# Instrukcje grupowania w języku SQL, etap I

- Wprowadzenie
- Przeczytaj
- Prezentacja multimedialna
- Sprawdź się
- Dla nauczyciela

# Instrukcje grupowania w języku SQL, etap I

Źródło: Tobias Fischer, domena publiczna.

Rzadko się zdarza, aby baza danych służyła wyłącznie do odczytywania zapisanych w niej informacji. Zazwyczaj chcemy także analizować zawartość tabel oraz wykonywać obliczenia, wykorzystując przechowywane w nich dane. Wówczas z pomocą przychodzą funkcje agregujące języka SQL.

Więcej informacji na temat instrukcji grupowania w języku SQL znajdziesz w e-materiałach:

- Instrukcje grupowania w języku SQL, etap II,
- Instrukcje grupowania w języku SQL, etap III,
- Instrukcje grupowania w języku SQL, etap IV.

> **Twoje cele**
>
> - Wyjaśnisz, czym są funkcje agregujące języka SQL.
> - Stworzysz zapytania z wykorzystaniem funkcji agregujących.
> - Wykorzystasz grupowanie danych.
> - Przeanalizujesz dane z wielu tabel.

# Przeczytaj

## Baza danych

W załączonych plikach znajdziesz bazę danych `Uczniowie` w formacie przeznaczonym dla programu LibreOffice Base (rozszerzenie `.odb`) oraz Microsoft Access (rozszerzenie `.accdb`). Pobierz odpowiedni plik i zapisz na dysku.

Baza danych `uczniowie.odb` dla LibreOffice Base.
Plik o rozmiarze 397.82 KB w języku polskim

Baza danych `uczniowie.accdb` dla Microsoft Access.
Plik o rozmiarze 868.00 KB w języku polskim

> **Ćwiczenie 1**
>
> W wybranym programie otwórz odpowiednią bazę. Wykorzystaj widok projektu tabeli oraz widok relacji, aby poznać schemat bazy danych.

> Podane dalej polecenia SQL wykonujemy po otwarciu projektu kwerendy w widoku SQL.

## Funkcje agregujące

Język SQL zawiera zestaw użytecznych funkcji agregujących, które służą do wykonywania obliczeń bądź innych operacji na grupach rekordów. Są to:

- `COUNT()` – zwraca liczbę rekordów spełniających podane kryteria;
- `AVG()` – zwraca średnią wartość wyliczoną dla pól zawierających liczby;
- `SUM()` – oblicza sumę dla pól zawierających liczby;
- `MIN()` – zwraca wartość minimalną zapisaną we wskazanych polach liczbowych i tekstowych;
- `MAX()` – zwraca wartość maksymalną zapisaną we wskazanych polach liczbowych i tekstowych.

Funkcji agregujących możemy używać między innymi do:

- wykonywania obliczeń na wynikach zwracanych przez inne klauzule,
- liczenia zgrupowanych rekordów,
- wyliczania średniej albo sumy wartości z wybranej kolumny,
- znajdowania wartości skrajnych we wskazanej kolumnie.

Wykorzystamy przedstawione funkcje do przeanalizowania informacji z przykładowej bazy.

Na początku sprawdźmy, ile kobiet i ilu mężczyzn zostało zapisanych w bazie. Informację na temat płci znajdziemy w tabeli *uczniowie* – jest to pole typu `BOOLEAN`, czyli przechowujące wartość `TRUE` (`PRAWDA, Tak`) lub `FALSE` (`FAŁSZ, Nie`). W naszym przypadku wartość `PRAWDA (Tak)` oznacza płeć żeńską.

Zacznijmy od kwerendy zliczającej wszystkie rekordy:

> **Przykład 1**
>
> ```
> SELECT COUNT(*)
> FROM uczniowie;
> ```
>
> Znak „*” jako argument funkcji `COUNT()` oznacza zliczanie wszystkich rekordów, w tym zawierających wartość `NULL`.

Wynikiem będzie liczba `156` oraz nagłówek pola `COUNT(*)` (LibreOffice Base) lub `Expr1000` (Microsoft Access).

Do poprzedniego zapytania dodamy warunek, dzięki któremu z tabeli wybrane zostaną rekordy, w których pole `plec` będzie miało wartość `PRAWDA`:

> **Przykład 2**
>
> ```
> SELECT COUNT(*) AS liczba_kobiet
> FROM uczniowie
> WHERE plec = TRUE;
> ```

Wynikiem będzie liczba `65` oraz nagłówek pola `liczba_kobiet` dzięki użyciu klauzuli `AS`, która pozwala podawać aliasy pól.

> **Ważne!**
>
> W zależności od systemu bazodanowego, a także od narzędzia, którego używamy do budowania zapytań, warunki nakładane na pola typu `BOOLEAN` można wyrażać w inny sposób, np.: `plec = 1` (działa w LibreOffice Base) lub `plec IS TRUE`.

Zliczać można również tylko wybrane pola, co jest przydatne np. wtedy, kiedy szukamy wartości unikalnych. Wtedy używamy klauzuli `DISTINCT`:

> **Przykład 3**
>
> Użycie klauzuli `DISTINCT` w LibreOffice Base:
>
> ```
> SELECT COUNT(DISTINCT imie) AS liczba_imion
> FROM uczniowie
> WHERE plec = TRUE;
> ```

Użycie klauzuli `DISTINCT` w Microsoft Access:

```
SELECT COUNT(*) AS liczba_imion
FROM
(SELECT DISTINCT imie FROM uczniowie WHERE plec = TRUE);
```

W Microsoft Access klauzula `DISTINCT` występuje w podzapytaniu, którego wyniki stają się źródłem dla kwerendy zliczającej.

Wynikiem kwerendy będzie wartość `31`, czyli liczba niepowtarzających się imion żeńskich zapisanych w tabeli.

Warunki służące do wskazania zliczanych wierszy można formułować nie tylko za pomocą klauzuli `WHERE`. Oto przykłady użycia instrukcji warunkowych:

**Przykład 4**

Użycie instrukcji `CASE warunek THEN wartość_prawda END` w LibreOffice Base:

```
SELECT
  COUNT(CASE WHEN plec = 1 THEN plec END) AS liczba_kobiet,
  COUNT(CASE WHEN plec = 0 THEN plec END) AS liczba_mezczyzn
FROM uczniowie;
```

Jeżeli warunek po klauzuli `WHEN` jest prawdziwy, zwracana jest podana `wartość_prawda`.

W programie Microsoft Access korzystamy z funkcji `IIF` lub `SWITCH` w podzapytaniu:

```
SELECT COUNT(lk) AS liczba_kobiet, COUNT(lm) AS liczba_mezczyzn
FROM
(SELECT IIf(plec = TRUE, 1) AS lk, IIf(plec = FALSE, 0) AS lm F
```

```
SELECT COUNT(lk) AS liczba_kobiet, COUNT(lm) AS liczba_mezczyzn
FROM
(SELECT SWITCH(plec = TRUE, 1) AS lk, SWITCH(plec = FALSE, 0) A
```

Ogólna składnia funkcji warunkowych dostępnych w Microsoft Access to:
`IIF (wyrażenie, wartość_prawda, wartość_fałsz)` oraz

`SWITCH (wyrażenie, wartość_prawda)`. Jeżeli wyrażenie jest prawdziwe, zwracana jest podana `wartość_prawda`.

W powyższych kwerendach jako `wartość_prawda` możemy podawać dowolne wartości, które mogą być później zliczone.

Wszystkie podane kwerendy zwrócą wyniki: `65` i `91`.
Informację o liczbie kobiet i mężczyzn można również uzyskać za pomocą klauzuli `GROUP BY`, która na początku grupuje rekordy (w tym wypadku według wartości prawda i fałsz), a później przekazuje je do funkcji agregującej:

```
SELECT plec, COUNT(plec) AS liczba
FROM uczniowie
GROUP BY plec;
```

Jeszcze innym sposobem uzyskania informacji dotyczącej liczby kobiet jest użycie funkcji `SUM()`. Ponieważ typ `BOOLEAN` jest podtypem liczbowym, możemy przekształcić go na liczbę i zsumować.

**Przykład 5**

W LO Base:

```
SELECT SUM(CAST (plec AS INT)) AS liczba_kobiet
FROM uczniowie;
```

W LO Base reprezentacją liczbową logicznej wartości `TRUE` jest wartość `1`. Instrukcja `CAST` pozwala przekształcać typy danych na inne, o ile to możliwe. W tym wypadku typ `BOOLEAN` zamieniamy na liczbę całkowitą, tj. typ `INT`.

W MS Access:

```
SELECT ABS(SUM(INT(plec))) AS liczba_kobiet
FROM uczniowie;
```

W MS Access reprezentacją liczbową logicznej wartości `TRUE` jest wartość `-1`. Funkcja `INT()` jest specyficzna dla MS Access. W podanym przykładzie można ją pominąć, ponieważ instrukcja `SUM(plec)` dokonuje konwersji automatycznie. Funkcja `ABS()` zwraca wartość bezwzględną liczby.

Teraz spróbujmy się dowiedzieć, jakie są średnie wyniki egzaminów z różnych przedmiotów:

**Przykład 6**

```
SELECT AVG(egz_hum), AVG(egz_mat), AVG(egz_jez)
FROM uczniowie;
```

Po wykonaniu kwerendy może się okazać, że dokładność wyników (liczba cyfr po przecinku) znacznie przekracza nasze potrzeby. Możemy jednak w prosty sposób ograniczyć długość części ułamkowej – użyjemy w tym celu funkcji `ROUND()`:

**Przykład 7**

```
SELECT ROUND(AVG(egz_hum), 2)
FROM uczniowie;
```

Oprócz wyniku średniego możemy wyświetlić wynik minimalny i maksymalny:

**Przykład 8**

```
SELECT MIN(egz_hum), ROUND(AVG(egz_hum), 2), MAX(egz_hum)
FROM uczniowie;
```

Dzięki funkcjom `MIN()` i `MAX()` wyszukamy też najlepsze i najgorsze oceny wśród uczniów wskazanej płci (w tym wypadku kobiet):

**Przykład 9**

```
SELECT MIN(egz_hum), MAX(egz_hum)
FROM uczniowie
WHERE plec = TRUE;
```

## Klauzula GROUP BY

Chcemy dowiedzieć się, jaka jest średnia ocen z wybranego przedmiotu we wskazanej klasie. W wynikach kwerendy chcemy zobaczyć nazwę przedmiotu, nazwę klasy i średnią ocen. Zacznijmy od kwerendy wybierającej odpowiednie pola z różnych tabel:

W LibreOffice Base napiszemy:

```
SELECT przedmioty.nazwa, klasy.nazwa, ocena
FROM oceny
INNER JOIN uczniowie ON uczniowie.id = oceny.id_ucznia
INNER JOIN przedmioty ON przedmioty.id = oceny.id_przedmiotu
```

```
INNER JOIN klasy ON klasy.id = uczniowie.id_klasy
WHERE przedmioty.nazwa='polski' AND klasy.nazwa='3A';
```

lub:

```
SELECT przedmioty.nazwa, klasy.nazwa, ocena
FROM oceny, przedmioty, uczniowie, klasy
WHERE oceny.id_przedmiotu = przedmioty.id
      AND oceny.id_ucznia = uczniowie.id
      AND uczniowie.id_klasy = klasy.id
      AND przedmioty.nazwa = 'polski'
      AND klasy.nazwa = '3A';
```

> Warunki złączeń w kwerendach wybierających dane z wielu tabel mogą być definiowane za pomocą klauzul `JOIN` lub `WHERE`.

W Microsoft Access składnia jest bardziej skomplikowana, musimy uwzględnić nawiasy po klauzuli `FROM`:

```
SELECT przedmioty.nazwa, klasy.nazwa, ocena
FROM (klasy INNER JOIN uczniowie ON klasy.id = uczniowie.id_klasy
     INNER JOIN (przedmioty INNER JOIN oceny ON przedmioty.id = o
WHERE przedmioty.nazwa="polski" AND klasy.nazwa="3A";
```

W powyższych kwerendach warto zauważyć, że pola zawierające nazwę przedmiotu i klasy mają taką samą nazwę, dlatego musieliśmy je poprzedzić nazwami tabel. Drugą ważną rzeczą jest uwzględnienie w warunkach złączeń tabeli *uczniowie*. Mimo że nie wybieramy z niej żadnego pola, tylko dzięki niej możemy odczytać oceny podanej klasy.

Kiedy mamy listę ocen, wystarczy je zgrupować według pól podanych w klauzuli `SELECT`, czyli nazwy przedmiotu i klasy, i użyć funkcji `AVG()` na polu `ocena`.

W LibreOffice Base:

```
SELECT przedmioty.nazwa, klasy.nazwa, ROUND(AVG(ocena), 2)
FROM oceny
INNER JOIN uczniowie ON uczniowie.id = oceny.id_ucznia
INNER JOIN przedmioty ON przedmioty.id = oceny.id_przedmiotu
INNER JOIN klasy ON klasy.id = uczniowie.id_klasy
WHERE przedmioty.nazwa='polski' AND klasy.nazwa='3A'
```

```
GROUP BY przedmioty.nazwa, klasy.nazwa;
```

W Microsoft Access:

```
SELECT przedmioty.nazwa, klasy.nazwa, ROUND(AVG(ocena), 2)
FROM (klasy INNER JOIN uczniowie ON klasy.id = uczniowie.id_klasy
WHERE przedmioty.nazwa="polski" AND klasy.nazwa="3A"
GROUP BY przedmioty.nazwa, klasy.nazwa;
```

Funkcja `ROUND()` zaokrągla podaną wartość do podanej liczby miejsc po przecinku, w tym wypadku dwóch. Wynik podanych kwerend to: `polski, 3A, 3,5`.

> **Ważne!**
>
> W zapytaniach grupujących dane pola, które wybieramy w klauzuli `SELECT` i które nie są objęte funkcjami agregującymi, muszą wystąpić również w klauzuli `GROUP BY`.

## Słownik

> **funkcje agregujące**
>
> funkcje umożliwiające wykonywanie obliczeń na grupach rekordów oraz wyszukiwanie i zliczanie rekordów spełniających określone warunki
> **instrukcja warunkowa**
>
> instrukcja, która pozwala nakładać warunki na wybierane rekordy; w zależności od systemu bazodanowego można korzystać z następujących funkcji:
> `CASE WHEN warunek1 THEN wynik1 ELSE wynik2 END` (mySQL, LibreOffice Base),
> `IFF (wyrażenie, wartość_jeżeli_prawda, wartość_jeżeli_fałsz)`,
> `SWITCH (wyrażenie, wartość)` (MicrosoftS Access)
> **podzapytanie**
>
> zapytanie `SELECT` umieszczone wewnątrz innego zapytania `SELECT`, podzapytanie wykonywane w pierwszej kolejności, a jego wyniki są źródłem dla zewnętrznego zapytania

# Prezentacja multimedialna

**Polecenie 1**

Poniżej zamieszczono bazę danych `Uczniowie` w wersji dla LibreOffice Base oraz Microsoft Access. Pobierz odpowiedni plik, otwórz w wybranej aplikacji i wykonaj w trybie SQL zapytania z poniższej prezentacji ilustrujące użycie funkcji agregujących i klauzuli `GROUP BY`.

Plik bazy `uczniowie.odb`
Plik o rozmiarze 397.82 KB w języku polskim

Plik bazy danych uczniowie.accdb
Plik o rozmiarze 936.00 KB w języku polskim



**Polecenie 2**

# Sprawdź się

Pokaż ćwiczenia:

- Ćwiczenie 1
- Ćwiczenie 2
- Ćwiczenie 3
- Ćwiczenie 4
- Ćwiczenie 5
- Ćwiczenie 6
- Ćwiczenie 7
- Ćwiczenie 8

# Dla nauczyciela

**Autor:** Robert Bednarz
**Przedmiot:** Informatyka

**Temat: Instrukcje grupowania w języku SQL, etap I**

**Grupa docelowa:**

Szkoła ponadpodstawowa, liceum ogólnokształcące, technikum, zakres rozszerzony

**Podstawa programowa:**

Cele kształcenia – wymagania ogólne

> II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

Treści nauczania – wymagania szczegółowe

> II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.
>
> > Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:
> >
> > > 4) przygotowując opracowania rozwiązań złożonych problemów, posługuje się wybranymi aplikacjami w stopniu zaawansowanym:
> > >
> > > > d) projektuje i tworzy relacyjną bazę złożoną z wielu tabel oraz sieciową aplikację bazodanową dla danych związanych z rozwiązywanym problemem, formułuje kwerendy, tworzy i modyfikuje formularze oraz raporty, stosuje język SQL do wyszukiwania informacji w bazie i do jej modyfikacji, uwzględnia kwestie integralności danych, bezpieczeństwa i ochrony danych w bazie,

**Kształtowane kompetencje kluczowe:**

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

**Cele operacyjne (językiem ucznia):**

- Wyjaśnisz, czym są funkcje agregujące języka SQL.
- Stworzysz zapytania z wykorzystaniem funkcji agregujących.
- Wykorzystasz grupowanie danych.
- Przeanalizujesz dane z wielu tabel.

**Strategie nauczania:**

- konstruktywizm;
- konektywizm.

**Metody i techniki nauczania:**

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- ćwiczenia praktyczne.

**Formy pracy:**

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

**Środki dydaktyczne:**

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda;
- oprogramowanie MySQL w wersji 8.0 (lub nowszej).

**Przebieg lekcji**

**Przed lekcją:**

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Instrukcje grupowania w języku SQL, etap I". Nauczyciel prosi uczniów o zapoznanie się z treściami w sekcji „Przeczytaj".

**Faza wstępna:**

1. Nauczyciel prosi wybraną osobę o odczytanie tematu lekcji, a następnie określa cele i kryteria sukcesu.
2. Prowadzący prosi uczniów, aby zgłaszali swoje propozycje pytań do tematu. Jedna osoba może zapisywać je na tablicy. Gdy uczniowie wyczerpią swoje pomysły, a pozostały jakieś ważne kwestie do poruszenia, nauczyciel je dopowiada.

**Faza realizacyjna:**

1. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Prezentacja multimedialna". Uczniowie wspólnie zapoznają się z jego treścią. Zapisują ewentualne problemy i pytania. Po czym następuje dyskusja, w trakcie której nauczyciel wyjaśnia niezrozumiałe treści.
2. **Ćwiczenie umiejętności.** Nauczyciel przechodzi do sekcji „Sprawdź się". Uczniowie indywidualnie rozwiązują ćwiczenia nr 1–7 na czas. Osoba, która poprawnie rozwiąże zadania jako pierwsza, wygrywa, a nauczyciel może nagrodzić ją oceną za aktywność.

**Faza podsumowująca:**

1. Na koniec zajęć nauczyciel raz jeszcze wyświetla na tablicy temat lekcji i cele zawarte w sekcji „Wprowadzenie". W odniesieniu do ich realizacji dokonuje szczegółowej oceny rozwiązania zastosowanego przez wybranego ucznia.
2. Wybrany uczeń podsumowuje zajęcia, zwracając uwagę na nabyte umiejętności, omawia ewentualne problemy podczas rozwiązania ćwiczeń.

**Praca domowa:**

1. Uczniowie wykonują ćwiczenie 8 z sekcji „Sprawdź się".

**Materiały pomocnicze:**

- Oficjalna dokumentacja techniczna dla systemu MySQL 8.0 (lub nowszej wersji).

**Wskazówki metodyczne:**

- Treści w sekcji „Prezentacja multimedialna" można wykorzystać jako materiał służący powtórzeniu materiału.