



Wprowadzenie do projektowania i wykorzystania baz danych

Relacje

Katarzyna Klessa



Dygresja nt. operatorów

SELECT 2²

SELECT 2³⁰

SELECT 50⁵⁰



Dygresja nt. operatorów

SELECT 2^30

--Bitwise exclusive OR (XOR, alternatywa wykluczająca, więcej): porównanie liczb w syst. dwójkowym, w wyniku wart. 1 wpisywana jest na tej pozycji, na której porównywane liczby się różniły, w pozostałych przypadkach 0, np. Dla 2^30:

--00010 - w syst. dziesiętnym: 2

--11110 - w syst. Dziesiętnym: 30

--11100 - w syst. dziesiętnym: 28

Zadanie: Oblicz, a potem sprawdź w SSMS:

SELECT 4^9

SELECT 2^2



Dygresja nt. operatorów

```
SELECT 2^2
```

```
--00010 - w syst. dziesiętnym: 2
```

```
--00010 - w syst. Dziesiętnym: 2
```

```
-----
```

```
--00000 - w syst. dziesiętnym: 0 (wszystkie pozycje  
identyczne wartości)
```

```
SELECT 4^9
```

```
-- 0100 - w syst. dziesiętnym: 4
```

```
-- 1001 - w syst. dziesiętnym: 9
```

```
-----
```

```
-- 1101 - w syst. dziesiętnym: 13
```



Dygresja nt. operatorów

W niektórych językach programowania znak ^ może oznaczać podniesienie do potęgi. W SQL jest to operator XOR, jak w przykładach wyżej.

Podniesienie do potęgi, pierwiastek w SQL:

```
SELECT POWER(2, 3)
```

```
SELECT SQRT(2)
```



Różne sposoby zabezpieczania danych i operacji w SSMS



Różne sposoby zabezpieczania danych i operacji w SSMS - “wstępny SELECT”

Prostym sposobem może być w wielu przypadkach kontrolny “wstępny **SELECT**”, który wykonujemy zanim wykonamy zapytanie modyfikujące dane – możemy dzięki temu wstępnie podejrzeć co stanie się z danymi po naszej operacji, np. ile wierszy zostanie zmodyfikowanych, w jaki sposób itp.



Różne sposoby zabezpieczania danych i operacji w SSMS - transakcje

BEGIN TRAN

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

-- następnie ALBO zatwierdzamy (COMMIT)

COMMIT

-- albo wycofujemy (ROLLBACK) zmiany

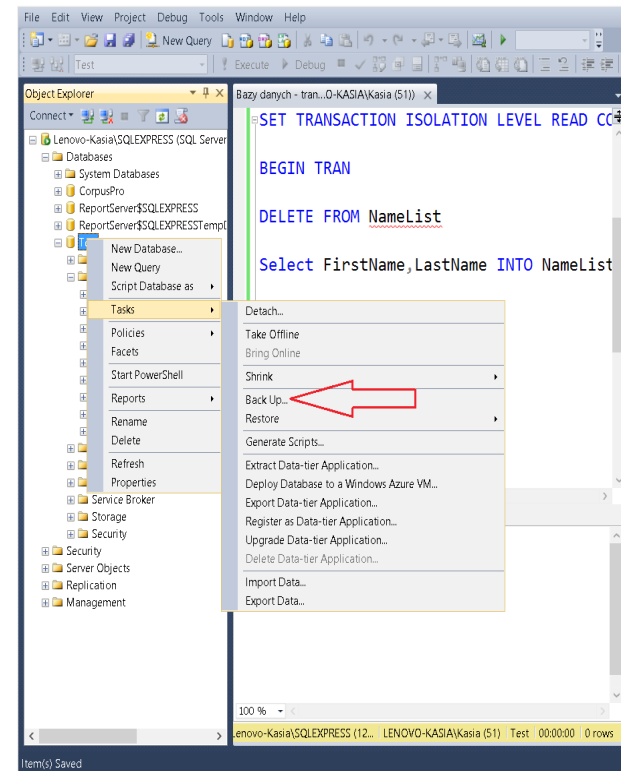
ROLLBACK



Różne sposoby zabezpieczania danych i operacji w SSMS - kopie bezpieczeństwa

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





RELACJE - składnia

```
ALTER TABLE Table1 WITH CHECK ADD CONSTRAINT  
RelationName  
FOREIGN KEY(ForeignKeyColumn)  
REFERENCES Table2 (KeyColumn)
```



RELACJE - składnia

```
ALTER TABLE Table1 WITH CHECK ADD CONSTRAINT  
RelationName  
FOREIGN KEY(ForeignKeyColumn)  
REFERENCES Table2 (KeyColumn)
```

Zapytanie modyfikuje tabelę Table1 (ALTER TABLE) ze sprawdzeniem wartości (WITH CHECK), dodaje ograniczenie (ADD CONSTRAINT) o nazwie RelationName na klucz obcy ForeignKeyColumn (FOREIGN KEY). Niech wskazuje na tabelę Table2, kolumnę KeyColumn (REFERENCES). Tworzenie relacji “ręcznie” za pomocą takiego zapytania to jeden sposób, drugi, wizualny zostanie omówiony w dalszej części wykładu.



RELACJE - opis

Relacje są jednym z mechanizmów ograniczeń (tzw. *constraints*) stosowanych w serwerach baz danych.

Relacja powstaje przez połączenie dwóch tabel. Łączymy klucz główny (*Primary Key*) w jednej tabeli z kluczem obcym (*Foreign Key*) w drugiej.

Relacje “pilnują”, aby nie można było przypadkowo usunąć danych użytych w innym miejscu. Można więc uznać je za kolejny sposób zabezpieczania danych w bazie.



RELACJE - przykład

Aby przyjrzeć się jak działają relacje, przydadzą nam się dwie tabele. Jak pamiętamy, tabele można stworzyć na kilka sposobów, np. za pomocą **SELECT INTO** na podstawie innej tabeli lub **CREATE**. Tu stworzymy 2 nowe tabele: Citizen i City.

--stworzenie tabeli Citizen

```
CREATE Table Citizen (  
  CitizenID int identity(1,1) not null Primary Key,  
  --stworzenie klucza głównego, int, samonumeracja,  
  nie może być pusty, klucz główny  
  FirstName varchar(50),  
  LastName varchar(50),  
  CityID int  
)
```



RELACJE - przykład

```
--stworzenie tabeli City  
CREATE Table City (  
CityID int identity(1,1) not null Primary Key,  
Name varchar(50),  
CitizenID int  
)
```



RELACJE - przykład

--nałożenie relacji

```
ALTER TABLE Citizen WITH CHECK ADD CONSTRAINT  
FK_Citizen_City  
FOREIGN KEY(CityID)  
REFERENCES City (CityID)
```

Zapytanie modyfikuje tabelę Citizen (ALTER TABLE) ze sprawdzeniem wartości (WITH CHECK), dodaje ograniczenie o nazwie FK_Citizen_City na klucz obcy CityID (ADD CONSTRAINT ... FOREIGN KEY). Niech wskazuje na tabelę City, kolumnę CityID (REFERENCES).



RELACJE - przykład

Dzięki relacjom dane są zabezpieczone przed przypadkową modyfikacją. Jeśli na tabele (np. Citizen i City, w których są dane osób z Warszawy i Poznania) jest założona relacja i spróbujemy usunąć wiersz z jednej z nich (np. wiersze dot. miasta Warszawy), to SSMS wyświetli błąd, np. taki:

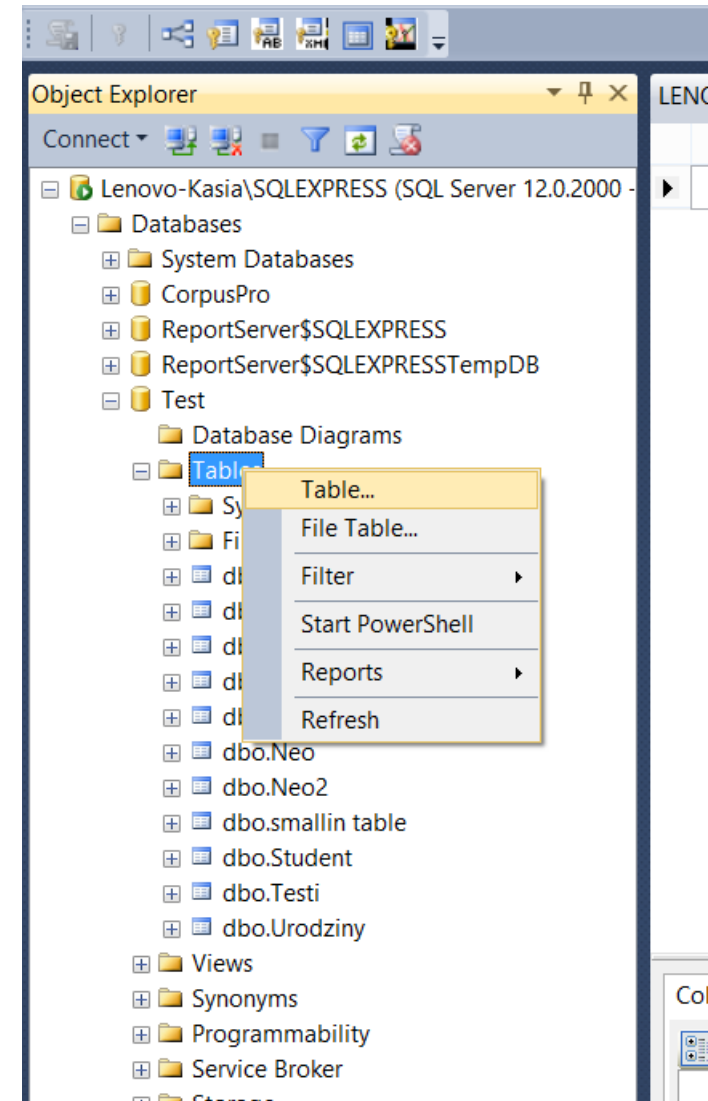
```
Msg 547, Level 16, State 0, Line 9
The DELETE statement conflicted with the REFERENCE constraint
"FK_Citizen_City". The conflict occurred in database "Test", table
"dbo.Citizen", column 'CityID'.
The statement has been terminated.
```

Aby było możliwe usunięcie danych dla miasta, musimy najpierw usunąć dane jego mieszkańców – musimy więc zrobić to świadomie, relacja zabezpiecza przed przypadkowym usuwaniem danych, które są powiązane relacjami z innymi danymi w bazie.



RELACJE - przykład

Tabele w SSMS można stworzyć także używając dostępnych tam **narzędzi graficznych** (choć nie wszyscy użytkownicy je preferują). Taki sam efekt jak w wyniku dwóch powyższych zapytań **CREATE** otrzymamy klikając prawym klawiszem na Tables po lewej stronie okna programu. P





RELACJE - przykład

Tabele – tworzenie za pomocą narzędzi graficznych

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. On the left, the 'Object Explorer' shows the database structure for 'Lenovo-Kasia\SQLEXPRESS (SQL Server 12.0.2000)'. The 'Tables' folder is expanded, showing various tables like 'dbo.Citizen', 'dbo.City', etc. The main window shows the 'Table Designer' for 'dbo.Table_1'. The 'Column Properties' tab is active, showing the properties for the 'CitizenID' column. The 'Data Type' is 'int', and the 'Identity Specification' is set to '(Is Identity)'. A red circle highlights the 'CitizenID' column in the table designer.

Column Name	Data Type	Allow Nulls
CitizenID	int	<input type="checkbox"/>

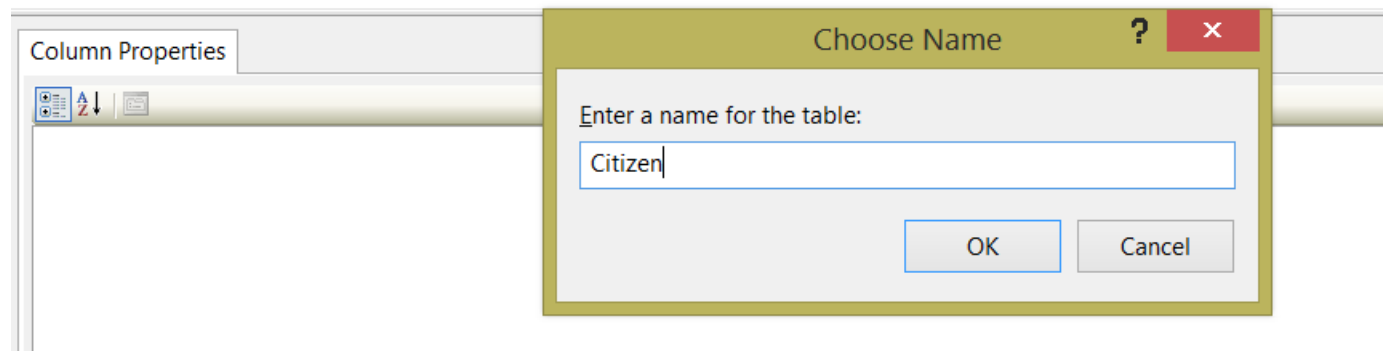
Property	Value
Data Type	int
Default Value or Binding	
Table Designer	
Collation	<database d
Computed Column Specification	
Condensed Data Type	int
Description	
Deterministic	Yes
DTS-published	No
Full-text Specification	No
Has Non-SQL Server Subscriber	No
Identity Specification	Yes
(Is Identity)	Yes
Identity Increment	1
Identity Seed	1
Indexable	Yes
Is Columnset	No
Is Sparse	No



RELACJE - przykład

Tabele – tworzenie za pomocą **narzędzi graficznych**. Dodajemy informacje dla kolejnych kolumn i zapisujemy tabelę.

	Column Name	Data Type	Allow Nulls
🔑	CitizenID	int	<input type="checkbox"/>
	FirstName	varchar(50)	<input type="checkbox"/>
	LastName	varchar(50)	<input type="checkbox"/>
	CityID	int	<input type="checkbox"/>
▶			<input type="checkbox"/>

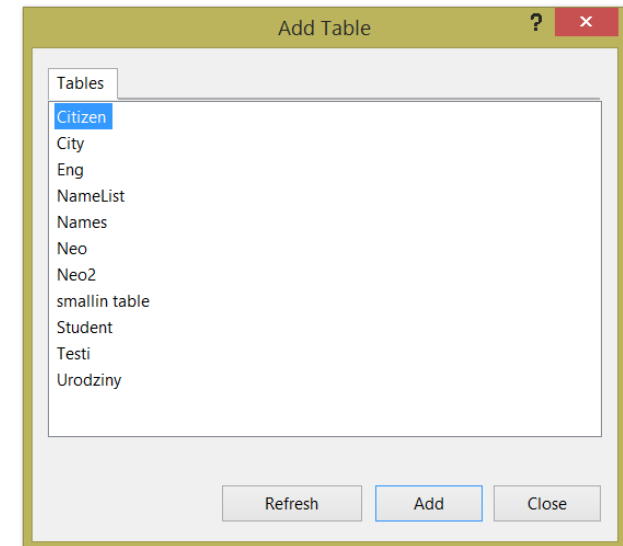
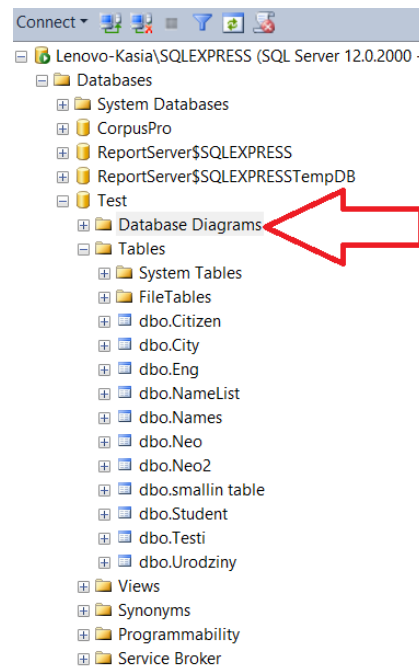




RELACJE - przykład

Edycja i podgląd wizualny relacji – diagramy.

Diagramy ERD (*Entity Relationship Diagrams*) – diagramy związków encji.

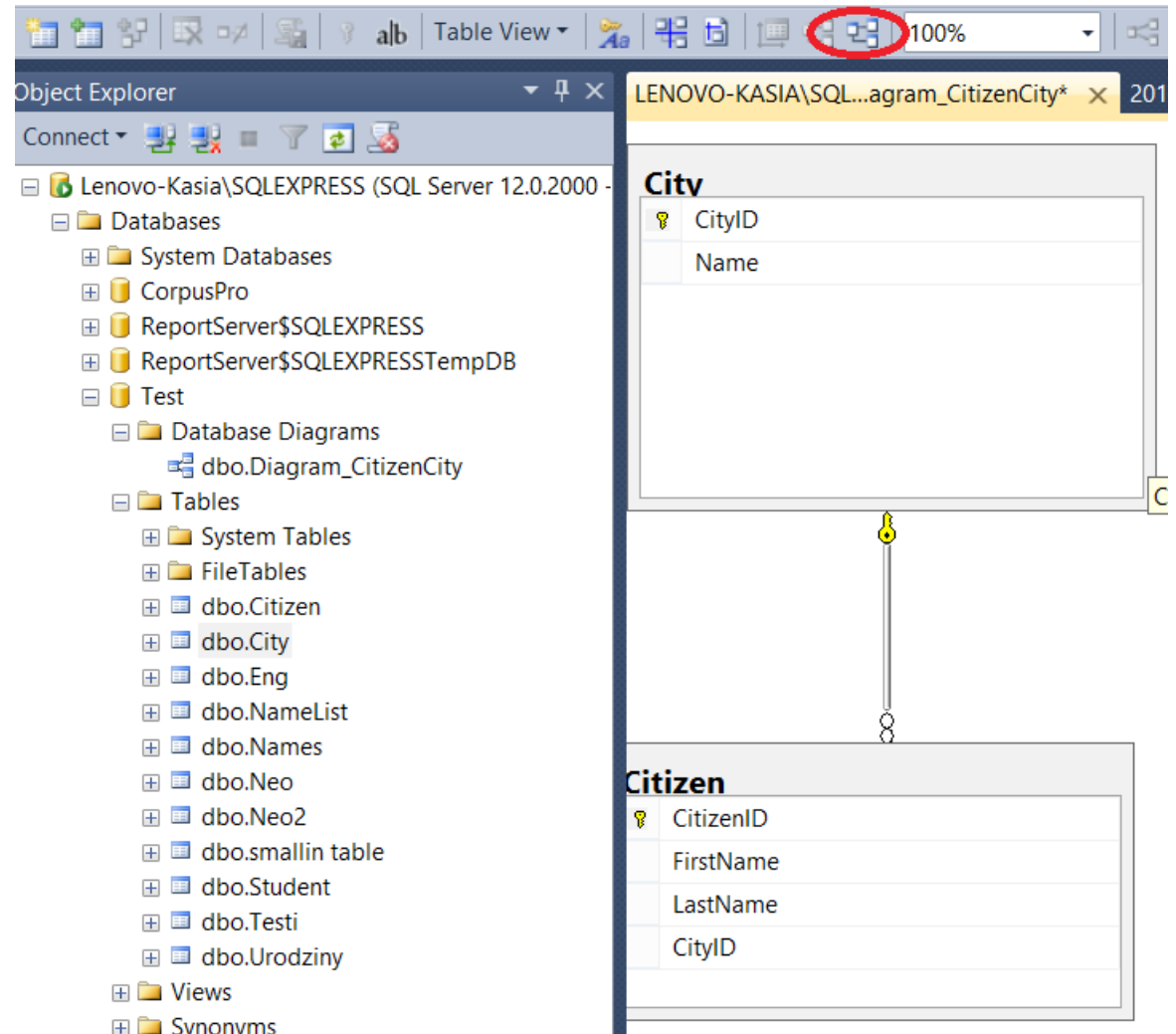




RELACJE - przykład

Edycja i podgląd
wizualny relacji –
diagramy

Relację pokazuje
nam linia łącząca
tabele w widoku
diagramu. Tę linię
możemy sami
wstawić (na różne
sposoby, zależnie
od rodzaju relacji)
albo usunąć.





RELACJE - przykład

Podgląd zapytań, np.

Script Table as...
m.in. wyświetli
komendę nakładania
relacji o składni jak
na slajdzie 9 wyżej

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Server Object Explorer' displays the 'Test' database structure, including 'Tables' and 'dbo.Citizen'. The 'dbo.Citizen' table is selected, and a context menu is open over it. The menu options include 'Table...', 'Design', 'Select Top 1000 Rows', 'Edit Top 200 Rows', 'Script Table as', 'View Dependencies', 'Full-Text index', 'Policies', 'Facets', 'Start PowerShell', 'Reports', 'Rename', 'Delete', 'Refresh', and 'Properties'. The 'Script Table as' option is expanded, showing a sub-menu with 'CREATE To', 'ALTER To', 'DROP To', 'DROP And CREATE To', 'SELECT To', 'INSERT To', 'UPDATE To', 'DELETE To', and 'EXECUTE To'. The 'CREATE To' option is also expanded, showing a further sub-menu with 'New Query Editor Window', 'File ...', 'Clipboard', and 'Agent Job ...'. On the right, the 'Table Designer' for 'Citizen' shows columns: CitizenID (primary key), FirstName, LastName, and CityID. Below it, the 'Table Designer' for 'City' shows columns: CityID (primary key) and Name. A relationship line connects the CityID column in the Citizen table to the CityID column in the City table.



RELACJE - typy

- Relacja jeden do wielu
- Relacje wiele-do-wielu
- Relacje jeden do jednego

Cdn.



Dziękuję za uwagę!

