In [1]:
```
mport pandas as pd
mport numpy as np
mport matplotlib.pyplot as plt
mport seaborn as sns
matplotlib inline
```

In [2]:
```
mport json
mport requests
```

Get Current Data from Lending Club

In [3]:
```
eaders = {'Authorization':'VEzi+drgj+ybnUSkbtrCTsngN/g='}
arameters = {'showAll': True}
rl='https://api.lendingclub.com/api/investor/v1/loans/listing'
=requests.get(url, headers=headers, params = parameters)
```

In [19]:
```
ata = r.json()
yData =data['loans']
```

In [20]:
```
ith open('current_list.txt', 'w') as outfile:
    json.dump(myData, outfile)
```

In [21]:
```
f_current=pd.read_json('current_list.txt')
```

In [23]:
```
f_current.describe()
```

Out[23]:

	accNowDelinq	accOpenPast24Mths	allUtil	annualInc	annualIncJoint	avgCurBal	bcOpenT
count	101.0	101.000000	101.000000	101.000000	14.000000	101.000000	101.0(
mean	0.0	4.940594	54.729703	100602.799208	128860.071429	11442.217822	20554.3
std	0.0	2.848936	19.249678	82763.440060	54008.380547	11063.858383	26733.8
min	0.0	0.000000	10.600000	15000.000000	71880.000000	190.000000	0.0(
25%	0.0	3.000000	41.100000	55000.000000	92000.000000	3007.000000	5350.0(
50%	0.0	5.000000	53.600000	82500.000000	110500.000000	7209.000000	11930.0(
75%	0.0	7.000000	66.800000	120000.000000	156500.000000	17598.000000	27026.0(
max	0.0	13.000000	108.600000	700000.000000	267000.000000	60874.000000	165485.0(

8 rows × 98 columns

In [29]:
```
f_current = pd.io.json.json_normalize(myData)
```

In [30]:
```
f_current.tail()
```

Out[30]:

	accNowDelinq	accOpenPast24Mths	acceptD	addrState	addrZip	allUtil	annualInc	annualIncJoir
96	0	3	2019-02-01T20:18:41.000-08:00	CA	914xx	27.5	100000.0	Nal
97	0	3	2019-01-31T17:36:31.000-08:00	CO	810xx	52.4	85000.0	Nal
98	0	9	2019-01-22T14:13:06.000-08:00	TN	378xx	76.1	15000.0	105000.
99	0	2	2019-02-03T04:53:36.000-08:00	NY	102xx	10.6	75000.0	Nal
100	0	6	2019-01-31T16:38:56.000-08:00	IN	465xx	81.6	62400.0	159661.

5 rows × 119 columns

Historical Data

In [4]:
```
f_2017_Q1 = pd.read_csv('LoanStats_securev1_2017Q1.csv', skiprows = 1)
f_2017_Q2 = pd.read_csv('LoanStats_securev1_2017Q2.csv', skiprows = 1)
f_2017_Q3 = pd.read_csv('LoanStats_securev1_2017Q3.csv', skiprows = 1)
f_2017_Q4 = pd.read_csv('LoanStats_securev1_2017Q4.csv', skiprows = 1)
f_2017_Q1.head()
```

```
:\Users\Melody Zhang\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3020:
typeWarning: Columns (0,118) have mixed types. Specify dtype option on import or set lo
_memory=False.
 interactivity=interactivity, compiler=compiler, result=result)
:\Users\Melody Zhang\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3020:
typeWarning: Columns (0) have mixed types. Specify dtype option on import or set low_me
ory=False.
 interactivity=interactivity, compiler=compiler, result=result)
```

Out[4]:

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	installment	grade	su
0	103771912	NaN	16000.0	16000.0	16000.0	60 months	12.74%	361.93	C	
1	104090257	NaN	11875.0	11875.0	11875.0	36 months	11.44%	391.26	B	
2	104045864	NaN	12000.0	12000.0	12000.0	36 months	7.99%	375.99	A	
3	104240334	NaN	25000.0	25000.0	25000.0	36 months	15.99%	878.81	C	
4	104170234	NaN	1500.0	1500.0	1500.0	36 months	5.32%	45.18	A	

5 rows × 151 columns

In [5]:
```
rames = [df_2017_Q1, df_2017_Q2, df_2017_Q3, df_2017_Q4]
f_2017 = pd.concat(frames)
f_2017.describe()
```

