Unit 10 Problem Set Submission Form

# Overview

|  |  |
| --- | --- |
| Your Name | Rutwik Ghag |
| Your SU Email | rughag@syr.edu |

# Instructions

Put your name and SU email at the top. Answer these questions all from the lab. When asked to include screenshots, please follow the screen shot guidelines from the first lab.

Remember as you complete the problem sets it is not only about getting it right / correct. We will discuss the answers in class so it’s important to articulate anything you would like to contribute to the discussion in your answer:

* If you feel the question is vague, include any assumptions you've made.
* If you feel the answer requires interpretation or justification provide it.
* If you do not know the answer to the question, articulate what you tried and how you are stuck.

This how you receive credit for answering questions which might not be correct.

# Questions

Answer these questions using the problem set submission template. You will need to consult the logical model in the overview section for details. For any screenshots provided, please follow the guidelines for submitting a screenshot.

Write the following as SQL programs. For each, include the SQL as a screenshot with the output of the query.

1. In the **TinyU** database,
   1. Write an SQL Stored procedure called **p\_upsert\_major** which given a major\_code (business key) and a major\_name does an Upsert, which is the following:
      1. Check if the major\_code exists in the table already.
      2. If yes, update the table and make the major\_name match the new major name.
      3. If no, insert the new major\_name and major\_code into the table. HINT: major\_id is not a surrogate key so you will need to determine the next ID yourself in code!
   2. Test your stored procedure by executing it to make these changes
      1. change : CSC – Computer Sciences to CSC – Computer Science and
      2. add: FIN – Finance.

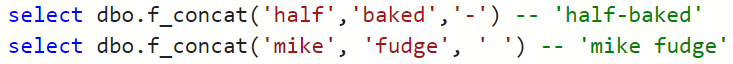
Make sure your screenshot captures all up/down code in 1.a AND another screen shot captures 1.b the output of your code execution to show that it works. SELECT the table before and after!

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

1. In the **TinyU** database,
   1. write a user-defined function called **f\_concat** which combines the any two varchars @a and @b together with a one-character @sep in between.   
      For example:  
      
   2. Now create a view called **v\_students** which displays the student\_id student name (first last), student name (last, first), gpa, and name of major. You should call the function you created in 2.a. After you create the view, execute it with a SELECT statement.

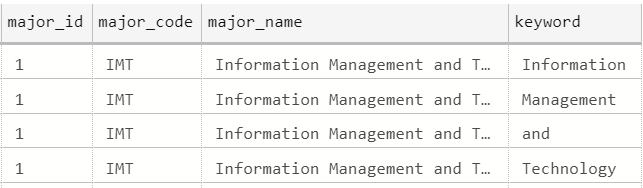
Make sure your screenshot captures all up/down code in 2.a AND another screen shot captures 2.b along with the output of the SELECT statement on the view (first few rows is fine).

Graphical user interface, text, application

Description automatically generated

Graphical user interface

Description automatically generated

1. In the **TinyU** database,
   1. Write a query on the **majors** table so that the major\_name is broken up into keywords one per row. HINT: you must use string\_split() with cross apply.   
      
   2. Then use the query in 3.a to create a table-valued function **f\_search\_majors** which allows you to search the majors by keyword. Demonstrate calling the TVF by querying all majors with the ‘Science’ keyword.

Your screenshot should include the query in 3.a Another screenshot should show the TVF in 3.b and the sample output from the SELECT statement calling the TVF.

Graphical user interface

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated

1. In the **TinyU** database,
   1. Alter the **students** table and add the following columns:
      1. student\_active char(1) default (‘Y’) not null
      2. student\_inactive\_date date null
   2. Create a trigger on the **students** table which when there is an student\_inactive\_date set will set student\_active to ‘N’, whenever there is not a student\_inactive\_date then student\_active is set to ‘Y’.
   3. Write SQL code to deactivate all the ‘Graduate’ students with a date of ‘2020-08-01’
   4. Write SQL code to re-activate all the ‘Graduate’ students.

Provide a screenshot of your code from 4.a. and 4.b working. Provide another screenshot demonstrating 4.c worked. Then a final screenshot of code and demonstration of 4.d working.

Graphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

Table

Description automatically generated with medium confidence

Table

Description automatically generated

# Reflection

Use this section to reflect on your learning. To achieve the highest grade on the assignment you must be as descriptive and personal as possible with your reflection.

1. What are the key things you learned through the process of completing this assignment?  
     
   The key things I learned from this assignment :
   1. Use built in value functions,
   2. Table value functions,
   3. Creating stored procedures,
   4. Triggers
   5. Creating views
2. What were the challenges or roadblocks (if any) you encountered on the way to completing it?  
     
   I faced some problem while solving the third problem.
3. Were you prepared for this assignment? What can you do to be better prepared?  
     
   I had referred professor’s Fudge’s videos prior to starting this assignment.
4. Now that you have completed the assignment rate your comfort level with this week’s material. This should be an honest assessment: (choose one)

4