

Лабораторная работа 8:

Триггеры

Создание триггера DML FOR

В этом задании вы создадите триггер для вставки данных. При выполнении финансовой операции с помощью инструкции INSERT в таблице **TransactionDetails.Transactions** надо изменить остаток на счете клиента. Остаток на счете требуется изменить после (AFTER) того, как строка вставлена в таблицу **TransactionDetails.Transactions**. Если при вставке возникнут проблемы, и инструкция INSERT не будет выполнена, то остаток на счете клиента не изменяется.

1. Запустите **Query Editor** (Редактор запросов) и в окне запроса введите следующий код T-SQL:

```
USE ApressFinancial
GO
CREATE TRIGGER TransactionDetails.trgInsTransactions
ON TransactionDetails.Transactions
AFTER INSERT
AS
UPDATE CustomerDetails.Customers
    SET CleareBalance = CleareBalance +
        (SELECT CASE WHEN CreditType = 0
            THEN i.Amount * -1
            ELSE i.Amount
        END
        FROM INSERTED AS i
        JOIN TransactionDetails.TransactionTypes AS tt
            ON tt.TransactionTypeId = i.TransactionType
        WHERE AffectCashBalance = 1 )
FROM CustomerDetails.Customers AS c
JOIN INSERTED AS i
    ON i.CustomerId = c.CustomerId;
```

2. Выполните код, чтобы создать триггер в БД.

Теперь протестируем триггер:

```
SELECT CleareBalance FROM CustomerDetails.Customers WHERE CustomerId=1;

INSERT INTO TransactionDetails.Transactions
    (CustomerId, TransactionType, Amount, RelatedProductId, DateEntered)
VALUES (1, 2, 200, 1, GETDATE())
```

```
SELECT CleareBalance FROM CustomerDetails.Customers WHERE CustomerId=1;
```

Результаты должны отображать ожидаемое уменьшение баланса на 200\$.

Смоделируем ненадежную транзакцию:

```
SELECT CleareBalance FROM CustomerDetails.Customers WHERE CustomerId=1;

INSERT INTO TransactionDetails.Transactions
    (CustomerId, TransactionType, Amount, RelatedProductId, DateEntered)
VALUES (1, 3, 200, 1, GETDATE())
```

```
SELECT CleareBalance FROM CustomerDetails.Customers WHERE CustomerId=1;
```

Появится сообщение об ошибке, т.к. в триггере подзапрос может вернуть значение NULL.

3. Изменим триггер с помощью инструкции ALTER TRIGGER:

```
ALTER TRIGGER TransactionDetails.trgInsTransactions
ON TransactionDetails.Transactions
AFTER INSERT
AS
UPDATE CustomerDetails.Customers
    SET CleareBalance = CleareBalance +
        ISNULL((SELECT CASE WHEN CreditType = 0
            THEN i.Amount * -1
```

```

ELSE i.Amount
END
FROM INSERTED AS i
JOIN TransactionDetails.TransactionTypes AS tt
    ON tt.TransactionTypesId = i.TransactionType
WHERE AffectCashBalance = 1 ), 0)
FROM CustomerDetails.Customers AS c
JOIN INSERTED AS i
    ON i.CustomerId = c.CustomerId;

```

Выполните код, чтобы изменить триггер.

Теперь еще раз выполните код, который привел к ошибке и отмене вставки:

```

SELECT ClearBalance FROM CustomerDetails.Customers WHERE CustomerId=1;

INSERT INTO TransactionDetails.Transactions
    (CustomerId,TransactionType,Amount,RelatedProductId,DateEntered)
VALUES (1, 3, 200, 1, GETDATE())

```

```

SELECT ClearBalance FROM CustomerDetails.Customers WHERE CustomerId=1;

```

Теперь код выполнится без ошибки. Изменения остатка на счете не будет.

	ClearBalance
1	387.9521

Query executed successfully

Создание триггера DDL

1. Создадим триггер DDL, который будет выполняться при создании хранимой процедуры, ее изменении или удалении. При выполнении любого из этих действий триггер проверит время дня, и если это рабочие часы, то действие будет отменено (будет выполнен откат), а также будет выдана ошибка и текст хранимой процедуры.

В окне запроса введите следующий код T-SQL:

```

CREATE TRIGGER trgSprocs
ON DATABASE
FOR CREATE_PROCEDURE, ALTER_PROCEDURE, DROP_PROCEDURE
AS
IF DATEPART(hh, GETDATE()) > 9 AND DATEPART(hh, GETDATE()) < 17
BEGIN
    DECLARE @Message nvarchar(max)
    SELECT @Message =
        'Completing work during core hours. Trying to release -'
        + EVENTDATA().value
        (' (/EVENT_INSTANCE/TSQLCommand/CommandText) [1]', 'nvarchar(max)')
    RAISERROR (@Message, 16, 1)
    ROLLBACK
    EXEC msdb.dbo.sp_send_dbmail
        @profile_name = 'SQL Server Database Mail Profile',
        @recipients = 'exam@limtu.spb.ru',
        @body = 'A stored procedure change',
        @subject = 'A stored procedure change has been initiated
                    and rolled back during core hours'
END;

```

Протестируйте триггер: попробуйте создать процедуру

```

CREATE PROC Test1
AS
SELECT 'Hello all';

```

Результат будет зависеть от времени тестирования.

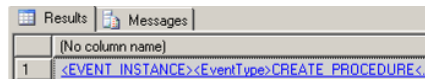
2. Создадим еще один триггер DDL, который будет выполняться по любому действию, осуществляемому в БД. В окне запроса введите следующий код T-SQL:

```
CREATE TRIGGER trgDBDump
ON DATABASE
FOR DDL_DATABASE_LEVEL_EVENTS
AS
    SELECT EVENTDATA();
```

Протестируйте триггер: попробуйте создать процедуру

```
CREATE PROC Test2
AS
    SELECT 'Hello all';
```

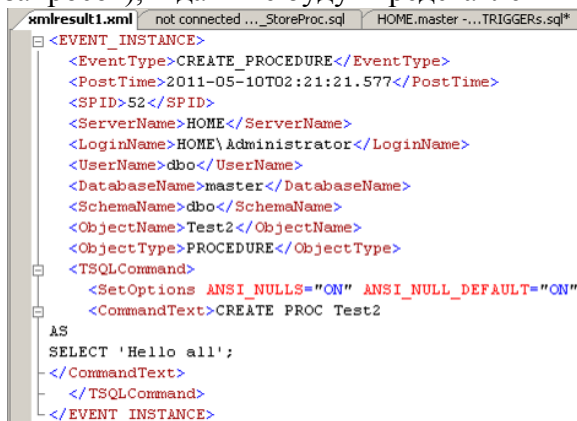
Код должен вернуть следующий результат:



Results		Messages
[No column name]		
1	<EVENT_INSTANCE><EventType>CREATE_PROCEDURE<...>	

Если щелкнуть ЛКМ по этой строке, то через несколько секунд откроется новая панель

Query Editor (Редактор запросов), и данные будут представлены в формате документа XML:



```
xmlresult1.xml  not connected ..._StoreProc.sql  HOME.master -...TRIGGERS.sql*
<EVENT_INSTANCE>
  <EventType>CREATE_PROCEDURE</EventType>
  <PostTime>2011-05-10T02:21:21.577</PostTime>
  <SPID>52</SPID>
  <ServerName>HOME</ServerName>
  <LoginName>HOME\Administrator</LoginName>
  <UserName>dbo</UserName>
  <DatabaseName>master</DatabaseName>
  <SchemaName>dbo</SchemaName>
  <ObjectName>Test2</ObjectName>
  <ObjectType>PROCEDURE</ObjectType>
  <TSQLCommand>
    <SetOptions ANSI_NULLS="ON" ANSI_NULL_DEFAULT="ON"
    <CommandText>CREATE PROC Test2
  AS
    SELECT 'Hello all';
  </CommandText>
  </TSQLCommand>
</EVENT_INSTANCE>
```

С помощью этого триггера можно просмотреть данные XML для любого события.

Удаление триггера DDL

Удалите триггеры DDL, созданные в предыдущем задании, с помощью инструкции DROP:

```
DROP TRIGGER trgSprocs ON DATABASE;
DROP TRIGGER trgDBDump ON DATABASE;
```

Примечание. Обратите внимание, что выполнение первой инструкции DROP вызвало срабатывание триггера *trgDBDump*!