Comprehensive Banking Portfolio & Risk Analysiss

This analysis explores the banking customer dataset to uncover key patterns in account holdings, loan portfolios, income profiles, and transactional behavior. By integrating both financial and demographic variables, we examine customer engagement, lending trends, and risk exposure across different segments. The insights aim to support data-driven decision-making for improving product strategies, managing portfolio risk, and enhancing overall banking performance.

Install MySQL Connector and Connect Database in Python Notebook

```
!pip install mysql-connector-python # Install MySQL Connector

import mysql.connector
import pandas as pd

# Connect to the database
conn = mysql.connector.connect(
    host="your_host",       # e.g., "localhost" or IP
    user="your_username",   # e.g., "root"
    password="your_password",
    database="your_database"
)

# Create a cursor object
cursor = conn.cursor()
print("Database connected successfully!")

# Fetch Data from MySQL into pandas dataframe
query = "SELECT * FROM banking_case.customer" # Define SQL query
df = pd.read_sql(query, conn)
conn.close() # Close the connection
```

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
df = pd.read_csv('Banking.csv')
```

In [3]:
```
df.shape
```

Out[3]: (3000, 25)

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

```
Displays concise summary of the DataFrame including:
- Index dtype and range
 - Column names, non-null counts, and data types
- Memory usage of the DataFrame
```

In [ ]:
```
# describe the descriptive statistics
df.describe()
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['Client ID', 'Name', 'Age', 'Location ID', 'Joined Bank',
       'Banking Contact', 'Nationality', 'Occupation', 'Fee Structure',
       'Loyalty Classification', 'Estimated Income', 'Superannuation Savings',
       'Amount of Credit Cards', 'Credit Card Balance', 'Bank Loans',
       'Bank Deposits', 'Checking Accounts', 'Saving Accounts',
       'Foreign Currency Account', 'Business Lending', 'Properties Owned',
       'Risk Weighting', 'BRId', 'GenderId', 'IAId'],
      dtype='object')
```

In [7]:
```
# Convert Joined Bank to datetime format
df['Joined Bank'] = pd.to_datetime(df['Joined Bank'], dayfirst=True)
```

In [8]:
```
df['Joined Bank'].head(4)
```

Out[8]:
```
0   2019-05-06
1   2001-12-10
2   2010-01-25
3   2019-03-28
Name: Joined Bank, dtype: datetime64[ns]
```

In [9]:
```
# Feature Engineering
df['Join Year'] = df['Joined Bank'].dt.year
df['Bank Tenure (Years)'] = (pd.to_datetime("today") - df['Joined Bank']).dt.days /
```

In [10]:
```
df['Join Year'].head(5)
```

Out[10]:
```
0    2019
1    2001
2    2010
3    2019
4    2012
Name: Join Year, dtype: int32
```

In [11]:
```
# Get column types
numerical_cols = df.select_dtypes(include='number').columns
categorical_cols = df.select_dtypes(include='object').columns.tolist()
```

In [12]:
```
# Remove identifier-like columns from numerical set
numerical_cols = [col for col in numerical_cols if col not in ['Location ID', 'Chec
```

In [13]:
```
categorical_cols
```

Out[13]:
```
['Client ID',
 'Name',
 'Banking Contact',
 'Nationality',
 'Occupation',
 'Fee Structure',
 'Loyalty Classification']
```

In [14]:
```
# Remove identifier-like columns from categorical set
categorical_cols = [col for col in categorical_cols if col not in ['Client ID', 'Na
```

In [15]:
```
categorical_cols
```

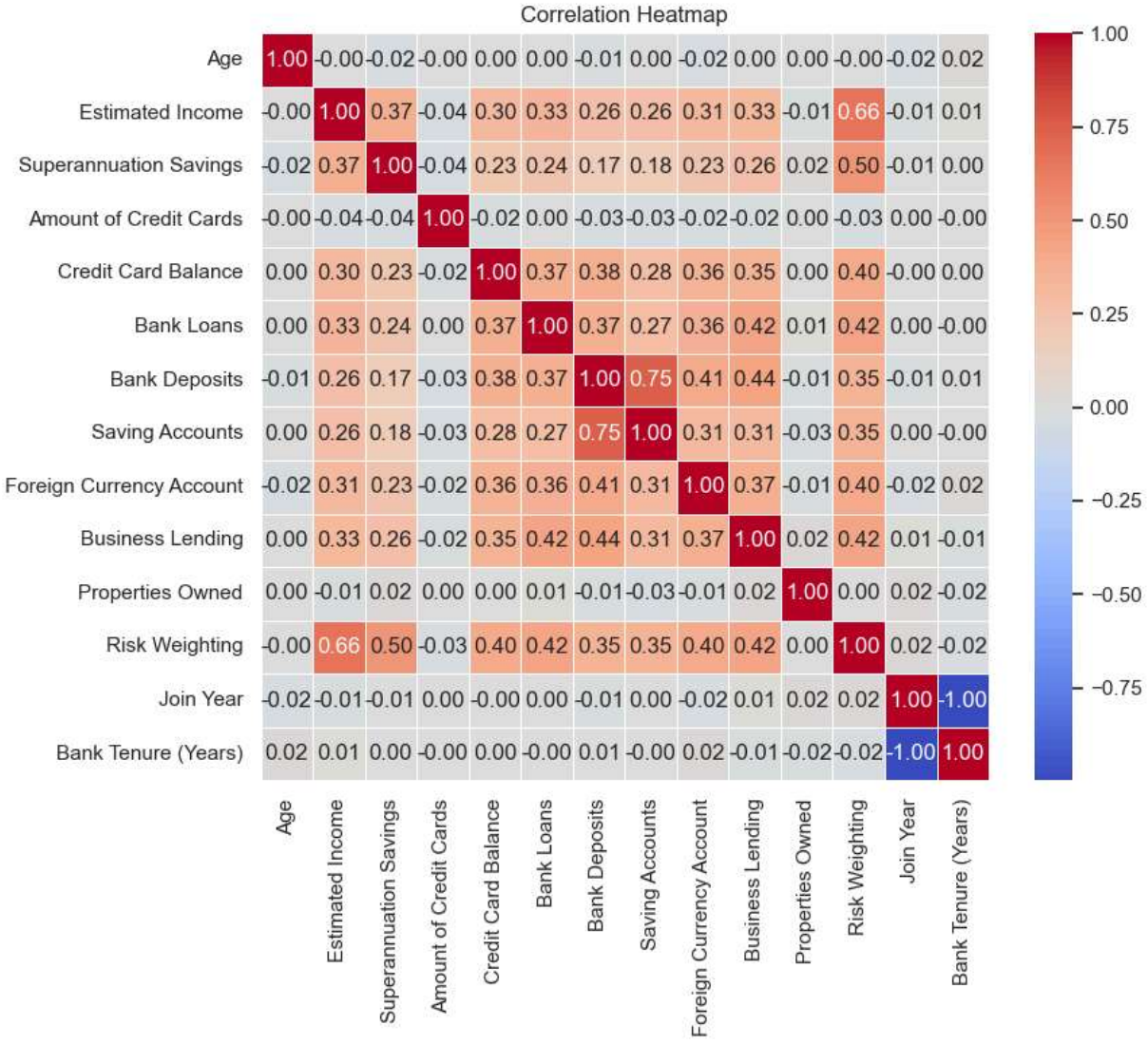
Out[15]:
```
['Nationality', 'Occupation', 'Fee Structure', 'Loyalty Classification']
```

Set visual style

In [16]:
```
sns.set(style="whitegrid")
plt.rcParams['figure.figsize'] = (10, 5)
```

Correlation Matrix

In [17]:
```
plt.figure(figsize=(9, 8))
corr = df[numerical_cols].corr()
sns.heatmap(corr, annot=True, fmt=".2f", cmap='coolwarm', linewidths=0.6)
plt.title("Correlation Heatmap")
plt.tight_layout()
plt.show()
```



🧮 Correlation Insights

From the correlation heatmap, we derive the following insights:

Strong Positive Correlations

Feature 1	Feature 2	Correlation	Insight
Checking Accounts	Bank Deposits	High	Customers with more checking accounts tend to hold higher deposit balances. Suggests a strong tie between account usage and deposits.
Bank Loans	Business Lending	High	Indicates that business customers who take loans often have multiple types of lending products.
Estimated Income	Superannuation Savings	High	Individuals with higher income are more likely to invest in retirement or long-term savings products.

Weak or No Correlation

Feature 1	Feature 2	Correlation	Insight
Amount of Credit Cards	Credit Card Balance	–0.02	There is almost no linear relationship. More cards doesn't mean higher balances. This may be due to card hoarding without usage or other behavioral patterns.
Risk Weighting	Monetary Variables	Low	Risk Weighting does not show strong linear correlation with key financial metrics, indicating it's influenced by other behavioral or demographic factors.

Interpretation Tips

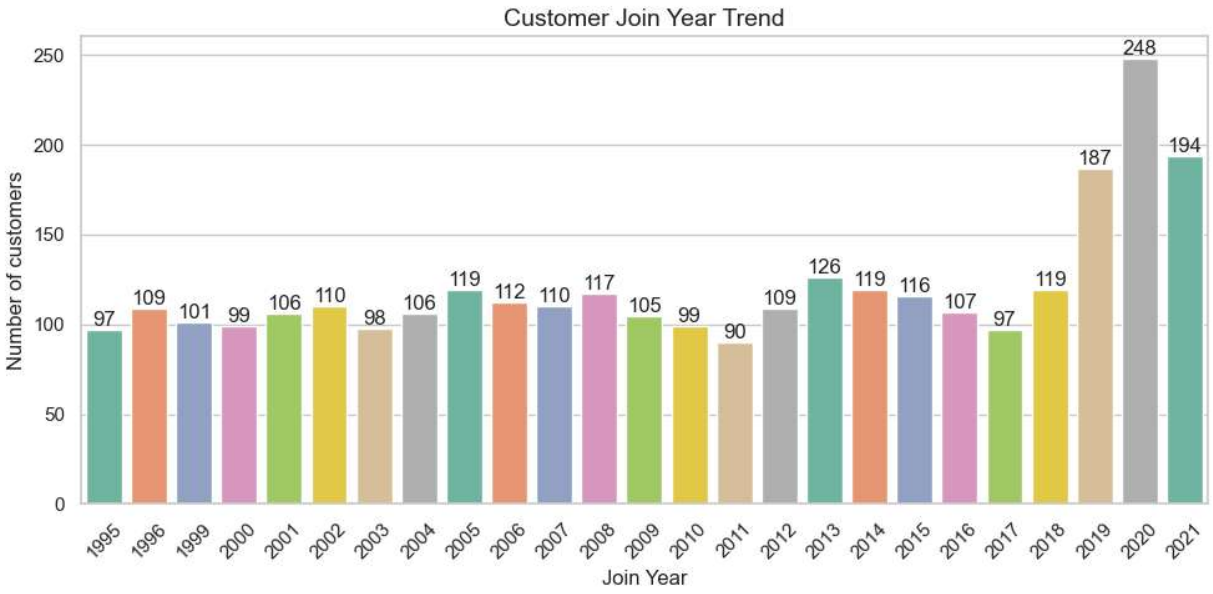
- Strong correlations suggest **feature redundancy** and should be examined for multicollinearity in models.
- Weak correlations can still be useful in **non-linear models** (e.g., trees, boosting).
- Consider creating **derived features** like:
  - `Avg. Credit Balance per Card`
  - `Deposit to Income Ratio`
  - `Loan to Deposit Ratio`

Customer Join Year Trend

In [18]:
```
ax = sns.countplot(data=df, x='Join Year', palette='Set2')

for bars in ax.containers:
    ax.bar_label(bars)

plt.title("Customer Join Year Trend", fontsize=14)
plt.xlabel("Join Year", fontsize=12)
plt.ylabel("Number of customers", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Customer Join Year Trend

Overview

The `Join Year` feature reflects the year in which customers joined the bank. Visual analysis using a count plot reveals notable trends in customer acquisition over time.

