CECS 323 LAB TRANSACTIONS DATAGRIP

OBJECTIVE: Get some first-hand experience with transactions and isolation levels.

INTRODUCTION: The Derby and MySQL databases default to performing a commit after each DML statement. This behavior helps to make sure that you do not lose data if the database goes offline unexpectedly, but it also means that you do not have a choice on the boundaries of your transactions. In this lab, we are going to show you how the database management system isolates transactions from each other so that they cannot interfere with each other.

See: https://db.apache.org/derby/docs/10.6/devguide/cdevconcepts15366.html for a discussion regarding locking and concurrency control in the Derby database.

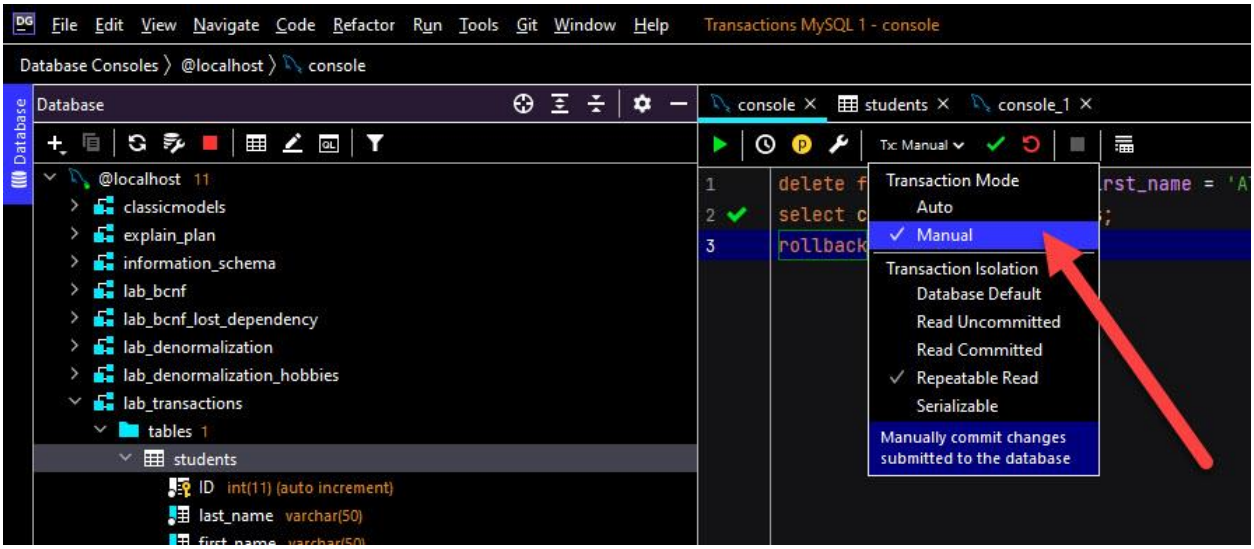
By contrast, the MySQL engine that we will default to uses multi-version concurrency control. A good article on that can be found at: https://medium.com/@ajones1_999/understanding-mysql-multiversion-concurrency-control-6b52f1bd5b7e. The wonderful thing about multi-version concurrency control is that readers never block another transaction and writers never block readers.

PROCEDURE:

Derby database:

1. In DataGrip, we can make several connections to the same Derby database, from the same embedded client, just by using two consoles within the same project.
    a. Open up DataGrip, go to the File | New | Project menu item and create a project called "Transactions".
    b. In the Database pane, select the "+" sign and create a connection to your lab Derby database.
    c. Right-click the connection object in the Database window and select "new schema" and create a schema called "Transactions".
2. At this point, you have one console window open into your database. Now you need to open a second console, that is connected to the **same** Derby database.
    a. Right-click the connection icon in the Database window.
    b. Select New | Query Console.
    c. From here on out, we will refer to window 1 and window 2. Window 1 is the first query console that you established, window 2 is the second.
3. Then in the **first transaction window**, run the SQL statements that you will find here.
    a. You can cut/paste those statements into your console window.
    b. Or you can run them by putting them into a .sql file and using the File | Run option from the menu.
4. In **both** windows we need to set the transaction isolation level and turn off the autocommit.

