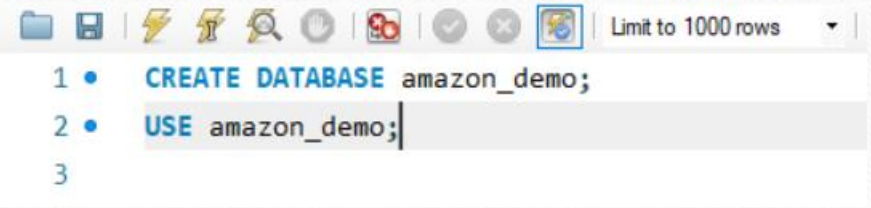
Project :-

amazon Analysis

Presented by :- Aman Dubey

Amazon — Sales, Discounts & Profit Analysis

1. This project shows how Amazon manages sales, discounts, and profit using structured SQL data.
2. We use a simplified database model to analyze transactions realistically.
3. SQL queries reveal revenue flow, discount patterns, and cost impacts.
4. Insights highlight both performance strengths and operational inefficiencies.



Data Model (tables)

1. Customers, products, Orders, Order items, Discounts, Payments, Returns.
2. Granularity: order -> Multiple order items; Discounts at item or order level.

```sql
CREATE TABLE customers(
customer_id INT PRIMARY KEY,
name VARCHAR(100),
email VARCHAR(150),
city VARCHAR(50),
signup_date DATE
);

CREATE TABLE products(
product_id INT PRIMARY KEY,
name VARCHAR(150),
category VARCHAR(50),
cost_price DECIMAL(10,2),
list_price DECIMAL(10,2)
);
```

```sql
CREATE TABLE orders(
order_id INT PRIMARY KEY,
customer_id INT,
order_date DATE,
order_status VARCHAR(30),
shipping_cost DECIMAL(10,2),
FOREIGN KEY (customer_id) REFERENCES customers(customer_id
);

CREATE TABLE order_items(
item_id INT PRIMARY KEY,
order_id INT,
product_id INT,
qty INT,
unit_price DECIMAL(10,2), -- selling price per unit
discount_amount DECIMAL(10,2) DEFAULT 0,
FOREIGN KEY(order_id) REFERENCES orders(order_id),
FOREIGN KEY(product_id) REFERENCES products(product_id)
);
```

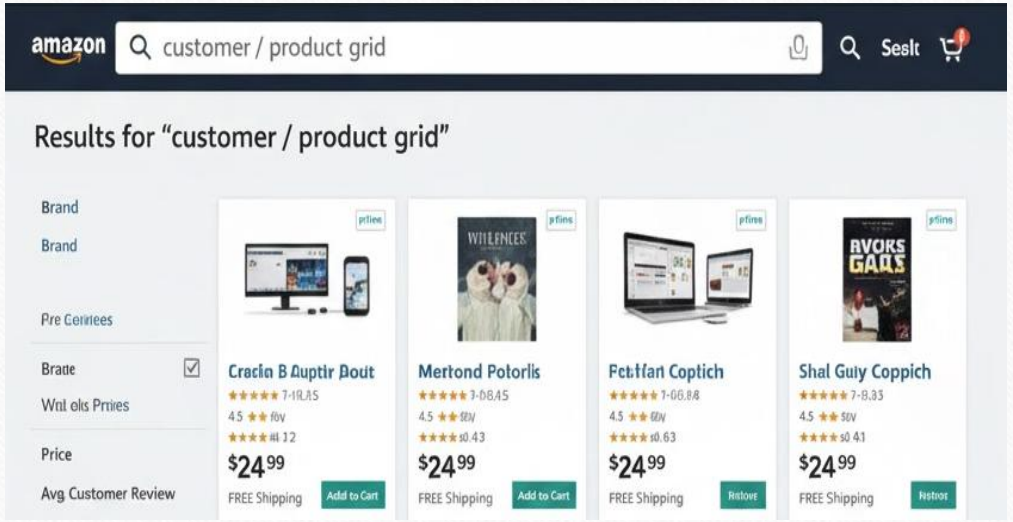
```sql
CREATE TABLE payments(
payment_id INT PRIMARY KEY,
order_id INT,
payment_method VARCHAR(50),
amount_paid DECIMAL(12,2),
payment_date DATE,
FOREIGN KEY(order_id) REFERENCES
orders(order_id)
);

CREATE TABLE returns(
return_id INT PRIMARY KEY,
order_id INT,
item_id INT,
return_date DATE,
return_amount DECIMAL(10,2)
);
```

Sample Master Data

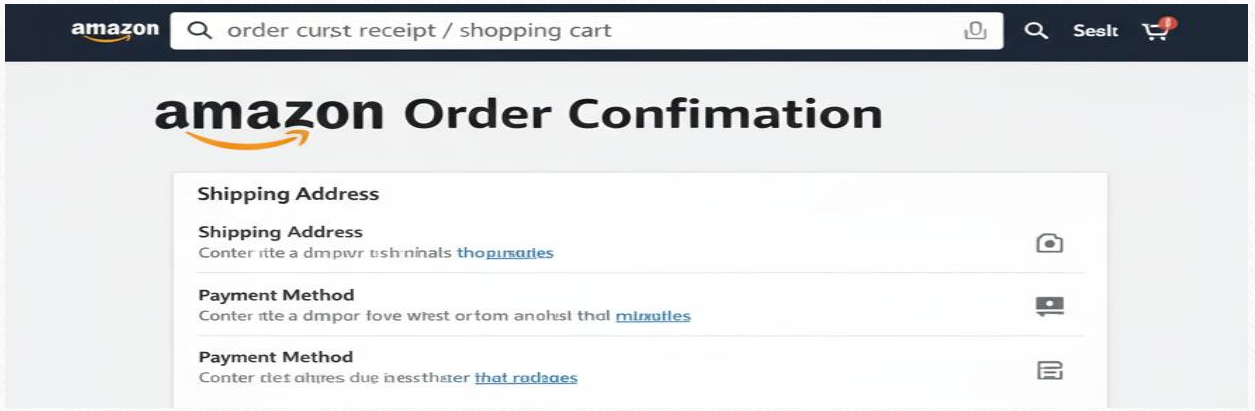
1. Customer data captures who buys, when they joined, and where they live.
2. Product data stores name, category, cost price, and list price.
3. These attributes allow margin, segment, and category-level insights.
4. Clean and structured master data improves downstream analysis accuracy.

```
58 • INSERT INTO customers VALUES
59   (1,'Ravi Kumar','ravi@example.com','Mumbai','2023-01-10'),
60   (2,'Sana Verma','sana@example.com','Delhi','2022-11-04'),
61   (3,'Ankit Rao','ankit@example.com','Bengaluru','2024-02-20');
62
63 • INSERT INTO products VALUES
64   (101,'Wireless Headphones','Electronics',1200.00,1999.00),
65   (102,'Stainless Steel Water Bottle','Home & Kitchen',150.00,399.00),
66   (103,'Running Shoes','Sports',800.00,1499.00),
67   (104,'LED Desk Lamp','Home & Kitchen',200.00,599.00);
```



Sample Transactional Data

1. Orders store total activity, status, dates, and shipping cost.
2. Order items define what was sold, quantity, selling price, and discount.
3. Payments link financial confirmation back to each order.
4. This layer forms the foundation for revenue, AOV, and demand analysis.




```
69 •  INSERT INTO orders VALUES
70    (5001,1,'2024-09-01','Completed',50.00),
71    (5002,2,'2024-09-03','Completed',40.00),
72    (5003,3,'2024-10-05','Completed',60.00);
73
74 •  INSERT INTO order_items VALUES
75    (1,5001,101,1,1799.00,100.00),
76    (2,5001,102,2,349.00,0.00),
77    (3,5002,103,1,1299.00,200.00),
78    (4,5003,104,3,549.00,0.00);
79
80 •  INSERT INTO payments VALUES
81    (9001,5001,'Credit Card',2497.00,'2024-09-01'),
82    (9002,5002,'UPI',1099.00,'2024-09-03'),
83    (9003,5003,'Netbanking',1707.00,'2024-10-05');
```

Discounts & Returns

```sql
CREATE TABLE discounts(
discount_id INT PRIMARY KEY,
name VARCHAR(100),
start_date DATE,
end_date DATE,
discount_pct DECIMAL(5,2)
);

INSERT INTO discounts VALUES
(1,'Festive Sale','2024-09-01','2024-09-10',10.00);

INSERT INTO returns VALUES
(1,5002,3,'2024-09-10',1299.00);
```

1. Discounts impact profitability and must be tracked item-by-item.
2. Campaign discounts influence volume but reduce margin.
3. Returns directly reduce recognized revenue and inflate logistics cost.
4. Evaluating both is essential for accurate net profit measurement.

Key SQL: Sales & Revenue

```sql
SELECT SUM(unit_price * qty) AS gross_sales
FROM order_items;

SELECT SUM(discount_amount) AS total_discounts
FROM order_items;

SELECT
(SUM(unit_price*qty) - SUM(discount_amount) + SUM(o.shipping_cost)
- COALESCE((SELECT SUM(return_amount) FROM returns),0))
AS net_revenue
FROM order_items i
JOIN orders o ON i.order_id = o.order_id;

SELECT AVG(order_total) FROM (
SELECT o.order_id, SUM(i.unit_price*i.qty - i.discount_amount)
+ o.shipping_cost AS order_total
FROM orders o JOIN order_items i ON o.order_id = i.order_id
GROUP BY o.order_id
) t;
```

1. Gross sales represent total item value before discounts.
2. Net revenue adjusts for discounts, returns, and shipping.
3. AOV shows the average customer spend per order.
4. These metrics help measure Amazon’s overall sales performance.

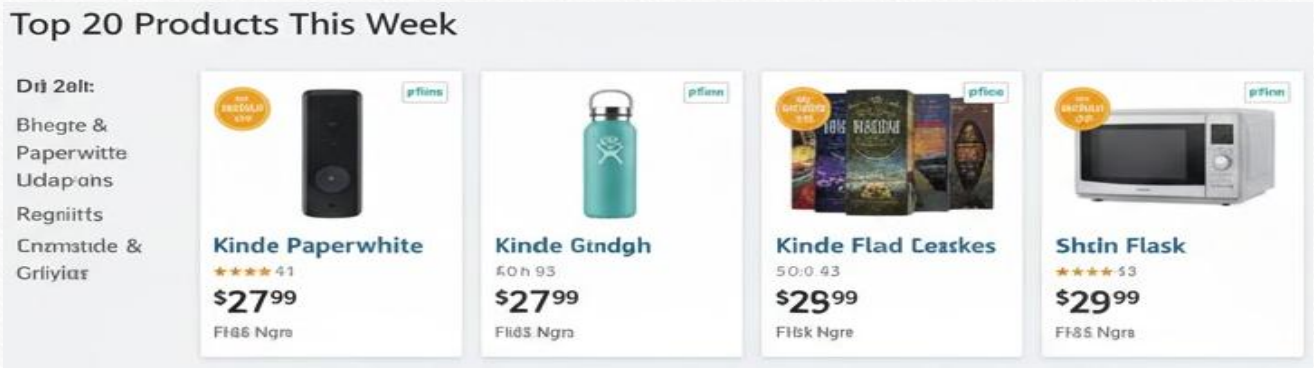
Key SQL: Profit & Margin

1. Profit depends on selling price, cost price, discounts, and shipping.
2. Product-level margin reveals which items drive profitability.
3. High revenue doesn’ t guarantee high profit — cost structure matters.
4. These insights guide decisions on pricing and inventory strategy.

```sql
SELECT
i.item_id,
i.order_id,
p.name,
(i.unit_price - p.cost_price) * i.qty - i.discount_amount
AS approx_profit
FROM order_items i
JOIN products p ON i.product_id = p.product_id;

SELECT
SUM((i.unit_price - p.cost_price)*i.qty - i.discount_amount)
- SUM(o.shipping_cost) AS est_profit
FROM order_items i
JOIN products p ON i.product_id = p.product_id
JOIN orders o ON i.order_id = o.order_id;
```

Analysis Queries: Customers & Products




```sql
SELECT p.product_id, p.name, SUM(i.unit_price*i.qty - i.discount_amount) AS revenue
FROM order_items i JOIN products p ON i.product_id = p.product_id
GROUP BY p.product_id,p.name
ORDER BY revenue DESC LIMIT 5;

SELECT COUNT(DISTINCT customer_id) AS total_customers,
SUM(CASE WHEN cnt>1 THEN 1 ELSE 0 END) AS repeat_customers
FROM (
SELECT customer_id, COUNT(order_id) AS cnt FROM orders GROUP BY customer_id
) t;
```

1. Top-selling products show where demand is strongest.
2. Repeat customer rate indicates long-term business health.
3. Discount sensitivity varies across categories and customer types.
4. Discount sensitivity varies across categories and customer types.

Time-series & Campaign Impact

1. Monthly revenue trends reveal growth patterns and seasonality.
2. Comparing pre-sale and sale periods measures campaign effectiveness.
3. Spikes in volume may not always mean higher profit.
4. Time-based analysis supports better inventory and pricing planning.

```sql
SELECT
    DATE_FORMAT(o.order_date, '%Y-%m') AS month,
    SUM((i.unit_price * i.qty) -
    i.discount_amount + o.shipping_cost) AS total_revenue
FROM orders o
JOIN order_items i ON o.order_id = i.order_id
GROUP BY DATE_FORMAT(o.order_date, '%Y-%m')
ORDER BY month;
```

Findings & Recommendations

1. Discounts boost sales volume but can erode margins if misused.
2. High-ticket returns significantly damage net revenue.
3. Customer retention must be improved with targeted strategies.
4. Better discount planning and product-level optimization can raise profit.

Recommended

```sql
SELECT
CASE WHEN discount_flag>0 THEN 'Discounted' ELSE 'FullPrice' END AS type,
AVG(order_profit) AS avg_order_profit
FROM (
SELECT o.order_id,
SUM((i.unit_price - p.cost_price)*i.qty - i.discount_amount) - o.shipping_cost AS order_profit,
SUM(CASE WHEN i.discount_amount>0 THEN 1 ELSE 0 END) AS discount_flag
FROM orders o
JOIN order_items i ON o.order_id = i.order_id
JOIN products p ON i.product_id = p.product_id
GROUP BY o.order_id
) t
GROUP BY type;
```