中山大学软件学院 **2011** 级软件工程专业(2012 春季学期)

# 《SE-304 数据库系统》期末试题试卷(B)

**(考试形式：开卷　　考试时间：2 小时)**

警示

**《中山大学授予学士学位工作细则》第六条**

**考试作弊不授予学士学位**

方向：＿＿＿＿＿＿＿＿　　姓名：＿＿＿＿＿　　学号：＿＿＿＿＿＿

出卷：＿＿＿＿＿＿＿＿＿＿＿＿　　审核：＿＿＿＿＿＿＿＿＿＿

**注意：答案一定要写在答卷中，写在本试题卷中不给分。本试卷要和答卷一起交回。**

**Question 1. (20 marks)** Consider relation R(A,B,C,D) with the following functional dependencies: FD = { AB → C, B→D, D→A, C→D}.

**(a)** Compute the Closure (闭包) of any set of attributes. Determine all the candidate keys (候选码) of relation R.

**(b)** Write down a minimal cover (最小覆盖) for the set FD.

**(c)** Is R in BCNF? If not, decompose R into a collection of BCNF relations. Show each step of the decomposition process.

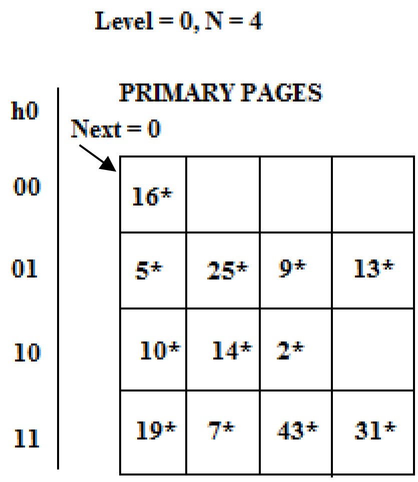
**Solution:**

**(a) Key: B**

**(b) B → C, D→A, C→D**

**(c) CD, ABC or AD,BC,CD**

**Question 2. (30 marks)** Suppose that we are using linear hashing index that take the following arriving search-key values K: 21(..101), 6(..110), 30(..110), 74(..010), 67(..011). The initial configuration is shown below and this setting is similar to the lecture slides where buckets can hold 4 keys (N =4), next pointer is defined by NEXT, h0 = h(K)mod(4) and so on.




Assuming the search-key values arrive in the given order (i.e. 21 being the first coming key and 67 being the last one). Show the linear hash structure for each insertion of the above key values file. (You should show clearly the stages of all essential parameters such as Next, hash function and overflow.)

**Solution:**

**(1) Insert 21:**

| h1 | h0 | | | | |
|---|---|---|---|---|---|
| 000 | 00 | 16 | | | |
| 001 | 01 | 5 | 25 | 9 | 13 |
| 010 | 10 | 10 | 14 | 2 | |
| 011 | 11 | 19 | 7 | 43 | 31 |
| 100 | 00 | | | | |

Next=1

| | | | |
|---|---|---|---|
| 21* | | | |

**(2) Insert 6:**

| h1 | h0 | | | | |
|---|---|---|---|---|---|
| 000 | 00 | 16 | | | |
| 001 | 01 | 5 | 25 | 9 | 13 |
| 010 | 10 | 10 | 14 | 2 | 6* |
| 011 | 11 | 19 | 7 | 43 | 31 |
| 100 | 00 | | | | |

Next=1

| | | | |
|---|---|---|---|
| 21* | | | |

**(3) Insert 30:**

| h1 | h0 | | | | |
|---|---|---|---|---|---|
| 000 | 00 | 16 | | | |
| 001 | 01 | | 25 | 9 | |
| 010 | 10 | 10 | 14 | 2 | 6 |
| 011 | 11 | 19 | 7 | 43 | 31 |
| 100 | 00 | | | | |
| 101 | 01 | 5 | 13 | | |

Next= 2

| | | | |
|---|---|---|---|
| 30* | | | |

**(4) Insert 74:**

| h1 | h0 | | | | |
|---|---|---|---|---|---|
| 000 | 00 | 16 | | | |
| 001 | 01 | | 25 | 9 | |
| 010 | 10 | 10 | 14 | 2 | 6 |
| 011 | 11 | 19 | 7 | 43 | 31 |
| 100 | 00 | | | | |
| 101 | 01 | 5 | 13 | | |

Next= 2

| | | | |
|---|---|---|---|
| 30* | 74* | | |

**(5) Insert 67:**

| h1 | h0 | | | | |
|---|---|---|---|---|---|
| 000 | 00 | 16 | | | |
| 001 | 01 | | 25 | 9 | |
| 010 | 10 | 10 | 74 | 2 | |
| 011 | 11 | 19 | 7 | 43 | 31 |
| 100 | 00 | | | | |
| 101 | 01 | 5 | 13 | | |
| 110 | 10 | 14 | 6 | 30 | |

Next = 3

| | | | |
|---|---|---|---|
| 67* | | | |

**Question 3. (20 marks)** Consider the following sales database:

- Customers (Name, Age, Gender, <u>CID</u>) having the size of 100,000 pages
- Purchases (CID, PID, Date, Location, Amount) having the size of 2,000,000 pages
- SalesCalls (CID, SID, Date, Result) having the size of 180,000 pages

The following is the workload estimation:

- 10% queries selecting on Customers.CID
- 30% queries selecting on Customers.Name
- 35% queries selecting on Purchases.PID
- 10% queries selecting on SalesCalls.SID
- 15% queries selecting on SalesCalls.Date

Suppose we can only build two indexes due to resource limitation. Detail a plan of building indexes with reasoning. You may assume that the indexes should have significant less cost than running a simple table scan.

**Solution:**

**Build indexes on Purhcases.PID and SalesCalls.Date**

**The savings obtained by building an index on an attribute is proportional to the number of pages in the corresponding relation multiplied by the number of queries using that index.**

**Purchases.PID is the most queried attribute. For the second index, we have to choose between Customers.Name and SalesCalls.Date. Of these two, Customers.Name is more useful since 0.3 x100000=30000> 0.15x180000=27000.**

**Question 4. (10 marks)** Consider the schedule S that consists of four transactions as follows: S = <T3_R(X), T4_W(X), T1_W(Y), T1_R(Y), T2_R(Y), T3_W(Z), T3_W(X), T2_R(X), T3_R(Z), T4_W(Z), T4_R(Z)>. The notation is self-explanatory. For example, T3_R(X) means that transaction T3 reads item X.

| T1 | T2 | T3 | T4 |
|---|---|---|---|
| | | R(X) | |
| | | | W(X) |
| W(Y) | | | |
| R(Y) | | | |
| | R(Y) | | |
| | | W(Z) | |
| | | W(X) | |
| | R(X) | | |
| | | R(Z) | |
| | | | W(Z) |
| | | | R(Z) |

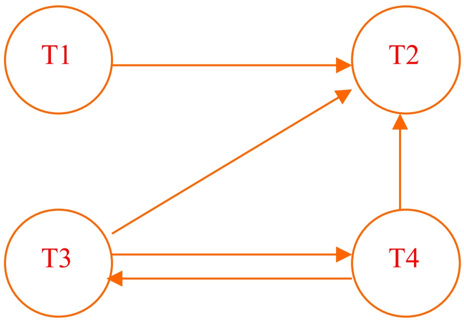
**(a)** Construct the precedence graph of S. Explain why or why not the schedule is conflict-serializable.

**(b)** If you find that S is **serializable** in (a), write down all equivalent serial schedules of S.

**Solution:**

**(a)**

**Precedence Graph of S**




**Cycle detected between T3 and T4→ Non Conflict-serializable.**

**(b)**

**No equivalent serial schedules**

**Question 5 (20 marks)** The relational database schema for an electronic company is given below:

Emp(<u>eid: integer</u>, ename: string, age: integer, salary: real)

Works(<u>eid: integer, did: integer</u>, pct_time: integer)

Dept(<u>did: integer</u>, dname: string, budget: real, managerid: integer)

Please answer the questions below with **SQL, Relational Algebra, and Tuple Relational Calculus**.

**(a)** Find out the names of all the employees that work in the department named "Human Resource".

**Solution:**

**SQL:**

**SELECT E.ename**

FROM    Emp E, Works W, Dept D

WHERE   D.dname = 'Human Resource' AND D.did = W.did AND W.eid = E.eid

RA:

$$\pi_{\text{ename}}((\pi_{\text{eid}}(\sigma_{\text{dname="Human Resource"}}\text{Dept} \bowtie \text{Works})) \bowtie \text{Emp})$$

TRC:

$$\{T \mid \exists E \in \text{Emp}\ (\exists D \in \text{Dept}\ (D.\text{dname} = \text{"Human Resource"} \wedge \exists W \in \text{Works}\ (W.\text{did} = D.\text{did} \wedge W.\text{eid} = E.\text{eid})) \wedge T.\text{ename} = E.\text{ename})\}$$

**(b)** Find out the names of the employees who make the highest salary.

**Solution:**

SQL:

```
SELECT   E1.ename
FROM     Emp E1
WHERE    E1.salary = (SELECT   MAX(E2.salary)
                      FROM     Emp E2)
```

RA:

$$\rho(\text{R1}, \text{Emp})$$

$$\rho(\text{R2}, \text{Emp})$$

$$\rho(\text{R3}(1 \to \text{eid}, 2 \to \text{ename}, 4 \to \text{salary}), \sigma_{\text{R1.salary} < R2.salary}(\text{R1} \times \text{R2}))$$

$$\pi_{\text{ename}}(\pi_{\text{eid,ename,salary}}\text{R1} - \text{R3})$$

TRC:

$$\{T \mid \exists E1 \in \text{Emp} \wedge \neg(\exists E2 \in \text{Emp}(E2.\text{salary} > E1.salary)) \wedge T.ename = E1.ename\}$$