

# Лабораторная работа 7.

## Триггеры

Цель работы: научиться создавать триггеры в среде SQL Server Management Studio.

### **Теоретические сведения**

При работе БД должна обеспечиваться целостность данных. При удалении записей из первичных таблиц автоматически должны удаляться связанные с ними записи из вторичных таблиц. В случае несоблюдения этого принципа со временем в БД накопится большое количество записей во вторичных таблицах, связанных с несуществующими записями в первичных таблицах, что приведёт к сбоям в работе БД и её засорению. Для обеспечения целостности данных в SQL Server используют триггеры.

Триггер – это сочетание хранимой в базе данных процедуры и события, которое заставляет ее выполняться. Такими событиями могут быть: ввод новой строки таблицы, изменение значений одного или нескольких ее столбцов и (или) удаление строки таблицы. При любом из этих событий автоматически запускаются один или несколько заранее созданных триггеров, которые производят проверку запрограммированных в них условий, и если они не выполняются, отменяют ввод, изменение или удаление, посылая об этом заранее подготовленное сообщение пользователю.

Триггеры похожи на процедуры и функции тем, что также являются именованными блоками и имеют раздел объявлений, выполняемый раздел и раздел обработки исключительных ситуаций. Подобно процедурам и функциям, триггеры хранятся как автономные объекты в базе данных.

Триггеры позволяют:

- реализовывать сложные ограничения целостности данных, которые невозможно реализовать через ограничения, устанавливаемые при создании таблицы;
- контролировать информацию, хранимую в таблице, посредством регистрации вносимых изменений и пользователей, производящих эти изменения;
- автоматически оповещать другие программы о том, что необходимо делать в случае изменения информации, содержащейся в таблице;
- публиковать информацию о различных событиях.

Триггеры также делятся на три основных типа.

- *Триггеры DML* активизируются предложениями ввода, обновления и удаления информации (INSERT, UPDATE, DELETE) до или после выполнения предложения, на уровне строки или таблицы.
- *Триггеры замещения (instead of)* можно создавать только для представлений (либо объектных, либо реляционных). В отличие от триггеров DML, которые выполняются в дополнение к предложениям DML, триггеры замещения выполняются вместо предложений DML, вызывающих их срабатывание. Триггеры замещения должны быть строковыми триггерами.
- *Системные триггеры* активизируется не на предложение DML, выполняемое над таблицей, а на системное событие, например, на запуск или останов базы данных. Системные триггеры срабатывают и на предложения DDL, такие как создание таблицы.

Создадим триггеры для таблицы «Artworks». Триггеры создаются отдельно для каждой таблицы и располагаются в обозревателе объектов в папке «Триггеры» (рис. 1).

Для начала создадим триггер, выводящий сообщение «Запись добавлена» при добавлении записи в таблицу «Artworks». Создадим новый триггер, щёлкнув правой кнопкой мыши по папке «Триггеры» в таблице «Artworks» и выбрав в появившемся меню пункт «Создать триггер». Появится окно с новым триггером (рис. 2).

Структура триггера:

- 1) область определения имени триггера (Trigger\_Name);
- 2) область, показывающая для какой таблицы создаётся триггер (Table\_Name);
- 3) область, показывающая, когда выполнять триггер (INSERT – при создании записи в таблице, DELETE – при удалении и UPDATE – при изменении);
- 4) тело триггера, содержащее команды языка T-SQL.

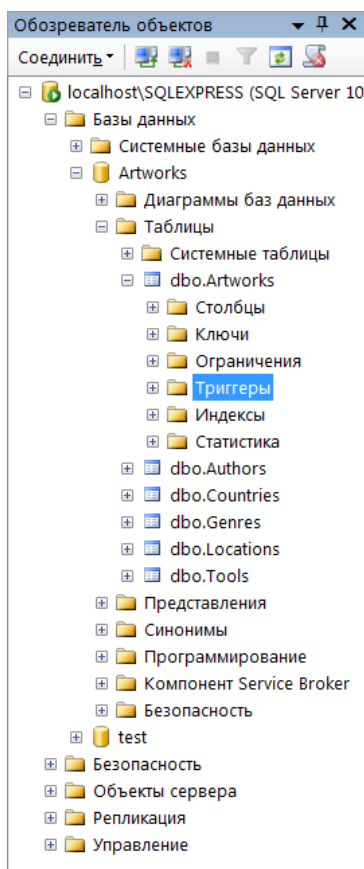


Рис. 1.

```
SQLQuery1.sql - localhost\...\ovm (52)
-- =====
-- Template generated from Template Explorer using:
-- Create Trigger (New Menu).SQL
--
-- Use the Specify Values for Template Parameters
-- command (Ctrl-Shift-M) to fill in the parameter
-- values below.
--
-- See additional Create Trigger templates for more
-- examples of different Trigger statements.
--
-- This block of comments will not be included in
-- the definition of the function.
-- =====

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =====
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =====
CREATE TRIGGER <Schema_Name, sysname, Schema_Name>.<Trigger_Name, sysname, Trigger_Name>
ON <Schema_Name, sysname, Schema_Name>.<Table_Name, sysname, Table_Name>
AFTER <Data_Modification_Statements, , INSERT,DELETE,UPDATE>
AS
BEGIN
-- SET NOCOUNT ON added to prevent extra result sets from
-- interfering with SELECT statements.
SET NOCOUNT ON;

-- Insert statements for trigger here

END
GO
```

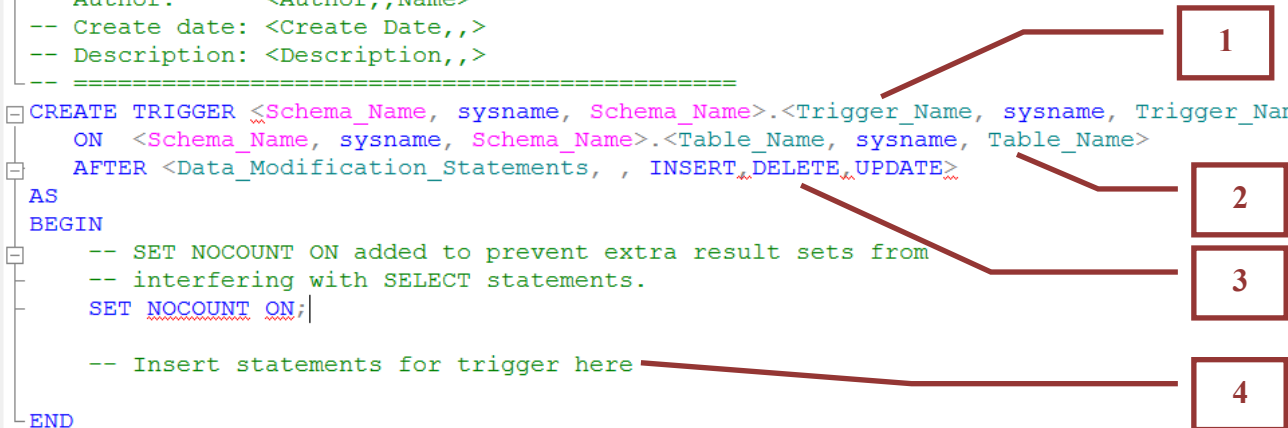


Рис. 2.

В окне нового триггера наберем следующий код (рис. 3).

Созданный триггер «Insert\_trigger» выполняется после добавления записи (AFTER INSERT) в таблицу «Artworks» (ON dbo.Artworks). После добавления записи триггер выведет на экран сообщение «Запись добавлена!» (PRINT 'Запись добавлена!'). Выполним набранный код, нажав кнопку «Выполнить» на панели инструментов. В нижней части окна с кодом появится сообщение «Выполнение команд успешно завершено».

```
SQLQuery2.sql - localhost\...\o... (53)* SQLQuery1.sql - localhost\...\ovm (52)
-- =====
-- Template generated from Template Explorer using:
-- Create Trigger (New Menu).SQL
--
-- Use the Specify Values for Template Parameters
-- command (Ctrl-Shift-M) to fill in the parameter
-- values below.
--
-- See additional Create Trigger templates for more
-- examples of different Trigger statements.
--
-- This block of comments will not be included in
-- the definition of the function.
-- =====

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

-- =====
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =====

CREATE TRIGGER Insert_trigger
ON dbo.Artworks
AFTER INSERT
AS
BEGIN
-- SET NOCOUNT ON added to prevent extra result sets from
-- interfering with SELECT statements.
SET NOCOUNT ON;

PRINT ('Запись добавлена!');

END
GO
```

Рис. 3.

Проверим, как работает новый триггер. Создадим новый пустой запрос и в нём выполним добавление данных в таблицу «Artworks» (рис.4):

```
SQLQuery2.sql - localhost\...\o... (53)* SQLQuery1.sql - localhost\...\ovm (52)
insert into Artworks (ArtworkTitle,GenreCode,ToolsCode,AuthorCode,CreationDate,CountryCode,PlacementCode,Price)
values ('Утро в сосновом лесу', 13, 3, 2, '1889-01-01', 1, 4, 1000000)
```

Сообщения  
Запись добавлена!  
(строк обработано: 1)

Рис. 4.

Выполним набранную команду, нажав кнопку «Выполнить» на панели инструментов. В таблицу будет добавлена новая запись, и триггер выведет сообщение «Запись добавлена!» (рис. 4).

Аналогичным образом создаются триггеры, выводящие сообщения при изменении данных в таблице (рис. 5) и при их удалении (рис. 6).

```
SQLQuery9.sql - localhost\... \o... (54)*  SQLQuery8.sql - localhost\... \ovm (52)  SQLQuery2.sql - localhost\... \o... (53)*
-- =====
-- Template generated from Template Explorer using:
-- Create Trigger (New Menu).SQL
--
-- Use the Specify Values for Template Parameters
-- command (Ctrl-Shift-M) to fill in the parameter
-- values below.
--
-- See additional Create Trigger templates for more
-- examples of different Trigger statements.
--
-- This block of comments will not be included in
-- the definition of the function.
-- =====

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

-- =====
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =====

CREATE TRIGGER Update_trigger
ON dbo.Artworks
AFTER UPDATE
AS
BEGIN
-- SET NOCOUNT ON added to prevent extra result sets from
-- interfering with SELECT statements.
SET NOCOUNT ON;

PRINT('Запись изменена!');

END
GO
```

Рис. 5.

Рассмотрим пример применения триггеров для обеспечения целостности данных. Создадим триггер «Delete\_author» (рис. 7), который при удалении записи из таблицы «Authors» сначала удаляет все связанные с ней записи из таблицы «Artworks», а затем удаляет саму запись из таблицы «Authors».

```
SQLQuery10.sql - localhost\..... (55)* SQLQuery9.sql - localhost\...o... (54)*
-- =====
-- Template generated from Template Explorer using:
-- Create Trigger (New Menu).SQL
--
-- Use the Specify Values for Template Parameters
-- command (Ctrl-Shift-M) to fill in the parameter
-- values below.
--
-- See additional Create Trigger templates for more
-- examples of different Trigger statements.
--
-- This block of comments will not be included in
-- the definition of the function.
-- =====

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

-- =====
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =====

CREATE TRIGGER Delete_trigger
ON dbo.Artworks
AFTER DELETE
AS
BEGIN
-- SET NOCOUNT ON added to prevent extra result sets from
-- interfering with SELECT statements.
SET NOCOUNT ON;

PRINT ('Запись удалена!');

END
GO
```

Рис. 6.

```
SQLQuery1.sql - localhost\...o... (53)* OVM-PC\SQLXPRES...s - dbo.Artworks
+
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

-- =====
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =====

CREATE TRIGGER Delete_author
ON dbo.Authors
INSTEAD OF DELETE
AS
BEGIN
-- SET NOCOUNT ON added to prevent extra result sets from
-- interfering with SELECT statements.
SET NOCOUNT ON;

-- Insert statements for trigger here
DELETE dbo.Artworks
FROM Deleted
WHERE Deleted.AuthorCode = Artworks.AuthorCode
DELETE dbo.Authors
FROM Deleted
WHERE Deleted.AuthorCode = Authors.AuthorCode

END
GO
```

Рис. 7.

При срабатывании триггера вместо удаления записи создаётся временная константа Deleted, содержащая имя таблицы, из которой должно было быть произведено удаление.

После срабатывания триггера из таблицы «Artworks» удаляется запись, у которой значение поля «AuthorCode» равно значению такого же поля у удаляемой записи из таблицы «Authors». Затем удаляется запись из таблицы «Authors», которую удаляли до срабатывания триггера.

Выполним набранный код, нажав кнопку «Выполнить» на панели инструментов. В нижней части окна с кодом появится сообщение «Выполнение команд успешно завершено.».

Проверим, как работает триггер «Author\_delete» (рис. 8). При срабатывании триггера сначала из таблицы «Artworks» удалятся все связанные с удаляемой записью записи, а затем удаляется сама удаляемая запись из таблицы «Artworks», при этом сохраняется целостность данных.

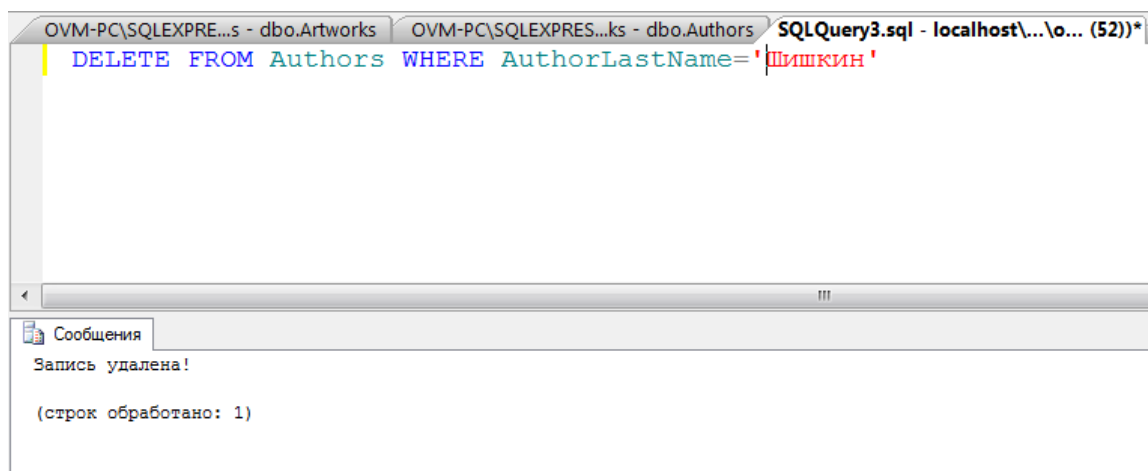


Рис. 8.

### ***Задание***

Создать триггеры в соответствии с заданием для своего варианта.