**Assignment Submission Form**

This form **must** be filled in and completed by the student submitting an assignment.

**Assignments submitted without the completed form will not be marked.**

|  |  |
| --- | --- |
| **Name:** | **Darren Carroll** |
| **School:** | **Victoria University (sim)** |
| **Course Code:** | **INFO 341** |
| **Assignment Title:** | **Assignment One** |
| **Submission Date/Time:** | **04/09/2015** |

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying.

Name (sign): *Darren Carroll*\_\_\_\_\_\_\_\_ Date: 03/09/2015

**Check and Tick all the boxes**

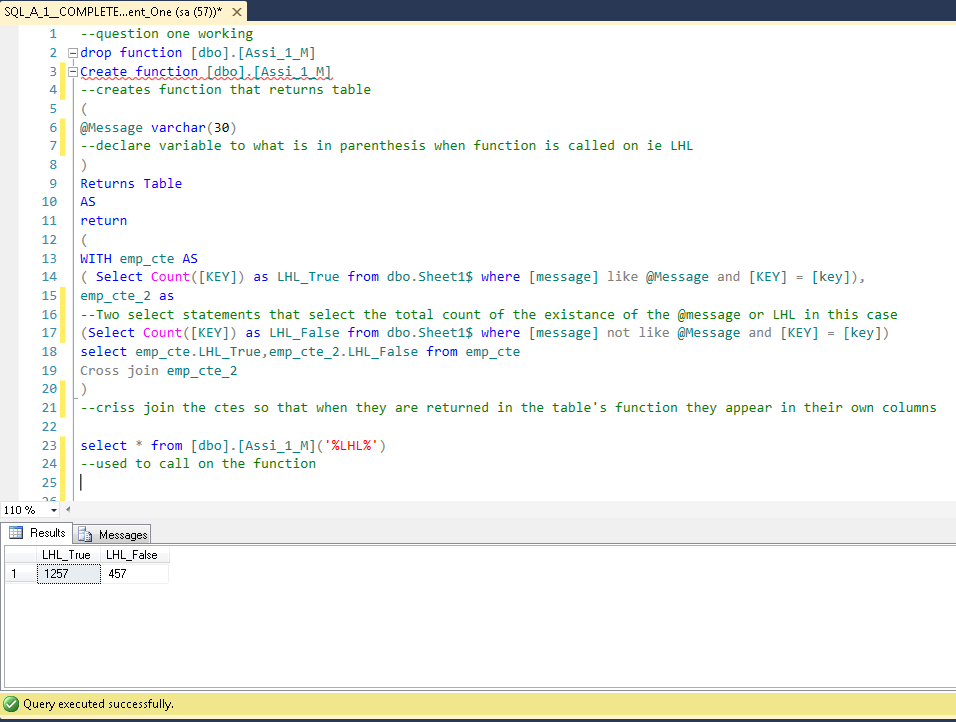
* Submit a paper copy by 9/9 2pm to BOX INFO341 located at the Mezzanine Level Rutherford house.

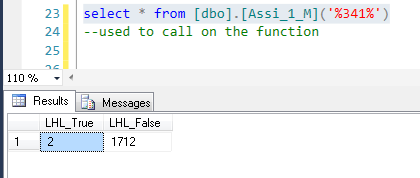
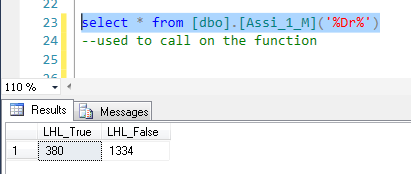
🞏 Upload a copy (we accept only .doc, .docx or pdf file) to BlackBoard by 9/9 2pm.

* Include the assignment submission form.

**Assignments submitted without a submission form will not be marked.**

Question one





Question one Code

--question one working

drop function [dbo].[Assi\_1\_M]

Create function [dbo].[Assi\_1\_M]

--creates function that returns table

(

@Message varchar(30)

--declare variable to what is in parenthesis when function is called on ie LHL

)

Returns Table

AS

return

(

WITH emp\_cte AS

( Select Count([KEY]) as LHL\_True from dbo.Sheet1$ where [message] like @Message and [KEY] = [key]),

emp\_cte\_2 as

--Two select statements that select the total count of the existance of the @message or LHL in this case