CECS 323 LAB TRANSACTIONS DATAGRIP



5. Then, using the same control, select the Repeatable Read option under Transaction Isolation. This will protect you from dirty reads and non-repeatable reads, but not from phantom reads.
6. Fill out the following table. Designate one of your Derby ij sessions as Transaction 1, and the other as Transaction 2. Perform the following commands in the designated transaction window, and in the specified order.
    a. In each case, look at the statement and try to predict what might happen.
    b. Put your prediction (theory) into the proper cell in the spreadsheet.
    c. If something different happens, specify that in the "Actual Results" column.

Remember, some of these actions will cause a lock to occur. Give it a minute or two for your transaction to time out, note that, and move on. **Do not abort any** of your transactions.

Step #	Transaction 1	Transaction 2	Expected Results	Actual Results
1.		SELECT COUNT(*) FROM students;		
2.		SELECT AVG(GPA) FROM students;	3.4124999999999996	
3.	INSERT INTO students (last_name, first_name, major, year_study, GPA) values ('Piggy', 'Miss', 'drama', 'senior', 3.99);			
4.		SELECT COUNT(*) FROM students;[1]		If you do **not** get a timeout

[1] This will take some time, be patient. The reason will become clear soon enough.

				waiting for a lock, let me know.
5.		SELECT COUNT(*) FROM students where last_name = 'the Frog';		
6.		select avg(GPA) from students;		
7.		SELECT COUNT(*) FROM (SELECT last_name FROM students WHERE last_name IN ('the Frog', 'Einstein')) abc;		
8.	commit;			
9.		SELECT AVG(GPA) FROM students;		
10.		commit;		
11.		SELECT AVG(GPA) FROM students;		

MySQL:

1. Either connect to the campus MySQL instance, or a local MySQL instance on your laptop. We will again create a students table, so be sure that will not conflict with an existing table in your schema. Use the DDL found here to build it in MySQL, it is slightly different from the DDL in Derby. That same file also has an insert statement to put in a little sample data.
2. If you are using MySQL workbench, you can easily create two connections to the same MySQL database. Just create the first connection as you always do, then push the home button in the upper left-hand corner and create a second connection. There will now be two tabs at the top of your MySQL window, one for each session. Each tab will have its own set of open SQL windows.
3. In **both** tabs, issue the statement "set transaction isolation level **repeatable read**;" to make sure that you have the same isolation level that you did in Derby. The repeatable read transaction isolation level is the default in MySQL for the InnoDB database engine, which is the default engine.
4. Run the following statements in the order and transaction indicated and fill out the following table, just as you did for Derby:

Step #	Transaction 1	Transaction 2	Expected Results	Actual Results
1.	START TRANSACTION;			
2.		START TRANSACTION;		

3.		SELECT AVG(GPA) FROM students;		
4.		SELECT COUNT(*) FROM students;		
5.	INSERT INTO students (last_name, first_name, major, year_study, GPA) values ('Piggy', 'Miss', 'drama', 'senior', 3.99);			
6.		SELECT COUNT(*) FROM students;		
7.		SELECT COUNT(*) FROM students WHERE last_name = 'the Frog';		
8.		SELECT AVG(GPA) FROM students;		
9.		SELECT COUNT(*) FROM (SELECT last_name FROM students WHERE last_name IN ('the Frog', 'Washington')) abc;		
10.	COMMIT;			
11.		SELECT AVG(GPA) FROM students;		
12.		COMMIT;		
13.		SELECT AVG(GPA) FROM students;		

WHAT TO TURN IN:

- Your Word document showing your expectations and the results from each of the above statements for both Derby and MySQL.
- Your team's Lab Collaboration Document. You can find the template for that at BeachBoard | Content | Student Helps | Lab Collaboration Document.