BANK LOAN ANALYSIS REPORT

Import Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
%matplotlib inline
%matplotlib notebook
```

```
df2=pd.read_excel('/content/drive/MyDrive/Colab Notebooks/Internship/financial_loan.xlsx')
```

```
df.head()
```

	id	address_state	application_type	emp_length	emp_title	grade	home_ownership	issue_date	last_credit_pull_date	last_payment_date	loan_status	purpose	term	verification_status	annual_income	dti	installment	int_rate	total_acc	total_pymnt	total_rec_int	total_rec_late_fee
0	1077501	GA	INDIVIDUAL	1 year	Bucher	C	RENT	2021-03-15	2021-06-15	2021-06-15	Fully Paid	other	36 months	Source Verified	55000.0	12.9100	183.60	5.7027	29.0	1209	209.60	0.0
1	1070922	CA	INDIVIDUAL	9 years	MRC Accounting	C	RENT	2021-02-14	2021-07-15	2021-07-15	Current	car	36 months	Source Verified	56000.0	15.9500	196.65	5.7884	3000	4	3936	0.0
2	1070921	CA	INDIVIDUAL	9 years	Sales Representative	C	RENT	2021-02-14	2021-07-15	2021-07-15	Current	other	36 months	Source Verified	60000.0	17.9700	197.79	5.7884	3000	4	3957.58	0.0
3	1067704	CA	INDIVIDUAL	9 years	Sales Manager	B	MORTGAGE	2021-02-12	2021-06-12	2021-06-12	Fully Paid	other	36 months	Source Verified	60000.0	12.8900	201.12	5.4884	3000	4	4021	0.0
4	1065002	IL	INDIVIDUAL	10+ years	Ibc Steel Inc.	B	MORTGAGE	2021-02-11	2021-12-14	2021-12-15	Current	other	36 months	Verified	55000.0	10.0201	136.05	5.8021	3000	20	3630	0.0

5 rows × 26 columns

```
df.tail()
```

	id	address_state	application_type	emp_length	emp_title	grade	home_ownership	issue_date	last_credit_pull_date	last_payment_date	loan_status	purpose	term	verification_status	annual_income	dti	installment	int_rate	total_acc	total_pymnt	total_rec_int	total_rec_late_fee
38651	803443	TN	INDIVIDUAL	7 years	Health Net Solutions	C	MORTGAGE	2021-06-15	2021-12-15	2021-12-15	Fully Paid	other	36 months	Verified	150000.0	17.1899	441.08	5.1208	24.0	15	3168	0.0
38652	803277	LA	INDIVIDUAL	5 years	Gas Group	C	RENT	2021-06-15	2021-06-15	2021-06-15	Fully Paid	other	36 months	Verified	55000.0	23.8499	379.79	5.1208	29.0	13	2479.9	0.0
38653	803274	LA	INDIVIDUAL	5 years	Health Care Partners	C	RENT	2021-06-15	2021-06-15	2021-06-15	Fully Paid	other	36 months	Verified	65000.0	17.4700	427.06	5.1208	29.0	15	3037	0.0
38654	803087	NY	INDIVIDUAL	5 years	Andrew Rankings	B	RENT	2021-06-15	2021-06-15	2021-06-15	Fully Paid	other	36 months	Verified	66000.0	13.1899	672.72	5.1208	24.0	5	3967	0.0
38655	802862	NY	INDIVIDUAL	5 years	Allen Edwards	F	RENT	2021-06-15	2021-06-15	2021-06-15	Fully Paid	other	36 months	Verified	60000.0	0.3800	448.88	5.1208	18.0	7	2767.9	0.0

5 rows × 26 columns

Metadata of data

```
print("No of Rows: ",df.shape[0])
```

```
No of Rows: 38655
```

```
print("No of Columns: ",df.shape[1])
```

```
No of Columns: 26
```

```
df.info()
```

```
<class'pandas.core.frame.DataFrame'>
RangeIndex:38655 entries,0 to 38654
Data columns (total 26 columns):
 #   Column                  Non-Nullcount  Dtype
---  ------                  --------------  -----
 0   id                      38655 non-null  int64
 1   address_state           38655 non-null  object
 2   application_type        38655 non-null  object
 3   emp_length              38655 non-null  object
 4   emp_title               38655 non-null  object
 5   grade                   38655 non-null  object
 6   home_ownership          38655 non-null  object
 7   issue_date              38655 non-null  datetime64[ns]
 8   last_credit_pull_date   38655 non-null  datetime64[ns]
 9   last_payment_date       38655 non-null  datetime64[ns]
 10  loan_status             38655 non-null  object
 11  purpose                 38655 non-null  object
 12  term                    38655 non-null  object
 13  verification_status     38655 non-null  object
 14  annual_income           38655 non-null  float64
 15  dti                     38655 non-null  float64
 16  installment             38655 non-null  float64
 17  int_rate                38655 non-null  float64
 18  total_acc               38655 non-null  int64
 19  total_pymnt             38655 non-null  int64
 20  total_rec_int           38655 non-null  int64
 21  total_rec_late_fee      38655 non-null  int64
dtypes:datetime64[ns]:(3),float64:(7),int64:(5),object:(11)
memory usage:6.8+MB
```

Data Types

```
df.dtypes
```

```
id                                int64
address_state                    object
application_type                 object
emp_length                       object
emp_title                        object
grade                            object
home_ownership                   object
issue_date               datetime64[ns]
last_credit_pull_date    datetime64[ns]
last_payment_date        datetime64[ns]
loan_status                      object
purpose                          object
term                             object
verification_status              object
annual_income                   float64
dti                             float64
installment                     float64
int_rate                        float64
total_acc                         int64
total_pymnt                       int64
total_rec_int                     int64
total_rec_late_fee                int64
dtype:object
```

```
df.sample(5)
```

	id	issue_date	last_credit_pull_date	last_payment_date	next_payment_date	member_id	annual_income	dti	installment	int_rate	total_acc	total_pymnt	total_rec_prncp
0	1077501	2021-03-15	2021-06-15	2021-06-15	NaN	1296599	55000.0	12.9100	183.60	0.057027	29.0	1209.0	1000.0
1	1070922	2021-02-14	2021-07-15	2021-07-15	2021-08-15	1314167	56000.0	15.9500	196.65	0.057884	3000.0	3936.65	2500.00
2	1070921	2021-02-14	2021-07-15	2021-07-15	2021-08-15	1313524	60000.0	17.9700	197.79	0.057884	3000.0	3957.58	2500.00
3	1067704	2021-02-12	2021-06-12	2021-06-12	NaN	1309779	60000.0	12.8900	201.12	0.054884	3000.0	4021.0	3000.00
4	1065002	2021-02-11	2021-12-14	2021-12-15	2021-12-15	1306957	55000.0	10.0201	136.05	0.058021	3000.0	3630.0	3000.00

Total Loan Applications

```
total_loan_applications = df['id'].count()
print("Total Loan Applications: ",total_loan_applications)
```

```
Total Loan Applications: 38655
```

MTD Total Loan Applications

```
lastmonth_loan_applications = df[df.issue_date.dt.month==12].shape[0]
latest_year = latest_issue_date.year
print("MTD Total Loan Applications: ",lastmonth_loan_applications)
```

```
MTD Total Loan Applications: 4576
```

```
mtd_loan_applications = mtd_data[['id'].count()
```

```
print("PMTD Total Loan Applications: ",lastmonth_loan_applications-mtd_loan_applications)
```

```
PMTD Total Loan Applications: -4538
```

Total Funded Amount

```
total_funded_amount=df.loan_amount.sum()
print("Total Funded Amount: $",round(total_funded_amount/1000000
print("Total Funded Amount: $11.23M", round(total_funded_amount/1000000,
```

```
Total Funded Amount: $11.23M
```

MTD Total Funded Amount

```
latest_year = latest_issue_date.year
latest_month = latest_issue_date.month
latest_year = latest_issue_date.year
latest_month = latest_issue_date.month
```

```
mtd_data = df[df[issue_date].dt.year==latest_year & df[issue_date].dt.month == latest_month]
```

```
mtd_funded_amount = mtd_data['loan_amount'].sum()
print("MTD Total Funded Amount: $",round(mtd_funded_amount/1000000
print("MTD Total Funded Amount: $1.23M", round(mtd_funded_amount/1000000,
```

```
MTD Total Funded Amount: $1.23M
```

Total Amount Received

```
total_amount_received=df['total_pymnt'].sum()
print("Total Amount Received: $",round(total_amount_received/1000000
print("Total Amount Received: $12.57M", round(total_amount_received/1000000,
```

```
Total Amount Received: $12.57M
```

MTD Total Amount Received

```
latest_year = latest_issue_date.year
latest_month = latest_issue_date.month
latest_year = latest_issue_date.year
latest_month = latest_issue_date.month
```

```
mtd_data = df[df[issue_date].dt.year==latest_year & df[issue_date].dt.month == latest_month]
```

```
mtd_total_amount_received = mtd_data['total_pymnt'].sum()
print("MTD Total Amount Received: $",round(mtd_total_amount_received/1000000
print("MTD Total Funded Amount: $1.23M", round(mtd_total_amount_received/1000000,
```

```
MTD Total Funded Amount: $0.29M
```

Average Interest rate

```
average_interest_rate = df['int_rate'].mean()*100
print("The Avg Interest rate: ",round(average_interest_rate)
```

```
The Avg Interest rate: 12
```

Average Debt-to-Income Ratio(DTI)

```
average_dti = df['dti'].mean()*100
print("The Avg DTI: ",round(average_dti)
```

```
The Avg DTI: 1307
```

Good Loan Metrics

```
good_loans = df[df['loan_status'].isin(['Fully Paid','Current'])] # Good Loans
```

```
total_good_applications = df['id'].count()
```

```
good_loan_applications = good_loans['id'].count()
good_loan_percentage = (good_loan_applications/total_loan_applications)*100
good_loan_received = good_loans['total_pymnt'].sum()
```

```
good_loan_funded_amount_billions = good_loan_funded_amount / 1000000
good_loan_received_billions = good_loan_received / 1000000
```

```
good_loan_percentage = round(good_loan_percentage,2)
good_loan_received_billions = round(good_loan_received_billions,2)
```

```
print("Good Loan Applications",good_loan_applications)
print("Good Loan Funded Amount in Millions: $",round(good_loan_funded_amount_billions)
print("Good Loan Received Amount in Millions: $",round(good_loan_received_billions)
print("Percentage of Good Loan Applications: ",round(good_loan_percentage))
```

```
Good Loan Applications 35386
Good Loan Funded Amount in Millions: $19.0M
Good Loan Received Amount in Millions: $19.7M
Percentage of Good Loan Applications: 92.0
```

Bad Loan Metrices

```
bad_loans = df[df['loan_status'].isin(['Charged Off'])] # Bad Loans
```

```
total_bad_applications = df['id'].count()
```

```
bad_loan_applications = bad_loans['id'].count()
bad_loan_percentage = (bad_loan_applications/total_loan_applications)*100
bad_loan_received = bad_loans['total_pymnt'].sum()
```

```
bad_loan_funded_amount_billions = bad_loan_funded_amount / 1000000
bad_loan_received_billions = bad_loan_received / 1000000
```

```
bad_loan_percentage = round(bad_loan_applications / total_loan_applications) * 100
```

```
print("Bad Loan Applications",bad_loan_applications)
print("Bad Loan Funded Amount in Millions: $",round(bad_loan_funded_amount_billions)
print("Bad Loan Received Amount in Millions: $",round(bad_loan_received_billions)
print("Percentage of Bad Loan Applications: ",round(bad_loan_percentage))
```

```
Bad Loan Applications 3269
Bad Loan Funded Amount in Millions: $0.3M
Bad Loan Received Amount in Millions: $0.3M
Percentage of Bad Loan Applications: 8.0
```

Monthly Trends by Issue Date for Total Funded Amount

```
monthly_trends=df
df.set_index('issue_date')
monthly_trends.groupby(monthly_trends['issue_date'].dt.strftime('%Y-%')
.agg({'loan_amount':'sum'})
.reset_index()
.sort_values(
'issue_date')
monthly_trends.head(5)
```

	issue_date	loan_amount
0	2021-01	1154350
1	2021-02	1772475
2	2021-03	2290250
3	2021-04	2475000
4	2021-05	2965000

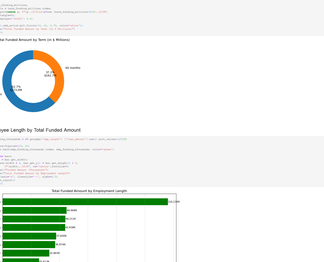
```
plt.figure(figsize=(5,5))
plt.plot(monthly_trends['issue_date'], monthly_trends['loan_amount'],marker='o')
plt.title('Total Funded Amount by Month')
plt.xlabel('Month')
plt.ylabel('Total Funded Amount')
plt.show()
```

Monthly Trends by Issue Date for Total Funded Amount Received

```
monthly_received=df
df.set_index('issue_date')
monthly_received.groupby(monthly_received['issue_date'].dt.strftime('%Y-%')
.agg({'total_pymnt':'sum'})
.reset_index()
.sort_values(
'issue_date')
monthly_received.head(5)
```

	issue_date	total_pymnt
0	2021-01	1154350
1	2021-02	1772475
2	2021-03	2290250
3	2021-04	2475000
4	2021-05	2965000

