Area Of Project : Customer Churn Prediction /Customer Churn dataset

---

Dataset Overview

The dataset contains 10,000 rows and 14 columns. The columns represent customer-related information from a banking system. Below is a summary of the columns:

- **RowNumber**: Serial number of the row.
- **CustomerId**: Unique identifier for a customer.
- **Surname**: Customer's surname.
- **CreditScore**: Credit score of the customer.
- **Geography**: Country of residence.
- **Gender**: Customer's gender.
- **Age**: Age of the customer.
- **Tenure**: Number of years the customer has been with the bank.
- **Balance**: Bank account balance of the customer.
- **NumOfProducts**: Number of products the customer uses.
- **HasCrCard**: Whether the customer owns a credit card (1 for yes, 0 for no).
- **IsActiveMember**: Whether the customer is an active member (1 for yes, 0 for no).
- **EstimatedSalary**: Estimated salary of the customer.
- **Exited**: Whether the customer left the bank (1 for yes, 0 for no).

Plan for Analysis

1. **Entities, Attributes, and Relationships**:
   - Identify logical groupings for entities such as Customer, Account, and other related components.
2. **ER Diagram**:
   - Design an Entity-Relationship diagram to map the identified entities and relationships.
3. **Database Design**:
   - Create MySQL table creation scripts.

I will first analyze the dataset for potential entities and relationships.

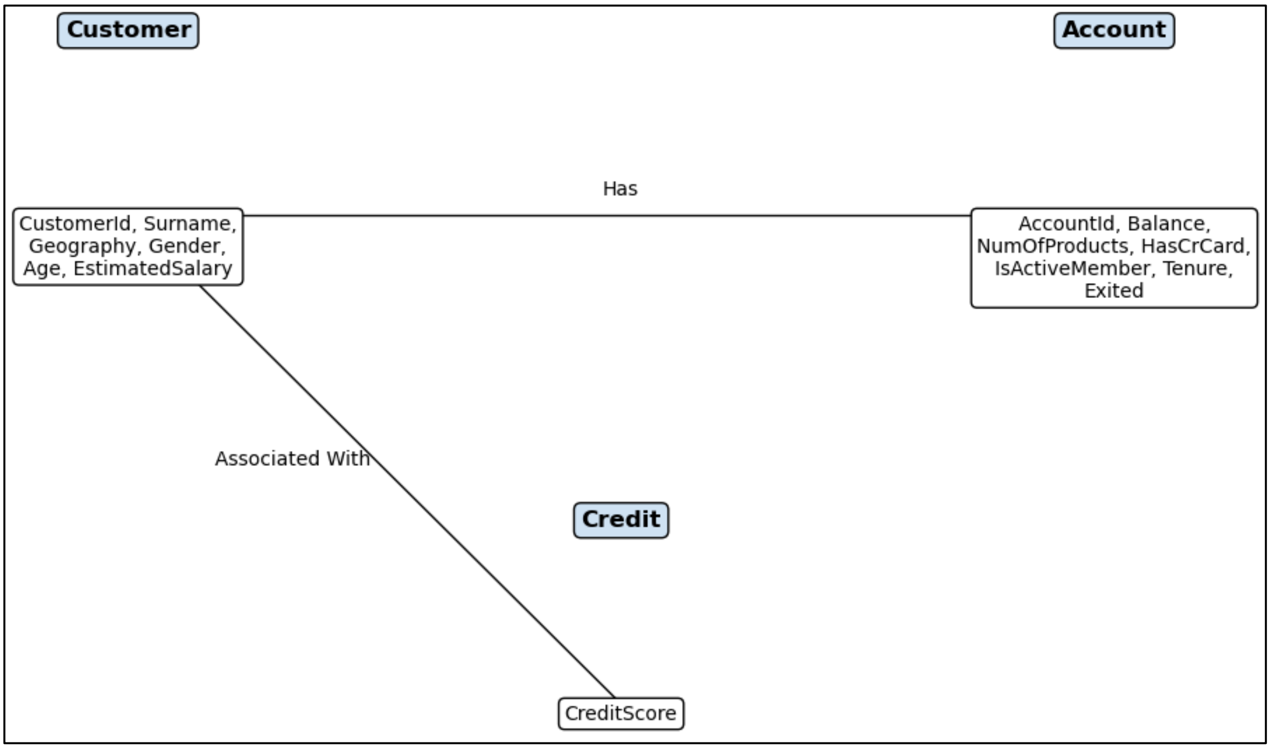
Identified Entities and Attributes

1. **Customer**
   - Attributes: CustomerId, Surname, Geography, Gender, Age, EstimatedSalary
   - Relationship: Owns an account and uses banking products.
2. **Account**
   - Attributes: AccountId (unique ID), Balance, NumOfProducts, HasCrCard, IsActiveMember, Tenure, Exited
   - Relationship: Linked to a customer.
3. **Credit**
   - Attributes: CreditScore
   - Relationship: Linked to a customer and potentially influences account behavior.

Relationships

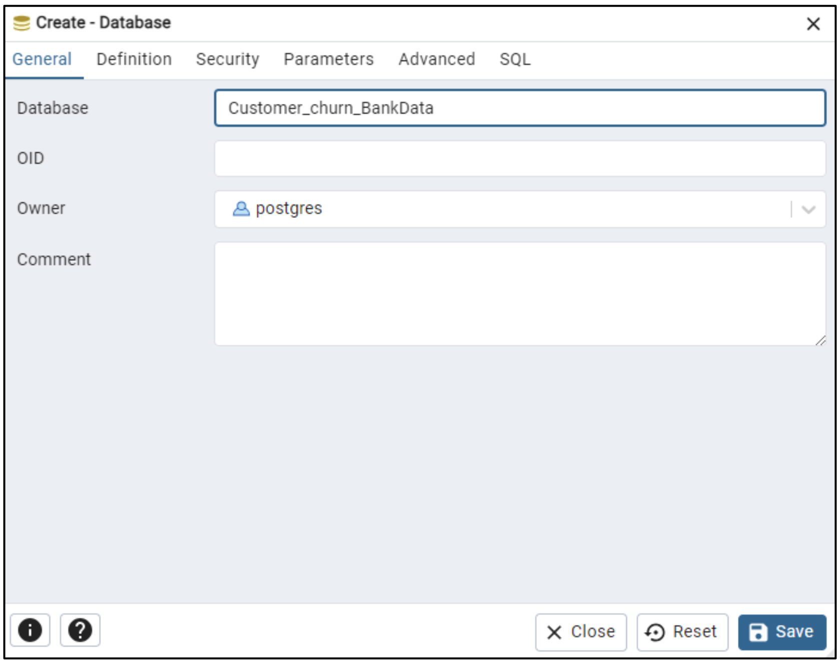
- A **Customer** has one **Account**.
- An **Account** has attributes like balance and product usage.
- **Credit** is associated with a **Customer**, reflecting their financial history.

ER Diagram –

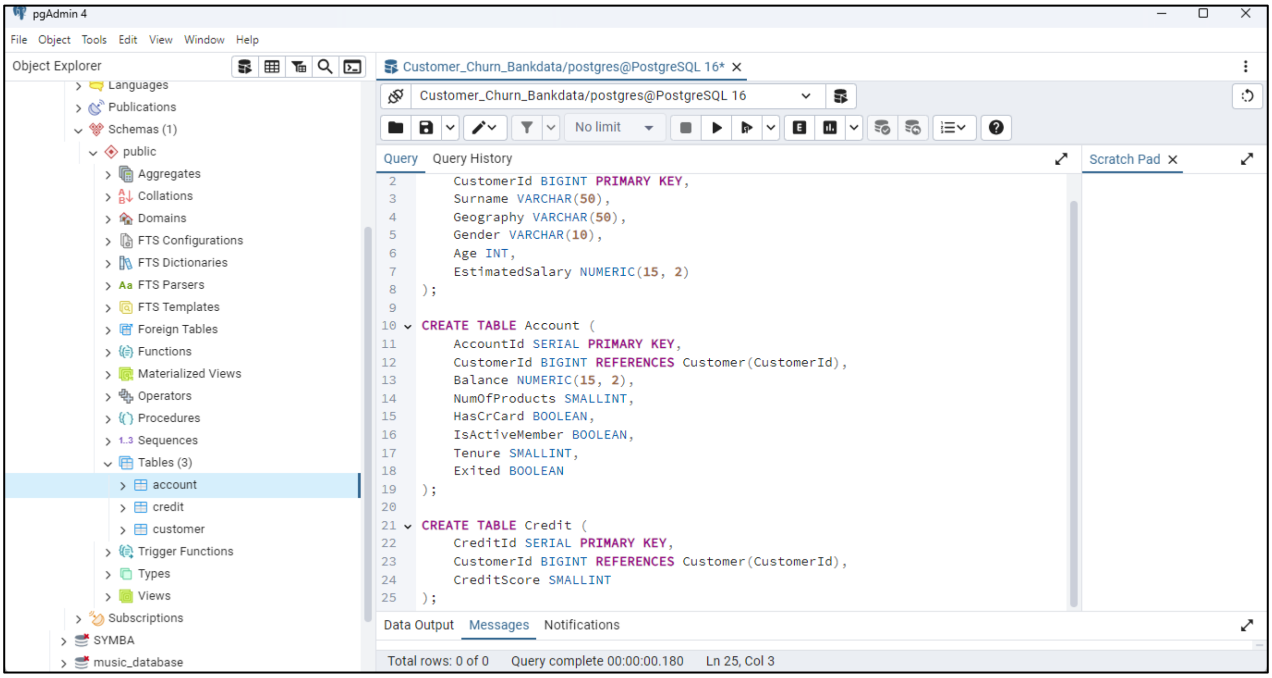


Screen Shots of Tables Created in MySQL (PostgreSQL)

Creating Database :



Query Of Creating Tables :



Tables Screenshots :

- Tables (3)
  - account
  - credit
  - customer

Table – Account

```sql
Select * from account
```

Data Output Messages Notifications

	customerid bigint	balance numeric (15,2)	numofproducts smallint	hascrcard boolean	isactivemember boolean	tenure smallint	exited boolean
1	15634602	0.00	1	true	true	2	true
2	15647311	83807.86	1	false	true	1	false
3	15619304	159660.80	3	true	false	8	true
4	15701354	0.00	2	false	false	1	false
5	15737888	125510.82	1	true	true	2	false
6	15574012	113755.78	2	true	false	8	true
7	15592531	0.00	2	true	true	7	false
8	15656148	115046.74	4	true	false	4	true
9	15792365	142051.07	2	false	true	4	false
10	15592389	134603.88	1	true	true	2	false
11	15767821	102016.72	2	false	false	6	false
12	15737173	0.00	2	true	false	3	false

Table – Customer

```
Select * from customer
```

Data Output Messages Notifications

	customerid [PK] bigint	surname character varying (50)	geography character varying (50)	gender character varying (10)	age integer	estimatedsalary numeric (15,2)
1	15634602	Hargrave	France	Female	42	101348.88
2	15647311	Hill	Spain	Female	41	112542.58
3	15619304	Onio	France	Female	42	113931.57
4	15701354	Boni	France	Female	39	93826.63
5	15737888	Mitchell	Spain	Female	43	79084.10
6	15574012	Chu	Spain	Male	44	149756.71
7	15592531	Bartlett	France	Male	50	10062.80
8	15656148	Obinna	Germany	Female	29	119346.88
9	15792365	He	France	Male	44	74940.50
10	15592389	H?	France	Male	27	71725.73
11	15767821	Bearce	France	Male	31	80181.12
12	15737173	Andrews	Spain	Male	24	76390.01
13	15632264	Kay	France	Female	34	26260.98

Table – Credit

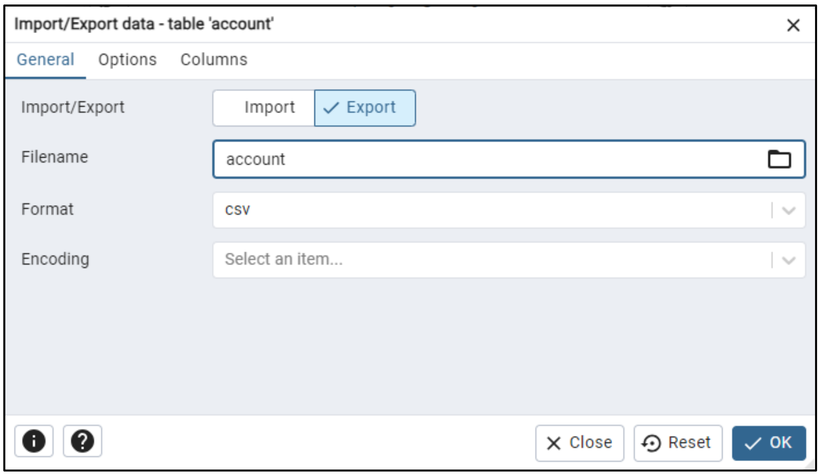
```
Select * from credit
```

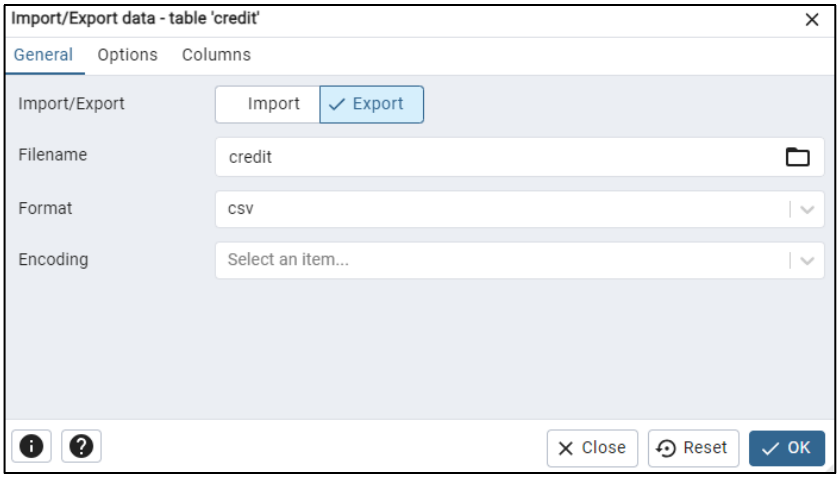
Data Output Messages Notifications

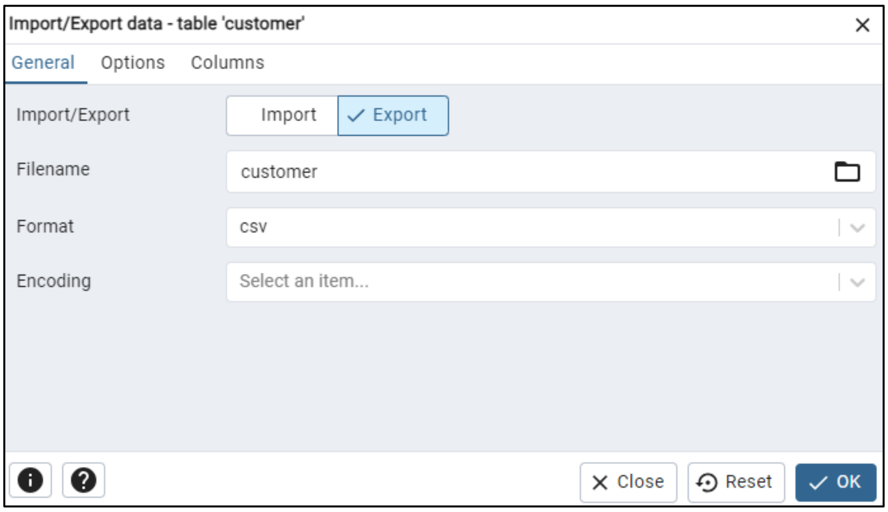
	customerid bigint	creditscore integer
1	15634602	619
2	15647311	608
3	15619304	502
4	15701354	699
5	15737888	850
6	15574012	645
7	15592531	822
8	15656148	376
9	15792365	501
10	15592389	684

Exporting Data in Excel/Csv format :

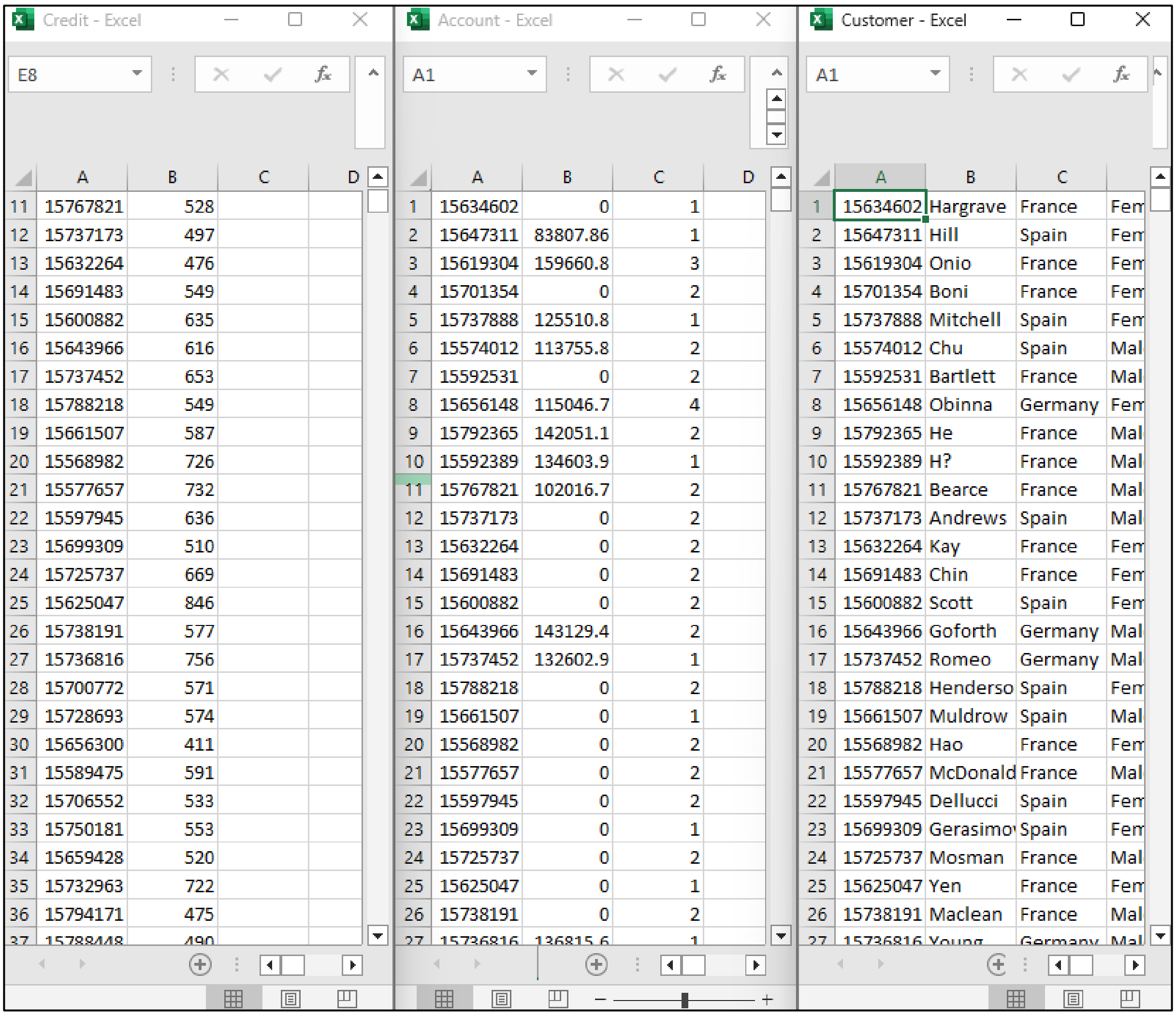
Exporting Tables –







Exported Excel Sheets Screenshot-



Credit - Excel

	A	B	C	D
11	15767821	528		
12	15737173	497		
13	15632264	476		
14	15691483	549		
15	15600882	635		
16	15643966	616		
17	15737452	653		
18	15788218	549		
19	15661507	587		
20	15568982	726		
21	15577657	732		
22	15597945	636		
23	15699309	510		
24	15725737	669		
25	15625047	846		
26	15738191	577		
27	15736816	756		
28	15700772	571		
29	15728693	574		
30	15656300	411		
31	15589475	591		
32	15706552	533		
33	15750181	553		
34	15659428	520		
35	15732963	722		
36	15794171	475		
37	15788448	490		

Account - Excel

	A	B	C	D
1	15634602	0	1	
2	15647311	83807.86	1	
3	15619304	159660.8	3	
4	15701354	0	2	
5	15737888	125510.8	1	
6	15574012	113755.8	2	
7	15592531	0	2	
8	15656148	115046.7	4	
9	15792365	142051.1	2	
10	15592389	134603.9	1	
11	15767821	102016.7	2	
12	15737173	0	2	
13	15632264	0	2	
14	15691483	0	2	
15	15600882	0	2	
16	15643966	143129.4	2	
17	15737452	132602.9	1	
18	15788218	0	2	
19	15661507	0	1	
20	15568982	0	2	
21	15577657	0	2	
22	15597945	0	2	
23	15699309	0	1	
24	15725737	0	2	
25	15625047	0	1	
26	15738191	0	2	
27	15736816	136815.6	1	

Customer - Excel

	A	B	C	
1	15634602	Hargrave	France	Fem
2	15647311	Hill	Spain	Fem
3	15619304	Onio	France	Fem
4	15701354	Boni	France	Fem
5	15737888	Mitchell	Spain	Fem
6	15574012	Chu	Spain	Mal
7	15592531	Bartlett	France	Mal
8	15656148	Obinna	Germany	Fem
9	15792365	He	France	Mal
10	15592389	H?	France	Mal
11	15767821	Bearce	France	Mal
12	15737173	Andrews	Spain	Mal
13	15632264	Kay	France	Fem
14	15691483	Chin	France	Fem
15	15600882	Scott	Spain	Fem
16	15643966	Goforth	Germany	Mal
17	15737452	Romeo	Germany	Mal
18	15788218	Henderso	Spain	Fem
19	15661507	Muldrow	Spain	Mal
20	15568982	Hao	France	Fem
21	15577657	McDonald	France	Mal
22	15597945	Dellucci	Spain	Fem
23	15699309	Gerasimo	Spain	Fem
24	15725737	Mosman	France	Mal
25	15625047	Yen	France	Fem
26	15738191	Maclean	France	Mal
27	15736816	Young	Germany	Mal

ML Model – Decision Tree

In [6]:
```
import pandas as pd  # For handling and manipulating data in DataFrame format
import numpy as np  # For numerical operations and array manipulations

# Visualization libraries
import matplotlib.pyplot as plt  # For creating static visualizations like plots an
```

In [7]:
```
# Data preparation and splitting
from sklearn.model_selection import train_test_split   # For splitting the dataset ifrom sklearn.preprocessing
import LabelEncoder  # For encoding categorical variablefrom sklearn.preprocessing import StandardScaler  #
For standardizing features by s
```

In [10]:
```
# Classification algorithms
from sklearn.tree import DecisionTreeClassifier  # For decision tree-based classififrom sklearn.neighbors impor
KNeighborsClassifier  # For k-nearest neighbors classfrom sklearn.ensemble import RandomForestClassifier  #
For random forest ensemble-b
```

In [11]:
```
# Model evaluation metrics
from sklearn.metrics import classification_report  # For generating precision, recafrom sklearn.metrics import
confusion_matrix # For creating a confusion matrix tofrom sklearn.metrics import accuracy_score  # For
computing the accuracy of the modfrom sklearn.metrics import RocCurveDisplay  # For plotting the ROC curve
of binary
```

In [13]:
```
df=pd.read_csv("Banking_Customers_Dataset.csv")
```

In [14]:
```
df.head()
```

Out[14]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance
0	1	15634602	Hargrave	619	France	Female	42	2	0.00
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86
2	3	15619304	Onio	502	France	Female	42	8	159660.80
3	4	15701354	Boni	699	France	Female	39	1	0.00
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82

In [15]:
```
df.drop(columns=['RowNumber'],inplace=True)  # drop not useful column
```

In [16]:
```
df.columns # names of columns
```

Out[16]:
```
Index(['CustomerId', 'Surname', 'CreditScore', 'Geography', 'Gender', 'Age',
       'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard', 'IsActiveMember',
       'EstimatedSalary', 'Exited'],
      dtype='object')
```

In [17]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>RangeIndex:
10000 entries, 0 to 9999Data columns (total 13
columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   CustomerId       10000 non-null  int64
 1   Surname          10000 non-null  object
 2   CreditScore      10000 non-null  int64
 3   Geography        10000 non-null  object
 4   Gender           10000 non-null  object
 5   Age              10000 non-null  int64
 6   Tenure           10000 non-null  int64
 7   Balance          10000 non-null  float64
 8   NumOfProducts    10000 non-null  int64
 9   HasCrCard        10000 non-null  int64
 10  IsActiveMember   10000 non-null  int64
 11  EstimatedSalary  10000 non-null  float64
 12  Exited           10000 non-null  int64
dtypes: float64(2), int64(8), object(3)
memory usage: 1015.8+ KB
```

In [18]:
```
df.describe().T #describe numric columns
```

Out[18]:

	count	mean	std	min	25%	50%	
CustomerId	10000.0	1.569094e+07	71936.186123	15565701.00	15628528.25	1.569074e+07	1.5
CreditScore	10000.0	6.505288e+02	96.653299	350.00	584.00	6.520000e+02	7.1
Age	10000.0	3.892180e+01	10.487806	18.00	32.00	3.700000e+01	4.4
Tenure	10000.0	5.012800e+00	2.892174	0.00	3.00	5.000000e+00	7.0
Balance	10000.0	7.648589e+04	62397.405202	0.00	0.00	9.719854e+04	1.2
NumOfProducts	10000.0	1.530200e+00	0.581654	1.00	1.00	1.000000e+00	2.0
HasCrCard	10000.0	7.055000e-01	0.455840	0.00	0.00	1.000000e+00	1.0
IsActiveMember	10000.0	5.151000e-01	0.499797	0.00	0.00	1.000000e+00	1.0
EstimatedSalary	10000.0	1.000902e+05	57510.492818	11.58	51002.11	1.001939e+05	1.4
Exited	10000.0	2.037000e-01	0.402769	0.00	0.00	0.000000e+00	0.0

In [19]:
```
df.describe(include='O').T  # describe  category  columns
```

Out[19]:

	count	unique	top	freq
Surname	10000	2932	Smith	32
Geography	10000	3	France	5014
Gender	10000	2	Male	5457

In [20]:
```
df.isna().sum() #Not null value
```

Out[20]:
```
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
```

In [21]:
```
#Analyis and Visualisation
```

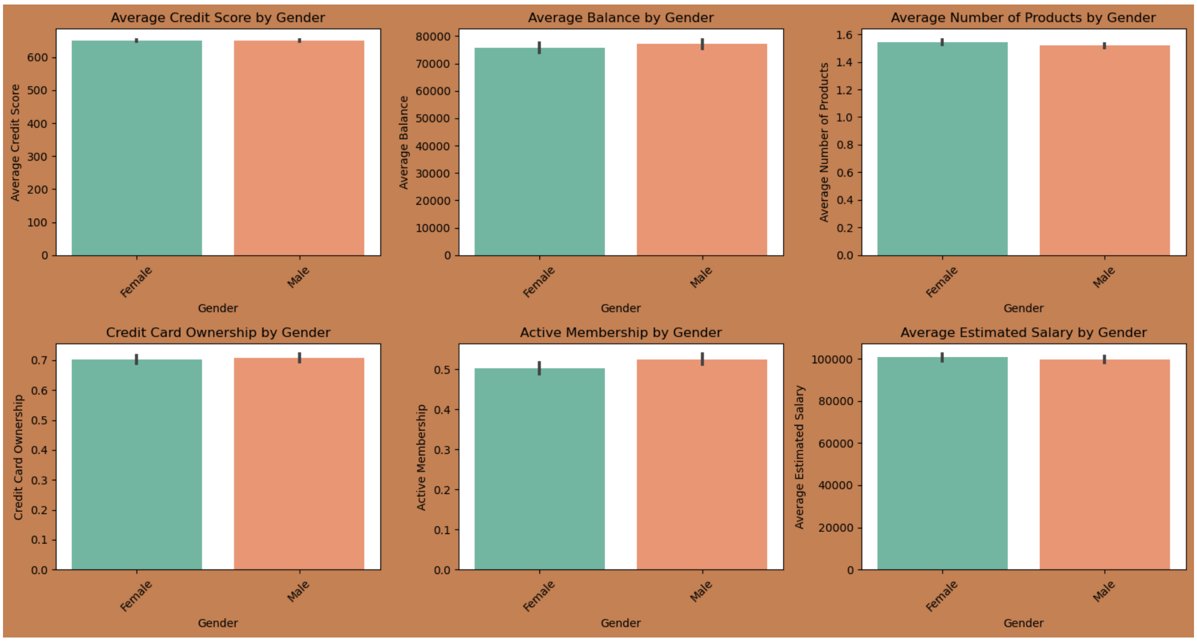
In [22]:
```
df.groupby(['Surname','Geography',])['Balance'].max().nlargest(5)
```

Out[22]:
```
Surname   Geography
Lo        Spain        250898.09
To Rot    France       238387.56
Haddon    Spain        222267.63
McIntosh  Spain        221532.80
Shaw      Spain        216109.88
Name: Balance, dtype: float64
```

In [23]:
```
df.groupby('NumOfProducts')['Balance'].max().nlargest(5) # num of product don't aff
```

Out[23]:
```
NumOfProducts
3    250898.09
1    238387.56
2    214346.96
4    195238.29
Name: Balance, dtype: float64
```

In [25]:
```
plt.figure(figsize=(15, 8), facecolor="#C38154")

features = ['CreditScore', 'Balance', 'NumOfProducts', 'HasCrCard', 'IsActiveMember
titles = ['Average Credit Score by Gender', 'Average Balance by Gender',
          'Average Number of Products by Gender', 'Credit Card Ownership by Gender'
          'Active Membership by Gender', 'Average Estimated Salary by Gender']
for i, feature in enumerate(features, 1):
    plt.subplot(2, 3, i)
    sns.barplot(x='Gender', y=feature, data=df, palette='Set2')
    plt.title(titles[i-1])
    plt.xlabel('Gender')
    plt.ylabel(titles[i-1].split(' by ')[0])
    plt.xticks(rotation=45)

plt.tight_layout()
plt.show()
```



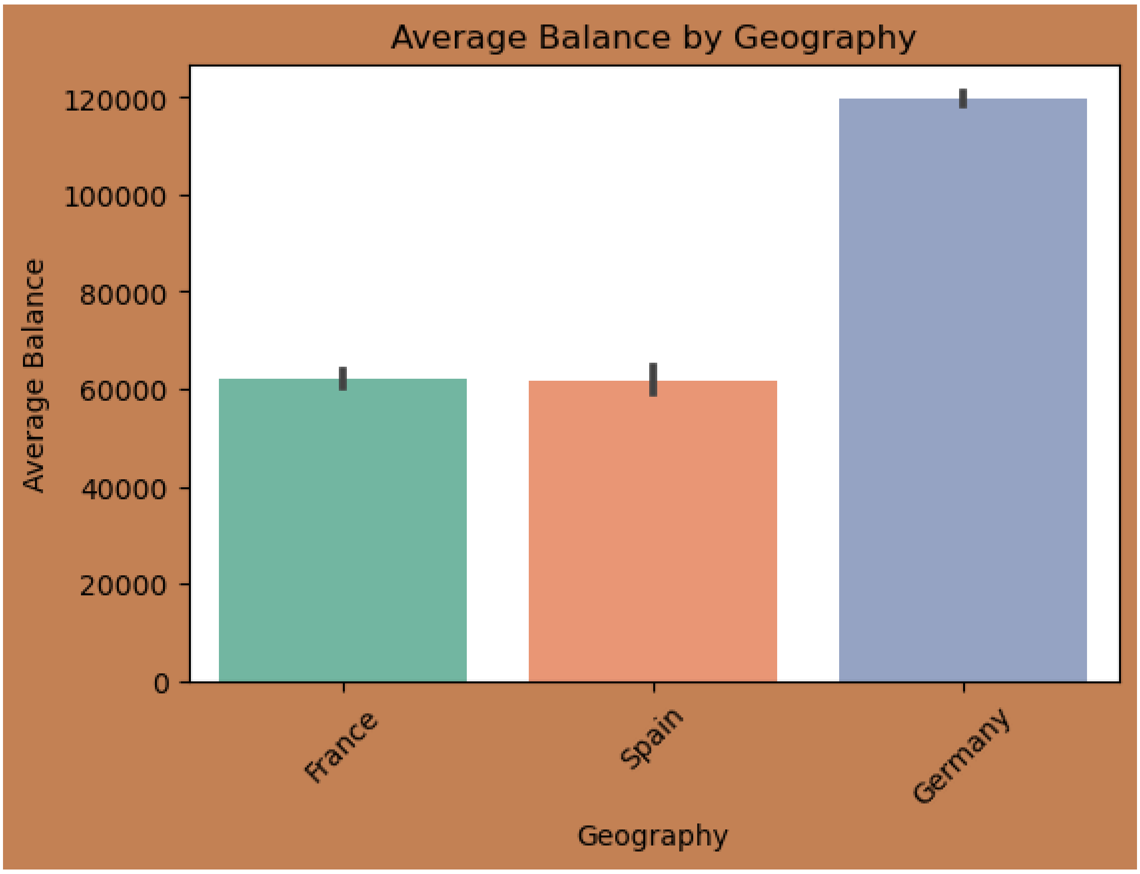
In [26]:
```
df.groupby('Geography')['NumOfProducts'].sum() # sum of product for each country# that is reason for heigh balance
```

Out[26]:
```
Geography
France     7676
Germany    3813
Spain      3813
Name: NumOfProducts, dtype: int64
```

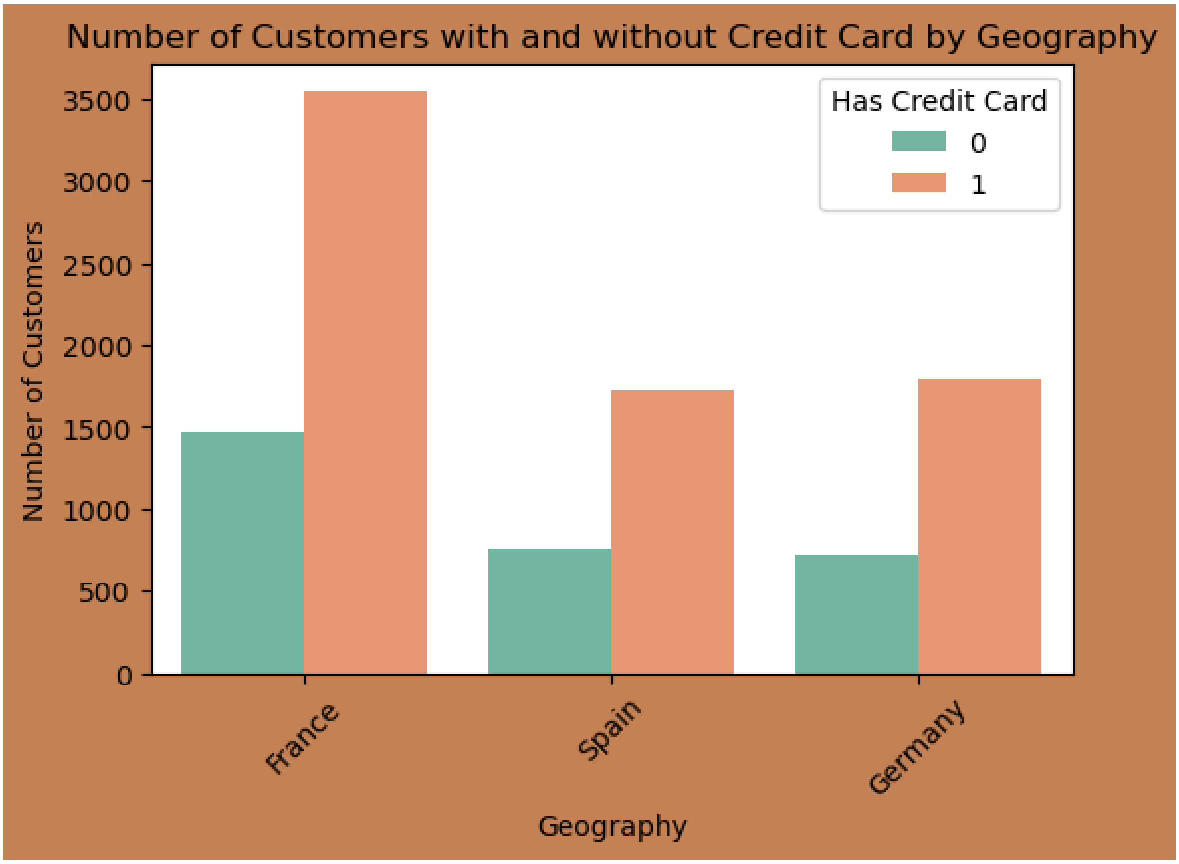
In [27]:
```
df.groupby('Geography')['Balance'].sum() #sum of balance for each country
```

Out[27]:
```
Geography
France     3.113325e+08
Germany    3.004029e+08
Spain  1.531236e+08
Name: Balance, dtype: float64
```

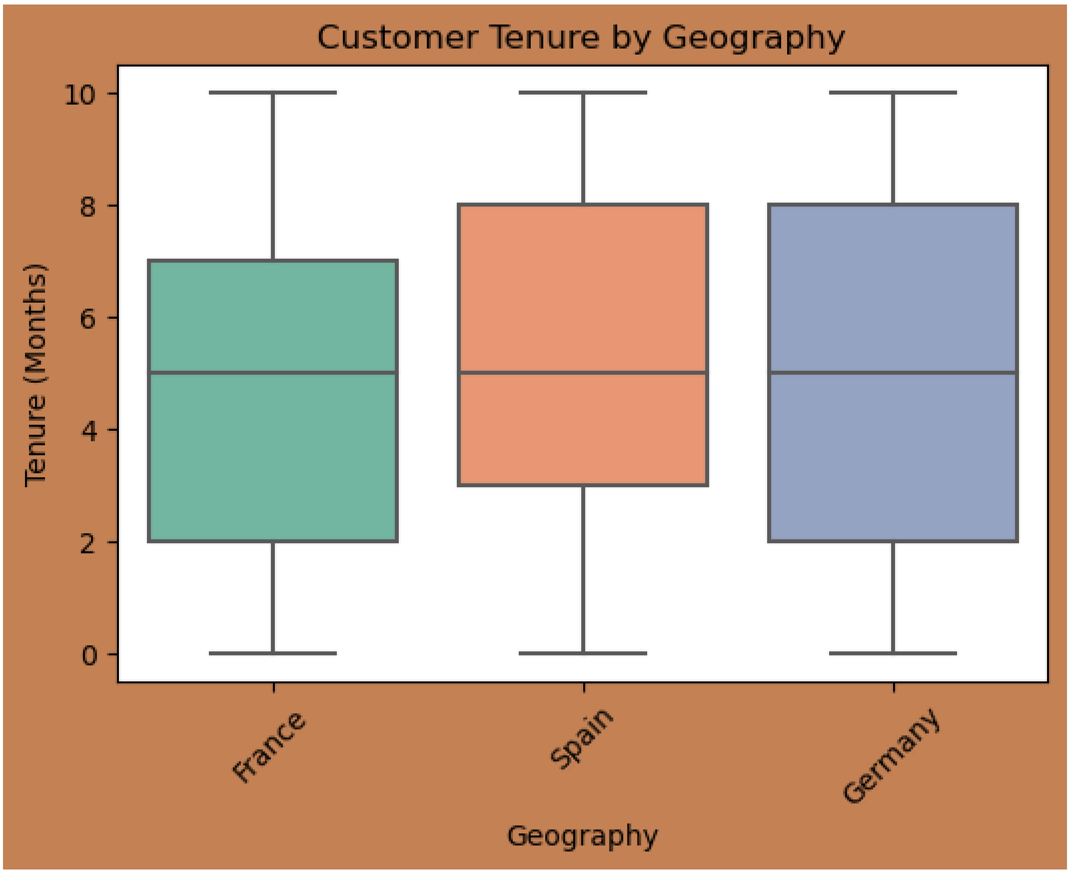
In [28]:
```
plt.figure(figsize=(6, 4), facecolor="#C38154")
sns.barplot(x='Geography', y='Balance', data=df, palette='Set2')
plt.title('Average Balance by Geography')
plt.xlabel('Geography')
plt.ylabel('Average  Balance')
plt.xticks(rotation=45)
plt.show()
```



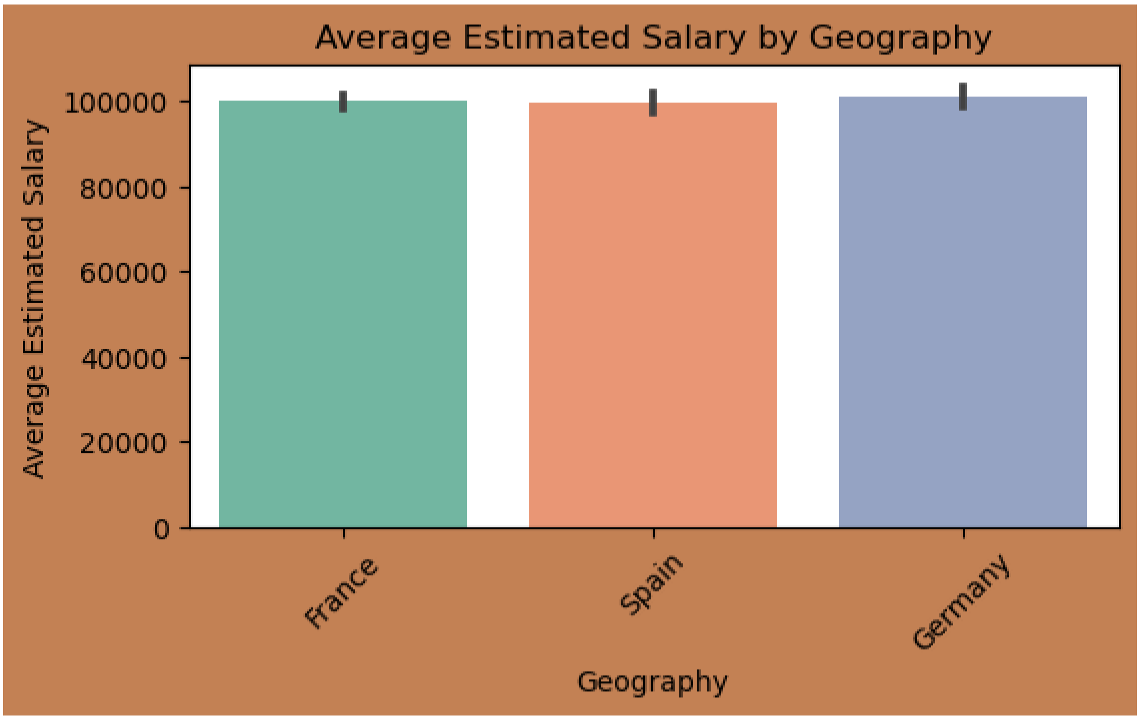
In [29]:
```
plt.figure(figsize=(6, 4), facecolor="#C38154")
