Initial Setup

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib
import seaborn as sns
color = sns.color_palette
import plotly.graph_objs as go
import plotly.offline as py
from plotly import tools
```

In [2]:

```
%matplotlib inline

import warnings
warnings.filterwarnings('ignore')
```

Importing the dataset

In [3]:

```
heroes = pd.read_csv("dataset/superhero-set/heroes_information.csv")
heroes.head(10)
```

Out[3]:

	Unnamed: 0	name	Gender	Eye color	Race	Hair color	Height	Publisher	Skin color	Alignn
0	0	A-Bomb	Male	yellow	Human	No Hair	203.0	Marvel Comics	-	g
1	1	Abe Sapien	Male	blue	Icthyo Sapien	No Hair	191.0	Dark Horse Comics	blue	g
2	2	Abin Sur	Male	blue	Ungaran	No Hair	185.0	DC Comics	red	g
3	3	Abomination	Male	green	Human / Radiation	No Hair	203.0	Marvel Comics	-	
4	4	Abraxas	Male	blue	Cosmic Entity	Black	-99.0	Marvel Comics	-	
5	5	Absorbing Man	Male	blue	Human	No Hair	193.0	Marvel Comics	-	
6	6	Adam Monroe	Male	blue	-	Blond	-99.0	NBC - Heroes	-	g
7	7	Adam Strange	Male	blue	Human	Blond	185.0	DC Comics	-	g
8	8	Agent 13	Female	blue	-	Blond	173.0	Marvel Comics	-	g
9	9	Agent Bob	Male	brown	Human	Brown	178.0	Marvel Comics	-	g

In [4]:

```
heroes.loc[heroes['Skin color'] != '-', ['name','Gender', 'Height']].head() # df.loc[row_idices_criteria, columns]
```

Out[4]:

	name	Gender	Height
1	Abe Sapien	Male	191.0
2	Abin Sur	Male	185.0
17	Alien	Male	244.0
34	Apocalypse	Male	213.0
39	Archangel	Male	183.0

In [5]:

```
heroes.iloc[3:18, [1,2,6]].head() # specify row/column indices
```

Out[5]:

	name	Gender	Height
3	Abomination	Male	203.0
4	Abraxas	Male	-99.0
5	Absorbing Man	Male	193.0
6	Adam Monroe	Male	-99.0
7	Adam Strange	Male	185.0

In [6]:

```
heroes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 734 entries, 0 to 733
Data columns (total 11 columns):
Unnamed: 0    734 non-null int64
name          734 non-null object
Gender        734 non-null object
Eye color     734 non-null object
Race          734 non-null object
Hair color    734 non-null object
Height        734 non-null float64
Publisher     719 non-null object
Skin color    734 non-null object
Alignment     734 non-null object
Weight        732 non-null float64
dtypes: float64(2), int64(1), object(8)
memory usage: 63.2+ KB
```

In [7]:

```
heroes.loc[heroes['Weight'].isna()]
```

Out[7]:

	Unnamed: 0	name	Gender	Eye color	Race	Hair color	Height	Publisher	Skin color	Alignment
286	286	Godzilla	-	-	Kaiju	-	108.0	NaN	grey	bad
389	389	King Kong	Male	yellow	Animal	Black	30.5	NaN	-	good

In [8]:

```
print("missing value count in Publisher: ", heroes['Publisher'].isnull().sum())
print("missing value count in Weight: ", heroes['Weight'].isnull().sum())


print("missing value count in Skin color: ", len(heroes.loc[heroes['Skin color']=='-'
]))
print("missing value count in Hair color: ", len(heroes.loc[heroes['Hair color']=='-'
]))
print("missing value count in Eye color: ", len(heroes.loc[heroes['Eye color']=='-']))
print("missing value count in Gender: ", len(heroes.loc[heroes['Gender']=='-']))
```

```
missing value count in Publisher:  15
missing value count in Weight:  2
missing value count in Skin color:  662
missing value count in Hair color:  172
missing value count in Eye color:  172
missing value count in Gender:  29
```

In [9]:

```
heroes.columns
```

Out[9]:

```
Index(['Unnamed: 0', 'name', 'Gender', 'Eye color', 'Race', 'Hair color',
       'Height', 'Publisher', 'Skin color', 'Alignment', 'Weight'],
      dtype='object')
```

In [10]:

```
list(heroes.Gender.unique())
```

Out[10]:

```
['Male', 'Female', '-']
```

In [11]:

```
def print_missing_values(df):

    for column in df.columns:
        if df[column].isna().any():
            print(column, " has ", df[column].isna().sum(), " NaN values")

        if '-' in list(df[column].unique()):
            print(column, " has ", len(df.loc[df[column]=='-']), " values represented a
s -")


print_missing_values(heroes)
```

```
Gender  has  29  values represented as -
Eye color  has  172  values represented as -
Race  has  304  values represented as -
Hair color  has  172  values represented as -
Publisher  has  15  NaN values
Skin color  has  662  values represented as -
Alignment  has  7  values represented as -
Weight  has  2  NaN values
```

In [12]:

```
heroes.drop(['Unnamed: 0'], axis=1)
heroes.columns
```

Out[12]:

```
Index(['Unnamed: 0', 'name', 'Gender', 'Eye color', 'Race', 'Hair color',
       'Height', 'Publisher', 'Skin color', 'Alignment', 'Weight'],
      dtype='object')
```

In [13]:

```
# dropping off unnecssary column
heroes.drop(['Unnamed: 0'], axis=1, inplace=True)
heroes.columns
```

Out[13]:

