FIT1045 INTRODUCTION TO DATA SCIENCE

ASSIGNMENT 1

George Tan Juan Sheng
30884128

Introduction

This assignment is mainly on finding infos from the data of a respective cinema. We will be analysing the data and answering the questions 1 by 1 starting from Question 1. Along the way, we will be explaining on how we are able to extract the info we want from each data as well, at the same time few other data frames would also be created. We will also be using in built functions like max, min to get our findings in order to answer each questions. Chart visualizations like bar graph will also be used in order to reassure our answer. We would also be providing business insights after each question is answered in order to improve the cinema's business.

Importing the necessary libraries

In [1]:
```
import pandas as pd
import matplotlib.pylab as plt
%matplotlib inline
```

Size and Statistic of Files

File FIT1043-ticket-trx.csv

The file 'FIT1043-ticket-trx.csv' is named as df1 and the size of file 'FIT1043-ticket-trx.csv' is 88477 x 35. In other words this file consists of 88477 rows and 35 columns.

In [2]:
```
df1 = pd.read_csv('FIT1043-ticket-trx.csv')
df1.shape
```

Out[2]:
```
(88477, 35)
```

These are the basic Statistics of the Values in File 'FIT1043-ticket-trx.csv'.

In [3]:
```
df1.describe()
```

Out[3]:

	Transaction.Number	Transaction.Sequence.Number	Ticket.Type.Code	Admits	Gro
count	88477.000000	88477.000000	88477.000000	88477.000000	
mean	726713.927156	5.204211	31.250246	0.988359	
std	20475.126596	23.963779	9.905059	0.152143	
min	689235.000000	1.000000	1.000000	-1.000000	
25%	708132.000000	1.000000	36.000000	1.000000	
50%	726091.000000	2.000000	36.000000	1.000000	
75%	744559.000000	2.000000	36.000000	1.000000	
max	761953.000000	392.000000	47.000000	1.000000	

8 rows × 21 columns

File FIT1043-ticket-seating.csv

The file 'FIT1043-ticket-seating.csv' is named as df2 and the size of file 'FIT1043-ticket-seating.csv' is 88477 x 16. In other words this file consists of 88477 rows and 16 columns.

In [4]:
```
df2 = pd.read_csv('FIT1043-ticket-seating.csv')
df2.shape
```

Out[4]: (88477, 16)

These are the basic Statistics of the Values in File 'FIT1043-ticket-seating.csv'.

In [5]:
```
df2.describe()
```

Out[5]:

	Transaction.Number	Transaction.Sequence.Number	Session.Id	Seat.Number	Grid.
count	88477.000000	88477.000000	88477.000000	88327.000000	88477.0
mean	726713.927156	5.204211	12278.738395	10.706262	11.6
std	20475.126596	23.963779	797.306817	5.921585	6.2
min	689235.000000	1.000000	10893.000000	1.000000	0.0
25%	708132.000000	1.000000	11646.000000	6.000000	7.0
50%	726091.000000	2.000000	12193.000000	10.000000	11.0
75%	744559.000000	2.000000	12797.000000	15.000000	15.0
max	761953.000000	392.000000	13879.000000	28.000000	30.0

Reading Files and merging files

File FIT1043-ticket-trx.csv and FIT1043-ticket-seating.csv were merged together on Transaction.Number and Transaction.Sequence into a dataframe named 'df3'. The purpose of merging these two files on the Transaction.Number and Transaction.Sequence is to form a unique identifier with both of these columns. There is also no duplicated fields after merging.

In [6]:
```
df3 = pd.merge(df1,df2, on = ['Transaction.Number','Transaction.Sequence.Numb
df3
```

Out[6]:

	Transaction.Number	Transaction.Sequence.Number	Transaction.Date.Time	Type.Of.Trans
0	689235	5	2/4/2017 0:34	Refund F
1	689235	6	2/4/2017 0:34	Refund F
2	691991	5	2/4/2017 0:33	Refund F
3	691991	6	2/4/2017 0:33	Refund F
4	692271	1	1/4/2017 6:01	Ticket Ref
...	...	...	...	
88472	761950	2	1/5/2017 0:21	Ticke
88473	761953	1	1/5/2017 0:25	Ticke
88474	761953	2	1/5/2017 0:25	Ticke
88475	761953	3	1/5/2017 0:25	Ticke
88476	761953	4	1/5/2017 0:25	Ticke

88477 rows × 49 columns

The size of the merged dataframe is now 88477 x 49. In other words this file consists of 88477 rows and 49 columns.

In [7]:
```
df3.shape
```

Out[7]: (88477, 49)

These are now the statistics of the newly merged dataframe df3.

In [8]:
```
df3.describe()
```

Out[8]:

	Transaction.Number	Transaction.Sequence.Number	Ticket.Type.Code	Admits	Gro
count	88477.000000	88477.000000	88477.000000	88477.000000	
mean	726713.927156	5.204211	31.250246	0.988359	
std	20475.126596	23.963779	9.905059	0.152143	
min	689235.000000	1.000000	1.000000	-1.000000	
25%	708132.000000	1.000000	36.000000	1.000000	
50%	726091.000000	2.000000	36.000000	1.000000	
75%	744559.000000	2.000000	36.000000	1.000000	
max	761953.000000	392.000000	47.000000	1.000000	

8 rows × 25 columns

QUESTION 1

Now, we will be finding the total revenue of each film, which can be found by finding the sum of Gross.Box.Office as this column shows the price of each ticket was sold including the two taxes.

This fullfils the term revenue as we are not looking for profits, hence we would not be using Net.BoxOffice in this case as Net.BoxOffice shows the net profit from tickets.

In [9]:

```
HR = df3.groupby('Film').agg({'Gross.Box.Office':'sum'})
HR= HR.reset_index()
HR.rename(columns = {'Gross.Box.Office':'TotalRevenue'}, inplace = True)
HR #HIGHEST REVENUE
```

Out[9]:

	Film	TotalRevenue
0	Film 1	3375.5
1	Film 10	31590.0
2	Film 11	4872.0
3	Film 13	4446.0
4	Film 14	12488.5
5	Film 16	4376.0
6	Film 17	3969.5
7	Film 20	11321.5
8	Film 21	1682.0
9	Film 24	2219.0
10	Film 25	58725.0
11	Film 27	2638.0
12	Film 3	48343.0
13	Film 30	7410.0
14	Film 31	37826.0
15	Film 32	80.0
16	Film 33	36.0
17	Film 34	15309.0
18	Film 35	18225.0
19	Film 36	3780.0
20	Film 37	35775.5
21	Film 38	2675.0
22	Film 39	9260.5
23	Film 4	28629.5
24	Film 40	1404.0
25	Film 41	2540.0
26	Film 5	321248.5
27	Film 6	68564.0
28	Film 7	7786.0
29	Film 8	71.5

From here, we can find which film has the highest revenue.
We can do this by finding which film's revenue is equal to the maximum of the total revenues of all films.
In the end, we were able to discover that Film 5 has the highest revenue of 321248.5$ when compared to other films.

In [10]:
```
HR.loc[HR['TotalRevenue']==
max(HR['TotalRevenue'])]
```

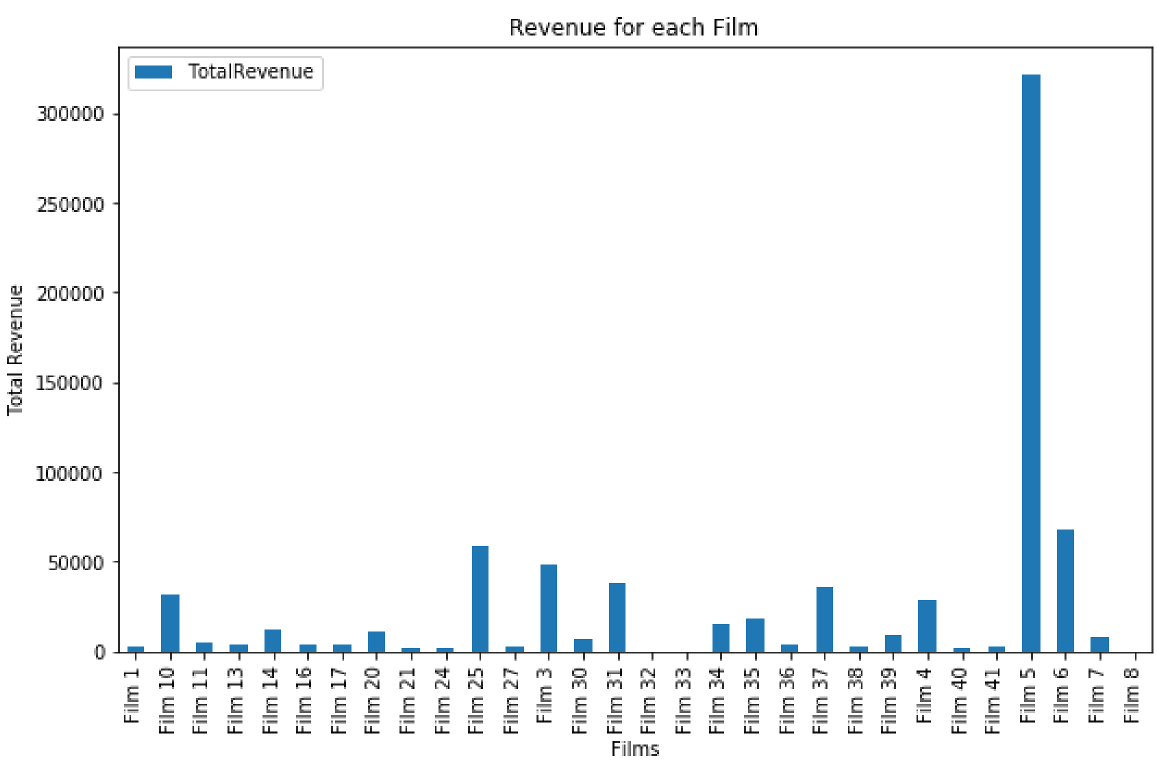
Out[10]:

	Film	TotalRevenue
26	Film 5	321248.5

We can also plot a bar graph in order to reassure our findings.
From the bar graph, we were also able to see that Film 5's Revenue were the highest and over 300000$.
Hence, we are able to conclude that Film 5 is the film that has the highest revenue.

In [11]:
```
bar = HR.plot.bar(figsize= (10,6))
bar.set_xticklabels(HR['Film'])
plt.xlabel('Films')
plt.ylabel('Total Revenue')
plt.title('Revenue for each Film')
plt.show
```

Out[11]: <function matplotlib.pyplot.show(*args, **kw)>



As shown as above, we were able to find out that Film 5 is the best selling film as it has generated a very high amount of revenue to the respective cinema when compared to other films.

QUESTION 2

Next, we will be looking for the least popular day to watch a movie.

We do this by first setting the column 'Session.Screening.Time' to datetime.

The reason why we chose to analyse the column 'Session.Screening.Time' and not 'Transaction.Date.Time' is because 'Transaction.Date.Time' is just the time where a ticket was sold, where 'Session.Screening.Time' shows the date and time of when the respective movie will be screening, and this will be able to show us how many people actually chose to purchase tickets to watch movies on that respective day.

We have also formatted the date and time in order to prevent mistake of values, for examples, the day will be mistaken as month and vice versa.

In [12]:
```
df3['Session.Screening.Time']= pd.to_datetime(df3['Session.Screening.Time'],f
```

Then, we created a new column called 'Screening.Day' and we find the name of the day according to the respective date.
We then group the 'Screening.Days' together and the number of audience would be number of times a respective day, among all the audiences had chose to purchase his ticket on.

In [13]:
```
df3['Screening.Day']= df3['Session.Screening.Time'].dt.strftime('%A')
LPD = df3.groupby('Screening.Day').agg({'Screening.Day':'count'})
LPD = LPD.rename(columns={'Screening.Day':'NumberOfAudience'}).reset_index()
LPD #LEAST POPULAR DAY
```

Out[13]:

	Screening.Day	NumberOfAudience
0	Friday	15356
1	Monday	8432
2	Saturday	23112
3	Sunday	19147
4	Thursday	9132
5	Tuesday	5763
6	Wednesday	7535

After we have the dataframe, we are able to find out which day has the least number of audience by finding which 'Screening.Day' has the minimum number of audience.

Hence, we were able to find that Tuesday had the lowest number of audience.

In [14]:
```
LPD.loc[LPD['NumberOfAudience']==min(LPD['NumberOfAudience'])]
```

Out[14]:

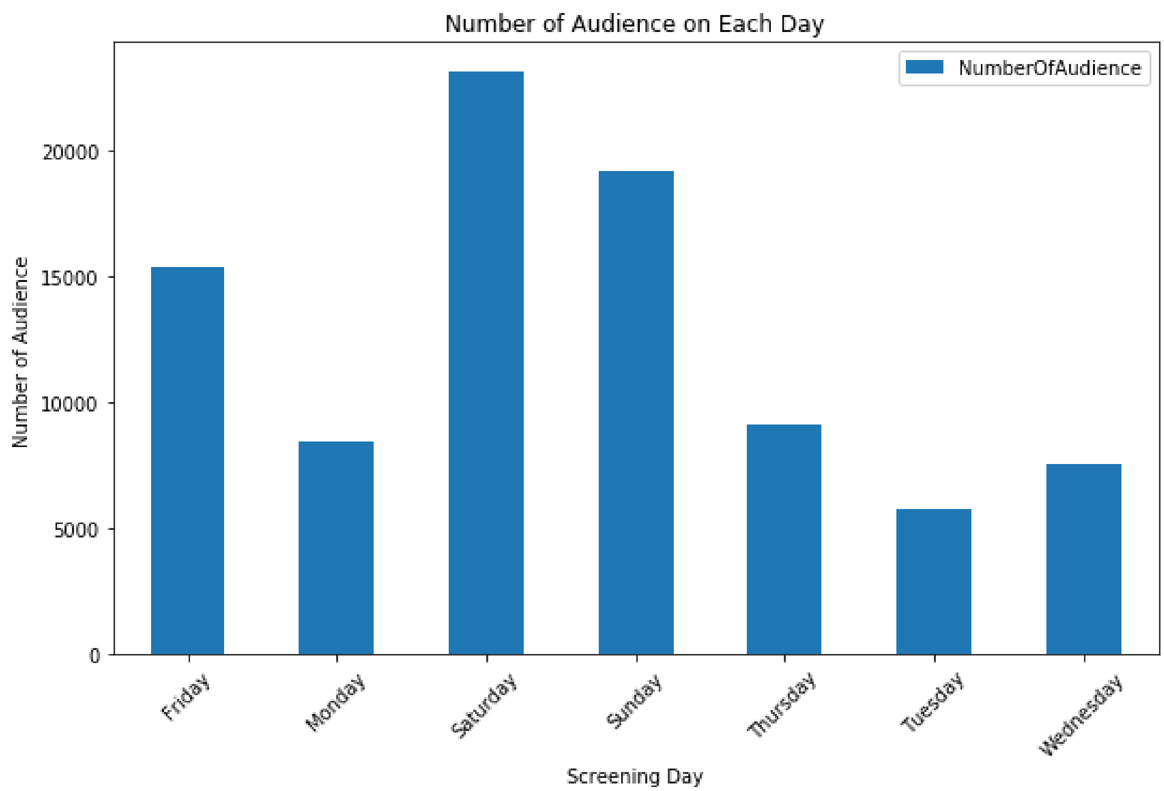
	Screening.Day	NumberOfAudience
5	Tuesday	5763

We were also able to find which day that is least popular to watch a movie by plotting a bar graph.

We can plot a bar graph and observe which day has the least number of audience.

In [15]:
```
bar2 = LPD.plot.bar(figsize=(10,6))
bar2.set_xticklabels(LPD['Screening.Day'],rotation = 45)
plt.xlabel('Screening Day')
plt.ylabel('Number of Audience')
plt.title('Number of Audience on Each Day')
```

Out[15]:
```
Text(0.5, 1.0, 'Number of Audience on Each Day')
```



After plotting a bar graph, we can clearly see that Tuesday has the lowest number of audience.

Hence, we can conclude that Tuesday is the least popular to watch a movie.

QUESTION 3

Now, we are looking to find the time of the day where it is the most popular to watch a movie. We will be getting the hours of each film's screening time and according to the hours , we will be able to determine which times of the day will be the most popular among people.

In [16]:
```
df3['Screening.Hour']=df3['Session.Screening.Time'].dt.strftime('%H')
```

In our analysis, we categorized time betwwen 5 AM and before 12 PM as Morning, 12 PM as noon, between 1 PM and before 5 PM as Afternoon, between 5 PM and before 7 PM as Early Evening, between 7 PM and before 8 PM as Late Evening, between 8 PM and before 12 AM as Night, and lastly, betwwen 12 AM and before 5 AM as Midnight.

In [17]:
```
df3.loc[(df3['Screening.Hour']>= '05') & (df3['Screening.Hour'] < '12'), 'Tim
df3.loc[df3['Screening.Hour'] == '12', 'Time.Group'] = 'Noon'
df3.loc[(df3['Screening.Hour']>= '13') & (df3['Screening.Hour'] < '17'), 'Tim
df3.loc[(df3['Screening.Hour']>= '17') & (df3['Screening.Hour'] < '19'), 'Tim
df3.loc[(df3['Screening.Hour']>= '19') & (df3['Screening.Hour'] < '20'), 'Tim
df3.loc[(df3['Screening.Hour']>= '20') & (df3['Screening.Hour'] <= '23'), 'Ti
df3.loc[(df3['Screening.Hour']>= '00') & (df3['Screening.Hour'] < '05'), 'Tim
df3['Time.Group']
```

Out[17]:
```
0        LateEvening
1        LateEvening
2          Afternoon
3          Afternoon
4            Morning
            ...
88472       Midnight
88473       Midnight
88474       Midnight
88475       Midnight
88476       Midnight
Name: Time.Group, Length: 88477, dtype: object
```

From our analysis, this is the data that we have collected and we are able to see that during every time of the day, there will be people watching films at the cinema.

In [18]:
```
MPT=df3.groupby('Time.Group').agg({'Time.Group':'count'})
MPT = MPT.rename(columns = {'Time.Group':'NumberOfAudience'}).reset_index()
MPT #MOST POPULAR TIME
```

Out[18]:

	Time.Group	NumberOfAudience
0	Afternoon	31511
1	EarlyEvening	14331
2	LateEvening	7209
3	Midnight	596
4	Morning	5275
5	Night	25533
6	Noon	4022

Here, we can see that the time of the day that has the highest number of people watching films is during the Afternoon, with a total number of audiences of 31511 out of the 88477 people.

In [19]:
```
MPT.loc[MPT['NumberOfAudience']==max(MPT['NumberOfAudience'])]
```

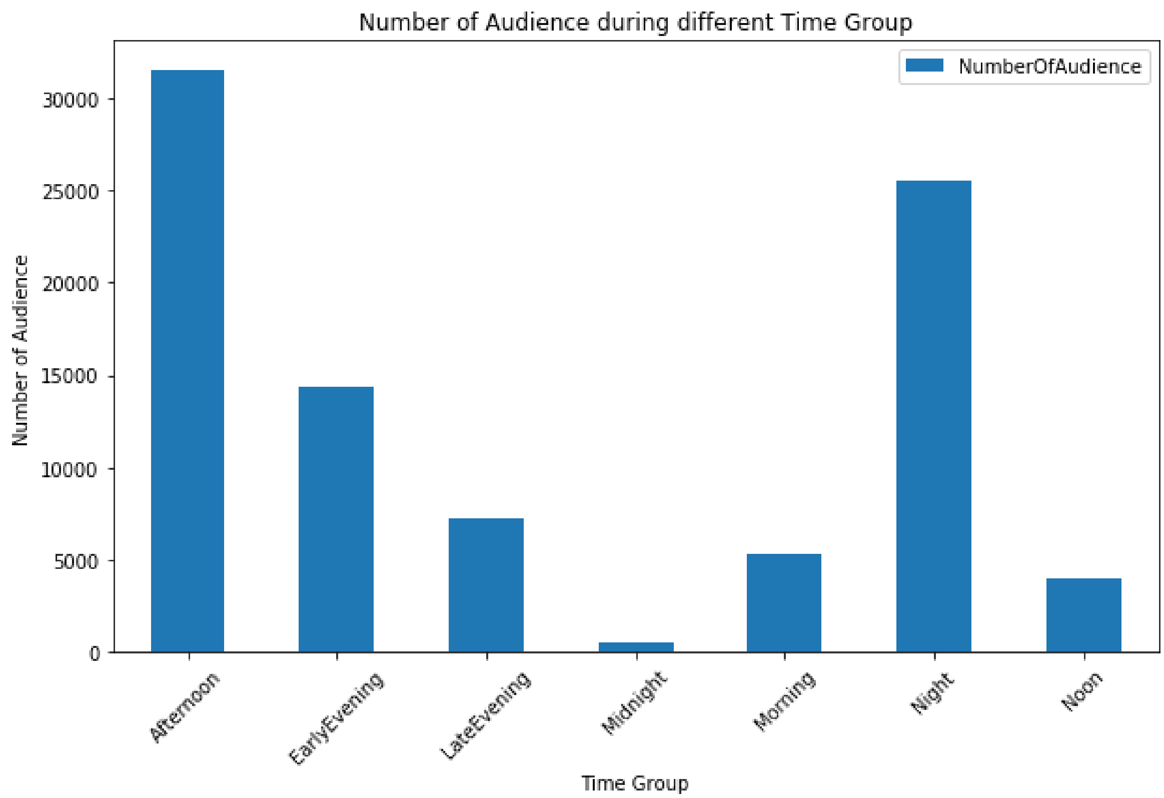
Out[19]:

	Time.Group	NumberOfAudience
0	Afternoon	31511

We could also plot a bar graph to reassure our answer.
From the graph,we would be able to clearly see which time had the most and least people.

In [20]:
```
bar3 = MPT.plot.bar(figsize=(10,6))
bar3.set_xticklabels(MPT['Time.Group'], rotation = 45)
plt.xlabel('Time Group')
plt.ylabel('Number of Audience')
plt.title('Number of Audience during different Time Group')
```

Out[20]:
```
Text(0.5, 1.0, 'Number of Audience during different Time Group')
```



And clearly from the graph, we can see that the number of audience during Afternoon is the highest and is roughly exceeding 30000 number of audiences.
We can also notice that the second most popular time of the day to watch movies would be during Night.

QUESTION 4

For this question, we will be finding which user has the best averaged order time and is the most efficient when handling ticket sales.
We will be excluding the web tickets as those transactions are automated and not handled by

physical users. It would be easier to just filter out the web tickets rather than filtering the users as there are WEB and NoShow, which are both assigned to web ticket transactions. NoShow typically stands for people not showing up or there are maybe some circumstances that are not recorded. WEB basically just refers to automated ticket handling in the website.
We will only be taking into account for the other users that handle tickets such as the standards, early birds etc.

In [21]:

```
df4= df3.loc[:,('User','Ticket.Type','Order.Time..Secs.')]
df4 = df4[df4['Ticket.Type'].str.contains('Web')==False]
df4
```

Out[21]:

	User	Ticket.Type	Order.Time..Secs.
45	User_01	Standard $9.00	26
46	User_01	Standard $9.00	42
47	User_01	Standard $9.00	42
48	User_01	Standard $9.00	42
49	User_01	Standard $9.00	42
...	...	...	...
88472	User_18	Standard $10.00	73
88473	User_18	Standard $9.00	44
88474	User_18	Standard $9.00	44
88475	User_18	Standard $9.00	44
88476	User_18	Standard $9.00	44

76116 rows × 3 columns

In [22]:
```
AOT = df4.groupby('User').agg({'Order.Time..Secs.':'mean'}).reset_index()
AOT = AOT.rename(columns={'Order.Time..Secs.':'Average.Order.Time'})
AOT #AVERAGE ORDER TIME
```

Out[22]:

	User	Average.Order.Time
0	Mgr_01	63.928012
1	Mgr_02	51.057462
2	Mgr_03	63.657247
3	Mgr_04	55.588410
4	Mgr_05	58.430380
5	Mgr_06	108.167665
6	User_01	50.144928
7	User_02	59.410838
8	User_03	60.672909
9	User_04	76.284364
10	User_05	72.029255
11	User_06	58.981793
12	User_07	53.854359
13	User_08	73.897017
14	User_09	62.739976
15	User_10	62.209181
16	User_11	64.889959
17	User_12	64.961033
18	User_13	49.155963
19	User_14	62.197136
20	User_15	53.827586
21	User_16	53.144948
22	User_17	69.664723
23	User_18	61.253000
24	User_19	63.969216
25	User_20	60.247510
26	User_21	67.582164
27	User_22	73.419619
28	User_23	52.451726

As we can see from our analysis, the user that has the lowest average order time is User_13, hence this makes User_13 the most efficient as User_13 takes the lowest amount of time to handle ticket sales.

In [23]:
```
AOT[AOT['Average.Order.Time']==min(AOT['Average.Order.Time'])]
```

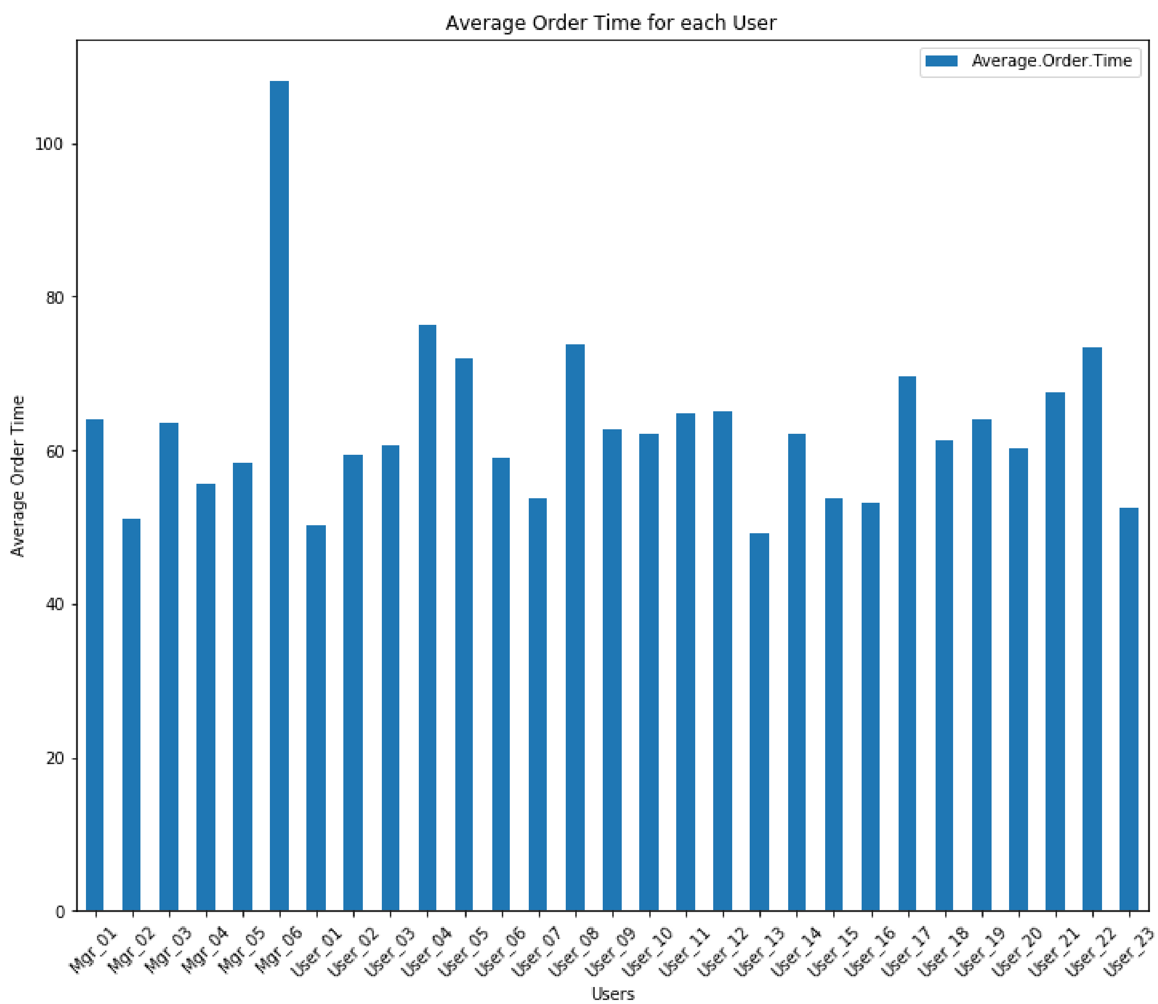
Out[23]:

	User	Average.Order.Time
18	User_13	49.155963

Other than that, We can also reassure our findings by plotting a bar graph and observe which user has the lowest average order time.

In [24]:
```
bar4 = AOT.plot.bar(figsize=(12,10))
bar4.set_xticklabels(AOT['User'],rotation = 45)
plt.xlabel('Users')
plt.ylabel('Average Order Time')
plt.title('Average Order Time for each User')
```

Out[24]:
```
Text(0.5, 1.0, 'Average Order Time for each User')
```



From the bar graph, we can find few relatively low average order time and efficient users. However, the user that has the lowest average order time and hence the most efficient, is User_13.

To end our report,

I would like to provide a few business insights on how to improve the cinema's overall business and performance.

1. Starting off, from what we have known from our analysis in Question 1, Film 5 is the best selling film and has the highest revenue. Hence, I would recommend that the respective cinema would maybe increase the screening time of Film 5 in order to generate more revenue for the cinema. At the same time, screening time for films that did not generate a reasonable amount of revenue should also be decreased, such as Film 32 , Film 33, and Film 8

2. Secondly, from Question 2 we have found out that Tuesday is the least popular day to watch a movie. Hence, the cinema should avoid having excessive amount of screening time on Tuesdays as there are least people that watch movies on Tuesdays. The respective cinema could also incorporate promotions or discounts on Tuesdays in order to attract customers to come watch movies on Tuesdays.

3. Moving on, we have also discovered that the most popular time of the day for movie goers are Afternoon. And so, the respective cinema should also use this to their advantage into making more profits. For example, the respective cinema could incorporate more sets for food and beverages during the afternoon. As we all know, it is a basic habit to purchase some popcorn or even sodas before we watch movies at the cinema. And so, due to the high amount of people coming to watch movies on Afternoons, the cinema will be able to generate a reasonable amount of profit from the F&B section.

4. Lastly, we had also found out that User_13 has done an amazing job of being very efficient with handling the ticket sales as User_13 has the lowest average order time. On the other hand, we also notice that user Mgr_06 has a very high average order time when compared to other users. Hence, I would strongly suggest the cinema's management team to keep an eye at user Mgr_06's performance and help the respective user if he is facing any kind of difficulties when handling the ticket sales. If the problem is with the respective user's attitude or work ethic, then I would suggest finding other workers or prospects that are of a better fit to replace that respective user.