UNIWERSYTET IM. ADAMA MICKIEWICZA W POZNANIU

Wprowadzenie do projektowania i wykorzystania baz danych

Katarzyna Klessa

POWTÓRKA Z PIERWSZYCH ZAJĘĆ

Lista słówek - do zapamiętania na początek

Z podstaw SQL:

CREATE - Tworzenie tabeli, czyli *Coś czego zwykle nie będę musiał/a robić ręcznie*
INSERT - Dodanie czegoś do tabeli
SELECT - Wyświetlenie
UPDATE - Zmodyfikowanie czegoś w tabeli
DELETE - Usunięcie z tabeli, czyli ***Ratunku, nie ma cofnij!***
DROP - Usunięcie tabeli, czyli *Ratunku, wyrzuciłem/am wszystko!*

Transakcje

Transakcje służą do zabezpieczenia bazy danych. Rozpatruje się sytuacje, gdy kilka osób pracuje jednocześnie na tych samych danych lub dodawane są dane w kilku miejscach i są one ze sobą powiązane, a więc w celu zapewnienia ich spójności zależy nam na tym, aby zmieniały się w sposób kontrolowany.

Transakcje - dwie osoby zmieniają te same dane

Gdy jedna osoba zmienia dane w tabeli, a druga osoba na drugim komputerze używa tej tabeli do wyliczeń czy innych operacji może powstać sytuacja, że druga osoba pobierze dane, które były jeszcze w trakcie modyfikacji. Na przykład został dodany wiersz dla nowego studenta, ale nie miał jeszcze wpisanej średniej, ponieważ program osoby wprowadzającej studentów właśnie tę średnią przeliczał. Użytkownik drugi pobrał dane studenta, widząc że on istnieje w bazie, ale nie wiedział, że wiersz nie jest wypełniony w całości. Transakcje umożliwiają zabezpieczenie danych i gwarantują, że modyfikowane dane staną się dostępne dopiero po całkowitym zakończeniu ich wprowadzania.

Transakcje - jedna osoba zmienia dane w dwóch miejscach

Ta sytuacja powstaje, gdy trzeba zapisać dane w kilku tabelach, np. wprowadzamy dane osobowe studenta i jego oceny do osobnych tabel. Chodzi nam konkretnie o sytuacje, gdy wszystkie dane muszą znaleźć się w bazie jednocześnie i chcemy wykluczyć możliwość tego, że dane osobowe będą już w bazie, a oceny dopiero za 3 milisekundy. W tym krótkim czasie może się zdarzyć, że program “poleci w kosmos” i oceny nie zostaną wprowadzone lub będą wprowadzone częściowo. Co więcej użytkownik nie będzie miał pojęcia czy oceny zostały umieszczone w bazie czy nie. Transakcje gwarantują, że zmiany w bazie danych pojawią się albo wszystkie albo wcale.

Transakcje - polecenia

BEGIN TRAN – rozpoczyna transakcję

COMMIT – potwierdza i kończy transakcję

ROLLBACK – anuluje wszystkie zmiany i kończy transakcję

Transakcje – uwaga praktyczna

UWAGA: Czas pomiędzy BEGIN TRAN a COMMIT powinien być jak najkrótszy, ponieważ jeśli zmieniliśmy coś w jakiejś tabeli, będzie ona niedostępna dla innego użytkownika do czasu wykonania przez nas COMMIT. Nie powinno się więc w praktyce wywoływać samego BEGIN TRAN, a potem wpisywać inne zapytania i dopiero po jakimś czasie COMMIT. **Zapytanie zawierające transakcje powinno być wykonywane jako jeden skrypt, w całości.**

Transakcje

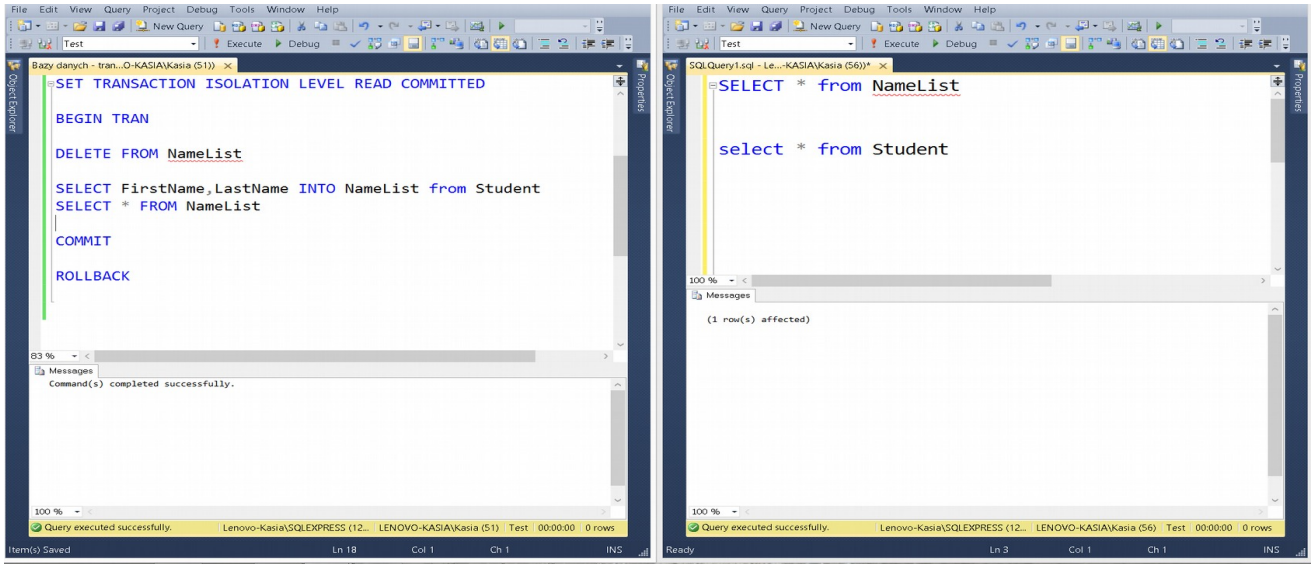
```
BEGIN TRAN
-- tu wpisujemy zapytanie/a np. Z UPDATE,
-- INSERT...

-- następnie ALBO zatwierdzamy (COMMIT)

COMMIT

-- albo wycofujemy (ROLLBACK) zmiany

ROLLBACK
```

Transakcje

Różne poziomy zabezpieczenia danych, często stosowane to READ COMMITTED – pozwala na odczyt danych już zatwierdzonych (przez użycie COMMIT)

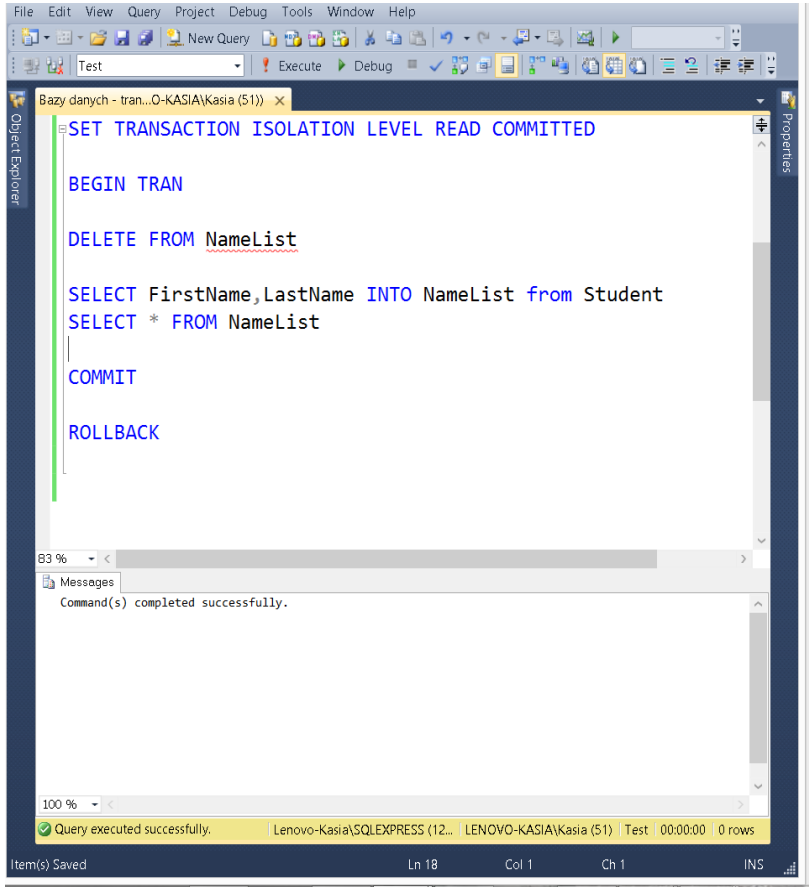
```
--SET TRANSACTION ISOLATION LEVEL
--    { READ UNCOMMITTED
--    | READ COMMITTED
--    | REPEATABLE READ
--    | SNAPSHOT
--    | SERIALIZABLE
--    }
--[ ; ]
```

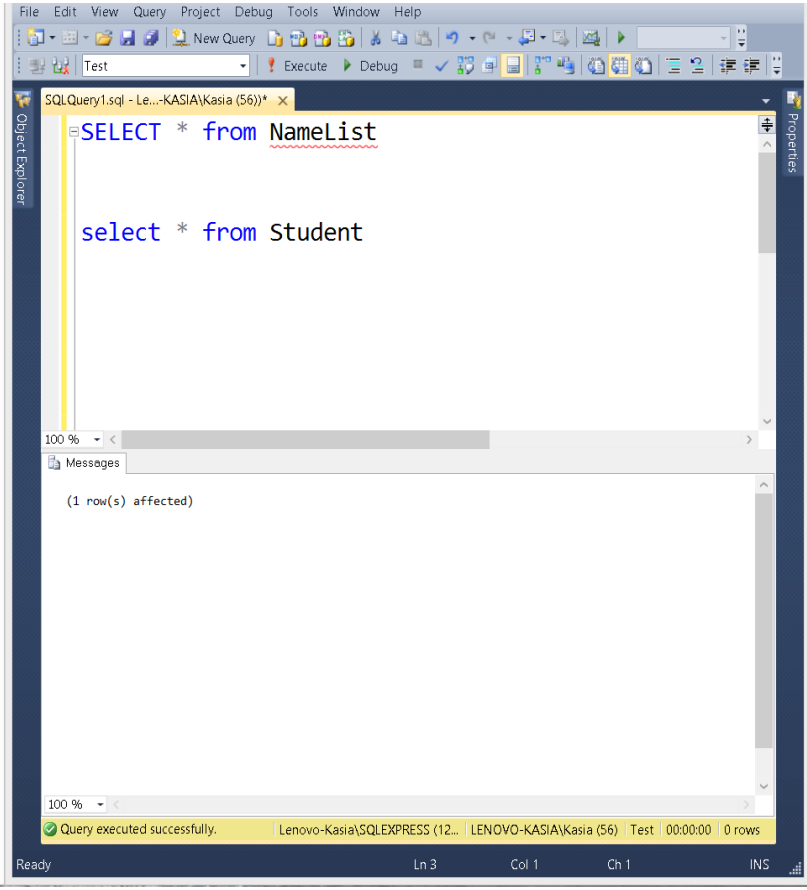
Transakcje – testy robocze



Transakcje można przetestować, otwierając SSMS w dwóch oknach. Pamiętajmy jednak, że w praktyce, zwłaszcza współpracując z innymi ludźmi tak się tego nie robi, ponieważ jak już wspomniano wyżej, powinny one trwać jak najrócej, aby nie blokować bazy danych (powinny być **wykonywane jako jeden skrypt, w całości**).

Transakcje

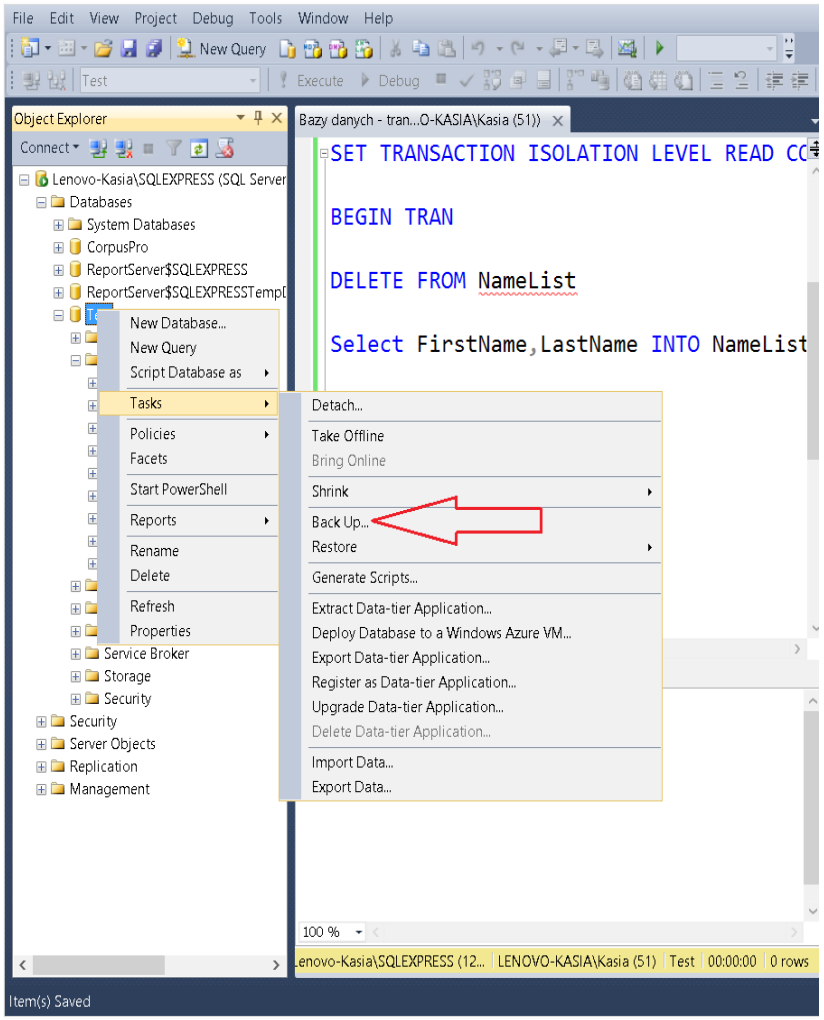

File Edit View Query Project Debug Tools Window Help
New Query
Test
Execute Debug
Bazy danych - tran...O-KASIA\Kasia (51))
SET TRANSACTION ISOLATION LEVEL READ COMMITTED
BEGIN TRAN
DELETE FROM NameList
SELECT FirstName,LastName INTO NameList from Student
SELECT * FROM NameList
COMMIT
ROLLBACK
Messages
Command(s) completed successfully.
Query executed successfully.
Item(s) Saved


File Edit View Query Project Debug Tools Window Help
New Query
Test
Execute Debug
SQLQuery1.sql - Le...-KASIA\Kasia (56))*
SELECT * from NameList
select * from Student
Messages
(1 row(s) affected)
Query executed successfully.
Ready

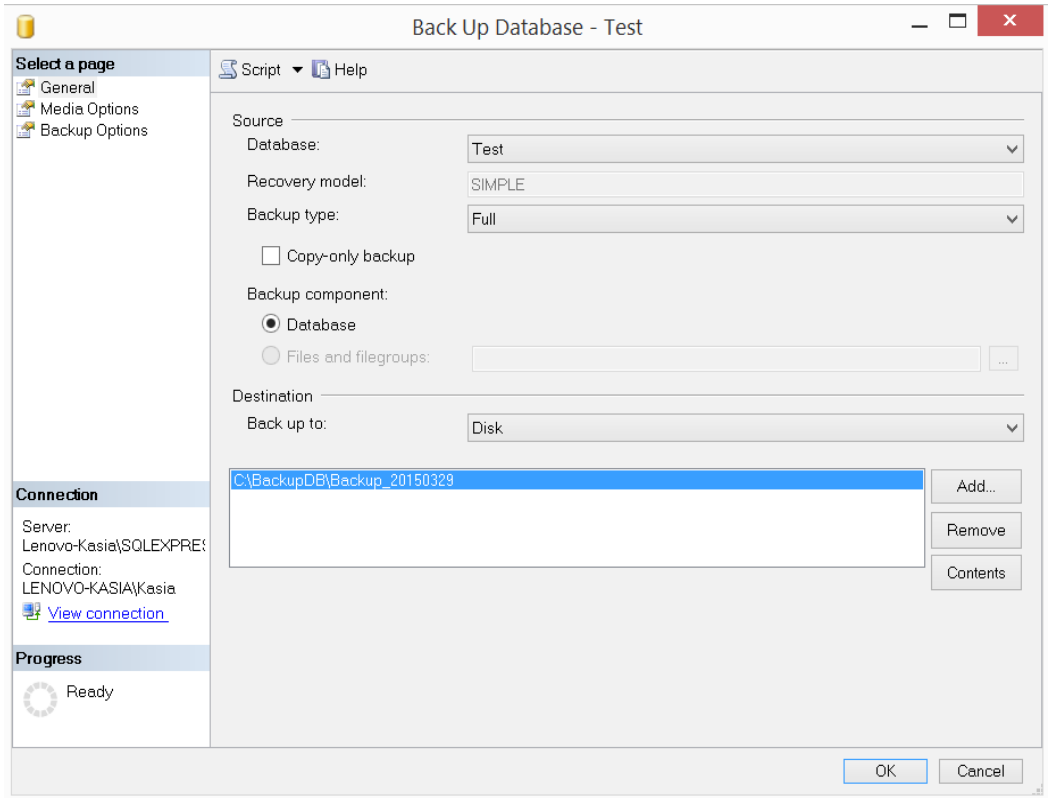
Kopie bezpieczeństwa w SSMS

Ratunku, nie ma cofnij!

Mimo, że w SQL istnieją pewne mechanizmy zabezpieczające, takie jak transakcje, to najbezpieczniejszym założeniem w pracy z bazami danych jest założenie, że z zasady raczej nie cofa się operacji. Dlatego tym bardziej ważnym nawykiem w pracy z wszystkimi rodzajami baz jest wykonywanie kopii bezpieczeństwa danych przed operacjami modyfikującymi dane. W SSMS backup możemy wykonać, klikając prawym klawiszem na nazwie bazy, następnie Tasks->Back Up



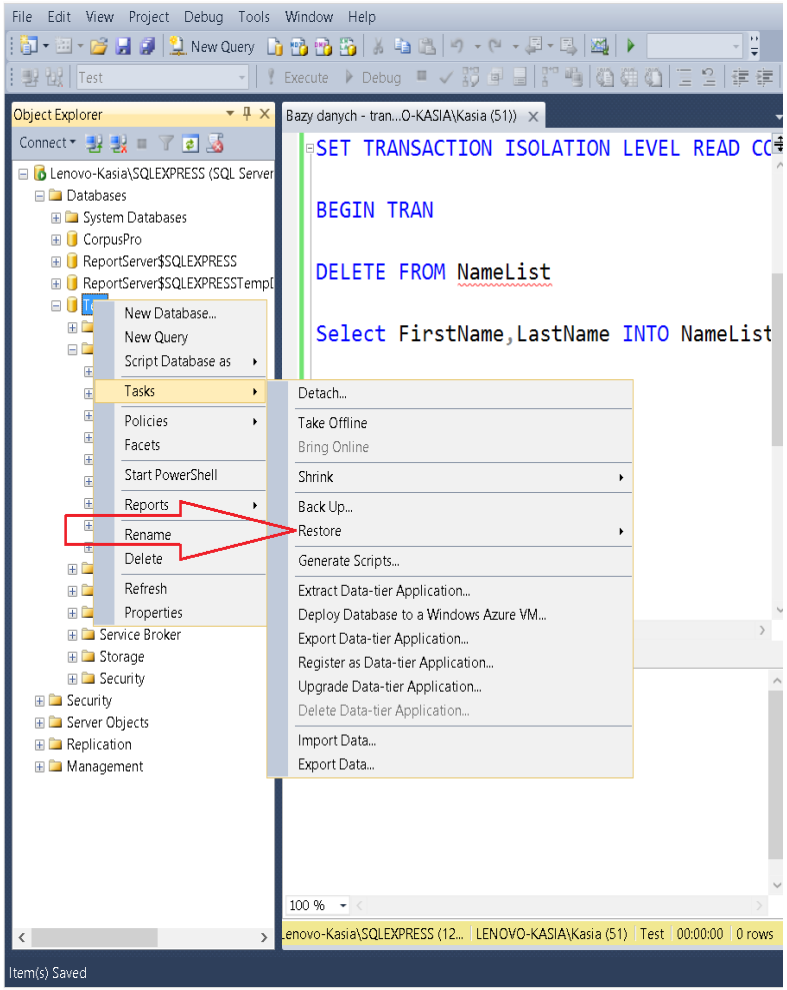
Kopie bezpieczeństwa w SSMS



Backup może mieć różne cechy, m.in. może być pełen (full) lub różnicowy (differential – zapisywana jest tylko różnica w stosunku do poprzedniego backupu). Przed zapisem możemy wybrać lokalizację dla plików z kopiami bezpieczeństwa.

Kopie bezpieczeństwa w SSMS

W SSMS dane z kopii bezpieczeństwa możemy przywrócić klikając prawym klawiszem na nazwie bazy, następnie Tasks->Restore i wskazując odpowiedni plik kopii bezpieczeństwa.



Dziękuję za uwagę!