

Łączenie wyników zapytań w MySQL

W relacyjnych bazach danych często zachodzi potrzeba **łączenia wyników kilku zapytań SELECT w jeden zestaw wyników**. Operacje tego typu należą do tzw. **operacji zbiorowych (set operations)**, ponieważ działają na zbiorach rekordów podobnie jak operacje matematyczne na zbiorach.

W systemie **MySQL** do łączenia wyników zapytań stosuje się przede wszystkim operatory:

- UNION
- UNION ALL
- INTERSECT
- EXCEPT

Operatory te pozwalają zestawiać wyniki wielu zapytań w jeden wynik końcowy, pod warunkiem spełnienia kilku wymagań dotyczących struktury zapytań.

Warunki używania operacji zbiorowych

Aby możliwe było połączenie wyników zapytań za pomocą operatorów zbiorowych, zapytania muszą spełniać określone warunki:

1. **Ta sama liczba kolumn**
Każde zapytanie **SELECT** musi zwracać identyczną liczbę kolumn.
2. **Zgodność typów danych**
Typy danych odpowiadających sobie kolumn powinny być takie same lub kompatybilne.
3. **Kolejność kolumn**
Kolumny są dopasowywane **według pozycji**, a nie według nazw.

Przykład poprawnej struktury:

```
SELECT column1, column2  
FROM tabela1
```

UNION

```
SELECT column1, column2  
FROM tabela2;
```

Zastosowanie operacji zbiorowych

Operatory zbiorowe są szczególnie przydatne w sytuacjach, gdy:

- chcemy **połączyć wyniki z różnych tabel** w jeden zestaw danych,
- chcemy **porównać wyniki zapytań**,
- potrzebujemy znaleźć **część wspólną danych**,

- chcemy określić **różnice między zbiorami rekordów**.

W praktyce operacje te stosuje się m.in. do:

- integracji danych z wielu tabel,
- wyszukiwania powtarzających się wartości,
- filtrowania danych,
- analiz porównawczych w bazach danych.

Podsumowanie

Operacje zbiorowe w MySQL umożliwiają elastyczne łączenie wyników wielu zapytań `SELECT`. Każdy operator realizuje inny typ operacji na zbiorach danych:

Operator	Działanie
UNION	łączy wyniki zapytań i usuwa duplikaty
UNION ALL	łączy wyniki zapytań i pozostawia duplikaty
INTERSECT	zwraca część wspólną wyników
EXCEPT	zwraca różnicę zbiorów

1. UNION – połączenie wyników bez duplikatów

Przykład 1

Wyświetlenie nazw państw i nazw miast z Europy w jednym wyniku.

```
SELECT Name AS nazwa
FROM country
WHERE Continent = 'Europe'

UNION

SELECT Name AS nazwa
FROM city
WHERE CountryCode IN (
    SELECT Code
    FROM country
    WHERE Continent = 'Europe'
);
```

[Komentarz](#)

To zapytanie:

- pobiera nazwy państw z tabeli `country`, które należą do Europy,
- następnie pobiera nazwy miast z tabeli `city`, które znajdują się w krajach europejskich,
- łączy oba wyniki w jeden zbiór za pomocą `UNION`.

Operator `UNION` usuwa duplikaty, więc jeśli jakaś nazwa wystąpi zarówno jako kraj, jak i miasto, pojawi się tylko raz.

Przykład 2

Wyświetlenie nazw krajów afrykańskich i azjatyckich jako jednego zbioru.

```
SELECT Name
FROM country
WHERE Continent = 'Africa'
```

`UNION`

```
SELECT Name
FROM country
WHERE Continent = 'Asia';
```

[Komentarz](#)

Zapytanie zwraca jedną wspólną listę nazw państw z dwóch kontynentów: Afryki i Azji. Ponieważ użyto `UNION`, każdy kraj zostanie pokazany tylko jeden raz.

2. UNION ALL – połączenie wyników z duplikatami

Przykład 1

Wyświetlenie nazw państw i miast bez usuwania powtórzeń.

```
SELECT Name AS nazwa
FROM country
```

`UNION ALL`

```
SELECT Name AS nazwa
FROM city;
```

[Komentarz](#)

To zapytanie:

- pobiera wszystkie nazwy krajów,
- pobiera wszystkie nazwy miast,
- łączy je w jeden wynik.

Operator `UNION ALL` zachowuje wszystkie rekordy, także te powtarzające się. Jeżeli jakaś nazwa występuje kilka razy, wszystkie wystąpienia będą widoczne.

Przykład 2

Wyświetlenie populacji dużych krajów i dużych miast.

```
SELECT Name, Population, 'kraj' AS typ
FROM country
WHERE Population > 100000000
```

```
UNION ALL
```

```
SELECT Name, Population, 'miasto' AS typ
FROM city
WHERE Population > 10000000;
```

[Komentarz](#)

Zapytanie łączy:

- kraje o populacji większej niż 100 milionów,
- miasta o populacji większej niż 10 milionów.

Dodatkowa kolumna `typ` pozwala odróżnić, czy rekord dotyczy kraju czy miasta.

`UNION ALL` jest tu dobrym wyborem, ponieważ nie ma potrzeby usuwania duplikatów.

3. INTERSECT – część wspólna wyników

Przykład 1

Wyświetlenie nazw, które występują zarówno w tabeli `country`, jak i w tabeli `city`.

```
SELECT Name
FROM country
```

```
INTERSECT
```

```
SELECT Name
FROM city;
```

[Komentarz](#)

Zapytanie zwraca tylko te nazwy, które są wspólne dla obu tabel.

Oznacza to, że wynik zawiera wartości będące jednocześnie nazwami krajów i nazwami miast.

Przykład 2

Wyświetlenie nazw regionów, które występują także jako nazwy dystryktów.

```
SELECT Region AS nazwa  
FROM country
```

INTERSECT

```
SELECT District AS nazwa  
FROM city;
```

[Komentarz](#)

Tutaj porównywane są:

- wartości z kolumny `Region` w tabeli `country`,
- wartości z kolumny `District` w tabeli `city`.

Wynik pokaże nazwy wspólne dla obu zbiorów.

Odpowiednik `INTERSECT` dla starszego MySQL

Jeżeli wersja MySQL nie obsługuje `INTERSECT`, można zastosować `JOIN`:

```
SELECT DISTINCT c.Name  
FROM country c  
JOIN city ci  
      ON c.Name = ci.Name;
```

[Komentarz](#)

To zapytanie daje podobny efekt jak `INTERSECT`, ponieważ zwraca wspólne nazwy z obu tabel.

4. EXCEPT – różnica zbiorów

Przykład 1

Wyświetlenie nazw krajów, które nie są nazwami miast.

```
SELECT Name  
FROM country
```

EXCEPT

```
SELECT Name  
FROM city;
```

[Komentarz](#)

Zapytanie zwraca nazwy z pierwszego zbioru, które nie pojawiają się w drugim zbiorze. W tym przypadku będą to państwa, których nazwy nie występują jako nazwy miast.

Przykład 2

Wyświetlenie języków występujących jako oficjalne, ale nie jako nieoficjalne.

```
SELECT Language
FROM countrylanguage
WHERE IsOfficial = 'T'
```

EXCEPT

```
SELECT Language
FROM countrylanguage
WHERE IsOfficial = 'F';
```

[Komentarz](#)

To zapytanie pokazuje języki, które występują wyłącznie jako urzędowe i nie pojawiają się jako nieurzędowe.

Odpowiednik `EXCEPT` dla starszego MySQL

Jeżeli dana wersja MySQL nie obsługuje `EXCEPT`, można użyć `NOT EXISTS`:

```
SELECT c.Name
FROM country c
WHERE NOT EXISTS (
    SELECT 1
    FROM city ci
    WHERE ci.Name = c.Name
);
```

[Komentarz](#)

To rozwiązanie daje efekt podobny do `EXCEPT`, bo eliminuje rekordy obecne w drugiej tabeli.

Przykłady praktyczne z bardziej rozbudowanym wynikiem

Przykład 1 – miasta i stolicy

Wyświetlenie nazw stolic oraz wszystkich pozostałych miast w jednym zbiorze.

```

SELECT ci.Name AS nazwa_miasta, 'stolica' AS rodzaj
FROM city ci
JOIN country co
    ON ci.ID = co.Capital

UNION

SELECT ci.Name AS nazwa_miasta, 'inne miasto' AS rodzaj
FROM city ci
WHERE ci.ID NOT IN (
    SELECT Capital
    FROM country
    WHERE Capital IS NOT NULL
);

```

Komentarz

Wynik zawiera:

- miasta będące stolicami,
- miasta, które nie są stolicami.

W tym przykładzie zastosowano `UNION`, więc identyczne rekordy nie będą się powtarzać.

Przykład 2 – porównanie języków w Europie i Azji

```

SELECT Language
FROM countrylanguage
WHERE CountryCode IN (
    SELECT Code
    FROM country
    WHERE Continent = 'Europe'
)

INTERSECT

SELECT Language
FROM countrylanguage
WHERE CountryCode IN (
    SELECT Code
    FROM country
    WHERE Continent = 'Asia'
);

```

Komentarz

Zapytanie wyszukuje języki, które występują zarówno w krajach europejskich, jak i azjatyckich.

Przykład 3 – kraje bez dużych miast

```
SELECT Name  
FROM country
```

EXCEPT

```
SELECT DISTINCT co.Name  
FROM country co  
JOIN city ci  
      ON co.Code = ci.CountryCode  
WHERE ci.Population > 5000000;
```

[Komentarz](#)

Wynik pokazuje kraje, które nie mają miasta o populacji większej niż 5 milionów mieszkańców.

Podsumowanie: Połączenie tabel a operacje zbiorowe

Warto pamiętać, że operacje takie jak UNION, INTERSECT i EXCEPT różnią się od JOIN.

- JOIN służy do **łączenia tabel poziomo**, czyli dołączania kolumn.
- operatory zbiorowe służą do **łączenia wyników pionowo**, czyli dokładania rekordów jeden pod drugim albo porównywania całych zestawów wyników.