

Lab 13: Triggers

Nidhi Surya Prakash
SUID:215895619

Purpose:

Create a trigger.¹

Create a trigger¹ on insert/delete/update of the Enrollment table:

Scripts common:

```
select * from nsuryapr.CourseEnrollment
select * from nsuryapr.Courses
```

Tables before changes:

EnrollmentId	StudentId	CourseId	FinalGrade
1	01-HJPotter	1	96
2	03-HJGranger	1	92
3	03-HJGranger	4	99
4	02-RBWeasley	1	91
5	02-RBWeasley	6	NULL
6	01-HJPotter	2	NULL
7	03-HJGranger	2	NULL
8	03-HJGranger	2	NULL
9			
10			
11			
12			
13			
14			
15			

¹ You can create 3 triggers, one per action. If you create a single trigger that can do all 3, you get an extra 5% bonus.

The screenshot shows the SSMS interface. In the Object Explorer on the left, under SERVERS, there is a tree view of database objects for 'nsuryapr'. The 'Courses' table is expanded, showing columns like CourseId, CourseCode, CourseTitle, Faculty, and OpenSeats. The 'LetterGrades' table is also expanded, showing columns like LetterGrade and Description. Other tables like 'users' and 'CourseEnrollment' are listed under their respective schemas.

In the center, there are two query panes: 'SQLQuery_2' and 'SQLQuery_1'. The 'SQLQuery_2' pane contains the following T-SQL script:

```

1 select * from nsuryapr.CourseEnrollment
2 select * from nsuryapr.Courses
3
4
5
6
7
8
9

```

The 'RESULTS' pane displays the output of the second query, which lists seven rows of course information:

CourseId	CourseCode	CourseTitle	Faculty	OpenSeats
1	DADA101	Defence Again...	16-Rhagrid	3
2	DADA201	Defence Again...	16-Rhagrid	1
3	DADA301	Defence Again...	16-Rhagrid	1
4	CHMS101	Charms BASIC	11-Fflitwick	1
5	CHMS201	Charms INTER...	11-Fflitwick	0
6	CHMS301	Charms ADV...	11-Fflitwick	4
7	HOM101	History of Magi...	NULL	10

The 'MESSAGES' pane at the bottom shows the execution details:

- Started executing query at Line 2
- (7 rows affected)
- Total execution time: 00:00:00.005

- when inserting a new record, update the Courses table's OpenSeats for the course into which the student is being enrolled.

Script:

```

CREATE TRIGGER courseTriggers1 ON nsuryapr.CourseEnrollment
AFTER INSERT
AS
SET NOCOUNT ON;
BEGIN
UPDATE nsuryapr.Courses
SET OpenSeats = OpenSeats - (select count(*) from inserted where inserted.CourseId =
nsuryapr.Courses.CourseId)
WHERE Courses.CourseId in (select CourseId from inserted)
END;

```

Insert statement

```

Insert into nsuryapr.CourseEnrollment(StudentId ,CourseId,FinalGrade)
VALUES ('03-HJGranger',7,null),('02-RBWeasley',2,null),('01-HJPotter',7,null);

```

Execution of script for insert trigger

The screenshot shows the SQL Server Management Studio interface. On the left is the Object Explorer tree, which includes nodes for SERVERS, nsuryapr.CourseGrade, nsuryapr.Courses, and nsuryapr.LetterGrades. The nsuryapr.LetterGrades node is currently selected. On the right is the Results pane, which displays the output of a T-SQL script. The script creates two triggers, courseTriggers1 and courseTriggers2, on the nsuryapr.CourseEnrollment table. The courseTriggers1 trigger is defined as follows:

```
1 CREATE TRIGGER courseTriggers1 ON nsuryapr.CourseEnrollment
2 AFTER INSERT
3 AS
4 SET NOCOUNT ON;
5 BEGIN
6 UPDATE nsuryapr.Courses
7 SET OpenSeats = OpenSeats - (select count(*) from inserted where inserted.CourseId = nsuryapr.Courses.CourseId)
8 WHERE Courses.CourseId in (select CourseId from inserted)
9 END;
```

The courseTriggers2 trigger is defined as follows:

```
12 /*CREATE TRIGGER courseTriggers2 ON nsuryapr.CourseEnrollment
13 AFTER DELETE
14 AS
15 SET NOCOUNT ON;
16 BEGIN
17 UPDATE nsuryapr.Courses
18 SET OpenSeats = OpenSeats + (select count(*) from deleted where deleted.CourseId = Courses.CourseId)
19 WHERE Courses.CourseId in (select CourseId from deleted)
20 END;*/
```

The Messages pane at the bottom shows the execution log:

```
10:39:08 AM Started executing query at Line 1
Commands completed successfully.
Total execution time: 00:00:00.005
```

Execution of insert statement

The screenshot shows the SQL Server Management Studio interface. The Object Explorer tree is identical to the previous screenshot, with nsuryapr.LetterGrades selected. The Results pane displays the output of an insert statement into the nsuryapr.CourseEnrollment table:

```
1 select * from nsuryapr.CourseEnrollment
2 select * from nsuryapr.Courses
3
4
5
6
7 Insert into nsuryapr.CourseEnrollment(StudentId ,CourseId,FinalGrade)
8 VALUES ('03-HJGranger',7,null),('02-RBWeasley',2,null),('01-HJPotter',7,null);
9
10 Delete from nsuryapr.CourseEnrollment where StudentId = '01-HJPotter' AND CourseId = 4
11
12 Update nsuryapr.CourseEnrollment
13 set CourseId=3, FinalGrade=null
14 WHERE EnrollmentId in (1,3)
15
```

The Messages pane shows the execution log:

```
10:39:43 AM Started executing query at Line 7
(3 rows affected)
Total execution time: 00:00:00.021
```

CourseEnrollment table after insert trigger

Inserted for enrollment id = 12,13,14

I had previously added and deleted for enrollment id 10,11 and did not reset.

The screenshot shows the Object Explorer on the left with the 'nsuryapr.CourseGrade' and 'nsuryapr.Courses' nodes expanded. The 'nsuryapr.CourseGrade' node has its 'Triggers' folder expanded, showing 'LetterGradeD' and 'GradeValueD'. The 'nsuryapr.Courses' node has its 'Triggers' folder expanded, showing 'OpenSeatsD'. The 'nsuryapr.LetterGrades' node is selected. The 'SQL Query' window on the right contains the following code:

```
1 select * from nsuryapr.CourseEnrollment
2 select * from nsuryapr.Courses
```

The 'RESULTS' tab displays the data from the 'CourseEnrollment' table:

	EnrollmentId	StudentId	CourseId	FinalGrade
1	1	01-HJPotter	1	96
2	3	03-HJGranger	1	92
3	4	03-HJGranger	4	99
4	5	02-RBWeasley	1	91
5	6	02-RBWeasley	6	NULL
6	7	01-HJPotter	2	NULL
7	8	03-HJGranger	2	NULL
8	9	03-HJGranger	2	NULL
9	12	03-HJGranger	7	NULL
10	13	02-RBWeasley	2	NULL
11	14	01-HJPotter	7	NULL

The 'MESSAGES' tab shows the execution details:

```
10:42:02 AM Started executing query at Line 1
(11 rows affected)
Total execution time: 00:00:00.005
```

Courses table after insert trigger

CourseID = 7 open seats became 8 from 10 and for CourseID = 2 openseats became 0 from 1

The screenshot shows the Object Explorer on the left with the 'nsuryapr.CourseGrade' and 'nsuryapr.Courses' nodes expanded. The 'nsuryapr.CourseGrade' node has its 'Triggers' folder expanded, showing 'GradingScaleD' and 'LetterGradeD'. The 'nsuryapr.Courses' node has its 'Triggers' folder expanded, showing 'OpenSeatsD'. The 'nsuryapr.LetterGrades' node is selected. The 'SQL Query' window on the right contains the following code:

```
1 select * from nsuryapr.CourseEnrollment
2 select * from nsuryapr.Courses
```

The 'RESULTS' tab displays the data from the 'Courses' table:

	CourseId	CourseCode	CourseTitle	Faculty	OpenSeats
1	1	DADA101	Defence Again...	16-Rhagrid	3
2	2	DADA201	Defence Again...	16-Rhagrid	0
3	3	DADA301	Defence Again...	16-Rhagrid	1
4	4	CHMS101	Charms BASIC	11-Filitwick	1
5	5	CHMS201	Charms INTER...	11-Filitwick	0
6	6	CHMS301	Charms ADVA...	11-Filitwick	4
7	7	HOM101	History of Magi...	NULL	8

The 'MESSAGES' tab shows the execution details:

```
10:40:12 AM Started executing query at Line 2
(7 rows affected)
Total execution time: 00:00:00.007
```

- b. when deleting a record, update the Courses table's OpenSeats for the course from which the student was deleted.

