



**Wyższa Szkoła Bankowa  
Gdańsk**

**Gdynia**



# Systemy baz danych. Podstawy SQL.

Krzysztof Ziółkowski



**[www.wsb.pl](http://www.wsb.pl)**

# Pojęcie bazy danych

- Baza danych – zbiór danych zapisanych zgodnie z określonymi regułami.



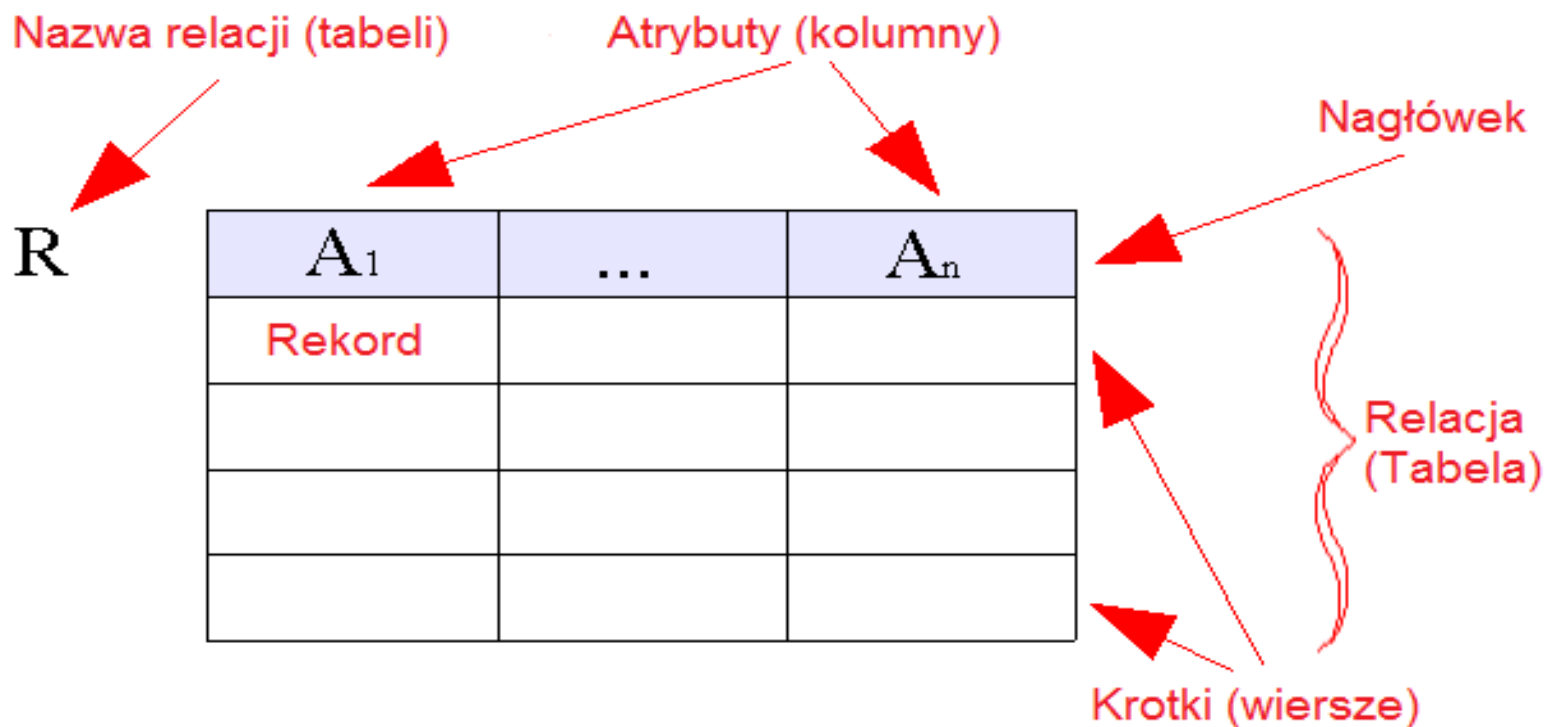
# Wymień znane Ci bazy danych

- Książka telefoniczna,
- Katalog biblioteczny,
- System ewidencji ludności (PESEL).

# Relacyjne bazy danych

- **Relacja** – odzwierciedlenie oddziaływania między dwoma bądź większą liczbą podmiotów
- **Relacyjna baza danych** (ang. Relational Database Management Systems, RDBMS) – jest logicznie wydzielonym zbiorem danych, zorganizowanych w formie dwuwymiarowej tabeli, składającej się z wierszy (krotek) dzielonych na kolumny (atrybuty).
- Najpopularniejszymi wolnodostępnymi RDBMS są MySQL, PostgreSQL i SQLite; natomiast zamkniętymi: Oracle, Microsoft SQL Server, IBM DB2, Microsoft Access i SAP HANA.

# Relacyjne bazy danych



# Pojęcie encji

- Encja (ang. entity) – reprezentacja wyobrażonego lub rzeczywistego obiektu (grupy obiektów).
- Encja stosowana jest przy modelowaniu danych podczas projektowania systemów informatycznych.
- Przykładowe encje:
  - Rzeczywiste: osoba, towar, pojazd, książka, dom, komputer, telefon,
  - Niematerialne: konto bankowe, kupno towaru, wypożyczenie książki, rezerwacja biletu lotniczego,

# Atrybut encji

- Encje mają właściwości, zwane **atrybutami**, które każdej encji ze zbioru encji przypisują pewną wartość z dziedziny wartości danego atrybutu.
- Przykłady encji (i atrybuty w encji):
  - Osoby (imię, nazwisko, PESEL)
  - Pojazdy (wysokość, szerokość, długość, sposób poruszania się)

# Związek

- Diagram związków encji lub Diagram ERD (od ang. Entity-Relationship Diagram) - logiczne powiązanie dwóch lub więcej zbiorów encji.
- Notacja Martina – notacja używana w modelowaniu diagramów związków encji.



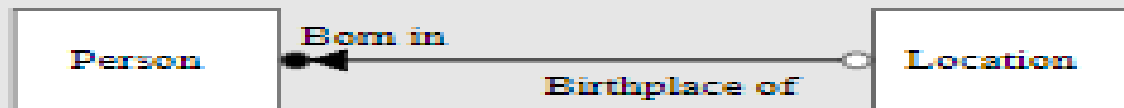
Chen



IDEFIX



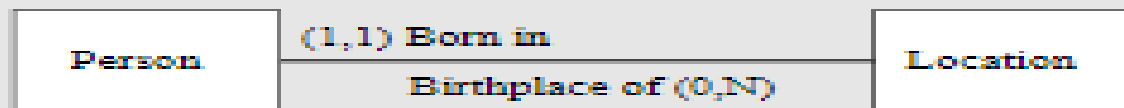
Bachman



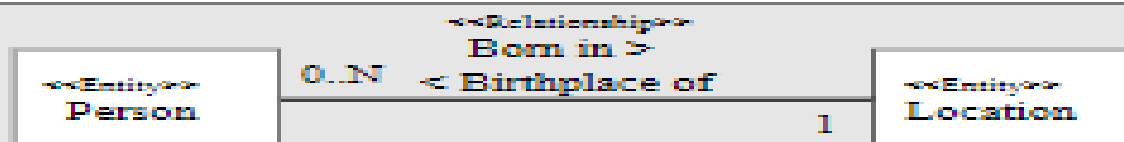
Martin / IE /  
Crow's Foot



Min-Max / ISO



UML



# ERD

- **Krotność** – określającą ile encji wchodzi w skład związku:
- 1:1 („jeden do jeden”) – encji odpowiada dokładnie jedna encja,
- 1:N („jeden do wielu”) – encji odpowiada jedna lub więcej encji,
- M:N („wiele do wielu”) – jednej lub więcej encjom odpowiada jedna lub więcej encji.

# Postać normalna

- **Postać normalna** – postać relacji w bazie danych, w której nie występuje redundancja.



# 1NF

- Relacja jest w pierwszej postaci normalnej, jeśli:

< Dziedziny atrybutów muszą być elementarne (nierozkładalne/atomowe/proste). >

kolejność wierszy może być dowolna (znaczenie danych nie zależy od kolejności wierszy).

# 1NF



Płeć	Imię
Męska	Marek, Rafał, Krzysztof
Żeńska	Malwina, Magda, Maja

Płeć	Imię
Męska	Marek
Męska	Rafał
Męska	Krzysztof
Żeńska	Malwina
Żeńska	Magda
Żeńska	Maja



## 2NF

- Musi zachodzić 1NF i żadna informacja w krotce nie może zależeć tylko od części klucza głównego.

## 2NF

Imię	Nazwisko	Płeć	Stanowisko	Stawka za godzinę
Marek	Markowski	Męska	Ochroniarz	11 zł
Maja	Majewska	Żeńska	Kasjerka	20 zł
Magda	Wołek	Żeńska	Kasjerka	20 zł

Imię	Nazwisko	Stanowisko	Stawka za godzinę
Marek	Markowski	Ochroniarz	11 zł
Magda	Wołek	Kasjerka	20 zł
Maja	Majewska	Kasjerka	20 zł

Imię	Płeć
Marek	Męska
Magda	Żeńska
Maja	Żeńska

## 3NF

- Musi zachodzić 2NF i żaden atrybut, który nie jest kluczem podstawowym nie zależy od niczego innego.



## 3NF

Imię	Nazwisko	Stanowisko	Stawka za godzinę
Marek	Markowski	Ochroniarz	11 zł
Magda	Wołek	Kasjerka	20 zł
Maja	Majewska	Kasjerka	20 zł

Imię	Nazwisko	Stanowisko
Marek	Markowski	Ochroniarz
Magda	Wołek	Kasjerka
Maja	Majewska	Kasjerka

Stanowisko	Stawka za godzinę
Ochroniarz	11 zł
Kasjerka	20 zł

# MS SQL

- Microsoft SQL Server (MS SQL) – system zarządzania bazą danych, charakteryzuje się tym, iż jako język zapytań używany jest przede wszystkim Transact-SQL, który stanowi rozwinięcie standardu ANSI/ISO.

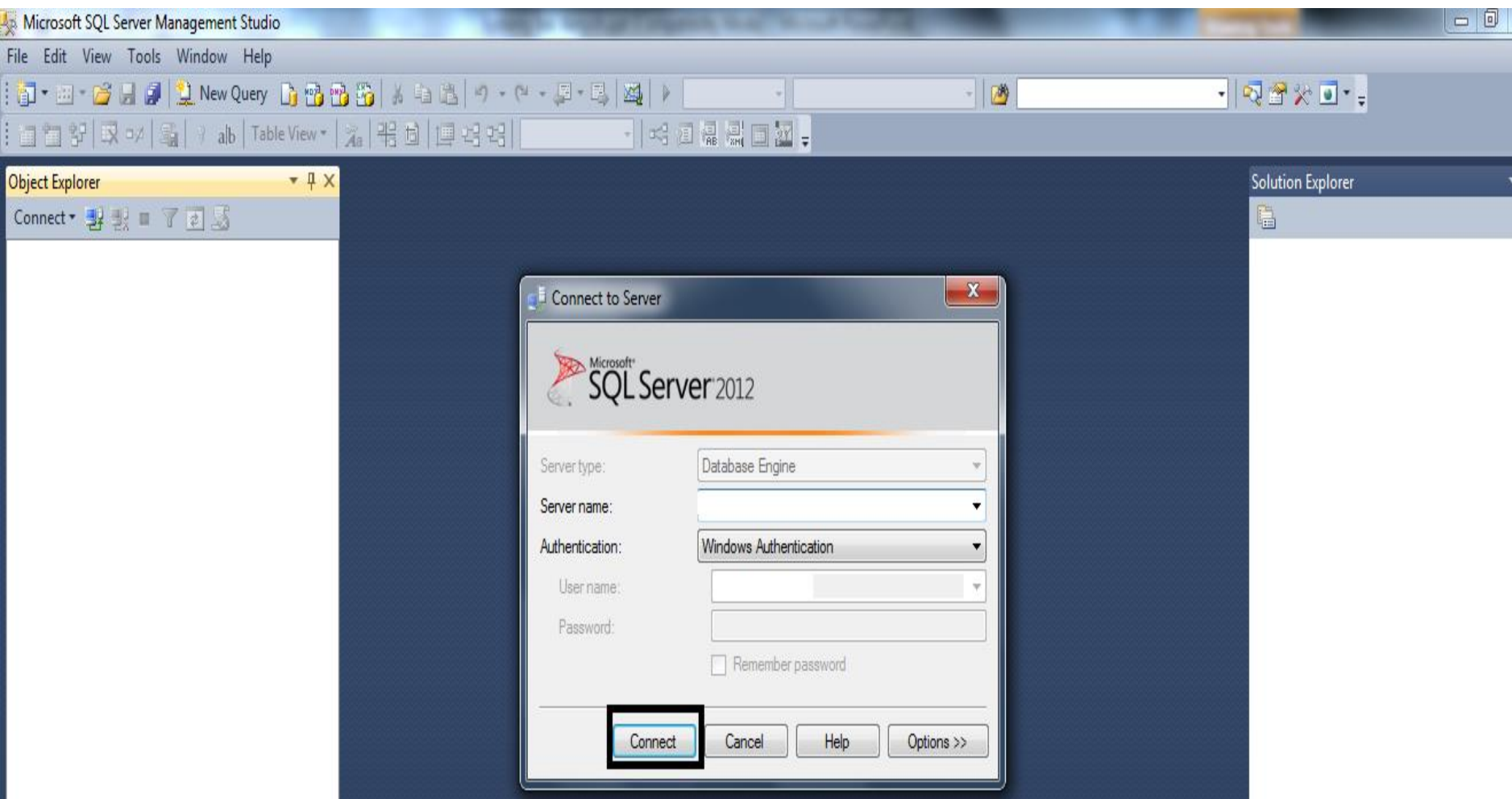
# MS SQL

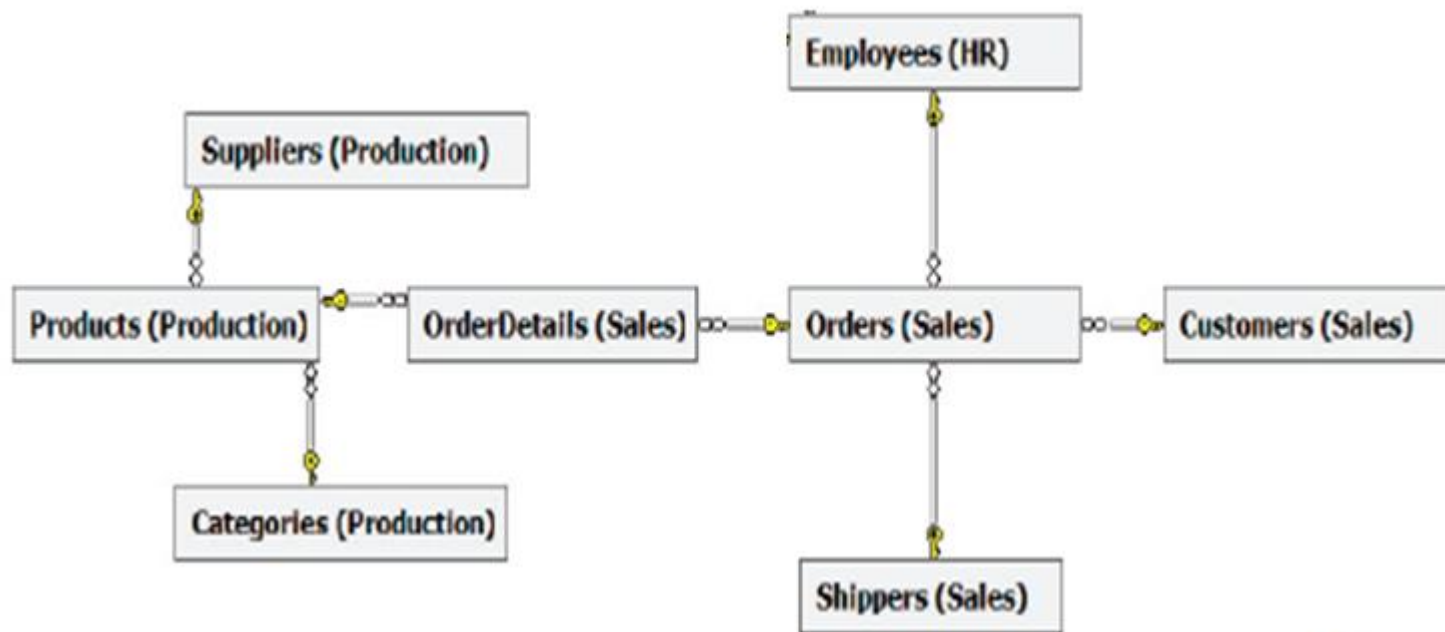
- MS SQL Server for OS/2 began as a project to port Sybase SQL Server onto OS/2 in 1989, by Sybase, Ashton-Tate, and Microsoft.
- SQL Server 4.2 for NT 1993
- SQL Server 6.0 is released in 1995
- SQL Server 7.0 is released in 1998
- SQL Server 2005, released in 2005
- SQL Server 2008[4]
- SQL Server 2008 R2
- SQL Server 2012
- SQL Server 2014
- SQL Server 2016
- SQL Server 2017

# Jak zacząć?

- <https://www.microsoft.com/en-us/download/details.aspx?id=42299>
- <https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-2017>

- 
- A screenshot of a Windows Start menu search results window. The search term is 'Microsoft SQL Server 2012'. The results list includes:
- Microsoft SQL Server 2012
  - Download Microsoft SQL Server Cor
  - Import and Export Data (32-bit)
  - Import and Export Data (64-bit)
  - SQL Server Data Tools
  - SQL Server Management Studio** (highlighted with a black box)
  - Configuration Tools
  - Documentation & Community
  - Integration Services
  - Microsoft SQL Server 2017
  - Microsoft SQL Server Tools 17
  - Microsoft SQL Server Management S
  - Performance Tools
  - Microsoft System Center





# Klucze

- <https://msdn.microsoft.com/pl-pl/library/encyklopedia-sql--klucze-glowne--primary-key-i-identity.aspx>



# Zadanie

- TR\_SCHEMA zawiera następujące tabele:

T\_PRODUKTY                      T\_ZAMOWIENIA                      T\_KLIENCI    T\_PRODUKCJA\_ZESPOL    T\_ZESPOL  
T\_PRACOWNICY

Opis schematu:

Klient może złożyć kilka zamówień, a produkt może być zamawiany wielokrotnie przez różnych klientów. (Innymi słowy, klient może kupić kilka produktów, a jeden produkt może zostać zakupiony przez wielu klientów).

Produkty są obsługiwane przez zespoły - jeden zespół może obsługiwać kilka produktów, a jeden produkt może być obsługiwany przez różne zespoły. (Podpowiedź: użyj tabeli T\_PRODUKCJA\_ZESPOL)

Zespoły zawsze składają się z kilku pracowników, ale pracownik może należeć tylko do jednego zespołu.

Narysuj ten schemat jako diagram relacji między jednostkami, wyświetlając nazwy tabel, połączenia typu "jeden do jednego" między tabelami. Upewnij się, że nie ma nienormalizowanych połączeń wiele do wielu.

# T-SQL

## Data Manipulation Language (DML\*)

- Statements for querying and modifying data
- SELECT, INSERT, UPDATE, DELETE

## Data Definition Language (DDL)

- Statements for object definitions
- CREATE, ALTER, DROP

## Data Control Language (DCL)

- Statements for security permissions
- GRANT, REVOKE, DENY

# SELECT

Naturalnym odruchem jest chęć sprawdzenia, czy informacje, które zostały wstawione do odpowiedniej tabeli, faktycznie się tam znalazły. Do wyświetlania zawartości danych z tabel służy polecenie SELECT. Na początku wykonajmy poniższe zapytanie:

```
SELECT columnName1, columnName2  
FROM tableName;
```

```
SELECT [co?] FROM [skąd?]
```

```
SELECT 'Witaj Świecie!';
```

# T-SQL

Elements:	Predicates and Operators:
Predicates	IN, BETWEEN, LIKE
Comparison Operators	=, >, <, >=, <=, <>, !=, !>, !<
Logical Operators	AND, OR, NOT
Arithmetic Operators	+, -, *, /, %
Concatenation	+

**T-SQL enforces operator precedence**

# T-SQL

Order of Evaluation	Operators
1	( ) Parentheses
2	*, /, % (Multiply, Divide, Modulo)
3	+, - (Add/Positive/Concatenate, Subtract/Negative)
4	=, <, >, >=, <=, !=, !>, !< (Comparison)
5	NOT
6	AND
7	BETWEEN, IN, LIKE, OR

# T-SQL

## String Functions

- SUBSTRING
- LEFT, RIGHT
- LEN
- DATALENGTH
- REPLACE
- REPLICATE
- UPPER, LOWER
- RTRIM, LTRIM

## Date and Time Functions

- GETDATE
- SYSDATETIME
- GETUTCDATE
- DATEADD
- DATEDIFF
- YEAR
- MONTH
- DAY

## Aggregate Functions

- SUM
- MIN
- MAX
- AVG
- COUNT

# T-SQL

```
/*
```

```
    This is a block  
    of commented code
```

```
*/
```

```
-- This line of text will be ignored
```

Element	Expression	Role
SELECT	<select list>	Defines which columns to return
FROM	<table source>	Defines table(s) to query
WHERE	<search condition>	Filters rows using a predicate
GROUP BY	<group by list>	Arranges rows by groups
HAVING	<search condition>	Filters groups using a predicate
ORDER BY	<order by list>	Sorts the output



5: SELECT <select list>

1: FROM <table source>

2: WHERE <search condition>

3: GROUP BY <group by list>

4: HAVING <search condition>

6: ORDER BY <order by list>

SELECT empid, YEAR(orderdate) AS orderyear  
FROM Sales.Orders  
WHERE custid =71  
GROUP BY empid, YEAR(orderdate)  
HAVING COUNT(\*) > 1  
ORDER BY empid, orderyear;

# SELECT DISTINCT

Select country  
From Sales.Customers;

- Specifies that only unique rows can appear in the result set
- Removes duplicates based on column list results, not source table
- Provides uniqueness across set of selected columns
- Removes rows already operated on by WHERE, HAVING, and GROUP BY clauses
- Some queries may improve performance by filtering out duplicates prior to execution of SELECT clause

# Aliasy


```
SELECT orderid, unitprice, qty AS quantity  
FROM Sales.OrderDetails;
```

```
SELECT orderid, unitprice, quantity = qty  
FROM Sales.OrderDetails;
```


```
SELECT orderid, unitprice quantity  
FROM Sales.OrderDetails;
```

# Aliasy - tabele

```
SELECT orderid, unitprice, qty  
FROM Sales.OrderDetails AS OD;
```



```
SELECT orderid, unitprice, qty  
FROM Sales.OrderDetails OD;
```



```
SELECT OD.orderid, OD.unitprice, OD.qty  
FROM Sales.OrderDetails AS OD;
```

# Where

SELECT kolumna1, kolumna2, ..., kolumnna FROM Tabela  
WHERE [warunki logiczne wyciągania rekordów]

# Where

```
SELECT orderid, unitprice, qty AS quantity  
FROM Sales.OrderDetails  
WHERE quantity > 10;
```

```
SELECT orderid, unitprice, qty AS quantity  
FROM Sales.OrderDetails  
WHERE qty > 10;
```

# Like

SELECT kolumna1, kolumna2, ... , kolumnaN FROM Tabela WHERE  
kolumna LIKE wzorzec\_do\_porównania

Wzorzec	Opis
%	Dopasowanie do jakiegokolwiek łańcucha znakowego, w szczególności pusty ciąg znaków.
_	Dopasowanie do jakiegokolwiek pojedynczego znaku.
[]	Dopasowanie do ciągu znaków wymienionych w zakresie ograniczonym przez [].
[^]	Dopasowanie do ciągu znaków, oprócz zbioru wymienionego w [^].

Źródło: <https://msdn.microsoft.com/pl-pl/library/>



# Like

dotyczą schematów zaczynających się na literę 'P' (1),  
zawierają trzycyfrowy identyfikator logu (2),  
zostały wykonane na obiektach, których druga litera to 'a', a  
trzecią nie jest 'l' (3).

```
SELECT DatabaseLogID, [Schema], [Object] FROM  
DatabaseLog WHERE [Schema] LIKE 'P%' -- (1) AND  
DatabaseLogId LIKE '[0-9][0-9][0-9]' -- (2)  
AND [Object] LIKE '_a[^l]%' -- (3)
```

Źródło: <https://msdn.microsoft.com/pl-pl/library/>

# Order By

```
SELECT orderid, unitprice, qty AS quantity  
FROM Sales.OrderDetails  
ORDER BY quantity;
```



# CASE

**CASE w SQL Server** to instrukcja wyboru porównywalna do instrukcji warunkowej **IF THEN ELSE** w innych językach programowania (switch w C++).

```
CASE wartość_lub_nazwa_kolumny  
WHEN wartość_1 THEN wynik_1  
WHEN wartość_2 THEN wynik_2  
WHEN wartość_n THEN wynik_n  
[ ELSE wynik_gdy_zaden_z_warunków_nie_jest_spełniony ]  
END
```

# CASE

```
SELECT productid, productname, categoryid,  
CASE categoryid  
WHEN 1 THEN 'Beverages'  
WHEN 2 THEN 'Condiments'  
WHEN 2 THEN 'Confections'  
ELSE 'Unknown Category'  
END AS categoryname  
FROM Production.Categories
```

# JOIN

[https://pl.wikibooks.org/wiki/SQL/Typy\\_z%C5%82%C4%85cze%C5%84](https://pl.wikibooks.org/wiki/SQL/Typy_z%C5%82%C4%85cze%C5%84)

[https://pl.wikipedia.org/wiki/Join\\_\(SQL\)](https://pl.wikipedia.org/wiki/Join_(SQL))

[https://www.w3schools.com/sql/sql\\_join.asp](https://www.w3schools.com/sql/sql_join.asp)

<https://docs.microsoft.com/en-us/sql/relational-databases/performance/joins?view=sql-server-2017>

# Join

Aby wypełnić tabelę wirtualną utworzoną przez klauzulę FROM w instrukcji SELECT, SQL Server używa łączenia operatorów. Operatory te dodają lub usuwają wiersze z wirtualnej tabeli, zanim zostaną przekazane kolejne fazy logiczne instrukcji SELECT:

- Operator łączenia krzyżowego (CROSS JOIN) dodaje wszystkie możliwe kombinacje dwóch wierszy tabel danych wejściowych do wiersza wirtualna tabela. Każde filtrowanie wierszy będzie miało miejsce w klauzuli WHERE. Dla większości zapytań, tego operatora należy unikać.
- Wewnętrzny operator łączenia (INNER JOIN lub po prostu JOIN) najpierw tworzy produkt kartezjański, a następnie filtruje wyniki przy użyciu predykatu dostarczonego w klauzuli ON, usuwając wszystkie wiersze z wirtualnej tabeli, które nie spełniają predykatu. Typ łączenia wewnętrznego jest bardzo powszechnym typem łączenia dla pobierania wierszy atrybuty pasujące do różnych tabel, na przykład dopasowywanie klientów do zamówień według wspólnego klienta.
- Zewnętrzny operator łączenia (LEFT OUTER JOIN, RIGHT OUTER JOIN, FULL OUTER JOIN) najpierw tworzy produkt kartezjański, podobnie jak sprzężenie wewnętrzne, filtruje wyniki, aby znaleźć wiersze pasujące do każdej tabeli.

Jednak wszystkie wiersze z jednej tabeli są zachowywane, dodawane do wirtualnej tabeli po początkowym filtrze stosowany. Wartości NULL są umieszczane na atrybutach, w których nie znaleziono zgodnych wartości.

# Inner Join

```
SELECT ...  
FROM Table1 JOIN Table2  
ON <on_predicate>
```

```
FROM t1 JOIN t2  
ON t1.column = t2.column
```

# Outer Joins

Podczas pisania zapytań z użyciem zewnętrznych sprzężeń, weź pod uwagę następujące wskazówki:

- Jak widzieliśmy, aliasy tabel są preferowane nie tylko dla listy SELECT, ale także dla wyrażania klauzula ON.
- Połączenia zewnętrzne wyrażane są za pomocą słów kluczowych LEFT, RIGHT lub OUTER poprzedzającego ZŁĄCZE ZEWNĘTRZNE. Celem słowa kluczowego jest wskazanie, która tabela (po której stronie słowa kluczowego JOIN) powinna być zachowana i wszystkie wyświetlane wiersze, dopasowanie lub brak zgodności.
- Podobnie jak w przypadku połączeń wewnętrznych, łączenia zewnętrzne mogą być wykonywane na jednym dopasowanym atrybucie, lub mogą być wykonywane na wielu pasujących atrybutach.
- W przeciwieństwie do połączeń wewnętrznych, kolejność, w jakiej tabeli są wymienione i dołączone do klauzuli FROM ma znaczenie, jako i to, czy wybierzesz LEWA, czy PRAWĄ dla stron swojego łączenia.



# Outer Joins

```
FROM t1 LEFT OUTER JOIN t2 ON  
    t1.col = t2.col
```

```
FROM t1 RIGHT OUTER JOIN t2 ON  
    t1.col = t2.col
```

```
FROM t1 LEFT OUTER JOIN t2 ON  
    t1.col = t2.col  
WHERE    t2.col IS NULL
```

# TOP

- SELECT TOP (N) <column\_list>
- FROM <table\_source>
- WHERE <search\_condition>;



# OFFSET i FETCH

- Umożliwia wygodne stronicowanie danych zwracanych przez zapytanie. Powyższe polecenia umieszczane są po klauzuli **ORDER BY**, która w ich przypadku jest wymagana. Składnia:
  - **OFFSET *offset\_value* { ROW | ROWS } [FETCH { FIRST | NEXT } *fetch\_value* { ROW | ROWS } ONLY]**

Źródło: <https://mndevnotes.wordpress.com/2012/04/22/nawosci-w-sql-server-2012-stronicowanie-wynikow-offset-fetch/>

# OFFSET i FETCH

- ***offset\_value*** – ilość rekordów do pominięcia  
***fetch\_value*** – ilość rekordów do pobrania
- Wartości dla OFFSET i FETCH możemy podawać jako liczba, zmienna typu int lub zapytanie zwracające wartość typu int;
- Słów kluczowych ROW i ROWS możemy używać zamiennie, to samo dotyczy słów FIRST i NEXT;
- Polecenie FETCH jest opcjonalne, bez niego zwrócone zostaną rekordy od wartości OFFSET + 1 do końca zbioru;

Źródło: <https://mndevnotes.wordpress.com/2012/04/22/nawosci-w-sql-server-2012-stronicowanie-wynikow-offset-fetch/>

# Typy danych T-SQL

- <https://msdn.microsoft.com/pl-pl/library/encyklopedia-sql--typy-danych-t-sql.aspx>

CHAR -> VARCHAR -> NVARCHAR -> TINYINT -> INT ->  
DECIMAL -> TIME -> DATE -> DATETIME2 -> XML

## Funkcje tekstowe T-SQL

- `LEFT( expression, int_value )` oraz `RIGHT(expression, int_value )` – to jedne z najprostszych funkcji tekstowych. Zwracają `int_value` znaków od lewej lub prawej z wyrażenia `expression` na którym działa

## Funkcje tekstowe T-SQL

- CHARINDEX ( expression1 , expression2 , start\_location ) – szuka pierwszego wystąpienia ciągu znaków expression1 w wartości znakowej podanej jako argument expression2.
- UPPER (string)/LOWER(string) zamienia wszystkie litery na duże/małe.

## Funkcje tekstowe T-SQL

- `LEN(expression)` – zwraca wartość integer, równej liczbie długości wyrażenia `expression`.
- `DATALENGTH(expression)`
- `SUBSTRING (expression, start_location , int_value)` – zwraca fragment tekstu, liczbę `int_value` znaków z wyrażenia podanego w `expression`, zaczynając od miejsca `start_location`.



## Funkcje tekstowe T-SQL

- REPLACE ( expression , string\_pattern , string\_replacement) – podmienia każde wystąpienie ciągu znaków string\_pattern na string\_replacement w przeszukiwanym wyrażeniu expression.
- Funkcja FORMAT pozwala na formatowanie podanej wartości zgodnie z podanym schematem

- <https://technet.microsoft.com/pl-pl/library/sql-server-2012--nowe-funkcje-daty-i-czasu.aspx>
- <https://docs.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql?view=sql-server-2017>
- <https://docs.microsoft.com/en-us/sql/t-sql/functions/cast-and-convert-transact-sql?view=sql-server-2017>

- <https://docs.microsoft.com/en-us/sql/t-sql/language-elements/coalesce-transact-sql?view=sql-server-2017>
- <https://docs.microsoft.com/en-us/sql/t-sql/functions/logical-functions-choose-transact-sql?view=sql-server-2017>