```sql
--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
)
```

```sql
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
```

```sql
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
```