SQL ZADANIA

Zadanie 1

Pracownik systemu potrzebuje sprawdzić wszystkie zamówienia złożone przez klientów. Wypisz nazwy klientów i identyfikatory zamówień.

```
SELECT customername, orderid
FROM Customers INNER JOIN Orders ON
customers.customerid=orders.customerid;

lub

SELECT customers.customername, orders.orderid
FROM Customers,Orders
WHERE customers.customerid=orders.customerid;
```

Zadanie 2

Wyświetl listę produktów, które zostały zamowione, wraz z identyfikatorami zamówień oraz

- połącz tabele Products i OrderDetails na postawie productID
- połącz wynik z tabelą Orders na podstawie OrderID

```
SELECT products.productname, orderdetails.OrderID
FROM (Products INNER JOIN Orderdetails ON
products.productID=Orderdetails.productID)
INNER JOIN Orders ON Orders.orderID=Orderdetails.orderID;
```

Zadanie 3

Kierownik chce zobaczyć lite klientów którzy jeszcze nie złozyli zamówienia:

- połącz tabele Customers and Orders , używając left Join aby uwzględnić wszystkich klientów którzy nie mają zamówień
- filtruj wynik aby wyświetlić klientów którzy nie maja zamowień (OrderID is NULL)

```
SELECT customers.customername, orders.orderID
FROM Customers LEFT JOIN Orders ON customers.customerID=Orders.customerID
WHERE orders.orderID is NULL;
```

Zadanie 4

Manager chce wiedzieć , którzy pracownicy obsługiwali konkretne zamówienia:

- połącz tabele Orders and Employees na podstawie EmployeeID, aby uzyskać informacje o zamówienniach i pracownikach , którzy je obsługiwali.

```
SELECT employees.lastname, employees.firstname, orders.orderID
FROM Orders INNER JOIN Employees ON
orders.employeeID=Employees.employeeID
```

Zadanie 5

Potrzebujesz wyświetlić liste wszystkich produktów , które zostały zamówione więcej niż raz.

- Połącz tabele Products i Orderdetails na podstawie ProductID za pomocą INNER JOIN
- Grupuj wynik według nazwy produktu.
- Filtruj wyniki aby wyświetlić tylko produkty zamówione więcej niż raz, używając Having.

```
SELECT products.productname, COUNT(orderdetails.quantity)
FROM Products INNER JOIN Orderdetails ON
products.productID=Orderdetails.productID
GROUP BY products.productname
HAVING COUNT(orderdetails.quantity)>1
```

Zadanie 6

Pracownik systemu potrzebuje sprawdzic wszystkie zamówienia złozone przez klientów. Wypisz nazwy klientów i identyfikatory ich zamówień.

```
SELECT customers.customername, orders.orderID
FROM Customers INNER JOIN Orders ON
orders.customerID=customers.customerID;
```

Zadanie 7

Pracownik systemu potrzebuje sprawdzic wszystkie zamówienia złozone przez klientów, nawet tych którzy nie złożyli jeszcze ani jednego zamówienia. Wypisz nazwy klientów i identyfikatory ich zamówień.

W tym przypadku użyje LEFT JOIN wynik zwróci wszystkich

klientów z lewej tabeli i dopasowania z prawej wraz z NULL jeśli nie było zamówienia

```
SELECT customers.customername, orders.orderID
FROM Customers LEFT JOIN Orders ON
orders.customerID=customers.customerID;
```

Użycie **RIGHT JOIN** zwróciło ny nam wszystkie zamówienia bez przypisanych klientów jest to błąd lub dane historyczne. W takim wypadku wynik zwróci wszystkie zamówienia po prawej i po lewej klientów dopasowanych jeśli będzie błąd po lewej wystąpi „NULL”

Zadanie 8

Wyświetl wszystkie zamówienia, których identyfikatory znajdują się na liście: 10250,10254,10334,10384.

```
SELECT*
FROM Orders
WHERE OrderID IN (10250, 10254, 10334, 10384);
```

Zadanie 9

Wyświetl wszystkie produkty, których ceny należą do listy wartośći: 10,20,30.

```
SELECT productname, price
FROM Products
WHERE Price IN (10,20,30);
```

Zadanie 10

Pokaż wszystkich klientów, którzy mieszkają w jednym z wymienionych miast: Berlin, Paris, London.

```
SELECT customername, city
FROM Customers
WHERE City IN ('Berlin', 'Paris', 'London');
```

Zadanie 11

Wyświetl wszystkie zamówienia złożone przez klientów, którzy mieszkają w 'London'

```
SELECT orders.orderID, customers.customername,customers.city
FROM Orders INNER JOIN Customers ON
orders.customerID=customers.customerID
WHERE customers.City= 'London';
```

Zadanie 12

Pokaż nazwę poduktu i jego cene dla wszystkich produktów , których cena jest wieksza niż 50.

```
SELECT productname, price
FROM Products
WHERE price>50
```

Zadanie 13

Wyświetl listę klientów, którzy nie mają jeszcze przypisanego sprzedawcy (osoba do kontaktu to osoba zajmująca się sprawami danego klienta)

```
SELECT customername,Contactname
FROM Customers
WHERE Contactname is NULL;
```

Zadanie 14

Potrzebujesz wyświetlić listę wszystkich produktów które, zostały zamówione więcej niż raz.

- Polącz tabele Orderdetails i Products za pomocą klucza
- Grupuj wyniki według nazwy produktu
- Filtruj wyniki za pomocą HAVING

```
SELECT products.productname, COUNT(orderdetails.productID) AS result
FROM Products
INNER JOIN orderdetails ON products.productID=orderdetails.productID
GROUP BY products.productname
HAVING COUNT(orderdetails.productID)>1;
```

Zadanie 15

Potrzebujesz wyświetlić lite zamówień i nazw firm przewozowych, które obsługują te zamówienia.

```
SELECT orders.orderID, shippers.shipperName
FROM Orders INNER JOIN Shippers ON orders.shipperid=shippers.shipperid;
```

Zadanie 16

Chcemy zobaczyć wszystkie zamówienia nawet te, które nie mają przypisanej firmy przewozowej (nawiązując do zadania 15) aby zidentyfikowaći naprawić potencjalne błędy w bazie danych.

```
SELECT orders.orderID, shippers.shipperName
FROM Orders LEFT JOIN Shippers ON orders.shipperid=shippers.shipperid;
```

LEFT JOIN – zwróci nam wyniki z lewej tabeli i dopasowania z prawej , jęśli nie będzie dopasowań to po prawej stronie będzie przypisana wartość „NULL”

a jeśli chcielibyśmy pogrupować te wyniki ? I aby dla każdej firmy przewozowej policzyć liczbę zamówień?

```
SELECT COUNT(orders.orderID) AS total, shippers.shipperName
FROM Orders LEFT JOIN Shippers ON orders.shipperid=shippers.shipperid
GROUP BY shippers.shipperName
```

Zadanie 17

Znajdz klientów z barkującymi danymi. Napisz zapyatnie, które znajdzie klientów, dla których brakuje danych Contactname, Country, Postalcode and City.

```
SELECT customername, customerID, Contactname, Country, Postalcode, City
FROM Customers
WHERE
Contactname IS NULL OR
Country IS NULL OR
PostalCode IS NULL OR
City IS NULL;
```

- policz klientów z brakującymi danymi ?

```
SELECT COUNT(*) AS Total
FROM Customers
WHERE
Contactname IS NULL OR
Country IS NULL OR
PostalCode IS NULL OR
City IS NULL;
```

Zadanie 18

Weryfikacja zamówienia z dużymi kwotami.

- Użyj tabeli Orders and OrderDetails
- Napisz zapytanie, które obliczy całkowitą wartość zamówienia (Quantity) dla każdego OrderID.
- Znajdz zamówienia, których wartość przekracza 500.

```
SELECT SUM(orderdetails.quantity) AS total_Quantity,orders.OrderID
FROM Orders INNER JOIN OrderDetails ON
orders.orderID=orderdetails.OrderID
GROUP BY orders.OrderID
HAVING SUM(orderdetails.quantity)> 500;
```

Zadanie 19

Znalezienie klientów z nieprawidłowym rozkładem zamówień

- znajdz klientów, którzy złożyli zamówienia w mniej niż 2 różnych miesiącach.

```
SELECT (DISTINCT EXTRACT (MONTH FROM OrderDate)),customers.customerName,
OrderID
FROM Orders INNER JOIN Customers ON
orders.customerID=customers.customerID
GROUP BY customers.customerName, orderID, OrderDate
HAVING (DISTINCT EXTRACT (MONTH FROM OrderDate)) < 2;

lub

SELECT COUNT(DISTINCT MONTH(orders.OrderDate)) AS uniqmonth,
customers.customerName
FROM Orders INNER JOIN Customers ON
orders.customerID=customers.customerID
GROUP BY customers.customerName, orders.customerID
HAVING COUNT(DISTINCT MONTH(orders.OrderDate))<2;

Uwaga:funcja DISTINCT (*) EXTRACT nie będzie obsługiwana z windows
database
```

Zadanie 20

Analiza aktywnośći klientów.

- Znajdz klientów, którzy złozyli mniej niż 3 zamówienia w systemie
- dla kazdego z nich wyswietl nazwę klienta, kraj i liczbę zamówień.
- Wykorzystaj COUNT(), i grupowanie (GROUP BY)

```
SELECT customers.customerName, customers.customerID, customers.Country,
COUNT(orders.orderID) AS order_result
FROM Customers LEFT JOIN Orders ON customers.customerID=orders.customerID
GROUP BY customers.customerName, customers.customerID, customers.Country
HAVING COUNT(orders.orderID) < 3;
```

Zadanie 21

Walidacja poprawnośći danych w zamówieniach.

- Znajdz wszystkie zamówienia, które nie mają żadnych szczegółów w tabeli **Orderdetails.**
- Wyświetl ich orderid i customerid.
- Wykorzystaj LEFT JOIN lub NOT EXISTS.

```
SELECT orders.orderID, orders.CustomerID
FROM Orders LEFT JOIN Orderdetails ON orders.orderID=orderdetails.orderID
WHERE orderdetails.orderID IS NULL
```

Zadanie 22

Znajdz najlepszych klientów, którzy złozyli zamówienia o łącznej wartośći większej niż 10 000.

- wyświetl nazwę klienta i sumę ich zamówień.
- Połącz 3 tabele

```
SELECT customers.customerName, orderdetails.productID,
SUM(Quantity) AS TOTAL
FROM (Customers INNER JOIN Orders ON
customers.customerID=orders.customerID)
INNER JOIN Orderdetails ON orders.orderID=orderdetails.orderID
GROUP BY customers.customerName,orderdetails.productID
HAVING SUM(Quantity) > 10000;
```

Zadanie 23

Znajdz produkty z najwyższym dochodem, pokazując ich nazwę oraz całkowity dochód (ilośćxcena).

- Posortuj wyniki malejąco według dochodu

```
SELECT products.productname, SUM(Price*Quantity) AS total_price
FROM Products INNER JOIN Orderdetails ON
products.productID=orderdetails.productID
GROUP BY products.productname
ORDER BY SUM(Price*Quantity) DESC
```

Zadanie 24

Wyświetl listę klientów i przydziel ich do kategori według złożonych zamówień.

- Zero zamówień to „brak zamówień”
- 1-5 licza zamówień to „mało zamówień”
- więcej niż 5 zamówień to „dużo zamówień”

```
SELECT customer.customerID,customers.customername, COUNT(orders.orderID)
AS Ordercount,
CASE
WHEN COUNT(orders.orderID) = 0 THEN 'brak zamowień'
WHEN COUNT(orders.orderID) BETWEEN 1 AND 5 THEN 'mało zamowien'
ELSE 'dużo zamówień'
END AS OrderCategory
FROM Customers LEFT JOIN Orders ON customers.customerID=orders.customerID
GROUP BY customer.customerID, customers.customername;
```