SCRIPT:

```

CREATE TRIGGER courseTiggers2 ON nsuryapr.CourseEnrollment
AFTER DELETE
AS
SET NOCOUNT ON;
BEGIN
UPDATE nsuryapr.Courses
SET OpenSeats = OpenSeats + (select count(*) from deleted where deleted.CourseId =
Courses.CourseId)
WHERE Courses.CourseId in (select CourseId from deleted)
END;

```

Delete statement

```
Delete from nsuryapr.CourseEnrollment where EnrollmentId > 12
```

Execution of script for delete trigger

The screenshot shows the SSMS interface with two query panes. The left pane displays the database structure for the 'nsuryapr' database, specifically the 'Courses' and 'CourseEnrollment' tables. The right pane shows the execution of a script. The script creates a trigger 'courseTiggers2' on the 'CourseEnrollment' table that updates the 'OpenSeats' column in the 'Courses' table after a delete operation. It then deletes rows from the 'CourseEnrollment' table where the 'EnrollmentId' is greater than 12. The 'MESSAGES' pane at the bottom right shows the execution log, indicating the start of the query at 10:50:18, successful completion, and a total execution time of 0:00:00.005.

```

CREATE TRIGGER courseTiggers2 ON nsuryapr.CourseEnrollment
AFTER DELETE
AS
SET NOCOUNT ON;
BEGIN
UPDATE nsuryapr.Courses
SET OpenSeats = OpenSeats + (select count(*) from deleted where deleted.CourseId =
Courses.CourseId)
WHERE Courses.CourseId in (select CourseId from deleted)
END;

```

10:50:18 AM Started executing query at Line 12.
Commands completed successfully.
Total execution time: 0:00:00.005

Execution of delete statement

The screenshot shows the SQL Server Management Studio interface. On the left is the Object Explorer tree, which includes nodes for 'nsuryapr.CourseGrade' and 'nsuryapr.Courses'. The 'nsuryapr.Courses' node is expanded, showing columns like CourseId, CourseCode, CourseTitle, Faculty, and OpenSeats. The 'nsuryapr.CourseEnrollment' node is also visible under 'nsuryapr'. In the center is the 'SQLQuery_2' window containing the following SQL code:

```
1 select * from nsuryapr.CourseEnrollment
2 select * from nsuryapr.Courses
3
4
5
6 Insert into nsuryapr.CourseEnrollment(StudentId ,CourseId,FinalGrade)
7 VALUES ('03-HJGranger',7,null),('02-RBWeasley',2,null),('01-HJPotter',7,null);
8
9 Delete from nsuryapr.CourseEnrollment where EnrollmentId > 12
10
```

Below the code, the 'MESSAGES' pane shows the execution results:

```
10:55:01 AM Started executing query at Line 9
(2 rows affected)
Total execution time: 00:00:00.017
```

At the bottom, the status bar indicates 0 rows, 00:00:00, LCS-VC-MSSQL.ad.syr.edu : CSE581labs, Ln 10, Col 1 (62 selected), Spaces: 4, UTF-8, LF, SQL, MSSQL, and a notification icon.

CourseEnrollment table after delete trigger

CourseID = 13,14 DELETED SINCE >12

The screenshot shows the SQL Server Management Studio interface. The Object Explorer tree is identical to the previous screenshot. The 'nsuryapr.CourseEnrollment' node is selected in the center window. The 'RESULTS' pane displays the data from the CourseEnrollment table:

	EnrollmentId	StudentId	CourseId	FinalGrade
1	1	01-HJPotter	1	96
2	3	03-HJGranger	1	92
3	4	03-HJGranger	4	99
4	5	02-RBWeasley	1	91
5	6	02-RBWeasley	6	NULL
6	7	01-HJPotter	2	NULL
7	8	03-HJGranger	2	NULL
8	9	03-HJGranger	2	NULL
9	12	03-HJGranger	7	NULL

The 'MESSAGES' pane shows the execution results:

```
10:55:07 AM Started executing query at Line 1
(9 rows affected)
Total execution time: 00:00:00.005
```

At the bottom, the status bar indicates 9 rows, 00:00:00, LCS-VC-MSSQL.ad.syr.edu : CSE581labs, Ln 2, Col 1 (40 selected), Spaces: 4, UTF-8, LF, SQL, MSSQL, and a notification icon.