```
plt.figure(figsize=(5,5))
plt.plot(monthly_received['issue_date'], monthly_received['total_pymnt'],marker='o')
plt.title('Total Received Amount by Month')
plt.xlabel('Month')
plt.ylabel('Total Received Amount')
plt.show()
```

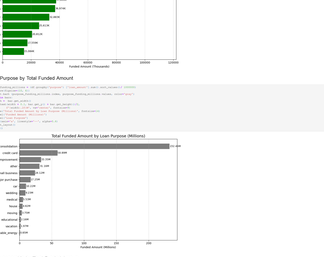


Monthly Trends by Issue Date for Total Loan applications

```
monthly_applications=df
df.set_index('issue_date')
monthly_loan_applications.groupby(monthly_loan_applications['issue_date'].dt.strftime('%Y-%')
.agg({'id':'count'})
.reset_index()
.sort_values(
'issue_date')
monthly_loan_applications.head(5)
```

	issue_date	id
0	2021-01	2290
1	2021-02	3516
2	2021-03	4544
3	2021-04	4910
4	2021-05	5880

```
plt.figure(figsize=(5,5))
plt.plot(monthly_loan_applications['issue_date'], monthly_loan_applications['id'],marker='o')
plt.title('Total Loan Applications by Month')
plt.xlabel('Month')
plt.ylabel('Total Loan Applications')
plt.show
```

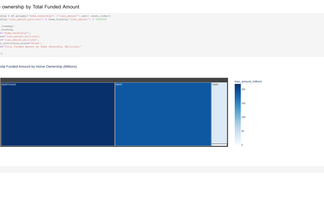


Regional Analysis by State for Total Funded Amount

```
state_funding = df.groupby('address_state').['loan_amount'].sum(.sort_values(ascending=False)
print("Regional Analysis by Total Funding:")
state_funding
```

```
plt.figure(figsize=(5,5))
bars = plt.bar(state_funding.index, state_funding.values/1000000 values, color='lightgray')
```

```
for bar in bars:
    height = bar.get_height()
    if bar.get_x() < 5:
        plt.text(bar.get_x() + bar.get_width()/ 2 ,
                 height + 0.5,
                 f'{height:.1f}',
plt.xlabel('States')
plt.ylabel('Total Funded Amount (in Millions)')
plt.show()
```



Long Term Analysis by Total Funded Amount

```
term_funding_applications = df.groupby('term').['loan_amount'].sum( .sort_values(ascending=False)
print("Regional Analysis by Term:")
term_funding_applications
```

```
plt.figure(figsize=(5,5))
term_funding_applications.plot(kind='pie',
                               autopct=lambda p: f'{p:.1f}%\n({int(round(p/100 * sum(term_funding_applications)/1000000,
                               colors=['lightgray','lightblue'],
                               startangle=90,
                               labels=['36 months', '60 months'])
plt.pie(term_funding_applications, labels=term_funding_applications.index, colors=['lightgray', 'lightblue'],
plt.title('Total Funded Amount By Term (in & Millions)')
plt.show()
```


Employee Length by Total Funded Amount

```
emp_funding_applications = df.groupby('emp_length').['loan_amount'].sum( .sort_values(ascending=False)
print("Regional Analysis by:")
emp_funding_applications
```

```
plt.figure(figsize=(5,5))
bars = plt.barh(emp_funding_applications.index, emp_funding_applications/1000000, color='gray')
```

```
for bar in bars:
    height = bar.get_height()
    if bar.get_x() < 5:
        plt.text(bar.get_x() + bar.get_width()/ 2 ,
                 height + 0.5,
plt.xlabel('Total Funded Amount by Employment Length')
plt.ylabel('Employee Length')
plt.show()
plt.show()
```


Loan Purpose by Total Funded Amount

```
purpose_funding = df.groupby('purpose').['loan_amount'].sum( .sort_values(ascending=False)
print("Regional Analysis by:")
purpose_funding
```

```
plt.figure(figsize=(5,5))
bars = plt.barh(purpose_funding.index, purpose_funding.values/1000000, color='gray')
```

```
for bar in bars:
    height = bar.get_height()
    if bar.get_x() < 5:
        plt.text(bar.get_x() + bar.get_width()/ 2 ,
                 height + 0.5,
plt.xlabel('Total Funded Amount by Loan Purpose (Millions)')
plt.ylabel('Loan Purpose')
plt.show()
plt.show()
```


Home ownership by Total Funded Amount

```
home_funding = df.groupby('home_ownership').['loan_amount'].sum( .sort_values(ascending=False)
home_funding['loan_amount'] = home_funding['loan_amount'] / 1000000
```

```
fig = px.treemap(
    home_funding,
    path=['home_ownership'],
    values='loan_amount',
    color='loan_amount',
    color_continuous_scale='Blues',
    title='Total Funded Amount by Home Ownership (Millions)'
)
fig.show()
```