

# Unit 10 Problem Set Submission Form

## Overview

|               |                  |
|---------------|------------------|
| Your Name     | Jainish Savaliya |
| Your SU Email | jsavaliy@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.

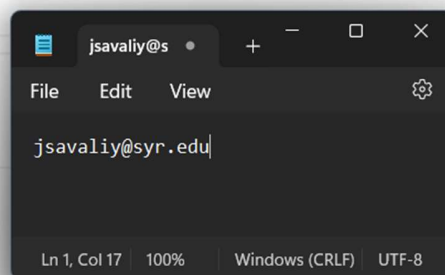
1. In the **TinyU** database,
  - a. Write an SQL Stored procedure called **p\_upsert\_major** which given a major\_code (business key) and a major\_name does an Upsert, which is the following:
    - i. Check if the major\_code exists in the table already.
    - ii. If yes, update the table and make the major\_name match the new major name.
    - iii. 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!
  - b. Test your stored procedure by executing it to make these changes
    - i. change : CSC – Computer Sciences to CSC – Computer Science and
    - ii. 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!

```
Run Cancel Disconnect Change Connection tinyu Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook
1 drop PROCEDURE if exists p_upsert_major
2 GO
3 create PROCEDURE p_upsert_major (
4     @new_major_code varchar(3),
5     @new_major_name varchar(20),
6     @new_major_id INT
7 ) as BEGIN
8 if exists (select major_code from dbo.majors where major_code = @new_major_code)
9 BEGIN
10 update dbo.majors
11 set major_name=@new_major_name where major_code=@new_major_code
12 return NULL
13 end
14 ELSE
15 BEGIN
16 print 'no major found'
17 insert into dbo.majors (major_id,major_code,major_name)
18 values (@new_major_id,@new_major_code,@new_major_name)
19 return NULL
20 END
21 END
22
```

#### Messages

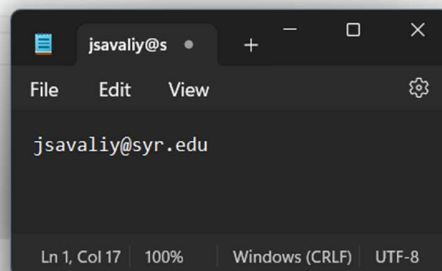
```
8:35:04 PM Started executing query at line 1
Commands completed successfully.
8:35:04 PM Started executing query at line 3
Commands completed successfully.
Total execution time: 00:00:00.045
```



```
Run Cancel Disconnect Change Connection tinyu Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook
1 drop PROCEDURE if exists p_upsert_major
2 GO
3 create PROCEDURE p_upsert_major (
4     @new_major_code varchar(3),
5     @new_major_name varchar(20),
6     @new_major_id INT
7 ) as BEGIN
8 if exists (select major_code from dbo.majors where major_code = @new_major_code)
9 BEGIN
10 update dbo.majors
11 set major_name=@new_major_name where major_code=@new_major_code
12 return NULL
13 end
14 ELSE
15 BEGIN
16 print 'no major found'
17 insert into dbo.majors (major_id,major_code,major_name)
18 values (@new_major_id,@new_major_code,@new_major_name)
19 return NULL
20 END
21 END
22
23 exec p_upsert_major @new_major_code = 'CSC',@new_major_name = "Computer Science",@new_major_id='4'
24 exec p_upsert_major @new_major_code = 'FIN',@new_major_name = "FINANCE",@new_major_id='6'
25
26 select * from dbo.majors
```

#### Messages

```
8:33:23 PM Started executing query at line 1
Commands completed successfully.
8:33:23 PM Started executing query at line 3
Commands completed successfully.
Total execution time: 00:00:00.020
```



2. In the **TinyU** database,
  - a. 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:

```
select dbo.f_concat('half','baked','-') -- 'half-baked'
select dbo.f_concat('mike','fudge',' ') -- 'mike fudge'
```

- b. 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).

The screenshot displays the SQL Server Enterprise Manager interface. The top pane shows the SQL script for creating a function and a view. The bottom pane shows the execution messages and the results of the view.

**SQL Script:**

```
1 drop function if exists f_concat
2 go
3 create function f_concat (
4     @a VARCHAR(50),
5     @b VARCHAR(50)
6 ) returns varchar(50) as BEGIN
7     return (@a+ '-' + @b)
8 end
9 go
10 drop view if exists v_students
11 go
```

**Execution Messages:**

```
8:37:53 PM Started executing query at line 1
Commands completed successfully.
8:37:53 PM Started executing query at line 3
Commands completed successfully.
8:37:53 PM Started executing query at line 11
Commands completed successfully.
Total execution time: 00:00:00.017
```

**Results:**

| student_id | student_firstname | student_lastname | student_year_name | student_major_id | student_gpa | student_notes |
|------------|-------------------|------------------|-------------------|------------------|-------------|---------------|
| 1          | Robin             | Banks            | Freshman          | 3                | 4.000       |               |
| 2          | Victor            | Edance           | Freshman          | 2                | 2.404       |               |
| 3          | Erin              | Yortires         | Junior            | 1                | 2.401       |               |
| 4          | Aurora            | Borealis         | Senior            | 1                | 3.024       |               |
| 5          | Tuck              | Androll          | Senior            | 2                | 3.333       |               |
| 6          | Eura              | Quittin          | Senior            | 2                | 3.372       |               |
| 7          | Willie            | Survive          | Sophomore         | 2                | 2.608       |               |
| 8          | Lola              | Dabridgeda       | Freshman          | 1                | 2.732       |               |
| 9          | Doris             | Closed           | Senior            | 3                | 3.173       |               |
| 10         | Phil              | McCup            | Freshman          | 2                | 2.705       |               |

| student_id | firstlast_concat | lastfirst_concat | student_gpa | major_name                      |
|------------|------------------|------------------|-------------|---------------------------------|
| 1          | Robin-Banks      | Banks-Robin      | 4.000       | Accounting                      |
| 2          | Victor-Edance    | Edance-Victor    | 2.404       | Applied Data Science            |
| 3          | Erin-Yortires    | Yortires-Erin    | 2.401       | Information Management and T... |
| 4          | Aurora-Borealis  | Borealis-Aurora  | 3.024       | Information Management and T... |
| 5          | Tuck-Androll     | Androll-Tuck     | 3.333       | Applied Data Science            |
| 6          | Eura-Quittin     | Quittin-Eura     | 3.372       | Applied Data Science            |
| 7          | Willie-Survive   | Survive-Willie   | 2.608       | Applied Data Science            |
| 8          | Lola-Dabridgeda  | Dabridgeda-Lola  | 2.732       | Information Management and T... |
| 9          | Doris-Closed     | Closed-Doris     | 3.173       | Accounting                      |

3. In the **TinyU** database,

- a. 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.

| major_id | major_code | major_name                      | keyword     |
|----------|------------|---------------------------------|-------------|
| 1        | IMT        | Information Management and T... | Information |
| 1        | IMT        | Information Management and T... | Management  |
| 1        | IMT        | Information Management and T... | and         |
| 1        | IMT        | Information Management and T... | Technology  |

- b. 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.

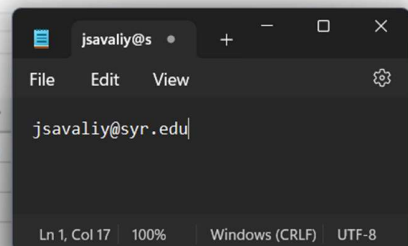
Your screenshot should include the query in 3.a Another screenshot should show the TVF in 3.b and the sample output from the SELECT statement calling the TVF.

```

Run Cancel Disconnect Change Connection tinyu
1 select * from dbo.majors cross apply string_split(major_name, ' ')
2 go
3 drop function if exists f_search_majors
4 go
5 create function f_search_majors (
6     @search varchar(50)
7 ) RETURNS TABLE
8 AS
9 return
10 select * from (select major_id, major_code, major_name, value search from majors cross apply string_split(major_name, ' ')) sear
11 where sear.search=@search
12 GO
13 select * from f_search_majors('science')

```

| Results |          |            |                                 |             |
|---------|----------|------------|---------------------------------|-------------|
|         | major_id | major_code | major_name                      | value       |
| 1       | 1        | IMT        | Information Management and T... | Information |
| 2       | 1        | IMT        | Information Management and T... | Management  |
| 3       | 1        | IMT        | Information Management and T... | and         |
| 4       | 1        | IMT        | Information Management and T... | Technology  |
| 5       | 2        | ADS        | Applied Data Science            | Applied     |
| 6       | 2        | ADS        | Applied Data Science            | Data        |
| 7       | 2        | ADS        | Applied Data Science            | Science     |
| 8       | 3        | ACC        | Accounting                      | Accounting  |
| 9       | 4        | CSC        | Computer Sciences               | Computer    |
| 10      | 4        | CSC        | Computer Sciences               | Sciences    |
|         | major_id | major_code | major_name                      | search      |
| 1       | 2        | ADS        | Applied Data Science            | Science     |



4. In the **TinyU** database,

- Alter the **students** table and add the following columns:
  - student\_active char(1) default ('Y') not null
  - student\_inactive\_date date null
- Create a trigger on the **students** table which when there is a 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'.
- Write SQL code to deactivate all the 'Graduate' students with a date of '2020-08-01'
- 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.

Run Cancel Disconnect Change Connection tinyu Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook

```
1 drop trigger if exists t_after_update_student_status
2 go
3 CREATE trigger t_after_update_student_status
4 on students
5 after insert, update
6 as BEGIN
7 if update(student_inactive_data) BEGIN
8 update students
9 set student_active = case
10 when student_inactive_data is not null then 'N'
11 when student_inactive_data is null then 'Y'
12 END
13 END
14 END
```

jsavaliy@s + - □ ×

File Edit View

jsavaliy@syr.edu

Ln 1, Col 17 100% Windows (CRLF) UTF-8

#### Messages

9:12:45 PM Started executing query at line 1  
Commands completed successfully.

9:12:45 PM Started executing query at line 3  
Commands completed successfully.  
Total execution time: 00:00:00.016

Run Cancel Disconnect Change Connection tinyu Estimated Plan Enable Actual Plan Enable SQLCMD Export as Notebook

```
1 alter table students
2 add student_active char(1) , student_inactive_data date null
3 GO
4 if exists (select * from INFORMATION_SCHEMA.TABLE_CONSTRAINTS
5 where constraint_name='df_students_student_active')
6 alter table students drop constraint df_students_student_active
7 go
8 alter table students
9 add CONSTRAINT df_students_student_active
10 default 'Y' for student_active
11 go
12
```

#### Messages

9:11:01 PM Started executing query at line 1  
Commands completed successfully.

9:11:01 PM Started executing query at line 4  
Commands completed successfully.

9:11:01 PM Started executing query at line 8  
Commands completed successfully.  
Total execution time: 00:00:00.046

jsavaliy@s + - □ ×

File Edit View

jsavaliy@syr.edu

The screenshot shows a database management interface. At the top, there's a toolbar with buttons like 'Run', 'Cancel', 'Disconnect', and 'Change Connection'. Below this, a SQL query is entered in a text area:

```

1 update students
2 set student_inactive_data = '2020-08-01'
3
4 GO
5 select * from students
6 GO

```

Below the query, there's a 'Results' tab showing a table of student data. The table has columns: student\_id, student\_firstname, student\_lastname, student\_year, student\_gpa, student\_status, student\_academic\_status, and student\_inactive\_data. The data is as follows:

| student_id | student_firstname | student_lastname | student_year | student_gpa | student_status | student_academic_status | student_inactive_data |
|------------|-------------------|------------------|--------------|-------------|----------------|-------------------------|-----------------------|
| 1          | Robin             | Banks            | Freshman     | 3           | 4.000          | N                       | 2020-08-01            |
| 2          | Victor            | Edance           | Freshman     | 2           | 2.404          | N                       | 2020-08-01            |
| 3          | Erin              | Yortires         | Junior       | 1           | 2.401          | N                       | 2020-08-01            |
| 4          | Aurora            | Borealis         | Senior       | 1           | 3.024          | N                       | 2020-08-01            |
| 5          | Tuck              | Androll          | Senior       | 2           | 3.333          | N                       | 2020-08-01            |
| 6          | Eura              | Quittin          | Senior       | 2           | 3.372          | N                       | 2020-08-01            |
| 7          | Willie            | Survive          | Sophomore    | 2           | 2.608          | N                       | 2020-08-01            |
| 8          | Lola              | Dabridgeda       | Freshman     | 1           | 2.732          | N                       | 2020-08-01            |
| 9          | Doris             | Closed           | Senior       | 3           | 3.173          | N                       | 2020-08-01            |
| 10         | Phil              | McCup            | Freshman     | 2           | 2.705          | N                       | 2020-08-01            |
| 11         | Jack              | Itupp            | Sophomore    | 3           | 3.386          | N                       | 2020-08-01            |
| 12         | Val               | Idation          | Senior       | 1           | 3.500          | N                       | 2020-08-01            |
| 13         | Ida               | Knowe            | Junior       | 4           | 2.724          | N                       | 2020-08-01            |
| 14         | Lee               | Hvmeehom         | Junior       | 2           | 1.916          | meet with student       | 2020-08-01            |
| 15         | Ginger            | Beer             | Graduate     | 2           | 4.000          | N                       | 2020-08-01            |
| 16         | Buck              | Naked            | Freshman     | 2           | 2.434          | N                       | 2020-08-01            |
| 17         | Val               | Uation           | Junior       | 4           | 3.384          | N                       | 2020-08-01            |
| 18         | Robin             | Eue              | Sophomore    | 2           | 3.006          | N                       | 2020-08-01            |
| 19         | Tera              | Dactyl           | Junior       | 1           | 2.367          | N                       | 2020-08-01            |

## 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.

- What are the key things you learned through the process of completing this assignment?  
-> **The key things I would say I learnt in this assignment are views, Triggers and stored procedures.**
- What were the challenges or roadblocks (if any) you encountered on the way to completing it?  
-> **I watched the video and understood the concepts so this one was smooth compared to other assignments**
- Were you prepared for this assignment? What can you do to be better prepared?  
-> **I think I was prepared for this assignment. The better I can do is to dig deep into the concepts and try solving new problems of this topics.**
- 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 ==> I understand this material and can explain it to others.**