(Select Count([KEY]) as LHL\_False from dbo.Sheet1$ where [message] not like @Message and [KEY] = [key])

select emp\_cte.LHL\_True,emp\_cte\_2.LHL\_False from emp\_cte

Cross join emp\_cte\_2

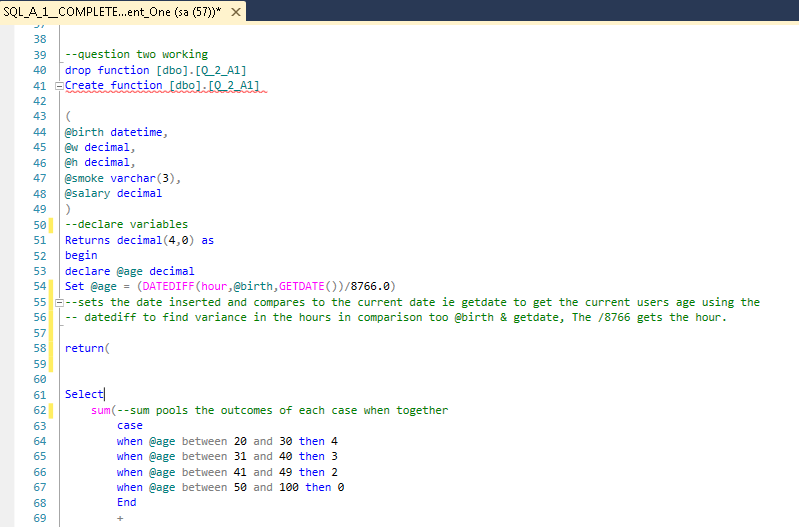
)

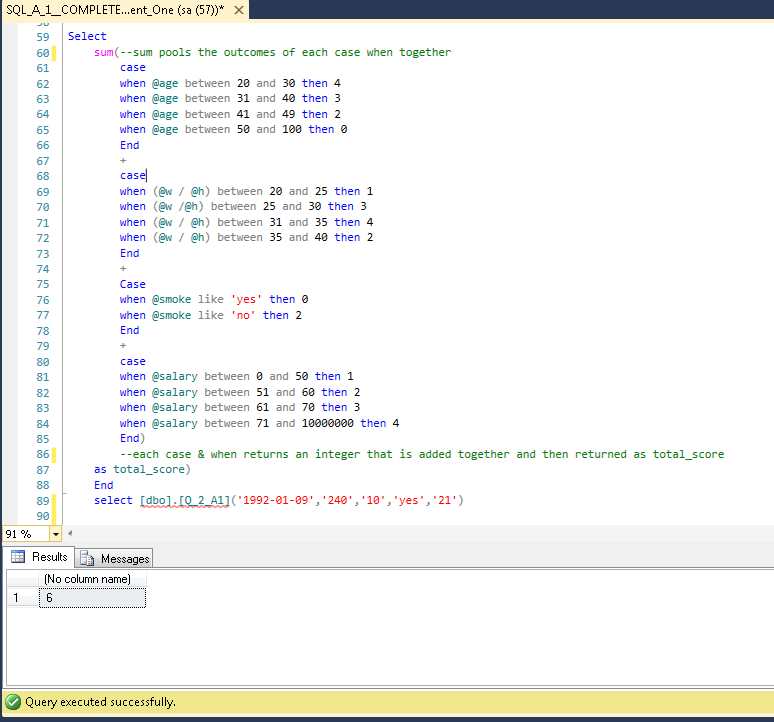
--criss join the ctes so that when they are returned in the table's function they appear in their own columns

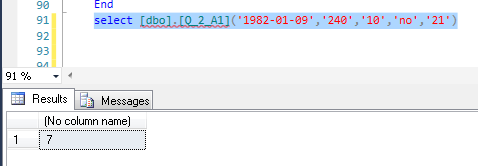
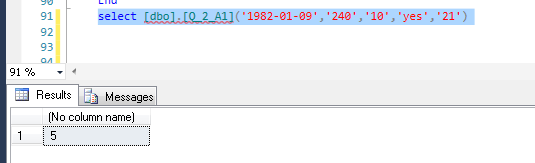
select \* from [dbo].[Assi\_1\_M]('%LHL%')

