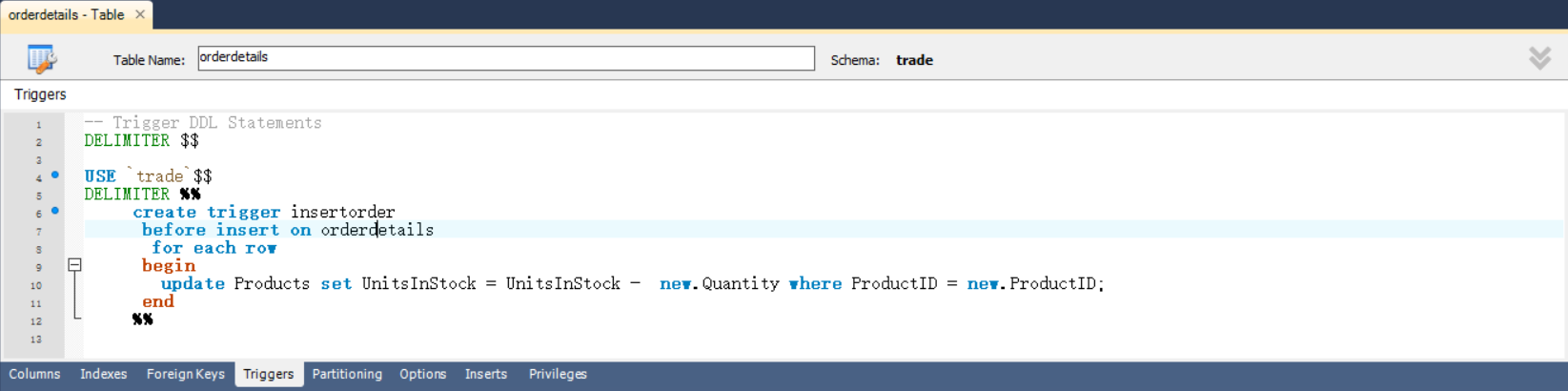


3）更新修改产品表中相关产品的库存量，减去订购的数量

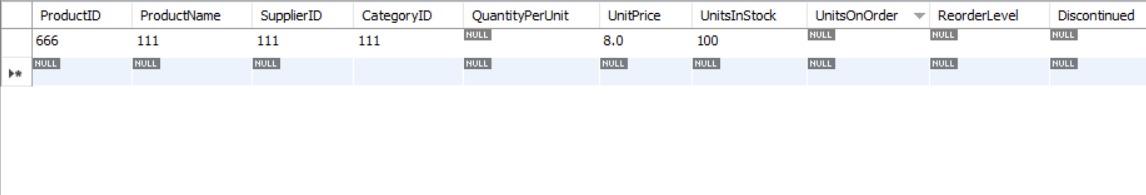
重点在2),3)之间的业务逻辑的一致性！

在OrderDetails 表上设计一个插入触发器，自动实现3）的更新库存的操作。

添加触发器的相关语句：



结果，到最后的产品总数量UnitsInStock由100变成90



变成下面图：

