

# SQL Podstawy 2

---

*DO WHAT YOU CAN'T*

~Casey Neistat

<https://youtu.be/jG7dSXcfVqE>

W SQL, polecenie **SELECT** zwraca zestaw wyników danych z zbioru takiego jak: tabela albo widok. **SELECT** może być używany z innymi warunkami np. **WHERE**, **GROUP BY** i **ORDER BY**, które precyzują pożądane rezultaty.

## 2.1 Filtrowanie wierszy używając warunku **WHERE**:

Klauzula **WHERE** filtruje tylko rekordy, które spełniają warunek:

```
SELECT * FROM nazwaTabeli WHERE warunek --np. ID > 10
```

## 2.2 Sortowanie wyniku przy pomocy **ORDER BY**:

Warunek **ORDER BY** służy do posortowania zwracanego rezultatu według konkretnej z kolumn lub wyrażenia:

```
SELECT * FROM nazwaTabeli ORDER BY nazwaKolumny
```

## 2.3 Grupowanie rezultatu wykorzystując **GROUP BY**:

**GROUP BY** grupuje wiersze rezultatu według określonej wartości:

```
SELECT type, count(*) FROM nazwaTabeli GROUP BY wartosc
-- count() zlicza ilość wierszy
```

## 2.4 Filtrowanie grup używając **HAVING**:

Jest czasami taki moment, że chcemy coś pogrupować, ale dodatkowo chcielibyśmy wyszczególnić pewne wartości. W takiej sytuacji niestety nie możemy użyć wcześniej poznanego **WHERE**. Tu pojawia się nowy warunek: **HAVING**.

```
SELECT type, count(*) FROM nazwaTabeli GROUP BY nazwa kolumny HAVING
warunek
-- np. count(*) < 10
```

## 2.5 Zwracanie tylko N pierwszych wierszy

Kiedy chcemy zwrócić tylko kilka pierwszych wierszy używamy warunku **TOP**:

```
SELECT TOP 10 * FROM nazwaTabeli
```

## 2.6 Daty

W T-SQL jest wbudowana funkcja **GETDATE()**, która zwraca obecną datę.

```
SELECT GETDATE() -- 2018-01-01 01:01:01.111
```

Można również konwertować daty na format, który nas interesuje:

```
SELECT CONVERT(VARCHAR(30), GETDATE(), kodFormatu)
```

Przykładowe kody	
KOD	REZULTAT
100	"Jul 21 2018 7:56AM"
102	"21/07/2018"
104	"21.07.2016"
113	"21 Jul 2018 07:57:59:553"

Drugim sposobem jest użycie funkcji CONVERT():

```
SELECT FORMAT(GETDATE(), dddd, MMMM dd, yyyy hh:mm:ss tt')
```

Wynik: Monday, September 05, 2016 12:01:02 AM

Lista argumentów: (**rozmiar ma znaczenie**)

yyyy	2016
yy	16
MMMM	September
MM	09
M	9
dddd	Monday
ddd	Mon
dd	05
d	5
HH	00
H	0
hh	12
h	12
mm	01
m	1
ss	02
s	2
tt	AM
t	A

## 2.6.1 Operacje na datach

```
SELECT DATEADD(coZmieniamy , wartość , data)
```

Przykład:

```
DECLARE @now DATETIME2 = GETDATE();
SELECT @now; --2016-07-21 14:39:46.4170000
SELECT DATEADD(YEAR, 1, @now) --2017-07-21 14:39:46.4170000
SELECT DATEADD(QUARTER, 1, @now) --2016-10-21 14:39:46.4170000
SELECT DATEADD(WEEK, 1, @now) --2016-07-28 14:39:46.4170000
SELECT DATEADD(DAY, 1, @now) --2016-07-22 14:39:46.4170000
SELECT DATEADD(HOUR, 1, @now) --2016-07-21 15:39:46.4170000
SELECT DATEADD(MINUTE, 1, @now) --2016-07-21 14:40:46.4170000
SELECT DATEADD(SECOND, 1, @now) --2016-07-21 14:39:47.4170000
SELECT DATEADD(MILLISECOND, 1, @now) --2016-07-21 14:39:46.4180000
```

Możemy też podać tu ujemne wartość:

```
DECLARE @now DATETIME2 = GETDATE();
SELECT @now; --2016-07-21 14:39:46.4170000
SELECT DATEADD(YEAR, -1, @now) --2015-07-21 14:39:46.4170000
```