Chris Harms

DBMS HW 7

* 1. The balance will be $450, since the last transaction is the one that writes and commits a balance change the latest.
  2. If there was concurrency control, the balance would be $50.
  3. They would get stuck in a deadlock, since they both read at the same time placing a shared lock, then they both request exclusive locks, so they both get stuck waiting for each other to finish.
  4. No deadlocks will occur, since transaction 1 reads with a shared lock, and would request for an exclusive lock before the last transaction requests a shared lock to read, so transaction 1 will go through all the way, and transaction 3 will have to wait before it can read until transaction 1 is committed.
  5. The 3 transactions will all commit changes on separate copies of the database, when they’re compared, the DBMS will see that they will conflict and revert all changes, so nothing will change.
  6. You could combine the player and specialty tables into just a player table, you could combine team and manager into just team
  7. Create indexes on every primary and foreign key, and create partitions on tables where it applies, such as partitioning location based on population size.
     1. Create indexes on all the primary keys, create an index on teamid for players since that’ll be in the join, and create an index on team name since that’ll be in the where or group by clauses.
     2. Create an index on specialtycode in player since that’ll be in the join, and the usual indexes on the primary keys in the tables. You wouldn’t index the description since it’s a longer and messier field.
     3. Indexes on pk’s, an index on PlayerSpecialtyCode since its in the join, and an index on salary since it’ll be in the where clause.
     4. Indexes on pk’s, indexes on teamid in player, and teamlocation in team since they’ll be in joins between the team, player and location tables, and an index on cityname in location since it’ll be in a group by or where clause.
  8. You could partition specialty by different salary ranges based on the 3rd query.
  9. Transaction S will need to be redone, since part of it isn’t stored in the checkpoint that the database will get reset to, but the entire transaction will be stored in a log file.
  10. Transaction T will need to be undone since none of it is stored in the checkpoint and only part of it made it into a log file, so it’s always going to be incomplete
  11. Q and R aren’t affected since they’re completely in the checkpoint
  12. X will be 20, Y will be 5, Z will be 12, and W will stay 2