CS 1555/2055 – Database Management Systems (Fall 2019)

Dept. of Computer Science, University of Pittsburgh

Assignment #6: SQL/PL: Functions, Procedures, and Triggers

Release: Oct. 11, 2019 Due: 8PM, Thursday, Oct. 17, 2019

Goal

Practice advanced SQL. In this homework you will have to use transactions appropriately and create functions, procedures and triggers using SQL/PL. You will continue with the US forest database of previous assignments.

Description

- Assume the following relational database schema that records information related to US forests. All primary key constraints are not deferrable, all foreign key constraints are initially deferred deferrable, and all the other constraints are initially immediate deferrable. You can see the schema definitions in `hw6-db.sql`.
  - FOREST = (Forest_No, Name, Area, Acid_Level, MBR_XMin, MBR_XMax, MBR_YMin, MBR_YMax)
  - STATE = (Name, Abbreviation, Area, Population)
  - COVERAGE = (Forest_No, State, Percentage, Area)
    FK (Forest_No) → FOREST(Forest_No)
    FK (State) → STATE(Abbreviation)
  - ROAD = (Road_No, Name, Length)
  - INTERSECTION = (Forest_No, Road_No)
    FK (Forest_No) → FOREST(Forest_No)
    FK (Road_No) → ROAD(Road_No)
  - SENSOR = (Sensor_Id, X, Y, Last_Charged, Maintainer, Last_Read)
    FK (Maintainer) → WORKER(SSN)
  - REPORT = (Sensor_Id, Report_Time, Temperature)
    FK (Sensor_Id) → SENSOR(Sensor_Id)
  - WORKER = (SSN, Name, Age, Rank)
  - EMERGENCY = (Sensor_Id, Report_Time)
    FK (Sensor_Id, Report_Time) → REPORT(Sensor_Id, Report_Time)

- Use `hw6-db.sql` to populate the database and answer the following questions with the sample data from previous assignments [for a total of 100 points]:
  1. [15 points each] Implement the following procedure and function in PostgreSQL and Oracle.
     (a) Create stored procedure pro_Update_Last_Read that updates the Last_Read attribute of a specific sensor. The procedure has the following inputs:
       - input_sensor_id: the ID of the sensor to be updated
       - read_time: the value to be used to update the Last_Read attribute.

(b) Create the function fun_Compute_Percentage that, given a specific amount area_covered as part of a forest's area covered by some state, returns the percentage of area_covered out of the whole area of the forest. The function has the following inputs:
   - input_forest_no: the ID of the forest in consideration
   - area_covered: the part of the forest's area covered by some state.

   For example: given the input values {input_forest_no = 1, area_covered = 16000}, the function will return 0.4 (40%), because the area of the forest whose Forest_No equals 1 is 40000. If the input area_covered is larger than the forest's area, the function returns 1 (100%).

2. [15 points each] Implement the following triggers in PostgreSQL and Oracle.
   (a) Define a trigger tri_Last_Read so that when a sensor reports a new temperature, the trigger will automatically update the value of Last_Read of that sensor. The trigger body should use the procedure pro_Update_Last_Read created above.
   (b) Define a trigger tri_Percentage so that when the area of a forest covered by some state is updated, the trigger automatically update the corresponding value of "percentage", using the function fun_Compute_Percentage created above.
   (c) Define a trigger tri_Emergency so that when a new report is inserted with the reported temperature greater than 100, the trigger inserts a corresponding tuple into table Emergency.

3. [5 points each] Verify the triggers created in Question 2 in PostgreSQL and Oracle.
   (a) Test the trigger tri_Last_Read in 3 steps:
      i. Display the content of SENSOR.
      ii. Add a new report with a report time.
      iii. Display the content of SENSOR.
   (b) Test the trigger tri_Percentage in 3 steps:
      i. Display the content of COVERAGE.
      ii. Update the area of a forest covered by some state.
      iii. Display the content of COVERAGE.
   (c) Test the trigger tri_Emergency in 3 steps:
      i. Display the content of EMERGENCY.
      ii. Add a new report with a temperature greater than 100.
      iii. Display the content of EMERGENCY.

4. [10 points total] Write a procedure with a cursor and call the procedure to list the names of the forests in PostgreSQL and Oracle. You need to implement the cursor both explicitly and implicitly for each DBMS.
   For the explicit cursor, you need to declare the cursor, open it, and fetch records from it.
   For the implicit cursor, you don't need to declare the cursor, only use the "for loop" to fetch records.
   Use the following command to print the names:
   - For PostgreSQL: `raise notice '%', forest_record.name;`
   - For Oracle (SQL Plus): `dbms_output.put_line(forest_record.name);`
     (Run `SET SERVEROUTPUT ON` at program start)
   - For Oracle (DataGrip): `dbms_output.put_line(forest_record.name);`
     (Click the `Enable SYS.DBMS_OUTPUT` button first (Command+F8 on MacOS, Ctrl+F8 on Windows))

What to submit

You are required to submit **exactly four** files under your **pitt_user_name** (e.g, abc01).

- **pitt_user_name-query.sql** (e.g., abc01-query.sql)
  In this file, please submit the answers to question 1-4 for PostgreSQL.
  In addition to providing the answers, you are expected to:
  - **include your name and pitt user name at the top of the file** using SQL comments
  - identify the question number before each answer using SQL comments
  - write simple queries at the beginning of this file to list the content of all the tables.

  The entire text file should be composed of **valid SQL statements**.

- **pitt_user_name-output.txt** (e.g., abc01-output.txt)
  In this file, please submit the output after running the pitt_user_name-query.sql file. You should use the PostgreSQL interactive terminal `psql` to generate the output. You can find the guide to start and use `psql` on class server at `QuickGuidePostgres_1555.pdf`. Briefly speaking, you need to add `psql` to your system's `PATH`, run your query file in `psql` and output the query output to a text file.
  In addition to providing the query output, you are expected to:
  - **include your name and pitt user name at the top of the file**
  - identify the question number before each answer

- **pitt_user_name-query-oracle.sql** (e.g., abc01-query-oracle.sql)
  In this file, please submit the answers to question 1-4 for Oracle.
  In addition to providing the answers, you are expected to:
  - **include your name and pitt user name at the top of the file** using SQL comments
  - identify the question number before each answer using SQL comments
  - write simple queries at the beginning of this file to list the content of all the tables.

  The entire text file should be composed of **valid SQL statements**.

- **pitt_user_name-output-oracle.txt** (e.g., abc01-output-oracle.txt)
  In this file, please submit the output after running the pitt_user_name-query-oracle.sql file. You need to use the Oracle interactive terminal `sqlplus` to generate the output. In `sqlplus`, you could use the command `spool` to begin a recording session and the command `start` to run your query file. More details about how to run and use `sqlplus` can be found at: `db.cs.pitt.edu/courses/cs1555/current.term/handouts/OracleEnvironment_1555.pdf`.

How to submit your assignment

1. Submit your assignment (the 4 files described above) through the Web-base submission interface (i.e., go to the class web page `http://db.cs.pitt.edu/courses/cs1555/current.term` and click the Submit button). **It is your responsibility to make sure the assignment was properly submitted.**
2. Submit your assignment by the due date (8PM, Thursday, Oct. 17, 2019). There is no late submission.

Academic Honesty

The work in this assignment is to be done *independently.* Discussions with other students on the assignment should be limited to understanding the statement of the problem. Cheating in any way, including giving your work to someone else will result in an F for the course and a report to the appropriate University authority.