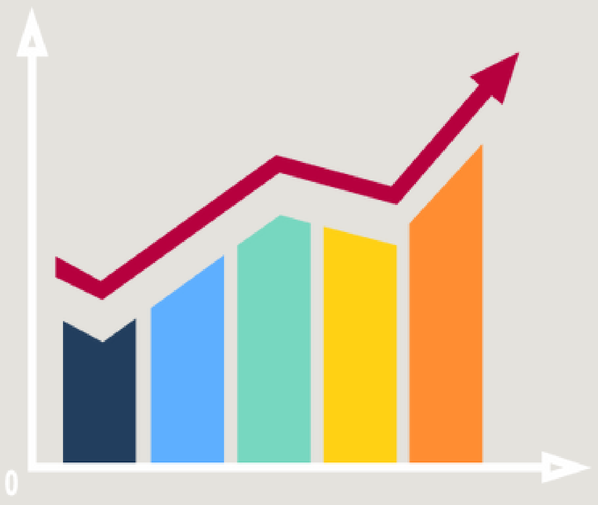
# Analyzing Banking Trends: Customer Transactions and Regional Impact

C H E L S E A
W I L L I A M S



# Analyzing Banking Trends: Customer Transactions and Regional Impact

## PROJECT DESCRIPTION

In the ever-evolving world of banking and finance, understanding customer behavior and the regional impact of transactions plays a crucial role in decision-making and strategic planning.

This project aims to explore and analyze the vast troves of transaction data to gain valuable insights into customer behavior patterns and their implications on different world regions.

## OBJECTIVE

The objective of this project is to explore customer transaction data and identify trends that may impact regional economies and financial systems.

# Analyzing Banking Trends: Customer Transactions and Regional Impact

## TECHNICAL TOOLS

- Python
- Microsoft SQL Server

## DATA SOURCES

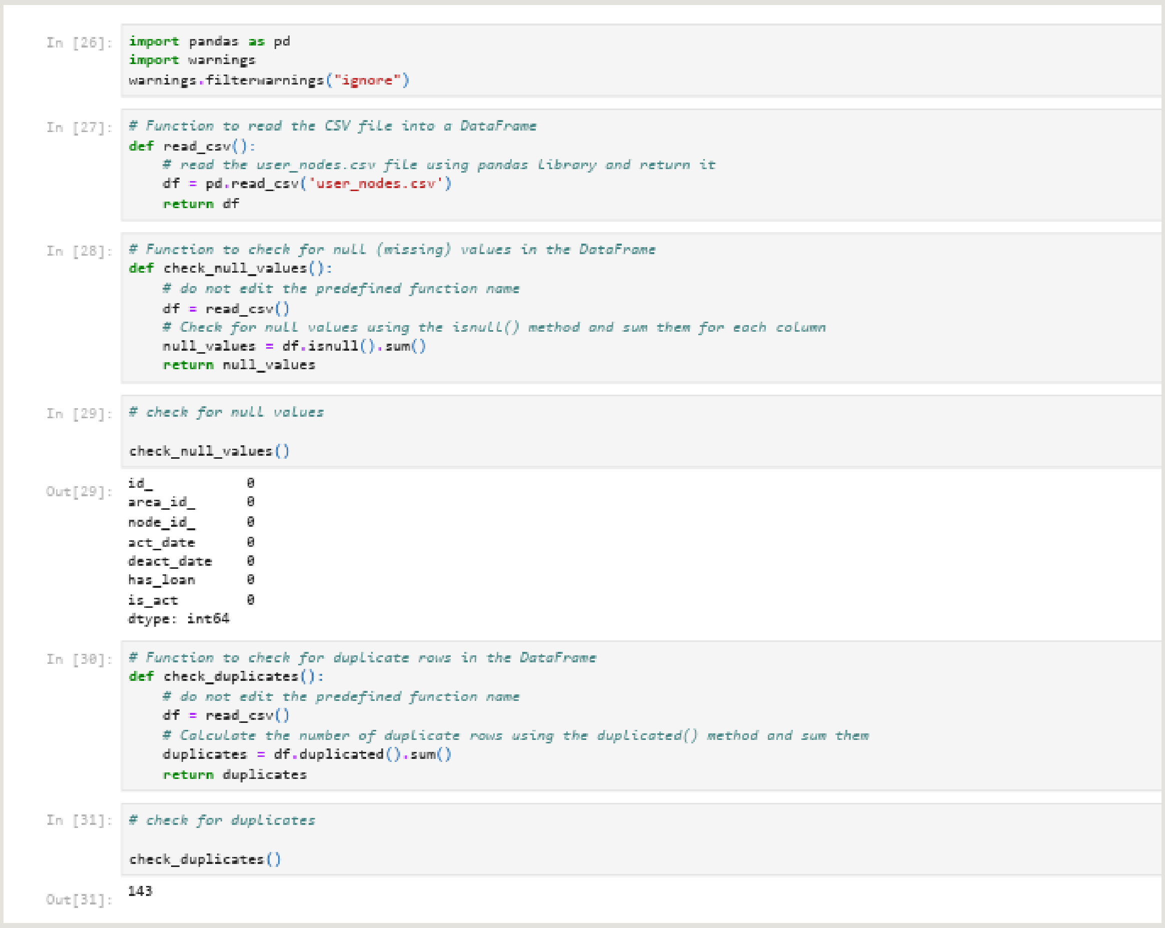
The project leverages three key tables that provide valuable information for analysis:

1. world_regions: This table contains data on various world regions and their corresponding codes and names. It serves as a reference to categorize customers based on their regional affiliation.

2. user_nodes: This table holds details about consumers' banking nodes, including their unique consumer IDs, associated region IDs, node IDs, start dates, and end dates. This data enables us to identify the specific banking nodes to which customers are connected and their duration of association.

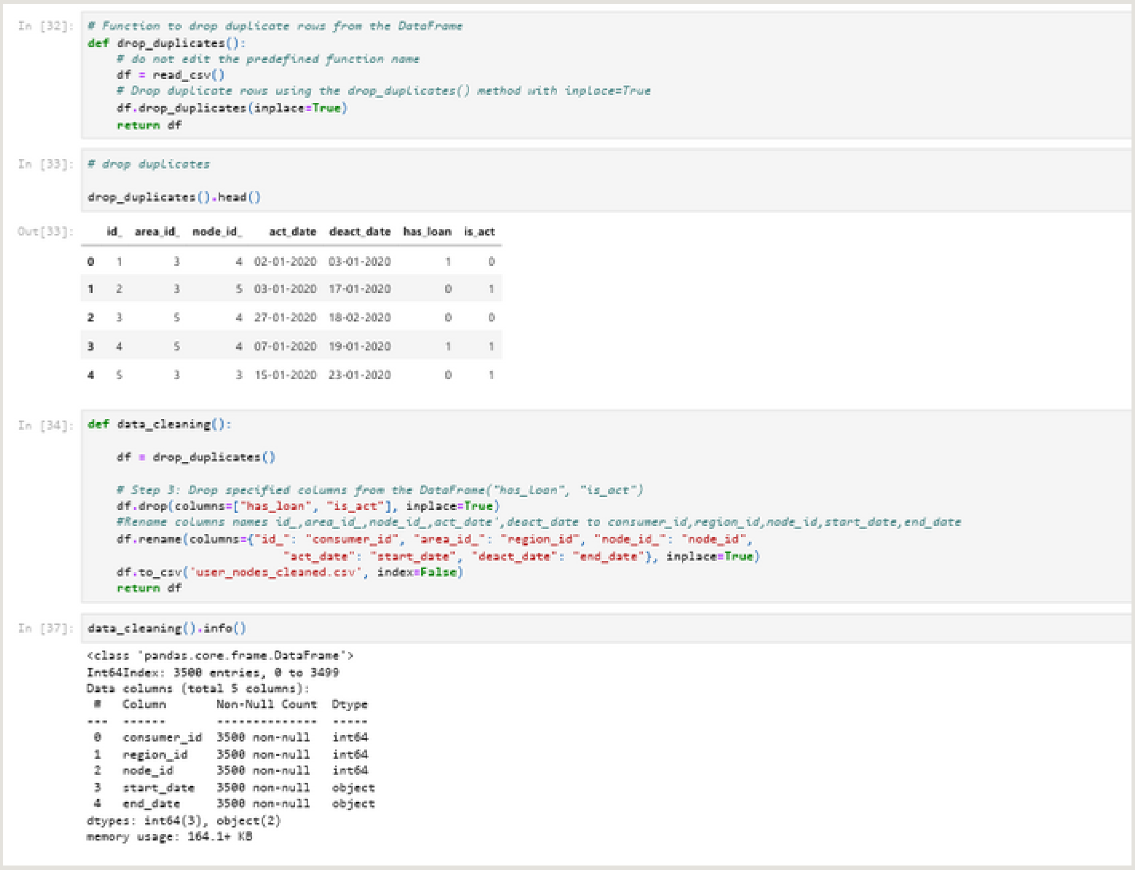
3. user_transaction: This table is a repository of customer transactions, containing data such as consumer IDs, transaction dates, types of transactions, and transaction amounts. Analyzing this data allows us to uncover patterns in customer spending and financial behaviors

# Analyzing Banking Trends: Customer Transactions and Regional Impact



```
In [26]: import pandas as pd
         import warnings
         warnings.filterwarnings("ignore")

In [27]: # Function to read the CSV file into a DataFrame
         def read_csv():
             # read the user_nodes.csv file using pandas library and return it
             df = pd.read_csv('user_nodes.csv')
             return df

In [28]: # Function to check for null (missing) values in the DataFrame
         def check_null_values():
             # do not edit the predefined function name
             df = read_csv()
             # Check for null values using the isnull() method and sum them for each column
             null_values = df.isnull().sum()
             return null_values

In [29]: # check for null values

         check_null_values()

Out[29]: id_           0
         area_id_      0
         node_id_      0
         act_date      0
         deact_date    0
         has_loan      0
         is_act        0
         dtype: int64

In [30]: # Function to check for duplicate rows in the DataFrame
         def check_duplicates():
             # do not edit the predefined function name
             df = read_csv()
             # Calculate the number of duplicate rows using the duplicated() method and sum them
             duplicates = df.duplicated().sum()
             return duplicates

In [31]: # check for duplicates

         check_duplicates()

Out[31]: 143
```

Each data source was cleaned, checked for null and duplicate values .



```
In [32]: # Function to drop duplicate rows from the DataFrame
         def drop_duplicates():
             # do not edit the predefined function name
             df = read_csv()
             # Drop duplicate rows using the drop_duplicates() method with inplace=True
             df.drop_duplicates(inplace=True)
             return df

In [33]: # drop duplicates

         drop_duplicates().head()

Out[33]:
```

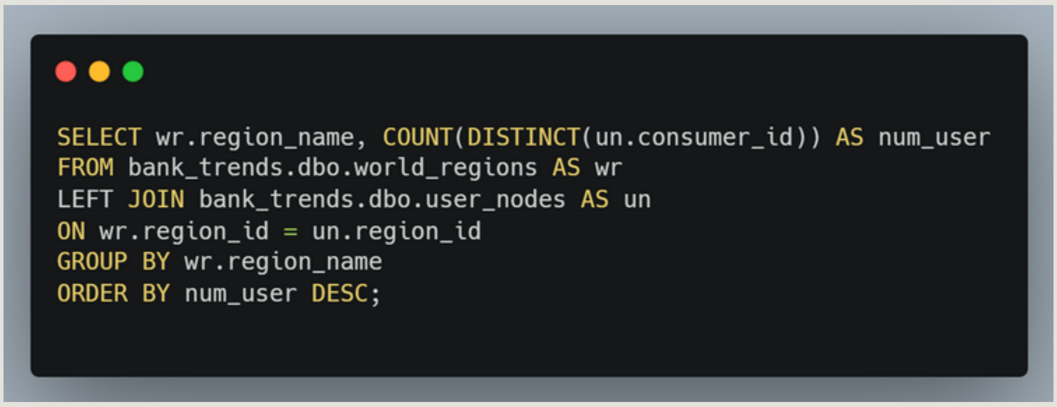
|   | id_ | area_id_ | node_id_ | act_date | deact_date | has_loan | is_act |
|---|---|---|---|---|---|---|---|
| **0** | 1 | 3 | 4 | 02-01-2020 | 03-01-2020 | 1 | 0 |
| **1** | 2 | 3 | 5 | 03-01-2020 | 17-01-2020 | 0 | 1 |
| **2** | 3 | 5 | 4 | 27-01-2020 | 18-02-2020 | 0 | 0 |
| **3** | 4 | 5 | 4 | 07-01-2020 | 19-01-2020 | 1 | 1 |
| **4** | 5 | 3 | 3 | 15-01-2020 | 23-01-2020 | 0 | 1 |

```
In [34]: def data_cleaning():

             df = drop_duplicates()

             # Step 3: Drop specified columns from the DataFrame("has_loan", "is_act")
             df.drop(columns=["has_loan", "is_act"], inplace=True)
             #Rename columns names id_,area_id_,node_id_,act_date',deact_date to consumer_id,region_id,node_id,start_date,end_date
             df.rename(columns={"id_": "consumer_id", "area_id_": "region_id", "node_id_": "node_id",
                                "act_date": "start_date", "deact_date": "end_date"}, inplace=True)
             df.to_csv('user_nodes_cleaned.csv', index=False)
             return df

In [37]: data_cleaning().info()

         <class 'pandas.core.frame.DataFrame'>
         Int64Index: 3500 entries, 0 to 3499
         Data columns (total 5 columns):
          #   Column       Non-Null Count  Dtype
         ---  ------       --------------  -----
          0   consumer_id  3500 non-null   int64
          1   region_id    3500 non-null   int64
          2   node_id      3500 non-null   int64
          3   start_date   3500 non-null   object
          4   end_date     3500 non-null   object
         dtypes: int64(3), object(2)
         memory usage: 164.1+ KB
```

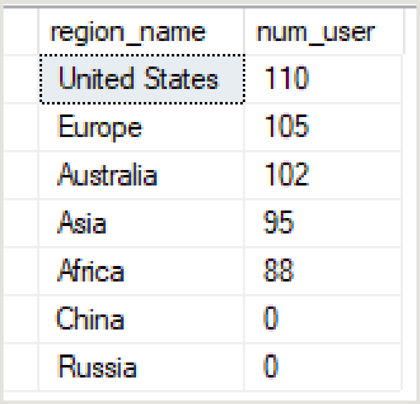
Duplicates and null values were then dropped and files were exported to a csv.

# Analyzing Banking Trends: Customer Transactions and Regional Impact

## List all regions along with the number of users assigned to each region.

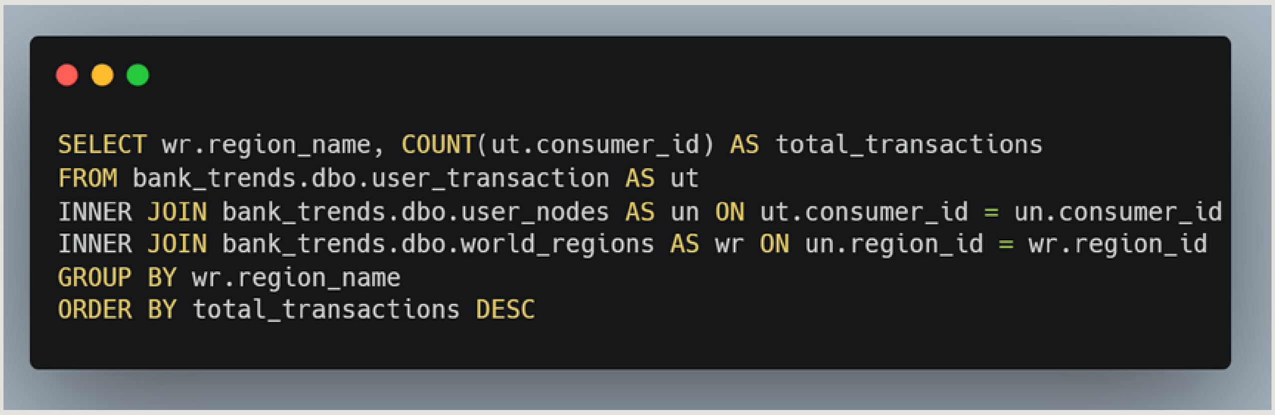


```sql
SELECT wr.region_name, COUNT(DISTINCT(un.consumer_id)) AS num_user
FROM bank_trends.dbo.world_regions AS wr
LEFT JOIN bank_trends.dbo.user_nodes AS un
ON wr.region_id = un.region_id
GROUP BY wr.region_name
ORDER BY num_user DESC;
```



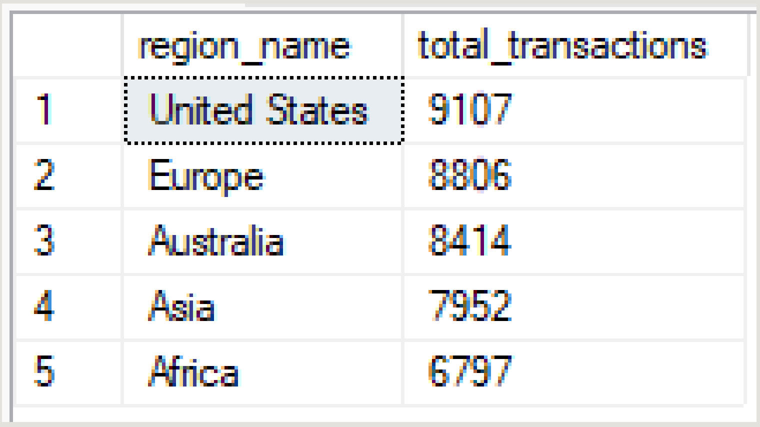
| region_name | num_user |
|---|---|
| United States | 110 |
| Europe | 105 |
| Australia | 102 |
| Asia | 95 |
| Africa | 88 |
| China | 0 |
| Russia | 0 |

## Retrieve the total number of transactions for each region.



```sql
SELECT wr.region_name, COUNT(ut.consumer_id) AS total_transactions
FROM bank_trends.dbo.user_transaction AS ut
INNER JOIN bank_trends.dbo.user_nodes AS un ON ut.consumer_id = un.consumer_id
INNER JOIN bank_trends.dbo.world_regions AS wr ON un.region_id = wr.region_id
GROUP BY wr.region_name
ORDER BY total_transactions DESC
```

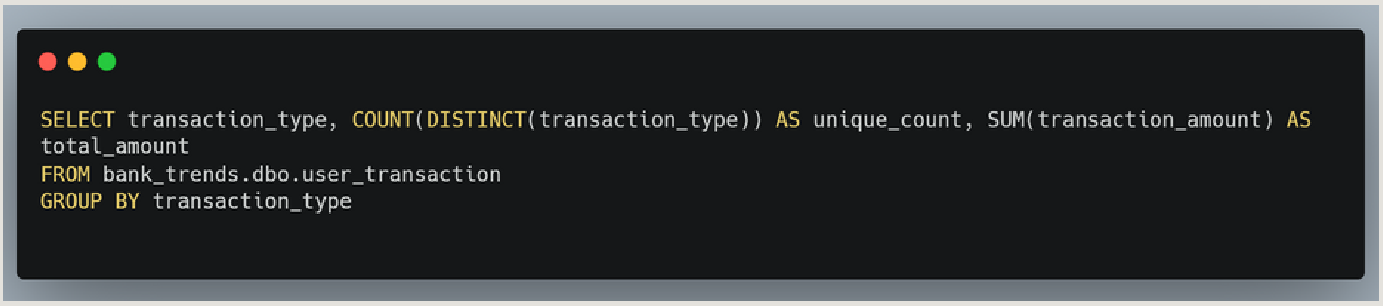
*This code shows the total number of transactions for each region. United States has the most transactions and Africa with the least amount of transactions.*



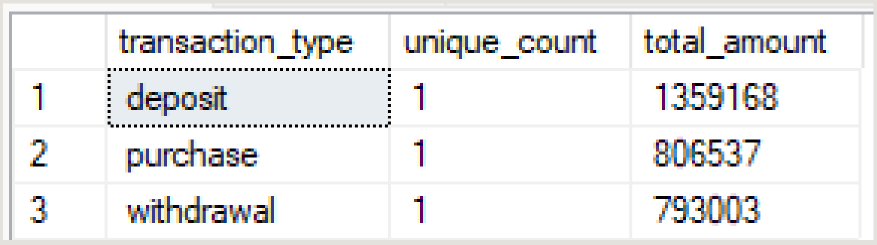
| | region_name | total_transactions |
|---|---|---|
| 1 | United States | 9107 |
| 2 | Europe | 8806 |
| 3 | Australia | 8414 |
| 4 | Asia | 7952 |
| 5 | Africa | 6797 |

# Analyzing Banking Trends: Customer Transactions and Regional Impact

## What is the unique count and total amount for each transaction type?



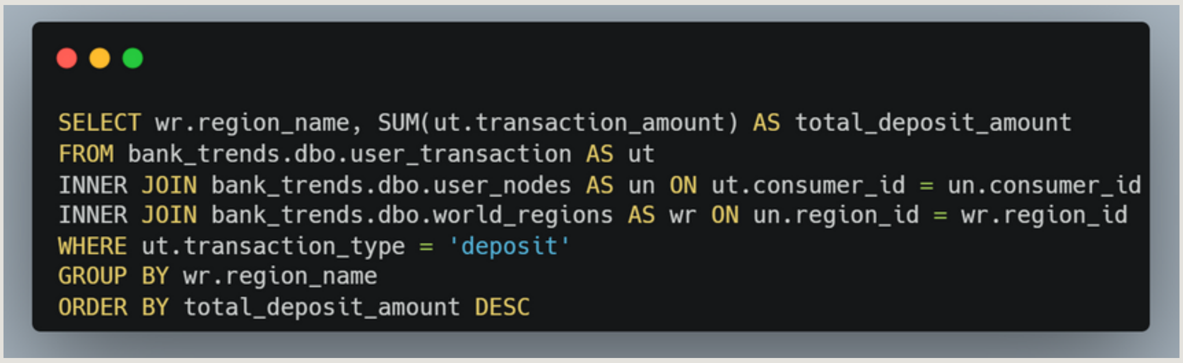
```
SELECT transaction_type, COUNT(DISTINCT(transaction_type)) AS unique_count, SUM(transaction_amount) AS
total_amount
FROM bank_trends.dbo.user_transaction
GROUP BY transaction_type
```



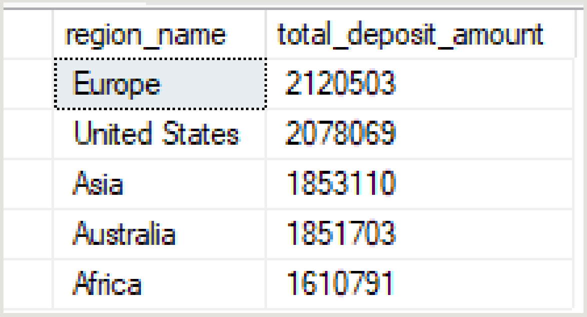
| | transaction_type | unique_count | total_amount |
|---|---|---|---|
| 1 | deposit | 1 | 1359168 |
| 2 | purchase | 1 | 806537 |
| 3 | withdrawal | 1 | 793003 |

*This query shows the total amount of deposits, purchases, and withdrawals made.*

## Write a query to find the total deposit amount for each region in the user_transaction table.



```
SELECT wr.region_name, SUM(ut.transaction_amount) AS total_deposit_amount
FROM bank_trends.dbo.user_transaction AS ut
INNER JOIN bank_trends.dbo.user_nodes AS un ON ut.consumer_id = un.consumer_id
INNER JOIN bank_trends.dbo.world_regions AS wr ON un.region_id = wr.region_id
WHERE ut.transaction_type = 'deposit'
GROUP BY wr.region_name
ORDER BY total_deposit_amount DESC
```

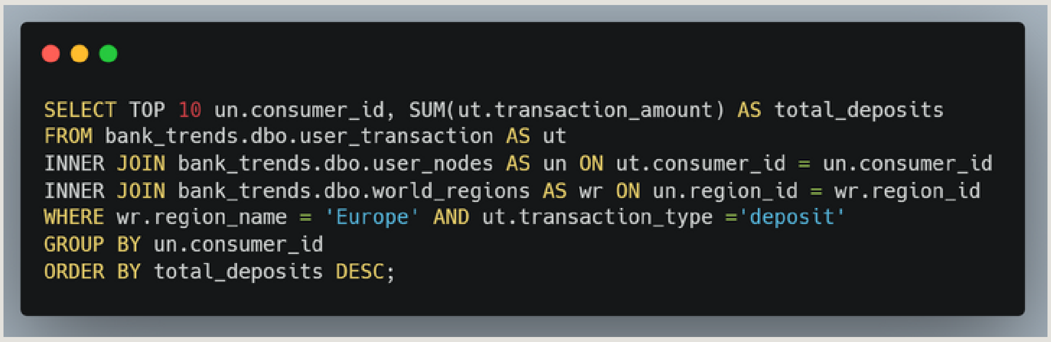


| region_name | total_deposit_amount |
|---|---|
| Europe | 2120503 |
| United States | 2078069 |
| Asia | 1853110 |
| Australia | 1851703 |
| Africa | 1610791 |

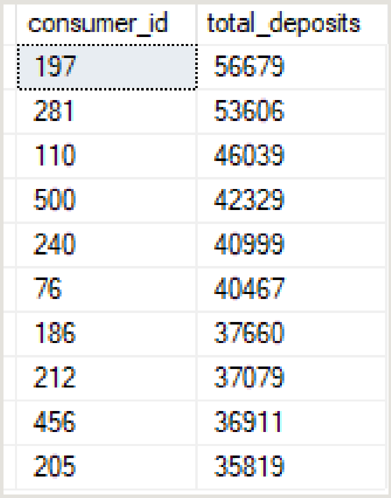
*This query shows customers in Europe have deposited the most amount of funds.*

# Analyzing Banking Trends: Customer Transactions and Regional Impact

## Calculate the total amount deposited for the top 10 users in the "Europe" region.



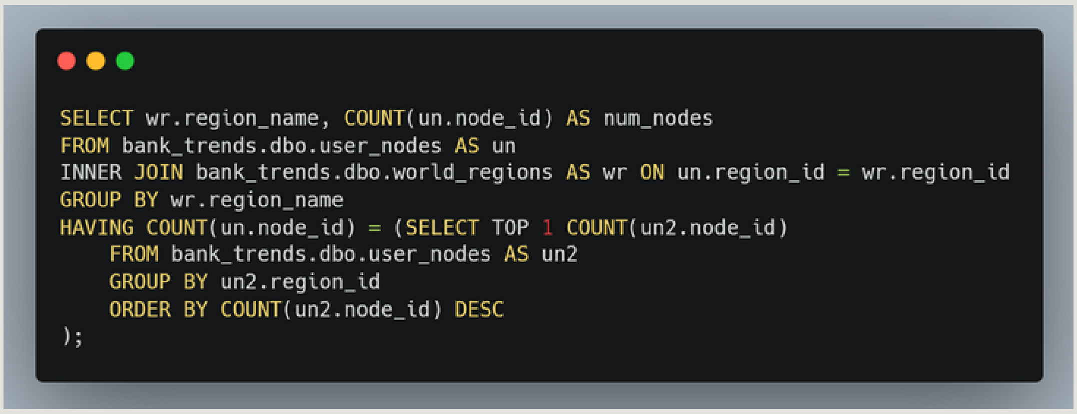
```sql
SELECT TOP 10 un.consumer_id, SUM(ut.transaction_amount) AS total_deposits
FROM bank_trends.dbo.user_transaction AS ut
INNER JOIN bank_trends.dbo.user_nodes AS un ON ut.consumer_id = un.consumer_id
INNER JOIN bank_trends.dbo.world_regions AS wr ON un.region_id = wr.region_id
WHERE wr.region_name = 'Europe' AND ut.transaction_type ='deposit'
GROUP BY un.consumer_id
ORDER BY total_deposits DESC;
```



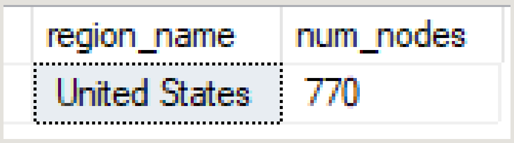
| consumer_id | total_deposits |
|---|---|
| 197 | 56679 |
| 281 | 53606 |
| 110 | 46039 |
| 500 | 42329 |
| 240 | 40999 |
| 76 | 40467 |
| 186 | 37660 |
| 212 | 37079 |
| 456 | 36911 |
| 205 | 35819 |

*Top 10 consumers and the total amount deposited*

## Find the regions with the highest number of nodes assigned to them.



```sql
SELECT wr.region_name, COUNT(un.node_id) AS num_nodes
FROM bank_trends.dbo.user_nodes AS un
INNER JOIN bank_trends.dbo.world_regions AS wr ON un.region_id = wr.region_id
GROUP BY wr.region_name
HAVING COUNT(un.node_id) = (SELECT TOP 1 COUNT(un2.node_id)
    FROM bank_trends.dbo.user_nodes AS un2
    GROUP BY un2.region_id
    ORDER BY COUNT(un2.node_id) DESC
);
```

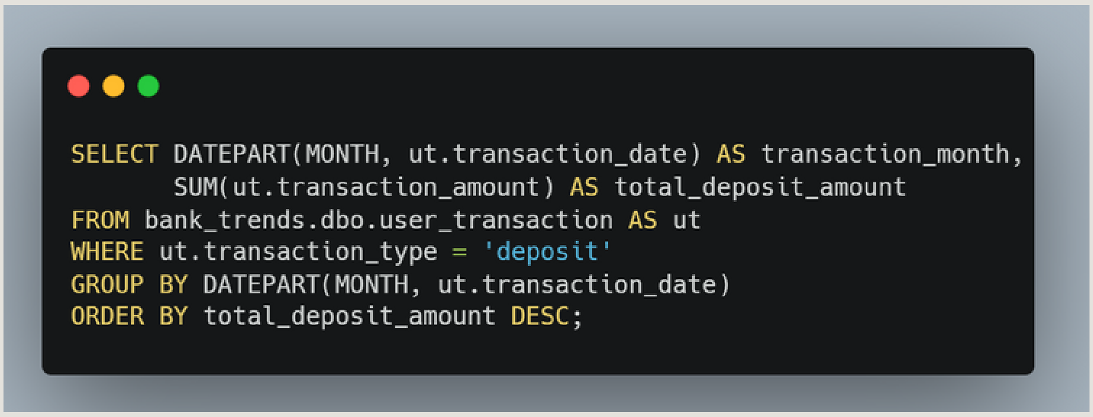


| region_name | num_nodes |
|---|---|
| United States | 770 |

*The United States has the highest number of nodes assigned.*

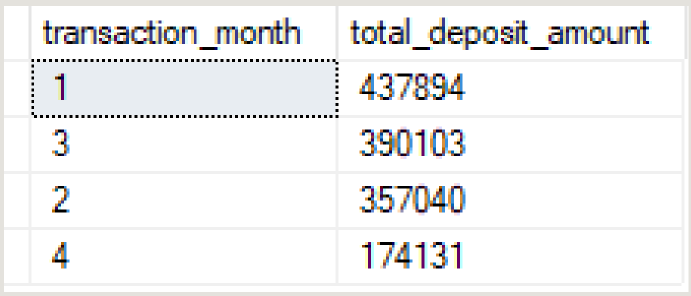
# Analyzing Banking Trends: Customer Transactions and Regional Impact

## Which month had the highest transaction amount for deposits?



```
SELECT DATEPART(MONTH, ut.transaction_date) AS transaction_month,
       SUM(ut.transaction_amount) AS total_deposit_amount
FROM bank_trends.dbo.user_transaction AS ut
WHERE ut.transaction_type = 'deposit'
GROUP BY DATEPART(MONTH, ut.transaction_date)
ORDER BY total_deposit_amount DESC;
```

*We saw the most deposits in January*



| transaction_month | total_deposit_amount |
|---|---|
| 1 | 437894 |
| 3 | 390103 |
| 2 | 357040 |
| 4 | 174131 |

## KEY TAKEAWAYS

- Most prominent transaction type is deposit across all regions.
- United States and Europe stand out with the most substantial deposit volume.
- January is the most popular month for deposits, possibly due to the holiday season.
- There are currently no users assigned in China and Russia,  potential areas for growth.