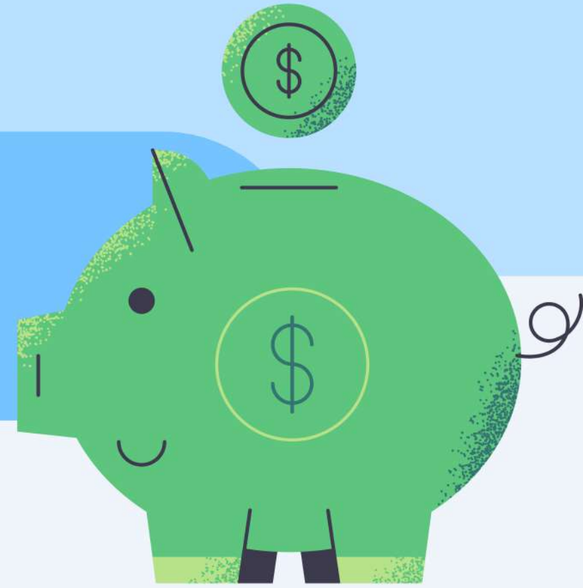
# Banking Customer Loan Analysis

Shubham Kumar | Data Analyst

# Objective



To analyze customer financial behavior and identify loan-eligible clients using data-driven KPIs while mitigating default risk.

# Dataset Overview

- Source: Banking.csv
- Key Columns: Client_ID, Estimated_Income, Bank_Deposits, Saving_Accounts, Bank_Loans, Credit_Card_Balance, Business_Lending, etc.
- Rows: 3000
- Brief on EDA approach (correlation insights)

# Formatting Joined_Bank Column

```sql
ALTER TABLE `bankloandb`.`banking` ADD COLUMN `Joined_Bank_Date` DATE;

UPDATE `bankloandb`.`banking`
SET `Joined_Bank_Date` = STR_TO_DATE(`Joined_Bank`, '%d-%m-%Y');

-- Drop the old column
ALTER TABLE `bankloandb`.`banking`
DROP COLUMN `Joined_Bank`;

-- Rename the new column
ALTER TABLE `bankloandb`.`banking`
CHANGE COLUMN `Joined_Bank_Date` `Joined_Bank` DATE;

SELECT Joined_Bank FROM banking
```

Before formatting

| Joined Bank |
|---|
| 6/5/2019 |
| 10/12/2001 |
| 25-01-2010 |
| 28-03-2019 |
| 20-07-2012 |
| 7/2/2019 |
| 2/6/2002 |
| 3/11/2000 |
| 7/4/2015 |
| 20-11-1995 |
| 18-10-2014 |
| 12/9/2005 |

After formatting

| Joined_Bank |
|---|
| 2019-05-06 |
| 2001-12-10 |
| 2010-01-25 |
| 2019-03-28 |
| 2012-07-20 |
| 2019-02-07 |
| 2002-06-02 |
| 2000-11-03 |
| 2015-04-07 |
| 1995-11-20 |
| 2014-10-18 |

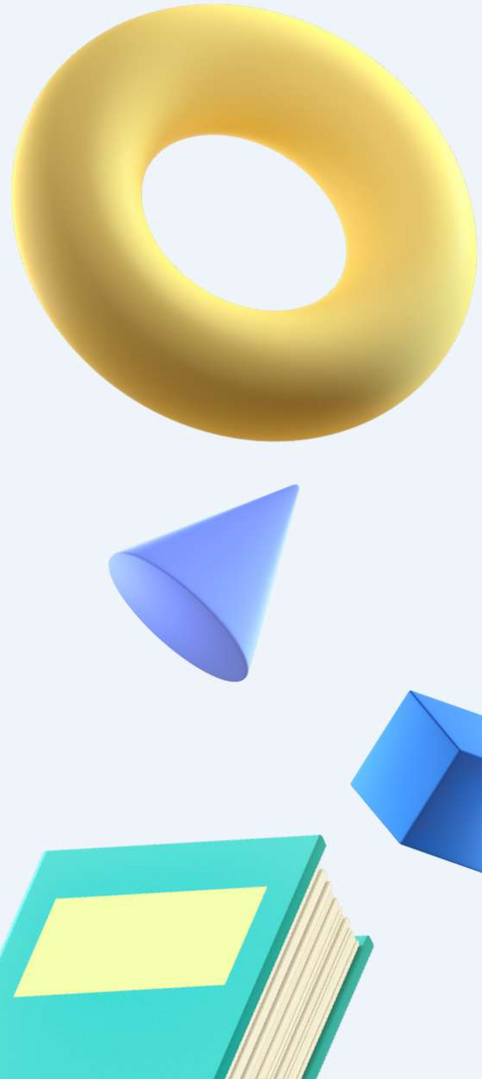
# Engagement_Timeframe of the clients in banks

```sql
ALTER TABLE `bankloandb`.`banking`
ADD COLUMN `Engagement_Timeframe` VARCHAR(20);

UPDATE `bankloandb`.`banking`
SET `Engagement_Timeframe` = CASE
    WHEN TIMESTAMPDIFF(YEAR, `Joined_Bank`, CURDATE()) < 1 THEN '<1 years'
    WHEN TIMESTAMPDIFF(YEAR, `Joined_Bank`, CURDATE()) < 5 THEN '<5 years'
    WHEN TIMESTAMPDIFF(YEAR, `Joined_Bank`, CURDATE()) < 10 THEN '<10 years'
    WHEN TIMESTAMPDIFF(YEAR, `Joined_Bank`, CURDATE()) < 20 THEN '<20 years'
    ELSE '>=20 years'
END;

SELECT `Client_ID`,`Name`,`Engagement_Timeframe` FROM banking
```

| Client_ID | Name | Engagement_Timeframe |
|---|---|---|
| IND81288 | Raymond Mills | <10 years |
| IND65833 | Julia Spencer | >=20 years |
| IND47499 | Stephen Murray | <20 years |
| IND72498 | Virginia Garza | <10 years |
| IND60181 | Melissa Sanders | <20 years |
| IND78532 | Samuel Hudson | <10 years |



# Creating bins Income < 100000 as low, <300000 as Mid and else High as Income_Band

```sql
ALTER TABLE bankloandb.banking
ADD COLUMN Income_Band VARCHAR(20);

UPDATE bankloandb.banking
SET Income_Band = CASE
    WHEN Estimated_Income < 100000 THEN 'Low'
    WHEN Estimated_Income < 300000 THEN 'Mid'
    WHEN Estimated_Income IS NULL THEN 'Unknown'
    ELSE 'High'
END;

SELECT Client_ID, Name, Estimated_Income, Income_Band FROM banking;
```

| Client_ID | Name | Estimated_Income | Income_Band |
|---|---|---|---|
| IND81288 | Raymond Mills | 75384.77 | Low |
| IND65833 | Julia Spencer | 289834.31 | Mid |
| IND47499 | Stephen Murray | 169935.23 | Mid |
| IND72498 | Virginia Garza | 356808.11 | High |
| IND60181 | Melissa Sanders | 130711.68 | Mid |
| IND78532 | Samuel Hudson | 118326.96 | Mid |

# Total Deposits per Customer

```sql
SELECT
    Client_ID,
    IFNULL(Bank_Deposits, 0) + IFNULL(Checking_Accounts, 0) + IFNULL(Saving_Accounts, 0) + IFNULL(Foreign_Currency_Account, 0) AS Total_Deposits
FROM banking;
```

Business Value: **Measures customer liquidity and usage of banking services**

| Client_ID | Total_Deposits |
|---|---|
| IND81288 | 2709028.94 |
| IND65833 | 1276801.6300000001 |
| IND47499 | 1968202.41 |
| IND72498 | 2388513.65 |
| IND60181 | 1092790.3699999999 |
| IND78532 | 2306822.7 |
| IND95683 | 129473.79 |
| IND40785 | 309665.83 |
| IND13570 | 1877335.24 |
| IND53299 | 592691.61 |
| IND76263 | 120059.63999999998 |
| IND56452 | 1490329.74 |
| IND28766 | 969683.15 |
| IND17897 | 1309483.8900000001 |
| IND86325 | 300650.46 |

# Savings-to-Income Ratio

```sql
SELECT
    Client_ID,
    ROUND(IFNULL(Saving_Accounts, 0) / Estimated_Income, 2) AS Savings_To_Income_Ratio
FROM Banking
WHERE Estimated_Income > 0;
```

Business Value: **Indicates saving behavior relative to income.**

| Client_ID | Savings_To_Income_Ratio |
|---|---|
| IND81288 | 8.06 |
| IND65833 | 1.19 |
| IND47499 | 1.19 |
| IND72498 | 0.66 |
| IND60181 | 0.98 |
| IND78532 | 2.01 |
| IND95683 | 0.43 |
| IND40785 | 0.72 |
| IND13570 | 3.18 |
| IND53299 | 1.77 |
| IND76263 | 0.7 |
| IND56452 | 4.61 |

# Loan Exposure (Personal + Business)

```sql
SELECT
    Client_ID,
    ROUND(IFNULL(Bank_Loans, 0) + IFNULL(Business_Lending, 0), 2) AS Total_Loan_Exposure
FROM Banking;
```

Business Value: **Captures total liability exposure..**

| Client_ID | Total_Loan_Exposure |
|---|---|
| IND81288 | 1910718.22 |
| IND65833 | 3271141.53 |
| IND47499 | 1600853.42 |
| IND72498 | 1269597.35 |
| IND60181 | 2722714.07 |
| IND78532 | 2157933.56 |
| IND95683 | 363021.31 |
| IND40785 | 2312161.48 |
| IND13570 | 1268004.74 |
| IND53299 | 968611.84 |
| IND76263 | 567447.75 |

# Deposit Diversification Score

```sql
SELECT Client_ID,
        CASE
          WHEN Checking_Accounts > 0 THEN 1 ELSE 0
        END +
        CASE
          WHEN Saving_Accounts > 0 THEN 1 ELSE 0
        END +
        CASE
          WHEN Foreign_Currency_Account > 0 THEN 1 ELSE 0
        END AS Deposit_Diversification_Score
FROM Banking;
```

| Client_ID | Deposit_Diversification_Score |
|---|---|
| IND27269 | 3 |
| IND97061 | 3 |
| IND94872 | 3 |
| IND86593 | 1 |
| IND84908 | 3 |
| IND32885 | 3 |
| IND70495 | 3 |
| IND45220 | 3 |

Business Value: **Reflects how spread out a customer's deposits are (across checking, saving, foreign accounts).**

# High Balance Customers (Wealthy Segment)

```sql
SELECT Client_ID
FROM Banking
WHERE Estimated_Income > 100000
  AND (Bank_Deposits + Checking_Accounts + Saving_Accounts + Foreign_Currency_Account) > 50000;
```

| Client_ID |
|---|
| IND65833 |
| IND47499 |
| IND72498 |
| IND60181 |
| IND78532 |
| IND56452 |
| IND28766 |
| IND17897 |
| IND86325 |
| IND28503 |

Business Value: **Identify top-tier customers for targeted marketing or premium services.**

# Credit Risk Score (based on Credit Card Balance and Loans)

```sql
SELECT
    Client_ID,
    IFNULL(Credit_Card_Balance, 0) + IFNULL(Bank_Loans, 0) AS Credit_Risk_Score
FROM Banking
ORDER BY Credit_Risk_Score DESC
```

| Client_ID | Credit_Risk_Score |
|---|---|
| IND23172 | 2674996.02 |
| IND22626 | 2593144.79 |
| IND99978 | 2552479.96 |
| IND99607 | 2441035.19 |
| IND70050 | 2389233.9699999997 |
| IND19422 | 2384128.85 |
| IND31640 | 2356570.0100000002 |
| IND70565 | 2350841.4099999997 |
| IND52376 | 2336178.86 |
| IND63638 | 2333134.2600000002 |
| IND40901 | 2312258.43 |
| IND51890 | 2301411.78 |

Business Value: **Helps assess financial risk and creditworthiness.**

# Superannuation Coverage Ratio

```sql
SELECT Client_ID,
       ROUND(Superannuation_Savings / Estimated_Income, 4) AS Superannuation_Coverage
FROM Banking
WHERE Estimated_Income > 0;
```

| Client_ID | Superannuation_Coverage |
|---|---|
| IND81288 | 0.2345 |
| IND65833 | 0.06 |
| IND47499 | 0.252 |
| IND72498 | 0.0153 |
| IND60181 | 0.3678 |
| IND78532 | 0.0509 |
| IND95683 | 0.2429 |
| IND40785 | 0.4772 |
| IND13570 | 0.1068 |
| IND53299 | 0.4186 |
| IND76263 | 0.2258 |
| IND56452 | 0.1568 |

Business Value: **Tracks retirement savings sufficiency relative to income.**

# Cross-Product Holding Score

```sql
SELECT Client_ID,
       (CASE WHEN Credit_Card_Balance > 0 THEN 1 ELSE 0 END +
        CASE WHEN Bank_Loans > 0 THEN 1 ELSE 0 END +
        CASE WHEN Saving_Accounts > 0 THEN 1 ELSE 0 END +
        CASE WHEN Checking_Accounts > 0 THEN 1 ELSE 0 END +
        CASE WHEN Business_Lending > 0 THEN 1 ELSE 0 END) AS Product_Holding_Score
FROM Banking;
```

| Client_ID | Product_Holding_Score |
|---|---|
| IND19011 | 5 |
| IND86703 | 5 |
| IND23640 | 5 |
| IND93407 | 3 |
| IND40136 | 5 |
| IND76215 | 5 |
| IND93216 | 5 |
| IND54572 | 5 |
| IND18649 | 5 |
| IND51790 | 5 |
| IND17740 | 5 |

Business Value: **Measures how many different product types a customer uses (helps with customer engagement analysis).**

# Customer Segmentation Rules

| Client_ID | Segment |
|---|---|
| IND81288 | General Customer |
| IND65833 | Affluent Saver |
| IND47499 | Affluent Saver |
| IND72498 | Affluent Saver |
| IND60181 | Affluent Saver |
| IND78532 | Affluent Saver |
| IND95683 | General Customer |
| IND40785 | General Customer |
| IND13570 | General Customer |
| IND53299 | General Customer |
| IND76263 | General Customer |
| IND56452 | Affluent Saver |

```sql
SELECT Client_ID,
       CASE
         WHEN Estimated_Income > 100000
              AND (Bank_Deposits + Checking_Accounts + Saving_Accounts + Foreign_Currency_Account) > 50000
              THEN 'Affluent Saver'
         WHEN (Bank_Loans + Credit_Card_Balance) > 50000 AND Saving_Accounts < 5000
              THEN 'High-Risk Borrower'
         WHEN Business_Lending > 0
              AND Bank_Deposits = 0
              AND Checking_Accounts = 0
              AND Saving_Accounts = 0
              AND Bank_Loans = 0
              AND Credit_Card_Balance = 0
              THEN 'Commercial Client'
         ELSE 'General Customer'
       END AS Segment
FROM Banking;
```

🏦 **Affluent Saver**

Rule: High Estimated Income AND High Total Deposits
(e.g., EstimatedIncome > 100,000 and TotalDeposits > 50,000)

⚠️ **High-Risk Borrower**

Rule: High Loan Exposure AND Low Savings
(e.g., BankLoans + CreditCardBalance > 50,000 and SavingAccounts < 5,000)

📄 **Commercial Client**

Rule: Has Business Lending, but No Personal Financial Products
(like savings/checking/deposits)

# Debt-to-Income Ratio (DTI)

```sql
SELECT Client_ID,
       ROUND((Bank_Loans + Credit_Card_Balance + Business_Lending) / Estimated_Income, 3) AS Debt_To_Income_Ratio
FROM Banking
WHERE Estimated_Income > 0
ORDER BY Debt_To_Income_Ratio ASC
```

| Client_ID | Debt_To_Income_Ratio |
|---|---|
| IND57908 | 0.391 |
| IND87020 | 0.524 |
| IND74305 | 0.601 |
| IND90745 | 0.607 |
| IND69834 | 0.724 |
| IND26629 | 0.828 |
| IND34216 | 0.854 |
| IND38237 | 0.898 |
| IND93832 | 0.907 |
| IND30559 | 0.999 |
| IND71913 | 1.004 |
| IND28522 | 1.032 |

Business Value: **Indicates the loan burden vs. income. A lower DTI suggests the customer has capacity to take on more debt.**

Use case: **Offer loans only to customers with DTI < 0.4 (or 40%).**

# Loan Repayment Capacity Score

```sql
SELECT Client_ID,
       ROUND((Estimated_Income + Saving_Accounts + Bank_Deposits) / (Bank_Loans + Credit_Card_Balance + Business_Lending), 2) AS Repayment_Capacity_Score
FROM Banking
WHERE (Bank_Loans + Credit_Card_Balance + Business_Lending)>0
order by Repayment_Capacity_Score DESC;
```

| Client_ID | Repayment_Capacity_Score |
|---|---|
| IND47000 | 14.69 |
| IND73620 | 11.33 |
| IND57908 | 10.71 |
| IND90745 | 10.39 |
| IND95041 | 9.03 |
| IND74305 | 8.88 |
| IND21371 | 8.2 |
| IND82767 | 7.67 |
| IND59444 | 7.16 |
| IND94106 | 7.09 |
| IND86149 | 7.06 |
| IND12762 | 6.91 |
| IND96454 | 6.49 |
| IND97484 | 6.29 |
| IND29427 | 6.02 |

Business Value: **Shows the customer's financial buffer vs. their liabilities.**

# Loan Default Risk Indicator

```sql
SELECT Client_ID,
       CASE
         WHEN Bank_Loans > 50000 AND Estimated_Income < 30000 AND Saving_Accounts < 5000
         THEN 'High Risk'
         ELSE 'Low/Medium Risk'
       END AS Default_Risk_Flag
FROM Banking;
```

| Client_ID | Default_Risk_Flag |
|---|---|
| IND81288 | Low/Medium Risk |
| IND65833 | Low/Medium Risk |
| IND47499 | Low/Medium Risk |
| IND72498 | Low/Medium Risk |
| IND60181 | Low/Medium Risk |
| IND78532 | Low/Medium Risk |
| IND95683 | Low/Medium Risk |
| IND40785 | Low/Medium Risk |
| IND13570 | Low/Medium Risk |
| IND53299 | Low/Medium Risk |
| IND76263 | Low/Medium Risk |
| IND56452 | Low/Medium Risk |
| IND28766 | Low/Medium Risk |

Business Value: **Create a flag based on high loan + low income + low savings, to identify risky borrowers.**

Rule-Based Example:

- BankLoans > 50,000
- EstimatedIncome < 30,000
- Savings < 5,000

# Customer Liquidity Score

```sql
SELECT Client_ID,
       ROUND((Bank_Deposits + Checking_Accounts + Saving_Accounts) / (Bank_Loans + Credit_Card_Balance),2) AS Liquidity_Score
FROM Banking
WHERE (Bank_Loans + Credit_Card_Balance)>0
ORDER BY Liquidity_Score DESC
```

| Client_ID | Liquidity_Score |
|---|---|
| IND33193 | 860783.79 |
| IND60970 | 84266.68 |
| IND90914 | 14629.09 |
| IND65488 | 8937.44 |
| IND32633 | 647.31 |
| IND73620 | 582.94 |
| IND63102 | 565.99 |
| IND80929 | 480.67 |
| IND79637 | 434.54 |
| IND47000 | 416.85 |
| IND76462 | 383.56 |
| IND74531 | 377.16 |
| IND76818 | 349.87 |
| IND26855 | 302.68 |

Business Value: **High liquidity means the customer can pay off liabilities quickly if needed.**

# Credit Utilization Ratio

```sql
SELECT Client_ID,
       ROUND(Credit_Card_Balance / (Bank_Deposits + Checking_Accounts + Saving_Accounts), 2) AS Credit_Utilization
FROM Banking
ORDER BY Credit_Utilization DESC;
```

| Client_ID | Credit_Utilization |
|---|---|
| IND68994 | 0.07 |
| IND20002 | 0.06 |
| IND83050 | 0.06 |
| IND44048 | 0.06 |
| IND21547 | 0.06 |
| IND70450 | 0.05 |
| IND86703 | 0.05 |
| IND62132 | 0.05 |
| IND26575 | 0.05 |
| IND51423 | 0.05 |
| IND23568 | 0.05 |
| IND83694 | 0.05 |
| IND70183 | 0.05 |
| IND16777 | 0.05 |

Business Value: **High ratio = higher risk of maxing out available funds.**

# Product Penetration Score

```sql
SELECT Client_ID,
       (CASE WHEN Bank_Loans > 0 THEN 1 ELSE 0 END +
        CASE WHEN Credit_Card_Balance > 0 THEN 1 ELSE 0 END +
        CASE WHEN Saving_Accounts > 0 THEN 1 ELSE 0 END +
        CASE WHEN Checking_Accounts > 0 THEN 1 ELSE 0 END +
        CASE WHEN Business_Lending > 0 THEN 1 ELSE 0 END) AS Product_Penetration_Score
FROM Banking;
```

| Client_ID | Product_Penetration_Score |
|---|---|
| IND96454 | 5 |
| IND14292 | 5 |
| IND67206 | 5 |
| IND90177 | 5 |
| IND60970 | 4 |
| IND24003 | 5 |
| IND87800 | 5 |
| IND75987 | 5 |
| IND70083 | 5 |
| IND31444 | 5 |
| IND70450 | 5 |
| IND47520 | 5 |
| IND69076 | 5 |

Business Value: **Measure how many financial products (accounts, loans, etc.) a customer uses to cross-sell and upsell effectively.**

# Product Penetration Score

```sql
SELECT
    CAST(SUM(`Bank_Loans` + `Business_Lending` + `Credit_Card_Balance`) / 1000000 AS DECIMAL(10,2)) AS Total_Loan_Millions
FROM banking;
```

| Total_Loan_Millions |
| --- |
| 4383.97 |

= $4383.97 M

# Recommendations

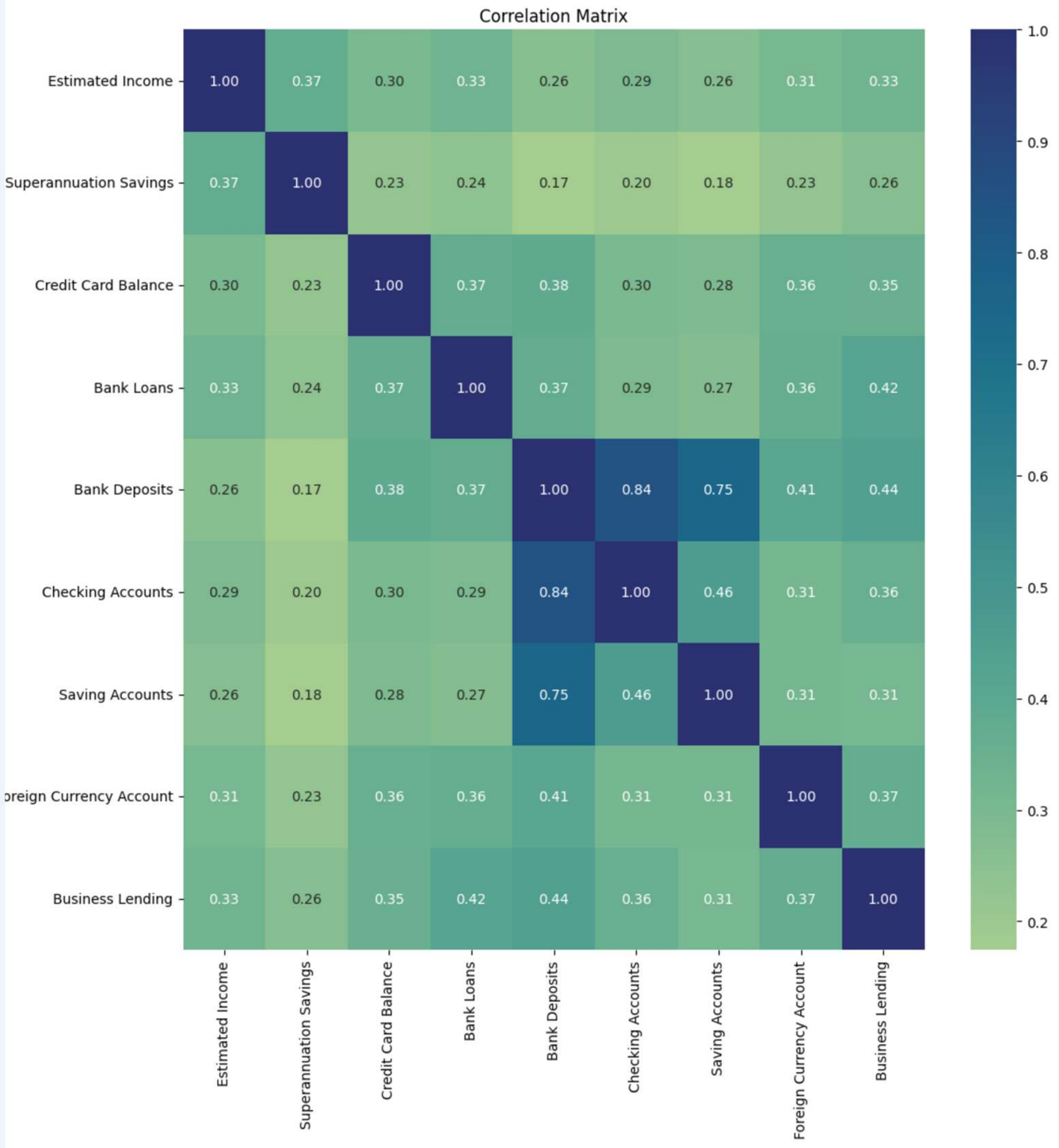
- Focus loan marketing on high eligibility scores
- Avoid issuing loans where DTI > 0.4 or Liquidity < 1
- Promote savings behavior to increase loan readiness


# Tools & Skills Used

- Python (Pandas, EDA)
- SQL (KPI extraction)
- Power BI / Excel for visualization (optional)
- Data Cleaning, Segmentation, KPI Modeling


# Insights of EDA:

**Interest** can be defined as as the amount of money paid when borrowing another person's money. It can also be the amount you are paid when you lend money.



# Thank you!