重庆交通大学信息科学与工程学院
设　计　性　实　验　报　告

专　　业：计算机科学与技术（大数据与人工智能）

班　　级：2021 级曙光班

学　　号：632107060506

姓　　名：李幸洋

课 程 名 称：数据库原理课程实验

实验项目性质：设计性实验

实验所属课程：《数据库原理》

实验室(中心)：信息中心实验室

指 导 教 师 ：王家伟

实验完成时间：2023 年 6 月 18 日

总 成 绩	
教师签名	
日　　期	

评分标准及成绩

实验名称	评分细则	评分
商品销售信息系统设计与实现(100 分)	报告表述的清晰程度和完整性（20 分）	
	概念模型的合理性（20 分）	
	逻辑模型的合理性（10 分）	
	物理模型的合理性(10 分)	
	功能合理性（20 分）	
	功能完善性（10 分）	
	个人体会（10 分）	
	总成绩	

目录

一、功能分析与设计

1.1 用户登录/注册

该电商系统主要用户分两类，一是普通用户，可在系统中进行线下下订单；二是供货商，在系统中为一些商品进行供货。系统提供两类用户登录和注册的功能，两类用户登录后系统给出不同的界面

1.2 商品基本管理

同一种商品能够由多家供货商供货，供货商在其界面选择商品供货后，商品的库存会增加。

1.2.1 商品的基本信息管理

包含商品代码（唯一）、商品名称、规格、型号、计量单位、市场价、销售价、成本价、商品缩略图、商品介绍，库存数量等信息；可对这些信息进行修改、删除。

- 商品品牌管理

  对商品的品牌进行管理，可以在界面上增加品牌、修改品牌信息以及删除品牌等；修改与删除与商品中的品牌对应同步修改和删除，删除品牌后商品、订单也会删除。

- 商品分类管理

  对商品的类别进行管理，可以在界面上增加类别、修改类别信息以及删除类别等；修改与删除与商品中的类别对应同步修改和删除，删除类别后商品、订单也会删除。

- 商品规格管理

  对商品的规格进行管理，可以在界面上增加规格、修改类别规格以及删除规格等；商品与规格之间是多对多的联系，在界面上可以对商品与规格的联系可以进行增加、修改和删除。

1.2.2 商品的查询

根据相关的查询条件完成商品品牌、商品供应商、商品分类和商品基本信息的查询；根据商品品牌和商品分类进行统计。系统提供将此功能面向所有用户，系统根据用户的输入以及选择的查询方式进行查询。

1.3 商品销售

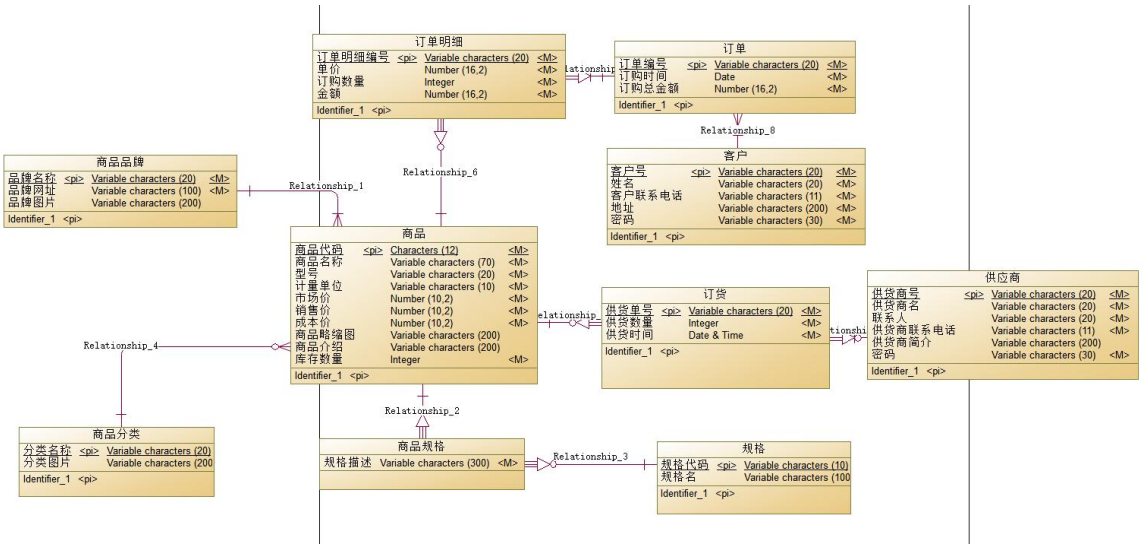
客户根据其自身需求在线上下单，选择商品以及数量等，在其界面进行下单；系统根据商品库存判断是否能下单，下单过程使用事务管理，如果有某件商品库存不足或出现其他错误，则无法下单；下单后生成订单以及订单明细，可供用户查询统计。

