```
Dataset Information
This dataset contains information on default payments,
demographic factors, credit data, history of payment, and
bill statements of credit card clients from April 2005 to September 2005.
```

In [1]:
```python
#importing packages
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
from pandas import set_option
```

In [7]:
```python
#loading data
file = 'C:\\Users\\HP\\Documents\\WORKSPACE\\default_credit.xls'
df = pd.read_excel(file)
```

In [8]:
```python
df.head(10)
```

Out[8]:

| | ID | LIMIT_BAL | SEX | EDUCATION | MARRIAGE | AGE | PAY_0 | PAY_2 | PAY_3 | PAY_4 | ... | BILL_AMT4 | BILL_AMT5 | BILL_AMT6 | PAY_AMT1 | PAY_AMT2 | PAY_AMT3 | PAY_AM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 20000 | 2 | 2 | 1 | 24 | 2 | 2 | -1 | -1 | ... | 0 | 0 | 0 | 0 | 689 | 0 | |
| **1** | 2 | 120000 | 2 | 2 | 2 | 26 | -1 | 2 | 0 | 0 | ... | 3272 | 3455 | 3261 | 0 | 1000 | 1000 | 10 |
| **2** | 3 | 90000 | 2 | 2 | 2 | 34 | 0 | 0 | 0 | 0 | ... | 14331 | 14948 | 15549 | 1518 | 1500 | 1000 | 10 |
| **3** | 4 | 50000 | 2 | 2 | 1 | 37 | 0 | 0 | 0 | 0 | ... | 28314 | 28959 | 29547 | 2000 | 2019 | 1200 | 11 |
| **4** | 5 | 50000 | 1 | 2 | 1 | 57 | -1 | 0 | -1 | 0 | ... | 20940 | 19146 | 19131 | 2000 | 36681 | 10000 | 90 |
| **5** | 6 | 50000 | 1 | 1 | 2 | 37 | 0 | 0 | 0 | 0 | ... | 19394 | 19619 | 20024 | 2500 | 1815 | 657 | 10 |
| **6** | 7 | 500000 | 1 | 1 | 2 | 29 | 0 | 0 | 0 | 0 | ... | 542653 | 483003 | 473944 | 55000 | 40000 | 38000 | 202 |
| **7** | 8 | 100000 | 2 | 2 | 2 | 23 | 0 | -1 | -1 | 0 | ... | 221 | -159 | 567 | 380 | 601 | 0 | 5 |
| **8** | 9 | 140000 | 2 | 3 | 1 | 28 | 0 | 0 | 2 | 0 | ... | 12211 | 11793 | 3719 | 3329 | 0 | 432 | 10 |
| **9** | 10 | 20000 | 1 | 3 | 2 | 35 | -2 | -2 | -2 | -2 | ... | 0 | 13007 | 13912 | 0 | 0 | 0 | 130 |

10 rows × 25 columns

# DATA EXPLORATORY ANALYSIS

In [9]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30000 entries, 0 to 29999
Data columns (total 25 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   ID                          30000 non-null  int64
 1   LIMIT_BAL                   30000 non-null  int64
 2   SEX                         30000 non-null  int64
 3   EDUCATION                   30000 non-null  int64
 4   MARRIAGE                    30000 non-null  int64
 5   AGE                         30000 non-null  int64
 6   PAY_0                       30000 non-null  int64
 7   PAY_2                       30000 non-null  int64
 8   PAY_3                       30000 non-null  int64
 9   PAY_4                       30000 non-null  int64
 10  PAY_5                       30000 non-null  int64
 11  PAY_6                       30000 non-null  int64
 12  BILL_AMT1                   30000 non-null  int64
 13  BILL_AMT2                   30000 non-null  int64
 14  BILL_AMT3                   30000 non-null  int64
 15  BILL_AMT4                   30000 non-null  int64
 16  BILL_AMT5                   30000 non-null  int64
 17  BILL_AMT6                   30000 non-null  int64
 18  PAY_AMT1                    30000 non-null  int64
 19  PAY_AMT2                    30000 non-null  int64
 20  PAY_AMT3                    30000 non-null  int64
 21  PAY_AMT4                    30000 non-null  int64
 22  PAY_AMT5                    30000 non-null  int64
 23  PAY_AMT6                    30000 non-null  int64
 24  default payment next month  30000 non-null  int64
dtypes: int64(25)
memory usage: 5.7 MB
```

There are no missing values in the dataset

In [6]:
```
df.columns
```
Out[6]:
```
Index(['ID', 'LIMIT_BAL', 'SEX', 'EDUCATION', 'MARRIAGE', 'AGE', 'PAY_0',
       'PAY_2', 'PAY_3', 'PAY_4', 'PAY_5', 'PAY_6', 'BILL_AMT1', 'BILL_AMT2',
       'BILL_AMT3', 'BILL_AMT4', 'BILL_AMT5', 'BILL_AMT6', 'PAY_AMT1',
       'PAY_AMT2', 'PAY_AMT3', 'PAY_AMT4', 'PAY_AMT5', 'PAY_AMT6',
       'default payment next month'],
      dtype='object')
```
In [6]:
```
df.describe()
```
Out[6]:

| | ID | LIMIT_BAL | SEX | EDUCATION | MARRIAGE | AGE | PAY_0 | PAY_2 | PAY_3 | PAY_4 | ... | BILL_AMT4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 30000.000000 | 30000.000000 | 30000.000000 | 30000.000000 | 30000.000000 | 30000.000000 | 30000.000000 | 30000.000000 | 30000.000000 | 30000.000000 | ... | 30000.000000 |
| **mean** | 15000.500000 | 167484.322667 | 1.603733 | 1.853133 | 1.551867 | 35.485500 | -0.016700 | -0.133767 | -0.166200 | -0.220667 | ... | 43262.948967 |
| **std** | 8660.398374 | 129747.661567 | 0.489129 | 0.790349 | 0.521970 | 9.217904 | 1.123802 | 1.197186 | 1.196868 | 1.169139 | ... | 64332.856134 |
| **min** | 1.000000 | 10000.000000 | 1.000000 | 0.000000 | 0.000000 | 21.000000 | -2.000000 | -2.000000 | -2.000000 | -2.000000 | ... | -170000.000000 |
| **25%** | 7500.750000 | 50000.000000 | 1.000000 | 1.000000 | 1.000000 | 28.000000 | -1.000000 | -1.000000 | -1.000000 | -1.000000 | ... | 2326.750000 |
| **50%** | 15000.500000 | 140000.000000 | 2.000000 | 2.000000 | 2.000000 | 34.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | ... | 19052.000000 |
| **75%** | 22500.250000 | 240000.000000 | 2.000000 | 2.000000 | 2.000000 | 41.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | ... | 54506.000000 |
| **max** | 30000.000000 | 1000000.000000 | 2.000000 | 6.000000 | 3.000000 | 79.000000 | 8.000000 | 8.000000 | 8.000000 | 8.000000 | ... | 891586.000000 |

8 rows × 25 columns

In [ ]:
```
#plot to see the distrubution accross each features
```

In [7]:

```python
df.hist()
plt.gcf().set_size_inches(20,20)
plt.show()
```


There is an abnormal distrubution accross the features, but due to the fact that, we wont be buidling a model with this dataset, we will not be dealing with the scales since this does not affect visualization and relationships accross the features.

```
UNIVARIATE ANALYSIS
Analyse the key features to uncover more insight about the data.
```

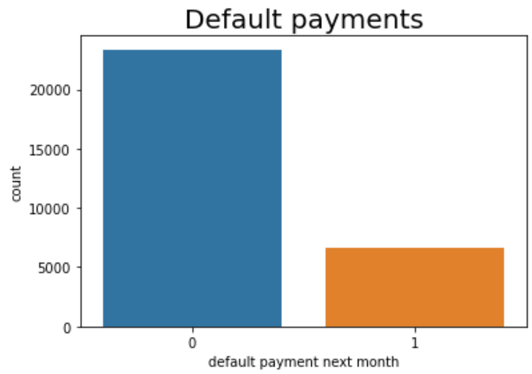
In [8]:

```python
print (df['default payment next month'].value_counts())
print(df['default payment next month'].value_counts(normalize=True))
```

```
0    23364
1     6636
Name: default payment next month, dtype: int64
0    0.7788
1    0.2212
Name: default payment next month, dtype: float64
```

In [9]:

```
%matplotlib inline
sb.countplot(data = df, x='default payment next month')
plt.title('Default payments', fontsize = 20)
plt.show()
```




From the above, we can see that 78%(23,364) of the customers ended up paying thier loans whilst 22%(6,636) ended up defaulting on their loans.

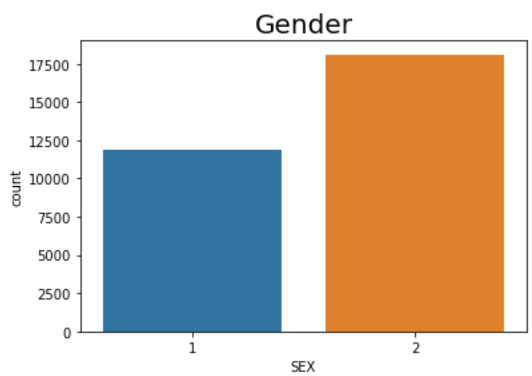
In [10]:

```
print (df['SEX'].value_counts())
print(df['SEX'].value_counts(normalize=True))
```

```
2    18112
1    11888
Name: SEX, dtype: int64
2    0.603733
1    0.396267
Name: SEX, dtype: float64
```

In [11]:

```
sb.countplot(data = df, x='SEX')
plt.title('Gender', fontsize = 20)
plt.show()
```



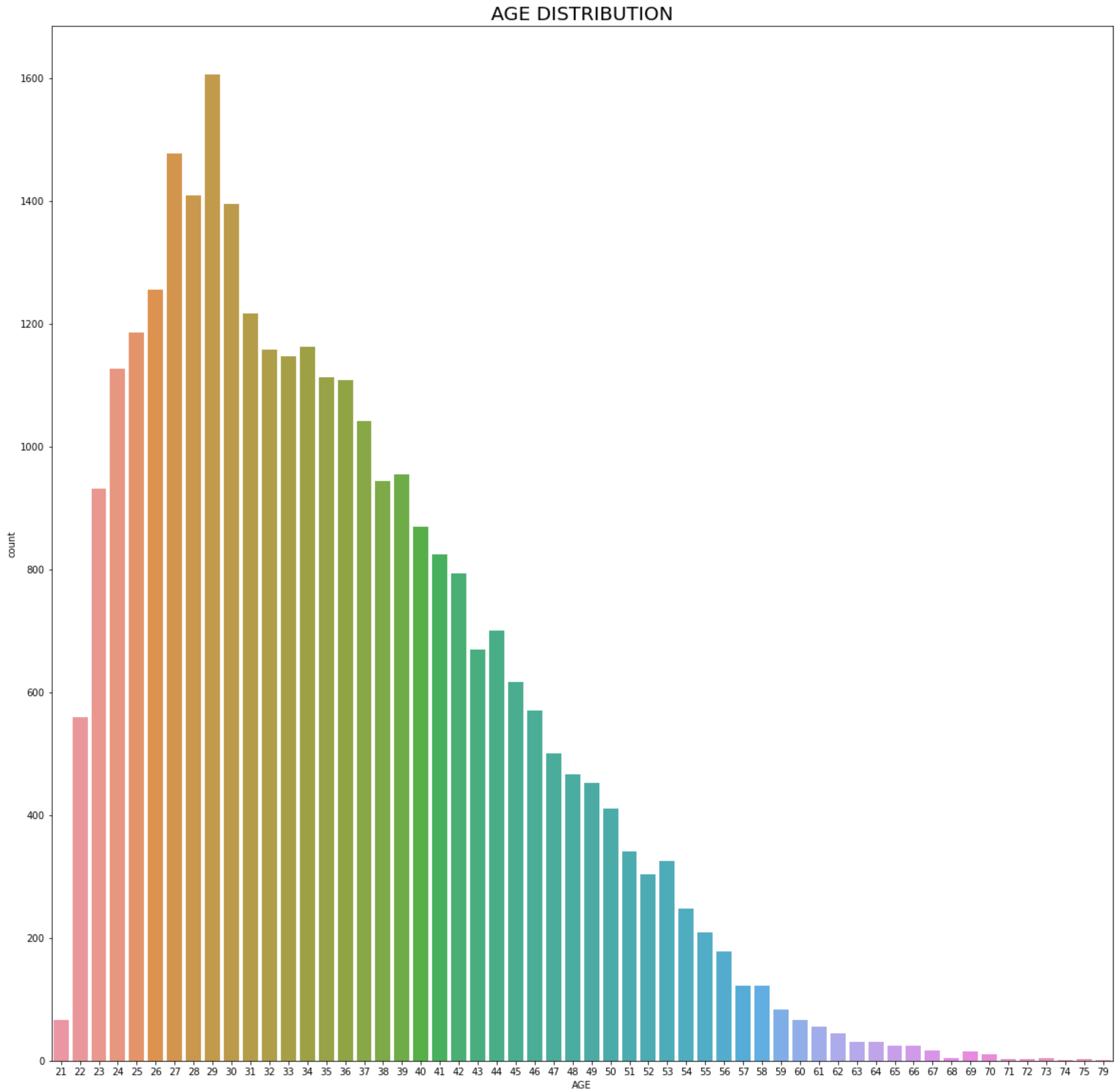

The customers comprises of 60% female (18,112) and 39% male (11,888)

In [12]:

```
print (df['AGE'].value_counts())
```

```
29    1605
27    1477
28    1409
30    1395
26    1256
31    1217
25    1186
34    1162
32    1158
33    1146
24    1127
35    1113
36    1108
37    1041
39     954
38     944
23     931
40     870
41     824
42     794
44     700
43     670
45     617
46     570
22     560
47     501
48     466
49     452
50     411
51     340
53     325
52     304
54     247
55     209
56     178
58     122
57     122
59      83
21      67
60      67
61      56
62      44
64      31
63      31
66      25
65      24
67      16
69      15
70      10
68       5
73       4
75       3
72       3
71       3
74       1
79       1
Name: AGE, dtype: int64
```

In [13]:
```
plt.figure(figsize = (20,20))
sb.countplot(data = df, x='AGE')
plt.title('AGE DISTRIBUTION', fontsize = 20)
plt.show()
```




The distrubution of ages tells us that the chunk of the customers lie
between the ages of 21-70, with the age of 29 being the most frequent age group.
This can help us understand which age group does most of the defualters come from as we dig deeper.

In [14]:
```
subset = (df.EDUCATION==0)|(df.EDUCATION==5)|(df.EDUCATION==6)
```

In [15]:
```
df.loc[subset,'EDUCATION']=4
```

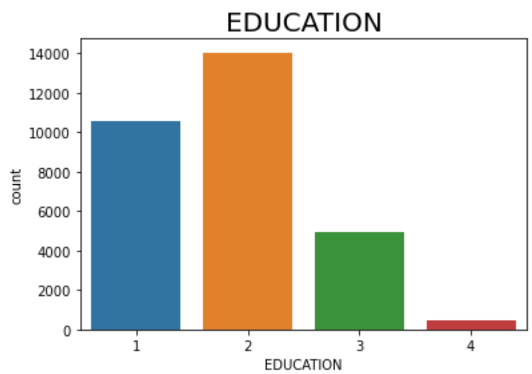
In [16]:
```
sorted(df.EDUCATION.unique())
```

Out[16]: [1, 2, 3, 4]

In [17]:
```
print (df['EDUCATION'].value_counts())
print(df['EDUCATION'].value_counts(normalize=True))
```

```
2    14030
1    10585
3     4917
4      468
Name: EDUCATION, dtype: int64
2    0.467667
1    0.352833
3    0.163900
4    0.015600
Name: EDUCATION, dtype: float64
```

In [18]:
```
sb.countplot(data = df, x='EDUCATION')
plt.title('EDUCATION', fontsize = 20)
plt.show()
```




```
Most of the customers are educated, with a chunk of them being in the university.
University 46% (14,030)
Graduate School 35% (10,585)
High School 16%(4,917)
could some of the default be coming from the fact that students are using these loans for accademic
purpose ?
```

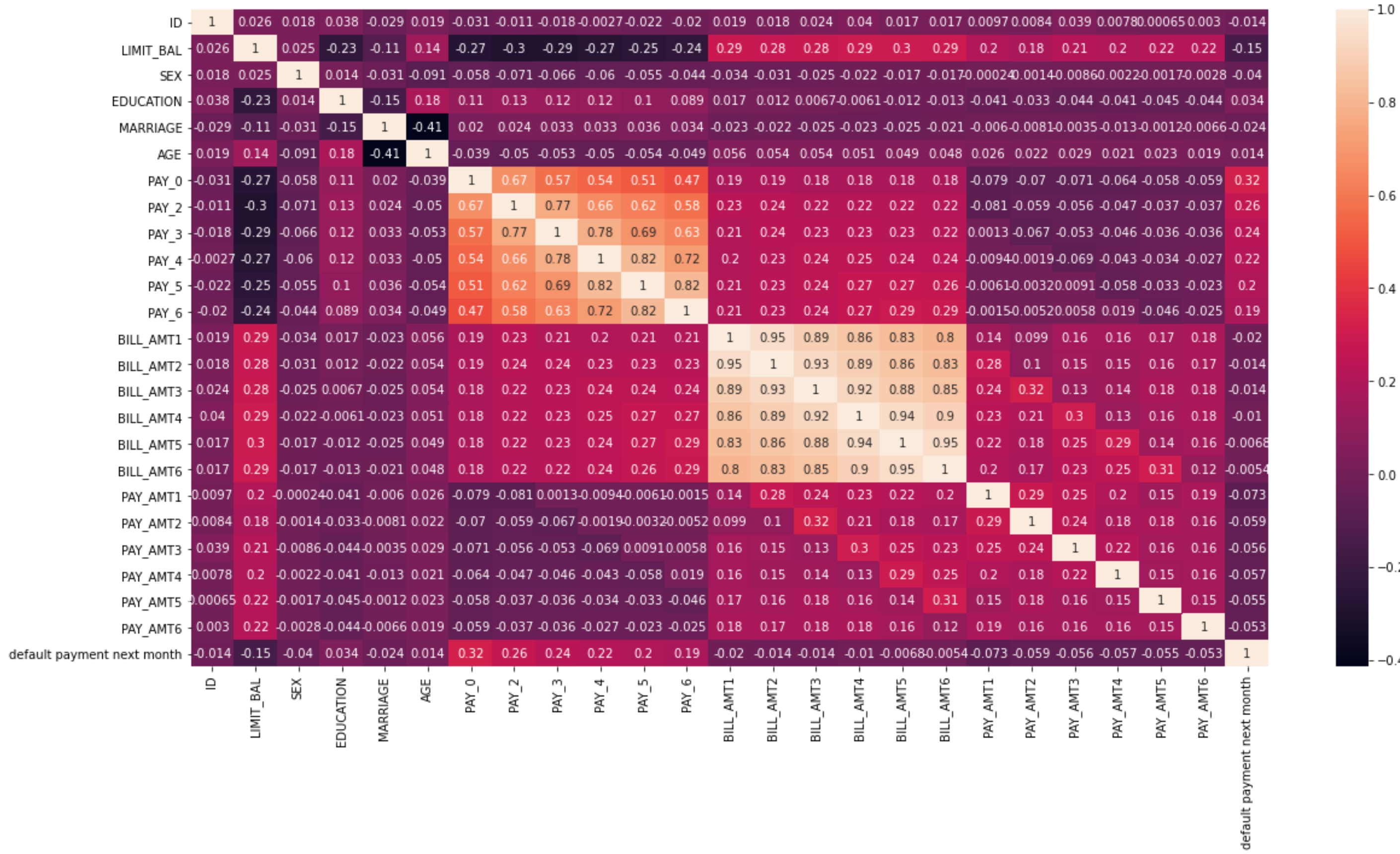
In [19]:
```
print (df['MARRIAGE'].value_counts())
print(df['MARRIAGE'].value_counts(normalize=True))
```

```
2    15964
1    13659
3      323
0       54
Name: MARRIAGE, dtype: int64
2    0.532133
1    0.455300
3    0.010767
0    0.001800
Name: MARRIAGE, dtype: float64
```

In [20]:
```
sb.countplot(data = df, x='MARRIAGE')
plt.title('MARRIAGE STATUS', fontsize = 20)
plt.show()
```


