华北科技学院计算机学院综合性实验

实 验 报 告

课程名称 数据库原理与应用

实验学期 2023 至 2024 学年 第 2 学期

学生所在院部 计算机科学与工程

年级 大二 专业班级 信管 B221

学生姓名 李智文 学号 202207034116

成绩评定：

1、工作量： A（ ），B（ ），C（ ），D（ ），F（ ）

2、难易度： A（ ），B（ ），C（ ），D（ ），F（ ）

3、答辩情况：

基本操作： A（ ），B（ ），C（ ），D（ ），F（ ）

代码理解： A（ ），B（ ），C（ ），D（ ），F（ ）

4、报告规范度： A（ ），B（ ），C（ ），D（ ），F（ ）

5、学习态度： A（ ），B（ ），C（ ），D（ ），F（ ）

总评成绩:

指导教师: 郭慧

计算机学院制

《数据库原理与应用》课程综合性实验报告

开课实验室： 基础二 2024 年 6 月 6 日

实验题目：电子商务网站数据库设计

一、实验目的

根据学习的数据库设计理论和方法，设计电子商务网站数据库，包括概要设计、逻辑设计和物理设计。设计完成后，使用 mysql 数据库管理系统创建数据库并插入数据，再在建表的基础上创建索引等优化数据库相关表查询性能。

二、设备与环境

硬件：多媒体计算机
软件：windows，mysql 数据库管理系统

三、实验内容

1. 系统概述

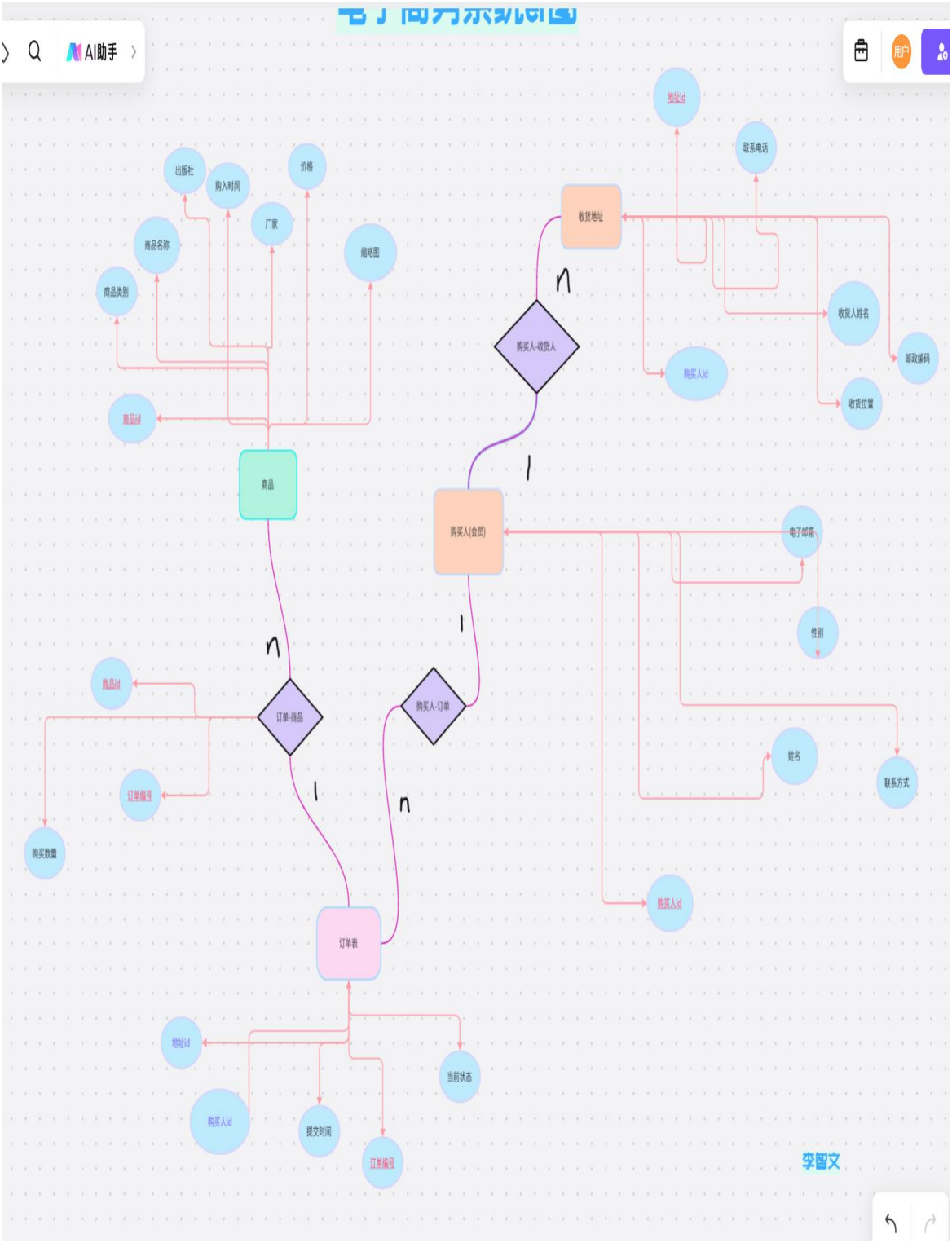
实验以电子商务网站（如淘宝，京东）为基础，根据网站提供的各种功能来进行有关电子商务网站数据库的设计，我们将电子商务网站的业务分为了商品管理部与商品销售部，其中商品管理部面向商户，主要负责管理商品的各项数据，包括商品名称，价格，购入时间等。商品销售部面向客户，要负责管理有关客户的相关信息与商品的购买，同时包括了订单，购买人，收货人收货地址的相关信息。两部门之间通过订单商品关系表相关联。