sns.countplot(x='Geography', hue='HasCrCard', data=df, palette='Set2')
plt.title('Number of Customers with and without Credit Card by Geography')
plt.xlabel('Geography')
plt.ylabel('Number of Customers')
plt.legend(title='Has Credit Card')
plt.xticks(rotation=45)
plt.show()
```



In [30]:
```
plt.figure(figsize=(6, 4), facecolor="#C38154")
sns.boxplot(x='Geography', y='Tenure', data=df, palette='Set2')
plt.title('Customer Tenure by Geography')
plt.xlabel('Geography')
plt.ylabel('Tenure (Months)')
plt.xticks(rotation=45)
plt.show()
```

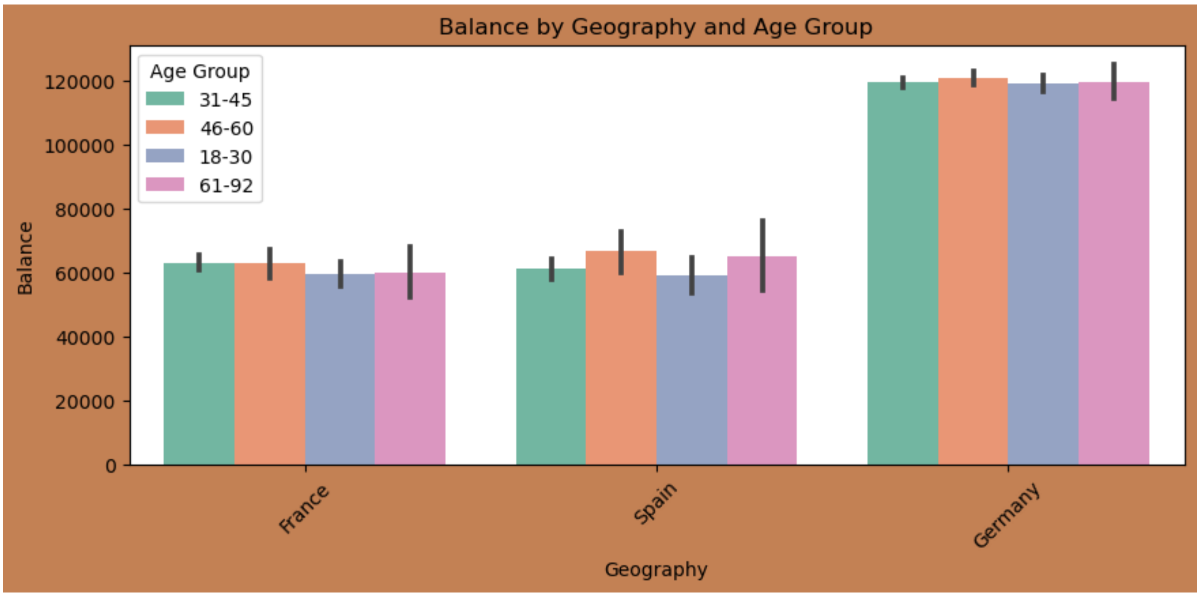


In [31]:
```
plt.figure(figsize=(6, 3), facecolor="#C38154")
sns.barplot(x='Geography', y='EstimatedSalary', data=df, palette='Set2')
plt.title('Average Estimated Salary by Geography')
plt.xlabel('Geography')
plt.ylabel('Average Estimated Salary')
plt.xticks(rotation=45)
plt.show()
```

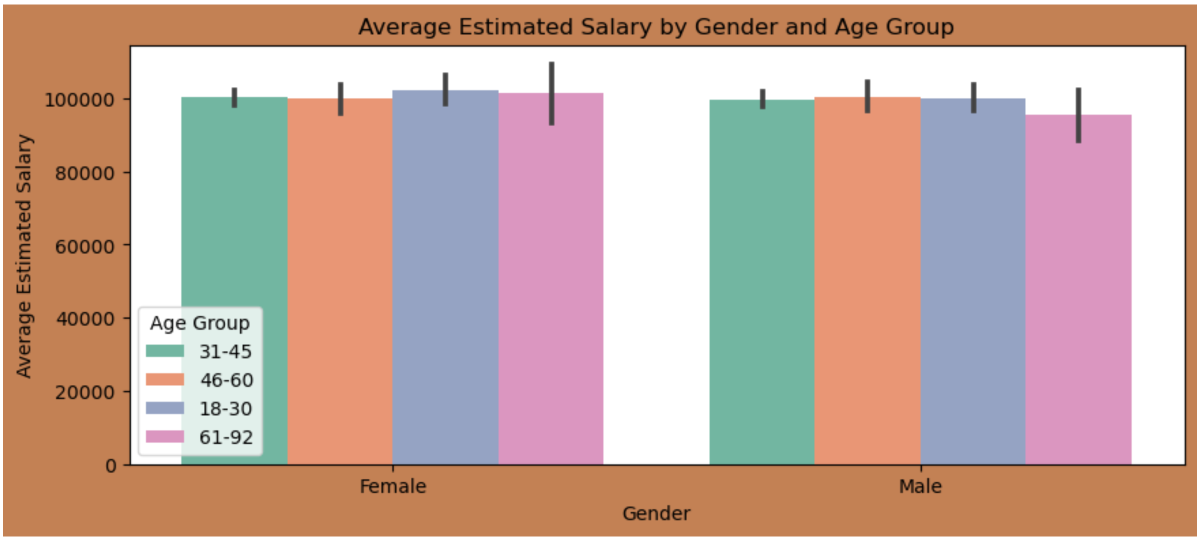


In [32]:
```
def  categorize_age(age):
    if 18 <= age <= 30:

        return '18-30'

    elif 31 <= age <= 45:

        return '31-45'

    elif 46 <= age <= 60:

        return '46-60'

    elif 61 <= age <= 92:
```

In [33]:
```
plt.figure(figsize=(10,  4),  facecolor="#C38154")
sns.barplot(x='Geography',  y='Balance',  hue='AgeGroup',  data=df,  palette='Set2')
plt.title('Balance by Geography and Age Group')
plt.xlabel('Geography')
plt.ylabel('Balance')
plt.xticks(rotation=45)
plt.legend(title='Age  Group')
plt.show()
```

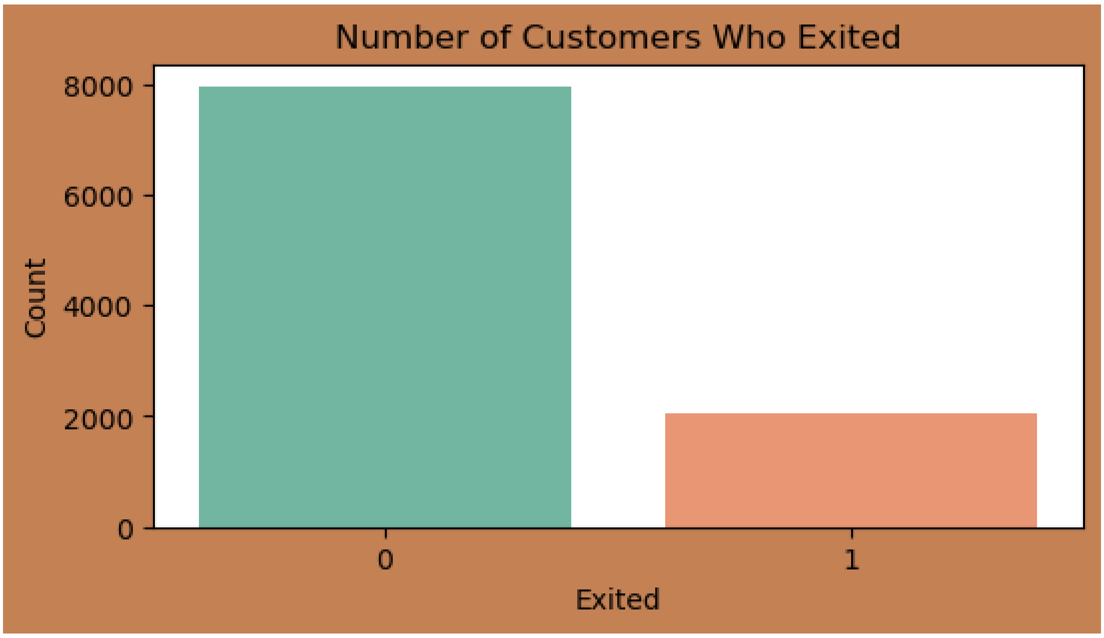


In [34]:
```
plt.figure(figsize=(10, 4), facecolor="#C38154")
sns.barplot(x='Gender', y='EstimatedSalary', hue='AgeGroup', data=df, palette='Set
plt.title('Average Estimated Salary by Gender and Age Group')
plt.xlabel('Gender')
plt.ylabel('Average Estimated Salary')
plt.legend(title='Age Group')
plt.show()
```



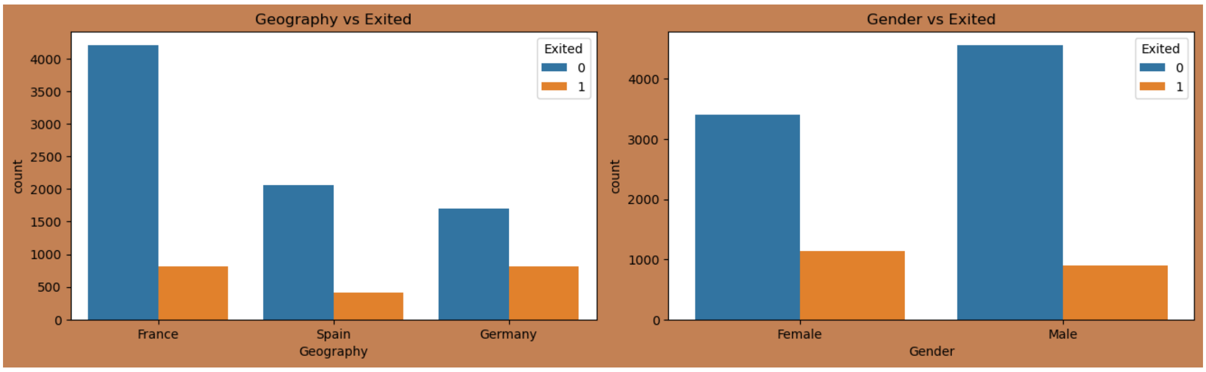
In [35]:
```
target = 'Exited'
```

In [36]:
```
plt.figure(figsize=(6, 3), facecolor="#C38154")
sns.countplot(x='Exited', data=df, palette='Set2')
plt.title('Number of Customers Who Exited')
plt.xlabel('Exited')
plt.ylabel('Count')
plt.show()
```

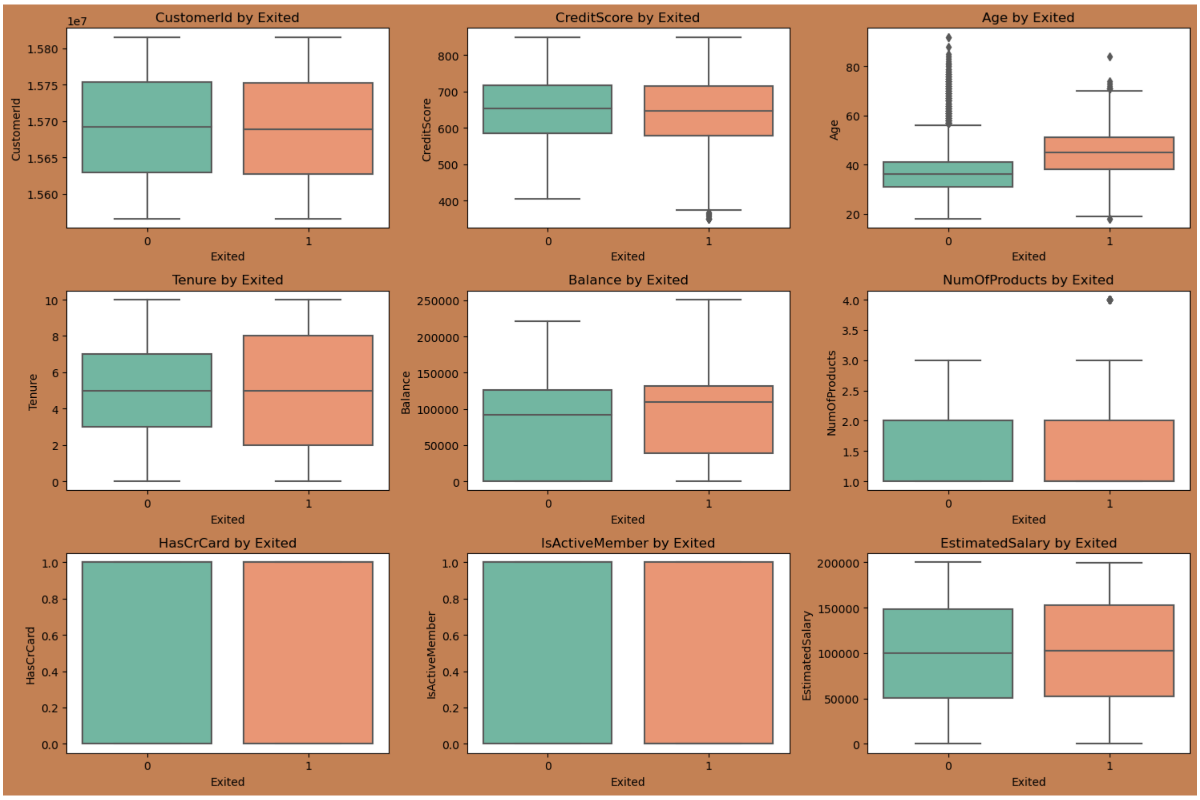


In [37]:
```
categorical_columns =['Geography','Gender']
plt.figure(figsize=(20, 15),facecolor="#C38154")
for i, col in enumerate(categorical_columns, 1):plt.subplot(4, 3, i)
    top_10_values = df[col].value_counts().nlargest(10).index
    sns.countplot(x=col, hue=target, data=df[df[col].isin(top_10_values)])plt.title(f'{col} vs {target}')
    plt.legend(title=target)
```

```
plt.tight_layout()plt.show()
```

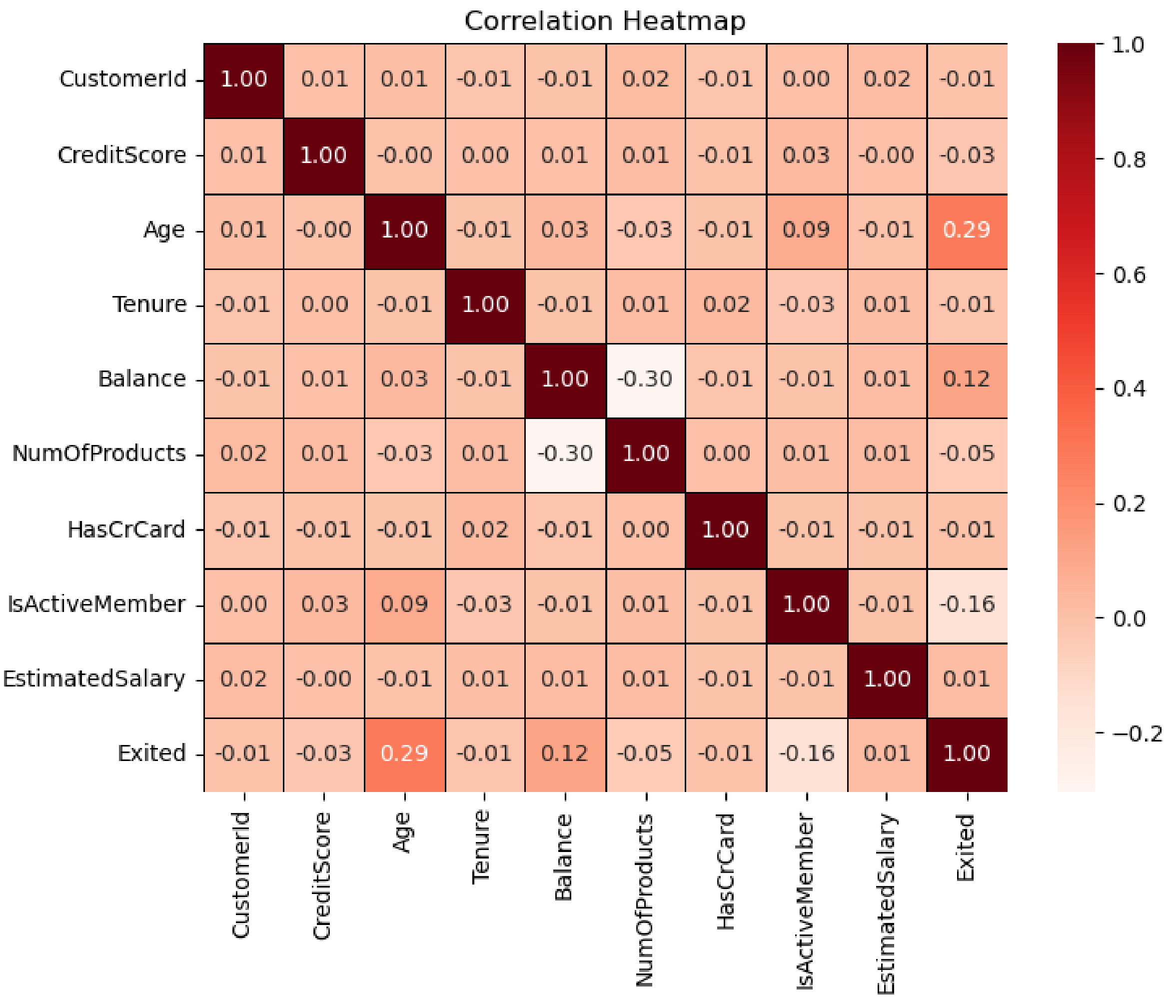


In [38]:
```
plt.figure(figsize=(15,  10),facecolor="#C38154")

numerical_columns  =  df.select_dtypes(include=['int64',  'float64']).drop(columns='E

num_columns = len(numerical_columns)
n_rows = (num_columns + 2) // 3


for i, column in enumerate(numerical_columns, 1):
    plt.subplot(n_rows, 3, i)
    sns.boxplot(x='Exited', y=column, data=df, palette='Set2')
    plt.title(f'{column} by Exited')
    plt.xlabel('Exited')
    plt.ylabel(column)

plt.tight_layout()
plt.show()
```



In [39]:
```
numeric_df  =  df.select_dtypes(include=['float64',  'int64'])


#Calculate correlation matrix

corr_matrix  =  numeric_df.corr(method='pearson')
```

```
sns.heatmap(corr_matrix,  annot=True,cmap='Reds',  linecolor='black',fmt='.2f',  lineplt.title('Correlation Heatmap')

plt.show()
```



In [40]:
```
#Predictive  Modelling
```

In [41]:
```
le = LabelEncoder()
columns_to_encode  =  ['Geography',  'Gender']
for column in columns_to_encode:

    df[column]  =  le.fit_transform(df[column])
```

In [42]:
```
from  sklearn.model_selection  import  train_test_split

x = df[['CreditScore', 'Geography', 'Gender', 'Age', 'Tenure', 'Balance', 'NumOfPr
y  = df['Exited']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_st
```

In [43]:
```
#Decsion Tree
```

In [44]:
```
from  sklearn.metrics  import  accuracy_score,  classification_report



# Initialize and train the Decision Tree model

dt_model  =  DecisionTreeClassifier(random_state=42)dt_model.fit(x_train, y_train)



# Predictions

y_train_pred_dt  =  dt_model.predict(x_train)y_test_pred_dt=
dt_model.predict(x_test)
```

```
accuracy_test_dt = accuracy_score(y_test, y_test_pred_dt)

# Print accuracy

print("Decision Tree - Training Accuracy:", accuracy_train_dt)print("Decision Tree -
Testing Accuracy:", accuracy_test_dt)



# Print the classification report

print("Decision Tree - Training Classification Report:")
```

```
Decision Tree - Training Accuracy: 1.0
Decision Tree - Testing Accuracy: 0.782
Decision Tree - Training Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6356
           1       1.00      1.00      1.00      1644

    accuracy                           1.00      8000
   macro avg       1.00      1.00      1.00      8000
weighted avg       1.00      1.00      1.00      8000

Decision Tree - Testing Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.85      0.86      1607
           1       0.45      0.52      0.49       393

    accuracy                           0.78      2000
   macro avg       0.67      0.68      0.67      2000
weighted avg       0.80      0.78      0.79      2000
```

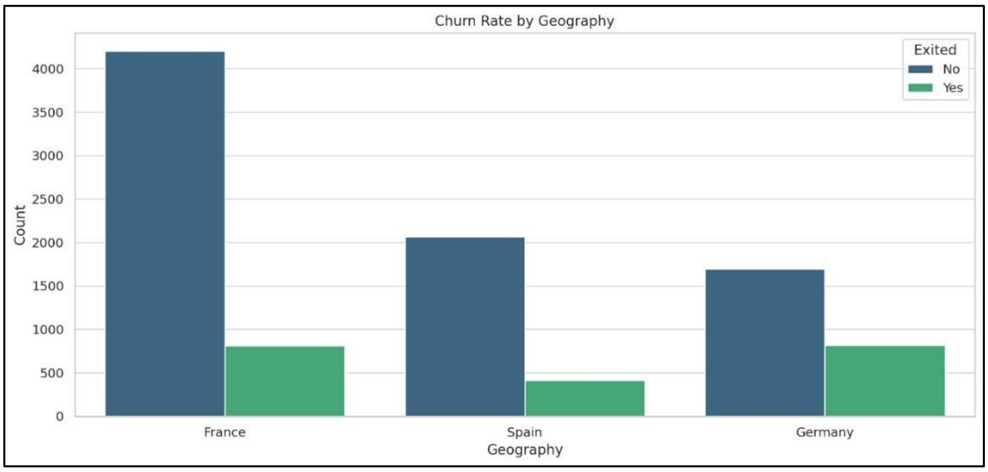
In [45]:

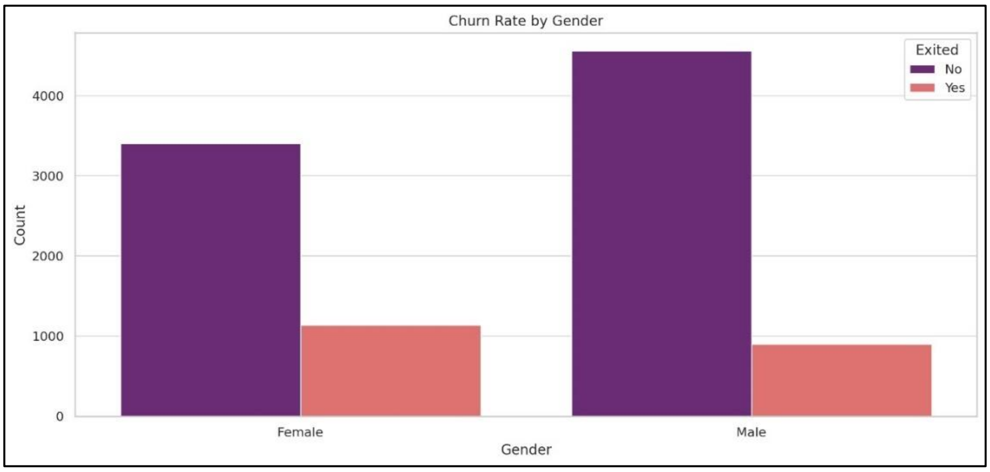
```
#Random Forest

rf_model = RandomForestClassifier(random_state=42)
rf_model.fit(x_train, y_train)

# Predictions

y_train_pred_rf = rf_model.predict(x_train)
y_test_pred_rf = rf_model.predict(x_test)

# Calculate accuracy

accuracy_train_rf = accuracy_score(y_train, y_train_pred_rf)
accuracy_test_rf = accuracy_score(y_test, y_test_pred_rf)
# Print accuracy

print("Random Forest - Training Accuracy:", accuracy_train_rf)

# Print the classification report

print("Random Forest - Training Classification Report:")
print(classification_report(y_train, y_train_pred_rf))
```

```
Random Forest - Training Accuracy: 1.0
Random Forest - Training Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6356
           1       1.00      1.00      1.00      1644

    accuracy                           1.00      8000
   macro avg       1.00      1.00      1.00      8000
weighted avg       1.00      1.00      1.00      8000

Random Forest - Testing Accuracy: 0.8645
Random Forest - Testing Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.96      0.92      1607
           1       0.75      0.47      0.57       393

    accuracy                           0.86      2000
   macro avg       0.82      0.71      0.75      2000
weighted avg       0.85      0.86      0.85      2000
```

In [ ]:

Visualisations Using Excel/PowerBi:



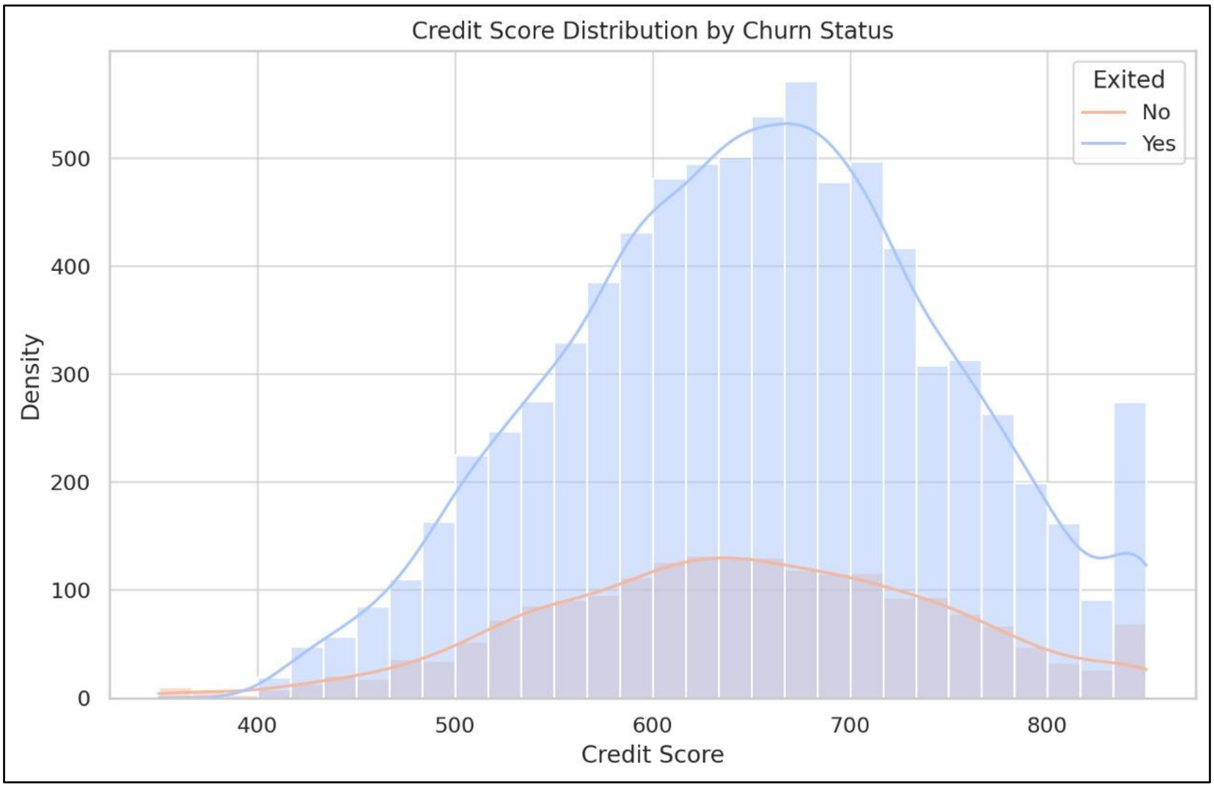


Churn Rate by Geography and Gender

Explanation: This graph displays the churn rate segmented by customer location (Geography) and gender. Bars represent the number of churned and active customers, with color coding to distinguish between them.

Insights:

- Differences in churn rate by location may indicate that certain regions (e.g., France, Germany, Spain) have higher or lower churn. This could reflect either cultural or competitive influences in these regions.
- Analyzing by gender may show whether males or females are more likely to churn. If one gender has a significantly higher churn rate, targeted retention efforts could be applied.

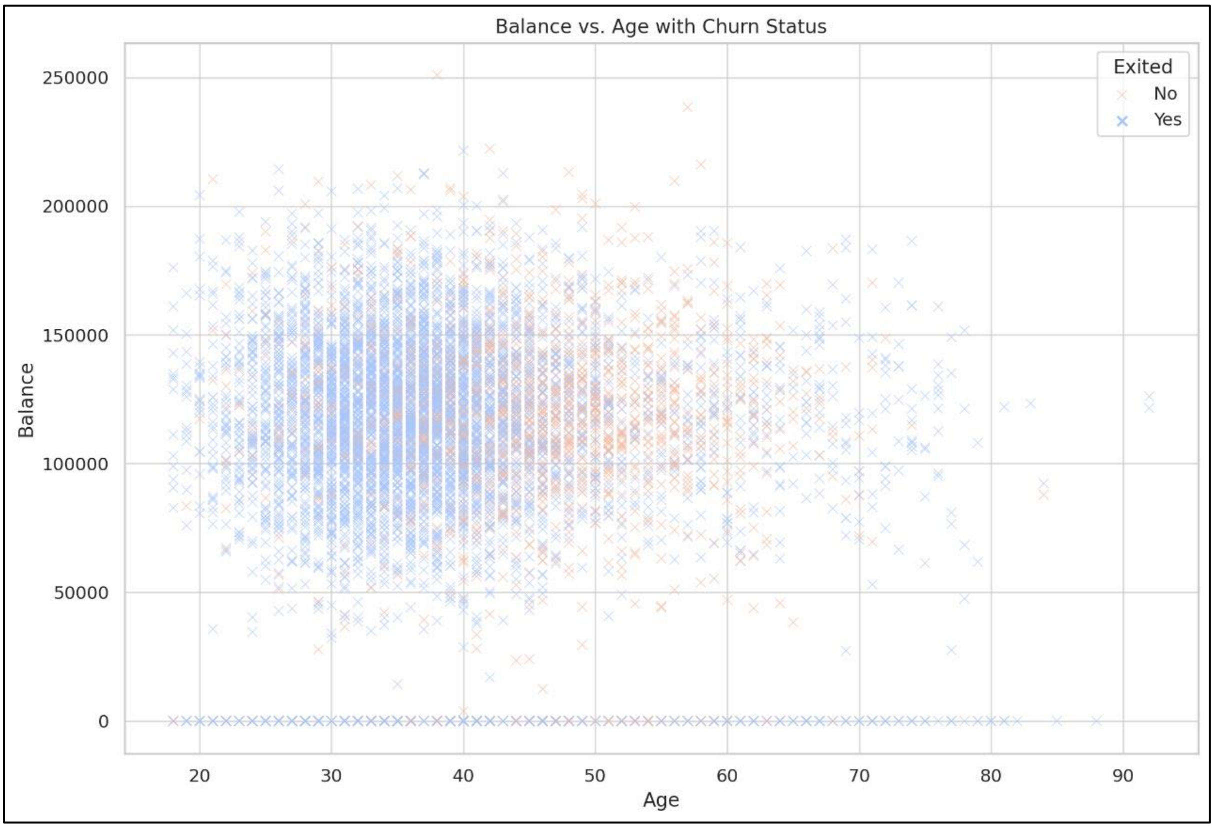


<u>Credit Score Distribution by Churn Status</u>

Explanation: This histogram shows the distribution of credit scores for churned vs. active customers. The distribution is split by churn status (churned or active), helping us see if there is a relationship between credit score and churn.

Insights:

- If churned customers have a lower credit score on average, it may suggest that customers with lower credit are more likely to leave.
- Conversely, if churned customers have higher credit scores, this may indicate a disconnect where high-credit customers are dissatisfied despite their positive credit history, requiring further investigation.

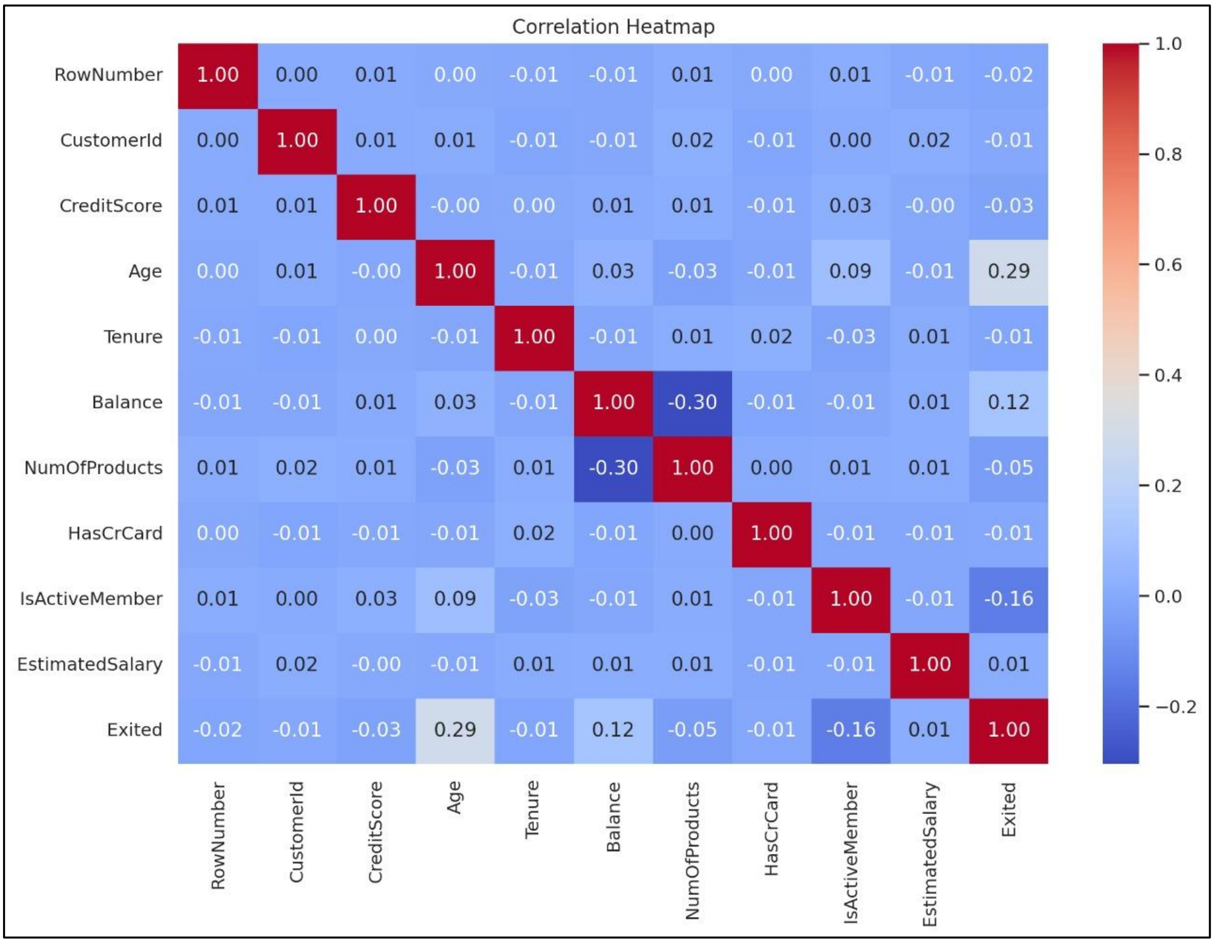


Balance vs. Age with Churn Status

Explanation: This scatter plot maps customer balance against age, with color coding to distinguish churned from active customers.

Insights:

- If churned customers cluster around low balances, it may suggest that customers with fewer assets in the bank are less engaged or feel less tied to the bank.
- Alternatively, if high-balance customers are also churning, this could indicate issues with services that appeal to affluent customers. This insight could lead to personalized service initiatives for high-balance customers to improve satisfaction and retention.



<u>Correlation Heatmap</u>

Explanation: The heatmap visualizes the correlation between numerical variables in the dataset, with each cell showing the strength and direction of the correlation.

Insights:

- A strong correlation between variables (e.g., Age and Balance) could indicate underlying patterns or customer behavior trends.
- If churn is highly correlated with a particular variable, this can serve as a flag for areas to address. For example, if balance or credit score has a high correlation with churn, it could highlight **the need to address issues with these customer segments.**

Overall Insights

These visualizations together provide a holistic view of the factors influencing customer churn. Key patterns include:

Regional and Gender Differences: The churn rate may vary significantly by geography and gender, suggesting targeted retention strategies could be effective.

Age and Financial Engagement: Younger customers or those with lower balances may be more likely to leave, indicating that financial products catering to young professionals or financially engaged customers could improve retention.

Tenure and Onboarding: High churn among new customers emphasizes the importance of early engagement, while long-tenured customers churning might indicate the need for evolving services.

These insights can guide strategic efforts to reduce churn and increase customer satisfaction by addressing the needs of specific customer segments.

Conclusion

In this project, our team successfully demonstrated the full life cycle of Business Data Analytics (BDA) through hands-on application in our chosen area,.Our goal was to integrate multiple stages of data analytics, from database design to data visualization, to reveal insights and patterns that would otherwise be hidden.

Key Accomplishments:

1. **Database Design and Creation**: We carefully analyzed the selected domain to identify the necessary entities, attributes, and relationships, culminating in a comprehensive E-R Diagram. Using MySQL, we transformed this conceptual design into a functional database, creating tables and establishing relationships to enable efficient data management.
2. **Data Export and Transformation**: The database tables were exported to Excel/CSV files, which facilitated further analysis. These files not only served as an input for our machine learning model but also supported effective visualization in platforms like Tableau and PowerBI.
3. **Machine Learning Model**: Leveraging the exported data, we developed a machine learning model to generate predictive insights. This model allowed us to better understand trends and patterns within the data, showcasing the potential impact of analytics on informed decision-making.
4. **Data Visualization**: Using Tableau and/or PowerBI, we translated our findings into visual representations, making the data accessible and actionable. These visualizations provided a clear picture of the data insights, adding value to stakeholders by enhancing data interpretation and decision-making.

Through this project, we gained a deeper understanding of the technical and analytical skills essential for executing a BDA project. By integrating tools like MySQL, machine learning models, and data visualization platforms, we showcased the importance of a structured approach in data analytics to drive actionable insights. This end-to-end project highlights the transformative power of analytics in understanding and addressing complex business problems.