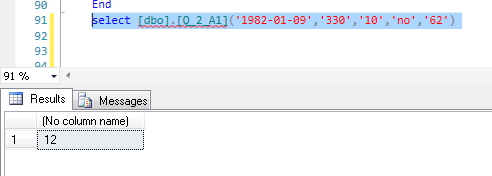
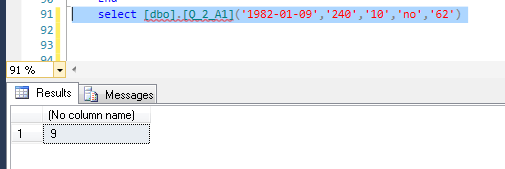
--used to call on the function

Question Two A









Question two A code

Create function [dbo].[Q\_2\_A1]

(

@birth datetime,

@w decimal,

@h decimal,

@smoke varchar(3),

@salary decimal

)

--declare variables

Returns decimal(4,0) as

begin

declare @age decimal

Set @age = (DATEDIFF(hour,@birth,GETDATE())/8766.0)

--sets the date inserted and compares to the current date ie getdate to get the current users age using the

-- datediff to find variance in the hours in comparison too @birth & getdate, The /8766 gets the hour

return(

Select

sum(--sum pools the outcomes of each case when together

case

when @age between 20 and 30 then 4

when @age between 31 and 40 then 3

when @age between 41 and 49 then 2

when @age between 50 and 100 then 0

End

+

case

when (@w / @h) between 20 and 25 then 1

when (@w /@h) between 25 and 30 then 3

when (@w / @h) between 31 and 35 then 4

when (@w / @h) between 35 and 40 then 2

End

+

Case

when @smoke like 'yes' then 0

when @smoke like 'no' then 2

End

+

case

when @salary between 0 and 50 then 1

when @salary between 51 and 60 then 2

when @salary between 61 and 70 then 3

when @salary between 71 and 10000000 then 4

End)

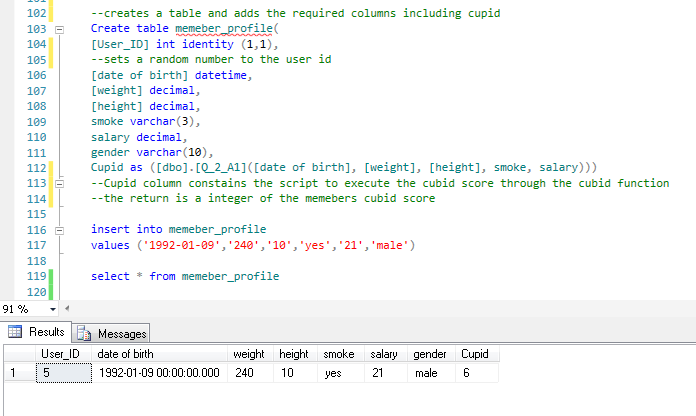
--each case & when returns an integer that is added together and then returned as total\_score

as total\_score)

End

select [dbo].[Q\_2\_A1]('1982-01-09','330','10','no','62')

Question two B



Question two B code

--creates a table and adds the required columns including cupid

Create table memeber\_profile(

[User\_ID] int identity (1,1),

--sets a random number to the user id

[date of birth] datetime,

[weight] decimal,

[height] decimal,

smoke varchar(3),

salary decimal,

gender varchar(10),

Cupid as ([dbo].[Q\_2\_A1]([date of birth], [weight], [height], smoke, salary)))

--Cupid column constains the script to execute the cubid score through the cubid function

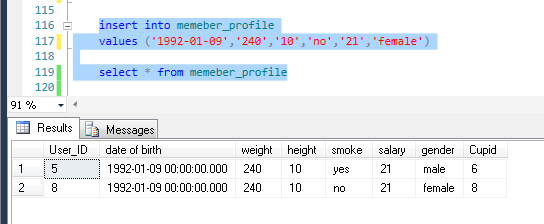
--the return is a integer of the memebers cubid score

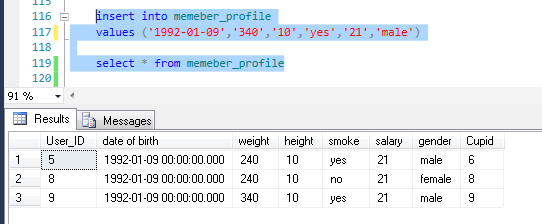
insert into memeber\_profile

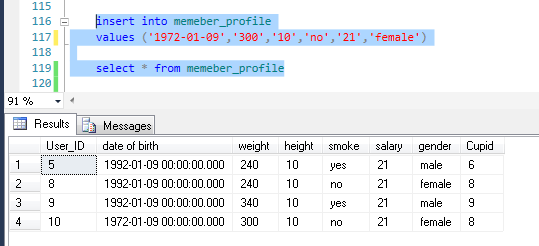
values ('1972-01-09','300','10','yes','30.0','male')

