

Loading Snowboard data.

This notebook is all about cleaning the snowboard data, checking some condition as per user's requirement.

- Loaded the Snowboard data with all fields.
- Handling of hexadecimal values.
- Null data handling.
- Datatype change.
- Addressed user's requirement.
- Discounted Damage calculation.
- Filtered the data based on the duration of each scenario.
- Total damage calculation.
- Handling for Total Damage over 50000\$.
- Exporting the filtered data into a new .csv file, which will be used in final notebook.

```

In [ ]: import pandas as pd
df =pd.read_csv('Low_Anchor.tsv', sep='\t+',skiprows=[0,2, 4]+list(range(1,1614,2
))
        + [1614], names = ['StartDate', 'EndDate',
'ResponseType',
'IP Address',
'Progress',
'Duration',
'Finished',
'RecordedDate',
'ResponseID',
'RecipientLastName','RecipientFirstName','RecipientEmail',
'ExternalDataReference','LocationLatitude', 'LocationLongitude',
'DistributionChannel', 'UserLanguage', 'Participation_in_this_project.',
'Browser Meta Info - Browser',
'Browser Meta Info - Version',
'Browser Meta Info - Operating System',
'Browser Meta Info - Resolution',
'What number did you hear?',
'What word did you see?',
'What is your sex?',
'How old are you?',
'Which of the following best describes your ethnicity?',
'Are you Spanish/Hispanic/Latino',
'What is the highest degree or level of school you have completed?',
'This is an attention check. Select 200.',
'Which of the following best describes your total household income?',
'Where would you place yourself on this scale?',
'What is your zip code?',
'Timing - First Click','Timing - Last Click',
'Timing - Page Submit', 'Timing - Click Count',
'Timing - First Click.1', 'Timing - Last Click.1',
'Timing - Page Submit.1',
'Timing - Click Count.1', 'Timing - First Click.2',
'Timing - Last Click.2',
'Timing - Page Submit.2','Timing - Click Count.2',
'Timing - First Click.3','Timing - Last Click.3',
'Timing - Page Submit.3','Timing - Click Count.3',
'Timing - First Click.4', 'Timing - Last Click.4',
'Timing - Page Submit.4','Timing - Click Count.4',
'Timing - First Click.5', 'Timing - Last Click.5',
'Timing - Page Submit.5','Timing - Click Count.5',
'Timing - First Click.6', 'Timing - Last Click.6',
'Timing - Page Submit.6', 'Timing - Click Count.6',
'Timing - First Click.7','Timing - Last Click.7',
'Timing - Page Submit.7', 'Timing - Click Count.7',
'Identify the statement that correctly describes the facts of this case. (This is
the attention check)',
'Was_snowboard_sold_McNeil_defective_14',
'Is_substantial_factor_McNeil_injuries_14",
'Non_economic_damages_McNeil_suffered_14',
'Damages_words_14',
'Was_McNeil_negligent',
'McNeil_negligence_substantial_factor_for_injuries',
'Percentage_of_responsibility_X5',
'Percentage_of_responsibility_McNeil',
'Was_snowboard_sold_McNeil_defective_58',
'Is_substantial_factor_McNeil_injuries_58",
'Economic_damages_McNeil_suffer_58',
'Economic_Damages_In_Word_58',
'Non_economic_damages_McNeil_suffered_58',

```

```
'Non_Economic_Damages_In_Word_58',
'Please explain why you arrived at your decision? (50 character minimum)',
'Q40',#'Did the fact that X5 added core inserts to the Later Carve 3000 model, affect your view as to whether the original Carve 3000 was defective?',
'Q41', #'Were you able to ignore the fact that X5 added core inserts to the Later Carve 3000 model when deciding whether the original Carve 3000 was defective?',
'Path']])
```

```
In [252]: print(df.shape)
```

```
(804, 84)
```

Replacing hexadecimal value of damages'/x00' to "(empty string)

The data set contains so many hexadecimal values so we have replaced them with empty string.

```
In [253]: for i in range(len(df)):
    df['Was_snowboard_sold_McNeil_defective_14'].values[i] = df['Was_snowboard_sold_McNeil_defective_14'].values[i].replace('\x00','')
    df['Is_substantial_factor_McNeil_injuries_14'].values[i] = df['Is_substantial_factor_McNeil_injuries_14'].values[i].replace('\x00','')
    df['Non_economic_damages_McNeil_suffered_14'].values[i] = df['Non_economic_damages_McNeil_suffered_14'].values[i].replace('\x00','')
    df['Damages_words_14'].values[i] = df['Damages_words_14'].values[i].replace('\x00','')
    df['Was_McNeil_negligent'].values[i] = df['Was_McNeil_negligent'].values[i].replace('\x00','') ;
    df['McNeil_negligence_substantial_factor_for_injuries'].values[i] = df['McNeil_negligence_substantial_factor_for_injuries'].values[i].replace('\x00','') ;
    df['Percentage_of_responsibility_X5'].values[i] = df['Percentage_of_responsibility_X5'].values[i].replace('\x00','') ;
    df['Percentage_of_responsibility_McNeil'].values[i] = df['Percentage_of_responsibility_McNeil'].values[i].replace('\x00','') ;
    df['Was_snowboard_sold_McNeil_defective_58'].values[i] = df['Was_snowboard_sold_McNeil_defective_58'].values[i].replace('\x00','') ;
    df['Is_substantial_factor_McNeil_injuries_58'].values[i] = df['Is_substantial_factor_McNeil_injuries_58'].values[i].replace('\x00','') ;
    df['Economic_damages_McNeil_suffer_58'].values[i] = df['Economic_damages_McNeil_suffer_58'].values[i].replace('\x00','') ;
    df['Economic_Damages_In_Word_58'].values[i] = df['Economic_Damages_In_Word_58'].values[i].replace('\x00','') ;
    df['Non_economic_damages_McNeil_suffered_58'].values[i] = df['Non_economic_damages_McNeil_suffered_58'].values[i].replace('\x00','') ;
    df['Non_Economic_Damages_In_Word_58'].values[i] = df['Non_Economic_Damages_In_Word_58'].values[i].replace('\x00','') ;
    df['Path'].values[i] = df['Path'].values[i].replace('\x00','') ;
    df['Q40'].values[i] = df['Q40'].values[i].replace('\x00','') ;
    df['Q41'].values[i] = df['Q41'].values[i].replace('\x00','') ;
    df['Duration'].values[i] = df['Duration'].values[i].replace('\x00','') ;
    df['What is the highest degree or level of school you have completed?'].values[i] = df['What is the highest degree or level of school you have completed?'].values[i].replace('\x00','') ;
    df['Which of the following best describes your total household income?'].values[i] = df['Which of the following best describes your total household income?'].values[i].replace('\x00','') ;
    #df['Was the Carve 3000 snowboard X5 sold Connor McNeil defective?'].values[i] = df['Was the Carve 3000 snowboard X5 sold Connor McNeil defective?'].values[i].replace('\x00','') ;
```

After dealing with Special Character in data, lets change the Data type of required columns

```
In [254]: df.StartDate = pd.to_datetime(df.StartDate)
df.EndDate = pd.to_datetime(df.EndDate)
#df.Was_snowboard_sold_McNeil_defective_14 = pd.to_numeric(df.Was_snowboard_sold_McNeil_defective_14)
df.Is_substantial_factor_McNeil_injuries_14 = pd.to_numeric(df.Is_substantial_factor_McNeil_injuries_14)
df.Non_economic_damages_McNeil_suffered_14 = pd.to_numeric(df.Non_economic_damages_McNeil_suffered_14)
df.Was_McNeil_negligent = pd.to_numeric(df.Was_McNeil_negligent)
df.McNeil_negligence_substantial_factor_for_injuries = pd.to_numeric(df.McNeil_negligence_substantial_factor_for_injuries)
df.Percentage_of_responsibility_X5 = pd.to_numeric(df.Percentage_of_responsibility_X5)
df.Percentage_of_responsibility_McNeil = pd.to_numeric(df.Percentage_of_responsibility_McNeil)
#df.Was_snowboard_sold_McNeil_defective_58 = pd.to_numeric(df.Was_snowboard_sold_McNeil_defective_58)
df.Is_substantial_factor_McNeil_injuries_58 = pd.to_numeric(df.Is_substantial_factor_McNeil_injuries_58)
df.Economic_damages_McNeil_suffer_58 = pd.to_numeric(df.Economic_damages_McNeil_suffer_58)
df.Non_economic_damages_McNeil_suffered_58 = pd.to_numeric(df.Non_economic_damages_McNeil_suffered_58)
df.Q40 = pd.to_numeric(df.Q40)
#df.Q41 = pd.to_numeric(df.Q41)
# Handling for Path
df.Path = pd.to_numeric(df.Path)
df['Path'].fillna(0,inplace = True)
df.Duration = pd.to_numeric(df.Duration)
df.Duration = df.Duration.astype(int)
df.Path = df.Path.astype(int)
df.dtypes
```

```
Out[254]: StartDate          datetime64[ns]
          EndDate            datetime64[ns]
          ResponseType       object
          IP Address         object
          Progress           object
          Duration           int32
          Finished           object
          RecordedDate       object
          ResponseID         object
          RecipientLastName  object
          RecipientFirstName object
          RecipientEmail     object
          ExternalDataReference object
          LocationLatitude   object
          LocationLongitude  object
          DistributionChannel object
          UserLanguage       object
          Participation_in_this_project. object
          Browser Meta Info - Browser object
          Browser Meta Info - Version object
          Browser Meta Info - Operating System object
          Browser Meta Info - Resolution object
          What number did you hear? object
          What word did you see? object
          What is your sex? object
          How old are you? object
          Which of the following best describes your ethnicity? object
          Are you Spanish/Hispanic/Latino object
          What is the highest degree or level of school you have completed? object
          This is an attention check. Select 200. object
```

...

```

Timing - Last Click.5
                                object
Timing - Page Submit.5
                                object
Timing - Click Count.5
                                object
Timing - First Click.6
                                object
Timing - Last Click.6
                                object
Timing - Page Submit.6
                                object
Timing - Click Count.6
                                object
Timing - First Click.7
                                object
Timing - Last Click.7
                                object
Timing - Page Submit.7
                                object
Timing - Click Count.7
                                object
Identify the statement that correctly describes the facts of this case. (This is
the attention check)
                                object
Was_snowboard_sold_McNeil_defective_14
                                object
Is_substantial_factor_McNeil_injuries_14
                                float64
Non_economic_damages_McNeil_suffered_14
                                float64
Damages_words_14
                                object
Was_McNeil_negligent
                                float64
McNeil_negligence_substantial_factor_for_injuries
                                float64
Percentage_of_responsibility_X5
                                float64
Percentage_of_responsibility_McNeil
                                float64
Was_snowboard_sold_McNeil_defective_58
                                object
Is_substantial_factor_McNeil_injuries_58
                                float64
Economic_damages_McNeil_suffer_58
                                float64
Economic_Damages_In_Word_58
                                object
Non_economic_damages_McNeil_suffered_58
                                float64
Non_Economic_Damages_In_Word_58
                                object
Please explain why you arrived at your decision? (50 character minimum)
                                object
Q40
                                float64
Q41
                                object
Path
                                int32
Length: 84, dtype: object

```

*** Note Failed parsing df.Q41 =pd.to_numeric(df.Q41) Checked the data It has one invalid row '1,3'

```
In [255]: df.Q41.unique()
```

```
Out[255]: array(['', '3', '1', '"1,3"'], dtype=object)
```

Extracting the required columns and storing it in "newdf" data frame.


```

In [256]: newdf =pd.DataFrame(df[['StartDate', 'EndDate','Duration',
    'Was_snowboard_sold_McNeil_defective_14',
    "Is_substantial_factor_McNeil_injuries_14",
    'Non_economic_damages_McNeil_suffered_14',

    'Was_McNeil_negligent',
    'McNeil_negligence_substantial_factor_for_injuries',

    'Percentage_of_responsibility_X5',
    'Percentage_of_responsibility_McNeil'

    ,
    'Was_snowboard_sold_McNeil_defective_58',
    "Is_substantial_factor_McNeil_injuries_58",
    'Economic_damages_McNeil_suffer_58',
    'Non_economic_damages_McNeil_suffered_58',
    'Q40',
    'Q41',
    'Path',
    'What is the highest degree or level of school you have completed?',
    'Which of the following best describes your total household income?',
    ]])

newdf.sample(5)

```

Out[256]:

	StartDate	EndDate	Duration	Was_snowboard_sold_McNeil_defective_14	Is_substantial_fact
643	2018-04-06 13:31:00	2018-04-06 14:03:00	1925	4	5.0
118	2018-04-06 13:18:00	2018-04-06 13:38:00	1207	4	5.0
612	2018-04-06 13:20:00	2018-04-06 14:00:00	2407		NaN
715	2018-04-06 14:06:00	2018-04-06 14:27:00	1237	4	6.0
738	2018-04-06 14:14:00	2018-04-06 14:33:00	1109	6	NaN

```
In [257]: newdf.rename(columns=
{"What is the highest degree or level of school you have completed?": "Education",
"Which of the following best describes your total household income?": "Income",
},inplace=True)
```

Handling Percentage Calculation

We have two columns that save the percentage of responsibility for X5 and McNeil. The total sum should be 100. If it is less than 100 or greater than 100, then we need to change to a relative percentage, so that it should be round to 100.

Let's see what are the data in these columns and if there are any null/NaN values, then we have to deal with that.

```
In [258]: print("Unique values for _X5 ", newdf.Percentage_of_responsibility_X5.unique())
print("Unique values for _McNeil ", newdf.Percentage_of_responsibility_McNeil.unique())

Unique values for _X5 [ nan  50.  65.  75.  90.  80.  70.  25.  60.  40.  20.  15.
 5.  10.
 35.  85.  55. 100.  45.  30.]
Unique values for _McNeil [nan 50. 35. 25. 10. 20. 30. 75. 40. 60. 80. 85. 95. 90. 6
 5. 15. 45.  0.
 55. 70.]
```

In Both the columns, we have NaN values. Before replacing NaN with 0s, lets first check which rows have 0 values.

```
In [259]: newdf.query("Percentage_of_responsibility_McNeil == 0")
```

Out[259]:

	StartDate	EndDate	Duration	Was_snowboard_sold_McNeil_defective_14	Is_substantial_fact
510	2018-04-06 13:51:00	2018-04-06 13:53:00	136		NaN

As there is one row with 0 value, we are replacing NaN with some negative values say `-1'`

```
In [260]: newdf['Percentage_of_responsibility_X5'].fillna(-1,inplace=True)
newdf['Percentage_of_responsibility_McNeil'].fillna(-1,inplace=True)
newdf.Percentage_of_responsibility_McNeil.unique()
```

```
Out[260]: array([-1., 50., 35., 25., 10., 20., 30., 75., 40., 60., 80., 85., 95.,
 90., 65., 15., 45.,  0., 55., 70.])
```

Lets see the distribution of total percentage.

```
In [261]: newdf['Total_perc'] = (newdf['Percentage_of_responsibility_X5']
+newdf['Percentage_of_responsibility_McNeil'])

newdf.query('Total_perc < 100 & Total_perc > 0 |Total_perc > 100')
```

Out[261]:

	StartDate	EndDate	Duration	Was_snowboard_sold_McNeil_defective_14	Is_substantial_factor_
--	-----------	---------	----------	--	------------------------

So for all cases each percentage are summing to 100 and there is no "Total Percentage" greater than or less than 100.

Lets convert the value to % for calculation of discounted damages and replacinng -ve value with 1.

```
In [262]: newdf['Percentage_of_responsibility_X5']=newdf['Percentage_of_responsibility_X5']/100
newdf['Percentage_of_responsibility_X5'].replace([-0.01],[1], inplace = True)
newdf['Percentage_of_responsibility_McNeil']=newdf['Percentage_of_responsibility_McNeil']/100
newdf['Percentage_of_responsibility_McNeil'].replace([-0.01],[1], inplace = True)

print(newdf['Percentage_of_responsibility_McNeil'].head())
print(newdf['Percentage_of_responsibility_X5'].head())
```

```
0    1.0
1    1.0
2    1.0
3    1.0
4    1.0
Name: Percentage_of_responsibility_McNeil, dtype: float64
0    1.0
1    1.0
2    1.0
3    1.0
4    1.0
Name: Percentage_of_responsibility_X5, dtype: float64
```

See how many missing data points we have

Ok, now we know that we do have some missing values. Let's see how many we have in each column.

```
In [263]: import numpy as np
missing_values_count = newdf.isnull().sum()
print(missing_values_count)
total_cells = np.product(newdf.shape)
total_missing = missing_values_count.sum()
print('Percent of data that is missing: ',(total_missing/total_cells) * 100)
```

```
StartDate          0
EndDate            0
Duration           0
Was_snowboard_sold_McNeil_defective_14      0
Is_substantial_factor_McNeil_injuries_14    618
Non_economic_damages_McNeil_suffered_14     637
Was_McNeil_negligent                        489
McNeil_negligence_substantial_factor_for_injuries 489
Percentage_of_responsibility_X5             0
Percentage_of_responsibility_McNeil         0
Was_snowboard_sold_McNeil_defective_58      0
Is_substantial_factor_McNeil_injuries_58    629
Economic_damages_McNeil_suffer_58          656
Non_economic_damages_McNeil_suffered_58     656
Q40                                         33
Q41                                         0
Path                                       0
Education                                  0
Income                                     0
Total_perc                                 0
dtype: int64
Percent of data that is missing: 26.162935323383085
```

```
In [264]: newdf.shape
```

```
Out[264]: (804, 20)
```

```
In [265]: newdf.dtypes
```

```
Out[265]: StartDate          datetime64[ns]
EndDate            datetime64[ns]
Duration           int32
Was_snowboard_sold_McNeil_defective_14      object
Is_substantial_factor_McNeil_injuries_14    float64
Non_economic_damages_McNeil_suffered_14     float64
Was_McNeil_negligent                        float64
McNeil_negligence_substantial_factor_for_injuries float64
Percentage_of_responsibility_X5             float64
Percentage_of_responsibility_McNeil         float64
Was_snowboard_sold_McNeil_defective_58      object
Is_substantial_factor_McNeil_injuries_58    float64
Economic_damages_McNeil_suffer_58          float64
Non_economic_damages_McNeil_suffered_58     float64
Q40                                         float64
Q41                                         object
Path                                       int32
Education                                  object
Income                                     object
Total_perc                                 float64
dtype: object
```

As we are just working on from path 1 to 8, Lets remove path with value 0.

So the number of rows to be removed having Path as 0 can be checked usig the ".shape"

```
In [266]: newdf[newdf.Path <=0].shape
```

```
Out[266]: (13, 20)
```

So we have 13 rows . Let's have a look on those rows.

```
In [267]: newdf[newdf.Path <=0].head()
```

```
Out[267]:
```

	StartDate	EndDate	Duration	Was_snowboard_sold_McNeil_defective_14	Is_substantial_factor
0	2018-04-06 13:14:00	2018-04-06 13:15:00	34		NaN
1	2018-04-06 13:15:00	2018-04-06 13:16:00	42		NaN
3	2018-04-06 13:17:00	2018-04-06 13:18:00	69		NaN
5	2018-04-06 13:19:00	2018-04-06 13:19:00	14		NaN
6	2018-04-06 13:17:00	2018-04-06 13:20:00	197		NaN

 As we can see there are 13 observation with path value equal to 0. We are removing these observation

```
In [268]: newdf = newdf[newdf.Path > 0]
```

Filter the data based on the duration of each scenario

As per the requirement we have to filter the experiment one data based on the length of each scenario. The lengths are as follows:

- Scenario 1 14:47 (887 seconds)
- Scenario 2 15:11 (911 seconds)
- Scenario 3 15:50 (950 seconds)
- Scenario 4 16:23 (983 seconds)
- Scenario 5 16:37 (997 seconds)
- Scenario 6 16:56 (1016 seconds)
- Scenario 7 17:40 (1060 seconds)
- Scenario 8 18:04 (1084 seconds)

Let's filter anyone who spent less than 10 second less than the whole time. e.g 14:37(877), 15:01(901) etc.

```
In [ ]: newdf=newdf[((newdf.Path == 1) & (df.Duration>=877))|((newdf.Path == 2)
& (df.Duration>=901))|((newdf.Path == 4)
& (df.Duration>=973))|((newdf.Path == 3)
& (df.Duration>=940)) | ((newdf.Path == 5)
& (df.Duration>=987)) | ((newdf.Path == 6)
& (df.Duration>=1006))|((newdf.Path == 7)
& (df.Duration>=1050)) | ((newdf.Path == 8)
& (df.Duration>=1074))]
```

```
In [270]: newdf.head()
```

Out[270]:

	StartDate	EndDate	Duration	Was_snowboard_sold_McNeil_defective_14	Is_substantial_facto
25	2018-04-06 13:15:00	2018-04-06 13:32:00	1039		NaN
26	2018-04-06 13:17:00	2018-04-06 13:33:00	915	4	6.0
27	2018-04-06 13:15:00	2018-04-06 13:33:00	1051	4	5.0
28	2018-04-06 13:15:00	2018-04-06 13:33:00	1092	6	NaN
29	2018-04-06 13:14:00	2018-04-06 13:33:00	1135	4	5.0

Replacing the Null Values with empty string(Easy to convert to other datatypes Later)

```
In [271]: print(pd.isnull(newdf).any())
newdf = newdf[np.isfinite(newdf['Path'])]
newdf['Is_substantial_factor_McNeil_injuries_14'].fillna("",inplace=True)
newdf['Non_economic_damages_McNeil_suffered_14'].fillna("",inplace=True)
newdf['Was_McNeil_negligent'].fillna("",inplace=True)
newdf['McNeil_negligence_substantial_factor_for_injuries'].fillna("",inplace=True)
newdf['Percentage_of_responsibility_X5'].fillna("",inplace=True)
newdf['Percentage_of_responsibility_McNeil'].fillna("",inplace=True)
newdf['Was_snowboard_sold_McNeil_defective_58'].fillna("",inplace=True)
newdf['Is_substantial_factor_McNeil_injuries_58'].fillna("",inplace=True)
newdf['Economic_damages_McNeil_suffer_58'].fillna("",inplace=True)
newdf['Non_economic_damages_McNeil_suffered_58'].fillna("",inplace=True)
newdf['Q40'].fillna("",inplace=True)
newdf['Q41'].fillna("",inplace=True)
# Printing the first 5 lines.
#print(newdf.head(5))
```

StartDate	False
EndDate	False
Duration	False
Was_snowboard_sold_McNeil_defective_14	False
Is_substantial_factor_McNeil_injuries_14	True
Non_economic_damages_McNeil_suffered_14	True
Was_McNeil_negligent	True
McNeil_negligence_substantial_factor_for_injuries	True
Percentage_of_responsibility_X5	False
Percentage_of_responsibility_McNeil	False
Was_snowboard_sold_McNeil_defective_58	False
Is_substantial_factor_McNeil_injuries_58	True
Economic_damages_McNeil_suffer_58	True
Non_economic_damages_McNeil_suffered_58	True
Q40	True
Q41	False
Path	False
Education	False
Income	False
Total_perc	False
dtype: bool	

Graph showing the responses of jurors for each path(1-8)

There are two separate columns in our dataset having the juror response.

- Was_snowboard_sold_McNeil_defective_14 : keeping response from path 1 to 4
- Was_snowboard_sold_McNeil_defective_58 : keeping response from path 5 to 8

For plotting a single graph for all the path, we merge these two columns into a new column called **"Liability"**.

```
In [272]: newdf['Liability'] =(newdf['Was_snowboard_sold_McNeil_defective_14']
+ newdf['Was_snowboard_sold_McNeil_defective_58'])
```

Liability columns have numeric values. We have replaced it with 4 for 'Yes' and 6 for 'No'. Liability with blank is dropped.

```
In [273]: newdf['Liability'].replace('', np.nan, inplace=True)
newdf.dropna(subset=['Liability'], inplace=True)
newdf['Liability'].replace(['4', '6'], ['Yes', 'No'], inplace = True)

newdf.Liability.unique()
```

```
Out[273]: array(['No', 'Yes'], dtype=object)
```

Total Damage Calculation

For the box plot, we need to replace the empty string with 0. But before Filling the NaN values with 0 , lets first check if any juror has put 0 intentionally.

We need to change the data type of damages. There are 3 different columns that have the damages information. From previous data type check, we found that there are so many missing values for damages. So we replaced them with 0.

For simplicity to plot Path vs damages we combined all damages into one column and named it as "Total_Damages".

```
In [274]: newdf.Economic_damages_McNeil_suffer_58= pd.to_numeric(newdf.Economic_damages_McNeil_suffer_58)
newdf.Non_economic_damages_McNeil_suffered_58 = pd.to_numeric(newdf.Non_economic_damages_McNeil_suffered_58)
newdf.Non_economic_damages_McNeil_suffered_14 = pd.to_numeric(newdf.Non_economic_damages_McNeil_suffered_14)
```

```
In [275]: newdf.query('Non_economic_damages_McNeil_suffered_14 == 0 | Non_economic_damages_McNeil_suffered_58 == 0 | Economic_damages_McNeil_suffer_58 ==0')
```

```
Out[275]:
```

	StartDate	EndDate	Duration	Was_snowboard_sold_McNeil_defective_14	Is_substantial_fact
246	2018-04-06 13:19:00	2018-04-06 13:42:00	1404		

1 rows × 21 columns

We found that one row has 0 value for Non_economic damages McNeil suffered.

```
In [276]: newdf.Economic_damages_McNeil_suffer_58.fillna(0, inplace = True)
newdf.Non_economic_damages_McNeil_suffered_58.fillna(0, inplace = True)
newdf.Non_economic_damages_McNeil_suffered_14.fillna(0, inplace = True)

newdf['Total_Damages'] = (newdf['Economic_damages_McNeil_suffer_58']
                        + newdf['Non_economic_damages_McNeil_suffered_58']
                        + newdf['Non_economic_damages_McNeil_suffered_14'])
```


Lets see how many rows have Total Damage as 0.

```
In [277]: (newdf.Total_Damages==0).sum()
```

```
Out[277]: 435
```

As per requirement Damages above 500000 should be converted to 500000.

```
In [278]: newdf['Total_Damages'].replace([1000000], [500000], inplace = True)
```

```
In [279]: newdf.query("Total_Damages > 500000")
```

```
Out[279]:
```

	StartDate	EndDate	Duration	Was_snowboard_sold_McNeil_defective_14	Is_substantial_factor_
--	-----------	---------	----------	--	------------------------

0 rows × 22 columns

As per user's requirement we need to discount the percentage X5 responsible from the Total Damages.

```
In [280]: newdf["Discounted_Damages"] = (newdf["Total_Damages"]
                                         * newdf["Percentage_of_responsibility_X5"])
```

```
In [281]: newdf.Liability.unique()
```

```
Out[281]: array(['No', 'Yes'], dtype=object)
```

```
In [282]: newdf.shape
```

```
Out[282]: (729, 23)
```

```
In [283]: newdf.Liability.unique()
```

```
Out[283]: array(['No', 'Yes'], dtype=object)
```

As we need to check if juror education and Income has any impact while awarding liability, we included there two fields in the data frame.

```
In [284]: newdf.Education=newdf.Education.astype(int)
newdf.Education.unique()
newdf.Income=newdf.Income.astype(int)
newdf.Income.unique()
```

```
Out[284]: array([2, 3, 5, 4, 1, 6], dtype=int64)
```

Saving the file to CSV so that we can use in the other notebook.

```
In [285]: newdf.to_csv("cleaning.csv", sep=',')
```