Example Paper

THE UNIVERSITY OF SYDNEY

SEAT NUMBER: ....

STUDENT ID: ....

COMP9120
Relational Database Management Systems

Practice Questions Part 2
Semester 1, 2018

This examination paper consists of 6 pages

INSTRUCTIONS TO CANDIDATES

1. In a real exam these questions would take 62 minutes, plus 10 minutes reading time. You may find it helpful to time yourself to see how much you can do in this period.
2. You should try to do all calculations in your head or with a simple calculator.
3. The paper comprises 2 questions each with multiple parts. You should attempt both questions.
4. The mark available for each question is indicated beside the question heading. Each question consists of several parts, and the associated points are also indicated.
5. Answer all questions in the spaces provided on this question paper.
6. If you need more space to write, use additional paper and attach to the end.
7. Take care to write legibly. Write your final answers in ink, not pencil.

Question	Mark	Out of
Question 1		/23
Question 2		/28
Total		/51

Question 1: Transactions [23 points]

a) Give a brief explanation for each of the four ACID properties that should ideally be guaranteed by database transactions. **[8 marks]**

不记得Atomic的定义了

Atomocity
Consistency
I Isolation
D Durable

TEMP UPDATE - ~~COMMITED READ~~ READ COMMITED
INCORRECT SUM - REPEATABLE READ
LOST UPDATE - SERELIAZABLE SERIALIZABLE

b) Determine for each of the following schedules whether they are conflict-serializable or not.

(i) $r_1(X)$ $r_2(X)$ $w_1(X)$ $w_2(X)$ **[2 marks]**

not 2PL

X

(ii) $w_1(X)$ $r_2(Y)$ $r_1(Y)$ $r_2(X)$ **[2 marks]**

✓

(iii) $r_1(X)$ $r_1(Y)$ $w_1(X)$ $r_2(Y)$ $w_3(Y)$ $w_1(X)$ $r_2(Y)$ **[2 marks]**

X

如果 w3(Y) 能执行，说明它获得了 X-lock(Y)；

那就意味着其他持有 Y 的锁（T1、T2 **的** S-lock）必须已经释放；

但 T1 不能释放锁（Strict 2PL 要求它写过 X，必须等 commit）；

所以，如果 w3(Y) 真执行了，说明违反了 Strict 2PL；

c) Briefly explain Strict Two Phase Locking (Strict 2PL), and show whether the above schedules can be produced by the protocol. **[9 marks]**

Can only release lock after you no more add lock

Strict 2PL requires that all locks be held until the transaction commits or aborts. No lock can be released until the transaction ends

Question 2: Storage and Indexing [28 points]

4 16 230

A database contains the relation *Computers(comp_id, room_id, comp_desc).* Each record is 250 bytes, with *comp_id* using 4 bytes and *room_id* using 16 bytes. The relation holds 30,000 entries. The relation is stored with a sparse integrated B+ tree index on *comp_id*. A secondary B+ tree index is also constructed on *room_id*. Entries in both indexes are of the form <*search_key*, *rowid*>, where *rowid* pointers take up 4 bytes. The page size for the DBMS is 4096 bytes, with 100 bytes reserved for header data.

a) Estimate the number of pages required to hold the data described (including indexes), stating any assumptions you have made in your calculations. **[12 marks]**

page without head = 4096 - 100 = 3996

Data page
# of record/page = floor(3996/250) = 15
# of data page = ceil(30,000/15) = 2000

First index tree
index_size = 4 + 4 = 8
# of index/page = floor(3996/8) = 499
# of first layer index page = ceil(2000/499) = 5
# of second layer index page = ceil(5/499) = 1

Second index tree
index_size = 16+4 = 20
# of index/page = floor(3996/20) = 199
# of first layer index page = ceil(2000/199) = 11
# of second layer index page = ceil(11/199) = 1

In total
we have 2000 + 5 + 1 + 11 + 1 = 2018

老师说我的写法是正确的
Second index tree
index_size = 16+4 = 20
# of index/page = floor(3996/20) = 199
# of first layer index page = ceil(30000/199) = 151
# of second layer index page = ceil(151/199) = 1

In total
we have 2000 + 5 + 1 + 151 + 1 = 2158

b) The Computers relation is updated and now contains 50,000 records, distributed uniformly across 500 *room_id* values, stored in 4000 pages. The updated primary B+-tree index has 3 levels (including the root but excluding the leaf-level record pages). The secondary B+-tree index also has three levels and a fan-out of 150.

(i) Calculate the I/O cost to obtain the records in the following query via the most appropriate index: **[2 marks]**

SELECT * FROM Computers C WHERE C.comp_id =4;

(ii) Explain why the cost of the following query, again using the most appropriate index, will be higher than for the previous query: **[3 marks]**

SELECT * FROM Computers C WHERE C.room _id>=150 AND C.room_id<155;

c) A second relation *Users(emp_id, comp_id, acces_priv)* exists in the database, with *comp_id* a foreign key to the *Computers* relation. The relation consists of 100,000 records stored in 600 pages. It also has an unclustered index on *comp_id*, with three levels (again, including the root). *Users* and the updated *Computers* relation are used to find employees with access to computers in certain rooms with the following query:

SELECT U.emp_id

FROM Computers C, Users U

WHERE C.comp_id =U.comp_id AND C.room _id>=150 AND C.room_id<155;

(i) Estimate the I/O cost of evaluating the query with a block-nested join (using blocks of one page for each table) of the two relations, followed by in-memory selection and projection of the matching tuples. **[3 marks]**

(ii) Consider an alternative plan in which a range search first finds records from *Computers* with matching *room_id* values, followed by an index nested loop join with *Users* and in-memory projection of the results. Explain what factors affect the I/O cost of this plan, and how these differ to the previous plan. **[8 marks]**