

**《数据库课程实践》课程报告**



**京东数据库的设计与实现**

**学 号 3019244 3019244233 3019244**

**姓 名 李润泽 王红阳 杨琨**

**学 院 智能与计算学部**

**专 业 计算机科学与技术**

**年 级 大三**

# 一、需求分析

TODO：引入语：大数据时代，电子商务已经走入我们生活的每个角落，深入了解电子商务平台是如何运转是十分必要的.....................

因此，本次数据库实践，我们小组以京东购物网页的后台数据库作为数据库设计的参考，以深刻学习构建工业级别数据库的内部流程及各种内在规律。

# 二、总体设计

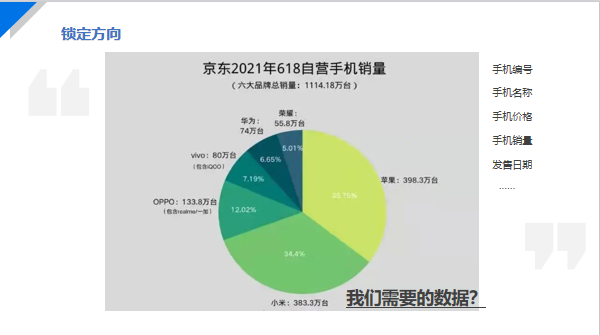
## 1.设计过程

电子商务平台数据具有海量数据，要想洞悉电子购物平台的内在运行规律，势必要对数据库进行重构化简。

然而，哪些数据才是交互过程中实际重要的？

本小组，对京东购物平台进行了深入调研，深入洞察真正重要的数据。

下面以最常用的手机数据为模板来分析我们构建数据库的过程。



首先，我们选择了一个每个人都离不开的商品：手机，以此为出发点，探寻构建购物平台所需的数据库的定位



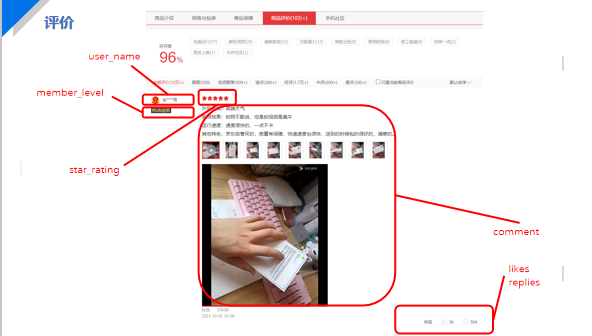
根据商品主页，我们确定出一个商品应该有的数据：商品名称，商品类型，价格，商品描述以及商品ID（以方便对商品进行唯一标识）等等



选好商品后，下一步就是下单了，在这个过程中，需要构建订单的ID标识，下单时间，订单状态，以及购买者的地址，购买者的ID标识，昵称等。



随后，商品开始进行配送，在这个过程中，对于购买者地址的确切描述是十分重要的，它关系着后续的配送能否高效进行，可以构建 一个table，名为address，里面存储用户的ID，用户名，国家，省份，城市，区县，详细地址描述，以细致描述购买者的地址



在商品到货后，用户还可对商品进行评价，对其所购买的商品进行打分，其余用户可对其发布的评价进行点赞，评论等

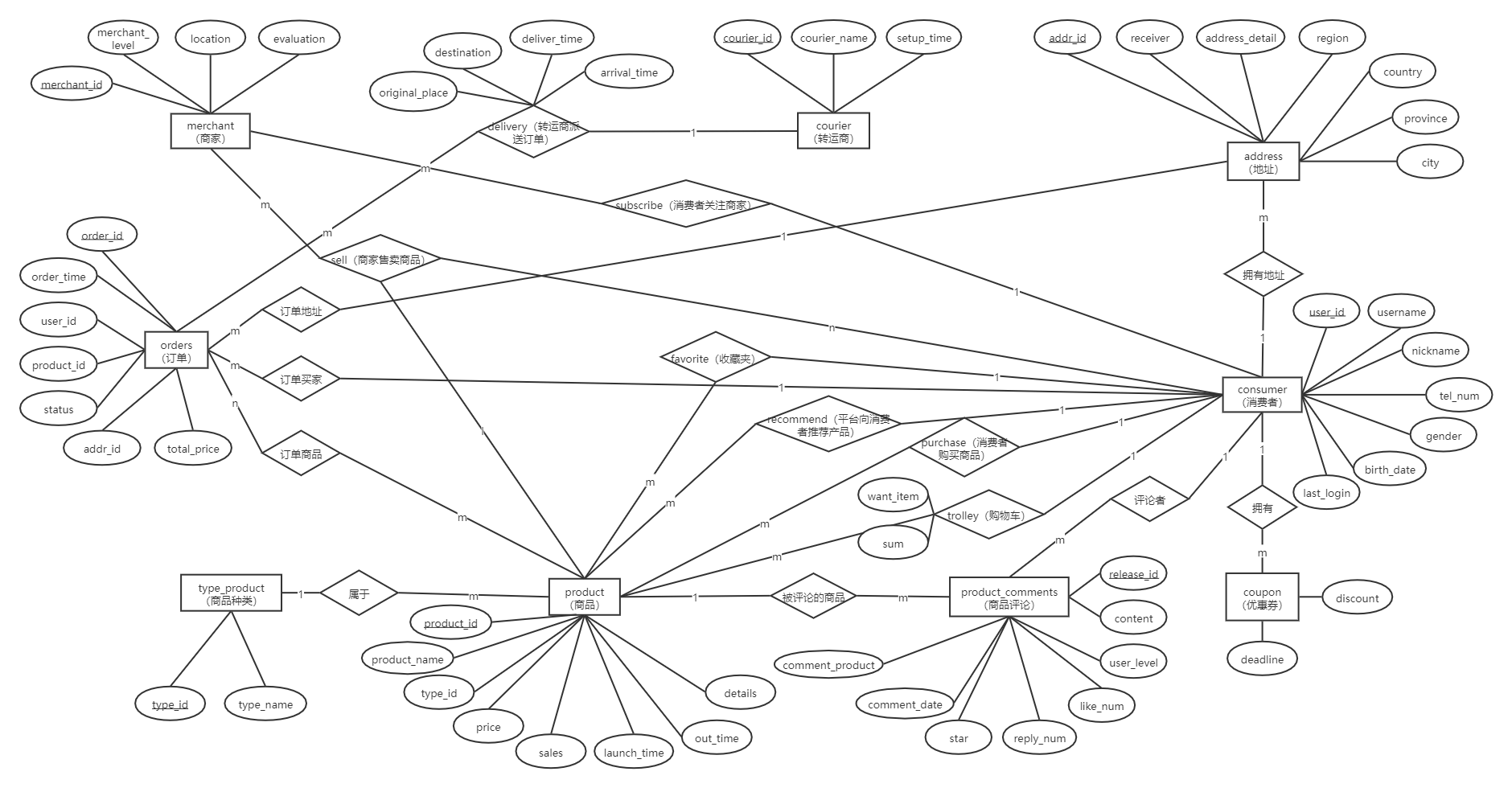
以上只是最基本的功能，实际上京东购物平台的界面所包含的内容远远多余这些：

经过仔细查看交互界面以及京东后端api，我们还对数据库添加了以下实体：消费者对店铺的订阅、用户购物车、电商平台最擅长的优惠券、必不可少的快递公司、转运商等近20个实体

## 2.实现方式

* 基于PgSQL平台实现了所有的数据库代码
  + 建表
  + 创建触发器
  + 插入数据
  + 更新数据
  + ...........
* 通过Python Tkinter 进行简单的交互展示
  + 有登录/注册界面，用于登录账号
  + 用户可以选购商品,
  + 用户可以添加地址
  + 用户可以添加评论
  + 用户可以查看评论
  + ..............

## 3.E-R 图

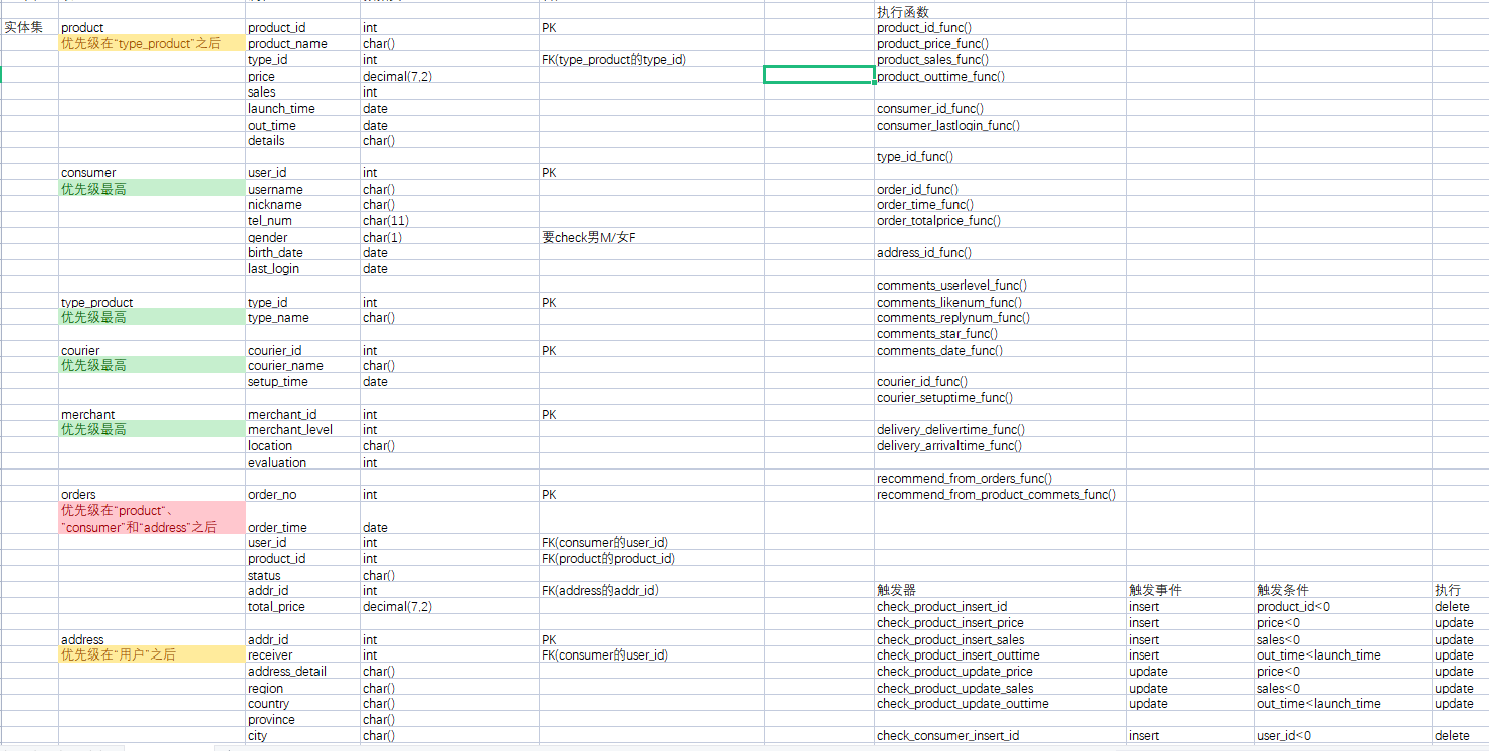


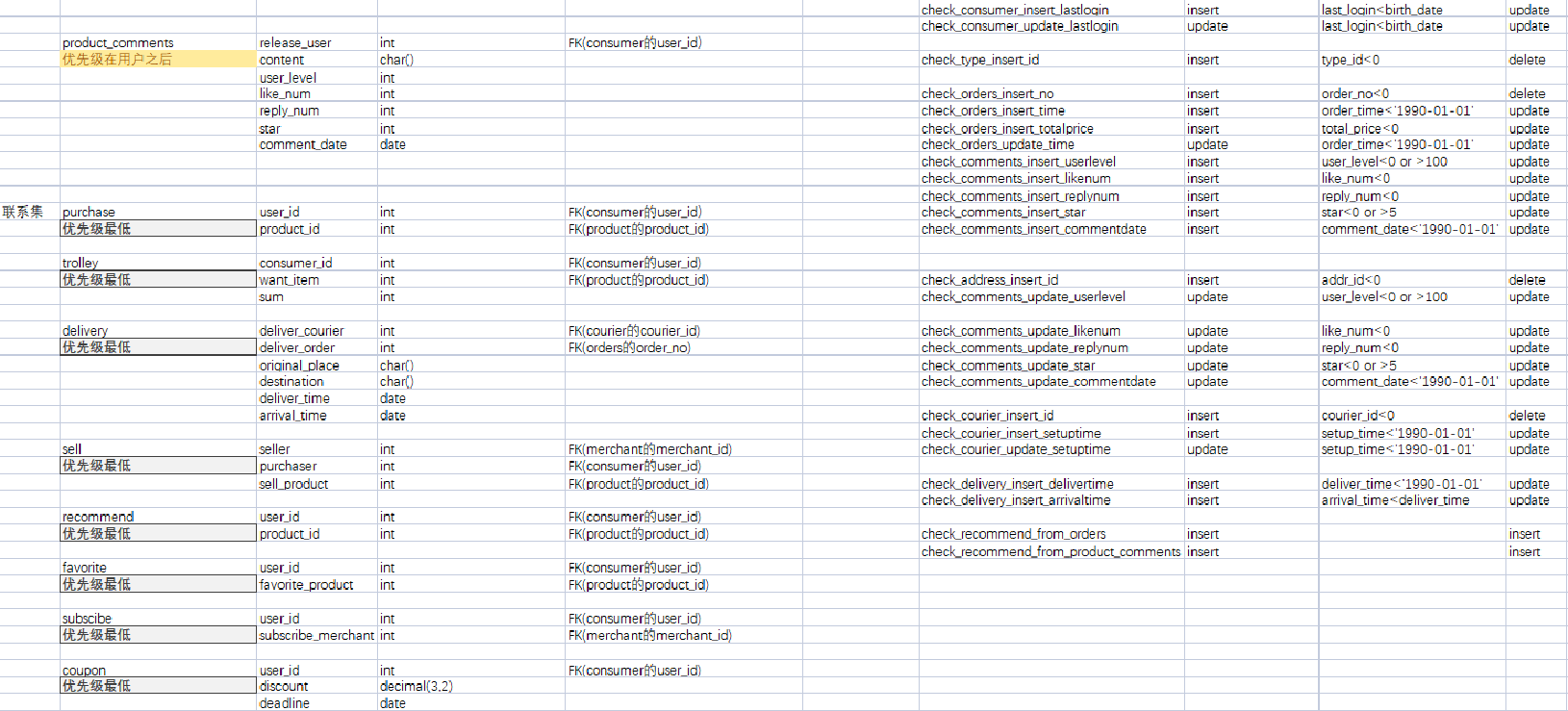
注：在这里可能看不清，详见压缩包里的ER图.png

TODO：

实现了包括商品实体、商品类型、订单、用户实体、配送地址、商品售后、商品评价等多个表，并根据实际生活中对于购物平台的真实需求，创建了一些触发器，约束等数据库应用组件，并使用了基于python tkinter的前端界面用于展示，成功地复现了一个简易的数据购物平台

## 4.需求表格





## 5.table

### 商品类型：type\_product

#### 属性：

* 类型编号：type\_id
* 类型名称：type\_name

#### 约束：

* 主键：type\_id

### 商品: product

#### 属性：

* 商品编号：product\_id
* 商品名称：product\_name
* 商品类型：type\_id
* 价格：price
* 销量：sales
* 上架时间：launch\_time
* 下架时间：off\_time
* 商品描述：details
* 商店名称：shop\_name

#### 约束：

* 主键：商品编号product\_id
* 商品名不能为空
* 必须在table type\_product存在该type\_id

### 用户:consumer

#### 属性：

* 用户账号：user\_id
* 用户密码：passwd
* 用户名：username
* 昵称：nickname
* 手机号：tel\_num
* 性别：gender
* 生日：birth\_date
* 上次登录时间：last\_login

#### 约束：

* 主键：用户账号user\_id
* 性别只能为'M' 或 'F'
* 密码不能为空
* 用户名不能为空
* 电话号不能为空

### 配送地址:address

#### 属性：

* 地址编号：addr\_id
* 接收者：receiver
* 详细地点：address\_detail
* 地区名：region
* 国家：country
* 省份：province
* 城市：city

#### 约束：

* 主键：地址编号addr\_id
* 必须在table consumer的user\_id存在该receiver

### 订单: orders

#### 属性：

* 订单号：order\_no
* 下单时间：order\_time
* 用户编号：user\_id
* 商品编号：product\_id
* 订单状态：status
* 订单地址编号：addr\_id

#### 约束：

* 主键：订单号order\_no
* 必须在table consumer的user\_id存在本table中的user\_id
* 必须在table product的product\_id存在本table中的product\_id
* 必须在table address的addr\_id存在本table中的addr\_id

### 商品评价: product\_comments

#### 属性：

* 用户编号：user\_id
* 评论内容：contents
* 会员级别：user\_level
* 点赞数：likes\_num
* 回复数：reply\_num
* 评价星级：star
* 评论时间：comment\_time

#### 约束：

* 必须在table product\_comments 的user\_id存在本table中的user\_id

### 购买记录: purchase

#### 属性：

* 用户编号：user\_id
* 商品编号：product\_id

#### 约束：

* 必须在table consumer的user\_id存在本table中的user\_id
* 必须在table product的product\_id存在本table中的product\_id

### ：couier

#### 属性：

* : courier\_id
* : courier\_name
* : setup\_name

#### 约束：

* 主键：courier\_id

### :merchant

#### 属性:

* merchant\_id
* merchant\_level
* location
* evaluation

#### 约束：

TODO：有的表没写完

## 触发器

TODO：有些触发器的意思不太懂，没办法写

# 三、实验总结

TODO

通过这次实验，我们深刻了解了一个电子商务平台在初始化，运行，后台调试过程中，对于数据库的操作过程及原理，对构建大型复杂工业级数据库有了充足经验，并建立起书写大型管理系统的信心

# 四、实验代码

注：由于前端代码过于零散，在这里仅展示核心的PgSQL代码

----------------------------------------------------  
-- 初始化，先把表情况  
drop table if exists sell,favorite,subscribe,coupon,trolley,merchant;  
drop table if exists recommend,type\_product, product,consumer,address,orders,product\_comments,purchase,courier,delivery;  
-- drop SEQUENCE if exists orders\_id\_seq;  
----------------------------------------------------  
  
----------------------------------------------------  
-- 建表  
-- 创建商品类型表  
CREATE TABLE IF NOT EXISTS type\_product(  
 type\_id BIGINT,  
 type\_name VARCHAR(100),  
 PRIMARY KEY ( type\_id )  
);  
  
-- 创建商品表  
CREATE TABLE IF NOT EXISTS product(  
 product\_id BIGINT ,  
 product\_name VARCHAR(100) NOT NULL,  
 type\_id int,  
 price int,  
 sales int,  
 launch\_time DATE,  
 out\_time DATE,  
 details varchar(100),  
 shop\_name varchar(100),  
 PRIMARY KEY (product\_id ),  
 CONSTRAINT fk\_pro\_type FOREIGN KEY(type\_id) REFERENCES type\_product(type\_id)  
);  
  
-- 创建顾客表  
CREATE TABLE IF NOT EXISTS consumer (  
 user\_id BIGINT,  
 passwd varchar(20) NOT NULL,   
 username VARCHAR(100) NOT NULL,  
 nickname VARCHAR(100) NOT NULL,  
 tel\_num VARCHAR(11) NOT NULL,  
 gender CHAR(1),  
 birth\_date DATE,  
 last\_login DATE,  
 PRIMARY KEY ( user\_id ),  
 CONSTRAINT user\_sex CHECK (  
 gender = 'M' OR gender = 'F'  
 )  
) ;  
  
-- 创建转运商表  
CREATE TABLE IF NOT EXISTS courier (  
 courier\_id BIGINT,  
 courier\_name VARCHAR(100) NOT NULL,  
 setup\_date DATE,  
 PRIMARY KEY ( courier\_id )  
);  
  
-- 创建卖家店铺表  
CREATE TABLE IF NOT EXISTS merchant (  
 merchant\_id BIGINT,  
 merchant\_level BIGINT,  
 location VARCHAR(100),  
 evaluation BIGINT,  
 PRIMARY KEY ( merchant\_id )  
);  
  
-- 创建地址表  
CREATE TABLE IF NOT EXISTS address (  
 addr\_id serial PRIMARY KEY,  
 receiver INT REFERENCES consumer(user\_id),  
 address\_detail VARCHAR(100),  
 region VARCHAR(20),  
 country VARCHAR(20),  
 province VARCHAR(20),  
 city VARCHAR(20)  
) ;  
  
-- 创建订单表  
CREATE TABLE IF NOT EXISTS orders (  
 order\_no serial,  
 order\_time DATE,  
 user\_id INT REFERENCES consumer(user\_id),  
 product\_id INT REFERENCES product(product\_id),  
 status VARCHAR(10),  
 addr\_id INT REFERENCES address(addr\_id),  
 total\_price DECIMAL(7,2),  
 PRIMARY KEY(order\_no)  
) ;  
  
-- --增加自增序列  
-- CREATE SEQUENCE orders\_id\_seq   
-- INCREMENT 1   
-- START 1   
-- NO MINVALUE   
-- NO MAXVALUE   
-- CACHE 2;  
-- --增加键id  
-- alter table orders add column order\_no int;  
-- --修改键id为自增序列  
-- alter table orders alter column order\_no set default nextval('orders\_id\_seq');  
  
  
-- 创建商品评论表  
CREATE TABLE IF NOT EXISTS product\_comments (  
 release\_user BIGINT REFERENCES consumer(user\_id),  
 contents VARCHAR(100),  
 user\_level INT,  
 like\_num INT,  
 reply\_num INT,  
 star INT,  
 comment\_date DATE,  
 product\_id BIGINT  
) ;  
  
-- 创建购买记录表  
CREATE TABLE IF NOT EXISTS purchase (  
 user\_id BIGINT REFERENCES consumer(user\_id),  
 product\_id BIGINT REFERENCES product(product\_id)  
) ;  
  
-- 创建购物车记录表  
CREATE TABLE IF NOT EXISTS trolley (  
 consumer\_id BIGINT REFERENCES consumer(user\_id),  
 want\_item BIGINT REFERENCES product(product\_id),  
 sum BIGINT  
) ;  
  
-- 创建发送货物记录表  
CREATE TABLE IF NOT EXISTS delivery (  
 deliver\_courier BIGINT REFERENCES courier(courier\_id),  
 deliver\_order BIGINT REFERENCES orders(order\_no),  
 original\_place VARCHAR(100) NOT NULL,  
 destination VARCHAR(100) NOT NULL,  
 deliver\_time DATE,  
 arrival\_time DATE  
) ;  
  
-- 创建售卖商品列表  
CREATE TABLE IF NOT EXISTS sell (  
 seller BIGINT REFERENCES merchant(merchant\_id),  
 purchaser BIGINT REFERENCES consumer(user\_id),  
 sell\_product BIGINT REFERENCES product(product\_id)  
) ;  
  
-- 创建推荐表  
CREATE TABLE IF NOT EXISTS recommend (  
 user\_id BIGINT REFERENCES consumer(user\_id),  
 product\_id BIGINT REFERENCES product(product\_id)  
) ;  
  
-- 创建收藏夹列表  
CREATE TABLE IF NOT EXISTS favorite (  
 user\_id BIGINT REFERENCES consumer(user\_id),  
 favorite\_product BIGINT REFERENCES product(product\_id)  
) ;  
  
-- 创建订阅列表  
CREATE TABLE IF NOT EXISTS subscribe (  
 user\_id BIGINT REFERENCES consumer(user\_id),  
 subscribe\_merchant BIGINT REFERENCES merchant(merchant\_id)  
) ;  
  
-- 创建优惠券列表  
CREATE TABLE IF NOT EXISTS coupon (  
 user\_id BIGINT REFERENCES consumer(user\_id),  
 discount DECIMAL(3,2),  
 deadline DATE  
) ;  
----------------------------------------------------  
  
  
----------------------------------------------------  
-- 创建执行函数  
  
-- 创建商品product\_id执行函数  
CREATE OR REPLACE FUNCTION product\_id\_func()  
returns trigger  
language plpgsql  
AS $$  
BEGIN  
 delete   
 from product  
 where (product\_id < 0);  
 return new;  
END;  
$$;  
  
-- 创建商品price执行函数  
CREATE OR REPLACE FUNCTION product\_price\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 update product  
 set price = 99999  
 where price < 0;  
 return new;  
END;  
$$;  
  
-- 创建商品sales执行函数  
CREATE OR REPLACE FUNCTION product\_sales\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 update product  
 set sales = 0  
 where sales < 0;  
 return new;  
END;  
$$;  
  
-- 创建商品out\_time执行函数  
CREATE OR REPLACE FUNCTION product\_outtime\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 update product  
 set out\_time = launch\_time  
 where out\_time < launch\_time;  
 return new;  
END;  
$$;  
  
-- 创建消费者id执行函数  
CREATE OR REPLACE FUNCTION consumer\_id\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 delete  
 from consumer  
 where user\_id < 0;  
 return new;  
END;  
$$;  
  
-- 创建消费者last\_login执行函数  
CREATE OR REPLACE FUNCTION consumer\_lastlogin\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 update consumer  
 set last\_login = birth\_date  
 where last\_login < birth\_date;  
 return new;  
END;  
$$;  
  
-- 创建商品类型id执行函数  
CREATE OR REPLACE FUNCTION type\_id\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 delete  
 from type\_product  
 where type\_id < 0;  
 return new;  
END;  
$$;  
  
-- 创建订单id执行函数  
CREATE OR REPLACE FUNCTION order\_id\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 delete  
 from orders  
 where order\_no < 0;  
 return new;  
END;  
$$;  
  
-- 创建订单order\_time执行函数  
CREATE OR REPLACE FUNCTION order\_time\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 update orders  
 set order\_time = '1990-01-01'  
 where order\_time < '1990-01-01';  
 return new;  
END;  
$$;  
  
-- 创建订单total\_price执行函数  
CREATE OR REPLACE FUNCTION order\_totalprice\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 update orders  
 set total\_price = 0  
 where total\_price < 0;  
 return new;  
END;  
$$;  
  
-- 创建地址id执行函数  
CREATE OR REPLACE FUNCTION address\_id\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
BEGIN  
 delete  
 from address  
 where addr\_id < 0;  
 return new;  
END;  
$$;  
  
-- 创建商品评论user\_level执行函数  
CREATE OR REPLACE FUNCTION comments\_userlevel\_func()  
returns trigger   
language 'plpgsql'  
AS $$  
BEGIN  
 update product\_comments  
 set user\_level = 0  
 where user\_level < 0 or user\_level > 100;  
 return new;  
END;  
$$;  
  
-- 创建商品评论like\_num执行函数  
CREATE OR REPLACE FUNCTION comments\_likenum\_func()  
returns trigger   
language 'plpgsql'  
AS $$  
BEGIN  
 update product\_comments  
 set like\_num = 0  
 where like\_num < 0;  
 return new;  
END;  
$$;  
  
-- 创建商品评论reply\_num执行函数  
CREATE OR REPLACE FUNCTION comments\_replynum\_func()  
returns trigger   
language 'plpgsql'  
AS $$  
BEGIN  
 update product\_comments  
 set reply\_num = 0  
 where reply\_num < 0;  
 return new;  
END;  
$$;  
  
-- 创建商品评论star执行函数  
CREATE OR REPLACE FUNCTION comments\_star\_func()  
returns trigger   
language 'plpgsql'  
AS $$  
BEGIN  
 update product\_comments  
 set star = 0  
 where star < 0 or star > 5;  
 return new;  
END;  
$$;  
  
-- 创建商品评论comment\_date执行函数  
CREATE OR REPLACE FUNCTION comments\_date\_func()  
returns trigger   
language 'plpgsql'  
AS $$  
BEGIN  
 update product\_comments  
 set comment\_date = '1990-01-01'  
 where comment\_date < '1990-01-01';  
 return new;  
END;  
$$;  
  
-- 创建转运商id执行函数  
CREATE OR REPLACE FUNCTION courier\_id\_func()  
returns trigger   
language 'plpgsql'  
AS $$  
BEGIN  
 delete  
 from courier  
 where courier\_id < 0;  
 return new;  
END;  
$$;  
  
-- 创建转运商setup\_time执行函数  
CREATE OR REPLACE FUNCTION courier\_setuptime\_func()  
returns trigger   
language 'plpgsql'  
AS $$  
BEGIN  
 update courier  
 set setup\_date = '1990-01-01'  
 where setup\_date < '1990-01-01';  
 return new;  
END;  
$$;  
  
-- 创建发送货物deliver\_time执行函数  
CREATE OR REPLACE FUNCTION delivery\_delivertime\_func()  
returns trigger   
language 'plpgsql'  
AS $$  
BEGIN  
 update delivery  
 set deliver\_time = '1990-01-01'  
 where deliver\_time < '1990-01-01';  
 return new;  
END;  
$$;  
  
-- 创建发送货物arrival\_time执行函数  
CREATE OR REPLACE FUNCTION delivery\_arrivaltime\_func()  
returns trigger   
language 'plpgsql'  
AS $$  
BEGIN  
 update delivery  
 set arrival\_time = deliver\_time  
 where arrival\_time < deliver\_time;  
 return new;  
END;  
$$;  
  
-- 创建推荐列表recommend\_from\_orders的函数  
CREATE OR REPLACE FUNCTION recomend\_from\_orders\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
begin  
 insert into recommend VALUES(new.user\_id, new.product\_id);  
 return NULL;  
end;  
$$;  
  
-- 创建推荐列表recommend\_from\_product\_comments的函数  
CREATE OR REPLACE FUNCTION recomend\_from\_product\_comments\_func()  
returns trigger  
language 'plpgsql'  
AS $$  
begin  
 if new.star >= 4 then  
 insert into recommend VALUES(new.release\_user, new.product\_id);  
 end if;  
 return NULL;  
end;  
$$;  
----------------------------------------------------  
  
----------------------------------------------------  
-- 创建触发器  
  
-- 创建商品product\_id触发器  
CREATE TRIGGER check\_product\_insert\_id  
after insert on product  
for each row  
execute procedure product\_id\_func();  
  
-- 创建商品price触发器  
CREATE TRIGGER check\_product\_insert\_price  
after insert on product  
for each row  
execute procedure product\_price\_func();  
  
CREATE TRIGGER check\_product\_up\_price  
after update on product  
for each row  
execute procedure product\_price\_func();  
  
-- 创建商品sales触发器  
CREATE TRIGGER check\_product\_insert\_sales  
after insert on product  
for each row  
execute procedure product\_sales\_func();  
  
CREATE TRIGGER check\_product\_update\_sales  
after update on product  
for each row  
execute procedure product\_sales\_func();  
  
-- 创建商品out\_time触发器  
CREATE TRIGGER check\_product\_insert\_outtime  
after insert on product  
for each row  
execute procedure product\_outtime\_func();  
  
CREATE TRIGGER check\_product\_update\_outtime  
after update on product  
for each row  
execute procedure product\_outtime\_func();  
  
-- 创建消费者id触发器  
CREATE TRIGGER check\_consumer\_insert\_id  
after insert on consumer  
for each row  
execute procedure consumer\_id\_func();  
  
-- 创建消费者last\_login触发器  
CREATE TRIGGER check\_consumer\_insert\_lastlogin  
after insert on consumer  
for each row  
execute procedure consumer\_lastlogin\_func();  
  
CREATE TRIGGER check\_consumer\_update\_lastlogin  
after update on consumer  
for each row  
execute procedure consumer\_lastlogin\_func();  
  
-- 创建商品类型id触发器  
CREATE TRIGGER check\_type\_insert\_id  
after insert on type\_product  
for each row  
execute procedure type\_id\_func();  
  
-- 创建订单id触发器  
CREATE TRIGGER check\_orders\_insert\_id  
after insert on orders  
for each row  
execute procedure order\_id\_func();  
  
-- 创建订单order\_time触发器  
CREATE TRIGGER check\_orders\_insert\_time  
after insert on orders  
for each row  
execute procedure order\_time\_func();  
  
CREATE TRIGGER check\_orders\_update\_time  
after update on orders  
for each row  
execute procedure order\_time\_func();  
  
-- 创建订单total\_price触发器  
CREATE TRIGGER check\_orders\_insert\_totalprice  
after insert on orders  
for each row  
execute procedure order\_totalprice\_func();  
  
CREATE TRIGGER check\_orders\_update\_totalprice  
after update on orders  
for each row  
execute procedure order\_totalprice\_func();  
  
-- 创建地址id触发器  
CREATE TRIGGER check\_address\_insert\_id  
after insert on address  
for each row  
execute procedure address\_id\_func();  
  
-- 创建商品评论user\_level触发器  
CREATE TRIGGER check\_comments\_insert\_userlevel  
after insert on product\_comments  
for each row  
execute procedure comments\_userlevel\_func();  
  
CREATE TRIGGER check\_comments\_update\_userlevel  
after update on product\_comments  
for each row  
execute procedure comments\_userlevel\_func();  
  
-- 创建商品评论like\_num触发器  
CREATE TRIGGER check\_comments\_insert\_likenum  
after insert on product\_comments  
for each row  
execute procedure comments\_likenum\_func();  
  
CREATE TRIGGER check\_comments\_update\_likenum  
after update on product\_comments  
for each row  
execute procedure comments\_likenum\_func();  
  
-- 创建商品评论reply\_num触发器  
CREATE TRIGGER check\_comments\_insert\_replynum  
after insert on product\_comments  
for each row  
execute procedure comments\_replynum\_func();  
  
CREATE TRIGGER check\_comments\_update\_replynum  
after update on product\_comments  
for each row  
execute procedure comments\_replynum\_func();  
  
-- 创建商品评论star触发器  
CREATE TRIGGER check\_comments\_insert\_star  
after insert on product\_comments  
for each row  
execute procedure comments\_star\_func();  
  
CREATE TRIGGER check\_comments\_update\_star  
after update on product\_comments  
for each row  
execute procedure comments\_star\_func();  
  
-- 创建商品评论comment\_date触发器  
CREATE TRIGGER check\_comments\_insert\_date  
after insert on product\_comments  
for each row  
execute procedure comments\_date\_func();  
  
CREATE TRIGGER check\_comments\_update\_date  
after update on product\_comments  
for each row  
execute procedure comments\_date\_func();  
  
-- 创建转运商id触发器  
CREATE TRIGGER check\_courier\_insert\_id  
after insert on courier  
for each row  
execute procedure courier\_id\_func();  
  
-- 创建转运商setup\_date触发器  
CREATE TRIGGER check\_courier\_insert\_setuptime  
after insert on courier  
for each row  
execute procedure courier\_setuptime\_func();  
  
CREATE TRIGGER check\_courier\_update\_setuptime  
after update on courier  
for each row  
execute procedure courier\_setuptime\_func();  
  
-- 创建发送货物deliver\_time触发器  
CREATE TRIGGER check\_delivery\_insert\_delivertime  
after insert on delivery  
for each row  
execute procedure delivery\_delivertime\_func();  
  
-- 创建发送货物arrival\_time触发器  
CREATE TRIGGER check\_delivery\_insert\_arrivaltime  
after insert on delivery  
for each row  
execute procedure delivery\_arrivaltime\_func();  
  
-- 创建推荐列表recommend\_from\_orders触发器  
CREATE TRIGGER recomend\_from\_orders  
after insert on orders  
for each row  
execute procedure recomend\_from\_orders\_func();  
  
-- 创建推荐列表recommend\_from\_product\_comments触发器  
CREATE TRIGGER recomend\_from\_product\_comments  
after insert on product\_comments  
for each row  
execute procedure recomend\_from\_product\_comments\_func();  
----------------------------------------------------  
  
----------------------------------------------------  
-- 数据库操作：增删改查  
  
-- 添加商品类型  
insert into type\_product (type\_id, type\_name) VALUES (1,'电子设备');  
insert into type\_product (type\_id, type\_name) VALUES (2,'日用百货');  
insert into type\_product (type\_id, type\_name) VALUES (3,'文具');  
insert into type\_product (type\_id, type\_name) VALUES (4,'书籍');  
  
-- 添加商品数据  
insert into product (product\_id, product\_name, type\_id, price, sales, launch\_time, out\_time,details, shop\_name) VALUES (01,'iPhone 12', 1, 5000.0,500,'2021-10-30','2031-10-30','一部手机','苹果手机店');  
insert into product (product\_id, product\_name, type\_id, price, sales, launch\_time, out\_time,details, shop\_name) VALUES (02,'OPPO reno 6', 1, 6000.0,600,'2021-05-30','2031-06-30','一部手机','OPPO手机店');  
insert into product (product\_id, product\_name, type\_id, price, sales, launch\_time, out\_time,details, shop\_name) VALUES (03,'蓝月亮洗衣液', 2, 30.00,56787,'2013-05-30','2041-06-30','洗衣液','蓝月亮官网');  
insert into product (product\_id, product\_name, type\_id, price, sales, launch\_time, out\_time,details, shop\_name) VALUES (04,'乐扣水杯', 2, 30.0,60858,'2007-07-27','2071-07-12','新款运动塑料水杯学生杯便携随手带杯子两件套','乐扣乐扣京东自营旗舰店');  
insert into product (product\_id, product\_name, type\_id, price, sales, launch\_time, out\_time,details, shop\_name) VALUES (05,'晨光水笔替芯', 3, 16.8,4274,'2004-05-30','2031-04-24','  
晨光(M&G) 中性替芯水笔芯 黑 0.7mm学生办公文具 黑色 20支/盒','晨光文具京东自营');  
insert into product (product\_id, product\_name, type\_id, price, sales, launch\_time, out\_time,details, shop\_name) VALUES (06,'高等数学同济第七版', 1, 68.8,214,'2005-05-30','2022-06-30','高等数学上下册：教材+习题全解指南','高等教育出版社');  
  
-- 创建顾客数据  
insert into consumer (user\_id, passwd, username, nickname, tel\_num, gender, birth\_date, last\_login) VALUES (001,'why','why','giao','15222168550','M','2004-05-30','2021-09-30');  
insert into consumer (user\_id, passwd, username, nickname, tel\_num, gender, birth\_date, last\_login) VALUES (002,'changqingaas','changqingaas','JiaRan','110','F','2010-05-30','2021-09-30');  
  
-- 添加地址数据   
insert into address (receiver, address\_detail, region, country, province, city) VALUES ( 001, '海河路250号', '津南区', '中国','天津市','天津市');  
  
-- 添加订单数据  
insert into orders (order\_time, user\_id, product\_id, status, addr\_id, total\_price) VALUES ('2021-09-18',001,01,'状态：未送达',0001,5000);  
insert into orders (order\_time, user\_id, product\_id, status, addr\_id, total\_price) VALUES ('2000-09-18',002,02,'状态：未送达',0001,6000);  
  
-- 创建商品评论数据  
insert into product\_comments (release\_user, contents, user\_level, like\_num, reply\_num, star, comment\_date) VALUES (001,'good',87,77,2,5,'2021-09-08');  
  
-- 创建购买记录   
insert into purchase(user\_id, product\_id) VALUES (001,01);  
insert into purchase(user\_id, product\_id) VALUES (002,02);  
----------------------------------------------------