**Task 1: Recursive Query for Analyzing ETF Transaction Hierarchies**

Suppose you have a dataset of ETF transactions, where each transaction may be linked to another one (for example, an original transaction and its associated corrections or cancellations).

**Table transactions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **transaction\_id** | **etf\_symbol** | **amount** | **transaction\_date** | **parent\_transaction\_id** |
| 1 | SPY | 1000.00 | 2024-01-01 09:15:00 | NULL |
| 2 | SPY | 200.00 | 2024-01-02 14:30:00 | 1 |
| 3 | QQQ | 300.00 | 2024-01-03 11:45:00 | 1 |
| 4 | IWM | 500.00 | 2024-01-04 12:00:00 | NULL |
| 5 | IWM | 150.00 | 2024-01-05 14:00:00 | 4 |
| 6 | SPY | 400.00 | 2024-01-10 09:00:00 | 1 |
| 7 | QQQ | 600.00 | 2024-01-12 13:30:00 | 3 |
| 8 | SPY | 200.00 | 2024-01-15 15:45:00 | 2 |
| 9 | IWM | 700.00 | 2024-01-20 12:00:00 | NULL |
| 10 | IWM | 250.00 | 2024-01-22 14:15:00 | 9 |

**Task**:

1. Using a **recursive CTE**, calculate the total transaction amount for each primary transaction (sum all associated transactions, including corrections and cancellations).
2. Return the total spending and the number of associated transactions for each primary transaction.

**Task 2: Sales Trend Analysis Using Window Functions**

Suppose you have sales data for ETFs in different regions. Your task is to compare the sales trends over the months to identify the largest fluctuations in sales.

**Table sales**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **sale\_id** | **region** | **amount** | **sale\_date** | **etf\_symbol** |
| 1 | East | 500.00 | 2024-01-01 | SPY |
| 2 | West | 300.00 | 2024-01-01 | QQQ |
| 3 | East | 700.00 | 2024-02-01 | SPY |
| 4 | West | 800.00 | 2024-02-01 | IWM |
| 5 | East | 600.00 | 2024-03-01 | QQQ |
| 6 | West | 750.00 | 2024-03-01 | SPY |
| 7 | East | 400.00 | 2024-04-01 | IWM |
| 8 | West | 500.00 | 2024-04-01 | QQQ |
| 9 | East | 900.00 | 2024-05-01 | SPY |
| 10 | West | 650.00 | 2024-05-01 | IWM |

**Task**:

1. Use **window functions** (such as **LAG()**, **LEAD()**) to calculate the percentage change in sales for each region between two consecutive months.
2. Identify the **top 3 regions** that had the **highest sales increase** over the last 6 months.

**Task 3: Sales Analysis by ETF Segment**

You have transaction data for ETFs and detailed customer information. You need to compare how different customer segments spend on various ETF categories.

**Table customers**

|  |  |  |  |
| --- | --- | --- | --- |
| **customer\_id** | **name** | **signup\_date** | **country** |
| 1 | John Doe | 2022-01-15 | USA |
| 2 | Alice Smith | 2023-05-10 | UK |
| 3 | Bob Johnson | 2021-11-20 | Canada |
| 4 | Susan Lee | 2023-02-25 | Germany |
| 5 | James Brown | 2020-07-15 | USA |

**Table transactions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **transaction\_id** | **customer\_id** | **amount** | **transaction\_date** | **etf\_symbol** |
| 1001 | 1 | 500.00 | 2024-02-15 | SPY |
| 1002 | 2 | 100.00 | 2024-03-01 | QQQ |
| 1003 | 1 | 700.00 | 2024-02-25 | IWM |
| 1004 | 3 | 400.00 | 2024-02-10 | SPY |
| 1005 | 2 | 600.00 | 2024-03-05 | QQQ |
| 1006 | 4 | 1200.00 | 2024-04-01 | SPY |
| 1007 | 5 | 1300.00 | 2024-04-10 | IWM |
| 1008 | 1 | 400.00 | 2024-05-15 | QQQ |
| 1009 | 2 | 1500.00 | 2024-05-20 | SPY |
| 1010 | 3 | 350.00 | 2024-06-01 | IWM |

**Task**:

1. **Segment customers** based on their total spending:

Assign each customer to a segment based on their total spending.

* + Segment 1: Customers who spent less than 500 USD
  + Segment 2: Customers who spent between 500 and 1500 USD
  + Segment 3: Customers who spent more than 1500 USD

1. For each segment, calculate the **average transaction value** and the **percentage share** spent in each ETF symbol (e.g., "SPY", "QQQ", "IWM").

**PYTHON**

**Task 1: ETF Data Merging and Aggregation**

You are provided with two dataframes, df\_etf\_prices and df\_etf\_trades. The df\_etf\_prices dataframe contains the daily closing price of different ETFs, while thedf\_etf\_trades dataframe contains the trading volumes for each ETF per day.

**Dataframes:**

df\_etf\_prices:

|  |  |  |
| --- | --- | --- |
| **date** | **etf\_symbol** | **closing\_price** |
| 2025-01-01 | SPY | 450.25 |
| 2025-01-01 | QQQ | 375.30 |
| 2025-01-02 | SPY | 455.20 |
| 2025-01-02 | QQQ | 380.50 |
| 2025-01-03 | SPY | 460.10 |
| 2025-01-03 | QQQ | 385.10 |

df\_etf\_trades:

|  |  |  |
| --- | --- | --- |
| **date** | **etf\_symbol** | **trade\_volume** |
| 2025-01-01 | SPY | 1200 |
| 2025-01-01 | QQQ | 800 |
| 2025-01-02 | SPY | 1000 |
| 2025-01-02 | QQQ | 900 |
| 2025-01-03 | SPY | 1500 |
| 2025-01-03 | QQQ | 1100 |

**Task:**

1. Merge df\_etf\_prices and df\_etf\_trades on the date and etf\_symbol columns.
2. Calculate the **dollar volume** for each ETF per day, defined as: dollar\_volume=closing\_price×trade\_volume
3. Return the merged dataframe with the **date**, **etf\_symbol**, **closing\_price**, **trade\_volume**, and **dollar\_volume**.

**Task 2: ETF Price and Volume Correlation Analysis**

You are provided with a dataframe, df\_etf\_data, which contains the closing prices and trading volumes of several ETFs over a month. Your task is to compute the correlation between price and volume for each ETF.

**Dataframe:**

df\_etf\_data:

|  |  |  |  |
| --- | --- | --- | --- |
| **date** | **etf\_symbol** | **closing\_price** | **trade\_volume** |
| 2025-01-01 | SPY | 450.25 | 1200 |
| 2025-01-01 | QQQ | 375.30 | 800 |
| 2025-01-02 | SPY | 455.20 | 1000 |
| 2025-01-02 | QQQ | 380.50 | 900 |
| 2025-01-03 | SPY | 460.10 | 1500 |
| 2025-01-03 | QQQ | 385.10 | 1100 |
| ... | ... | ... | ... |

**Task:**

1. For each ETF, calculate the **Pearson correlation** between the closing\_price and trade\_volume over the available dates.
2. Return a dataframe with **etf\_symbol** and the **correlation coefficient**.

**Task 3: ETF Portfolio Performance Calculation and Rebalancing**

Suppose you have data on an ETF portfolio consisting of multiple ETFs, and the portfolio weights for each ETF. You need to calculate the overall **portfolio return** and rebalance the portfolio based on changes in the prices of ETFs.

**Dataframe:**

df\_etf\_portfolio:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **etf\_symbol** | **weight** | **return\_2025-01-01** | **return\_2025-01-02** | **return\_2025-01-03** |
| SPY | 0.5 | 0.02 | 0.01 | 0.015 |
| QQQ | 0.3 | 0.015 | 0.018 | 0.01 |
| IWM | 0.2 | 0.01 | 0.012 | 0.014 |

**Task:**

1. For each day, calculate the **weighted return** for the portfolio using the formula:

portfolio\_return=∑(weight×return\_of\_etf\_on\_day)

Return the **date** and the **portfolio return** for each day.

1. Rebalance the portfolio based on changes in ETF prices:
   * Assume that on day 1 (2025-01-01), you set a total portfolio value (e.g., $100,000) and allocate it across ETFs based on their initial weights.
   * On each subsequent day, calculate the new value of each ETF based on its price change and rebalance the portfolio to keep the portfolio’s weights the same (e.g., if ETF A has increased in price, you sell some of ETF A and buy more of ETF B to maintain the original weightings).
2. Implement the rebalancing step by calculating the portfolio value for each ETF on each day and adjusting the holdings of each ETF based on the percentage change in price. Then, rebalance the portfolio so that it continues to reflect the original weights.