

## Instructions

Read these instructions all the way through before you start.

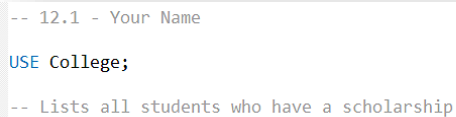
Download the answer document. Be sure to put your name at the top of the answer document where that's indicated. Answer the questions below in that document and when you are done, upload it into Blackboard.

## Overview

In this unit you'll get a feel on how to create and use stored procedures and user functions.

## Preparing your SQL

Make sure you include the comment and USE statement at the top of your SQL

A screenshot of a text editor window showing SQL code. The code is as follows:

```
-- 12.1 - Your Name  
USE College;  
-- Lists all students who have a scholarship
```

and don't forget to beautify your SQL, by hand if needed. In your screen shot of the Result Grid you need to show only first 6-10 rows. If you're not sure about these, see Unit 02 instruction for details on how to do that.

## Important Tips

You can speed up your work by automatically dropping a stored procedure, if it's already there, before you recreate it. Use this kind of code

```
DROP procedure IF EXISTS Students_With_Scholarships ;
```

## Steps

1. [5] Create and call a stored procedure named **Students\_With\_Scholarships** that lists all students who have a scholarship, sorted by last name. Call the procedure. Its results should look like this.

Last Name	First Name	Scholarship
Armstrong	Jose	2000.00
Bailey	Christina	1000.00

2. [5] Create and call a stored procedure named **Students\_Major** that takes as it's IN parameter the ID of a major and lists all students who have that major. Call the procedure, providing the ID of a major.

Last Name	First Name	Major
-----------	------------	-------

3. [8] Create a stored procedure name **Student\_Add** that takes arguments for LastName, FirstName, Sex, MajorID, and DateOfBirth, and adds the person to the Student table. Your stored procedure should automatically set EnrolledDate to today's date and automatically create the email address from the student's name. You'll need to use the CONCAT function to create the email address in the form FirstLast@college.edu.

Call your stored procedure to add yourself as a student, and then run this SELECT statement to list out the new record.

```
SELECT
    *
FROM
    student
WHERE
    student.id = (SELECT MAX(ID) FROM Student);
```

4. [8] Create and call a stored procedure named **Faculty\_GPA** that takes as it's IN parameter the ID of a faculty member and returns the average grade across all the sections the professor teaches. Call the procedure for faculty member Richard Graham and write code to display the average GPA of his students.

Your code to call the stored procedure should look like this

```
CALL Faculty_GPA(2, @averageGPA) ;  
  
SELECT @averageGPA as GPA ;
```

5. [10] Create and call a function named **Student\_Section\_Count** that takes as it's IN parameter the ID of a student and returns the number of sections the student has registered for. Use this function in the following SELECT statement.

```
SELECT  
    LastName,  
    FirstName,  
    Student_Section_Count(ID) AS 'Registered Sections'  
FROM  
    Student  
LIMIT 6;
```

6. [10] Create a Stored Procedure of your own design. This procedure must take one IN argument and must have a useful purpose. It should contain a SELECT statement that joins together at least three tables and not be a statement used earlier in the course. Write a brief description of the purpose of your stored procedure.