

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```
df1=pd.read_csv(r"c:\Users\DELL\Downloads\Finance1cd.csv")
print(df1.head())
```

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	
term \						
0	54734	80364	25000	25000	19080.057200	36
months						
1	55742	114426	7000	7000	672.803839	36
months						
2	57245	138150	1200	1200	1200.000000	36
months						
3	57416	139635	10800	10800	10691.551050	36
months						
4	58915	153417	7500	5025	557.087228	36
months						

	int_rate	installment	grade	sub_grade	...	verification_status
Issue_d \						
0	11.89%	829.10	B	B4	...	Verified
2009-08-01						
1	10.71%	228.22	B	B5	...	Not Verified
2008-05-01						
2	13.11%	40.50	C	C2	...	Not Verified
2010-03-01						
3	13.57%	366.86	C	C3	...	Not Verified
2009-11-01						
4	10.08%	162.34	B	B3	...	Not Verified
2008-04-01						

	loan_status	pymnt_plan	
desc \			
0	Fully Paid	n	Due to a lack of personal finance education an...
1	Fully Paid	n	Just want to pay off the last bit of credit ca...
2	Fully Paid	n	If funded, I would use this loan consolidate t...
3	Fully Paid	n	I currently have a personal loan with Citifina...
4	Fully Paid	n	Hi : Thanks for stopping by. I've accr...

	purpose	title
zip_code \		
0	debt_consolidation	Debt consolidation for on-time payer 941xx

1	credit_card	Credit Card payoff	112xx
2	debt_consolidation	zxcvb	777xx
3	debt_consolidation	Nicolechr1978	067xx
4	debt_consolidation	sdguy	921xx

	addr_state	dti
0	CA	19.48
1	NY	14.29
2	TX	5.47
3	CT	11.63
4	CA	8.10

[5 rows x 24 columns]

```
import pandas as pd
df2=pd.read_csv(r"c:\Users\DELL\OneDrive\Documents\Finance2cd.csv")
print(df2.head())
```

	id	earliest_cr_line	revol_bal	total_pymnt	last_pymnt_d	\
0	54734	Feb-94	28854	29330.356700	2011-10-01	
1	55742	Oct-00	33623	8215.537060	2011-06-01	
2	57245	Jan-85	2584	1457.819531	2013-03-01	
3	57416	Dec-96	3511	13207.763300	2012-11-01	
4	58915	Nov-00	33667	5843.875367	2011-04-01	

	last_pymnt_amnt	last_credit_pull_d
0	7392.08	2012-08-01
1	228.48	2012-08-01
2	42.73	2016-05-01
3	398.00	2016-05-01
4	162.63	2014-03-01

```
print(df1.info())
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 39717 entries, 0 to 39716
```

```
Data columns (total 24 columns):
```

#	Column	Non-Null Count	Dtype
---	-----	-----	-----
0	id	39717 non-null	int64
1	member_id	39717 non-null	int64
2	loan_amnt	39717 non-null	int64
3	funded_amnt	39717 non-null	int64
4	funded_amnt_inv	39717 non-null	float64
5	term	39717 non-null	object
6	int_rate	39717 non-null	object

```

7  installment      39717 non-null float64
8  grade            39717 non-null object
9  sub_grade        39717 non-null object
10 emp_title         37258 non-null object
11 emp_length        38642 non-null object
12 home_ownership    39717 non-null object
13 annual_inc        39717 non-null float64
14 verification_status 39717 non-null object
15 Issue_d           39717 non-null object
16 loan_status        39717 non-null object
17 pymnt_plan         39717 non-null object
18 desc              26775 non-null object
19 purpose            39717 non-null object
20 title              39706 non-null object
21 zip_code           39717 non-null object
22 addr_state         39717 non-null object
23 dti                39717 non-null float64
dtypes: float64(4), int64(4), object(16)
memory usage: 7.3+ MB
None

```

#question

#Year Wise Loan Amount Stats

```

grpdl=(df1.groupby('Issue_d')['loan_amnt'].sum())
print(grpdl)
formatted1=grpdl.apply(lambda x: "${:,.0f}".format(x))
print(formatted1)

```

```

Issue_d
2007-06-01      7500
2007-07-01     171700
2007-08-01     208475
2007-09-01     146025
2007-10-01     318775
2007-11-01     380625
2007-12-01     986175
2008-01-01    1761050
2008-02-01    1679375
2008-03-01    2693925
2008-04-01    1556300
2008-05-01     418900
2008-06-01     379975
2008-07-01     524925
2008-08-01     412200
2008-09-01     184925
2008-10-01     780750
2008-11-01    1703450

```

2008-12-01	2294500
2009-01-01	2186225
2009-02-01	2341075
2009-03-01	2653250
2009-04-01	2678150
2009-05-01	3115150
2009-06-01	3192350
2009-07-01	3486450
2009-08-01	3933700
2009-09-01	4580125
2009-10-01	5611900
2009-11-01	6167225
2009-12-01	6490725
2010-01-01	6579725
2010-02-01	6784725
2010-03-01	7638325
2010-04-01	8657075
2010-05-01	9932700
2010-06-01	10474400
2010-07-01	11183000
2010-08-01	11185475
2010-09-01	11233400
2010-10-01	12163250
2010-11-01	12503550
2010-12-01	13714575
2011-01-01	14824575
2011-02-01	14585275
2011-03-01	16503275
2011-04-01	17522075
2011-05-01	19062225
2011-06-01	20808400
2011-07-01	21271900
2011-08-01	23149075
2011-09-01	25762300
2011-10-01	27392550
2011-11-01	28105075
2011-12-01	31519850

Name: loan\_amnt, dtype: int64

Issue\_d

2007-06-01	\$7,500
2007-07-01	\$171,700
2007-08-01	\$208,475
2007-09-01	\$146,025
2007-10-01	\$318,775
2007-11-01	\$380,625
2007-12-01	\$986,175
2008-01-01	\$1,761,050
2008-02-01	\$1,679,375
2008-03-01	\$2,693,925

2008-04-01	\$1,556,300
2008-05-01	\$418,900
2008-06-01	\$379,975
2008-07-01	\$524,925
2008-08-01	\$412,200
2008-09-01	\$184,925
2008-10-01	\$780,750
2008-11-01	\$1,703,450
2008-12-01	\$2,294,500
2009-01-01	\$2,186,225
2009-02-01	\$2,341,075
2009-03-01	\$2,653,250
2009-04-01	\$2,678,150
2009-05-01	\$3,115,150
2009-06-01	\$3,192,350
2009-07-01	\$3,486,450
2009-08-01	\$3,933,700
2009-09-01	\$4,580,125
2009-10-01	\$5,611,900
2009-11-01	\$6,167,225
2009-12-01	\$6,490,725
2010-01-01	\$6,579,725
2010-02-01	\$6,784,725
2010-03-01	\$7,638,325
2010-04-01	\$8,657,075
2010-05-01	\$9,932,700
2010-06-01	\$10,474,400
2010-07-01	\$11,183,000
2010-08-01	\$11,185,475
2010-09-01	\$11,233,400
2010-10-01	\$12,163,250
2010-11-01	\$12,503,550
2010-12-01	\$13,714,575
2011-01-01	\$14,824,575
2011-02-01	\$14,585,275
2011-03-01	\$16,503,275
2011-04-01	\$17,522,075
2011-05-01	\$19,062,225
2011-06-01	\$20,808,400
2011-07-01	\$21,271,900
2011-08-01	\$23,149,075
2011-09-01	\$25,762,300
2011-10-01	\$27,392,550
2011-11-01	\$28,105,075
2011-12-01	\$31,519,850

Name: loan\_amnt, dtype: object

```
df1['Issue_d'] = pd.to_datetime(df1['Issue_d'], errors='coerce')
```

```
print(df1['Issue_d'].dtype)
```

```
datetime64[ns]
```

```
df1['Year'] = df1['Issue_d'].dt.year
```

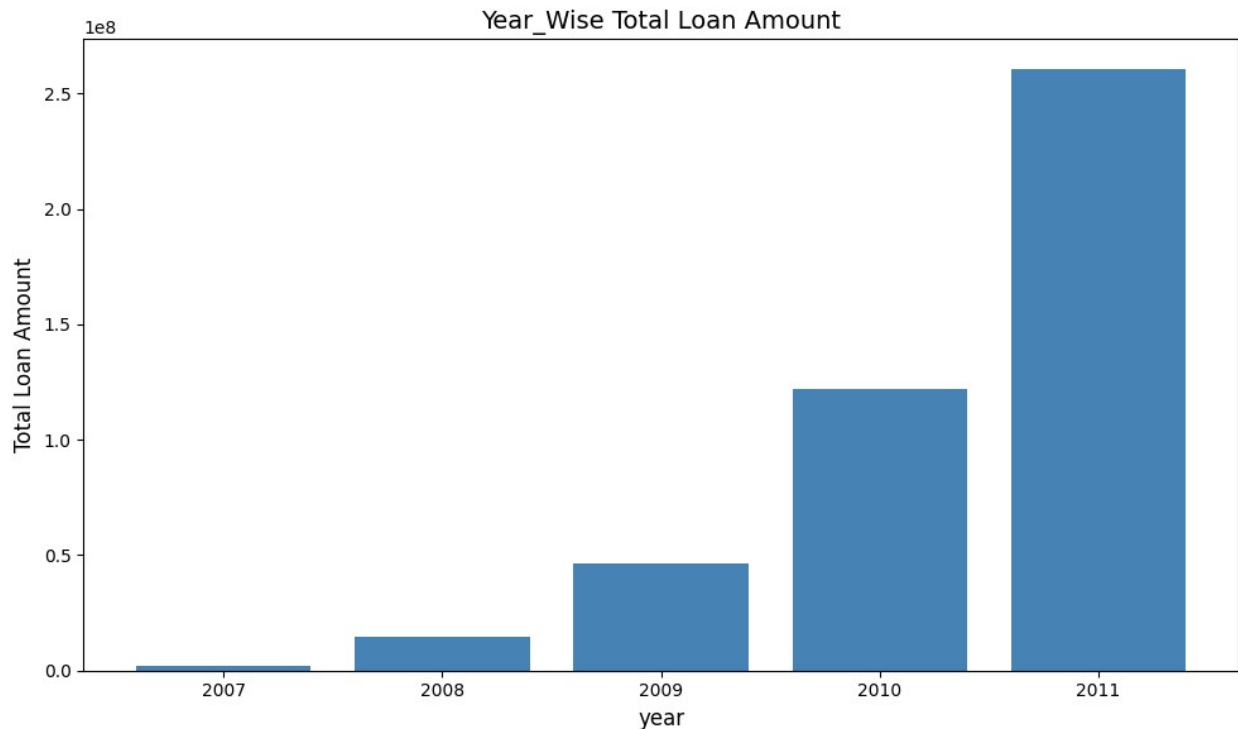
```
yearly_loan = df1.groupby('Year', as_index=False)['loan_amnt'].sum()  
print(yearly_loan)
```

	Year	loan_amnt
0	2007	2219275
1	2008	14390275
2	2009	46436325
3	2010	122050200
4	2011	260506575

```
formatted=yearly_loan.copy()  
formatted['loan_amnt']=formatted['loan_amnt'].apply(lambda x: "$  
{:, .0f}".format(x))  
print(formatted)
```

	Year	loan_amnt
0	2007	\$2,219,275
1	2008	\$14,390,275
2	2009	\$46,436,325
3	2010	\$122,050,200
4	2011	\$260,506,575

```
df1['Issue_d']=pd.to_datetime(df1['Issue_d'],errors='coerce')  
df1['year']=df1['Issue_d'].dt.year  
plt.figure(figsize=(10,6))  
plt.bar(yearly_loan['Year'].astype(str),yearly_loan['loan_amnt'],color  
='steelblue')  
plt.title("Year_Wise Total Loan Amount",fontsize=14)  
plt.xlabel("year",fontsize=12)  
plt.ylabel("Total Loan Amount",fontsize=12)  
plt.tight_layout()  
plt.show()
```



#question

#Grade and Subgrade Wise Revol\_bal

```
import pandas as pd
merged=pd.merge(df1,df2,on='id',how='inner')
print(merged)
```

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	\
0	54734	80364	25000	25000	19080.057200	
1	55742	114426	7000	7000	672.803839	
2	57245	138150	1200	1200	1200.000000	
3	57416	139635	10800	10800	10691.551050	
4	58915	153417	7500	5025	557.087228	
...	...	...	...	...	...	...
39712	1075358	1311748	3000	3000	3000.000000	
39713	1076863	1277178	10000	10000	10000.000000	
39714	1077175	1313524	2400	2400	2400.000000	
39715	1077430	1314167	2500	2500	2500.000000	
39716	1077501	1296599	5000	5000	4975.000000	

	term	int_rate	installment	grade	sub_grade	...
addr_state \						
0	36 months	11.89%	829.10	B	B4	...
CA						
1	36 months	10.71%	228.22	B	B5	...
NY						
2	36 months	13.11%	40.50	C	C2	...

TX						
3	36 months	13.57%	366.86	C	C3	...
CT						
4	36 months	10.08%	162.34	B	B3	...
CA						
...	...	...	...	...	...	...
.						
39712	60 months	12.69%	67.79	B	B5	...
OR						
39713	36 months	13.49%	339.31	C	C1	...
CA						
39714	36 months	15.96%	84.33	C	C5	...
IL						
39715	60 months	15.27%	59.83	C	C4	...
GA						
39716	36 months	10.65%	162.87	B	B2	...
AZ						

	dti	Year	year	earliest_cr_line	revol_bal	total_pymnt	\
0	19.48	2009	2009	Feb-94	28854	29330.356700	
1	14.29	2008	2008	Oct-00	33623	8215.537060	
2	5.47	2010	2010	Jan-85	2584	1457.819531	
3	11.63	2009	2009	Dec-96	3511	13207.763300	
4	8.10	2008	2008	Nov-00	33667	5843.875367	
...	...	...	...	...	...	...	
39712	17.94	2011	2011	Jan-96	27783	3513.330000	
39713	20.00	2011	2011	Feb-96	5598	12231.890000	
39714	8.72	2011	2011	Nov-01	2956	3005.666844	
39715	1.00	2011	2011	Apr-99	1687	1008.710000	
39716	27.65	2011	2011	Jan-85	13648	5863.155187	

	last_pymnt_d	last_pymnt_amnt	last_credit_pull_d
0	2011-10-01	7392.08	2012-08-01
1	2011-06-01	228.48	2012-08-01
2	2013-03-01	42.73	2016-05-01
3	2012-11-01	398.00	2016-05-01
4	2011-04-01	162.63	2014-03-01
...	...	...	...
39712	2016-05-01	67.79	2016-05-01
39713	2015-01-01	357.48	2016-04-01
39714	2014-06-01	649.91	2016-05-01
39715	2013-04-01	119.66	2013-09-01
39716	2015-01-01	171.62	2016-05-01

[39717 rows x 32 columns]

```

grpdp2=merged.groupby(['grade', 'sub_grade'])
['revol_bal'].sum().reset_index()
grpdp2['format2']=grpdp2['revol_bal'].apply(lambda x: "$

```



```
{:,.0f}").format(x))
print(grpd2)
```

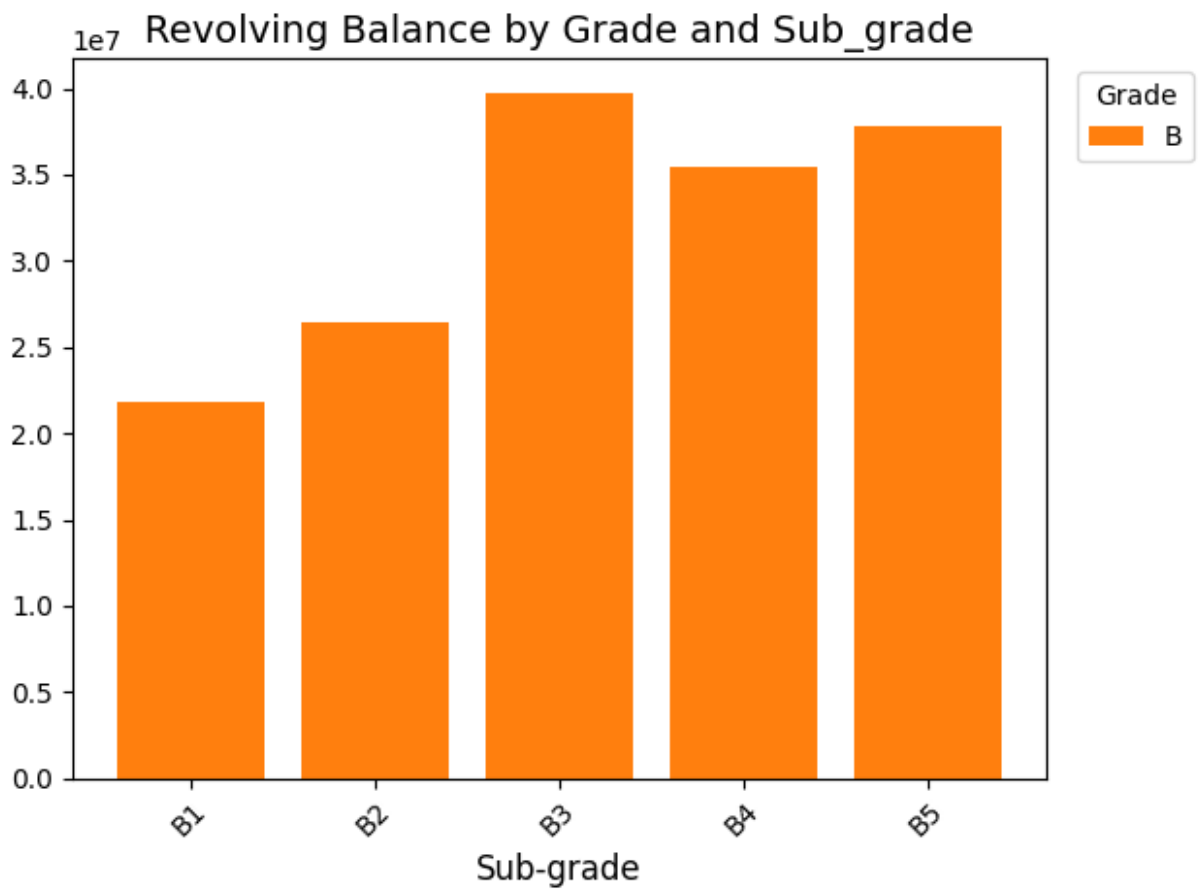
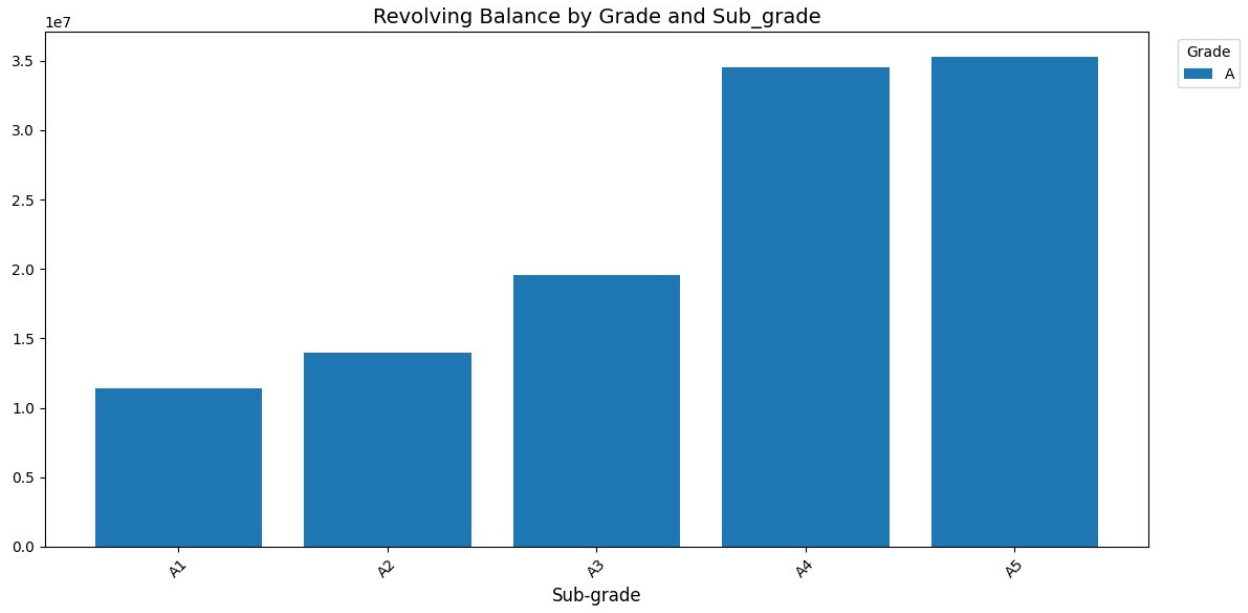
	grade	sub_grade	revol_bal	format2
0	A	A1	11365196	\$11,365,196
1	A	A2	14004780	\$14,004,780
2	A	A3	19543922	\$19,543,922
3	A	A4	34557156	\$34,557,156
4	A	A5	35303045	\$35,303,045
5	B	B1	21842079	\$21,842,079
6	B	B2	26478439	\$26,478,439
7	B	B3	39723554	\$39,723,554
8	B	B4	35405811	\$35,405,811
9	B	B5	37858666	\$37,858,666
10	C	C1	29384926	\$29,384,926
11	C	C2	27321114	\$27,321,114
12	C	C3	20531370	\$20,531,370
13	C	C4	16867691	\$16,867,691
14	C	C5	16015609	\$16,015,609
15	D	D1	12130255	\$12,130,255
16	D	D2	18570972	\$18,570,972
17	D	D3	16793781	\$16,793,781
18	D	D4	13742947	\$13,742,947
19	D	D5	13252474	\$13,252,474
