```
--1-Total revenue(sum of price+freight value) for delievered orders:
select sum(oi.price+oi.freight_value) as total_revenue
from order_items oi
join orders o on oi.order_id=o.order_id
where o.order_status='delivered'

--2-Expected revenue (sum of price+freight value) for approved orders:
SELECT SUM(oi.price + oi.freight_value) AS expected_revenue
FROM order_items oi
JOIN orders o ON oi.order_id = o.order_id
WHERE o.order_status = 'approved'

--3-Net profit(price-freight value) for delivered orders:
Select sum(oi.price-oi.freight_value) As net_profits
from order_items oi
join orders o on oi.order_id=o.order_id
where order_status='delivered'

--4-Net profit margin %= (net profit/total revenue)*100 :
SELECT (SUM(oi.price - oi.freight_value) * 100.0 /
NULLIF(SUM(oi.price + oi.freight_value), 0)) AS net_profit_margin_percentage
FROM order_items oi
JOIN orders o ON oi.order_id = o.order_id
WHERE o.order_status = 'delivered';

--5-Average order value(total revenue/ number of delivered orders:
```

```
SELECT SUM(oi.price + oi.freight_value) / COUNT(DISTINCT o.order_id) AS
average_order_value

FROM order_items oi

JOIN orders o ON oi.order_id = o.order_id

WHERE o.order_status = 'delivered';


--6-Revenue by product category:

SELECT  p.product_category_name,

SUM(oi.price + oi.freight_value) AS category_revenue

FROM order_items oi

JOIN products p ON oi.product_id = p.product_id

JOIN orders o ON oi.order_id = o.order_id

WHERE o.order_status = 'delivered'

GROUP BY p.product_category_name

ORDER BY category_revenue DESC


-- percentage of revenue for each category

WITH category_revenue AS (

  SELECT

    p.product_category_name,

    SUM(oi.price + oi.freight_value) AS category_revenue

  FROM order_items oi

  JOIN products p ON oi.product_id = p.product_id

  JOIN orders o ON oi.order_id = o.order_id

  WHERE o.order_status = 'delivered'

  GROUP BY p.product_category_name
),
total_revenue AS (
   SELECT SUM(category_revenue) AS total
   FROM category_revenue
)

SELECT
   cr.product_category_name,
   cr.category_revenue,
   ROUND((cr.category_revenue * 100.0 / tr.total), 2) AS revenue_percentage
FROM category_revenue cr
CROSS JOIN total_revenue tr
ORDER BY cr.category_revenue DESC;
--7-Revenue Per seller:
SELECT s.seller_id, SUM(oi.price + oi.freight_value) AS seller_revenue
FROM order_items oi
JOIN sellers s ON oi.seller_id = s.seller_id
JOIN orders o ON oi.order_id = o.order_id
WHERE o.order_status = 'delivered'
GROUP BY s.seller_id
ORDER BY seller_revenue DESC;

--8-Revenue per customer:
SELECT o.customer_id,SUM(oi.price + oi.freight_value) AS customer_revenue
FROM order_items oi
JOIN orders o ON oi.order_id = o.order_id
```

```
WHERE o.order_status = 'delivered'

GROUP BY o.customer_id

ORDER BY customer_revenue DESC;


--9-Orders with freight > product price:

WITH order_stats AS (

   SELECT

 COUNT(DISTINCT oi.order_id) AS total_orders,

SUM(CASE WHEN oi.freight_value > oi.price THEN 1 ELSE 0 END) AS freight_heavy_orders

FROM order_items oi

JOIN orders o ON oi.order_id = o.order_id

 WHERE o.order_status = 'delivered'

)

SELECT

   (freight_heavy_orders * 100.0 / total_orders) AS percentage_freight_heavy

FROM order_stats;


--10- Monthly Revenue forecast:

WITH MonthlyRevenue AS (

   SELECT

      YEAR(o.order_purchase_timestamp) AS Year,

      MONTH(o.order_purchase_timestamp) AS Month,

      SUM(oi.price + oi.freight_value) AS MonthlyRevenue

   FROM orders o

   JOIN order_items oi ON o.order_id = oi.order_id

   WHERE o.order_status = 'delivered'
```

```
  GROUP BY YEAR(o.order_purchase_timestamp), MONTH(o.order_purchase_timestamp)
)
SELECT  Year,Month, MonthlyRevenue,
 AVG(MonthlyRevenue) OVER (ORDER BY Year, Month ROWS BETWEEN 2 PRECEDING AND CURRENT ROW) AS MovingAvg3Months
 from MonthlyRevenue


--11- Number of unique customers:
SELECT COUNT(DISTINCT customer_id) AS unique_customers
FROM customers
--OR
SELECT
COUNT(DISTINCT COALESCE(customer_id, 'MISSING_ID')) AS unique_customers
FROM customers


--12-Returning Customers:
WITH CustomerOrders AS (
  SELECT
    customer_id,
    COUNT(DISTINCT order_id) AS OrderCount
  FROM orders
  GROUP BY customer_id
)
SELECT
  COUNT(*) AS ReturningCustomers
FROM CustomerOrders
```

```
WHERE OrderCount > 1
```

```
--13-Customer retention rate:

SELECT ROUND( COUNT(DISTINCT returning.customer_id) * 100.0 /
NULLIF(COUNT(DISTINCT initial.customer_id), 0), 2)

AS retention_rate_percentage

FROM (SELECT DISTINCT customer_id FROM Orders

WHERE order_purchase_timestamp BETWEEN '2024-01-01' AND '2024-03-31') initial

LEFT JOIN (SELECT DISTINCT customer_id FROM Orders

WHERE order_purchase_timestamp BETWEEN '2024-04-01' AND '2024-06-30') returning

ON initial.customer_id = returning.customer_id;

--14-Customer Churn Rate (if time range allows)

SELECT (COUNT(DISTINCT CASE WHEN last_order_date < start_date THEN customer_id
END) * 100.0) / COUNT(DISTINCT customer_id)

AS churn_rate_percentage FROM (SELECT customer_id, MAX(order_purchase_timestamp)
AS last_order_date

FROM Orders GROUP BY customer_id) AS customer_activity;

--15-Customer Lifetime Value (CLV)

SELECT c.customer_id, SUM(oi.price + oi.freight_value) AS total_revenue,

COUNT(DISTINCT o.order_id) AS order_count, DATEDIFF(day,
MIN(o.order_purchase_timestamp),

MAX(o.order_purchase_timestamp)) / NULLIF(COUNT(DISTINCT o.order_id) - 1, 0) AS
avg_days_between_orders,

SUM(oi.price + oi.freight_value) * (COUNT(DISTINCT o.order_id) / NULLIF(DATEDIFF(day,
MIN(o.order_purchase_timestamp),

MAX(o.order_purchase_timestamp)) + 1, 0)) AS estimated_clv
```

```
FROM Orders o JOIN dbo.order_items oi

ON o.order_id = oi.order_id JOIN dbo.customers c ON o.customer_id = c.customer_id
GROUP BY c.customer_id;

--16-Top Customer Locations

 SELECT c.customer_state, COUNT(DISTINCT o.customer_id) AS customer_count,

 COUNT(DISTINCT o.order_id) AS order_count, SUM(oi.price + oi.freight_value) AS
total_revenue

 FROM Orders o JOIN dbo.customers c ON o.customer_id = c.customer_id

 JOIN dbo.order_items oi ON o.order_id = oi.order_id GROUP BY c.customer_state ORDER
BY total_revenue DESC;

 SELECT TOP 20 c.customer_city, c.customer_state, COUNT(DISTINCT o.customer_id) AS
customer_count,

 COUNT(DISTINCT o.order_id) AS order_count, SUM(oi.price + oi.freight_value) AS
total_revenue

 FROM Orders o JOIN customers c ON o.customer_id = c.customer_id

 JOIN order_items oi ON o.order_id = oi.order_id GROUP BY c.customer_city,
c.customer_state

 ORDER BY total_revenue DESC;

 --17-Average Review Score

 SELECT AVG(CAST(review_score AS DECIMAL(3,1))) AS average_review_score

 FROM order_reviews WHERE review_score IS NOT NULL

 --18-% of 5-Star Reviews

 SELECT (COUNT(CASE WHEN review_score = 5 THEN 1 END) * 100.0) / COUNT(*)

 AS five_star_percentage FROM order_reviews WHERE review_score IS NOT NULL

 --19-Top Products with Low Ratings (1–2 stars)

 SELECT TOP 20 p.product_id, p.product_category_name, COUNT(r.review_id) AS
review_count,

 AVG(CAST(r.review_score AS DECIMAL(3,1))) AS avg_score,
```

```sql
COUNT(CASE WHEN r.review_score IN (1, 2) THEN 1 END) AS low_rating_count
FROM order_reviews r
JOIN order_items oi ON r.order_id = oi.order_id
JOIN products p ON oi.product_id = p.product_id
GROUP BY p.product_id, p.product_category_name
HAVING AVG(CAST(r.review_score AS DECIMAL(3,1))) <= 2.5
ORDER BY low_rating_count DESC
--20-Average Review Score per Seller
SELECT s.seller_id, COUNT(r.review_id) AS review_count,
AVG(CAST(r.review_score AS DECIMAL(3,1))) AS avg_score
FROM order_reviews r
JOIN order_items oi ON r.order_id = oi.order_id
JOIN sellers s ON oi.seller_id = s.seller_id
GROUP BY s.seller_id
ORDER BY avg_score DESC
--21-Average Review Score per Product Category
SELECT p.product_category_name, COUNT(r.review_id) AS review_count,
AVG(CAST(r.review_score AS DECIMAL(3,1))) AS avg_score
FROM order_reviews r
JOIN order_items oi ON r.order_id = oi.order_id
JOIN products p ON oi.product_id = p.product_id
WHERE p.product_category_name IS NOT NULL
GROUP BY p.product_category_name
ORDER BY avg_score DESC;
--22-Average Delivery Time
```

```sql
SELECT AVG(DATEDIFF(day, order_purchase_timestamp,
order_delivered_customer_date)) AS avg_delivery_time_days

FROM Orders WHERE order_delivered_customer_date IS NOT NULL AND order_status =
'delivered';

--23-Late Delivery Rate

SELECT CAST(COUNT(CASE WHEN order_delivered_customer_date >
order_estimated_delivery_date THEN 1 END) AS FLOAT) * 100.0 / COUNT(*)

AS late_delivery_rate_percentage

FROM Orders

WHERE order_delivered_customer_date IS NOT NULL AND order_status = 'delivered';

--24- Delivery Delay (Avg. Days Late)

SELECT AVG(DATEDIFF(day, order_estimated_delivery_date,
order_delivered_customer_date))

AS avg_delivery_delay_days

FROM Orders

WHERE order_delivered_customer_date > order_estimated_delivery_date AND
order_delivered_customer_date

IS NOT NULL AND order_status = 'delivered';

--25-Average Time from Order to Delivery

SELECT ROUND( COUNT(DISTINCT returning.customer_id) * 100.0 /
NULLIF(COUNT(DISTINCT initial.customer_id), 0), 2)

AS retention_rate_percentage FROM (SELECT DISTINCT customer_id

FROM Orders WHERE order_purchase_timestamp BETWEEN '2024-01-01' AND '2024-03-
31') initial

LEFT JOIN (SELECT DISTINCT customer_id FROM Orders

WHERE order_purchase_timestamp BETWEEN '2024-04-01' AND '2024-06-30') returning

ON initial.customer_id = returning.customer_id;

--26- Shipping Time
```

```
 SELECT order_id,

DATEDIFF(day, order_delivered_carrier_date, order_delivered_customer_date) AS
shipping_time_days

FROM orders

WHERE order_delivered_customer_date IS NOT NULL

AND order_delivered_carrier_date IS NOT NULL

--27-Order Handling Time

SELECT order_id,

DATEDIFF(hour, order_purchase_timestamp, order_approved_at) AS handling_time_hours

FROM orders

WHERE order_approved_at IS NOT NULL

--28-% of Orders with Missing Delivery Date

SELECT COUNT(CASE WHEN order_delivered_customer_date IS NULL THEN 1 END) *
100.0 / COUNT(*)

AS pct_missing_delivery_date

FROM orders

--29-Popular Payment Methods

SELECT payment_type,COUNT(*) AS order_count,

COUNT(*) * 100.0 / (SELECT COUNT(*) FROM order_payments) AS pct_of_total

FROM order_payments

GROUP BY payment_type

ORDER BY order_count DESC;

--30-Average Number of Installments

SELECT  AVG(payment_installments) AS avg_installments

FROM order_payments

WHERE payment_installments > 0;

--31-Installment Use Rate
```

```
SELECT COUNT(CASE WHEN payment_installments > 1 THEN 1 END) * 100.0 /

COUNT(*) AS installment_use_rate

FROM order_payments

-- 32. Average Payment per Installment

SELECT

SUM(payment_value) / SUM(payment_installments) AS avg_payment_per_installment

FROM order_payments

WHERE payment_installments > 0;

--33-Total Payments Received

SELECT

SUM(payment_value) AS total_payments_received

FROM order_payments

--34-Revenue per Payment Method

SELECT payment_type, SUM(payment_value) AS total_revenue

FROM order_payments

GROUP BY payment_type

ORDER BY total_revenue DESC;

--35-Total Number of Orders

SELECT COUNT(*) AS total_orders from orders

--36-Number of Orders by Category (English translated names)

SELECT product_category_name AS category_name,

 COUNT(DISTINCT oi.order_id) AS order_count

FROM order_items oi

JOIN products p ON oi.product_id = p.product_id

GROUP BY p.product_category_name

ORDER BY order_count DESC
```

```sql
--or
SELECT
   p.product_category_name,
   COUNT(DISTINCT oi.order_id) AS order_count
FROM order_items oi
JOIN products p ON oi.product_id = p.product_id
WHERE p.product_category_name IS NOT NULL
GROUP BY p.product_category_name
ORDER BY order_count DESC;
--37-Canceled Orders
SELECT COUNT(*) AS canceled_orders
FROM orders
WHERE order_status = 'canceled'
--38-Pending Orders
SELECT COUNT(*) AS pending_orders
FROM orders
WHERE order_status IN ('created', 'approved', 'processing', 'invoiced')
--39-Delivered Orders by Month
SELECT
   FORMAT(order_delivered_customer_date, 'yyyy-MM') AS delivery_month,
   COUNT(*) AS delivered_orders
FROM orders
WHERE order_status = 'delivered' AND order_delivered_customer_date IS NOT NULL
GROUP BY FORMAT(order_delivered_customer_date, 'yyyy-MM')
ORDER BY delivery_month desc
--40-Top 10 Most Ordered Products
```

```
SELECT TOP 10
p.product_id,
 p.product_category_name,
   COUNT(*) AS order_count
FROM order_items oi
JOIN products p ON oi.product_id = p.product_id
GROUP BY p.product_id, p.product_category_name
ORDER BY order_count DESC;
--41-Top 10 Most Reviewed Products
SELECT TOP 10
   p.product_id,
   p.product_category_name,
   COUNT(r.review_id) AS review_count,
   AVG(r.review_score) AS avg_review_score
FROM order_reviews r
JOIN order_items oi ON r.order_id = oi.order_id
JOIN products p ON oi.product_id = p.product_id
GROUP BY p.product_id, p.product_category_name
ORDER BY review_count DESC;
--42-Orders with Product Return Risk
SELECT
   oi.order_id,
   p.product_id,
   p.product_category_name,
   AVG(r.review_score) AS avg_review_score
FROM order_items oi
```

```
JOIN order_reviews r ON oi.order_id = r.order_id

JOIN products p ON oi.product_id = p.product_id

GROUP BY oi.order_id, p.product_id, p.product_category_name

HAVING AVG(r.review_score) < 3;

--43-Number of Sellers

SELECT COUNT(DISTINCT seller_id) AS seller_count FROM sellers

--44-Top Sellers by Revenue or Orders

SELECT TOP 10

   s.seller_id,

   s.seller_city,

   s.seller_state,

   SUM(oi.price) AS total_revenue,

   COUNT(DISTINCT oi.order_id) AS order_count

FROM order_items oi

JOIN sellers s ON oi.seller_id = s.seller_id

GROUP BY s.seller_id, s.seller_city, s.seller_state

ORDER BY total_revenue DESC;

--45-Orders by Customer Region/State

SELECT

   c.customer_state,

   COUNT(DISTINCT o.order_id) AS order_count

FROM orders o

JOIN customers c ON o.customer_id = c.customer_id

GROUP BY c.customer_state

ORDER BY order_count DESC;

--46-Monthly Orders Overview
```

```
SELECT
   FORMAT(order_purchase_timestamp, 'yyyy-MM') AS month,
   COUNT(*) AS total_orders,
   COUNT(CASE WHEN order_status = 'delivered' THEN 1 END) AS delivered_orders,
   COUNT(CASE WHEN order_status = 'canceled' THEN 1 END) AS canceled_orders,
   COUNT(CASE WHEN order_status IN ('created', 'approved', 'processing', 'invoiced') THEN
1 END) AS pending_orders
FROM orders
GROUP BY FORMAT(order_purchase_timestamp, 'yyyy-MM')
ORDER BY month desc
-- 47. % Revenue Reconciliation
-- (Total Revenue from delivered orders / Expected Revenue from approved orders) * 100
SELECT
   SUM(CASE WHEN o.order_status = 'delivered' THEN oi.price + oi.freight_value ELSE 0
END) AS actual_revenue,
   SUM(CASE WHEN o.order_status = 'approved' THEN oi.price + oi.freight_value ELSE 0
END) AS expected_revenue,
   (SUM(CASE WHEN o.order_status = 'delivered' THEN oi.price + oi.freight_value ELSE 0
END) /
    NULLIF(SUM(CASE WHEN o.order_status = 'approved' THEN oi.price + oi.freight_value
ELSE 0 END), 0)) * 100 AS revenue_reconciliation_pct
FROM orders o
JOIN order_items oi ON o.order_id = oi.order_id
WHERE o.order_status IN ('delivered', 'approved');
--48-  interactive (Slicers for Order Status, Payment Type, Date)
-- For order status slicer:
SELECT DISTINCT order_status FROM orders
```

```
-- For payment type slicer:
SELECT DISTINCT payment_type FROM order_payments-- For date range slicer (min/max
dates):
SELECT
   MIN(order_purchase_timestamp) AS min_date,
   MAX(order_purchase_timestamp) AS max_date
FROM orders
--49- Drill-Down to Order-Level Details
-- Simplified Order Drill-Down (KPI 49)
SELECT
   o.order_id,
   o.order_status,
   o.order_purchase_timestamp,
   o.order_approved_at,
   o.order_delivered_carrier_date,
   o.order_delivered_customer_date,
   c.customer_city,
   c.customer_state,
   COUNT(oi.product_id) AS product_count,
   SUM(oi.price) AS subtotal,
   SUM(oi.freight_value) AS shipping_cost,
   SUM(oi.price + oi.freight_value) AS order_total,
   (SELECT SUM(payment_value)
    FROM order_payments p
    WHERE p.order_id = o.order_id) AS total_paid,
   (SELECT AVG(review_score)
    FROM order_reviews r
    WHERE r.order_id = o.order_id) AS avg_rating
FROM orders o
JOIN customers c ON o.customer_id = c.customer_id
JOIN order_items oi ON o.order_id = oi.order_id
GROUP BY
   o.order_id,
   o.order_status,
   o.order_purchase_timestamp,
   o.order_approved_at,
   o.order_delivered_carrier_date,
   o.order_delivered_customer_date,
   c.customer_city,
   c.customer_state;
```