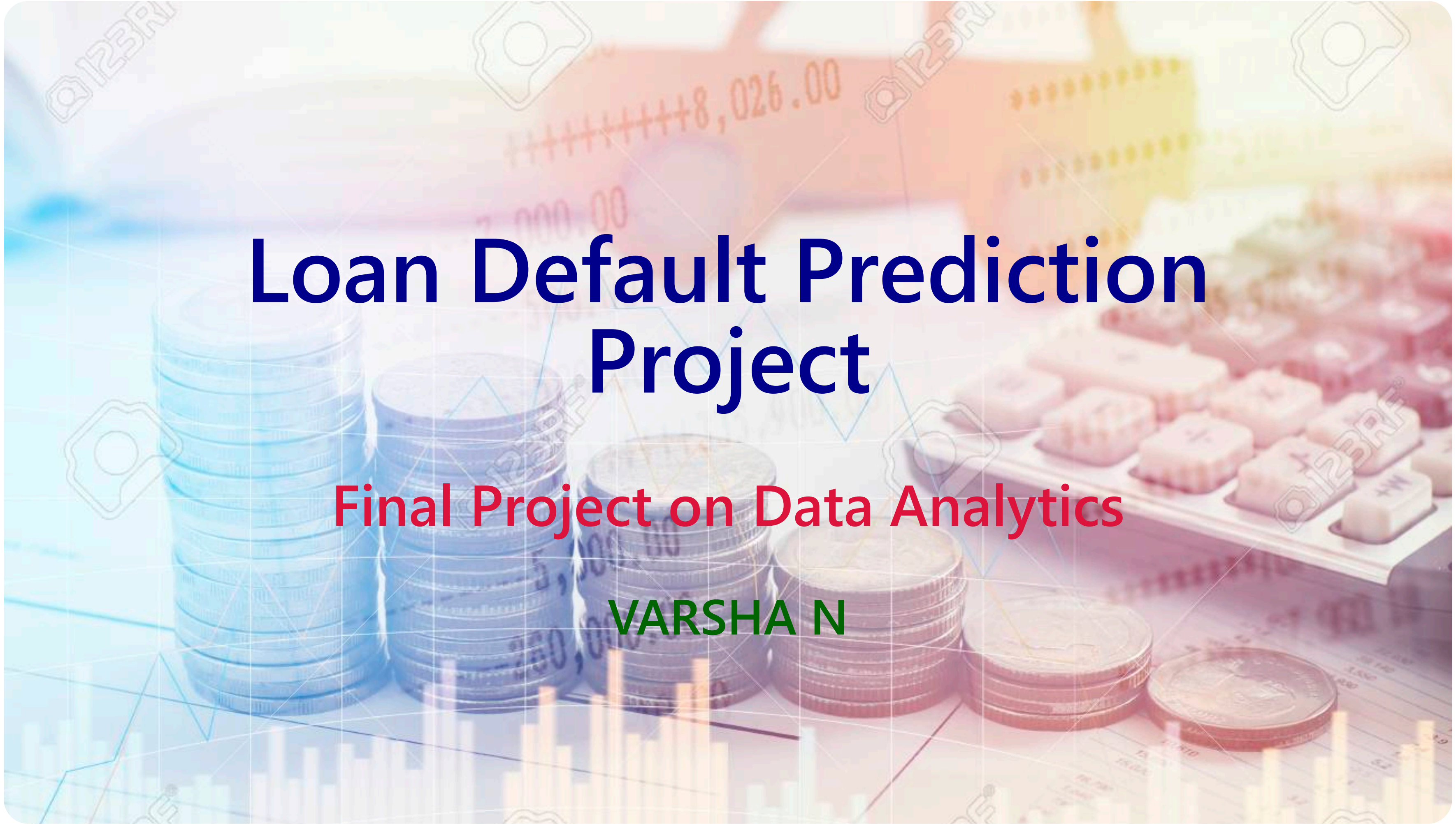


Introduction

Loans are a major part of the banking system. However, many applicants default on their loans, causing financial losses for banks. Predicting loan default in advance can help financial institutions minimize risk and improve decision-making. With the help of data analytics, we can analyze borrower characteristics to predict the probability of loan default.

🔷 Aim

The aim of this project is to analyze loan applicant data and develop a predictive model that can determine the likelihood of a borrower defaulting on a loan.

- 📊 Understand borrower characteristics such as income, age, credit score, loan amount, and employment details.
- 🔍 Identify patterns and key factors that influence repayment or default.
- 🤖 Build a predictive model for loan default.
- 📈 Support banks in better lending decisions and risk reduction.

🔷 Dataset Description

- Source -> Kaagle
- Rows Count -> 322375
- Column Count -> 18

🔷 Column-wise Description

1. LoanID

- **Type:** Identifier(String)
- **Meaning:** A unique ID assigned to each loan record
- **Use in Analysis:**
  - Not used in prediction(it's just an identifier).

2. Age

- **Type:** Integer
- **Meaning:** Age of the Borrower.

- **Importance:**
  - Younger borrowers may have lower credit history and higher default risk.
  - Middle-aged borrowers may be more stable financially.
  - Very old borrowers may have limited income sources.

3. Income

- **Type:** Integer
- **Meaning:** Annual Income of the Borrower.
- **Importance:**
  - Higher Income usually means lower default risk(borrower can repay).
  - Very low income as a higher chance of dafault ,if loan amount is high.

4. LoanAmount

- **Type:** Integer
- **Meaning:** Amount of money borrowed.
- **Importance:**
  - High loan amount compared to income may increase risk.
  - Used along with income to calculate affordability ratios.

5. CreditScore

- **Type:** Integer
- **Meaning:**Credit score of the borrower(measure of credit worthiness).
- **Importance:**
  - Higher score,borrower is financially reliable.
  - Low score,higher chance of default.

6. MonthsEmployed

- **Type:** Integer
- **Meaning:** Number of months borrower has been employed.
- **Importance:**

- If the borrower has a long term employment then job is stable and has less risks.
- Unstable jobs or short term employment then default risk is higher.

7. NumCreditLines

- **Type:** Integer
- **Meaning:**Number of credit lines borrower has opened(credit cards,loans) .
- **Importance:**
  - If someone has too many credit lines,it means they are borrowing too much, higher chance of default.
  - If someone has very few or none,the bank cannot judge their repayment history well and it's harder to trust.

8. InterestRate

- **Type:** Float
- **Meaning:** Interest rate charged on loan.
- **Importance:**
  - Higher interest rates,high repayment burden.
  - Loans with high rates are riskier and often given to high-risk borrowers.

9. LoanTerm

- **Type:** Integer
- **Meaning:** Duration of the loan in months.
- **Importance:**
  - Short term loans,higher monthly payments but less overall risks.
  - Long term loans,lesser monthly payments but higher long term risks.

10. DTIRatio(Debt-to-Income Ratio)

- **Type:** Float
- **Meaning:** Ratio of borrower's debt to income.
- **Importance:**
  - Low DTI,Person has enough free income left and it's safer for banks.
  - High DTI,most income goes to paying existing debts and riskier to give loans.

- Moreover,DTI tells if the borrower can handle another loan or not.

11. Education

- **Type:** String(phD,Master's,Bachelor's,High School)
- **Meaning:** Highest level education of borrower.
- **Importance:**
  - Higher education often correlates with higher income potential.
  - Low education may limit future earning capacity.

12. EmploymentType

- **Type:** String ((Full-time,Part-time,Self-Employed,Unemployed)
- **Meaning:** The borrower's current employment status.
- **Importance:**
  - Borrower's with stable jobs(full time),usually have lower default risk.
  - While part-time,self-employed, unemployed applicants are riskier.

13. MaritalStatus

- **Type:** String(Single,Married,Divorced)
- **Meaning:** Borrower's marital status.
- **Importance:**
  - Married borrowers may have shared finacial responsibilities(sometimes more stable).
  - Single borrowers may have fewer dependents.
  - Divorced borrowers may face financial stress.

14. HasMortgage

- **Type:** String(Yes/No)
- **Meaning:** Whether borrower already has a mortgage.
- **Importance:**
  - If there is any existing mortgage, then it will be a larger debt and risk increases.
  - If nothing is there then lower debt burden.

15. HasDependents

- **Type:** String(Yes/No)
- **Meaning:** Whether borrower already has dependents(children/others to support).
- **Importance:**
    - Dependents increase expenses,repayment becomes harder.
    - If no dependents,borrower has more disposable income.

16. HasCoSigner

- **Type:** String(Yes/No)
- **Meaning:** Whether loan has a co-signer(another person guaranteeing repayment).
- **Importance:**
    - Co-signer reduces risk,it's their legal responsibility.
    - If no Co-signer,lender takes more risk.

17. LoanPurpose

- **Type:** String
- **Meaning:** The reason why the borrower has taken the loan.
- **Importance:**
    - Business loans can be riskier than home, education and other loan types.

18. Default

- **Type:** Integer(0/1)
- **Meaning:**
    - 0=Loan was repaid successfully(No Default)
    - 1= Borrower failed to repay(Default)
- **Importance:**
    - It is the target variable that indicates whether the borrower repaid(0) or defaulted(1)

🔹 Project Scope

1. Data Loading and Intial Overview

➡️ Import the dataset using Pandas and provide an Overview

In [5]:
```
# Importing libraries
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv("C:\python1\Loan_default_uncleaned_2.csv")
```

-> Number of rows and columns

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

Out[6]:
```
(322375, 18)
```

In [7]:
```
# To get only Rows
df.shape[0]
```

Out[7]:
```
322375
```

In [8]:
```
# To get only Columns
df.shape[1]
```

Out[8]:
```
18
```

In [9]:
```
# To get Column Names
df.columns
```

Out[9]:
```
Index(['LoanID', 'Age', 'Income', 'LoanAmount', 'CreditScore',
       'MonthsEmployed', 'NumCreditLines', 'InterestRate', 'LoanTerm',
       'DTIRatio', 'Education', 'EmploymentType', 'MaritalStatus',
       'HasMortgage', 'HasDependents', 'LoanPurpose', 'HasCoSigner',
       'Default'],
      dtype='object')
```

-> Data types of each column

In [10]:
```
df.dtypes
```

Out[10]:
```
LoanID             object
Age               float64
Income            float64
LoanAmount        float64
CreditScore       float64
MonthsEmployed      int64
NumCreditLines      int64
InterestRate      float64
LoanTerm            int64
DTIRatio          float64
Education          object
EmploymentType     object
MaritalStatus      object
HasMortgage        object
HasDependents      object
LoanPurpose        object
HasCoSigner        object
Default             int64
dtype: object
```

In [11]:
```
# To get datatype of single column
df["LoanAmount"].dtype
```

Out[11]:
```
dtype('float64')
```

-> Initial Observations

In [12]:
```
# HEAD()
```

```
df.head()   #Shows the first few rows of the dataset(default 5 rows)
```

Out[12]:

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRatio	Educatio
0	I38PQUQS96	56.0	85994.0	50587.0	520.0	80	4	15.23	36	0.44	Bachelor'
1	HPSK72WA7R	69.0	50432.0	124440.0	458.0	15	1	4.81	60	0.68	Master'
2	C1OZ6DPJ8Y	46.0	84208.0	129188.0	451.0	26	3	21.17	24	0.31	Master'
3	V2KKSFM3UN	32.0	31713.0	44799.0	743.0	0	3	7.07	24	0.23	Higl Schoc
4	EY08JDHTZP	60.0	20437.0	9139.0	633.0	8	4	6.51	48	0.73	Bachelor'

In [13]:

```
df.head(10)   # Shows first 10 rows of the dataset
```

Out[13]:

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRatio	Educatio
0	I38PQUQS96	56.0	85994.0	50587.0	520.0	80	4	15.23	36	0.44	Bachelo
1	HPSK72WA7R	69.0	50432.0	124440.0	458.0	15	1	4.81	60	0.68	Maste
2	C1OZ6DPJ8Y	46.0	84208.0	129188.0	451.0	26	3	21.17	24	0.31	Maste
3	V2KKSFM3UN	32.0	31713.0	44799.0	743.0	0	3	7.07	24	0.23	Hi Schc
4	EY08JDHTZP	60.0	20437.0	9139.0	633.0	8	4	6.51	48	0.73	Bachelo
5	A9S62RQ7US	25.0	90298.0	90448.0	720.0	18	2	22.72	24	0.10	Hi Schc
6	H8GXPAOS71	38.0	111188.0	177025.0	429.0	80	1	19.11	12	0.16	Bachelo
7	0HGZQKJ36W	56.0	126802.0	155511.0	531.0	67	4	8.15	60	0.43	Pł
8	1R0N3LGNRJ	36.0	42053.0	92357.0	827.0	83	1	23.94	48	0.20	Bachelo
9	CM9L1GTT2P	40.0	132784.0	228510.0	480.0	114	4	9.09	48	0.33	Hi Schc

In [14]:

```
# TAIL()
df.tail()  #Shows last 5 default rows of the dataset
```

Out[14]:

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRatio	Ec
322370	PYQE3BDKX3	NaN	NaN	NaN	NaN	94	3	2.88	24	0.82	M
322371	0MF69ZQSJB	NaN	NaN	NaN	NaN	4	2	19.60	12	0.63	
322372	7ISLSUD0G2	NaN	NaN	NaN	NaN	85	1	15.03	60	0.47	M
322373	FJ6QE5V305	NaN	NaN	NaN	NaN	43	4	17.56	12	0.54	
322374	NWANQE3AER	NaN	NaN	NaN	NaN	69	2	5.74	36	0.13	b.

In [15]:

```
df.tail(10)  #Shows last 10 rows of the dataset
```

Out[15]:

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRatio	Ec
322365	YDZLFA17NO	NaN	NaN	NaN	NaN	108	3	5.74	60	0.19	M
322366	TQY19MUA96	NaN	NaN	NaN	NaN	82	4	6.99	12	0.73	
322367	CQ35I264AJ	NaN	NaN	NaN	NaN	66	1	16.19	24	0.43	b
322368	WNB79DK668	NaN	NaN	NaN	NaN	57	2	23.01	60	0.10	M
322369	CFJ1ZTVHVV	NaN	NaN	NaN	NaN	7	3	20.63	12	0.69	b
322370	PYQE3BDKX3	NaN	NaN	NaN	NaN	94	3	2.88	24	0.82	M
322371	0MF69ZQSJB	NaN	NaN	NaN	NaN	4	2	19.60	12	0.63	
322372	7ISLSUD0G2	NaN	NaN	NaN	NaN	85	1	15.03	60	0.47	M
322373	FJ6QE5V305	NaN	NaN	NaN	NaN	43	4	17.56	12	0.54	
322374	NWANQE3AER	NaN	NaN	NaN	NaN	69	2	5.74	36	0.13	b

In [16]:

```
# INDEX()
df.index  # Shows the row labels(index) of the dataframe
```

Out[16]:

```
RangeIndex(start=0, stop=322375, step=1)
```

In [17]:

```
# INFO()
# Number of rows
# Column names
# Data Types
# Non-null counts
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 322375 entries, 0 to 322374
Data columns (total 18 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   LoanID          322375 non-null  object 
 1   Age             312799 non-null  float64
 2   Income          312799 non-null  float64
 3   LoanAmount      312799 non-null  float64
 4   CreditScore     312799 non-null  float64
 5   MonthsEmployed  322375 non-null  int64  
 6   NumCreditLines  322375 non-null  int64  
 7   InterestRate    322375 non-null  float64
 8   LoanTerm        322375 non-null  int64  
 9   DTIRatio        322375 non-null  float64
 10  Education       322375 non-null  object 
 11  EmploymentType  322375 non-null  object 
 12  MaritalStatus   322375 non-null  object 
 13  HasMortgage     322375 non-null  object 
 14  HasDependents   322375 non-null  object 
 15  LoanPurpose     322375 non-null  object 
 16  HasCoSigner     322375 non-null  object 
 17  Default         322375 non-null  int64  
dtypes: float64(6), int64(4), object(8)
memory usage: 44.3+ MB
```

In [18]:
```
df.describe()   # Numerical Columns
```

Out[18]:

	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	
count	312799.000000	312799.000000	312799.000000	312799.000000	322375.000000	322375.000000	322375.000000	322375.000000	322
mean	43.500011	82492.467060	127598.987104	574.378064	59.567432	2.500278	13.497270	36.036144	
std	14.994360	38961.793881	70837.575367	158.864188	34.640502	1.116941	6.635982	16.987138	
min	18.000000	15000.000000	5000.000000	300.000000	0.000000	1.000000	2.000000	12.000000	
25%	31.000000	48811.000000	66190.500000	437.000000	30.000000	2.000000	7.780000	24.000000	
50%	43.000000	82484.000000	127506.000000	574.000000	60.000000	2.000000	13.460000	36.000000	
75%	56.000000	116191.500000	189062.500000	712.000000	90.000000	3.000000	19.260000	48.000000	
max	69.000000	149999.000000	249999.000000	849.000000	119.000000	4.000000	25.000000	60.000000	

In [19]:

```
df.describe(include='all')    # All Columns
```

Out[19]:

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	
count	322375	312799.000000	312799.000000	312799.000000	312799.000000	322375.000000	322375.000000	322375.000000	322
unique	255347	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
top	26XYW14YHT	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
freq	3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
mean	NaN	43.500011	82492.467060	127598.987104	574.378064	59.567432	2.500278	13.497270	
std	NaN	14.994360	38961.793881	70837.575367	158.864188	34.640502	1.116941	6.635982	
min	NaN	18.000000	15000.000000	5000.000000	300.000000	0.000000	1.000000	2.000000	
25%	NaN	31.000000	48811.000000	66190.500000	437.000000	30.000000	2.000000	7.780000	
50%	NaN	43.000000	82484.000000	127506.000000	574.000000	60.000000	2.000000	13.460000	
75%	NaN	56.000000	116191.500000	189062.500000	712.000000	90.000000	3.000000	19.260000	
max	NaN	69.000000	149999.000000	249999.000000	849.000000	119.000000	4.000000	25.000000	

---

2. Data Pre-Processing

➡️ Perform all neccessary cleaning steps such as :

➡️ Handling Missing Values

In [20]:

```
#Count missing values of each columns
df.isnull().sum()
```

Out[20]:

```
LoanID              0
Age              9576
Income           9576
LoanAmount       9576
CreditScore      9576
MonthsEmployed      0
NumCreditLines      0
InterestRate        0
LoanTerm            0
DTIRatio            0
Education           0
EmploymentType      0
MaritalStatus       0
HasMortgage         0
HasDependents       0
LoanPurpose         0
HasCoSigner         0
Default             0
dtype: int64
```

In [21]:

```
#Filling missing values
df["Age"].fillna(df["Age"].median(), inplace=True)
df["Income"].fillna(df["Income"].mean(), inplace=True)
df["LoanAmount"].fillna(df["LoanAmount"].mean(), inplace=True)
df["CreditScore"].fillna(df["CreditScore"].mean(), inplace=True)
df.tail(55)
```

Out[21]:

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRa
322320	QQVV5KC2G5	43.0	82492.46706	127598.987104	574.378064	65	3	4.44	12	0
322321	A59MTVVARI	43.0	82492.46706	127598.987104	574.378064	52	2	21.67	24	0
322322	BB67I4RKLN	43.0	82492.46706	127598.987104	574.378064	2	1	8.29	60	0
322323	E9O7GNC8TE	43.0	82492.46706	127598.987104	574.378064	112	3	6.90	12	0
322324	BW8206KOSJ	43.0	82492.46706	127598.987104	574.378064	76	4	3.52	24	0
322325	ZE2YUDK21G	43.0	82492.46706	127598.987104	574.378064	8	3	24.50	12	0
322326	OHZAKSMZBW	43.0	82492.46706	127598.987104	574.378064	59	3	12.13	12	0
322327	LDDMTHQW7M	43.0	82492.46706	127598.987104	574.378064	107	1	24.85	60	0
322328	NUG31RTOU0	43.0	82492.46706	127598.987104	574.378064	74	2	24.58	12	0
322329	IMQO8C8K57	43.0	82492.46706	127598.987104	574.378064	1	4	11.36	24	0
322330	PVJEONSUIH	43.0	82492.46706	127598.987104	574.378064	106	2	11.65	36	0
322331	903OT2UQPI	43.0	82492.46706	127598.987104	574.378064	117	4	12.92	48	0
322332	8926FZRIQ8	43.0	82492.46706	127598.987104	574.378064	101	3	19.67	60	0
322333	WLOMJZMA5P	43.0	82492.46706	127598.987104	574.378064	13	3	20.83	24	0
322334	7MLU510SSQ	43.0	82492.46706	127598.987104	574.378064	106	3	2.51	48	0
322335	QLA01DFD76	43.0	82492.46706	127598.987104	574.378064	96	4	12.83	36	0
322336	UL8Z7TQOJV	43.0	82492.46706	127598.987104	574.378064	100	1	3.06	24	0
322337	843Z1BY8NI	43.0	82492.46706	127598.987104	574.378064	97	2	10.53	48	0
322338	XYMXOWZBNM	43.0	82492.46706	127598.987104	574.378064	75	1	6.31	36	0

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRa
322339	W1BPGEIRAZ	43.0	82492.46706	127598.987104	574.378064	118	2	4.66	24	0
322340	L1LHJ4XIGW	43.0	82492.46706	127598.987104	574.378064	32	3	16.87	12	0
322341	SJZ1CQORGD	43.0	82492.46706	127598.987104	574.378064	90	3	13.43	24	0
322342	P6GGW4XVDU	43.0	82492.46706	127598.987104	574.378064	33	3	19.98	48	0
322343	XL2GH7B5T1	43.0	82492.46706	127598.987104	574.378064	28	3	14.57	24	0
322344	8MMH2KDN6J	43.0	82492.46706	127598.987104	574.378064	65	3	15.43	24	0
322345	GFS3R8GENH	43.0	82492.46706	127598.987104	574.378064	111	3	18.88	24	0
322346	II35MQ6ZYA	43.0	82492.46706	127598.987104	574.378064	75	1	7.34	36	0
322347	11IBUB4UYI	43.0	82492.46706	127598.987104	574.378064	6	3	2.94	60	0
322348	9FWEI1DYWZ	43.0	82492.46706	127598.987104	574.378064	77	3	8.79	24	0
322349	GQ5UJRZZ9H	43.0	82492.46706	127598.987104	574.378064	60	2	10.48	24	0
322350	IY9MMAP1E4	43.0	82492.46706	127598.987104	574.378064	42	3	24.99	60	0
322351	3GATXJRKU9	43.0	82492.46706	127598.987104	574.378064	113	4	10.44	24	0
322352	5ORYQ2EG5Y	43.0	82492.46706	127598.987104	574.378064	106	4	5.05	36	0
322353	USD54KQRQ7	43.0	82492.46706	127598.987104	574.378064	63	1	14.78	48	0
322354	JV3M5ES6WY	43.0	82492.46706	127598.987104	574.378064	70	1	4.72	12	0
322355	8YNLV79BC4	43.0	82492.46706	127598.987104	574.378064	14	4	8.53	60	0
322356	3CWMCCZZIK	43.0	82492.46706	127598.987104	574.378064	84	1	10.37	48	0
322357	7PI6DB1N4M	43.0	82492.46706	127598.987104	574.378064	119	4	9.20	36	0

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRa
322358	T21EXWIKTY	43.0	82492.46706	127598.987104	574.378064	115	2	22.11	24	0
322359	EAFUROBVPQ	43.0	82492.46706	127598.987104	574.378064	38	3	3.24	60	0
322360	X4TUGNIET0	43.0	82492.46706	127598.987104	574.378064	39	2	13.30	36	0
322361	94UEE939YF	43.0	82492.46706	127598.987104	574.378064	76	4	10.97	12	0
322362	W61H219UDU	43.0	82492.46706	127598.987104	574.378064	8	2	2.07	60	0
322363	J7CBWFLXFN	43.0	82492.46706	127598.987104	574.378064	73	2	21.08	24	0
322364	1IVWOLUAU4	43.0	82492.46706	127598.987104	574.378064	24	1	12.38	36	0
322365	YDZLFA17NO	43.0	82492.46706	127598.987104	574.378064	108	3	5.74	60	0
322366	TQY19MUA96	43.0	82492.46706	127598.987104	574.378064	82	4	6.99	12	0
322367	CQ35I264AJ	43.0	82492.46706	127598.987104	574.378064	66	1	16.19	24	0
322368	WNB79DK668	43.0	82492.46706	127598.987104	574.378064	57	2	23.01	60	0
322369	CFJ1ZTVHVV	43.0	82492.46706	127598.987104	574.378064	7	3	20.63	12	0
322370	PYQE3BDKX3	43.0	82492.46706	127598.987104	574.378064	94	3	2.88	24	0
322371	0MF69ZQSJB	43.0	82492.46706	127598.987104	574.378064	4	2	19.60	12	0
322372	7ISLSUD0G2	43.0	82492.46706	127598.987104	574.378064	85	1	15.03	60	0
322373	FJ6QE5V305	43.0	82492.46706	127598.987104	574.378064	43	4	17.56	12	0
322374	NWANQE3AER	43.0	82492.46706	127598.987104	574.378064	69	2	5.74	36	0

In [22]:
```
# Verifying if missing values are filled
print(df.isnull().sum())
```

```
LoanID            0
Age               0
Income            0
LoanAmount        0
CreditScore       0
MonthsEmployed    0
NumCreditLines    0
InterestRate      0
LoanTerm          0
DTIRatio          0
Education         0
EmploymentType    0
MaritalStatus     0
HasMortgage       0
HasDependents     0
LoanPurpose       0
HasCoSigner       0
Default           0
dtype: int64
```

In [23]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 322375 entries, 0 to 322374
Data columns (total 18 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   LoanID          322375 non-null  object 
 1   Age             322375 non-null  float64
 2   Income          322375 non-null  float64
 3   LoanAmount      322375 non-null  float64
 4   CreditScore     322375 non-null  float64
 5   MonthsEmployed  322375 non-null  int64  
 6   NumCreditLines  322375 non-null  int64  
 7   InterestRate    322375 non-null  float64
 8   LoanTerm        322375 non-null  int64  
 9   DTIRatio        322375 non-null  float64
 10  Education       322375 non-null  object 
 11  EmploymentType  322375 non-null  object 
 12  MaritalStatus   322375 non-null  object 
 13  HasMortgage     322375 non-null  object 
 14  HasDependents   322375 non-null  object 
 15  LoanPurpose     322375 non-null  object 
 16  HasCoSigner     322375 non-null  object 
 17  Default         322375 non-null  int64  
dtypes: float64(6), int64(4), object(8)
memory usage: 44.3+ MB
```

✅ Removing Multiple Spaces, Leading /Trailing Spaces(Stripping),Changing to Proper Cases

In [24]:
```
#Convert to proper case and fix apostrophes
categorical_cols = ["Education", "EmploymentType", "MaritalStatus", "LoanPurpose"]
df[categorical_cols] = df[categorical_cols].apply(
    lambda col: col.astype(str)
                    .str.strip()                              # remove leading/trailing spaces
                    .str.replace(r"[^a-zA-Z\s']", "", regex=True) # keep only alphabets & '
                    .str.replace(r"\s+", " ", regex=True)     # collapse multiple spaces
                    .str.title()                              # proper case
)
print(df[categorical_cols].tail(15))
```

```
          Education EmploymentType MaritalStatus LoanPurpose
322360          Phd       Parttime       Married        Auto
322361     Master'S       Parttime      Divorced        Home
322362   Bachelor'S     Unemployed        Single        Home
322363  High School   Selfemployed       Married        Home
322364  High School   Selfemployed      Divorced       Other
322365     Master'S     Unemployed       Married   Education
322366          Phd   Selfemployed        Single        Home
322367   Bachelor'S       Parttime      Divorced       Other
322368     Master'S       Fulltime       Married       Other
322369   Bachelor'S       Parttime       Married        Home
322370     Master'S       Fulltime       Married       Other
322371  High School       Fulltime      Divorced  Education
322372     Master'S     Unemployed      Divorced        Auto
322373          Phd       Parttime        Single  Education
322374   Bachelor'S   Selfemployed       Married        Home
```

In [25]:

```python
# Remove hash symbols from LoanPurpose column
df['LoanPurpose'] = df['LoanPurpose'].astype(str).str.replace('#', '', regex=False)

# Check again if hash symbol exists in LoanPurpose column
hash_exists = df['LoanPurpose'].str.contains('#').any()

print("Does '#' still exist in LoanPurpose?:", hash_exists)
```

```
Does '#' still exist in LoanPurpose?: False
```

In [26]:

```python
# Clean 'LoanPurpose' column
df['LoanPurpose'] = df['LoanPurpose'].str.strip()       # remove spaces
df['LoanPurpose'] = df['LoanPurpose'].str.lower()       # make lowercase
```

In [27]:

```python
df['LoanPurpose'].unique()
```

Out[27]:

```
array(['other', 'auto', 'business', 'home', 'education'], dtype=object)
```

In [28]:

```python
# Convert all values in Education column to proper case (first letter capitalized)
df["Education"] = df["Education"].str.strip().str.title()
```

In [29]:

```python
print(df["Education"].unique())
```

```
["Bachelor'S" "Master'S" 'High School' 'Phd']
```

In [30]:
```python
# Convert all values in EmploymentType column to proper case (first letter capitalized)
df["EmploymentType"] = df["EmploymentType"].str.strip().str.title()
```

In [31]:
```python
print(df["EmploymentType"].unique())
```

```
['Fulltime' 'Unemployed' 'Selfemployed' 'Parttime']
```

In [32]:
```python
# Convert all values in MaritalStatus column to proper case (first letter capitalized)
df["MaritalStatus"] = df["MaritalStatus"].str.strip().str.title()
```

In [33]:
```python
print(df["MaritalStatus"].unique())
```

```
['Divorced' 'Married' 'Single']
```

In [34]:
```python
#Checking if any row as null values
print(df[df.isnull().any(axis=1)])
```

```
Empty DataFrame
Columns: [LoanID, Age, Income, LoanAmount, CreditScore, MonthsEmployed, NumCreditLines, InterestRate, LoanTerm, DTIRatio, Educa
tion, EmploymentType, MaritalStatus, HasMortgage, HasDependents, LoanPurpose, HasCoSigner, Default]
Index: []
```

In [35]:
```python
#Checking if all values are null or  not
print(df.isnull().sum())
print(df.isnull().any().any())
```

```
LoanID            0
Age               0
Income            0
LoanAmount        0
CreditScore       0
MonthsEmployed    0
NumCreditLines    0
InterestRate      0
LoanTerm          0
DTIRatio          0
Education         0
EmploymentType    0
MaritalStatus     0
HasMortgage       0
HasDependents     0
LoanPurpose       0
HasCoSigner       0
Default           0
dtype: int64
False
```

➡️ Removing Duplicates

In [36]:
```
# Count the number of duplicated rows in dataset
df.duplicated().sum()
```

Out[36]:
```
np.int64(60644)
```

In [37]:
```
#Removing duplicates across the dataset
df = df.drop_duplicates()
```

In [38]:
```
#After removing duplicates the total number of rows.
print(df.shape)
```

```
(261731, 18)
```

➡️ Correcting Data Types

In [39]:
```
#Correcting datatypes makes the dataset clean, efficient, and ready for analysis.

# Making a copy of the original  dataframe df.
df_cleaned = df.copy()

# 1. Converting Yes/No columns to boolean of HasDependents,HasCosigner
bool_columns = ["HasDependents", "HasCoSigner"]
for col in bool_columns:
    df_cleaned[col] = df_cleaned[col].map({"Yes": True, "No": False})

# 2. Converting Yes/No columns to boolean of HasMortgage
    df_cleaned["HasMortgage"] = df_cleaned["HasMortgage"].astype(bool)


# 3. Converting categorical text columns to category dtype
categorical_columns = ["Education", "EmploymentType", "MaritalStatus", "LoanPurpose"]
for col in categorical_columns:
    df_cleaned[col] = df_cleaned[col].astype("category")

# 4. Converting Default column to boolean (if needed otherwise keep it as integer)
df_cleaned["Default"] = df_cleaned["Default"].astype(bool)

# 5. Converting Age to nullable integer (inplace of float)
df["Age"] = df["Age"].astype("Int64")


#  Now checking the updated datatypes
print(df_cleaned.dtypes)
```

```
LoanID               object
Age                 float64
Income              float64
LoanAmount          float64
CreditScore         float64
MonthsEmployed        int64
NumCreditLines        int64
InterestRate        float64
LoanTerm              int64
DTIRatio            float64
Education          category
EmploymentType     category
MaritalStatus      category
HasMortgage            bool
HasDependents          bool
LoanPurpose        category
HasCoSigner            bool
Default                bool
dtype: object
```

➡️ Creating derived columns

In [40]:
```
# Derived columns  created from existing columns in your dataset.
# Their main purpose is to extract more meaningful information or make patterns easier to detect for analysis.
```

Loan_to_Income

In [41]:
```
# how much of loan amount compared to annual income
df['Loan_to_Income'] = df['LoanAmount'] / df['Income']
```

Credit_Utilization_Score

In [42]:
```
#loan utilization based on credit lines.
#Measures how much loan per credit line the borrower is using.
df['Credit_Utilization'] = df['LoanAmount']/(df['NumCreditLines']+1)
```

Years_Employed

In [43]:
```
#Converted months of employment into years.
df['Years_Employed'] = df['MonthsEmployed']/12
```

Age_Group

In [44]:
```
#Categorized customers(young,mid-age,Mature,Senior)
df['Age_Group'] = pd.cut(df['Age'], bins=[18, 30, 45, 60, 100],
                         labels=['Young', 'Mid-age', 'Mature', 'Senior'])
```

Risk_Score

In [45]:
```
#Based on Risk_Score, categorizing
df['Risk_Score'] =  (df['CreditScore']/850) - df['DTIRatio'] - df['Loan_to_Income']
```

In [46]:
```
# To see the whole (maximum number of columns) for displaying the output
pd.set_option('display.max_columns', None)
df.head()   # now all columns will be displayed
```

Out[46]:

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRatio	Educatio
0	I38PQUQS96	56	85994.0	50587.0	520.0	80	4	15.23	36	0.44	Bachelor'
1	HPSK72WA7R	69	50432.0	124440.0	458.0	15	1	4.81	60	0.68	Master'
2	C1OZ6DPJ8Y	46	84208.0	129188.0	451.0	26	3	21.17	24	0.31	Master'
3	V2KKSFM3UN	32	31713.0	44799.0	743.0	0	3	7.07	24	0.23	Higl Schoc
4	EY08JDHTZP	60	20437.0	9139.0	633.0	8	4	6.51	48	0.73	Bachelor'

➡️ Filtering Data

In [47]:
```
#Filtering means selecting only the rows that meet certain conditions from your dataset
```

High Income Borrowers

In [48]:
```
# Borrowers with Income > 50000
high_income = df[df['Income'] > 50000]
print(high_income.head(10))
```

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	\
0	I38PQUQS96	56	85994.0	50587.0	520.0	80	
1	HPSK72WA7R	69	50432.0	124440.0	458.0	15	
2	C1OZ6DPJ8Y	46	84208.0	129188.0	451.0	26	
5	A9S62RQ7US	25	90298.0	90448.0	720.0	18	
6	H8GXPAOS71	38	111188.0	177025.0	429.0	80	
7	0HGZQKJ36W	56	126802.0	155511.0	531.0	67	
9	CM9L1GTT2P	40	132784.0	228510.0	480.0	114	
10	IA35XVH6ZO	28	140466.0	163781.0	652.0	94	
11	Y8UETC3LSG	28	149227.0	139759.0	375.0	56	
13	GX5YQOGROM	53	117550.0	95744.0	395.0	112	

	NumCreditLines	InterestRate	LoanTerm	DTIRatio	Education	\
0	4	15.23	36	0.44	Bachelor'S	
1	1	4.81	60	0.68	Master'S	
2	3	21.17	24	0.31	Master'S	
5	2	22.72	24	0.10	High School	
6	1	19.11	12	0.16	Bachelor'S	
7	4	8.15	60	0.43	Phd	
9	4	9.09	48	0.33	High School	
10	2	9.08	48	0.23	High School	
11	3	5.84	36	0.80	Phd	
13	4	3.58	24	0.73	High School	

	EmploymentType	MaritalStatus	HasMortgage	HasDependents	LoanPurpose	\
0	Fulltime	Divorced	Yes	Yes	other	
1	Fulltime	Married	No	No	other	
2	Unemployed	Divorced	Yes	Yes	auto	
5	Unemployed	Single	Yes	No	business	
6	Unemployed	Single	Yes	No	home	
7	Fulltime	Married	No	No	home	
9	Selfemployed	Married	Yes	No	other	
10	Unemployed	Married	No	No	education	
11	Fulltime	Divorced	No	No	education	
13	Unemployed	Single	No	No	auto	

	HasCoSigner	Default	Loan_to_Income	Credit_Utilization	Years_Employed	\
0	Yes	0	0.588262	10117.400000	6.666667	
1	Yes	0	2.467481	62220.000000	1.250000	
2	No	1	1.534154	32297.000000	2.166667	
5	Yes	1	1.001661	30149.333333	1.500000	

```
6          Yes      0      1.592123      88512.500000      6.666667
7          Yes      0      1.226408      31102.200000      5.583333
9          Yes      0      1.720915      45702.000000      9.500000
10          No      0      1.165983      54593.666667      7.833333
11         Yes      1      0.936553      34939.750000      4.666667
13         Yes      0      0.814496      19148.800000      9.333333

   Age_Group  Risk_Score
0     Mature   -0.416497
1     Senior   -2.608657
2     Mature   -1.313565
5      Young   -0.254602
6    Mid-age   -1.247417
7     Mature   -1.031702
9    Mid-age   -1.486209
10     Young   -0.628924
11     Young   -1.295377
13    Mature   -1.079790
```

Risky Customers

In [49]:
```
risky_customers = df[df['CreditScore'] < 600]
print(risky_customers.head(15))
```

```
       LoanID  Age    Income  LoanAmount  CreditScore  MonthsEmployed  \
0   I38PQUQS96   56   85994.0     50587.0        520.0              80   
1   HPSK72WA7R   69   50432.0    124440.0        458.0              15   
2   C1OZ6DPJ8Y   46   84208.0    129188.0        451.0              26   
6   H8GXPAOS71   38  111188.0    177025.0        429.0              80   
7   0HGZQKJ36W   56  126802.0    155511.0        531.0              67   
9   CM9L1GTT2P   40  132784.0    228510.0        480.0             114   
11  Y8UETC3LSG   28  149227.0    139759.0        375.0              56   
13  GX5YQOGROM   53  117550.0     95744.0        395.0             112   
15  O5DM5MPPNA   41   74064.0    230883.0        432.0              31   
16  ZDDRGVTEXS   20  119704.0     25697.0        313.0              49   
18  O1IKKLC69B   19   40718.0     78515.0        319.0             119   
19  F7487UU2BF   41  123419.0    161146.0        376.0              65   
20  7ASF0IHRIT   61   30142.0    133714.0        429.0              96   
21  A22KI1B6SE   47  146113.0    100621.0        419.0              55   
22  1MUSHWD9TW   55  132058.0    130912.0        583.0              48   

    NumCreditLines  InterestRate  LoanTerm  DTIRatio    Education  \
0                4         15.23        36      0.44   Bachelor'S   
1                1          4.81        60      0.68     Master'S   
2                3         21.17        24      0.31     Master'S   
6                1         19.11        12      0.16   Bachelor'S   
7                4          8.15        60      0.43          Phd   
9                4          9.09        48      0.33  High School   
11               3          5.84        36      0.80          Phd   
13               4          3.58        24      0.73  High School   
15               2          5.00        60      0.89     Master'S   
16               1          9.63        24      0.28          Phd   
18               2         14.00        24      0.17   Bachelor'S   
19               4         16.96        60      0.39  High School   
20               1         15.58        12      0.65          Phd   
21               1          9.32        12      0.38   Bachelor'S   
22               4          5.82        60      0.47  High School   

   EmploymentType MaritalStatus HasMortgage HasDependents LoanPurpose  \
0        Fulltime      Divorced         Yes           Yes       other   
1        Fulltime       Married          No            No       other   
2      Unemployed      Divorced         Yes           Yes        auto   
6      Unemployed        Single         Yes            No        home   
7        Fulltime       Married          No            No        home   
9    Selfemployed       Married         Yes            No       other   
```

```
11      Fulltime     Divorced          No            No   education
13    Unemployed       Single          No            No        auto
15    Unemployed      Married         Yes            No        auto
16    Unemployed       Single         Yes            No        home
18  Selfemployed     Divorced         Yes            No   education
19  Selfemployed       Single         Yes            No       other
20      Parttime     Divorced          No           Yes    business
21    Unemployed      Married         Yes           Yes    business
22    Unemployed      Married          No           Yes    business

   HasCoSigner  Default  Loan_to_Income  Credit_Utilization  Years_Employed  \
0          Yes        0        0.588262        10117.400000        6.666667
1          Yes        0        2.467481        62220.000000        1.250000
2           No        1        1.534154        32297.000000        2.166667
6          Yes        0        1.592123        88512.500000        6.666667
7          Yes        0        1.226408        31102.200000        5.583333
9          Yes        0        1.720915        45702.000000        9.500000
11         Yes        1        0.936553        34939.750000        4.666667
13         Yes        0        0.814496        19148.800000        9.333333
15          No        0        3.117344        76961.000000        2.583333
16          No        0        0.214671        12848.500000        4.083333
18          No        1        1.928263        26171.666667        9.916667
19         Yes        0        1.305682        32229.200000        5.416667
20          No        0        4.436136        66857.000000        8.000000
21          No        0        0.688652        50310.500000        4.583333
22         Yes        0        0.991322        26182.400000        4.000000

   Age_Group  Risk_Score
0     Mature   -0.416497
1     Senior   -2.608657
2     Mature   -1.313565
6    Mid-age   -1.247417
7     Mature   -1.031702
9    Mid-age   -1.486209
11     Young   -1.295377
13    Mature   -1.079790
15   Mid-age   -3.499109
16     Young   -0.126436
18     Young   -1.722969
19   Mid-age   -1.253329
20    Senior   -4.581430
```

```
21    Mature   -0.575711
22    Mature   -0.775440
```

Married Borrowers with Dependents

In [50]:
```
married_dependents = df[(df['MaritalStatus'] == 'Married') &
                        (df['HasDependents'] == 'Yes')]
print(married_dependents.head(15))
```

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	\
21	A22KI1B6SE	47	146113.0	100621.0	419.0	55	
22	1MUSHWD9TW	55	132058.0	130912.0	583.0	48	
41	RT511ZZNF2	54	21487.0	84115.0	386.0	32	
43	XG5WPXX0TY	24	105732.0	33013.0	400.0	58	
44	2XUD7N4OJ1	38	73117.0	221628.0	639.0	84	
46	Z7UFZIW3MK	56	24796.0	37657.0	498.0	88	
49	LMQOF5PTTT	42	143244.0	240591.0	393.0	96	
52	130U8TZ0HW	26	92153.0	214064.0	634.0	33	
55	QHPWTGMTU8	37	37173.0	234059.0	455.0	20	
61	Z3WJUIM1DZ	64	110606.0	137139.0	669.0	11	
66	ZOPZLF57NR	67	77135.0	159839.0	529.0	59	
68	ZY51VR44DK	21	142860.0	238718.0	318.0	35	
70	MZRL2WMB52	23	17142.0	110469.0	802.0	56	
75	5KCTUT4OSE	43	130625.0	17462.0	303.0	22	
76	94DS7MC3PV	61	49191.0	93324.0	487.0	51	

	NumCreditLines	InterestRate	LoanTerm	DTIRatio	Education	\
21	1	9.32	12	0.38	Bachelor'S	
22	4	5.82	60	0.47	High School	
41	2	20.89	60	0.78	Phd	
43	3	15.64	36	0.48	High School	
44	4	13.05	36	0.56	Bachelor'S	
46	1	8.20	48	0.73	High School	
49	3	23.15	36	0.16	High School	
52	3	2.34	24	0.32	Phd	
55	3	15.12	48	0.74	Phd	
61	3	18.01	60	0.66	Master'S	
66	4	4.14	36	0.41	High School	
68	3	13.06	48	0.36	High School	
70	3	12.23	12	0.71	High School	
75	3	17.91	60	0.65	Master'S	
76	2	2.65	12	0.25	High School	

	EmploymentType	MaritalStatus	HasMortgage	HasDependents	LoanPurpose	\
21	Unemployed	Married	Yes	Yes	business	
22	Unemployed	Married	No	Yes	business	
41	Unemployed	Married	Yes	Yes	business	
43	Fulltime	Married	Yes	Yes	auto	
44	Parttime	Married	No	Yes	home	
46	Parttime	Married	Yes	Yes	auto	

```
49   Selfemployed       Married           No            Yes    business
52       Fulltime       Married           No            Yes   education
55       Fulltime       Married           No            Yes       other
61       Parttime       Married           No            Yes        home
66       Fulltime       Married          Yes            Yes       other
68       Parttime       Married           No            Yes    business
70       Fulltime       Married          Yes            Yes        auto
75   Selfemployed       Married           No            Yes    business
76     Unemployed       Married           No            Yes        home

   HasCoSigner  Default  Loan_to_Income  Credit_Utilization  Years_Employed  \
21          No        0        0.688652        50310.500000        4.583333
22         Yes        0        0.991322        26182.400000        4.000000
41         Yes        0        3.914693        28038.333333        2.666667
43         Yes        0        0.312233         8253.250000        4.833333
44         Yes        0        3.031142        44325.600000        7.000000
46          No        0        1.518672        18828.500000        7.333333
49          No        0        1.679589        60147.750000        8.000000
52          No        0        2.322919        53516.000000        2.750000
55          No        0        6.296479        58514.750000        1.666667
61          No        0        1.239888        34284.750000        0.916667
66          No        0        2.072198        31967.800000        4.916667
68         Yes        0        1.670993        59679.500000        2.916667
70         Yes        0        6.444347        27617.250000        4.666667
75          No        0        0.133680         4365.500000        1.833333
76         Yes        0        1.897176        31108.000000        4.250000

   Age_Group  Risk_Score
21    Mature   -0.575711
22    Mature   -0.775440
41    Mature   -4.240575
43     Young   -0.321645
44   Mid-age   -2.839377
46    Mature   -1.662790
49   Mid-age   -1.377236
52     Young   -1.897037
55   Mid-age   -6.501185
61    Senior   -1.112829
66    Senior   -1.859845
68     Young   -1.656875
70     Young   -6.210818
```

```
75   Mid-age   -0.427210
76    Senior   -1.574235
```

High Interest Rate Loans

In [51]:
```
High_Interest_Rate_Loans = df[df['InterestRate'] > 15]
print(High_Interest_Rate_Loans.tail(15))
```

	LoanID	Age	Income	LoanAmount	CreditScore	\
318835	I9X9Q5416F	43	82492.46706	127598.987104	574.378064	
318845	22IGDI7J5V	43	82492.46706	127598.987104	574.378064	
318859	J7CBWFLXFN	43	82492.46706	127598.987104	574.378064	
318890	PGVWWAD1VU	43	82492.46706	127598.987104	574.378064	
318928	BPJ2A0NEVA	43	82492.46706	127598.987104	574.378064	
318931	OK6MMILY5J	43	82492.46706	127598.987104	574.378064	
318986	TT14R1P1M7	43	82492.46706	127598.987104	574.378064	
319010	LBY7AT792Z	43	82492.46706	127598.987104	574.378064	
319017	SS4ACX50Q8	43	82492.46706	127598.987104	574.378064	
319025	F0NIB82WJL	43	82492.46706	127598.987104	574.378064	
319067	VJ7YALTZZN	43	82492.46706	127598.987104	574.378064	
319076	CPSCTZ5QGZ	43	82492.46706	127598.987104	574.378064	
319078	NVR7SNNIHQ	43	82492.46706	127598.987104	574.378064	
319086	4VPUOQDT79	43	82492.46706	127598.987104	574.378064	
319103	21Q7Q1LBT1	43	82492.46706	127598.987104	574.378064	

	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRatio	\
318835	85	3	19.77	12	0.57	
318845	46	4	19.79	12	0.48	
318859	73	2	21.08	24	0.37	
318890	100	3	15.60	12	0.49	
318928	43	4	17.21	48	0.18	
318931	73	3	16.98	48	0.33	
318986	5	2	20.77	48	0.30	
319010	57	4	23.13	12	0.34	
319017	13	4	18.05	24	0.63	
319025	36	3	17.91	12	0.78	
319067	46	4	19.93	12	0.89	
319076	58	1	23.90	36	0.11	
319078	80	4	21.18	48	0.38	
319086	22	3	24.92	60	0.50	
319103	27	1	23.16	12	0.78	

	Education	EmploymentType	MaritalStatus	HasMortgage	HasDependents	\
318835	High School	Parttime	Married	No	Yes	
318845	Phd	Parttime	Divorced	No	No	
318859	High School	Selfemployed	Married	No	No	
318890	Bachelor'S	Selfemployed	Married	No	No	
318928	High School	Fulltime	Married	No	No	
318931	Master'S	Fulltime	Married	No	Yes	

318986	Master'S	Parttime	Married	No	No
319010	High School	Unemployed	Married	Yes	Yes
319017	Bachelor'S	Fulltime	Single	Yes	Yes
319025	Bachelor'S	Selfemployed	Married	No	Yes
319067	Bachelor'S	Fulltime	Divorced	Yes	Yes
319076	Phd	Unemployed	Single	Yes	No
319078	Bachelor'S	Selfemployed	Divorced	Yes	Yes
319086	Phd	Parttime	Divorced	Yes	No
319103	Master'S	Selfemployed	Divorced	No	No

	LoanPurpose	HasCoSigner	Default	Loan_to_Income	Credit_Utilization	\
318835	education	Yes	0	1.546796	31899.746776	
318845	other	No	0	1.546796	25519.797421	
318859	home	No	0	1.546796	42532.995701	
318890	auto	Yes	0	1.546796	31899.746776	
318928	auto	No	0	1.546796	25519.797421	
318931	education	No	0	1.546796	31899.746776	
318986	education	Yes	0	1.546796	42532.995701	
319010	home	No	0	1.546796	25519.797421	
319017	business	No	0	1.546796	25519.797421	
319025	home	Yes	0	1.546796	31899.746776	
319067	home	No	0	1.546796	25519.797421	
319076	other	Yes	0	1.546796	63799.493552	
319078	auto	Yes	0	1.546796	25519.797421	
319086	business	Yes	0	1.546796	31899.746776	
319103	home	Yes	0	1.546796	63799.493552	

	Years_Employed	Age_Group	Risk_Score
318835	7.083333	Mid-age	-1.441057
318845	3.833333	Mid-age	-1.351057
318859	6.083333	Mid-age	-1.241057
318890	8.333333	Mid-age	-1.361057
318928	3.583333	Mid-age	-1.051057
318931	6.083333	Mid-age	-1.201057
318986	0.416667	Mid-age	-1.171057
319010	4.750000	Mid-age	-1.211057
319017	1.083333	Mid-age	-1.501057
319025	3.000000	Mid-age	-1.651057
319067	3.833333	Mid-age	-1.761057
319076	4.833333	Mid-age	-0.981057
319078	6.666667	Mid-age	-1.251057

```
319086        1.833333   Mid-age   -1.371057
319103        2.250000   Mid-age   -1.651057
```

Senior Citizens

In [52]:
```
Senior_Citizens = df[df['Age'] > 60]
print(Senior_Citizens.head(20))
```

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	\
1	HPSK72WA7R	69	50432.0	124440.0	458.0	15	
20	7ASF0IHRIT	61	30142.0	133714.0	429.0	96	
28	BJNLQ0H95H	61	62519.0	29676.0	462.0	16	
30	GAA8OQN796	66	39568.0	58945.0	604.0	37	
36	8NTWNU4HTY	64	102463.0	218433.0	506.0	24	
37	ASYFXCP452	68	85409.0	44772.0	540.0	105	
38	4857M8R0YI	61	26470.0	19818.0	695.0	47	
39	5FENBP2UV8	69	87295.0	16281.0	707.0	94	
42	KAC7P2RE1X	68	111716.0	215851.0	747.0	99	
51	N3KDA4UM9K	67	100949.0	145906.0	439.0	36	
57	DSOB0M5AQ4	64	49565.0	245711.0	655.0	76	
59	0VEK3DDM69	61	49113.0	222046.0	771.0	47	
61	Z3WJUIM1DZ	64	110606.0	137139.0	669.0	11	
66	ZOPZLF57NR	67	77135.0	159839.0	529.0	59	
76	94DS7MC3PV	61	49191.0	93324.0	487.0	51	
82	SQ3MA71EVN	65	80770.0	217916.0	574.0	9	
90	I7Z9O3RG38	62	138569.0	248333.0	453.0	101	
92	CL2BS2EEJE	64	93741.0	232727.0	747.0	44	
96	D51F6APQ5Y	62	25274.0	134289.0	701.0	91	
100	XL0LIO7XMD	62	47383.0	13109.0	553.0	73	

	NumCreditLines	InterestRate	LoanTerm	DTIRatio	Education	\
1	1	4.81	60	0.68	Master'S	
20	1	15.58	12	0.65	Phd	
28	1	23.91	48	0.12	Bachelor'S	
30	4	6.67	12	0.10	High School	
36	2	9.23	60	0.86	Master'S	
37	1	2.96	36	0.17	Master'S	
38	2	20.00	36	0.69	Phd	
39	1	13.82	60	0.75	Bachelor'S	
42	3	18.84	60	0.40	High School	
51	2	3.74	48	0.58	Bachelor'S	
57	4	23.41	24	0.37	High School	
59	3	22.80	12	0.24	High School	
61	3	18.01	60	0.66	Master'S	
66	4	4.14	36	0.41	High School	
76	2	2.65	12	0.25	High School	
82	3	20.97	36	0.49	High School	
90	2	14.10	24	0.44	High School	
92	1	24.29	24	0.62	Master'S	

96	1	13.01	24	0.57	Master'S
100	3	19.57	12	0.42	Bachelor'S

	EmploymentType	MaritalStatus	HasMortgage	HasDependents	LoanPurpose	\
1	Fulltime	Married	No	No	other	
20	Parttime	Divorced	No	Yes	business	
28	Unemployed	Divorced	Yes	No	home	
30	Unemployed	Divorced	Yes	Yes	auto	
36	Unemployed	Married	No	No	auto	
37	Unemployed	Divorced	No	No	auto	
38	Unemployed	Divorced	Yes	Yes	auto	
39	Parttime	Single	No	No	other	
42	Unemployed	Divorced	No	Yes	home	
51	Unemployed	Married	No	No	auto	
57	Unemployed	Single	No	Yes	auto	
59	Unemployed	Divorced	No	Yes	other	
61	Parttime	Married	No	Yes	home	
66	Fulltime	Married	Yes	Yes	other	
76	Unemployed	Married	No	Yes	home	
82	Parttime	Single	Yes	No	home	
90	Parttime	Married	Yes	Yes	other	
92	Fulltime	Divorced	Yes	Yes	other	
96	Unemployed	Married	No	No	business	
100	Unemployed	Married	Yes	No	business	

	HasCoSigner	Default	Loan_to_Income	Credit_Utilization	Years_Employed	\
1	Yes	0	2.467481	62220.000000	1.250000	
20	No	0	4.436136	66857.000000	8.000000	
28	Yes	0	0.474672	14838.000000	1.333333	
30	Yes	0	1.489714	11789.000000	3.083333	
36	Yes	0	2.131823	72811.000000	2.000000	
37	No	0	0.524207	22386.000000	8.750000	
38	Yes	0	0.748697	6606.000000	3.916667	
39	No	0	0.186506	8140.500000	7.833333	
42	No	0	1.932140	53962.750000	8.250000	
51	Yes	0	1.445344	48635.333333	3.000000	
57	Yes	0	4.957349	49142.200000	6.333333	
59	No	0	4.521125	55511.500000	3.916667	
61	No	0	1.239888	34284.750000	0.916667	
66	No	0	2.072198	31967.800000	4.916667	
76	Yes	0	1.897176	31108.000000	4.250000	

```
82           No        0        2.697982        54479.000000        0.750000
90          Yes        0        1.792125        82777.666667        8.416667
92           No        0        2.482660       116363.500000        3.666667
96          Yes        0        5.313326        67144.500000        7.583333
100          No        0        0.276660         3277.250000        6.083333

    Age_Group  Risk_Score
1      Senior   -2.608657
20     Senior   -4.581430
28     Senior   -0.051142
30     Senior   -0.879126
36     Senior   -2.396529
37     Senior   -0.058913
38     Senior   -0.621050
39     Senior   -0.104741
42     Senior   -1.453317
51     Senior   -1.508873
57     Senior   -4.556761
59     Senior   -3.854066
61     Senior   -1.112829
66     Senior   -1.859845
76     Senior   -1.574235
82     Senior   -2.512688
90     Senior   -1.699184
92     Senior   -2.223836
96     Senior   -5.058620
100    Senior   -0.046072
```

Months Employed less than 1 year

In [53]:
```
Less_Than_Year=df[df['MonthsEmployed'] < 12]
print(Less_Than_Year.head(5))
```

```
       LoanID  Age    Income  LoanAmount  CreditScore  MonthsEmployed  \
3   V2KKSFM3UN   32   31713.0     44799.0        743.0               0   
4   EY08JDHTZP   60   20437.0      9139.0        633.0               8   
32  KD97QJJFD8   59  102292.0     55337.0        840.0               6   
47  RSP1YD80Z7   35   95963.0     77552.0        560.0               8   
53  E9NX4IRVSU   43   70113.0    215064.0        657.0               0   

    NumCreditLines  InterestRate  LoanTerm  DTIRatio    Education  \
3                3          7.07        24      0.23  High School   
4                4          6.51        48      0.73   Bachelor'S   
32               1         16.11        60      0.44     Master'S   
47               2          6.63        24      0.86     Master'S   
53               2          6.00        24      0.76   Bachelor'S   

   EmploymentType MaritalStatus HasMortgage HasDependents LoanPurpose  \
3        Fulltime       Married          No            No    business   
4      Unemployed      Divorced          No           Yes        auto   
32     Unemployed       Married         Yes            No        auto   
47   Selfemployed      Divorced         Yes           Yes        home   
53   Selfemployed        Single          No           Yes        home   

   HasCoSigner  Default  Loan_to_Income  Credit_Utilization  Years_Employed  \
3           No        0        1.412638        11199.750000        0.000000   
4           No        0        0.447179         1827.800000        0.666667   
32          No        0        0.540971        27668.500000        0.500000   
47          No        1        0.808145        25850.666667        0.666667   
53         Yes        0        3.067391        71688.000000        0.000000   

   Age_Group  Risk_Score  
3    Mid-age   -0.768521  
4     Mature   -0.432473  
32    Mature    0.007264  
47   Mid-age   -1.009321  
53   Mid-age   -3.054450  
```

High DTI Ratio

In [54]:
```
High_DTI_Ratio = df[df['DTIRatio'] > 0.5] # (>0.5-> risky)
print(High_DTI_Ratio.tail(10))
```

```
           LoanID  Age       Income     LoanAmount  CreditScore  \
319024  6BEKC0R79X   43  82492.46706  127598.987104   574.378064
319025  F0NIB82WJL   43  82492.46706  127598.987104   574.378064
319054  12IT24XQX5   43  82492.46706  127598.987104   574.378064
319066  0ZYOV6QRT8   43  82492.46706  127598.987104   574.378064
319067  VJ7YALTZZN   43  82492.46706  127598.987104   574.378064
319088  SFU0VSRSIU   43  82492.46706  127598.987104   574.378064
319094  Q5QSJCH8PO   43  82492.46706  127598.987104   574.378064
319103  21Q7Q1LBT1   43  82492.46706  127598.987104   574.378064
319130  LQT2S0YRTT   43  82492.46706  127598.987104   574.378064
319174  X8W1CFI5N4   43  82492.46706  127598.987104   574.378064

        MonthsEmployed  NumCreditLines  InterestRate  LoanTerm  DTIRatio  \
319024              13               4         10.74        60      0.51
319025              36               3         17.91        12      0.78
319054             111               2         10.32        48      0.68
319066              45               2         11.99        12      0.72
319067              46               4         19.93        12      0.89
319088              57               4         10.34        60      0.82
319094              28               4          7.08        12      0.59
319103              27               1         23.16        12      0.78
319130              82               2          6.22        36      0.75
319174              48               4         14.35        48      0.71

          Education EmploymentType MaritalStatus HasMortgage HasDependents  \
319024  High School       Parttime      Divorced          No           Yes
319025   Bachelor'S   Selfemployed       Married          No           Yes
319054  High School   Selfemployed       Married         Yes           Yes
319066  High School   Selfemployed      Divorced         Yes           Yes
319067   Bachelor'S       Fulltime      Divorced         Yes           Yes
319088  High School       Fulltime        Single         Yes            No
319094          Phd     Unemployed        Single          No           Yes
319103     Master'S   Selfemployed      Divorced          No            No
319130     Master'S     Unemployed      Divorced          No            No
319174     Master'S   Selfemployed        Single          No            No

       LoanPurpose HasCoSigner  Default  Loan_to_Income  Credit_Utilization  \
319024   education          No        0        1.546796        25519.797421
319025        home         Yes        0        1.546796        31899.746776
319054   education          No        0        1.546796        42532.995701
319066    business          No        0        1.546796        42532.995701
```

```
319067       home     No   0   1.546796    25519.797421
319088  education    Yes   0   1.546796    25519.797421
319094       home     No   0   1.546796    25519.797421
319103       home    Yes   0   1.546796    63799.493552
319130       auto    Yes   0   1.546796    42532.995701
319174   business     No   0   1.546796    25519.797421

        Years_Employed Age_Group  Risk_Score
319024        1.083333   Mid-age   -1.381057
319025        3.000000   Mid-age   -1.651057
319054        9.250000   Mid-age   -1.551057
319066        3.750000   Mid-age   -1.591057
319067        3.833333   Mid-age   -1.761057
319088        4.750000   Mid-age   -1.691057
319094        2.333333   Mid-age   -1.461057
319103        2.250000   Mid-age   -1.651057
319130        6.833333   Mid-age   -1.621057
319174        4.000000   Mid-age   -1.581057
```

➡️ Aggregating Data

Loan Purposes By Count

In [55]:
```
loan_purposes = df['LoanPurpose'].value_counts().head()
print("loan_Purposes by count:")
print(loan_purposes)
```

```
loan_Purposes by count:
LoanPurpose
home         52612
business     52543
education    52233
other        52211
auto         52132
Name: count, dtype: int64
```

Average Loan Amount by Education

In [56]:
```
avg_loan_by_edu = df.groupby("Education")["LoanAmount"].mean().sort_values(ascending=False).head()
print("\nAverage Loan Amount by Education:")
```

```
print(avg_loan_by_edu)
```

```
Average Loan Amount by Education:
Education
Phd            127818.417905
Master'S       127708.290883
High School    127398.648988
Bachelor'S     127396.014892
Name: LoanAmount, dtype: float64
```

Default Rate by Marital Status

In [57]:
```
default_by_marital = df.groupby("MaritalStatus")["Default"].mean().sort_values(ascending=False)
print("\nDefault Rate by Marital Status:")
print(default_by_marital)
```

```
Default Rate by Marital Status:
MaritalStatus
Divorced    0.125220
Single      0.119128
Married     0.103956
Name: Default, dtype: float64
```

Average Credit Score by Loan Purpose

In [58]:
```
avg_credit_by_purpose = df.groupby("LoanPurpose")["CreditScore"].mean().sort_values(ascending=False).head(10)
print("\nAverage Credit Score by Loan Purpose:")
print(avg_credit_by_purpose)
```

```
Average Credit Score by Loan Purpose:
LoanPurpose
home         575.043798
auto         574.571030
other        574.407124
business     574.069594
education    573.240236
Name: CreditScore, dtype: float64
```

Employment Types by Count

In [59]:
```
employment_types = df['EmploymentType'].value_counts().head()
print("\nEmployment Types by count:")
```

```
print(employment_types)
```

```
Employment Types by count:
EmploymentType
Parttime        65733
Selfemployed    65381
Unemployed      65378
Fulltime        65239
Name: count, dtype: int64
```

---

3. Exploratory Data Analysis (EDA)

➡️ Conduct descriptive and exploratory analysis to uncover pattern and trends :

➡️ Univariate,Bivariate and Multivariate Analysis

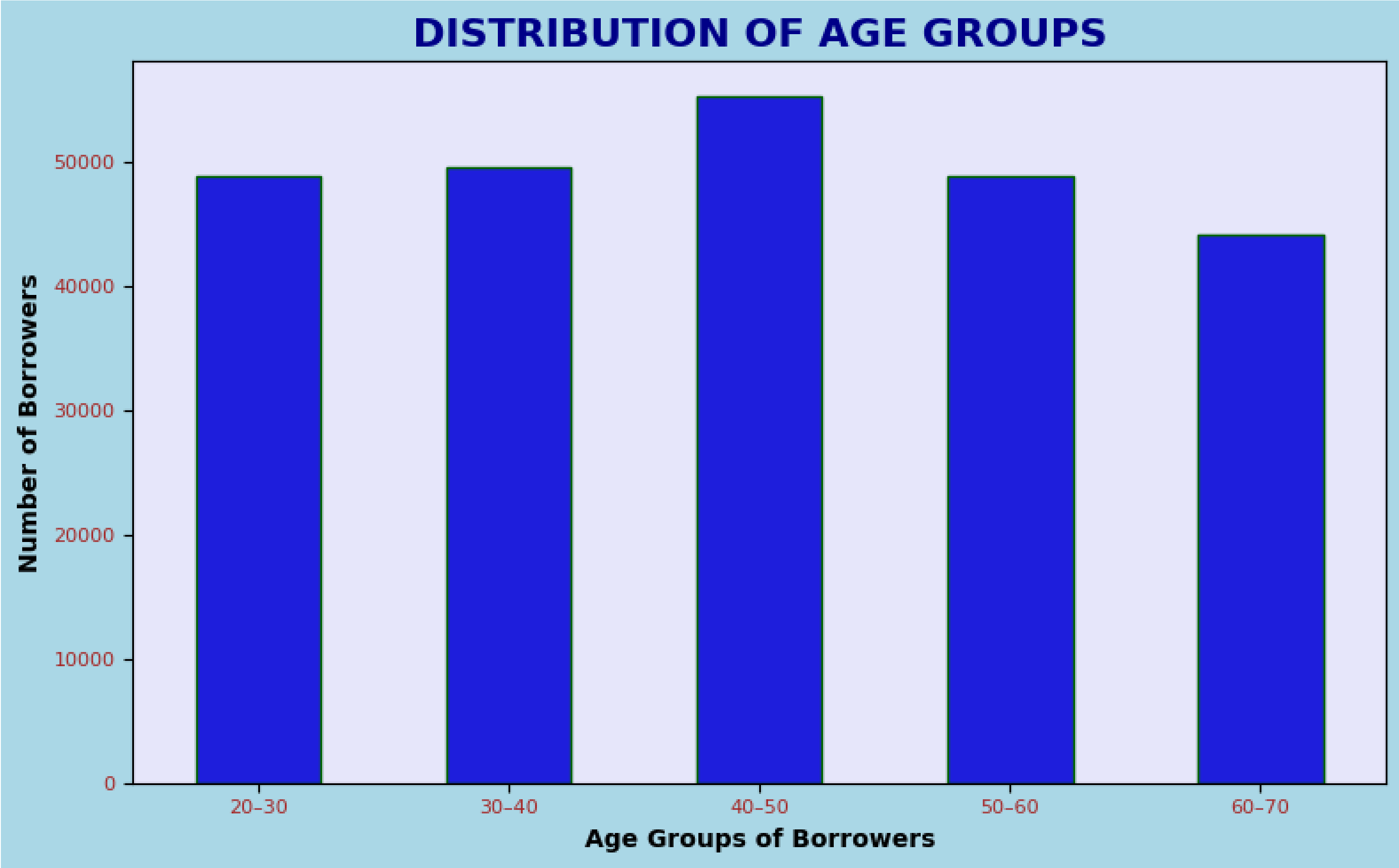
✅ Univariate Analysis(Single Variable Analysis)

Age

In [101…

```
# Distribution of Borrower age-> Are most borrowers middle aged or not?

# Create Age Groups
df['AgeGroup'] = pd.cut(
    df['Age'],
    bins=[20,30,40,50,60,70],
    labels=["20-30","30-40","40-50","50-60","60-70"]
)

plt.figure(figsize=(8,5), facecolor="lightblue")

# Bar plot for age groups
sns.countplot(x='AgeGroup', data=df, color="blue", edgecolor="darkgreen",width=0.5)

# Title
plt.title("DISTRIBUTION OF AGE GROUPS", fontsize=16, weight="bold", color="darkblue")
```

```python
# X and Y axis labels
plt.xlabel("Age Groups of Borrowers", fontsize=10, weight="bold", color="black")
plt.ylabel("Number of Borrowers", fontsize=10, weight="bold", color="black")

# Change background inside plot
plt.gca().set_facecolor("lavender")

# Change tick labels size and color
plt.xticks(color="brown", fontsize=8)
plt.yticks(color="brown", fontsize=8)

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



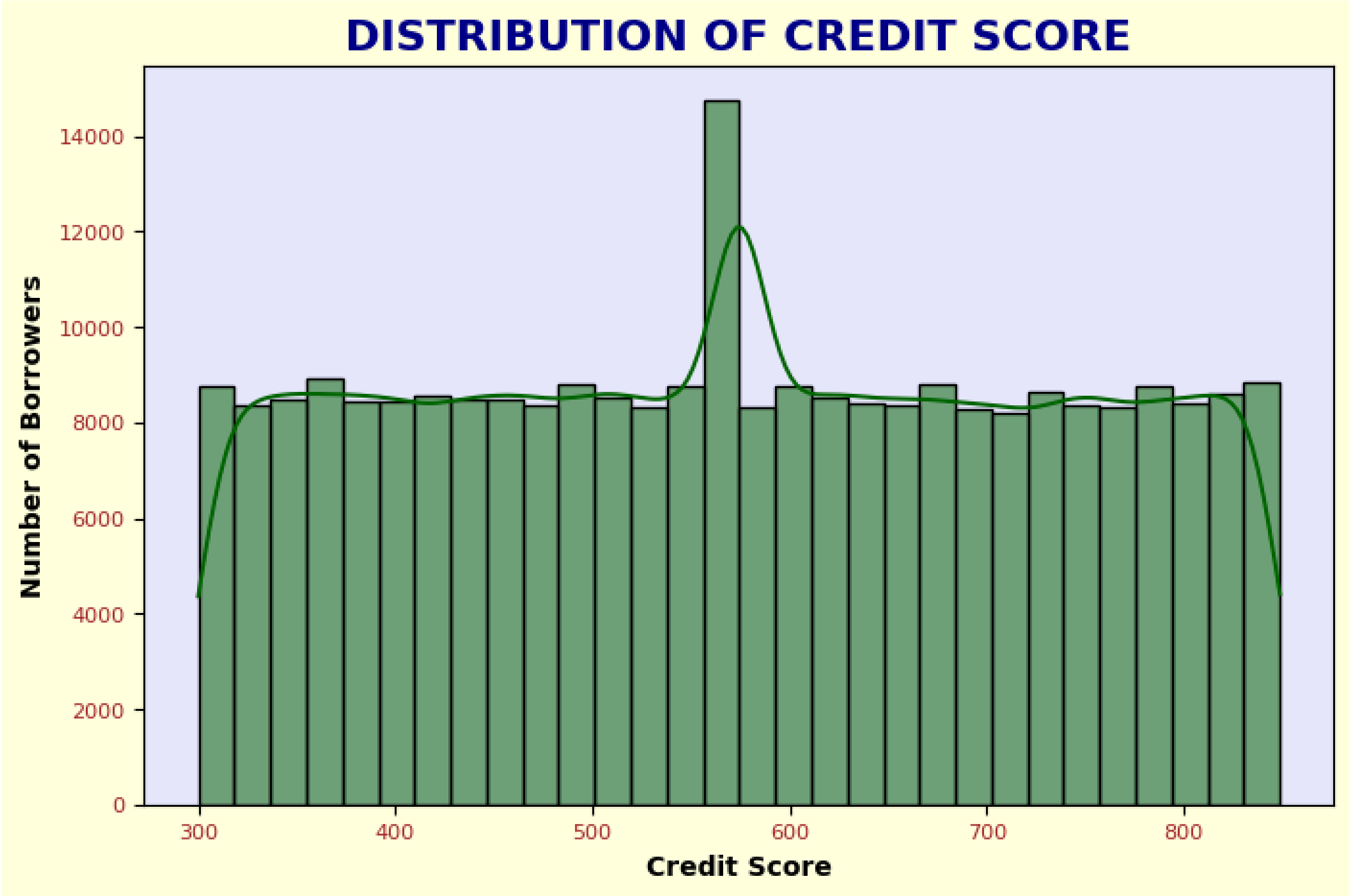
Insights

- The number of borrowers in 20-30,30-40,50-60 age groups are very close to each other,evenly distributed.
- The number of borrowers in 40-50 age group has a spike,as most stable point in their career.
- Drop in borrowers 60-70 age group possibly due to reduced income ,near to retirement.

CreditScore

In [100…

```
#Check if scores are normally distributed or skewed.

#Histogram
plt.figure(figsize=(8,5), facecolor="lightyellow")

# Histogram for Credit Score
sns.histplot(df['CreditScore'], kde=True, bins=30, color="darkgreen")

plt.title("DISTRIBUTION OF CREDIT SCORE", fontsize=16,weight="bold", color="darkblue")


plt.xlabel("Credit Score", fontsize=10,weight="bold", color="black")
plt.ylabel("Number of Borrowers", fontsize=10,weight="bold", color="black")

# Change background inside plot
plt.gca().set_facecolor("lavender")

# Change tick labels size and color
plt.xticks(color="brown", fontsize=8)
plt.yticks(color="brown", fontsize=8)
plt.show()
```



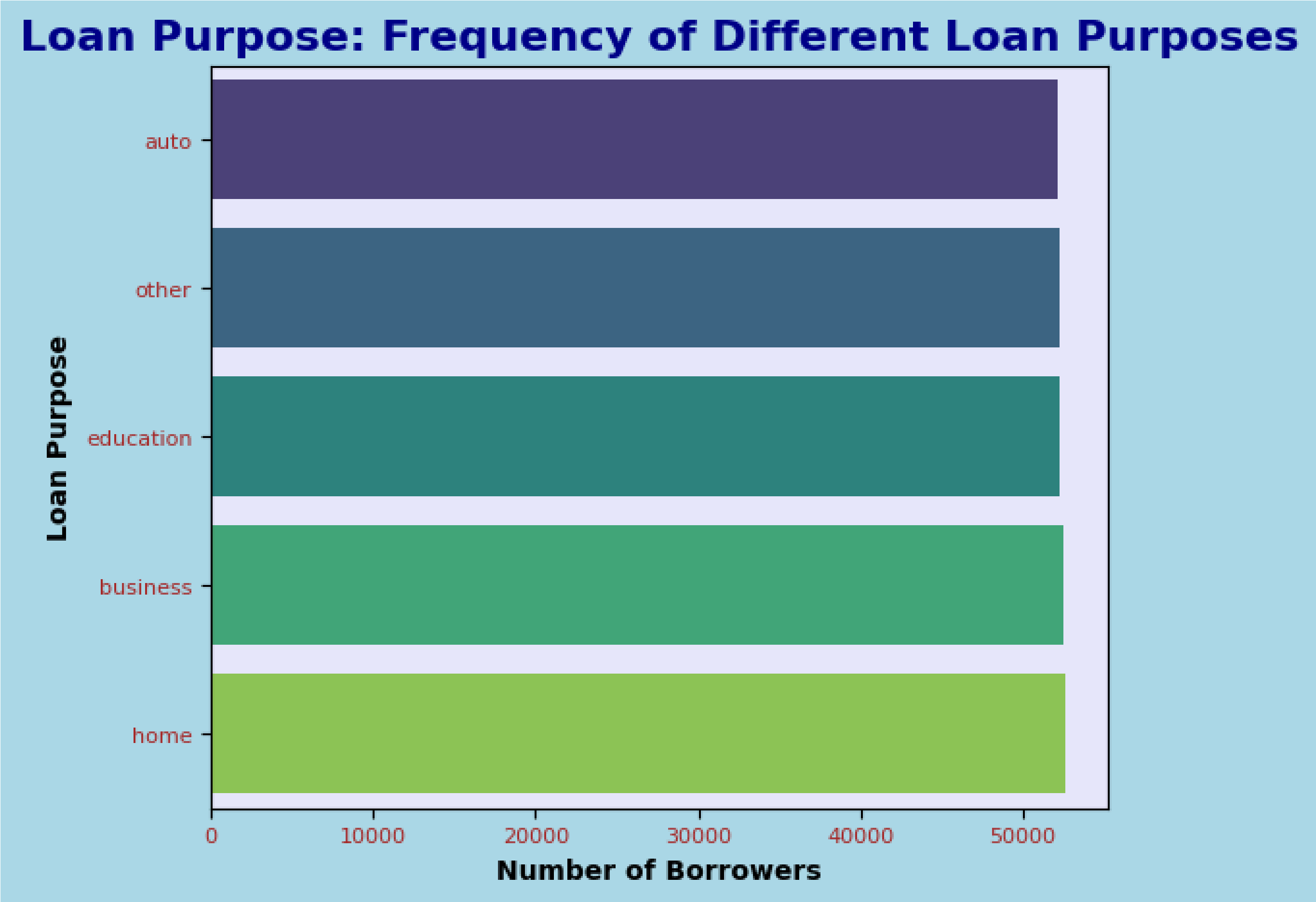
Insights

- The distribution appears fairly uniform across most of the range.
- A noticeable sharp spike occurs near 600 credit score.
- But at the 600 mark, the count jumps to above 14,000, making it an outlier cluster.

Loan Purpose

In [99]:
```
# Frequency of different loan purposes.
plt.figure(figsize=(6,5), facecolor="lightblue")
```

```python
# Bar chart for Loan Purpose
sns.countplot(
    y='LoanPurpose',
    data=df,
    order=df['LoanPurpose'].value_counts().index[::-1],
    palette="viridis"
)

# Title
plt.title("Loan Purpose: Frequency of Different Loan Purposes", fontsize=16,weight="bold", color="darkblue")

# X and Y axis labels
plt.xlabel("Number of Borrowers", fontsize=10,weight="bold", color="black")
plt.ylabel("Loan Purpose", fontsize=10,weight="bold", color="black")

# Change background inside plot
plt.gca().set_facecolor("lavender")


# Change tick labels size and color
plt.xticks(color="brown", fontsize=8)
plt.yticks(color="brown", fontsize=8)

plt.show()
```



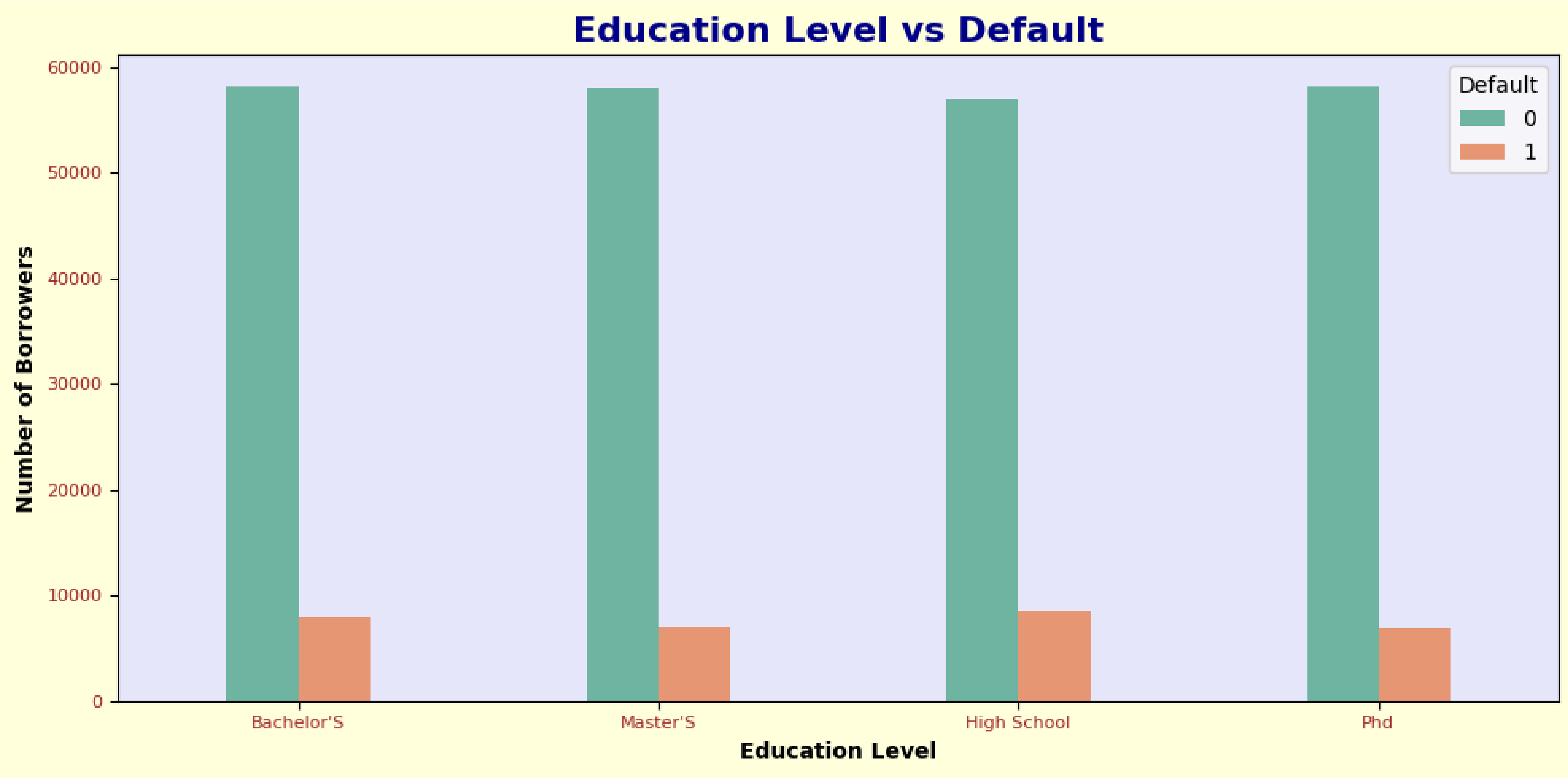
Insights

- Auto have the highest number of borrowers.
- Other category loans are also very frequent,slightly lower than auto loans.
- Business and Education loans are in middle range.
- Home loans have the lowest frequency among all categories.

✅ **Bivariate Analysis(Two Variable Analysis)**

Education vs Default

In [98]:
```
#it’s comparing how many borrowers in each education group defaulted.
plt.figure(figsize=(10,5), facecolor="lightyellow")

sns.countplot(
    x="Education",
    hue="Default",
    data=df,
    width =0.4,
    palette="Set2"
)

plt.title("Education Level vs Default",
          fontsize=16, weight="bold", color="darkblue")

plt.xlabel("Education Level", fontsize=10, weight="bold", color="black")
plt.ylabel("Number of Borrowers", fontsize=10, weight="bold", color="black")

plt.gca().set_facecolor("lavender")


plt.xticks(color="brown", fontsize=8)
plt.yticks(color="brown", fontsize=8)

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



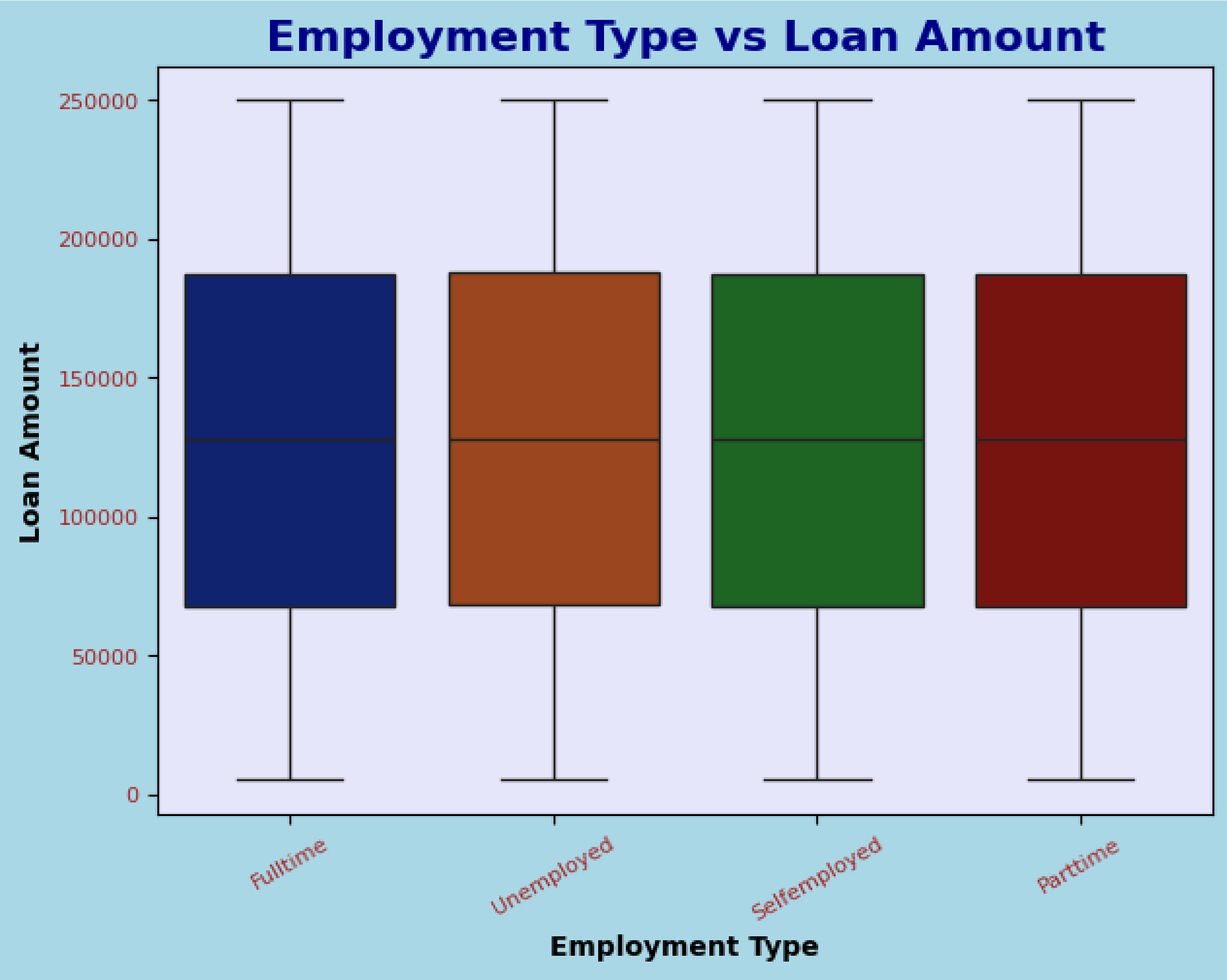
Insights

- Majority of borrowers did not default (Default = 0) across all education levels.
- Higher education levels(Bachelor's,Master's,phD) have more borrowers,they tend to take more loans.
- High school-educated borrowers may have a higher default rate than those with higher education, because of less stable jobs and low incomes.

EmploymentType vs LoanAmount

In [97]:
```
# It checks about borrower’s employment type affect the loan amount they take
plt.figure(figsize=(7,5), facecolor="lightblue")
```

```python
# Create boxplot (EmploymentType vs LoanAmount)
sns.boxplot(
    x="EmploymentType",
    y="LoanAmount",
    data=df,
    palette="dark"    )

plt.title("Employment Type vs Loan Amount", fontsize=16, weight="bold", color="darkblue")

# X and Y axis labels
plt.xlabel("Employment Type", fontsize=10, weight="bold")
plt.ylabel("Loan Amount", fontsize=10, weight="bold")

plt.gca().set_facecolor("lavender")

plt.xticks(rotation=30,color="brown",fontsize=8)
plt.yticks(color="brown",fontsize=8)


plt.show()
```

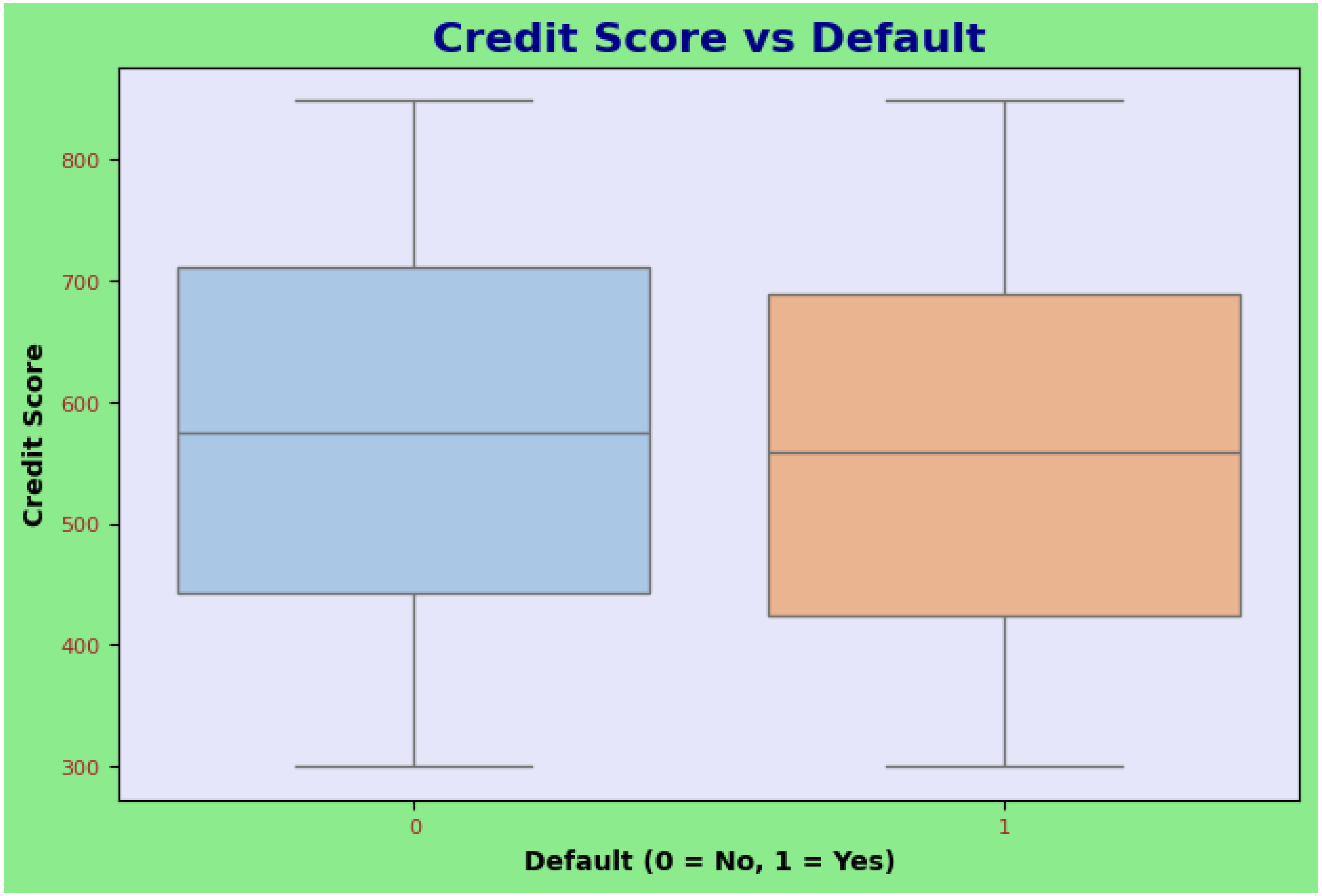


Insights

- The middle lines in the boxes are very close across all employment types.
- The boxes (interquartile range or IQR, from 25th to 75th percentile) are almost equal in size.
- The whiskers (lines extending from boxes) show that,The maximum loan amounts are nearly the same across all employment types.

CreditScore vs Default

In [96]:
```
#This is used to find out how reliable credit score is for predicting loan repayment risk.
plt.figure(figsize=(8,5), facecolor="lightgreen")

sns.boxplot(
    x="Default",
    y="CreditScore",
    data=df,
    palette="pastel"
)

plt.title("Credit Score vs Default", fontsize=16, weight="bold", color="darkblue")
plt.xlabel("Default (0 = No, 1 = Yes)", fontsize=10, weight="bold")
plt.ylabel("Credit Score", fontsize=10, weight="bold")

plt.xticks(color="brown",fontsize=8)
plt.yticks(color="brown",fontsize=8)

plt.gca().set_facecolor("lavender")
plt.show()
```



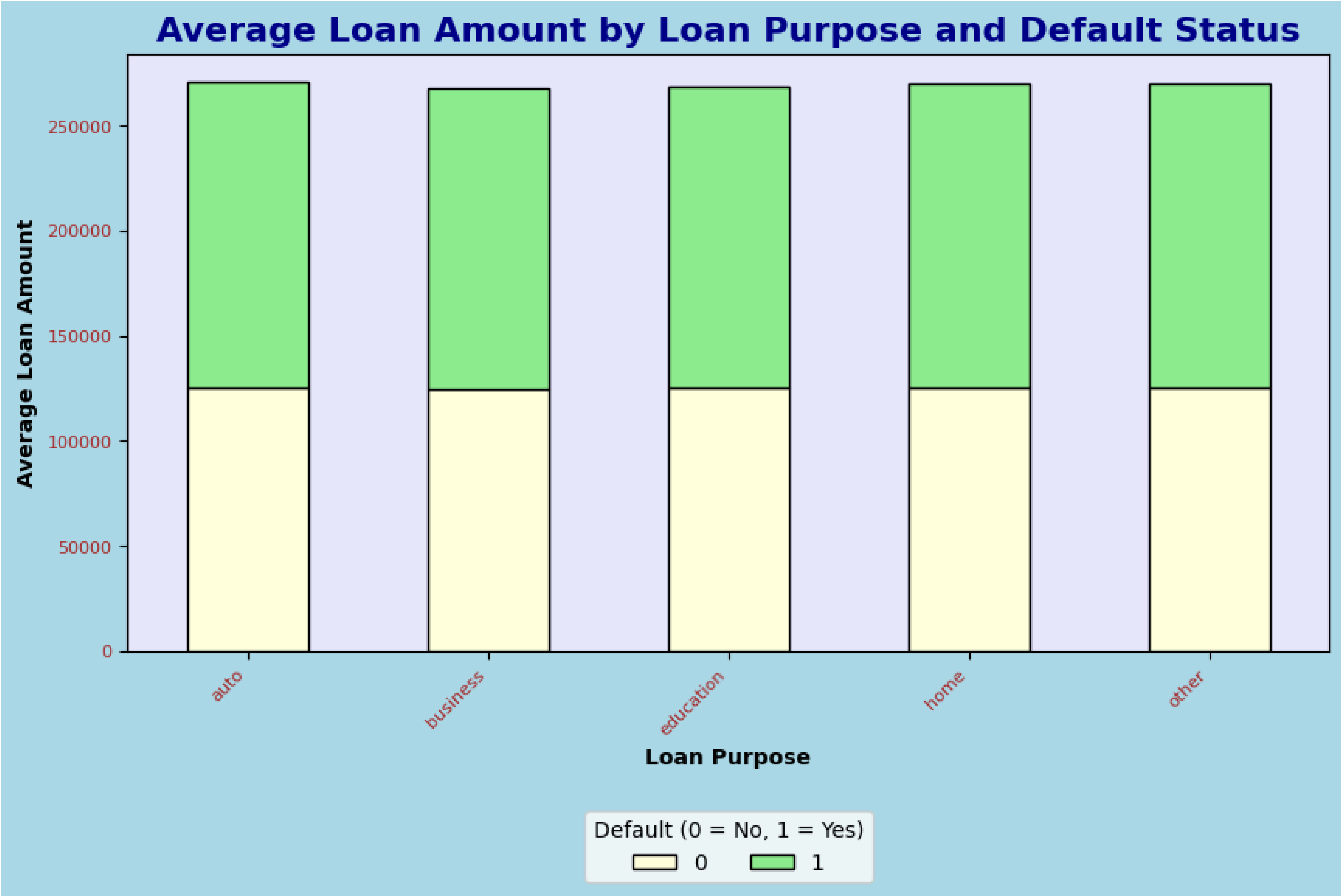
Insights

- Credit Scores are Higher for Non-Defaulters (0):The median credit score for people who did not default is higher than for those who did.
- Lower Credit Scores for Defaulters (1):The boxplot for defaulted loans (1) shows a lower median and lower quartiles, indicating: Individuals with lower credit scores are more likely to default.

✅ **Multivariate Analysis(Three or more Variable Analysis)**

LoanPurpose vs LoanAmount vs Default

In [95]:
```
# For each type of loan, what is the average loan amount, and how is it distributed between defaulters and non-default
grouped = df.groupby(["LoanPurpose", "Default"])["LoanAmount"].mean().unstack()
plt.figure(figsize=(10,5), facecolor="lightblue")

# Stacked bar chart
grouped.plot(
    kind="bar",
    stacked=True,
    color=["lightyellow", "lightgreen"],
    edgecolor="black",
    ax=plt.gca()
)


plt.title("Average Loan Amount by Loan Purpose and Default Status", fontsize=16, weight="bold", color="darkblue")
plt.xlabel("Loan Purpose", fontsize=10, weight="bold")
plt.ylabel("Average Loan Amount", fontsize=10, weight="bold")

plt.legend(
    title="Default (0 = No, 1 = Yes)",
    loc="upper center",
    bbox_to_anchor=(0.5, -0.25),
    ncol=2)


plt.gca().set_facecolor("lavender")


plt.xticks(color="brown",fontsize=8,rotation=45, ha="right")
plt.yticks(color="brown",fontsize=8)

plt.show()
```



Insights

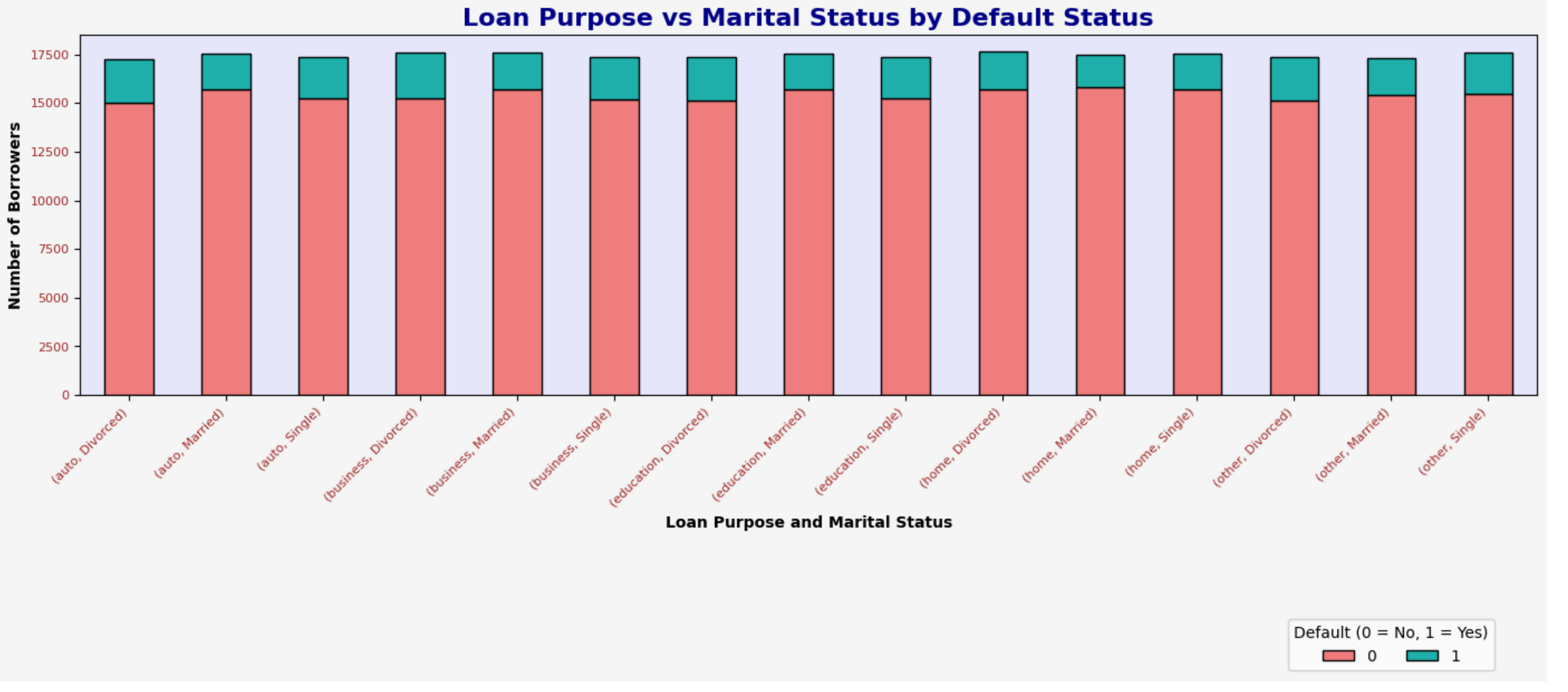
- All Loan Purposes Show Defaults.
- Business Loans Have the Highest Average Loan Amount.

- Education Loans Show Significant Defaults.
- Auto Loans Have Lower Average Loan Amounts.
- Home Loans Appear More Stable.
- Other Category is Mid-Range.

LoanPurpose vs MaritalStatus vs Default

In [94]:
```
# Helps  to see which loan purposes and marital statuses have higher default rates.
grouped = df.groupby(["LoanPurpose", "MaritalStatus", "Default"]).size().unstack()

plt.figure(figsize=(14,8), facecolor="whitesmoke")

# stacked bar chart
grouped.plot(
    kind="bar",
    stacked=True,
    color=["lightcoral", "lightseagreen"],
    edgecolor="black",
    ax=plt.gca()
)

plt.title("Loan Purpose vs Marital Status by Default Status",
          fontsize=16, weight="bold", color="darkblue")

plt.xlabel("Loan Purpose and Marital Status", fontsize=10, weight="bold")

plt.ylabel("Number of Borrowers", fontsize=10, weight="bold")

plt.legend(title="Default (0 = No, 1 = Yes)", fontsize=8)

plt.gca().set_facecolor("lavender")

plt.xticks(color="brown",fontsize=8,rotation=45, ha="right")
plt.yticks(color="brown",fontsize=8)


plt.legend(
    title="Default (0 = No, 1 = Yes)",
```

```
    loc="upper center",
    bbox_to_anchor=(0.9, -0.60),
    ncol=2)

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



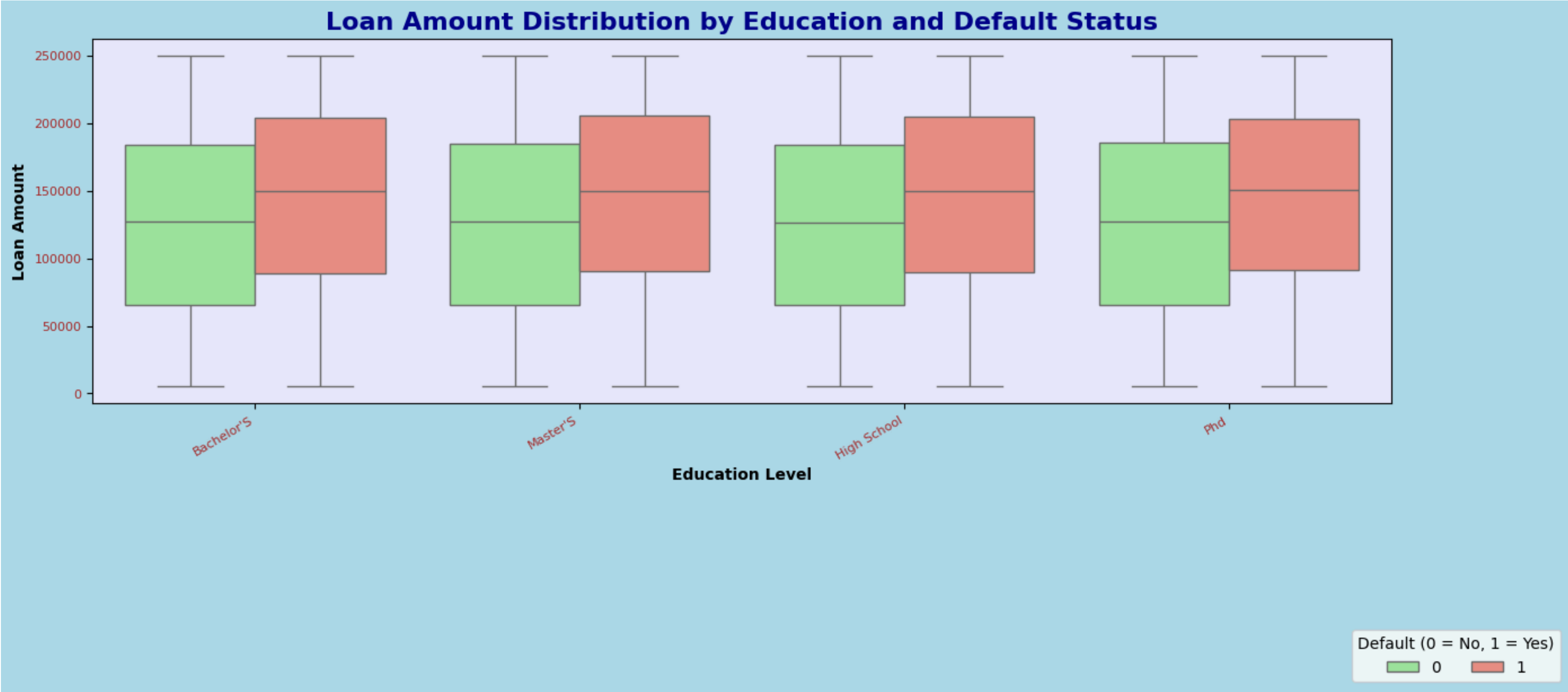
Insights

- Most borrowers do not default.
- Defaults are relatively small,The green portion (Default = 1 → Yes) is much smaller, but still present across all loan purposes and marital statuses.
- Loan purpose does not affect default rate.
- There is no major difference between married vs single borrowers within each loan purpose, Both groups show similar proportions.

- Total borrowers per category are similar,That indicates a balanced dataset where loan purpose + marital status combinations have similar sample sizes.

LoanAmount vs Education vs Default

In [93]:
```
# Shows whether education level impacts loan amounts and default.
plt.figure(figsize=(14,8), facecolor="lightblue")

sns.boxplot(
    data=df,
    x="Education",
    y="LoanAmount",
    hue="Default",
    palette={0: "lightgreen", 1: "salmon"}
)


plt.title("Loan Amount Distribution by Education and Default Status",
          fontsize=16, weight="bold", color="darkblue")

plt.xlabel("Education Level", fontsize=10, weight="bold")

plt.ylabel("Loan Amount", fontsize=10, weight="bold")

plt.xticks(color="brown",fontsize=8,rotation=30, ha="right")
plt.yticks(color="brown",fontsize=8)

plt.gca().set_facecolor("lavender")

plt.legend(
    title="Default (0 = No, 1 = Yes)",
    loc="upper center",
    bbox_to_anchor=(1.05, -0.60),
    ncol=2)

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



Insights

- Loan amount distribution is similar across education levels.
- Defaults (red) vs. Non-defaults (green) overlap a lot,The distributions for defaulted and non-defaulted loans are very similar within each education group.
- Median loan amounts are slightly higher for defaults in some groups,In a few education categories, the red box (defaults) seems shifted slightly upward compared to green (non-defaults).
- people who default may have slightly higher loan amounts on average.

➡️ Use Groupby,pivot tables and correlation analysis

✅ GroupBy

Average loan amount and default rate per loan purpose

In [70]:
```
#groupby() is used to split data into groups based on categories, and then apply functions (like mean, median, su
```

In [71]:
```
#summarizing average loan amount and default rate per loan purpose.
grouped = df.groupby(["LoanPurpose", "Default"])["LoanAmount"].mean().reset_index()
print(grouped.head(10))
```

```
  LoanPurpose  Default     LoanAmount
0        auto        0  125503.266310
1        auto        1  145251.407479
2    business        0  124901.361781
3    business        1  143218.011808
4   education        0  125562.188847
5   education        1  143192.162933
6        home        0  125675.133056
7        home        1  144665.988530
8       other        0  125394.468985
9       other        1  144320.680388
```

Median Income by Education & Default

In [72]:
```
#checks if more educated borrowers have high income,and fewer defaults.
grouped_income = df.groupby(["Education", "Default"])["Income"].median().reset_index()
print(grouped_income.head(10))
```

```
     Education  Default       Income
0   Bachelor'S        0  82492.46706
1   Bachelor'S        1  68664.00000
2  High School        0  82583.00000
3  High School        1  68308.50000
4     Master'S        0  83230.50000
5     Master'S        1  69044.00000
6          Phd        0  82492.46706
7          Phd        1  68438.00000
```

Average Debt-to-Income ratio(DTI) by MaritalStatus

In [73]:
```
#Shows whether single/married/divorced borrowers have higher debt burdens.
grouped_dti = df.groupby("MaritalStatus")["DTIRatio"].mean().sort_values(ascending=False)
print(grouped_dti)
```

```
MaritalStatus
Single      0.501448
Married     0.499850
Divorced    0.498966
Name: DTIRatio, dtype: float64
```

✅ Pivot Tables

In [74]:
```
#A pivot table is like a more flexible version of groupby().It summarizes data in a matrix format-> rows × column
```

LoanAmount by Education & Marital Status

In [75]:
```
#This pivot table reveals how education & marital status influence loan amounts.
pivot3 = pd.pivot_table(
    df,
    values="LoanAmount",
    index="Education",
    columns="MaritalStatus",
    aggfunc="mean"
)
print(pivot3)
```

```
MaritalStatus       Divorced        Married         Single
Education
Bachelor'S     128349.391174  126585.252670  127282.931928
High School    126809.346542  127010.633461  128379.654490
Master'S       127627.914873  127637.427950  127860.419582
Phd            128445.844559  127824.132804  127179.511309
```

Default Rate by Education & LoanPurpose

In [76]:
```
#checks which education groups default more often for each loan purpose.
pivot4 = pd.pivot_table(
    df,
    values="Default",
    index="Education",
    columns="LoanPurpose",
    aggfunc="mean"
)
print(pivot4)
```

```
LoanPurpose       auto  business  education      home     other
Education
Bachelor'S    0.120239  0.129342   0.124036  0.106007  0.124595
High School   0.131442  0.137676   0.128670  0.119591  0.128583
Master'S      0.111120  0.114835   0.113083  0.093888  0.111034
Phd           0.112872  0.109529   0.106504  0.090361  0.107741
```

Income vs LoanPurpose

In [77]:

```
#for each loan purpose, what will be typical borrower income.
pivot5 = pd.pivot_table(
    df,
    values="Income",
    index="LoanPurpose",
    aggfunc="mean"
)
print(pivot5)
```

```
                   Income
LoanPurpose
auto         82507.908340
business     82887.874284
education    82224.204766
home         82404.468720
other        82469.617299
```

✅ Correlation Analysis

In [78]:

```
# Correlation analysis measures the strength and direction of the linear relationship between two numerical varial
```

DTI Ratio vs Default

In [92]:

```
#This analysis is done to check if DTI ratio is an important risk indicator for loan default.
dti_default_corr = df["DTIRatio"].corr(df["Default"])

plt.figure(figsize=(14,8), facecolor="lightblue")

sns.violinplot(x="Default", y="DTIRatio", data=df,hue="Default", palette={0:"lightgreen",1:"salmon"})

plt.title("DTIRatio vs Default ",fontsize=16, weight="bold", color="darkblue")
```

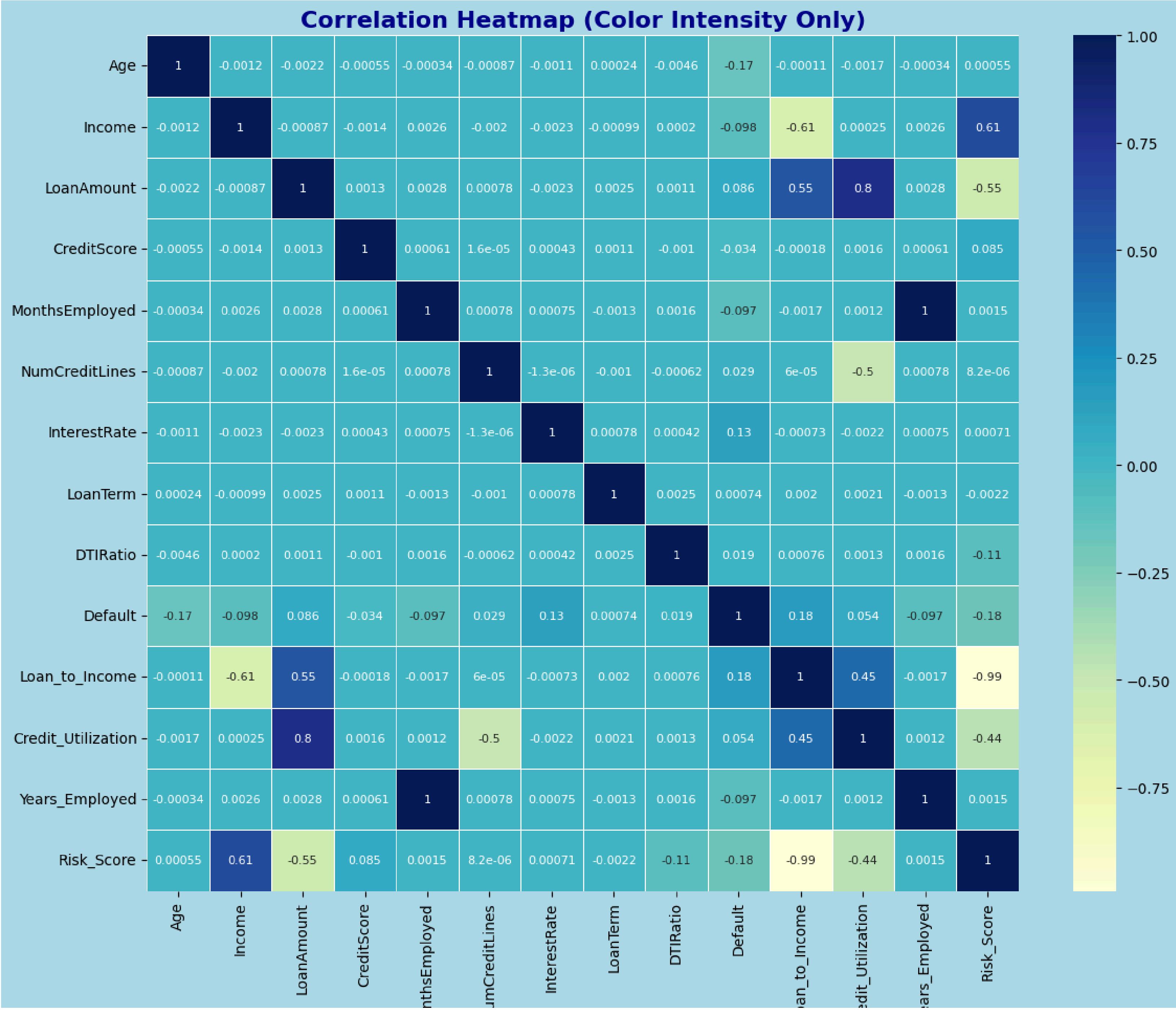
```
plt.xlabel("Default Status (0=No, 1=Yes)", fontsize=10, weight="bold")
plt.ylabel("Debt-to-Income Ratio", fontsize=10, weight="bold")

plt.xticks(color="brown", fontsize=8)
plt.yticks(color="brown", fontsize=8)

plt.gca().set_facecolor("lavender")

plt.legend(title="Default (0 = No, 1 = Yes)",
           loc="upper center", bbox_to_anchor=(1.05, -0.10), ncol=2)

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

Insights

- DTI ratio distribution is very similar for both groups,Both defaulted and non-defaulted borrowers have almost identical shaped violin plots.
- Most borrowers have moderate DTI ratios,The widest portion of both violins is around the middle , most borrowers fall within this range.
- Green violin plot is wider,more non-defaulters at those DTI values,Red violin plot is narrower with fewer defaulters, more evenly spread DTI values.

Heatmap with colour intensity

In [91]:
```
#This analysis is used to understand what increases or decreases the chance of loan default and to decide which


numeric_df = df.select_dtypes(include=["int64","float64"])

# Create heatmap
plt.figure(figsize=(12,10), facecolor="lightblue")
sns.heatmap(numeric_df.corr(), cmap="YlGnBu", cbar=True, annot=True,annot_kws={"size":8}, linewidths=0.5, lineco


plt.title("Correlation Heatmap (Color Intensity Only)", fontsize=16, weight="bold", color="darkblue")

plt.gca().set_facecolor("lavender")


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


Correlation Heatmap (Color Intensity Only)
Age 1 -0.0012 -0.0022 -0.00055 -0.00034 -0.00087 -0.0011 0.00024 -0.0046 -0.17 -0.00011 -0.0017 -0.00034 0.00055
Income -0.0012 1 -0.00087 -0.0014 0.0026 -0.002 -0.0023 -0.00099 0.0002 -0.098 -0.61 0.00025 0.0026 0.61
LoanAmount -0.0022 -0.00087 1 0.0013 0.0028 0.00078 -0.0023 0.0025 0.0011 0.086 0.55 0.8 0.0028 -0.55
CreditScore -0.00055 -0.0014 0.0013 1 0.00061 1.6e-05 0.00043 0.0011 -0.001 -0.034 -0.00018 0.0016 0.00061 0.085
MonthsEmployed -0.00034 0.0026 0.0028 0.00061 1 0.00078 0.00075 -0.0013 0.0016 -0.097 -0.0017 0.0012 1 0.0015
NumCreditLines -0.00087 -0.002 0.00078 1.6e-05 0.00078 1 -1.3e-06 -0.001 -0.00062 0.029 6e-05 -0.5 0.00078 8.2e-06
InterestRate -0.0011 -0.0023 -0.0023 0.00043 0.00075 -1.3e-06 1 0.00078 0.00042 0.13 -0.00073 -0.0022 0.00075 0.00071
LoanTerm 0.00024 -0.00099 0.0025 0.0011 -0.0013 -0.001 0.00078 1 0.0025 0.00074 0.002 0.0021 -0.0013 -0.0022
DTIRatio -0.0046 0.0002 0.0011 -0.001 0.0016 -0.00062 0.00042 0.0025 1 0.019 0.00076 0.0013 0.0016 -0.11
Default -0.17 -0.098 0.086 -0.034 -0.097 0.029 0.13 0.00074 0.019 1 0.18 0.054 -0.097 -0.18
Loan_to_Income -0.00011 -0.61 0.55 -0.00018 -0.0017 6e-05 -0.00073 0.002 0.00076 0.18 1 0.45 -0.0017 -0.99
Credit_Utilization -0.0017 0.00025 0.8 0.0016 0.0012 -0.5 -0.0022 0.0021 0.0013 0.054 0.45 1 0.0012 -0.44
Years_Employed -0.00034 0.0026 0.0028 0.00061 1 0.00078 0.00075 -0.0013 0.0016 -0.097 -0.0017 0.0012 1 0.0015
Risk_Score 0.00055 0.61 -0.55 0.085 0.0015 8.2e-06 0.00071 -0.0022 -0.11 -0.18 -0.99 -0.44 0.0015 1
Age Income LoanAmount CreditScore nthsEmployed umCreditLines InterestRate LoanTerm DTIRatio Default an_to_Income edit_Utilization ears_Employed Risk_Score
1.00 0.75 0.50 0.25 0.00 −0.25 −0.50 −0.75

Insights

- Strong positive correlations-LoanAmount vs Loan_to_Income,Income vs Risk_Score.
- Strong negative correlation- Default vs Risk_Score,risk,Risk_Score vs Credit_Utilization.

➡️ Include Statistical summaries to support findings

In [81]:
```
df.describe()
```

Out[81]:

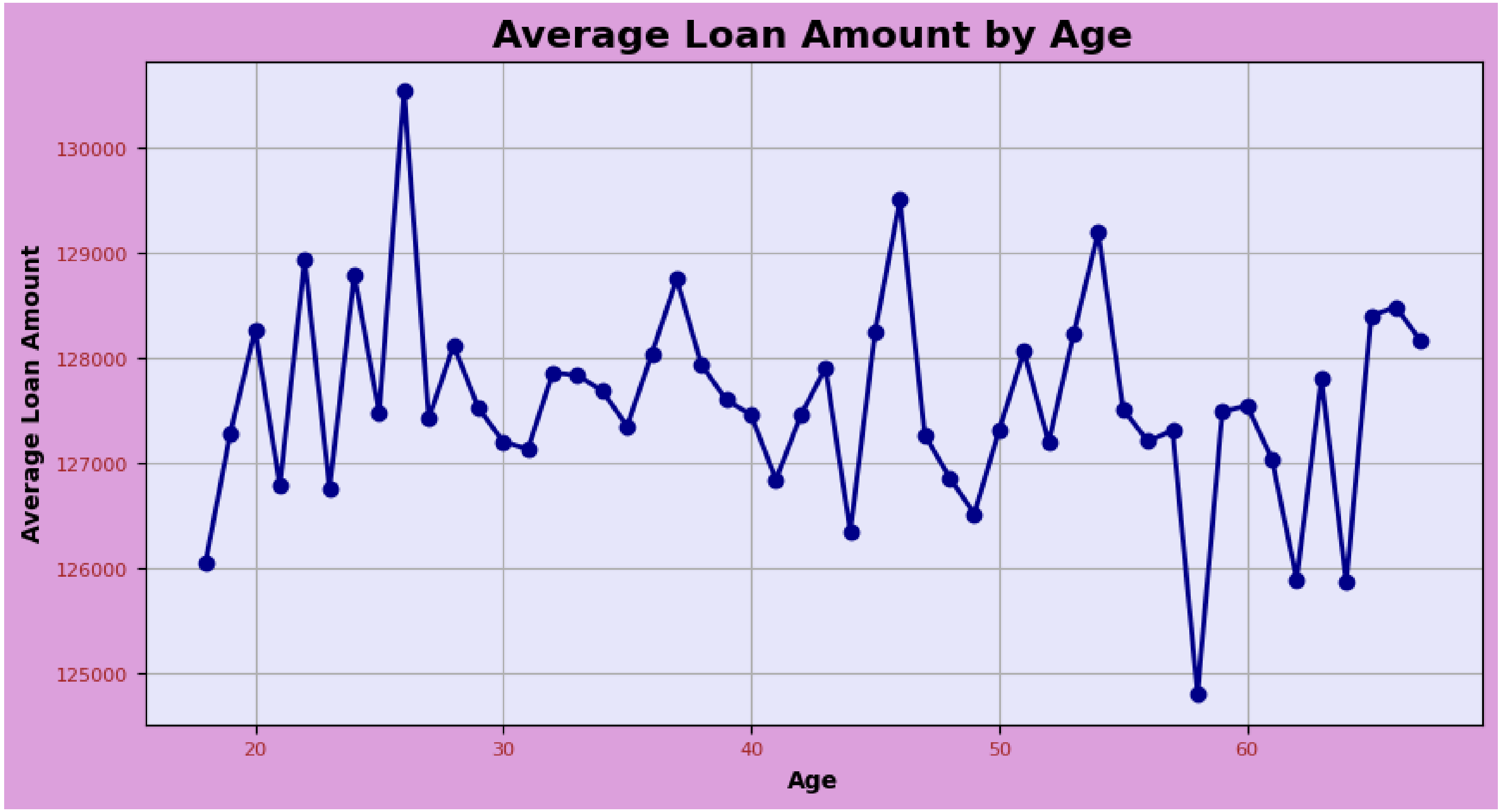
	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	Lo
count	261731.0	261731.000000	261731.000000	261731.000000	261731.000000	261731.000000	261731.000000	26173
mean	43.486152	82499.137820	127579.356307	574.267120	59.547948	2.500835	13.495161	36
std	14.806512	38484.896118	69971.415387	156.953948	34.640943	1.117029	6.636820	16
min	18.0	15000.000000	5000.000000	300.000000	0.000000	1.000000	2.000000	12
25%	31.0	49638.000000	67645.000000	440.000000	30.000000	2.000000	7.770000	24
50%	43.0	82492.467060	127598.987104	574.378064	60.000000	2.000000	13.460000	36
75%	56.0	115394.500000	187464.000000	708.000000	90.000000	3.000000	19.250000	48
max	69.0	149999.000000	249999.000000	849.000000	119.000000	4.000000	25.000000	60

4. Visualisations

➡️ Use Matplotlib/Seaborn/Plotly to generate meaningful visualisations:

Average LoanAmount by Age

In [90]:
```
#Taking average, how much loan do people of each age take.
avg_loan_by_age = df.groupby("Age")["LoanAmount"].mean()


avg_loan_by_age = avg_loan_by_age.head(50)

 #Create the line chart
plt.figure(figsize=(10,5),facecolor="plum")
avg_loan_by_age.plot(
    kind="line",
    marker="o",
    linewidth=2,
    color="darkblue"
)


plt.title("Average Loan Amount by Age", fontsize=16, weight="bold")
plt.gca().set_facecolor("lavender")
plt.xlabel("Age",fontsize=10,color="black",weight="bold")
plt.ylabel("Average Loan Amount",fontsize=10,color="black",weight="bold")
plt.xticks(color="brown", fontsize=8)
plt.yticks(color="brown", fontsize=8)

plt.grid(True)


plt.show()
```



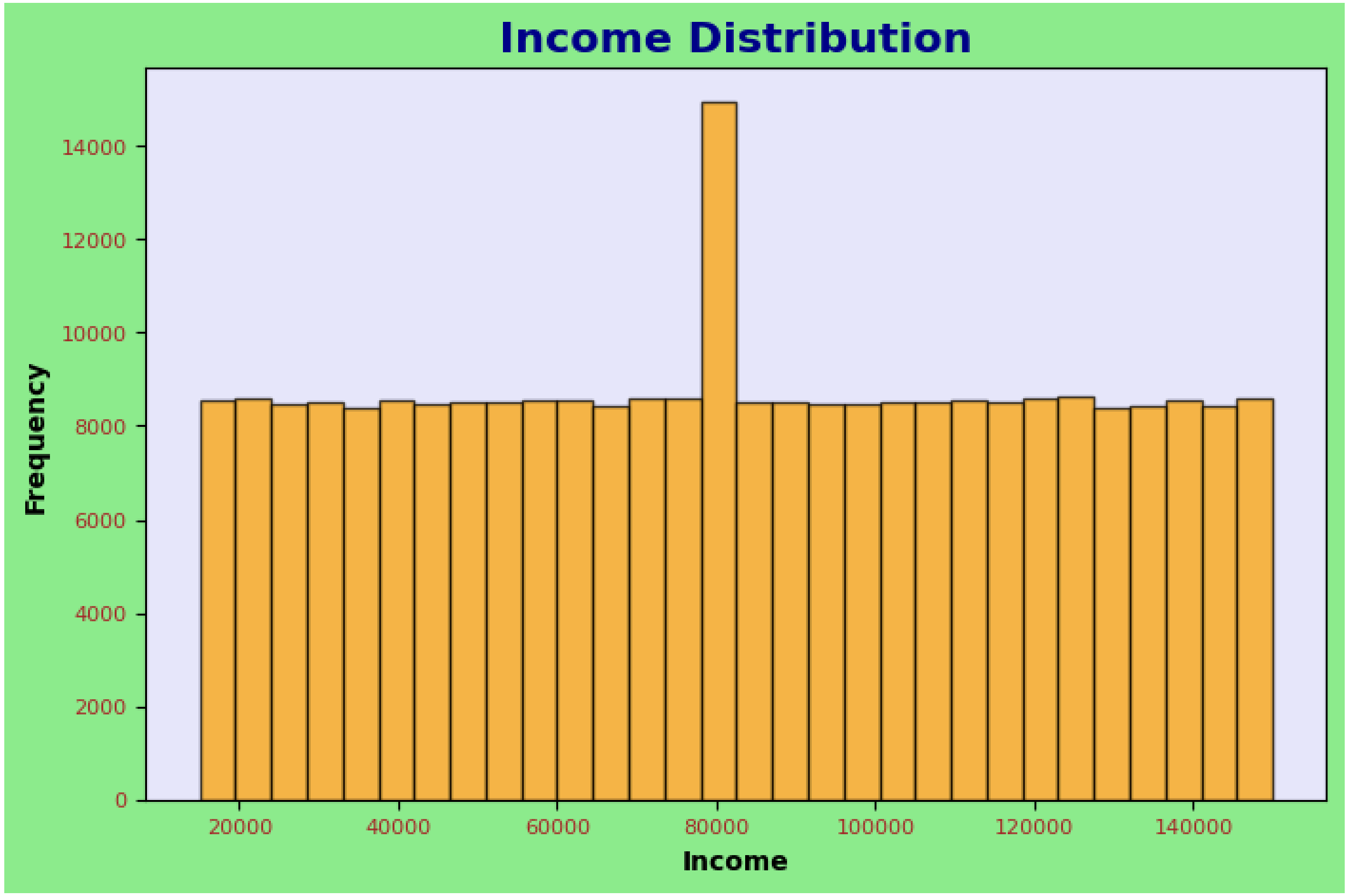
Insights

- The line goes up and down, average loan amount does not increase or decrease consistently.
- A spike around age ~22, and a drop around age ~60.

Income Distribution

In [89]:
```
#how incomes are spread out (who earns less, who earns more, and where most people lie).

plt.figure(figsize=(8,5),facecolor="lightgreen")
plt.hist(
    df["Income"],
```

```
    bins=30,
    color="orange",
    edgecolor="black",
    alpha=0.7
)


plt.title("Income Distribution",color="darkblue", fontsize=16, weight="bold")
plt.gca().set_facecolor("lavender")
plt.xlabel("Income",color="black",weight="bold",fontsize="10")
plt.ylabel("Frequency",color="black",weight="bold",fontsize="10")
plt.xticks(color="brown",fontsize=8)
plt.yticks(color="brown",fontsize=8)

plt.show()
```



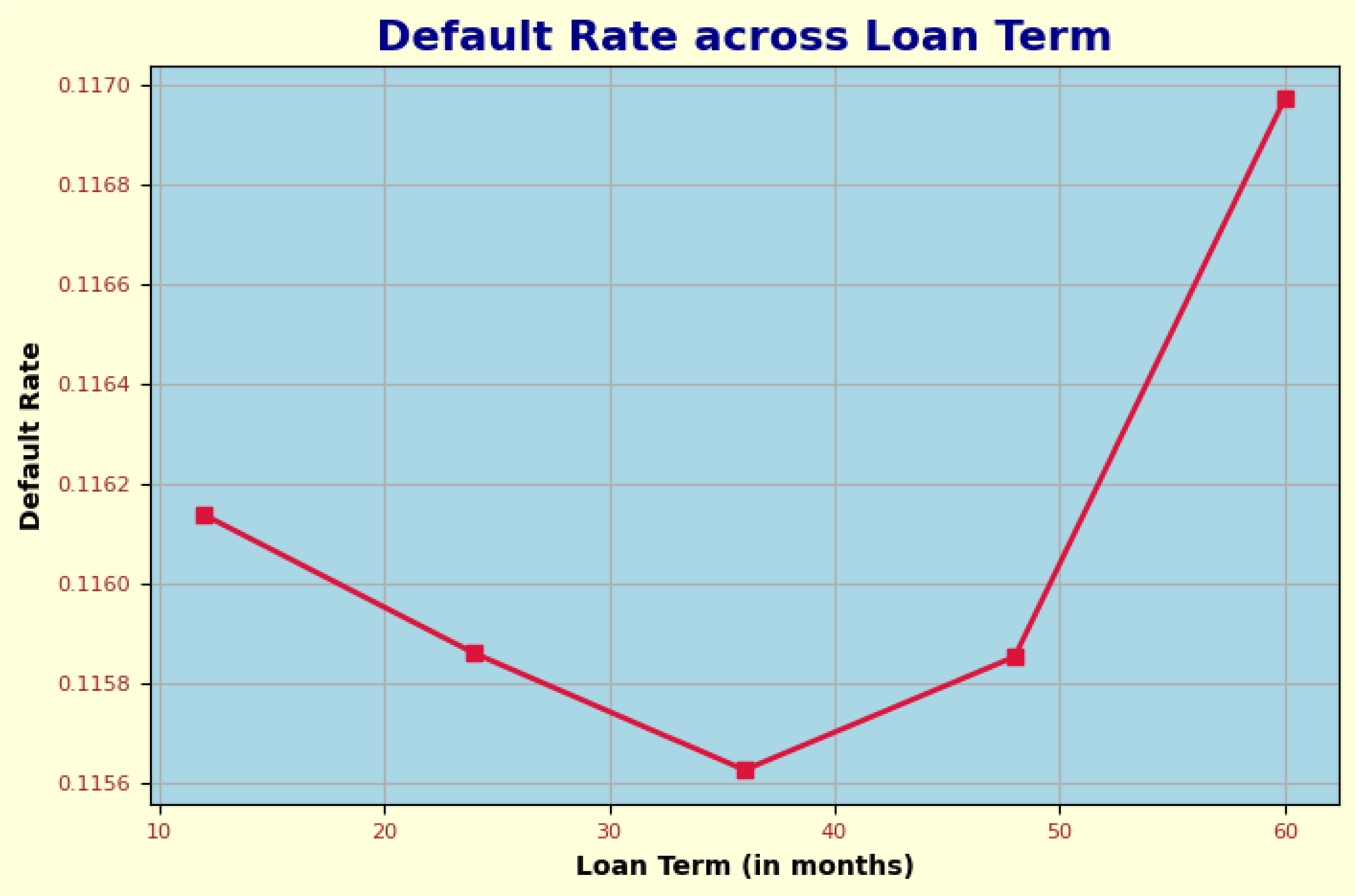
Insights

- Mostly uniform distribution, and an unusual spike at 80,000.

Default rate by Loan term

In [88]:
```
#checks if  the loanterm is longer or shorter, does people default more or less.

default_rate_term = df.groupby("LoanTerm")["Default"].mean()

#  Create the line chart
```

```
plt.figure(figsize=(8,5),facecolor="lightyellow")
default_rate_term.plot(
    kind="line",
    marker="s",
    linewidth=2,
    color="crimson"
)

plt.title("Default Rate across Loan Term", fontsize=16,color="darkblue", weight="bold")
plt.gca().set_facecolor("lightblue")
plt.xlabel("Loan Term (in months)",fontsize=10,weight="bold")
plt.ylabel("Default Rate",fontsize=10,weight="bold")
plt.xticks(color="brown",fontsize=8)
plt.yticks(color="brown",fontsize=8)

plt.grid(True)

plt.show()
```

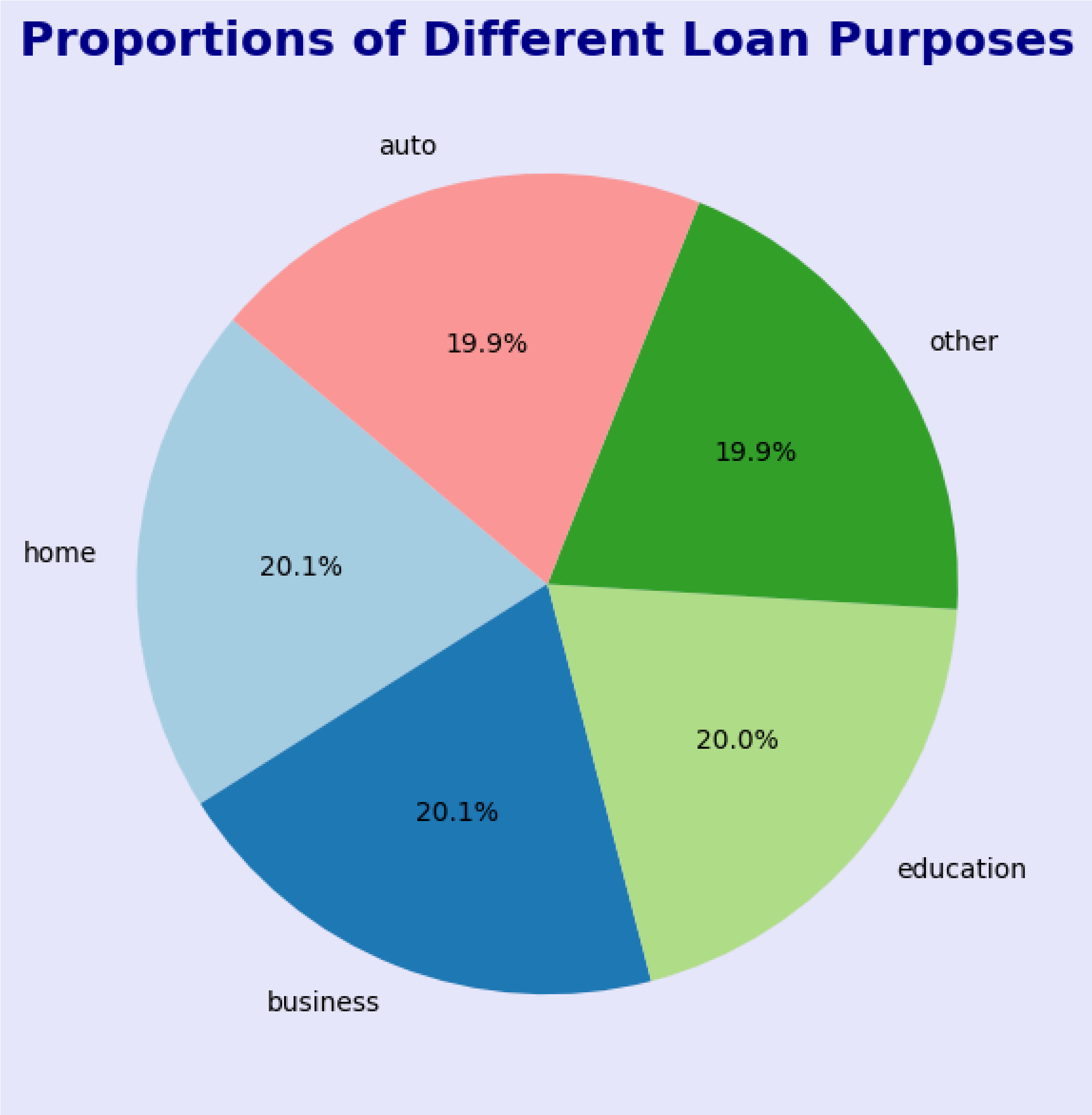


Insights

- Short loans(10-20 months):slightly higher chance of default.
- Medium loans(30-40 months):safest,lowest default rate.
- Long loans(60 months):Borrowers are much more likely to default, longer repayment increases financial stress or uncertainty.

Loan Purpose

In [87]:
```
# Count the frequency of each Loan Purpose
loanpurpose_counts = df["LoanPurpose"].value_counts()

# Create the pie chart
plt.figure(figsize=(7,7),facecolor="lavender")
plt.pie(loanpurpose_counts,
        labels=loanpurpose_counts.index,
        autopct="%1.1f%%",
        startangle=140,
        colors=plt.cm.Paired.colors)

plt.title("Proportions of Different Loan Purposes",color="darkblue", fontsize=18, weight="bold")
plt.show()
```



Insights

- All four categories are almost equally distributed.
- Home and Business have the highest share of 20.1%.

- Auto,Education and Other are slightly lower.

◆ Insights

- People with low credit scores are more likely to default.
- Borrowers who take a big loan compared to their income default more often.
- Borrowers with stable jobs (longer employment) are less likely to default.
- Having too many credit lines increases the chance of default.
- Younger borrowers (20–30 age group) tend to default more often than older borrowers.
- Defaults cluster in lower income groups → borrowers with small salaries but high loan requests struggle to repay.
- High risk borrowers → young, new to jobs, low income, low credit score, many loans, risky loan purpose
- Low risk borrowers → older, stable jobs, good income, high credit score, fewer loans, productive loan purpose.

◆ Recommendations

- Provide loans mainly to people with good credit scores (above 650).
- Avoid approving loans that are too large compared to income.
- Mostly Prefer borrowers with stable jobs or ask new employees to get a co-signer.
- Make sure to check how many loans/credit lines a borrower already has before giving a new one.
- Apply stricter rules for younger borrowers like smaller loans, co-signer.

- Encourage borrowers with short employment history to build stability before taking big loans.
- Create income-based loan limit,by not allowing to take loans more than a certain percentage of income.
- Use Credit Score & Loan-to-Income ratio as the main approval criteria.
- Build a risk score system by combining credit score, income, loan purpose, employment length and then approve loans only if the risk score is acceptable.

****The End****

In [ ]: