HH Exploratory Data Analysis

Objective

- What are the overall sales trend?
- Revenue generated from each course
- What are top 3 courses by sales?
- What are the number of course orders handled by each counsellor?
- How does the mean revenue collected per month vary with time?

IMPORTING REQUIRED LIBRARIES

In [1]:
```
import random
import pandas as pd
import matplotlib.pyplot as plt
import datetime as dt
import numpy as np
%matplotlib inline

import seaborn as sns
from sklearn.linear_model import LinearRegression
```

IMPORTING THE DATASET

In [2]:
```
ds = pd.read_excel('cleaned_data.xlsx')
df = ds.sample(frac=1)
pd.options.display.float_format = '{:.2f}'.format
```

DATA AUDIT

In [3]:
```
df.head()
```

Out[3]:

	Unnamed: 0	Month	Counsellor Name	Course Name	Fees Total	Fees Received	Fees Pending	Unnamed: 7	Unnamed: 8	Unn
439	444	2020-08-01	NC	CDCW	9320.00	9320.00	0	NaN	NaN	
4412	4842	2020-08-01	Manish	SSBB	14205.96	14584.61	612.04	NaN	NaN	
3097	3124	2020-07-01	Deepak	GST	3569.72	3569.72	0	NaN	NaN	
1082	1091	2020-12-01	SS	CDCW	10000.00	10000.00	0	NaN	NaN	
745	752	2020-10-01	SK	CDCW	2000.00	2000.00	0	NaN	NaN	

In [4]:
```
df.tail()
```

Out[4]:

	Unnamed: 0	Month	Counsellor Name	Course Name	Fees Total	Fees Received	Fees Pending	Unnamed: 7	Unnamed: 8	Unn
5003	5443	2020-10-01	Manish	SSGB	11000.00	11000.00	0	NaN	NaN	
3837	4256	2020-12-01	Manish	SSBB	18000.00	18000.00	0	NaN	NaN	
2520	2547	2020-04-01	Deepak	CDCW	8550.00	8550.00	0	NaN	NaN	
1745	1769	2021-04-01	MM	CDCW	10000.00	10000.00	0	NaN	NaN	
1761	1785	2021-04-01	Manisha	CDCW	20000.00	20000.00	0	NaN	NaN	

In [5]:
```
df.shape
```

Out[5]:
```
(6360, 10)
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['Unnamed: 0', 'Month', 'Counsellor Name', 'Course Name', 'Fees Total',
       'Fees Received', 'Fees Pending', 'Unnamed: 7', 'Unnamed: 8',
       'Unnamed: 9'],
      dtype='object')
```

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6360 entries, 439 to 1761
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype         
---  ------           --------------  -----         
 0   Unnamed: 0       6360 non-null   int64         
 1   Month            6360 non-null   datetime64[ns]
 2   Counsellor Name  6360 non-null   object        
 3   Course Name      6360 non-null   object        
 4   Fees Total       6360 non-null   float64       
 5   Fees Received    6360 non-null   float64       
 6   Fees Pending     6360 non-null   object        
 7   Unnamed: 7       0 non-null      float64       
 8   Unnamed: 8       0 non-null      float64       
 9   Unnamed: 9       0 non-null      float64       
dtypes: datetime64[ns](1), float64(5), int64(1), object(3)
memory usage: 546.6+ KB
```

In [8]:
```
df.drop('Unnamed: 7', axis=1,inplace=True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6360 entries, 439 to 1761
Data columns (total 9 columns):
```

```
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Unnamed: 0       6360 non-null   int64
 1   Month            6360 non-null   datetime64[ns]
 2   Counsellor Name  6360 non-null   object
 3   Course Name      6360 non-null   object
 4   Fees Total       6360 non-null   float64
 5   Fees Received    6360 non-null   float64
 6   Fees Pending     6360 non-null   object
 7   Unnamed: 8       0 non-null      float64
 8   Unnamed: 9       0 non-null      float64
dtypes: datetime64[ns](1), float64(4), int64(1), object(3)
memory usage: 496.9+ KB
```

In [9]:
```
df.drop(['Unnamed: 8','Unnamed: 9'], axis=1,inplace=True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6360 entries, 439 to 1761
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Unnamed: 0       6360 non-null   int64
 1   Month            6360 non-null   datetime64[ns]
 2   Counsellor Name  6360 non-null   object
 3   Course Name      6360 non-null   object
 4   Fees Total       6360 non-null   float64
 5   Fees Received    6360 non-null   float64
 6   Fees Pending     6360 non-null   object
dtypes: datetime64[ns](1), float64(2), int64(1), object(3)
memory usage: 397.5+ KB
```

In [10]:
```
# Checking missing values
df.isnull().sum()
```

Out[10]:
```
Unnamed: 0         0
Month              0
Counsellor Name    0
Course Name        0
Fees Total         0
Fees Received      0
Fees Pending       0
dtype: int64
```

In [11]:
```
# Getting descriptive statistics summary
df.describe()
```

Out[11]:

	Unnamed: 0	Fees Total	Fees Received
count	6360.00	6360.00	6360.00
mean	3395.56	12746.30	1559008.97
std	2025.87	8296.17	123308464.27
min	1.00	0.00	0.00
25%	1612.75	7700.04	7600.00
50%	3206.50	11750.00	11875.00
75%	5203.25	15000.00	15000.00

	Unnamed: 0	Fees Total	Fees Received
max	6839.00	89000.00	9833814498.00

EXPLORATORY DATA ANALYSIS

- ## WHAT IS THE OVERALL SALES TREND?

In [12]:
```
df['Month'].min()
```

Out[12]:
```
Timestamp('2020-04-01 00:00:00')
```

In [13]:
```
df['Month'].max()
```

Out[13]:
```
Timestamp('2021-07-01 00:00:00')
```

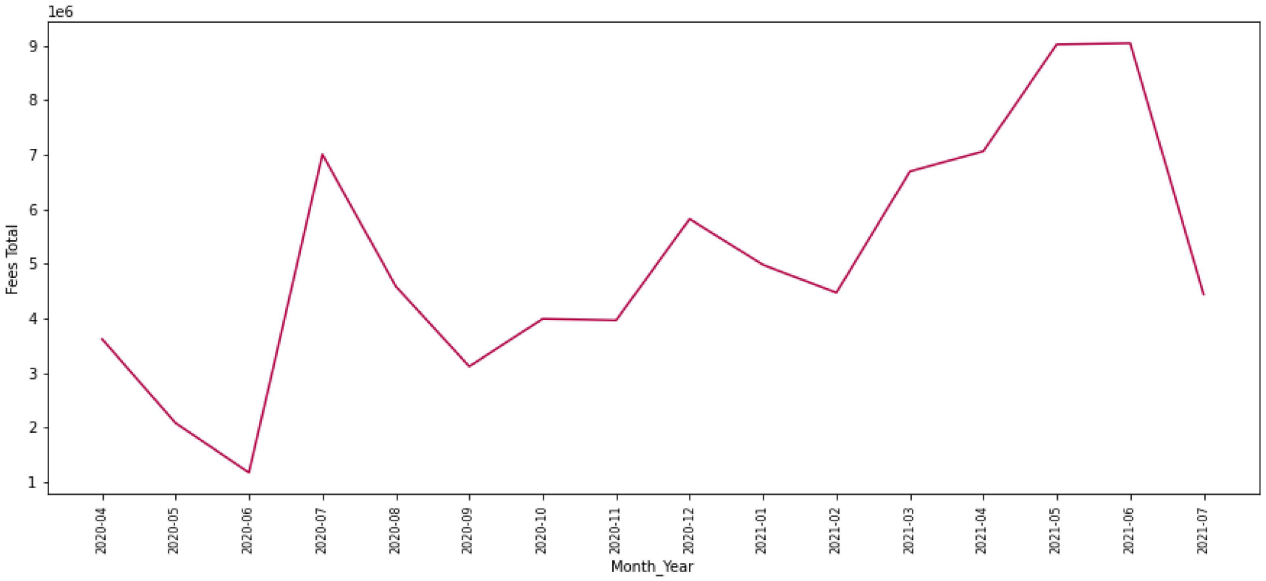
In [14]:
```
df['month_year'] = df['Month'].apply(lambda x: x.strftime('%Y-%m'))
```

In [15]:
```
# Grouping month year
df_trend = df.groupby('month_year').sum()['Fees Total'].reset_index()
df_trend
```

Out[15]:

	month_year	Fees Total
0	2020-04	3618720.00
1	2020-05	2080410.00
2	2020-06	1170850.00
3	2020-07	7004012.89
4	2020-08	4585163.11
5	2020-09	3118683.92
6	2020-10	3991449.00
7	2020-11	3963611.00
8	2020-12	5820778.73
9	2021-01	4979998.00
10	2021-02	4470147.00
11	2021-03	6692073.29
12	2021-04	7061907.52
13	2021-05	9021952.00
14	2021-06	9045863.00
15	2021-07	4440819.00

In [16]:
```
# Setting the figure size
plt.figure(figsize=(15,6))
plt.plot(df_trend['month_year'], df_trend['Fees Total'], color='#b80045')
plt.xlabel('Month_Year', fontsize=10)
plt.ylabel('Fees Total', fontsize=10)
plt.xticks(rotation='vertical', size=8)
plt.show()
```



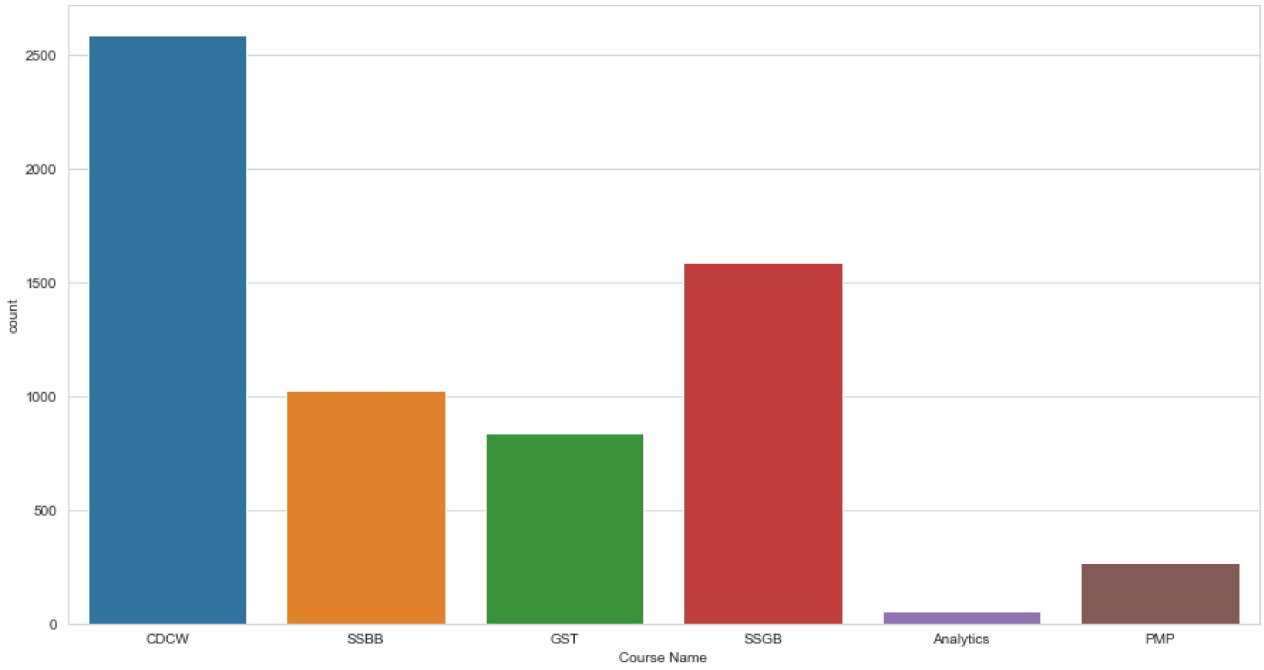
- WHAT ARE THE SALES OF EACH COURSE?

In [17]:
```
#Sales of each course
course_count = pd.DataFrame(df.groupby('Course Name').count()['Fees Total'])
course_count
```

Out[17]:

Course Name	Fees Total
Analytics	53
CDCW	2590
GST	834
PMP	268
SSBB	1026
SSGB	1589

In [18]:
```
plt.figure(figsize=(15,8))
sns.set_style("whitegrid")
sns.countplot(x=df['Course Name'])
plt.show()
```



- WHAT ARE THE TOP 3 COURSES BY SALES?

In [19]:
```
course_sales = pd.DataFrame(df.groupby('Course Name').count()['Fees Total'].rename('Cou
```

In [20]:
```
course_sales = course_sales.sort_values('Count', ascending=False)
```

In [21]:
```
#Top 3 courses by sales
course_sales[:3]
```

Out[21]:

Course Name	Count
CDCW	2590
SSGB	1589
SSBB	1026

- WHAT IS THE REVENUE GENERATED FROM EACH COURSE?

In [22]:
```
course_rev = pd.DataFrame(df.groupby('Course Name').sum()['Fees Total'].rename('Revenue
course_rev
```

Out[22]:

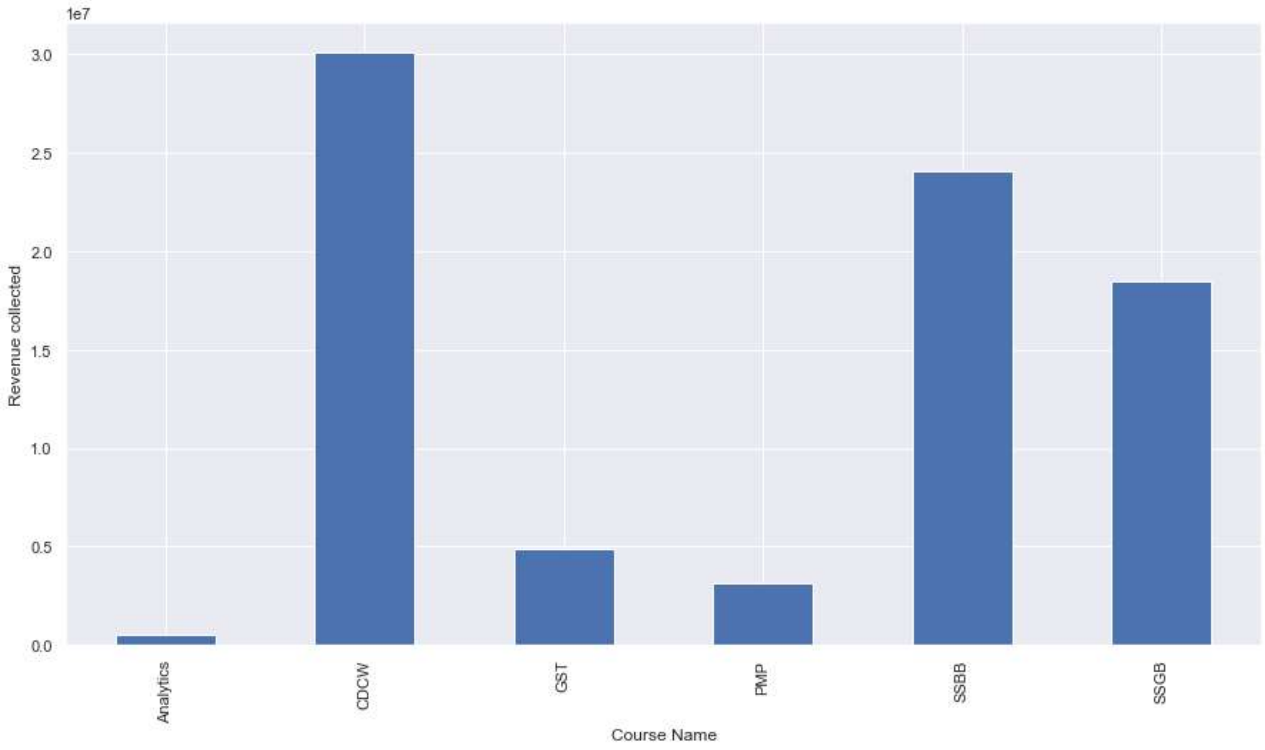
Course Name	Revenue Generated
Analytics	510547.65
CDCW	30087032.40

Course Name	Revenue Generated
GST	4854866.89
PMP	3104577.00
SSBB	24054030.52
SSGB	18455384.00

- Bar Graph

In [23]:
```
plt.figure(figsize=(15,8))
sns.set_theme(style="darkgrid")
df.groupby('Course Name').sum()['Fees Total'].rename('Revenue Generated').plot(kind="ba
```

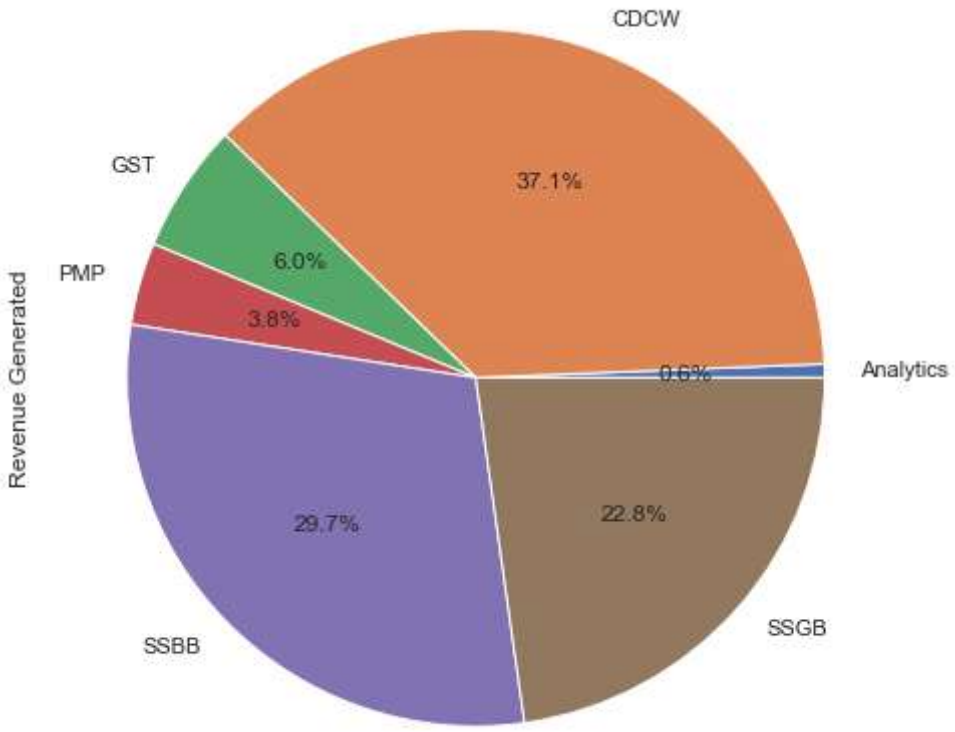
Out[23]: <AxesSubplot:xlabel='Course Name', ylabel='Revenue collected'>



- Pie Chart

In [24]:
```
plt.figure(figsize=(15,8))
sns.set_style("whitegrid")
df.groupby('Course Name').sum()['Fees Total'].rename('Revenue Generated').plot(kind="pi
```

Out[24]: <AxesSubplot:ylabel='Revenue Generated'>



- WHAT ARE THE NUMBER OF COURSE ORDERS HANDLED BY EACH COUNSELLOR?

In [25]:
```
df.groupby('Counsellor Name').count()['Course Name']
```

Out[25]:
```
Counsellor Name
Abhishek       76
Abhishek C    127
Akansha       139
Ankit         247
Ayushi          3
Bhoomika        9
CD            222
Chandrika      15
Deepak        366
Deepender      17
Deepti         17
Garima         54
Hemlata         9
Honey           7
IJ            329
Iqra           13
KIM             2
Khalid         70
Kim           201
Komal          13
Laiba         206
MANISH          2
MM            338
Manish        659
Manisha       124
```

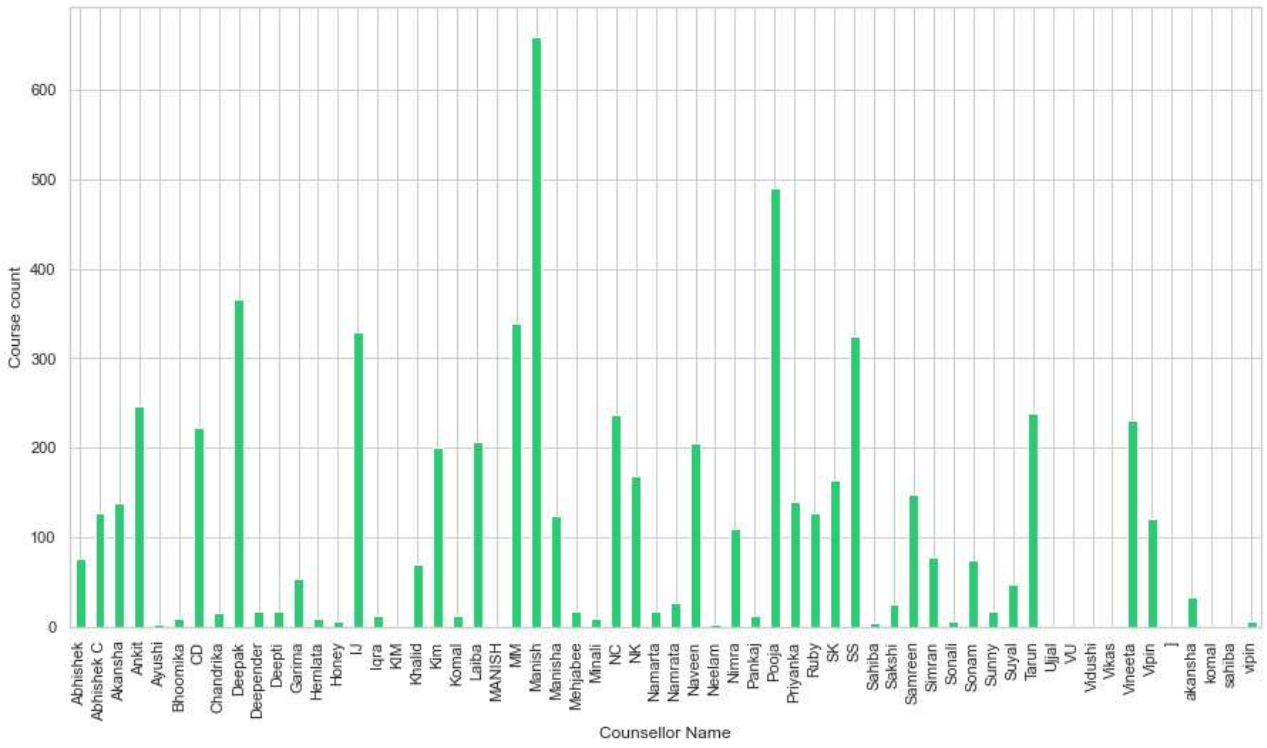
```
Mehjabee       18
Minali          9
NC            237
NK            169
Namarta        17
Namrata        27
Naveen        205
Neelam          3
Nimra         110
Pankaj         13
Pooja         490
Priyanka      140
Ruby          127
SK            164
SS            324
Sahiba          5
Sakshi         26
Samreen       148
Simran         78
Sonali          7
Sonam          74
Sunny          17
Suyal          48
Tarun         238
Ujjal           2
VU              2
Vidushi         1
Vikas           1
Vineeta       230
Vipin         121
]               1
akansha        33
komal           2
sahiba          2
vipin           6
Name: Course Name, dtype: int64
```

In [26]:

```
plt.figure(figsize=(15,8))
df.groupby('Counsellor Name').count()['Course Name'].plot(kind="bar", ylabel="Course co
```

Out[26]:

```
<AxesSubplot:xlabel='Counsellor Name', ylabel='Course count'>
```



- HOW DOES THE MEAN REVENUE COLLECTED PER MONTH VARY WITH TIME?

- Line Plot

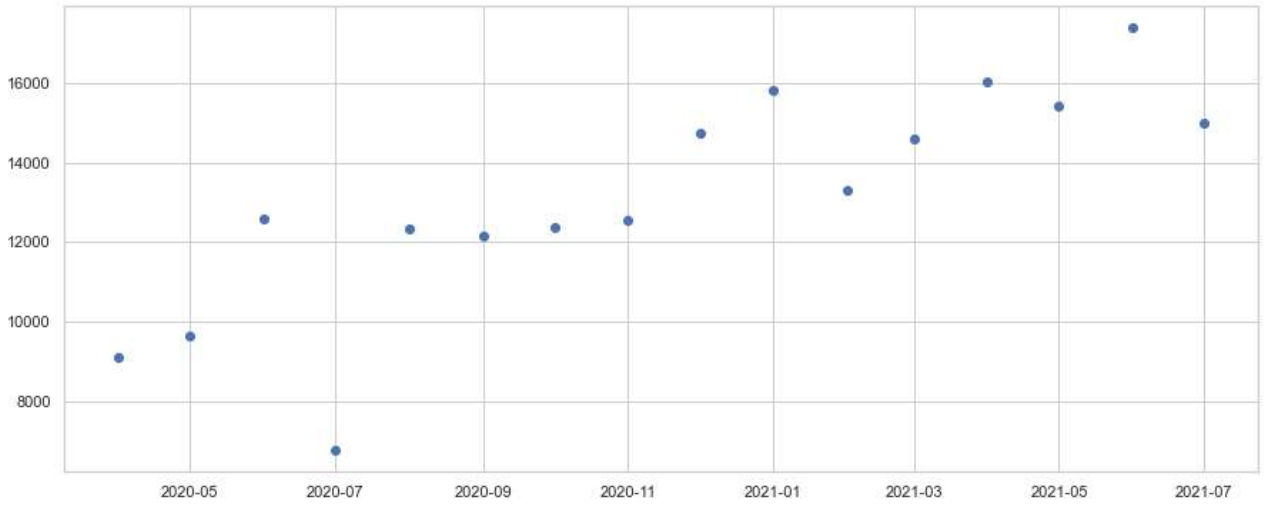
In [27]:
```
course_ran = df.groupby('Month', as_index=True).mean()['Fees Total']
print(course_ran)
date = df['Month'].map(dt.datetime.toordinal)
plt.figure(figsize=(15,6))
plt.plot(course_ran, color="red")
plt.xticks(rotation='vertical', size=15)
plt.show()
```

```
Month
2020-04-01    9092.26
2020-05-01    9631.53
2020-06-01   12589.78
2020-07-01    6747.60
2020-08-01   12325.71
2020-09-01   12134.96
2020-10-01   12357.43
2020-11-01   12543.07
2020-12-01   14736.15
2021-01-01   15809.52
2021-02-01   13304.01
2021-03-01   14579.68
2021-04-01   16013.40
2021-05-01   15422.14
2021-06-01   17395.89
2021-07-01   15002.77
Name: Fees Total, dtype: float64
```

- **Scatter Plot**

In [28]:
```
plt.figure(figsize=(15,6))
a = pd.DataFrame(df.groupby('Month', as_index=True).mean()['Fees Total']).values
b = df['Month'].unique()
c = np.sort(b)
plt.scatter(c,a)
```

Out[28]: <matplotlib.collections.PathCollection at 0x2005baac9a0>



- **Linear Regression Model**

In [29]:
```
p = df['Month'].map(dt.datetime.toordinal).unique()
q= np.sort(p)
q = q.reshape(-1,1)
q.shape
```

Out[29]: (16, 1)

In [30]:
```
model = LinearRegression()
```

```
model.fit(q,a)
```

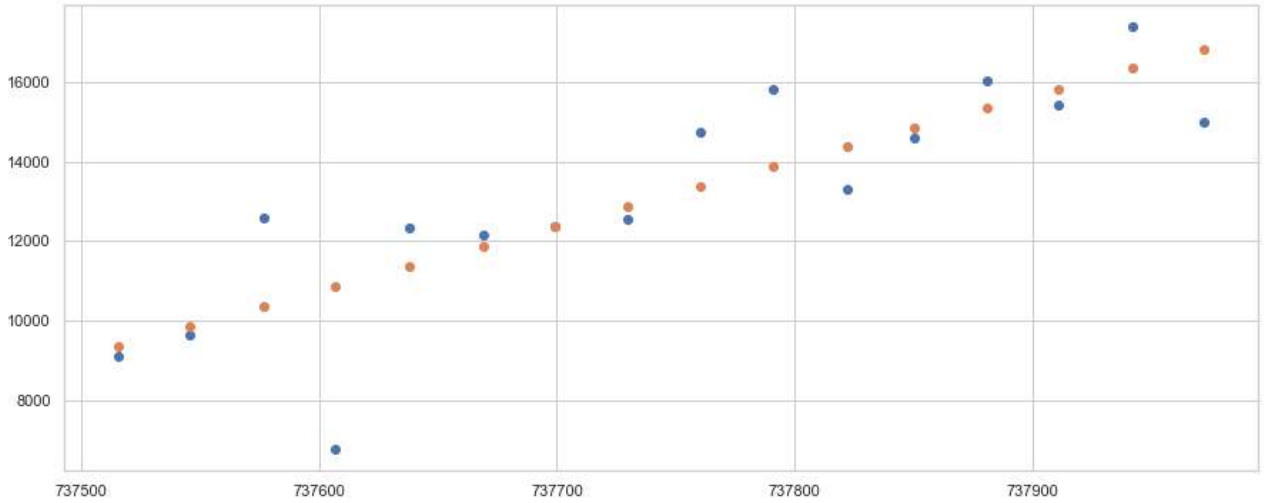
Out[30]:
```
LinearRegression()
```

In [31]:
```
# Getting predicted monthly mean revenues in the form of an array
a_pred = model.predict(q)
a_pred
```

Out[31]:
```
array([[ 9365.85492169],
       [ 9856.95378934],
       [10364.42261925],
       [10855.5214869 ],
       [11362.99031681],
       [11870.45914672],
       [12361.55801437],
       [12869.02684428],
       [13360.12571193],
       [13867.59454184],
       [14375.06337175],
       [14833.42231489],
       [15340.8911448 ],
       [15831.99001245],
       [16339.45884236],
       [16830.55771001]])
```

In [32]:
```
plt.figure(figsize=(15,6))
plt.scatter(q,a)
plt.scatter(q,a_pred)
```

Out[32]:
```
<matplotlib.collections.PathCollection at 0x2005bb11d60>
```



In [33]:
```
model.score(q,a)
```

Out[33]:
```
0.7065517159854393
```

The linear regression model has an accuracy of approximately 71%

In [34]:
```
# Viewing the ".map(dt.datetime.toordinal)" equivalent of "Month" column of dataset
q
```

Out[34]:
```
array([[737516],
       [737546],
       [737577],
       [737607],
       [737638],
       [737669],
       [737699],
       [737730],
       [737760],
       [737791],
       [737822],
       [737850],
       [737881],
       [737911],
       [737942],
       [737972]], dtype=int64)
```

In [35]:
```
# Predicting mean revenues for upcoming months
x_pred = np.array([737975,738000]).reshape(-1,1)
y_pred = model.predict(x_pred)
y_pred
```

Out[35]:
```
array([[16879.66759678],
       [17288.91665315]])
```