2. 数据库概要设计（E-R 图）

设计思路：用户在网页会浏览商品，其中展示的是商品表的各项信息，用户在选购商品时会先将商品加入购物车，此时后台会产生订单以及订单编号（用户不可见且订单状态为未提交），用户选购完商品时进入购物车检查购买的所有商品以及数量，此时向用户展示的是订单商品表，确认无误后用户提交订单，这时要求用户输入送货地址，由于送货地址与收获人信息在送货地址表中存放，用户可直接选择数据库中存放的送货地址，也可以新增地址存到送货地址表中。（这样把送货地址单独存放一张表就可以解决用户同时提交多个订单而每个订单收货地址不同的问题，订单表中只需要一个地址 id 外键关联到收货地址表就能找到相应的收货地址与收货人信息）当用户提交订单之后可将订单表展示给用户等待用户支付。

设计细节：收货地址中包含一个属性为购买人 id，外键关联到购买人表中的主键 id，这样既可以看到每个收货地址由哪个用户创建，也可以防止多个购买人给同一个人买东西时相关信息无法匹配的问题。

在订单商品关系表中，订单编号与商品 id 为联合主键，同时也作为外键关联到商品表与订单表，可保证数据之间的关联而不会出错



3. 数据库逻辑设计

将 E-R 图转换成关系模式如下

设计逻辑：在 E-R 图中“购买人-订单”和“购买人-收货人”两表为弱关系类型，不单独建表，两表之间直接通过外键关联。而订单商品属于强关系表，单独建表

3. 数据库实现

(1) 数据库表结构

表名：商品

名	类型	长度	小数点	不是 null	虚拟	键	注释
商品id	int	11	0	☑	☐	1	
商品名称	varchar	100	0	☑	☐		
商品类别	varchar	10	0	☐	☐		
价格	decimal	10	2	☑	☐		
生产厂家	varchar	100	0	☐	☐		
上次购入时间	date	0	0	☐	☐		
详细信息	text	0	0	☐	☐		
缩略图路径	varchar	255	0	☐	☐		

表名：购买人

名	类型	长度	小数点	不是 null	虚拟	键	注释
购买人id	int	11	0	☑	☐	1	
姓名	varchar	50	0	☑	☐		
性别	varchar	5	0	☑	☐		
联系方式	text	0	0	☐	☐		
电子邮箱	varchar	100	0	☐	☐		

表名：收货地址

名	类型	长度	小数点	不是 null	虚拟	键	注释
地址id	int	11	0	☑	☐	1	
购买人id	int	11	0	☐	☐		
收货人姓名	varchar	50	0	☑	☐		
收货地址	varchar	100	0	☑	☐		
邮政编码	varchar	20	0	☑	☐		
联系电话	varchar	20	0	☑	☐		

外键

字段 索引 外键 触发器 选项 注释 SQL 预览

名	字段	参考模式	参考表	参考字段	删除时	更新时
收货地址_ibfk_1	购买人id	dzsw	购买人	购买人id	RESTRICT	RESTRICT

表名：订单表

注：订单表中提交时间属性类型为 timestamp，默认情况下为插入数据时间

字段 索引 外键 触发器 选项 注释 SQL 预览

名	类型	长度	小数点	不是 null	虚拟	键	注释
订单编号	int	11	0	☑	☐	🔑1	
提交时间	timestamp	0	0	☑	☐		
当前状态	varchar	10	0	☑	☐		
购买人id	int	11	0	☐	☐		
地址id	int	11	0	☐	☐		

外键：

字段 索引 外键 触发器 选项 注释 SQL 预览

名	字段	参考模式	参考表	参考字段	删除时	更新时
订单表_ibfk_1	购买人id	dzsw	购买人	购买人id	RESTRICT	RESTRICT
订单表_ibfk_2	地址id	dzsw	收货地址	地址id	RESTRICT	RESTRICT

表名：订单商品表

字段 索引 外键 触发器 选项 注释 SQL 预览

名	类型	长度	小数点	不是 null	虚拟	键	注释
订单编号	int	11	0	☑	☐	🔑1	
商品id	int	11	0	☑	☐	🔑2	
购买数量	int	11	0	☐	☐		

外键：

字段 索引 外键 触发器 选项 注释 SQL 预览

名	字段	参考模式	参考表	参考字段	删除时	更新时
订单商品表_ibfk	订单编号	dzsw	订单表	订单编号	RESTRICT	RESTRICT
订单商品表_ibfk	商品id	dzsw	商品	商品id	RESTRICT	RESTRICT

表名：商品日志（触发器使用）

字段 索引 外键 触发器 选项 注释 SQL 预览

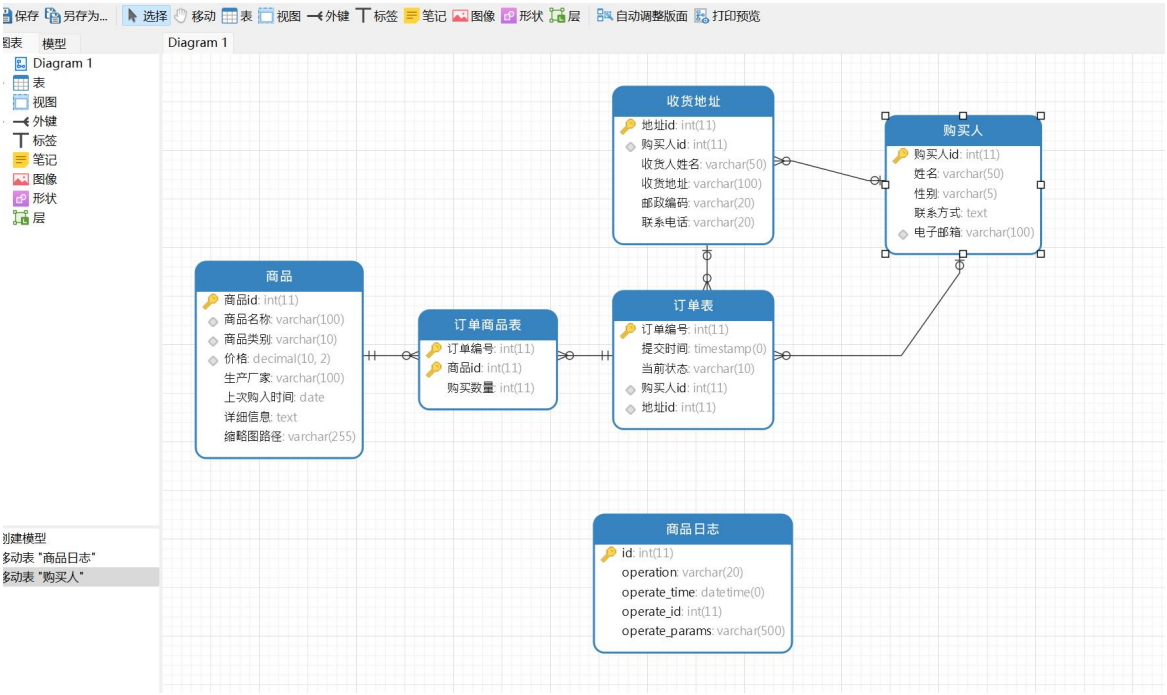
名	类型	长度	小数点	不是 null	虚拟	键	注释
*		0	0	☐	☐		
id	int	11	0	☑	☐	🔑1	
operation	varchar	20	0	☑	☐		操作类型, insert/update/delete
operate_time	datetime	0	0	☑	☐		操作时间
operate_id	int	11	0	☑	☐		操作的ID
operate_params	varchar	500	0	☐	☐		操作参数

（2）在 mysql 中创建数据表

```
1.  CREATE TABLE 商品 (
2.      商品 id INT AUTO_INCREMENT PRIMARY KEY,
3.      商品名称 VARCHAR(100) NOT NULL,
4.      商品类别 varchar(10),
5.      价格 DECIMAL(10, 2) NOT NULL,
6.      生产厂家 VARCHAR(100),
7.      上次购入时间 date,
8.      详细信息 TEXT,
9.      缩略图路径 VARCHAR(255)
10. );
```

```
11.
12.  CREATE TABLE 购买人 (
13.      购买人id INT AUTO_INCREMENT PRIMARY KEY,
14.      姓名 VARCHAR(50) NOT NULL,
15.      性别 varchar(5) NOT NULL,
16.      联系方式 TEXT,
17.      电子邮箱 VARCHAR(100) UNIQUE
18.  );
19.
20.  CREATE TABLE 收货地址 (
21.      地址id INT AUTO_INCREMENT PRIMARY KEY,
22.      购买人id INT,
23.      收货人姓名 VARCHAR(50) NOT NULL,
24.      收货地址 VARCHAR(100) NOT NULL,
25.      邮政编码 VARCHAR(20) NOT NULL,
26.      联系电话 VARCHAR(20) NOT NULL,
27.      FOREIGN KEY (购买人id) REFERENCES 购买人(购买人id)
28.  );
29.
30.  CREATE TABLE 订单表 (
31.      订单编号 INT AUTO_INCREMENT PRIMARY KEY,
32.      提交时间 TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
33.      当前状态 VARCHAR(10) NOT NULL,
34.      购买人id INT,
35.      地址id int,
36.      FOREIGN KEY (购买人id) REFERENCES 购买人(购买人id),
37.      FOREIGN KEY (地址id) REFERENCES 收货地址(地址id)
38.  );
39.
40.  CREATE TABLE 订单商品表 (
41.      订单编号 INT,
42.      商品id INT,
43.      购买数量 INT,
44.      PRIMARY KEY (订单编号,商品id),
45.      FOREIGN KEY (订单编号) REFERENCES 订单表(订单编号),
46.      FOREIGN KEY (商品id) REFERENCES 商品(商品id)
47.  );
48.
49.  create table 商品日志(
50.    id int(11) not null auto_increment,
51.    operation varchar(20) not null comment '操作类型, insert/update/delete',
52.    operate_time datetime not null comment '操作时间',
53.    operate_id int(11) not null comment '操作的ID',
54.    operate_params varchar(500) comment '操作参数',
55.    primary key(`id`)
56.  );
```

建完表后，我们可以利用 Navicat 提供的逆向表到模型来查看各个表之间的关系



如图，表之间有关联的都通过外键连线相关联，可看到各个表之间都互相关联，可保证数据不易出错

（3）插入数据

以下是相关表的数据

购买人：

开始事务 文本 筛选 排序 导入 导出

购买人id	姓名	性别	联系方式	电子邮箱
1	张三	男	123456	zhangsan@
2	李四	女	654321	lisi@examp
3	王五	男	33143	wangwu@c
4	赵四	男	22316	zhaosi@qq.
5	小红	女	73462	xiaohong@

收货地址：

注：一个购买人可能存有多个地址

地址id	购买人id	收货人姓名	收货地址	邮政编码	联系电话
1	1	张三收	北京市朝阳区XX街	100020	138001380
2	2	李四收	上海市浦东新区X	200120	139123456
3	3	齐天大圣收	花果山水帘洞	135247	100293726
4	3	猪八戒收	高老庄	250250	132749237
5	4	刘德华收	华北科技学院218	383742	123455324

商品：

商品id	商品名称	商品类别	价格	生产厂家	上次购入时间	详细信息	缩略图路径
1	编程入门	图书	49.99	清华大学出版	2023-01-15	适合初学者的编程教材	images/book_thumb1.j
2	iPhone 15	手机	799.99	Apple	2023-02-10	最新款苹果手机	images/phone_thumb1
3	Canon EOS	相机	2499.99	佳能	2023-03-05	专业级全画幅相机	images/camera_thumb
4	Dell XPS 13	笔记本	199.99	戴尔	2023-04-01	轻薄便携笔记本	images/laptop_thumb
5	数据库原理与	图书	29.9	北京大学出版	2023-07-13	一本精通mysql数据库	images/book/mysql
6	java基础	图书	39.9	清华大学出版	2024-06-06	学Java，这一本就够	images/book/java

订单表：

注：一个购买人可能有多个订单在进行（购买人 3），订单也指向相同或不同的地址

开始事务 文本 筛选 排序 导入 导出

订单编号	提交时间	当前状态	购买人id	地址id
1	2024-06-05	未付款	1	1
2	2024-06-05	已付款	2	2
3	2024-06-06	未付款	3	4
4	2024-06-06	未付款	3	3
5	2024-06-06	已付款	4	5

订单商品表：

注：由于一个订单可能会包含多个不同商品，（如前三条记录表示在订单编号 1 中，购买人买了商品 1 三件，商品 2 四件，商品 3 两件），这么设计是为了方便利用存储语句来计算订单总金额，同时方便商家与客户直观看到自己售卖或购买的商品

开始事务 文本 筛选 排序

订单编号	商品id	购买数量
1	1	3
1	2	4
1	3	2
2	2	1
3	4	2
4	2	5
4	1	3
5	3	4
5	6	1

（4）创建数据库对象

（一）触发器

为了更方便地管理商品，跟踪每个商品信息价格等属性的更改，我为商品表设计了三种（插入，修改，删除）触发器，该触发器可以记录商品表的一系列操作，将操作信息存储到商品日志表中

以下是三种触发器的构建语句：

```
create trigger 插入商品触发器
    after insert on 商品 for each row
begin
    insert into 商品日志(id, operation, operate_time, operate_id, operate_params) VALUES
    (null, 'insert', now(), new.商品 id, concat('插入的数据内容为：商品 id=',new.商品 id,',商品名称=',new.商品名称, ', 上次购入时间=', NEW.上次购入时间, ', 价格=', NEW.价格, ', 详细信息=', NEW.详细信息));
end;

create trigger 修改商品触发器
    after update on 商品 for each row
begin
    insert into 商品日志(id, operation, operate_time, operate_id, operate_params) VALUES
    (null, 'update', now(), new.商品 id,
      concat('更新之前的数据：商品 id=',old.商品 id,',商品名称=',old.商品名称, ', 上次购入时间=', old.上次购入时间, ', 价格=', old.价格, ', 详细信息=', old.详细信息,
        ' | 更新之后的数据：商品 id=',new.商品 id,',商品名称=',new.商品名称, ', 上次购入时间=', NEW.上次购入时间, ', 价格=', NEW.价格, ', 详细信息=', NEW.详细信息));
end;

create trigger 删除商品触发器
    after delete on 商品 for each row
begin
    insert into 商品日志(id, operation, operate_time, operate_id, operate_params) VALUES
    (null, 'delete', now(), old.商品 id,
      concat('删除之前的数据：商品 id=',old.商品 id,',商品名称=',old.商品名称, ', 上次购入时间=', old.上次购入时间, ', 价格=', old.价格, ', 详细信息=', old.详细信息));
end;
```

