1 实验环境

请填写用到的操作系统和主要开发工具。

Windows 操作系统、MySQL8.0 关系数据库管理系统、数据库设计工具 PowerDesigner16.5。

2 实验过程

2.1 系统功能

*请结合文字、图表等方式清晰描述系统的功能。如有亮点功能请用*标志。*

1、用户注册（管理员、普通用户）；

2、普通用户可在平台发布二手物品（物品类别、物品名称、购买年份、新旧程度、转让价格……）；

3、普通用户可查询浏览二手物品，下订单（模拟网络支付）；

4、管理员浏览物品，下架违规物品；

5、普通用户可对某物品进行收藏、添加购物车；

6、普通用户可对某单交易进行投诉（选择投诉类别，填入投诉原因）；

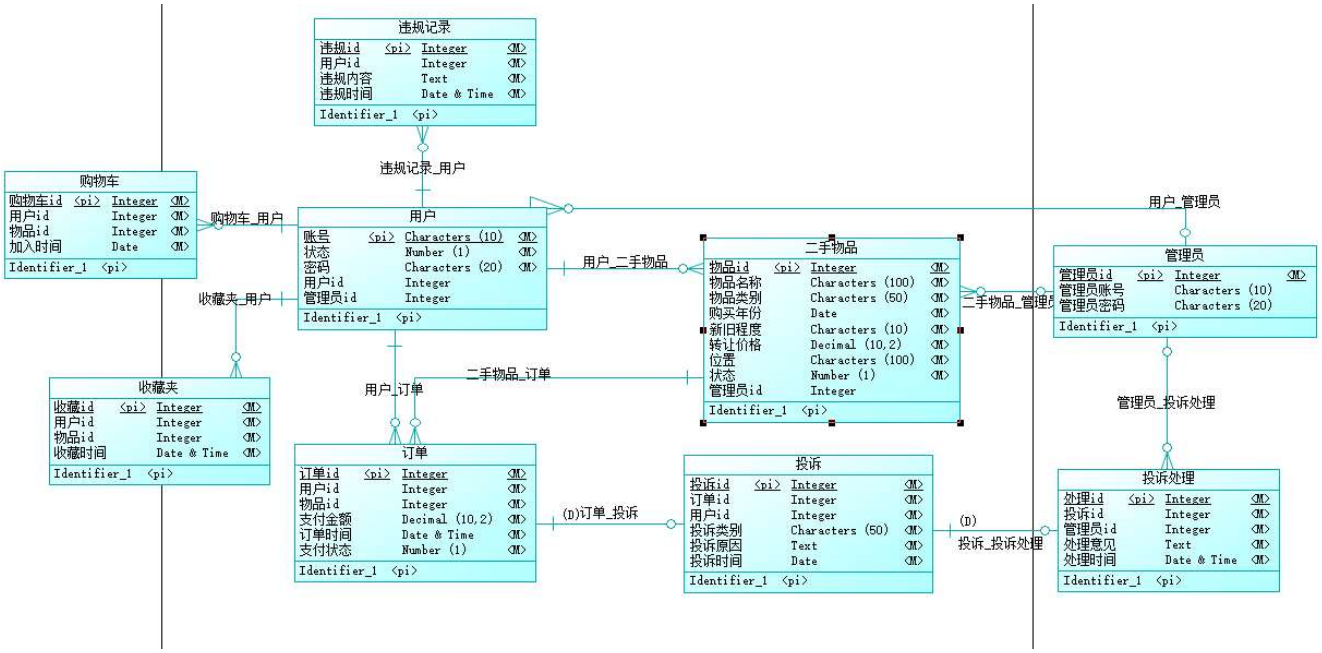
7、管理员可处理投诉（输入处理意见）；

8、对于多次违规的用户管理员可冻结其账号。

2.2 数据库设计

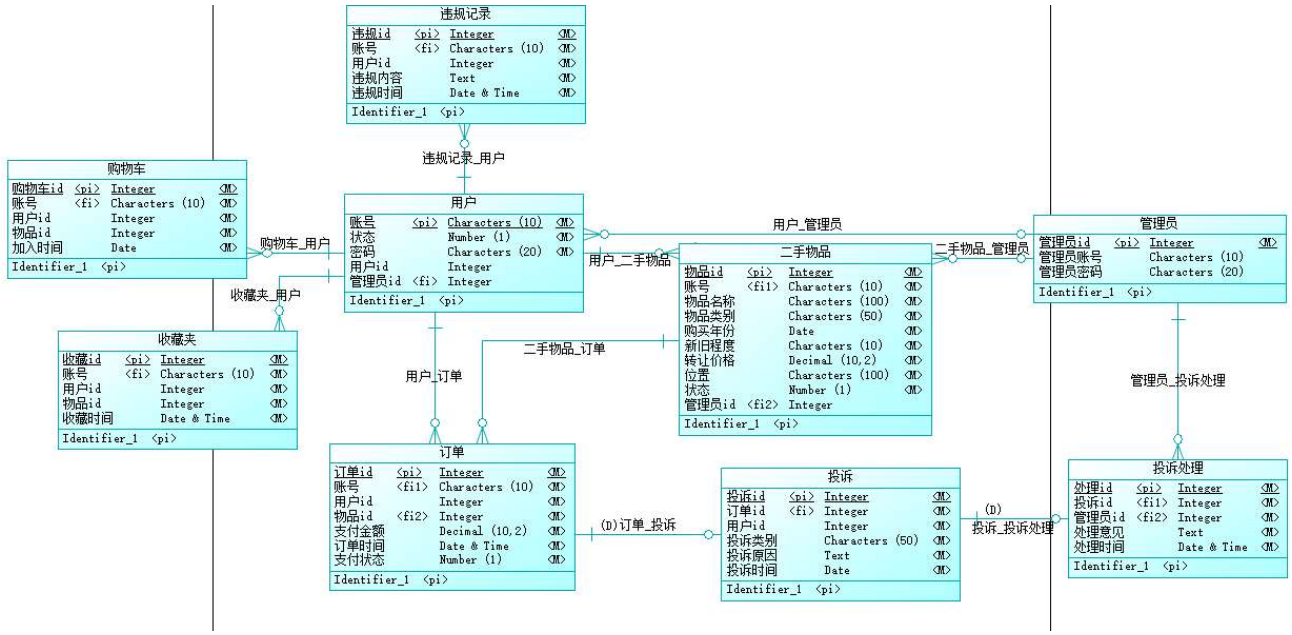
2.1.1 ER 图

要求：截图务必清晰，如果图太大可截图一个总图，然后再分块截图。如果看不清截图会影响成绩。



2.1.2 LDM 图

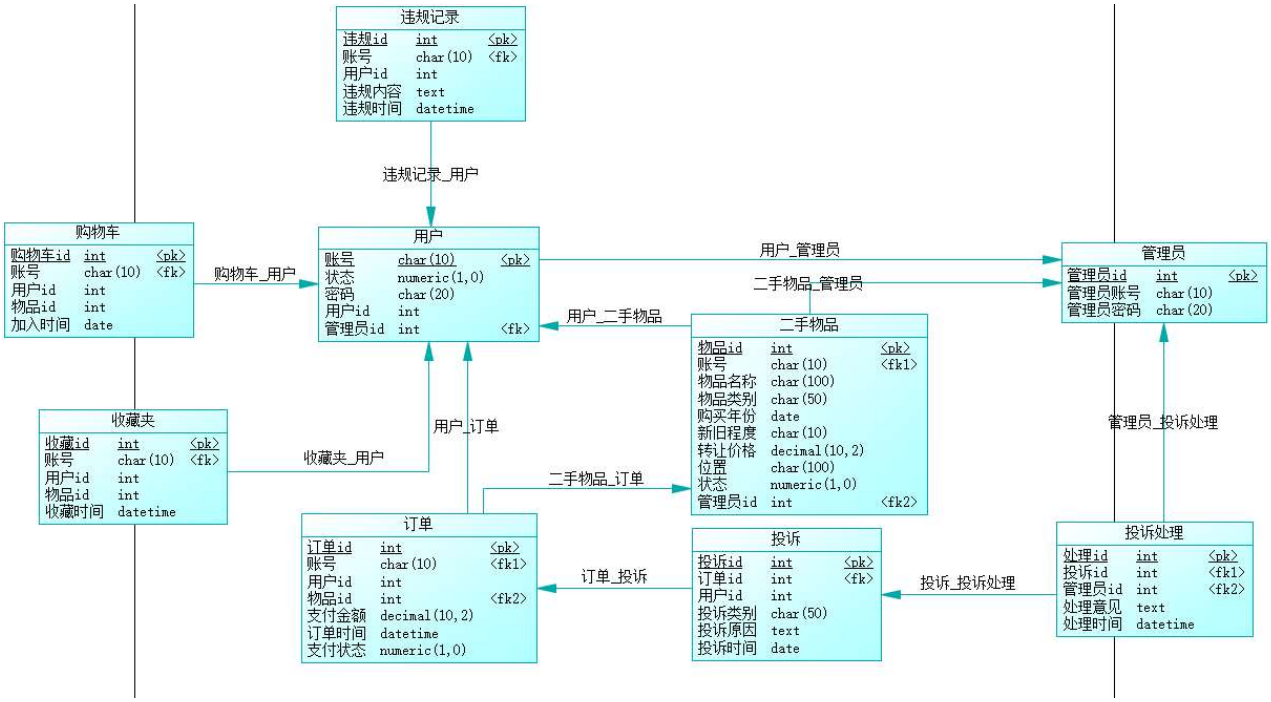
要求：截图务必清晰，如果图太大可截图一个总图，然后再分块截图。如果看不清截图会影响成绩。



2.1.3 PDM 图

要求：截图务必清晰，如果图太大可截图一个总图，然后再分块截图。如果看不清

截图会影响成绩。



2.1.4 数据库表结构

1、 表结构

选取 2-3 个比较有代表性的表结构截图，体现主键约束、外键约束、空值约束等。

二手物品表 item

cart 表

主键约束：cart_id 是主键。

外键约束：user_no 是外键，引用 userinfo 表中的 user_no，表明购物车属于哪个用户。

空值约束：所有字段都不能为 null。

```
create table cart
(
   cart_id              int not null  comment '',
   user_no              char(10) not null  comment '',
   user_id              int not null  comment '',
   item_id              int not null  comment '',
   added_time           date not null  comment '',
   primary key (cart_id)
);

alter table cart add constraint FK_CART_CART_USER_USERINFO foreign key (user_no)
      references userinfo (user_no) on delete restrict on update restrict;
```

orders 表

主键约束：order_id 是主键。

外键约束：user_no 和 item_id 是外键，分别引用 userinfo 和 item 表中的 user_no 和 item_id。

空值约束：所有字段都不能为 null。

```
create table orders
(
   order_id             int not null  comment '',
   user_no              char(10) not null  comment '',
   user_id              int not null  comment '',
   item_id              int not null  comment '',
   amount               decimal(10,2) not null  comment '',
   order_time           datetime not null  comment '',
   payment_status       numeric(1,0) not null  comment '',
   primary key (order_id)
);

alter table orders add constraint FK_ORDERS_ITEM_ORDE_ITEM foreign key (item_id)
      references item (item_id) on delete restrict on update restrict;

alter table orders add constraint FK_ORDERS_USER_ORDE_USERINFO foreign key (user_no)
      references userinfo (user_no) on delete restrict on update restrict;
```

admin 表

主键约束：admin_id 是主键，不允许为空。

空值约束：admin_no 和 admin_password 是必填字段，不能为 null。

```
create table admin
(
   admin_id             int not null  comment '',
   admin_no             char(10)  comment '',
   admin_password       char(20)  comment '',
   primary key (admin_id)
);
```

2、 索引

1) 索引截图

```
CREATE INDEX idx_item_price
ON item (price ASC);
```

2) 使用场景说明（用途）

便于用户可能会按价格范围筛选商品，如查询价格低于某个值的商品，或者按价格升序/降序排序。

3、 视图

1) 视图截图

```
create view UserCartView as
select
    u.user_id, u.user_no, i.item_id, i.item_name, i.category, i.price, i.status, c.added_time
from userinfo u
join cart c on u.user_id = c.user_id
join item i on c.item_id = i.item_id
```

2) 使用场景说明（用途）

展示普通用户的购物车信息，包含商品的基本信息以及添加时间

4、 触发器

1) 触发器截图

```
AFTER UPDATE ON orders
FOR EACH ROW
BEGIN
    -- 检查支付状态是否从未支付（0）更新为支付完成（1）
    IF NEW.payment_status = 1 AND OLD.payment_status = 0 THEN
        -- 更新对应的 item 表中 item_id 的状态为已售出（设定 status = 0 表示已售出）
        UPDATE item
        SET status = 0
        WHERE item_id = NEW.item_id;
    END IF;
END;
//
DELIMITER ;
```

2) 使用场景说明（用途）

当某个商品在 orders 表中被标记为支付完成（payment_status = 1）时，自动更新 item 表中对应商品的状态为 "已售出"（设定 status = 0 表示已售出）

3) 验证触发器

插入一个新商品到 item 表（先建立管理员和用户）

```
INSERT INTO admin (admin_id, admin_no, admin_password)
VALUES (1, 'A123456789', 'adminpass');

INSERT INTO userinfo (user_no, user_state, user_password, user_id, admin_id)
VALUES ('U123456789', 1, 'password123', 101, 1);

INSERT INTO item (item_id, user_no, item_name, category, purchase_year, conditions, price, location, status)
VALUES (1, 'U123456789', '二手书籍', '书籍', '2021-01-01', '9成新', 50.00, '图书馆', 1);  -- status = 1 表示商品可用
```

插入一个新的未支付订单到 orders 表

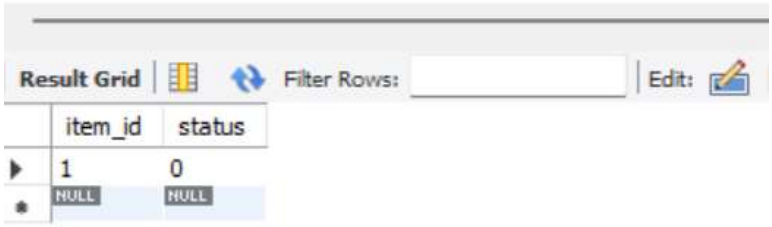
```
INSERT INTO orders (order_id, user_no, user_id, item_id, amount, order_time, payment_status)
VALUES (1, 'U123456789', 101, 1, 50.00, NOW(), 0);  -- payment_status = 0 表示未支付
```

更新订单状态并触发触发器

```sql
UPDATE orders
SET payment_status = 1
WHERE order_id = 1;
```

验证触发器是否生效

```sql
SELECT item_id, status
FROM item
WHERE item_id = 1;
```



查询结果显示商品状态 status = 0，表示商品已售出。

5、 存储过程或存储函数

1) 存储过程或存储函数截图

```sql
DELIMITER //

CREATE PROCEDURE FreezeUserAccount(
    IN p_user_no CHAR(10),          -- 输入参数，用户编号
    IN p_admin_id INT               -- 输入参数，管理员 ID
)
BEGIN
    -- 检查该用户是否存在
    DECLARE user_exists INT;
    SELECT COUNT(*) INTO user_exists
    FROM userinfo
    WHERE user_no = p_user_no;

    IF user_exists = 0 THEN
        -- 如果用户不存在，抛出错误信息
```

```
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = '用户不存在';
        ELSE
            -- 更新用户状态为 0（冻结状态）
            UPDATE userinfo
            SET user_state = 0, admin_id = p_admin_id
            WHERE user_no = p_user_no;

            -- 记录操作日志或显示成功消息
            SELECT CONCAT('用户 ', p_user_no, ' 已被冻结。') AS Result;
        END IF;
    END;

    //
    DELIMITER ;
```

2)　　使用场景说明（用途）

管理员通过输入 user_no（用户编号）来冻结用户账户

比如冻结用户编号为 U123456789 的用户，管理员的 ID 为 1

```
1 •  CALL FreezeUserAccount('U123456789', 1);
2 •  SELECT user_no, user_state FROM userinfo WHERE user_no = 'U123456789';
3
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content

user_no	user_state
U123456789	0
NULL	NULL

user_state 更新为 0，表示该用户已经成功被冻结