

BAZY DANYCH

WYKŁAD VI

WYZWALACZE, PROCEDURY WBUDOWANE,
TRANSAKCJE

WYZWALACZE

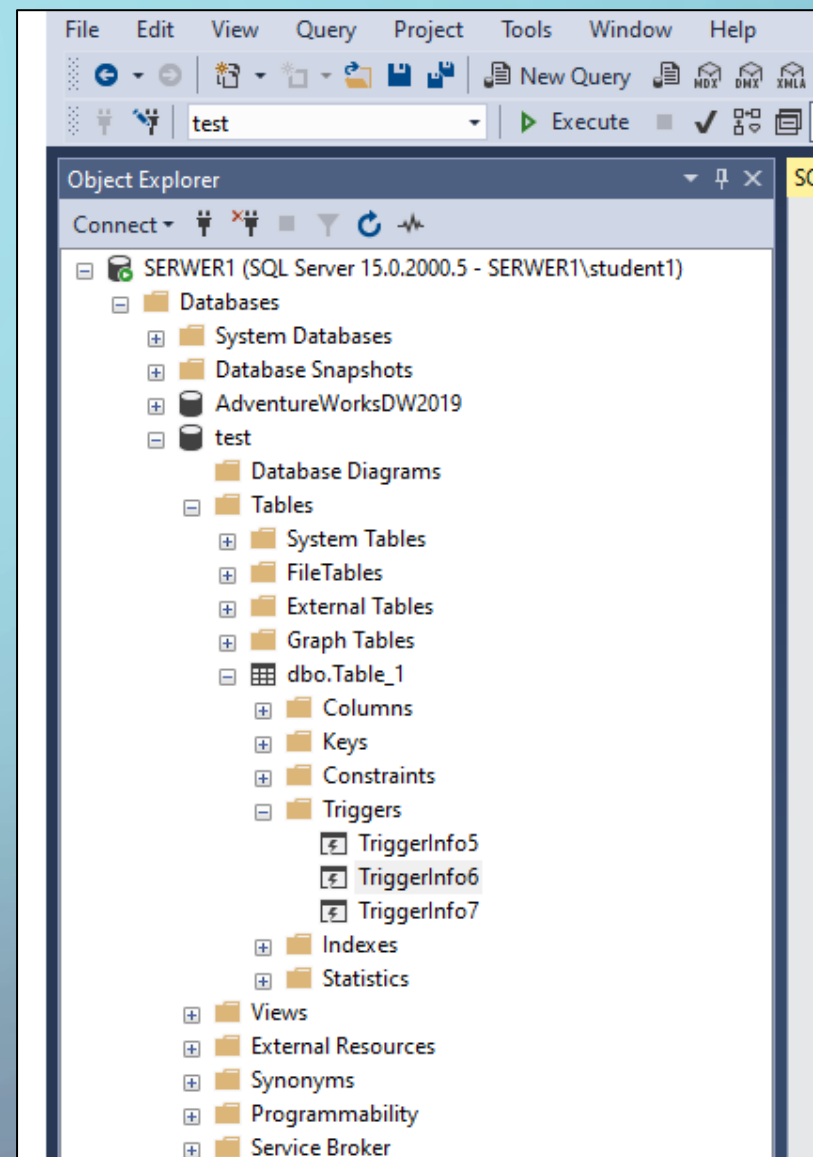
Obiekty bazy danych

- Tabele
- Widoki
- Reguły i ograniczenia
 - Typ danych
 - Ograniczenie CHECK
 - Ograniczenie PRIMARY KEY
 - Ograniczenie FOREIGN KEY
- Procedury składowane
- Wyzwalacze
- Transakcje (?)

WYZWALACZE

Trigger (wyzwalacz) jest obiektem bazy danych, inicjowanym automatycznie jako reakcja po zdarzeniu na serwerze bazy danych. W rzeczywistości wyzwalacze często są określane jako „specjalny rodzaj procedury składowanej”. Wyzwalacze mogą ograniczać dostęp do pewnych danych, rejestrować zmiany danych lub nadzorować modyfikacje danych.

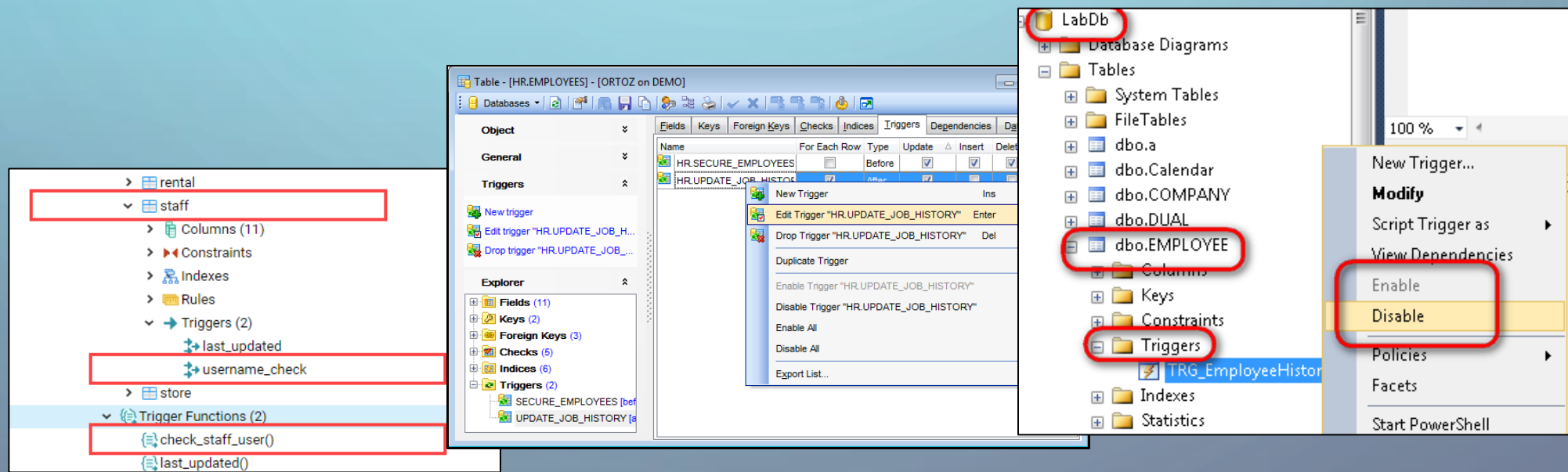
W przeciwieństwie do zwykłej procedury składowanej wyzwalacz nie może zostać jawnie wywołany.



WYZWALACZE

Wiele systemów baz danych posiada wyzwalacze: Microsoft SQL Server, PostgreSQL, Sybase, Oracle, Firebird, SQLite, InterBase SQL, MySQL (od wersji 5.0.2).

Standard SQL definiuje wyzwalacze dopiero od wersji 99, wobec czego różne systemy bazodanowe opracowały własną składnię tworzenia wyzwalaczy, dlatego też składnia wyzwalaczy w wielu systemach różni się od wytycznych.



WYZWALACZE

Istnieją trzy typowe zdarzenia powodujące wykonanie wyzwalaczy:

- dopisanie nowego rekordu do bazy danych w wyniku wykonania instrukcji INSERT,
- zmiana zawartości rekordu w wyniku wykonania instrukcji UPDATE,
- usunięcie rekordu w wyniku wykonania instrukcji DELETE.

Główne cechy wyzwalaczy to:

- nie mogą mieć parametrów (ale mogą zapisywać dane w tabelach tymczasowych),
- nie mogą zatwierdzać transakcji (COMMIT) ani ich wycofywać (ROLLBACK), ponieważ działają w kontekście instrukcji SQL, która spowodowała ich uruchomienie
- mogą generować dodatkowe błędy, jeżeli są źle napisane.

WYZWALACZE

Wyzwalacze mogą być wykorzystywane w wielu sytuacjach, np. do:

- złożonych warunków poprawności dla kolumny lub wiersza
- implementacji złożonych zachowań dla integralności referencyjnej
- obliczania wartości kolumn
- inspekcji działania użytkowników

Wyzwalacze mogą wykonywać następujące czynności:

- porównywać wersje danych z „przed” i „po” modyfikacji
- anulować nieprawidłowe modyfikacje
- odczytywać dane z innych tabel
- modyfikować inne tabele
- wykonywać procedury składowane

WYZWALACZE

Cechy wyzwalaczy:

- Wyzwalacz jest wykonywany tylko raz dla danego polecenia niezależnie od tego ile wierszy modyfikuje te polecenie.
- Wyzwalacz jest integralną częścią transakcji związanej z realizacją polecenia
- Dla każdej instrukcji dotyczącej tych samych danych oraz tych samych obiektów jak i logowań można utworzyć kilka wyzwalaczy działających niezależnie między sobą.
- Wyzwalacze DML korzystają z tabel logicznych, tak zwanych koncepcyjnych, czyli DELETED oraz INSERTED zawierających stare oraz nowe wartości wierszy.
- Wyzwalacze nie powinny zwracać żadnych wartości, powinny natomiast sprawdzać oraz modyfikować dane w oparciu o instrukcje modyfikowania jak i definiowania danych.
- Często wykorzystywane są do logowania zmian zachodzących w bazie danych, czyli:
 - kto dokonał zmiany
 - kiedy zmiana miała miejsce
 - z czego na co nastąpiła zmiana

WYZWALACZE

W definicji wyzwalacza określamy:

- nazwę wyzwalacza,
- tabelę dla której tworzymy wyzwalacz,
- akcje na które wyzwalacz będzie reagował,
- typ wyzwalacza,
- ciało wyzwalacza (odpowiednik ciała procedury składowanej) – czyli kod wykonywany przez wyzwalacz .

Rodzaje wyzwalaczy

- a) Wyzwalacze DML (Data Manipulation Language). Reagują one na „zdarzenia” Insert, Update, Delete,
- b) Wyzwalacze DDL (Data Definition Language). Reagują one na „zdarzenia” CREATE, ALTER, DROP.
- c) Wyzwalacze LOGON

WYZWALACZE

Struktura wyzwalaczy

```
CREATE TRIGGER [nazwa_schematu.]nazwa_wyzwalacza
ON { table | view }
[ WITH <dml_opcje_wyzwalacza> [,...n] ]
{ FOR | AFTER | INSTEAD OF }
[ INSERT ] [ , ] [ UPDATE ] [ , ] [ DELETE ]
[ WITH APPEND ]
[ NOT FOR REPLICATION ]
AS wyrażenie_sql [ ; ] [,...n]
```

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

```
CREATE TRIGGER nazwa_wyzwalacza
ON { ALL SERVER | DATABASE }
[ WITH <ddl_opcje_wyzwalacza> [,...n] ]
{ FOR | AFTER } { rodzaj_zdarzenia | event_group } [,...n]
AS wyrażenie_sql [ ; ] [,...n]
```

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

```
CREATE TRIGGER nazwa_wyzwalacza
ON ALL SERVER
[ WITH <opcje_wyzwalacza_logon> [,...n] ]
{ FOR | AFTER } LOGON
AS wyrażenie_sql [ ; ] [,...n]
```

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

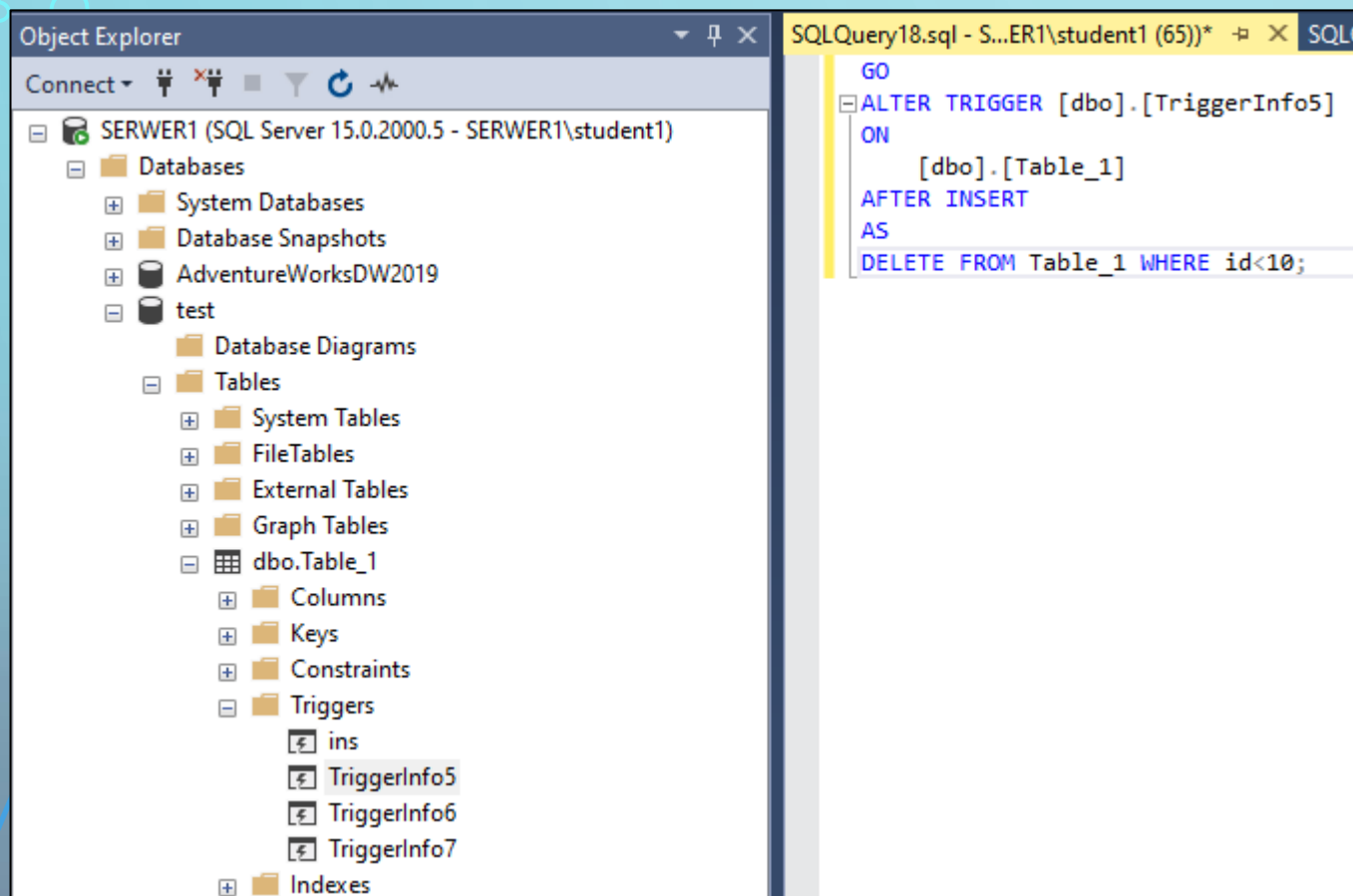
WYZWALACZE

Rodzaje wyzwalaczy DML:

- Typ AFTER (po operacji DML)
- Typ BEFORE (przed operacją DML)
- Typ INSTEAD OF (zamiast operacji DML)

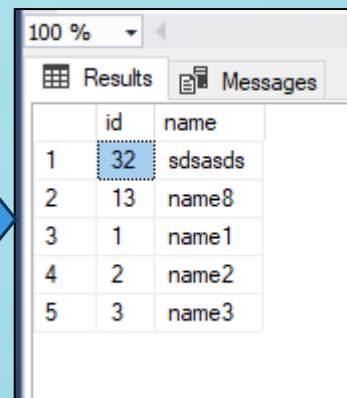
WYZWALACZE

Przykład wyzwalacza AFTER INSERT w MSSQL



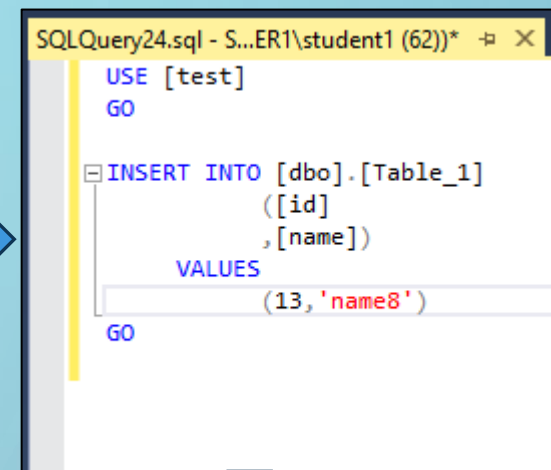
The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Object Explorer' pane displays the database structure for 'SERWER1 (SQL Server 15.0.2000.5 - SERWER1\student1)'. The 'test' database is expanded, showing 'Tables' and 'dbo.Table_1'. On the right, the 'SQLQuery18.sql' window shows the following SQL code:

```
GO
ALTER TRIGGER [dbo].[TriggerInfo5]
ON
    [dbo].[Table_1]
AFTER INSERT
AS
DELETE FROM Table_1 WHERE id<10;
```



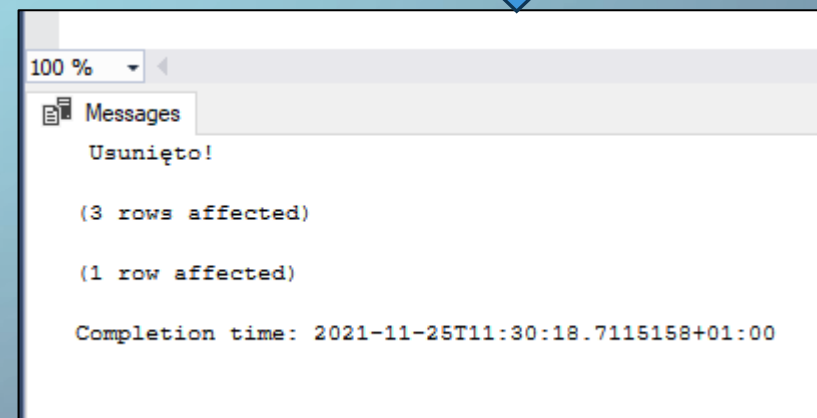
The 'Results' pane shows the data in 'Table_1' after the trigger execution. The table has two columns: 'id' and 'name'. The data is as follows:

id	name
1	32
2	13
3	name8
4	1
5	name1
6	2
7	name2
8	3
9	name3



The 'SQLQuery24.sql' window shows the following SQL code:

```
USE [test]
GO
INSERT INTO [dbo].[Table_1]
    ([id]
    , [name])
VALUES
    (13, 'name8')
GO
```



The 'Messages' pane shows the execution results of the SQL query. The messages are:

```
Usunięto!


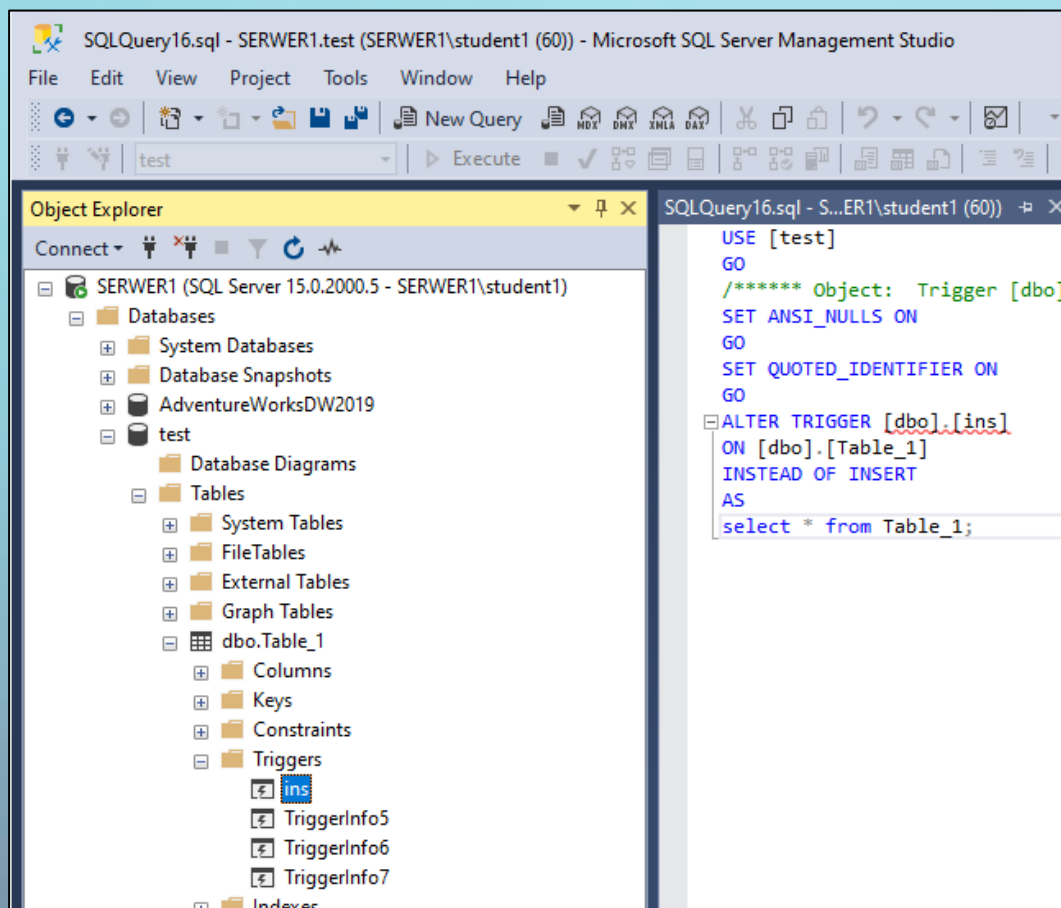
(3 rows affected)

(1 row affected)


Completion time: 2021-11-25T11:30:18.7115158+01:00
```

WYZWALACZE

Przykład wyzwalacza INSTEAD OF w MSSQL



```
USE [test]
GO
INSERT INTO [dbo].[Table_1]
([id]
,[name])
VALUES
(6, 'name')
GO
```

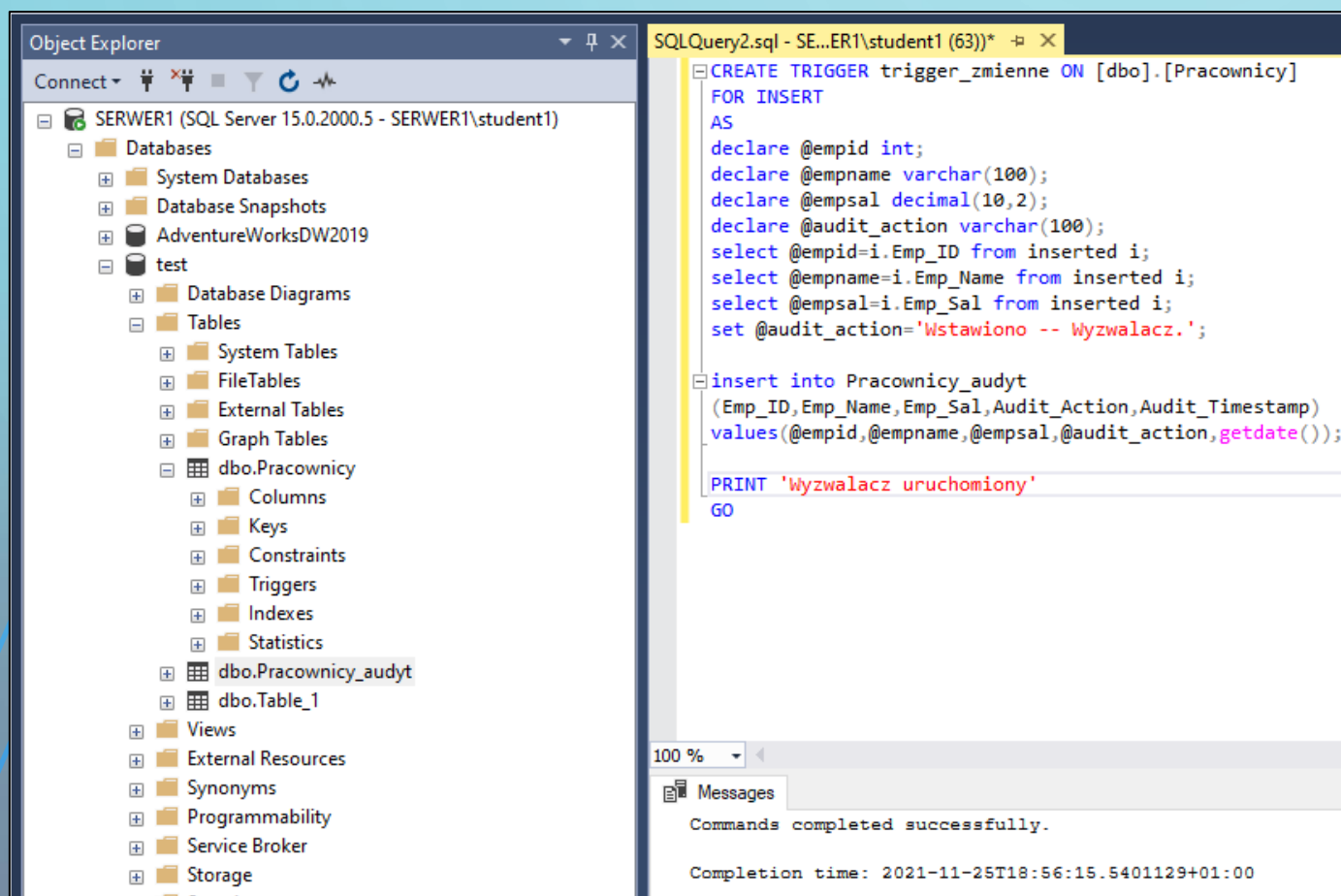


100 %

Results		Messages
id	name	
1	32	sdsasds

WYZWALACZE

Przykład wyzwalacza z wykorzystaniem zmiennych (MSSQL)



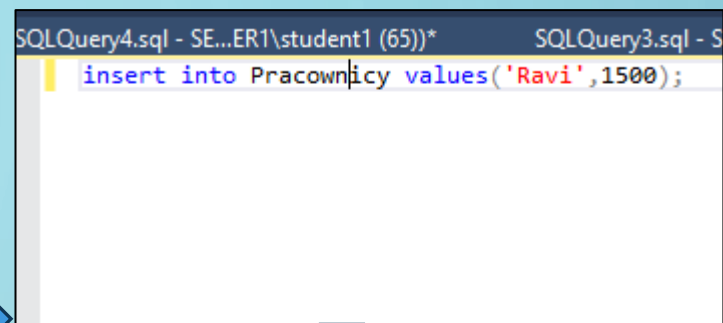
The screenshot shows the SQL Server Enterprise Manager interface on the left, displaying the database structure for 'SERWER1'. The right pane shows a SQL query window with the following code:

```
CREATE TRIGGER trigger_zmienne ON [dbo].[Pracownicy]
FOR INSERT
AS
declare @empid int;
declare @empname varchar(100);
declare @empsal decimal(10,2);
declare @audit_action varchar(100);
select @empid=i.Emp_ID from inserted i;
select @empname=i.Emp_Name from inserted i;
select @empsal=i.Emp_Sal from inserted i;
set @audit_action='Wstawiono -- Wyzwalacz.';

insert into Pracownicy_audyt
(Emp_ID,Emp_Name,Emp_Sal,Audit_Action,Audit_Timestamp)
values(@empid,@empname,@empsal,@audit_action,getdate());

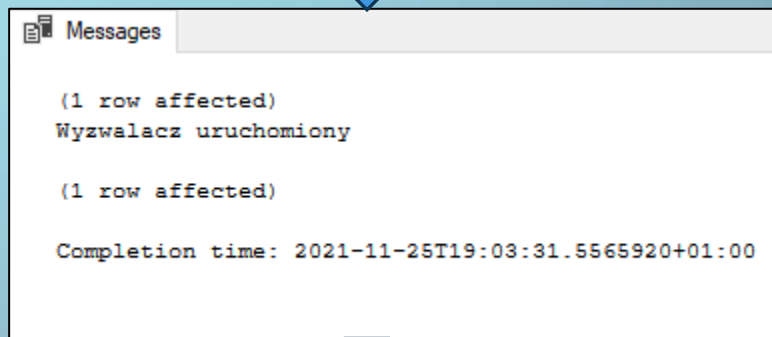
PRINT 'Wyzwalacz uruchomiony'
GO
```

The Messages pane at the bottom shows: "Commands completed successfully." and "Completion time: 2021-11-25T18:56:15.5401129+01:00".



SQLQuery4.sql - SE...ER1\student1 (65))* SQLQuery3.sql - S

```
insert into Pracownicy values('Ravi',1500);
```

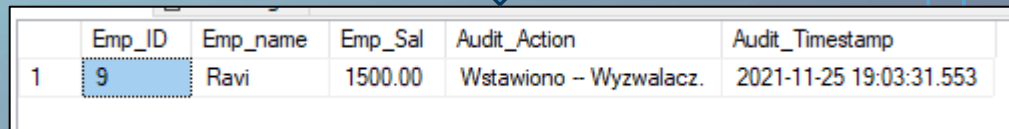


Messages

(1 row affected)
Wyzwalacz uruchomiony

(1 row affected)

Completion time: 2021-11-25T19:03:31.5565920+01:00



	Emp_ID	Emp_name	Emp_Sal	Audit_Action	Audit_Timestamp
1	9	Ravi	1500.00	Wstawiono - Wyzwalacz.	2021-11-25 19:03:31.553

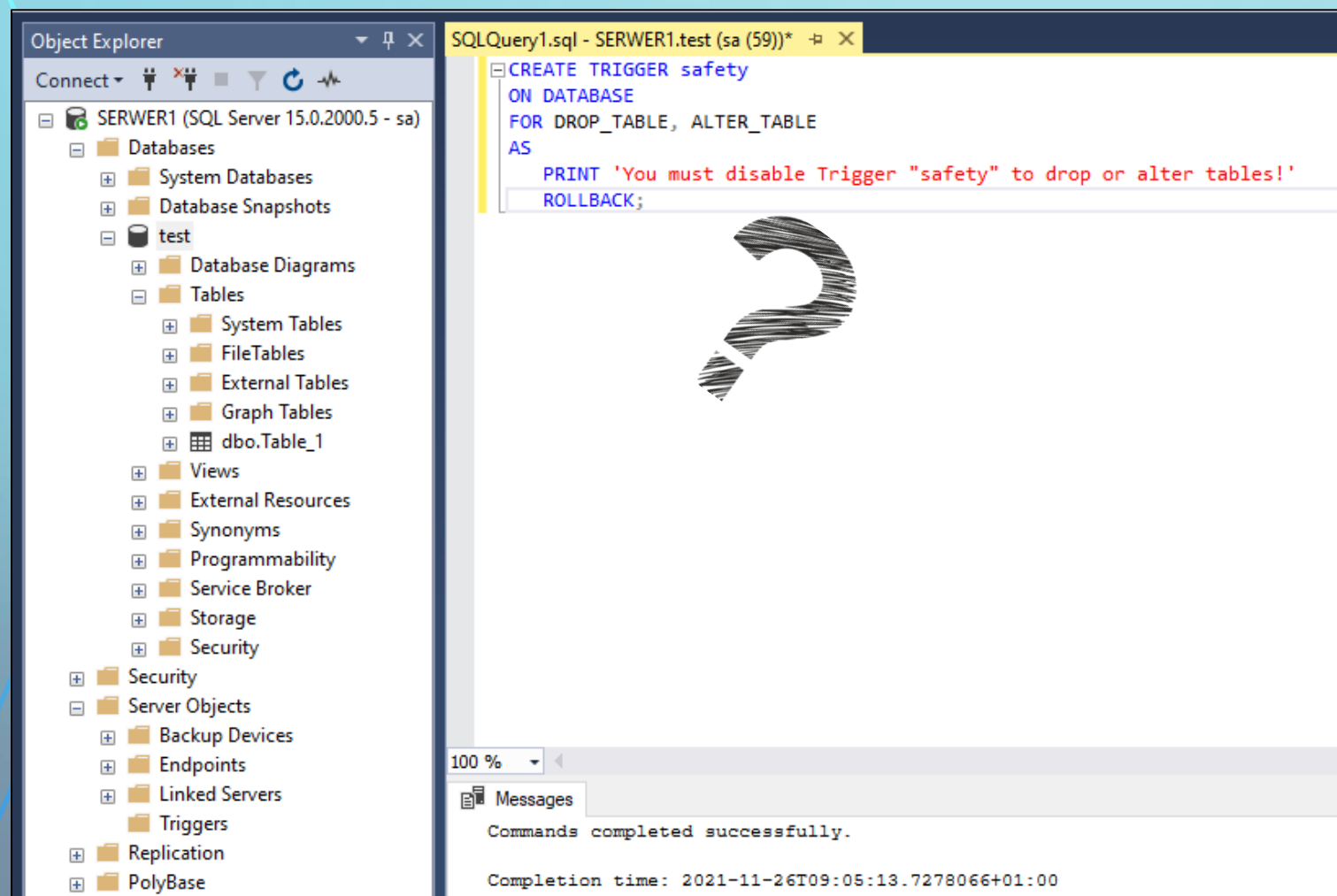
WYZWALACZE

Wyzwalacze DDL

- Sq to wyzwalacze działające w kontekście bazy danych
- Sq to także wyzwalacze działające w kontekście serwera

WYZWALACZE

Przykład wyzwalacza DDL w MSSQL



Object Explorer

- SERVER1 (SQL Server 15.0.2000.5 - sa)
 - Databases
 - System Databases
 - Database Snapshots
 - test
 - Database Diagrams
 - Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.Table_1
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Service Broker
 - Storage
 - Security
 - Security
 - Server Objects
 - Backup Devices
 - Endpoints
 - Linked Servers
 - Triggers
 - Replication
 - PolyBase

SQLQuery1.sql - SERVER1.test (sa (59))*

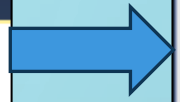
```
CREATE TRIGGER safety
ON DATABASE
FOR DROP_TABLE, ALTER_TABLE
AS
    PRINT 'You must disable Trigger "safety" to drop or alter tables!'
    ROLLBACK;
```

100 %

Messages


Commands completed successfully.

Completion time: 2021-11-26T09:05:13.7278066+01:00



SQLQuery3.sql - SERVER1.test (sa (67))*

```
DROP TABLE [dbo].[Table_1]
```



Messages

You must disable Trigger "safety" to drop or alter tables!
Msg 3609, Level 16, State 2, Line 1
The transaction ended in the trigger. The batch has been aborted.

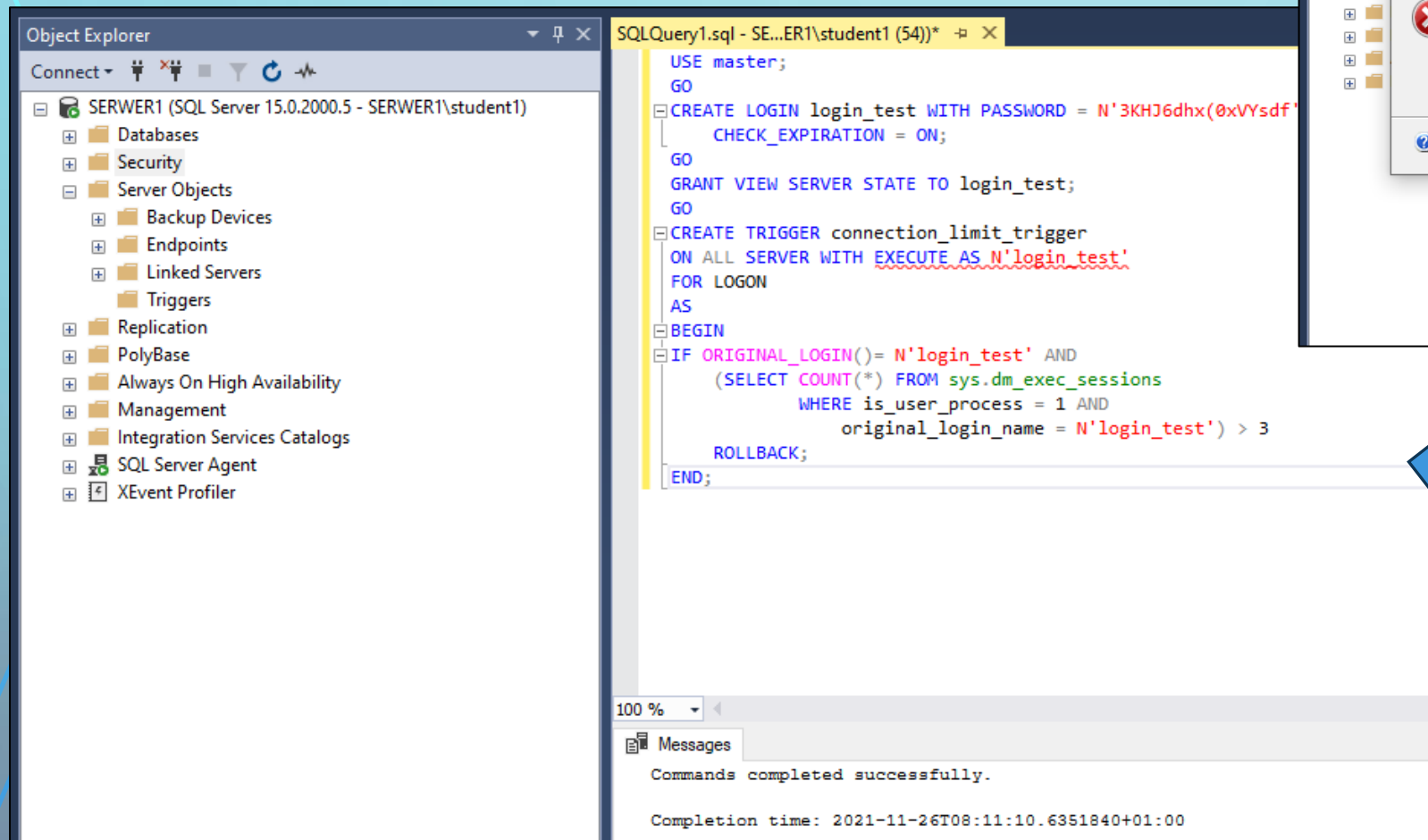
Completion time: 2021-11-26T09:09:00.9157325+01:00

WYZWALACZE

Wyzwalaczy logowania można używać do inspekcji i kontroli sesji serwera, na przykład przez śledzenie aktywności logowania, ograniczanie logowania do programu SQL Server lub ograniczanie liczby sesji dla określonego logowania.

WYZWALACZE

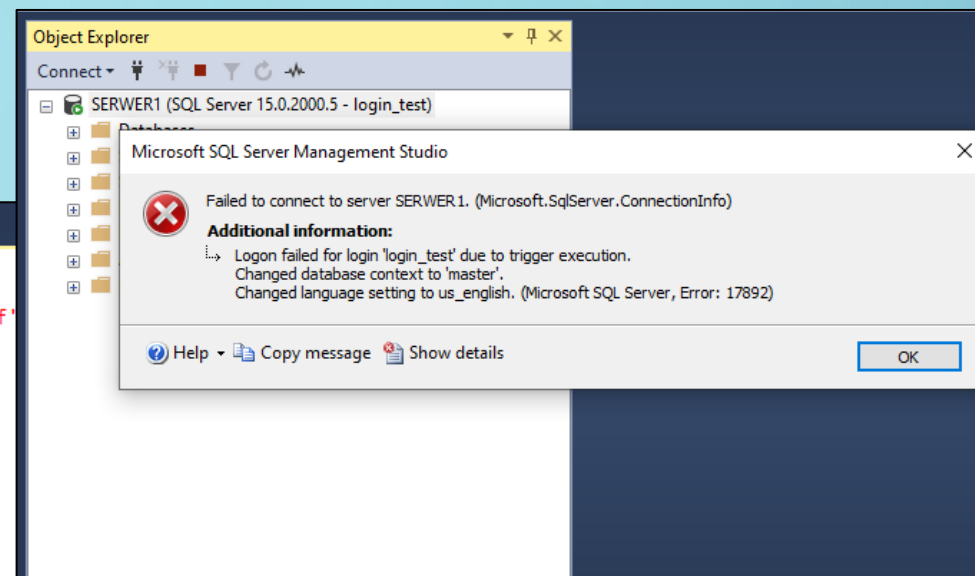
Przykład wyzwalacza LOGON w MSSQL



The screenshot displays the Microsoft SQL Server Enterprise Manager interface. On the left, the 'Object Explorer' pane shows the server hierarchy for 'SERWER1 (SQL Server 15.0.2000.5 - SERWER1\student1)'. The 'Security' folder is expanded, showing 'Logins'. The main pane shows a SQL query window titled 'SQLQuery1.sql - SE...ER1\student1 (54))*'. The query contains the following T-SQL code:

```
USE master;
GO
CREATE LOGIN login_test WITH PASSWORD = N'3KHJ6dhx(0xVYsdf'
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;
```

At the bottom of the window, the 'Messages' pane shows the output: 'Commands completed successfully.' and 'Completion time: 2021-11-26T08:11:10.6351840+01:00'.



The screenshot shows an error message dialog box titled 'Microsoft SQL Server Management Studio'. The message states: 'Failed to connect to server SERWER1. (Microsoft.SqlServer.ConnectionInfo)'. Below this, the 'Additional information:' section provides details: 'Logon failed for login 'login_test' due to trigger execution. Changed database context to 'master'. Changed language setting to us_english. (Microsoft SQL Server, Error: 17892)'. The dialog box includes buttons for 'Help', 'Copy message', 'Show details', and 'OK'. A blue arrow points from the SQL query window towards this error dialog.

WYZWALACZE

Zalety:

- automatyzacja działań wynikających z logiki projektu lub logiki biznesowej,
- możliwość implementacji bardzo złożonych reguł sprawdzających integralność danych,
- usprawnienie procesów związanych z administrowaniem bazą danych,
- minimalizowanie prawdopodobieństwa utraty spójności danych,
- możliwość poprawy wydajności (kolumny wyliczane).

Wady:

- spadek wydajności związany z dodatkowymi działaniami podczas realizacji poleceń DDL lub DML,
- zwiększenie skomplikowania logiki bazy danych,
- możliwość generowania skomplikowanych błędów w sytuacji nieumiejętnego wykorzystania mechanizmu wyzwalaczy.

PROCEDURY SKŁADOWE

Procedura składowana (ang. stored procedure) jest nazwanym zbiorem poleceń w języku SQL, pozwalającym na wykonanie w środowisku bazodanowym pewnych działań programistycznych. Procedury są przechowywane na serwerze baz danych a ich kompilacja następuje przy pierwszym wykonaniu.

PROCEDURY SKŁADOWE

Proces wykonania pojedynczego zapytania w MSSQL Server dzieli się na następujące etapy:

- sprawdzenie i rozdzielenie kodu na fragmenty - dokonywany jest podział kodu na fragmenty nazywane symbolami (analiza leksykalna),
- sprawdzenie kodu pod względem poprawności semantycznej i syntaktycznej – dokonywane jest sprawdzenie, czy kod nie odwołuje się do nieistniejących obiektów lub nie używa nieistniejących poleceń, sprawdzana się także jest poprawność użytej składni,
- standaryzacja wyodrębnionej części kodu – silnik wykonywania zapytań zapisuje kod w jednoznacznej postaci (np. usuwa niepotrzebne symbole),
- optymalizacja – wybierany jest optymalny sposób dostępu do danych, tzn. taki plan wykonania zapytania, w którym serwer będzie przeszukiwał najmniejszą ilość stron danych (na optymalizację szczególny wpływ ma struktura indeksów oraz sposób łączenia tabel),
- kompilacja i wykonanie – zapytanie jest kompilowane według optymalnego planu wykonania i w takiej postaci wykonywane,
- zwrócenie wyników – wyniki zapytania zwracane są do klienta.

PROCEDURY SKŁADOWE

Wykonywanie procedur składowanych różni się w stosunku do procesu wykonywania pojedynczego zapytania SQL. Utworzenie i pierwsze wykonanie procedury w Microsoft SQL Server można podzielić na następujące kroki:

- zdefiniowanie procedury składowanej – tworzenie odbywa się za pomocą polecenia `CREATE PROCEDURE`,
- analiza poprawności kodu procedury - kontrola poprawności syntaktycznej kodu procedury,
- zapisanie procedury w bazie danych – nazwa procedury wraz z kodem jest zapisywana do odpowiednich widoków systemowych bazy danych (`sysobjects` oraz `syscomments`),
- wywołanie procedury – wywołanie wraz z odpowiednimi parametrami za pomocą polecenia `EXEC`,
- właściwe wykonanie procedury – optymalizacja planu wykonania i kompilacja,
- buforowanie planu wykonania – skompilowany optymalny plan wykonania jest zapisywany w tzw. buforze procedur, skąd jest wczytywany przy następnym wywołaniu procedury.

PROCEDURY SKŁADOWE

Rekompilacja procedur składowanych:

Gdy procedura jest kompilowana po raz pierwszy lub ponownie, plan zapytań procedury jest optymalizowany pod kątem bieżącego stanu bazy danych i jej obiektów. Jeśli baza danych ulegnie znaczącym zmianom (może tak być z wielu powodów, np. zmiany struktury indeksów lub zapisania dużej ilości rekordów), ponowna kompilacja procedury aktualizuje i optymalizuje plan zapytań procedury pod kątem tych zmian. Może to poprawić wydajność przetwarzania procedury.

Rekompilacji, czyli ponownej kompilacji procedury można dokonać poprzez:

- ponowne uruchomienie programu SQL Server,
- dodając w definicji procedury klauzulę `WITH RECOMPILE`.
- używając specjalnej systemowej procedury rekompilującej (w Microsoft SQL Server jest to procedura `sp_recompile`).

PROCEDURY SKŁADOWE

Struktura procedury składowanej:

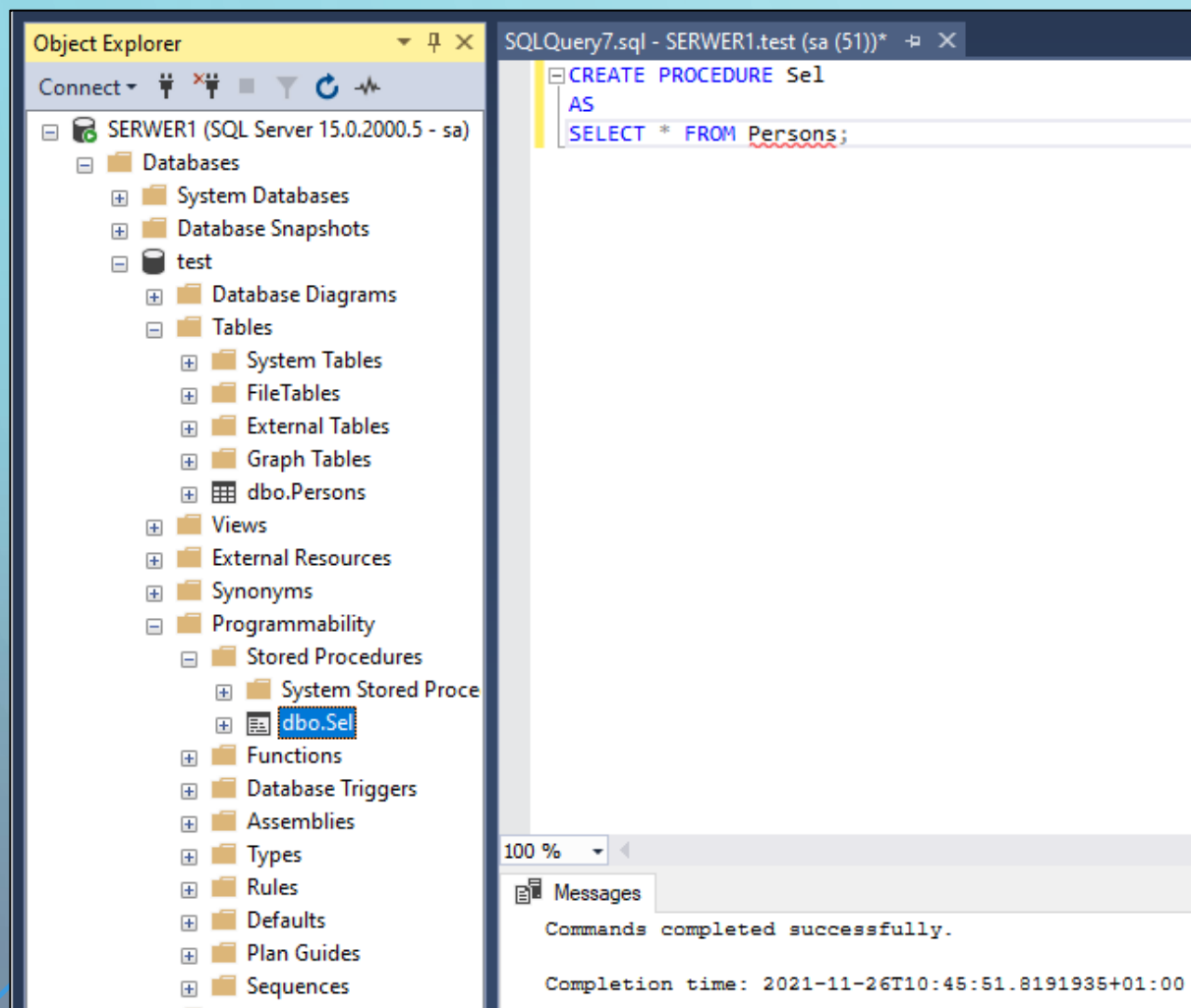
```
CREATE { PROC | PROCEDURE } [nazwa_schematu.]nazwa_procedury  
  [ @parametr typ_danych[,...n] ]  
  [ WITH <opcje_procedury> [,...n] ]  
  [ FOR REPLICATION ]  
AS  
  ciało_procedury  
  
<opcja_procedury> ::=  
  [ ENCRYPTION ]  
  [ RECOMPILE ]  
  [ EXECUTE_AS-Clause ]
```

W definicji procedury składowanej określamy:

- nazwę procedury
- nazwy
- typy danych
- kierunek działania parametrów procedury
- ciało procedury (kod wykonywany przez procedurę)

PROCEDURE SKŁADOWE

Przykład:

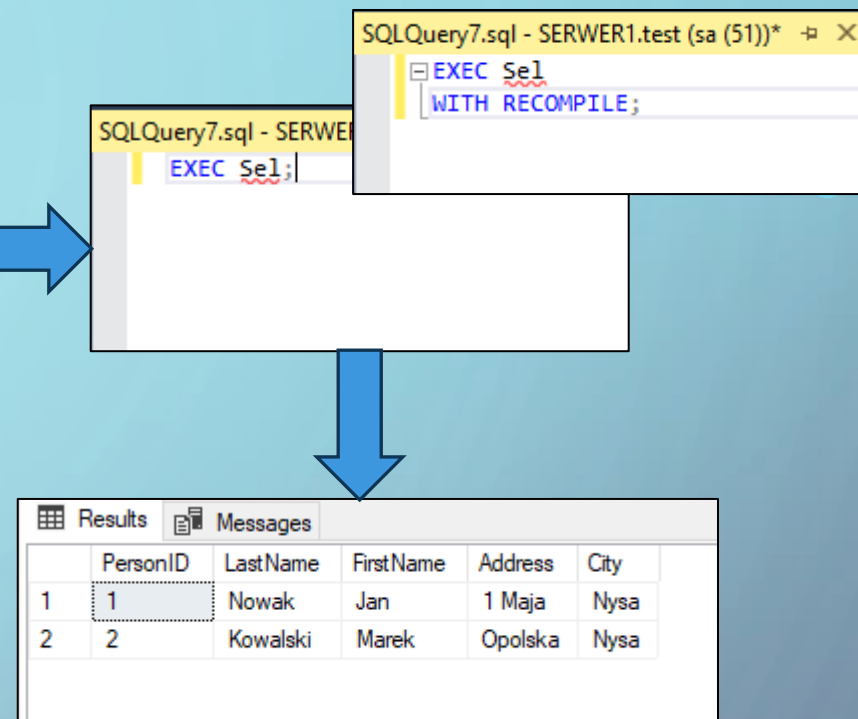


The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the 'test' database structure, with 'dbo.Sel' highlighted under 'Stored Procedures'. The main pane shows the SQL query editor with the following code:

```
CREATE PROCEDURE Sel
AS
SELECT * FROM Persons;
```

At the bottom, the Messages pane displays the following text:

```
Commands completed successfully.
Completion time: 2021-11-26T10:45:51.8191935+01:00
```



The diagram illustrates the execution of the stored procedure. A blue arrow points from the SQL query editor to a box containing the execution command:

```
EXEC Sel;
```

Another blue arrow points from this box to a 'Results' pane, which displays the following data:

	PersonID	LastName	FirstName	Address	City
1	1	Nowak	Jan	1 Maja	Nysa
2	2	Kowalski	Marek	Opolska	Nysa

PROCEDURY SKŁADOWE

Procedury składowane mogą przyjmować parametry wywołania. Ilość i typ danych, które należy podać przy wywołaniu procedury składowanej, określamy w trakcie tworzenia procedury. W zależności od tego, czy parametry będą potrzebne do wykonania procedury, czy też mają być one przez procedurę zwrócone, wyróżniamy dwa rodzaje parametrów: wejściowe (INPUT) oraz wyjściowe (OUTPUT).

PROCEDURY SKŁADOWE

Przykład:

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the server hierarchy for 'SERWER1 (SQL Server 15.0.2000.5 - sa)'. The 'Programmability' folder is expanded, showing 'Stored Procedures' and 'System Stored Procedures'. The 'dbo.LN' procedure is highlighted. The main query window, titled 'SQLQuery10.sql - SERWER1.test (sa (53))*', contains the following SQL code:

```
CREATE PROCEDURE LN
@nazwisko varchar(40)
AS
SELECT *
FROM Persons
WHERE LastName=@nazwisko
GO
```

At the bottom, the 'Messages' window shows the status: 'Commands completed successfully.' and 'Completion time: 2021-11-26T11:18:40.6084676+01:00'.

A callout box shows the execution of the stored procedure. The window title is 'SQLQuery10.sql - SERWER1.test (sa (53))*'. The code entered is:

```
exec LN @nazwisko='Nowak';
```

A callout box shows the results of the stored procedure execution. The window title is 'SQLQuery10.sql - SERWER1.test (sa (53))*'. The 'Results' tab is active, displaying a table with the following data:

	PersonID	LastName	FirstName	Address	City
1	1	Nowak	Jan	1 Maja	Nysa

PROCEDURY SKŁADOWE

Procedury składowane dzięki temu, że są zapisane na serwerze oraz dzięki skompilowanemu planowi wykonania przechowywanemu w buforze procedur posiadają dwie zasadnicze zalety:

- zwiększają wydajność bazy danych,
- ograniczają ruch w sieci (przesyłane są tylko nazwy procedur i wartości parametrów).

PROCEDURY SKŁADOWE

Procedury składowane mają wiele zalet z punktu widzenia programistów aplikacji bazodanowych:

- zapewniają jedną logikę biznesową dla wszystkich aplikacji klienckich.
- przesłaniają szczegóły tabel w bazie danych (przezroczystość struktury dla zwykłego użytkownika aplikacji).
- umożliwiają modyfikację danych bez bezpośredniego dostępu do tabel bazy danych.
- dostarczają mechanizmów bezpieczeństwa (można nadawać uprawnienia do wykonywania procedur poszczególnym użytkownikom bazy danych).
- umożliwiają programowania modułarne (procedurę można wielokrotnie wywołać; procedurę tworzy osoba wyspecjalizowana w bazach danych a programista aplikacji jedynie ją uruchamia).
- dzięki procedurom możliwe jest szybsze wykonanie zapytań (wykonują się one na serwerze, są optymalizowane i umieszczane w pamięci przy pierwszym wykonaniu).
- zmniejszają ruch sieciowy (wiele zapytań można zastąpić wywołaniem jednej procedury składowanej).

PROCEDURY SKŁADOWE

Niektóre z aspektów działania procedur składowych można potraktować jako ich wady:

- następstwem rekompilacji czasem jest zmniejszenie wydajności procedury (rekompilacja musi być przeprowadzana w odpowiednim momencie).
- w przypadku zagnieżdżania procedur składowanych zmienia się kontekst ich wykonania (procedura zagnieżdżana wykonuje się z uprawnieniami innej procedury).
- aby tworzyć poprawnie działające procedury składowane niezbędne jest poznanie zaawansowanych mechanizmów języka programowania baz danych, takich jak zmienne, funkcje i procedury systemowe czy obsługa błędów.

TRANSAKCJE

Transakcją określa się sekwencję operacji wykonywanych na bazie danych stanowiących w istocie pewną całość. Każda transakcja ma cztery kluczowe wartości, które są określone skrótem ACID. Jest to akronim od Atomic Consistent Isolated Durability (atomicity - atomowość, consistency - spójność, isolation - izolacja, durability - trwałość).

TRANSAKCJE

ACID:

- **ATOMIC** oznacza, iż cała operacja jest traktowana jako pojedyncza jednostka. Wykonywane jest wszystko albo nic.
- **CONSISTENT** oznacza, że zakończona transakcja pozostawi bazę danych w spójnym stanie wewnętrznym.
- **ISOLATED** oznacza, że transakcja widzi bazę danych we wspomnianym powyżej stanie spójnym. Jeżeli dwie transakcje próbują zaktualizować tą samą tabelę najpierw zostanie wykonana pierwsza z nich a potem druga.
- **DURABILITY** oznacza, że wyniki transakcji są stale przechowywane w systemie.

TRANSAKCJE

Transakcja składa się zawsze z 3 etapów: rozpoczęcia, wykonania i zamknięcia.

Istotne jest, aby transakcja trwała jak najkrócej, ponieważ równolegle może być dokonywanych wiele transakcji i część operacji musi zostać wykonana w pewnej kolejności.

Każdy etap transakcji jest logowany, dzięki czemu w razie awarii systemu (dzięki zawartości logów) można odtworzyć stan bazy danych sprzed transakcji, która nie została zamknięta.

Część systemów baz danych umożliwia używanie punktów pośrednich (ang. save point), są to zapamiętane w systemie etapy transakcji, do których w razie wystąpienia błędu można się wycofać, bez konieczności anulowania wszystkich wykonanych działań.

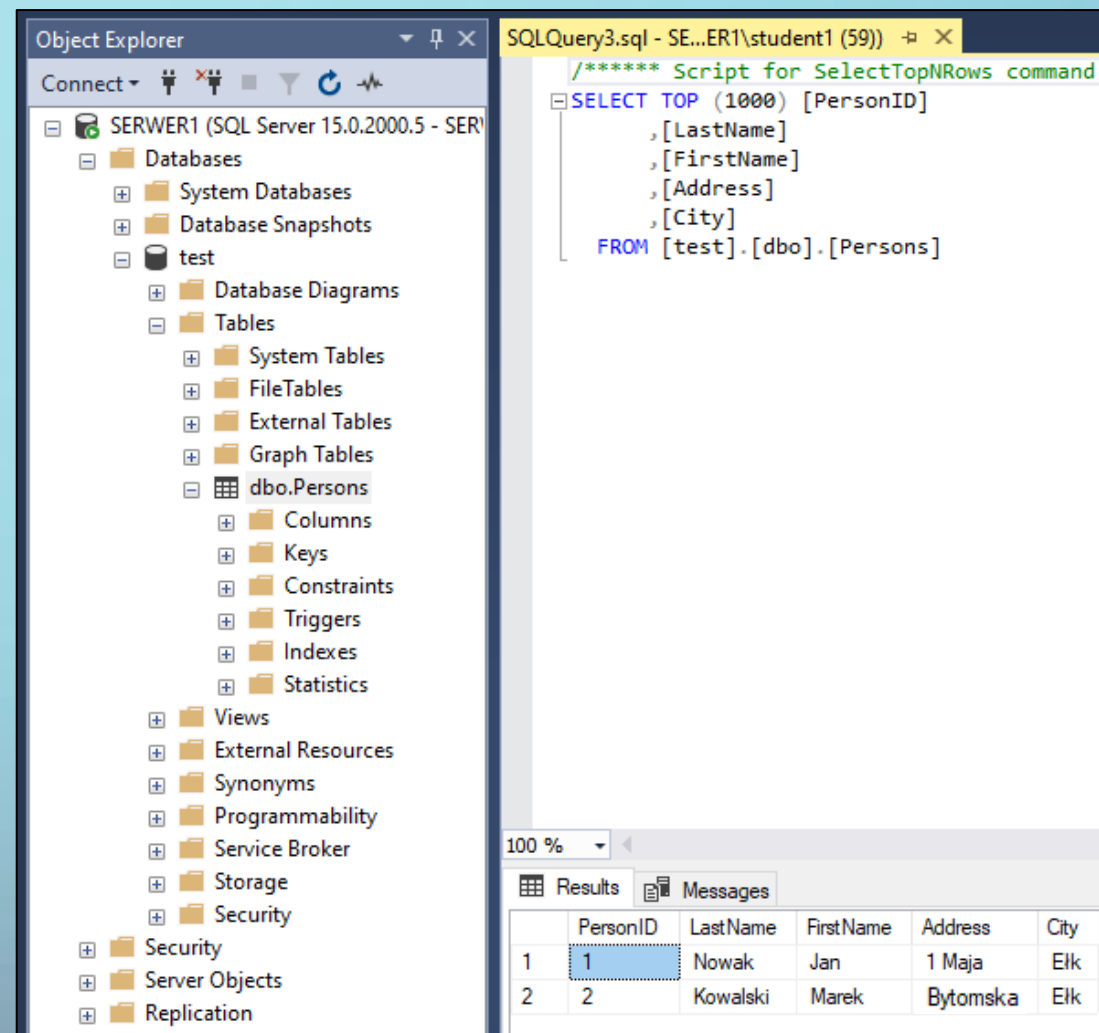
TRANSAKCJE

W systemach baz danych realizujących standard SQL następujące polecenia dotyczą transakcji:

- `BEGIN` lub `BEGIN WORK` – rozpoczęcie transakcji,
- `COMMIT` – zatwierdzenie zmian wykonanych w obrębie transakcji,
- `ROLLBACK` – odrzucenie zmian wykonanych w obrębie transakcji,
- `SAVEPOINT nazwa` – zdefiniowanie punktu pośredniego o określonej nazwie,
- `RELEASE SAVEPOINT nazwa` – skasowanie punktu pośredniego (nie wpływa w żaden sposób na stan transakcji),
- `ROLLBACK TO SAVEPOINT nazwa` – wycofanie transakcji do stanu zapamiętanego w podanym punkcie pośrednim.

TRANSAKCJE

Aby transakcje były bardziej przydatne należy przeprowadzać w nich dwie lub więcej operacji.

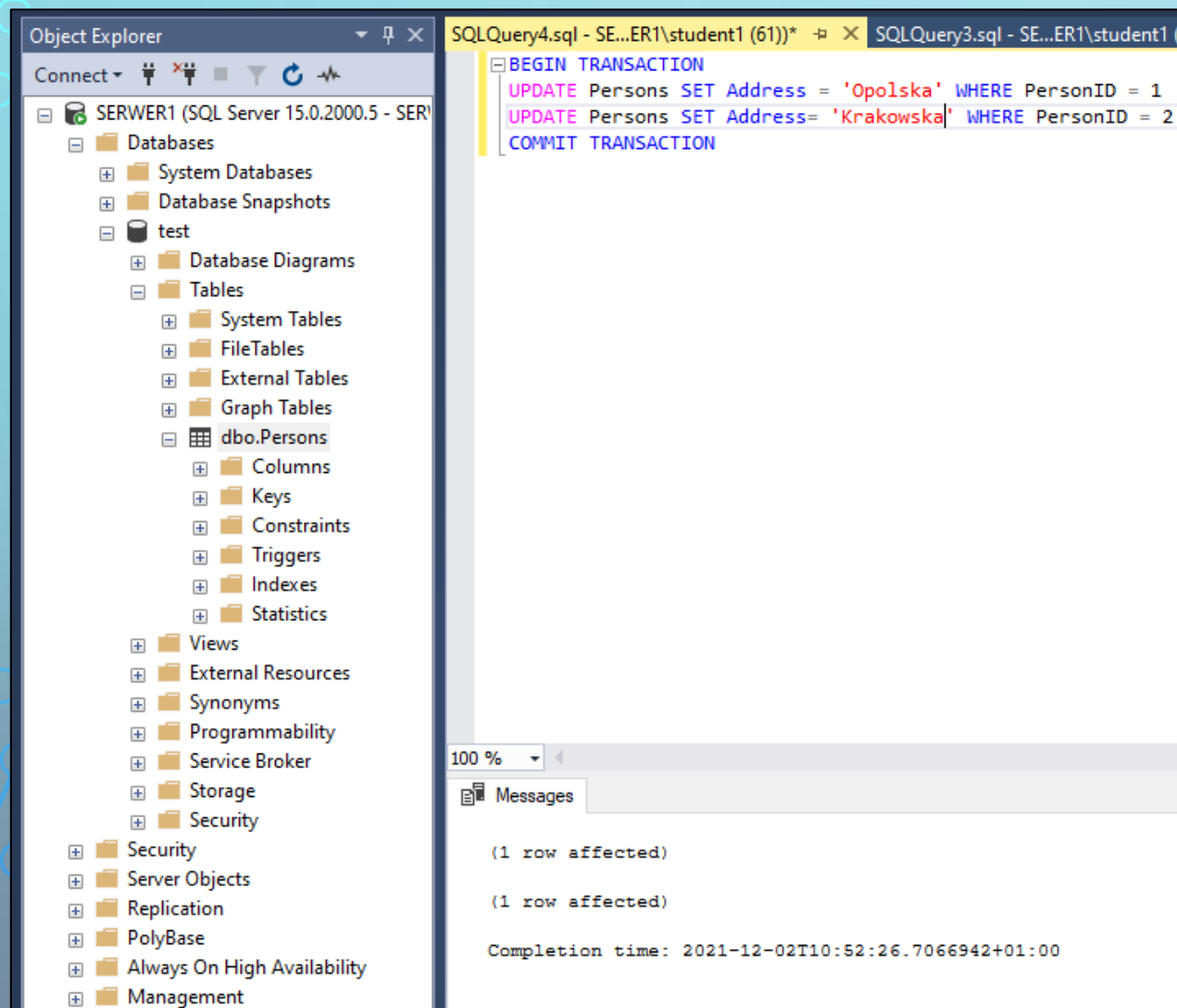


The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Object Explorer' pane shows the server hierarchy for 'SERWER1 (SQL Server 15.0.2000.5 - SER1)'. The 'test' database is expanded, showing its 'dbo' schema and the 'Persons' table. On the right, the 'SQLQuery3.sql' window shows a query script for the 'SelectTopNRows' command. The query selects the top 1000 rows from the 'Persons' table, ordered by 'PersonID'. The results are displayed in a table at the bottom right.

```
/****** Script for SelectTopNRows command *****  
SELECT TOP (1000) [PersonID]  
      , [LastName]  
      , [FirstName]  
      , [Address]  
      , [City]  
FROM [test].[dbo].[Persons]
```

	PersonID	LastName	FirstName	Address	City
1	1	Nowak	Jan	1 Maja	Elk
2	2	Kowalski	Marek	Bytomska	Elk

TRANSAKCJE

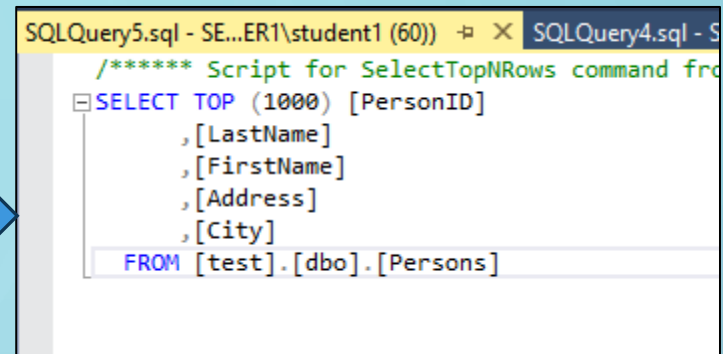


The screenshot shows the SQL Server Enterprise Manager interface on the left, with the 'test' database selected. The 'dbo.Persons' table is highlighted. On the right, the SQL Query window displays a transaction script:

```
BEGIN TRANSACTION
UPDATE Persons SET Address = 'Opolska' WHERE PersonID = 1
UPDATE Persons SET Address = 'Krakowska' WHERE PersonID = 2
COMMIT TRANSACTION
```

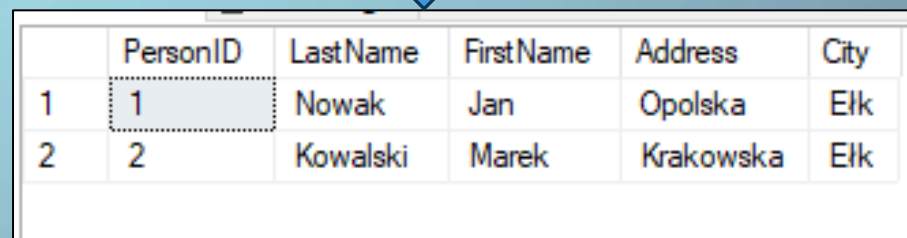
Below the script, the Messages pane shows the execution results:

```
(1 row affected)
(1 row affected)
Completion time: 2021-12-02T10:52:26.7066942+01:00
```



The SQL Query window displays a script for the 'SelectTopNRows' command:

```
SELECT TOP (1000) [PersonID]
, [LastName]
, [FirstName]
, [Address]
, [City]
FROM [test].[dbo].[Persons]
```



	PersonID	LastName	FirstName	Address	City
1	1	Nowak	Jan	Opolska	Elk
2	2	Kowalski	Marek	Krakowska	Elk

TRANSAKCJE

COMMIT – Zatwierdza wykonane polecenia transakcji.

```
BEGIN TRANSACTION
INSERT Persons (City,FirstName, LastName, Address) VALUES ('X', 'X', 'X','X')
SELECT * FROM Persons
IF @@ROWCOUNT = 5
    ROLLBACK TRANSACTION
ELSE
    BEGIN
        PRINT 'INFO'
        COMMIT TRANSACTION
    END
```

	PersonID	LastName	FirstName	Address	City
1	2	Nowak	Jan	1 Maja	Nysa
2	4	Nowak	Jan	Bytomska	Wałbrzych
3	7	Mucha	Anna	ul. Szkolna	Opole
4	32	Lewicki	Franciszek	Arłamów	Brzeg
5	33	X	X	X	X

Object Explorer

Connect

SERWER1 (SQL Server 15.0.2000.5 - SER)

- Databases
 - System Databases
 - Database Snapshots
 - test
 - Database Diagrams
 - Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.Persons
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Service Broker

SQLQuery20.sql - S...ER1\student1 (51))

```
/****** Script for SelectTopNRows command from S
```

```
SELECT TOP (1000) [PersonID]
      ,[LastName]
      ,[FirstName]
      ,[Address]
      ,[City]
FROM [test].[dbo].[Persons]
```

100 %

Results Messages

	PersonID	LastName	FirstName	Address	City
1	2	Nowak	Jan	1 Maja	Nysa
2	4	Nowak	Jan	Bytomska	Wałbrzych
3	7	Mucha	Anna	ul. Szkolna	Opole
4	32	Lewicki	Franciszek	Arłamów	Brzeg

Results Messages

	PersonID	LastName	FirstName	Address	City
1	2	Nowak	Jan	1 Maja	Nysa
2	4	Nowak	Jan	Bytomska	Wałbrzych
3	7	Mucha	Anna	ul. Szkolna	Opole
4	32	Lewicki	Franciszek	Arłamów	Brzeg

TRANSAKCJE

ROLLBACK – Cofa wykonane operacje w momencie w którym nie wszystkie polecenia transakcji zostały wykonane poprawnie.

SQLQuery14.sql - S...ER1\student1 (62)) SQLQuery13.s

```
/****** Script for SelectTopNRows command *****  
SELECT TOP (1000) [PersonID]  
    , [LastName]  
    , [FirstName]  
    , [Address]  
    , [City]  
FROM [test].[dbo].[Persons]
```

100 %

Results Messages

	PersonID	LastName	FirstName	Address	City
1	2	Nowak	Jan	1 Maja	Nysa
2	4	Nowak	Jan	Bytomska	Nysa

Object Explorer

Connect

SERWER1 (SQL Server 15.0.2000.5 - SER)

- Databases
 - System Databases
 - Database Snapshots
 - test
 - Database Diagrams
 - Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.PPersons
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Service Broker
 - Storage
 - Security
 - Security
 - Server Objects
 - Replication

SQLQuery14.sql - S...ER1\student1 (62)) SQLQuery13.sql - S...ER1\student

```
BEGIN TRANSACTION  
INSERT INTO Persons (FirstName, LastName, Address, City)  
VALUES ('Marek', 'Jeż', 'Krucza', 'Gliwice')  
ROLLBACK TRANSACTION
```

100 %

Messages

(1 row affected)

Completion time: 2021-12-02T12:02:36.2041817+01:00

DZIĘKUJĘ ZA UWAGĘ