Unit 10 Problem Set Submission Form

Overview

Your Name	Hendi Kushta
Your SU Email	hkushta@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.

-

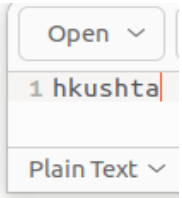
Questions

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!

```
SELECT * from majors

DROP PROCEDURE IF EXISTS p_upsert_major
GO
create PROCEDURE p_upsert_major (
    @major_id int,
    @major_c CHAR(3),
    @major_n VARCHAR(50)
)as BEGIN
if exists (select * from majors WHERE major_code = @major_c) Begin
    print 'Major alreary in the majors table'
    UPDATE majors
        SET major_name = @major_n
        WHERE major_code = @major_c
    end
ELSE
    BEGIN
        insert into majors(major_id, major_code, major_name)
            VALUES(@major_id, @major_c, @major_n)
    end
end
```

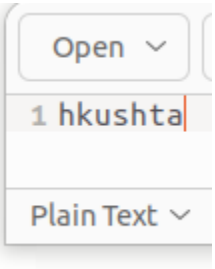


b. Test your stored procedure by executing it to make these changes
    i. change : CSC – Computer Sciences to CSC – Computer Science and
    ii. add: FIN – Finance.

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!

Table before

	major_id	major_code	major_name
1	1	IMT	Information Management an…
2	2	ADS	Applied Data Science
3	3	ACC	Accounting
4	4	CSC	Computer Sciences
5	5	BSK	Basket Weaving




```
EXEC p_upsert_major
    @major_id = 6,
    @major_c='CSC',
    @major_n = "Computer Science and"

EXEC p_upsert_major
    @major_id = 7,
    @major_c='FIN',
    @major_n = "Finance"
```

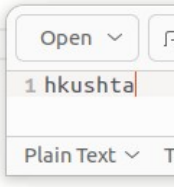
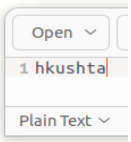


Table after

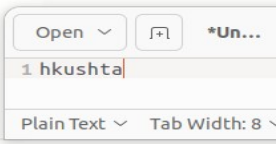
	major_id	major_code	major_name
1	1	IMT	Information Management and…
2	2	ADS	Applied Data Science
3	3	ACC	Accounting
4	4	CSC	Computer Science and
5	5	BSK	Basket Weaving
6	7	FIN	Finance



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'
```

```
66
67    DROP FUNCTION  if exists f_concat
68    GO
69    CREATE FUNCTION f_concat (
70            @a VARCHAR(50),
71            @b VARCHAR(50),
72            @sep CHAR(1)
73
74    ) returns varchar(100) AS BEGIN
75    DECLARE
76            @full_name VARCHAR(100)
77            set
78                @full_name = @a + @sep + @b
79            RETURN @full_name
80    END
81    GO
82
83    SELECT dbo.[f_concat]('half', 'baked', '-') -- half-baked
84    SELECT dbo.[f_concat]('mike', 'fudge', '-') -- mike-fudge
85
```



Results **Messages**

	(No column name)
1	half-baked

b.

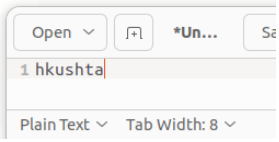
	(No column name)
1	mike-fudge

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

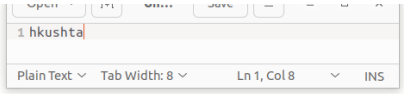
major. You should call the function you created in 2.a. After you create the view, execute it with a SELECT statement.

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).

```
CREATE VIEW v_students AS
    SELECT
        student_id,
        dbo.[f_concat](student_firstname, student_lastname, ' ') as student_name_first_space_last,
        dbo.[f_concat](student_lastname, student_firstname, ', ') as student_name_last_comma_first,
        student_gpa,
        major_name
    from students
        inner join majors on majors.major_id = students.student_major_id
```




```
100
101   SELECT * from v_students
102
103   GO
104
```



Results Messages

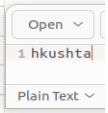
	student_id	student_name_first_space_last	student_name_last_comma_first	student_gpa	major_name
1	1	Robin Banks	Banks,Robin	4.000	Accounting
2	2	Victor Edance	Edance,Victor	2.404	Applied Data Science
3	3	Erin Yortires	Yortires,Erin	2.401	Information Management and…
4	4	Aurora Borealis	Borealis,Aurora	3.024	Information Management and…
5	5	Tuck Androll	Androll,Tuck	3.333	Applied Data Science
6	6	Eura Quittin	Quittin,Eura	3.372	Applied Data Science
7	7	Willie Survive	Survive,Willie	2.608	Applied Data Science

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

```
select * from majors
cross apply string_split(major_name, ' ')
```

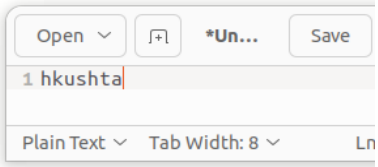
	major_id	major_code	major_name	value
1	1	IMT	Information Management and…	Information
2	1	IMT	Information Management and…	Management
3	1	IMT	Information Management and…	and
4	1	IMT	Information Management and…	Technology
5	2	ADS	Applied Data Science	Applied
6	2	ADS	Applied Data Science	Data
7	2	ADS	Applied Data Science	Science
8	3	ACC	Accounting	Accounting
9	4	CSC	Computer Science and	Computer
10	4	CSC	Computer Science and	Science
11	4	CSC	Computer Science and	and
12	5	BSK	Basket Weaving	Basket
13	5	BSK	Basket Weaving	Weaving
14	7	FIN	Finance	Finance



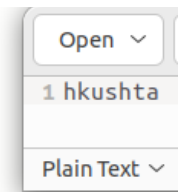
b. Then use the query in 3.a to create a table-valued function **f_search_majors** which allows you to search the majors by keyword. Demonstrate calling the TVF by querying all majors with the 'Science' keyword.

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.

```
DROP FUNCTION  if exists f_search_majors
GO
CREATE FUNCTION f_search_majors (
    @value VARCHAR(50)
)RETURNS TABLE

AS

RETURN
    select * from majors
        cross apply string_split(major_name, ' ')

        GROUP BY major_id, major_code, major_name, value
        having [value] = @value
GO

select major_id, major_code, major_name from f_search_majors('Science')
```



	major_id	major_code	major_name
1	2	ADS	Applied Data Science
2	4	CSC	Computer Science and



4. In the **TinyU** database,
    a. Alter the **students** table and add the following columns:
        i. student_active char(1) default (‘Y’) not null
        ii. student_inactive_date date null

```
ALTER TABLE students
ADD student_active char(1) default 'Y' not null,
    student_inactive_date DATE;
```

```
1 hkushta
```

	student_id	student_firstname	student_lastname	student_year_name	student_major_id	student_gpa	student_notes	student_active	student_inactive_date
1	1	Robin	Banks	Freshman	3	4.000		Y	NULL
2	2	Victor	Edance	Freshman	2	2.404		Y	NULL
3	3	Erin	Yortires	Junior	1	2.401		Y	NULL
4	4	Aurora	Borealis	Senior	1	3.024		Y	NULL
5	5	Tuck	Androll	Senior	2	3.333			
6	6	Eura	Quittin	Senior	2	3.372			
7	7	Willie	Survive	Sophomore	2	2.608			
8	8	Lola	Dabridgeda	Freshman	1	2.732			

Open *Un... Save

```
1 hkushta
```

    b. 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’.

```
DROP TRIGGER t_student_activity
GO
CREATE TRIGGER t_student_activity
    on students
    after INSERT, UPDATE, DELETE
    AS
        BEGIN
        UPDATE students
            SET student_active = 'N'
            from inserted
            WHERE students.student_inactive_date is not NULL
    END;
    BEGIN
        UPDATE students
            SET student_active = 'Y'
            from inserted
            WHERE students.student_inactive_date is NULL
    END;
```

Open

```
1 hkushta
```

Plain Text

    c. Write SQL code to deactivate all the ‘Graduate’ students with a date of ‘2020-08-01’

```
185
186   SELECT * from students WHERE student_year_name = 'Graduate'
187   UPDATE students SET student_inactive_date = '2020-08-01' WHERE student_year_name = 'Graduate'
188   SELECT * from students WHERE student_year_name = 'Graduate'
189
```

```
1 hkushta
```

Plain Text Tab Width: 8 Ln 1, Col 8

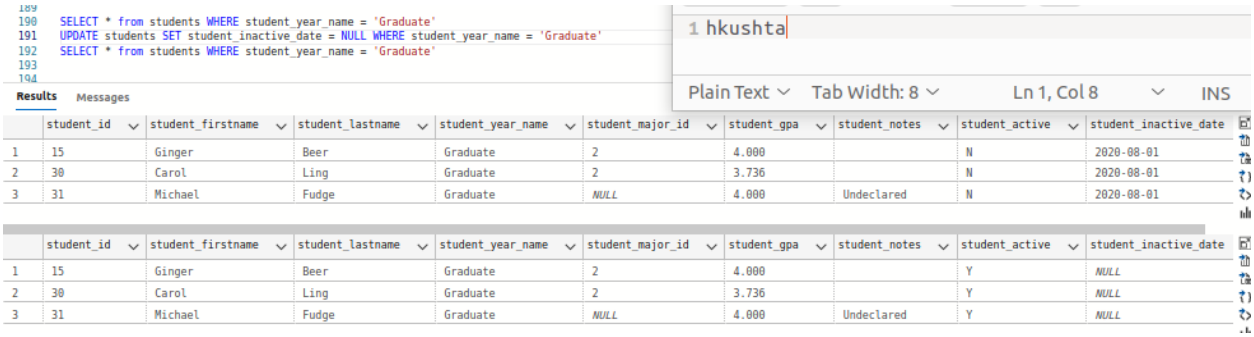
Results Messages

	student_id	student_firstname	student_lastname	student_year_name	student_major_id	student_gpa	student_notes	student_active	student_inactive_dat
1	15	Ginger	Beer	Graduate	2	4.000		Y	NULL
2	30	Carol	Ling	Graduate	2	3.736		Y	NULL
3	31	Michael	Fudge	Graduate	NULL	4.000	Undeclared	Y	NULL

	student_id	student_firstname	student_lastname	student_year_name	student_major_id	student_gpa	student_notes	student_active	student_inactive_dat
1	15	Ginger	Beer	Graduate	2	4.000		N	2020-08-01
2	30	Carol	Ling	Graduate	2	3.736		N	2020-08-01
3	31	Michael	Fudge	Graduate	NULL	4.000	Undeclared	N	2020-08-01

d. 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.



```
189
190  SELECT * from students WHERE student_year_name = 'Graduate'
191  UPDATE students SET student_inactive_date = NULL WHERE student_year_name = 'Graduate'
192  SELECT * from students WHERE student_year_name = 'Graduate'
193
194
```

```
1 hkushta
```

Plain Text | Tab Width: 8 | Ln 1, Col 8 | INS

Results | Messages

	student_id	student_firstname	student_lastname	student_year_name	student_major_id	student_gpa	student_notes	student_active	student_inactive_date
1	15	Ginger	Beer	Graduate	2	4.000		N	2020-08-01
2	30	Carol	Ling	Graduate	2	3.736		N	2020-08-01
3	31	Michael	Fudge	Graduate	NULL	4.000	Undeclared	N	2020-08-01

	student_id	student_firstname	student_lastname	student_year_name	student_major_id	student_gpa	student_notes	student_active	student_inactive_date
1	15	Ginger	Beer	Graduate	2	4.000		Y	NULL
2	30	Carol	Ling	Graduate	2	3.736		Y	NULL
3	31	Michael	Fudge	Graduate	NULL	4.000	Undeclared	Y	NULL

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?

   How to create a trigger using inserted and deleted tables.

2. What were the challenges or roadblocks (if any) you encountered on the way to completing it?

3. Were you prepared for this assignment? What can you do to be better prepared?

   Yes, I was.

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 ==> I understand this material and can explain it to others.
   3 ==> I understand this material.
   2 ==> I somewhat understand the material but sometimes need guidance from others.
   1 ==> I understand very little of this material and need extra help.