Recommendations: According to the data, large monthly EMIs increase the likelihood of loan default for loans with a shorter 36-month term. Also prone to default are clients who have mortgaged or rented their properties. Individuals earning less than $50,000 per year are also at a higher risk. The fact that confirmed clients defaulted more frequently than non-certified ones raises the possibility of corruption or a verification failure. Due to large sums, debt consolidation loan recipients frequently experienced defaults.

In [53]:

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

In [3]:

```
#Load the loan dataset
loan = pd.read_csv("loan.csv")
```

```
/var/folders/rj/jgvbh2h573g0q1kmy76d0bx00000gp/T/ipykernel_40878/3592418794.py:1: DtypeWarning: Columns (47) have mixed types. Specify dtype option on import or set low_memory=False.
  loan = pd.read_csv("loan.csv")
```

In [5]:

```
#Reading top few lines on loan dataset to
loan.head()
```

Out[5]:

| | id | member_id | loan_amnt | funded_amnt |
|---|---|---|---|---|
| 0 | 1077501 | 1296599 | 5000 | 5000 |
| 1 | 1077430 | 1314167 | 2500 | 2500 |
| 2 | 1077175 | 1313524 | 2400 | 2400 |
| 3 | 1076863 | 1277178 | 10000 | 10000 |
| 4 | 1075358 | 1311748 | 3000 | 3000 |

5 rows × 111 columns

In [17]:

```
#Dropping rows where all values are missin
loan.dropna(how='all')
```

Out[17]:

| | id | member_id | loan_amnt | funded_am |
|---|---|---|---|---|
| 0 | 1077501 | 1296599 | 5000 | 50 |
| 1 | 1077430 | 1314167 | 2500 | 25 |
| 2 | 1077175 | 1313524 | 2400 | 24 |
| 3 | 1076863 | 1277178 | 10000 | 100 |
| 4 | 1075358 | 1311748 | 3000 | 30 |
| ... | ... | ... | ... | |
| 39712 | 92187 | 92174 | 2500 | 25 |
| 39713 | 90665 | 90607 | 8500 | 85 |
| 39714 | 90395 | 90390 | 5000 | 50 |
| 39715 | 90376 | 89243 | 5000 | 50 |
| 39716 | 87023 | 86999 | 7500 | 75 |

39717 rows × 111 columns

In [33]:

```
#Dropping columns with all or at least 1 n
loan = loan.dropna(axis=1)
```

In [ ]:

```
#Filtering for Defaulters from loan datase
```

In [39]:

```
#checking all loan status values
loan['loan_status'].unique
```

Out[39]:

```
<bound method Series.unique of 0          Fully Paid
1        Charged Off
2         Fully Paid
3         Fully Paid
4            Current
            ...
39712     Fully Paid
39713     Fully Paid
39714     Fully Paid
39715     Fully Paid
39716     Fully Paid
Name: loan_status, Length: 39717, dtype: object>
```

In [43]:

```
#Filtering for loan status with value "Cha
loan_default = loan[loan['loan_status'].st
```

In [45]:

```
loan_default.info
```

Out[45]:

```
<bound method DataFrame.info of
id  member_id  loan_amnt  funded_amnt  funded_amnt_inv  \
1      1077430    1314167       2500
2500            2500.0
8      1071795    1306957       5600
5600            5600.0
9      1071570    1306721       5375
5375            5350.0
12     1064687    1298717       9000
9000            9000.0
14     1069057    1303503      10000
10000           10000.0
...        ...        ...        ...
...              ...
39667   118823     118026       2500
2500             675.0
39668   118533     117783       2500
2500             825.0
39669   118523     118519       6500
6500             225.0
39678   113179     113093       1000
1000             950.0
39688   111227     111223      20000
20000           2800.0

            term int_rate  installment grade sub_grade  ... total_rec_prncp  \
1      60 months   15.27%        59.83
C        C4  ...          456.46
8      60 months   21.28%       152.39
F        F2  ...          162.02
9      60 months   12.69%       121.45
B        B5  ...          673.48
12     36 months   13.49%       305.38
C        C1  ...         1256.14
14     36 months   10.65%       325.74
B        B2  ...         5433.47
...          ...      ...          ...
...        ...  ...             ...
39667  36 months   12.80%        84.00
D        D4  ...         1706.01
39668  36 months    9.64%        80.26
B        B4  ...         1730.83
39669  36 months   15.01%       225.37
F        F1  ...         2886.21
39678  36 months   10.59%        32.55
C        C2  ...          544.02
39688  36 months   13.43%       678.08
E        E1  ...        16077.42

       total_rec_int total_rec_late_fee recoveries collection_recovery_fee  \
1             435.17               0.00
117.08                   1.1100
8             294.94               0.00
189.06                   2.0900
9             533.42               0.00
269.29                   2.5200
12            570.26               0.00
444.30                   4.1600
14           1393.42               0.00
645.10                   6.3145
...              ...                ...
...                     ...
39667         477.21               1.69
35.70                    0.3800
39668         354.44               1.36
0.00                     0.0000
39669        1168.14               0.00
0.00                     0.0000
39678         138.64               0.00
21.29                    0.2300
39688        4262.24               0.00
0.00                     0.0000

      last_pymnt_amnt policy_code application_type acc_now_delinq delinq_amnt
1              119.66           1       INDIVIDUAL              0           0
8              152.39           1       INDIVIDUAL              0           0
9              121.45           1       INDIVIDUAL              0           0
12             305.38           1       INDIVIDUAL              0           0
14             325.74           1       INDIVIDUAL              0           0
...               ...         ...
...               ...         ...
39667            1.76           1       INDIVIDUAL              0           0
39668            1.40           1       INDIVIDUAL              0           0
39669          225.37           1       INDIVIDUAL              0           0
39678           32.55           1       INDIVIDUAL              0           0
39688          678.08           1       INDIVIDUAL              0           0

[5627 rows x 43 columns]>
```

In [47]:

```
loan_default.shape
```

Out[47]:

```
(5627, 43)
```

In [49]:

```
loan_default.columns
```

Out[49]:

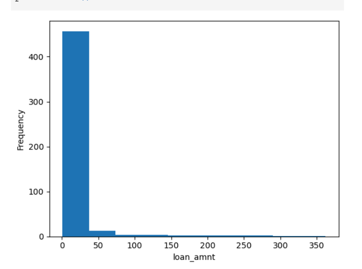
```
Index(['id', 'member_id', 'loan_amnt', 'funded_amnt', 'funded_amnt_inv',
       'term', 'int_rate', 'installment', 'grade', 'sub_grade',
       'home_ownership', 'annual_inc', 'verification_status', 'issue_d',
       'loan_status', 'pymnt_plan', 'url', 'purpose', 'zip_code', 'addr_state',
       'dti', 'delinq_2yrs', 'earliest_cr_line', 'inq_last_6mths', 'open_acc',
       'pub_rec', 'revol_bal', '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_amnt', 'policy_code',
       'application_type', 'acc_now_delinq', 'delinq_amnt'],
      dtype='object')
```

In [ ]:

```
#DATA ANALYSIS
```

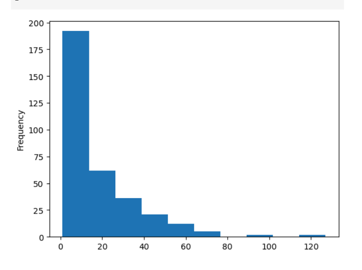
In [127]:

```
#UNI VARIATE ANALYSIS
#Variable 1 = loan_amnt
loan_default['loan_amnt'].value_counts().p
plt.xlabel("loan_amnt")
plt.show()
```



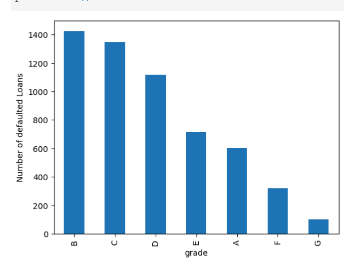

In [69]:

```
#Variable 3 = int_rate
loan_default['int_rate'].value_counts().pl
plt.show()
```



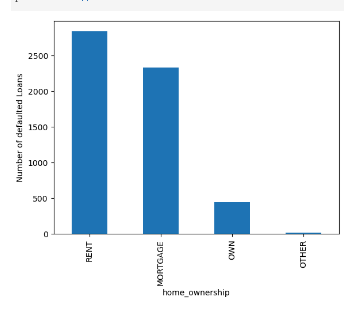

In [135]:

```
#Variable 4 = grade
loan_default['grade'].value_counts().plot.
plt.ylabel("Number of defaulted Loans")
plt.show()
```



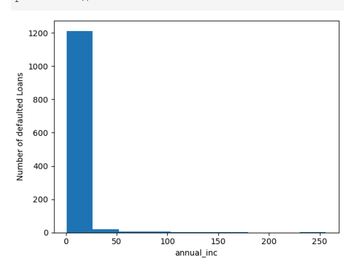

In [137]:

```
#Variable 5 = home_ownership
loan_default['home_ownership'].value_count
plt.ylabel("Number of defaulted Loans")
plt.show()
```



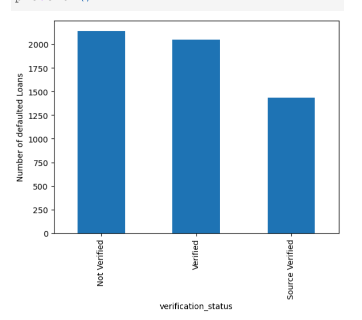

In [147]:

```
#Variable 6 = annual_inc
loan_default['annual_inc'].value_counts().
plt.xlabel("annual_inc")
plt.ylabel("Number of defaulted Loans")
plt.show()
```



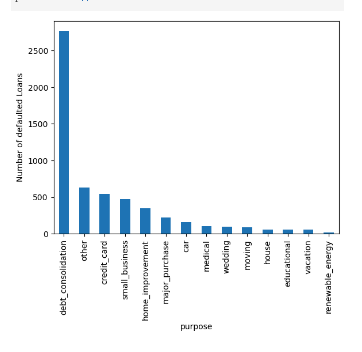

In [141]:

```
#Variable 7 = Verification Status
loan_default['verification_status'].value_
plt.ylabel("Number of defaulted Loans")
plt.show()
```



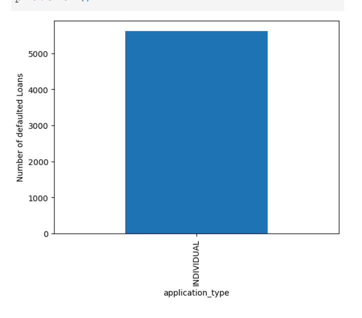

In [143]:

```
#Variable 8 = purpose
loan_default['purpose'].value_counts().plo
plt.ylabel("Number of defaulted Loans")
plt.show()
```




In [145]:

```
#Variable 9 = application_type
loan_default['application_type'].value_cou
plt.ylabel("Number of defaulted Loans")
plt.show()
```

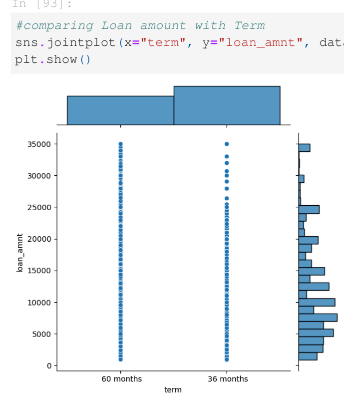



In [ ]:

```
#BI VARIATE ANALYSIS
```

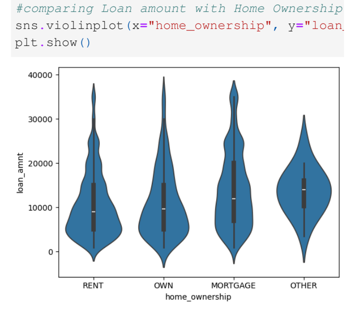
In [93]:

```
#comparing Loan amount with Term
sns.jointplot(x="term", y="loan_amnt", dat
plt.show()
```



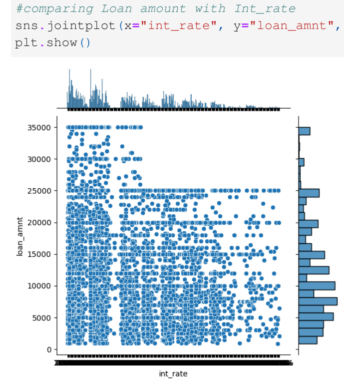

In [113]:

```
#comparing Loan amount with Home Ownership
sns.violinplot(x="home_ownership", y="loan
plt.show()
```




In [97]:

```
#comparing Loan amount with Int_rate
sns.jointplot(x="int_rate", y="loan_amnt",
plt.show()
```



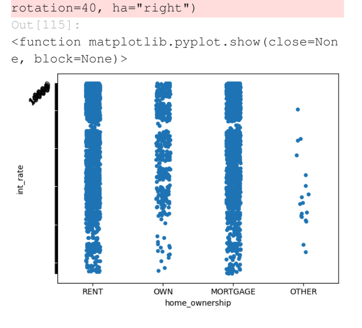

In [115]:

```
#comparing home ownership with int_rate
ay = sns.stripplot(x="home_ownership", y="
ay.set_yticklabels(ay.get_yticklabels(), r
plt.show
```

```
/var/folders/rj/jgvbh2h573g0q1kmy76d0bx00000gp/T/ipykernel_40878/3848597446.py:3: UserWarning: set_ticklabels() should only be used with a fixed number of ticks, i.e. after set_ticks() or using a FixedLocator.
  ay.set_yticklabels(ay.get_yticklabels(), rotation=40, ha="right")
```

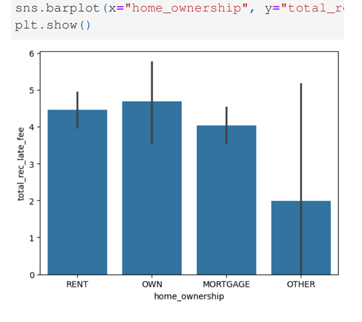
Out[115]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```




In [119]:

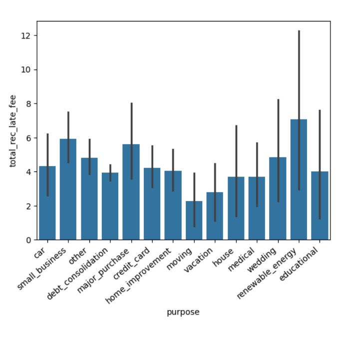
```
#comparing home ownership with total_rec_l
sns.barplot(x="home_ownership", y="total_r
plt.show()
```




In [123]:

```
#comparing purpose with total_rec_late_fee
x = sns.barplot(x="purpose", y="total_rec_
x.set_xticklabels(ax.get_xticklabels(), ro
plt.show()
```

```
/var/folders/rj/jgvbh2h573g0q1kmy76d0bx00000gp/T/ipykernel_40878/426503065.py:3: UserWarning: set_ticklabels() should only be used with a fixed number of ticks, i.e. after set_ticks() or using a FixedLocator.
  x.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")
```




```
In [ ]:
```

Final Summary: These recommendations from this analysis should be taken seriously for future loan awards, and more research is necessary to expedite the loan verification procedure. By doing this, we can lower the number of loan defaults in the future.