Problem Statement

You are working as a data analyst for a retail company, and you are tasked with analyzing their sales data to uncover key insights that can guide business decisions. The database consists of the following tables:

1. **Sales:** Contains information about sales transactions, including `SaleDate`, `Amount`, `Boxes`, `Customers`, and foreign keys to other tables.
2. **People:** Lists salespersons and their teams, with columns like `Salesperson`, `Team`, and `SPID`.
3. **Products:** Includes details of products with columns like `Product`, `Category`, and `PID`.
4. **Geo:** Contains geographical data with columns like `Geo`, `GeoID`.

Your objectives include:

1. • **Calculate Sales Performance Metrics**
   How can we calculate key performance metrics like revenue per unit sold to assess efficiency?
2. • **Filter Sales Data Based on Specific Conditions**
   How can we filter sales data based on thresholds, date ranges, or specific attributes like geography?
3. • **Categorize Sales Data Using Conditional Logic (CASE)**
   How can we group sales into meaningful categories for easier analysis and decision-making?
4. • **Sort and Order Data**
   How can we organize data by priority, such as sorting sales by highest amount or latest date?
5. • **Aggregate Functions to Summarize Data**
   How can we summarize large datasets using metrics like total sales, averages, or counts?
6. • **Subqueries for Nested Insights**
   How can we retrieve nested insights, such as finding the second-highest sale or comparing to averages?
7. • **Data Retrieval and Filtering Using String and Pattern Matching**
   How can we filter or retrieve data based on specific text patterns or attributes like names or teams?
8. • **Date-Based Analysis**
   How can we analyze data for specific time periods, such as filtering by year, date range, or day of the week?
9. • **Joining Multiple Tables**
   How can we combine data from multiple tables to generate insights across dimensions like products, geography, or teams?
10. • **Top-N Analysis**
    How can we identify the top-performing entities, such as the best-selling products or highest revenue-generating regions?

1. Calculate Sales Performance Metrics (e.g., "Amount per box")

- **Formula:**

  Sql code:
  ```
  SELECT SaleDate, Amount, Boxes, Amount / Boxes AS "Amount per box"
  FROM sales;
  ```

- **Purpose:** Calculates the revenue generated per box sold to measure sales efficiency.

---

2. Filter Sales Data Based on Specific Conditions

- **Filter by Amount Exceeding a Threshold:**

  Sql code:
  ```
  SELECT * FROM sales WHERE amount > 10000;
  ```

- **Filter by Date Range:**

  Sql code:
  ```
  SELECT * FROM sales WHERE amount > 10000 AND SaleDate >= '2022-01-
  01';
  ```

- **Filter by Geography:**

  Sql code:
  ```
  SELECT * FROM sales WHERE GeoID = 'g1' ORDER BY PID, Amount DESC;
  ```

- **Filter by Box Range (Using Operators):**

  Sql code:
  ```
  SELECT * FROM sales WHERE boxes > 0 AND boxes <= 50;
  ```

- **Filter by Box Range (Using BETWEEN):**

  Sql code:
  ```
  SELECT * FROM sales WHERE boxes BETWEEN 0 AND 50;
  ```

---

3. Categorize Sales Data Using Conditional Logic (CASE)

- **Formula:**

  ```
  SELECT SaleDate, Amount,
         CASE
             WHEN amount < 1000 THEN 'Under 1k'
             WHEN amount < 5000 THEN 'Under 5k'
             WHEN amount < 10000 THEN 'Under 10k'
             ELSE '10k or more'
         END AS "Amount category"
  FROM sales;
  ```

- **Purpose:** Groups sales into meaningful categories for easier analysis.

---

4. Sort and Order Data

- **Sort Sales by Amount Descending:**

  Sql code:
  ```
  SELECT * FROM sales WHERE amount > 10000 ORDER BY amount DESC;
  ```

- **Sort Sales by Geography and Product ID:**

  Sql code:
  ```
  SELECT * FROM sales WHERE GeoID = 'g1' ORDER BY PID, Amount DESC;
  ```

- **Sort Sales by Date and Amount in Descending Order:**

  Sql code:
  ```
  SELECT p.Salesperson, s.Amount, s.SaleDate
  FROM sales s
  JOIN people p ON p.SPID = s.SPID
  WHERE SaleDate >= '2022-01-01' AND SaleDate <= '2022-01-30'
  ORDER BY SaleDate, Amount DESC;
  ```

---

5. Aggregate Functions to Summarize Data

- **Total and Average Sales by Product:**

  Sql code:
  ```
  SELECT Product, SUM(Amount) AS "Total Amount"
  FROM sales s
  JOIN products pr ON pr.PID = s.PID
  GROUP BY Product
  ORDER BY SUM(Amount) DESC;
  ```

- **Sales Count and Sum by Team and Category:**

  Sql code:
  ```
  SELECT Category, Team, SUM(Amount), SUM(Boxes)
  FROM sales s
  JOIN people p ON p.SPID = s.SPID
  JOIN products pr ON pr.PID = s.PID
  WHERE Team <> ''
  GROUP BY Team, Category
  ORDER BY Category;
  ```

- **Sales Count by Team:**

  Sql code:
  ```
  SELECT Team, COUNT(*) FROM people GROUP BY Team;
  ```

---

6. Subqueries for Nested Insights

- **Find Second-Highest Sale Amount:**

```
Sql code:
SELECT MAX(Amount)
FROM sales
WHERE Amount < (SELECT MAX(Amount) FROM sales);
```

- **Find Sales Below the Average Amount:**

```
Sql code:
SELECT Amount, Customers, (SELECT AVG(Amount) FROM sales)
FROM sales
WHERE Amount < (SELECT AVG(Amount) FROM sales);
```

---

7. Data Retrieval and Filtering Using String and Pattern Matching

- **Filter by Team Using OR Operator:**

```
Sql code:
SELECT * FROM people WHERE team = 'Delish' OR team = 'Jucies';
```

- **Filter by Team Using IN Operator:**

```
Sql code:
SELECT * FROM people WHERE team IN ('Delish', 'Jucies');
```

- **Filter by Salesperson Name Starting with 'B':**

```
Sql code:
SELECT * FROM people WHERE salesperson LIKE 'B%';
```

- **Filter by Salesperson Name Containing 'B':**

```
Sql code:
SELECT * FROM people WHERE salesperson LIKE '%B%';
```

---

8. Date-Based Analysis

- **Filter by Weekday:**

```
Sql code:
SELECT SaleDate, Amount, Boxes, WEEKDAY(SaleDate) AS "Day of week"
FROM sales
WHERE WEEKDAY(SaleDate) = 4;
```

- **Filter by Year:**

```
Sql code:
SELECT SaleDate, Amount
FROM sales
WHERE Amount > 10000 AND YEAR(SaleDate) = 2022
ORDER BY Amount DESC;
```

---

9. Joining Multiple Tables

- **Join and Filter by Geography and Amount:**

Sql code :

```
SELECT s.SaleDate, s.Amount, p.Salesperson, pr.Product, g.Geo
FROM sales s
JOIN people p ON p.SPID = s.SPID
JOIN products pr ON pr.PID = s.PID
JOIN geo g ON g.GeoID = s.GeoID
WHERE s.Amount > 10000 AND g.Geo IN ('India', 'New Zealand')
ORDER BY Product, Geo;
```

---

10. Top-N Analysis

- **Top 5 Products by Sales Amount:**

Sql code:

```
SELECT Product, SUM(Amount) AS "Total Amount"
FROM sales s
JOIN products pr ON pr.PID = s.PID
GROUP BY Product
ORDER BY SUM(Amount) DESC
LIMIT 5;
```

This structure now includes all your SQL formulas and ensures they are grouped under relevant categories for clarity. Let me know if you'd like further adjustments!