

# Lekcja

## Temat: Zaawansowane zapytania JOIN

```
DROP TABLE IF EXISTS zamowienia;  
DROP TABLE IF EXISTS klienci;
```

```
CREATE TABLE klienci (  
  id INT PRIMARY KEY,  
  imie VARCHAR(50)  
);
```

```
CREATE TABLE zamowienia (  
  id INT PRIMARY KEY,  
  id_klienta INT ,  
  produkt VARCHAR(50),  
  FOREIGN KEY (id_klienta) REFERENCES klienci(id)  
);
```

```
INSERT INTO klienci (id, imie) VALUES  
(1, 'Anna'),  
(2, 'Jan'),  
(3, 'Ola'),  
(4, 'Piotr');
```

```
INSERT INTO zamowienia (id, id_klienta, produkt) VALUES  
(1, 1, 'Laptop'),  
(2, 1, 'Myszka'),  
(3, 2, 'Telefon'),  
(4, null, 'Monitor'); -- ten klient (id=5) nie istnieje w tabeli klienci
```

### ● INNER JOIN

```
SELECT k.imie, z.produkt  
FROM klienci k  
INNER JOIN zamowienia z ON k.id = z.id_klienta;
```

| imie | produkt |
|------|---------|
| Anna | Laptop  |
| Anna | Myszka  |
| Jan  | Telefon |

## ● LEFT JOIN

```
SELECT k.imie, z.produkt
FROM klienci k
LEFT JOIN zamowienia z ON k.id = z.id_klienta;
```

| imie  | produkt |
|-------|---------|
| Anna  | Laptop  |
| Anna  | Myszka  |
| Jan   | Telefon |
| Ola   | NULL    |
| Piotr | NULL    |

## ● RIGHT JOIN

```
SELECT k.imie, z.produkt
FROM klienci k
RIGHT JOIN zamowienia z ON k.id = z.id_klienta;
```

| imie | produkt |
|------|---------|
| Anna | Laptop  |
| Anna | Myszka  |
| Jan  | Telefon |
| NULL | Monitor |

## ● FULL JOIN (symulowany)

```
SELECT k.imie, z.produkt
FROM klienci k
LEFT JOIN zamowienia z ON k.id = z.id_klienta
```

UNION

```
SELECT k.imie, z.produkt
FROM klienci k
RIGHT JOIN zamowienia z ON k.id = z.id_klienta;
```

| imie | produkt |
|------|---------|
|------|---------|

|       |         |
|-------|---------|
| Anna  | Laptop  |
| Anna  | Myszka  |
| Jan   | Telefon |
| Ola   | NULL    |
| Piotr | NULL    |
| NULL  | Monitor |

```
DROP TABLE IF EXISTS zamowienia;
DROP TABLE IF EXISTS produkty;
DROP TABLE IF EXISTS sklepy;
```

```
CREATE TABLE sklepy (
  id INT PRIMARY KEY,
  nazwa VARCHAR(50)
);
```

```
CREATE TABLE produkty (
  id INT PRIMARY KEY,
  nazwa VARCHAR(50),
  id_sklepu INT,
  FOREIGN KEY (id_sklepu) REFERENCES sklepy(id)
);
```

```
CREATE TABLE zamowienia (
  id INT PRIMARY KEY,
  id_produkту INT,
  ilosc INT,
  FOREIGN KEY (id_produkту) REFERENCES produkty(id)
);
```

```
INSERT INTO sklepy (id, nazwa) VALUES
(1, 'Sklep A'),
(2, 'Sklep B'),
(3, 'Sklep C');
```

```
INSERT INTO produkty (id, nazwa, id_sklepu) VALUES
(1, 'Laptop', 1),
(2, 'Myszka', 1),
(3, 'Monitor', 2),
(4, 'Klawiatura', 3);
```

```
INSERT INTO zamowienia (id, id_produkту, ilosc) VALUES
(1, 1, 5),
(2, 1, 3),
(3, 2, 10),
(4, 3, 2);
```

### Zestawienie sklepów i produktów, łącznie z tymi, dla których nie odnotowano zamówień:

```
SELECT s.nazwa AS sklep,  
       p.nazwa AS produkt,  
       COALESCE(SUM(z.ilosc), 0) AS sprzedane_sztuki  
FROM sklepy s  
LEFT JOIN produkty p ON p.id_sklepu = s.id  
LEFT JOIN zamowienia z ON z.id_produktu = p.id  
GROUP BY s.id, p.id  
ORDER BY s.id, sprzedane_sztuki DESC;
```

| sklep   | produkt    | sprzedane_sztuki |
|---------|------------|------------------|
| Sklep A | Myszka     | 10               |
| Sklep A | Laptop     | 8                |
| Sklep B | Monitor    | 2                |
| Sklep C | Klawiatura | 0                |

#### Wyjaśnienie:

- ☐ LEFT JOIN produkty → bierzemy wszystkie sklepy, nawet jeśli nie mają produktów.
- ☐ LEFT JOIN zamowienia → bierzemy wszystkie produkty, nawet jeśli nie mają zamówień.
- ☐ SUM(z.ilosc) → sumujemy liczbę sprzedanych sztuk dla każdego produktu.
- ☐ COALESCE(..., 0) → jeśli produkt nie ma zamówień, pokazujemy 0 zamiast NULL.
- ☐ GROUP BY s.id, p.id → agregujemy dane po sklepie i produkcie.
- ☐ ORDER BY s.id, sprzedane\_sztuki DESC → sortujemy dane po sklepie i liczbie sprzedanych sztuk.

### Pokazuje wszystkie zamówienia, nawet jeśli nie ma dopasowanego produktu lub sklepu:

```
SELECT s.nazwa AS sklep,  
       p.nazwa AS produkt,  
       z.ilosc AS sprzedane_sztuki  
FROM sklepy s  
RIGHT JOIN produkty p ON p.id_sklepu = s.id  
RIGHT JOIN zamowienia z ON z.id_produktu = p.id  
GROUP BY s.id, p.id;
```

| sklep   | produkt | sprzedane_sztuki |
|---------|---------|------------------|
| Sklep A | Laptop  | 3                |
| Sklep A | Laptop  | 5                |
| Sklep A | Myszka  | 10               |
| Sklep B | Monitor | 2                |

**Wyjaśnienie:**

- ☐ **RIGHT JOIN** produkty p ON p.id\_sklepu = s.id → bierzemy wszystkie produkty, nawet jeśli nie mają sklepu.
- ☐ **RIGHT JOIN** zamówienia z ON z.id\_produktu = p.id → bierzemy wszystkie zamówienia, nawet jeśli nie mają przypisanego produktu.
- ☐ Jeśli w tabeli produkty lub sklepy brakuje dopasowania → kolumny będą **NULL**.