触发器演示：

在我们插入数据时，触发器生效并将一些数据存入日志，以下是在插入商品表插入数据时触发器生成的数据

id	operation	operate_time	operate_id	operate_params
1	insert	2024-06-05 23:42:41	1	插入的数据内容为:商品id=1,商品名称=编程入门,上次购入时间=2023-01-15,价格=49.99,详细信息=适合初学者的
2	insert	2024-06-05 23:42:41	2	插入的数据内容为:商品id=2,商品名称=iPhone 15,上次购入时间=2023-02-10,价格=799.99,详细信息=最新款苹
3	insert	2024-06-05 23:42:41	3	插入的数据内容为:商品id=3,商品名称=Canon EOS R5,上次购入时间=2023-03-05,价格=2499.99,详细信息=专业
4	insert	2024-06-05 23:42:41	4	插入的数据内容为:商品id=4,商品名称=Dell XPS 13,上次购入时间=2023-04-01,价格=1199.99,详细信息=轻薄便
5	insert	2024-06-06 09:23:13	5	插入的数据内容为:商品id=5,商品名称=数据库原理与应用,上次购入时间=2023-07-13,价格=29.90,详细信息=一本
6	insert	2024-06-06 15:25:43	6	插入的数据内容为:商品id=6,商品名称=java基础,上次购入时间=2024-06-06,价格=39.90,详细信息=学Java，这一

接下来我们修改一些商品的价格，可以发现日志中有了新的数据
update 商品 set 价格=34.9 where 商品 id=5；
update 商品 set 价格=49 where 商品 id=6；

接下来是删除触发器，我们将商品 id 为 7 的商品从表中删除

触发器优点：与查看二进制日志中记录的的 SQL 语句来检查数据变化不同，该触发器可以更加方便地展示商品数据的变化，使数据库更加方便维护

（二）存储过程

围绕着订单商品这个关系表，我构建了一个可以直接算出订单总价的存储过程，该存储过程有一个 in 参数和一个 out 参数，在使用时只需要把需要计算的订单编号传入，就可以穿出该订单需要支付的价格，过程中使用了游标
以下是该存储过程的构建语句：

```
1.  CREATE PROCEDURE 订单价格(IN orderID INT, OUT money DECIMAL(10, 2))
2.  BEGIN
3.    DECLARE done INT DEFAULT FALSE;
4.    DECLARE productID INT;
5.    DECLARE quantity INT;
6.    DECLARE price DECIMAL(10, 2);
7.    DECLARE cur CURSOR FOR SELECT 商品 id, 购买数量 FROM 订单商品表 WHERE 订单编号 = orderID;
8.    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
9.
10.   OPEN cur;
11.
12.   SET money = 0;
13.
14.   read_loop: LOOP
15.     FETCH cur INTO productID, quantity;
16.     IF done THEN
17.       LEAVE read_loop;
18.     END IF;
19.
20.     SELECT 价格 INTO price FROM 商品 WHERE 商品 id = productID;
21.     IF price IS NOT NULL THEN
22.       SET money = money + (price * quantity);
23.     END IF;
24.   END LOOP;
```

```
25.
26.     CLOSE cur;
27. END;
```

演示：比如我们要查询订单 1 的总金额

存储过程优点：有时候订单商品表中记录太多，找起来十分复杂，还要去找对应商品表中每个商品的价格，这过程中一单商品价格发生更改又需要重新统计计算，而该存储过程实时通过表中的数据计算总金额，也方便记账

（三）索引

由于 mysql 会对每个表的主键构建主键索引，所以该实验中不考虑主键索引的情况。

我在此围绕商品表为例，根据生活中的实际情况，为增加查询速度建立了三个二级索引。

（1）商品类别索引：

我们在浏览商品时通常是为了买某种物品（比如图书），这时候就要用到分类查询

create index idx_goods_name on 商品(商品类别)

接下来我们查看分类查询时的执行计划

explain select 商品名称 from 商品 where 商品类别='图书'

```
mysql> explain select 商品名称 from 商品 where 商品类别='图书';
+----+-------------+------+------------+------+-----------------------------------+---------------+---------+-------+------+----------+-------+
| id | select_type | table | partitions | type | possible_keys                     | key           | key_len | ref   | rows | filtered | Extra |
+----+-------------+------+------------+------+-----------------------------------+---------------+---------+-------+------+----------+-------+
|  1 | SIMPLE      | 商品  | NULL       | ref  | idx_goods_name,idx_商品_商品类别_价格 | idx_goods_name | 33      | const |    3 |   100.00 | NULL  |
+----+-------------+------+------------+------+-----------------------------------+---------------+---------+-------+------+----------+-------+
1 row in set, 1 warning (0.00 sec)
```

可以发现索引成功生效

（2）商品名称索引

我们有事在购物网站是明确为了一个商品去的（比如 java 基础），这时就需要用到商品名称索引。

create index idx_goods_商品名称 on 商品(商品名称)

执行计划

```
mysql> explain select 商品id from 商品 where 商品名称='java基础';
+----+-------------+------+------------+------+-----------------------------------+-------------------+---------+-------+------+----------+-------------+
| id | select_type | table | partitions | type | possible_keys                     | key               | key_len | ref   | rows | filtered | Extra       |
+----+-------------+------+------------+------+-----------------------------------+-------------------+---------+-------+------+----------+-------------+
|  1 | SIMPLE      | 商品  | NULL       | ref  | idx_goods_商品名称,idx_goods_name1 | idx_goods_商品名称 | 302     | const |    1 |   100.00 | Using index |
+----+-------------+------+------------+------+-----------------------------------+-------------------+---------+-------+------+----------+-------------+
1 row in set, 1 warning (0.00 sec)
```

（3）价格排序索引

在我们浏览商品时，通常会用到价格升序/降序排序，我们先来尝试不建立排序索引的情况

```
mysql> select 商品名称,价格 from 商品 where 商品类别='图书' order by 价格;
+-----------------+-------+
| 商品名称        | 价格  |
+-----------------+-------+
| 数据库原理与应用 | 29.90 |
| 编程入门        | 49.99 |
+-----------------+-------+
2 rows in set (0.00 sec)

mysql> explain select 商品名称,价格 from 商品 where 商品类别='图书' order by 价格;
+----+-------------+------+------------+------+---------------+---------------+---------+-------+------+----------+-----------------------------------------+
| id | select_type | table | partitions | type | possible_keys | key           | key_len | ref   | rows | filtered | Extra                                   |
+----+-------------+------+------------+------+---------------+---------------+---------+-------+------+----------+-----------------------------------------+
|  1 | SIMPLE      | 商品  | NULL       | ref  | idx_goods_name | idx_goods_name | 33      | const |    2 |   100.00 | Using index condition; Using filesort |
+----+-------------+------+------------+------+---------------+---------------+---------+-------+------+----------+-----------------------------------------+
1 row in set, 1 warning (0.00 sec)
```

可以看到在不建立排序索引时数据库也使用了之前建立的商品类别索引，但是 extra 一列中出现了 Using filesort 也就是说明查询使用了文件排序的方法，这样会大大降低查询速度，所以需要建立排序索引

create index idx_商品_商品类别_价格 on 商品(商品类别,价格)

这里价格默认升序排序不做更改，继续查看查询计划

```
mysql> explain select 商品名称,价格 from 商品 where 商品类别='图书' order by 价格;
+----+-------------+------+------------+------+-----------------------------------+-----------------------------------+---------+-------+------+----------+--------------------------+
| id | select_type | table | partitions | type | possible_keys                     | key                               | key_len | ref   | rows | filtered | Extra                    |
+----+-------------+------+------------+------+-----------------------------------+-----------------------------------+---------+-------+------+----------+--------------------------+
|  1 | SIMPLE      | 商品  | NULL       | ref  | idx_goods_name,idx_商品_商品类别_价格 | idx_商品_商品类别_价格             | 33      | const |    2 |   100.00 | Using index condition    |
+----+-------------+------+------------+------+-----------------------------------+-----------------------------------+---------+-------+------+----------+--------------------------+
1 row in set, 1 warning (0.00 sec)
```

索引优点：在商品表中存在大量数据时查询速度会大大降低，这时候就需要借助索引进行 sql 优化，可大大提高查询速度

缺点：由于商品表是需要经常修改添加的，所以索引会减低表的性能，同时也索引要考虑覆盖索引和回表查询的情况，防止索引失效

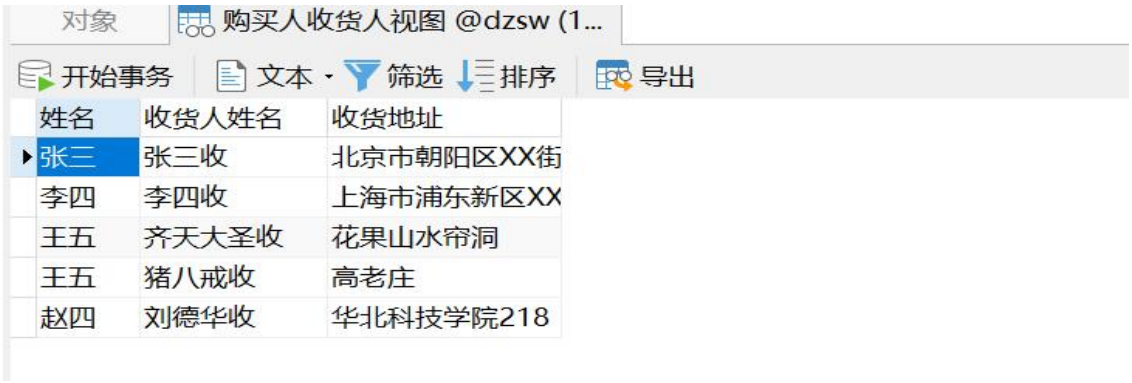
（四）视图

简单创建了两个用于给特定数据库用户查看的视图

create view 商品价格视图 as select 商品 id,商品名称,价格 from 商品

商品id	商品名称	价格
1	编程入门	49.99
2	iPhone 15	799.99
3	Canon EOS R5	2499.99
4	Dell XPS 13	1199.99
5	数据库原理与应用	34.9
6	java基础	50

create view 购买人收货人视图 as select 姓名,收货人姓名,收货地址 from 购买人,收货地址 where 购买人.购买人 id=收货地址.购买人 id



姓名	收货人姓名	收货地址
张三	张三收	北京市朝阳区XX街
李四	李四收	上海市浦东新区XX
王五	齐天大圣收	花果山水帘洞
王五	猪八戒收	高老庄
赵四	刘德华收	华北科技学院218

四、实验结果以及分析

1. 数据库后续运维

该系统完成了电子商务网站的数据库设计，因为这个设计只是针对两个部门的业务所建立的数据库系统，具体功能的实现还需要借助 javaweb 相关知识完善电子商务系统的功能。

若要将该数据库系统投入实际使用，还需要再将数据库进行一系列的优化，比如针对并发事务的问题我们要在数据中引入相应的事务隔离级别，当数据库中数据量非常大时要考虑建立多个服务器进行分库分表，面对大数据量，实施分库分表策略可以显著提高系统处理能力和响应速度，通过哈希、范围等方式将数据分散存储在不同的服务器或表中，减轻单个服务器的压力。还可以配置主从表方便必要时对数据库进行维护，以防主服务器宕机后业务无法进行。

针对数据库系统投入之后的运维还有更多细节需要处理，比如数据库用户的创建与授权，及时对日志进行清理检查，系统业务发生变更时要对数据库进行相应的调整，还可以通过慢查询日志来检查有哪些 sql 操作需要建立索引。

对存储引擎的使用我未在建立系统时做过多考虑，默认为 innodb 存储引擎，因为其支持事务和行锁的特性符合该系统业务需要。

2. 实验中发现的报错

在建完所有表并插入数据后，删除商品表中的一条数据发现删除失败，检查后发现该商品 id 已经存在于订单商品表中，也就是被人购买并记录了，而在建表时订单商品表的商品 id 为外键关联到商品表中，故无法删除。而还未被人购买的商品可以正常删除；

关于索引的建立有很多方面待完善：首先商品表是一个更新频率较高的表，不宜建立过多的索引，会降低 mysql 性能，也占用磁盘空间。再者索引是针对一个或多个字段建立的，在使用时要符合覆盖索引（防止回表查询耗费时间），也要满足最左前缀法则，不然容易导致索引失效，但目前还不知道具体业务中会经常用到哪些查询，所以此问题还不能很好解决。

还有一些细节方面比如外键引用错误，这要求在进行数据操作时，确保所引用的记录在关联表中是存在的。同时在插入数据时，可能会遇到因数据类型不匹配而导致的错误，例如试图将字符串数据插入到数值型字段中。这些都是比较小的错误可以避免。