AMAZON SALES SQL QUERIES

1. Find total no of orders, total quantity sold, and total sales amount.

```
SELECT * FROM sales;

SELECT COUNT(*) AS no_of_orders, SUM(qty) AS total_qty_sold, SUM(amount) AS
total_sales

FROM sales;
```

2. Some orders have amount = 0 or qty = 0. Write a query to find how many such orders exist and what percentage they form of total orders.

```
WITH ordersummary AS(

SELECT COUNT(order_ID) AS totalorders,

        COUNT(CASE WHEN amount = 0 OR qty = 0 THEN order_ID

                END) AS zeroordercount

FROM sales

)

SELECT totalorders, zeroordercount, ROUND((zeroordercount*100.0/totalorders),2)
orderpercent

FROM ordersummary;
```

3. Analyze total quantity and revenue by product category.

```
SELECT category, SUM(qty) AS total_qty, SUM(amount) AS revenue

FROM sales

GROUP BY category
```

4. Identify top 5 states by total sales amount.

```
SELECT shipState, SUM(amount) AS total_sales

FROM sales

GROUP BY shipState
```

```
ORDER BY 2 DESC
LIMIT 5;
```

5. Find cities with highest order volume.

```
SELECT shipCity, COUNT(*) AS highest_orders
FROM sales
GROUP BY shipCity
ORDER BY 2 DESC;
```

6. Calculate total sales and quantity sold per month.

```
SELECT TO_CHAR(cleanDate, 'MM-YYYY') AS month, SUM(amount) total_sales,
SUM(qty) total_qty
FROM sales
GROUP BY TO_CHAR(cleanDate, 'MM-YYYY')
ORDER BY 2 DESC;
```

7. For each category, calculate its percentage contribution to total sales.

```
SELECT category, SUM(amount) cat_sales,
        ROUND(SUM(amount)*100.0/(SELECT SUM(amount) FROM sales),2) AS
cat_percentage
FROM sales
GROUP BY 1
ORDER by 2 DESC;
```

OR

```
SELECT category,
        ROUND(SUM(amount) OVER(PARTITION BY category)*100.0/SUM(amount)
OVER(),2) AS cat_percentage
FROM sales
```

```
ORDER BY 2 DESC;
```

8. Compare monthly sales to identify growth or decline trends.

```
SELECT month_year, total_sales, LAG(total_sales, 1) OVER(ORDER BY month_year ) AS previous_month_sales
FROM
        (SELECT TO_CHAR(cleanDate, 'MM-YYYY') AS month_year, SUM(amount) AS total_sales
         FROM sales
         GROUP BY 1) t
ORDER BY 1;
```

9. Analyze week-wise order volume to understand short-term demand patterns.

```
SELECT EXTRACT(WEEK FROM cleanDate) AS Week, EXTRACT(YEAR FROM cleanDate) AS year, COUNT(order_ID) AS noOfOrders
FROM sales
GROUP BY 1,2
ORDER BY 1;
```

10. Calculate percentage of orders that are Shipped vs On the Way.

```
SELECT (shippedOrders*100.0/total_orders) AS shipPercentage,
(OnWayOrders*100.0/total_orders) AS onWayPercentage
FROM (SELECT COUNT(order_ID) AS total_orders,
        SUM(CASE WHEN courierStatus = 'Shipped' THEN 1 ELSE 0 END) AS shippedOrders,
        SUM(CASE WHEN courierStatus = 'On the Way' THEN 1 ELSE 0 END) AS OnWayOrders
                FROM sales) t
;
```

11. Compare total orders and sales by fulfillment method (fulfilledby).

```
SELECT CASE
                WHEN fulfilledby = 'Easy Ship' THEN 'Easy Ship'
                ELSE 'Non-Easy Ship'
                END AS fulfillmentMethod,
        COUNT(order_ID) AS total_orders,
        SUM(amount) AS total_sales
FROM sales
GROUP BY 1
ORDER BY 3;
```

12. Analyze whether expedited shipping results in higher sales or order volume.

```
SELECT shipServiceLevel, COUNT(order_ID) AS order_vol, SUM(amount) AS sales
FROM sales
GROUP BY 1
;
```

13. Compare sales performance between B2B and non-B2B orders.

```
SELECT CASE
                WHEN B2B = FALSE THEN 'B2C sales'
                ELSE 'B2B sales'
                END typeofsale,
        SUM(amount) totalsales
FROM sales
GROUP BY 1
ORDER BY 2 DESC;
```

14. Identify categories with high revenue but low quantity sold.

```
SELECT category, SUM(qty) AS qty_sold, SUM(amount) AS total_rev
FROM sales
GROUP BY 1
ORDER BY 3 DESC, 2 ASC;
```

15. Identify the top 5 highest-value orders based on total order amount.
(Assume an order can have multiple line items.)

```
SELECT order_ID, SUM(COALESCE(amount,0)) AS total_amount
FROM sales
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5;
```

16. Find the number of orders that are not shipped yet (i.e., not ‘Shipped’)
and group them by courier status.

```
SELECT courierStatus, COUNT(order_ID) AS not_shipped_orders
FROM sales
WHERE courierStatus = 'Cancelled' OR courierStatus = 'Unshipped' OR courierStatus = 'On the Way'
GROUP BY courierStatus
ORDER BY 2 DESC;
```

OR

```
SELECT courierStatus, COUNT(order_ID) AS not_shipped_orders
FROM sales
WHERE courierStatus <> 'Shipped'
GROUP BY courierStatus
ORDER BY 2 DESC;
```

17. Find cities where: Order volume is high but total sales are comparatively low.

```
SELECT shipCity, COUNT(order_ID) AS order_vol, SUM(amount) AS total_sales
FROM sales
GROUP BY 1
ORDER BY 2 DESC, 3 ASC;
```