-- Category name for which maximum subcategories are present

Processes the SQL queries involve:

1. Column Selection: Selects category key, category name, and subcategory count.
2. Table Join: Joins category_lookup with subcategory_lookup on ProductCategoryKey.
3. Subcategory Counting: Counts subcategories per category.
4. Grouping: Groups results by category key and name.
5. Ordering: Orders categories by subcategory count in descending order.

Use of the Query:

1. Analyzes the number of subcategories in each product category.
2. Identifies categories with the most subcategories.
3. Validates data integrity.
4. Provides insights for inventory management and reporting.

Category and Subcategory Distribution:

Processes the SQL Query Includes:

1. Column Selection: Selects category name, subcategory name, and product count.
2. Table Joins:

Joins product_lookup with subcategory_lookup on ProductSubcategoryKey.

Joins subcategory_lookup with category_lookup on ProductCategoryKey.

3. Product Counting: Counts distinct products in each subcategory.
4. Grouping: Groups results by category name and subcategory name.
5. Ordering: Orders categories by name and product count in descending order.

Use of the Query:

1. Analyzes the distribution of products across categories and subcategories.
2. Identifies which subcategories have the highest number of products within each category.
3. Helps in understanding product organization and hierarchy.

-- Top 5 Subcategories with Most Products

Processes the SQL Query Includes:

1. Column Selection: Selects the ProductSubcategoryKey and the count of distinct productkey.
2. Product Counting: Counts distinct products for each subcategory.
3. Grouping: Groups results by ProductSubcategoryKey.
4. Ordering: Orders subcategories by product count in descending order.
5. Limiting: Limits the results to the top 5 subcategories.

Use of the Query:

1. Identifies the top 5 subcategories with the highest number of products.
2. Provides insights into product distribution across subcategories.
3. Assists in inventory management by highlighting subcategories with the most products.

--convert the orderdatetime column to datetime

Processes the SQL Query Includes:

1. Column Modification: Alters the OrderDate column in three tables (sales_2022, sales_2021, sales_2020) to change its data type to Datetime.

Use of the Query:

2. Ensures that the OrderDate column is stored as a Datetime data type, allowing for accurate date and time operations.
3. Facilitates date-based queries, comparisons, and calculations, enhancing data analysis and reporting capabilities.
4. Standardizes the data type across multiple tables for consistency and integrity in the database schema.

-- Quarterly Sales Analysis by Year for 2020, 2021, and 2022

Processes the SQL Query Includes:

1. Subquery Execution: Executes three subqueries, each querying sales data from sales_2022, sales_2021, and sales_2020 tables, joined with product_lookup to calculate total sales for each quarter.
2. Quarterly Aggregation: Aggregates sales data by quarter (quart) and year (year) using QUARTER() and YEAR() functions.
3. Conditional Summation: Uses CASE statements within SUM() functions to sum total sales based on the year (2022, 2021, 2020).
4. Data Round-Up: Rounds the total sales amount to ensure accuracy in financial reporting.
5. Final Grouping: Groups results by quarter (quart) to consolidate sales data across years.
6. Sorting: Orders results by quarter (quart) in descending order to show the latest quarters first.

Use of the Query:

1. Yearly Comparison: Provides a comparative analysis of quarterly sales across three consecutive years (2020, 2021, and 2022).
2. Performance Insights: Helps in identifying trends and patterns in sales performance over quarters, aiding in strategic planning and decision-making.
3. Financial Reporting: Facilitates accurate financial reporting by summarizing sales data by quarter and year, crucial for budgeting and forecasting.

-- Average order value for the subcategory for the year 2021

Processes the SQL Query Includes:

1. Subquery Execution: Executes a subquery to retrieve productkey from sales_2021 table to filter relevant products in product_lookup.
2. Filtering and Aggregation: Filters product_lookup to include only products sold in sales_2021, calculates the average productprice per ProductSubcategoryKey, and truncates the result to two decimal places.
3. Grouping: Groups results by ProductSubcategoryKey to calculate the average order value for each subcategory.
4. Sorting: Orders results in descending order based on the average order value (average_order_value).

Use of the Query:

1. Performance Measurement: Calculates the average order value for each subcategory based on sales data from the year 2021, providing insights into customer spending patterns.
2. Strategic Insights: Helps in identifying high-value subcategories where customers tend to spend more per order, aiding in pricing strategies and promotional planning.

-- Product Names with Order Quantity Exceeding 10 for the year 2022

Processes the SQL Query Includes:

1. Subquery Execution: Executes an inner subquery to calculate the total orderquantity for each ProductKey in sales_2022.
2. Filtering: Filters results using the HAVING clause to include only products with a total orderquantity greater than or equal to 10.
3. Joining Tables: Joins the filtered results (sub subquery) with product_lookup table on ProductKey to retrieve product details (productname, productprice) for products meeting the criteria.

Use of the Query:

1. Identifying Products: Retrieves product names (productname), prices (productprice), and total order quantities (order_quantity) for products that had orders exceeding 10 units in the year 2022.
2. Sales Analysis: Helps in identifying popular products or products with high demand based on the volume of orders, aiding in inventory management and sales strategy.

-- Average stock period of products

Processes SQL Query Includes:

1. Join Condition: Joins the sales_2020 table (s20) with product_lookup (pl) on ProductKey to associate each sale with its corresponding product.
2. Average Calculation: Computes the average stock period using DATEDIFF() to calculate the difference in days between orderdate and stockdate.
3. Grouping: Groups results by subcategoryname and productname to calculate the average stock period for each product within its subcategory.
4. Sorting: Orders results in descending order based on the calculated stock_period to show products with the longest average stock periods first.

Use of the Query:

1. Stock Management: Provides insights into how long products typically remain in stock before being sold, aiding in inventory management and restocking strategies.
2. Performance Analysis: Helps identify products and subcategories where inventory turnover might be slower or faster, influencing procurement and sales forecasting.
3. Data Integration: Integrates sales data (sales_2020) with product details (product_lookup and subcategory_lookup) to analyze stock periods across different product categories.

-- Checking for null values in table
Processes the Revised SQL Query Includes:

1. Conditional Counting: Uses CASE statements inside SUM() functions to count occurrences where each specified column (orderdate, productkey, customerkey) is null.
2. Result Condition: Evaluates the summed counts to determine if any null values exist

Use of the Query:

1. Null Value Check: Effectively checks for null values in specific columns (orderdate, productkey, customerkey) within the sales_2021 table.
2. Data Quality Assurance: Assures data completeness and integrity, crucial for data analysis and reporting.
3. Error Prevention: Helps in identifying potential issues related to missing data that could impact analytical results or operations relying on complete data sets.

-- Top 5 Customers with Maximum Purchased Quantity Along with Their Annual Income

Processes the SQL Query Includes:

1. Joining Tables: Joins customer_lookup (cl) with sales_2021 (s21) on CustomerKey to link customer data with sales transactions.
2. Grouping and Counting: Groups the results by CustomerKey and annualincome from customer_lookup, and counts the number of transactions (COUNT(*)) for each customer.
3. Sorting: Orders the results in descending order based on item_count, which represents the number of items purchased by each customer (DESC).
4. Limiting Results: Limits the output to the top 5 customers with the highest item_count, providing a focused view of the top purchasers.

Use of the Query:

1. Identifying Top Customers: Retrieves the top 5 customers who made the most purchases in terms of item count in 2021.
2. Customer Segmentation: Provides insights into customer behavior and preferences based on purchasing habits, potentially informing targeted marketing strategies.
3. Financial Insights: Links customer purchasing behavior with their annual income, offering a holistic view of customer spending patterns and potential economic demographics

-- Names and prices of the most expensive and least expensive

Use of the Query:

1. Extreme Price Identification: Retrieves the names and prices of both the most expensive and least expensive products from the product_lookup table.
2. Price Comparison: Provides a quick overview of price extremes within the product catalog, useful for pricing strategy analysis and product positioning.

-- Most ordered subcategory name

Use of the Query:

1. Identifying Popular Subcategories: Determines which subcategory (subcategoryname) has the highest total count of orders across the top 5 most ordered products in 2022.
2. Sales Analysis: Provides insights into product demand within different subcategories, aiding in inventory management and marketing strategies.
3. Data Integration: Integrates sales data (sales_2022) with product details (product_lookup) and subcategory information (subcategory_lookup) to perform comprehensive analysis of product performance by subcategory.