Midterm Spring 2020
Solutions & Rubrics

Q1 (3+3 = 6 points)

Consider a table, with several (2 or more) columns that contain repeating data values (within each column).

a. when is this acceptable? Explain, with an example.

A. It is acceptable when it is a bridge table, where redundancies cannot be avoided (eg. an 'Enrollment' bridge entity with a (StudentID,CourseID) PK where values can repeat in each column.

(+3 for correct answer which relative to not PK or before normalization or uniquely identifies, exist redundancies)

Partial Credit (+1.5) if no examples are provided.

b. when is this not acceptable? Explain, with an example.

A. Not acceptable when it is in the form of a 1NF table - here we have NEEDLESS repetition, which calls for the table to be segmented using normalization principles (eg. the 'Project' example from the lecture).

(+3 repeated rows/PK or after normalization, uniquely identifies)

Partial Credit (+1.5) if no examples are provided.

Q2 (5 points)

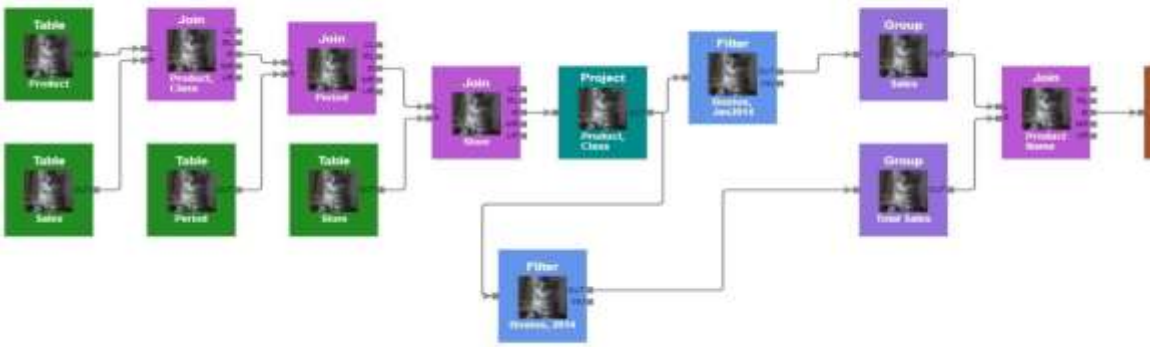
A 'shared lock' is granted to a set of transactions that all want to read data (unlike an 'exclusive lock' that is granted to a single transaction that wants to write data). Why is this even a thing, ie. why does reading (which causes no change to contents being read) require locking at all?

A. Because while the readings are occurring other transactions might write to the cells that being read, causing non-repeatable reads.

Rubric:
+5 points awarded for correct answer.
Answers related to dirty reads, non-repeatable reads, inconsistency retrieval are considered correct.
Partial Credits:
No points awarded for answers not related to data integrity
+2 points for not mentioning data inconsistency but explaining why shared locks are given to multiple read transactions
+4 points for mentioning data inconsistency without explaining read-write conflicts, (explanation must briefly explain the conflict or mention at least one error which can cause such conflicts or give an example explaining this conflict.)
+4 points for mentioning data inconsistency but mentioning exclusive lock (write lock) for a resource can be acquired during share lock.

Q3 (3+3 = 6 points)

The diagram below, shows a SQL dataflow graph. Users can construct non-trivial queries by dragging and dropping 'ops' (operators/nodes) on to a palette (graph area), and wiring them up by connecting inputs and outputs as shown - in other words, queries can be built up by chaining nodes, as shown (eg. in the graph shown, we are reading four tables using a 'Table' op, and performing joins using 'Join', projecting using 'Project', etc.).



a. what is the advantage of such visual construction, over coding queries by hand using SQL commands (like you did for HW2)?

A. The advantage is that non-programmers (casual users, business analysts...) can create SQL queries without having to write code.

(+3 for correct answer which is very similar to this)

(+1.5 partial credit if written some other advantages but not similar to the correct answer)

b. conversely, what are the advantages of being able to hand-code queries?

A. Hand-coding offers flexibility, control and power - queries that can't be visually constructed (eg ones with two or three levels of subquery nesting, correlated subqueries etc) can always be explicitly coded.

(+3 if correct answer(if talked about flexibility)

Partial credit +1.5 - if didn't talk about flexibility but some other advantages)

Q4 (3+3 = 6 points)

As you know, in 2PL, a transaction acquires locks during the locking phase, carries out the transaction, then releases locks during the unlocking phase (there is no interleaving of locking and unlocking).

a. what issue can arise, during the locking phase, how is it resolved?

A. Deadlocks can arise, on account of two or more transactions in the process of lock acquisition being unable to complete the step on account of mutual dependencies. Deadlock detection and mitigation strategies will help resolve the issue (the dependency cycles need to be broken).

Partial credit:(+1.5) for issues during locking phase. ie 2 or more processes competing for a lock. Partial credit:(+1.5) for correct solution to resolve deadlock. (break the deadlock state/dependency cycle).

-0.5 If solution does not contain the term- 'deadlock'

b. what issue can arise during the unlocking phase, how is that resolved?

A. During sequential unlocking, the transaction doing the unlocking might need to abort, in which case other transactions that acquired the released locks and started their own transaction operations will also need to be aborted ('cascading aborts'). To

resolve (prevent) this, we hold on to the locks until our transaction is committed (or aborted), and release the locks after (using a protocol called 'Strict 2PL').

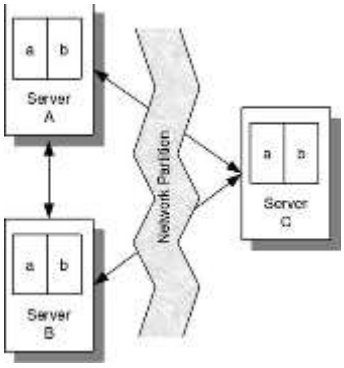
Partial credit: +1.5 For cascading aborts/rollbacks [or explanation for the situation]

+1.5 for any valid solution that prevents cascading rollbacks [cascadeless schedule, strict 2PL]

Q5 (3+2+1 = 6 points)

During a network partition (failure), a portion of a distributed DB (with replicated data stored in multiple nodes, and the cluster operating as a single unit) gets separated from the rest (eg. on account of a network switch failure).

In the figure below, a partition separates nodes A and B (which remain connected to each other), from node C - this will lead to AB, and C, operating as two independent DB copies, each of which can receive read/write requests.



Consider a write request arrives at node C.

Rubrics:

1. Full credit if answers are according to this solution.
2. Full credit if the student has used the words 'consistency' and 'availability' instead of 'accept' and 'reject'.
3. Partial credit (as per correctness of answer) if the student has used 2PC or do-undo-redo or distributed/remote request/access or failure/location transparency or push/pull, etc instead of this solution.
4. Anything other than this -- TO BE DECIDED as per the student's answer.
5. Many students have given answers like -- 'Partition tolerance from CAP' instead of writing just CAP -- that's ok; give full credit.
6. If only mentioned any one of them, then partial credit. -0.5

a. what are the two options that C would choose from?

A. C can accept (perform) the write operation, or it can refuse the request.

1.5 points for each option

b. what are the implications of choosing either option?

A. If C carries out the write, we will have an inconsistency (between C, and AB) - we are prioritizing availability; if we refuse, the transaction requesting the write would need to be aborted and possibly retried - we are prioritizing consistency.

1 point for an explanation of implication for each option

c. what distributed DB principle/idea/'law'/theorem are we referring to?

A. The CAP Theorem.

partial credit 0.5 if only giving any among C (consistency),A (availability)

or P (partition tolerance)

Q6 (6 points)

Consider the following table (PrinterControl), which is used to assign specific printers to named users as well as guests, in a workgroup (the first three entries are named users):

```
PrinterControl
user_id_start    user_id_finish   printer_name
printer_description
=================================================
'chacha'   'chacha'        'LPT1'   'First floor's printer'
'lee'      'lee'           'LPT2'   'Second floor's printer'
'thomas'   'thomas'        'LPT3'   'Third floor's printer'
'aaaaaaaa' 'mzzzzzzzz'     'LPT4'   'Common printer #1 '
'naaaaaaa' 'zzzzzzzz'      'LPT5'   'Common printer #2'
```

Explain what the following query does, when the :my_id variable can contain a variety of userIDs, of named users and guests (eg. it could contain 'lee' or 'archit' or 'yiming') - the BETWEEN keyword returns a boolean if a given string lexicographically (alphabetically) lies between two others. Be very specific in your explanation (eg. be sure to explain why we use MIN()).

```
SELECT MIN(printer_name)
  FROM PrinterControl
 WHERE :my_id BETWEEN user_id_start AND user_id_finish;
```

A. The query assigns (selects) a printer, based on incoming username - if the user is chacha or lee or thomas, the assigned printer is LPT1, LPT2 or LPT3, respectively; for all other users, LPT4 is assigned if their username lies in a..m, or LPT5 otherwise (n..z). The MIN operation selects LPT1 instead of LPT4 for chacha (and likewise for the other two named users).

Rubrics:

1)Full credit if answer is according to the solution mentioned.

2)Split up:

+2 points if the student has mentioned that the query assigns printer based on username.(Give marks if the student has given a general idea of what the query does.)

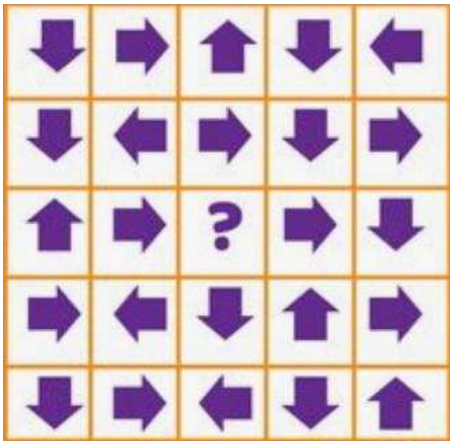
+2 points if the student has explained how each of the printers get assigned based on the username.

+2 points if the student has mentioned the role of min operation

Bonus (1 point)

Note - this bonus is optional - if you do get it right, you get 1 point, which is counted only when you don't have 35/35 already :) In other words, the max you can get for the whole test is 35, not 36.

In which direction should the missing arrow point (there is only ONE right answer :))?



A. Down - starting from top-left, follow this spiralling path where 'Down-Right-UpDown-Left-Right' repeats: