ECE356 - Database Systems
Lab 3 - More exercises on SQL Queries, Stored Procedures, Transactions and DB Performance
Fall 2019

Due Date: **November 18, 2019 @ 11:59PM**

Individual work policy: In this lab, students are expected to work on their own.

In this lab, we provide you with more practice on SQL queries and ask you to practice writing and invoking stored procedures. In addition, you will practice writing a transaction that completes or rolls back updates to the database depending on a certain condition. Finally, we ask you to improve the performance of a database by examining the current query plans, and adding indexes that may be needed.

Part 0 of this lab is not required for submission. However, it is recommended to go through this part because it gets you familiar with using SELECT statements in large scale datasets. In addition, this part helps you know the procedure of loading the database you will use in parts 1 to 3. The latter parts (1-3) are required for submission.

The questions start on the next page.

Part 0: Load Yelp database *[Required]* and Run Queries *[Optional]*

Important Note: Before continuing reading this part, you **must** accept the license as stated in https://www.yelp.com/dataset/documentation/faq. To accept this license, please visit https://www.yelp.com/dataset/download.

Yelp is a website that maintains consumer reviews of businesses. Its data is very similar to that within a data warehouse: a small number of tables with a very, very large quantity of data. Yelp regularly provides access to a small subset of that data together with a schema for it (https://www.yelp.ca/dataset_challenge).

Your task is to write the SELECT statements to answer the questions proposed here. This task is not graded, but it is provided to get you familiar with SELECT statements at large scale and help you know the procedure of loading a large dataset such as Yelp.

In order to reduce the load and running time of the queries for this exercise, we will use a reduced version of the Yelp database, which contains only 20% of the reviews in the original database. This database will be created on your own *personal machines*. To ensure consistency among students, save you time and avoid all the hassle of creating table attributes, we simplify the process by providing you a MySQL script titled as *load_yelp_db_small* that loads the data for you using the LOAD DATA method. Inside this MySQL script, you will find MySQL commands you need to import the Yelp dataset for your reference. Note that, along with this MySQL script, you will be provided with six CSV files containing the filtered data (both the MySQL script and CSV files can be downloaded from LEARN). To import the data using the provided script, simply follow the steps below:

1. Download the CSV files and the MySQL script *load_yelp_db_small* and save them on your computer.
2. Open the MySQL script *load_yelp_db_small* and go to lines 77, 264, 285, 312, 335, and 373.
3. On these lines, insert the path to the corresponding CSV file where you saved it (make sure to add .csv at the end of the path).
4. Run the script and wait for the database to be created.

After the database is created, you can check if the data was properly loaded by executing simple query statements. Examples of such queries can be found commented in the provided MySQL script "*load_yelp_db_small*". Search for these queries, copy them, and execute them if you wish. You can also use your own queries if you like.

Create SQL queries to answer the following questions:

(a) What is the id and review count of the user whose name is 'Shila'?

(b) Which business has received the greatest number of reviews? (provide the business id, business name, and review count)

(c) Find the average number of reviews of all users in the database.

(d) The average rating written by a user can be determined in two ways:

   1 By direct reading from the Users table "average stars" column

   2 By computing an average of the ratings issued by a user for businesses reviewed

   For how many users is the difference between these two quantities larger than 0.50?

(e) What fraction of users have written more than 10 reviews?

(f) For each user who wrote more than 10 reviews, what is the average number of characters of these reviews?

Part 1: Stored Procedures

Start with the normalized Employee database shown in Figure 1. You should use *createTables.sql* MySQL script to create the appropriate tables and load the necessary data.

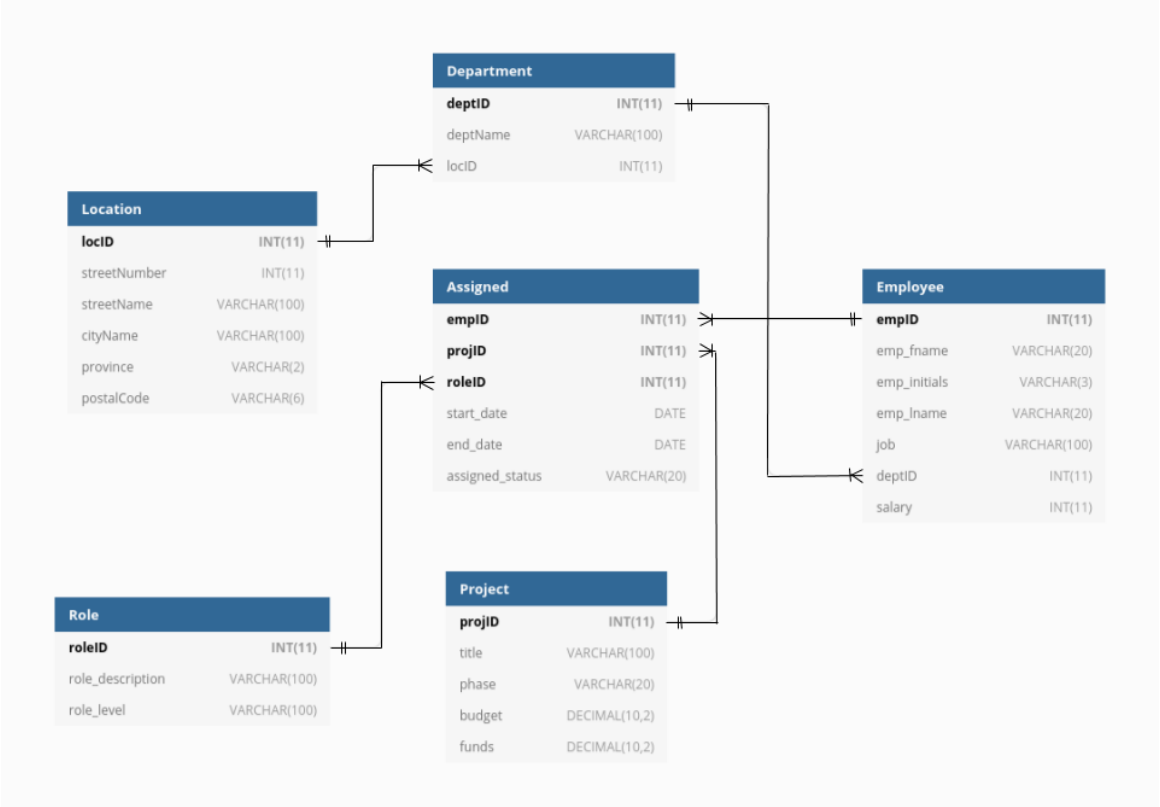

Figure 1: Normalized Employee Database Schema

Write a stored procedure called **"sp_pay_raise"**. The procedure should try to raise the salary of a given employee by a given percentage and return an error code associated with this operation. The percentage specified must be greater than 0% and less than 10%. If the percentage is not within the valid range, the salary is not increased.

Tables 1 and 2 show the input parameters for this procedure, and the error codes that must be returned under different conditions.

Table 1: Parameters

Input/Output	Name	Data type
Input	inEmpID	int
Input	inPercentageRaise	Double(4,2)
Output	errorCode	int

Table 2: Error Code Values

Error Code Value	Condition
-3	Employee does not exist AND the percentage specified is outside the range between 0% and 10% inclusive.
-2	Employee does not exist AND the percentage specified is greater than 0% or less than 10% Salary is not raised.
-1	Employee exists AND the percentage specified is outside the range between 0% and 10% inclusive.
0	Salary was raised. No error

Table 3 contains examples of sample outputs that you can use to verify your procedure. Please make sure that you run your procedure with different test cases.

Table 3: Example Input/Output Pairs For the Given Stored Procedure

inEmpID	inPercentageRaise	errorCode
12	5	-2

Please note that your stored procedure will always be evaluated on a fresh database.

Save your stored procedure in the provided *part1.sql*.

Part 2: Transaction Management

Start with the normalized Employee database given in Lab 2 and shown in Figure 1. After finishing Part 1, please make sure that you reload all your tables using the provided *createTables.sql* SQL script.

Create a transaction within a stored procedure that will raise the salary of all employees in Kitchener by 3%, provided that no salaries of any employee is more than $50,000 after being raised. If any salary after the raise in Kitchener exceeds $50,000, then the pay raise is cancelled for all employees — that is, the transaction is rolled back (see COMMIT/ROLLBACK commands). If the transaction should be committed, you should return 1 in your stored procedure. Otherwise, you should return 0.

No credit will be given if the solution does not use a transaction to perform/rollback the raise. Please note that your solution will always be evaluated on a fresh database.

Save your answers in the provided *part2.sql*.

Part 3: Database Performance

The ***explain*** command determines the query execution plan for your query. The query plan will show you where primary keys, foreign keys or additional indexes are used and allow you to determine where they may be needed (i.e. you will find out where a sequential search is used since there is no index yet).

In this part of the lab, you will improve the performance of two queries to the Yelp database on your personal machine. Please make sure that you set up MySQL on your personal machine.

In order to reduce the load and running time of the queries for this exercise, we will use a reduced version of the Yelp database called *yelp_db_small*. The steps to load this database on your local machine were explained in Part 0.

Once you have created the reduced Yelp database, use the explain command to analyze the following queries:

1. Count the reviews written in May 2014. You should use the Review table and name the output column as *count*.
2. A user *(Review.userid='KGYM_D6JOkjwnzslWO0QHg')* exists in your database who has written some reviews. Write a query that lists the review count (review_count) and average stars (avg_stars) for this user in the user table, as well as all the review ids (review_ids), and business names (business_names) assigned in each review. You have to use User, Review, and Business tables in this query and use a join operation. *Hint: Your output should have a total of 13 rows.*

Your task is to examine the query plan for each of these queries, and add any other indexes that may improve the performance of the query.

For each query:

a) Write your query.

b) Write the explain command to output the query execution plan for your query.

c) Add one index that can lead to improving the running time of your query according to your query execution plan.

e) Write the explain command to output the improved execution plan.

Write your answers for the first query in *part3-1.sql* and the second query in *part3-2.sql*.

Package your finished submission of all files using the provided *package.sh* and submit the resulting tarball to LEARN.