EAST TENNESSEE STATE
UNIVERSITY

CSCI 2020 – Database Fundamentals

Lab 11 – Stored Programs

Header Comments

Enter the standard header comments into your SQL Worksheet in SQL Developer. Failure to do so will result in a 10% (4 point) reduction in your grade. Save your file as LastnameFirstinitial_Lab11.sql (e.g., RezwanaT_Lab11.sql).

PL/SQL (40 total points)

Today's lab will introduce you to PL/SQL – a native Oracle programming language that uses regular SQL. Place a slash (/) after every PL/SQL block and after ever SQL statement so the file will run as a batch.

1. User-Defined Function (15 Points)

User-defined functions allow you to perform manipulations within a pre-defined block of code and then return a value from that block.

a) Create a stored function get_month () that takes a parameter of type Date and returns the Month corresponding to that date. (HINT: Use the TO_CHAR function to convert the input date to a month string).

Once you have successfully created the stored function, query the **Academy_Prep schema** to test your function.

b) Write a query that displays the number of students graduated in each month within the year 2007. Your query MUST use the get_month() function that you created.

c) Use the get_month() function to execute a query that shows the average SAT test scores taken by male and female students during each month of 2010. Your results should display the following columns:

Month	Gender	Average SAT Score

2. Triggers (10 Points)

Create the following two tables. (Copy and Paste these into your solution file).

```
DROP TABLE Student_Example;
CREATE TABLE Student_Example (
   Student_ID INTEGER,
   Student_First_Name VARCHAR2(30),
   Student_Last_Name VARCHAR2(30),
   Student_Classification VARCHAR2(2)
);
```

```
DROP TABLE Student_Example_Log;
CREATE TABLE Student_Example_Log (
   Student_ID INTEGER,
   Student_First_Name VARCHAR2(30),
   Student_Last_Name VARCHAR2(30),
   Student_Classification VARCHAR2(2),
   Activity_Date DATE,
   Username VARCHAR2(20)
);
```

Create the following two triggers.

1. `BEFORE INSERT` into Student_Example, get the highest Student ID in the table, and add one to it. Set the new value of Student_ID to the value you calculated in your trigger. Name this Trigger `Student_Example_ID`. Note that you will need to handle the situation when there are no rows in the table. One way to do this is by first counting the rows. If there are zero rows, return 1, else grab the highest ID and add 1.
2. `AFTER INSERT` or `UPDATE` of Student_Example, insert a log record into Student_Example_Log that contains the new values. The Activity_Date should be set using the keyword `SYSDATE`, and the Username should be set using the keyword `USER`. Name this Trigger `Student_Example_Logger`.

Both Triggers should be defined `FOR EACH ROW`.

3. User-Defined Procedure (15 Points)

Create a user-defined function (using `CREATE OR REPLACE`) named `Trigger_Tester` that inserts three rows into Student_Example. Your insert statements should only add Student_First_Name, Student_Last_Name, and Student_Classification. If your triggers are correctly defined, Student_ID will be added automatically. Finally, UPDATE one of the rows that you just inserted. Be sure to add a COMMIT statement at the end of your procedure definition.

Following your procedure definition, run it using the `EXECUTE` command and then select all rows from Student_Example and Student_Example_Log to ensure it worked properly.

Submission

Submit your file, named LastnameFirstinitial_Lab11.sql, to the Lab 11 Dropbox area on D2L by **Friday, April 4, 2022, 11:59 PM.**