Out[5]:

	member_id	loan_amnt	funded_amnt	funded_amnt_inv	installment	annual_inc	desc	
count	0.0	443579.000000	443579.000000	443579.000000	443579.000000	4.435790e+05	0.0	4430
mean	NaN	14845.060463	14845.060463	14840.839712	442.495150	8.010556e+04	NaN	
std	NaN	9632.634949	9632.634949	9630.180034	283.153059	2.041056e+05	NaN	
min	NaN	1000.000000	1000.000000	975.000000	7.610000	0.000000e+00	NaN	
25%	NaN	7200.000000	7200.000000	7200.000000	237.060000	4.700000e+04	NaN	
50%	NaN	12000.000000	12000.000000	12000.000000	364.940000	6.600000e+04	NaN	
75%	NaN	20000.000000	20000.000000	20000.000000	593.960000	9.500000e+04	NaN	
max	NaN	40000.000000	40000.000000	40000.000000	1719.830000	1.100000e+08	NaN	9

8 rows × 112 columns

In [5]:
```
f_2017.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 443587 entries, 0 to 118649
Columns: 151 entries, id to settlement_term
dtypes: float64(112), object(39)
memory usage: 514.4+ MB
```

In [40]:
```
rint('len of num_feature is:', len(num_feature), 'ob_feature:', len(ob_feature))
```

```
len of num_feature is: 110 ob_feature: 39
```

In [6]:
```
f_2017 = df_2017.dropna(axis = 0, thresh=10, how="any")
f_2017 = df_2017.dropna(axis = 1, thresh=400000, how="any")
f_2017 = df_2017.dropna(axis = 1, how = 'all')
f_2017
```

Out[6]:

	id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	installment	grade	sub_grad
0	103771912	16000.0	16000.0	16000.0	60 months	12.74%	361.93	C	C
1	104090257	11875.0	11875.0	11875.0	36 months	11.44%	391.26	B	B
2	104045864	12000.0	12000.0	12000.0	36 months	7.99%	375.99	A	A
3	104240334	25000.0	25000.0	25000.0	36 months	15.99%	878.81	C	C
4	104170234	1500.0	1500.0	1500.0	36 months	5.32%	45.18	A	A
5	104190252	6000.0	6000.0	6000.0	36 months	11.44%	197.69	B	B
6	103951365	20000.0	20000.0	20000.0	36 months	8.24%	628.95	B	B
7	104190255	35000.0	35000.0	35000.0	60 months	25.49%	1037.38	E	E
8	103912095	5000.0	5000.0	5000.0	36 months	7.99%	156.66	A	A
9	103618433	17000.0	17000.0	17000.0	60 months	13.99%	395.48	C	C
10	103508738	16000.0	16000.0	16000.0	60 months	12.74%	361.93	C	C
11	102675947	14000.0	14000.0	14000.0	60 months	15.99%	340.38	C	C
12	104048783	13000.0	13000.0	13000.0	36 months	11.44%	428.32	B	B
13	104090121	20000.0	20000.0	20000.0	36 months	15.99%	703.05	C	C
14	104049706	6000.0	6000.0	6000.0	36 months	8.24%	188.69	B	B
15	98692718	20000.0	20000.0	20000.0	36 months	12.74%	671.38	C	C
16	104220223	5000.0	5000.0	5000.0	36 months	14.99%	173.31	C	C
17	100075977	13000.0	13000.0	13000.0	36 months	6.99%	401.35	A	A
18	104046412	5000.0	5000.0	5000.0	36 months	8.24%	157.24	B	B
19	99892449	7100.0	7100.0	7100.0	36 months	11.39%	233.76	B	B
20	104047067	10000.0	10000.0	10000.0	36 months	7.99%	313.32	A	A
21	104048642	10000.0	10000.0	10000.0	36 months	7.49%	311.02	A	A
22	104280288	3025.0	3025.0	3025.0	36 months	6.99%	93.39	A	A
23	104028593	4600.0	4600.0	4600.0	36 months	11.39%	151.45	B	B
24	104070168	9000.0	9000.0	9000.0	36 months	11.39%	296.32	B	B