```
Index(['name', 'Gender', 'Eye color', 'Race', 'Hair color', 'Height',
       'Publisher', 'Skin color', 'Alignment', 'Weight'],
      dtype='object')
```

In [14]:

```
heroes.Gender.unique()
```

Out[14]:

```
array(['Male', 'Female', '-'], dtype=object)
```

In [15]:

```
heroes.Gender.value_counts()
```

Out[15]:

```
Male      505
Female    200
-          29
Name: Gender, dtype: int64
```

In [16]:

```
heroes.Weight.value_counts()
```

Out[16]:

```
-99.0     237
 79.0      23
 54.0      23
 81.0      22
 90.0      19
         ...
 132.0      1
 38.0       1
 412.0      1
 320.0      1
 855.0      1
Name: Weight, Length: 135, dtype: int64
```

In [17]:

```
# replacing ALL negative values with NaN
heroes.replace(-99.0, np.nan, inplace=True)
```

In [18]:

```
heroes.Height.isna().sum()
```

Out[18]:

```
217
```

In [19]:

```
heroes.Weight.isna().sum()
```

Out[19]:

```
239
```

In [20]:

```
heroes.head(2)
```

Out[20]:

	name	Gender	Eye color	Race	Hair color	Height	Publisher	Skin color	Alignment	Weight
0	A-Bomb	Male	yellow	Human	No Hair	203.0	Marvel Comics	-	good	441.0
1	Abe Sapien	Male	blue	Icthyo Sapien	No Hair	191.0	Dark Horse Comics	blue	good	65.0

In [21]:

```
ht_wt = heroes[['Height', 'Weight']]

ht_wt.head(3)
```

Out[21]:

	Height	Weight
0	203.0	441.0
1	191.0	65.0
2	185.0	90.0

In [24]:

```
# imputing missing values with median
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy="median")
X = imputer.fit_transform(ht_wt)
```

In [25]:

```
type(X)
```

Out[25]:

```
numpy.ndarray
```

In [26]:

```
heroes_h_w = pd.DataFrame(X, columns=ht_wt.columns)
heroes_h_w.head(2)
```

Out[26]:

	Height	Weight
0	203.0	441.0
1	191.0	65.0

In [27]:

```
heroes_without_h_w = heroes.drop(['Height', 'Weight'], axis=1)

heroes = pd.concat([heroes_without_h_w, heroes_h_w], axis=1)

heroes.head(10)
```

Out[27]:

	name	Gender	Eye color	Race	Hair color	Publisher	Skin color	Alignment	Height	Weigh
0	A-Bomb	Male	yellow	Human	No Hair	Marvel Comics	-	good	203.0	441.(
1	Abe Sapien	Male	blue	Icthyo Sapien	No Hair	Dark Horse Comics	blue	good	191.0	65.(
2	Abin Sur	Male	blue	Ungaran	No Hair	DC Comics	red	good	185.0	90.(
3	Abomination	Male	green	Human / Radiation	No Hair	Marvel Comics	-	bad	203.0	441.(
4	Abraxas	Male	blue	Cosmic Entity	Black	Marvel Comics	-	bad	183.0	81.(
5	Absorbing Man	Male	blue	Human	No Hair	Marvel Comics	-	bad	193.0	122.(
6	Adam Monroe	Male	blue	-	Blond	NBC - Heroes	-	good	183.0	81.(
7	Adam Strange	Male	blue	Human	Blond	DC Comics	-	good	185.0	88.(
8	Agent 13	Female	blue	-	Blond	Marvel Comics	-	good	173.0	61.(
9	Agent Bob	Male	brown	Human	Brown	Marvel Comics	-	good	178.0	81.(

Some Insights

In [28]:

```
# seeing the distribution of super heroes from each of the Publishers
```

In [29]:

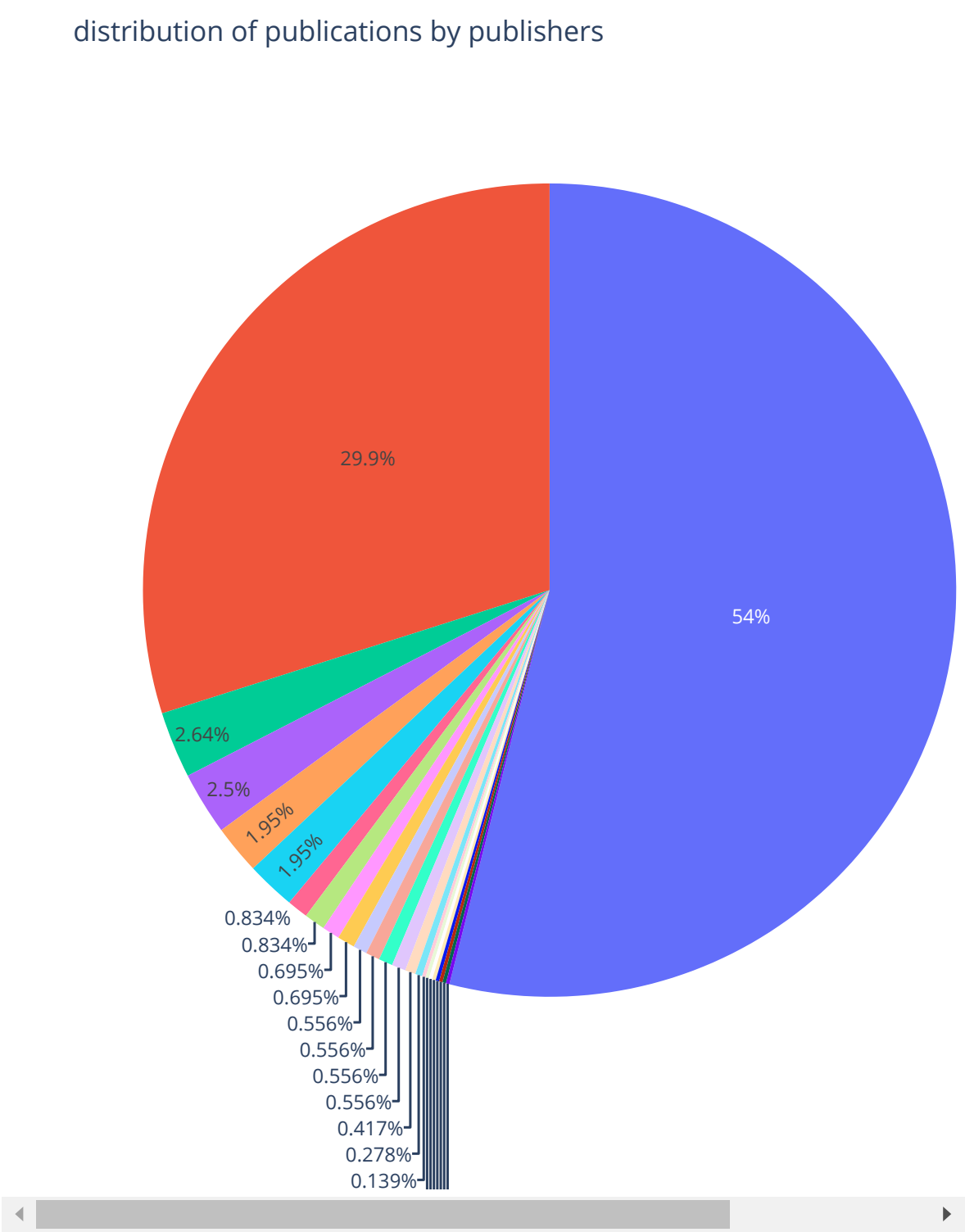
```
publisher_series = heroes.Publisher.value_counts()
```

In [30]:

```
publishers = list(publisher_series.index)
publications = list((publisher_series/publisher_series.sum())*100)
```

In [31]:

```
trace = go.Pie(labels=publishers, values=publications)

layout = go.Layout(title='distribution of publications by publishers', height=750, widt
h=750)

data = [trace]
fig = go.Figure(data=data, layout=layout)
py.iplot(fig, filename='comic-wise-distribution-by-publishers')
```



In [32]:

```
heroes.Alignment.value_counts()
```

Out[32]:

```
good       496
bad        207
neutral     24
-            7
Name: Alignment, dtype: int64
```

In [33]:

```
heroes.loc[heroes['Alignment']=='-', 'Alignment'] = 'unknown'
```

In [36]:

```
heroes.loc[heroes['name']=='Hulk']
```

Out[36]:

	name	Gender	Eye color	Race	Hair color	Publisher	Skin color	Alignment	Height	Weight
331	Hulk	Male	green	Human / Radiation	Green	Marvel Comics	green	good	244.0	630.0

In [37]:

```
# No. of Super heros vs no. of super villan

tot_pub = (heroes.Publisher.value_counts().index)
col_names = ['Publisher', 'total_heroes', 'total_villians', 'total_neutral', 'total_unk
nown']

df = pd.DataFrame(columns=col_names)

for publisher in tot_pub:
    data = []
    # adding the publisher names into list
    data.append(publisher)
    # slicing up the df for heroes from this publisher,
    # converting to list, taking length -> total count of super heros from this publish
er
    data.append(len(list(heroes.loc[(heroes['Alignment']=='good') & (heroes['Publisher'
]==publisher),'name'])))
    # slicing up the df for villians from this publisher,
    # converting to list, taking length -> total count of villians from this publisher
    data.append(len(list(heroes.loc[(heroes['Alignment']=='bad') & (heroes['Publisher']
==publisher),'name'])))
    # slicing up the df for neutral characters from this publisher,
    # converting to list, taking length -> total count of neutral characters from this
 publisher
    data.append(len(list(heroes.loc[(heroes['Alignment']=='neutral') & (heroes['Publish
er']==publisher),'name'])))
    # slicing up the df for unknowns from this publisher,
    # converting to list, taking length -> total count of unknown characters from this
 publisher
    data.append(len(list(heroes.loc[(heroes['Alignment']=='unknown') & (heroes['Publish
er']==publisher),'name'])))

    # append everything to as individual row into df
    df.loc[len(df)] = data
```

In [38]:

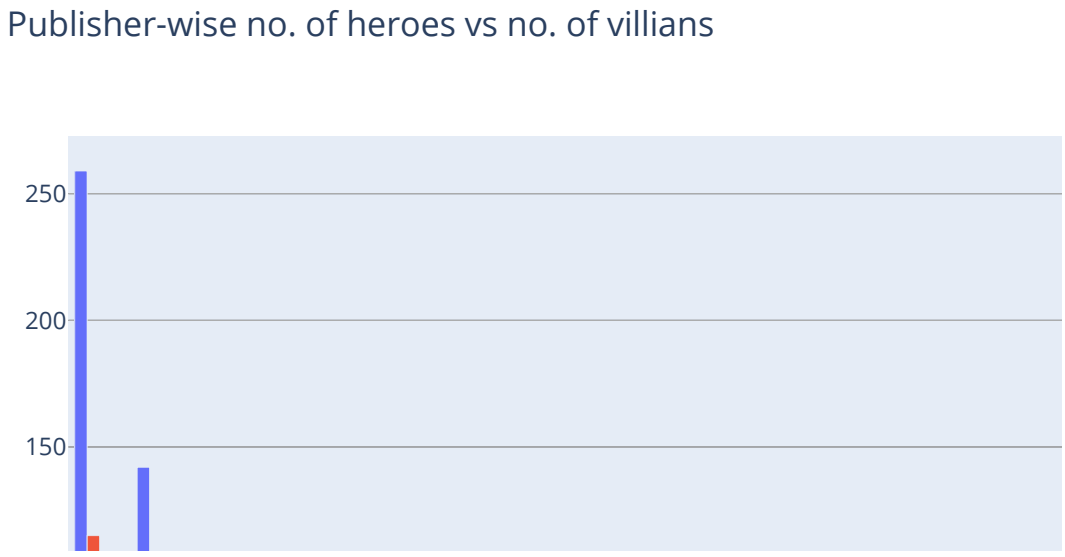
```
df.head(10)
```

Out[38]:

	Publisher	total_heroes	total_villians	total_neutral	total_unknown
0	Marvel Comics	259	115	11	3
1	DC Comics	142	59	13	1
2	NBC - Heroes	16	3	0	0
3	Dark Horse Comics	12	6	0	0
4	Image Comics	2	11	0	1
5	George Lucas	8	6	0	0
6	Star Trek	5	0	0	1
7	HarperCollins	6	0	0	0
8	Team Epic TV	4	1	0	0
9	SyFy	5	0	0	0

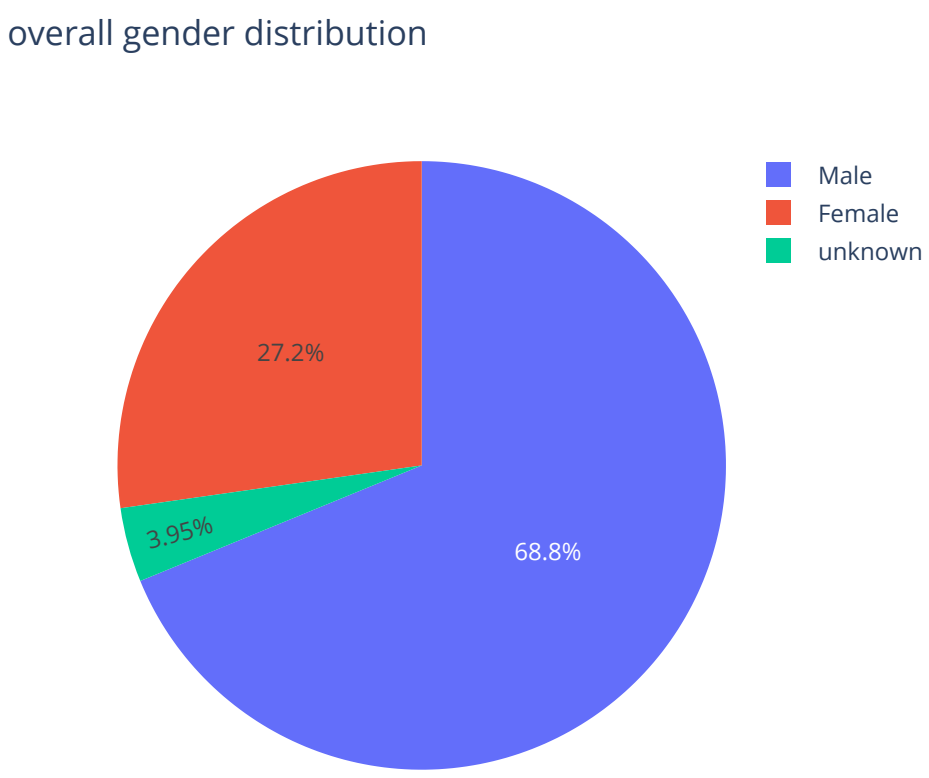
In [39]:

```
trace1 = go.Bar(
    x=list(df.Publisher),
    y=list(df.total_heroes),
    name='total_heroes')

trace2 = go.Bar(
    x=list(df.Publisher),
    y=list(df.total_villians),
    name='total_villians')

trace3 = go.Bar(
    x=list(df.Publisher),
    y=list(df.total_neutral),
    name='total_neutral')

trace4 = go.Bar(
    x=list(df.Publisher),
    y=list(df.total_unknown),
    name='total_unknown')

data = [trace1, trace2, trace3, trace4]
layout = go.Layout(title='Publisher-wise no. of heroes vs no. of villians', barmode='gr
oup')

fig = go.Figure(data=data, layout=layout)
py.iplot(fig, filename='heroes-vs-villians by Publishers')
```



In [40]:

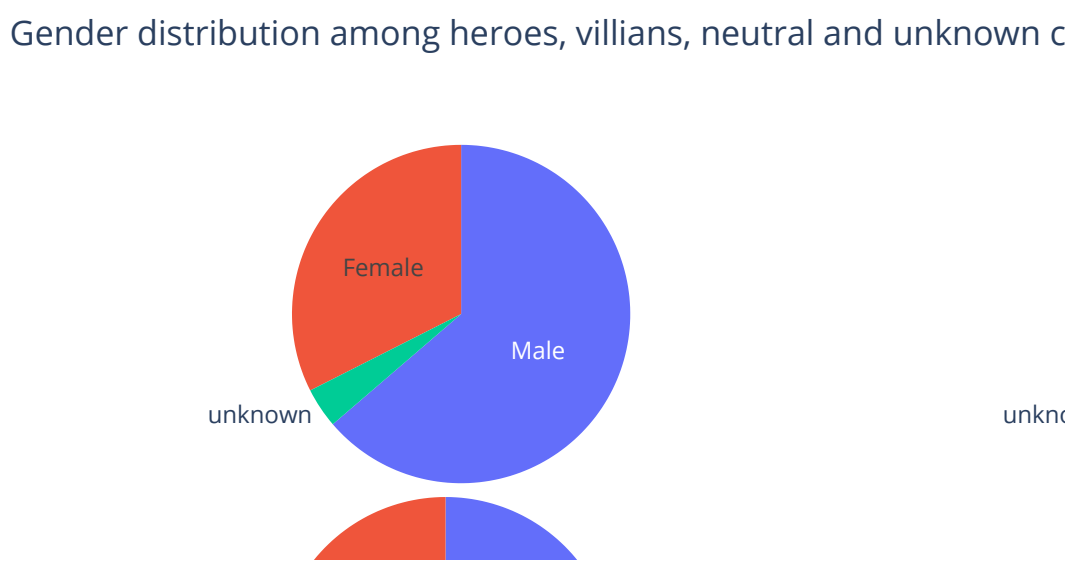
```
# replace dash by unknown
heroes.replace('-', 'unknown', inplace=True)

# Gender Distribution
gender_series = heroes['Gender'].value_counts()
genders = list(gender_series.index)
distribution = list((gender_series/gender_series.sum())*100)

trace = go.Pie(labels=genders, values=distribution)
layout = go.Layout(
    title='overall gender distribution',
    height=500,
    width=500
)

data = [trace]
fig = go.Figure(data=data, layout=layout)
py.iplot(fig, filename='gender-distribution')
```



In [41]:

```
# gender distribution by good alignment
heroes_gender_series = heroes.loc[heroes['Alignment']=='good', 'Gender'].value_counts()
heroes_gender = list(heroes_gender_series.index)
heroes_distribution = list((heroes_gender_series/heroes_gender_series.sum())*100)

# gender distribution by bad alignment
villian_gender_series = heroes.loc[heroes['Alignment']=='bad', 'Gender'].value_counts()
villian_gender = list(heroes_gender_series.index)
villian_distribution = list((heroes_gender_series/heroes_gender_series.sum())*100)

# gender distribution by neutral alignment
neutral_gender_series = heroes.loc[heroes['Alignment']=='neutral', 'Gender'].value_coun
ts()
neutral_gender = list(heroes_gender_series.index)
neutral_distribution = list((heroes_gender_series/heroes_gender_series.sum())*100)

# gender distribution by unknown alignment
unknown_gender_series = heroes.loc[heroes['Alignment']=='unknown', 'Gender'].value_coun
ts()
unknown_gender = list(heroes_gender_series.index)
unknown_distribution = list((heroes_gender_series/heroes_gender_series.sum())*100)

fig = {
    "data": [
        {
            "labels": heroes_gender,
            "values": heroes_distribution,
            "type": "pie",
            "name": "heroes",
            "domain": {'x': [0, 0.5],
                       'y': [0.51, 1]
                      },
            "textinfo": "label"
        },
        {
            "labels": villian_gender,
            "values": villian_distribution,
            "type": "pie",
            "name": "villians",
            "domain": {'x': [0.52, 1],
                       'y': [0.51, 1]
                      },
            "textinfo": "label"
        },
        {
            "labels": neutral_gender,
            "values": neutral_distribution,
            "type": "pie",
            "name": "neutral",
            "domain": {'x': [0, 0.48],
                       'y': [0, 0.49]
                      },
            "textinfo": "label"
        },
        {
            "labels": unknown_gender,
            "values": unknown_distribution,
            "type": "pie",
            "name": "unknown",
```

```
            "domain": {'x': [0.52, 1],
                       'y': [0, 0.49]
                      },
            "textinfo": "label"
        }
    ],
    "layout": {"title": "Gender distribution among heroes, villians, neutral and unknow
n characters", "showlegend": True}
}

py.iplot(fig, filename="gender distribution")
```

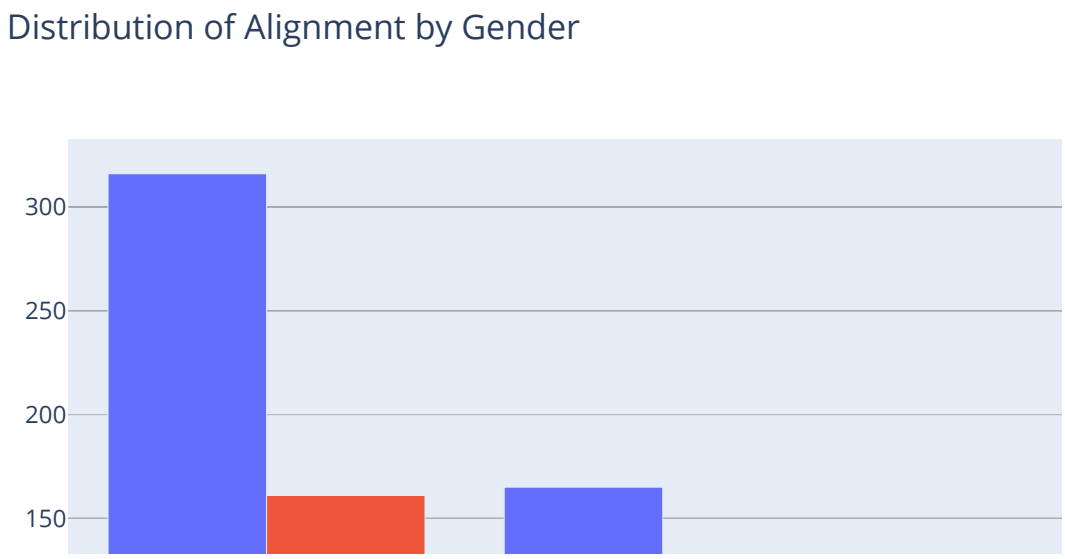


In [42]:

```
# alignment of characters by alignment
male_df = heroes.loc[heroes['Gender']=='Male']
female_df = heroes.loc[heroes['Gender']=='Female']
```

In [43]:

```
trace_m = go.Bar(
    x = male_df['Alignment'].value_counts().index,
    y = male_df['Alignment'].value_counts().values,
    name="male"
)

trace_f = go.Bar(
    x = female_df['Alignment'].value_counts().index,
    y = female_df['Alignment'].value_counts().values,
    name="female"
)

data = [trace_m, trace_f]
layout = go.Layout(
    title="Distribution of Alignment by Gender",
    barmode="group"
)

fig = go.Figure(data=data, layout=layout)
py.iplot(fig, filename="alignment by gender")
```



In [47]:

```
heroes.head(2)
```

Out[47]:

	name	Gender	Eye color	Race	Hair color	Publisher	Skin color	Alignment	Height	Weight
0	A-Bomb	Male	yellow	Human	No Hair	Marvel Comics	unknown	good	203.0	441.0
1	Abe Sapien	Male	blue	Icthyo Sapien	No Hair	Dark Horse Comics	blue	good	191.0	65.0

In [48]:

```
# but lets also plot the Professor Xavier like charater

heroes['Hair color'].unique()
```

Out[48]:

```
array(['No Hair', 'Black', 'Blond', 'Brown', 'unknown', 'White', 'Purple',
       'Orange', 'Pink', 'Red', 'Auburn', 'Strawberry Blond', 'black',
       'Blue', 'Green', 'Magenta', 'Brown / Black', 'Brown / White',
       'blond', 'Silver', 'Red / Grey', 'Grey', 'Orange / White',
       'Yellow', 'Brownn', 'Gold', 'Red / Orange', 'Indigo',
       'Red / White', 'Black / Blue'], dtype=object)
```

In [49]:

```
heroes['bald_or_not'] = heroes['Hair color'].where(heroes['Hair color']=='No Hair', oth
er='Hair')
```

In [50]:

```
heroes['bald_or_not'].value_counts()
```

Out[50]:

```
Hair       659
No Hair     75
Name: bald_or_not, dtype: int64
```

In [51]:

```
trace = go.Bar(
    x = heroes.bald_or_not.value_counts().index,
    y = heroes.bald_or_not.value_counts().values,
    name = 'bald or not',
    text = ['not-bald','bald']
)

layout = go.Layout(
    title = 'bald or not'
)

fig = go.Figure(data=[trace], layout=layout)
py.iplot(fig, filename='distribution by baldness')
```

In [58]:

```
heroes.loc[heroes['bald_or_not']=='No Hair', ['name', 'bald_or_not']].head()
```

Out[58]:

	name	bald_or_not
0	A-Bomb	No Hair
1	Abe Sapien	No Hair
2	Abin Sur	No Hair
3	Abomination	No Hair
5	Absorbing Man	No Hair

In [59]:

```
heroes.Race.unique()
```

Out[59]:

```
array(['Human', 'Icthyo Sapien', 'Ungaran', 'Human / Radiation',
       'Cosmic Entity', 'unknown', 'Cyborg', 'Xenomorph XX121', 'Android',
       'Vampire', 'Mutant', 'God / Eternal', 'Symbiote', 'Atlantean',
       'Alien', 'Neyaphem', 'New God', 'Alpha', 'Bizarro', 'Inhuman',
       'Metahuman', 'Demon', 'Human / Clone', 'Human-Kree',
       'Dathomirian Zabrak', 'Amazon', 'Human / Cosmic',
       'Human / Altered', 'Kryptonian', 'Kakarantharaian',
       'Zen-Whoberian', 'Strontian', 'Kaiju', 'Saiyan', 'Gorilla',
       'Rodian', 'Flora Colossus', 'Human-Vuldarian', 'Asgardian',
       'Demi-God', 'Eternal', 'Gungan', 'Bolovaxian', 'Animal',
       'Czarnian', 'Martian', 'Spartoi', 'Planet', 'Luphomoid',
       'Parademon', 'Yautja', 'Maiar', 'Clone', 'Talokite', 'Korugaran',
       'Zombie', 'Human-Vulcan', 'Human-Spartoi', 'Tamaranean',
       'Frost Giant', 'Mutant / Clone', "Yoda's species"], dtype=object)
```

loading the powers dataset

In [65]:

```
powers = pd.read_csv("dataset/superhero-set/super_hero_powers.csv")
powers.head(2)
```

Out[65]:

	hero_names	Agility	Accelerated Healing	Lantern Power Ring	Dimensional Awareness	Cold Resistance	Durability	Stealth	A
0	3-D Man	True	False	False	False	False	False	False	
1	A-Bomb	False	True	False	False	False	True	False	

2 rows × 168 columns

In [66]:

```
powers = powers * 1
powers.head()
```

Out[66]:

	hero_names	Agility	Accelerated Healing	Lantern Power Ring	Dimensional Awareness	Cold Resistance	Durability	Stealth	A
0	3-D Man	1	0	0	0	0	0	0	
1	A-Bomb	0	1	0	0	0	1	0	
2	Abe Sapien	1	1	0	0	1	1	0	
3	Abin Sur	0	0	1	0	0	0	0	
4	Abomination	0	1	0	0	0	0	0	

5 rows × 168 columns

In [76]:

```
powers.loc[:, 'total_powers'] = powers.iloc[:, 1:].sum(axis=1)
powers.head(2)
```

Out[76]:

	hero_names	Agility	Accelerated Healing	Lantern Power Ring	Dimensional Awareness	Cold Resistance	Durability	Stealth	A
0	3-D Man	1	0	0	0	0	0	0	
1	A-Bomb	0	1	0	0	0	1	0	

2 rows × 169 columns

In [82]:

```
# top 10 most powerful characters

powers[['hero_names', 'total_powers']].sort_values('total_powers', ascending=False).hea
d(10).reset_index().drop('index', axis=1)
```

Out[82]:

	hero_names	total_powers
0	Spectre	49
1	Amazo	44
2	Martian Manhunter	35
3	Living Tribunal	35
4	Man of Miracles	34
5	Captain Marvel	33
6	T-X	33
7	Galactus	32
8	T-1000	32
9	One-Above-All	31

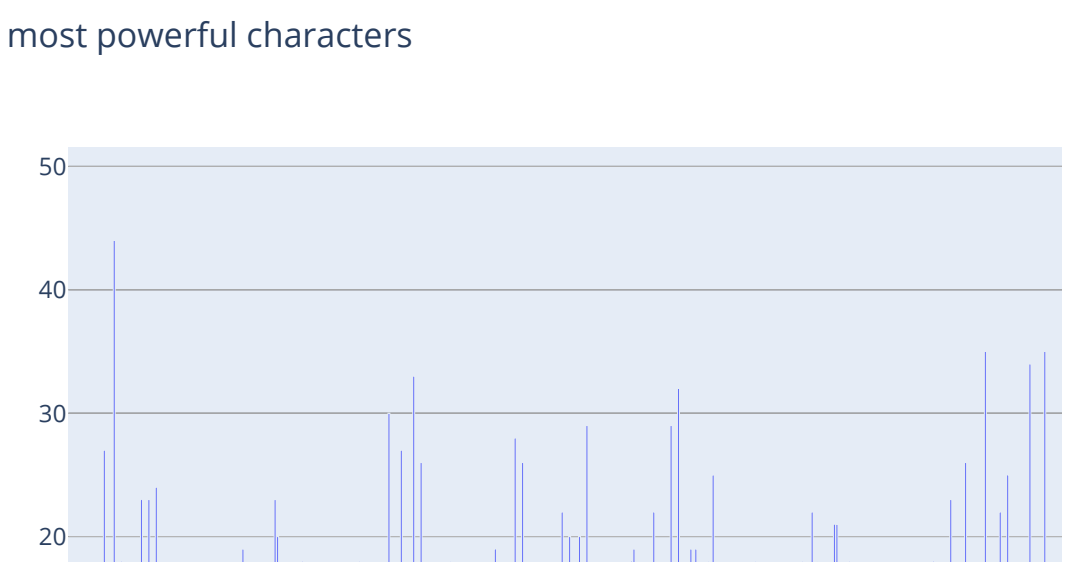
In [83]:

```
powers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 667 entries, 0 to 666
Columns: 169 entries, hero_names to total_powers
dtypes: int32(167), int64(1), object(1)
memory usage: 445.7+ KB
```

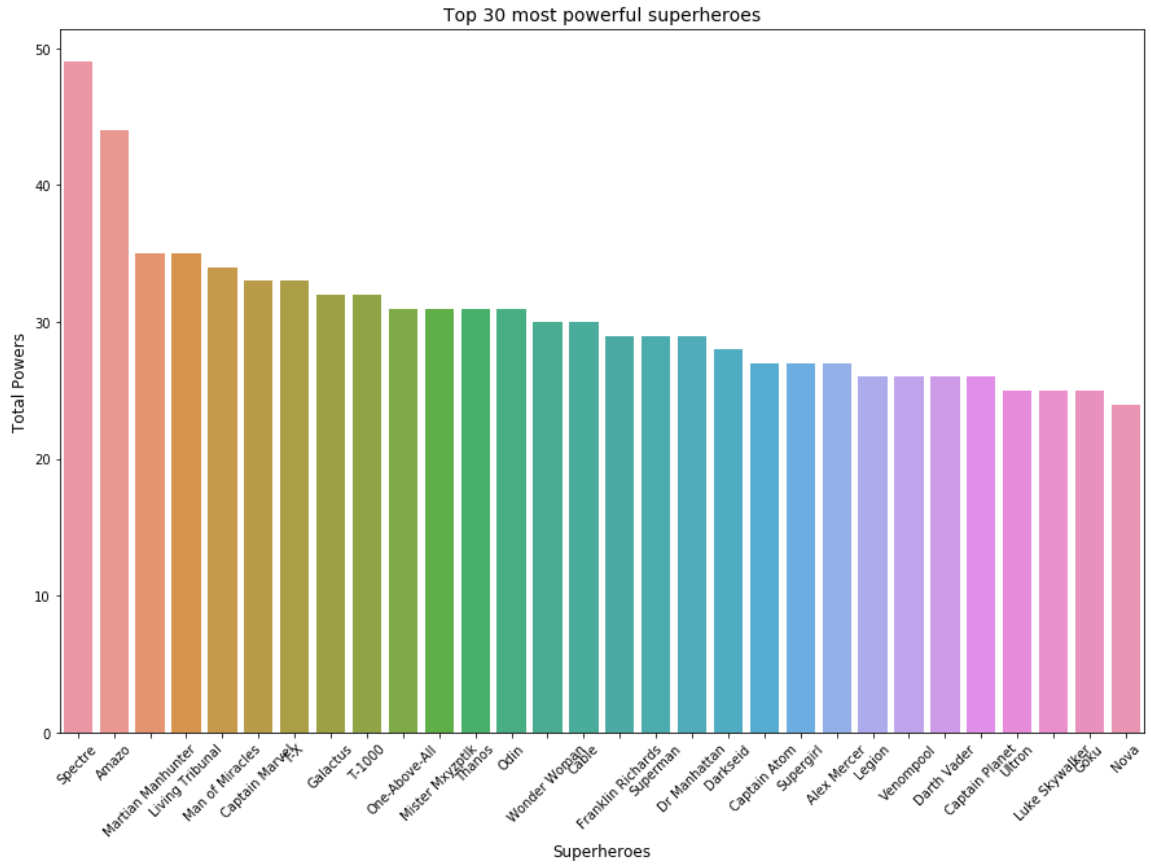
In [84]:

```
trace = go.Bar(
    x = powers['hero_names'],
    y = powers['total_powers'],
    text = ['names', 'total_powers']
)

layout = go.Layout(title="most powerful characters")

fig = go.Figure(data=[trace], layout=layout)
py.iplot(fig, filename='most-powerful-characters')
```



In [86]:

```
top_30_powerful_ones = powers.sort_values('total_powers', ascending=False).head(30)

plt.figure(figsize=(15,10))
sns.barplot(top_30_powerful_ones['hero_names'], top_30_powerful_ones['total_powers'], a
lpha=1)
plt.xticks(rotation=45)
plt.xlabel("Superheroes", fontsize=12)
plt.ylabel("Total Powers", fontsize=12)
plt.title("Top 30 most powerful superheroes", fontsize=14)
plt.show()
```



In [87]:

```
powers.columns
```

Out[87]:

```
Index(['hero_names', 'Agility', 'Accelerated Healing', 'Lantern Power Rin
g',
       'Dimensional Awareness', 'Cold Resistance', 'Durability', 'Stealt
h',
       'Energy Absorption', 'Flight',
       ...
       'Reality Warping', 'Odin Force', 'Symbiote Costume', 'Speed Force',
       'Phoenix Force', 'Molecular Dissipation', 'Vision - Cryo',
       'Omnipresent', 'Omniscient', 'total_powers'],
      dtype='object', length=169)
```

In [93]:

```
# how many of these can fly?

len(powers.loc[powers['Flight']==1, 'hero_names'].value_counts().index)
```

Out[93]:

212

In [96]:

```
# how many can heal on their own like Wolverine

len(powers[powers['Accelerated Healing']==1])
```

Out[96]:

178

Calculating BMI (to estimate who's the fittest of'em all)

$$BMI = \frac{W_{kg}}{H_m^2}$$

In [97]:

```
heroes['BMI'] = np.divide(heroes['Weight'], np.square(heroes['Height']/100))
heroes.head(3)
```

Out[97]:

	name	Gender	Eye color	Race	Hair color	Publisher	Skin color	Alignment	Height	Weight	I
0	A-Bomb	Male	yellow	Human	No Hair	Marvel Comics	unknown	good	203.0	441.0	
1	Abe Sapien	Male	blue	Icthyo Sapien	No Hair	Dark Horse Comics	blue	good	191.0	65.0	
2	Abin Sur	Male	blue	Ungaran	No Hair	DC Comics	red	good	185.0	90.0	

In [100]:

```
obese_heroes = heroes.loc[(heroes['BMI'] > 30.0) & (heroes['Alignment']=='good'), ['nam
e','BMI']]
obese_heroes.head()
```

Out[100]:

	name	BMI
0	A-Bomb	107.015458
36	Aqualad	33.455372
37	Aquaman	42.658875
42	Ares	78.889701
47	Atlas	30.159157

In [102]:

```
# top 5 obese heroes
obese_heroes.sort_values(by='BMI', ascending=False).head()
```

Out[102]:

	name	BMI
389	King Kong	870.733674
422	Machine Man	114.365911
0	A-Bomb	107.015458
331	Hulk	105.818328
575	Sasquatch	96.748186

In [112]:

```
# top 5 heaviest characters who are also agile

powers.loc[powers['hero_names'].isin(
    heroes.sort_values(by='Weight', ascending=False).head(10)['name'].values
), ['hero_names', 'Agility']].sort_values('Agility', ascending=False)
```

Out[112]:

	hero_names	Agility
107	Bloodaxe	1
180	Darkseid	1
296	Hulk	1
525	Sasquatch	1
601	Thanos	1
1	A-Bomb	0
255	Giganta	0
334	Juggernaut	0
500	Red Hulk	0
654	Wolfsbane	0

In [ ]:

In [ ]: