SQL Cheat Sheet Advanced

sklep.ninjadata.pl

Podzapytania

Pobiera produkty, które mają cenę wyższą niż średnia cena wszystkich produktów — WHERE

```
SELECT Nazwa, Cena
FROM Produkty
WHERE Cena > (SELECT AVG(Cena) FROM Produkty);
```

Oblicza średnią sprzedaż dla każdego produktu i wyświetla tylko te produkty, które sprzedają się lepiej niż 100 sztuk — FROM

```
SELECT Nazwa, SredniaSprzedaz
FROM (
    SELECT Produkt AS Nazwa, AVG(Ilosc) AS SredniaSprzedaz
    FROM Zamowienia
    GROUP BY Produkt
) AS Podzapytanie
WHERE SredniaSprzedaz > 100;
```

Wyświetla listę produktów z sumą zamówionych ilości oraz najwcześniejszą datą zamówienia dla każdego produktu — SELECT

```
SELECT Produkt,
       SUM(Ilosc) AS LacznaIlosc,
       (SELECT MIN(DataZamowienia)
        FROM Zamowienia z
        WHERE z.Produkt = Zamowienia.Produkt) AS DataZamowienia
FROM Zamowienia
GROUP BY Produkt;
```

Wyświetla produkty, które są droższe niż średnia cena w ich kategorii — Skorelowane podzapytanie

```
SELECT Nazwa, Cena, Kategoria
FROM Produkty AS P1
WHERE Cena > (SELECT AVG(Cena)
              FROM Produkty AS P2
              WHERE P1.Kategoria = P2.Kategoria);
```

Wyświetla klientów, którzy złożyli przynajmniej jedno zamówienie — EXISTS

```
SELECT Klient
FROM Klienci k
WHERE EXISTS (
    SELECT 1
    FROM Zamowienia z
    WHERE z.Klient_ID = k.ID
);
```

Wyświetla klientów, którzy nie złożyli przynajmniej jedno zamówienie — NOT EXISTS

```
SELECT Klient
FROM Klienci k
WHERE NOT EXISTS (
    SELECT 1
    FROM Zamowienia z
    WHERE z.Klient_ID = k.ID
);
```

Wyświetla produkty zamawiane przez klientów z Polski — IN

```
SELECT Nazwa
FROM Produkty
WHERE ID IN (
    SELECT Produkt_ID
    FROM Zamowienia
    WHERE Kraj = 'Polska'
);
```

Funkcje okna

Nadaje rangę produktom w każdej kategorii na podstawie sprzedaży – produkty o tej samej sprzedaży otrzymują tę samą rangę, a następne miejsce jest pomijane — RANK

```
SELECT
      Produkt, Kategoria, Sprzedaz,
      RANK() OVER (PARTITION BY Kategoria ORDER BY Sprzedaz DESC) AS Ranga
FROM Produkty;
```

Nadaje rangę produktom w każdej kategorii na podstawie sprzedaży, ale nie pomija miejsc w przypadku remisów — DENSE_RANK

```
SELECT
      Produkt, Kategoria, Sprzedaz,
      DENSE_RANK() OVER (PARTITION BY Kategoria ORDER BY Sprzedaz DESC) AS
GestaRanga
FROM Produkty;
```

Przypisuje unikalny numer każdemu produktowi w ramach kategorii, niezależnie od remisów — ROW_NUMBER

```
SELECT
      Produkt, Kategoria, Sprzedaz,
      ROW_NUMBER() OVER (PARTITION BY Kategoria ORDER BY Sprzedaz DESC) AS
NumerWiersza
FROM Produkty;
```

Dzieli produkty na cztery grupy (kwartyle) na podstawie sprzedaży w każdej kategorii — NTILE

```
SELECT
      Produkt, Kategoria, Sprzedaz,
      NTILE(4) OVER (PARTITION BY Kategoria ORDER BY Sprzedaz DESC) AS
Kwartyla
FROM Produkty;
```

Oblicza łączną sprzedaż w każdej kategorii dla każdego produktu — SUM

```
SELECT
      Produkt, Kategoria, Sprzedaz,
      SUM(Sprzedaz) OVER (PARTITION BY Kategoria) AS LaczaSprzedazKategorii
FROM Produkty;
```

Wyświetla sprzedaż poprzedniego produktu w każdej kategorii — LAG

```
SELECT
      Produkt, Kategoria, Sprzedaz,
      LAG(Sprzedaz) OVER (PARTITION BY Kategoria ORDER BY Sprzedaz DESC) AS
PoprzedniaSprzedaz
FROM Produkty;
```

Wyświetla sprzedaż następnego produktu w każdej kategorii — LEAD

```
SELECT
       Produkt, Kategoria, Sprzedaz,
       LEAD(Sprzedaz) OVER (PARTITION BY Kategoria ORDER BY Sprzedaz DESC)
AS NastepnaSprzedaz
FROM Produkty;
```

Wyrażenia regularne

Zamiana wszystkich wystąpień słowa 'stary' na 'nowy' w kolumnie 'Opis' — REGEX_REPLACE

```
SELECT
      REGEXP_REPLACE(Opis, 'stary', 'nowy') AS NowyOpis
FROM Produkty;
```

Wyciąganie pierwszego słowa z tekstu w kolumnie 'Opis' — REGEX_SUBSTR

```
SELECT
      REGEXP_SUBSTR(Opis, '[A-Za-z]+') AS PierwszeSlowo
FROM Produkty;
```

Złożone transformacje danych

Łączna sprzedaż w poszczególnych regionach dla każdego produktu — PIVOT

```
SELECT *
FROM (
    SELECT Produkt, Region, Sprzedaz
    FROM Sprzedaz
) AS Dane
PIVOT (
    SUM(Sprzedaz) FOR Region IN ([Polska], [Niemcy], [Francja])
) AS RegionySprzedaz;
```

Zamiana tabeli z agregatami sprzedaży na dane wierszowe — UNPIVOT

```
SELECT *
FROM (
      SELECT Produkt, [Polska], [Niemcy], [Francja]
      FROM RegionySprzedaz
) AS Dane
UNPIVOT (
    Sprzedaz FOR Region IN ([Polska], [Niemcy], [Francja])
) AS SprzedazRegiony;
```

Najlepsze produkty według sprzedaży z użyciem CTE — CTE

```
WITH Sprzedaz AS (
 SELECT Produkt, SUM(Ilosc) AS SumaSprzedazy
 FROM Zamówienia
 GROUP BY Produkt
),
Najlepsze AS (
 SELECT Produkt, SumaSprzedazy
 FROM Sprzedaz
 WHERE SumaSprzedazy > 1000
)
SELECT Produkt, SumaSprzedazy FROM Najlepsze;
```

Zaawansowane grupowanie danych

Generuje podsumowania w sposób hierarchiczny, zaczynając od szczegółowych danych (np. sprzedaż w regionie i kategorii), a następnie dodaje podsumowania dla regionów i na końcu sumę całkowitą dla wszystkich danych — ROLLUP

```
SELECT
      Region, Kategoria, SUM(Sprzedaz) AS SumaSprzedazy
FROM Produkty
GROUP BY ROLLUP(Region, Kategoria);
```

Generuje wszystkie możliwe podsumowania – dla każdej kategorii, każdego regionu, łącznie dla wszystkich kategorii i regionów, oraz sumę ogólną — CUBE

```
SELECT
      Region, Kategoria, SUM(Sprzedaz) AS SumaSprzedazy
FROM Produkty
GROUP BY CUBE(Region, Kategoria);
```

Pozwala wybrać konkretne podsumowania, np. tylko dla regionu, tylko dla kategorii lub sumę całkowitą — GROUPING SETS

```
SELECT
      Region, Kategoria, SUM(Sprzedaz) AS SumaSprzedazy
FROM Produkty
GROUP BY GROUPING SETS ((Region, Kategoria), (Region), (Kategoria), ());
```