

--Q1) En çok sipariş veren ilk 10 müşteri + toplam harcama

```
SELECT
    c.customer_id,
    c.full_name,
    COUNT(o.order_id) AS order_count,
    ROUND(SUM(o.total_amount)) AS total_spend
FROM customers c
JOIN orders o ON o.customer_id = c.customer_id
WHERE o.status <> 'cancelled'
GROUP BY c.customer_id, c.full_name
ORDER BY order_count DESC, total_spend DESC
LIMIT 10;
```

--Q2) Kategori bazında ciro (revenue) (quantity * unit price)

```
SELECT
    p.category,
    ROUND(SUM(oi.quantity * oi.unit_price), 2) AS revenue
FROM order_items oi
JOIN products p ON p.product_id = oi.product_id
JOIN orders o ON o.order_id = oi.order_id
WHERE o.status <> 'cancelled'
GROUP BY p.category
ORDER BY revenue DESC;
```

--Q3) Aylık sipariş trendi sipariş sayısı + ciro

```
SELECT
    DATE_TRUNC('month', o.order_date)::date AS month,
    COUNT(*) AS orders_count,
    ROUND(SUM(o.total_amount), 2) AS revenue
FROM orders o
WHERE o.status <> 'cancelled'
GROUP BY month
ORDER BY month;
```

--Q4) Teslimat gecikmesi analizi: ortalama teslim süresi

```
SELECT
    s.carrier,
    COUNT(*) AS delivered_shipments,
    ROUND(AVG(EXTRACT(EPOCH FROM (s.delivered_at - s.shipped_at)) / 86400.0), 2)
AS avg_delivery_days
FROM shipments s
WHERE s.delivered_at IS NOT NULL
GROUP BY s.carrier
ORDER BY avg_delivery_days DESC;
```

--Q5) Geç teslim oranı carrier bazında

```
WITH delivered AS (
    SELECT
        s.*,
        (EXTRACT(EPOCH FROM (s.delivered_at - s.shipped_at)) / 86400.0) AS
delivery_days
    FROM shipments s
```

```

        WHERE s.delivered_at IS NOT NULL
    )
    SELECT
        carrier,
        COUNT(*) AS delivered_shipments,
        SUM(CASE WHEN delivery_days > 3 THEN 1 ELSE 0 END) AS late_shipments,
        ROUND(100.0 * SUM(CASE WHEN delivery_days > 3 THEN 1 ELSE 0 END) / COUNT(*),
2) AS late_rate_pct
    FROM delivered
    GROUP BY carrier
    ORDER BY late_rate_pct DESC, delivered_shipments DESC;

```

--Q6) İade Oranı kategori bazında

```

WITH category_orders AS (
    SELECT DISTINCT
        p.category,
        o.order_id
    FROM orders o
    JOIN order_items oi ON oi.order_id = o.order_id
    JOIN products p ON p.product_id = oi.product_id

    WHERE o.status <> 'cancelled'
),
returned_orders AS (
    SELECT
        r.order_id
    FROM returns r
)
SELECT
    co.category,
    COUNT(*) AS total_orders_in_category,
    SUM(CASE WHEN ro.order_id IS NOT NULL THEN 1 ELSE 0 END) AS
returned_orders_in_category,
    ROUND(
        100.0 * SUM(CASE WHEN ro.order_id IS NOT NULL THEN 1 ELSE 0 END) /
COUNT(*)
    , 2) AS return_rate_pct
    FROM category_orders co
    JOIN returned_orders ro ON ro.order_id = co.order_id
    GROUP BY co.category
    ORDER BY return_rate_pct DESC, total_orders_in_category;

```

--Q7) En çok iade edilen reason + toplam refund

```

SELECT
    reason,
    COUNT(*) AS return_count,
    ROUND(SUM(refund_amount), 2) AS total_refund
    FROM returns
    GROUP BY reason
    ORDER BY return_count DESC, total_refund DESC;

```

--Q8) Repeat rate: 2+ sipariş veren müşteri oranı

```

WITH customer_orders AS (
    SELECT

```

```

        customer_id,
        COUNT(*) AS order_count
    FROM orders
    WHERE status <> 'cancelled'
    GROUP BY customer_id
)
SELECT
    COUNT(*) AS total_customers_with_orders,
    SUM(CASE WHEN order_count >= 2 THEN 1 ELSE 0 END) AS repeat_customers,
    ROUND(100.0 * SUM(CASE WHEN order_count >= 2 THEN 1 ELSE 0 END) / COUNT(*), 2)
AS repeat_rate_pct
FROM customer_orders;

```

--Q9) Window: Her müşteri için siparişleri tarihe göre sırala

```

SELECT
    o.customer_id,
    c.full_name,
    o.order_id,
    o.order_date,
    o.status,
    o.total_amount,
    ROW_NUMBER() OVER(PARTITION BY o.customer_id ORDER BY o.order_date) AS
order_rank_for_customer
FROM orders o
JOIN customers c ON c.customer_id = o.customer_id
WHERE o.status <> 'cancelled'
ORDER BY o.customer_id DESC, order_rank_for_customer DESC;

```

--Q10) Recency / Frequency / Monetary

```

WITH ref AS (
    SELECT MAX(order_date)::date AS as_of_date
    FROM orders
    WHERE status <> 'cancelled'
),
rfm AS (
    SELECT
        o.customer_id,
        MAX(o.order_date)::date AS last_order_date,
        COUNT(*) AS frequency,
        ROUND(SUM(o.total_amount), 2) AS monetary
    FROM orders o
    WHERE o.status <> 'cancelled'
    GROUP BY o.customer_id
),
scored AS (
    SELECT
        r.customer_id,
        r.last_order_date,
        (SELECT as_of_date FROM ref) AS as_of_date,
        ((SELECT as_of_date FROM ref) - r.last_order_date) AS recency_days,
        r.frequency,
        r.monetary,
        CASE
            WHEN r.frequency >= 3 AND r.monetary >= 3000 THEN 'VIP'

```

```
        WHEN r.frequency >= 2 THEN 'Regular'
        ELSE 'One-time'
      END AS segment
    FROM rfm r
  )
SELECT
  s.customer_id,
  c.full_name,
  s.segment,
  s.recency_days,
  s.frequency,
  s.monetary,
  s.last_order_date,
  s.as_of_date
FROM scored s
JOIN customers c ON c.customer_id = s.customer_id
ORDER BY
  CASE s.segment WHEN 'VIP' THEN 1 WHEN 'Regular' THEN 2 ELSE 3 END,
  s.monetary DESC;
```