

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Систем автоматизированного проектирования

ОТЧЕТ
по лабораторной работе №11
по дисциплине «Базы данных»
Тема: Создание триггеров

Студенты гр. 2308

Рымарь М.И.

Мелихов М.А.

Придчин В.Е.

Преподаватель

Горяинов С.В.

Санкт-Петербург

2024

Цель работы

Научиться создавать триггеры. В этой лабораторной работе используется база данных AdventureWorks.

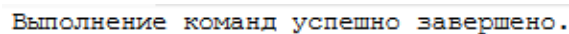
Выполнение работы

Упражнение 1 – создание новой таблицы.

Создать новую таблицу, используя следующие команды:

```
USE [AdventureWorks]
GO
CREATE TABLE [HumanResources].[JobCandidateHistory] (
    [JobCandidateID] [int] NOT NULL UNIQUE,
    [Resume] [xml] NULL,
    [Rating] [int] NOT NULL CONSTRAINT
    [DF_JobCandidateHistory_Rating] Default (5),
    [RejectedDate] [datetime] NOT NULL,
    [ContactID] [int] NULL,
    CONSTRAINT [FK_JobCandidateHistory_Contact_ContactID]
    FOREIGN KEY (ContactID) REFERENCES [Person].[Contact]
    (ContactID),
    CONSTRAINT [CK_JobCandidateHistory_Rating]
    CHECK ([Rating] >= 0 AND [Rating] <= 10)
) ON [PRIMARY]
```

Результат выполнения показан на рисунке 1.



Выполнение команд успешно завершено.

Время выполнения: 2024-11-04T20:18:52.5044103+03:00

Рисунок 1 – Создание новой таблицы

Упражнение 2 – создание триггера для таблицы JobCandidate схемы HumanResources.

Выполнение следующих команд для создания триггера:

```

USE [AdventureWorks]
GO
CREATE TRIGGER [HumanResources].[dJobCandidate]
ON [HumanResources].[JobCandidate] AFTER DELETE
AS
BEGIN
    INSERT INTO [HumanResources].[JobCandidateHistory] (
        JobCandidateID, Resume, RejectedDate, ContactID
    )
    SELECT deleted.JobCandidateID, deleted.Resume,
GETDATE(), NULL
    FROM deleted;
END;
GO

```

Результат представлен на рисунке 2.

Выполнение команд успешно завершено.

Время выполнения: 2024-11-11T16:25:59.8604517+03:00

Рисунок 2 – Результат выполнения команд

Упражнение 3 – проверка работы триггера.

Выполнение следующей команды:

```

USE AdventureWorks
GO
DELETE FROM HumanResources.JobCandidate
WHERE JobCandidateID =
(SELECT MIN (JobCandidateID)
FROM HumanResources.JobCandidate)

```

Результат выполнения показан на рисунке 3.

(затронута одна строка)

(затронута одна строка)

Время выполнения: 2024-11-11T17:22:34.2731436+03:00

Рисунок 3 – Результат выполнения команды

В результате срабатывания триггера на удаление данные о кандидате должны быть скопированы в таблицу JobCandidateHistory. Выполнение следующего запроса:

```
SELECT * FROM [HumanResources].[JobCandidateHistory]
```

Результат выполнения показан на рисунке 4.

	JobCandidateID	Resume	Rating	RejectedDate	ContactID
1	1	<ns:Resume xmlns:ns="http://schemas.microsoft.co...	5	2024-11-11 17:22:34.257	NULL

(затронута одна строка)

Время выполнения: 2024-11-11T17:26:16.4188310+03:00

Рисунок 4 – Результат выполнения запроса

Удаление данных из таблицы JobCandidateHistory с помощью команды:

```
TRUNCATE TABLE [HumanResources].[JobCandidateHistory]
```

Результат выполнения команды представлен на рисунке 5.

Выполнение команд успешно завершено.

Время выполнения: 2024-11-11T17:33:14.7948597+03:00

Рисунок 5 – Результат выполнения команды

Упражнение 4 – создание триггера на обновление и вставку.

Создание триггера OrderDetailNotDiscontinued на таблицу Sales.SalesOrderDetail. Этот триггер отвергает попытки ввода заказов на товары, приём которых на склад прекращён. Информация о прекращении поставок товара находится в таблице Production.Product. Если поставки товара прекращены, то значение поля DiscontinuedDate будет иметь значение, отличное от NULL. При попытке заказать такой товар должен выдавать сообщение с

помощью команды RAISERROR и откатить транзакцию. Запрос представлен ниже. Результат выполнения запроса – на рисунке 6.

```
USE AdventureWorks;
GO
CREATE TRIGGER OrderDetailNotDiscontinued
ON Sales.SalesOrderDetail AFTER INSERT
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM inserted i
        JOIN Production.Product p ON i.ProductID = p.ProductID
        WHERE p.DiscontinuedDate IS NOT NULL
    )
    BEGIN
        RAISERROR('Невозможно заказать товар, поставки которого
прекращены.', 16, 1);

        ROLLBACK TRANSACTION;
    END
END;
GO
```

Выполнение команд успешно завершено.

Время выполнения: 2024-11-11T17:36:37.1262169+03:00

Рисунок 6 – Результат выполнения команд

Выполнение проверки триггера. В базу данных необходимо ввести хотя бы одну строку, при обнаружении которой во время выполнения кода триггера должна активироваться ошибка. Для этого в таблицу Production.Product необходимо ввести данные хотя бы об одном товаре, поставка которого

прекращена. Проверим, есть ли подходящие данные в таблице Product с помощью запроса:

```
SELECT ProductID, Name
FROM Production.Product
WHERE DiscontinuedDate IS NOT NULL
```

Результат выполнения запроса представлен на рисунке 7.

(затронуто строк: 0)

Время выполнения: 2024-11-11T17:44:24.3563040+03:00

Рисунок 7 – Результат выполнения запроса

По результату выполнения запроса можно понять, что подходящих записей нет, поэтому добавим запись искусственно с помощью следующих команд:

```
UPDATE Production.Product
SET DiscontinuedDate = GETDATE()
WHERE ProductID = 680
```

Результат выполнения представлен на рисунке 8.

(затронута одна строка)

Время выполнения: 2024-11-11T23:18:23.7841120+03:00

Рисунок 8 – Результат выполнения

Проверка триггера с помощью следующей команды:

```
INSERT Sales.SalesOrderDetail
(SalesOrderID, OrderQty, ProductID, SpecialOfferID,
UnitPrice, UnitPriceDiscount)
VALUES (43660, 5, 680, 1, 1431, 0)
```

Результат выполнения представлен на рисунке 9.

```
сообщение: 50000, уровень: 16, состояние: 1, процедура: OrderDetailNotDiscontinued  
Невозможно заказать товар, поставки которого  
прекращены.  
Сообщение 3609, уровень 16, состояние 1, строка 1  
Транзакция завершилась в триггере. Выполнение пакета прервано.  
Время выполнения: 2024-11-11T23:20:41.3531316+03:00
```

Рисунок 9 – Результат выполнения команды

В результате срабатывает триггер и попытка ввода отвергается с ошибкой «Невозможно заказать товар, поставки которого прекращены». Триггер сработал корректно.

Выводы

В ходе лабораторной работы были изучены принципы создания триггеров в SQL Server, позволяющих автоматически выполнять определенные действия при возникновении событий в базе данных. Практические упражнения продемонстрировали, как создавать триггеры для различных типов событий, таких как вставка, удаление и обновление данных в таблицах.

В первом упражнении была создана новая таблица JobCandidateHistory для хранения информации о кандидатах, которые были отклонены. Это действие было необходимо для последующей демонстрации работы триггера.

Второе упражнение продемонстрировало создание триггера dJobCandidate на таблицу JobCandidate. Этот триггер срабатывает после удаления записи из таблицы JobCandidate и автоматически копирует информацию о кандидате в таблицу JobCandidateHistory.

В третьем упражнении была проверена работоспособность триггера dJobCandidate. Была удалена запись из таблицы JobCandidate, после чего данные о кандидате успешно скопировались в таблицу JobCandidateHistory, что подтверждает корректную работу триггера.

В четвертом упражнении был создан триггер OrderDetailNotDiscontinued на таблицу Sales.SalesOrderDetail. Этот триггер предотвращает попытку ввода

заказа на товары, поставки которых прекращены. При попытке добавить в заказ такой товар триггер выдает сообщение об ошибке и откатывает транзакцию.

В ходе лабораторной работы было сделано интересное наблюдение:

- Триггеры предоставляют мощный механизм для автоматизации бизнес-логики в базе данных. Они позволяют гарантировать целостность данных, контролировать действия пользователей и выполнять необходимые действия без необходимости ручного вмешательства.

Полученные знания и навыки в рамках данной лабораторной работы позволяют эффективно использовать триггеры для решения различных задач, связанных с управлением данными в SQL Server, обеспечивая целостность и консистентность базы данных, а также реализацию сложных бизнес-правил.