Jerzy STANKIEWICZ[1]

PODSTAWOWE I ZŁOŻONE POLECENIA SQL W IMPLEMENTACJACH MS SQL SERVER 2008

Streszczenie

W artykule w oparciu o wcześniej utworzoną bazę danych zostały sformułowane przykładowe zapytania SQL-owe w implementacji MS SQL Server 2008. Pokazano różne możliwości pisania poleceń manipulujących danymi w bazach. Zaprezentowano zapytania generujące wyniki w formatach relacyjnych i nierelacyjnych i omówiono przykłady zadań nietypowych, pozornie bardzo trudnych i złożonych oraz niebanalnych rozwiązań wykorzystujących możliwości języka SQL.

Abstract

The article presents examples of SQL queries in the implementation of MS SQL Server 2008, basing on a database created beforehand. It presents different possibilities of creating commands to manipulate data in the databases. It presents queries generating results in relational and non-relational formats. It also presents not typical tasks, with complex and original solutions using SQL language.

1. WPROWADZENIE

SQL (ang. *Structured Query Language* – strukturalny język zapytań) to uniwersalny język stosowany w większości systemów zarządzania bazami danych. Zaliczany jest do tak zwanych języków deklaratywnych (języki czwartej generacji 4GL) zorientowanych na wynik.

W językach tego typu użytkownik definiuje, jaki efekt końcowy chce osiągnąć, w wyniku działania wybranego polecenia, bez określania, w jaki sposób należy to wykonać.

[1] Mgr inż. Jerzy Stankiewicz jest wykładowcą w Warszawskiej Wyższej Szkole Informatyki.

Model relacyjny systemu zarządzania bazą danych został zaproponowany w 1970 przez E. F. Codda z IBM Research Laboratory w San Jose w Kalifornii.

Polecenia języka Język SQL można podzielić na trzy główne grupy:

- **DML** (ang. *Data Manipulation Language)* – Język Manipulacji Danymi – część języka SQL umożliwiająca dokonywanie operacji na danych takich jak wstawianie wiersza do tabeli, modyfikowanie istniejących wierszy, usuwanie wierszy oraz pobieranie danych (zapytania). W skład DML wchodzą polecenia: INSERT (wstaw wiersze do tabeli), UPDATE (zmodyfikuj wiersze w tabeli), DELETE (usuń wiersze z tabeli) i SELECT(pobierz dane)
- **DDL** – (ang. *Data Definition Language*) – Język Definiowania Danych – część języka umożliwiająca definiowanie struktur bazy danych (tabele, widoki, procedury, indeksy i wiele, wiele innych obiektów bazy danych) oraz ich modyfikację. W skład DDL wchodzą następujące polecenia: CREATE (zdefiniuj obiekt bazy danych), DROP (usuń obiekt z bazy danych) i ALTER (zmień definicję istniejącego obiektu)
- **DCL** (ang. *Data Control Language)* – Język Kontroli Danych – część języka SQL umożliwiająca sterowanie prawami dostępu do danych. W skład DCL wchodzą polecenia GRANT (przydziel prawo) i REVOKE (pozbawienie przydzielonego prawa)

1.1. Model relacyjny

- Informacja w relacyjnej bazie danych jest reprezentowana przez wartości zapisane w tabelach. W systemie relacyjnym **tabele** mają (poziome) **wiersze** i (pionowe) **kolumny.**
- Pojęcia tabela, wiersz, kolumna są pojęciami powszechnie używanymi w komercyjnych systemach zarządzania relacyjnymi bazami danych. Można w literaturze spotkać takie pojęcia jak **relacja, krotka (encja) i atrybut** są synonimami tabeli, wiersza i kolumny (podobnie jak plik, rekord, pole).
- Zbiór powiązanych tabel tworzy bazę danych.
- Każdy wiersz tabeli opisuje pojedyncze wystąpienie encji (osoby, przedsiębiorstwa, sprzedaży).
- Każda kolumna opisuje pewną charakterystyczną cechę encji np. nazwisko, imię, adres.
- SQL jest stosowany głównie do operowania danymi: wyszukiwania danych i ich modyfikowania.

- Modyfikowanie danych oznacza ich dodawanie, usuwanie lub modyfikację.
- Operacje wyszukiwania danych (często zwane **zapytaniami**) polegają na przeszukaniu bazy danych, pobraniu żądanej informacji i wizualne jej przedstawienie.
- We wszystkich zapytaniach SQL używa się słowa kluczowego **SELECT.**

1.2. Podstawowa składnia SELECT

SELECT Lista kolumn
FROM Źródło Listy
[**WHERE** Warunek wybierający]
[**GROUP BY** Warunek grupowania]
[**HAVING** Szczegóły grupowania]
[**ORDER BY** Wyrażenie] [ASC | DESC]
* Pogrubione są słowa kluczowe

Źródło listy – w przypadku odwołania do:

- pojedynczej tabeli podajemy nazwę tabeli;
- więcej niż jednej tabeli używa się poniższą składnię:

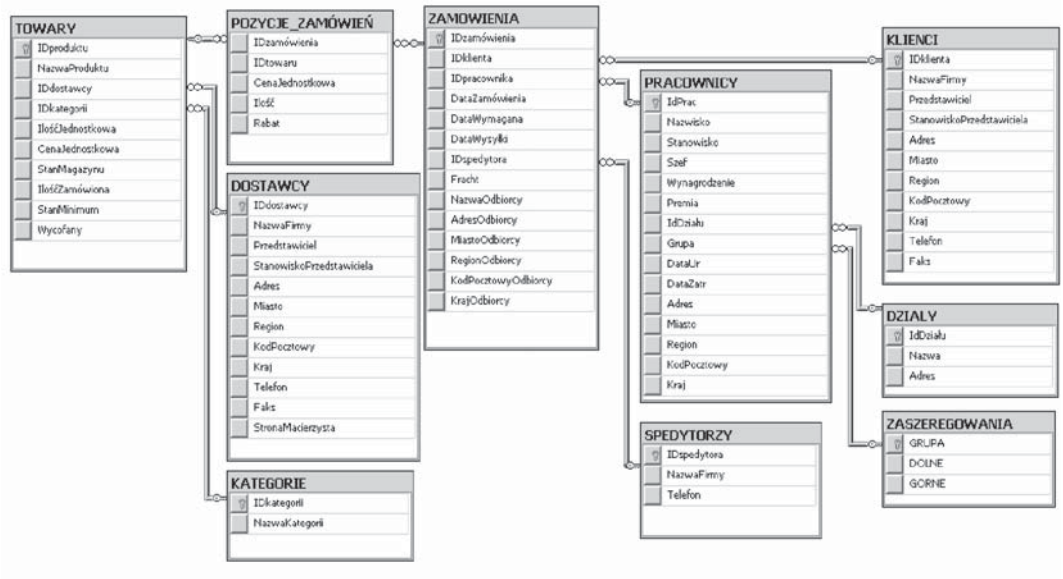
FROM <tabela>
<operator_zlaczenia> <tabela>
ON <warunek zlaczenia>
<operator_zlaczenia> <tabela>
ON <warunek zlaczenia>

Gdzie <operator złącza> to JOIN (INNER JOIN), LEFT OUTER JOIN, RIGHT OUTER JOIN

1.3. Przykładowa baza danych

Do omówienia poleceń realizujących zapytania wykorzystano przedstawioną poniżej przykładową bazę danych. Zaprezentowana w formie diagramu baza zawiera tabele, dzięki którym można ewidencjonować zamówienia klientów w sklepie. W bazie można wyróżnić szereg tabel zwanych tabelami słownikowymi. Są nimi między innymi tabele Towary, Klienci, Pracownicy, Dostawcy wykorzystane w dalszej części do prezentacji zapytań.

Elektroniczny Sklep



1.4. Możliwości polecenia SQL (SELECT)

Instrukcja SELECT jest z wielu względów „motorem" SQL. Umożliwia wyszukanie i oglądanie danych na wiele sposobów. Dane w bazie relacyjnej są zapisywane w postaci dwuwymiarowych tabel i wynik zapytania też jest (wyjątek opisana dalej klauzula COMPUTE) dwuwymiarową tabelą. W praktyce każde zapytanie jest wypadkową trzech operacji opisanych poniżej:

- Projekcja (kolumna lub kolumny – po słowie kluczowym SELECT, które mają być włączone do wyników)
- Selekcja (warunek wybierania – po słowie kluczowym WHERE)
- Łączenie (złączenie tabel w celu otrzymania pełnej analizy danych w oparciu o klucz obcy jednej tabeli i klucz podstawowy tabeli dołączanej)

1.5. Wybieranie wierszy – klauzura WHERE

Wykonując instrukcję SELECT zawierającą klauzulę WHERE podaje się warunki wyszukiwania – określa się, jaki podzbiór wierszy ma zostać zwrócony. SQL dostarcza szereg operatorów i słów kluczowych służących do formułowania warunków wyszukiwania. Należą do nich:

- Operatory logiczne

- Kombinacje lub logiczne negacje warunków
- Przedziały
- Listy
- Wartości nieznane
- Zgodność znakowa

Operatory logiczne
Logiczne operatory: { = | <> | != | > | >= | !> | < | <= | !< }
= równy
> większy niż
< mniejszy niż
>= większy lub równy
<= mniejszy lub równy
<>różny od

Przykłady
nazwisko = ‘Kowalski’, wiek < 25, Data = ‘11-JUL-1996‘, Data > ‘1996-07-11’

Przedziały
BETWEEN i NOT BETWEEN należy używać do określenia przedziału domkniętego, w którym poszukuje się wartości najmniejszej, największej i wszystkich wartości zawartych między nimi.

Przykłady
WHERE wiek BETWEEN 30 and 40
WHERE Data BETWEEN ‘11-JUL-1996’ AND ‘18-JUL-1996’

Listy
Słowo kluczowe IN i NOT IN umożliwia wybranie wartości pokrywającej się z którąkolwiek wymienioną na liście wartości.

Przykłady
WHERE litera IN (‘A’, ‘B’, ‘C’)
lub inaczej
WHERE litera = ‘A’ OR litera = ‘B’ OR LITERA = ‘C’
WHERE liczba IN (10, 20, 30)

Wartości nieznane

NULL i NOT NULL – Wartość NULL oznacza brak danych. Uwaga! Wartość NULL jest czymś innym niż 0 w przypadku liczb czy też pusty string w przypadku znaków.

Przykłady

WHERE Fax. IS NULL

Zgodność znakowa

Niektóre problemy nie można rozwiązać na drodze dokładnego porównania (np. znany jest częściowo kontekst znakowy). Dlatego zaleca się używania między innymi operatorów LIKE, LEFT czy RIGHT opisanych poniżej.

Operator LIKE

- % – zastępuje dowolne znaki (od zera do wielu znaków)
  np WHERE imie LIKE '%ar%' wszystkie osoby o imieniu gdzie są litery „ar"
- _ (podkreślenie) (dowolny, pojedynczy znak)
  np WHERE slowo LIKE '_an' 3 znakowe słowa np Jan Pan itp.
- [ ] wiele pojedynczych znaków z listy [a-f] LUB [abcdef]
  np. WHERE nazwisko LIKE '[C-P]arsen' znajdzie np Carsen, Larsen
- [^] wiele pojedynczych znaków nie z listy [a-f] lub [abcdef]
  np. WHERE nazwisko LIKE '[^C-P]arsen' znajdzie np Barsen, Sarsen

Operator LEFT, RIGHT

Przykłady:

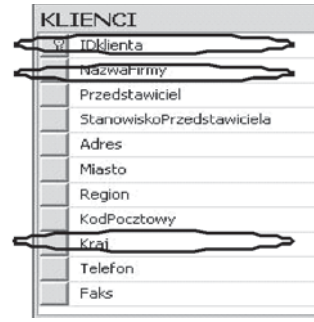
```
WHERE (LEFT(nazwisko,1)='K')
WHERE (RIGHT(RTRIM(IMIE),1)='A')
```

Co oznacza: Od prawej (z pominięciem spacji) pierwszy znak to 'A'

2. POLECENIE SELECT – DOSTĘP DO DANYCH Z JEDNEJ TABELI – PROSTY WARUNEK WYBORU

W rozdziale tym zaprezentowano kilka prostych zapytań kierowanych do jednej tabeli gdzie w wyniku realizowana jest operacja projekcji, selekcji oraz kombinacja tych operacji

- **Podać pełne dane (wszystkie kolumny) o wszystkich klientach**

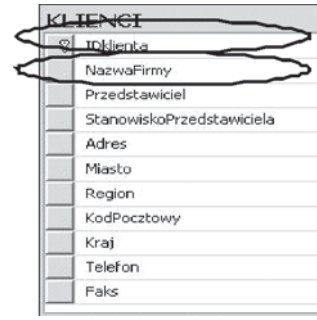



```
SELECT    *
FROM      klienci
```

- **Podać wykaz wszystkich klientów:**
  dla każdego klienta podać identyfikator nazwę firmy i kraj.

```
SELECT    Idklienta, NazwaFirmy, Kraj
FROM      klienci
```

- **Podać wykaz klientów z Krakowa:**
  dla każdego klienta podać jego identyfikator oraz nazwę firmy w której pracuje.




```
SELECT    IDklienta, NazwaFirmy
FROM      Klienci
WHERE     Miasto = 'Kraków'
```

- **Podać wykaz wszystkich towarów kategorii RYBY (idkategorii=8).**
  Dla każdego towaru podać IDproduktu, Nazwę produktu i IDdostawcy.

```
SELECT    IDproduktu, NazwaProduktu, IDdostawcy
FROM      Towary
WHERE     IDkategorii = 8
```

- **Podać wykaz dostawców z krajów różnych od Australii.**
  Dla każdego dostawcy podać nazwę firmy, miasto i kraj.

```
SELECT    Nazwafirmy, Miasto, Kraj
FROM      Dostawcy
WHERE     Kraj <> 'Australia'
```

- **Podać wykaz zamówień o wartości przekraczającej 500 złotych.**
  Dla każdego takiego zamówienia podać jego numer, identyfikator klienta i wartość zamówienia.

```
SELECT    IDzamówienia, Idklienta, Fracht
FROM      Zamówienia
WHERE     Fracht > 500
```

- **Podać wykaz wszystkich zamówień złożonych między 11/07/1996 a 12/07/1996. Dla każdego zamówienia podać numer, datę i wartość**

```
SELECT    IDzamówienia, DataZamówienia, Fracht
FROM      Zamowienia
WHERE     DataZamówienia BETWEEN '11-JUL-1996' AND '18-JUL-1996'
```

- **Podać wykaz krajów i miast, w których mieszczą się firmy dostawców.**

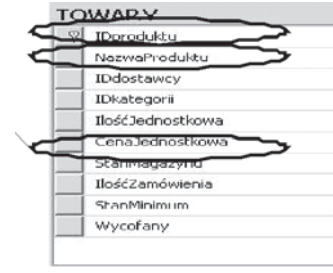
```
SELECT DISTINCT Miasto, Kraj
FROM      Dostawcy
```

* *Wybór wierszy spełniających zadany warunek i kolumn wskazanych nazwą*
* *Użycie słowa kluczowego DISTINCT w celu wyeliminowania powtórzeń*

3. POLECENIE SELECT – DOSTĘP DO DANYCH Z JEDNEJ TABELI – ZŁOŻONY WARUNEK WYBORU

W rozdziale tym zaprezentowano kilka zapytań ze złożonym warunkiem wyboru kierowanych do jednej tabeli gdzie w wyniku realizowana jest operacja projekcji, selekcji oraz kombinacja tych operacji

- **Podać wykaz towarów kategorii=1 od dostawcy=18. Dla każdego modelu podać numer katalogowy, nazwę produktu i cenę jednostkową**




```
SELECT    IDproduktu, NazwaProduktu, CenaJednostkowa
FROM      Towary
WHERE     IDkategorii = 1
   AND    IDdostawcy = 18
```

- **Podać wykaz wszystkich towarów od dostawcy 1 i 2.**
  Dla każdego artykułu podać nazwę produktu, iddostawcy i oznaczenie towaru(IDtowaru).

```
SELECT    IDproduktu, NazwaProduktu, IDdostawcy
FROM      Towary
WHERE     IDdostawcy = 1 OR IDdostawcy = 2
```

- **Podać wykaz wszystkich towarów o cenach mieszczących się w granicach od 10 do 21.50 złotych od dostawcy=1. Dla każdego modelu podać numer katalogowy, nazwę produktu i cenę.**

```
SELECT    IDtowaru, NazwaProduktu, CenaJednostkowa
FROM      Towary
WHERE     IDdostawcy= 1
  AND     CenaJednostkowa BETWEEN 10 AND 21.50
```

- **Podać wykaz wszystkich zamówień o wartości przekraczającej 200 złotych, złożonych między 15/04/1996 a 15/12/1996. Dla każdego zamówienia podać numer, datę i wartość.**

```
SELECT IDzamówienia, Datazamówienia, Fracht
FROM      Zamowienia
WHERE     Datazamówienia BETWEEN '15-APR-1996' AND '15-DEC-1996'
  AND     Fracht > 200
```

- **Podać wykaz towarów kategorii=2 od dostawcy 1 lub 2 lub 3, którego zapas w magazynie jest mniejszy niż 100 sztuk, podać IDtowaru, Nazwę IDdostawcy i stan magazynu**

```
SELECT    IDtowaru, NazwaProduktu, IDdostawcy, StanMagazynu
FROM      Towary
WHERE     IDkategorii= 2
  AND     (IDdostawcy = 1
  OR      IDdostawcy = 2
  OR      IDdostawcy = 3)
  AND     StanMagazynu < 100
```

- **Polecenie z zadania poprzedniego można zapisać w następującej postaci korzystając z operatora IN:**

```
SELECT    IDtowaru, NazwaProduktu, IDdostawcy, StanMagazynu
FROM      Towary
WHERE     IDkategorii= 2 AND
          IDdostawcy in (1,2,3) AND
          StanMagazynu < 100
```

Użycie operatora LIKE

- **Podać wykaz towarów kategorii=2 o oznaczeniu rozpoczynającym się od słowa "G". Dla każdego towaru podać IDproduktu i nazwę produktu.**

```
SELECT    IDproduktu, NazwaProduktu
FROM      Towary
WHERE     IDkategorii = 2
   AND    NazwaProduktu LIKE 'G%'
```

- **Podać wykaz towarów kategorii=2, takich że wartość zapasów danego towaru przekracza 500 złotych. Dla każdego towaru podać IDproduktu, nazwę produktu i IDdostawcy.**

```
SELECT    IDproduktu, NazwaProduktu, IDdostawcy
FROM      Towary
WHERE     IDkategorii = 2
   AND    StanMagazynu * CenaJednostkowa > 500
```

- **Podać wykaz zamówień przyjętych w okresie od 01/04/1996 do 12/04/1996, dla których czas oczekiwania przekroczył 15 dni. Dla każdej dostawy podać numer zamówienia i identyfikator klienta.**

```
SELECT    IDzamówienia, Idklienta
FROM      Zamowienia
WHERE     DATEDIFF(„dd",datazamówienia, datawysyłki)>15
   and    Datazamówienia BETWEEN '01-APR-1996' AND '12-APR-1996'
```

Uporządkowanie wierszy wyniku za pomocą specyfikacji ORDER BY

- **Podać wykaz klientów z Londynu Dla każdego klienta podać identyfikator, nazwę firmy i adres. Uporządkować alfabetycznie rosnąco wg nazwy firmy.**

```
SELECT    Idklienta, Nazwafirmy, Adres
FROM      Klienci
WHERE     Miasto = 'Londyn'
ORDER BY  Nazwafirmy ASC
```

Uporządkowanie wierszy malejąco

```
SELECT    Idklienta, Nazwafirmy, Adres
FROM      Klienci
WHERE     Miasto = 'Londyn'
ORDER BY  Nazwafirmy DESC
```

- **Podać wykaz wszystkich zamówień od klienta o identyfikatorze = 1, złożonych w okresie od 01/12/1996. Dla każdego zamówienia podać numer zamówienia, datę i wartość. Uporządkować wynik według wartości od największej do najmniejszej.**

```
SELECT    IDzamówienia, Datazamówienia, Fracht
FROM      Zamówienie
WHERE     IDklienta = 1
   AND    Datazamówienia >= '01-DEC-1996'
ORDER BY  Fracht DESC
```

- **Podać wykaz wszystkich dostawców. Dla każdego dostawcy podać nazwę firmy, miasto i kraj. Uporządkować alfabetycznie według krajów, a gdy kraj się powtarza – kolejnym kryterium sortowania jest miasto.**

```
SELECT    Nazwafirmy, Miasto, Kraj
FROM      Dostawcy
ORDER BY  Kraj, Miasto
```

Lub

```
SELECT    Nazwafirmy, Miasto, Kraj
FROM      Dostawcy
ORDER BY  2, 3
```

Dołączanie kolumn z wartościami wyliczanymi.

- **Podać wykaz wartości zapasów (cena jednostkowa * stan magazynu) dla poszczególnych towarów kategorii=2**

```
SELECT    IDproduktu, NazwaProduktu, Cenajednostkowa * StanMagazynu
FROM      Towary
WHERE     IDkategorii = 2
```

Wybranie wierszy w których wartość we wskazanej kolumnie jest określona/ nieokreślona

- **Podać wykaz klientów (identyfikator, nazwa firmy), którzy nie mają faksu.**

```
SELECT    Idklienta, NazwaFirmy
FROM      Klienci
WHERE     Faks IS NULL
```

4. POLECENIE SELECT – DOSTĘP DO DANYCH Z DWÓCH LUB WIĘCEJ TABEL

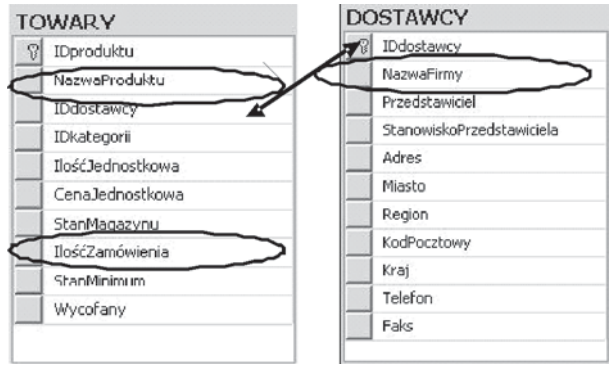
W rozdziale tym zaprezentowano kilka zapytań ze złożonym warunkiem wyboru kierowanych do więcej niż jednej tabeli gdzie w wyniku realizowana jest operacja łączenia, projekcji, selekcji oraz kombinacja tych operacji

- **Podać wykaz wszystkich zarejestrowanych zamówień. Dla każdego zamówienia podać: numer i datę zamówienia oraz nazwę firmy zamawiającej i miasto, w którym ta firma się znajduje.**

```
SELECT    Zamowienia.idzamówienia,
          Zamowienia.Datazamówienia,
          Klienci.Nazwafirmy,
          Klienci.Miasto
FROM      Zamowienia
   JOIN   Klienci
   ON     Zamowienia.IDklienta = Klienci.IDklienta
```

- **Podać wykaz wszystkich zarejestrowanych dostaw. Dla każdej dostawy podać nazwę towaru, zamówioną ilość towaru i nazwę firmy dostawcy**

```
SELECT    Towary.NazwaProduktu,
              Towary.IlośćZamówiona,
          Dostawcy.Nazwafirmy
FROM      Towary JOIN Dostawcy
   ON     Towary.IDdostawcy = Dostawcy.IDdostawcy
```



- **Podać wykaz zamówień o wartości przekraczającej 500 złotych. Dla każdego takiego zamówienia podać identyfikator klienta, nazwę firmy, numer zamówienia i wartość zamówienia.**

```
SELECT    Klienci.Idklienta,
          Klienci.Nazwafirmy,
          Zamowienia.idzamówienia,
          Zamowienia.fracht
FROM      Klienci JOIN Zamowienia
   ON     Klienci.Idklienta = Zamowienia.Idklienta
WHERE     Zamowienia.fracht > 500
```

- **Podać wykaz zamówień o wartości przekraczającej 100 złotych, złożonych przez klientów z Genewy. Dla każdego takiego zamówienia podać identyfikator klienta, nazwę firmy, numer zamówienia i wartość zamówienia.**

```
SELECT    Klienci.IDklienta,
          Klienci.Nazwafirmy,
          Zamowienia.IDzamówienia,
          Zamowienia.Fracht,
          Klienci.Miasto
   FROM   Klienci JOIN Zamowienia
```

```
ON      Klienci.IDklienta = Zamowienia.IDklienta
WHERE   Klienci.Miasto = 'Genewa'
AND     Zamowienia.Fracht > 100
```

Wiersze tabeli wynikowej powstają z par wierszy spełniających warunek złączenia z dodatkową selekcją

- **Podać wykaz produktów Cukier zamówionych u dostawców mających siedzibę w USA. Dla każdej dostawy podać nazwę firmy dostawcy, numer zamówienia, ilość zamówienia.**

```
SELECT    NazwaFirmy, Idzamówienia, NazwaProduktu, kraj, ilość
FROM      Pozycje_Zamówień
JOIN      Towary ON Pozycje_Zamówień.IDtowaru=Towary.IDproduktu
JOIN      Dostawcy ON Towary.iddostawcy = dostawcy.iddostawcy
WHERE     dostawcy.kraj = 'USA'
  AND     NazwaProduktu ='Cukier'
```

Złączenia zewnętrzne

- **Podać pełny wykaz klientów wraz z danymi opisującymi złożone przez nich zamówienia (numer i data). Zakładamy, że są zarejestrowani w bazie danych klienci, dla których nie ma zarejestrowanych zamówień.**

```
SELECT    Klienci.Idklienta,
          Klienci.Nazwafirmy,
          Zamowienia.IDzamówienia,
          Zamowienia.DataZamówienia,
          miasto
FROM      Klienci
  LEFT OUTER JOIN Zamowienia
ON        Klienci.Idklienta = Zamowienia.Idklienta
```

- **Pokaż klientów, którzy nie dokonali żadnej transakcji**

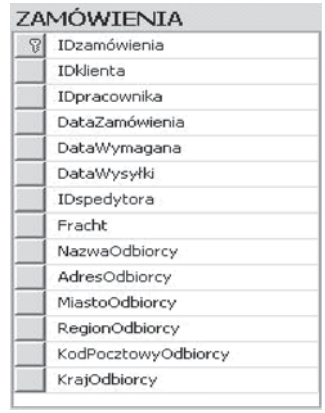
```
SELECT    Klienci.Idklienta,
          Klienci.Nazwafirmy,
          Zamowienia.IDzamówienia,
          Zamowienia.DataZamówienia, miasto
FROM      Klienci
  LEFT OUTER JOIN Zamowienia
  ON      Klienci.Idklienta = Zamowienia.Idklienta
WHERE     IDzamówienia is NULL
```

5. POLECENIE SELECT – OBLICZENIA ZBIOROWE

Grupowanie (GROUP BY) używamy do zaprezentowania wybranych danych (kolumn) bez ich powtarzania się np. pokazać identyfikatory klientów (idklienta) z tabeli Zamówienia

Doskonale wraz z tą dyrektywą spisują się wszelkie funkcje agregujące.
Są nimi między innymi:
SUM – suma wartości w zbiorze danych,
COUNT – liczność zbioru danych,
MIN – wartość minimalna w zbiorze danych,
MAX – wartość maksymalna w zbiorze danych,
AVG – wartość średnia w zbiorze danych.

Dyrektywa HAVING służy do ograniczania wyświetlanych rekordów. Gdybyśmy chcieli wyświetlić tylko osoby, które występują więcej niż 2 razy w powyższym wykazie należy użyć dyrektywy HAVING.



Przykład wybranych danych (wybranych kolumn) z tabeli Zamówienia

ZAMÓWIENIA		
Idzamówienia	Idklienta	Fracht
1	1	100
2	2	200
3	1	200
4	1	300
5	2	400

Grupowanie po klientach

Idklienta	Kolumna kalkulacyjna1	Kolumna kalkulacyjna2
1		
2		

- **Dla każdego klienta podać sumę wartości wszystkich zarejestrowanych zamówień**

```
SELECT    Idklienta, SUM (Fracht) AS Suma_wartość
FROM      Zamowienia
GROUP BY  Idklienta
```

- **Dla każdego klienta podać średnią wartość zamówienia (policzoną po wszystkich zarejestrowanych zamówieniach pochodzących od tego klienta) i najwyższą wartość zamówienia.**

```
SELECT    Idklienta, AVG(fracht) AS średnia_wartość, MAX(fracht) AS Maks_wartość
FROM      Zamowienia
GROUP BY  Idklienta
```

- **Dla każdego klienta podać sumę wartości wszystkich zarejestrowanych zamówień po 02/01/98**

```
SELECT    Idklienta, SUM (Fracht) AS Suma_wartość
FROM      Zamowienia
WHERE     DataZamówienia > '02-JAN-1998'
GROUP BY  Idklienta
```

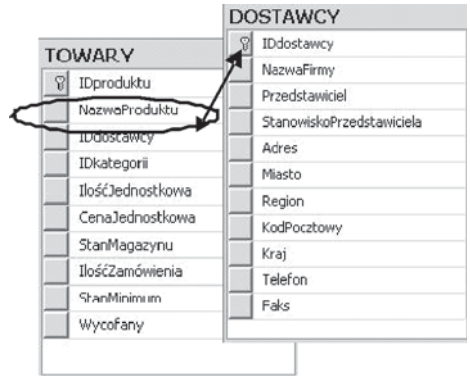
<u>Obliczenia zbiorcze w grupach wierszy we wskazanych tabelach</u>

- **Dla każdego towaru, na który były składane zamówienia w okresie od 01/04/1996 do 30/04/2004 podać średnią i największą liczbę sztuk zamawianych w ramach jednej pozycji zamówienia.**

```
SELECT      Towary.NazwaProduktu,
            AVG(Pozycje_zamówień .Ilość) AS średnia_ilość,
            MAX(Pozycje_zamówień.Ilość) AS Maks_ilość
FROM        Towary
   JOIN     Pozycje_zamówień
   ON       Towary.idproduktu = Pozycje_zamówień.idtowaru
   JOIN     Zamówienia
   ON       Pozycje_zamówień.IDzamówienia = Zamówienia.IDzamówienia
WHERE       Zamowienia.Datazamówienia BETWEEN '01-APR-1996' AND '01-APR-2004'
GROUP BY    Towary.NazwaProduktu
```

Zapytania z zagnieżdżonymi podzapytaniami

- **Podać wykaz towarów oferowanych przez dostawcę o nazwie 'Pavlova,Ltd.**




```
SELECT      Towary.NazwaProduktu
FROM        Towary
WHERE       IDdostawcy IN
   (SELECT IDdostawcy
   FROM     Dostawcy
    WHERE  Nazwafirmy='Pavlova, Ltd.')
```

Sformułowanie zadania, które odzwierciedla konstrukcję podaną wyżej jest następujące: podać wykaz towarów, takich że IDdostawcy należy do zbioru IDdostawcy występujących w ofertach od dostawcy o nazwie 'Pavlowa.Ltd

```
SELECT      Towary.NazwaProduktu,Dostawcy.IDdostawcy
FROM        Towary JOIN Dostawcy
   ON       Towary.IDdostawcy = Dostawcy.IDdostawcy
WHERE       Nazwafirmy='Pavlova, Ltd.'
```

- **Podać wykaz klientów (identyfikator, nazwa firmy), dla których są zamówienia złożone po 04-01-1998.**

```
SELECT      Idklienta, Nazwafirmy
FROM        Klienci
WHERE       Idklienta IN
    (SELECT DISTINCT Idklienta
    FROM    Zamowienia
    WHERE   Zamowienia.Datazamówienia > '04-JAN-1998')
```

Lub

```
SELECT DISTINCT     Klienci.Idklienta, Klienci.Nazwafirmy
FROM        Klienci JOIN Zamowienia
    ON      Klienci.Idklienta = Zamowienia.Idklienta
WHERE       Zamowienia.Datazamówienia > '04-JAN-1998'
```

- **Podać wykaz klientów (identyfikator, nazwa firmy), dla których nie ma zamówień o wartości większej niż 150złotych.**

```
SELECT      Idklienta, Nazwafirmy
FROM            Klienci
WHERE       Idklienta NOT IN
    ( SELECT DISTINCT Idklienta
    FROM    Zamowienia
    WHERE   Zamowienia.fracht > 150) ;
```

CUBE, ROLLUP, COMPUTE, COMPUTE BY

Rozszerzona składnia SELECT

SELECT [DISTINCT] [TOP n [PERCENT]] Lista kolumn
FROM Źródło Listy
[**WHERE** Warunek wybierający]
[**GROUP BY** Warunek grupowania]
[**HAVING** Szczegóły grupowania]
[**WITH** { **CUBE** | **ROLLUP** }]]
[**ORDER BY** Wyrażenie [ASC| DESC]]
[**COMPUTE** { { AVG | COUNT | MAX | MIN | SUM } (wyrazenie) } [,...n] [BY wyrazenie [,...n]]]

*** Pogrubione są słowa kluczowe**

CUBE

Operator CUBE może być wykorzystany do tworzenia i podsumowania wszystkich możliwych kombinacji grup (w formacie relacyjnym) na podstawie klauzuli, GROUP BY

ROLLUP

Operator ROLLUP umożliwia uzyskanie danych w standardowym formacie relacyjnym. Służy do podsumowania wyników grupowania

COMPUTE lub COMPUTE BY

Klauzule COMPUTE lub COMPUTE BY, generują dodatkowe wiersze podsumowań danych w formacie nie relacyjnym, który nie jest zgodny ze standardem ANSI. Format ten jest użyteczny do przeglądania, jednak dane nie są odpowiednio przystosowane do generowania zestawów wyników, które mogą być wykorzystane z innymi aplikacjami.

Poznając kolejne funkcje agregujące mamy zamiar dojść do wyniku podobnego do Tabeli przestawnej z Excela

ROK	KWARTAŁ 1	KWARTAŁ 2	KWARTAŁ 3	KWARTAŁ 4	SUMA
1995	14	16	13	14	57
1996	21	22	23	29	95
SUMA	35	38	36	43	152

ROZLICZENIA		
ROK	KWARTAŁ	KWOTA
1995	1	11
1995	1	3
1995	2	12
1995	2	4
1995	3	13
1995	4	14
1996	1	21
1996	2	22
1996	3	23
1996	4	24
1996	4	25

CUBE wyświetli nam wyniki z GRUP BY włącznie z wartościami NULL (z roku, kwartału oraz wspólnego). Jednocześnie pogrupuje wartości wyświetlając niepowtarzające się wartości sumy, najpierw według kwartału, następnie według. roku, a na koniec suma całości.

```
SELECT ROK, KWARTAL, SUM(KWOTA)
FROM TABELA
GROUP BY ROK, KWARTAL WITH CUBE
```

ROZLICZENIA		
ROK	KWARTAŁ	KWOTA
1995	1	11
1995	1	3
1995	2	12
1995	2	4
1995	3	13
1995	4	14
1996	1	21
1996	2	22
1996	3	23
1996	4	24
1996	4	25

ROK	KWARTAŁ	SUMA
1995	1	14
1996	1	21
NULL	1	35
1995	2	16
1996	2	22
NULL	2	38
1995	3	13
1996	3	23
NULL	3	36
1995	4	14
1996	4	49
NULL	4	63
NULL	NULL	172
1995	NULL	57
1996	NULL	115

ROLLUP wyświetli nam zsumowany każdy kwartał każdego roku, następnie sumę latami i na koniec podsumowanie za całość

```
SELECT ROK, KWARTAL, SUM(KWOTA)
FROM TABELA
GROUP BY ROK, KWARTAL WITH ROLLUP
```

ROZLICZENIA		
ROK	KWARTAŁ	KWOTA
1995	1	11
1995	1	3
1995	2	12
1995	2	4
1995	3	13
1995	4	14
1996	1	21
1996	2	22
1996	3	23
1996	4	24
1996	4	25

ROK	KWARTAŁ	SUMA
1995	1	14
1995	2	16
1995	3	13
1995	4	14
1995	NULL	57
1996	1	21
1996	2	22
1996	3	23
1996	4	49
1996	NULL	115
NULL	NULL	172

PIVOT

TWORZENIE TABELI PRZESTAWNEJ (KLAUZULA PIVOT)

Tabele przestawne są bardzo przydatne przy tworzeniu różnych zestawień, pewną wartość odnajdujemy na przecięciu wiersza i kolumny.

W poniższym przykładzie tworzona jest tabela przestawna, która w pierwszej kolumnie wyświetla ROK a kolejne kolumny reprezentują 4 kwartały roku (1, 2, 3, 4)

– wartości na przecięciu kolumny i wiersza mogą liczyć SUM COUNT AVG MAX MIN kwoty.

– Wartości NULL występują wtedy, gdy w danym roku nie było wartości w określonym kwartale

ROZLICZENIA		
ROK	KWARTAŁ	KWOTA
1995	1	11
1995	1	3
1995	2	12
1995	2	4
1995	3	13
1995	4	14
1996	1	21
1996	2	22
1996	3	23
1996	4	24
1996	4	25

ROK	1	2	3	4
1995	14	16	13	14
1996	21	22	23	49

```
SELECT *
FROM
    (
     SELECT ROK, KWARTAŁ, KWOTA
     FROM ROZLICZENIA
    ) AS A
PIVOT
    (COUNT(KWOTA) FOR KWARTAŁ in ([1],[2],[3],[4])) AS B

SELECT *
FROM
    (
     SELECT ROK, KWARTAŁ, KWOTA
     FROM ROZLICZENIA
    ) AS A
PIVOT
    (SUM(KWOTA) FOR KWARTAŁ in ([1],[2],[3],[4])) AS B
```

OPERACJA UNION

UNION nie oznacza złączenia, ale zapewnia możliwość połączenia danych z wielu zapytań w jeden wynik. Operacja UNION jest użyteczna, gdy chce się obejrzeć podobne dane z dwóch lub więcej tabel w czasie jednego wyświetlania

Składnia

Instrukcja Select
UNION
Instrukcja Select

Przykład: Trzy zapytania o „Londyńczyków" dostawców, klientów i pracowników

```
SELECT NazwaFirmy, Przedstawiciel, MIASTO
FROM DOSTAWCY
WHERE Miasto = 'LONDYN'

SELECT NazwaFirmy, Przedstawiciel, MIASTO
FROM KLIENCI
WHERE Miasto = 'LONDYN'

SELECT Nazwisko, Stanowisko, MIASTO
FROM PRACOWNICY
WHERE Miasto = 'LONDYN'
```

Operacja Union

```
SELECT    'Dostawcy' as Typ,NazwaFirmy, Przedstawiciel,MIASTO
          FROM DOSTAWCY
          WHERE Miasto = 'LONDYN'
UNION
          SELECT 'Klienci',NazwaFirmy, Przedstawiciel,MIASTO
          FROM KLIENCI
          WHERE Miasto = 'LONDYN'
UNION
          SELECT 'Pracownicy',Nazwisko, Stanowisko,MIASTO
          FROM PRACOWNICY
          WHERE Miasto = 'LONDYN'
          ORDER BY 1,2
```

Zauważmy, że trzy listy wyboru zawierają tę samą liczbę elementów i że typy danych są zgodne.

TYP	NazwaFirmy	Przedstawiciel	Miasto
Dostawcy	Exotic Liquids	Charlotte Cooper	Londyn
Klienci	Around the Horn	Thomas Hardy	Londyn
Klienci	B's Beverages	Victoria Ashworth	Londyn
Klienci	Consolidated Holdings	Elizabeth Brown	Londyn
Klienci	Eastern Connection	Ann Devon	Londyn
Klienci	North/South	Simon Crowther	Londyn
Klienci	Seven Seas Imports	Hari Kumar	Londyn
Pracownicy	Buchanan	SALESMAN	Londyn
Pracownicy	Dodsworth	SALESMAN	Londyn
Pracownicy	Ford	ANALYST	Londyn
Pracownicy	King	MANAGER	Londyn
Pracownicy	Miller	CLERK	Londyn
Pracownicy	Suyama	MANAGER	Londyn

OPERACJA EXCEPT

EXCEPT połączenia danych z wielu zapytań w jeden wynik. Operacja EXCEPT jest użyteczna, gdy chce się obejrzeć różnicę zbiorów z podobnymi danymi

Składnia

Instrukcja Select
EXCEPT
Instrukcja Select

Przykład

```
SELECT Nazwisko, Stanowisko,MIASTO
FROM PRACOWNICY
EXCEPT
SELECT Nazwisko, Stanowisko,MIASTO
FROM PRACOWNICY
WHERE Miasto = 'LONDYN
```

Inny sposób zapisu

```
SELECT Nazwisko, Stanowisko, MIASTO
FROM PRACOWNICY
WHERE Miasto <> 'LONDYN'
```

lub

SELECT Nazwisko, Stanowisko, MIASTO
FROM PRACOWNICY
WHERE Miasto NOT IN ('LONDYN')

OPERACJA INTERSECT

INTERSECT połączenia danych z wielu zapytań w jeden wynik. Operacja INTERSECT jest użyteczna, gdy chce się obejrzeć część wspólną zbiorów z podobnymi danymi

Składnia

Instrukcja Select
INTERSECT
Instrukcja Select

Przykład

```
SELECT Nazwisko, Stanowisko,MIASTO
FROM PRACOWNICY
WHERE Stanowisko ='CLERK'
INTERSECT
SELECT Nazwisko, Stanowisko,MIASTO
FROM PRACOWNICY
WHERE Miasto = 'LONDYN
```

Można inaczej zapisać

SELECT Nazwisko, Stanowisko, MIASTO
FROM PRACOWNICY
WHERE Miasto = 'LONDYN' AND Stanowisko = 'CLERK'

Literatura

Marcin Zawadzki, *SQL Serwer 2005i,* MIKOM, 2006
Scott Cameron, *Microsoft SQL Server 2008 Analysis Services,* A.P.N. Promise, 2009
Eric Johnson, Joshua Jones, *Modelowanie danych w SQL Server 2005 i 2008. Przewodnik*, Helion, 2009
Itzik Ben-Gan, Lubor Kollar, Dejan Sarka, Steve Kass, *Microsoft SQL Server 2008 od środka Zapytania w języku T-SQL*, A.P.N. Promise, 2009