Carnegie Mellon University
15-415 - Database Applications
Fall 2009, Faloutsos
Assignment 10: Concurrency Control

Due: 12/1, 1:30pm, in class - hard-copy please

Reminders

- Weight: **5%** of the homework grade i.e. **1.5%** of the course grade.
- Out of **100** points.
- Lead TA: B. Aditya Prakash

Notes:

- Rough time-estimate: 2∼4 hours (about 30-60mins per question).
- You are encouraged to **type** your answers. Illegible handwriting may get no points, at the discretion of the grader.
- Whenever you are making an assumption, please state it clearly.

Question 1: Serializability [15 points, 5 points each]

Consider the following schedules. The actions are listed in the order they are scheduled, and prefixed with the transaction name.

S1: `T1:R(X), T2:R(X), T1:W(Y), T2:W(Y), T1:R(Y), T2:R(Y)`
S2: `T3:W(X), T1:R(X), T1:W(Y), T2:R(Z), T2:W(Z), T3:R(Z)`

For **each** of the schedules, answer the following questions:

Q1.1 What is the precedence graph for the schedule?
Q1.2 Is the schedule conflict-serializable? If so, what are all the conflict equivalent serial schedules?
Q1.3 Is the schedule view-serializable? If so, what are all the view equivalent serial schedules?

Question 2: Two-Phase Locking [10 points, 5 points each]

Consider the following schedules:

S1: `T1:R(X), T2:W(Y), T2:R(X), T1:W(Y), T1:Commit, T2:Commit`

S2: `T1:R(X), T2:R(Y), T1:W(Z), T1:Commit, T3:R(Y), T3:R(Z), T2:W(Y), T3:W(X), T2:Commit, T3:Commit`

For **each** schedule, state which of the following concurrency control protocols allows it, that is, allows the actions to occur in exactly the order shown.

- 2PL
- Strict 2PL

Please provide a brief explanation for your answer...If YES, show where the lock requests could have happened; If NO, explain briefly.

Question 3: Deadlock Management [20 points]

Consider the following sequence of actions, listed in the order it is submitted to the DBMS (`S` is a shared lock, `X` is an exclusive lock):

S1: `T1:S(A), T2:X(A), T3:X(B), T1:X(B), T3:S(A)`
S1: `T1:X(A), T2:S(C), T1:S(B), T2:S(B), T3:X(B), T2:X(A)`

For **both** the sequences S1 and S2 given above:

Q3.1 Mention for each request whether the request is granted or blocked by the lock manager. [**4x2 = 8 points**]

Q3.2 Show the waits-for graph and indicate whether there will be a deadlock or not at the end of each sequence. [**4x2 = 8 points**]

Q3.3 If there is no deadlock, is there a lock or upgrade request by some transaction which can result in a deadlock? If yes, give such a request. If not, explain briefly. [**2x2 = 4 points**]

Question 4: Hierarchical Locking [20 points, 4 points each]

Consider the following database schema:

```
ACTOR (actor_name, actor_id, address)
PLAYSIN (movie_id, actor_id)
```

Suppose that the database is organized in terms of the following hierarchy of objects: The database itself is an object (D), and it contains two files (`ACTOR` and `PLAYSIN`). The `ACTOR` file contains 200 pages (`A1....A200`) while the `PLAYSIN` file contains 2000 pages (`P1....P2000`). Each page contains 100 records, and records are identified as `p:i`, where `p` is the page identifier and `i` is the slot of the record on that page.

Multiple granularity locking is used, with S, X, IS, IX and SIX locks, and database-level file-level, page-level and record-level locking. For efficiency reasons assume that if 60% of

the children require a certain lock, then the DBMS obtains that lock on the parent node itself (thus, it does not need repeating the same operation on the children).

For each of the following operations, indicate the sequence of **lock and unlock** requests that must be generated by a transaction that wants to carry out (just) these operations:

Q4.1 Read record `P600:4`.
Q4.2 Read pages `P10` through `P80` (included).
Q4.3 Read all pages in `ACTOR` and (based on the values read) modify 10 pages (refer to these 10 pages as `TENPGS`).
Q4.4 Update record `S10:40`.
Q4.5 Delete all records from both tables.

Question 5: B+ Tree Locking [15 points, 5 points each]

Consider the B+ Tree shown in Figure 17.5, Page 563 of the textbook and the *crabbing* algorithm from the lecture slides (Lecture 22 (file `23CC2.pdf`), Slide 36). Use `S(A)` to indicate that the transaction requests a shared lock on node `A` (similarly for `X(A)`, `U(A)`). Assuming that lock requests and lock upgrades are always granted, please give the sequence of lock/unlock requests for each of the following transactions:

Q5.1 Search for data entry 40*.
Q5.2 Insert data entry 62* into the tree.
Q5.3 Insert data entry 40* into the (original) tree.

Reminder: Make sure that the lock/unlock requests in your answer are given in the correct order.

Question 6: Crash Recovery [15 points, 5 points each]

Assume that the ARIES algorithm is implemented in the DBMS. Consider the following execution log:

LSN	log entry	prevLSN	undonextLSN
00	begin checkpoint		
10	end checkpoint		
20	T1 update Page P11		
30	T1 update Page P12		
40	T2 update Page P25		
50	T3 update Page P34		
60	T3 commit		
70	T2 update Page P25		
80	T2 update Page P23		
90	T2 abort		

Q6.1 Fill in the appropriate values in the `prevLSN` and `undonextLSN` columns.

Q6.2 Describe the actions taken to rollback transaction T2.

Q6.3 Show the log after T2 is rolled back, including all `prevLSN` and `undonextLSN` values in log records.

Question 7: Crash Recovery again [5 points]

Now consider the same execution log given in Question 6 and imagine there is a CRASH after LSN 90. Thus this is the log which the recovery manager will see. Which transactions will be redone and undone?