中山大学本科生期末考试

考试科目：《数据库系统》（A 卷）

学年学期：2015 学年第三学期	姓　　名：____________________
学 院/系：数据科学与计算机学院	学　　号：____________________
考试方式：开卷	年级专业：____________________
考试时长：120 分钟	班　　别：____________________

警示 **《中山大学授予学士学位工作细则》第八条：“考试作弊者，不授予学士学位。”**

--------------以下为试题，共七道大题，总分 100 分,考生请在答题纸上作答--------------

Question 1. (15 marks) Consider the following relational schema:

Likes(drinker, beer)
Frequents(drinker, bar)
Serves(bar, beer)

(i) (5 marks) For each bar, count the number of drinkers that frequent that bar and that like both 'zhujiang' and 'haerbing'.

Answer

```
SELECT x.bar, count(distinct x.drinker)
FROM frequents x, likes y, likes z
WHERE x.drinker = y.drinker AND x.drinker = z.drinker
   AND y.beer = ' zhujiang ' AND z.beer = 'haerbing'
GROUP BY x.bar
```

(ii) (5 marks) Find all drinkers that frequent some bar that serves some beer that they like.

Answer

```
SELECT DISTINCT x.drinker
```

FROM frequents x, serves y, likes z

WHERE x.bar = y.bar AND y.beer = z.beer AND x.drinker = z.drinker

(iii) (5 marks) Give an expression in the **relational algebra** to formulate the query in **(ii).**

Answer

One possible solution:

$$\pi_{x.drinker}\ (\sigma_{\text{x.bar = y.bar AND y.beer = z.beer AND x.drinker = z.drinker}}\ (x \times y \times z))$$

Question 2. (15 marks) A store maintains the following relations for its employee records:

employee (officeNum, SSN, phone, managerName, deptNum)

Given the following functional dependencies:

officeNum $\rightarrow$ phone
SSN $\rightarrow$ officeNum, deptNum
deptNum $\rightarrow$ managerName

(i) (5 marks) List one key of the employee relation, and compute the attribute closure of the key you have listed. You are required to show the major steps of computing the attribute closure.

Answer

SSN+ = SSN
= SSN officeNum deptNum
= SSN officeNum deptNum phone
= SSN officeNum deptNum phone managerName

(ii) (10 marks) Is the employee relation in BCNF? If so, write "Yes" below. Otherwise, decompose it into BCNF and underline all keys and foreign keys in the final relations.

Answer

One possible solution:

The relation is not in BCNF.

From officeNum $\rightarrow$ phone,
R1={officeNum, phone}, R2={officeNum, SSN, deptNum, managerName}

R1 is in BCNF.

From deptNum → managerName,
R21={deptNum, managerName}, R22={deptNum, SSN, officeNum}
R21 is in BCNF.

From SSN → officeNum, deptNum,
R22 is in BCNF.

The final relations are:
R1={officeNum, phone},
R21={deptNum, managerName},
R22={deptNum, SSN, officeNum}, where deptNum and officeNum in R22 are foreign keys to R21(deptNum) and R1(officeNum) respectively.

Question 3. (10 marks) Consider the following two schedules S1 and S2, where the notation is self-explanatory (For example, R2(Y) means that transaction T2 reads data item Y). Draw the precedence graph for each schedule and decide if the schedule (S1 or S2) is conflict serializable. If it is, give an equivalent serial schedule of transactions (e.g., T2, T1, T3 – you don't need to list the individual read/write steps in the serial schedule). If the schedule is not conflict serializable, explain why not. Label the edges of the precedence graphs with the data item that causes the conflict (e.g., X, Y, or Z).

(i) (5 marks) Schedule S1: R2(Y); W2(Y); R3(Y); R1(X); W1(X); W3(Y); R2(X); R1(Y); W1(Y)

Answer

This is not conflict-serializable because of the cycles in the precedence graph.

(ii) (5 marks) Schedule S2: R3(Y); R3(Z); R1(X); W1(X); W3(Y); R2(Z); R1(Y); R2(X); W1(Y); W2(X)

Answer

This is conflict-serializable because of no cycles in the precedence graph. The equivalent serial schedule is T3; T1; T2.

Question 4. (10 marks) Consider the following schedules S, where the notation is self-explanatory (For example, R1(X) means that transaction T1 reads data item X; similarly, L1_S(X) and L1_X(X) mean that transaction T1 locks data item X in *shared* lock and *exclusive* lock respectively; U1(X) means that transaction T1 removes the lock on data item X.)

Schedule S2: R1(X); W1(X); R2(X); R1(Y); W1(Y); R2(Y); W2(Y); W2(X)

Could this schedule be produced by a scheduler that is using the Two-Phase Locking protocol (2PL)? Explain why or why not. (If it is possible, an easy way to demonstrate that is to rewrite (copy) the schedule with lock and unlock operations inserted in the appropriate places.)