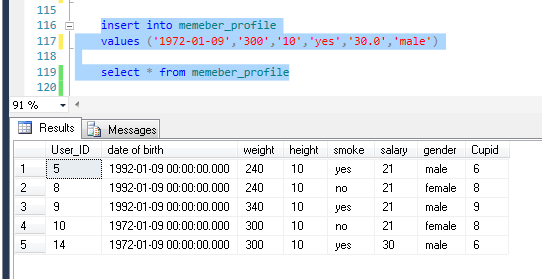
select \* from memeber\_profile

Question two C

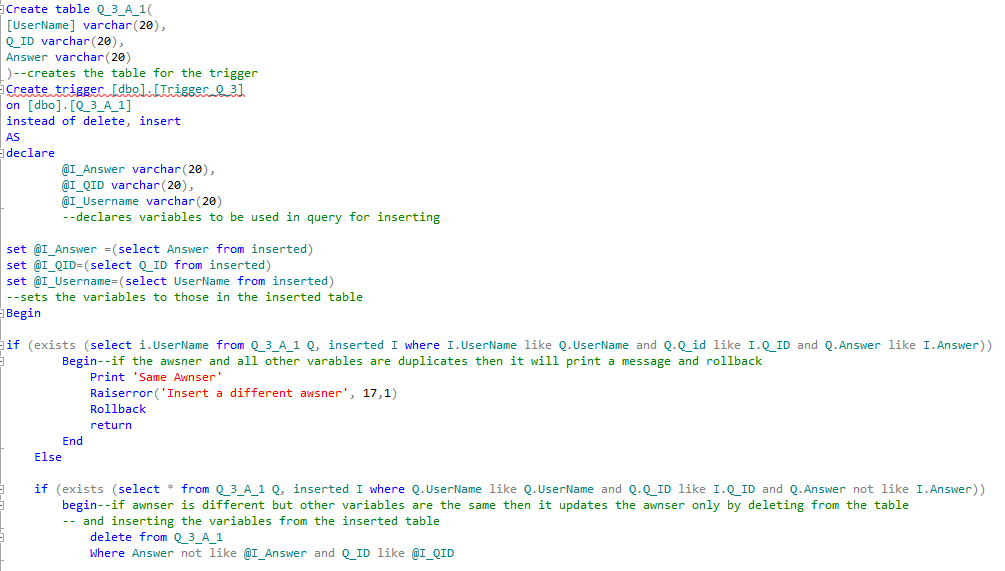


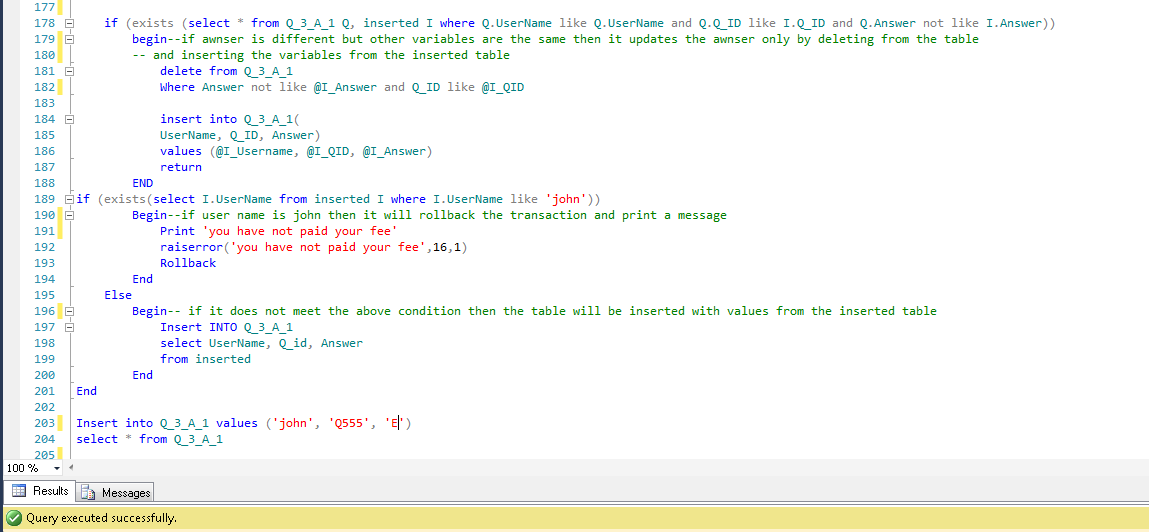






Question three





\*Testing is documented on page 10

Question three Code

Create table Q\_3\_A\_1(

[UserName] varchar(20),

Q\_ID varchar(20),

Answer varchar(20)

)--creates the table for the trigger

Create trigger [dbo].[Trigger\_Q\_3]

on [dbo].[Q\_3\_A\_1]

instead of delete, insert

AS

declare

@I\_Answer varchar(20),

@I\_QID varchar(20),

@I\_Username varchar(20)

--declares variables to be used in query for inserting

set @I\_Answer =(select Answer from inserted)

set @I\_QID=(select Q\_ID from inserted)

set @I\_Username=(select UserName from inserted)

--sets the variables to those in the inserted table

Begin

if (exists (select i.UserName from Q\_3\_A\_1 Q, inserted I where I.UserName like Q.UserName and Q.Q\_id like I.Q\_ID and Q.Answer like I.Answer))

Begin--if the awsner and all other varables are duplicates then it will print a message and rollback

Print 'Same Awnser'

Raiserror('Insert a different awsner', 17,1)

Rollback

return

End

Else

if (exists (select \* from Q\_3\_A\_1 Q, inserted I where Q.UserName like Q.UserName and Q.Q\_ID like I.Q\_ID and Q.Answer not like I.Answer))

begin--if awnser is different but other variables are the same then it updates the awnser only by deleting from the table

-- and inserting the variables from the inserted table

delete from Q\_3\_A\_1

Where Answer not like @I\_Answer and Q\_ID like @I\_QID

insert into Q\_3\_A\_1(

UserName, Q\_ID, Answer)

values (@I\_Username, @I\_QID, @I\_Answer)

return

END

if (exists(select I.UserName from inserted I where I.UserName like 'john'))

Begin--if user name is john then it will rollback the transaction and print a message

Print 'you have not paid your fee'

raiserror('you have not paid your fee',16,1)

Rollback

End

Else

Begin-- if it does not meet the above condition then the table will be inserted with values from the inserted table

Insert INTO Q\_3\_A\_1

select UserName, Q\_id, Answer

from inserted

End

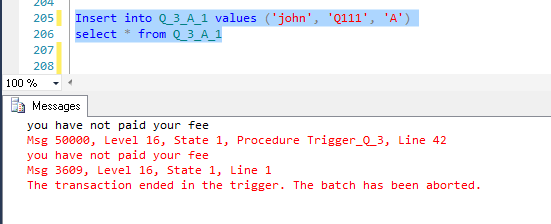
End

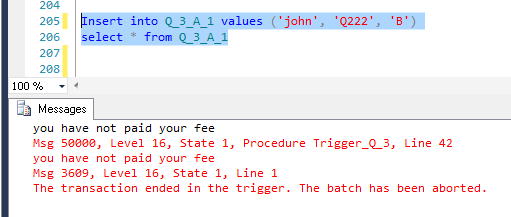
--Insert into Q\_3\_A\_1 values ('john', 'Q555', 'E')

--select \* from Q\_3\_A\_1

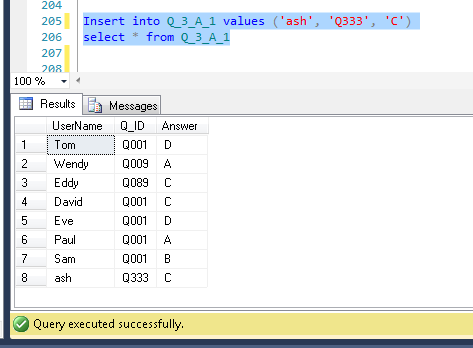
Question three testing

Attempting to insert ‘john’ into the table

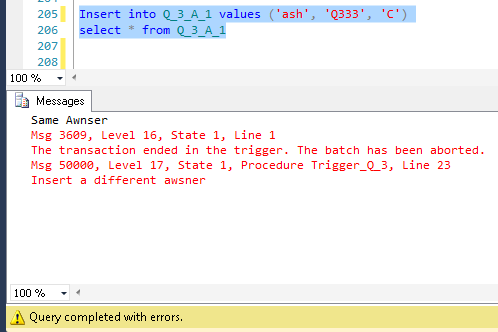




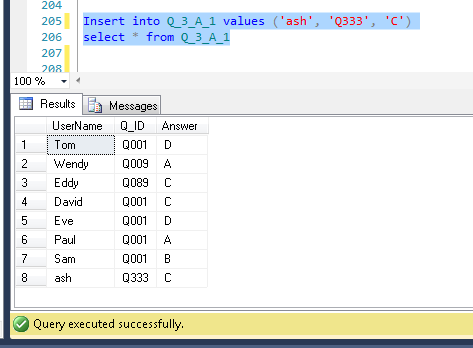
When inserting name that is not john

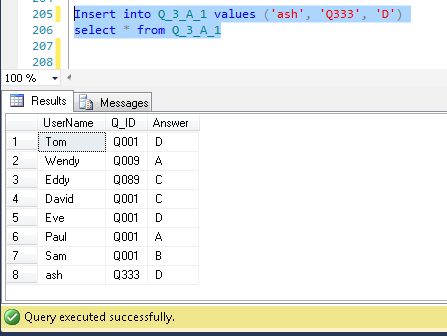


When inserting the same username, QID and Answer.

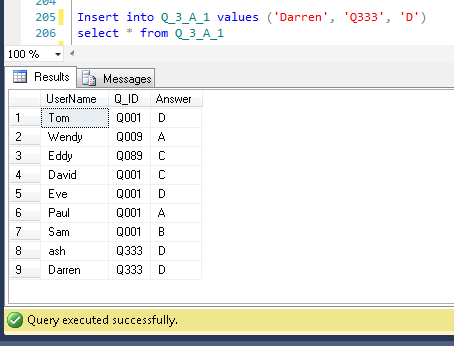


When inserting Different Answer trigger updates row 8

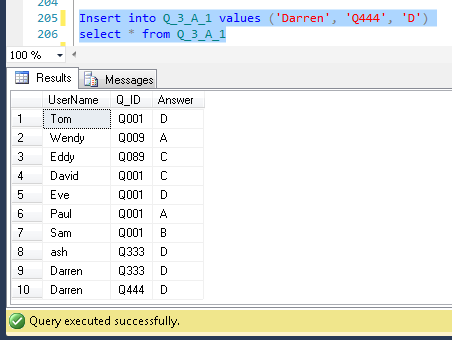




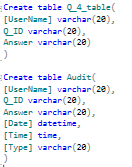
When inserting different username with same QID and Answer

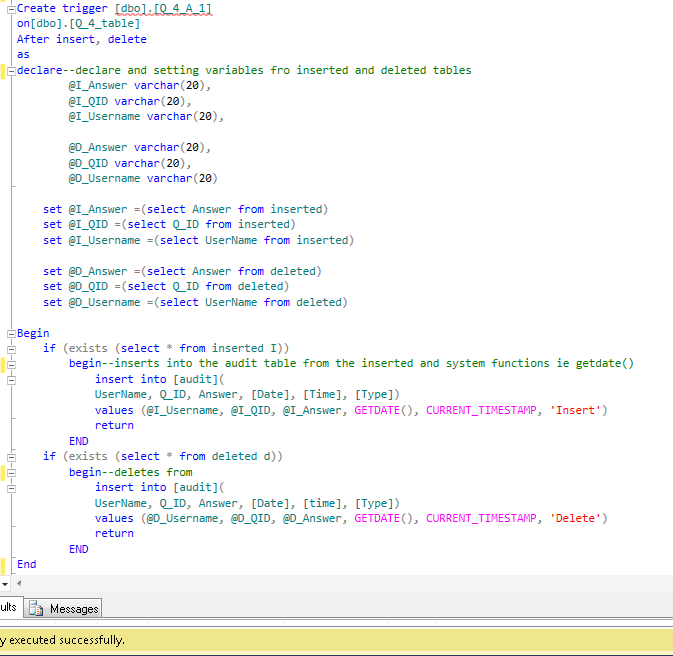


When inserting Different QID with same username and answer



Question four





\*Testing is documented on page 15

Question four code

Create trigger [dbo].[Q\_4\_A\_1]

on[dbo].[Q\_4\_table]