Courses table after delete trigger

Courseid = 7 openseats increased by 1 and for courseid = 2 increased by 1

The screenshot shows the SSMS interface with the following details:

- Servers:** LCS-VC-MSSQL.ad.syr.edu
- SQL Query:** CSE581labs
- Script:**

```
1 select * from nsuryapr.CourseEnrollment
2 select * from nsuryapr.Courses
3
4
5
6 Insert into nsuryapr.CourseEnrollment(StudentId ,CourseId,FinalGrade)
7 VALUES ('03-HGranger',7,null),('02-RWeasley',2,null),('01-HPotter',7,null);
8
9 Delete from nsuryapr.CourseEnrollment where EnrollmentId > 12
10
```
- Results:** A table showing the Courses table with 7 rows. The last row (CourseID 7) has OpenSeats increased to 9.

CourseId	CourseCode	CourseTitle	Faculty	OpenSeats
1	DADA101	Defence Again...	16-Rhagrid	3
2	DADA201	Defence Again...	16-Rhagrid	1
3	DADA301	Defence Again...	16-Rhagrid	1
4	CHMS101	Charms BASIC	11-Flitwick	1
5	CHMS201	Charms INTER...	11-Flitwick	0
6	CHMS301	Charms ADVA...	11-Flitwick	4
7	HOM101	History of Magi...	NULL	9

- Messages:**
 - 10:55:15 AM Started executing query at Line 2
 - (7 rows affected)
 - Total execution time: 00:00:00.005
- Status Bar:** 7 rows 00:00:00 LCS-VC-MSSQL.ad.syr.edu : CSE581labs Ln 3, Col 1 (31 selected) Spaces: 4 UTF-8 LF SQL MSSQL

- c. if an update happened on the CourseId (student switched courses), adjust the Courses table's OpenSeats accordingly.

SCRIPT:

```
CREATE TRIGGER courseTiggers3 ON nsuryapr.CourseEnrollment
AFTER UPDATE
AS
IF UPDATE(CourseId)
BEGIN

UPDATE nsuryapr.Courses
SET Courses.OpenSeats = Courses.OpenSeats + (Select COUNT(*) from deleted
where deleted.CourseId = Courses.CourseId)
WHERE Courses.CourseId in (select CourseId from deleted)

UPDATE nsuryapr.Courses
SET OpenSeats = OpenSeats - (select count(*) from inserted
where inserted.CourseId = nsuryapr.Courses.CourseId)
WHERE Courses.CourseId in (select CourseId from inserted)

END;
```

Update statement:

```
Update nsuryapr.CourseEnrollment  
set CourseId=3  
WHERE EnrollmentId in (1,3)
```

Execution of script for update trigger

The screenshot shows the SQL Server Management Studio interface. On the left is the Object Explorer tree, which includes nodes for 'nsuryapr.CourseGrade' (Columns, Keys, Constraints, Triggers, Indexes, Statistics), 'nsuryapr.Courses' (Columns, Keys, Constraints, Triggers, Indexes, Statistics), and 'nsuryapr.LetterGrades'. In the center is a query window titled 'SQLQuery_2' with the following script:

```
17 UPDATE nsuryapr.Courses  
18 SET OpenSeats = OpenSeats + (select count(*) from deleted where deleted.CourseId = Courses.CourseId)  
19 WHERE Courses.CourseId in (select CourseId from deleted)  
20 END;/  
21  
22  
23 CREATE TRIGGER courseTiggers3 ON nsuryapr.CourseEnrollment  
24 AFTER UPDATE  
25 AS  
26 IF UPDATE(CourseId)  
27 BEGIN  
28  
29 UPDATE nsuryapr.Courses  
30 SET Courses.OpenSeats = Courses.OpenSeats + (Select COUNT(*) from deleted  
where deleted.CourseId = Courses.CourseId)  
31 WHERE Courses.CourseId in (select CourseId from deleted)  
32  
33  
34  
35 UPDATE nsuryapr.Courses  
36 SET OpenSeats = OpenSeats - (select count(*) from inserted  
where inserted.CourseId = nsuryapr.Courses.CourseId)  
37 WHERE Courses.CourseId in (select CourseId from inserted)  
38  
39  
40 END;  
41  
42
```

Below the script is the 'MESSAGES' pane, which displays the following output:

```
11:00:48 AM Started executing query at Line 23  
Commands completed successfully.  
Total execution time: 00:00:00.007
```

At the bottom of the screen, the status bar shows: 0 rows 00:00:00 LCS-VC-MSSQL.ad.syr.edu : CSE581labs Ln 40, Col 5 (504 selected) Spaces: 4 UTF-8 LF SQL MSSQL 1 1.

Execution of update statement

The screenshot shows the SQL Server Management Studio interface. On the left is the Object Explorer tree, identical to the previous screenshot. In the center is a query window titled 'SQLQuery_2' with the following script:

```
1 select * from nsuryapr.CourseEnrollment  
2 select * from nsuryapr.Courses  
3  
4  
5  
6 Insert into nsuryapr.CourseEnrollment(StudentId ,CourseId,FinalGrade)  
7 VALUES ('03-HGGranger',7,null),('02-RBWeasley',2,null),('01-HJPotter',7,null);  
8  
9 Delete from nsuryapr.CourseEnrollment where EnrollmentId > 12  
10  
11 Update nsuryapr.CourseEnrollment  
12 set CourseId=3  
13 WHERE EnrollmentId in (1,3)  
14
```

Below the script is the 'MESSAGES' pane, which displays the following output:

```
11:06:36 AM Started executing query at Line 11  
(1 row affected)  
(1 row affected)  
(2 rows affected)  
Total execution time: 00:00:00.022
```

At the bottom of the screen, the status bar shows: 0 rows 00:00:00 LCS-VC-MSSQL.ad.syr.edu : CSE581labs Ln 13, Col 28 (75 selected) Spaces: 4 UTF-8 LF SQL MSSQL 1 1.

CourseEnrollment table after update trigger

The screenshot shows the Object Explorer on the left with the 'nsuryapr.CourseGrade' node expanded. The 'Tables' node under 'nsuryapr' is also expanded. The 'SQLQuery_2' tab in the center contains the following SQL code:

```

1 select * from nsuryapr.CourseEnrollment
2 select * from nsuryapr.Courses
3
4
5
6 Insert into nsuryapr.CourseEnrollment(StudentId ,CourseId,FinalGrade)
7 VALUES ('03-HJGranger',7,null),('02-RBWeasley',2,null),('01-HJPotter',7,null);
8
9 Delete from nsuryapr.CourseEnrollment where EnrollmentId > 12
10
11 Update nsuryapr.CourseEnrollment
12 set CourseId=3
13 WHERE EnrollmentId in (1,3)
14

```

The 'RESULTS' pane shows the updated data in the CourseEnrollment table:

	EnrollmentId	StudentId	CourseId	FinalGrade
1	1	01-HJPotter	3	96
2	3	03-HJGranger	3	92
3	4	03-HJGranger	4	99
4	5	02-RBWeasley	1	91
5	6	02-RBWeasley	6	NULL
6	7	01-HJPotter	2	NULL
7	8	03-HJGranger	2	NULL
8	9	03-HJGranger	2	NULL
9	12	03-HJGranger	7	NULL

The 'MESSAGES' pane shows the execution details:

- Started executing query at Line 1
- (9 rows affected)
- Total execution time: 00:00:00.096

Courses table after update trigger

The screenshot shows the Object Explorer on the left with the 'nsuryapr.CourseGrade' node expanded. The 'Tables' node under 'nsuryapr' is also expanded. The 'SQLQuery_2' tab in the center contains the same SQL code as the previous screenshot:

```

1 select * from nsuryapr.CourseEnrollment
2 select * from nsuryapr.Courses
3
4
5
6 Insert into nsuryapr.CourseEnrollment(StudentId ,CourseId,FinalGrade)
7 VALUES ('03-HJGranger',7,null),('02-RBWeasley',2,null),('01-HJPotter',7,null);
8
9 Delete from nsuryapr.CourseEnrollment where EnrollmentId > 12
10
11 Update nsuryapr.CourseEnrollment
12 set CourseId=3
13 WHERE EnrollmentId in (1,3)
14

```

The 'RESULTS' pane shows the updated data in the Courses table:

	CourseId	CourseCode	CourseTitle	Faculty	OpenSeats
1	1	DADA101	Defence Again...	16-Rhagrid	5
2	2	DADA201	Defence Again...	16-Rhagrid	1
3	3	DADA301	Defence Again...	16-Rhagrid	-1
4	4	CHMS101	Charms BASIC	11-Fflitwick	1
5	5	CHMS201	Charms INTER...	11-Fflitwick	0
6	6	CHMS301	Charms ADVA...	11-Fflitwick	4
7	7	HOM101	History of Magi...	NULL	9

The 'MESSAGES' pane shows the execution details:

- Started executing query at Line 2
- (7 rows affected)
- Total execution time: 00:00:00.005