20	E	E1	11132588	\$11,132,588
21	E	E2	10242033	\$10,242,033
22	E	E3	9039059	\$9,039,059
23	E	E4	7990991	\$7,990,991
24	E	E5	7669868	\$7,669,868
25	F	F1	5840746	\$5,840,746
26	F	F2	4528248	\$4,528,248
27	F	F3	3175435	\$3,175,435
28	F	F4	2551064	\$2,551,064
29	F	F5	2187323	\$2,187,323
30	G	G1	1808763	\$1,808,763
31	G	G2	1729627	\$1,729,627
32	G	G3	832193	\$832,193
33	G	G4	1390628	\$1,390,628
34	G	G5	701515	\$701,515

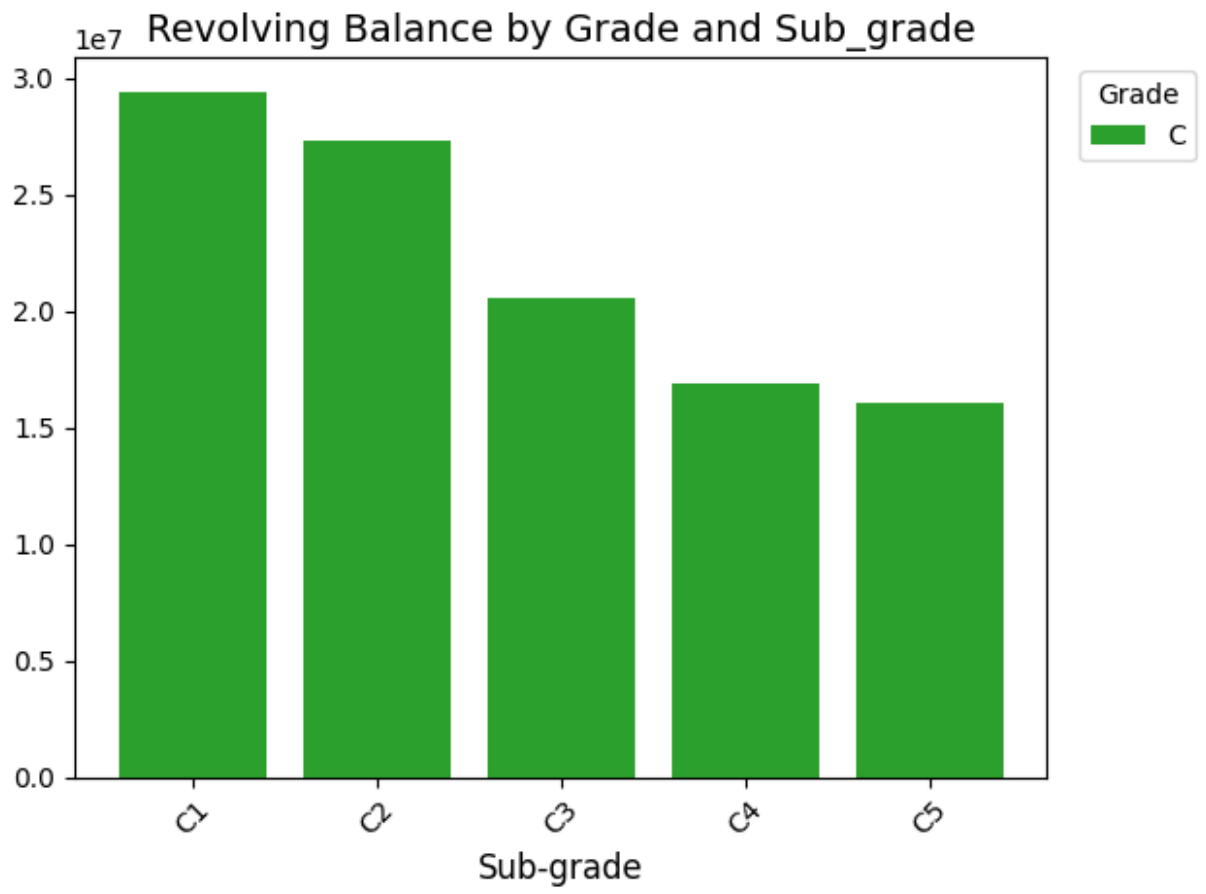
```
import pandas as pd
import matplotlib.pyplot as plt
#Assuming 'merged' already contains the columns we need
merged.columns=merged.columns.str.strip().str.lower()
#Rename grade x/sub_grade_x for convenience
merged=merged.rename(columns={'grade_x':'grade','sub_grade_x':'sub_grade'})
#group and sum revolving balance
grpd2=(merged.groupby(['grade','sub_grade'])
['revol_bal'].sum().reset_index())
```

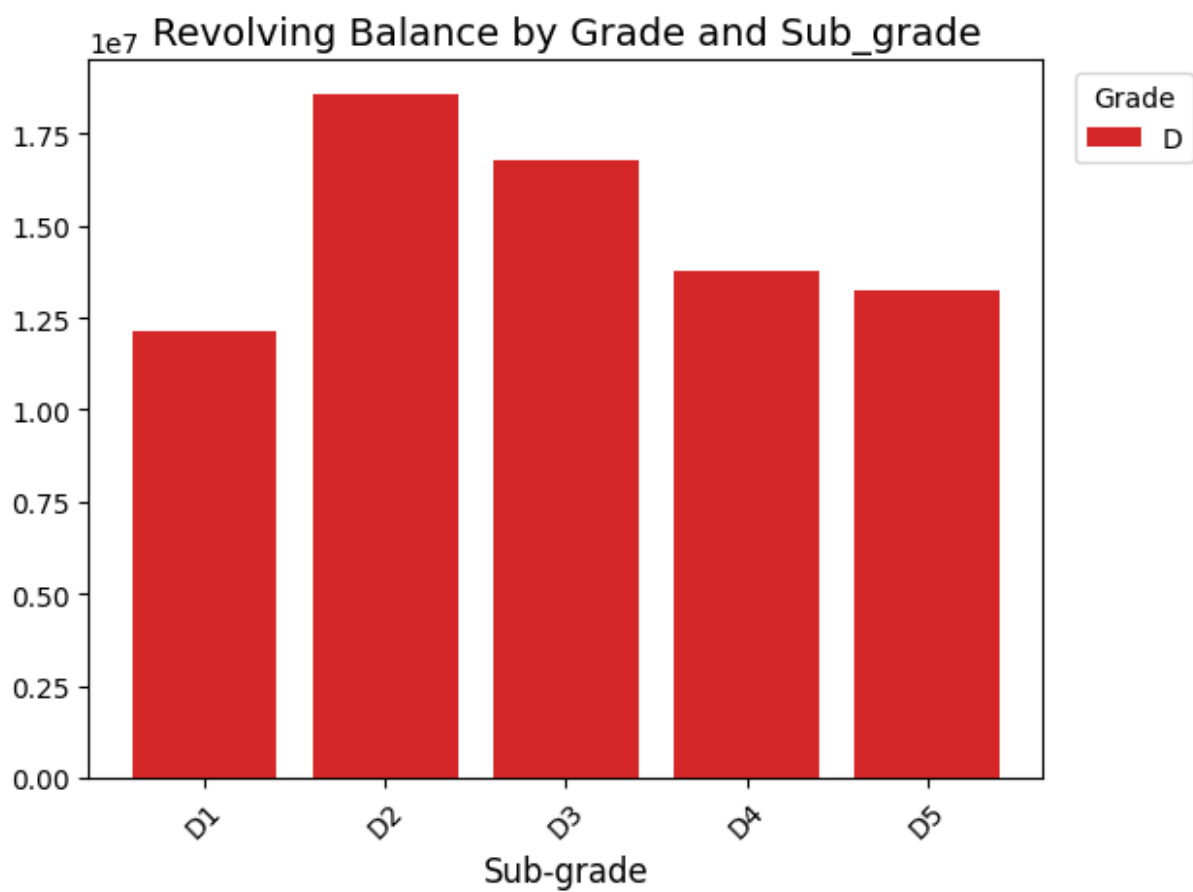
```

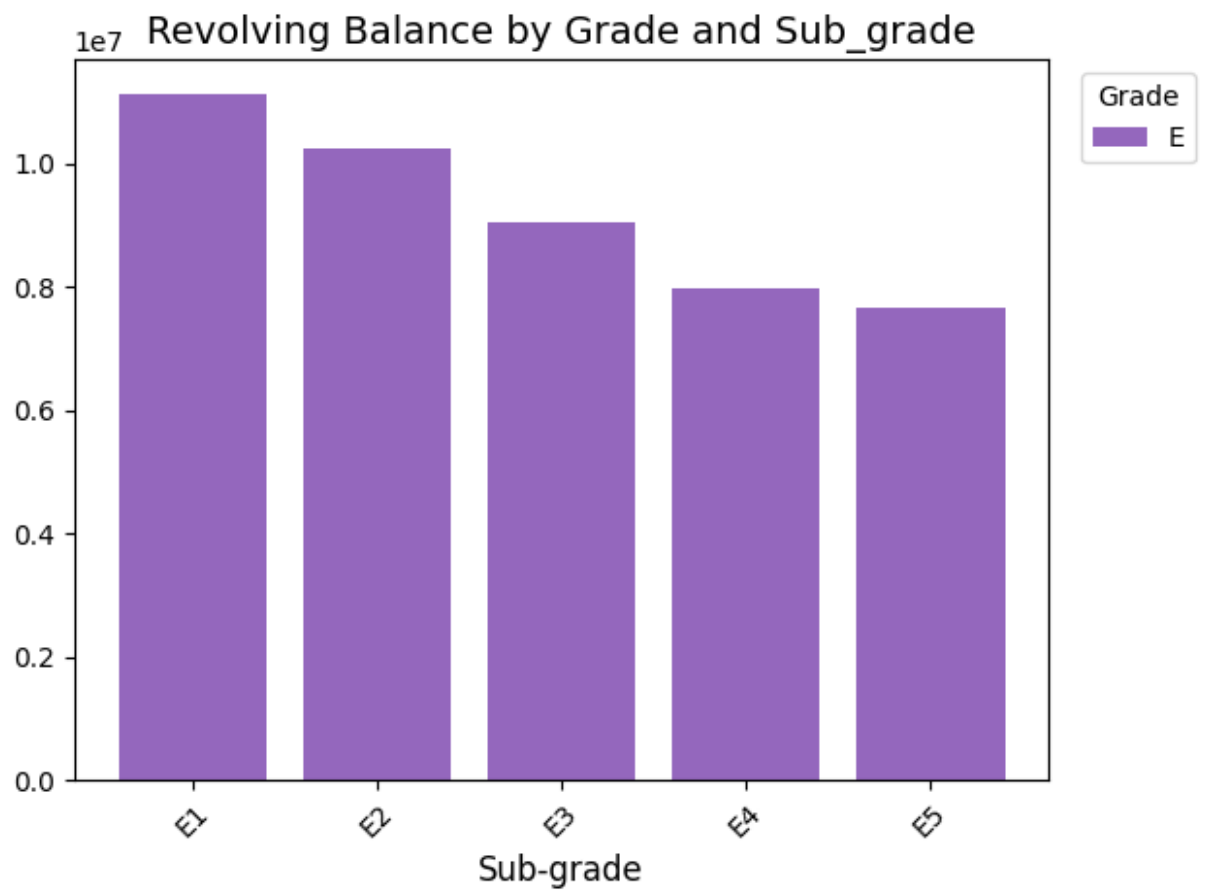
#Add a formatted string column(optional)
grp2['format2']=grp2['revol_bal'].apply(lambda x: "$
{:,.0f}".format(x))
#sort sub-grades for natural ordering
grp2=grp2.sort_values('sub_grade')
#set up the plot
plt.figure(figsize=(12,6))
#colors for each grade
grades=sorted(grp2['grade'].unique())
palette=plt.get_cmap('tab10').colors[:len(grades)]
#plot bars grouped by grade
for i,grade in enumerate(grades):
    subset=grp2[grp2['grade']==grade]
    plt.bar(
        subset['sub_grade'],
        subset['revol_bal'],
        label=grade,
        color=palette[i]
    )
#labels & formatting
plt.title("Revolving Balance by Grade and Sub_grade",fontsize=14)
plt.xlabel("Sub-grade",fontsize=12)
plt.xticks(rotation=45)
plt.legend(title='Grade',bbox_to_anchor=(1.02,1),loc='upper left')
#optional:add value labels above bars
#max_val=grp2['revol_bal'].max()
#for idx,row in grp2.iterrows():
# x=row['sub_grade']
# y=row['revol_bal']
#plt.text(
#    x,y+max_val*0.01,
#    f"${y:,.0f}"
#    ha='center', va='bottom',fontsize=9
# )
plt.tight_layout()
plt.show()

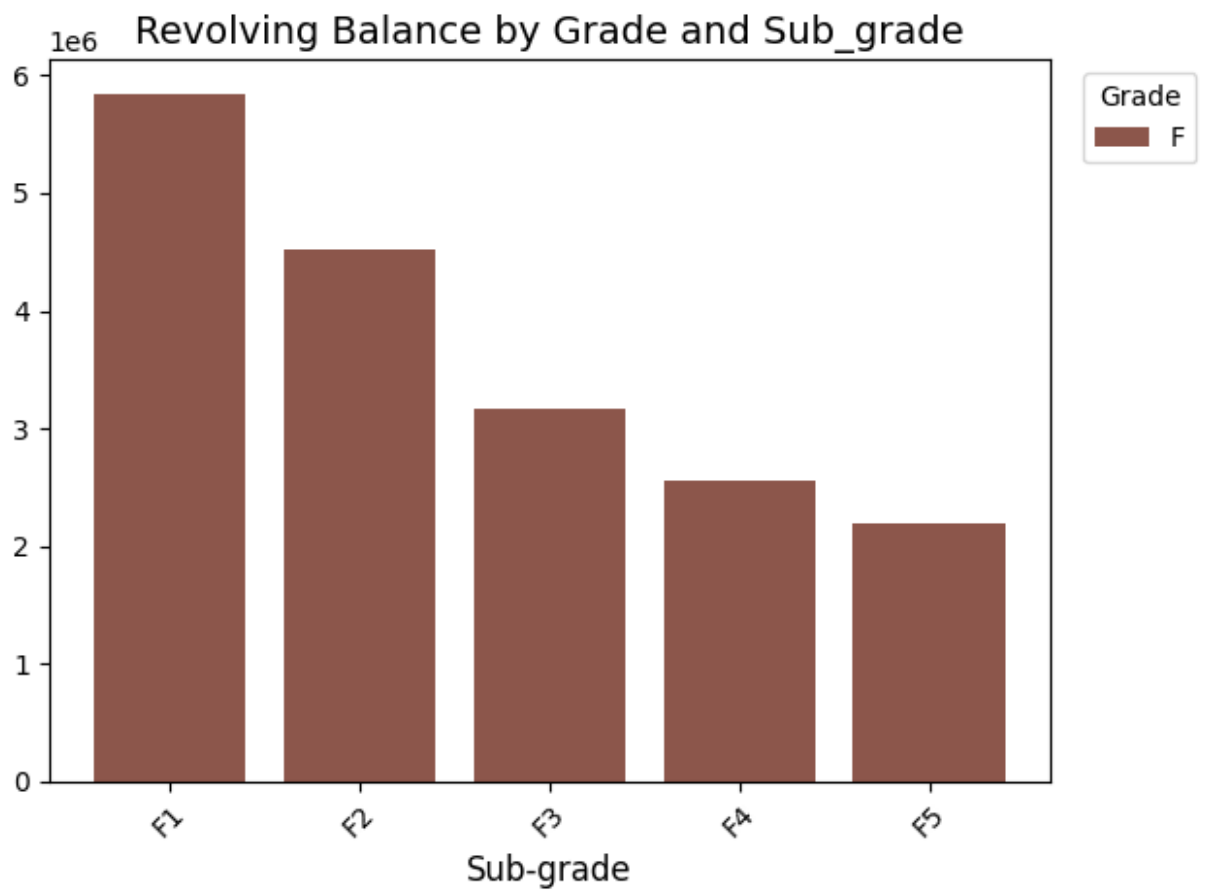
```

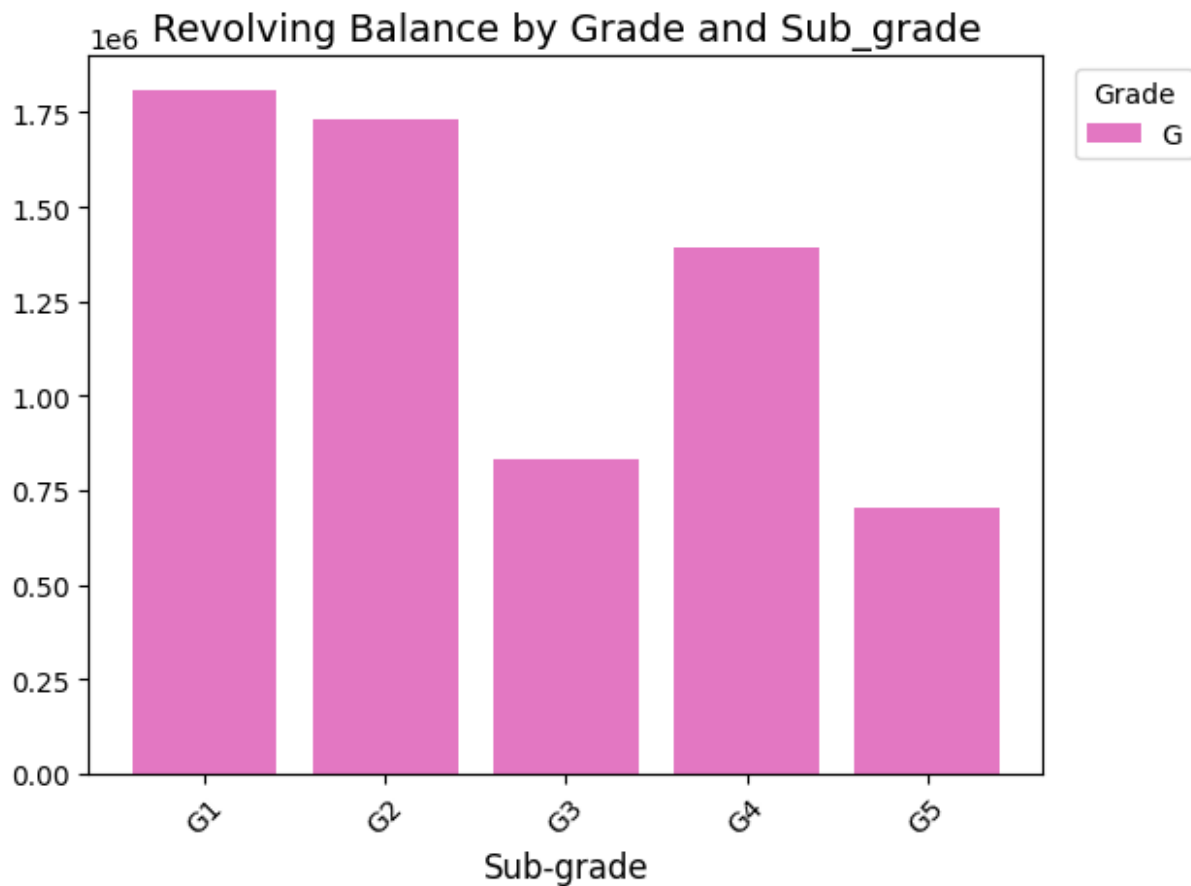












#question 3

#Total Payment For Verified Status vs Total Payment for Non verified Status

```
verified= merged.groupby('verification_status')
['total_pymnt'].sum().reset_index()
verified['format3']=verified['total_pymnt'].apply(lambda x: "$
{:, .0f}".format(x))
print(verified)
```

	verification_status	total_pymnt	format3
0	Not Verified	1.535414e+08	\$153,541,418
1	Source Verified	1.092707e+08	\$109,270,668
2	Verified	2.198923e+08	\$219,892,308

```
import matplotlib.pyplot as plt
import pandas as pd
```

*# Simulate your grouped data*

```
verified = pd.DataFrame({
    'status': ['Not verified', 'Source verified', 'verified'],
    'total_pymnt': [1200000, 1000000, 1500000]
})
```



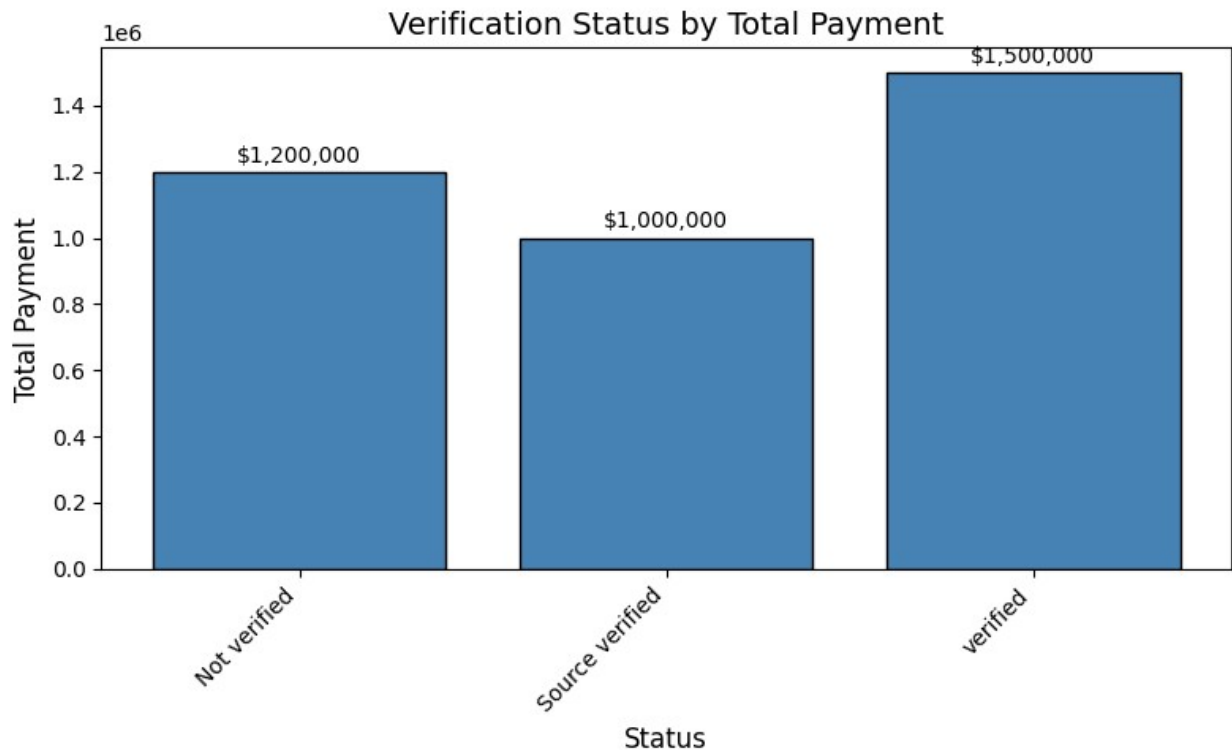
```
# Plot
plt.figure(figsize=(8,5))
plt.bar(
    verified['status'],
    verified['total_pymnt'],
    color='steelblue',
    edgecolor='black'
)

# Add titles and labels
plt.title("Verification Status by Total Payment", fontsize=14)
plt.xlabel("Status", fontsize=12)
plt.ylabel("Total Payment", fontsize=12)

# Rotate x-axis labels
plt.xticks(rotation=45, ha='right')

# Optionally annotate bar values
for status, payment in zip(verified['status'],
verified['total_pymnt']):
    plt.text(
        status,
        payment + verified['total_pymnt'].max() * 0.01,
        f"${payment:,.0f}",
        ha='center',
        va='bottom'
    )

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



#question 4

#State Wise And Month Wise Loan Status

```
print(merged.columns.tolist())

['id', 'member_id', 'loan_amnt', 'funded_amnt', 'funded_amnt_inv',
 'term', 'int_rate', 'installment', 'grade', 'sub_grade', 'emp_title',
 'emp_length', 'home_ownership', 'annual_inc', 'verification_status',
 'issue_d', 'loan_status', 'pymnt_plan', 'desc', 'purpose', 'title',
 'zip_code', 'addr_state', 'dti', 'year', 'year', 'earliest_cr_line',
 'revol_bal', 'total_pymnt', 'last_pymnt_d', 'last_pymnt_amnt',
 'last_credit_pull_d']

import pandas as pd
# Convert your date column to datetime
merged['issue_d'] = pd.to_datetime(merged['issue_d'], errors='coerce')
# Create a month column (integer)
merged['Month'] = merged['issue_d'].dt.month # or .dt.month_name()
q4 = merged.groupby(
    ['addr_state', 'Month', 'loan_status']
)['loan_amnt'].count().reset_index(name='count_loans')
print(q4.head())
```

	addr_state	Month	loan_status	count_loans
0	AK	1	Fully Paid	6
1	AK	2	Fully Paid	2

2	AK	3	Charged Off	2
3	AK	3	Fully Paid	6
4	AK	4	Charged Off	1

```
merged['YearMonth'] = merged['issue_d'].dt.to_period('M')
print(merged)
```

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	\
0	54734	80364	25000	25000	19080.057200	
1	55742	114426	7000	7000	672.803839	
2	57245	138150	1200	1200	1200.000000	
3	57416	139635	10800	10800	10691.551050	
4	58915	153417	7500	5025	557.087228	
...	...	...	...	...	...	
39712	1075358	1311748	3000	3000	3000.000000	
39713	1076863	1277178	10000	10000	10000.000000	
39714	1077175	1313524	2400	2400	2400.000000	
39715	1077430	1314167	2500	2500	2500.000000	
39716	1077501	1296599	5000	5000	4975.000000	

	term	int_rate	installment	grade	sub_grade	...	year
year \							
0	36 months	11.89%	829.10	B	B4	...	2009
2009							
1	36 months	10.71%	228.22	B	B5	...	2008
2008							
2	36 months	13.11%	40.50	C	C2	...	2010
2010							
3	36 months	13.57%	366.86	C	C3	...	2009
2009							
4	36 months	10.08%	162.34	B	B3	...	2008
2008							
...	...	...	...	...	...	...	...
..							
39712	60 months	12.69%	67.79	B	B5	...	2011
2011							
39713	36 months	13.49%	339.31	C	C1	...	2011
2011							
39714	36 months	15.96%	84.33	C	C5	...	2011
2011							
39715	60 months	15.27%	59.83	C	C4	...	2011
2011							
39716	36 months	10.65%	162.87	B	B2	...	2011
2011							

	earliest_cr_line	revol_bal	total_pymnt	last_pymnt_d
last_pymnt_amnt \				
0	Feb-94	28854	29330.356700	2011-10-01
7392.08				
1	Oct-00	33623	8215.537060	2011-06-01

228.48				
2	Jan-85	2584	1457.819531	2013-03-01
42.73				
3	Dec-96	3511	13207.763300	2012-11-01
398.00				
4	Nov-00	33667	5843.875367	2011-04-01
162.63				
...	...	...	...	...
...				
39712	Jan-96	27783	3513.330000	2016-05-01
67.79				
39713	Feb-96	5598	12231.890000	2015-01-01
357.48				
39714	Nov-01	2956	3005.666844	2014-06-01
649.91				
39715	Apr-99	1687	1008.710000	2013-04-01
119.66				
39716	Jan-85	13648	5863.155187	2015-01-01
171.62				

	last_credit_pull_d	Month	YearMonth
0	2012-08-01	8	2009-08
1	2012-08-01	5	2008-05
2	2016-05-01	3	2010-03
3	2016-05-01	11	2009-11
4	2014-03-01	4	2008-04
...	...	...	...
39712	2016-05-01	12	2011-12
39713	2016-04-01	12	2011-12
39714	2016-05-01	12	2011-12
39715	2013-09-01	12	2011-12
39716	2016-05-01	12	2011-12

[39717 rows x 34 columns]

```
q4.rename(columns={'loan_amnt': 'loan_count'}, inplace=True)
print(q4.columns.tolist())
print(q4.head())
```

```
['addr_state', 'Month', 'loan_status', 'count_loans']
addr_state  Month  loan_status  count_loans
0          AK      1    Fully Paid           6
1          AK      2    Fully Paid           2
2          AK      3    Charged Off           2
3          AK      3    Fully Paid           6
4          AK      4    Charged Off           1
```

```
plt.figure(figsize=(10, 6))
sns.barplot(
    data=q4,
```

```

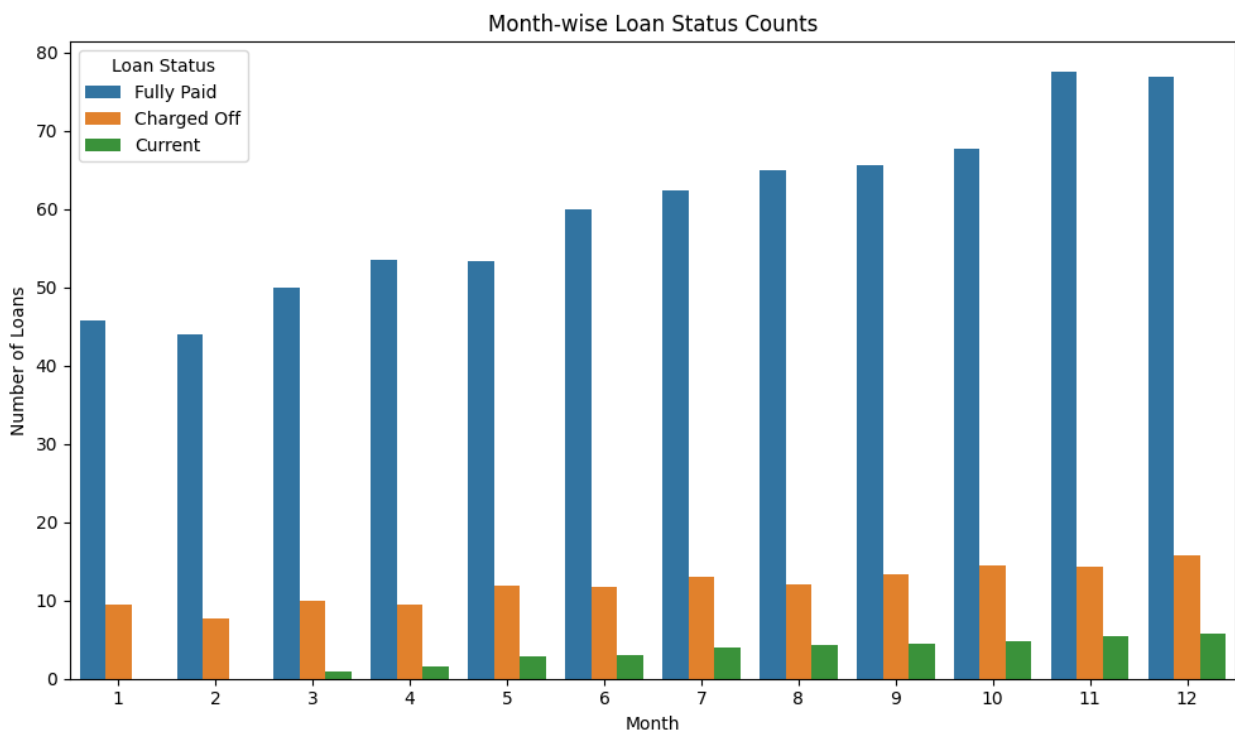
x='Month',
y='count_loans',      # use the actual column name
hue='loan_status',
ci=None
)
plt.title("Month-wise Loan Status Counts")
plt.xlabel("Month")
plt.ylabel("Number of Loans")
plt.legend(title='Loan Status')
plt.tight_layout()
plt.show()
# plt.title("Loans by State, Month & Status")
# plt.xlabel("Month")
# plt.ylabel("Number of Loans")
# plt.legend(title="Loan Status", bbox_to_anchor=(1.02, 1),
loc='upper left')
plt.tight_layout()
plt.show()

```

C:\Users\DELL\AppData\Local\Temp\ipykernel\_8684\3347346721.py:2:  
FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

```
sns.barplot(
```



<Figure size 640x480 with 0 Axes>

#question 5

#Home Ownership vs last payment date stats

```
q5=merged.groupby('home_ownership')['last_pymnt_d'].describe()
print(q5)
```

	count	unique	top	freq
home_ownership				
MORTGAGE	17645	100	2016-05-01	705
NONE	3	2	2010-09-01	2
OTHER	98	40	2011-03-01	7
OWN	3053	98	2016-05-01	94
RENT	18847	100	2014-12-01	524

```
q5=merged.groupby('home_ownership')['last_pymnt_amnt'].describe()
print(q5)
```

	count	mean	std	min	25%
home_ownership					
MORTGAGE	17659.0	3188.490051	5014.658942	0.00	253.6250
NONE	3.0	177.716667	119.353980	89.80	109.7800
OTHER	98.0	1767.746939	3455.543863	1.34	196.1475
OWN	3058.0	2690.963846	4665.771046	0.00	200.8025
RENT	18899.0	2205.759727	3743.250498	0.00	198.1600

	75%	max
home_ownership		
MORTGAGE	4204.5350	35613.68
NONE	221.6750	313.59
OTHER	1434.0125	20617.94
OWN	3145.2025	35479.89
RENT	2615.1350	36115.20

```
q5_mean = q5['mean']
# Plot
plt.figure(figsize=(10, 6))
plt.bar(q5_mean.index, q5_mean.values, color='teal')
plt.title('Mean Last Payment Amount by Home Ownership')
plt.xlabel('Home Ownership')
plt.ylabel('Mean Last Payment Amount')
```

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

