МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технологический университет «СТАНКИН»
(ФГБОУ ВО МГТУ «СТАНКИН»)

Институт
информационных технологий

Кафедра
информационных систем

Отчет по лабораторной работе №11

по дисциплине «**Управление данными**»
на тему: Работа с триггерами в SQL Server Management Studio

Студент **Музафаров К. Р.**
группа ИДБ–21–06

подпись

Руководитель **Быстрикова В. А.**
старший преподаватель

подпись

Москва 2023 г.

ЦЕЛЬ РАБОТЫ

Ознакомление с понятием триггера, изучение способов создания триггеров на языке Transact-SQL, а также приобретение практических навыков работы с триггерами в среде SQL Server Management Studio.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Триггер – это откомпилированная SQL-процедура, исполнение которой обусловлено наступлением определенных событий внутри реляционной базы данных.

Триггер представляет собой специальный тип хранимых процедур, запускаемых сервером автоматически при попытке изменения данных в таблицах, с которыми триггеры связаны. Каждый триггер привязывается к конкретной таблице. Все производимые им модификации данных рассматриваются как одна транзакция. В случае обнаружения ошибки или нарушения целостности данных происходит откат этой транзакции. Тем самым внесение изменений запрещается. Отменяются также все изменения, уже сделанные триггером.

Триггеры – особый инструмент SQL-сервера, используемый для поддержания целостности данных в базе данных. С помощью ограничений целостности, правил и значений по умолчанию не всегда можно добиться нужного уровня функциональности. Часто требуется реализовать сложные алгоритмы проверки данных, гарантирующие их достоверность и реальность. Кроме того, иногда необходимо отслеживать изменения значений таблицы, чтобы нужным образом изменить связанные данные. Триггеры можно рассматривать как своего рода фильтры, вступающие в действие после выполнения всех операций в соответствии с правилами, стандартными значениями и т.д.

Применение триггеров большей частью весьма удобно для пользователей базы данных. И все же их использование часто связано с дополнительными затратами ресурсов на операции ввода/вывода. В том

случае, когда тех же результатов (с гораздо меньшими непроизводительными затратами ресурсов) можно добиться с помощью хранимых процедур или прикладных программ, применение триггеров нецелесообразно.

В отличие от обычной подпрограммы, триггер выполняется неявно в каждом случае возникновения триггерного события, к тому же он не имеет аргументов. Приведение его в действие иногда называют запуском триггера.

С помощью триггеров достигаются следующие цели:

− проверка корректности введенных данных и выполнение сложных ограничений целостности данных, которые трудно, если вообще возможно, поддерживать с помощью ограничений целостности, установленных для таблицы;

− выдача предупреждений, напоминающих о необходимости выполнения некоторых действий при обновлении таблицы, реализованном определенным образом;

− накопление аудиторской информации посредством фиксации сведений о внесенных изменениях и тех лицах, которые их выполнили.

При условии правильного использования триггеры могут стать очень мощным механизмом. Основное их преимущество заключается в том, что стандартные функции сохраняются внутри базы данных и согласованно активизируются при каждом ее обновлении. Это может существенно упростить приложения. Тем не менее, следует упомянуть и о присущих триггеру недостатках:

− сложность: при перемещении некоторых функций в базу данных усложняются задачи ее проектирования, реализации и администрирования;

− скрытая функциональность: перенос части функций в базу данных и сохранение их в виде одного или нескольких триггеров иногда приводит к сокрытию от пользователя некоторых функциональных возможностей. Хотя это в определенной степени упрощает его работу, но, к сожалению, может стать причиной незапланированных, потенциально нежелательных и вредных

побочных эффектов, поскольку в этом случае пользователь не в состоянии контролировать все процессы, происходящие в базе данных;

− влияние на производительность: перед выполнением каждой команды по изменению состояния базы данных СУБД должна проверить триггерное условие с целью выяснения необходимости запуска триггера для этой команды. Выполнение подобных вычислений сказывается на общей производительности СУБД, а в моменты пиковой нагрузки ее снижение может стать особенно заметным. Очевидно, что при возрастании количества триггеров увеличиваются и накладные расходы, связанные с такими операциями.

ХОД ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

1. Создать триггер, который при добавлении записи в таблицу Exam автоматически вычисляет общее число студентов, сдавших этот же экзамен в этой же группе. Данное значение фиксируется в другой таблице Exam_Group (Группа, Предмет, Кол-во студентов), если строка с такой группой и таким предметом существует, или создается новая запись с соответствующими значениями.

   Предварительно проверить, что добавляется только одна запись.

   Код данного триггера представлен ниже:

```
USE [ExamSession]
GO
/**** Object:  Trigger [dbo].[a24]    Script Date: 21.11.2023 16:05:20 ****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TRIGGER [dbo].[a24] ON [dbo].[Exam] AFTER INSERT
AS
BEGIN
IF @@ROWCOUNT = 1
```

```
BEGIN
DECLARE @group VARCHAR(20), @count_stud SMALLINT, @subject NCHAR(20), @is_exist SMALLINT, @id_stud INT
SELECT @group = NameGroup FROM inserted JOIN Student ON Student.Id_Student = inserted.Id_Student
SELECT @subject = Subject FROM inserted
SELECT @id_stud = Id_Student FROM inserted
SELECT @count_stud = COUNT(*) FROM Exam JOIN Student ON Student.Id_Student = @id_stud WHERE NameGroup = @group AND Exam.Subject = @subject
SELECT @is_exist = COUNT(*) FROM Exam_Group WHERE NameGroup = @group AND Subject = @subject
IF @is_exist>0
 BEGIN
  UPDATE Exam_Group SET StudentsCount = @count_stud WHERE NameGroup = @group AND Subject = @subject
 END
ELSE
 BEGIN
  INSERT INTO Exam_Group VALUES (@group, @subject, @count_stud)
 END
END
ELSE
BEGIN
PRINT('Вы добавили более одной строки')
ROLLBACK
END
END
```

Для выполнения данного задания была создана таблица Exam_Group. На рис. 1 представлена её структура.

Column Name	Data Type	Allow Nulls
NameGroup	varchar(20)	☑
Subject	nchar(20)	☑
StudentsCount	smallint	☑
		☐

Рис. 1. Структура таблицы Exam_Group

Исходные данные для выполнения данного задания до срабатывания триггера представлены на рис. 2.

NameGroup	Subject	StudentsCount
NULL	*NULL*	*NULL*

Рис. 2. Исходные данные из таблицы Exam_Group

При добавлении в таблицу Exam строки с группой, которой еще нет в таблице Exam_Group, создается новая запись. Таблица Exam_Group добавления строки и срабатывания триггера представлена на рис. 3.

NameGroup	Subject	StudentsCount
ИБД-12-02	Компьютерная...	9
NULL	*NULL*	*NULL*

Рис. 3. Таблица Exam_Group после срабатывания триггера

При добавлении в таблицу Exam строки с группой, которая уже есть в таблице Exam_Group, изменяется уже созданная запись. Таблица Exam_Group после действия, описанного раннее, представлена на рис. 4.

ИДБ-21-06	УД	4
ИДБ-20-08	УД	5
NULL	*NULL*	*NULL*

Рис. 4. Таблица Exam_Group после добавления оценки другому студенту из другой группы

При попытке создания нескольких записей в таблице Exam выводится сообщение, представленное на рис. 5.

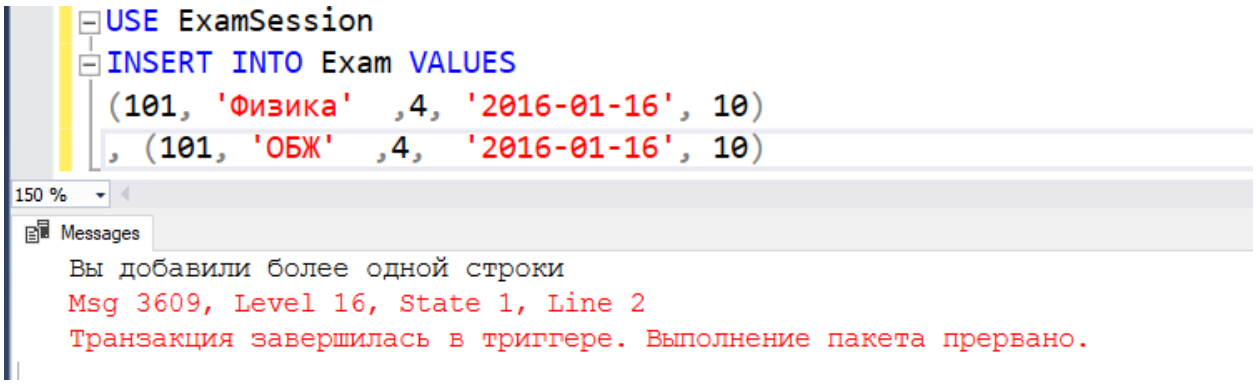

Рис. 5. Сообщение при попытке удаления нескольких строк в таблице Exam

2. Создать триггер, который при добавлении или изменении записи в таблицу Exam автоматически определяет худший результат того студента, который сдал данный экзамен. Словесное описание этого результата (отлично, с четверками, удовлетворительно, с двойками) фиксируется в другой таблице РезультатыСессии, если строка с таким студентом существует, или создается новая запись с соответствующими значениями.

Предварительно проверить, что добавляется (изменяется) только одна запись.

Код данного триггера представлен ниже:

```
USE [ExamSession]
GO
/**** Object:  Trigger [dbo].[v26]    Script Date: 21.11.2023 15:51:04 ****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER TRIGGER [dbo].[v26] ON [dbo].[Exam] INSTEAD OF UPDATE, INSERT
AS
BEGIN
```

```
IF @@ROWCOUNT = 1
BEGIN
DECLARE @result INT, @stud_id INT, @count SMALLINT
SELECT @stud_id = Id_Student FROM inserted
SELECT @result = MIN(Mark) FROM Exam WHERE Id_Student = @stud_id
INSERT Exam SELECT * FROM inserted

SELECT @count = COUNT(*) FROM SessionResult WHERE Id_Student = @stud_id
IF @count = 0
BEGIN
IF @result=2
INSERT INTO SessionResult VALUES(@stud_id, 'С двойками')
ELSE IF @result=3
INSERT INTO SessionResult VALUES(@stud_id, 'Удовлетварительно')
ELSE IF @result=4
INSERT INTO SessionResult VALUES(@stud_id, 'Хорошо')
ELSE IF @result=5
INSERT INTO SessionResult VALUES(@stud_id, 'Отлично')
END
ELSE
BEGIN
IF @result=2
UPDATE SessionResult SET Result = 'С двойками'
ELSE IF @result=3
UPDATE SessionResult SET Result = 'Удовлетварительно'
ELSE IF @result=4
UPDATE SessionResult SET Result = 'Хорошо'
```

```
ELSE IF @result=5
UPDATE SessionResult SET Result = 'Отлично'
END
END
ELSE
BEGIN
PRINT('Вы добавили более одной строки')
END
END
```

Для выполнения данного задания была создана таблица SessionResult. На рис. 6 представлена её структура.

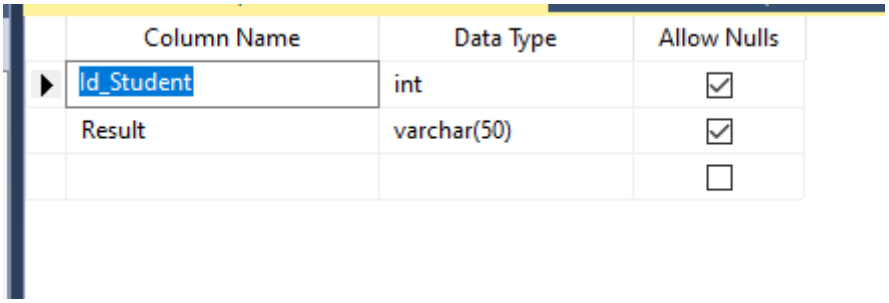

Рис. 6. Структура таблицы SessionResult

Исходные данные для выполнения данного задания до срабатывания триггера представлены на рис. 7.

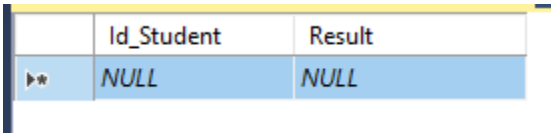

Рис. 7. Исходные данные из таблицы SessionResult

При добавлении в таблицу Exam строки со студентом, которого еще нет в таблице SessionResult, создается новая запись. Таблица SessionResult добавления строки и срабатывания триггера представлена на рис. 8.

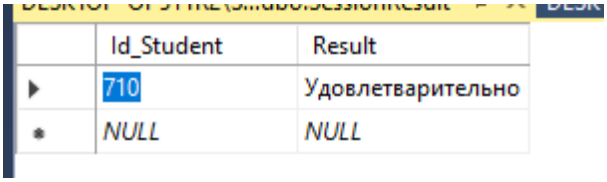

Рис. 8. Таблица SessionResultпосле срабатывания триггера

При добавлении в таблицу Exam строки со студентом, который уже есть в таблице SessionResult, изменяется уже созданная запись. Таблица SessionResult после действия, описанного раннее, представлена на рис. 9.

	Id_Student	Result
	710	Удовлетворительно
	101	С двойками
▶*	*NULL*	*NULL*

Рис. 9. Таблица SessionResult после удаления строки с группой ИДБ-22-05

При попытке изменения в таблице Exam сразу нескольких строк выводится сообщение, представленное на рис. 10.

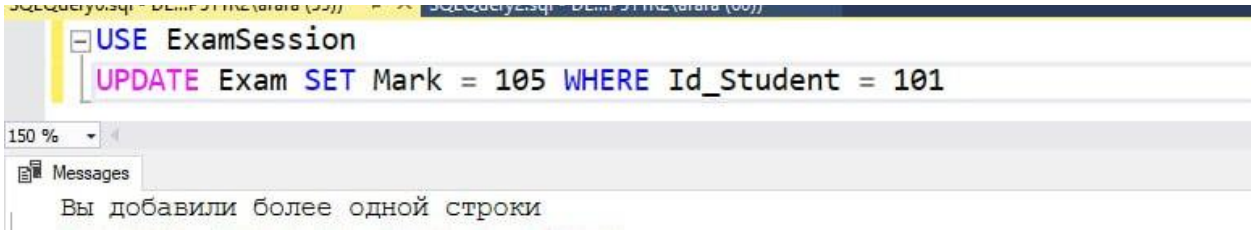

Рис. 10. Сообщение при попытке удаления нескольких строк в таблице Exam

ВЫВОДЫ

В ходе данной лабораторной работы ознакомились с понятием триггера, изучили способы создания триггеров на языке Transact-SQL, а также были приобретены практические навыки работы с триггерами в среде SQL Server Management Studio.

Приложение А

Id_Student	Subject	Mark	Exam_Date	Id_Lect
101	Архитектура Э...	4	2016-01-16	10
101	Операционны...	4	2016-01-12	13
101	Управление да...	5	2015-12-28	25
104	Архитектура Э...	4	2016-01-15	10
104	Управление да...	5	2016-01-19	35
110	Архитектура Э...	4	2016-01-20	12
110	Мат.статистик...	4	2016-01-10	35
110	Сети	4	2016-01-15	13
120	Архитектура Э...	4	2016-01-16	10
120	Операционны...	5	2016-01-12	13
120	Управление да...	4	2015-12-28	25
121	Архитектура Э...	4	2016-01-16	10
121	Операционны...	4	2016-01-12	13
121	Управление да...	4	2015-12-28	25
156	Архитектура Э...	3	2015-12-29	10
156	Операционны...	2	2016-01-16	13
156	Управление да...	3	2016-01-12	25
209	Архитектура Э...	5	2016-01-20	12
209	Сети	4	2016-01-15	13
218	Архитектура Э...	4	2016-01-15	10
218	Компьютерная...	4	2016-01-20	12
218	Физика	5	2015-12-29	50
218	Философия ...	4	2016-01-10	30

1 для 108

Рис. А1. Исходные данные таблицы “Exam”