Answer

Yes, this is possible.

One equivalent schedule with 2PL inserted is:
L1_X(X); L1_X(Y); R1(X); W1(X); U1(X); L2_X(X); R2(X); R1(Y); W1(Y); U1(Y); L2_X(Y); R2(Y); W2(Y); W2(X)

Question 5. (10 marks) Consider slotted pages that are used to store variable-length records. Assume a newly created page numbered P, where we allocate a slot in the slot directory when a record comes. The first slot is numbered 1, the second slot is numbered 2, and etc. Consider the following sequence of actions:

Insert record A of length 10 bytes. Insert record B of length 20. Insert record C of length 15. Insert record D of length 25. Delete record B. Delete record D. When we delete a record, we change the length to be -1 in the record's slot, and reduce by 1 the number of slots in the slot directory. Compact the page (that is, move all records to be contiguous to one another without changing the position of the corresponding slots in the slot directory, starting from the beginning of the page, to free up space for new records).

(i) (6 marks) Draw the page P after the above sequence of actions has been finished. You do not need to draw the pointers for deleted records.

Answer

(ii) (4 marks) List RIDs (record id) of record A and record C, respectively.

Answer

RID of A = <P, 1>

RID of C = <P, 3>

Question 6. (20 marks) Consider the database consisting of the following relations, which is used as the running example in the textbook:

Sailors (<u>*sid*: integer</u>, *sname*: string, *rating*: integer, *age*: real)

Reserves (<u>*sid*: integer, *bid*: integer, *day*: dates</u>, *rname*: string)

The relation Reserves has 1000 pages in total, with 100 tuples per page, where there are 100 different boats in the range of [1, 100]. The relation Sailors has 500 pages in total, with 80 tuples per page, where there are 10 different ratings in the range of [1, 10]. Assume we have 15 pages in our buffer pool. Assume we have no index on both relations. Use the assumption that both boats and ratings have uniform distribution in the computation of I/O cost.

(i) (10 marks) Consider the following query:

```
SELECT   S.sname
FROM   Reserves R, Sailors S
WHERE   R.sid=S.sid AND
      R.bid=100 AND S.rating>5
```

Suppose we use the optimization strategy of “Push Selects” as fully as we can, and use page-oriented nested loops to join Sailors and Reserves, where Sailors is taken as the outer relation. Assume we do not use temporary files on disk to keep intermediate results. Draw the corresponding query plan using the convention that the outer relation is the left child of the join operator. Compute the I/O cost of the query plan without counting the output of the final query result. You need to explain each part of the I/O cost in the computation.

Answer

I/O cost = 500 + 1000 = 1500
We need to read both Sailors and Reserves once, and the 10 pages of intermediate results of Reserves can be kept in the buffer pool.

(ii) (10 marks) Consider the same query as mentioned in (i), while we still we use the optimization strategy of “Push Selects” as fully as we can, we now use sort-merge join to join Sailors and Reserves. Assume we use temporary files on disk to keep intermediate results only when we have to. Draw the corresponding query plan using Reserves as the left child of the join operator. Compute the best possible I/O cost of the query plan without counting the output of the final query result. You need to explain each part of the I/O cost in the computation.

Answer

cost of plan:

Scan Reserves (1000) + write temp T1 (10 pages) = 1010.

Scan Sailors (500) + write temp T2 (250 pages) = 750.

sort T2 in 3 passes:

Pass 0: 250/15 +1 = 17 runs

Pass 1: 17/14 +1 = 2 runs

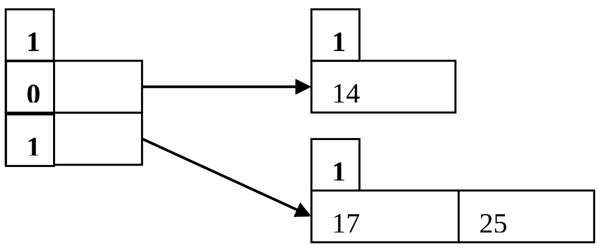
Pass 2: 2/14 + 1 = 1 run

Cost = (2*3*250) = 1500

merge (10+250) = 260, since 10 pages of Reserves can be sorted and saved in the buffer pool which has 15 pages.

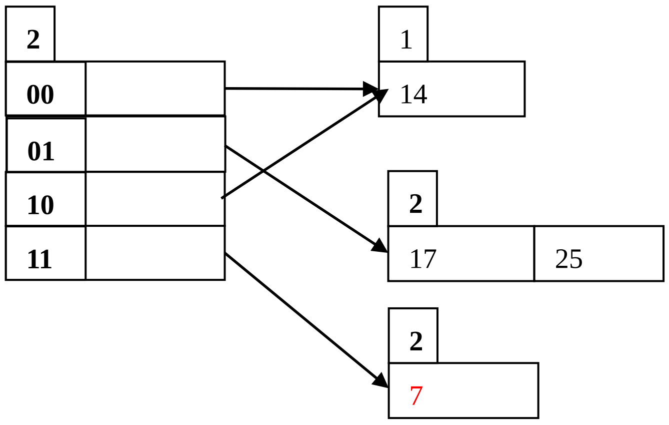
Total: 1010 + 750 + 1500 + 260 = 3520 page I/Os.

Question 7. (20 marks) You are given an initial extendable hash structure with three keys already inserted as below. You should follow the convention that binary hash indices start from the least significant bit. Draw *five* extendable hash structures corresponding for each insertion of the following search key values *K*: 7, 15, 20, 37, 18. Assume each bucket can hold two keys and the search key values arrive in the given order (i.e. 7 being the first coming key and 18 being the last one).

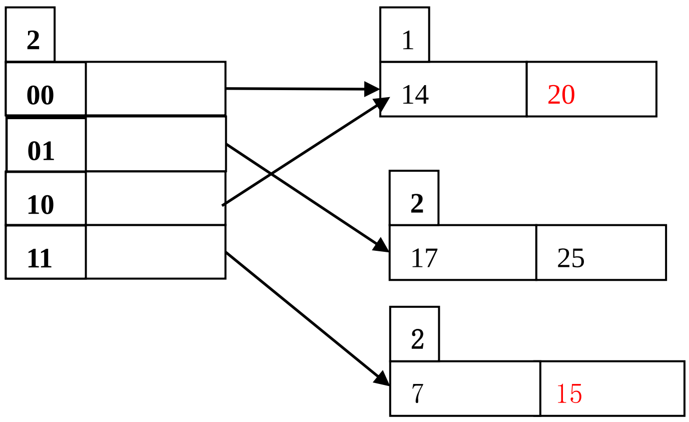


Answer

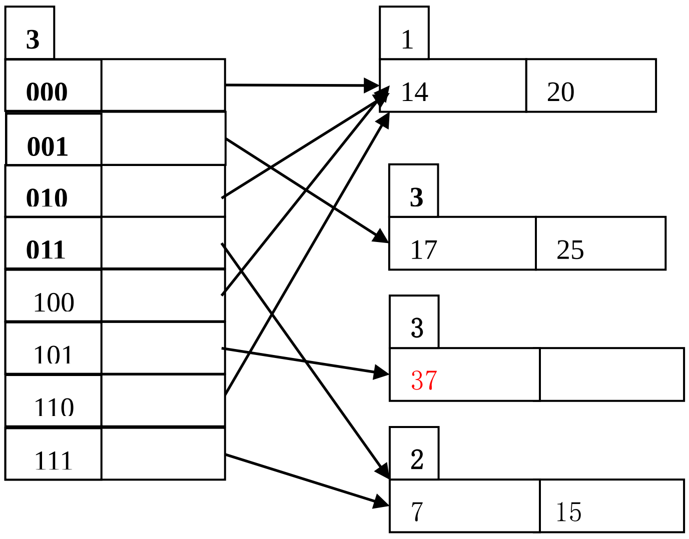
Insert 7



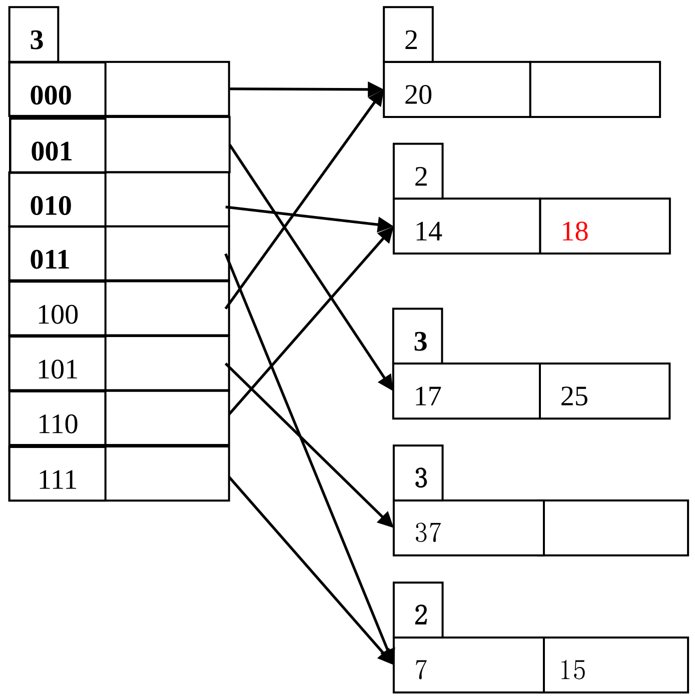
Insert 15 and 20



Insert 37



Insert 18


3
000
001
010
011
100
101
110
111
2
20
2
14
18
3
17
25
3
37
2
7
15