```
--Exploratory Data Analysis (EDA) with SQL

--1. Data Aggregation and Grouping:

--Order Amounts by Customer:


SELECT Customer_ID, SUM(Amount) AS Total_Spent FROM ORDER_ GROUP BY Customer_ID ORDER BY Total_Spent DESC;  --Sum=532500  --Avg=3328.12


--Shipping Status by Customer:

 SELECT Customer_ID, Status, COUNT(*) AS Shipments FROM Shipping GROUP BY Customer_ID, Status;


--2. Trend Analysis:

--Monthly Order Trends:

 --SELECT DATE_FORMAT(Order_Date, '%Y-%m') AS Month, SUM(Amount) FROM ORDER_ GROUP BY Month ORDER BY Month;  --No Order_Date


--Join Analysis (Customer Orders and Shipping):


SELECT o.Customer_ID, c.FIRST, o.Order_ID, s.Status

FROM ORDER_ o
```

```
JOIN Customer c ON o.Customer_ID = c.Customer_ID
LEFT JOIN Shipping s ON o.Customer_ID = s.Customer_ID;


--4. Outlier Detection:
--High Order Amounts:

SELECT * FROM Order_ WHERE Amount > (SELECT AVG(Amount) + 3 * STDDEV(Amount) FROM ORDER_);

--5. Pivoting and Unpivoting Data:
--Shipping Status Pivot by Customer:

SELECT * FROM (
  SELECT Customer_ID, Status FROM Shipping
) AS SourceTable
PIVOT (
  COUNT(Status) FOR Status IN ('Pending', 'Delivered')
) AS PivotTable;
```

```
--6. Segmentation Analysis:

--Customer Segmentation by Total Spend:


SELECT CASE
  WHEN Total_Spent > 1000 THEN 'High Value'
  WHEN Total_Spent BETWEEN 500 AND 1000 THEN 'Medium Value'
  ELSE 'Low Value'
END AS Segment, COUNT(*)
FROM (SELECT Customer_ID, SUM(Amount) AS Total_Spent FROM Order_ GROUP BY Customer_ID) AS Spending
GROUP BY Segment;



--7. Cumulative Analysis:

--Running Total of Orders by Date (if available):


--SELECT Order_Date, SUM(Amount) OVER (ORDER BY Order_Date) AS Running_Total FROM Order;


SELECT item, SUM( Amount) OVER (ORDER BY Order_Date) AS Running_Total FROM Order
--8. Drill-Down Analysis:
```

--Orders by Category and Item (if available):

```
SELECT  ITEM, SUM(Amount) FROM ORDER_ GROUP BY ITEM ORDER BY SUM(Amount) DESC;
```

--9. Hypothesis Testing (Basic):

--Comparing Average Order Amount by Shipping Status:

```
SELECT s.Status, AVG(o.Amount)
FROM ORDER_ o
JOIN Shipping s ON o.Customer_ID = s.Customer_ID
GROUP BY s.Status;
```