Unit 11 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.

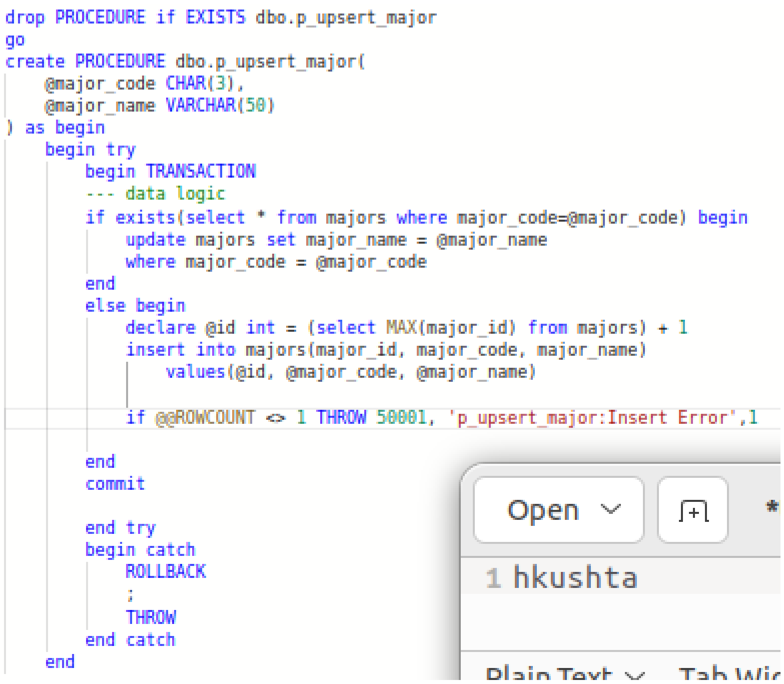
This how you receive credit for answering questions which might not be correct.

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 SQL Code.

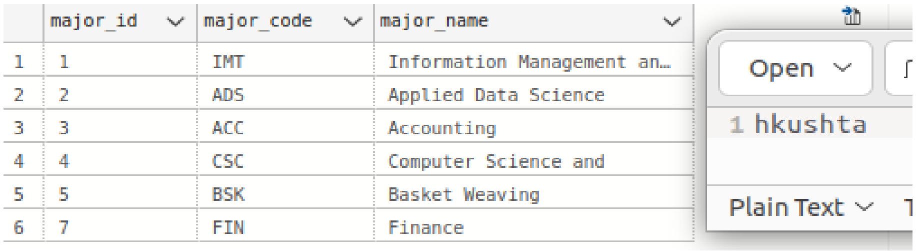
1. Provide a screenshot of your code execution from the walkthrough were you modified **p_upsert_major** in the **TinyU** database to be transaction-safe.



```
drop PROCEDURE if EXISTS dbo.p_upsert_major
go
create PROCEDURE dbo.p_upsert_major(
    @major_code CHAR(3),
    @major_name VARCHAR(50)
) as begin
    begin try
        begin TRANSACTION
        --- data logic
        if exists(select * from majors where major_code=@major_code) begin
            update majors set major_name = @major_name
            where major_code = @major_code
        end
        else begin
            declare @id int = (select MAX(major_id) from majors) + 1
            insert into majors(major_id, major_code, major_name)
                values(@id, @major_code, @major_name)

            if @@ROWCOUNT <> 1 THROW 50001, 'p_upsert_major:Insert Error',1

        end
        commit

        end try
        begin catch
            ROLLBACK
            ;
            THROW
        end catch
    end
```

