Project #2 (Part 2)

CSC 261/461 (Database Systems), Spring 2018,
University of Rochester
Due Date: 03/29/2018 (11:59 pm)
This is NOT a group project

Introduction

This part of the project will make use of MySQL triggers to monitor and maintain the integrity of your AuctionBase data. You will also add a "current time" feature to your AuctionBase database. Your project must be implemented in MySQL and validated on the betaweb machines.

Task A: Adding Support for Current Time

The original auction data that we provided for you in XML, which you translated into relations and loaded into the database in Part 1, represents a single point in time, specifically, one second after midnight on December 20th, 2001 (represented in MySQL as 2001-12-20 00:00:01). To fully test your functionality, and to simulate the true operation of an online auction system in which auctions close as time passes, you should maintain a fictitious "current time" in your database. First, add a new one-attribute table to your database that represents this current time. For Consistency, let's name this table and the attribute CurrentTime and timeNow.

This table should at all times contain a single row (i.e., a single value) representing the current time of your AuctionBase system.

For starters, the table should be initialized to match the single point in time we've previously mentioned: 2001-12- 20 00:00:01. To do this, modify your create.sql from Part 1 by adding the necessary SQL commands to create and initialize your `CurrentTime` table. Also, to make sure that you've initialized everything correctly, include a SELECT statement that reads the current time of your AuctionBase system. Your `create.sql` should now include the following:

```
DROP TABLE if exists CurrentTime ;
CREATE TABLE CurrentTime (timeNow DATETIME NOT NULL) ;
INSERT INTO CurrentTime values ('2001-12-20 00:00:01') ;
SELECT timeNow FROM CurrentTime ;
```

Task B: Adding Constraints and Triggers to Your Schema

For this task, you may want to refer to the MySQL documentation for the CREATE TRIGGER and DROP TRIGGER statements `https://dev.mysql.com/doc/refman/5.7/en/trigger-syntax.html`.

To ensure that the AuctionBase system at a given point in time represents a correct state of the real world, a number of real-world constraints are expected to hold. In particular, your database schema must adhere to the following constraints:

Primary Key and Foreign Key constraints

- No two users can share the same User ID.
- All sellers and bidders must already exist as users.
- No two items can share the same Item ID.

- Every bid must correspond to an actual item.
- An item cannot belong to a particular category more than once.
- All sellers and bidders must already exist as users.
- No user can make a bid of the same amount to the same item more than once.

- All this constraints can be imposed using Primary Key and Foreign Key constraints. If you have not done this already in Project 2 Part 1, please modify `create.sql` file to incorporate these constraints.

Constraints achieved by Triggers (You need to create Triggers)

Constraints for Bidding

1. A user may not bid on an item he or she is also selling.
2. No auction may have two bids at the exact same time.
3. No auction may have a bid before its start time or after its end time.
4. No user can make a bid of the same amount to the same item more than once.
5. Any new bid for a particular item must have a higher amount than any of the previous bids for that item.
6. All new bids must be placed at the time which matches the current time of your AuctionBase system. item.

Constraints for Items

7. The Current Price of an item must always match the Amount of the most recent bid for that item. (Note: This can be updated when a new bid for that item is inserted)

Constraints for Time

8. The current time of your AuctionBase system can only advance forward in time, not backward in time.

- We are only concerned about two trigger events, when a new bid is inserted, and when the current time gets updated. The current version of MySQL does not support two separate triggers for the same event at same trigger time. (We can have two different triggers for the same event if the trigger time is different. For example, Trigger for Constraint 7 should only get activated after a successful entry of a Bid). For this part, you need to create three (3) triggers to implement this 8 constraints.
- Also, this version of MySQL does not support SIGNAL command. A workaround for this is 'call'ing a function that does not exist, and the name of this 'non-existing' function gives us clue about the error!

To make it simple for you, we are providing you a sample trigger for the following constraint:
The end time for a bid must always be after its start time.

```
# Constraint: The end time for a bid must be after its start time.
DROP TRIGGER IF EXISTS item_check_before;

DELIMITER //
CREATE TRIGGER item_check_before BEFORE INSERT ON Item
FOR EACH ROW
BEGIN
IF NEW.ends < NEW.started THEN
CALL `Error0:The end time must be after its start time.`;
END IF;
END;//
DELIMITER ;
```

Create a file, `create_triggers.sql` which should contain code for creating triggers to satisfy all these constraints. Please make sure you drop the existing trigger before creating a new one (or creating after modification)

For automated grading, we mandate the following naming scheme for the triggers:

- **before_insert_bid**
- **after_insert_bid**
- **before_update_time**

Also, Number the Error Messages from 'Error1:< description of the error>' to 'Error8:<description of the error>' for Constraints 1-8. Note, You do not need to have any 'Error7: ...' as we are only updating a field in another table.

Task C: Test the triggers

Run some sample queries after creating the triggers to see if the triggers are reporting errors or updating fields as expected.

It is possible to create all the files, load them and then create all the triggers using the source command as we have done in Project 2 Part 1. For example,

```
mysql> source create.sql
mysql> source load.sql
mysql> source create_triggers.sql
```

For avoiding any deduction your submission must need to follow the following guidelines:

- 'timeNow' is the only attribute in your CurrentTime relation which holds the value '2001-12-20 00:00:01' in DATETIME format. The code for creating and loading the relation is in `create.sql` file.
- All the Primary Key and Foreign Key constraints hold.
- All the triggers in `create_triggers.sql` work.
- You are numbering the error messages based on the constraint numbers (from 1 to 8) and your error messages are descriptive.
- You are following the naming convention for the triggers.
- The constraints are numbered based on their priority. For any query, if two constraints i and j are violated, then Error$i$ should be reported if $i < j$.

Once all the requirements are satisfied, follow the submission instruction.

Submission instructions

To submit Part 2 of the project, first gather the following files in a single submission directory `proj2p2`:

```
create.sql
create_triggers.sql
```

Now, create an archive file (with .tar extension) for submission.

```
cd ..
tar -cvf proj2p2.tar proj2p2
ls
```

Submit the tar file using the command:

```
 ~csc261/submit p2p2 proj2p2.tar
```

You should get a feedback whether the submission is successful or not.

You may resubmit as many times as you like; however, only the latest submission (along with the timestamp) will be saved, and we will use your latest submission for grading your work. Submissions via email will not be accepted! No late submissions allowed.

Acknowledgement

Adapted from a project given in CS145 offered by Stanford University.