	id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	installment	grade	sub_grad
25	104210150	12000.0	12000.0	12000.0	36 months	6.99%	370.48	A	A
26	104048967	5000.0	5000.0	5000.0	36 months	25.49%	200.10	E	E
27	103961780	10000.0	10000.0	10000.0	36 months	7.49%	311.02	A	A
28	103608248	4000.0	4000.0	4000.0	36 months	10.49%	130.00	B	B
29	103991278	15000.0	15000.0	15000.0	60 months	13.49%	345.08	C	C
...	...	...	...	...	...	...	...	...	.
118618	119300207	20700.0	20700.0	20700.0	60 months	12.62%	466.98	C	C
118619	119695811	32400.0	32400.0	32375.0	36 months	20.00%	1204.11	D	D
118620	119703711	30000.0	30000.0	30000.0	36 months	21.45%	1137.20	D	D
118621	119699773	21000.0	21000.0	21000.0	36 months	17.09%	749.65	D	D
118622	119281569	15000.0	15000.0	15000.0	60 months	30.75%	492.24	F	F
118623	119402310	10000.0	10000.0	10000.0	36 months	9.44%	320.05	B	B
118624	117169380	20000.0	20000.0	20000.0	36 months	13.59%	679.58	C	C
118625	119260414	35000.0	35000.0	34725.0	60 months	30.75%	1148.55	F	F
118626	119394625	10500.0	10500.0	10500.0	36 months	13.59%	356.78	C	C
118627	119342626	16000.0	16000.0	16000.0	60 months	18.06%	406.82	D	D
118628	119321840	32000.0	32000.0	32000.0	60 months	28.72%	1010.30	F	F
118629	119373285	35000.0	35000.0	35000.0	36 months	7.97%	1096.29	A	A
118630	119314998	10000.0	10000.0	10000.0	60 months	28.72%	315.72	F	F
118631	117880015	35000.0	35000.0	35000.0	60 months	17.09%	871.54	D	D
118632	119300456	25000.0	25000.0	25000.0	60 months	30.65%	818.85	F	F
118633	118620436	15000.0	15000.0	15000.0	36 months	10.91%	490.45	B	B
118634	119231255	19000.0	19000.0	19000.0	60 months	14.08%	442.89	C	C
118635	116788357	18000.0	18000.0	18000.0	60 months	30.65%	589.57	F	F
118636	119344895	17950.0	17950.0	17950.0	60 months	30.17%	582.62	F	F
118637	118120119	30000.0	30000.0	30000.0	60 months	17.09%	747.03	D	D

	id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	installment	grade	sub_grad
118638	119317668	10000.0	10000.0	10000.0	36 months	12.62%	335.12	C	C
118639	119361010	15000.0	15000.0	15000.0	60 months	16.02%	364.94	C	C
118640	119130644	3500.0	3500.0	3500.0	36 months	10.91%	114.44	B	B
118641	119311699	21000.0	21000.0	21000.0	60 months	30.94%	691.60	G	G
118642	119297966	30000.0	30000.0	30000.0	60 months	30.75%	984.47	F	F
118643	119305438	12000.0	12000.0	12000.0	60 months	14.08%	279.72	C	C
118644	117579665	12000.0	12000.0	12000.0	60 months	25.82%	358.01	E	E
118645	118660967	10000.0	10000.0	10000.0	36 months	11.99%	332.10	B	B
118646	118667978	12000.0	12000.0	12000.0	60 months	21.45%	327.69	D	D
118647	118607680	16550.0	16550.0	16550.0	60 months	21.45%	451.94	D	D

443579 rows × 105 columns

In [7]:
```
ols=df_2017.columns.values
ols
```

Out[7]:
```
array(['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',
       'url', 'purpose', 'title', 'zip_code', 'addr_state', 'dti',
       'delinq_2yrs', 'earliest_cr_line', 'fico_range_low',
       'fico_range_high', 'inq_last_6mths', 'open_acc', 'pub_rec',
       'revol_bal', 'revol_util', 'total_acc', 'initial_list_status',
       'out_prncp', 'out_prncp_inv', 'total_pymnt', 'total_pymnt_inv',
       'total_rec_prncp', 'total_rec_int', 'total_rec_late_fee',
       'recoveries', 'collection_recovery_fee', 'last_pymnt_d',
       'last_pymnt_amnt', 'last_credit_pull_d', 'last_fico_range_high',
       'last_fico_range_low', 'collections_12_mths_ex_med', 'policy_code',
       'application_type', 'acc_now_delinq', 'tot_coll_amt',
       'tot_cur_bal', 'open_acc_6m', 'open_act_il', 'open_il_12m',
       'open_il_24m', 'mths_since_rcnt_il', 'total_bal_il', 'open_rv_12m',
       'open_rv_24m', 'max_bal_bc', 'all_util', 'total_rev_hi_lim',
       'inq_fi', 'total_cu_tl', 'inq_last_12m', 'acc_open_past_24mths',
       'avg_cur_bal', 'bc_open_to_buy', 'bc_util',
       'chargeoff_within_12_mths', 'delinq_amnt', 'mo_sin_old_il_acct',
       'mo_sin_old_rev_tl_op', 'mo_sin_rcnt_rev_tl_op', 'mo_sin_rcnt_tl',
       'mort_acc', 'mths_since_recent_bc', 'num_accts_ever_120_pd',
       'num_actv_bc_tl', 'num_actv_rev_tl', 'num_bc_sats', 'num_bc_tl',
       'num_il_tl', 'num_op_rev_tl', 'num_rev_accts',
       'num_rev_tl_bal_gt_0', 'num_sats', 'num_tl_120dpd_2m',
       'num_tl_30dpd', 'num_tl_90g_dpd_24m', 'num_tl_op_past_12m',
       'pct_tl_nvr_dlq', 'percent_bc_gt_75', 'pub_rec_bankruptcies',
       'tax_liens', 'tot_hi_cred_lim', 'total_bal_ex_mort',
       'total_bc_limit', 'total_il_high_credit_limit', 'hardship_flag',
       'disbursement_method', 'debt_settlement_flag'], dtype=object)
```

In [8]:
```
um_feature=[]
b_feature=[]
or col in cols:
   if df_2017[col].dtype == 'object':
       ob_feature.append(col)
   else:
       num_feature.append(col)
```

In [9]:
```
um_feature
```

Out[9]:
```
['loan_amnt',
 'funded_amnt',
 'funded_amnt_inv',
 'installment',
 'annual_inc',
 'dti',
 'delinq_2yrs',
 'fico_range_low',
 'fico_range_high',
 'inq_last_6mths',
 'open_acc',
 'pub_rec',
 'revol_bal',
 'total_acc',
 'out_prncp',
 'out_prncp_inv',
 'total_pymnt',
 'total_pymnt_inv',
 'total_rec_prncp',
 'total_rec_int',
 'total_rec_late_fee',
 'recoveries',
 'collection_recovery_fee',
 'last_pymnt_amnt',
 'last_fico_range_high',
 'last_fico_range_low',
 'collections_12_mths_ex_med',
 'policy_code',
 'acc_now_delinq',
 'tot_coll_amt',
 'tot_cur_bal',
 'open_acc_6m',
 'open_act_il',
 'open_il_12m',
 'open_il_24m',
 'mths_since_rcnt_il',
 'total_bal_il',
 'open_rv_12m',
 'open_rv_24m',
 'max_bal_bc',
 'all_util',
 'total_rev_hi_lim',
 'inq_fi',
 'total_cu_tl',
 'inq_last_12m',
 'acc_open_past_24mths',
 'avg_cur_bal',
 'bc_open_to_buy',
 'bc_util',
 'chargeoff_within_12_mths',
 'delinq_amnt',
 'mo_sin_old_il_acct',
 'mo_sin_old_rev_tl_op',
 'mo_sin_rcnt_rev_tl_op',
 'mo_sin_rcnt_tl',
 'mort_acc',
 'mths_since_recent_bc',
 'num_accts_ever_120_pd',
 'num_actv_bc_tl',
 'num_actv_rev_tl',
 'num_bc_sats',
 'num_bc_tl',
 'num_il_tl',
```

```
 'num_op_rev_tl',
 'num_rev_accts',
 'num_rev_tl_bal_gt_0',
 'num_sats',
 'num_tl_120dpd_2m',
 'num_tl_30dpd',
 'num_tl_90g_dpd_24m',
 'num_tl_op_past_12m',
 'pct_tl_nvr_dlq',
 'percent_bc_gt_75',
 'pub_rec_bankruptcies',
 'tax_liens',
 'tot_hi_cred_lim',
 'total_bal_ex_mort',
 'total_bc_limit',
 'total_il_high_credit_limit']
```

In [10]:
```
b_feature
```

Out[10]:
```
['id',
 'term',
 'int_rate',
 'grade',
 'sub_grade',
 'emp_title',
 'emp_length',
 'home_ownership',
 'verification_status',
 'issue_d',
 'loan_status',
 'pymnt_plan',
 'url',
 'purpose',
 'title',
 'zip_code',
 'addr_state',
 'earliest_cr_line',
 'revol_util',
 'initial_list_status',
 'last_pymnt_d',
 'last_credit_pull_d',
 'application_type',
 'hardship_flag',
 'disbursement_method',
 'debt_settlement_flag']
```

In [10]:
```
rint('len of num_feature is:', len(num_feature), 'ob_feature:', len(ob_feature))
```

```
len of num_feature is: 79 ob_feature: 26
```

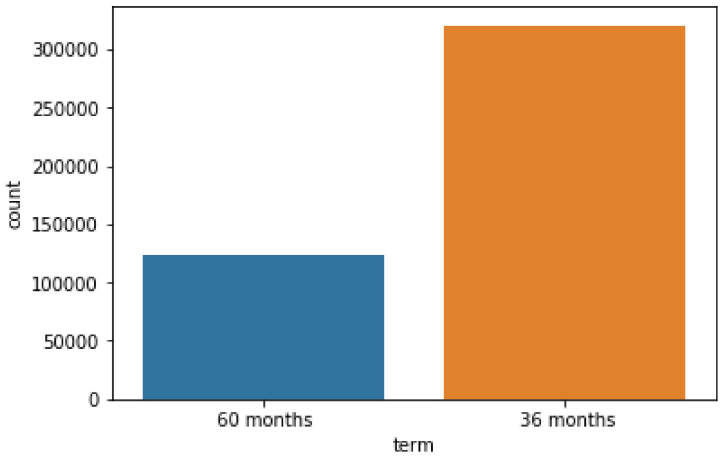
In [12]:
```
f_2017['grade'].value_counts()
```

Out[12]:
```
    145144
    133127
     78796
     56646
     20167
      6242
      3457
ame: grade, dtype: int64
```

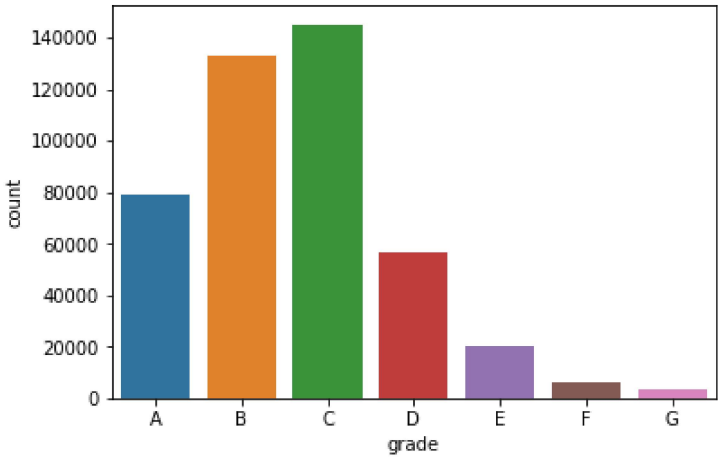
In [18]:
```
f_2017['term'].values
```

Out[18]:
```
array([' 60 months', ' 36 months', ' 36 months', ..., ' 36 months',
       ' 60 months', ' 60 months'], dtype=object)
```

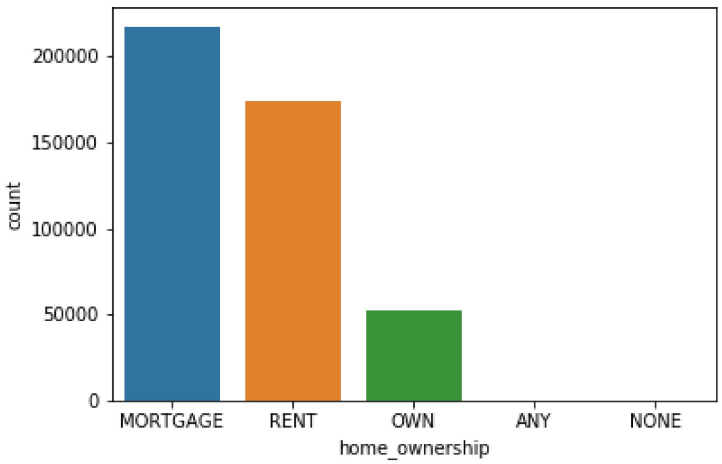
In [23]:
```
ns.countplot(x='term',data=df_2017)
lt.show()
```



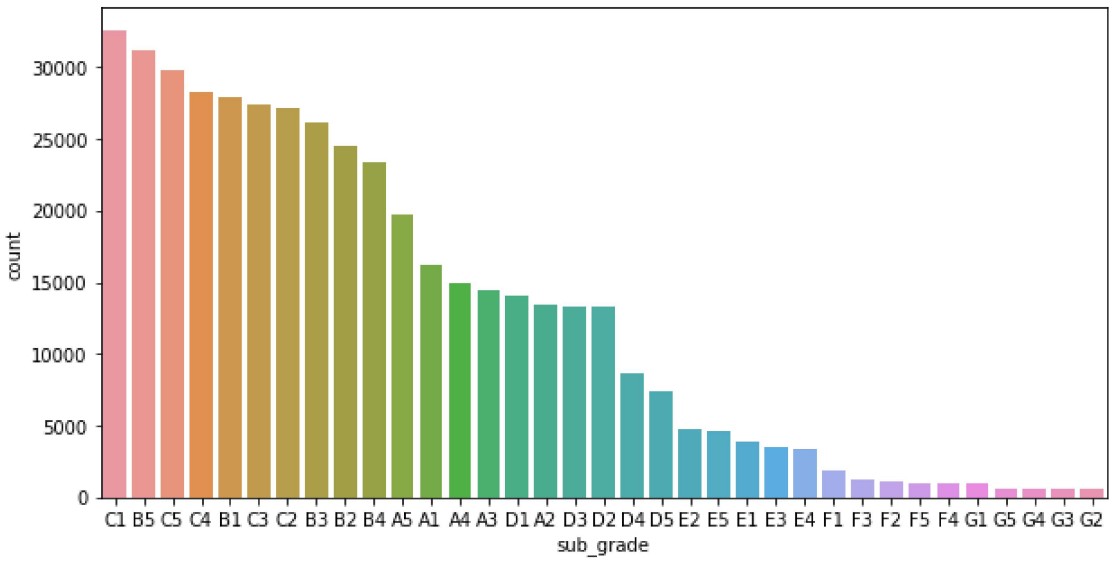
In [30]:
```
ns.countplot(x='grade',data=df_2017, order = ['A','B','C','D','E','F','G'])
lt.show()
```



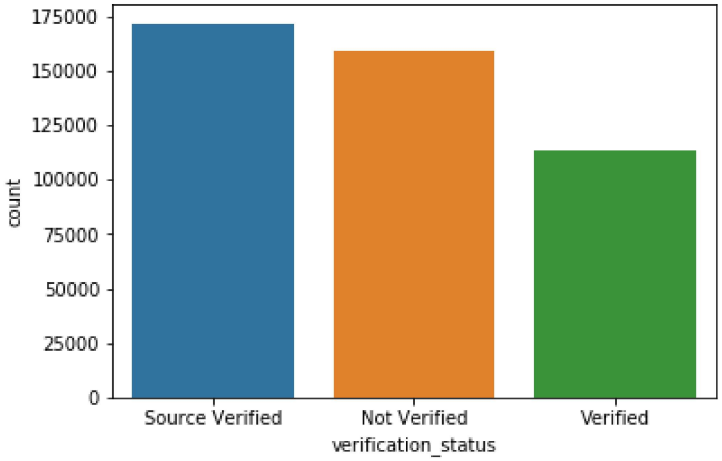
In [20]:
```
ns.countplot(x='home_ownership',data=df_2017, order = df_2017['home_ownership'].value_counts().index)
lt.show()
```



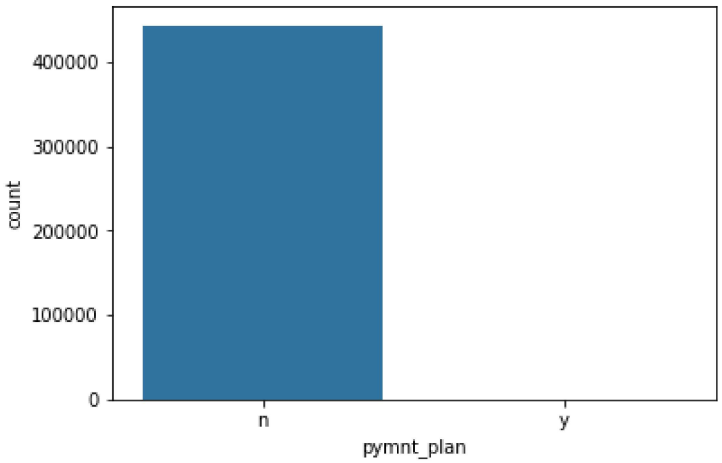
In [21]:
```
lt.figure(figsize=[10,5])
ns.countplot(x='sub_grade',data=df_2017, order = df_2017['sub_grade'].value_counts().index)
lt.show()
```



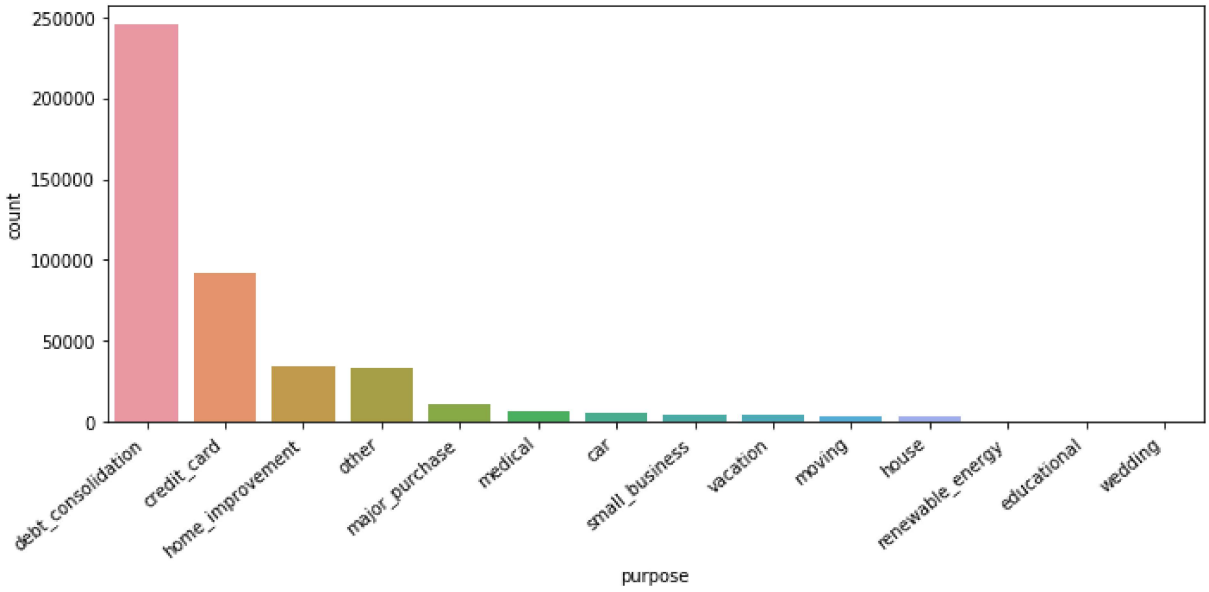
In [24]:
```
ns.countplot(x='verification_status',data=df_2017, order = df_2017['verification_statu
'].value_counts().index)
lt.show()
```



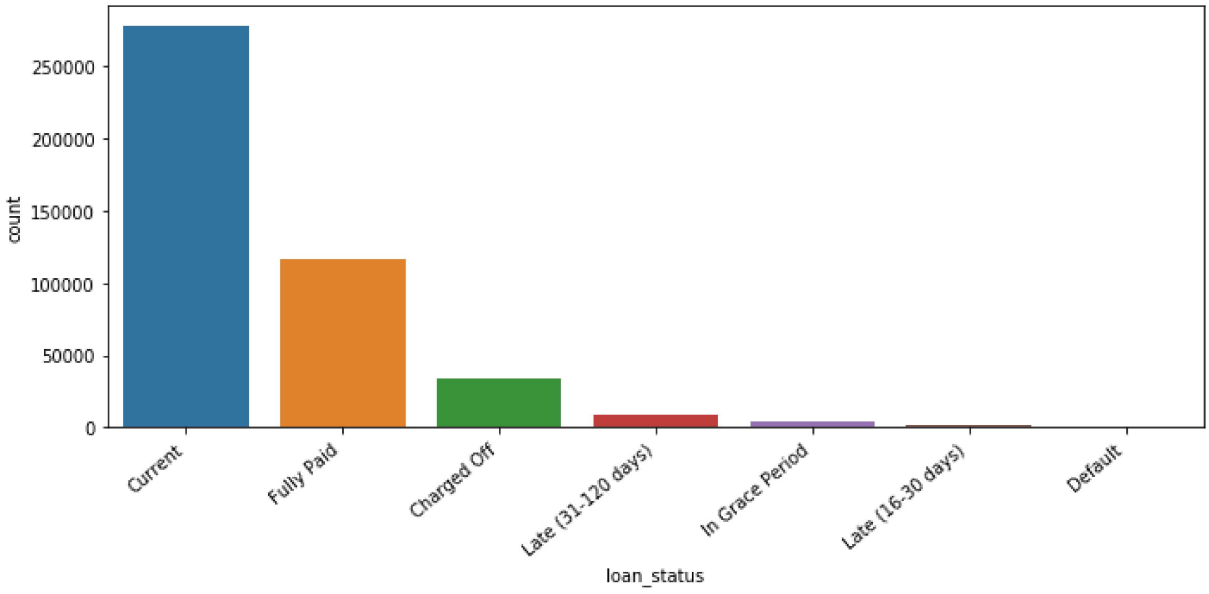
In [25]:
```
ns.countplot(x='pymnt_plan',data=df_2017, order = df_2017['pymnt_plan'].value_counts().
ndex)
lt.show()
```



In [29]:
```
lt.figure(figsize=[10,5])
x = sns.countplot(x='purpose',data=df_2017, order = df_2017['purpose'].value_counts().index)
x.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")
lt.tight_layout()
lt.show()
```



In [32]:
```
lt.figure(figsize=[10,5])
x = sns.countplot(x='loan_status',data=df_2017, order = df_2017['loan_status'].value_counts().index)
x.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")
lt.tight_layout()
lt.show()
```



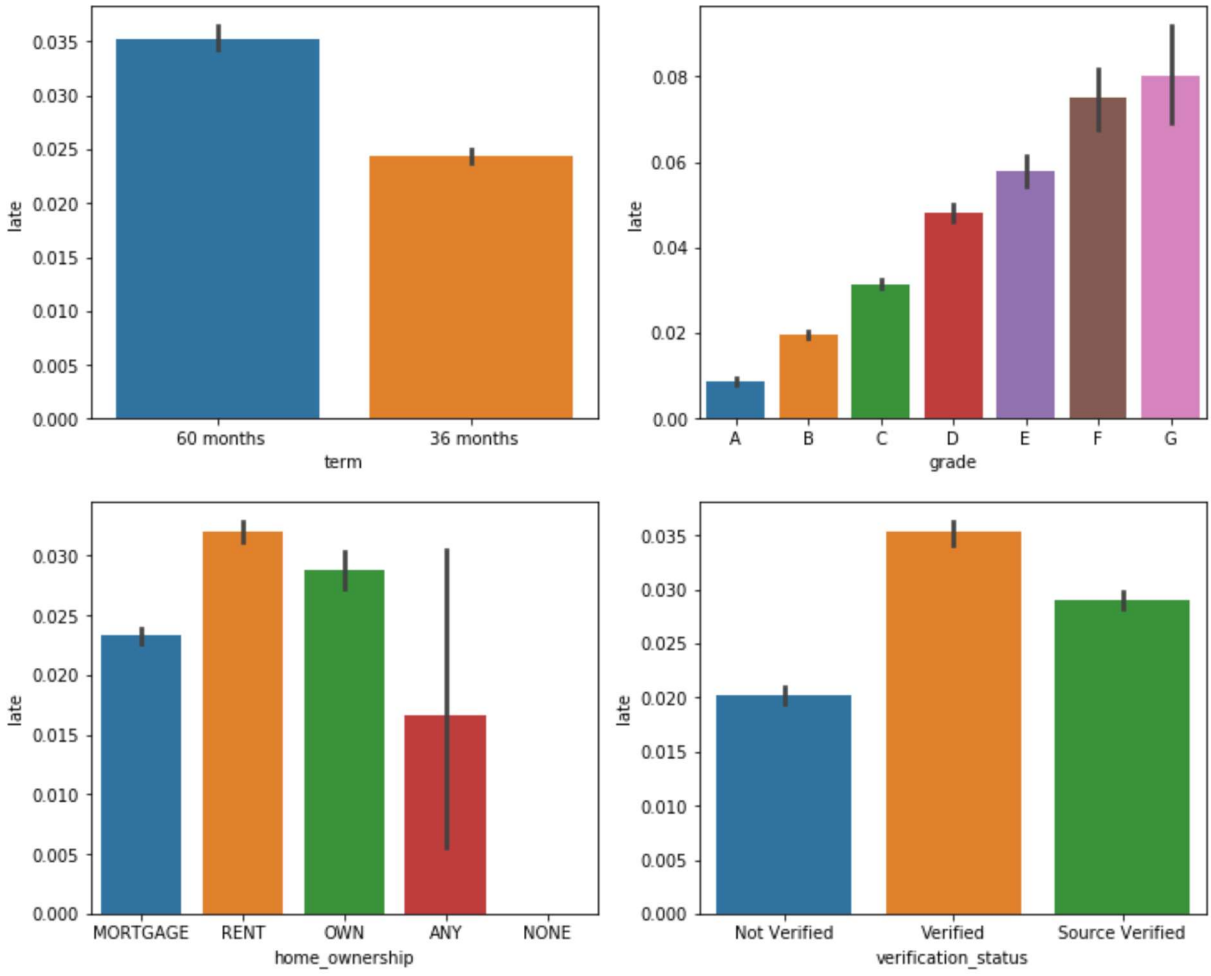
In [33]:
```
f_2017['loan_status'].value_counts()
```

Out[33]:
```
Current               278150
Fully Paid            116178
Charged Off            33595
Late (31-120 days)      9153
In Grace Period         4433
Late (16-30 days)       2058
Default                   12
Name: loan_status, dtype: int64
```

In [34]:
```
f_2017['late'] = df_2017['loan_status'].map({'Current': 0, 'Fully Paid': 0, 'Charged of
': 0, 'Late (31-120 days)': 1, 'In Grace Period': 0, 'Late (16-30 days)': 1, 'Default':
})
```

In [36]:
```
lt.figure(figsize=[12,10])

lt.subplot(221)
ns.barplot(x = 'term', y = 'late', data = df_2017)
People with higher socieconomic class had a higher rate of survival

lt.subplot(222)
ns.barplot(x = 'grade', y = 'late', data = df_2017, order = ['A','B','C','D','E','F',
'G'])
People embarked at C are more likely to survive

lt.subplot(223)
ns.barplot(x='home_ownership',y='late',data = df_2017)
People with a recorded Cabin number are more likely to survive

lt.subplot(224)
ns.barplot(x='verification_status',y='late',data = df_2017)
```

Out[36]: <matplotlib.axes._subplots.AxesSubplot at 0x1519dbce668>



In [43]:
```
 df_2017['int_rate'].value_counts()
f_2017['int_rate_float'] = df_2017['int_rate'].astype('str').map(lambda x: x[:-1]).asty
e(float)
```

In [42]:
```
lt.figure(figsize=[12,10])

lt.subplot(221)
ns.barplot(x = 'term', y = 'int_rate_float', data = df_2017)

lt.subplot(222)
ns.barplot(x = 'grade', y = 'int_rate_float', data = df_2017, order = ['A','B','C','D',
'E','F','G'])

lt.subplot(223)
ns.barplot(x='home_ownership',y='int_rate_float',data = df_2017)

lt.subplot(224)
ns.barplot(x='verification_status',y='int_rate_float',data = df_2017)
```

Out[42]: <matplotlib.axes._subplots.AxesSubplot at 0x151b9bca198>