天津大學

《数据库课程实践》课程报告

京东数据库的设计与实现

学　　号 3019244266　3019244233　3019244259

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 2019 级

一、需求分析

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。从最初的电话、电子邮件等方式到如今已经发展为通过网络实现商品的交易和结算，使得网络以及数据库的设计变得至关重要。

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

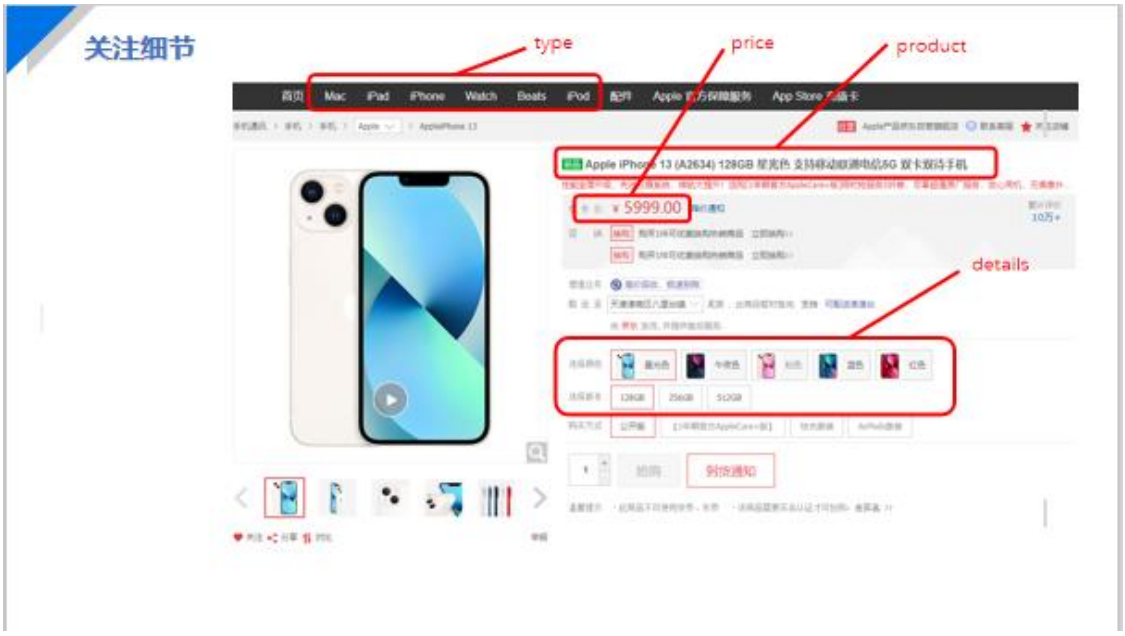
电子商务平台数据具有海量数据，要想洞悉电子购物平台的内在运行规律，势必要对数据库进行重构化简。

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。

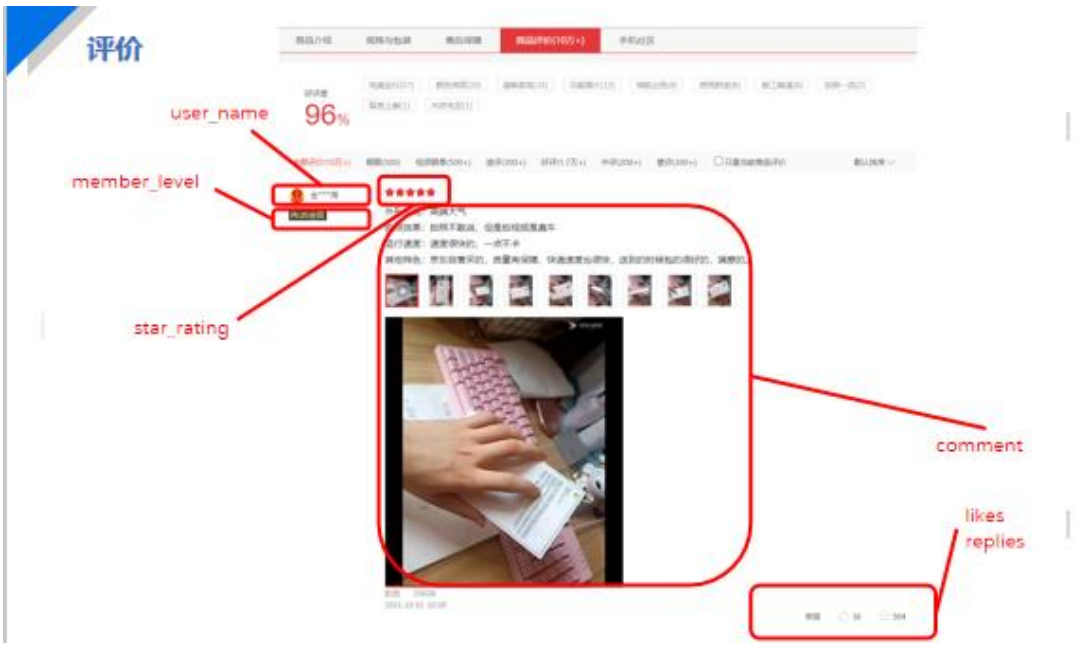
首先，我们选择了一个每个人都离不开的商品：手机，以此为出发点，探寻构建购物平台所需的数据库的定位



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

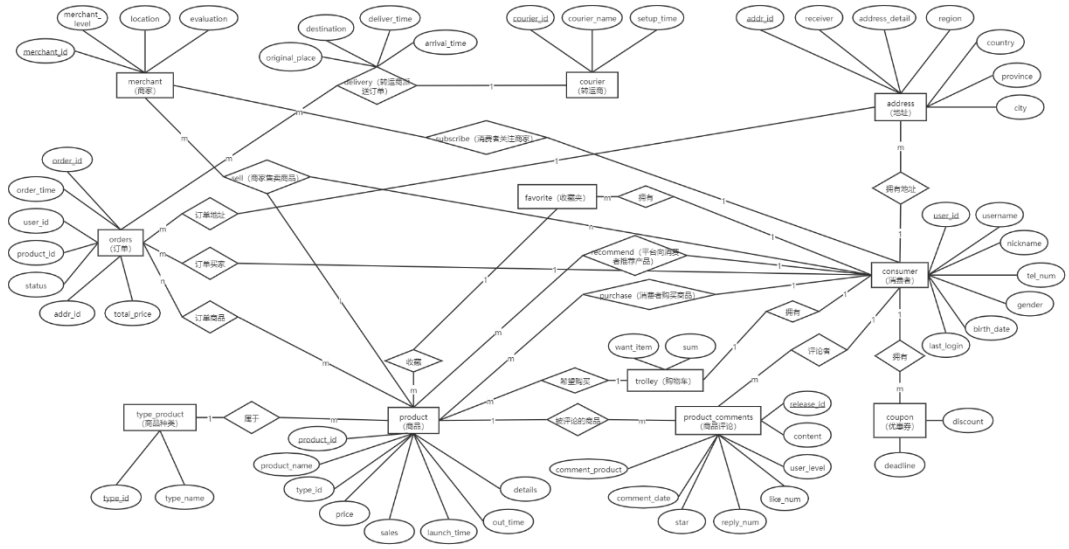
以上只是最基本的功能，实际上京东购物平台的界面所包含的内容远远多余这些：

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 PostgreSQL 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

在 E/R 图中， 我们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

					执行函数				
实体集	product	product_id	int	PK	product_id_func()				
	优先级在"type_product"之后	product_name	char()		product_price_func()				
		type_id	int	FK(type_product的type_id)	product_sales_func()				
		price	decimal(7,2)		product_outtime_func()				
		sales	int						
		launch_time	date		consumer_id_func()				
		out_time	date		consumer_lastlogin_func()				
		details	char()						
					type_id_func()				
	consumer	user_id	int	PK					
	优先级最高	username	char()		order_id_func()				
		nickname	char()		order_time_func()				
		tel_num	char(11)		order_totalprice_func()				
		gender	char(1)	要check男M/女F					
		birth_date	date		address_id_func()				
		last_login	date						
					comments_userlevel_func()				
	type_product	type_id	int	PK	comments_likenum_func()				
	优先级最高	type_name	char()		comments_replynum_func()				
					comments_star_func()				
	courier	courier_id	int	PK	comments_date_func()				
	优先级最高	courier_name	char()						
		setup_time	date		courier_id_func()				
					courier_setuptime_func()				
	merchant	merchant_id	int	PK					
	优先级最高	merchant_level	int		delivery_delivertime_func()				
		location	char()		delivery_arrivaltime_func()				
		evaluation	int						
					recommend_from_orders_func()				
	orders	order_no	int	PK	recommend_from_product_commets_func()				
	优先级在"product"、"consumer"和"address"之后	order_time	date						
		user_id	int	FK(consumer的user_id)					
		product_id	int	FK(product的product_id)					
		status	char()						
		addr_id	int	FK(address的addr_id)	触发器	触发事件	触发条件	执行	
		total_price	decimal(7,2)		check_product_insert_id	insert	product_id<0	delete	
					check_product_insert_price	insert	price<0	update	
	address	addr_id	int	PK	check_product_insert_sales	insert	sales<0	update	
	优先级在"用户"之后	receiver	int	FK(consumer的user_id)	check_product_insert_outtime	insert	out_time<launch_time	update	
		address_detail	char()		check_product_update_price	update	price<0	update	
		region	char()		check_product_update_sales	update	sales<0	update	
		country	char()		check_product_update_outtime	update	out_time<launch_time	update	
		province	char()						
		city	char()		check_consumer_insert_id	insert	user_id<0	delete	

					check_consumer_insert_lastlogin	insert	last_login<birth_date	update
					check_consumer_update_lastlogin	update	last_login<birth_date	update
	product_comments	release_user	int	FK(consumer的user_id)				
	优先级在用户之后	content	char()		check_type_insert_id	insert	type_id<0	delete
		user_level	int					
		like_num	int		check_orders_insert_no	insert	order_no<0	delete
		reply_num	int		check_orders_insert_time	insert	order_time<'1990-01-01'	update
		star	int		check_orders_insert_totalprice	insert	total_price<0	update
		comment_date	date		check_orders_update_time	update	order_time<'1990-01-01'	update
					check_comments_insert_userlevel	insert	user_level<0 or >100	update
					check_comments_insert_likenum	insert	like_num<0	update
					check_comments_insert_replynum	insert	reply_num<0	update
联系集	purchase	user_id	int	FK(consumer的user_id)	check_comments_insert_star	insert	star<0 or >5	update
	优先级最低	product_id	int	FK(product的product_id)	check_comments_insert_commentdate	insert	comment_date<'1990-01-01'	update
	trolley	consumer_id	int	FK(consumer的user_id)				
	优先级最低	want_item	int	FK(product的product_id)	check_address_insert_id	insert	addr_id<0	delete
		sum	int		check_comments_update_userlevel	update	user_level<0 or >100	update
	delivery	deliver_courier	int	FK(courier的courier_id)	check_comments_update_likenum	update	like_num<0	update
	优先级最低	deliver_order	int	FK(orders的order_no)	check_comments_update_replynum	update	reply_num<0	update
		original_place	char()		check_comments_update_star	update	star<0 or >5	update
		destination	char()		check_comments_update_commentdate	update	comment_date<'1990-01-01'	update
		deliver_time	date					
		arrival_time	date		check_courier_insert_id	insert	courier_id<0	delete
					check_courier_insert_setuptime	insert	setup_time<'1990-01-01'	update
	sell	seller	int	FK(merchant的merchant_id)	check_courier_update_setuptime	update	setup_time<'1990-01-01'	update
	优先级最低	purchaser	int	FK(consumer的user_id)				
		sell_product	int	FK(product的product_id)	check_delivery_insert_delivertime	insert	deliver_time<'1990-01-01'	update
					check_delivery_insert_arrivaltime	insert	arrival_time<deliver_time	update
	recommend	user_id	int	FK(consumer的user_id)				
	优先级最低	product_id	int	FK(product的product_id)	check_recommend_from_orders	insert		insert
					check_recommend_from_product_comments	insert		insert
	favorite	user_id	int	FK(consumer的user_id)				
	优先级最低	favorite_product	int	FK(product的product_id)				
	subscibe	user_id	int	FK(consumer的user_id)				
	优先级最低	subscribe_merchant	int	FK(merchant的merchant_id)				
	coupon	user_id	int	FK(consumer的user_id)				
	优先级最低	discount	decimal(3,2)					
		deadline	date					

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

转运商：courier

属性：

- 转运商编号：courier_id
- 转运商名称：courier_name
- 转运商建立时间：setup_date

约束：

- 主键：courier_id

商家（店铺）：merchant

属性：

- 商家编号：merchant_id

- 商家等级：merchant_level
- 商家地址：location
- 商家评分：evaluation

约束：

- 主键：merchant_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

购物车：trolley

属性:

- 用户编号：consumer_id
- 购物车内的商品：want_item
- 购物车内商品总量：sum

约束：

- 必须在 table consumer 的 user_id 存在本 table 中的 consumer_id
- 必须在 table product 的 product_id 存在本 table 中的 want_item

快递公司派送记录：delivery

属性：

- 派送公司：deliver_courier
- 派送订单：deliver_order
- 出发地址：original_place
- 派送目的地：destination
- 出发时间：deliver_time
- 到达时间：arrival_time

约束：

- 必须在 table courier 的 courier_id 存在本 table 中的 deliver_courier
- 必须在 table orders 的 order_no 存在本 table 中的 deliver_order

卖家售卖记录：sell

属性：

- 卖家：seller
- 买家：purchaser
- 售卖商品：sell_product

约束：

- 必须在 table merchant 的 merchant_id 存在本 table 中的 seller
- 必须在 table consumer 的 user_id 存在本 table 中的 purchaser
- 必须在 table product 的 product_id 存在本 table 中的 sell_product

推荐记录：recommend

属性：

- 用户编号：user_id
- 推荐商品：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

收藏夹：favorite

属性：

- 用户编号：user_id
- 收藏夹内商品：favorite_product

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 favorite_product

订阅记录：subscribe

属性：

- 用户编号：user_id
- 订阅商家：subscribe_merchant

约束：

- 必须在 table consumer 的 user_id 存在本 table 中的 user_id
- 必须在 table merchant 的 merchant_id 存在本 table 中的 subscribe_merchant

6. 触发器

触发器	触发事件	触发条件	执行
check_product_insert_id	insert	product_id<0	delete
check_product_insert_price	insert	price<0	update
check_product_insert_sales	insert	sales<0	update
check_product_insert_outtime	insert	out_time<launch_time	update
check_product_update_price	update	price<0	update
check_product_update_sales	update	sales<0	update
check_product_update_outtime	update	out_time<launch_time	update
check_consumer_insert_id	insert	user_id<0	delete
check_consumer_insert_lastlogin	insert	last_login<birth_date	update
check_consumer_update_lastlogin	update	last_login<birth_date	update
check_type_insert_id	insert	type_id<0	delete
check_orders_insert_no	insert	order_no<0	delete
check_orders_insert_time	insert	order_time<'1990-01-01'	update
check_orders_insert_totalprice	insert	total_price<0	update
check_orders_update_time	update	order_time<'1990-01-01'	update
check_comments_insert_userlevel	insert	user_level<0 or >100	update
check_comments_insert_likenum	insert	like_num<0	update
check_comments_insert_replynum	insert	reply_num<0	update
check_comments_insert_star	insert	star<0 or >5	update
check_comments_insert_commentdate	insert	comment_date<'1990-01-01'	update
check_address_insert_id	insert	addr_id<0	delete
check_comments_update_userlevel	update	user_level<0 or >100	update
check_comments_update_likenum	update	like_num<0	update
check_comments_update_replynum	update	reply_num<0	update
check_comments_update_star	update	star<0 or >5	update
check_comments_update_commentdate	update	comment_date<'1990-01-01'	update
check_courier_insert_id	insert	courier_id<0	delete
check_courier_insert_setuptime	insert	setup_time<'1990-01-01'	update
check_courier_update_setuptime	update	setup_time<'1990-01-01'	update
check_delivery_insert_delivertime	insert	deliver_time<'1990-01-01'	update
check_delivery_insert_arrivaltime	insert	arrival_time<deliver_time	update

check_recommend_from_orders	insert		insert
check_recommend_from_product_comments	insert	product_comments.star>=4	insert

三、实验总结

通过这次实验，我们深刻理解了一个电子商务平台在初始化与运行以及在后台调试过程中，对于数据库的操作过程及原理，小组成员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,p
roduct_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),
       PRIMARY KEY (product_id ),
       CONSTRAINT fk_pro_type FOREIGN KEY(type_id) REFERENCES type_product(typ
e_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_t
ime, 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_t
ime, out_time,details, shop_name) VALUES (02,'OPPO reno 6', 1, 6000.0,600,'202
1-05-30','2031-06-30','一部手机','OPPO 手机店');
insert into product (product_id, product_name, type_id, price, sales, launch_t
ime, out_time,details, shop_name) VALUES (03,'蓝月亮洗衣液', 2, 30.00,56787,'2
013-05-30','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_pr
```

```
ice) 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, re
ply_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);
-----------------------------------------------
```