After insert, delete

as

declare--declare and setting variables fro inserted and deleted tables

@I\_Answer varchar(20),

@I\_QID varchar(20),

@I\_Username varchar(20),

@D\_Answer varchar(20),

@D\_QID varchar(20),

@D\_Username varchar(20)

set @I\_Answer =(select Answer from inserted)

set @I\_QID =(select Q\_ID from inserted)

set @I\_Username =(select UserName from inserted)

set @D\_Answer =(select Answer from deleted)

set @D\_QID =(select Q\_ID from deleted)

set @D\_Username =(select UserName from deleted)

Begin

if (exists (select \* from inserted I))

begin--inserts into the audit table from the inserted and system functions ie getdate()

insert into [audit](

UserName, Q\_ID, Answer, [Date], [Time], [Type])

values (@I\_Username, @I\_QID, @I\_Answer, GETDATE(), CURRENT\_TIMESTAMP, 'Insert')

return

END

if (exists (select \* from deleted d))

begin--deletes from

insert into [audit](

UserName, Q\_ID, Answer, [Date], [time], [Type])

values (@D\_Username, @D\_QID, @D\_Answer, GETDATE(), CURRENT\_TIMESTAMP, 'Delete')

return

END

End

--delete from Q\_4\_table where UserName = 'Jerry';

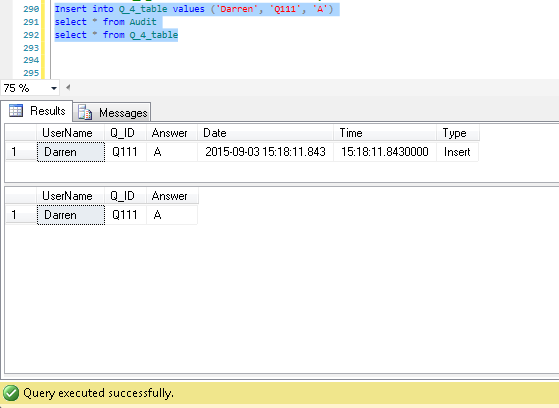
--Insert into Q\_4\_table values ('Jerry', 'Q222', 'B')

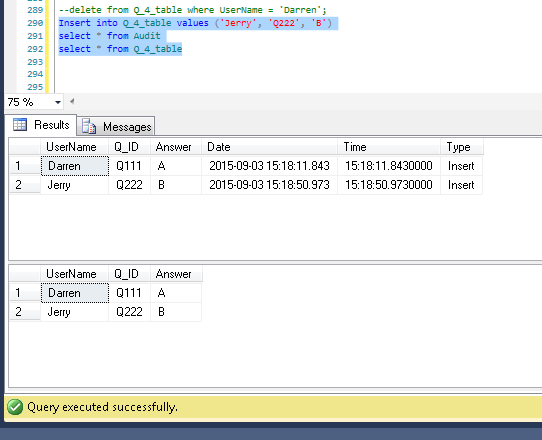
--select \* from Audit

--select \* from Q\_4\_table

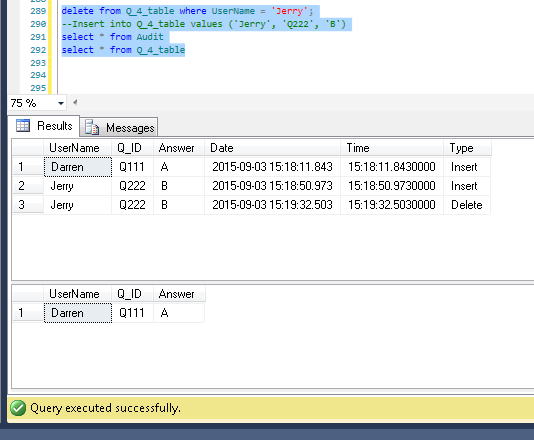
Question four testing

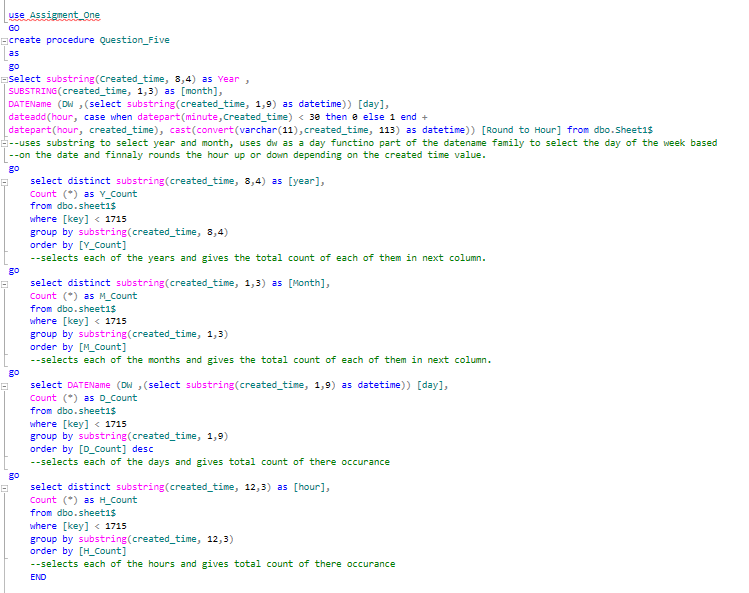
When inserting into the Q\_4\_table

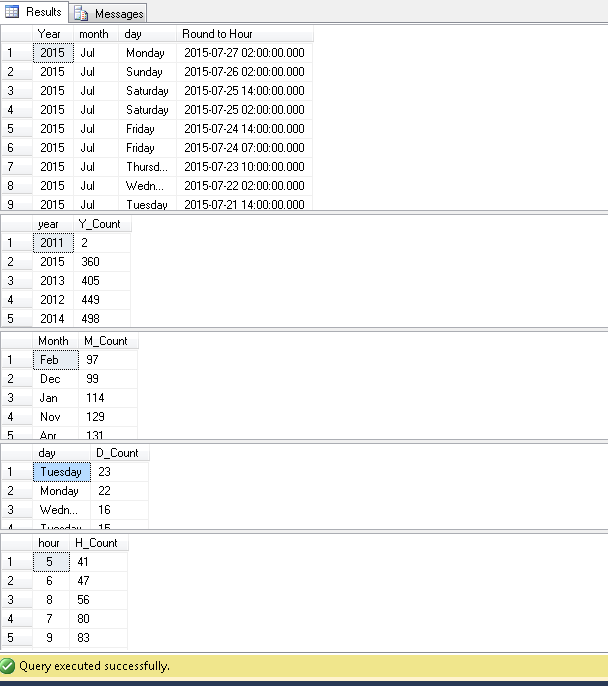




When deleting from the Q\_4\_table



Question five 



Question Five Code

use Assigment\_One

GO

create procedure Question\_Five

as

go

Select substring(Created\_time, 8,4) as Year ,

SUBSTRING(created\_time, 1,3) as [month],

DATEName (DW ,(select substring(created\_time, 1,9) as datetime)) [day],

dateadd(hour, case when datepart(minute,Created\_time) < 30 then 0 else 1 end +

datepart(hour, created\_time), cast(convert(varchar(11),created\_time, 113) as datetime)) [Round to Hour] from dbo.Sheet1$

--uses substring to select year and month, uses dw as a day functino part of the datename family to select the day of the week based

--on the date and finnaly rounds the hour up or down depending on the created time value.

go

select distinct substring(created\_time, 8,4) as [year],

Count (\*) as Y\_Count

from dbo.sheet1$

where [key] < 1715

group by substring(created\_time, 8,4)

order by [Y\_Count]

--selects each of the years and gives the total count of each of them in next column.

go

select distinct substring(created\_time, 1,3) as [Month],

Count (\*) as M\_Count

from dbo.sheet1$

where [key] < 1715

group by substring(created\_time, 1,3)

order by [M\_Count]

--selects each of the months and gives the total count of each of them in next column.

go

select DATEName (DW ,(select substring(created\_time, 1,9) as datetime)) [day],

Count (\*) as D\_Count

from dbo.sheet1$

where [key] < 1715

group by substring(created\_time, 1,9)

order by [D\_Count] desc

--selects each of the days and gives total count of there occurance

go

select distinct substring(created\_time, 12,3) as [hour],

Count (\*) as H\_Count

from dbo.sheet1$

where [key] < 1715

group by substring(created\_time, 12,3)

order by [H\_Count]

--selects each of the hours and gives total count of there occurance

END