Assignment 3

* Student Name: Zixin Hao
* Student ID: ******
* Tutorial Code: 05-P1
* Tutor: **** and ********

Task A

Q1: Before decompression, it is 110M. After decompression, it is 344M.

Code1:

```
$ ls -lh FB_Dataset.csv.zip
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ ls -lh FB_Dataset.csv.zip
-rwxrwx---+ 1 Zixin None 110M 10月 24 10:39 FB_Dataset.csv.zip
```

Explanation:

Check the size of the file (FB_Dataset.csv.zip) using the "-lh" option in ls; Ls means "list".

Code2:

```
$ unzip FB_Dataset.csv.zip
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ unzip FB_Dataset.csv.zip
Archive:  FB_Dataset.csv.zip
  inflating: FB_Dataset.csv
   creating: __MACOSX/
  inflating: __MACOSX/._FB_Dataset.csv
```

Explanation:

The code is used to decompress the zip file.

Code3:

```
$ ls -lh FB_Dataset.csv
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ ls -lh FB_Dataset.csv
-rw-r--r--+ 1 Zixin None 344M 9月  13 2019 FB_Dataset.csv
```

Explanation:

Check the size of the file (FB_Dataset.csv) using the -lh option in ls;
Ls means “list”;

Q2: Comma (‘,’) is the delimiter used to separate the columns in the file.

Code1:

```
$ cat FB_Dataset.csv|head -n2|less -S
```

Output:

```
page_name,post_id,page_id,post_name,message,description,caption,post_type,status>
abc-news,86680728811_272953252761568,86680728811,Chief Justice Roberts Responds >
~
~
```

Explanation:

“Cat”- load the file and output it to the terminal;
‘|’- pipe operator used to connect programs and pass data flow;
“head -n2”- take the first two line by using the parameter ‘-n2’;
“Less -s” – view the contents
So, the code is to show the first two line from the file loaded.

Code2:

```
$ cat FB_Dataset.csv|head -n1|less
```

Output:

```
page_name,post_id,page_id,post_name,message,description,caption,post_type,stat
us_type,likes_count,comments_count,shares_count,love_count,wow_count,haha_coun
t,sad_count,thankful_count,angry_count,post_link,picture,posted_at
```

Explanation:

The code is to show the first line of the file loaded.

Q3: 16

Code1:

```
$ awk -F ',' 'NR>1 {print $1}' FB_Dataset.csv | uniq | wc -l
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' 'NR>1 {print $1}' FB_Dataset.csv | uniq | wc -l
16
```

Explanation:

"Awk" is to process a text file one line at a time (break up each line into columns (considering ',' is the delimiter) and print the first column which is 'page_name').

The code is to select a subset of the columns without title line and show unique pages line by line, and then, count the number of lines to get the number by using "wc -l".

Q4: date range: from 1/1/12 0:30 to 7/11/16 23:45

Code1:

```
$ awk -F ',' 'NR>1 {print $21}' FB_Dataset.csv | head -1
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' 'NR>1 {print $21}' FB_Dataset.csv | head -1
1/1/12 0:30
```

Explanation:

The code is to break up each line into columns by comma and print the 21th column without head and then, take the first line. (it Assumes that the data is in order so the first line must be earliest.)

Code2:

```
$ awk -F ',' 'NR>1 {print $21}' FB_Dataset.csv | tail -1
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' 'NR>1 {print $21}' FB_Dataset.csv | tail -1
7/11/16 23:45
```

Explanation:

The code is to break up each line into columns by comma and print the 21th column without head, and then, take the last line. (it Assumes that the data is in order so the last line must be latest.)

Q5:

Times: 7610

When (first mention): 30/1/12 21:07

Post name: "Donald Trump Staff Reaching Out to Financers .. Campaign Managers to Explore Third Party Bid"

Assuming 'Angry_count' as a negative reaction and 'love_count' as a positive reaction.
'Angry_count'+'Love_count" as the total reactions

Code1:

```
$ awk -F ',' 'NR>1 {print $4}' FB_Dataset.csv | grep -i -o "Donald Trump" | wc -l
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' 'NR>1 {print $4}' FB_Dataset.csv | grep -i -o "Donald Trump" | wc -l
7610
```

Explanation:

The code is to choose the fourth column without head and pass the data flow to "grep" function which finds the name "Donald Trump" (ignore the case), finally count the number.

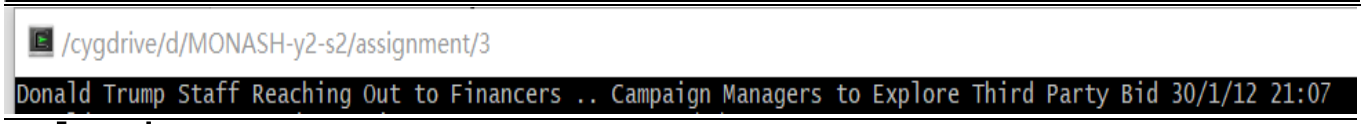
(the parameter of "-o" can let the function only print the matched parts of the file as list, no extra content is printed, so it won't miss any names on the same line with others)

Code2:

```
$ awk -F ',' 'NR>1 {print $4,$21}' FB_Dataset.csv | grep -i 'Donald Trump' |less -s
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' 'NR>1 {print $4,$21}' FB_Dataset.csv | grep -i 'Donald Trump' |less -s
```



Explanation:

The code is to break up line by line into columns by comma and print the 4th, 21th columns without head, and then to get lines which contains "Donald Trump" (ignore case) and view it.

Code3:

```
$ awk -F ',' '{print $2,$13,$18}' FB_Dataset.csv | less
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' '{print $2,$13,$18}' FB_Dataset.csv | less
```

/cygdrive/d/MONASH-y2-s2/assignment/3

```
post_id love_count angry_count
86680728811_272953252761568 0 0
86680728811_273859942672742 0 0
86680728811_10150499874478812 0 0
86680728811_244555465618151 0 0
86680728811_252342804833247 0 0
86680728811_200661383359612 0 0
86680728811_281125741936891 0 0
86680728811_10150500662053812 0 0
86680728811_10150500969563812 0 0
86680728811_10150501303143812 0 0
```

Explanation:

The code is to break up line by line into columns by comma and print the 2nd,13th, 18th columns and view to see people's positive and negative reactions for a post.

Code3:

```
$ awk -F ',' '{print $2, ($13+$18)}' FB_Dataset.csv | less
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' '{print $2,($13+$18)}' FB_Dataset.csv | less
```

```
86680728811_272953252761568 0
86680728811_273859942672742 0
86680728811_10150499874478812 0
86680728811_244555465618151 0
86680728811_252342804833247 0
86680728811_200661383359612 0
86680728811_281125741936891 0
86680728811_10150500662053812 0
86680728811_10150500969563812 0
86680728811_10150501303143812 0
86680728811_305275689511038 0
86680728811_10150501624593812 0
```

Explanation:

The code is to break up line by line into columns by comma and print the 2nd line (post_id) and the sum of 13th, 18th (total reactions).

Q6:

Code1:

```
$ cat FB_Dataset.csv | awk -F ',' '{print $2, $10}'|head -1 >>Trump.txt
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ cat FB_Dataset.csv | awk -F ',' '{print $2,$10}'|head -1 >>Trump.txt
```

```
post_id likes_count
```

Explanation:

The code is to load the file and pass it to "awk" function to break up lines into columns and print 2nd and 10th columns, and extract the first line to save it in Trump.txt file (attach the title into file).

Code2:

```
$ grep -i "Donald Trump" FB_Dataset.csv | awk -F ',' '$10>100 {print $2,
$10}'|head -5|sort -k2,2 -nr >> Trump.txt
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ grep -i "Donald Trump" FB_Dataset.csv | awk -F ',' '$10>100 {print $2, $10}'|head -5|sort -k2,2 -nr >> Trump.txt
```

```
post_id likes_count
86680728811_163656673780670 2023
86680728811_275700509178095 436
86680728811_322858374419787 349
86680728811_325466700825405 174
86680728811_147556555361835 124
```

Explanation:

The code is to filter all lines which contains "Donald Trump"(ignore case) will be passed to next function. Awk function break up lines into columns by comma and the filter is that the value of 10th column is bigger than 100. Then, extract first 5 lines and sort them based on second column. Finally, attach the data to Trump.txt file.

Task B1:

Q7:

Code1:

```
    $ awk -F ',' 'NR==1 {print $1, $8, $11}' FB_Dataset.csv | tr ' ' ',' >>
the-wall-street-journal.csv
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' 'NR==1 {print $1, $8, $11}' FB_Dataset.csv | tr ' ' ',' >> the-wall-street-journal.csv
```

	A	B	C	D	E	F	G	H	I	J	K
1	page_nam	post_type	comments_count								
2											
3											
4											
5											
6											
7											
8											
9											

Explanation

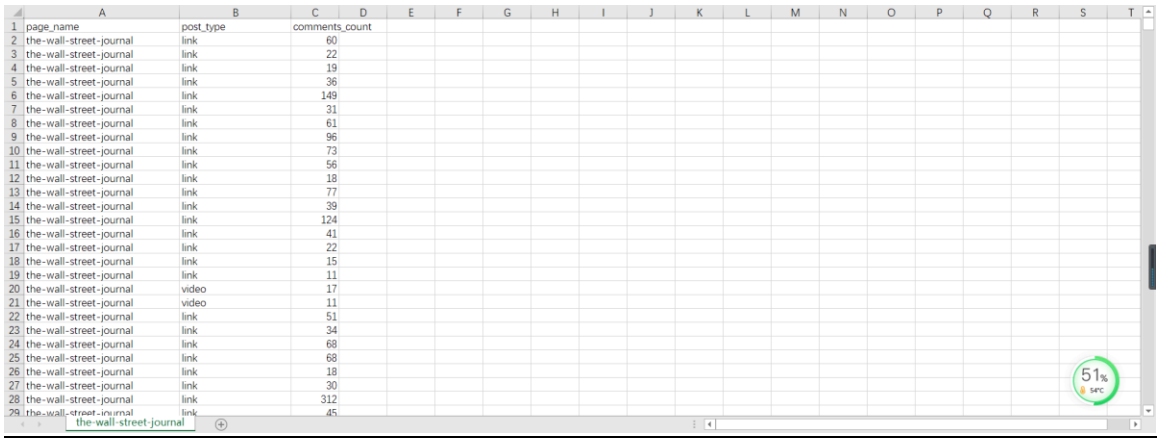
The code is to break up lines into columns and select the three columns header, then to replace delimiter(space) with comma. Save it as csv file.

Code2:

```
$ awk -F ',' '{print $1, $8, $11}' FB_Dataset.csv | grep -i "the-wall-street-
journal" | tr ' ' ',' >> the-wall-street-journal.csv
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' '{print $1,$8,$11}' FB_Dataset.csv|grep -i "the-wall-street-journal"|tr ' ' ',' >> the-wall-street-journal.csv
```



page_name	post_type	comments_count
the-wall-street-journal	link	60
the-wall-street-journal	link	22
the-wall-street-journal	link	19
the-wall-street-journal	link	36
the-wall-street-journal	link	149
the-wall-street-journal	link	31
the-wall-street-journal	link	61
the-wall-street-journal	link	96
the-wall-street-journal	link	73
the-wall-street-journal	link	56
the-wall-street-journal	link	18
the-wall-street-journal	link	77
the-wall-street-journal	link	39
the-wall-street-journal	link	124
the-wall-street-journal	link	41
the-wall-street-journal	link	22
the-wall-street-journal	link	15
the-wall-street-journal	link	11
the-wall-street-journal	video	17
the-wall-street-journal	video	11
the-wall-street-journal	link	51
the-wall-street-journal	link	34
the-wall-street-journal	link	68
the-wall-street-journal	link	68
the-wall-street-journal	link	18
the-wall-street-journal	link	30
the-wall-street-journal	link	312
the-wall-street-journal	link	45

Explanation:

The code is to break up lines into columns and select the three columns data (only consider lines which contain "the-wall-street-journal"), then to replace delimiter(space) with comma. Save it as csv file.

Code3:

```
library("ggplot2")
setwd("D:/MONASH-y2-s2/assignment/3")
mydata=read.csv("the-wall-street-journal.csv")
head(mydata)
mydata2 <- subset(mydata,mydata$comments_count<4000)
nrow(mydata)
nrow(mydata2)
```

Output:

```
> setwd("D:/MONASH-y2-s2/assignment/3")
> mydata=read.csv("the-wall-street-journal.csv")
> head(mydata)
                  page_name post_type comments_count
1 the-wall-street-journal      link             60
2 the-wall-street-journal      link             22
3 the-wall-street-journal      link             19
4 the-wall-street-journal      link             36
5 the-wall-street-journal      link            149
6 the-wall-street-journal      link             31
> mydata2 <- subset(mydata,mydata$comments_count<4000)
> nrow(mydata)
[1] 35574
> nrow(mydata2)
[1] 35569
```

Explanation:

The code is a pre-process for the data, set a word directory in which we can find

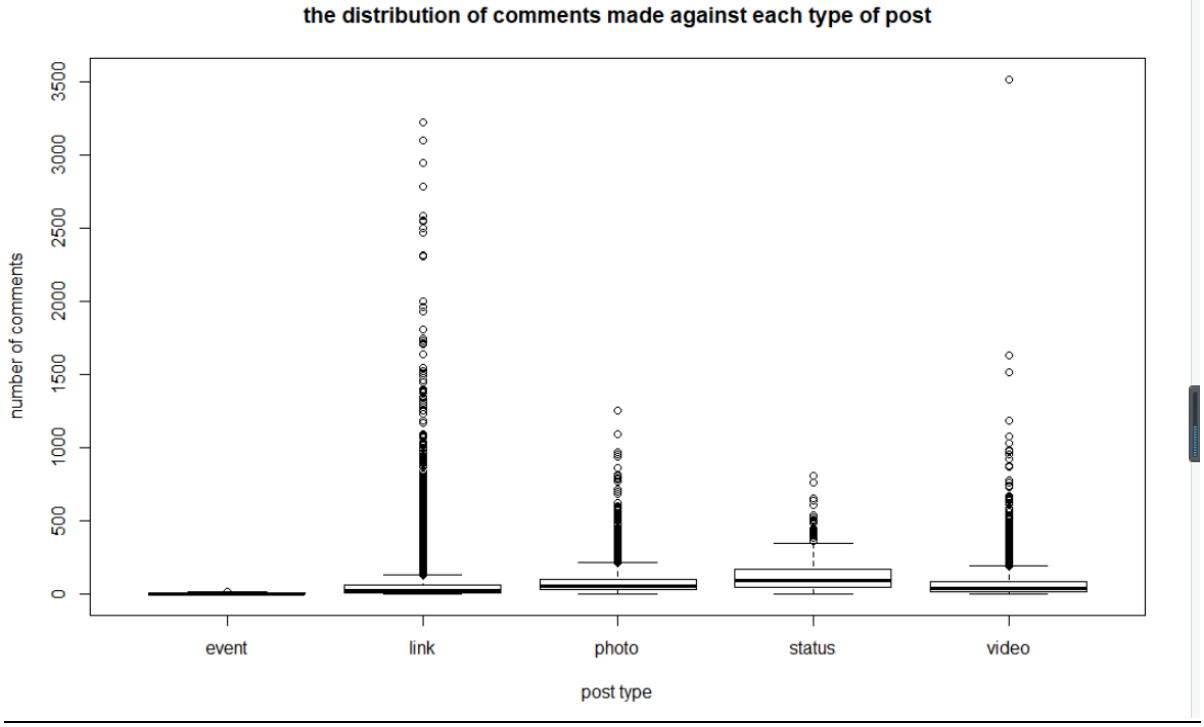
the file (the-wall-street-journal.csv). Read the excel and filter the data based on the requirements of “the-wall-street-journal” which is “comments_count” < 4000.

Code4:

```
Boxplot (mydata2$comments_count~mydata2$post_type, data=mydata2,
        xlab="post type", ylab="number of comments",
        main ="the distribution of comments made against each type of post")
```

Output:

```
> boxplot(mydata2$comments_count~mydata2$post_type, data=mydata2,
+         xlab="post type",ylab="number of comments",
+         main ="the distribution of comments made against each type of post")
```



Explanation:

Using ggplot2 library, to create a boxplot. (The “comments_count” column of the “mydata” dataframe against the “post_type” column.)

Question answer:

Infer:

75% data, for all boxes (all kinds of type), they are smaller than 500 and the location is so close. So, I infer, for most of the post, there are not too much affect made against each type of post to the number of comments.

The “Link” type has the most outliers and the value is much higher than other types’, and “event” type has only one outlier probably. So, I infer the number of comments for “event” type posts is concentrated. But “link” type post is more dispersive, I infer it may be caused by post content or people curiously want to know what the content behind the link so that there are many big value outliers.

Can you detect? :

I cannot detect which on is the most engaging post type because there are too many outliers and the boxes are so flat.

“Status” type box has a tiny higher upper & lower quartiles and median value than other types’, we can get, at least 75% data (75% posts), comment of this kind of type is little higher (the value is still smaller (around 300)).

