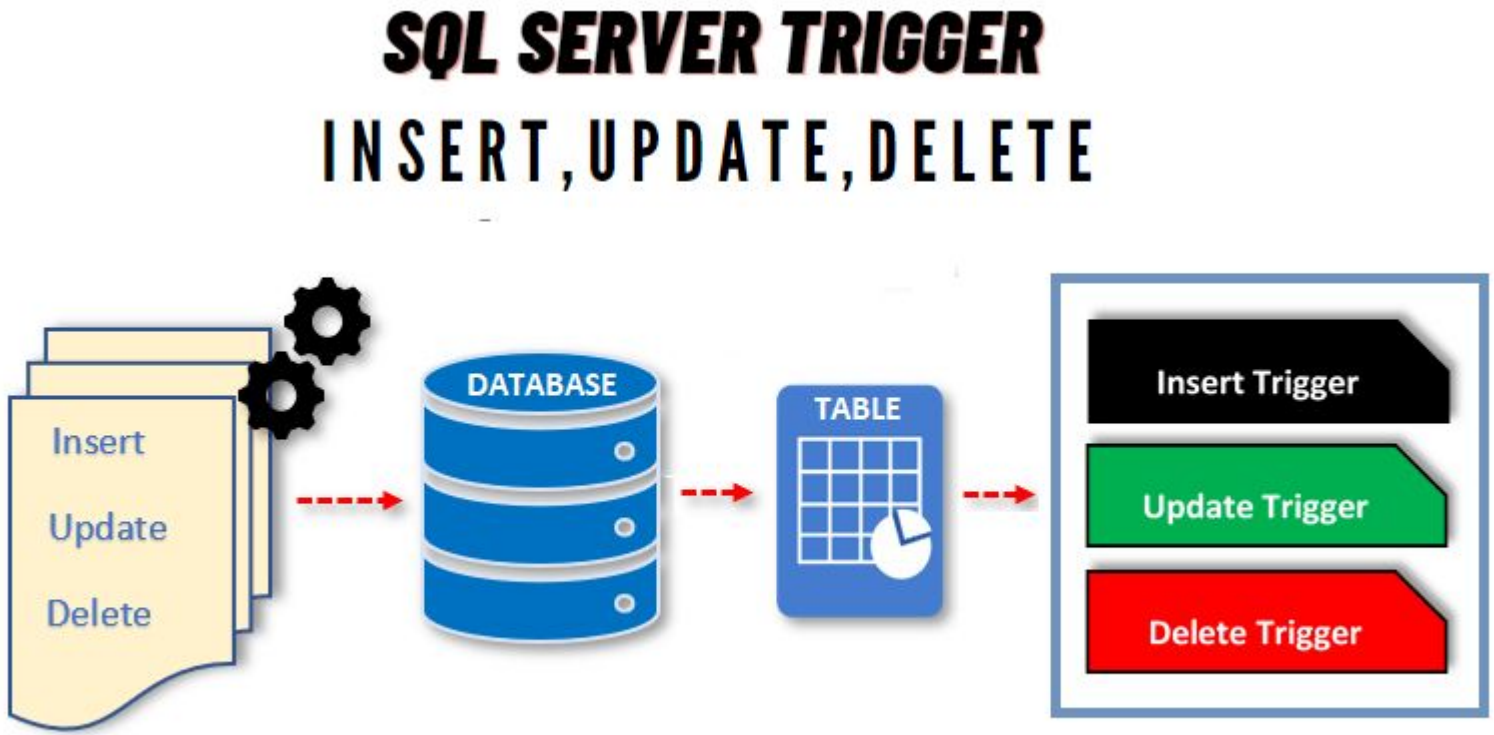
Типы триггеров

- *Триггеры DML* (язык манипулирования данными)
- *Триггеры DDL* (язык определения данных)
- *Триггеры входа в систему*
- *Триггер DDL среды CLR*



Типы триггеров

- *Триггеры DML* (язык манипулирования данными)
- *Триггеры DDL* (язык определения данных)
- *Триггеры входа в систему*
- *Триггер DDL среды CLR*

Триггеры DML позволяют нам выполнять код в ответ на изменение данных. Другими словами, они позволяют нам запускать дополнительный код в ответ на выполнение операторов вставки, обновления или удаления.

Триггеры DDL дают нам возможность выполнять код в ответ на изменения в структуре базы данных, такие как удаление или создание таблицы; или событие сервера, например, когда кто-то создает базу данных или логин для входа в систему. Триггеры DDL можно разделить на два разных типа в зависимости от их области действия.

- Триггеры DDL с областью действия базы данных
- Триггеры DDL на уровне сервера

Триггеры входа в систему — это частный случай триггеров DDL с областью действия сервера, которые срабатывают в ответ на событие LOGON, возникающее при установке сеанса пользователя.

Триггер CLR выполняет один или несколько методов, написанных в управляемом коде, которые являются членами сборки, созданной в платформа .NET Framework и переданной в SQL Server.

Триггеры DDL

Используйте триггеры DDL, если хотите сделать следующее.

- Предотвращать внесение определенных изменений в схему базы данных.
- Настроить выполнение в базе данных некоторых действий в ответ на изменения в схеме базы данных.
- Записывать изменения или события схемы базы данных.

Ограничения

- Триггеры DDL срабатывают только после выполнения соответствующих инструкций DDL. Триггеры DDL нельзя использовать в качестве триггеров INSTEAD OF.
- Триггеры DDL не срабатывают в ответ на события, влияющие на локальные или глобальные временные таблицы и хранимые процедуры.
- Триггеры DDL не создают специальные таблицы **inserted** и **deleted** .
- Сведения о событии, приведшем к срабатыванию триггера DDL, и последующих изменениях, выполненных триггером, можно получить при помощи функции ***EVENTDATA.***

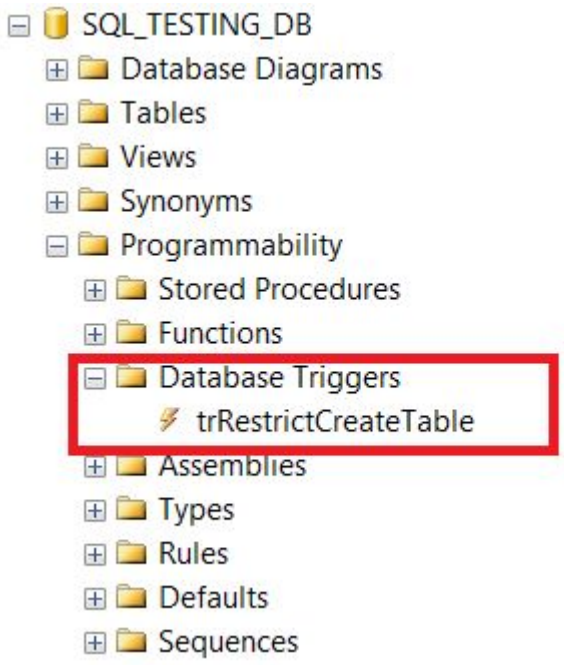
Триггеры DDL

Используйте триггеры DDL, если хотите сделать следующее.

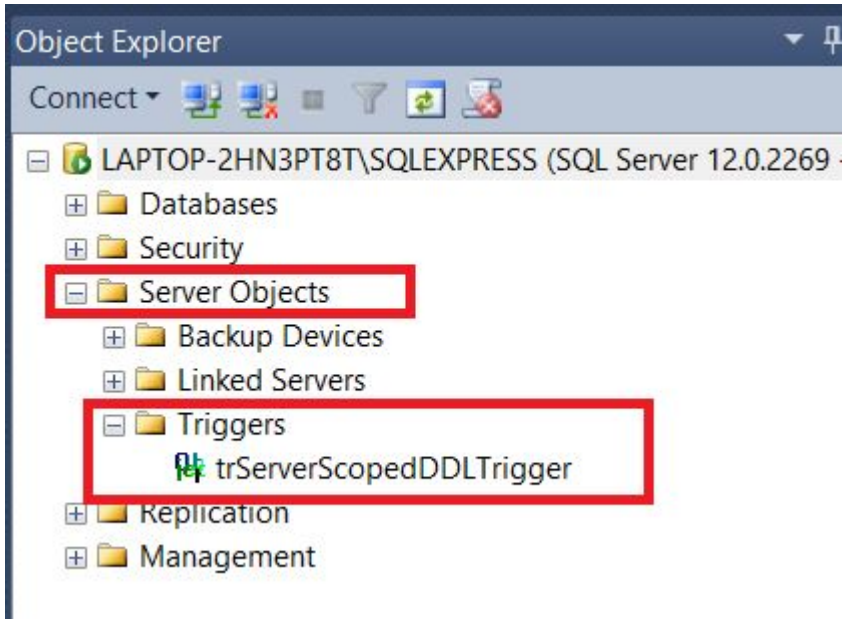
- Предотвращать внесение определенных изменений в схему базы данных.
- Настроить выполнение в базе данных некоторых действий в ответ на изменения в схеме базы данных.
- Записывать изменения или события схемы базы данных.

```
CREATE TRIGGER alert_table
ON database
FOR CREATE_TABLE, DROP_TABLE, ALTER_TABLE
AS
BEGIN
        IF IS_MEMBER ('db_owner') = 0
        BEGIN
            PRINT 'Please contact your Database Administrator'
            INSERT INTO master.dbo.event_data(in_)
            SELECT EVENTDATA();
            ROLLBACK TRANSACTION;
        END
END
GO
```

Триггеры DDL



Триггеры DDL, доступные в области базы данных



Триггеры DDL сервера

Триггеры DDL


```
-- Trigger on a CREATE, ALTER, DROP, GRANT, DENY,
-- REVOKE or UPDATE statement (DDL Trigger)

CREATE [ OR ALTER ] TRIGGER trigger_name
ON { ALL SERVER | DATABASE }
[ WITH <ddl_trigger_option> [ ,...n ] ]
{ FOR | AFTER } { event_type | event_group } [ ,...n ]
AS { sql_statement  [ ; ] [ ,...n ] | EXTERNAL NAME < method specifier >  [ ; ] }


 -- Trigger on a LOGON event (Logon Trigger)

 CREATE [ OR ALTER ] TRIGGER trigger_name
 ON ALL SERVER
 [ WITH <logon_trigger_option> [ ,...n ] ]
 { FOR| AFTER } LOGON
 AS { sql_statement  [ ; ] [ ,...n ] | EXTERNAL NAME < method specifier >  [ ; ] }

 <logon_trigger_option> ::=
     [ ENCRYPTION ]
     [ EXECUTE AS Clause ]
```

Триггеры DDL

```
CREATE [ OR ALTER ] TRIGGER trigger_name
ON { ALL SERVER | DATABASE }
[ WITH <ddl_trigger_option> [ ,...n ] ]
{ FOR | AFTER } { event_type | event_group } [ ,...n ]
AS { sql_statement  [ ; ] [ ,...n ] | EXTERNAL NAME < method specifier >  [ ; ] }

<ddl_trigger_option> ::=
    [ ENCRYPTION ]
    [ EXECUTE AS Clause ]
```

Argument	Description
ALL SERVER | DATABASE	Это установит область действия триггера. ALL SERVER : областью действия является весь экземпляр SQL Server. DATABASE: задает область действия триггера для конкретной базы данных, в которой создается триггер.
ENCRYPTION	Шифрует определение триггера в метаданных.
EXECUTE AS	Позволяет изменить контекст безопасности, в котором работает триггер. Например, вы можете настроить запуск триггера с правами db_owner, даже если событие, вызвавшее запуск триггера, имеет меньше прав.
DDL event	Имя события языка Transact-SQL, которое после запуска вызывает срабатывание триггера DDL. Полный список доступных событий : https://docs.microsoft.com/en-us/sql/relational-databases/triggers/ddl-events .
< method_specifier >	Указывает метод сборки для связывания с CLR-триггером.

Триггеры DDL


```
-- Trigger on a CREATE, ALTER, DROP, GRANT, DENY, REVOKE or UPDATE statement

DROP TRIGGER [ IF EXISTS ] trigger_name [ ,...n ]
ON { DATABASE | ALL SERVER }
[ ; ]

-- Trigger on a LOGON event (Logon Trigger)

DROP TRIGGER [ IF EXISTS ] trigger_name [ ,...n ]
ON ALL SERVER
```

События DDL

DDL-события, которые могут быть использованы для срабатывания триггера DDL или уведомления о событиях. Обратите внимание, что каждое событие соответствует инструкции Transact-SQL или хранимой процедуре с измененным синтаксисом инструкции для включения символа подчеркивания (_) между ключевыми словами.

CREATE_USER (Применяется в инструкции CREATE USER, процедурах **sp_adduser**и **sp_grantdbaccess**.)	ALTER_USER (Применяется в инструкции ALTER USER и процедуре **sp_change_users_login**.)	DROP_USER (Применяется в инструкции DROP USER, процедурах **sp_dropuser**и **sp_revokedbaccess**.)
CREATE_VIEW	ALTER_VIEW	DROP_VIEW

https://learn.microsoft.com/ru-ru/sql/relational-databases/triggers/ddl-events?view=sql-server-ver16

Группы событий DDL

Группы событий DDL, которые можно использовать для запуска триггера DDL или уведомления о событии.

Обратите внимание на иерархическую природу групп событий.

Например, триггер DDL или уведомление о событии, указывающее FOR DDL_TABLE_EVENTS (10018), охватывает инструкции CREATE TABLE, ALTER TABLE и DROP TABLE Transact-SQL. Триггер DDL или уведомление о событии, указывающее FOR DDL_TABLE_VIEW_EVENTS (10017), охватывает все инструкции Transact-SQL в типах DDL_TABLE_EVENTS, DDL_VIEW_EVENTS, DDL_INDEX_EVENTS и DDL_STATISTICS_EVENTS.

https://learn.microsoft.com/ru-ru/sql/relational-databases/triggers/ddl-event-groups?view=sql-server-ver16

Группы событий DDL

	Server Scope	Database Scope
DDL_SERVER_LEVEL_EVENTS (CREATE DATABASE, ALTER DATABASE, DROP DATABASE)	X	
DDL_ENDPOINT_EVENTS (CREATE ENDPOINT, ALTER ENDPOINT, DROP ENDPOINT)	X	
DDL_SERVER_SECURITY_EVENTS	X	
DDL_LOGIN_EVENTS (CREATE LOGIN, ALTER LOGIN, DROP LOGIN)	X	
DDL_GDR_SERVER_EVENTS (GRANT SERVER, DENY SERVER, REVOKE SERVER)	X	
DDL_AUTHORIZATION_SERVER_EVENTS (ALTER AUTHORIZATION SERVER)	X	
DDL_DATABASE_LEVEL_EVENTS		X
DDL_TABLE_VIEW_EVENTS		X
DDL_TABLE_EVENTS (CREATE TABLE, ALTER TABLE, DROP TABLE)		X
DDL_VIEW_EVENTS (CREATE VIEW, ALTER VIEW, DROP VIEW)		X
DDL_INDEX_EVENTS (CREATE INDEX, ALTER INDEX, DROP INDEX, CREATE XML INDEX)		X
DDL_STATISTICS_EVENTS (CREATE STATISTICS, UPDATE STATISTICS, DROP STATISTICS)		X
DDL_SYNONYM_EVENTS (CREATE SYNONYM, DROP SYNONYM)		X
DDL_FUNCTION_EVENTS (CREATE FUNCTION, ALTER FUNCTION, DROP FUNCTION)		X
DDL_PROCEDURE_EVENTS (CREATE PROCEDURE, ALTER PROCEDURE, DROP PROCEDURE)		X
DDL_TRIGGER_EVENTS (CREATE TRIGGER, ALTER TRIGGER, DROP TRIGGER)		X
DDL_EVENT_NOTIFICATION_EVENTS (CREATE EVENT NOTIFICATION, DROP EVENT NOTIFICATION)		X
DDL_ASSEMBLY_EVENTS (CREATE ASSEMBLY, ALTER ASSEMBLY, DROP ASSEMBLY)		X
DDL_TYPE_EVENTS (CREATE TYPE, DROP TYPE)		X
DDL_DATABASE_SECURITY_EVENTS		X
DDL_CERTIFICATE_EVENTS (CREATE CERTIFICATE, ALTER CERTIFICATE, DROP CERTIFICATE)		X
DDL_USER_EVENTS (CREATE USER, ALTER USER, DROP USER)		X
DDL_ROLE_EVENTS (CREATE ROLE, ALTER ROLE, DROP ROLE)		X
DDL_APPLICATION_ROLE_EVENTS (CREATE APPROLE, ALTER APPROLE, DROP APPROLE)		X
DDL_SCHEMA_EVENTS (CREATE SCHEMA, ALTER SCHEMA, DROP SCHEMA)		X
DDL_GDR_DATABASE_EVENTS (GRANT DATABASE, DENY DATABASE, REVOKE DATABASE)		X

Группы событий DDL


```
WITH DirectReports(name, parent_type, type, level, sort) AS
(
    SELECT CONVERT(varchar(255),type_name), parent_type, type, 1,
    CONVERT(varchar(255),type_name)
    FROM sys.trigger_event_types
    WHERE parent_type IS NULL
    UNION ALL
    SELECT  CONVERT(varchar(255), REPLICATE ('|    ' , level) +
    e.type_name),
        e.parent_type, e.type, level + 1,
    CONVERT (varchar(255), RTRIM(sort) + '|    ' + e.type_name)
    FROM sys.trigger_event_types AS e
        INNER JOIN DirectReports AS d
        ON e.parent_type = d.type
)
SELECT parent_type, type, name
FROM DirectReports
ORDER BY sort;
```

https://learn.microsoft.com/ru-ru/sql/relational-databases/triggers/ddl-event-groups?view=sql-server-ver16

События DDL и функция EVENTDATA()

Функция EVENTDATA позволяет получить сведения о событии, которое привело к срабатыванию триггера DDL. Эта функция возвращает значение типа **xml** . XML-схема содержит следующие сведения:

- время формирования события;
- идентификатор системного процесса (SPID), соответствующий соединению, во время которого был выполнен триггер;
- тип события, которое привело к срабатыванию триггера.

For example, the ALTER_TABLE event returns the following schema:

```
<EVENT_INSTANCE>
    <EventType>type</EventType>
    <PostTime>date-time</PostTime>
    <SPID>spid</SPID>
    <ServerName>name</ServerName>
    <LoginName>name</LoginName>
    <UserName>name</UserName>
    <DatabaseName>name</DatabaseName>
    <SchemaName>name</SchemaName>
    <ObjectName>name</ObjectName>
    <ObjectType>type</ObjectType>
    <TSQLCommand>command</TSQLCommand>
</EVENT_INSTANCE>
```

```
AS
DECLARE @data XML
SET @data = EVENTDATA()
INSERT ddl_log
   (PostTime, DB_User, Event, TSQL)
   VALUES
   (GETDATE(),
   CONVERT(nvarchar(100), CURRENT_USER),
   @data.value('(/EVENT_INSTANCE/EventType)[1]', 'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/TSQLCommand)[1]', 'nvarchar(2000)') ) ;
```

Получение сведений о триггерах DDL

Триггеры DDL уровня базы данных

Сведения о триггерах с областью действия на уровне базы данных
sys.triggers (Transact-SQL)

Сведения о событиях базы данных, вызывающих срабатывание триггеров
sys.trigger_events (Transact-SQL)

Определения триггеров уровня базы данных
sys.sql_modules (Transact-SQL)

Сведения о триггерах среды CLR уровня базы данных
sys.assembly_references (Transact-SQL)

Триггеры DDL уровня сервера

Сведения о триггерах с областью действия на уровне сервера
sys.server_triggers (Transact-SQL)

Сведения о событиях сервера, вызывающих срабатывание триггеров
sys.server_trigger_events (Transact-SQL)

Определения триггеров с областью действия на уровне сервера
sys.server_sql_modules (Transact-SQL)

Сведения о триггерах CLR с областью действия на уровне сервера
sys.server_assembly_modules (Transact-SQL)

Проблема

В идеальном мире привилегии **sa** были бы только у администратора баз данных, клавиша F5 нажималась бы только намеренно, каждое изменение проходило бы строгие процедуры контроля версий, и у нас были бы полные резервные копии всех баз данных каждую минуту.

Конечно, на самом деле мы имеем дело с совершенно другими обстоятельствами, и мы можем обнаружить себя (или услышать кого-то другого), говорящего: «Ой… как мне это исправить?» Один из наиболее распространенных сценариев, заключается в том, что кто-то редактирует хранимую процедуру несколько раз между резервными копиями или в каком-то цикле, а затем желает, чтобы у него была доступная версия (текущая - 1). Её еще нет в резервной копии, поэтому её нельзя восстановить; и пользователь, разумеется, закрыл свое окно без сохранения.

Проблема

У нас есть требование аудита по отключению xp_cmdshell. Однако я читал, что член роли системного администратора может повторно включить xp_cmdshell.
Есть ли способ предотвратить это?

Триггеры входа

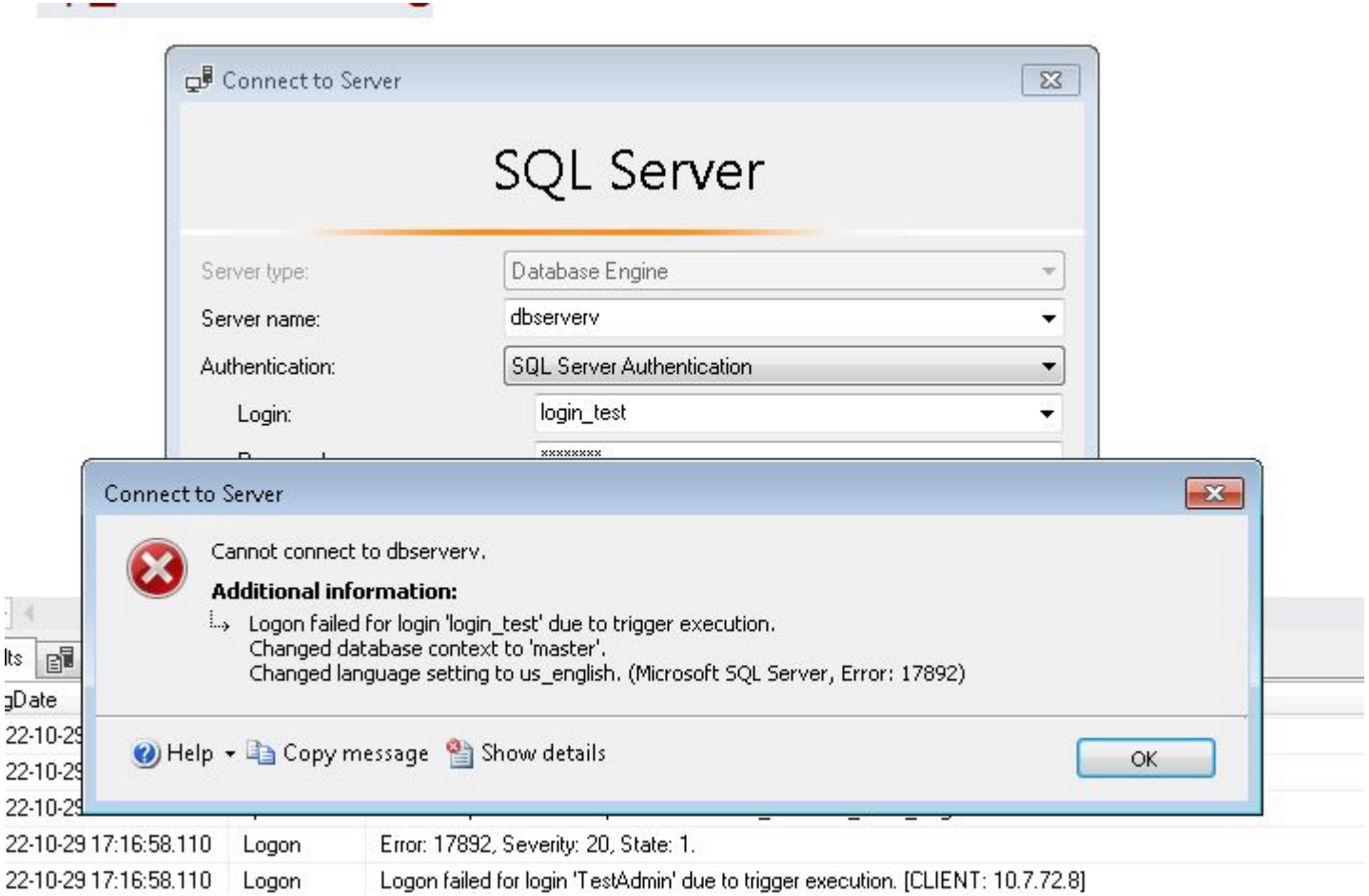
Триггеры входа вызывают срабатывание хранимых процедур в ответ на событие ***LOGON.*** Это событие вызывается при установке пользовательского сеанса с экземпляром SQL Server. Триггеры входа срабатывают после завершения этапа проверки подлинности при входе, но перед тем, как пользовательский сеанс реально устанавливается. Следовательно, все сообщения, которые возникают внутри триггера и обычно достигают пользователя, такие как сообщения об ошибках и сообщения от инструкции PRINT, перенаправляются в журнал ошибок SQL Server . Если проверка подлинности завершается сбоем, триггеры входа не срабатывают.
Можно использовать триггеры входа для проверки и управления сеансами сервера, например для отслеживания входов в систему, ограничения входов в SQL Server или ограничения числа сеансов для конкретного имени входа.

https://dotnettutorials.net/lesson/logon-trigger-in-sql-server/

Триггеры входа

```
USE master;
GO
CREATE LOGIN login_test WITH PASSWORD = N'3KHJ6dhx(0xVYsdf' MUST_CHANGE,
    CHECK_EXPIRATION = ON;
GO
GRANT VIEW SERVER STATE TO login_test;
GO
CREATE TRIGGER connection_limit_trigger
ON ALL SERVER WITH EXECUTE AS N'login_test'
FOR LOGON
AS
BEGIN
IF ORIGINAL_LOGIN()= N'login_test' AND
    (SELECT COUNT(*) FROM sys.dm_exec_sessions
            WHERE is_user_process = 1 AND
                original_login_name = N'login_test') > 3
    ROLLBACK;
END;
```

Если вы создали триггер для LOGON и у вас есть неверный код в триггере, при попытке входа в систему вы получите сообщение об ошибке, подобное показанному на рисунке 1.



Управление безопасностью триггеров

Как триггеры DML, так и триггеры DDL по умолчанию выполняются в контексте того пользователя, который вызывает триггер. Это пользователь, который выполняет инструкцию, вызывающую запуск триггера. Например, если пользователь **Mary** выполняет инструкцию DELETE, которая запускает триггер DML **DML_trigMary** , то код внутри **DML_trigMary** выполняется в контексте прав доступа пользователя **Mary**. Этим поведением по умолчанию могут воспользоваться те пользователи, которые хотят внести небезопасный код в экземпляр базы данных или сервера.

```
CREATE TRIGGER DDL_trigJohnDoe
ON DATABASE FOR ALTER_TABLE
AS
SET NOCOUNT ON;

BEGIN TRY
  EXEC(N'
    USE [master];
    GRANT CONTROL SERVER TO [JohnDoe];
');
END TRY
BEGIN CATCH
  DECLARE @DoNothing INT;
END CATCH;
GO
```

В этом триггере подразумевается, что как только пользователь с разрешением на выполнение инструкции GRANT CONTROL SERVER, например член предопределенной роли сервера **sysadmin**, выполнит инструкцию ALTER TABLE, пользователь **JohnDoe** получит разрешение CONTROL SERVER.
Другими словами, хотя **JohnDoe** не может предоставить разрешение CONTROL SERVER самому себе, он создал код триггера, который предоставит ему разрешение работать с повышенными правами доступа. Эта уязвимость относится как к триггерам DML, так и к триггерам DDL.

Способы защиты триггеров

Чтобы предотвратить выполнение кода триггера с повышенными правами доступа, примите следующие меры.

- Проверьте базу данных и экземпляр сервера на наличие триггеров DDL и DDL. Для этого выполните соответствующие запросы к представлениям каталогов sys.triggers и sys.server_triggers .

```
SELECT type, name, parent_class_desc FROM sys.triggers
UNION ALL
SELECT type, name, parent_class_desc FROM sys.server_triggers;
```

- с помощью инструкции DISABLE TRIGGER запретите триггеры, которые могут нарушить целостность базы данных или сервера при выполнении с повышенными правами доступа.

```
DISABLE TRIGGER ALL ON DATABASE;
```

```
DISABLE TRIGGER ALL ON ALL SERVER;
```

Способы защиты триггеров

```
DECLARE @schema_name sysname, @trigger_name sysname, @object_name sysname;
DECLARE @sql nvarchar(max);
DECLARE trig_cur CURSOR FORWARD_ONLY READ_ONLY FOR
    SELECT SCHEMA_NAME(schema_id) AS schema_name,
        name AS trigger_name,
        OBJECT_NAME(parent_object_id) AS object_name
    FROM sys.objects WHERE type IN ('TR', 'TA');

OPEN trig_cur;
FETCH NEXT FROM trig_cur INTO @schema_name, @trigger_name, @object_name;

WHILE @@FETCH_STATUS = 0
BEGIN
    SELECT @sql = N'DISABLE TRIGGER ' + QUOTENAME(@schema_name) + N'.'
        + QUOTENAME(@trigger_name)
        + N' ON ' + QUOTENAME(@schema_name) + N'.'
        + QUOTENAME(@object_name) + N'; ';
    EXEC (@sql);
    FETCH NEXT FROM trig_cur INTO @schema_name, @trigger_name, @object_name;
END;
GO

-- Verify triggers are disabled. Should return an empty result set.
SELECT * FROM sys.triggers WHERE is_disabled = 0;
GO

CLOSE trig_cur;
DEALLOCATE trig_cur;
```

Эта инструкция отключает все триггеры DML в текущей базе данных: