SQL PODSTAWY

SQL jest językiem manipulacji danych - **D**ata **M**anipulation **L**anguage (w skrócie - **DML**). Zawiera w sobie takie operacje jak `INSERT`, `SELECT` lub `UPDATE`.

1.1 Podstawowe funkcje w T-SQL

W tym skrypcie tworzymy tabelę, aby zademonstrować kilka podstawowych funkcji.

```
-- Tworzenie tabeli "HelloWorld"
CREATE TABLE HelloWorld (
Id INT IDENTITY, -- Id = nazwa kolumny, int = typ kolumny
Description VARCHAR(1000)
)

-- DML Operation INSERT, wstawianie wiersza do tabeli
INSERT INTO HelloWorld (Description) VALUES ('Hello World')

-- DML Operation SELECT, wyświetlanie tabeli
SELECT * FROM HelloWorld

-- Wyświetlanie wybranej kolumny z tabeli
SELECT Description FROM HelloWorld

-- Wyświetlanie ilości rekordów (wierszy) w tabeli
SELECT Count(*) FROM HelloWorld

-- DML Operation UPDATE, aktualizowanie wybranej kolumny w tabeli
UPDATE HelloWorld SET Description = 'Hello, World!' WHERE Id = 1

-- Wyświetlanie tabeli (widzisz jak zmieniła się wartość w kolumnie
Description po aktualizacji?)
SELECT * FROM HelloWorld

-- DML Operation - DELETE, usuwanie kolumny z tabeli
DELETE FROM HelloWorld WHERE Id = 1

--Wyświetlanie tabeli. Widoczne zmiany w tabeli po operacji DELETE
SELECT * FROM HelloWorld
```

1.2 Tworzenie zapytań

Następujące przykłady pokazują jak wykonywać zapytania tabeli:

```
USE Northwind;
GO
SELECT TOP 10 * FROM Customers
ORDER BY CompanyName
```

To zapytanie wyświetli pierwsze 10 rekordów z tabeli `Customers`, sortowane według kolumny `CompanyName` z bazy danych `Northwind` (która jest jedną z przykładowych baz danych Microsoft, można ją pobrać tutaj).

CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Region	PostalCode	Country	Phone	Fax
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	null	12209	Germany	030-0074321	030-0076545
ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner	Avda. de la Constitución 2222	México D.F.	null	05021	Mexico	(5) 555-4729	(5) 555-3745
ANTON	Antonio Moreno Taquería	Antonio Moreno	Owner	Mataderos 2312	México D.F.	null	05023	Mexico	(5) 555-3932	null
AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London	null	WA1 1DP	UK	(171) 555-7788	(171) 555-6750
BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator	Berguvsvägen 8	Luleå	null	S-958 22	Sweden	0921-12 34 65	0921-12 34 67
BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative	Forsterstr. 57	Mannheim	null	68306	Germany	0621-08460	0621-08924
BLONP	Blondesddsl père et fils	Frédérique Citeaux	Marketing Manager	24, place Kléber	Strasbourg	null	67000	France	88.60.15.31	88.60.15.32
BOLID	Bólido Comidas preparadas	Martín Sommer	Owner	C/ Araquil, 67	Madrid	null	28023	Spain	(91) 555 22 82	(91) 555 91 99
BONAP	Bon app'	Laurence Lebihan	Owner	12, rue des Bouchers	Marseille	null	13008	France	91.24.45.40	91.24.45.41
BOTTM	Bottom-Dollar Markets	Elizabeth Lincoln	Accounting Manager	23 Tsawassen Blvd.	Tsawassen	BC	T2F 8M4	Canada	(604) 555-4729	(604) 555-3745

Zauważ, że Use Northwind; zmienia domyślną bazę danych dla wszystkich następnych zapytań. Możesz wciąż odwołać się do bazy danych używając pełnej kwalifikowanej składni w formie [Database].[Schema].[Table]:

```
SELECT TOP 10 * FROM Northwind.dbo.Customers
ORDER BY CompanyName
SELECT TOP 10 * FROM Pubs.dbo.Authors
ORDER BY City
```

Jest to użyteczne gdy wykonujesz zapytanie z różnych baz danych.
Zauważ, że **dbo** podane “pomiędzy” jest nazywane schematem i musi być określone podczas używania pełnej kwalifikowanej składni.
Możesz myśleć o schemacie jak o folderze zawierającym Twoją bazę danych.
dbo jest domyślnym schematem. Domyślny schemat może być pominięty. Wszystkie inne zdefiniowane przez użytkownika schematy muszą
być określone.

1.3 Podstawowe funkcje T-SQL - rozwinięcie

1.3.1 CREATE TABLE

```
CREATE TABLE NazwaTabeli (
NazwaKolumny1 INT,
<Nazwa kolumny2> <typ kolumny2>,
...
)
```

Ilość kolumn w tabeli jest nieograniczona. Mimo wszystko, staraj się nie tworzyć tabel mających dużą ilość kolumn, gdyż nie zawsze jest to wydajne rozwiązanie.

1.3.2 INSERT

```
INSERT INTO NazwaTabeli (NazwaKolumny1) VALUES (10)
```

Podczas wstawiania danych do tabeli nie musisz wpisywać wartości wszystkich kolumn (chyba że każda z kolumn wymaga podania jakiejś wartości i tabela nie posiada klucza głównego numerującego wiersze).

1.3.3 SELECT

Wyświetlanie wszystkich kolumn z tabeli:

```
SELECT *
FROM table_name
```

Korzystanie z operatora gwiazdki * służy jako skrót do wybierania wszystkich kolumn w tabeli. Będą również wybrane wszystkie wiersze, ponieważ ta instrukcja `SELECT` nie zawiera klauzuli `WHERE`, aby określić dowolne kryteria filtrowania.

UWAGA: użycie `SELECT *` w kodzie produkcyjnym lub przechowywanych procedur może prowadzić do późniejszych problemów(gdy nowe kolumny są dodawane do tabeli lub jeśli kolumny są uporządkowane w tabeli), szczególnie jeśli twój kod zawiera proste założenia o kolejności kolumn lub liczbie zwróconych kolumn. Bezpieczniej jest więc zawsze jawnie określać nazwy kolumn w instrukcjach SELECT dla kodu produkcyjnego.

1.3.4 UPDATE

```
UPDATE HelloWorlds
SET HelloWorld = 'HELLO WORLD!!!'
WHERE Id = 5
```

Powyższy kod aktualizuje wartość pola "HelloWorld" na "HELLO WORLD !!!" dla rekordu, w którym "Id = 5" w tabeli HelloWorlds.

UWAGA: Podczas aktualizacji zaleca się użycie klauzuli "where", aby uniknąć aktualizacji całej tabeli, chyba że tak chcesz zrobić.

JOIN

Joiny są przydatne, jeśli chcesz wysyłać zapytania do pól, które nie istnieją w jednej tabeli, ale w wielu tabelach. Na przykład:
Chcesz zapytać o wszystkie kolumny z tabeli `Region` w bazie danych `Northwind`. Ale zauważyłeś, że do tego potrzebujesz również `RegionDescription`, który jest przechowywany w innej tabeli, `Region`. Istnieje jednak wspólny klucz, `RegionID` które można wykorzystać do połączenia tych informacji w jednym zapytaniu w następujący sposób(`TOP 5` wyświetli tylko pierwsze 5 wierszy, pomiń to, jeśli chcesz wyświetlić wszystkie wiersze):

```
SELECT TOP 5 Territories.*,
Regions.RegionDescription
FROM Territories
INNER JOIN Region
ON Territories.RegionID=Region.RegionID
ORDER BY TerritoryDescription
```

To zapytanie wyświetli wszystkie kolumny z `Territories` plus kolumnę `RegionDescription` z `Region`.

ALIAS

Gdy zapytanie wymaga odniesienia do dwóch lub więcej tabel, przydatne może być użycie Aliasu Tablicy. Aliasy tabel są skrótowymi odwołaniami do tabel, które mogą być używane zamiast pełnej nazwy tabeli i mogą skracać pisanie i edytowanie. Składnia do używania aliasu to:

```
<TableName> [as] <alias>
```

“As” jest opcjonalnym słowem kluczowym. Na przykład, poprzednie zapytanie może być zapisane w ten sposób:

```
SELECT TOP 5 t.*,
r.RegionDescription
FROM Territories t
INNER JOIN Region r
```

```
ON t.RegionID = r.RegionID
ORDER BY TerritoryDescription
```

Aliasy muszą być unikalne dla wszystkich tabel w zapytaniu, nawet jeśli używasz tej samej tabeli dwa razy. Na przykład, jeśli Twoja tabela `Employee` zawiera pole `SupervisorId`, możesz użyć tego zapytania, aby zwrócić pracownika i jego opiekuna:

```
SELECT e.*,
s.Name as SupervisorName -- Zmiana nazwy pola dla wyniku zapytania
FROM Employee e
INNER JOIN Employee s
ON e.SupervisorId = s.EmployeeId
WHERE e.EmployeeId = 111
```

https://docs.microsoft.com/en-us/previous-versions/sql/sql-server-2008-r2/ms143221(v=sql.105)

```
Tylu ludzi nie wie co robić w życiu, a to jest w sumie takie proste.
Wystarczy iść za własną ciekawością.
~K. Gonciarz
```