But the “link” type has many outliers of high value. We cannot clearly know which one is the most engaging post type.

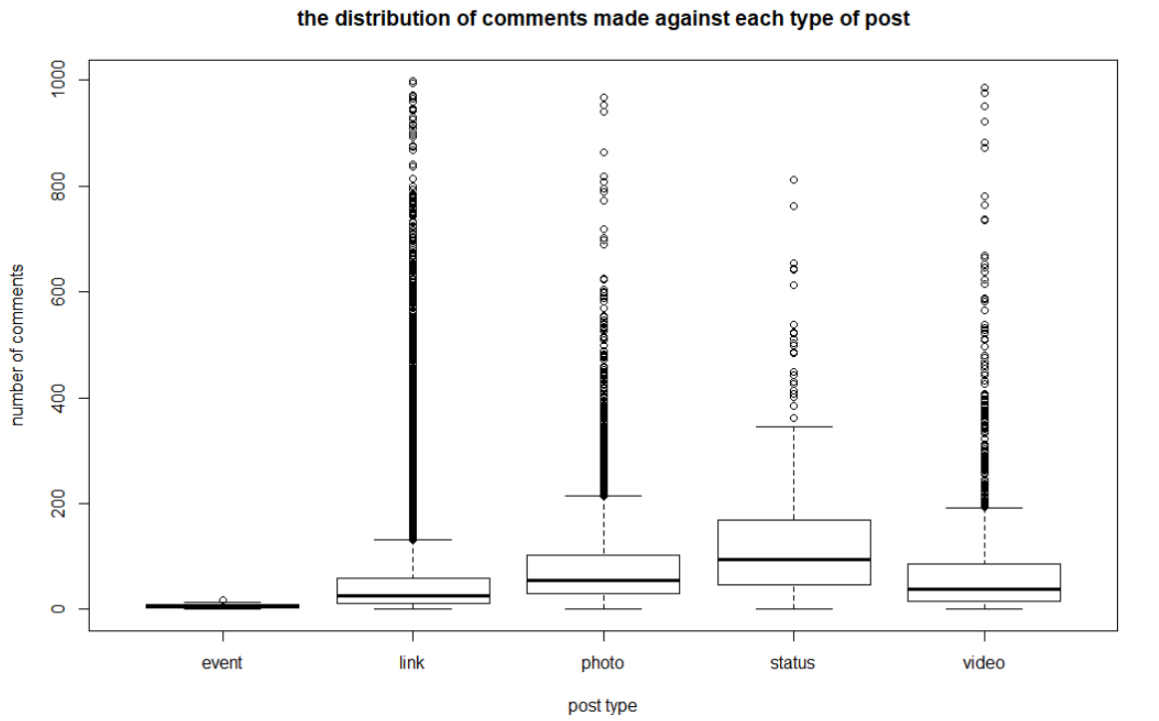
Q8:

Code1:

```
mydata3 <- subset (mydata, mydata$comments_count<=1000)
boxplot (mydata3$comments_count~mydata3$post_type, data=mydata3,
         xlab="post type", ylab="number of comments",
         main ="the distribution of comments made against each type of post")
```

Output:

```
> mydata3 <- subset(mydata,mydata$comments_count<=1000)
> boxplot(mydata3$comments_count~mydata3$post_type, data=mydata3,
+         xlab="post type",ylab="number of comments",
+         main ="the distribution of comments made against each type of post")
> |
```



Explanation:

filter the data based on the requirements of “the-wall-street-journal” which is “comments_count” <= 1000 to filter out those outliers. And Using ggplot2 library, to create a boxplot. (The “comments_count” column of the “mydata” dataframe against the “post_type” column)

Q9:

Code1:

(1)

```
library(dplyr)
aggregate(mydata$comments_count~mydata$post_type,mydata,median)
```

(2)

```
library(dplyr)
aggregate(mydata2$comments_count~mydata2$post_type,mydata,median)
```

(3)

```
library(dplyr)
aggregate(mydata3$comments_count~mydata3$post_type,mydata,median)
```

Output:

```
> aggregate(mydata$comments_count~mydata$post_type,mydata,median)
  mydata$post_type mydata$comments_count
1            event                     5
2             link                    25
3            photo                    54
4           status                    94
5            video                    38
> aggregate(mydata2$comments_count~mydata2$post_type,mydata,median)
  mydata2$post_type mydata2$comments_count
1             event                     5
2              link                    25
3             photo                    54
4            status                    94
5             video                    38
> aggregate(mydata3$comments_count~mydata3$post_type,mydata,median)
  mydata3$post_type mydata3$comments_count
1             event                     5
2              link                    25
3             photo                    54
4            status                    94
5             video                    38
```

Explanation:

Aggregate function to group the dataset against post type and calculate the median of each group.

Note:

Mydata$comments_count (the column of comments_count) against mydata$post_type (the column of post_type);

Using mydata/ mydata2/ mydata3 dataset;

Question answer:

"mydata" is the dataset has no filter;

"mydata2" is the dataset has filter that is commends number is less than 4000

"mydata3" is the dataset which has filtered out values (comments_count) greater than 1000

So, for dataset all kinds of datasets, **"status"** has the highest median comment count.

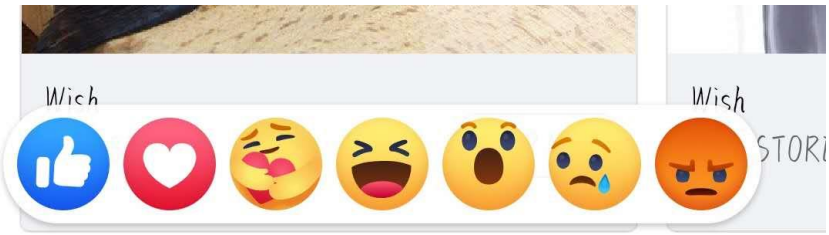
Task B2:

Q10:

I assume likes_counts, love_counts, wow_counts, haha_counts, sad_counts, thankful_counts and angry_counts are reactions people will be. They all represent a kind of reaction in different degree (positive or negative).

"Comments_counts" and "shares_counts" are not included because I think they cannot be considered a reaction, for example, the reasons of people giving a comment and sharing the post are many (maybe because of angry, maybe love, maybe because of something not related to reaction at all).

Besides, in Facebook, there are the 7 reactions.

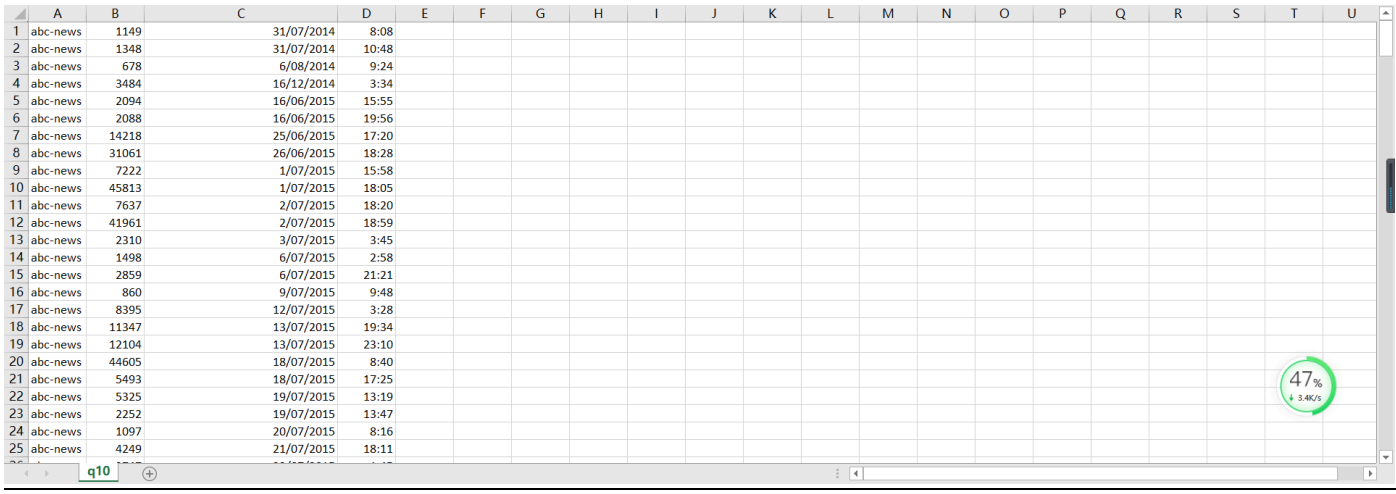


Code1:

```
    $ awk -F ',' ' $1=="abc-news" && $5~/[Dd][Oo][Nn][Aa][Ll][Dd]
[Tt][Rr][Uu][Mm][Pp]/ {print $1,$10+$13+$14+$15+$16+$17+$18,$21}'
FB_Dataset.csv |tr ' ' ',' > q10.csv
```

Output:

```
Zixin@LAPTOP-HG93CI3D /cygdrive/d/MONASH-y2-s2/assignment/3
$ awk -F ',' ' $1=="abc-news" && $5~/[Dd][Oo][Nn][Aa][Ll][Dd] [Tt][Rr][Uu][Mm][Pp]/ {print $1,$10+$13+$14+$15+$16+$17+$18,$21}' FB_Dataset.csv |tr ' ' ',' > q10.csv
```



	A	B	C	D
1	abc-news	1149	31/07/2014	8:08
2	abc-news	1348	31/07/2014	10:48
3	abc-news	678	6/08/2014	9:24
4	abc-news	3484	16/12/2014	3:34
5	abc-news	2094	16/06/2015	15:55
6	abc-news	2088	16/06/2015	19:56
7	abc-news	14218	25/06/2015	17:20
8	abc-news	31061	26/06/2015	18:28
9	abc-news	7222	1/07/2015	15:58
10	abc-news	45813	1/07/2015	18:05
11	abc-news	7637	2/07/2015	18:20
12	abc-news	41961	2/07/2015	18:59
13	abc-news	2310	3/07/2015	3:45
14	abc-news	1498	6/07/2015	2:58
15	abc-news	2859	6/07/2015	21:21
16	abc-news	860	9/07/2015	9:48
17	abc-news	8395	12/07/2015	3:28
18	abc-news	11347	13/07/2015	19:34
19	abc-news	12104	13/07/2015	23:10
20	abc-news	44605	18/07/2015	8:40
21	abc-news	5493	18/07/2015	17:25
22	abc-news	5325	19/07/2015	13:19
23	abc-news	2252	19/07/2015	13:47
24	abc-news	1097	20/07/2015	8:16
25	abc-news	4249	21/07/2015	18:11

Explanation:

The code is first to select rows which contains “Donald Trump” (ignore case) and “abc-news” from the file by using $1==”abc-news” (“$1==”abc-news”” means print those columns and lines only when the first column equals to “abc-news”). Then select these columns and add all reactions together as one column, Finally, replace space with comma and save it as a csv file to get a relatively clear table.

code2:

In [24]:
```
import pandas as pd
import time,datetime
import matplotlib.pyplot as plt
```

In [25]:
```
df = pd.read_csv('D:/MONASH-y2-s2/assignment/3/q10.csv',header=None)
```

In [26]:
```
df.columns=['page_name','reaction_count','posted_at','time']
df
```

Out[26]:

	page_name	reaction_count	posted_at	time
0	abc-news	1149	31/7/14	8:08
1	abc-news	1348	31/7/14	10:48
2	abc-news	678	6/8/14	9:24
3	abc-news	3484	16/12/14	3:34
4	abc-news	2094	16/6/15	15:55
...	...	...	...	...
289	abc-news	147	25/10/16	11:59
290	abc-news	4837	2/11/16	1:56
291	abc-news	2292	2/11/16	11:48
292	abc-news	285	3/11/16	11:09
293	abc-news	857	6/11/16	12:02

294 rows × 4 columns

explaination: the code is to import libraries I need; read data generated by unix shell from this path and rename columns'name

code3:

In [27]:
```
df.posted_at=pd.to_datetime(df.posted_at,dayfirst=True)
df['day']=df['posted_at'].dt.dayofweek
```

In [28]:
```
df
```

Out[28]:

	page_name	reaction_count	posted_at	time	day
0	abc-news	1149	2014-07-31	8:08	3
1	abc-news	1348	2014-07-31	10:48	3
2	abc-news	678	2014-08-06	9:24	2
3	abc-news	3484	2014-12-16	3:34	1
4	abc-news	2094	2015-06-16	15:55	1
...	...	...	...	...	...
289	abc-news	147	2016-10-25	11:59	1
290	abc-news	4837	2016-11-02	1:56	2
291	abc-news	2292	2016-11-02	11:48	2
292	abc-news	285	2016-11-03	11:09	3
293	abc-news	857	2016-11-06	12:02	6

294 rows × 5 columns

explaination: the code is to change datatype of the date and transform it into weekdays,then add one column to record it.

code4:

In [29]:
```
df2 = df.groupby('day').agg({'reaction_count':'sum'})
df2=df2.reset_index()
df2
```

Out[29]:

	day	reaction_count
0	0	152459
1	1	204462
2	2	129759
3	3	177277
4	4	172716
5	5	142866
6	6	201967

explaination: group the data by "day" (group by weekdays) and sum the reaction_count

code5:

In [30]:
```
def get_week_day(day):
    List=[]

    week_day_dict = {
            0 : 'Monday',
            1 : 'Tuesday',
            2 : 'Wednesday',
            3 : 'Thursday',
            4 : 'Friday',
            5 : 'Saturday',
            6 : 'Sunday',
      }
    for value in day:
        List.append(week_day_dict[value])

    daySer = pd.Series(List)
    return daySer
```

In [31]:
```
df2['week_day']=get_week_day(df2.day)
```

In [32]:
```
df2
```

Out[32]:

	day	reaction_count	week_day
0	0	152459	Monday
1	1	204462	Tuesday
2	2	129759	Wednesday
3	3	177277	Thursday
4	4	172716	Friday
5	5	142866	Saturday
6	6	201967	Sunday

explaination: the codes are to define a function that can transform the number in the "day" column into string. Save weekdays as a column.

code6:

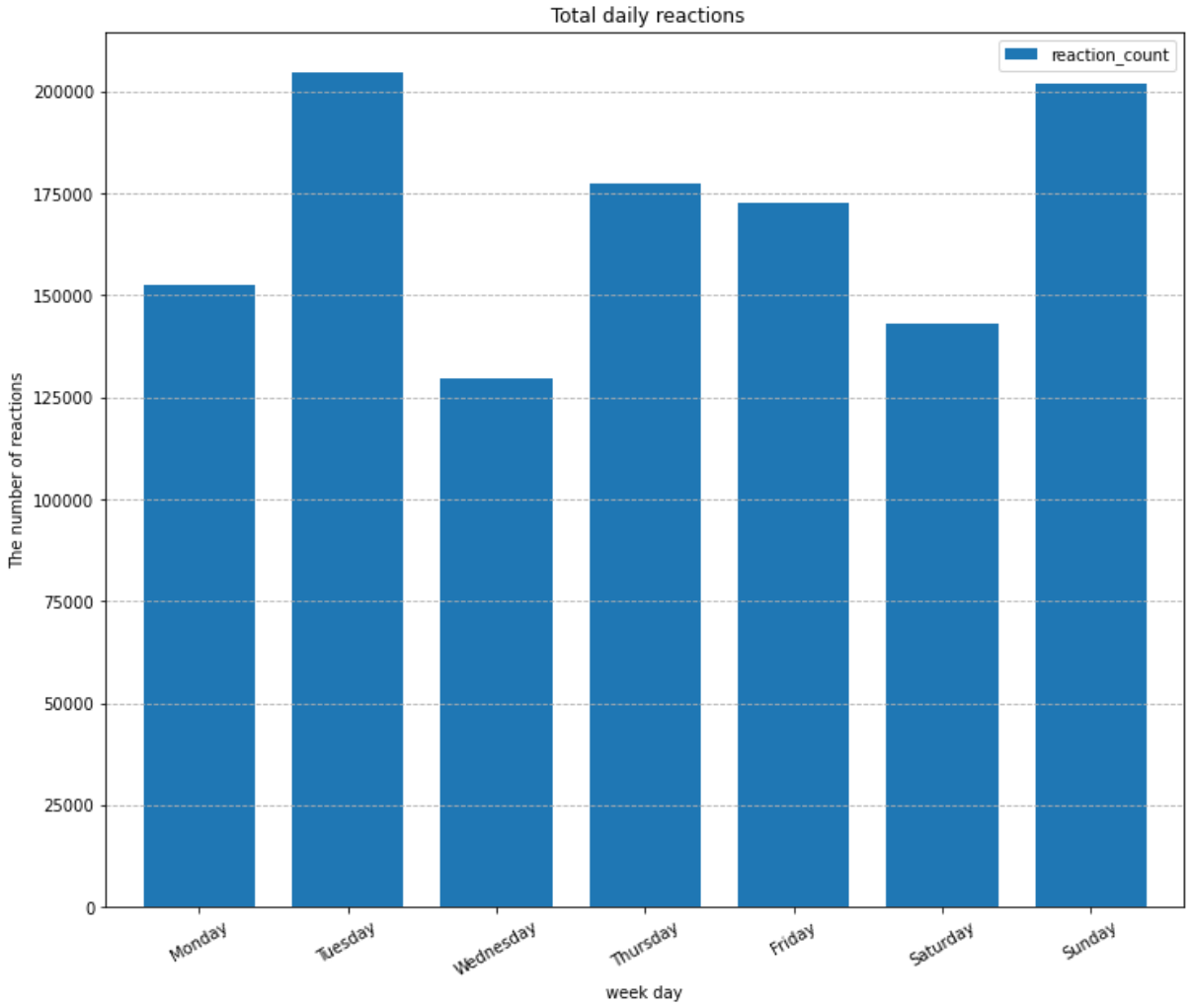
In [33]:
```
df2=df2[['reaction_count','week_day']]
df2
```

Out[33]:

	reaction_count	week_day
0	152459	Monday
1	204462	Tuesday
2	129759	Wednesday
3	177277	Thursday
4	172716	Friday
5	142866	Saturday
6	201967	Sunday

In [34]:
```
chart = df2.plot.bar(figsize=(12,10),width = 0.75)
chart.set_xticklabels(df2['week_day'],rotation=30)
plt.xlabel('week day')
plt.ylabel('The number of reactions')
plt.title(' Total daily reactions')

plt.grid(axis='y',linestyle='--')
```



explaination: the code is to select a sub-dataset to draw a bar chart

Q11

Tuesday and Sunday, users have shown the most reactions to the posts.

there are no big difference between the number of reactions on average during the weekdays and at the weekends.But Sundays of weekends have more people's reaction than most of weekdays, and Saturdays have fewer people's reaction than most of weekdays.

Q12

code1:

In [35]:
```
dft = df[(df.day==1)]
dfs = df[(df.day==6)]
dft=dft.reset_index()
dfs=dfs.reset_index()
```

explaination: the code is to filter these two day in which users have shown the most reactions to the posts, dataframes of them is represented by "dfs" and "dfm"

code2:

In [36]:
```
def get_day_hour(time):
    List=[]

    for value in time:

        hour=value[:-3]

        List.append(hour)


    hourSer = pd.Series(List)
    return hourSer
```

In [37]:
```
dft['hour']=get_day_hour(dft.time)
dfs['hour']=get_day_hour(dfs.time)
```

In [38]:
```
dft.head()
```

Out[38]:

	index	page_name	reaction_count	posted_at	time	day	hour
0	3	abc-news	3484	2014-12-16	3:34	1	3
1	4	abc-news	2094	2015-06-16	15:55	1	15
2	5	abc-news	2088	2015-06-16	19:56	1	19
3	24	abc-news	4249	2015-07-21	18:11	1	18
4	27	abc-news	775	2015-08-04	22:11	1	22

explaination: the code is a function that can extract the hour of post time for each post from the "time" column.

code3:

In [39]:
```
dft.hour = dft.hour.astype('int')
dft=dft.groupby('hour').agg({'reaction_count':'sum'}).reset_index().sort_values(by='hour')

dfs.hour = dfs.hour.astype('int')
dfs=dfs.groupby('hour').agg({'reaction_count':'sum'}).reset_index().sort_values(by='hour')
```

In [40]:
```
dft.head()
```

Out[40]:

	hour	reaction_count
0	0	21327
1	1	640
2	2	14190
3	3	5694
4	4	2280

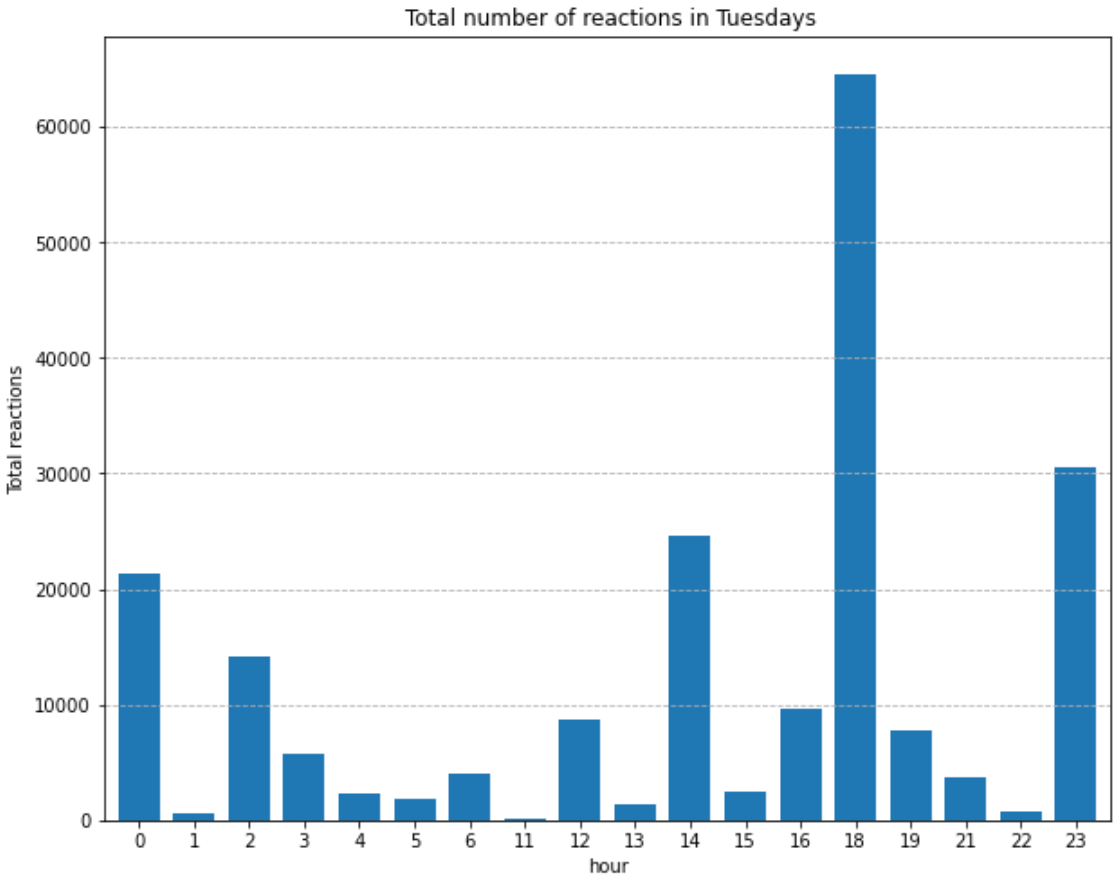
In [41]:
```
dfs.head()
```

Out[41]:

	hour	reaction_count
0	0	544
1	1	1370
2	2	14747
3	3	8395
4	4	3123

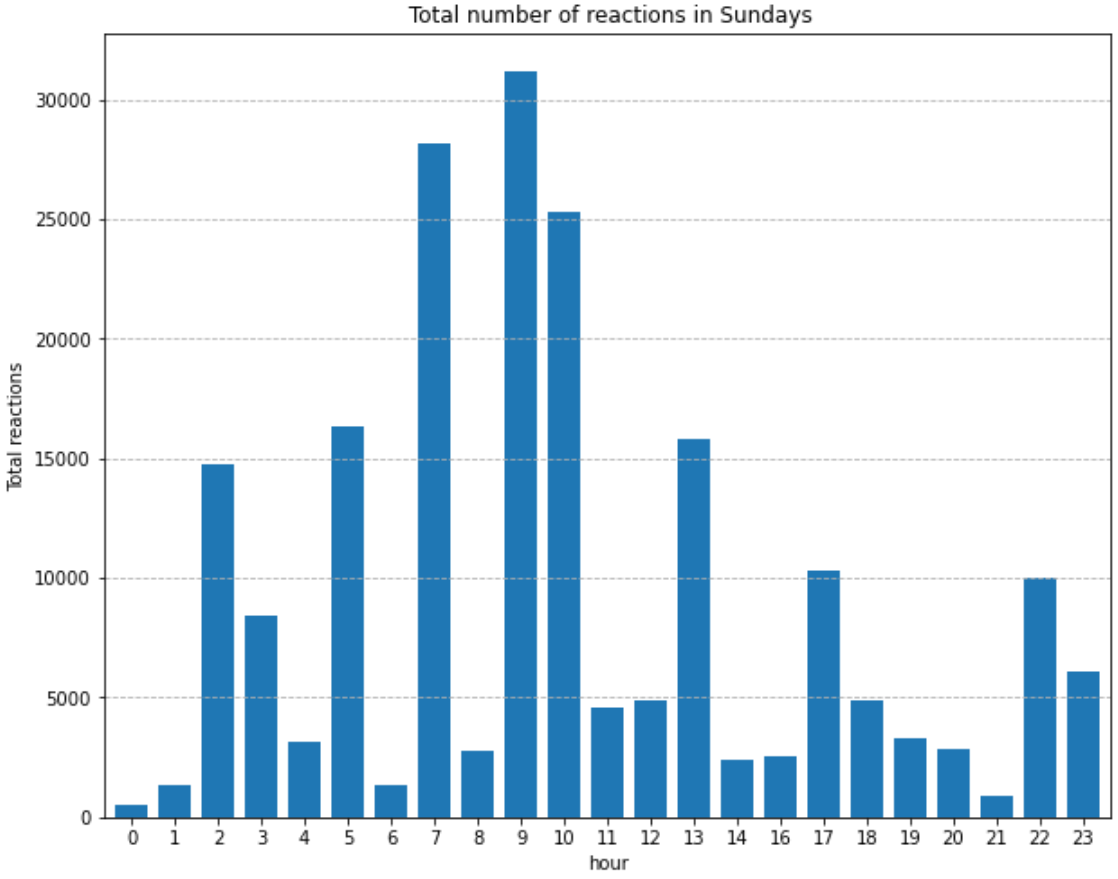
In [42]:
```
Tue = dft.reaction_count.plot.bar(figsize=(10,8),width = 0.75)
Tue.set_xticklabels(dft.hour,rotation=0)
plt.xlabel('hour')
plt.ylabel('Total reactions')
plt.title(' Total number of reactions in Tuesdays')

plt.grid(axis='y',linestyle='--')
```



In [43]:
```
Sun = dfs.reaction_count.plot.bar(figsize=(10,8),width = 0.75)
Sun.set_xticklabels(dfs.hour,rotation=0)
plt.xlabel('hour')
plt.ylabel('Total reactions')
plt.title(' Total number of reactions in Sundays')

plt.grid(axis='y',linestyle='--')
```



explaination: the code firstly change "hour" column's data type into integer,and sum all reactions after grouping by hours. Then sort the dataframe against the value of hours.Finally, draw the bar chart (hourly total reactions for each of two days)

```
Tuesdays: the most reactions happen around 18:00 pm
Sundays: the most reactions happen around 9 am

There are some similarity between the number of hourly reactions in these two days. First
ly,the general shapes(wave shape) of each days also are similar(like waves, the trend is
 first increasing and then decreasing and then increasing so on). There is a sub-period i
n the two days are similar, which is (2-4 hour), showing decreasing trend from the simila
r value(around 15000).
```

Q13

(a) 18:00pm Tuesdays in which had the maximum number of reactions

(b) I think it's a good idea.

I suggest publishing a general post about Trump in Sundays and Tuesdays and at the peak hours which is 9:00am, 18:00pm respectively. Because these two days and peak hours get the most people to look at facebook and to react for the post.

The posts posted in these time period have some benefits: Being seen by more people; increases its exposure(It had a bigger impact); For the publisher, half the effort is twice the result

In [ ]: