Czechoslovakia-Bank

I obtained the raw dataset from the provided sources, ensuring all necessary relationships between the datasets. The dataset comprises 8 tables, including account, client, disposition, permanent order, transaction, loan, credit card, and district. I converted the date column to datetime type. Then, I merged and integrated the various datasets to create a unified dataset suitable for comprehensive analysis. Additionally, I utilized bar graphs to visualize some aspects of the data.

Problem Statement:

The Czechoslovakia Bank aims to leverage its financial data to gain deeper insights and facilitate informed decision-making. With a focus on identifying trends, patterns, and potential risks within its operations, the bank intends to explore opportunities for introducing new financial products or services. To achieve these objectives, the bank has outlined the following key questions for analysis:

Analyze the demographic profile of clients and assess its variation across districts.

Identify the age and gender distribution of customers.

Examine the usage frequency and profitability of various credit card types to refine services.

Identify major expenses of the bank.

Analyze variations in the loan portfolio across purposes and client segments to inform risk management and lending strategies.

Enhance customer service and satisfaction to retain existing customers and attract new ones.

Explore opportunities to introduce new financial products or services to diversify offerings, attract a wider customer base, and increase profitability.

Calculate the bank’s total loans over the years, including year and month-wise breakdowns.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Import the 8 data into the python file

```
account = pd.read_csv(r"C:\Users\Dell\OneDrive\Desktop\account.csv")
account.head()
```

	account_id	district_id	frequency	date
0	576	55	POPLATEK MESICNE	1993-01-01
1	3818	74	POPLATEK MESICNE	1993-01-01
2	704	55	POPLATEK MESICNE	1993-01-01
3	2378	16	POPLATEK MESICNE	1993-01-01
4	2632	24	POPLATEK MESICNE	1993-01-02

```
card = pd.read_csv(r"C:\Users\Dell\OneDrive\Desktop\card.csv")
card.head()
```

	card_id	disp_id	type	issued
0	1005	9285	classic	1993-11-07 12:00:00
1	104	588	classic	1994-01-19 12:00:00
2	747	4915	classic	1994-02-05 12:00:00
3	70	439	classic	1994-02-08 12:00:00
4	577	3687	classic	1994-02-15 12:00:00

```
client = pd.read_csv(r"C:\Users\Dell\OneDrive\Desktop\client.csv")
client.head()
```

	Unnamed: 0	client_id	birth_number	district_id	gender	age	age_levels
0	1	1	701213	18	FEMALE	29	ADULT
1	2	2	450204	1	MALE	54	MIDDLE AGED
2	3	3	401009	1	FEMALE	59	MIDDLE AGED
3	4	4	561201	5	MALE	43	MIDDLE AGED
4	5	5	600703	5	FEMALE	39	MIDDLE AGED

```
disp = pd.read_csv(r"C:\Users\Dell\OneDrive\Desktop\disp.csv")
disp.head()
```

	disp_id	client_id	account_id	type
0	1	1	1	OWNER
1	2	2	2	OWNER
2	3	3	2	DISPONENT
3	4	4	3	OWNER
4	5	5	3	DISPONENT

```
district = pd.read_csv(r"C:\Users\Dell\Downloads\district.csv")
district.head()
```

	district_id	A2	Region	A4	A5	A6	A7	A8	A9	A10	Average_salary	A12	A13	A14	A15	A16
0	1	Hl.m. Praha	Prague	1204953	0	0	0	1	1	100.0	12541	0.29	0.43	167	85677	99107
1	2	Benesov	central Bohemia	88884	80	26	6	2	5	46.7	8507	1.67	1.85	132	2159	2674
2	3	Beroun	central Bohemia	75232	55	26	4	1	5	41.7	8980	1.95	2.21	111	2824	2813
3	4	Kladno	central Bohemia	149893	63	29	6	2	6	67.4	9753	4.64	5.05	109	5244	5892
4	5	Kolin	central Bohemia	95616	65	30	4	1	6	51.4	9307	3.85	4.43	118	2616	3040

```
loan = pd.read_csv(r"C:\Users\Dell\OneDrive\Desktop\loan.csv")
loan.head()
```

	loan_id	account_id	date	amount	duration	payments	status
0	5314	1787	1993-07-05 00:00:00	96396	12	8033	B
1	5316	1801	1993-07-11 00:00:00	165960	36	4610	A
2	6863	9188	1993-07-28 00:00:00	127080	60	2118	A
3	5325	1843	1993-08-03 00:00:00	105804	36	2939	A
4	7240	11013	1993-09-06 00:00:00	274740	60	4579	A

```
orde = pd.read_csv(r"C:\Users\Dell\OneDrive\Desktop\orde.csv")
orde.head()
```

	order_id	account_id	bank_to	account_to	amount	k_symbol
0	29401	1	YZ	87144583	2452.0	SIPO
1	29402	2	ST	89597016	3372.7	UVER
2	29403	2	QR	13943797	7266.0	SIPO
3	29404	3	WX	83084338	1135.0	SIPO
4	29406	3	AB	59972357	3539.0	POJISTNE

```
trans = pd.read_csv(r"C:\Users\Dell\OneDrive\Desktop\transac.csv")
trans.head()
```

	trans_id	account_id	date	type	operation	amount	balance	k_symbol	bank	account
0	637742	2177	05-01-1993 00:00	PRIJEM	PREVOD Z UCTU	5123	5923	DUCHOD	YZ	62457513
1	579374	1972	07-01-1993 00:00	PRIJEM	PREVOD Z UCTU	5298	5698	DUCHOD	UV	14132887
2	1049882	3592	10-01-1993 00:00	PRIJEM	PREVOD Z UCTU	6007	6607	DUCHOD	MN	73166322
3	171813	576	11-01-1993 00:00	PRIJEM	PREVOD Z UCTU	6207	7107	DUCHOD	YZ	30300313
4	689828	2357	12-01-1993 00:00	PRIJEM	PREVOD Z UCTU	6434	7234	DUCHOD	OP	34144538

Demographics

```
demographics = client.groupby('district_id').agg({
    'age': 'mean',  # Calculate mean of 'age'
    'client_id': 'count',  # Count the number of clients
    'gender': lambda x: x.value_counts().idxmax()  # Find the most
common gender
}).reset_index()

# Rename columns
demographics.columns = ['district_id', 'avg_age', 'client_count',
'most_common_gender']

# Print the resulting DataFrame

print(demographics.head(10))
```

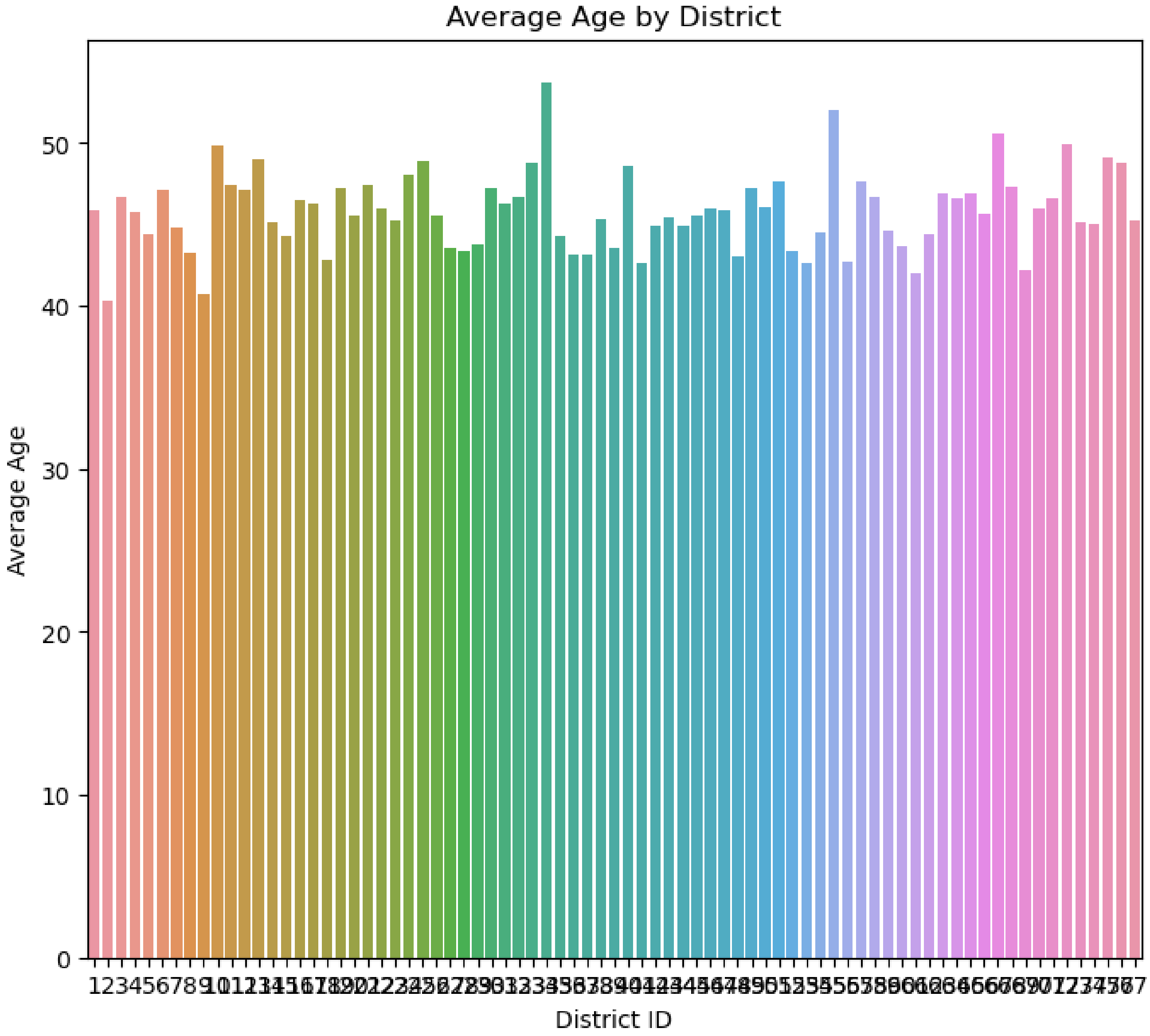
Output:

```
   district_id    avg_age  client_count most_common_gender
0            1  45.855204           663               MALE
1            2  40.369565            46             FEMALE
2            3  46.746032            63               MALE
3            4  45.780000            50               MALE
4            5  44.366197            71             FEMALE
5            6  47.132075            53             FEMALE
6            7  44.822222            45               MALE
7            8  43.275362            69               MALE
8            9  40.750000            60             FEMALE
9           10  49.888889            54             FEMALE
```

```
# Plot average age across districts

plt.figure(figsize=(8, 7))
sns.barplot(data=demographics, x='district_id', y='avg_age')
plt.title('Average Age by District')
plt.xlabel('District ID')
plt.ylabel('Average Age')
plt.savefig('Average age.png',format = 'png')
plt.show()
```

Average Age by District

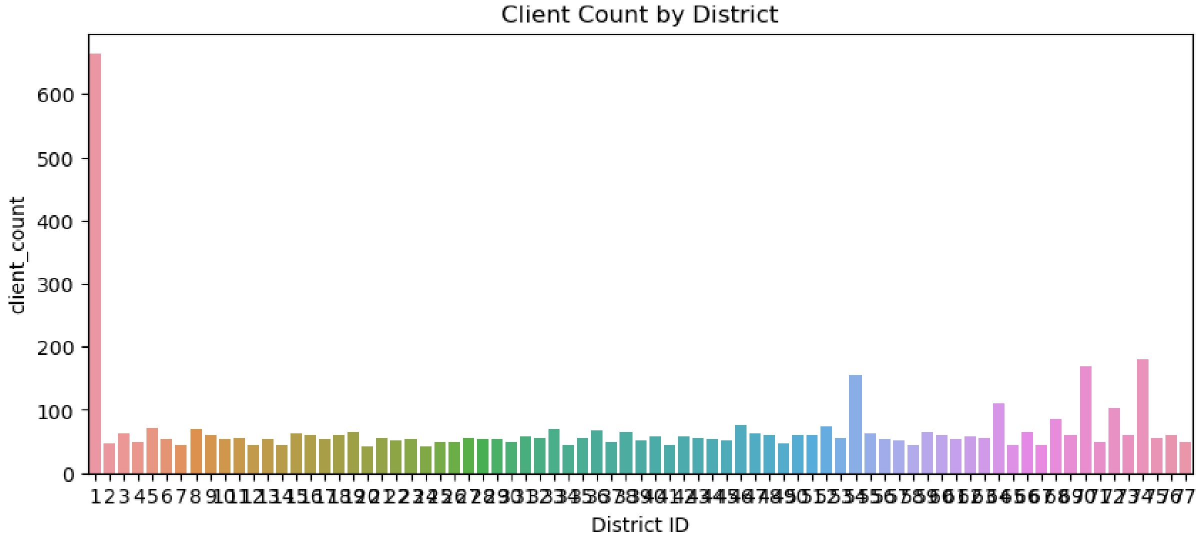


```
# Plot client count across districts

plt.figure(figsize=(10,4))
sns.barplot(data=demographics, x='district_id', y='client_count')
plt.title('Client Count by District')
plt.xlabel('District ID')

plt.savefig(' Client Count.png',format = 'png')
plt.show()
```

Client count by district



Porformance of Bank by year & month-wise.

```
# Join account and loan (data) on account_id

merge = pd.merge(account, loan, on='account_id', how='inner')
merge.head(10)
```

Out[22]:

	account_id	district_id	frequency	date_x	loan_id	date_y	amount	duration	payments	status
0	5270	44	POPLATEK MESICNE	1993-01-13	6077	1993-11-22 00:00:00	79608	24	3317	A
1	11265	15	POPLATEK MESICNE	1993-01-14	7284	1993-09-15 00:00:00	52788	12	4399	A
2	10364	55	POPLATEK MESICNE	1993-01-17	7121	1993-11-10 00:00:00	21924	36	609	A
3	3834	54	POPLATEK MESICNE	1993-01-19	5754	1994-09-28 00:00:00	23052	12	1921	A
4	9307	68	POPLATEK MESICNE	1993-01-24	6895	1994-09-19 00:00:00	41904	12	3492	A
5	5891	54	POPLATEK MESICNE	1993-01-25	6202	1994-04-18	65184	12	5432	A

	account_id	district_id	frequency	date_x	loan_id	date_y	amount	duration	payments	status
						00:00:00				
6	6473	12	POPLATEK MESICNE	1993-01-26	6316	1994-05-26 00:00:00	76908	12	6409	B
7	1843	12	POPLATEK MESICNE	1993-01-30	5325	1993-08-03 00:00:00	105804	36	2939	A
8	9265	1	POPLATEK MESICNE	1993-02-04	6888	1994-07-11 00:00:00	39576	12	3298	A
9	8051	1	POPLATEK TYDNE	1993-02-07	6647	1994-06-01 00:00:00	208320	48	4340	A

```
# Extract year and month from the 'date' column

loan['year'] = loan['date'].dt.year
loan['month'] = loan['date'].dt.month
loan_amounts = loan.groupby(['year', 'month'])['amount'].sum()
loan_amounts.head()
```

Output:

```
year  month
1993  7        389436
      8        105804
      9        590112
      10       154416
      11       218556
```

Gender population in percentage

```
# Count the number of men and women
gender_counts = client['gender'].value_counts()

# Calculate the percentage of men and women in the total population
total_clients = len(client)
percentage_male = (gender_counts['MALE'] / total_clients) * 100
percentage_female = (gender_counts['FEMALE'] / total_clients) * 100
```

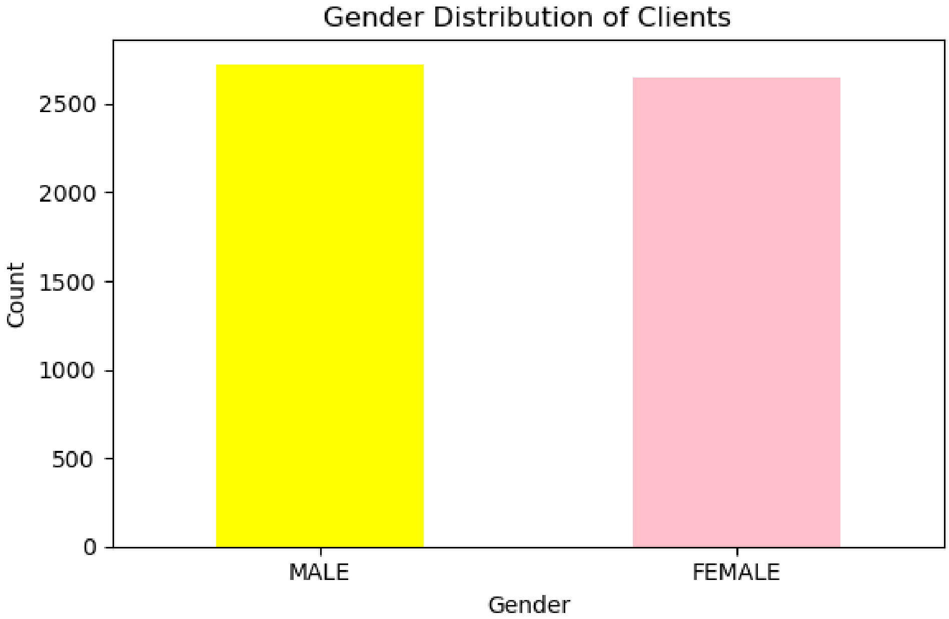
```
print("Percentage of Male Clients:", percentage_male)
print("Percentage of Female Clients:", percentage_female)
```

Output:

```
Percentage of Male Clients: 50.73570497299311
Percentage of Female Clients: 49.264295027006895
```

```
# Visualize the gender distribution

plt.figure(figsize=(6, 4))
gender_counts.plot(kind='bar', color=['yellow', 'pink'])
plt.title('Gender Distribution of Clients')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



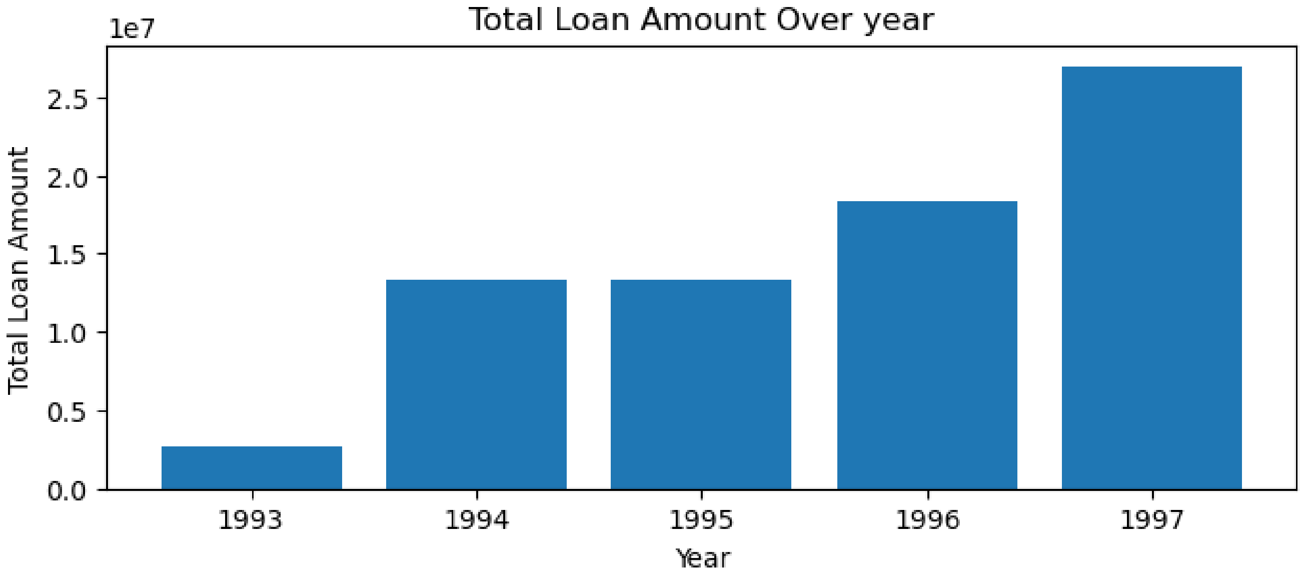
Year Wise

```
# Extract year from the 'date' column
loan['year'] = loan['date'].dt.year

# Group the data by year and calculate total loan amount for each year
yearly_loan_amounts = loan.groupby('year')['amount'].sum()

# Reset index to convert the groupby result to a DataFrame
yearly_loan_amounts = yearly_loan_amounts.reset_index()
```

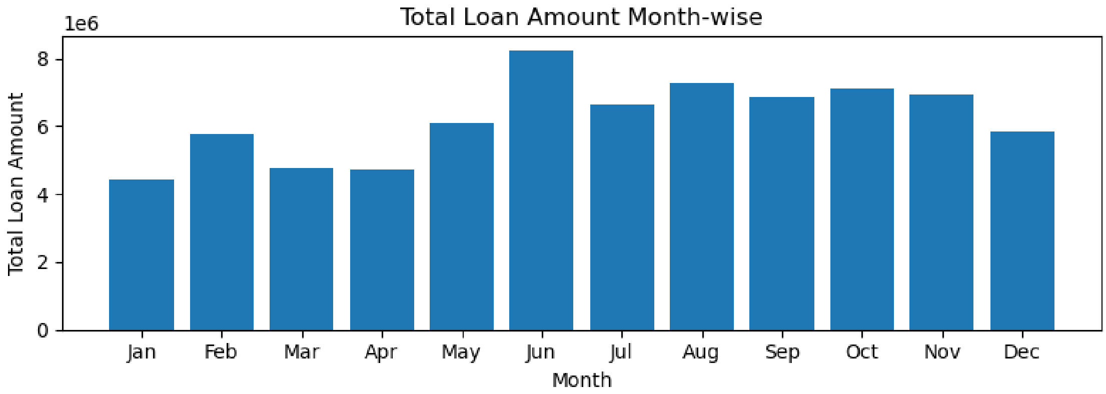
```
# Plot the loan amounts over time
plt.figure(figsize=(8, 3))
plt.bar(yearly_loan_amounts['year'], yearly_loan_amounts['amount'])
plt.xlabel('Year')
plt.ylabel('Total Loan Amount')
plt.title('Total Loan Amount Over year')
plt.savefig('Over year.png', format = 'png')
plt.show()
```



Month Wise

```
# Extract month from the 'date' column
loan['month'] = loan['date'].dt.month

# Group the data by month and calculate total loan amount for each month
monthly_loan_amounts = loan.groupby('month')['amount'].sum()
# Plot the loan amounts month-wise
plt.figure(figsize=(8, 3))
plt.bar(monthly_loan_amounts.index, monthly_loan_amounts.values)
plt.xlabel('Month')
plt.ylabel('Total Loan Amount')
plt.title('Total Loan Amount Month-wise')
plt.xticks(range(1, 13), ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul',
'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])
plt.tight_layout()
plt.show()
```



Common types of accounts and its different usage and profitability

```
# Count occurrences of each account type
account_type_counts = account['frequency'].value_counts()

# Display the most common account types
print("Most common types of accounts:")
print(account_type_counts)
Most common types of accounts:
frequency
POPLATEK MESICNE      925
POPLATEK TYDNE         57
POPLATEK PO OBRATU     18


# Calculate usage metrics
total_transactions = trans['amount'].sum()
average_transaction_amount = trans['amount'].mean()

print('total_transactions:',  total_transactions)
print('average_transaction_amount:', average_transaction_amount)
total_transactions: 194098
average_transaction_amount: 5545.657142857143

# Profitablity
total_loan_amount = loan['amount'].sum()
average_loan_amount = loan['amount'].mean()


print('total_loan_amount:', total_loan_amount)
print('average_loan_amount:', average_loan_amount)
```

Output:

```
total_loan_amount: 74620872
average_loan_amount: 149241.744
```

Type of cards used frequently

```
card_types = card['type'].unique()

card_types
array(['classic', 'junior', 'gold'], dtype=object)
card_usage_counts = card['type'].value_counts()

print(card_usage_counts)
```

Output:

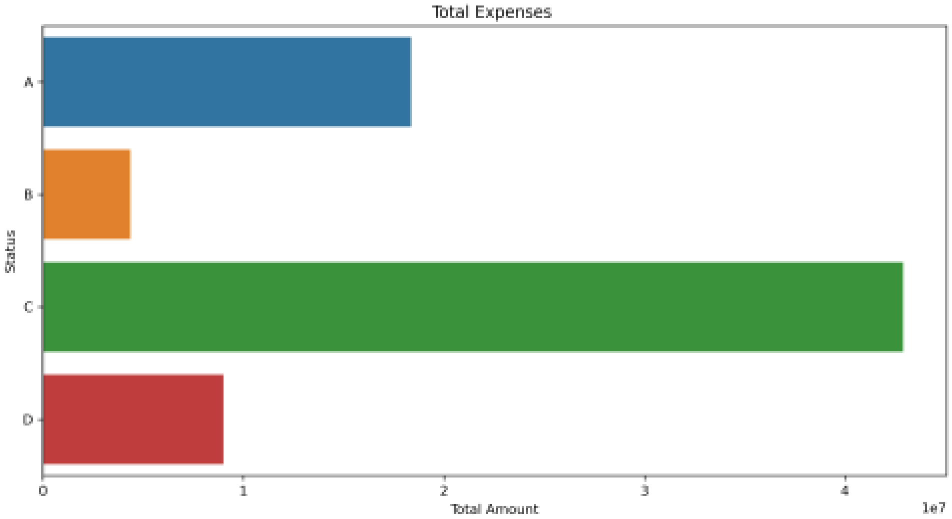
```
type
classic    659
junior     145
gold        88
```

Classic type of card is used most frequently

Major expenses of the bank

After a detailed analysis of the bank's expenses, including loan expenses and order expenses by merging the required columns in the data, we summed up all the expenses to create the total expenses.

Total expenses by the Status



The bank improves its customer service and satisfaction levels by:

By gathering comprehensive customer data including demographics, transaction history, and preferences.

Utilizing the advanced analytics to segment customers into distinct groups based on their financial behaviors and needs.

Developing the targeted marketing campaigns and promotions tailored to specific customer segments.

The bank introduce new financial products or services to attract more customers and increase profitability.

By analyzing loan performance, segmenting clients, and understanding transaction patterns, we can identify potential opportunities for the bank to introduce new financial products or services. This analysis will help the bank attract more customers and increase profitability.