1.4 商品供货

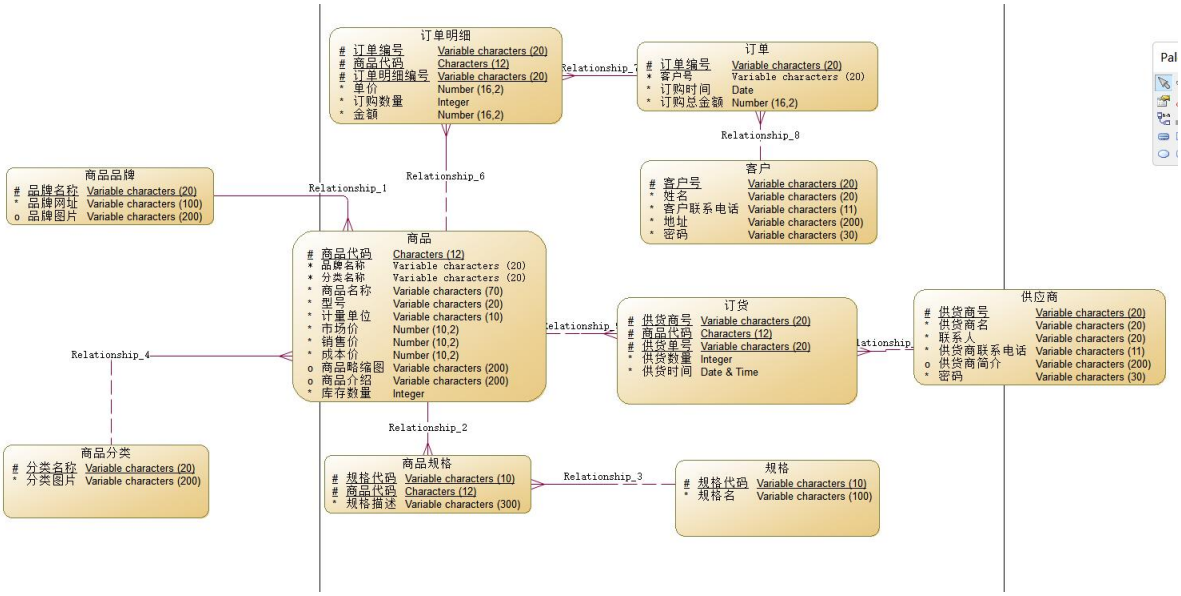
由供货商在其界面选择某一商品，提供相应数量的商品，商品被供货后，库存数量会增加。供货后，系统自动生成供货记录，供货商可以此查询自己的供货记录。

二、模型设计

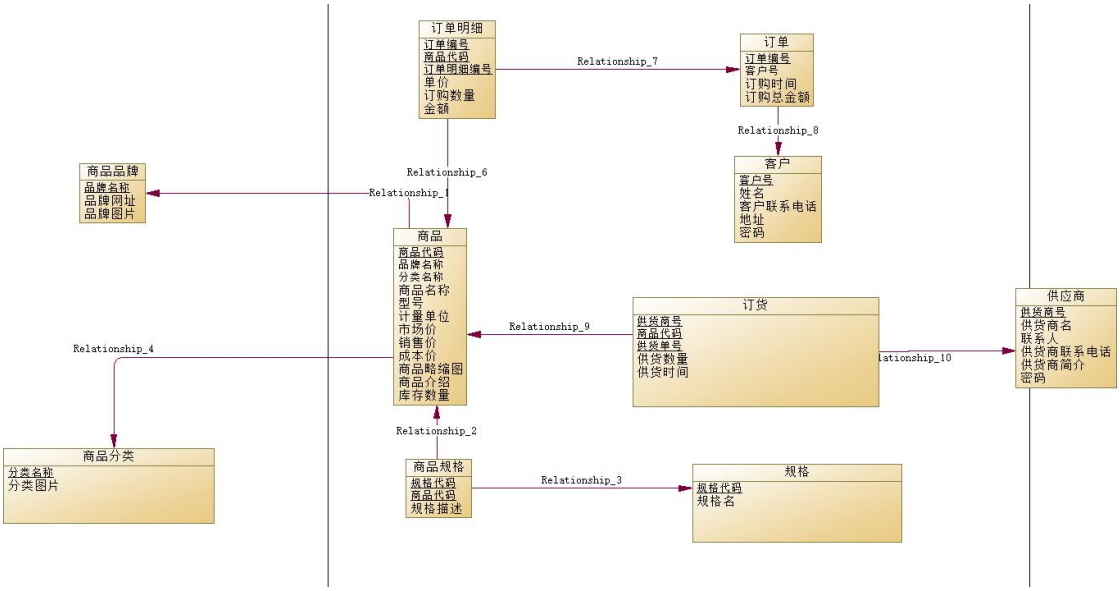
2.1 概念模型



2.2 逻辑模型



2.3 物理模型



三、功能实现

本系统采用的是 Java 和 Mybatis 连接 Mysql 数据库，Mybatis 支持原生 SQL 以及事务管理等。

3.1 用户登录/注册

登录的核心是用户名与密码的比对；注册则对应数据库插入一条数据

客户登录注册核心 SQL：

```
<select id="selectById" resultType="com.sql.cms.pojo.Customer">
    select * from customer where cus_no = #{cus_no}
</select>

<insert id="insertOne" parameterType="com.sql.cms.pojo.Customer">
    insert into customer(cus_no, cus_name, cus_tel, cus_site, cus_pwd)
    values(#{cusNo},#{cusName},#{cusTel},#{cusSite},#{cusPwd})
</insert>
```

供货商登录注册核心 SQL：

```
<select id="selectById" resultType="com.sql.cms.pojo.Supplier">
    select * from supplier where su_no = #{su_no}
</select>

<insert id="insertOne" parameterType="com.sql.cms.pojo.Supplier">
    insert into supplier(su_no, su_name, su_contact, su_con_tel, su_introduce, su_pwd)
    values(#{suNo},#{suName},#{suContact},#{suConTel},#{suIntroduce},#{suPwd})
</insert>
```

3.2 商品基本管理

商品、商品类别、商品分类以及商品规格的查看功能的核心均 SQL 的 select 语句

商品：

```
<select id="selectAll" resultType="com.sql.cms.pojo.Commodity">
    select * from commodity;
</select>
<select id="selectById" resultType="com.sql.cms.pojo.Commodity">
    select * from commodity where co_no = #{co_no}
</select>
```

商品品牌：

```
<select id="selectAll" resultType="com.sql.cms.pojo.Brand">
    select * from brand;
</select>

<select id="selectById" resultType="com.sql.cms.pojo.Brand">
    select * from brand where bra_name = #{bra_name}
</select>
```

商品类别：

```
<select id="selectAll" resultType="com.sql.cms.pojo.Category">
    select * from category;
</select>
<select id="selectById" resultType="com.sql.cms.pojo.Category">
    select * from category where cat_name = #{cat_name}
</select>
```

商品规格：

```
<select id="selectAll" resultType="com.sql.cms.pojo.Specification">
    select * from specification;
</select>
<select id="selectById" resultType="com.sql.cms.pojo.Specification">
    select * from specification where spe_no = #{spe_no}
</select>
```

3.2.1 商品的基本信息管理

商品以及其他信息的管理功能都是基于 SQL 的 update、delete 和 insert 实现的

```
<insert id="insertOne" parameterType="java.util.Map">
    insert into commodity
    values (
        #{coNo},
        #{braName},
        #{catName},
        #{coName},
        #{coType},
        #{coJl},
        #{coMarketProse},
        #{coSalePrise},
        #{coCostPrise},
        #{coPhoto},
        #{coIntroduce},
        #{coNum}
    )
</insert>
```

```
<update id="updateOne" parameterType="java.util.Map">
    update commodity set
        co_no=#{newCoNo},
        bra_name=#{braName},
        cat_name=#{catName},
        co_name=#{coName},
        co_type=#{coType},
        co_jl=#{coJl},
        co_market_prose=#{coMarketProse},
        co_sale_prise=#{coSalePrise},
        co_cost_prise=#{coCostPrise},
        co_photo=#{coPhoto},
        co_introduce=#{coIntroduce},
        co_num=#{coNum}
    where co_no=#{oldCoNo}
</update>
```

```
<delete id="deleteOne" parameterType="java.util.Map">
    delete from commodity where co_no = #{coNo}
</delete>
```

- 商品品牌管理

```
<update id="updateOne" parameterType="java.util.Map">
    update brand set
        bra_name=#{newBraName},
        bra_website=#{braWebsite},
        bra_photo=#{braPhoto}
    where bra_name=#{oldBraName}
</update>
<delete id="deleteOne" parameterType="java.util.Map">
    delete from brand where bra_name=#{braName}
</delete>

<insert id="insertOne" parameterType="com.sql.cms.pojo.Brand">
    insert into brand(bra_name, bra_website, bra_photo)
    values(#{braName},#{braWebsite}, #{braPhoto})
</insert>
```

- 商品分类管理

```
<delete id="deleteOne" parameterType="java.util.Map">
    delete from category where cat_name=#{catName}
</delete>

<insert id="insertOne" parameterType="com.sql.cms.pojo.Category">
    insert into category(cat_name, cat_photo)
    values(#{catName},#{catPhoto})
</insert>

<update id="updateOne" parameterType="java.util.Map">
    update category set
    cat_name=#{newCatName},
    cat_photo=#{catPhoto}
    where cat_name=#{oldCatName}
</update>
```

- 商品规格管理

```
<insert id="insertOne" parameterType="com.sql.cms.pojo.Specification">
    insert into specification(spe_no, spe_name)
    values(#{speNo},#{speName})
</insert>

<update id="updateOne" parameterType="java.util.Map">
    update specification set
    spe_no=#{newSpeNo},
    spe_name=#{speName}
    where spe_no=#{oldSpeNo}
</update>
```

```
<insert id="insertOne" parameterType="java.util.Map">
    insert into com_specification
    values ( #{coNo}, #{speNo}, #{comSpeDescription} )
</insert>

<update id="updateOne" parameterType="java.util.Map">
    update com_specification set
    com_spe_description=#{comSpeDescription}
    where co_no = #{coNo} and spe_no = #{speNo}
</update>
```

3.2.2 商品的查询

商品以及其信息的查询是基于SQL的select语句加上多条件以及表的连接等操作进行实现。

```
<select id="selectByBra" resultType="com.sql.cms.pojo.Commodity">
    select * from commodity where bra_name=#{braName}
</select>
<select id="selectByCat" resultType="com.sql.cms.pojo.Commodity">
    select * from commodity where cat_name=#{catName}
</select>
```

```
<select id="selectByOrder" resultType="java.util.Map">
    select commodity.co_name,co_jl,commodity.bra_name,ord_det_num,ord_det_prise
    from order_detail, commodity
    where commodity.co_no = order_detail.co_no and order_detail.ord_no = #{ordNo}
</select>
```

1.3 商品销售

客户下单功能具体流程是先由用户选择好需要的商品以及数量，点击下单后，系统先生成订单，再生成订单明细，写入订单明细前需要检查商品数量是否足够，不够则抛出异常，触发事务回滚。

修改商品数量触发器：

```
create definer = root@localhost trigger modify_commodity_num
    before insert
    on order_detail
    for each row
begin
        declare co_count int;
        select co_num into co_count from commodity where commodity.co_no=NEW.co_no;
        if(co_count >= NEW.ord_det_num) then
            update commodity set co_num=co_num-NEW.ord_det_num
                where co_no=NEW.co_no;
        else
            SIGNAL SQLSTATE 'HY000' SET MESSAGE_TEXT = co_count;
        end if;

    end;
```

商品订单的事务：

```
1 个用法  lxy *
@Override
@Transactional(propagation = Propagation.REQUIRED)
public Map<String, String> submitOrder(Map<String, String> map) {
    Map<String, String> resp = new HashMap<>();
    try {
        System.out.println(map);
        String cusNo = map.get("cusNo");
        String ordNo = NoUtils.CreateRandomNo( prefix: "order-");
        Date date = new Date(System.currentTimeMillis());
        JSONArray jsonArray = JSONArray.parseArray(map.get("coms"));
        List<OrderDetail> details = new ArrayList<>();

        double sum = 0;
```

```
            jsonObject = (JSONObject) jsonObject;

            String coNo = ((JSONObject) jsonObject)
                    .getString( key: "coNo");
            Double prise = Double.parseDouble(((JSONObject) jsonObject)
                    .getString( key: "coPrise"));
            Integer num = Integer.parseInt(((JSONObject) jsonObject)
                    .getString( key: "coNum"));
            double coSum = num * prise;
            details.add(new OrderDetail(
                    coNo,
                    ordNo,
                    NoUtils.CreateRandomNo( prefix: "ord-del"),
                    prise,
                    num,
                    coSum
            ));
            sum += coSum;
        }

        ordersMapper.insertOne(new Orders(ordNo, cusNo, date, sum));

        for (OrderDetail detail : details) {
            orderDetailMapper.insertOne(detail);
        }

    } catch (Exception e) {
        resp.put("error_info", "订单出错，部分商品库存不足");
        throw new RuntimeException("事务回滚");
    }
```

3.4 商品供货

由供货商在其界面选择某一商品，提供相应数量的商品，商品被供货后，库存数量会增加。供货后，系统自动生成供货记录，供货商可以此查询自己的供货记录。

供货商供货触发器：

```
create definer = root@localhost trigger supply_after
    after insert
    on orders_goods
    for each row
    update commodity set commodity.co_num= commodity.co_num+new.ord_go_num
where commodity.co_no=new.co_no;
```

```
<insert id="insertOne" parameterType="com.sql.cms.pojo.Supplier">
    insert into supplier(su_no, su_name, su_contact, su_con_tel, su_introduce, su_pwd)
    values(#{suNo},#{suName},#{suContact},#{suConTel},#{suIntroduce},#{suPwd})
</insert>
```

四、功能测试

4.1 商品管理测试

4.1.1 商品添加

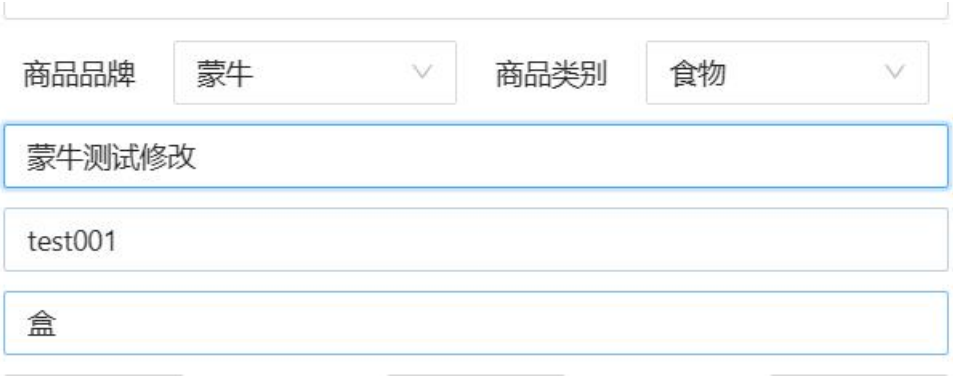
前端界面：

成功添加数据库：

co_no	bra_name	cat_name	co_name	co_type	co_jl	co_market_prose	co_s
nn001	蒙牛	食物	蒙牛牛奶	mn001	盒	3.00	
test001	蒙牛	食物	蒙牛测试	test001	盒	3.00	

4.1.2 商品修改

前端界面：



修改成功数据库：

	co_no	bra_name	cat_name	co_name	co_type	co_jl
1	nn001	蒙牛	食物	蒙牛牛奶	mn001	盒
2	test001	蒙牛	食物	蒙牛测试修改	test001	盒

4.1.3商品删除

	co_no	bra_name	cat_name	co_name	co_type	co_jl
1	nn001	蒙牛	食物	蒙牛牛奶	mn001	盒

4.2 品牌、类别管理测试

4.2.1 添加测试

品牌添加、类别添加：

bra_name	bra_website	
测试	https://pic2.zhimg.co…	ht
蒙牛	https://www.mengniu.c…	ht

	cat_name	cat_photo
1	测试	https://pic2.zhimg.co…
2	食物	https://pic3.zhimg.co…

	co_no	bra_name	cat_name	co_name	co_type
1	nn001	蒙牛	食物	蒙牛牛奶	mn001
2	test001	测试	测试	蒙牛测试	test001

4.2.2 修改测试

品牌、类别修改测试：

	bra_name	bra_website
1	修改测试	https://pic2.zhimg.co…
2	蒙牛	https://www.mengniu.c…

	cat_name	cat_photo
1	修改测试	https://pic2.zhimg.co…
2	食物	https://pic3.zhimg.co…

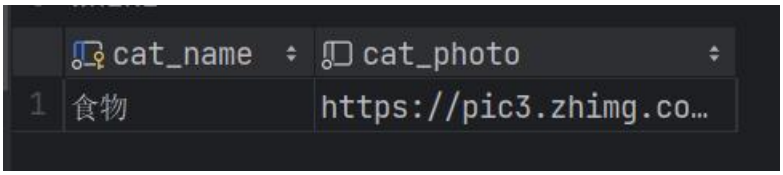
nn001	蒙牛	食物	蒙牛牛奶	mn001
test001	修改测试	修改测试	蒙牛测试	test001

	co_no	bra_name	cat_name	co_name	co_type
1	nn001	蒙牛	食物	蒙牛牛奶	mn001
2	test001	修改测试	修改测试	蒙牛测试	test001

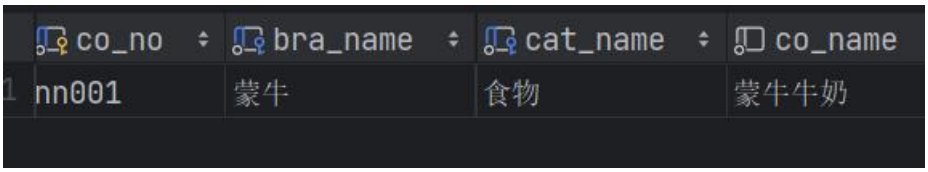
4.2.3 删除测试

品牌、类别删除：

	bra_name	bra_website	bra_photo
1	蒙牛	https://www.mengniu.c…	https://www.



	cat_name	cat_photo
1	食物	https://pic3.zhimg.co…



	co_no	bra_name	cat_name	co_name
1	nn001	蒙牛	食物	蒙牛牛奶

4.3 商品规格管理测试

4.3.1 商品规格添加

规格编号	规格名称
size001	大小
test001	测试规格
weight001	重量

	spe_no	spe_name
1	size001	大小
2	test001	测试规格
3	weight001	重量

nn001	size001	牛奶尺寸
nn001	test001	测试为商品添加规格
nn001	weight001	牛奶重量

co_no	spe_no	com_spe_description
nn001	size001	牛奶尺寸
nn001	test001	测试为商品添加规格
nn001	weight001	牛奶重量

4.3.2 商品规格修改

规格编号	规格名称
001modi	测试规格修改
size001	大小
weight001	重量

co_no	spe_no	com_spe_description
nn001	001modi	测试为商品添加规格
nn001	size001	牛奶尺寸
nn001	weight001	牛奶重量

商品编号	规格编号	规格描述
nn001	001modi	测试为商品添加规格
nn001	size001	牛奶尺寸
nn001	weight001	牛奶重量

4.3.3 商品规格删除

商品编号	规格编号	规格描述
nn001	size001	牛奶尺寸
nn001	weight001	牛奶重量

co_no	spe_no	com_spe_desc
nn001	size001	牛奶尺寸
nn001	weight001	牛奶重量

spe_no	spe_name
size001	大小
weight001	重量

4.4 客户下单测试

下单成功：

co_introduce	co_num
略	88

	ord_no	cus_no	ord_time	ord_sumprise
1	order-4cfd6c885601	123	2023-06-06 20:51:26	187.00
2	order-4df3d6048114	123	2023-06-06 09:12:33	48.00
3	order-ef5714463954	123	2023-06-07 16:44:24	48.00

	ord_det_no	ord_det_prise	ord_det_num	ord_det_sumprise
1	ord-delb90763885709	4.00	18	72.00
4	ord-deld1576f048208	4.00	12	48.00
4	ord-del83035d464053	4.00	12	48.00

事务测试：

下单失败：

在控制台观察到，再开始写入订单明细时发生异常，此时将已经写入的订单回滚，商品数量没有被修改

4.5 供货商供货测试

供货成功

单号	商品编号	数量	时间
SUCOM-10896b830700	nn001	12	2023-06-07 16:50:31

成功写入数据库，并触发了触发器修改了商品数量

co_introduce	co_num
略	100

co_no	su_no	ord_go_no	ord_go_num	ord_go_time
nn001	123	SUCOM-10896b830700	12	2023-06-07 16:50:31
nn001	123	SUCOM-2c-4a168586	10	2023-06-04 16:42:02
nn001	123	SUCOM-7df26a168586	66	2023-06-04 16:38:53
nn001	123	SUCOM-84495e168586	66	2023-06-04 16:14:02
nn001	123	SUCOM-c12d38168586	8	2023-06-04 16:40:42

五、实验心得

这次实验主要使用了数据库技术、Java 以及前端框架成功地实现了这个商品供销管理系统，其中最为核心的内容就是数据库技术。通过建立 E-R 模型，架构了整个整体的数据骨架，数据库技术为整个系统起到了绝对的支撑作用。

对比学习《数据库原理》、使用数据库技术之前所完成的系统，这次实验所完成的系统有了明显的提升；第一，通过数据库原理以及建立 E-R 模型的工作，让整个系统的数据存储以及读取有了清晰流程，数据的完整型、独立性等得到了质的飞跃。第二，数据库技术的引入提高了系统的安全性、减少了业务上逻辑代码的编写。同样对比没有引入数据库技术的系统，单表的操作会让业务代码变得复杂，极有可能出现数据的漏写或重写，而数据库技术则可以让这部分工作得以节省。最后数据库技术也提高了系统的可扩展性，对比文件存储的数据和没有完全使用数据库技术的系统，可扩展性、数据独立性太差，没有可复用性的架构会让业务拓展时变得复杂、难以下手。

对于数据库原理的学习以及这次实验的过程来说，无疑是收获巨大的。数据库在现代的软件系统来说是一个地基式的存在，没有良好的模型以及数据库技术的系统是很难拥有强健性的。此次实验，完成了一个比较复杂的商品供销存储系统，很大程度的让我了解和熟悉了数据模型和数据库技术，可以说将我的系统开发能力提高到了另一层面。最后，数据库的技术远不止如此，需要我学习的地方还有很多