```
53%(15,969) of the customers are single whilst 45%(13,659) are Married.
could relationship staus have an effect of defualtung ? we wil also find out as we dig deeper.
```

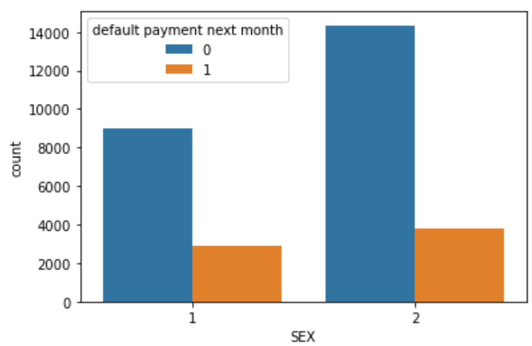
In [21]:
```
#corrlation analysis
plt.figure(figsize = (20,10))
sb.heatmap(df.corr(), annot = True)
set_option ('display.width', 1000)
```




From the above correlation heat map we can see that there seems to be a consistency of behaviour amogst customers.
There seems to be a high correlation in terms of repayment status amognst customers for each month that is April, May, June July, August. The same also applies to the bill statment for each month.

BIVARATE ANALYSIS

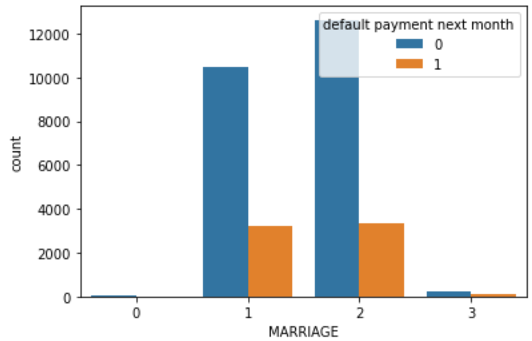
In [22]:
```
sb.countplot(x="SEX", hue="default payment next month", data=df)
plt.show()
```




Even though the females tend to pay more of thier loans than the male there seems to be no significant impact as to gender when it comes to defaulting.

In [23]:

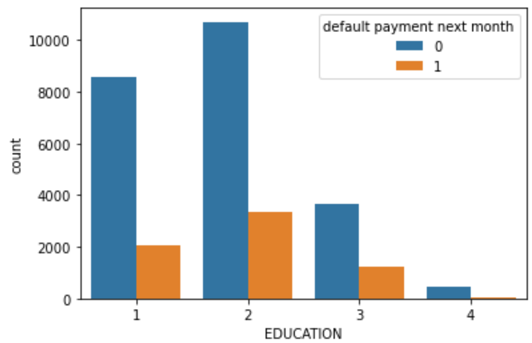
```
sb.countplot(x="MARRIAGE", hue="default payment next month", data=df)
plt.show()
```




from the above chart we can see that the single people
tend to pay back more loans but there also seem to be no significant impact on
relationship status when it comes to defaulting.

In [24]:

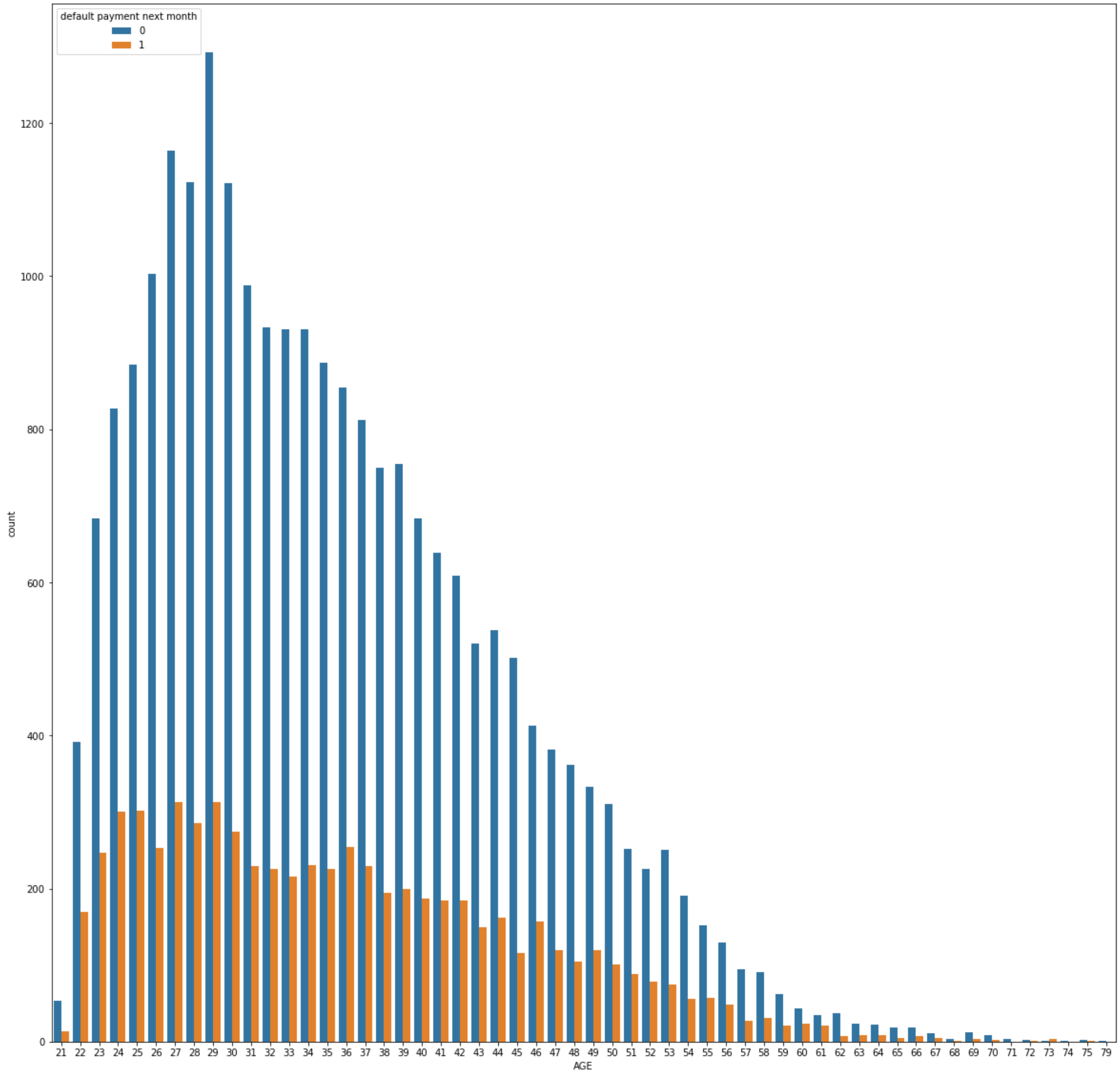
```
sb.countplot(x="EDUCATION", hue="default payment next month", data=df)
plt.show()
```




we can see from the chart that the customers in the university have the hghiest rate
of paying back loans and they also account for the highest defualters alonside graduate school.
it is safe to say education might not be the greatest indication for defualters.

In [25]:

```
plt.figure(figsize = (20,20))
sb.countplot(x="AGE", hue="default payment next month", data=df)
plt.show()
```




The average age of the customers is 35
Age group of 29 account for the highest number when it come to paying back loans
from the chart it can be seen that there is a consistency of default accross different
age groups with customers with ages 24,25,27 and 29, falling in the bracket of highest defaulters

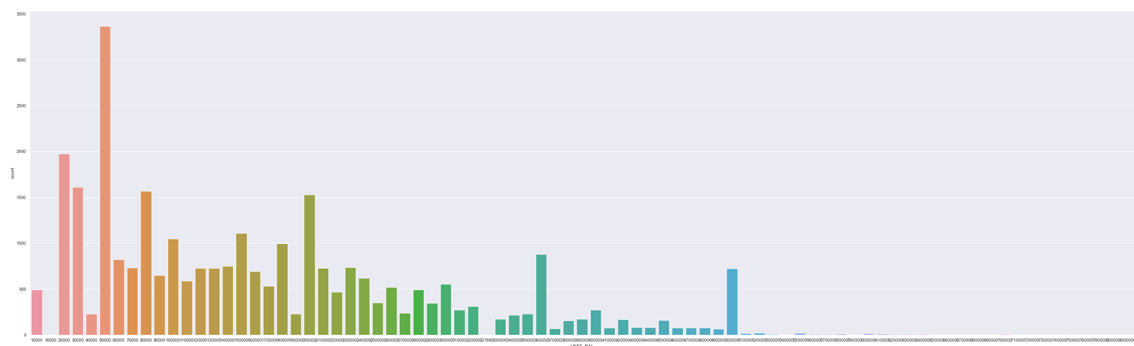
In [91]:

```
sb.set(style="darkgrid")
sb.countplot(data = df, x='PAY_0')
plt.title('Payment Status', fontsize = 15)
plt.show()
```


In [109]:

```
%matplotlib notebook
plt.figure(figsize = (50,15))
sb.countplot(x="LIMIT_BAL", data=df,)
plt.show()
```

<IPython.core.display.Javascript object>



In [106]:

```
print (df["LIMIT_BAL"].value_counts())
print(df["LIMIT_BAL"].value_counts(normalize=True))
```

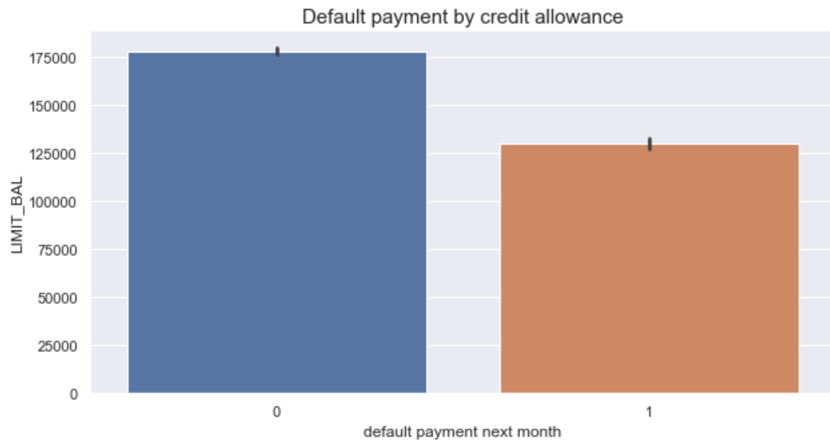
```
50000      3365
20000      1976
30000      1610
80000      1567
200000     1528
           ...
800000        2
1000000       1
760000        1
690000        1
327680        1
Name: LIMIT_BAL, Length: 81, dtype: int64
50000      0.112167
20000      0.065867
30000      0.053667
80000      0.052233
200000     0.050933
             ...
800000     0.000067
1000000    0.000033
760000     0.000033
690000     0.000033
327680     0.000033
Name: LIMIT_BAL, Length: 81, dtype: float64
```

11%(3,365) of the customers have a credit allowance of 50,000 it is the most frequent credit allowance given to customers. 10,000 is the minimum credit allowance and 100,000 being the maximum

In [107]:
```
sorted(df.LIMIT_BAL.unique())
```

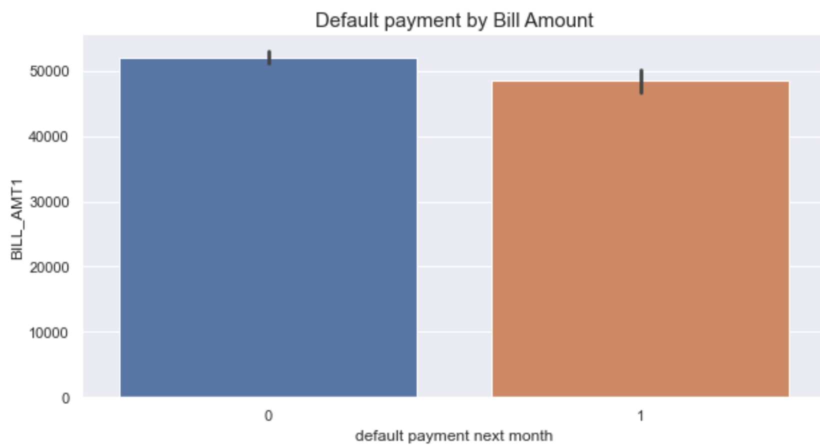
Out[107]:
```
[10000,
 16000,
 20000,
 30000,
 40000,
 50000,
 60000,
 70000,
 80000,
 90000,
 100000,
 110000,
 120000,
 130000,
 140000,
 150000,
 160000,
 170000,
 180000,
 190000,
 200000,
 210000,
 220000,
 230000,
 240000,
 250000,
 260000,
 270000,
 280000,
 290000,
 300000,
 310000,
 320000,
 327680,
 330000,
 340000,
 350000,
 360000,
 370000,
 380000,
 390000,
 400000,
 410000,
 420000,
 430000,
 440000,
 450000,
 460000,
 470000,
 480000,
 490000,
 500000,
 510000,
 520000,
 530000,
 540000,
 550000,
 560000,
 570000,
 580000,
 590000,
 600000,
 610000,
 620000,
 630000,
 640000,
 650000,
 660000,
 670000,
 680000,
 690000,
 700000,
 710000,
 720000,
 730000,
 740000,
 750000,
 760000,
 780000,
 800000,
 1000000]
```

In [80]:
```
plt.figure(figsize = (10,5))
plt.title("Default payment by credit allowance", fontsize =15)
sb.barplot(x="default payment next month", y= 'LIMIT_BAL', data=df)
plt.show()
```



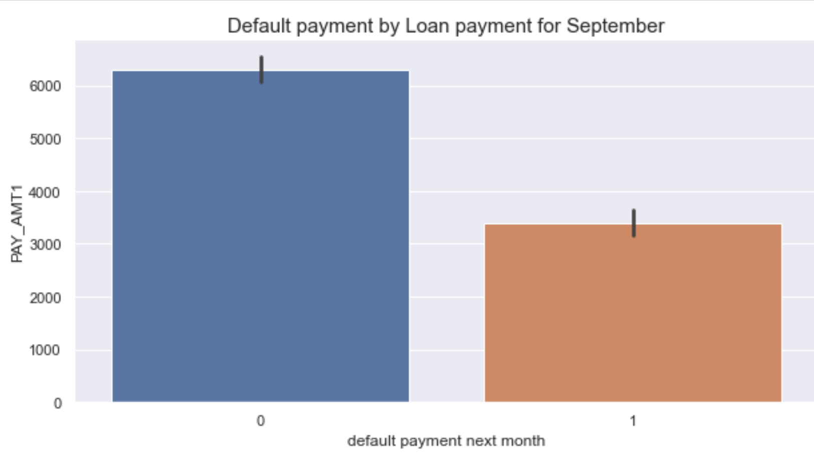

we can also see that Credit Allowance does not have a significant impact on whether a customer will default both customers with lower and and higher credit allowance tend to default.

In [81]:
```
plt.figure(figsize = (10,5))
plt.title("Default payment by Bill Amount", fontsize =15)
sb.barplot(x="default payment next month", y= 'BILL_AMT1', data=df)
plt.show()
```

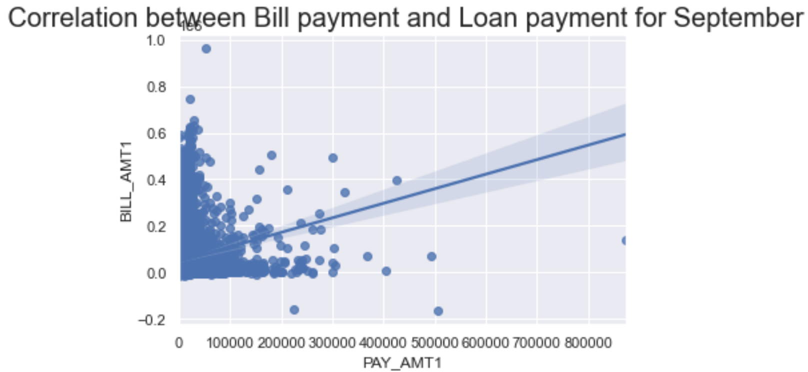



The more bills a customer pays, the likelihood is high that they will default on payments as it can be seen in the month of September.

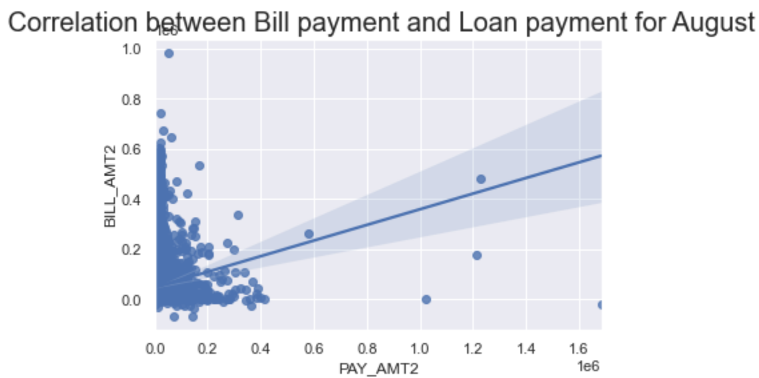
In [83]:
```
plt.figure(figsize = (10,5))
plt.title("Default payment by Loan payment for September", fontsize =15)
sb.barplot(x="default payment next month", y= "PAY_AMT1", data=df)
plt.show()
```



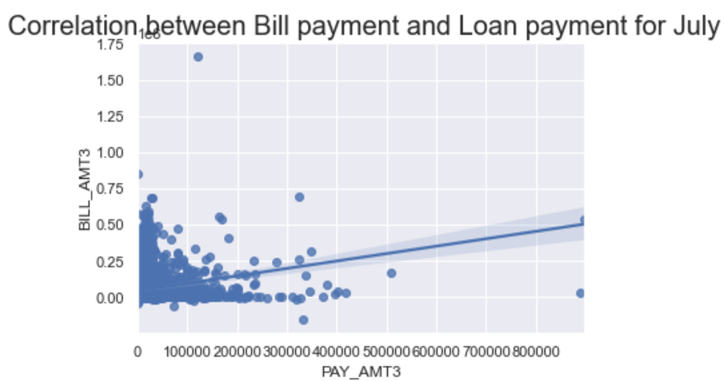
In [55]:
```python
plt.title(" Correlation between Bill payment and Loan payment for September", fontsize = 20)
