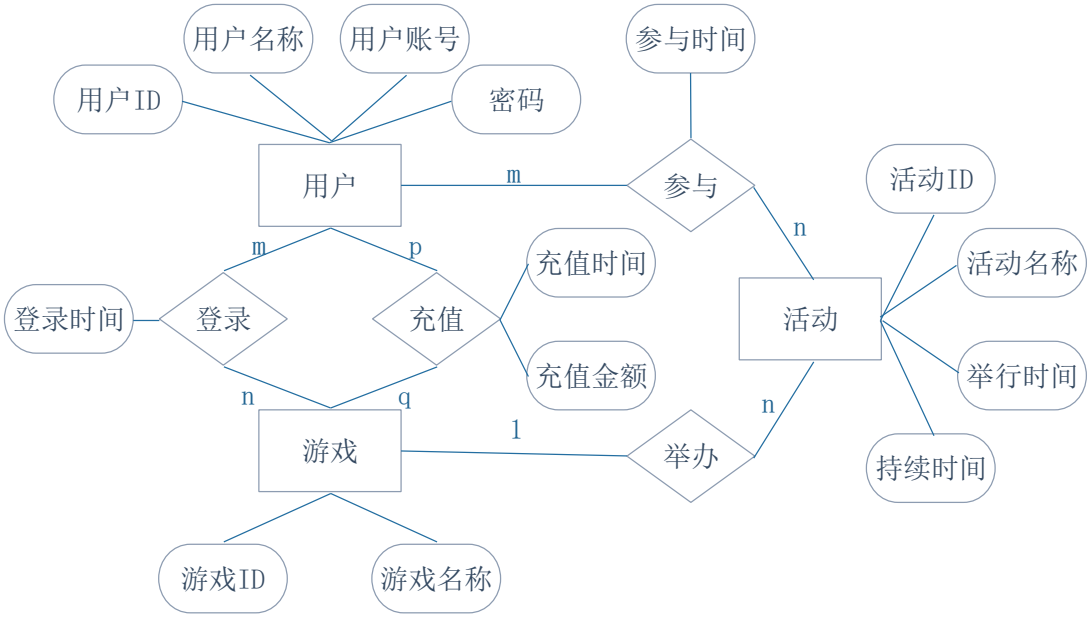
数据库原理与应用综合大作业

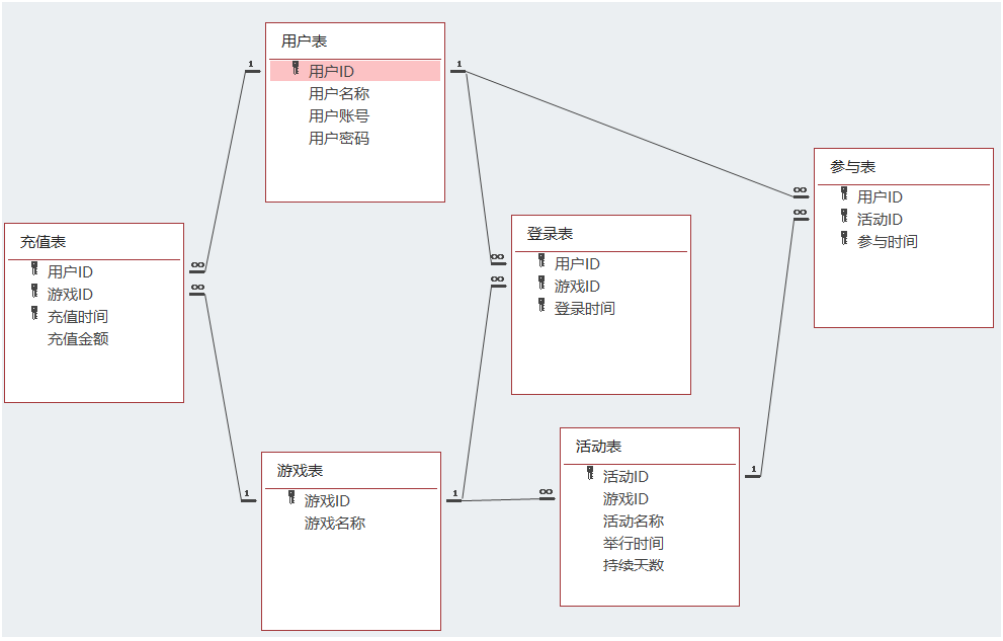
学号：20354027　　班级：2班　　姓名：方桂安

一、 概念设计 -- ER 图 （6 分）



二、逻辑设计 -- 关系模式 （4 分）

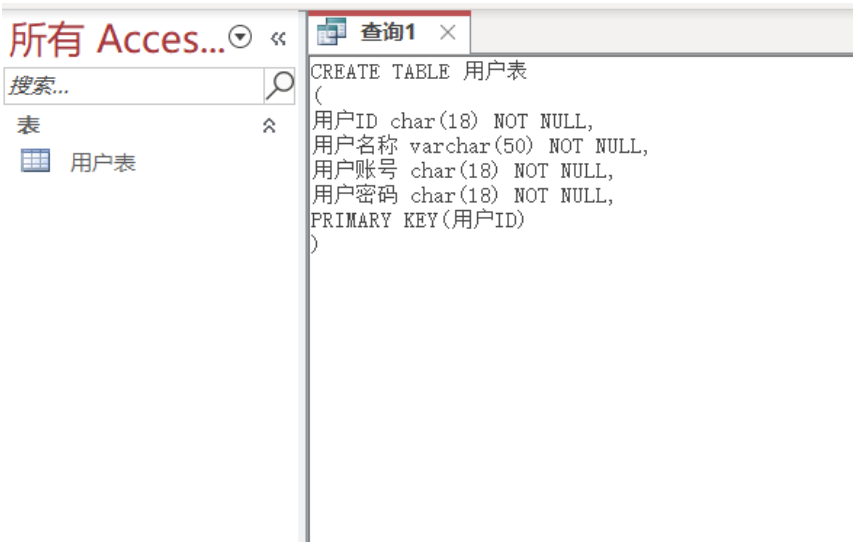
用户表（用户ID，用户名称，用户账号，用户密码）
游戏表（游戏ID，游戏名称）
活动表（活动ID，游戏ID，活动名称，举行时间，持续天数）
登录表（用户ID，游戏ID，登录时间）
充值表（用户ID，游戏ID，充值时间，充值金额）
参与表（用户ID，活动ID，参与时间）



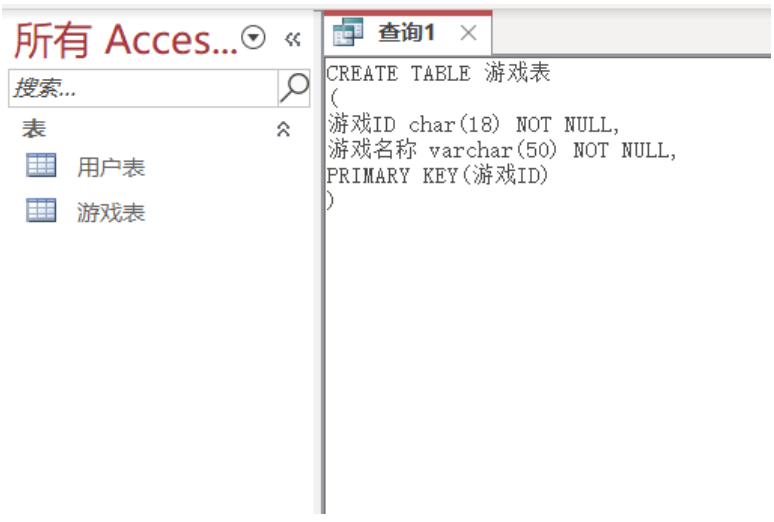
三、SQL代码（建表、查询、过程和函数、触发器）（18 分）

1. 建表

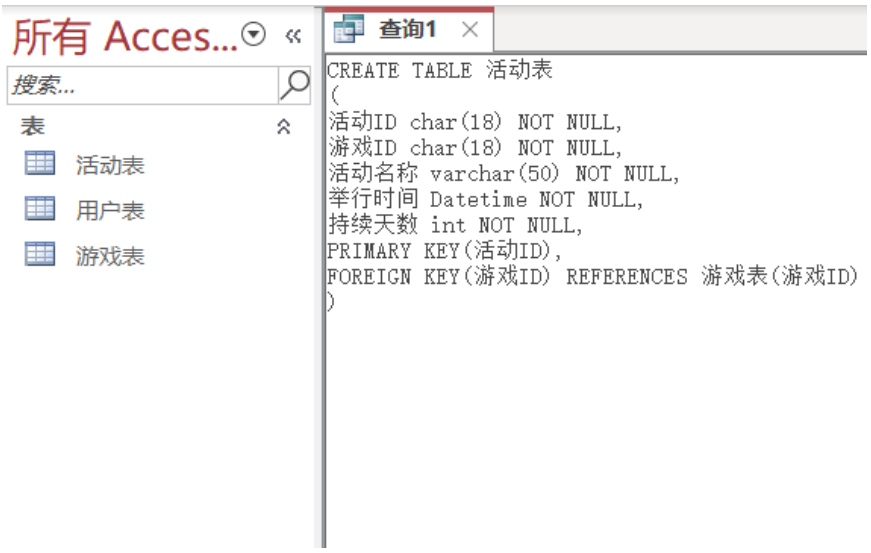
```
CREATE TABLE 用户表
(
用户ID char(18) NOT NULL,
用户名称 varchar(50) NOT NULL,
用户账号 char(18) NOT NULL,
用户密码 char(18) NOT NULL,
PRIMARY KEY(用户ID)
)
```



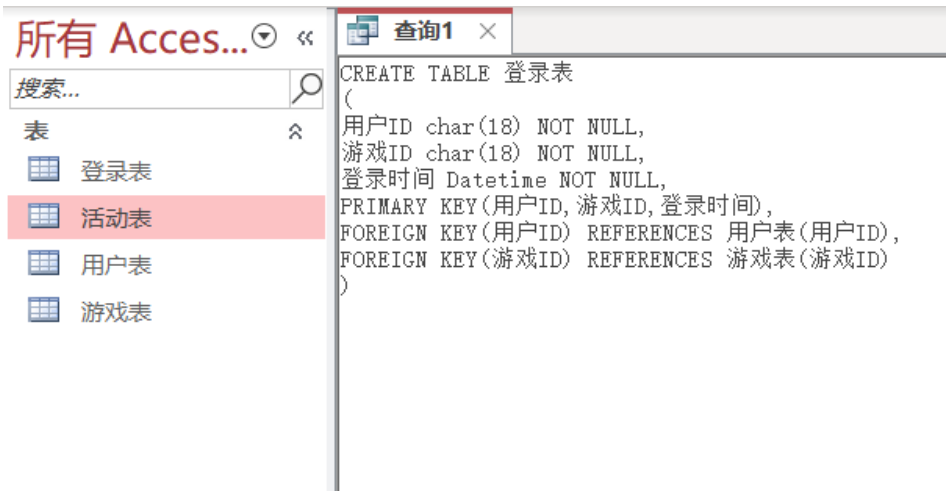

```
CREATE TABLE 游戏表
(
游戏ID char(18) NOT NULL,
游戏名称 varchar(50) NOT NULL,
PRIMARY KEY(游戏ID)
)
```



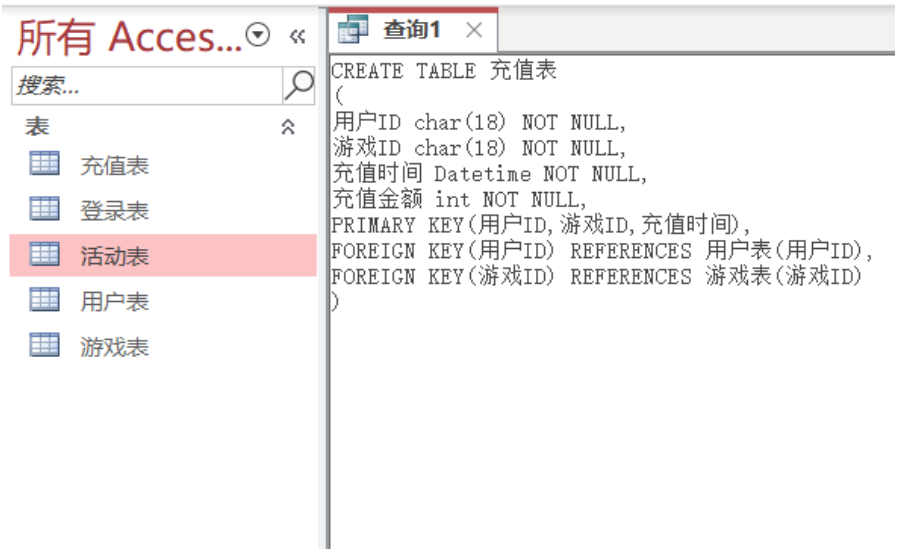
```
CREATE TABLE 活动表
(
活动ID char(18) NOT NULL,
游戏ID char(18) NOT NULL,
活动名称 varchar(50) NOT NULL,
举行时间 Datetime NOT NULL,
持续天数 int NOT NULL,
PRIMARY KEY(活动ID),
FOREIGN KEY(游戏ID) REFERENCES 游戏表(游戏ID)
)
```



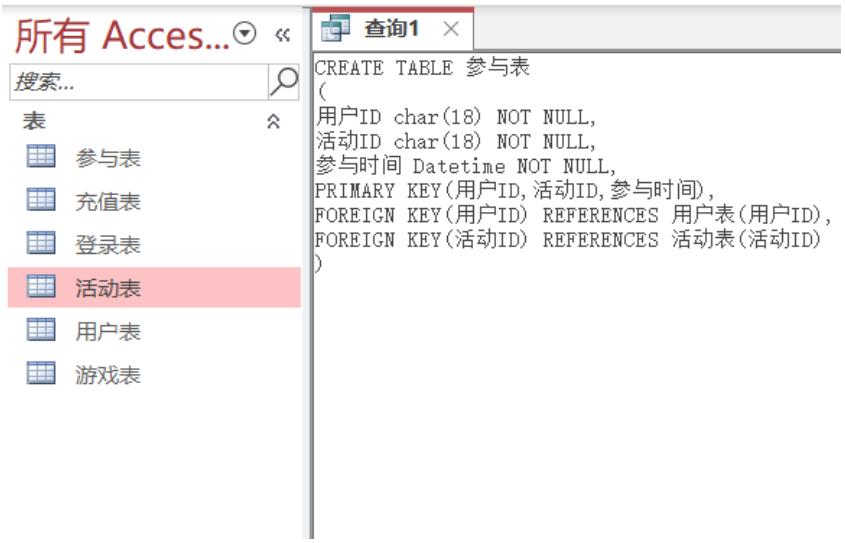

```
CREATE TABLE 登录表
(
用户ID char(18) NOT NULL,
游戏ID char(18) NOT NULL,
登录时间 Datetime NOT NULL,
PRIMARY KEY(用户ID,游戏ID,登录时间),
FOREIGN KEY(用户ID) REFERENCES 用户表(用户ID),
FOREIGN KEY(游戏ID) REFERENCES 游戏表(游戏ID)
)
```



```
CREATE TABLE 充值表
(
用户ID char(18) NOT NULL,
游戏ID char(18) NOT NULL,
充值时间 Datetime NOT NULL,
充值金额 int NOT NULL,
PRIMARY KEY(用户ID,游戏ID,充值时间),
FOREIGN KEY(用户ID) REFERENCES 用户表(用户ID),
FOREIGN KEY(游戏ID) REFERENCES 游戏表(游戏ID)
)
```

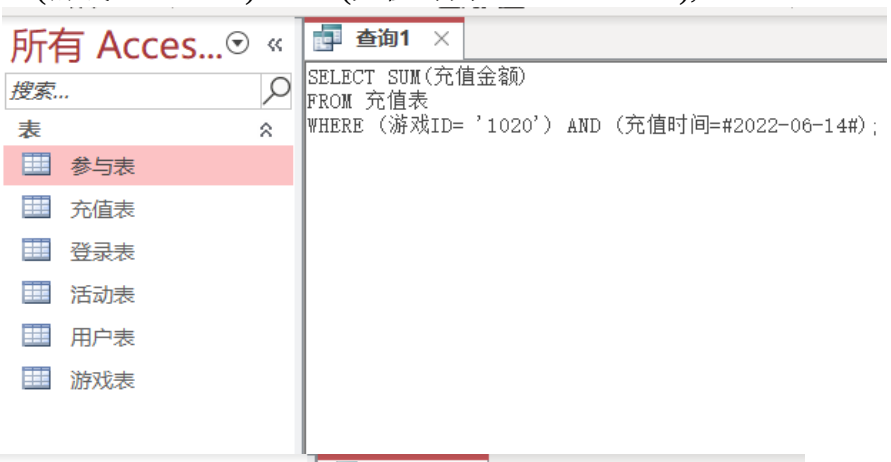


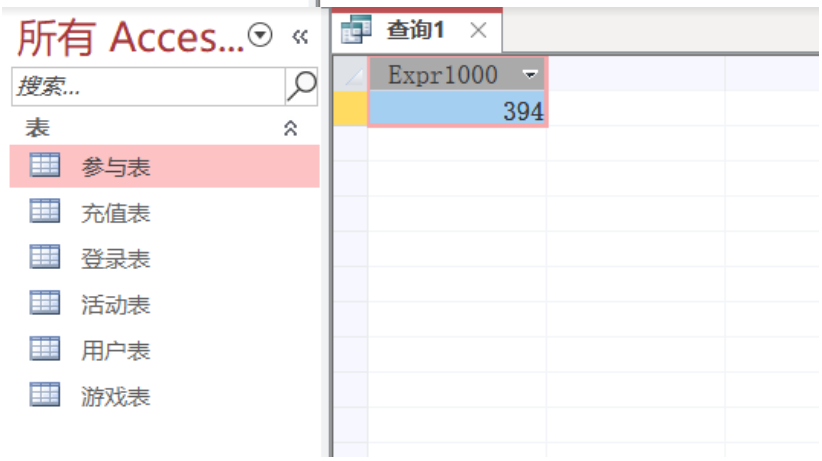

```
CREATE TABLE 参与表
(
用户ID char(18) NOT NULL,
活动ID char(18) NOT NULL,
参与时间 Datetime NOT NULL,
PRIMARY KEY(用户ID,活动ID,参与时间),
FOREIGN KEY(用户ID) REFERENCES 用户表(用户ID),
FOREIGN KEY(活动ID) REFERENCES 活动表(活动ID)
)
```



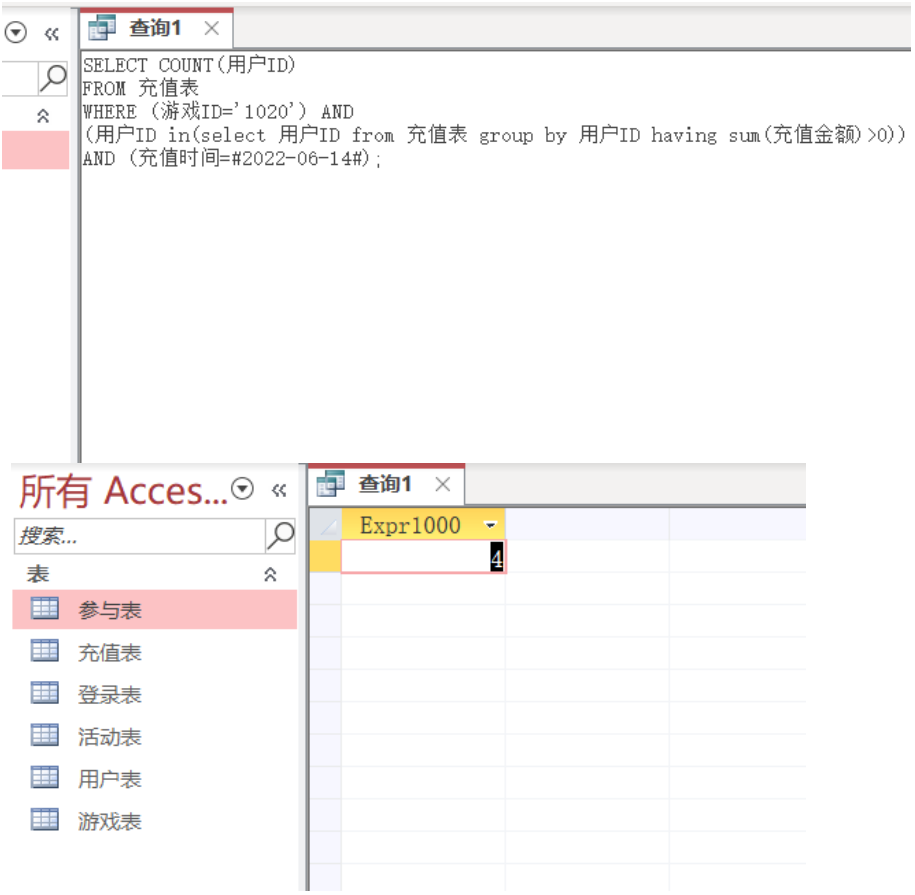
2. 查询

SELECT SUM(充值金额)
FROM 充值表
WHERE (游戏ID= '1020') AND (充值时间=#2022-06-14#);

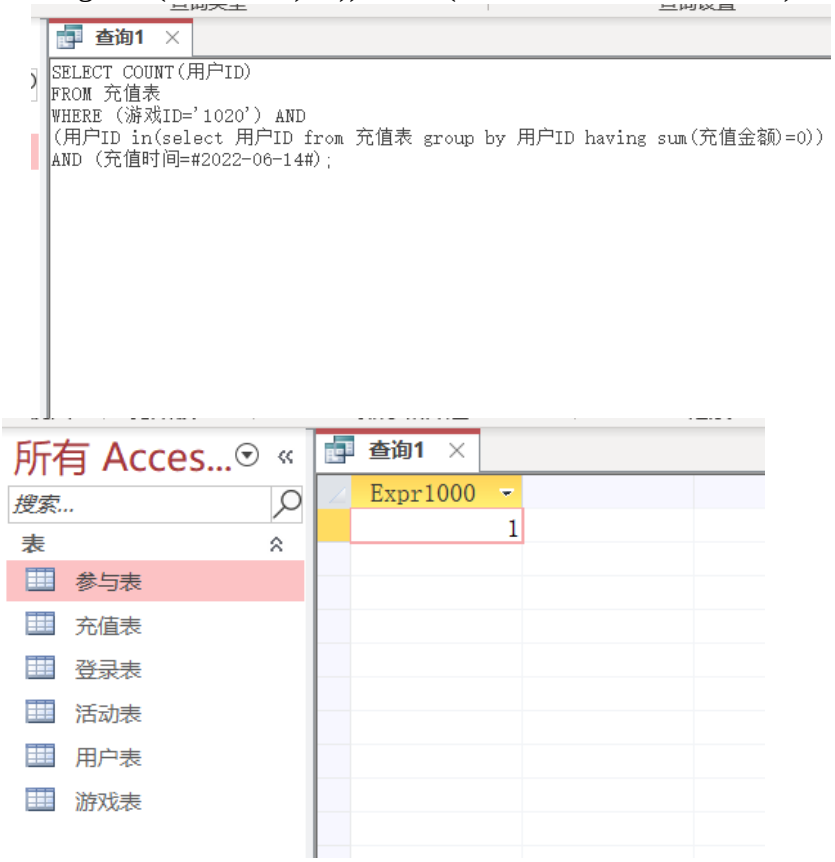




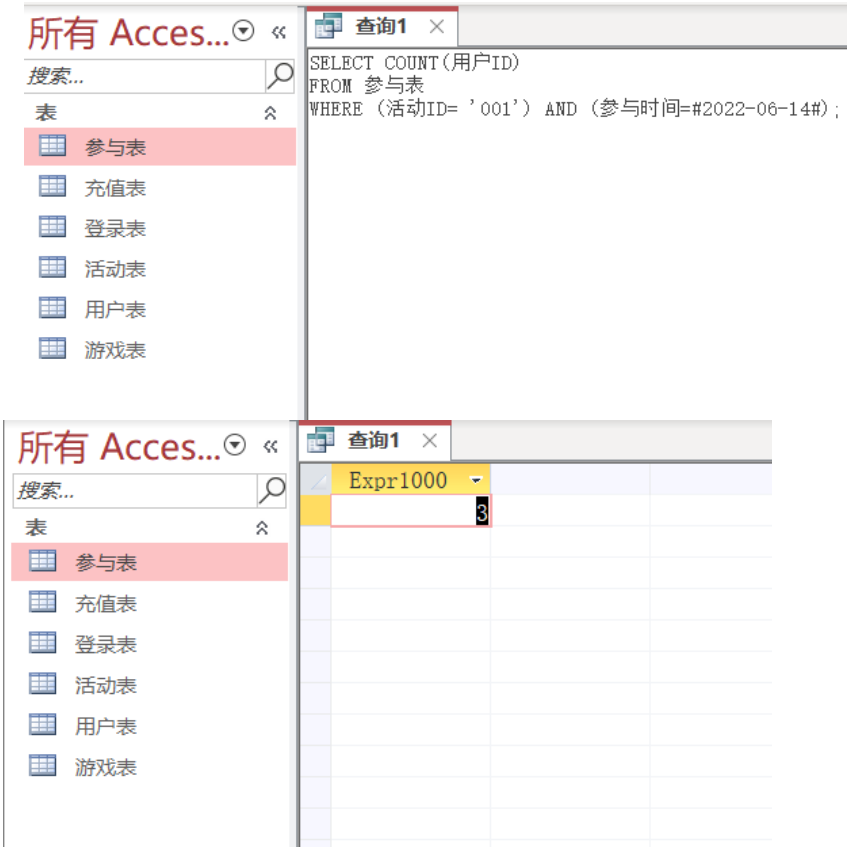
SELECT COUNT(用户ID)
FROM 充值表
WHERE (游戏ID='1020') AND (用户ID in(select 用户ID from 充值表 group by 用户ID having sum(充值金额)>0))
AND (充值时间=#2022-06-14#);



SELECT COUNT(用户ID)
FROM 充值表
WHERE (游戏ID='1020') AND (用户ID in(select 用户ID from 充值表 group by 用户ID having sum(充值金额)=0)) AND (充值时间=#2022-06-14#);



```
SELECT COUNT(用户ID)
FROM 参与表
WHERE (活动ID= '001') AND (参与时间=#2022-06-14#);
```



3. 过程和函数

```
CREATE OR REPLACE
FUNCTION login_number
(ID char,
时间 datetime)
RETURN integer
BEGIN
DECLARE login_day int;
SELECT COUNT(用户ID)
INTO login_day
FROM 登录表
WHERE (登录表.游戏ID=ID) AND (登录表.登录时间=时间);
RETURN login_day;
END

CREATE PROCEDURE pay_number
(in gameID char,out pay_man int)
BEGIN
SELECT COUNT(用户ID)
INTO pay_man
FROM 充值表
WHERE (充值表.游戏ID=gameID) AND (充值表.用户ID in(select 用户ID from
充值表 group by 用户ID having sum(充值金额)>0));
END
```

4. 触发器

```
CREATE TRIGGER delete_player
ON 参与表
AFTER DELETE
AS BEGIN DELETE 用户表
WHERE
用户表.用户ID IN(
SELECT 参与表.用户ID
FROM DELETED)
END
```

四、分析题（12 分）

1.Read1(B),Write2(B);
Read1(B),Write3(B);
Read2(B),Write3(B);
Read3(B),Write2(B);
Write2(B),Write3(B);
Read1(C),Write2(C);
Read2(C),Write1(C);
Write1(C),Write2(C);

2.No.w2(C) is in conflict with w1(C).w2(B) is in conflict with r1(B).So they can't be next to each other, this schedule isn't conflict serializable

T1	T2	T3
Read(A)		
	Read(B)	
		Read(A)
	Read(C)	
	Write(B)	
	Write(C)	
Read(B)		
Write(C)		
		Read(B)
		Write(B)

Its equivalent serial schedule:
Read2(B)Read2(C)Write2(B)Write2(C)Read1(A)Read1(B)Write1(C)Read3(A)Read3(B)Write3(B)

T1	T2	T3
	Read(B)	
	Read(C)	
	Write(B)	
	Write(C)	
Read(A)		
Read(B)		
Write(C)		
		Read(A)
		Read(B)
		Write(B)

3. <T_2 start>
<T_2,B,50,60>
<T_2,C,80,85>
<T_2 commit>
Let a checkpoint is set just after committing of Transaction T_2.
<T_1 start>
<T_1,C,85,90>
<T_1 commit>
<T_3 start>
<T_3,B,60,100>
<T_3 commit>