1. **Single-choice Questions (24 points, 3 points\*8)**

**Answer**

**A A B D D A A C**

**2. True or False Questions (12 points, 2 points \*6)**

**Answer**

**T F F F T T**

**3. SQL query (16 points)**

(1) **select distinct group\_name**

**from user**

**where user.gender = "male";**

语法/语义错写、漏写每个扣一分，如没写male条件、distinct没写

(2) **select user.user\_id, name**

**from user, followship**

**where user.user\_id = followship.follower\_id and followship.user\_id = "1001" and user.age > 18;**

4分，where中每个查询条件一分，select和from子句写对一分，

select中user\_id没指出表名扣一分；

where中join条件写错扣一分；

(3) 语法/语义错写、漏写每个扣一分，如没写female条件、group by条件写错、直接写max函数（而不是写在select里面）、limit 1扣一分

**select group\_name**

**from user**

**where user.gender = ‘female’**

**group by group\_name**

**having count(user\_id) >= all (**

**select count(user\_id)**

**from user**

**where user.gender = ‘female’**

**group by group\_name);**

或

**select group\_name**

**from user**

**where user.gender = ‘female’**

**group by group\_name**

**order by count(user\_id) desc**

**limit 1;**

(这种答案算部分对，因为只输出了一个结果，实际可能有多个，去年这种算对的)

或使用with 子句

**with T(group\_name, female\_count) as (select group\_name, count(user\_id)**

**from user**

**where user.gender = ‘female’**

**group by group\_name)**

**select group\_name**

**from T**

**where T.female\_count = (select max(female\_count) from T);**

(4)

语法/语义错写、漏写每个扣一分，如没写game条件、group by条件写错、join条件写错、直接写avg函数（而不是写在select里面）、limit 1扣一分

**select user.user\_id, name, follower\_count**

**from user, (select user.user\_id, count(follower\_id) as follower\_count**

**from user, followship**

**where user.user\_id = followship.user\_id and user.group\_name = ‘game’**

**group by user.user\_id) as T**

**where user.user\_id = T.user\_id and T.follower\_count > (select avg(follower\_count) from T);**

with 子句类型

**with T(user\_id, follower\_count) as (select user.user\_id, count(follower\_id)**

**from user, followship**

**where user.user\_id = followship.user\_id and user.group\_name = ‘game’**

**group by user.user\_id)**

**with R(avg\_follower\_count) as (select avg(follwer\_count) from T)**

**select user.user\_id, name, follower\_count**

**from user, T, R**

**where user.user\_id = T.user\_id and T.follower\_count > R. avg\_follower\_count;**

另有一解：

**select user\_id**

**from user natural join followship**

**where group\_name = ‘game’**

**group by user\_id**

**having count(follower\_id) >= ((**

**select count(follower-id)**

**from user natural join followship**

**where user.group****\_name =‘game’) /**

**(select count(\*)**

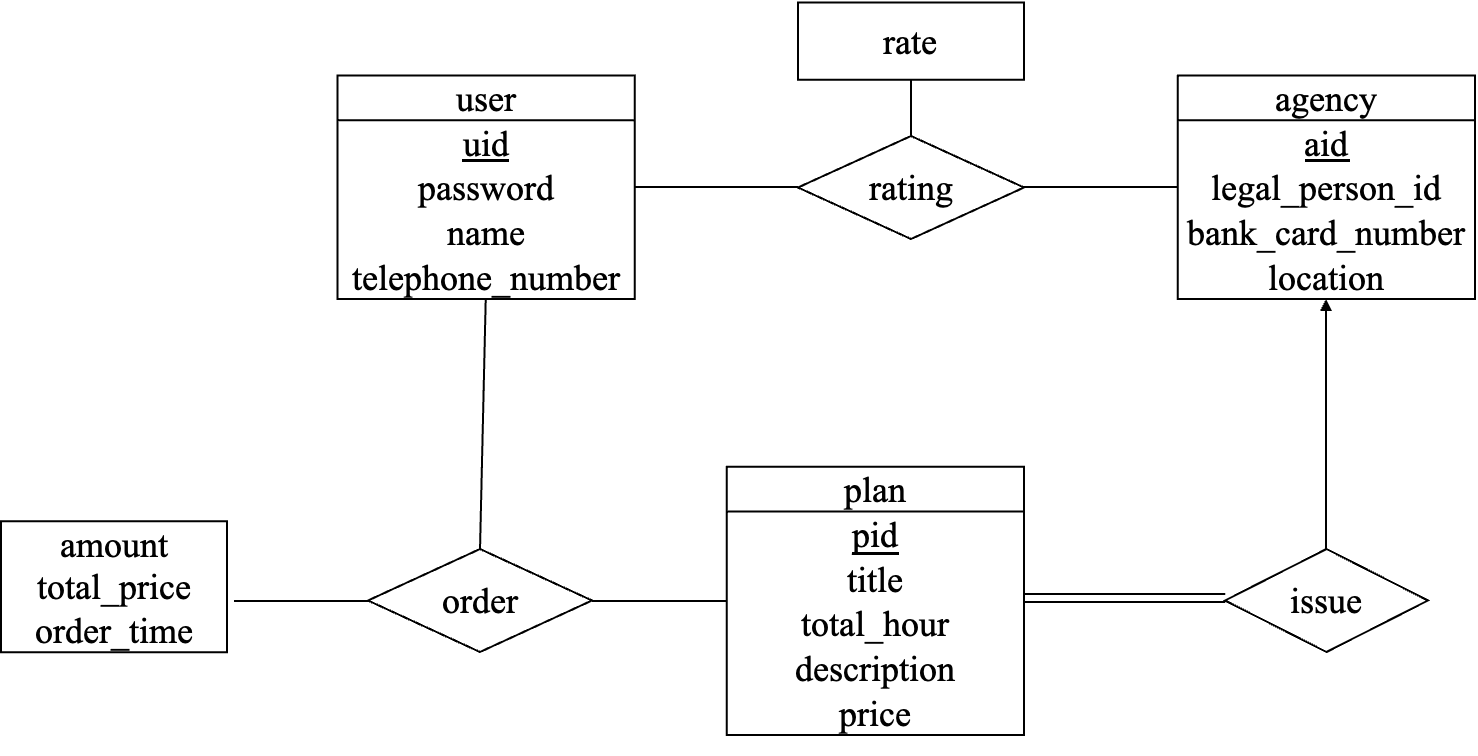
**from user**

**where group\_name =’game’));**

**4. Database design (16 points)**

**Answer**

(1)



评分细则：

每个实体（user、agency、plan）、关系（rating的多对多关系、order的多对多关系、issue的多对多关系）、rating和order关系衍生出的属性各一分，共8分；

每少写/错写减一分。

关系的名称随意，但是注意关系的类型和箭头；

注意order和rating每个多对多关系可以写成两个一对多关系，两个一对多关系需要注意箭头方向正确（指向一的实体）。

(2)

The relational schemas are as follows.

user(uid, password, name, telephone\_number)

agency(aid, legal\_person\_ID, bank\_card\_number, location)

travel\_plan(pid, title, total\_hour, description, price, agency\_ID, issue\_time)

order(oid, user\_ID, plan\_ID, amount, total\_price, order\_time)

rating(user\_ID, agency\_ID, rate)

The primary key of each table is underlined. (rating表如果有单独主键也给分，外键写明白即可)

The foreign keys are as follows:

* the agency\_ID of the travel plan table references the aid of the agency table
* the user\_ID of the order table references the uid of the user table
* the plan\_ID of the order table references the pid of the travel plan table
* the user\_ID of the user table references the uid of the user table
* the agency\_ID of the rating table references the aid of agency table

评分细则：

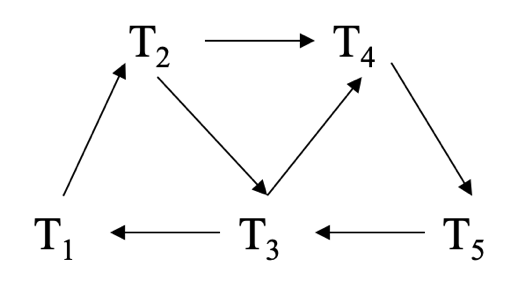
一共8分

* user和agency表正确，各一分
* travel\_plan表正确，主键和属性一分，外键一分；
* order表正确，主键和属性一分，外键一分；
* rating表正确，主键和属性一分，外键一分；

没有指出外键属性-3分；直接写create语句而不是schema扣1分。

**5. Concurrency Control (8 points)**

**Answer**



评分细则：图中的边多/少一条扣1分，全对给3分

(2) S is not conflict serializable，because there are cycles in the graph :

T1 🡪 T2 🡪 T3 🡪 T1

T4 🡪 T5 🡪 T3 🡪 T4

T1 🡪 T2 🡪 T4 🡪 T5 🡪 T3 🡪 T1

评分细则：答案错误扣3分，指出一个环算全对，全对给3分；没指出环扣1分。

(3) No，because every schedule generated by 2PL is serializable. :

评分细则：答案错误扣2分，理由错误扣一分，全对给2分

**6. ARIES Recovery. (12 points)**

**Answer:**

(1) **(5235, 8010, 8010)**. Log 8010 modifies page 5235, which should be added to the DirtyPageTable.

评分标准：2分。

(2) **8003**, which is the smallest RecLSN in the DirtyPageTable.

评分标准：2分。

(3) **<T4, 5235.1, 60>, <T2, 3462.1, 40>, <T2, 3462.1, 30>.**

The active transaction table after the analysis pass contains T2 and T4. The related update records should be redone.

评分标准：4分，写出3个各得1分；<T2, 3462.1, 40>在<T2, 3462.1, 30>前面得1分；每多写一个扣1分。

(4) **30, 80**. Location 3462.1 is modified by T2, which is undone. Location 6421.1 is modified by T1, which is committed.

评分标准：4分，1个2分。

**7. Buffer Tree. (12 points)**

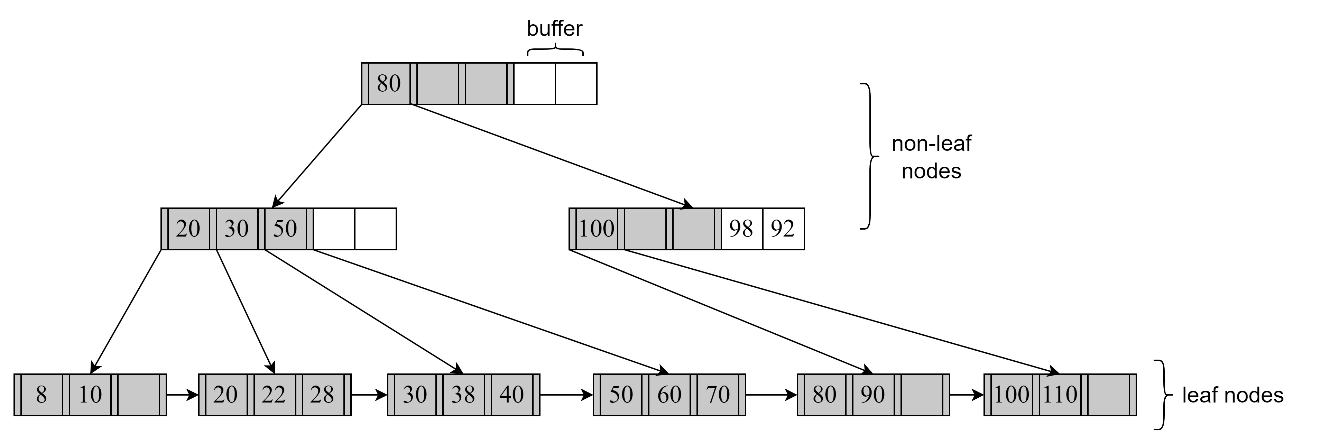
**Answer:**

(1) **4.** First, an equality search on 30 requires accessing the root node, the leftmost node on the second level, and the second leaf node on the left. Then, one additional leaf node needs to be searched to collect all items from 30 to 50.

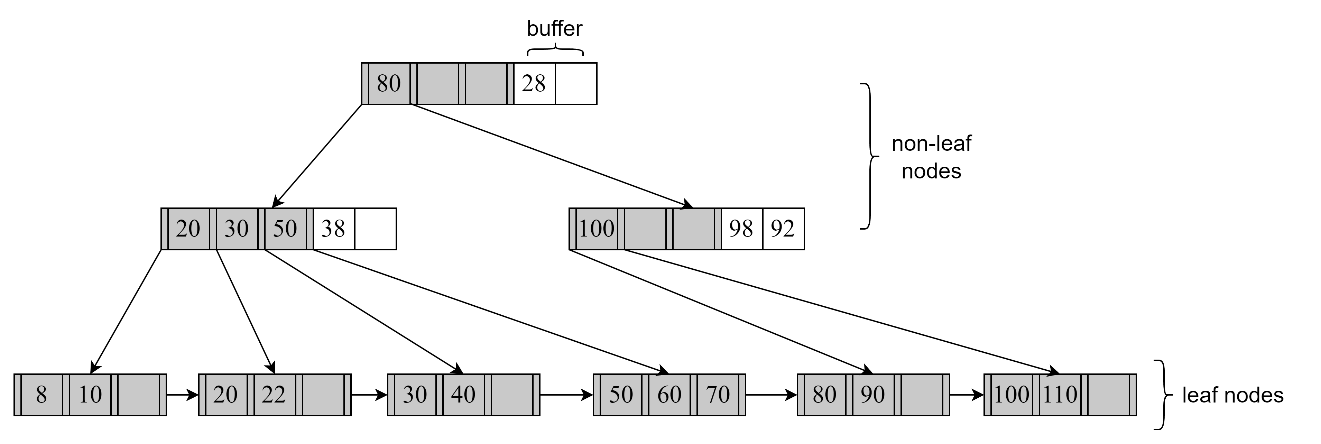
评分标准：数量答对得1分，解释合理给2分（只要体现出range query的查询思路，就可以得分）。

(2)

答案一：按照题目要求，如果buffer溢出，将所有元素下推。



答案二：也可以只下推前两个元素，将剩下的元素放进原来的buffer里。



评分标准：6分。根节点错扣1分，第二层两个节点错一个扣1分，第三层左边三个节点错一个扣1分，右边三个节点错任何一个扣1分，扣完为止。第二层右边的节点buffer如果写成92 98，算对。两种答案，以令本大题得分较高的为准。

(3) 若(2)为答案一，则为**0, 0, 5**；若(2)为答案二，则为**0, 0, 4。**

解析：

|  |  |
| --- | --- |
| 插入 | 描述 |
| 38 | 只插到根节点buffer |
| 92 | 只插到根节点buffer |
| 28 | 需要修改第二层两个buffer、左边三个（或者两个）叶子节点，第二层左边节点的指针，一共是5个（或者4个）节点 |

评分标准：3分，1个1分。