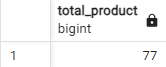
**NORTHWIND SQL SORGULARI**

**--- ÜRÜN-STOK ANALİZİ ---**

**-- Toplam Ürün Sayısı --**

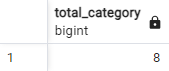
SELECT

COUNT(product\_id) AS total\_product

FROM

products;

**-- Toplam Kategori Sayısı --**

SELECT

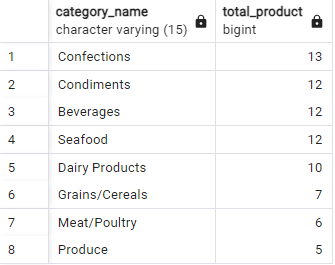
COUNT(category\_id) AS total\_category

FROM

categories;

**-- Kategorilere Göre Toplam Ürün Sayısı --**

SELECT

 category\_name,

COUNT(product\_id) AS total\_product

FROM

products p

JOIN

categories ca ON p.category\_id = ca.category\_id

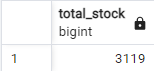
GROUP BY

category\_name

ORDER BY

total\_product DESC;

**-- Toplam Stok Miktarı --**

SELECT

SUM(unit\_in\_stock) AS total\_stock

FROM

products;

**--** **Toplam Siparişteki Ürün Sayısı --**

A screenshot of a computer

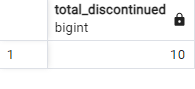
Description automatically generatedSELECT

SUM(unit\_on\_order) AS total\_unit\_on\_order

FROM

products;

**-- Toplam Üretilmeyen Ürünlerin Sayısı --**

SELECT

SUM(discontinued) AS total\_discontinued

FROM

products;

**-- Ürün Stok Durumu --**

SELECT

product\_name,

unit\_in\_stock,

reorder\_level,

CASE

WHEN unit\_in\_stock > reorder\_level THEN 'No Need'

ELSE 'Needed'

END AS stock\_status

FROM

products;

A screenshot of a computer

Description automatically generated

**--Ülke Bazında Ürün Sayısı--**

SELECT

c.country,

 COUNT(d.product\_id) AS total\_products

FROM

order\_details d

JOIN

orders o ON d.order\_id = o.order\_id

JOIN

customers c ON o.customer\_id = c.customer\_id

GROUP BY

c.country

ORDER BY

total\_products DESC;

**-- Stok Devir Oranı --**

SELECT

p.product\_name,

SUM(d.quantity) AS total\_sold,

ROUND(AVG(p.unit\_in\_stock)::NUMERIC, 2) AS average\_stock,

ROUND((SUM(d.quantity) / AVG(p.unit\_in\_stock))::NUMERIC, 2) AS inventory\_turnover

FROM

products p

JOIN

order\_details d ON p.product\_id = d.product\_id

WHERE

p.unit\_in\_stock <> 0

GROUP BY

product\_name;

A screenshot of a data

Description automatically generated

SELECT

ca.category\_name,

p.product\_name,

SUM(p.unit\_in\_stock) AS total\_stock,

SUM(d.quantity) AS total\_quantity,

SUM(p.reorder\_level) AS total\_reorder\_level,

ROUND(SUM(d.unit\_price)::NUMERIC, 2) AS total\_unit\_price,

SUM(d.discount) AS total\_discount

FROM

categories ca

JOIN

products p ON ca.category\_id = p.category\_id

JOIN

order\_details d ON p.product\_id = d.product\_id

WHERE

p.unit\_in\_stock <= p.reorder\_level

GROUP BY

ca.category\_name,

p.product\_name

ORDER BY

total\_stock DESC;

A screenshot of a computer

Description automatically generated

**--- ÇALIŞAN PERFORMANS ANALİZİ ---**

**-- Toplam Çalışan Sayısı --**

A screenshot of a computer

Description automatically generatedSELECT

COUNT(employee\_id) AS total\_emloyees

FROM

employees;

**-- Toplam Ülke Sayısı --**

A screenshot of a computer

Description automatically generatedSELECT

COUNT(country) AS country

FROM

employees;

**-- Çalışanlara ait Toplam Sipariş, İndirim ve İndirimsiz Satış Tutarları ve Ortalama İndirim Yüzdeleri--**

SELECT

CONCAT(e.first\_name, ' ', e.last\_name) AS employee\_full\_name,

e.title,

e.country,

COUNT (DISTINCT o.order\_id) AS total\_orders,

ROUND(SUM(od.quantity \* od.unit\_price):: NUMERIC, 2) AS total\_sale\_excluding\_discount,

ROUND(SUM((1 - od.discount)\*(od.quantity \* od.unit\_price))::NUMERIC, 2) AS total\_sale\_including\_discount,

ROUND(AVG(od.discount \* 100)::NUMERIC, 2) AS average\_discount\_percentage

FROM

employees e

INNER JOIN

orders o ON o.employee\_id = e.employee\_id

INNER JOIN

order\_details od ON o.order\_id = od.order\_id

INNER JOIN

products p ON od.product\_id = p.product\_id

GROUP BY

employee\_full\_name,

e.title,

e.country

ORDER BY

total\_sale\_excluding\_discount DESC;

A screenshot of a computer

Description automatically generated

**-- Çalışanların Prim Hakkediş Durumları --**

SELECT

CONCAT(e.first\_name, ' ', e.last\_name) AS employee\_full\_name,

e.title,

COUNT (DISTINCT o.order\_id) AS total\_orders,

CASE

WHEN COUNT (DISTINCT o.order\_id)> 120 THEN 'premium payment'

ELSE 'no premium payment'

END AS premium\_status

FROM

employees e

INNER JOIN

orders o ON o.employee\_id = e.employee\_id

INNER JOIN

order\_details od ON o.order\_id = od.order\_id

GROUP BY

employee\_full\_name,

e.title

ORDER BY

total\_orders DESC;

A screenshot of a computer

Description automatically generated

**--- MÜŞTERİ ANALİZİ ---**

A screenshot of a computer

Description automatically generated**-- Toplam Müşteri Sayısı --**

SELECT

COUNT(customer\_id) AS total\_customer

FROM

customers;

**-- Toplam Ülke Sayısı --**

A close up of a sign

Description automatically generatedSELECT

COUNT(country) AS country

FROM

customers;

**-- Müşteri Toplam Siparişi ve Tutarları --**

SELECT

c.customer\_id,

COUNT(d.order\_id) AS total\_order,

ROUND(SUM(quantity \* unit\_price)::NUMERIC, 2) AS total\_amount

FROM

order\_details d

JOIN

orders o ON d.order\_id = o.order\_id

JOIN

customers c ON o.customer\_id = c.customer\_id

GROUP BY

c.customer\_id

ORDER BY

total\_order DESC,

total\_amount DESC;

A screenshot of a computer

Description automatically generated

**-- Ülkelere Göre Müşteri Sayısı --**

SELECT

country,

COUNT(DISTINCT customer\_id) AS total\_customer

FROM

customers

GROUP BY

country

ORDER BY

total\_customer DESC;

A screenshot of a data

Description automatically generated

**--RFM--**

**-- Her müşteri için son sipariş tarihi, sipariş sayısı ve toplam harcama (amount) hesaplama**

WITH customer\_rfm AS (

SELECT

o.customer\_id,

MAX(o.order\_date) AS last\_order\_date,

COUNT(o.order\_id) AS frequency,

ROUND(SUM(d.quantity \* d.unit\_price)::NUMERIC, 2) AS monetary

FROM

orders o

JOIN order\_details d ON o.order\_id = d.order\_id

GROUP BY

customer\_id

),

**-- Analiz tarihini verideki en son sipariş tarihinden bir gün sonrası olarak belirleme**

reference\_date AS (

SELECT MAX(order\_date) + INTERVAL '1 day' AS analysis\_date FROM orders

),

**-- Recency değerini gün cinsinden hesaplama ve RFM değerlerini bir araya getirme**

rfm\_values AS (

SELECT

c.customer\_id,

EXTRACT(day FROM (r.analysis\_date - c.last\_order\_date)) AS recency,

c.frequency,

c.monetary,

**-- Recency Skoru**

CASE

WHEN EXTRACT(day FROM (r.analysis\_date - c.last\_order\_date)) <= 30 THEN 5

WHEN EXTRACT(day FROM (r.analysis\_date - c.last\_order\_date)) <= 60 THEN 4

WHEN EXTRACT(day FROM (r.analysis\_date - c.last\_order\_date)) <= 90 THEN 3

WHEN EXTRACT(day FROM (r.analysis\_date - c.last\_order\_date)) <= 120 THEN 2

ELSE 1

END AS r\_score,

**-- Frequency Skoru**

CASE

WHEN c.frequency >= 10 THEN 5

WHEN c.frequency >= 7 THEN 4

WHEN c.frequency >= 4 THEN 3

WHEN c.frequency >= 2 THEN 2

ELSE 1

END AS f\_score,

**-- Monetary Skoru**

CASE

WHEN c.monetary >= 1000 THEN 5

WHEN c.monetary >= 500 THEN 4

WHEN c.monetary >= 250 THEN 3

WHEN c.monetary >= 100 THEN 2

ELSE 1

END AS m\_score

FROM

customer\_rfm c

CROSS JOIN

reference\_date r

)

**-- RFM skorlarını birleştirme ve segment belirleme**

SELECT

customer\_id,

recency,

frequency,

monetary,

r\_score,

f\_score,

m\_score,

**-- RFM Skorunu birleştirme**

(r\_score::TEXT) || (f\_score::TEXT) || (m\_score::TEXT) AS RFM\_Score,

**-- Segment belirleme**

CASE

WHEN (r\_score = 5 AND f\_score = 5 AND m\_score = 5) THEN 'En İyi Müşteriler'

WHEN r\_score = 5 THEN 'Sadık Müşteriler'

WHEN f\_score = 5 THEN 'Potansiyel Müşteriler'

WHEN m\_score = 5 THEN 'Yüksek Değerli Müşteriler'

WHEN r\_score = 1 THEN 'Uyuyan Müşteriler'

ELSE 'Diğer'

END AS segment

FROM

rfm\_values;

A screenshot of a white sheet with numbers and symbols

Description automatically generated

**--- SİPARİŞ ANALİZİ ---**

**-- Toplam Sipariş Sayısı --**

A screenshot of a computer

Description automatically generatedSELECT

COUNT(order\_id) AS total\_orders

FROM

orders;

**-- Kategoriye Göre Toplam Sipariş Sayısı --**

SELECT

ca.category\_name,

COUNT(o.order\_id) AS total\_orders

A screenshot of a list of items

Description automatically generatedFROM

orders o

JOIN

order\_details od ON o.order\_id = od.order\_id

JOIN

products p ON od.product\_id = p.product\_id

JOIN

categories ca ON p.category\_id = ca.category\_id

GROUP BY

ca.category\_name

ORDER BY

total\_orders DESC;

**-- Yıllara Göre Sipariş Sayısı --**

SELECT

EXTRACT(YEAR FROM order\_date) AS order\_year,

COUNT(order\_id) AS total\_orders

A screenshot of a number

Description automatically generatedFROM

orders

GROUP BY

order\_year

ORDER BY

order\_year;

**-- Kategoriye Göre İndirimli ve İndirimsiz Sipariş Tutarları --**

SELECT

c.category\_name,

ROUND(SUM(od.quantity \* od.unit\_price)::NUMERIC, 2) AS total\_order\_amount\_excluding\_discount,

ROUND(SUM((od.quantity \* od.unit\_price) \* (1 - od.discount))::NUMERIC, 2) AS total\_order\_amount\_including\_discount

FROM

orders o

JOIN

order\_details od ON o.order\_id = od.order\_id

JOIN

products p ON od.product\_id = p.product\_id

JOIN

categories c ON p.category\_id = c.category\_id

GROUP BY

c.category\_name

ORDER BY

total\_order\_amount\_including\_discount DESC;

A screenshot of a computer

Description automatically generated

**-- Ürüne Göre Sipariş Durumu --**

SELECT

o.order\_id,

p.product\_name,

o.order\_date,

o.required\_date,

o.shipped\_date,

CASE

WHEN o.shipped\_date IS NOT NULL AND o.shipped\_date <= o.required\_date THEN 'Delivered On Time'

WHEN o.shipped\_date IS NOT NULL AND o.shipped\_date > o.required\_date THEN 'Late Delivery'

WHEN o.shipped\_date IS NULL AND CURRENT\_DATE > o.required\_date THEN 'Overdue'

WHEN o.shipped\_date IS NULL AND CURRENT\_DATE <= o.required\_date THEN 'Pending'

END AS order\_status

FROM

orders o

JOIN order\_details od ON o.order\_id = od.order\_id

JOIN products p ON od.product\_id = p.product\_id

ORDER BY

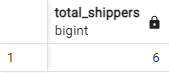
product\_name;

A screenshot of a computer

Description automatically generated

**--- NAKLİYE ANALİZİ ---**

**-- Toplam Nakliyeci Sayısı --**

SELECT

COUNT(shipper\_id) AS total\_shippers

FROM

shippers;

**-- Ortalama Teslimat Günü --**

SELECT

ROUND(AVG(EXTRACT(DAY FROM AGE(shipped\_date, order\_date)))::NUMERIC, 2) AS average\_delivery\_days

FROM

orders

WHERE

shipped\_date IS NOT NULL; -- Teslim edilmeyen siparişleri dışlamak için

A screenshot of a website

Description automatically generated

**-- Taşınan Yük(Nakliyeci) Maliyeti --**

SELECT

s.shipper\_id,

s.company\_name,

COUNT(o.order\_id) AS total\_shipments,

ROUND(AVG(EXTRACT(DAY FROM AGE(o.shipped\_date, o.order\_date)))::NUMERIC, 2) AS average\_delivery\_days,

ROUND(SUM(o.freight)::NUMERIC, 2) AS total\_freight\_cost

FROM

shippers s

JOIN

orders o ON s.shipper\_id = o.ship\_via

WHERE

o.shipped\_date IS NOT NULL -- Teslim edilmeyen siparişleri dışlamak için

GROUP BY

s.shipper\_id,

s.company\_name

A screenshot of a computer

Description automatically generated

**-- Teslimat Sürelerini Gün Cinsinden Hesaplama --**

SELECT

o.order\_id,

o.order\_date,

o.shipped\_date,

s.company\_name AS shipper,

EXTRACT(DAY FROM AGE(o.shipped\_date, o.order\_date)) AS delivery\_days,

CASE

WHEN EXTRACT(DAY FROM AGE(o.shipped\_date, o.order\_date)) >= 10 THEN 'Uzun Teslimat'

ELSE 'Kısa Teslimat'

END AS delivery\_type

FROM

orders o

JOIN

shippers s ON o.ship\_via = s.shipper\_id

WHERE

o.shipped\_date IS NOT NULL

ORDER BY

delivery\_days DESC;

A screenshot of a computer

Description automatically generated

**-- Teslimat Durumu --**

SELECT

o.order\_id,

o.order\_date,

o.required\_date,

o.shipped\_date,

EXTRACT(DAY FROM AGE(o.shipped\_date, o.required\_date)) AS delay\_days,

CASE

WHEN EXTRACT(DAY FROM AGE(o.shipped\_date, o.required\_date)) > 0 THEN 'Gecikmeli Teslimat'

ELSE 'Zamanında Teslimat'

END AS delivery\_status

FROM

orders o

WHERE

o.shipped\_date IS NOT NULL

ORDER BY

delay\_days DESC;

A screenshot of a data

Description automatically generated

**--- CİRO ANALİZİ ---**

**-- 1. Yıllık Ciro Analizi --**

SELECT

EXTRACT(YEAR FROM order\_date) AS order\_year,

ROUND(SUM(od.quantity \* od.unit\_price)::NUMERIC, 2) AS total\_revenue\_excluding\_discount,

ROUND(SUM((od.quantity \* od.unit\_price) \* (1 - od.discount))::NUMERIC, 2) AS total\_revenue\_including\_discount

FROM

orders o

JOIN

order\_details od ON o.order\_id = od.order\_id

WHERE

o.order\_date IS NOT NULL

GROUP BY

order\_year

ORDER BY

order\_year;

A screenshot of a computer

Description automatically generated

**-- 2. Aylık Ciro Analizi--**

SELECT

EXTRACT(YEAR FROM order\_date) AS order\_year,

EXTRACT(MONTH FROM order\_date) AS order\_month,

ROUND(SUM(od.quantity \* od.unit\_price)::NUMERIC, 2) AS total\_revenue\_excluding\_discount,

ROUND(SUM((od.quantity \* od.unit\_price) \* (1 - od.discount))::NUMERIC, 2) AS total\_revenue\_including\_discount

FROM

orders o

JOIN

order\_details od ON o.order\_id = od.order\_id

WHERE

o.order\_date IS NOT NULL

GROUP BY

order\_year, order\_month

ORDER BY

order\_year, order\_month;

A screenshot of a computer

Description automatically generated

**-- 3. Kategori Bazında Ciro Analizi --**

SELECT

c.category\_name,

ROUND(SUM(od.quantity \* od.unit\_price)::NUMERIC, 2) AS total\_revenue\_excluding\_discount,

ROUND(SUM((od.quantity \* od.unit\_price) \* (1 - od.discount))::NUMERIC, 2) AS total\_revenue\_including\_discount

FROM

order\_details od

JOIN

products p ON od.product\_id = p.product\_id

JOIN

categories c ON p.category\_id = c.category\_id

GROUP BY

c.category\_name

ORDER BY

total\_revenue\_including\_discount DESC;

A screenshot of a computer

Description automatically generated

**-- 4. Müşteri Bazında Ciro Analizi --**

SELECT

c.customer\_id,

c.company\_name,

ROUND(SUM(od.quantity \* od.unit\_price)::NUMERIC, 2) AS total\_revenue\_excluding\_discount,

ROUND(SUM((od.quantity \* od.unit\_price) \* (1 - od.discount))::NUMERIC, 2) AS total\_revenue\_including\_discount

FROM

orders o

JOIN

order\_details od ON o.order\_id = od.order\_id

JOIN

customers c ON o.customer\_id = c.customer\_id

GROUP BY

c.customer\_id, c.company\_name

ORDER BY

total\_revenue\_including\_discount DESC;

A screenshot of a screenshot of a computer

Description automatically generated

**-- 5. Ürün Bazında Ciro Analizi --**

SELECT

p.product\_name,

ROUND(SUM(od.quantity \* od.unit\_price)::NUMERIC, 2) AS total\_revenue\_excluding\_discount,

ROUND(SUM((od.quantity \* od.unit\_price) \* (1 - od.discount))::NUMERIC, 2) AS total\_revenue\_including\_discount

FROM

order\_details od

JOIN

products p ON od.product\_id = p.product\_id

GROUP BY

p.product\_name

ORDER BY

total\_revenue\_including\_discount DESC;

A screenshot of a computer

Description automatically generated

**-- 6. Çeyrek Bazında Ciro Analizi --**

SELECT

EXTRACT(YEAR FROM order\_date) AS order\_year,

EXTRACT(QUARTER FROM order\_date) AS order\_quarter,

ROUND(SUM(od.quantity \* od.unit\_price)::NUMERIC, 2) AS total\_revenue\_excluding\_discount,

ROUND(SUM((od.quantity \* od.unit\_price) \* (1 - od.discount))::NUMERIC, 2) AS total\_revenue\_including\_discount

FROM

orders o

JOIN

order\_details od ON o.order\_id = od.order\_id

GROUP BY

order\_year, order\_quarter

ORDER BY

order\_year, order\_quarter;

A screenshot of a computer

Description automatically generated

**-- 7. İndirim Etkisinin Analizi --**

SELECT

EXTRACT(YEAR FROM o.order\_date) AS order\_year,

ROUND(SUM(od.quantity \* od.unit\_price)::NUMERIC, 2) AS revenue\_without\_discount,

ROUND(SUM((od.quantity \* od.unit\_price) \* (1 - od.discount))::NUMERIC, 2) AS revenue\_with\_discount,

ROUND((SUM(od.quantity \* od.unit\_price) - SUM((od.quantity \* od.unit\_price) \* (1 - od.discount)))::NUMERIC, 2) AS total\_discount\_amount,

ROUND((AVG(od.discount) \* 100)::NUMERIC, 2) AS average\_discount\_percentage

FROM

orders o

JOIN

order\_details od ON o.order\_id = od.order\_id

GROUP BY

order\_year

ORDER BY

order\_year;

A screenshot of a computer

Description automatically generated