CSC 343 Fall 2019
Group Assignment 3
University of Toronto Mississauga
Due: Wednesday December 4th, 2019 by 11:59pm

Michael Liut

November 12, 2019

I. Database Design (25 marks)

Question 1 (10 marks)

Dr. Stephen Strange has recently been appointed by the Ancient One as the Professor responsible for coordinating all classes on Kamar-Taj. You have just been contracted by Dr. Strange to design a better schema for its current `Schedule` relation. The specifications of this schema are as follows:

```
Schedule (course, professor, teaching assistant, location, day, time)
```

where:
`course` represents the course code, `location` represents the room the course is held in, `day` represents the day of the week the course is offered, `time` represents the time of day the course is held, `professor` represents the name of the course instructor, and `teaching assistant` represents the name of the Head Teaching Assistant of the course.

From the `Schedule` relation, you are required to represent the following additional information:

a) No professor can be assigned to teach two (or more) courses on the same day and time.

b) There is at most one teaching assistant per course.

c) No two (or more) courses can be assigned to the same location on the same day and time.

d) No two (or more) professors can be assigned to the same location on the same day and time.

e) The combination of course, day, and time will uniquely determine what professor is teaching.

Answer the following questions:

1. Given the additional information, `a` – `e`, list all of the functional dependencies that can be inferred. [5 marks]
2. Dr. Strange will only be satisfied if your design is a good one (i.e. the schema satisfies either the 3NF or the BCNF). Is the design of your schema with the functional dependencies from part (1), above, a good one? Justify your answer. If the design is not a good one, provide a better one, using one of the decomposition algorithms discussed in class. [5 marks]

Question 2 (5 marks)

Given the following relation `R`:

A	B	C
a_1	b_1	c_1
a_1	b_1	c_2
a_2	b_1	c_1
a_2	b_1	c_3

1. List all of the functional dependencies that this `Relation Table` satisfies. [3 marks]
2. Now let's modify attribute **C**'s last record from c_3 to c_2. What functional dependencies, if any, from question (2) part (1), above, have been changed? List them. Explain your reasoning in no more than two sentences. [2 marks]

Question 3 (10 marks)

Consider a relation schema `R` with a set of attributes $\alpha = \{A, B, C, D, E, F, G, H\}$ and the set of functional dependencies $\mathcal{F} = \{\ A \rightarrow B,\ ABD \rightarrow FGH,\ AEH \rightarrow BD,\ BC \rightarrow EH,\ C \rightarrow ACG,\ C \rightarrow AFH,\ DE \rightarrow HB,\ DF \rightarrow AC,\ E \rightarrow F,\ H \rightarrow EA\}$

(a) [7 marks] Find all candidate keys (i.e., minimal keys) of relation `R`. You must show your work and **clearly state** which of Armstrong's axiom(s) were used to derive each key.

(b) [3 marks] True-or-False, prove by validity, or disprove by counter-example.

  i. Given `R`, with α and $\mathcal{F}$, a 3NF decomposition would result in the same candidate keys found in (a).

  ii. Given `R`, with α and $\mathcal{F}$, there can only be one unique 3NF decomposition.

II. Transactions and Concurrency (25 marks)

Question 4 (10 marks)

Consider the following classes of schedules: *serializable*, *conflict-serializable*, *view-serializable*, *recoverable*, *avoids-cascading-aborts*, and *strict*. For each of the following schedules, state which of the preceding classes it belongs to. If you cannot decide whether a schedule belongs in a certain class based on the listed actions, explain briefly.

The actions are listed in the order they are scheduled and prefixed with the transaction name. If a commit or abort is not shown, the schedule is incomplete; assume that abort or commit must follow all the listed actions.

1. T1:W(A), T2:R(B), T1:R(B), T2:R(A)
2. T1:R(A), T2:W(A), T1:W(A), T2:Abort, T1:Commit
3. T1:W(A), T2:R(A), T1:W(A), T2:Abort, T1:Commit
4. T2:R(A), T3:W(A), T3:Commit, T1:W(B), T1:Commit, T2:R(B), T2:W(C), T2:Commit
5. T1:R(A), T2:W(A), T2:Commit, T1:W(A), T1:Commit, T3:R(A), T3:Commit

Answers are to be selected in the table below; you are to use a check mark "✓" to identify which desirable properties are guaranteed and an "**X**" to identify the ones which are not guaranteed. For those that cannot be determined mark them with a "**?**" and add an explanation to the side.

Question \ Property	Serializable	Conflict-Serializabile	View-Serializabile	Recoverable	Avoids Cascading Aborts
1.					
2.					
3.					
4.					
5.					

Question 5 (15 marks)

You will simulate concurrent database queries, using two different database connections, against the `Accounts` table (run `createTable.ddl` prior to starting this).

You will open two terminal sessions, and execute the following commands to prepare your environments. For simplicity we will call the two sessions, **A** and **B**.

In Session A

1. Invoke MySQL on the CSC server using `mysql -u utorID -p -h 142.1.207.11 utorID_343`. Then enter the command `SET autocommit = 0;`. This command turns off the AUTO COMMIT feature in MySQL. You will now be at the MySQL command line.
2. Verify AUTO COMMIT is turned OFF by running the command `SHOW VARIABLES WHERE Variable_name='autocommit';`. You should see the value for "autocommit" as "OFF" in the second entry of the table.
3. Change the transaction isolation level to to 'Repeatable Read' by running the command `SET SESSION TRANSACTION ISOLATION LEVEL REPEATABLE READ;`. Transaction isolation levels define the degree of access and interaction among a set of concurrent transactions which operate against the same data. Please see the MySQL reference for more information on isolation levels in MySQL.
4. Verify your isolation level change by running the command `SHOW VARIABLES WHERE Variable_name='tx_isolation';`.

In Session B

1. Invoke MySQL on the CSC server using `mysql -u utorID -p -h 142.1.207.11 utorID_343` (do not turn AUTO COMMIT off).
2. Change the transaction isolation level to 'Repeatable Read' by running the command `SET SESSION TRANSACTION ISOLATION LEVEL REPEATABLE READ;`.
3. Verify your isolation level change by running the command `SHOW VARIABLES WHERE Variable_name='tx_isolation';`.

Transactions

Run the following commands (in the given order) and provide your answers to the stated questions:

1. In **Session A**, insert the record `('iron', 'Iron Man', 6000.00)`, followed by a `select * from Accounts`.
2. In **Session B**, run `select * from Accounts`. Is the output you get the same or different than in (1)? Why did this occur? What is a possible solution?
3. Return to **Session A**, and implement your solution from the previous step.
4. In **Session B**, do the SELECT * query again (to list all records). Provide your output.
5. We will update Accounts from two different transactions. In **Session A**, change the isolation level to READ COMMITTED. In **Session B**, change the isolation level to READ UNCOMMITTED. (You will have to disconnect and reconnect to the database.)
6. In **Session A**, transfer $750 from Thor's account into the Hulk's account.
7. In **Session B**, list all Accounts and their balances. Then issue a $300 transfer from Black Widow's account to the Hulk's account. What happens and why?
8. Commit the transaction in **Session A**. What is the Hulk's balance now?
9. Change the transaction isolation level in **Session A** to READ UNCOMMITTED.
10. Now, in **Session A**, transfer 80% of the Hulk's account balance to Captain America's account. Commit the transaction.
11. In **Session A**, transfer 50% of Iron Man's funds to Captain America's account. In **Session B**, list all data records. What is Captain America's balance? Does it reflect the latest transfer from Iron Man? Based on your transaction in step (7) and this step, what can you say about the allowed actions in READ COMMITTED and READ UNCOMMITTED isolation levels? How do these compare with the REPEATABLE READ isolation level?
12. Abort the Iron Man to Captain America transfer transaction in **Session A**, by executing the command `rollback;`.
13. In **Session A and B**, list all data records. What are the final balances for each user?

<u>**Please Note**</u>: questions 2, 4, 7, 8, 11, and 13 require written responses. Additionally, for question 13, you must display the tables for both sessions.

Grading

This is a group assignment to be completed in the pairs (i.e. a team of 2 people); selected in Assignment 0 unless advanced written approval has been given by the Course Instructor. This assignment is worth 10.0% of your final grade in this course and will be graded out of 50-points.

Submission

All files are to be submitted using the MarkUs platform (https://mcsmark.utm.utoronto.ca/csc343f19/). Only one person from each group is required to submit the files.

Please ensure your answers are typed and submissions are clearly legible. Include your, and your partner's, full name and student ID number in all files.

Upload one files with the indicated file extensions (no compression based `.tar`, `.zip`, `.rar` files). Submit your solutions to all questions in a file called **`A3.pdf`**.

Please note that late assignments will be docked 20% per day of lateness and after three (3) days, the assignment will no longer be accepted.

Plagiarism

Please refer to the course outline and introduction slides. To serve as a reminder: Turnitin will be used for all written work and MOSS for all code submissions. UTM's policy on Academic Integrity: http://academicintegrity.utoronto.ca/