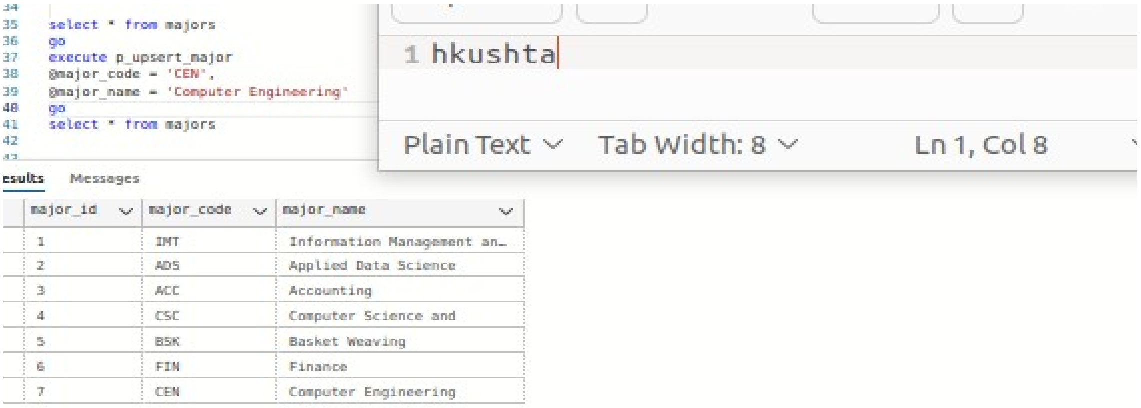
2. Provide a screenshot of examples of executing the **p_upsert_major** procedure to demonstrate it is transaction safe.

   Majors table before inserting or updating records.



	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 Science and
5	5	BSK	Basket Weaving
6	7	FIN	Finance

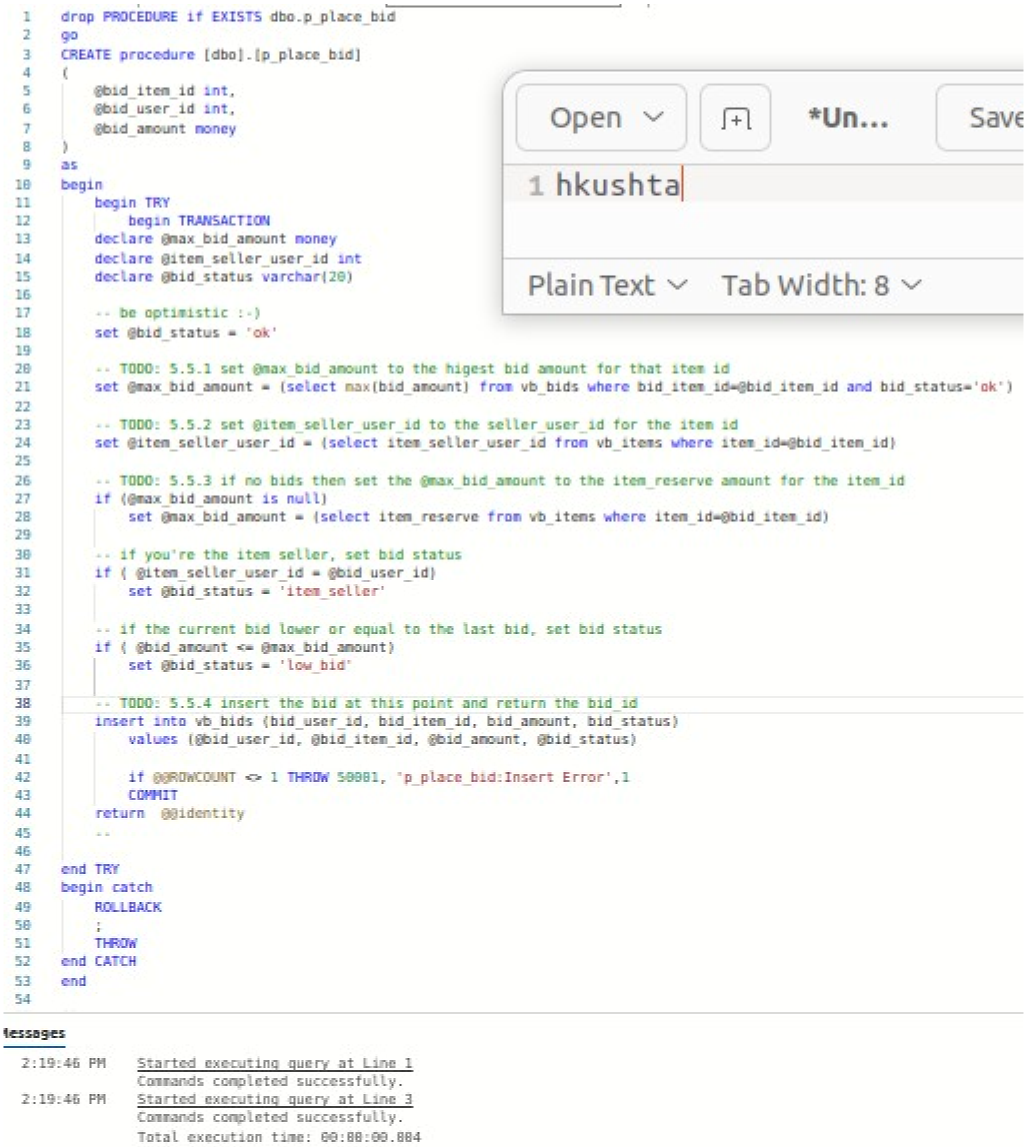
   Majors table after inserting



```
select * from majors
go
execute p_upsert_major
@major_code = 'CEN',
@major_name = 'Computer Engineering'
go
select * from majors
```

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

3. Re-write the **p_place_bid** stored procedure from the **vBay** database so that it is transaction safe. Provide a screenshot of the code and its execution.



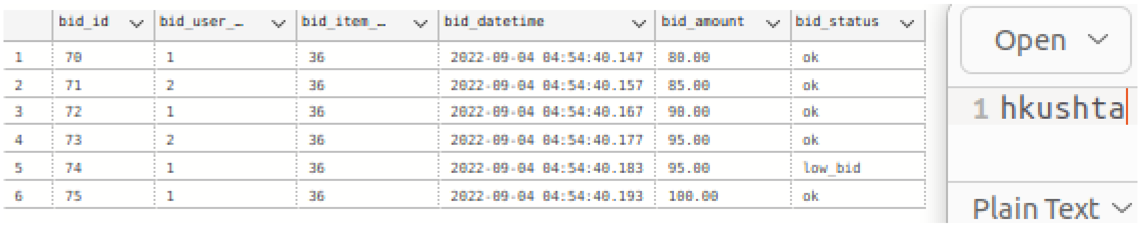
```sql
drop PROCEDURE if EXISTS dbo.p_place_bid
go
CREATE procedure [dbo].[p_place_bid]
(
    @bid_item_id int,
    @bid_user_id int,
    @bid_amount money
)
as
begin
    begin TRY
        begin TRANSACTION
    declare @max_bid_amount money
    declare @item_seller_user_id int
    declare @bid_status varchar(20)

    -- be optimistic :-)
    set @bid_status = 'ok'

    -- TODO: 5.5.1 set @max_bid_amount to the higest bid amount for that item id
    set @max_bid_amount = (select max(bid_amount) from vb_bids where bid_item_id=@bid_item_id and bid_status='ok')

    -- TODO: 5.5.2 set @item_seller_user_id to the seller_user_id for the item id
    set @item_seller_user_id = (select item_seller_user_id from vb_items where item_id=@bid_item_id)

    -- TODO: 5.5.3 if no bids then set the @max_bid_amount to the item_reserve amount for the item_id
    if (@max_bid_amount is null)
        set @max_bid_amount = (select item_reserve from vb_items where item_id=@bid_item_id)

    -- if you're the item seller, set bid status
    if ( @item_seller_user_id = @bid_user_id)
        set @bid_status = 'item_seller'

    -- if the current bid lower or equal to the last bid, set bid status
    if ( @bid_amount <= @max_bid_amount)
        set @bid_status = 'low_bid'

    -- TODO: 5.5.4 insert the bid at this point and return the bid_id
    insert into vb_bids (bid_user_id, bid_item_id, bid_amount, bid_status)
        values (@bid_user_id, @bid_item_id, @bid_amount, @bid_status)

        if @@ROWCOUNT <> 1 THROW 50001, 'p_place_bid:Insert Error',1
        COMMIT
    return  @@identity
    --

end TRY
begin catch
    ROLLBACK
    ;
    THROW
end CATCH
end
```

Messages

```
2:19:46 PM    Started executing query at Line 1
              Commands completed successfully.
2:19:46 PM    Started executing query at Line 3
              Commands completed successfully.
              Total execution time: 00:00:00.004
```

4. Execute your stored procedure in step 3 to demonstrate the procedure works. Make user 2, Bid $105 on item 36 and show the bid was placed with a SELECT.

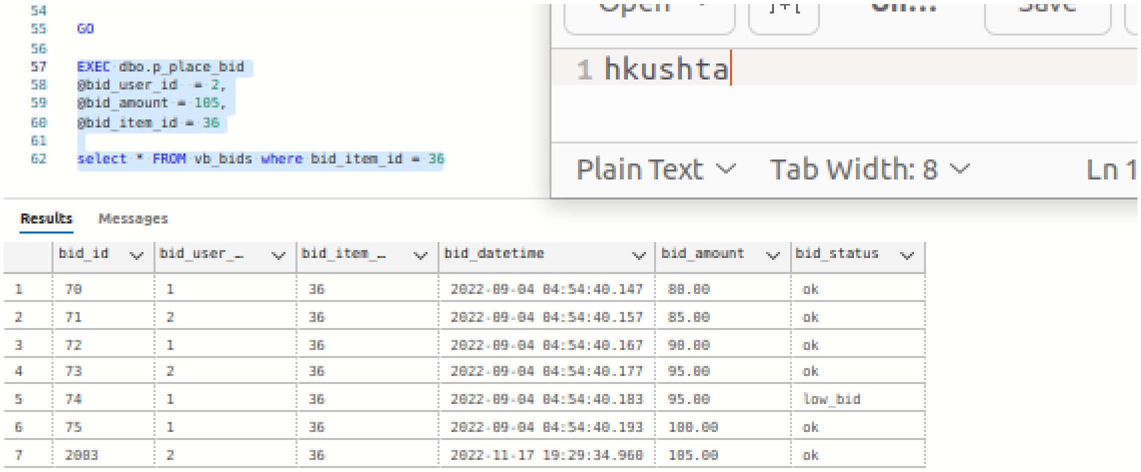
Table Before



	bid_id	bid_user_...	bid_item_...	bid_datetime	bid_amount	bid_status
1	70	1	36	2022-09-04 04:54:40.147	80.00	ok
2	71	2	36	2022-09-04 04:54:40.157	85.00	ok
3	72	1	36	2022-09-04 04:54:40.167	90.00	ok
4	73	2	36	2022-09-04 04:54:40.177	95.00	ok
5	74	1	36	2022-09-04 04:54:40.183	95.00	low_bid
6	75	1	36	2022-09-04 04:54:40.193	100.00	ok

1 hkushta

Table After



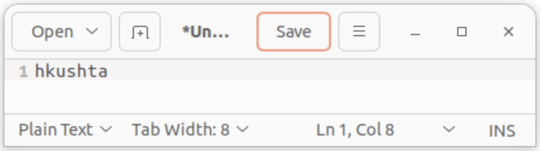
```
GO

EXEC dbo.p_place_bid
@bid_user_id  = 2,
@bid_amount = 105,
@bid_item_id = 36

select * FROM vb_bids where bid_item_id = 36
```

1 hkushta

	bid_id	bid_user_...	bid_item_...	bid_datetime	bid_amount	bid_status
1	70	1	36	2022-09-04 04:54:40.147	80.00	ok
2	71	2	36	2022-09-04 04:54:40.157	85.00	ok
3	72	1	36	2022-09-04 04:54:40.167	90.00	ok
4	73	2	36	2022-09-04 04:54:40.177	95.00	ok
5	74	1	36	2022-09-04 04:54:40.183	95.00	low_bid
6	75	1	36	2022-09-04 04:54:40.193	100.00	ok
7	2083	2	36	2022-11-17 19:29:34.960	105.00	ok

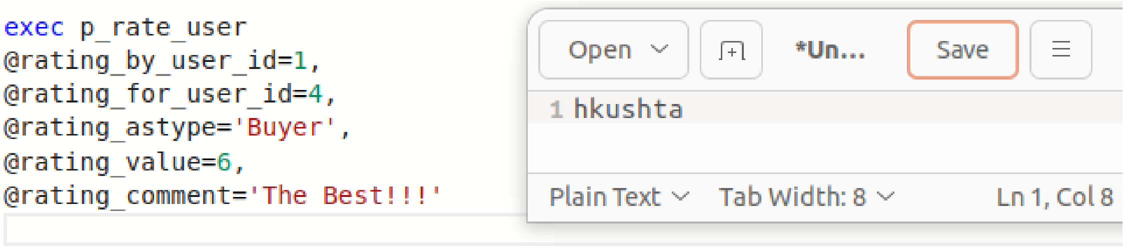
5. Re-write the **p_rate_user** stored procedure from the **VBay** database so that it is transaction safe. Provide a screenshot of the code and its execution.

```
drop PROCEDURE if exists p_rate_user
go
create PROCEDURE [dbo].[p_rate_user]
(
    @rating_by_user_id int,
    @rating_for_user_id int,
    @rating_astype varchar(20),
    @rating_value int,
    @rating_comment text
) as
BEGIN
begin TRANSACTION
begin TRY
insert into vb_user_ratings(rating_by_user_id, rating_for_user_id, rating_astype, rating_value, rating_comment)
VALUES(@rating_by_user_id, @rating_for_user_id, @rating_astype, @rating_value, @rating_comment)
commit
return @@identity
end try
begin CATCH
if not exists (select * from vb_user_ratings where rating_value>5)
throw 50001, 'rating_value only acepts ratings from 1-5',1
ROLLBACK
;
end catch
END
GO
```

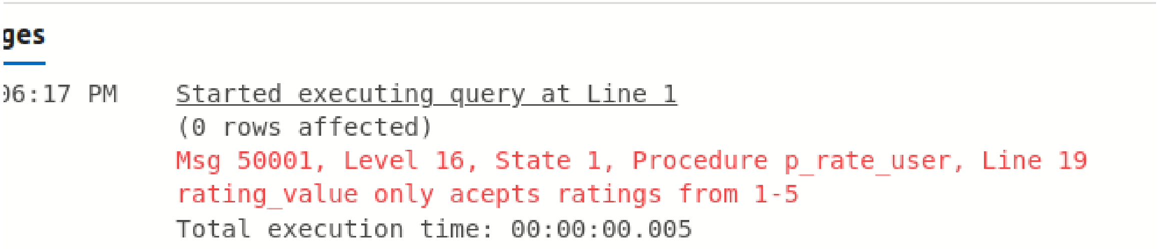


1 hkushta

6. Execute the stored procedure in step 5 to demonstrate the rollback works. You should give a 6 star rating and then execute again where someone attempts to rate themselves. Produce as screen shot as evidence the rollback worked.

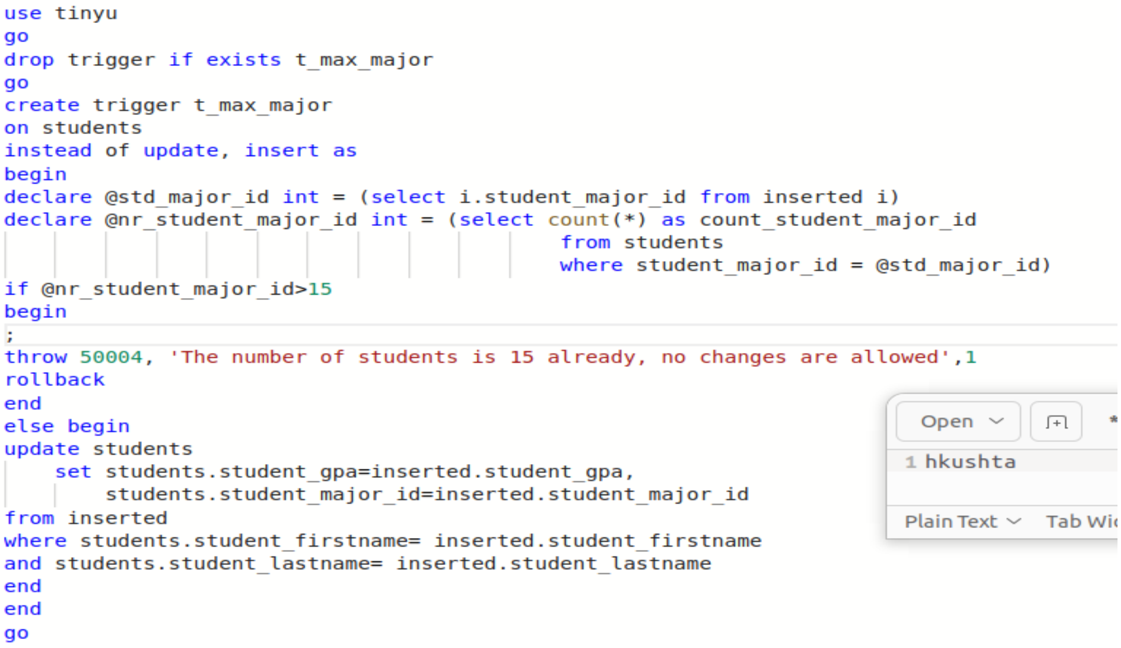


```
exec p_rate_user
@rating_by_user_id=1,
@rating_for_user_id=4,
@rating_astype='Buyer',
@rating_value=6,
@rating_comment='The Best!!!'
```



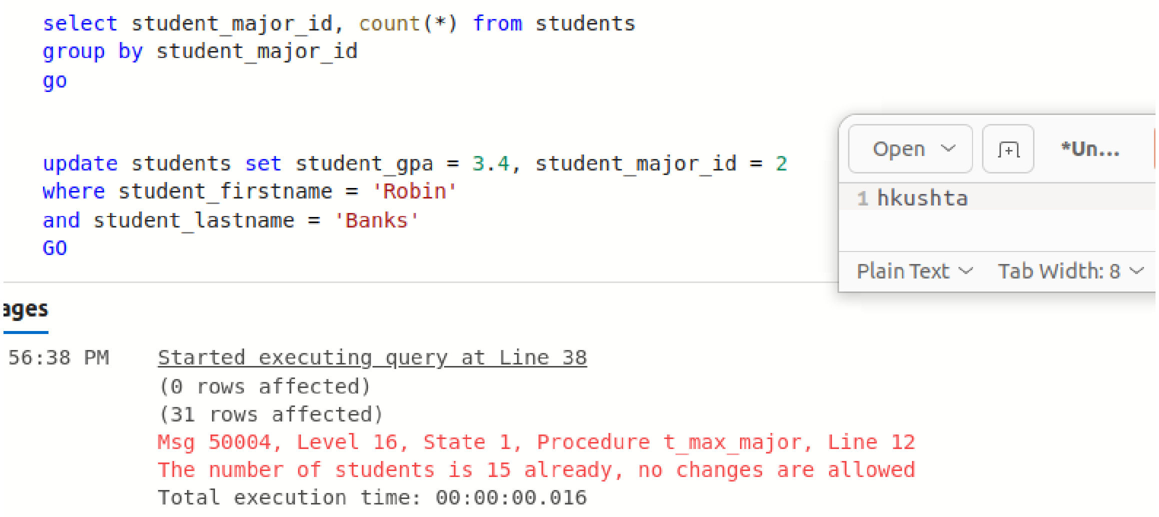
```
)6:17 PM     Started executing query at Line 1
             (0 rows affected)
             Msg 50001, Level 16, State 1, Procedure p_rate_user, Line 19
             rating_value only acepts ratings from 1-5
             Total execution time: 00:00:00.005
```

7. There is a conceptual data requirement which says that no **TinyU** major can have more than 15 students in it. (I know, this seems silly but think of the bigger problem – how to we enforce a specific minimum or maximum cardinality instead of just 1 or “many”?) Write data logic using an instead of trigger to do this.



```
use tinyu
go
drop trigger if exists t_max_major
go
create trigger t_max_major
on students
instead of update, insert as
begin
declare @std_major_id int = (select i.student_major_id from inserted i)
declare @nr_student_major_id int = (select count(*) as count_student_major_id
                                    from students
                                    where student_major_id = @std_major_id)
if @nr_student_major_id>15
begin
;
throw 50004, 'The number of students is 15 already, no changes are allowed',1
rollback
end
else begin
update students
    set students.student_gpa=inserted.student_gpa,
        students.student_major_id=inserted.student_major_id
from inserted
where students.student_firstname= inserted.student_firstname
and students.student_lastname= inserted.student_lastname
end
end
go
```

8. Test step 7 by trying to add or update a student and change their major to ADS. The ADS major has 15 students already. Your code should drop/create the trigger and also test the success and failure of the trigger.



```
select student_major_id, count(*) from students
group by student_major_id
go


update students set student_gpa = 3.4, student_major_id = 2
where student_firstname = 'Robin'
and student_lastname = 'Banks'
GO
```

```
56:38 PM     Started executing query at Line 38
             (0 rows affected)
             (31 rows affected)
             Msg 50004, Level 16, State 1, Procedure t_max_major, Line 12
             The number of students is 15 already, no changes are allowed
             Total execution time: 00:00:00.016
```

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?

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

   Question 7

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.