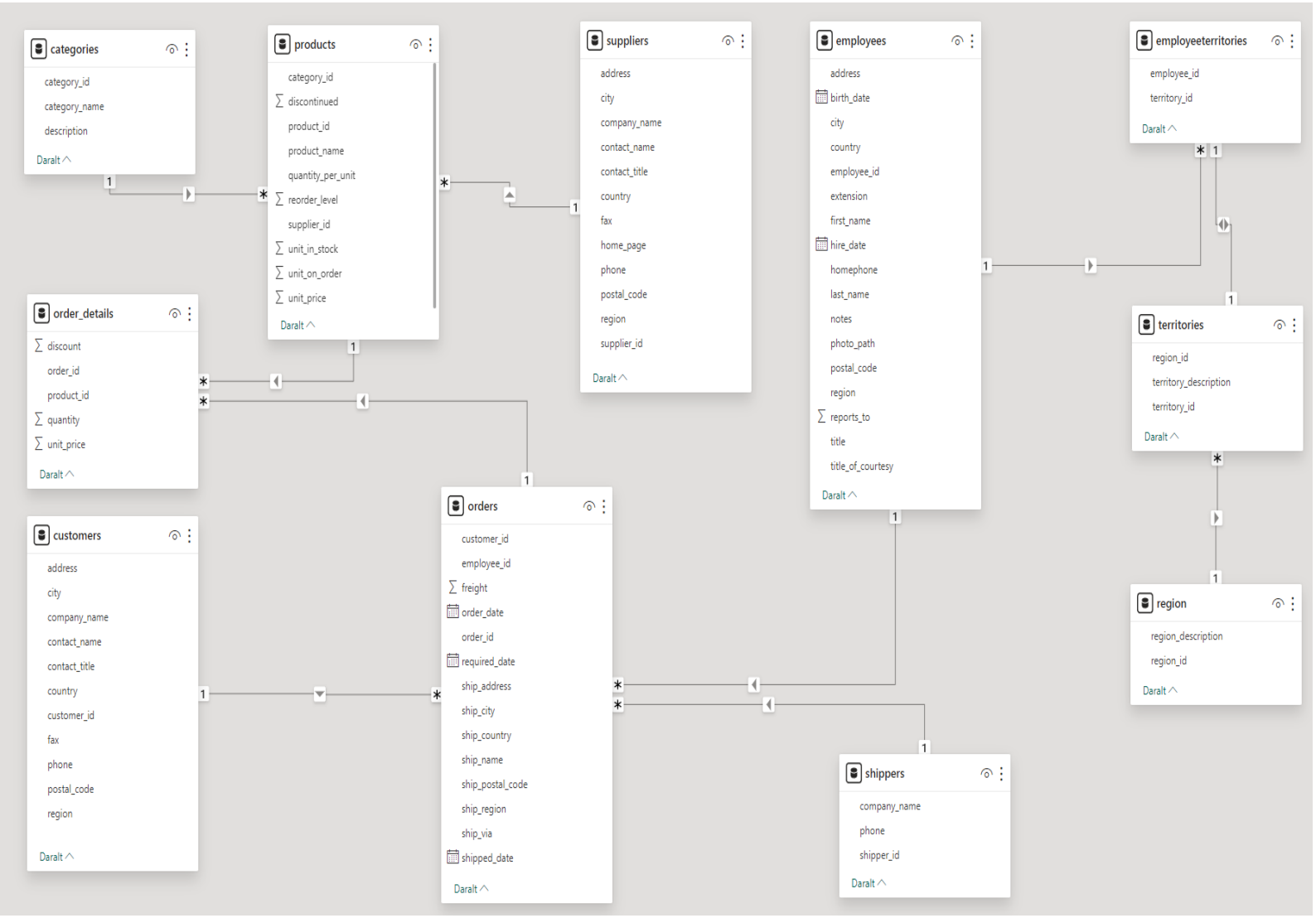
NORTHWIND


categories
category_id
category_name
description
Daralt
products
category_id
discontinued
product_id
product_name
quantity_per_unit
reorder_level
supplier_id
unit_in_stock
unit_on_order
unit_price
Daralt
suppliers
address
city
company_name
contact_name
contact_title
country
fax
home_page
phone
postal_code
region
supplier_id
Daralt
employees
address
birth_date
city
country
employee_id
extension
first_name
hire_date
homephone
last_name
notes
photo_path
postal_code
region
reports_to
title
title_of_courtesy
Daralt
employeeterritories
employee_id
territory_id
Daralt
territories
region_id
territory_description
territory_id
Daralt
region
region_description
region_id
Daralt
order_details
discount
order_id
product_id
quantity
unit_price
Daralt
customers
address
city
company_name
contact_name
contact_title
country
customer_id
fax
phone
postal_code
region
Daralt
orders
customer_id
employee_id
freight
order_date
order_id
required_date
ship_address
ship_city
ship_country
ship_name
ship_postal_code
ship_region
ship_via
shipped_date
Daralt
shippers
company_name
phone
shipper_id
Daralt

1.SATIŞ ANALİZİ

Satış verilerini inceleyerek satış performansını ve trendlerini belirlemek.

```
--- Ay Bazında Toplam Satış Geliri ---
SELECT
        TO_CHAR(o.order_date, 'YYYY-MM') AS "Ay",
    SUM(od.quantity) AS "Toplam Satılan Ürün Miktarı",
        CAST(SUM(od.quantity * od.unit_price) AS NUMERIC(10,2)) AS "Toplam Satış
Geliri"
FROM orders o
JOIN order_details od ON o.order_id = od.order_id
GROUP BY TO_CHAR(o.order_date, 'YYYY-MM')
ORDER BY 1;

--- Ürün Bazında Satış Geliri ---
SELECT
    p.product_name AS "Ürün İsmi",
    SUM(od.quantity) AS "Satılan Miktar",
    CAST(SUM(od.quantity * od.unit_price) AS NUMERIC(10, 2)) AS "Toplam Gelir"
FROM
    products p
JOIN
    order_details od ON p.product_id = od.product_id
GROUP BY
    p.product_name
ORDER BY
        3 DESC;
```

```sql
--- Müşteri Bazında Satış Geliri ---
SELECT
    c.customer_id AS "Müşteri ID",
    c.company_name AS "Şirket Adı",
    SUM(od.quantity) AS "Toplam Ürün Miktarı",
    CAST(SUM(od.unit_price * od.quantity * (1 - od.discount)) AS NUMERIC (10,2)) AS "Toplam Satış"
FROM
    customers c
JOIN
    orders o ON c.customer_id = o.customer_id
JOIN
    order_details od ON o.order_id = od.order_id
GROUP BY
    c.customer_id, c.company_name
ORDER BY
    4 DESC;

--- Şehir Bazında Satış Geliri ---
SELECT
    c.city AS "Şehir",
    SUM(od.quantity) AS "Toplam Ürün Miktarı",
    CAST(SUM(od.unit_price * od.quantity * (1 - od.discount)) AS NUMERIC(10,2)) AS "Toplam Satış"
FROM
    customers c
JOIN
    orders o ON c.customer_id = o.customer_id
JOIN
    order_details od ON o.order_id = od.order_id
GROUP BY
```

```
    c.city
ORDER BY
    3 DESC;


--- İndirim Oranı Bazında Satış Geliri ---
SELECT
    od.discount AS "İndirim Oranı",
    SUM(od.quantity) AS "Toplam Satılan Ürün Miktarı",
    CAST(SUM(od.quantity * od.unit_price * (1 - od.discount)) AS NUMERIC(10,2)) AS
"Toplam Satış Geliri"
FROM
    order_details od
GROUP BY
    od.discount
ORDER BY
    3   DESC;
```

En çok satan ürünleri, en karlı müşterileri, sezonluk satış değişikliklerini ve satış ekiplerinin performansını tespit edip bulduğumuz sonuçları satış stratejilerini geliştirmeye ve pazarlama kampanyalarını optimize etmeye yardımcı olur.

2. MÜŞTERİ ANALİZİ

Müşteri davranışlarını ve demografik özelliklerini inceleyerek müşteri segmentlerini ve sadakatini anlamak.

```
--- En yüksek toplam harcama yapan 1/3 müşteriye SMS ile kampanya tanımlayacağız. ---
WITH customer_spending AS (
    SELECT
        c.customer_id AS customer_id,
        CAST(SUM(od.quantity * od.unit_price * (1 - od.discount)) AS NUMERIC(10,2)) AS
total_spending
    FROM
```

```
        customers c
    JOIN
        orders o ON c.customer_id = o.customer_id
    JOIN
        order_details od ON o.order_id = od.order_id
    GROUP BY
        c.customer_id
)

SELECT
    c.customer_id AS "Müşteri ID",
    c.company_name AS "Şirket Adı",
    cs.total_spending AS "Toplam Harcama",
                c.phone AS "Telefon Numarası",
    CASE
        WHEN cs.total_spending <= (SELECT PERCENTILE_CONT(0.33) WITHIN GROUP
(ORDER BY total_spending) FROM customer_spending) THEN 'Düşük Harcama'
        WHEN cs.total_spending > (SELECT PERCENTILE_CONT(0.33) WITHIN GROUP
(ORDER BY total_spending) FROM customer_spending) AND cs.total_spending <=
(SELECT PERCENTILE_CONT(0.66) WITHIN GROUP (ORDER BY total_spending)
FROM customer_spending) THEN 'Orta Harcama'
        ELSE 'Yüksek Harcama'
    END AS "Harcama Grubu"
FROM
    customers c
JOIN
    customer_spending cs ON c.customer_id = cs.customer_id
ORDER BY
    3 DESC;
```

```sql
--- Hangi ülkeden kaç tane müşterilerim var? ---
SELECT
    country,
    COUNT(customer_id) AS customer_count
FROM
    customers
GROUP BY
    country
ORDER BY
    2 DESC;

--- En çok alışveriş yapan müşteriler? ---
SELECT
    c.customer_id AS "Müşteri ID",
    COUNT(DISTINCT o.order_id) AS "Sipariş Sayıları"
FROM
    customers c
LEFT JOIN
    orders o ON c.customer_id = o.customer_id
GROUP BY
    c.customer_id
ORDER BY
                2 DESC
```

Müşteri segmentasyonu yaparak hedeflenen pazarlama kampanyaları oluşturabilir, müşteri memnuniyetini artırabilir ve müşteri kaybını azaltabilir. Ayrıca, yeni müşteri edinme stratejilerini geliştirmeye yardımcı olur.

3.ÇALIŞAN ANALİZİ

Çalışanların performansını ve verimliliğini değerlendirmek.

```
--- Çalışanların Yıllık Sipariş Sayıları ---
SELECT
    e.employee_id AS "Çalışan ID",
    e.first_name || ' ' || e.last_name AS "Çalışan Ad-Soyad",
    EXTRACT(YEAR FROM o.order_date) AS "Yıl",
    SUM(od.quantity) AS "Toplam Sipariş Sayısı"
FROM
    employees e
JOIN
    orders o ON e.employee_id = o.employee_id
JOIN
    order_details od ON o.order_id = od.order_id
GROUP BY
    1, 2, 3
ORDER BY
    3, 4 DESC;

--- Çalışanların Hangi Şehirlerde Kaç Tane Müşterisi Var? ---
WITH employee_customer_counts AS (
    SELECT
        e.employee_id ,
        e.first_name || ' ' || e.last_name AS employee_name,
        c.city AS customer_city,
        COUNT(DISTINCT c.customer_id) AS total_customers
    FROM
        employees e
    LEFT JOIN
```

```
        orders o ON e.employee_id = o.employee_id
    LEFT JOIN
        customers c ON o.customer_id = c.customer_id
    GROUP BY
        e.employee_id, e.first_name, e.last_name, c.city
)
SELECT
    employee_id AS "Çalışan ID",
    employee_name AS "Çalışan Adı Soyadı",
    customer_city AS "Müşterinin Şehri",
    total_customers AS "Müşteri Sayısı"
FROM
    employee_customer_counts
ORDER BY
    4 DESC;

--- Aylık Tutar Olarak En Fazla Satış Yapan Çalışanlar ---
SELECT
    e.employee_id,
    e.first_name || ' ' || e.last_name AS employee_name,
    EXTRACT(YEAR FROM o.order_date) AS order_year,
    EXTRACT(MONTH FROM o.order_date) AS order_month,
    ROUND(SUM(od.unit_price * od.quantity)::numeric, 2) AS total_sales
FROM
    employees e
JOIN
    orders o ON e.employee_id = o.employee_id
JOIN
    order_details od ON o.order_id = od.order_id
GROUP BY
```

```
    1, 2, 3, 4
ORDER BY
    3, 4, 5 DESC;
```

En yüksek performans gösteren çalışanları ve satış ekiplerini belirler. Eğitim ve geliştirme ihtiyaçlarını tespit eder, çalışan motivasyonunu ve memnuniyetini artırır.

4. FİNANSAL ROI ANALİZİ

Finansal ROI (Return on Investment) analizi, yapılan yatırımın karlılığını değerlendirmek için kullanılan önemli bir araçtır.

```
--- ROI (Return on Investment - Yatırım Getirisi) ---
SELECT
    p.product_name,
    p.unit_price AS satis_fiyati,
    (od.unit_price * (1 + od.discount)) AS maliyet,
    (p.unit_price - (od.unit_price * (1 + od.discount))) AS kar,
    ((p.unit_price - (od.unit_price * (1 + od.discount))) / p.unit_price) * 100 AS kar_orani
FROM
    products p
JOIN
    order_details od ON p.product_id = od.product_id;
```

ROI, çeşitli yatırım seçeneklerinin karşılaştırılmasına olanak tanır, böylece hangi yatırımın daha karlı olduğunu belirlemek daha kolay olur. Yatırımların performansını ölçmek için basit ve anlaşılır bir metrik sağlar. Şirket içi projelerin veya departmanların verimliliğini değerlendirmek için kullanılabilir. Yatırımların getirisini hesaplayarak risklerin yönetilmesine yardımcı olur.

5. LOJİSTİK ANALİZ

Nakliye ve dağıtım süreçlerini inceleyerek lojistik performansını artırmak.

```
--- Gemi Şirketinin Toplam Sipariş Sayısı ---
SELECT s.company_name AS "Gemi Şirketi", COUNT(o.order_id) AS "Sipariş Sayısı"
FROM orders o
JOIN shippers s ON o.ship_via = s.shipper_id
GROUP BY 1
ORDER BY 2 DESC;

--- Siparişi Geciken Ürünlerin Nakliye Şirketleri ---
SELECT s.company_name AS "Gemi Şirketi",
      p.product_name AS "Ürün Adı",
      (o.shipped_date - o.required_date) AS "Gecikme Süresi"
FROM orders o
JOIN shippers s ON o.ship_via = s.shipper_id
JOIN order_details od ON o.order_id = od.order_id
JOIN products p ON od.product_id = p.product_id
WHERE o.shipped_date > o.required_date
ORDER BY 3 DESC;
```

Dağıtım maliyetlerini düşürür, teslimat sürelerini optimize eder ve müşteri memnuniyetini artırır. Tedarik zinciri verimliliğini ve stok yönetimini iyileştirir.

6. STOK ANALİZİ

Stok seviyelerini ve ürün hareketlerini incelemek.

```
--- Her ürün için mevcut stok miktarını, sipariş miktarını, yeniden sipariş seviyesini ve stok seviyelerini kontrol etme. ---
SELECT
    product_name,
    unit_in_stock AS "Mevcut Stok",
    unit_on_order AS "Sipariş Miktarı",
    reorder_level AS "Yeniden Sipariş Seviyesi",
    CASE
        WHEN unit_in_stock <= reorder_level THEN 'Yeniden Sipariş Edilmeli'
        ELSE 'Stok Seviyesi Yeterli'
    END AS "Stok Durumu"
FROM
    products
ORDER BY 5
--- Stok Değerleme: Ürün kategorisinde toplam Stok Maaliyeti ---
SELECT
    c.category_name AS "Kategori",
    CAST(SUM(p.unit_price * p.unit_in_stock) AS NUMERIC (10,2)) AS "Toplam Maliyet"
FROM
    products p
INNER JOIN
    categories c ON p.category_id = c.category_id
GROUP BY
    1
ORDER BY
                2 DESC
```

Stok devir hızını artırır, stok maliyetlerini azaltır ve stokta tükenme veya fazla stok sorunlarını önler. Talep tahmini yaparak doğru stok seviyelerini belirler.

7. RFM ANALİZİ

Müşterilerin satın alma davranışlarını (son satın alma tarihi, satın alma sıklığı, harcama miktarı) incelemek.

```
--- RFM Analizi ---
WITH RFM AS (
    SELECT
        c.customer_id,
        EXTRACT(DAY FROM '1998-05-06'::TIMESTAMP - MAX(o.order_date)) AS recency,
        COUNT(DISTINCT o.order_id) AS frequency,
        SUM(od.unit_price * od.quantity * (1 - od.discount)) AS monetary
    FROM
        customers c
    JOIN orders o ON c.customer_id = o.customer_id
    JOIN order_details od ON o.order_id = od.order_id
    GROUP BY
        c.customer_id
),

Recency_Scores AS (
    SELECT
        customer_id,
        recency,
        NTILE(5) OVER (ORDER BY recency DESC) AS recency_score
    FROM
        RFM
),

Frequency_Scores AS (
```

```
    SELECT
        customer_id,
        frequency,
        NTILE(5) OVER (ORDER BY frequency ASC) AS frequency_score
    FROM
        RFM
),

Monetary_Scores AS (
    SELECT
        customer_id,
        monetary,
        NTILE(5) OVER (ORDER BY monetary ASC) AS monetary_score
    FROM
        RFM
),

Total_Scores AS (
    SELECT
        r.customer_id,
        r.recency,
        r.recency_score,
        f.frequency,
        f.frequency_score,
        CAST(m.monetary AS NUMERIC(10,2)),
        m.monetary_score,
        (r.recency_score + f.frequency_score + m.monetary_score) AS total_score
    FROM
        Recency_Scores r
    JOIN Frequency_Scores f ON r.customer_id = f.customer_id
```

```
    JOIN Monetary_Scores m ON r.customer_id = m.customer_id
)

SELECT
    *
FROM
    Total_Scores
ORDER BY
    8 DESC;
```

Müşteri segmentasyonu yaparak hedeflenen pazarlama kampanyaları oluşturur. En değerli müşterileri belirler ve müşteri sadakat programlarını optimize eder. Pazarlama ve satış stratejilerini müşteri davranışlarına göre uyarlamaya yardımcı olur.