sb.regplot(x="PAY_AMT1", y="BILL_AMT1", data=df);
```



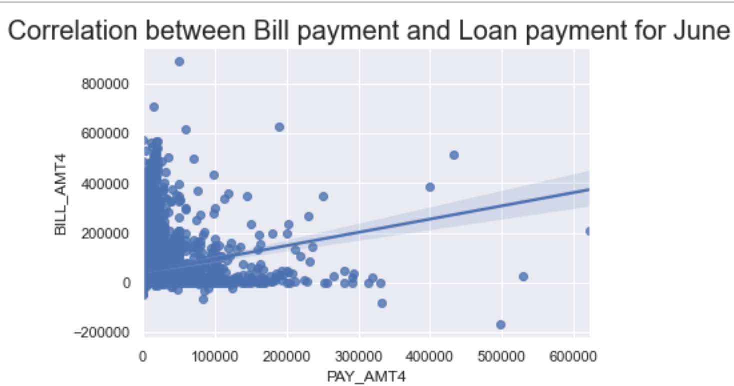

In [56]:
```python
plt.title(" Correlation between Bill payment and Loan payment for August", fontsize = 20)
sb.regplot(x="PAY_AMT2", y="BILL_AMT2", data=df);
```



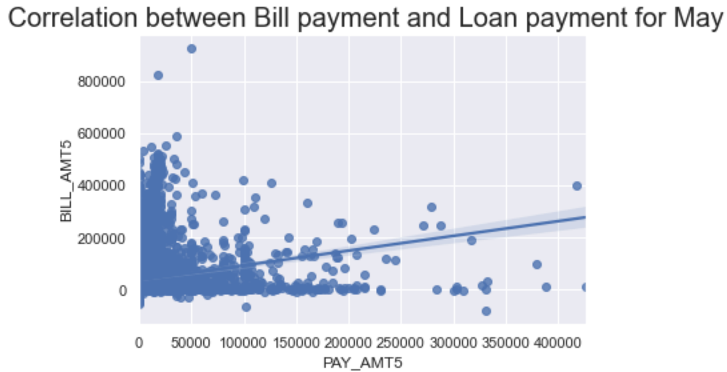

In [57]:
```python
plt.title(" Correlation between Bill payment and Loan payment for July", fontsize = 20)
sb.regplot(x="PAY_AMT3", y="BILL_AMT3", data=df);
```



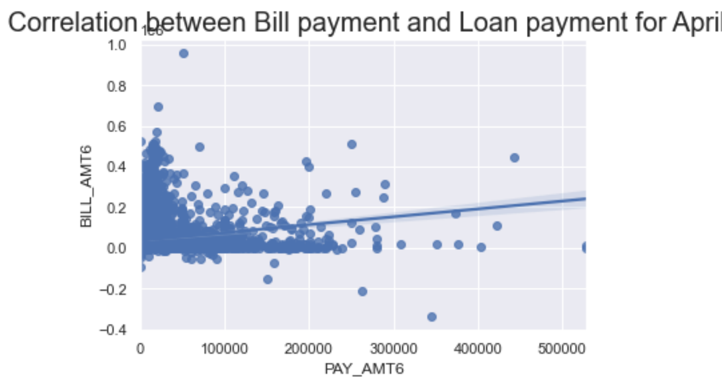

In [59]:
```python
plt.title(" Correlation between Bill payment and Loan payment for June", fontsize = 20)
sb.regplot(x="PAY_AMT4", y="BILL_AMT4", data=df);
```



In [62]:
```python
plt.title(" Correlation between Bill payment and Loan payment for May", fontsize = 20)
sb.regplot(x="PAY_AMT5", y="BILL_AMT5", data=df);
```

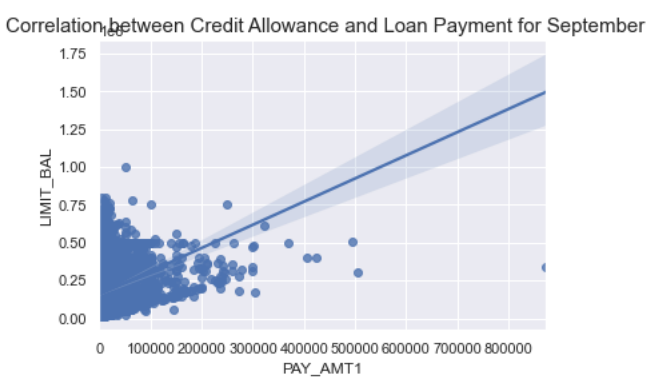



In [61]:
```python
plt.title(" Correlation between Bill payment and Loan payment for April", fontsize = 20)
sb.regplot(x="PAY_AMT6", y="BILL_AMT6", data=df);
```

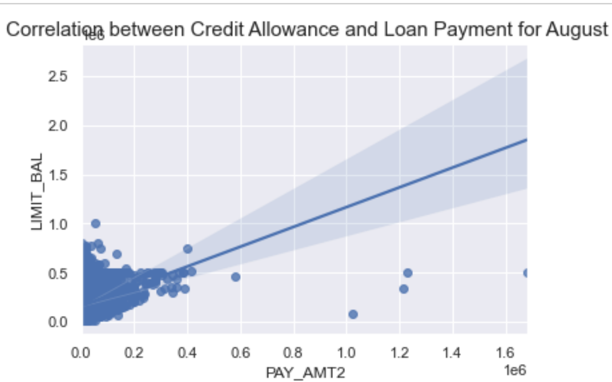



From the above charts we can see that, there is a strong positive correlation between Loan payments and amount spent on bills. even though they were servicing thier loans,they tend to be spedning more on bills. We can also see that the less amount spent on bills, the more they are able to pay for loans and vice versa as can be seen in the Months April and May having the highest payments in terms of loans.

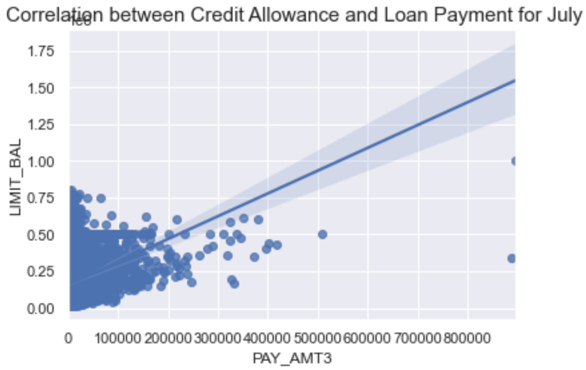
In [71]:
```python
plt.title(" Correlation between Credit Allowance and Loan Payment for September", fontsize =15)
sb.regplot(x="PAY_AMT1", y="LIMIT_BAL", data=df);
```



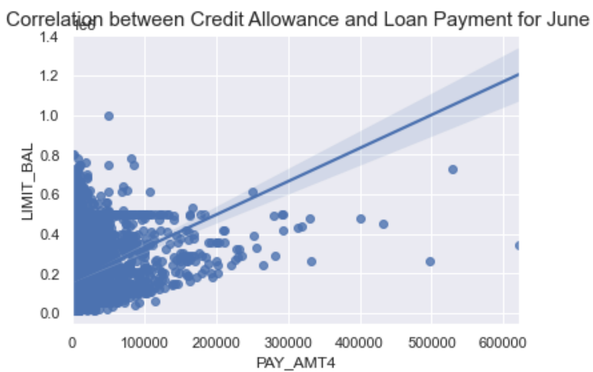

In [78]:
```python
plt.title(" Correlation between Credit Allowance and Loan Payment for August", fontsize =15)
sb.regplot(x="PAY_AMT2", y="LIMIT_BAL", data=df);
```



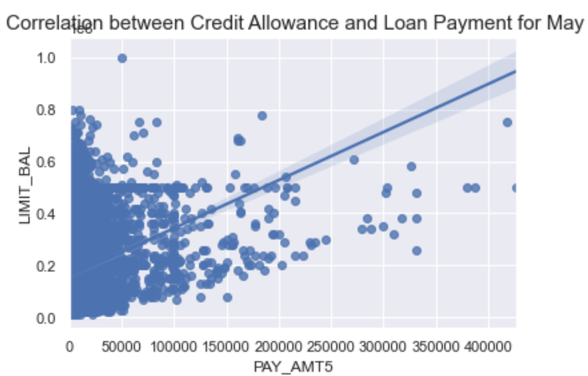
In [77]:
```python
plt.title(" Correlation between Credit Allowance and Loan Payment for July", fontsize =15)
sb.regplot(x="PAY_AMT3", y="LIMIT_BAL", data=df);
```



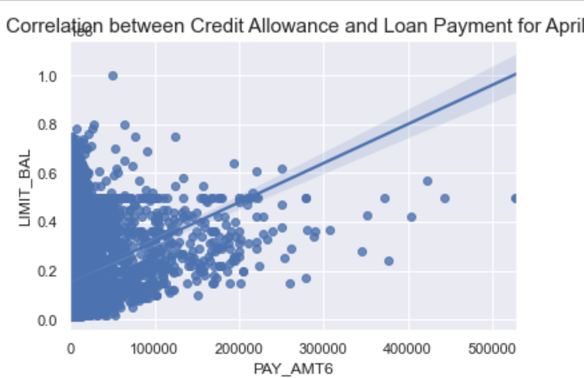

In [76]:
```python
plt.title(" Correlation between Credit Allowance and Loan Payment for June", fontsize =15)
sb.regplot(x="PAY_AMT4", y="LIMIT_BAL", data=df);
```




In [75]:
```python
plt.title(" Correlation between Credit Allowance and Loan Payment for May", fontsize =15)
sb.regplot(x="PAY_AMT5", y="LIMIT_BAL", data=df);
```

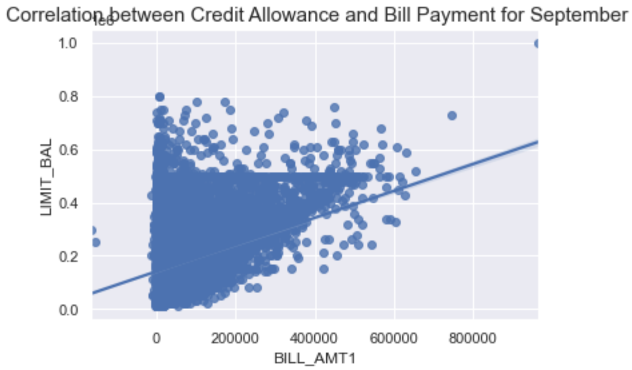



In [79]:
```python
plt.title(" Correlation between Credit Allowance and Loan Payment for April", fontsize =15)
sb.regplot(x="PAY_AMT6", y="LIMIT_BAL", data=df);
```



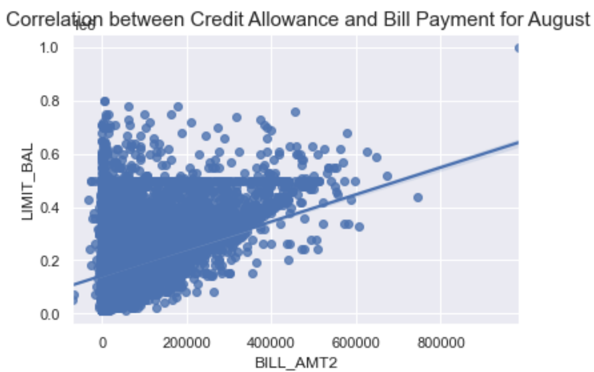

```
There is a strong positive correlation between credit allowance
and Loan payment amount across each month
customers with lower credit allowance tend to pay more
in terms of servicing their credits.
```

In [65]:
```python
plt.title(" Correlation between Credit Allowance and Bill Payment for September", fontsize =15)
sb.regplot(x="BILL_AMT1", y="LIMIT_BAL", data=df);
```

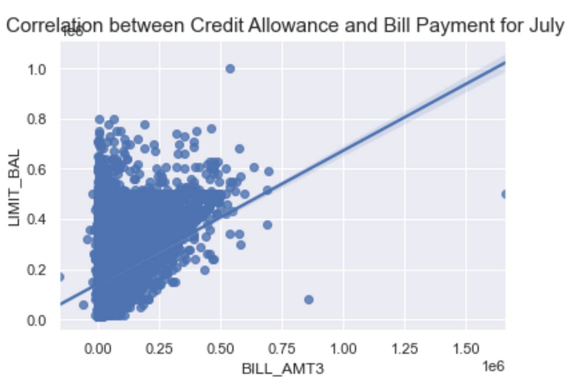



```
There is a strong positive correlation between Credit allowance and Bill amount
this means that the more credit allowance that a customer has, there is
also a high tendency to spend more on payment servicing bills.
```

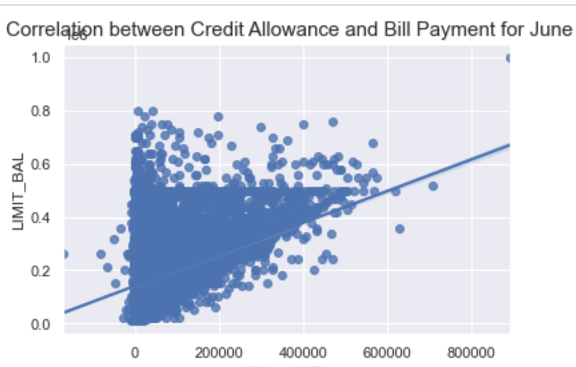
In [66]:
```python
plt.title(" Correlation between Credit Allowance and Bill Payment for August", fontsize =15)
sb.regplot(x="BILL_AMT2", y="LIMIT_BAL", data=df);
```



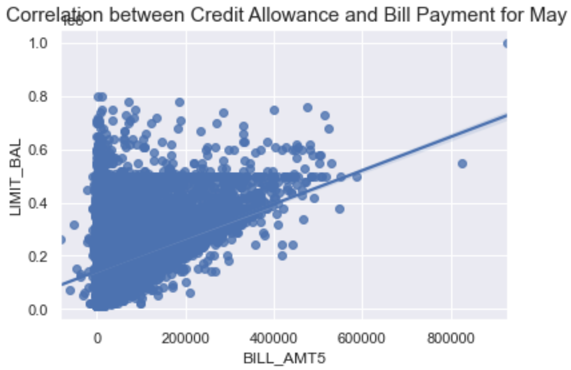

In [67]:
```python
plt.title(" Correlation between Credit Allowance and Bill Payment for July", fontsize =15)
sb.regplot(x="BILL_AMT3", y="LIMIT_BAL", data=df);
```



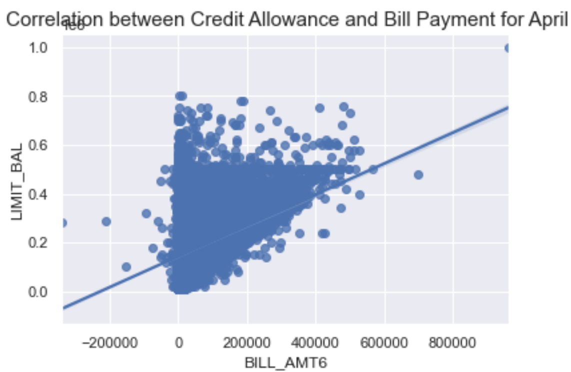

In [68]:
```python
plt.title(" Correlation between Credit Allowance and Bill Payment for June", fontsize =15)
sb.regplot(x="BILL_AMT4", y="LIMIT_BAL", data=df);
```



In [69]:
```
plt.title(" Correlation between Credit Allowance and Bill Payment for May", fontsize =15)
sb.regplot(x="BILL_AMT5", y="LIMIT_BAL", data=df);
```




In [70]:
```
plt.title(" Correlation between Credit Allowance and Bill Payment for April", fontsize =15)
sb.regplot(x="BILL_AMT6", y="LIMIT_BAL", data=df);
```




Across each month, there was a consistent trend of the higher the credit allowance
the more the amount spent servicing bills.

Profile of High-risk customers
From the analysis carried out, this is what the profile of a high-risk customer would look like
1. The females tend to payback loans more often than men.
2. Customers with ages 24,25,27 and 29, falls in the bracket of highest defaulters.
3. The higher the bills a customer pays, the higher the tendency for them to default on payments
4. Lower or higher credit allowance is not a guarantee that a customer wouldn’t default on payments.
5. customers with lower credit allowance tend to pay more
in terms of servicing their credits.
6. Customers with high credit allowance tend to spend more on bills.

In [ ]: