# 浙江大学 2015－2016 学年秋季学期

## 《数据库系统设计》课程期末考试试卷

课程号：21120302，开课学院：计算机学院

考试试卷：√A 卷、B 卷（请在选定项上打√）

考试形式：闭、开卷（请在选定项上打√），允许带 1 张 A4 纸笔记 入场

考试日期：2015 年 11 月 13 日，考试时间：120 分钟

**诚信考试，沉着应考，杜绝违纪。**

**考生姓名：_____学号：_____所属院系：_____**

| 题序 | 一 | 二 | 三 | 四 | 五 | 六 | 七 | 八 | 总 分 |
|---|---|---|---|---|---|---|---|---|---|
| 得分 | | | | | | | | | |
| 评卷人 | | | | | | | | | |

## Problem 1: Variable-length Records (10 points)

Variable-length records arise in database systems in several ways. Following SQL statement creates a table ***person***, in which ***name*** and ***address*** are variable-length attributes. In addition, ***age*** and ***address*** could be empty.

```
create table person
(   id char(4) primary key,
    name varchar(20) not null,
    age smallint,
    address varchar(20) )
```

Following is an instance of ***Person*** table with 3 variable-length records.

| id | name | age | address |
|---|---|---|---|
| 1001 | Bill Gates | 60 | NULL |
| 1002 | Bob Williams | 50 | 111, State Street, MA. |
| 1003 | John Harvard | NULL | NULL |

1) Show by example how to implement the representation of variable-length records.
2) Show by example how to store variable-length records in a block with slotted-page structure.
   (Assume that smallint type is 2 bytes, and block size is 1024 bytes.)

## Problem 2: B+ -Tree (16 points, 4 points per part)

1) Construct a B+-tree (n=4 ) for the following bulk of index entries:
   (0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100).
   Assume that the tree is initially empty, and the bottom-up B+-tree construction method is used to bulk loading the above index entries.
2) For the constructed B+-tree, draw the form of the tree after operation "insert 55".
3) Assume that the B+-tree (n=4) contains 10000 index items, please estimate the height of the B+-tree.
4) Assume that the B+-tree (n= 4)) contains 10000 index items, please estimate the size (i.e. the number of nodes) of the B+-tree.

## Problem 3: Cost Estimation (18 points, 9 points per part)

Consider the following relational schema and SQL query:

account(account_no: char(10), customer_name: char(10),
branch: char(20), balance: integer)
access (serial char(10), account_no: char(10), access_date: date, amount: integer)
*note: account_no of table* ***access*** *references to table* ***account****.*

**select** account.account_no, account.customer_name
**from** account , access
**where** account.account_no = access.account_no **and** year(access_date)=2012
**and** branch='Hangzhou' **and** amount between 10000 and 89999.

1) Estimate the size (i.e. number of records) returned by above SQL statement.
2) Estimate the cost, in the best case, for evaluating **account** ⋈ **access** with Block Nested-Loop Join method.

**Assumptions:**

- account ***table*** has 10,000 records; ***access*** table has 100,000 records
- ***branch*** attribute has 50 distinct values.
- The value of a***ccess_date*** attribute is between '2011-01-01' and '2014-12-31'
- The value of ***amount*** attribute is between 1 and 1,000,000.
- The values of all attributes are uniformly distributed.
- The integer type needs 4 bytes.
- The date type needs 4 bytes.
- The file system supports 4K byte blocks.
- There are 100 buffer blocks available in memory for evaluating the join operation.

## Problem 4: Materialized View (10 points, 5 points per part)

A materialized view is a view whose contents are computed and stored. Materialized views are important for improving query performance in some applications. Consider following materialized view ***big_deal*** over the tables given in problem 3:

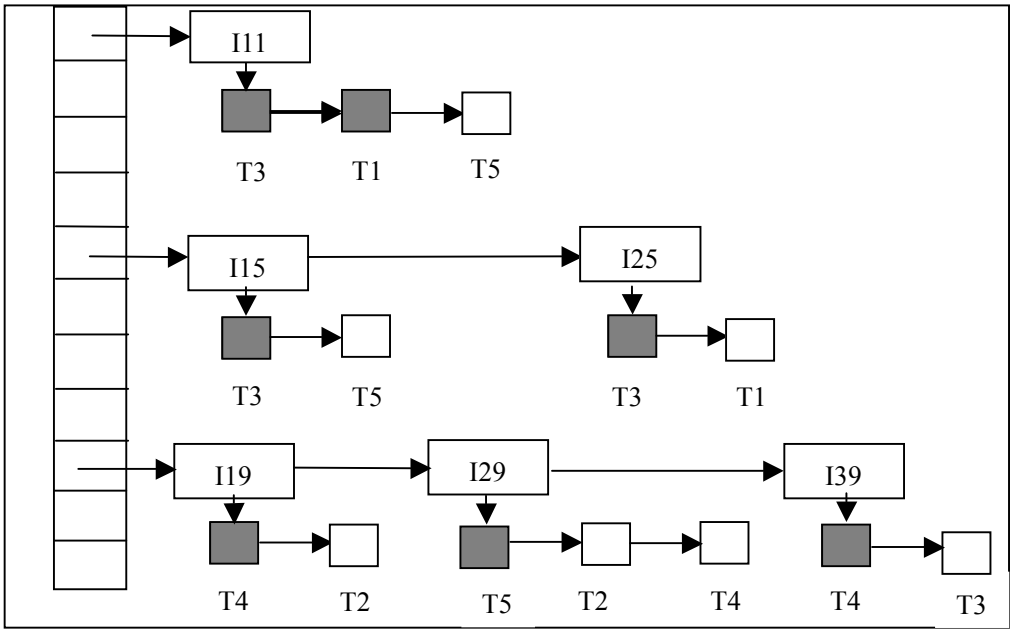
```
create materialized view big_deal(account_no, customer_name, access_date) as
select   account.account_no, account.customer_name, access.access_date
from     account , access
where    account.account_no = access.account_no and branch_name='Hangzhou'
         and amount >=50000;
```

1) Please identify a relational algebra expression that reflects the definition of the materialized view above.
2) If inserting a set ***S*** of records into the **access** table, explain how to incrementally maintain the materialized view ***big_deal***?

## Problem 5: Deadlock Handling (12 points, 4 points per part)

The following figure shows an instance of a lock table. There are 5 transactions (T1-T5) and 6 data items (I11, I15, I19, I25, I29, I39). Granted locks are filled (black) rectangles, while waiting requests are empty rectangles.

1) Which transactions are involved in deadlock?
2) In order to break the deadlock and release most lock resources, which transaction (victim) should be rolled back?
3) Please draw the lock table after the victim transaction of 2) is rolled back.



## Problem 6: Transactions (10 points)

In banking application, "Deposit & Withdraw" transactions have two operations:
**operation_A** updates the cash balance at a branch(a common item )
**operation_B** updates the individual account balance(a private item )
Explain how you can increase concurrency (and throughput) by ordering the operations of the "Deposit & Withdraw" transaction. Assume that all transactions follow the two-phase locking protocol.

## Problem 7: Crash Recovery (16 points, 4 points per part)

Consider the following log sequence of transactions T1, T2, T3, and T4. Supposing the system crashes just after the last log record. Please answer each of the following questions:

1) Which transactions should be redone? Which transactions should be undone?
2) What is the start point of redo phase? What is the end point of undo phase?
3) What are the values of A, B, C and D after recovery?
4) What log records are added to log file during recovery?

[1] <T1 **start**>
[2] <T1, A, 11, 22>
[3] <T2 **start**>
[4] <T2, B, 11, 22>
[5] <T3 **start**>
[6] <T3, C, 11, 22>
[7] <checkpoint { T1,T2,T3}>
[8] <T1, D, 11 ,22>
[9] <T2 **commit**>
[10] <T4 **start**>
[11] <T3, B, 22, 33>
[12] <checkpoint { T1, T3,T4}>
[13] <T1 **commit**>
[14] <T3, A, 22, 33>
[15] <T4, D, 22, 33>
[16] <T3 **commit**>

## Problem 8: Aries Recovery Method (8 points)

***Aries*** recovery method is widely used in industrial DBMSs. Please answer following questions about ***Aries***.

1) What contents are in the checkpoint log of ***Aries***?
2) Why checkpoint operation of ***Aries*** puts less side effects on normal transaction processing of DBMS?