Key Observations

- **Upward Trend Post-2017**:
  - A **consistent increase in customer acquisition** is observed after 2017.
  - This indicates a successful period of **business growth, marketing, or new branch openings**.
- **Peak in 2020**:
  - The highest number of customers joined in **2020**, with **248 accounts opened**.
  - This could be due to:
    - Digital banking push during COVID-19,
    - Customer migration to online platforms,
    - Attractive product offerings during uncertain economic periods.
- **Slight Decline in 2021**:
  - In 2021, new customer count dropped to **194.**
  - This might reflect:
    - Market saturation,
    - Decreased marketing efforts,
    - Economic slowdowns or lockdown fatigue.

Implications for Business Strategy

Insight	Application
Growth Period	Focus on analyzing what worked post-2017 to replicate success.
Product Analytics	Identify which products or campaigns were most effective during high-growth years.
Investigate 2021 Dip	Analyze internal (strategy, staffing) or external (pandemic, economy) causes for reduced joins.

| **Cohort Analysis** | Use `Join Year` for retention and churn behavior across cohorts.

Summary

- **Customer joins steadily increased after 2017**, peaking in **2020 with 248 new customers**.
- **A decline in 2021 (194 joins)** suggests a need to evaluate engagement or outreach strategies.
- Join Year analysis is crucial for **trend tracking**, **cohort analysis**, and **growth planning**. `

Countplot for categorical columns

In [19]:
```
categorical_cols
```

Out[19]:
```
['Nationality', 'Occupation', 'Fee Structure', 'Loyalty Classification']
```

In [20]:
```
df['Fee Structure'].value_counts()
```

Out[20]:
```
Fee Structure
High    1476
Mid      962
Low      562
Name: count, dtype: int64
```

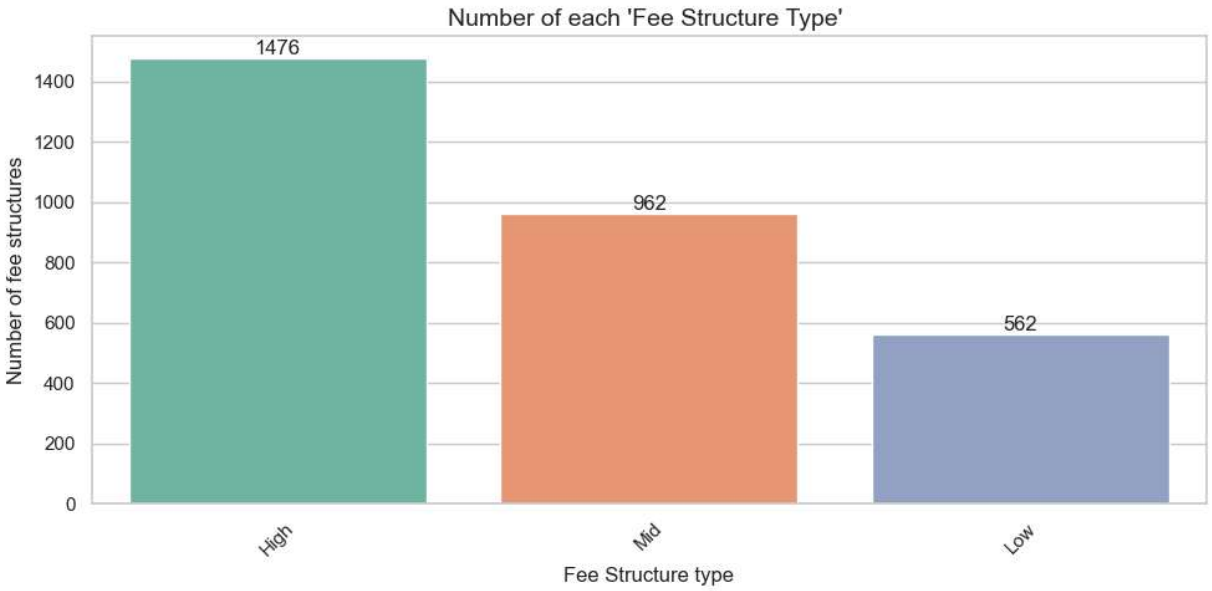
Fee Structure type

In [21]:
```
ax = sns.countplot(data=df, x='Fee Structure', palette='Set2')

for bars in ax.containers:
    ax.bar_label(bars)

plt.title("Number of each 'Fee Structure Type' ", fontsize=14)
plt.xlabel("Fee Structure type", fontsize=12)
plt.ylabel("Number of fee structures", fontsize=12)
plt.xticks(rotation=45)
```

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



Loyality classification type

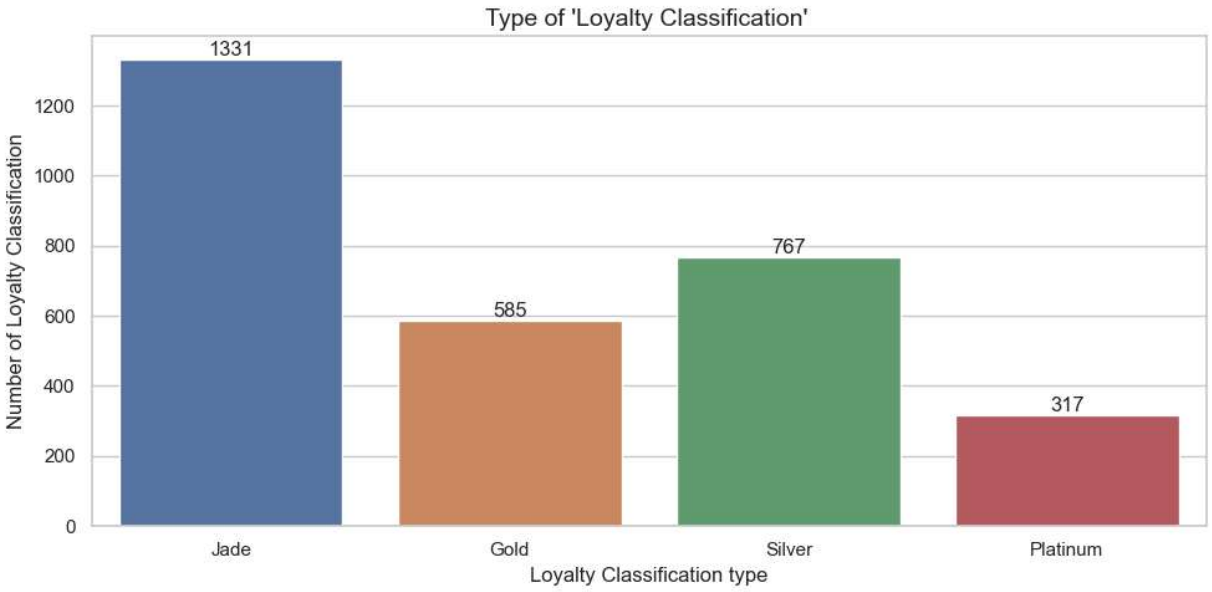
In [22]:
```
def format_percent(x):
    return f"{x:.2f}%"
```

In [23]:
```
ax = sns.countplot(data=df, x='Loyalty Classification', hue='Loyalty Classification

for bars in ax.containers:
    ax.bar_label(bars)

plt.title("Type of 'Loyalty Classification' ", fontsize=14)
plt.xlabel("Loyalty Classification type", fontsize=12)
plt.ylabel("Number of Loyalty Classification", fontsize=12)
plt.tight_layout()
plt.show()

Loyalty_classification_rate = df['Loyalty Classification'].value_counts(normalize=T
Loyalty_classification_rate = Loyalty_classification_rate.apply(format_percent)
print(Loyalty_classification_rate)
```




```
Loyalty Classification
Jade        44.37%
Silver      25.57%
Gold        19.50%
Platinum    10.57%
Name: proportion, dtype: object
```

Loyalty Classification Analysis

Distribution (%) by Tier:

Tier	Percentage	Interpretation
Jade	44.37%	Base/entry-level tier
Silver	25.57%	Mid-tier (common upgrade)
Gold	19.50%	Higher-tier (engaged customers)
Platinum	10.57%	Elite/premium tier (rarest)

Why Jade Dominates (44.37%):

1. **Entry-Level Accessibility**:
   - Jade is likely the **default tier** for new customers, requiring minimal spend/activity.
   - Example: Sign-up bonuses or automatic enrollment.
2. **Low Barrier to Retention**:
   - Customers may not spend enough to qualify for higher tiers (Silver/Gold/Platinum).
   - Tiers like Platinum often require **high lifetime value** (e.g., frequent purchases).
3. **Program Design**:
   - Loyalty programs often follow a **pyramid structure** (many at the base, few at the top).

- Encourages customers to "climb" tiers (e.g., Jade → Silver) via increased engagement.

Fee Structure Insight:

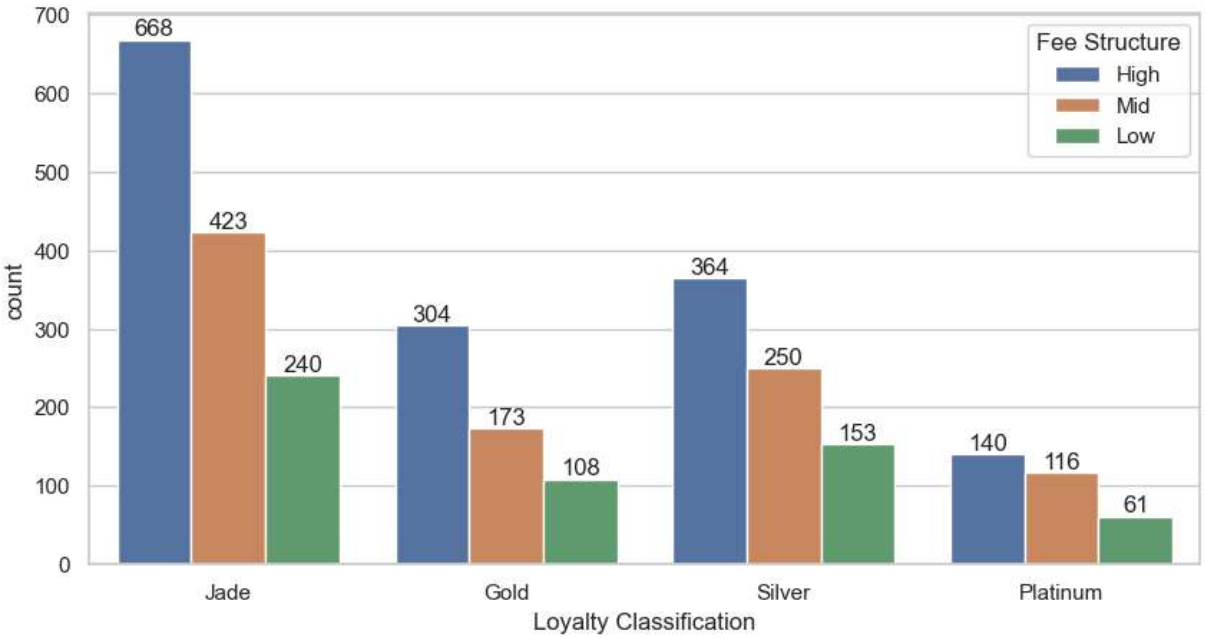
- **High fees dominate all tiers**, suggesting:
  - Revenue focus (e.g., annual fees for tier benefits).
  - Potential perks (e.g., Jade gets basic rewards; Platinum gets concierge services).

Actionable Takeaways:

- **Jade Tier**: Optimize for conversion (e.g., incentivize upgrades to Silver).
- **Platinum Tier**: Enhance exclusivity to retain high-value customers.

Loyality classification type vs Fees structure

In [24]:
```
ax = sns.countplot(data=df, x='Loyalty Classification', hue='Fee Structure')
for bars in ax.containers:
    ax.bar_label(bars)
```



Loyalty Classification Summary

Key Observations:

1. **Distribution of Loyalty Tiers**:
   - **Jade**: Most common tier.
   - **Silver**: Moderately common.
   - **Gold**: Less common than Silver but more than Platinum.
   - **Platinum**: Rarest tier.

2. **Fee Structure Dominance**:
    - **High** fees are the most prevalent across all tiers.
3. **Tier Hierarchy Implication**:
    - Likely order (ascending): Jade → Silver → Gold → Platinum (highest prestige).

Insights:

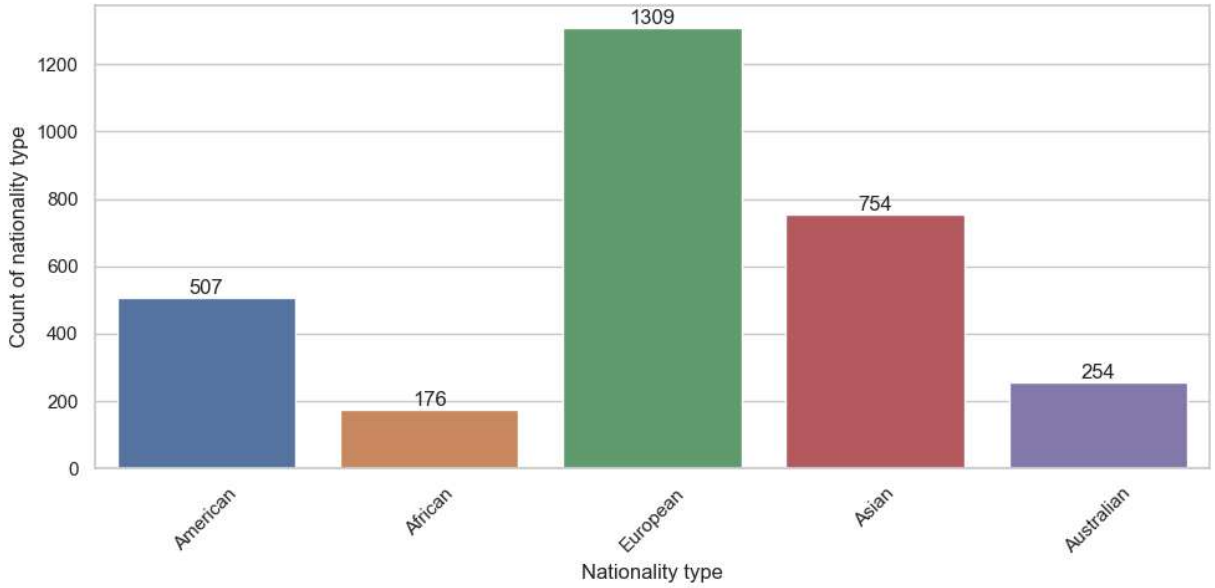
- Platinum customers may receive premium benefits (given their scarcity).
- High fees suggest the program prioritizes revenue or targets high-value customers.
- Jade's prevalence indicates it's the entry-level/default tier.

Nationality

In [25]:
```
ax = sns.countplot(data=df, x='Nationality', hue='Nationality')
for bars in ax.containers:
    ax.bar_label(bars)

plt.xlabel("Nationality type", fontsize=12)
plt.ylabel("Count of nationality type", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

Nationality_rate = df['Nationality'].value_counts(normalize=True) * 100
Nationality_rate = Nationality_rate.apply(format_percent)
print(Nationality_rate)
```



```
Nationality
European      43.63%
Asian         25.13%
American      16.90%
Australian     8.47%
African        5.87%
Name: proportion, dtype: object
```

Nationality Distribution in Banks

- **European**: 43.63%
- **Asian**: 25.13%
- **American**: 16.90%
- **Australian**: 8.47%
- **African**: 5.87%

Distribution Plot for numerical columns

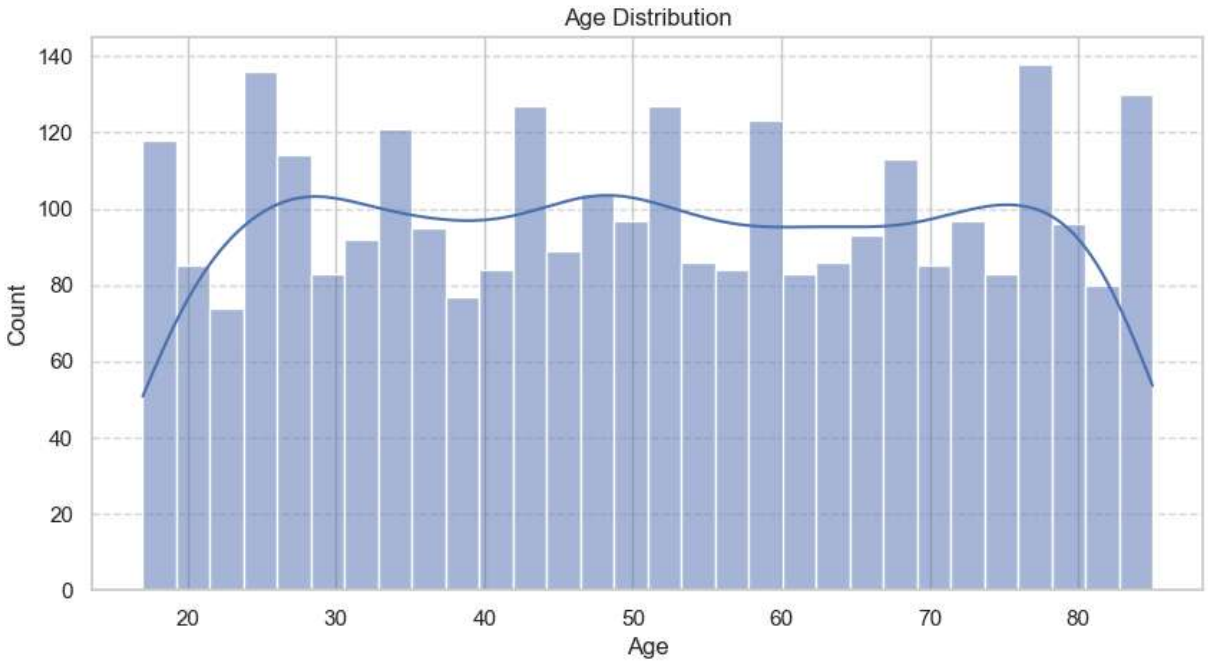
In [26]:
```
numerical_cols
```

Out[26]:
```
['Age',
 'Estimated Income',
 'Superannuation Savings',
 'Amount of Credit Cards',
 'Credit Card Balance',
 'Bank Loans',
 'Bank Deposits',
 'Saving Accounts',
 'Foreign Currency Account',
 'Business Lending',
 'Properties Owned',
 'Risk Weighting',
 'Join Year',
 'Bank Tenure (Years)']
```

Distribution of Age

In [27]:
```
 plt.grid(axis='y', linestyle='--', alpha=0.7)

sns.histplot(df['Age'], bins=30, kde=True)
plt.title("Age Distribution")
plt.show()
```



Age: Fairly Even Distribution with Peaks at 30 and 45

Distribution Overview

The `Age` variable shows a **fairly even distribution** across the adult age range, with **notable peaks near ages 30 and 45**.

> This suggests that a large portion of the bank's customer base falls into **young adult** and **mid-career** age groups.

Key Observations

- **Even Spread**: Customers are spread across a wide range of ages.
- **Peak at ~30**: Likely includes early-career professionals or individuals entering financial independence.
- **Peak at ~45**: May represent mid-career individuals with established income and more complex financial needs (loans, investments).
- **Lower Count <25 or >60**: Indicates fewer very young or retired individuals in the dataset.

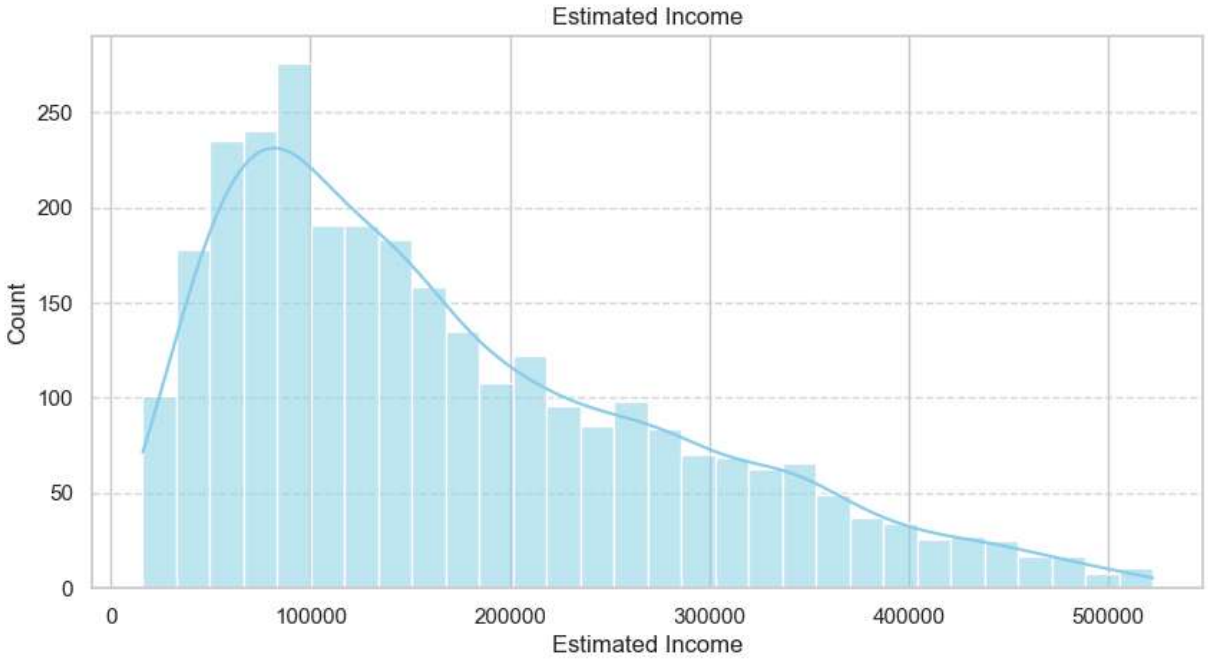
Summary

- `Age` is **fairly evenly distributed** with **two significant peaks at ages 30 and 45.**
- These patterns align with **financially active life stages** and can guide product development, risk modeling, and marketing efforts.

Distribution of Estimated Income

In [28]:
```
plt.grid(axis='y', linestyle='--', alpha=0.7)

sns.histplot(df['Estimated Income'], bins=30, kde=True, color='skyblue')
plt.title("Estimated Income")
plt.show()
```




```
Estimated Income: Right-skewed with most people earning under ₹2.5
Lakh.
```

Estimated Income: Right-Skewed Distribution

What is Observed?

The `Estimated Income` variable in the dataset follows a **right-skewed distribution**, which means:

- A **large portion of individuals earn under ₹2.5 Lakh**,
- While a **smaller group earns significantly more**, pulling the distribution's tail to the right.

> This is common in income data, where high-income earners are fewer but have disproportionately higher values.

Statistical Characteristics

- **Skewness**: Positive (right skew)
- **Concentration**: Majority of values lie between ₹0 – ₹2.5 Lakh
- **Tail**: A long right tail indicates **high-income outliers** (possibly executives, business owners, or investors)

Interpretation & Real-World Insight

Aspect	Insight
Income Inequality	Reflects real-world economic inequality — most people earn modestly, while a few earn much more.
Targeting Strategy	Useful for marketing and financial product segmentation (e.g., premium vs. basic credit cards).
Financial Inclusion	A large number of low-income users may require different product strategies or risk assessments.

Possible Feature Engineering

- `Income Bracket` (e.g., Low / Medium / High)
- `Income-to-Loan Ratio`
- `Normalized Income` (after log transform)
- `Relative Income Rank` (e.g., percentile rank)

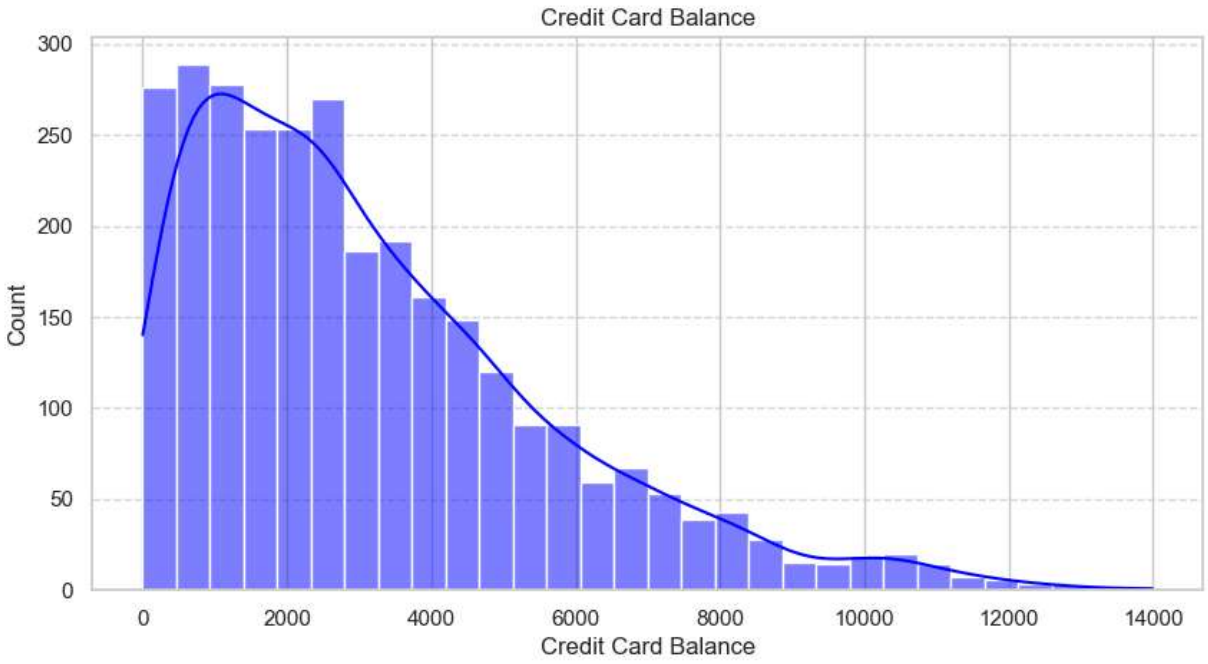
Summary

- `Estimated Income` is **right-skewed**, with most users earning less than ₹2.5 Lakh.
- High-income outliers exist and require **transformation or segmentation** before modeling.
- Proper handling of this feature is critical for **fair and effective predictive analysis**.

Distribution of Credit Card Balance

In [29]:
```
plt.grid(axis='y', linestyle='--', alpha=0.7)

sns.histplot(df['Credit Card Balance'], bins=30, kde=True, color='blue')
plt.title("Credit Card Balance")
plt.show()
```



Credit Card Balance: Right-Skewed Distribution

What is a Right-Skewed Distribution?

A **right-skewed (positively skewed)** distribution is one where:

- Most data points cluster on the **left (lower values)**,
- And the **tail extends to the right**, representing a few extremely high values.

In this context:

> **Most customers have low-to-moderate credit card balances, while a few carry very high balances.**

Observations in the Dataset

- The distribution of `Credit Card Balance` is **heavily concentrated at lower values**.
- A small group of customers have **exceptionally high balances**, possibly due to high spending behavior, lack of repayments, or premium cardholders.
- This results in a **long right tail**.

Interpretation Insights

- Right skewness in credit card balances is common in **consumer finance** — spending habits and financial discipline vary widely.
- May indicate the presence of **risky customers** or **high-value clients**.

- Enables **segmentation** into low spenders vs high spenders for marketing, credit risk analysis, etc.

Recommended Actions

- **Log-transform** `Credit Card Balance` to reduce skewness for modeling.
- **Detect and handle outliers** (e.g., top 1%) for robust statistical inference.
- Create derived metrics:
  - `Balance-to-Income Ratio`
  - `Average Monthly Credit Usage`

Summary

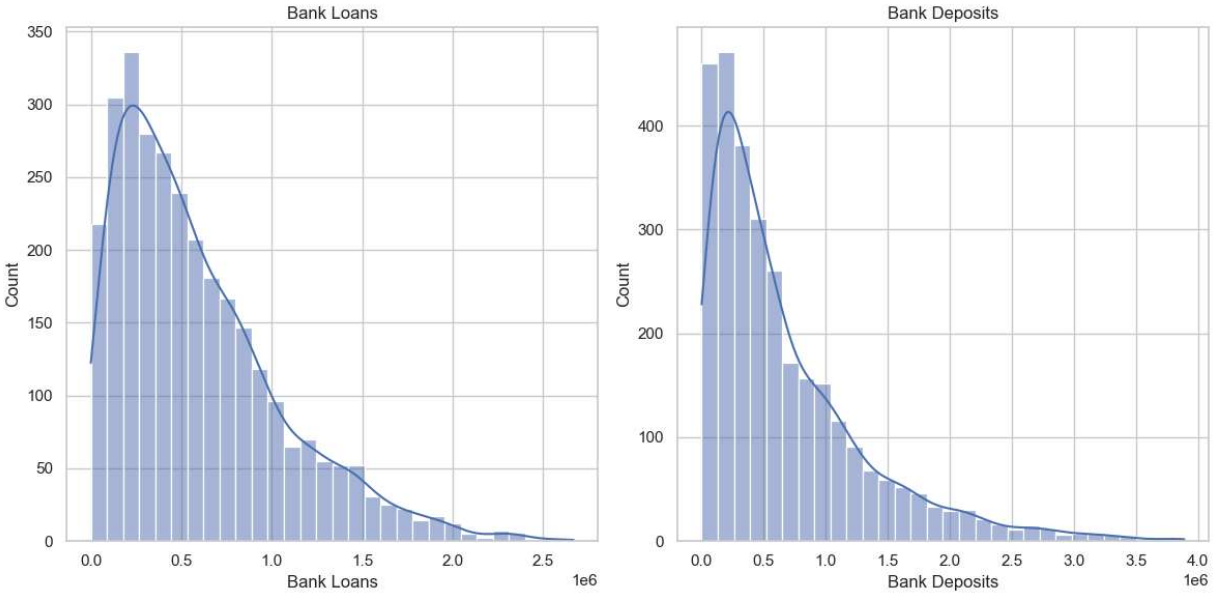
- `Credit Card Balance` exhibits a **right-skewed distribution**.
- This is typical in financial datasets and requires **preprocessing** before use in analytics or machine learning.
- Proper handling enhances **model performance** and ensures **fairer insights** across customer segments.

Distrubution of Bank Loans & Bank Deposits

In [30]:
```
categorical_col= ["Bank Loans", "Bank Deposits"]
plt.figure(figsize=(12, 6))

for i, col in enumerate(categorical_col):
    plt.subplot(1, 2, i + 1)  # 1 row, 2 columns
    sns.histplot(data=df[col], bins=30, kde=True)
    plt.title(col)

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



Bank Loans and Deposits: Heavy-Tailed Distributions

What is a Heavy-Tailed Distribution?

A **heavy-tailed distribution** is a probability distribution where:

- A **large number of observations are concentrated near the lower end** (e.g., small values), and
- A **small number of observations extend far into the higher range**, producing a "tail" that decays slowly.

In simple terms:

> **Most people have small to moderate values, but a few have extremely large amounts.**

Observations in the Dataset

Bank Loans

- Majority of customers have **low to moderate loan amounts**.
- A small group has **very high loans**, possibly business or premium customers.
- Distribution is **right-skewed** (long tail on the right side).

Bank Deposits

- Most customers maintain **low deposit balances**.
- A few customers have **very large deposits**, pulling the tail of the distribution to the right.
- Also **right-skewed**, showing typical wealth inequality patterns.

Summary

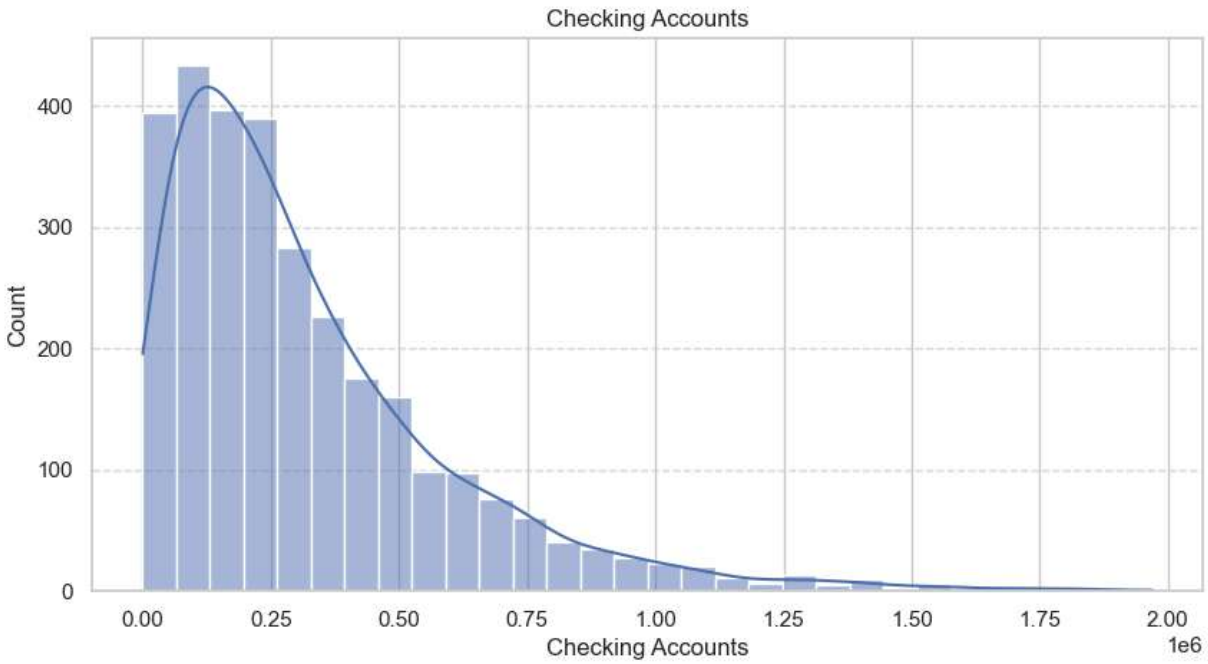
- Both **Bank Loans** and **Bank Deposits** follow a **heavy-tailed, right-skewed distribution**.
- This is typical in financial data where **a few high-value customers account for a disproportionate share of total value**.
- Careful preprocessing is needed before statistical analysis or modeling.

Distrubution of Checking Accounts

The ***Checking Accounts*** column in your dataset likely refers to the balances customers hold in their checking (or current) bank accounts.

In [31]:
```
plt.grid(axis='y', linestyle='--', alpha=0.7)

sns.histplot(df['Checking Accounts'], bins=30, kde=True)
plt.title("Checking Accounts")
plt.show()
```



Distribution Overview

The `Checking Accounts` variable displays a **right-skewed (positively skewed)** distribution, meaning:

- **Most customers have low checking account balances**,
- While a **few customers maintain significantly higher balances.**

> This distribution is typical in banking, where only a minority of clients hold large amounts in checking accounts.

Interpretation & Business Insights

Insight	Implication
Low Liquidity Users	Most users keep **small operational balances**, possibly using the account primarily for transactions.
High Liquidity Customers	A minority maintain high balances, possibly for business use, savings, or as HNWIs (High Net-Worth Individuals).
Segmentation Opportunity	Identifies premium customers who may be offered **tailored banking products** or investment services.

Summary

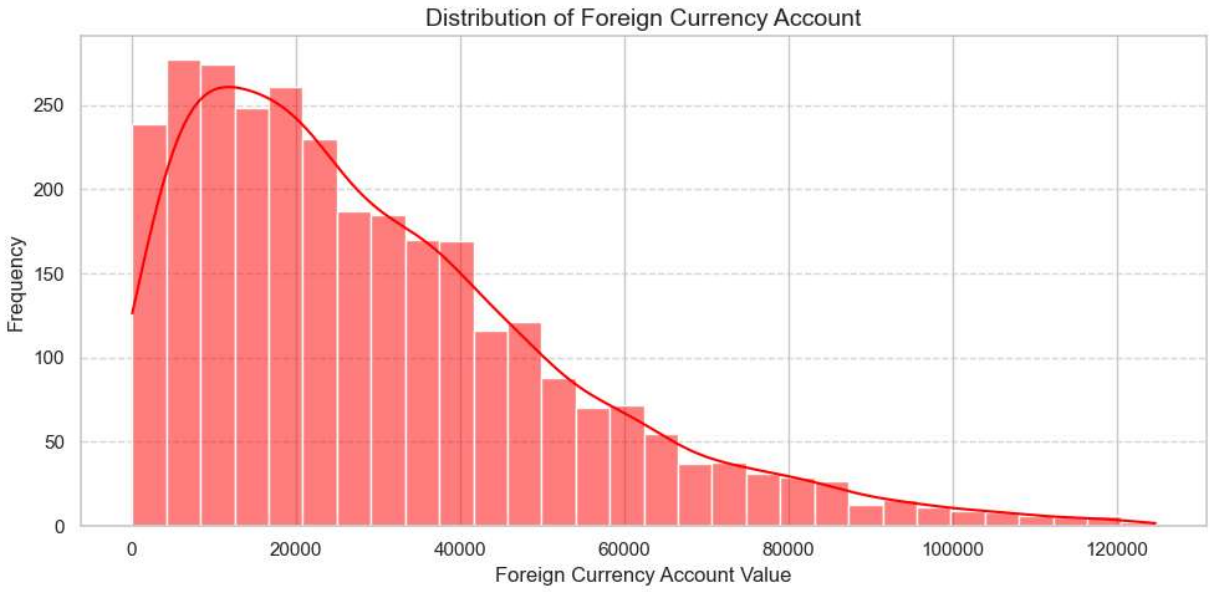
- `Checking Accounts` is **right-skewed**, with most customers holding small balances and a few holding significantly higher ones.
- This skewness is **financially meaningful** and should be carefully handled in **EDA, modeling, and customer profiling**.

Distrubution of Foreign Currency Account

In [32]:
```
sns.histplot(df['Foreign Currency Account'], bins=30, kde=True, color='red')

plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.title("Distribution of Foreign Currency Account", fontsize=14)
plt.xlabel("Foreign Currency Account Value", fontsize=12)
plt.ylabel("Frequency", fontsize=12)
plt.tight_layout()
plt.show()
```



Distribution Overview

The `Foreign Currency Account` variable shows a **right-skewed distribution**:

- Most customers hold **lower balances** in their foreign currency accounts.
- The frequency **declines sharply after ₹20,000**, with very few customers holding higher amounts.

> This pattern is typical in banking, where only a subset of customers deal actively in foreign currencies — often those involved in international business or travel.

Key Observations

- **Right-Skewed**: Majority of account values are concentrated below ₹20,000.
- **Long Tail**: A few customers maintain much larger balances (potential outliers or premium clients).
- **KDE Curve**: Density decreases steadily after ₹20,000, confirming the skew.

Business Interpretation

Insight	Implication
Low Usage Among Most Customers	Indicates foreign currency accounts are not commonly used across the customer base.
Premium Customer Indicator	High balances could indicate international businesses, frequent travelers, or NRIs.

Insight	Implication
Product Targeting	Useful for targeting forex services, investment products, or international banking solutions.

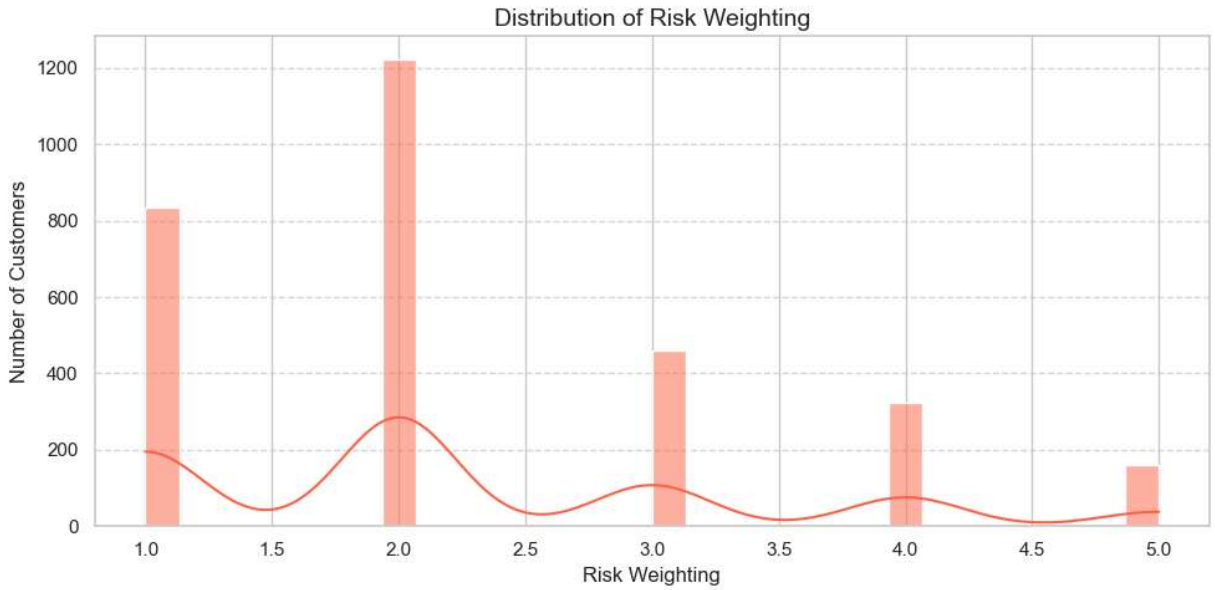
Summary

- `Foreign Currency Account` is **right-skewed**, with most customers holding **modest balances under ₹20,000.**
- The skew provides insight into **customer behavior and international banking engagement.**
- Should be carefully **transformed or engineered** before use in modeling pipelines.

Distrubution of Risk Weighting

In [33]:
```
sns.histplot(df['Risk Weighting'], bins=30, kde=True, color='tomato')

plt.title("Distribution of Risk Weighting", fontsize=14)
plt.xlabel("Risk Weighting", fontsize=12)
plt.ylabel("Number of Customers", fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



Distribution of Risk Weighting

This histogram visualizes the distribution of the **Risk Weighting** variable, which is assumed to reflect a customer's **credit or lending risk score** (higher values indicate greater risk).

Observations:

- The distribution is **right-skewed**, with the majority of customers having **lower risk weighting values.**
- A **long tail** on the right indicates a **small group of high-risk customers.**
- The **peak** is near the lower end, showing that most customers are categorized as **low-risk** based on this metric.
- The presence of a KDE (Kernel Density Estimate) line further confirms the **concentration of customers around the lower risk values.**

Interpretation:

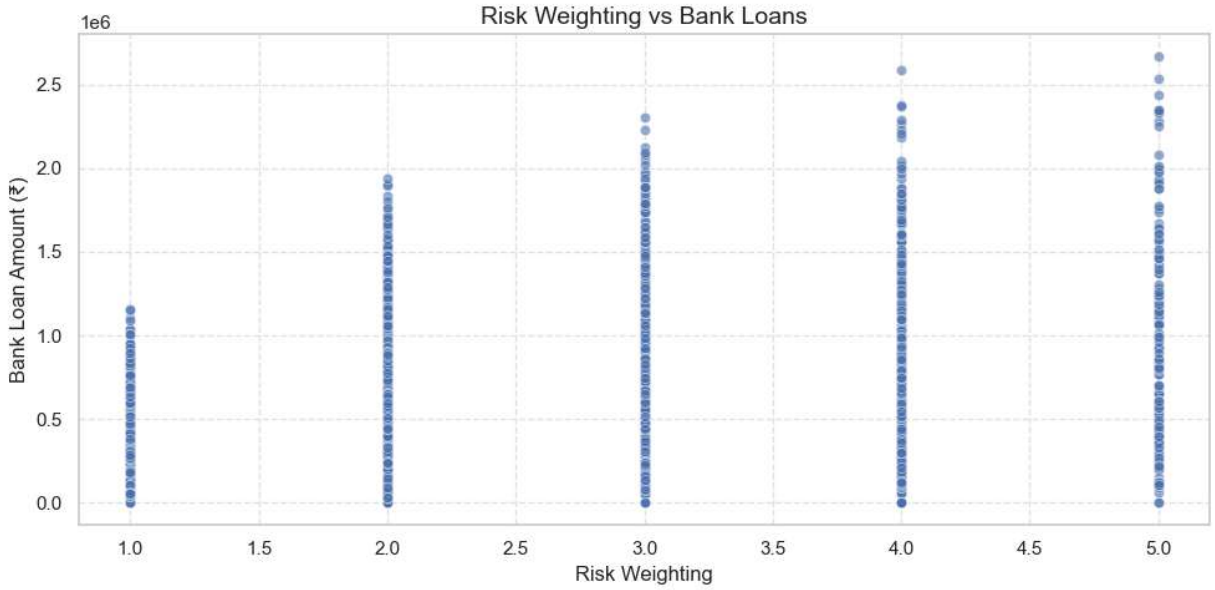
- The bank appears to have a **conservative customer base** — most clients fall under **low to moderate risk categories.**
- The **small population of high-risk customers** might warrant closer monitoring, more stringent approval criteria, or adjusted loan terms.
- Useful for **credit risk modeling**: this variable can be used as a **target** or as a **feature** in risk prediction models.

Risk Weighting vs Bank Loans

```
Risk Weighting is a score that helps banks quantify the risk of
lending money to a customer or investing in an asset. The higher
the score, the riskier the entity.
```

In [34]:
```
sns.scatterplot(data=df, x='Risk Weighting', y='Bank Loans', alpha=0.6)

plt.title("Risk Weighting vs Bank Loans", fontsize=14)
plt.xlabel("Risk Weighting", fontsize=12)
plt.ylabel("Bank Loan Amount (₹)", fontsize=12)
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```



Risk Weighting vs Bank Loans

This scatter plot illustrates the relationship between a customer's **Risk Weighting** and their **Bank Loan Amount**.

Observations

- There's **no strong linear relationship** between **Risk Weighting** and **Loan Amount**.
- Some customers with **high risk weighting** still received **large loans**, possibly due to:
  - High income
  - Long tenure with the bank
  - Existing strong customer-bank relationships
- Most **large loan approvals** are clustered around **low to moderate risk scores**, which aligns with expected banking behavior.

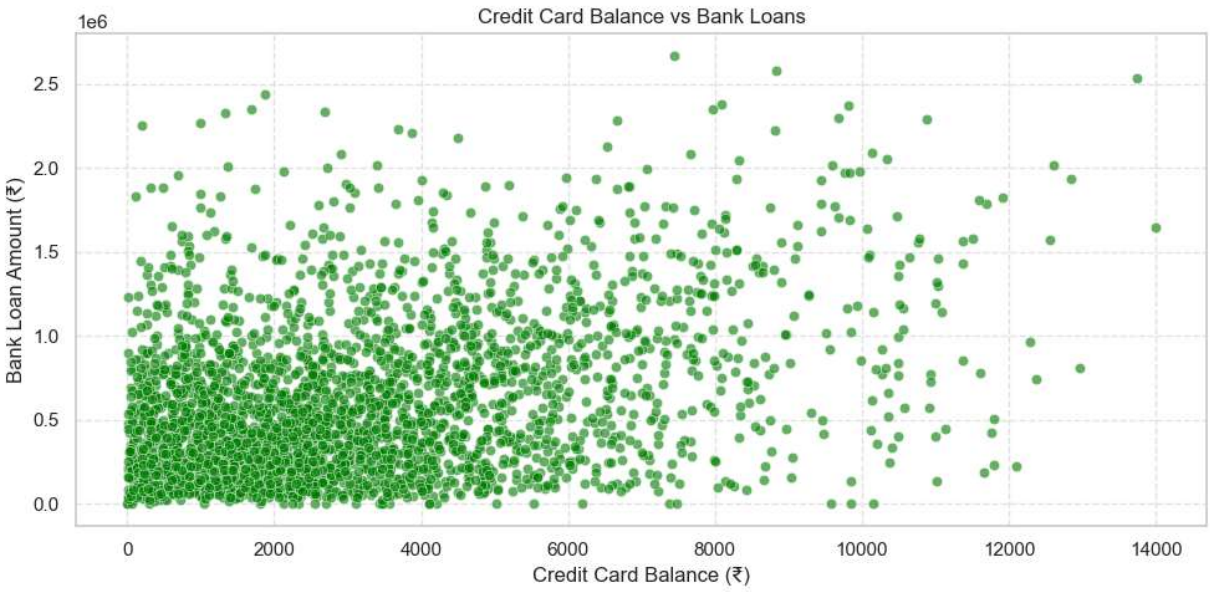
Conclusion

- In **real-world banking**, **risk weighting is a key factor** in deciding loan approvals — typically, a **higher risk score results in smaller or no loans**.
- However, in this dataset, the **relationship appears weak *(Some high-risk individuals still receive large loans)***, which may be due to:
  - Other influencing variables such as **income**, **property ownership**, or **deposits**
  - The bank's **historical lending strategy**
  - **Exceptional approvals** or **data quality issues** in the records

Credit Card Balance vs Bank Loans

In [35]:
```
sns.scatterplot(data=df, x='Credit Card Balance', y='Bank Loans', alpha=0.6, color=

plt.title("Credit Card Balance vs Bank Loans")
plt.xlabel("Credit Card Balance (₹)")
plt.ylabel("Bank Loan Amount (₹)")
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```



Credit Card Balance vs Bank Loans

This scatter plot explores the relationship between a customer's **credit card balance** and their **bank loan amount**.

Observations:

- The plot shows a **wide spread** of points with **no strong linear correlation** between the two variables.
- Some customers with **very high credit card balances** have **low or no bank loans**, and vice versa.
- The plot suggests that **credit card usage behavior is not a strong predictor** of loan amounts in this dataset.
- There are clusters of customers with:
  - Low credit card balances and low loans (likely low-engagement users).
  - Low credit card balances but high bank loans (possible installment loan users who avoid credit cards).
  - Few customers with both high credit card balances and high loan values — these may represent **higher-risk profiles**.

Interpretation:

- Since credit card balance and loan amount are **not tightly correlated**, they may represent **independent dimensions of credit usage**.
- This reinforces the need to **analyze multiple risk-related variables together** when assessing creditworthiness.

Risk Dimension

1. Loan Distribution by Income Band

Shows concentration of loans in certain income groups.

Risk: If too many loans are concentrated in low-income customers, default probability might be higher.

---

2. Loan Distribution by Nationality

Reveals geographic or demographic loan exposure.

Risk: Overexposure to a single group can lead to high losses if that segment faces economic trouble.

---

3. Risk Weighting

Even though the correlation was low with financial metrics, it highlights that non-financial factors (demographics, behavior) affect risk.

Risk: Ignoring these factors may lead to underestimating potential losses.

---

4. Credit Card Utilization Gaps

Many issued cards aren't being used actively.

Risk: Lower profitability and potential misuse or fraud exposure.

Loan Distribution by income band

In [36]:
```
def format_percent(x):
    return f"{x:.2f}%"
```

In [37]:
```
# Define income bands
bins = [0, 50000, 100000, 200000, float('inf')]
labels = ['Low', 'Medium', 'High', 'Very High']
df['Income_Band'] = pd.cut(df['Estimated Income'], bins=bins, labels=labels, right=

loan_by_income = df.groupby('Income_Band')['Bank Loans'].sum()
loan_by_income_pct = (loan_by_income / loan_by_income.sum() * 100).apply(format_per

print("Loan % by Income Band:\n", loan_by_income_pct)
```

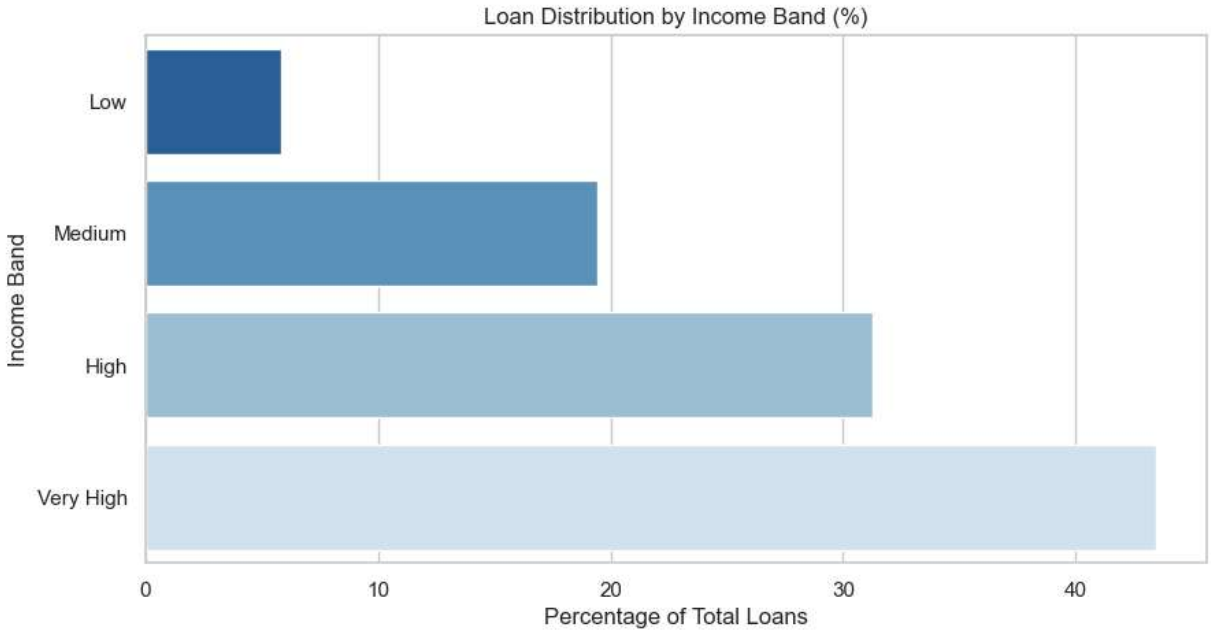
```
Loan % by Income Band:
 Income_Band
Low           5.81%
Medium       19.45%
High         31.28%
Very High    43.46%
Name: Bank Loans, dtype: object
```

Insights:

- **Very High income** group holds the largest share of loans (**43.46%**), suggesting significant borrowing capacity and possibly higher-value loans.
- **High income** band accounts for **31.28%**, indicating that the majority of loans are concentrated in the top two income groups.
- **Medium income** group contributes **19.45%** of loans.
- **Low income** group has the smallest share (**5.81%**), which could reflect limited borrowing eligibility or smaller loan sizes.

In [38]:
```
loan_by_income = df.groupby('Income_Band')['Bank Loans'].sum()
loan_by_income_pct = (loan_by_income / loan_by_income.sum() * 100).sort_values()

sns.barplot(x=loan_by_income_pct.values, y=loan_by_income_pct.index, palette="Blues
plt.title("Loan Distribution by Income Band (%)")
plt.xlabel("Percentage of Total Loans")
plt.ylabel("Income Band")
plt.show()
```



Loan Distribution by Nationality

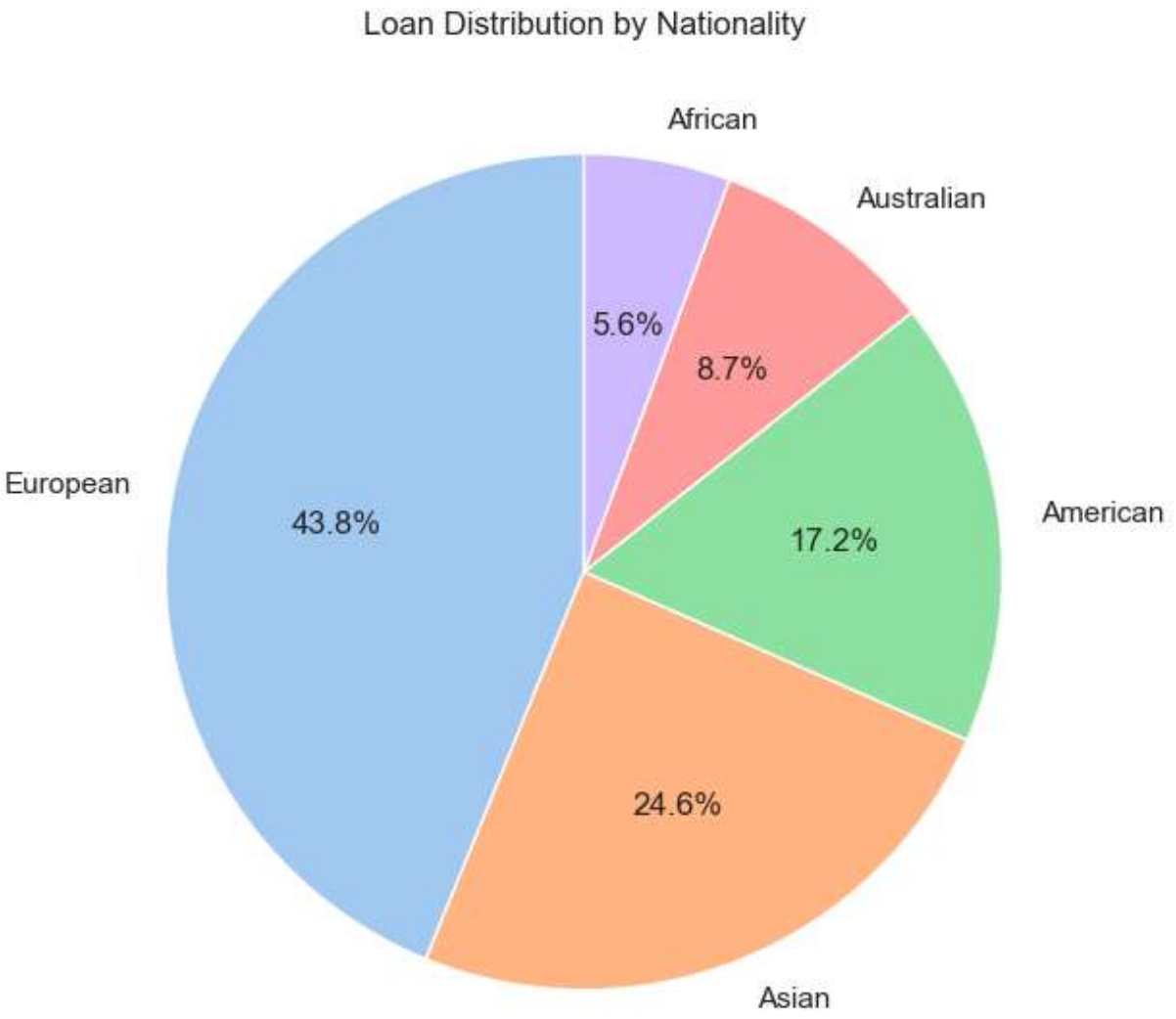
In [39]:
```
loan_by_nat = df.groupby('Nationality')['Bank Loans'].sum()
loan_by_nat_pct = (loan_by_nat / loan_by_nat.sum() * 100).apply(format_percent)
```

```
print("\nLoan % by Nationality:\n", loan_by_nat_pct)
```

```
Loan % by Nationality:
 Nationality
African         5.65%
American       17.25%
Asian          24.59%
Australian      8.69%
European       43.83%
Name: Bank Loans, dtype: object
```

In [40]:
```
loan_by_nat = df.groupby('Nationality')['Bank Loans'].sum()
loan_by_nat_pct = (loan_by_nat / loan_by_nat.sum() * 100).sort_values(ascending=Fal


plt.figure(figsize=(7, 8))
loan_by_nat_pct.plot(kind='pie', autopct='%1.1f%%', startangle=90, colors=sns.color
plt.title("Loan Distribution by Nationality")
plt.ylabel("")
plt.show()
```



Risk-Weighted Exposure

In [41]:
```
df['Risk_Weighted_Exposure'] = df['Bank Loans'] * (df['Risk Weighting'] / 100)
risk_weighted_by_cat = df.groupby(['Nationality', 'Occupation', 'Loyalty Classifica

print("\nRisk Weighted Exposure by Category:\n", risk_weighted_by_cat)
```

```
Risk Weighted Exposure by Category:
 Nationality  Occupation                  Loyalty Classification
African      Account Executive           Jade                      8335.5124
             Account Representative I    Gold                     11449.6022
                                         Jade                      3062.1424
             Account Representative II   Silver                   49068.8052
             Account Representative III  Silver                    2200.0052
                                                                       ...
European     Web Developer III           Silver                   67977.6120
             Web Developer IV            Gold                     29767.0200
                                         Jade                     16766.7115
                                         Platinum                  6205.9830
                                         Silver                  143670.0740
Name: Risk_Weighted_Exposure, Length: 1743, dtype: float64
```

Risk Weighted Exposure by Category

Nationality	Occupation	Loyalty Classification	Risk Weighted Exposure
African	Account Executive	Jade	8,335.51
African	Account Representative I	Gold	11,449.60
African	Account Representative I	Jade	3,062.14
African	Account Representative II	Silver	49,068.81
African	Account Representative III	Silver	2,200.01
...	...	...	...
European	Web Developer III	Silver	67,977.61
European	Web Developer IV	Gold	29,767.02
European	Web Developer IV	Jade	16,766.71
European	Web Developer IV	Platinum	6,205.98
European	Web Developer IV	Silver	143,670.07

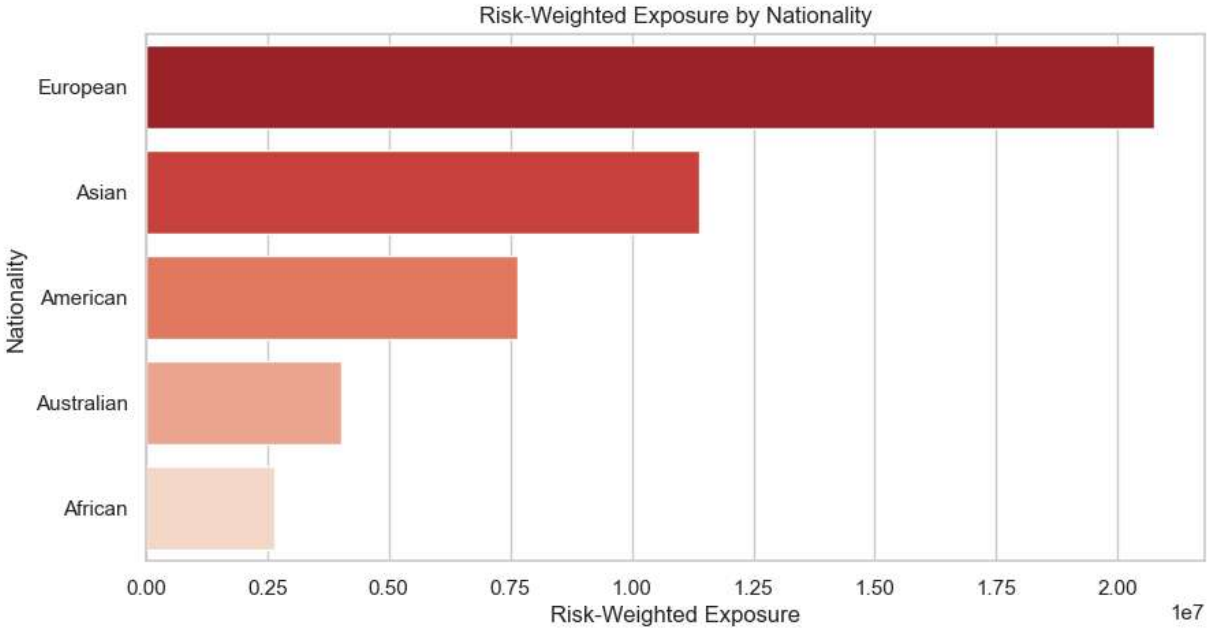
Insights:

- **High Exposure Roles:** Certain technical and senior positions like *Web Developer IV* (Silver classification) have particularly high exposure values.
- **Loyalty Classification Impact:** Silver classification appears frequently with large exposures, suggesting this segment represents significant portfolio risk.

- **Geographic Variation:** European roles, especially in tech, show some of the highest exposures compared to African roles in the sample.
- **Risk Concentration:** A small number of occupation–loyalty combinations hold a large share of total exposure, which could warrant closer monitoring.

In [42]:
```
df['Risk_Weighted_Exposure'] = df['Bank Loans'] * (df['Risk Weighting'] / 100)
risk_by_nat = df.groupby('Nationality')['Risk_Weighted_Exposure'].sum().sort_values

sns.barplot(x=risk_by_nat.head(10).values, y=risk_by_nat.head(10).index, palette="R
plt.title("Risk-Weighted Exposure by Nationality")
plt.xlabel("Risk-Weighted Exposure")
plt.ylabel("Nationality")
plt.show()
```



Why Non-Financial Factors Affect Risk Weighting

Money alone doesn't tell the whole story about risk.
Two people with the same income and savings can have very different risk levels because of **non-financial factors** such as:

- **Demographics** — Age, nationality, or location may affect job stability or economic opportunities.
- **Occupation** — Some jobs are more stable than others; for example, government employees often have steadier income than seasonal workers.
- **Behavior patterns** — Payment history, spending habits, or loyalty to the bank can show reliability (or the lack of it).

These factors can influence the likelihood of someone defaulting on a loan, even if their financial numbers look good on paper.

In short: Financial metrics tell you "*how much they have*", but non-financial factors tell you "*how likely they are to pay you back*".

Credit Card Utilization Gaps

In [43]:
```
# If credit limit column exists:
if 'Credit_Limit' in df.columns:
    df['Utilization_Rate'] = (df['Credit Card Balance'] / df['Credit_Limit']) * 100
    utilization_gap = 100 - df['Utilization_Rate']
else:
    df['Avg_Balance_per_Card'] = df['Credit Card Balance'] / df['Amount of Credit C


if 'Credit_Limit' in df.columns:
    print("\nCredit Card Utilization Gap:\n", utilization_gap.describe())
else:
    print("\nAverage Balance per Card:\n", df['Avg_Balance_per_Card'].describe())
```

```
Average Balance per Card:
 count     3000.000000
mean      2559.891708
std       2281.905475
min          1.170000
25%        854.877500
50%       1916.901667
75%       3592.447500
max      13749.830000
Name: Avg_Balance_per_Card, dtype: float64
```

In [44]:
```
df.columns
```

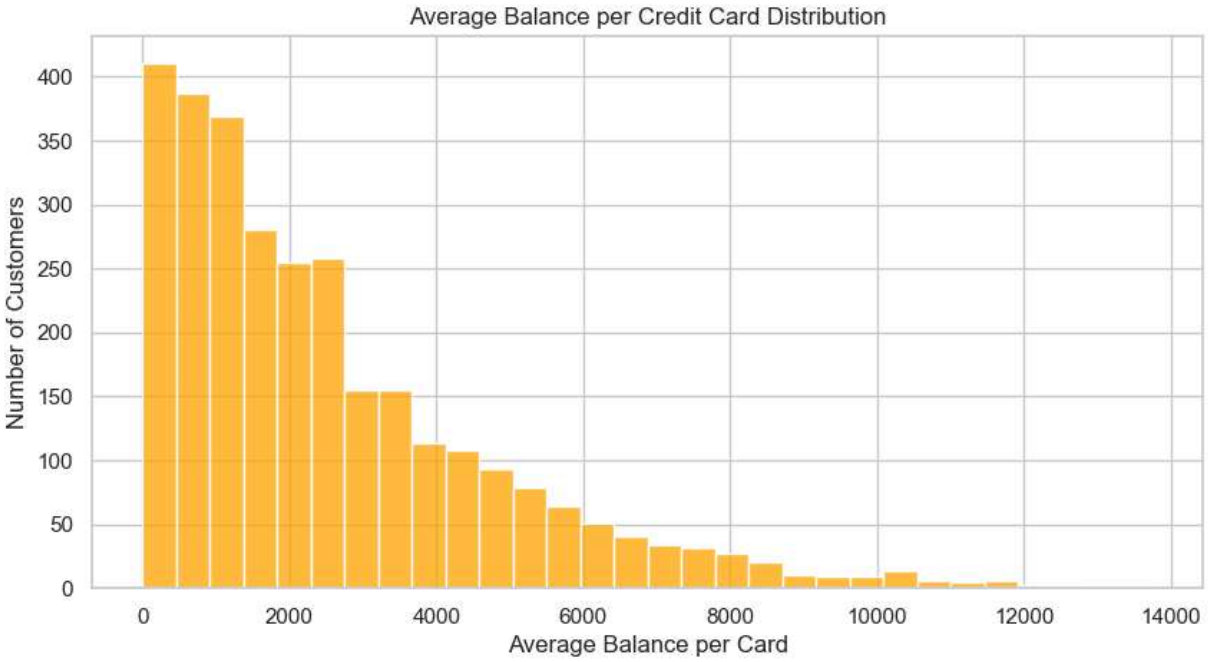
Out[44]:
```
Index(['Client ID', 'Name', 'Age', 'Location ID', 'Joined Bank',
       'Banking Contact', 'Nationality', 'Occupation', 'Fee Structure',
       'Loyalty Classification', 'Estimated Income', 'Superannuation Savings',
       'Amount of Credit Cards', 'Credit Card Balance', 'Bank Loans',
       'Bank Deposits', 'Checking Accounts', 'Saving Accounts',
       'Foreign Currency Account', 'Business Lending', 'Properties Owned',
       'Risk Weighting', 'BRId', 'GenderId', 'IAId', 'Join Year',
       'Bank Tenure (Years)', 'Income_Band', 'Risk_Weighted_Exposure',
       'Avg_Balance_per_Card'],
      dtype='object')
```

In [45]:
```
if 'Credit_Limit' in df.columns:
    df['Utilization_Rate'] = (df['Credit Card Balance'] / df['Credit_Limit']) * 100
    df['Utilization_Gap'] = 100 - df['Utilization_Rate']

    sns.histplot(df['Utilization_Gap'], bins=20, color="orange")
    plt.title("Credit Card Utilization Gap Distribution")
    plt.xlabel("Utilization Gap (%)")
    plt.ylabel("Number of Customers")
    plt.show()

else:
    df['Avg_Balance_per_Card'] = df['Credit Card Balance'] / df['Amount of Credit C
```

```
    sns.histplot(df['Avg_Balance_per_Card'], bins=30, color="orange")
    plt.title("Average Balance per Credit Card Distribution")
    plt.xlabel("Average Balance per Card")
    plt.ylabel("Number of Customers")
    plt.show()
```



Credit Card Utilization Analysis and Balance Analysis

The analysis examines customers' credit card usage based on the availability of the `Credit_Limit` column.

1. When `Credit_Limit` is available

- **Utilization Rate** = (`Credit Card Balance` / `Credit_Limit`) × 100
  Represents how much of their credit limit customers are using.
- **Utilization Gap** = 100 − `Utilization Rate`
  Indicates the remaining percentage of credit limit available for use.
- A histogram is plotted to show the **distribution of Utilization Gaps** across all customers.
- **Purpose:** Identify customers with low available credit (small gap) vs. customers with high available credit (large gap).

2. When `Credit_Limit` is not available

- **Average Balance per Card** = `Credit Card Balance` / `Amount of Credit Cards`
  Shows the mean outstanding balance per card.
- A histogram is plotted to show the **distribution of average balances per card.**
- **Purpose:** To understand spending levels per card across customers, which can help identify high-usage or low-usage cardholders

Insights that can be drawn:

- Customers with **low utilization gaps** may be close to maxing out their credit, indicating potential financial stress or risk.
- Customers with **high average balances per card** (when credit limits are unknown) may also carry higher default risk.
- The visualization helps in **risk assessment**, **customer segmentation**, and identifying **credit limit adjustment opportunities.**

Insights and Usage

- **Utilization Gap** analysis helps assess **credit risk**:
  - Lower gaps → Higher utilization → Potentially higher risk if repayment capacity is low.
  - Higher gaps → Lower utilization → Indicates available credit capacity.
- **Average Balance per Card** analysis helps in **customer segmentation** and **marketing strategies**:
  - High average balances might indicate premium customers.
  - Low balances may indicate under-utilization or dormant accounts.

Confidence Interval

In [46]:
```
import scipy.stats as st

data = df['Bank Loans'].dropna()

# Calculate statistics
mean_val = np.mean(data)
std_err = st.sem(data)
confidence = 0.95
ci_low, ci_high = st.t.interval(confidence, len(data)-1, loc=mean_val, scale=std_er
```

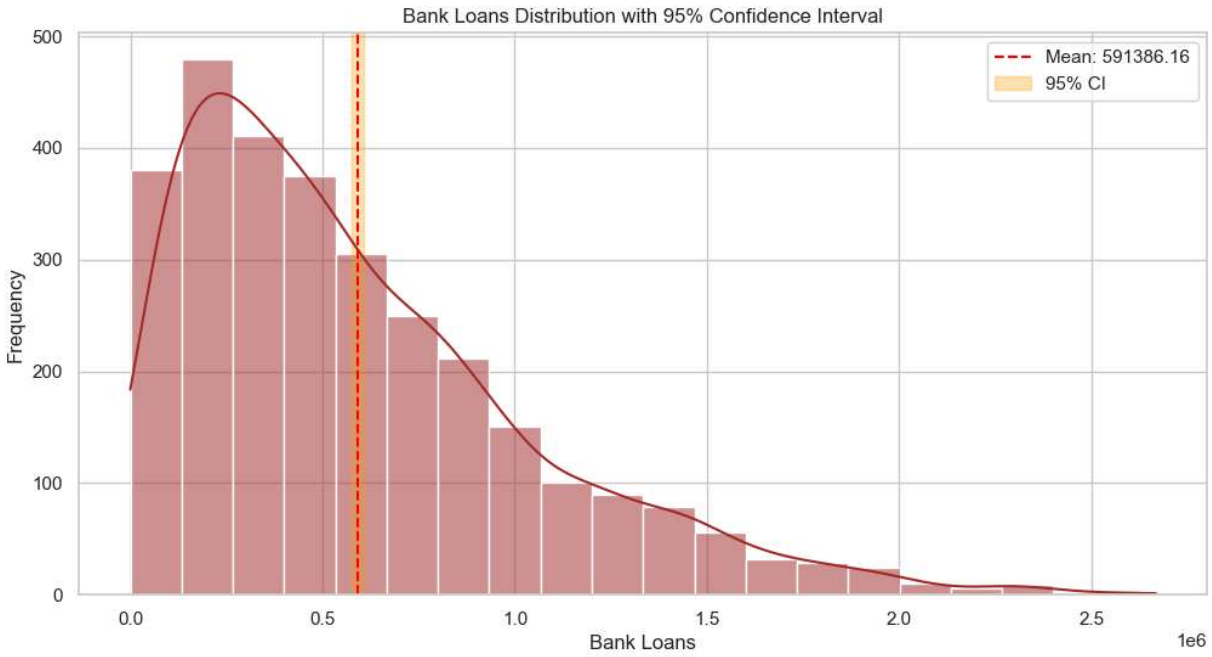
In [47]:
```
print(f"Mean: {mean_val:.2f}, CI: ({ci_low:.2f}, {ci_high:.2f})")
```

```
Mean: 591386.16, CI: (575006.38, 607765.93)
```

In [48]:
```
plt.figure(figsize=(12, 6))
sns.histplot(data, kde=True, color='Brown', bins=20)

# Add CI region
plt.axvline(mean_val, color='red', linestyle='--', label=f"Mean: {mean_val:.2f}")
plt.axvspan(ci_low, ci_high, color='orange', alpha=0.3, label=f"{int(confidence*100

plt.title("Bank Loans Distribution with 95% Confidence Interval")
plt.xlabel("Bank Loans")
plt.ylabel("Frequency")
plt.legend()
plt.show()
```



Connected with Me

Email Me

LinkedIn

GitHub

Made with ❤ by **Faisal Khan**

Powered by Jupyter Notebook

In [ ]: