Prepared by Asif Bhat

Data Visualization With Plotly (Part - 1)

```python
In [1]:  import numpy as np
         import pandas as pd
         import plotly.graph_objects as go
         import plotly.offline as po
         from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
         import matplotlib.pyplot as plt
         import dash
         import plotly.express as px
         import random
         import plotly.figure_factory as ff
```

Loading Datasets

```python
In [2]:  pokemon = pd.read_csv("C:/Users/DELL/Documents/GitHub/Public/Data-Visualization/Plotly/Datasets/pokemon_updated.csv")
         pokemon.head(10)
```

Out[2]:

	#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary	Total
0	1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	1	False	318
1	2	Ivysaur	Grass	Poison	60	62	63	80	80	60	1	False	405
2	3	Venusaur	Grass	Poison	80	82	83	100	100	80	1	False	525
3	3	VenusaurMega Venusaur	Grass	Poison	80	100	123	122	120	80	1	False	625
4	4	Charmander	Fire	NaN	39	52	43	60	50	65	1	False	309
5	5	Charmeleon	Fire	NaN	58	64	58	80	65	80	1	False	405
6	6	Charizard	Fire	Flying	78	84	78	109	85	100	1	False	534
7	6	CharizardMega Charizard X	Fire	Dragon	78	130	111	130	85	100	1	False	634
8	6	CharizardMega Charizard Y	Fire	Flying	78	104	78	159	115	100	1	False	634
9	7	Squirtle	Water	NaN	44	48	65	50	64	43	1	False	314

```
In [3]: stdperf = pd.read_csv("C:/Users/DELL/Documents/GitHub/Public/Data-Visualization/Plotly/Datasets/studentp.csv")
        stdperf.head(10)
```

Out[3]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75
5	female	group B	associate's degree	standard	none	71	83	78
6	female	group B	some college	standard	completed	88	95	92
7	male	group B	some college	free/reduced	none	40	43	39
8	male	group D	high school	free/reduced	completed	64	64	67
9	female	group B	high school	free/reduced	none	38	60	50

```
In [4]: corona = pd.read_csv('C:/Users/DELL/Documents/GitHub/Public/COVID-19/covid/data/countries-aggregated.csv' ,
                             index_col='Date' , parse_dates=True)
        corona.head(10)
```

Out[4]:

Date	Country	Confirmed	Recovered	Deaths
2020-01-22	Afghanistan	0	0	0
2020-01-22	Albania	0	0	0
2020-01-22	Algeria	0	0	0
2020-01-22	Andorra	0	0	0
2020-01-22	Angola	0	0	0
2020-01-22	Antigua and Barbuda	0	0	0
2020-01-22	Argentina	0	0	0
2020-01-22	Armenia	0	0	0
2020-01-22	Australia	0	0	0
2020-01-22	Austria	0	0	0

```python
In [5]: spotify = pd.read_csv("C:/Users/DELL/Documents/GitHub/Public/Data-Visualization/Plotly/Datasets/spotify.csv" , index_col="Date")
        spotify.head(10)
```

Out[5]:

Date	Shape of You	Despacito	Something Just Like This	HUMBLE.	Unforgettable
2017-01-06	12287078	NaN	NaN	NaN	NaN
2017-01-07	13190270	NaN	NaN	NaN	NaN
2017-01-08	13099919	NaN	NaN	NaN	NaN
2017-01-09	14506351	NaN	NaN	NaN	NaN
2017-01-10	14275628	NaN	NaN	NaN	NaN
2017-01-11	14372699	NaN	NaN	NaN	NaN
2017-01-12	14148108	NaN	NaN	NaN	NaN
2017-01-13	14536236	275178.0	NaN	NaN	NaN
2017-01-14	14173311	1144886.0	NaN	NaN	NaN
2017-01-15	12889849	1288198.0	NaN	NaN	NaN

```python
In [6]: housing = pd.read_csv('C:/Users/DELL/Documents/GitHub/Data-Visualization/housing.csv')
        housing.tail()
```

Out[6]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
20635	-121.09	39.48	25.0	1665.0	374.0	845.0	330.0	1.5603	78100.0	INLAND
20636	-121.21	39.49	18.0	697.0	150.0	356.0	114.0	2.5568	77100.0	INLAND
20637	-121.22	39.43	17.0	2254.0	485.0	1007.0	433.0	1.7000	92300.0	INLAND
20638	-121.32	39.43	18.0	1860.0	409.0	741.0	349.0	1.8672	84700.0	INLAND
20639	-121.24	39.37	16.0	2785.0	616.0	1387.0	530.0	2.3886	89400.0	INLAND

```python
In [7]: insurance = pd.read_csv('C:/Users/DELL/Documents/GitHub/Data-Visualization/insurance.csv')
        insurance.head(10)
```

Out[7]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
5	31	female	25.740	0	no	southeast	3756.62160
6	46	female	33.440	1	no	southeast	8240.58960
7	37	female	27.740	3	no	northwest	7281.50560
8	37	male	29.830	2	no	northeast	6406.41070
9	60	female	25.840	0	no	northwest	28923.13692

```
In [8]: employment = pd.read_excel("C:/Users/DELL/Documents/GitHub/Public/Data-Visualization/Plotly/Datasets/unemployment.xlsx")
        employment.head(10)
```

Out[8]:

	Age	Gender	Period	Unemployed
0	16 to 19 years	Men	2005-01-01	91000
1	20 to 24 years	Men	2005-01-01	175000
2	25 to 34 years	Men	2005-01-01	194000
3	35 to 44 years	Men	2005-01-01	201000
4	45 to 54 years	Men	2005-01-01	207000
5	55 to 64 years	Men	2005-01-01	101000
6	65 years and over	Men	2005-01-01	33000
7	16 to 19 years	Women	2005-01-01	38000
8	20 to 24 years	Women	2005-01-01	90000
9	25 to 34 years	Women	2005-01-01	142000

```
In [9]: helpdesk = pd.read_csv("C:/Users/DELL/Documents/GitHub/Public/Data-Visualization/Plotly/Datasets/helpdesk.csv")
        helpdesk.head(10)
```

Out[9]:

	ticket	requestor	RequestorSeniority	ITOwner	FiledAgainst	TicketType	Severity	Priority	daysOpen	Satisfaction
0	1	1929	1 - Junior	50	Systems	Issue	2 - Normal	0 - Unassigned	3	1 - Unsatisfied
1	2	1587	2 - Regular	15	Software	Request	1 - Minor	1 - Low	5	1 - Unsatisfied
2	3	925	2 - Regular	15	Access/Login	Request	2 - Normal	0 - Unassigned	0	0 - Unknown
3	4	413	4 - Management	22	Systems	Request	2 - Normal	0 - Unassigned	20	0 - Unknown
4	5	318	1 - Junior	22	Access/Login	Request	2 - Normal	1 - Low	1	1 - Unsatisfied
5	6	858	4 - Management	38	Access/Login	Request	2 - Normal	3 - High	0	0 - Unknown
6	7	1978	3 - Senior	10	Systems	Request	2 - Normal	3 - High	9	0 - Unknown
7	8	1209	4 - Management	1	Software	Request	2 - Normal	0 - Unassigned	15	0 - Unknown
8	9	887	2 - Regular	14	Software	Request	2 - Normal	2 - Medium	6	1 - Unsatisfied
9	10	1780	3 - Senior	46	Access/Login	Request	2 - Normal	1 - Low	1	1 - Unsatisfied

```
In [10]: fish= pd.read_csv("Fish.csv")
         fish.head(10)
```

Out[10]:

	Species	Weight	Length1	Length2	Length3	Height	Width
0	Bream	242.0	23.2	25.4	30.0	11.5200	4.0200
1	Bream	290.0	24.0	26.3	31.2	12.4800	4.3056
2	Bream	340.0	23.9	26.5	31.1	12.3778	4.6961
3	Bream	363.0	26.3	29.0	33.5	12.7300	4.4555
4	Bream	430.0	26.5	29.0	34.0	12.4440	5.1340
5	Bream	450.0	26.8	29.7	34.7	13.6024	4.9274
6	Bream	500.0	26.8	29.7	34.5	14.1795	5.2785
7	Bream	390.0	27.6	30.0	35.0	12.6700	4.6900
8	Bream	450.0	27.6	30.0	35.1	14.0049	4.8438
9	Bream	500.0	28.5	30.7	36.2	14.2266	4.9594

```
In [11]: exercise = pd.read_csv("C:/Users/DELL/Documents/GitHub/Public/Data-Visualization/Plotly/Datasets/exercise.csv")
         exercise.head(10)
```

Out[11]:

	id	diet	pulse	time	kind
0	1	low fat	85	1 min	rest
1	1	low fat	85	15 min	rest
2	1	low fat	88	30 min	rest
3	2	low fat	90	1 min	rest
4	2	low fat	92	15 min	rest
5	2	low fat	93	30 min	rest
6	3	low fat	97	1 min	rest
7	3	low fat	97	15 min	rest
8	3	low fat	94	30 min	rest
9	4	low fat	80	1 min	rest

```
In [12]: suicide = pd.read_csv("C:/Users/DELL/Documents/GitHub/Public/Data-Visualization/Plotly/Datasets/suicide.csv")
         suicide.head(10)
```

Out[12]:

	country	year	sex	age	suicides_no	population	suicides/100k pop	country-year	HDI for year	gdp_for_year ($)	gdp_per_capita ($)	generation
0	Albania	1987	male	15-24 years	21	312900	6.71	Albania1987	NaN	2,156,624,900	796	Generation X
1	Albania	1987	male	35-54 years	16	308000	5.19	Albania1987	NaN	2,156,624,900	796	Silent
2	Albania	1987	female	15-24 years	14	289700	4.83	Albania1987	NaN	2,156,624,900	796	Generation X
3	Albania	1987	male	75+ years	1	21800	4.59	Albania1987	NaN	2,156,624,900	796	G.I. Generation
4	Albania	1987	male	25-34 years	9	274300	3.28	Albania1987	NaN	2,156,624,900	796	Boomers
5	Albania	1987	female	75+ years	1	35600	2.81	Albania1987	NaN	2,156,624,900	796	G.I. Generation
6	Albania	1987	female	35-54 years	6	278800	2.15	Albania1987	NaN	2,156,624,900	796	Silent
7	Albania	1987	female	25-34 years	4	257200	1.56	Albania1987	NaN	2,156,624,900	796	Boomers
8	Albania	1987	male	55-74 years	1	137500	0.73	Albania1987	NaN	2,156,624,900	796	G.I. Generation
9	Albania	1987	female	5-14 years	0	311000	0.00	Albania1987	NaN	2,156,624,900	796	Generation X

```
In [13]: iris = pd.read_csv("iris.csv")
         iris.head()
```

Out[13]:

	Id	Sepal Length (cm)	Sepal Width (cm)	Petal Length (cm)	Petal Width (cm)	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

```
In [14]: canada = pd.read_csv("C:/Users/DELL/Documents/GitHub/Public/Data-Visualization/Plotly/Datasets/canada.csv")
         canada.head()
```

Out[14]:

	Type	Coverage	OdName	AREA	AreaName	REG	RegName	DEV	DevName	1980	...	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
0	Immigrants	Foreigners	Afghanistan	935	Asia	5501	Southern Asia	902	Developing regions	16	...	2978	3436	3009	2652	2111	1746	1758	2203	2635	2004
1	Immigrants	Foreigners	Albania	908	Europe	925	Southern Europe	901	Developed regions	1	...	1450	1223	856	702	560	716	561	539	620	603
2	Immigrants	Foreigners	Algeria	903	Africa	912	Northern Africa	902	Developing regions	80	...	3616	3626	4807	3623	4005	5393	4752	4325	3774	4331
3	Immigrants	Foreigners	American Samoa	909	Oceania	957	Polynesia	902	Developing regions	0	...	0	0	1	0	0	0	0	0	0	0
4	Immigrants	Foreigners	Andorra	908	Europe	925	Southern Europe	901	Developed regions	0	...	0	0	1	1	0	0	0	0	1	1

5 rows × 43 columns

```python
In [15]:  canada.columns
```

```
Out[15]:  Index(['Type', 'Coverage', 'OdName', 'AREA', 'AreaName', 'REG', 'RegName',
                 'DEV', 'DevName', '1980', '1981', '1982', '1983', '1984', '1985',
                 '1986', '1987', '1988', '1989', '1990', '1991', '1992', '1993', '1994',
                 '1995', '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003',
                 '2004', '2005', '2006', '2007', '2008', '2009', '2010', '2011', '2012',
                 '2013'],
                dtype='object')
```

```python
In [16]:  canada.drop(columns=['AREA' , 'DEV', 'DevName' , 'REG', 'Type', 'Coverage' , 'AreaName', 'RegName' ], inplace=True)
          canada.head()
```

Out[16]:

	OdName	1980	1981	1982	1983	1984	1985	1986	1987	1988	...	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
0	Afghanistan	16	39	39	47	71	340	496	741	828	...	2978	3436	3009	2652	2111	1746	1758	2203	2635	2004
1	Albania	1	0	0	0	0	0	1	2	2	...	1450	1223	856	702	560	716	561	539	620	603
2	Algeria	80	67	71	69	63	44	69	132	242	...	3616	3626	4807	3623	4005	5393	4752	4325	3774	4331
3	American Samoa	0	1	0	0	0	0	0	1	0	...	0	0	1	0	0	0	0	0	0	0
4	Andorra	0	0	0	0	0	0	2	0	0	...	0	0	1	1	0	0	0	0	1	1

5 rows × 35 columns

```python
In [17]:  canada.rename(columns={'OdName':'Country'} , inplace=True)
          canada.set_index(canada.Country,inplace=True)
          canada.head()
```

Out[17]:

	Country	1980	1981	1982	1983	1984	1985	1986	1987	1988	...	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
Country																					
Afghanistan	Afghanistan	16	39	39	47	71	340	496	741	828	...	2978	3436	3009	2652	2111	1746	1758	2203	2635	2004
Albania	Albania	1	0	0	0	0	0	1	2	2	...	1450	1223	856	702	560	716	561	539	620	603
Algeria	Algeria	80	67	71	69	63	44	69	132	242	...	3616	3626	4807	3623	4005	5393	4752	4325	3774	4331
American Samoa	American Samoa	0	1	0	0	0	0	0	1	0	...	0	0	1	0	0	0	0	0	0	0
Andorra	Andorra	0	0	0	0	0	0	2	0	0	...	0	0	1	1	0	0	0	0	1	1

5 rows × 35 columns

```
In [18]: canada2 = canada.copy()
         canada2.head()
```

Out[18]:

	Country	1980	1981	1982	1983	1984	1985	1986	1987	1988	...	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
Country																					
Afghanistan	Afghanistan	16	39	39	47	71	340	496	741	828	...	2978	3436	3009	2652	2111	1746	1758	2203	2635	2004
Albania	Albania	1	0	0	0	0	0	1	2	2	...	1450	1223	856	702	560	716	561	539	620	603
Algeria	Algeria	80	67	71	69	63	44	69	132	242	...	3616	3626	4807	3623	4005	5393	4752	4325	3774	4331
American Samoa	American Samoa	0	1	0	0	0	0	0	1	0	...	0	0	1	0	0	0	0	0	0	0
Andorra	Andorra	0	0	0	0	0	0	2	0	0	...	0	0	1	1	0	0	0	0	1	1

5 rows × 35 columns

```
In [19]: canada.index.name=None
         canada.head()
```

Out[19]:

	Country	1980	1981	1982	1983	1984	1985	1986	1987	1988	...	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
Afghanistan	Afghanistan	16	39	39	47	71	340	496	741	828	...	2978	3436	3009	2652	2111	1746	1758	2203	2635	2004
Albania	Albania	1	0	0	0	0	0	1	2	2	...	1450	1223	856	702	560	716	561	539	620	603
Algeria	Algeria	80	67	71	69	63	44	69	132	242	...	3616	3626	4807	3623	4005	5393	4752	4325	3774	4331
American Samoa	American Samoa	0	1	0	0	0	0	0	1	0	...	0	0	1	0	0	0	0	0	0	0
Andorra	Andorra	0	0	0	0	0	0	2	0	0	...	0	0	1	1	0	0	0	0	1	1

5 rows × 35 columns

```
In [20]: del canada['Country']
         canada.head()
```

Out[20]:

	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	...	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
Afghanistan	16	39	39	47	71	340	496	741	828	1076	...	2978	3436	3009	2652	2111	1746	1758	2203	2635	2004
Albania	1	0	0	0	0	0	1	2	2	3	...	1450	1223	856	702	560	716	561	539	620	603
Algeria	80	67	71	69	63	44	69	132	242	434	...	3616	3626	4807	3623	4005	5393	4752	4325	3774	4331
American Samoa	0	1	0	0	0	0	0	1	0	1	...	0	0	1	0	0	0	0	0	0	0
Andorra	0	0	0	0	0	0	2	0	0	0	...	0	0	1	1	0	0	0	0	1	1

5 rows × 34 columns

```
In [21]: canada = canada.transpose()
```

`canada.head()`

Out[22]:

	Afghanistan	Albania	Algeria	American Samoa	Andorra	Angola	Antigua and Barbuda	Argentina	Armenia	Australia	...	Uzbekistan	Vanuatu	Venezuela (Bolivarian Republic of)	Viet Nam	Western Sahara	Yemen	Zambia	Zimbabwe	Unknown	Total
1980	16	1	80	0	0	1	0	368	0	702	...	0	0	103	1191	0	1	11	72	44000	143137
1981	39	0	67	1	0	3	0	426	0	639	...	0	0	117	1829	0	2	17	114	18078	128641
1982	39	0	71	0	0	6	0	626	0	484	...	0	0	174	2162	0	1	11	102	16904	121175
1983	47	0	69	0	0	6	0	241	0	317	...	0	0	124	3404	0	6	7	44	13635	89185
1984	71	0	63	0	0	4	42	237	0	317	...	0	0	142	7583	0	0	16	32	14855	88272

5 rows × 197 columns

Scatter Plot

In [23]:
```python
#Simple Scatter Plot
random_x = np.random.randint(1,100,50)
random_y = np.random.randint(1,100,50)

data = [go.Scatter(
                x = random_x,
                y = random_y,
                mode = 'markers'
                )
    ]

layout = go.Layout(
                xaxis=dict(
                        showgrid=False, # Hide Gridlines
                        showline=False, # Hide X-Axis
                        ),

                yaxis=dict(
                        showgrid=False, # Hide Gridlines
                        showline=False, # Hide X-Axis
                        ),
                )
fig = go.Figure(data=data,layout=layout)
iplot(fig)
```
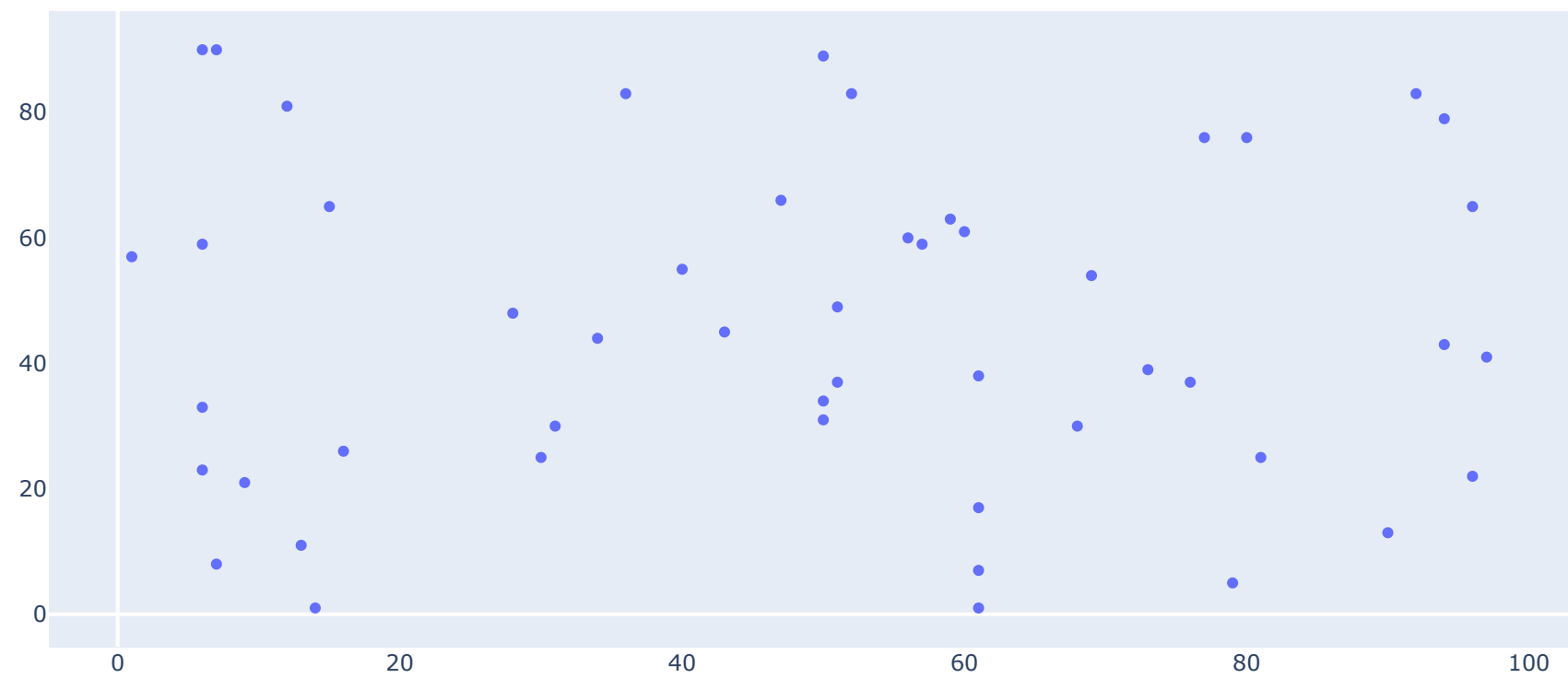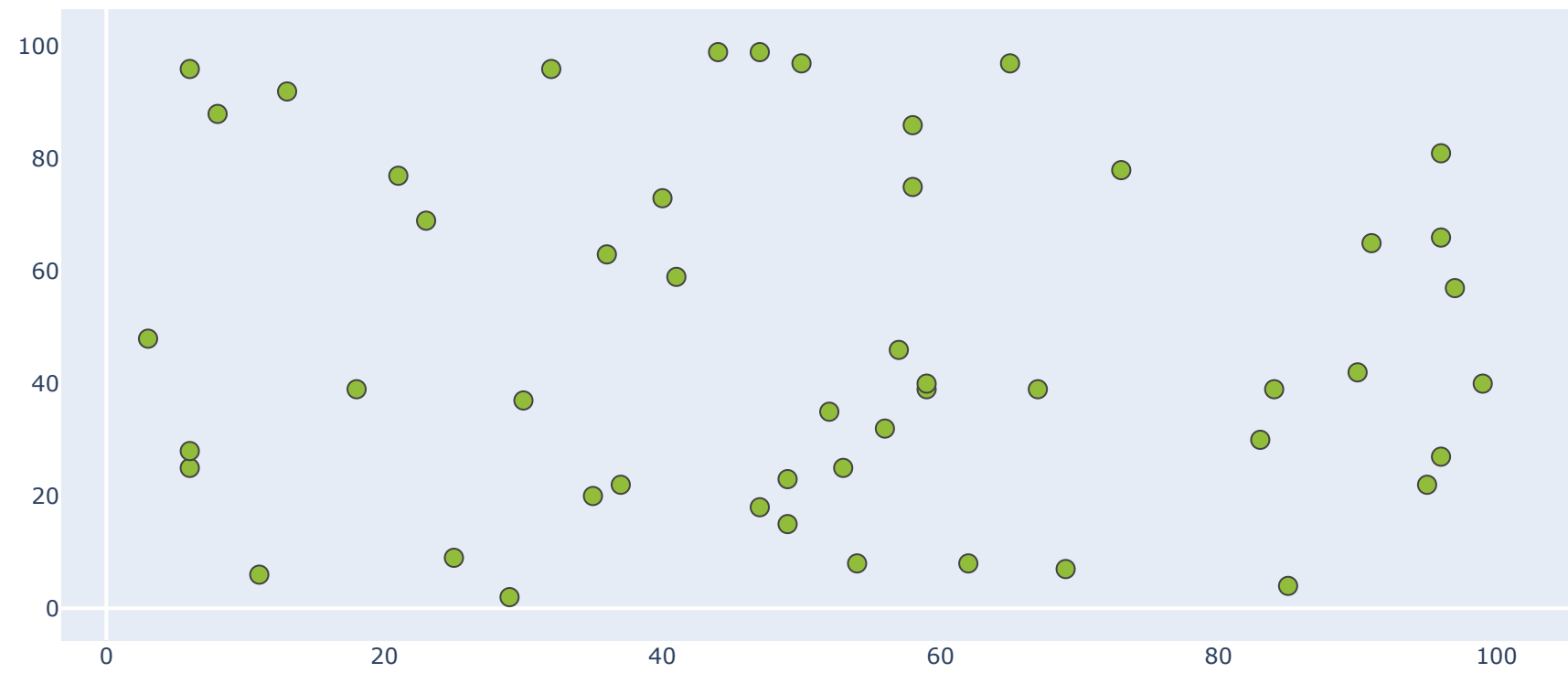
```
In [24]:  # Changing Marker size , shape & color using Marker parameter
          x_val = np.random.randint(1,100,50)
          y_val = np.random.randint(1,100,50)

          data = [go.Scatter(
                             x = x_val,
                             y = y_val,
                             mode = 'markers',
                             marker = dict(
                                          size = 10,
                                          color = '#91bd3a', #color of marker
                                          symbol = 'circle', # Shape of scatter plot
                                          line = dict(width = 1) #width of boundary
                                        )
                           )
                 ]

          layout = go.Layout(
                            xaxis=dict(
                                     showgrid=False, # Hide Gridlines
                                     showline=False, # Hide X-Axis
                                   ),

                            yaxis=dict(
                                     showgrid=False, # Hide Gridlines
                                     showline=False, # Hide X-Axis
                                   ),
                           )

          fig = go.Figure(data=data,layout=layout)
          iplot(fig)
```

```python
# Defining Labels (X-Axis & Y-Axis label , Graph tile)
x_val = np.random.randint(1,100,50)
y_val = np.random.randint(1,100,50)

data = [go.Scatter(
                x = x_val,
                y = y_val,
                mode = 'markers',
                marker = dict(
                        size = 10,
                        color = '#4ED700',
                        symbol = 'circle',
                        line = dict(width = 1,color = '#0E8700')
                        )
                )
        ]
layout = go.Layout(
                title = '$Scatter Plot$', # Title
                xaxis = dict(title = '$X-Axis$',showgrid=False,showline=False), # x-axis label
                yaxis = dict(title = '$Y-Axis$',showgrid=False,showline=False), # y-axis label
                )
fig = go.Figure(data=data, layout=layout)
iplot(fig)
```


Scatter Plot
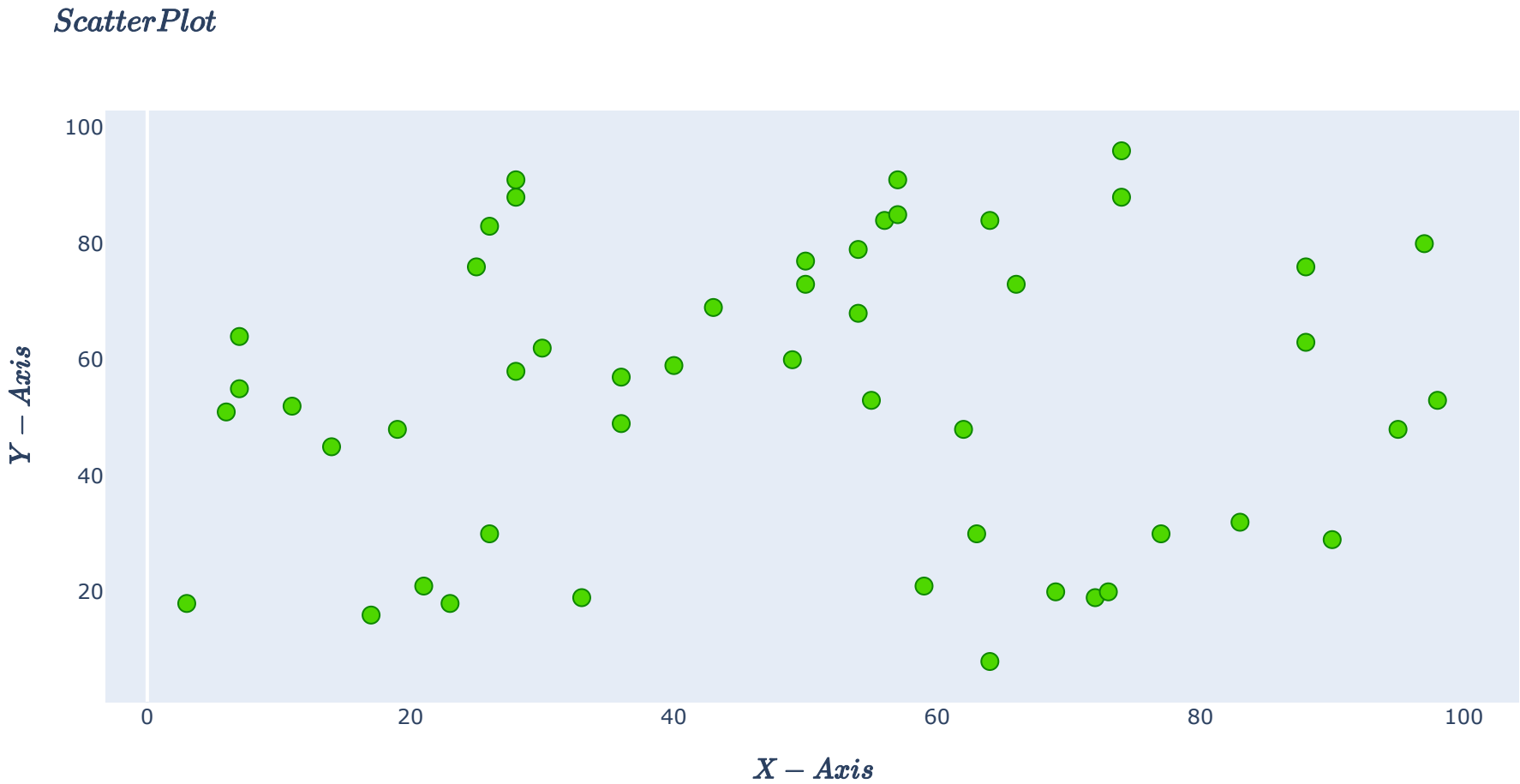
```python
In [26]:  x_values = np.linspace(0, 100, 100) # 100 evenly spaced values
          y_values = np.random.randn(100)     # 100 random values

          trace0 = go.Scatter(
                            x = x_values,
                            y = y_values,
                            mode = 'markers',
                            marker = dict(
                                        size = 7,
                                        color = '#F4D03F',
                                        symbol = 'circle',
                                        line = dict(width = 1,color = '#0E8700')
                                        )
                            )

          trace1 = go.Scatter(
                            x = x_values,
                            y = y_values-5,
                            mode = 'markers',
                            marker = dict(size = 7,
                                        color = '#A9DFBF',
                                        symbol = 'circle',
                                        line = dict(width = 1,color = '#0E8700')
                                        )
                            )

          data = [trace0, trace1]

          layout = go.Layout(
                            xaxis=dict(
                                    showgrid=False, # Hide Gridlines
                                    showline=False, # Hide X-Axis
                                    ),

                            yaxis=dict(
                                    showgrid=False, # Hide Gridlines
                                    showline=False, # Hide X-Axis
                                    ),
                            )


          fig = go.Figure(data=data,layout=layout)
          iplot(fig)
```
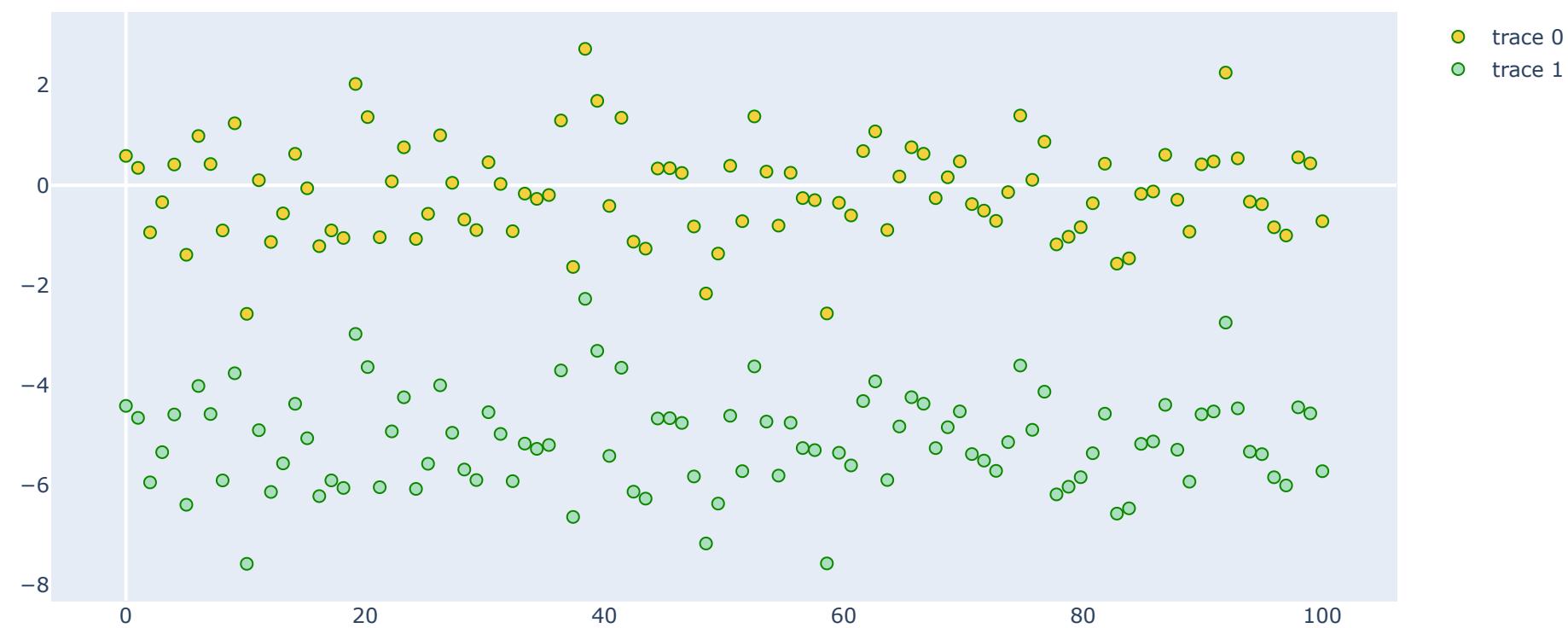
In [27]: `insurance.head(10)`

Out[27]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
5	31	female	25.740	0	no	southeast	3756.62160
6	46	female	33.440	1	no	southeast	8240.58960
7	37	female	27.740	3	no	northwest	7281.50560
8	37	male	29.830	2	no	northeast	6406.41070
9	60	female	25.840	0	no	northwest	28923.13692

```
In [28]: data = [go.Scatter(
                         x = insurance.bmi,
                         y = insurance.charges,
                         mode = 'markers',
                         marker = dict(size = 7,
                                       color = '#4ED700',
                                       symbol = 'circle',
                                       line = dict(width = 1,color = '#0E8700')
                                       )
                         )
               ]

         layout = go.Layout(
                         title = '$Insurance$', # Chart Title
                         xaxis = dict(title = '$BMI$',showgrid=False,showline=False), # x-axis label
                         yaxis = dict(title = '$Charges$',showgrid=False,showline=False), # y-axis label
                         )

         fig = go.Figure(data=data, layout=layout)

         iplot(fig)
```
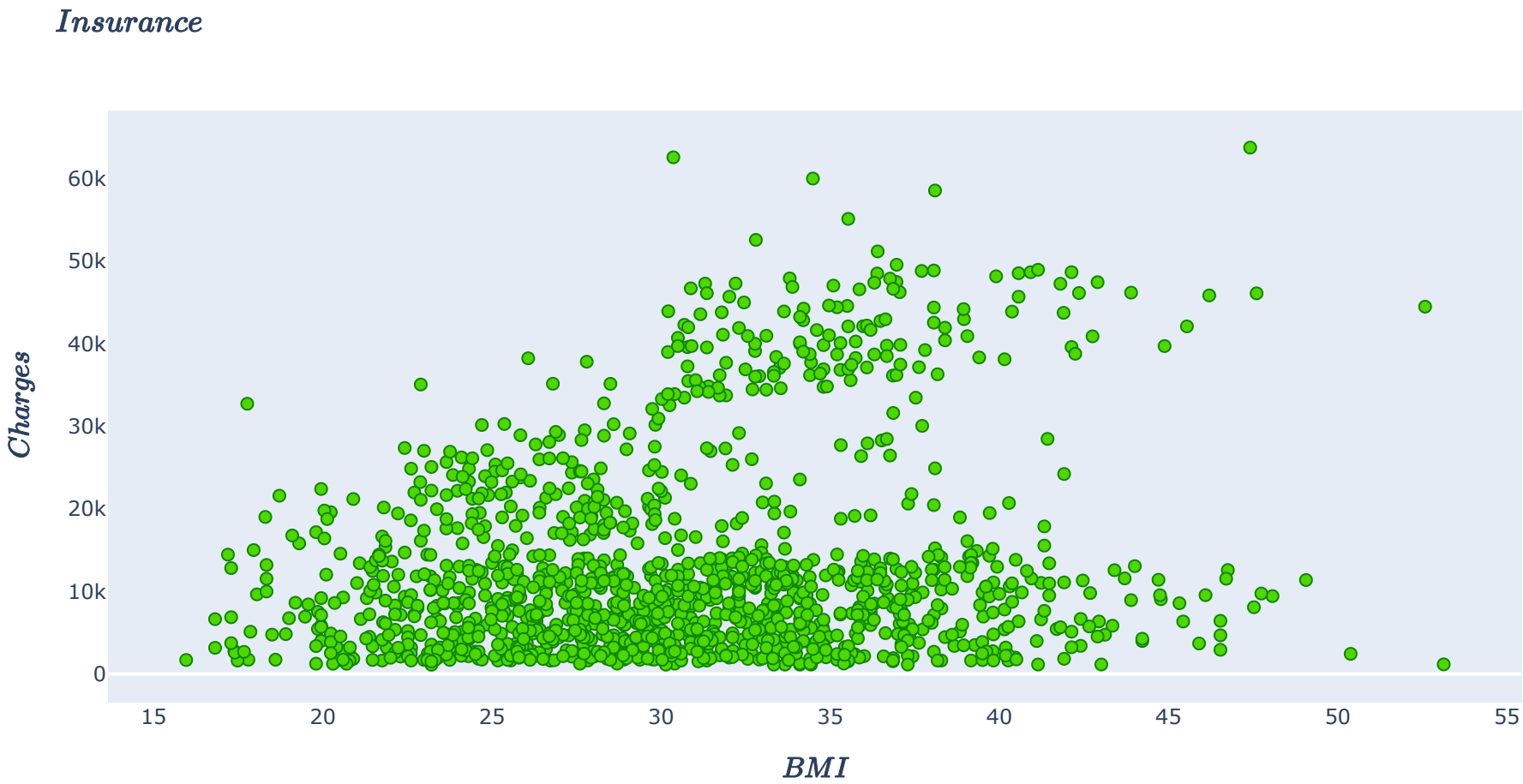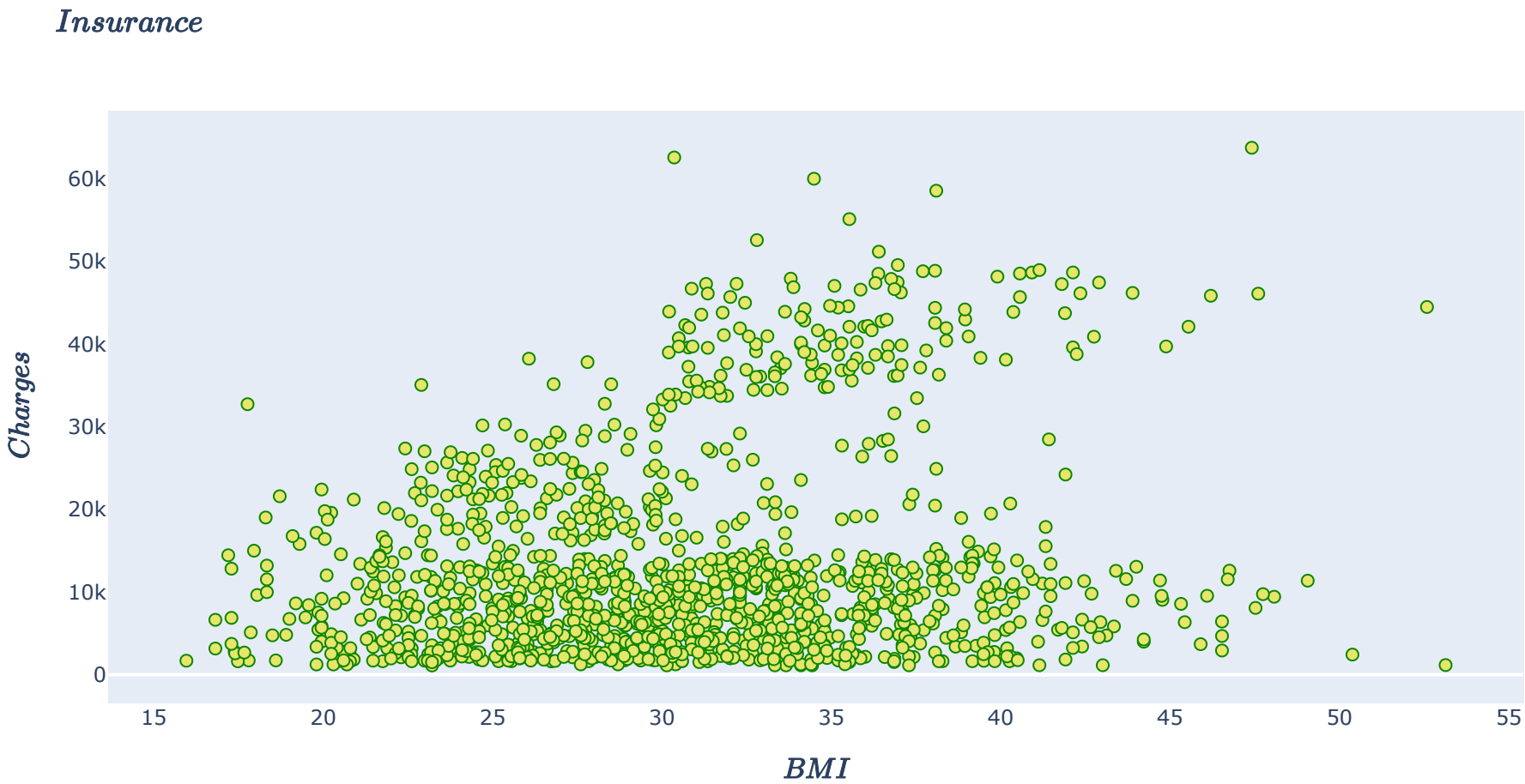

Insurance

```
In [29]: data = [go.Scatter(
                        x = insurance.bmi,
                        y = insurance.charges,
                        mode = 'markers',
                        marker = dict(size = 7,
                                      color = '#4ED700',
                                      symbol = 'circle',
                                      line = dict(width = 1,color = '#0E8700')
                                      )
                        )
             ]

         layout = go.Layout(
                        title = '$Insurance$', # Title
                        xaxis = dict(title = '$BMI$',showgrid=False,showline=False), # x-axis label
                        yaxis = dict(title = '$Charges$',showgrid=False,showline=False), # y-axis label
                        )

         fig = go.Figure(data=data, layout=layout)

         # Updating Traces
         fig.update_traces(
                        marker=dict(color="#e6e56c"),
                        )

         iplot(fig)
```

```
In [30]:  # trace0 will capture all smokers
          trace0 = go.Scatter(
                              x = insurance[insurance.smoker=='yes'].bmi,
                              y = insurance[insurance.smoker=='yes'].charges,
                              mode = 'markers',
                              name = 'Smoker',
                          marker = dict(size = 7, color = '#F39C12',symbol = 'circle',line = dict(width = 1))
                              )

          # trace1 will capture all non-smokers
          trace1 = go.Scatter(
                              x = insurance[insurance.smoker=='no'].bmi,
                              y = insurance[insurance.smoker=='no'].charges,
                              mode = 'markers',
                              name = 'Non-Smoker',
                              marker = dict(size = 7, color = '#8BC34A',symbol = 'circle',line = dict(width = 1))
                              )

          layout = go.Layout(
                              title = '$Scatter Plot$', # Title
                              xaxis = dict(title = '$BMI$',showgrid=False,showline=False), # x-axis label
                              yaxis = dict(title = '$Charges$',showgrid=False,showline=False), # y-axis label
                              )

          data = [trace0, trace1]
          fig = go.Figure(data=data,layout=layout)
          iplot(fig)
```
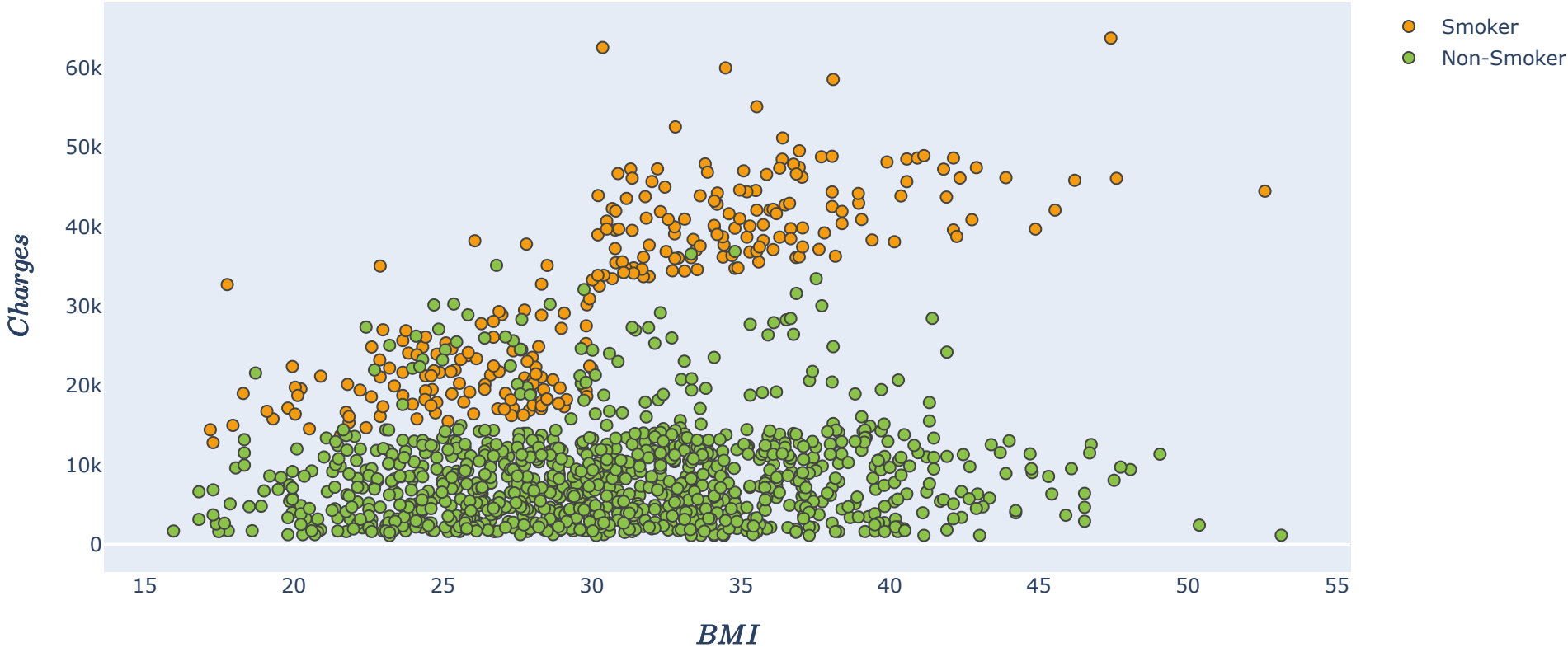
```python
In [31]: # trace0 will capture all smokers
trace0 = go.Scatter(
                    x = insurance[insurance.smoker=='yes'].bmi,
                    y = insurance[insurance.smoker=='yes'].charges,
                    mode = 'markers',
                    name = 'Smoker',
                    marker = dict(size = 14, color = '#9ca4c4',symbol = 'circle',
                                line = dict(width = 1,color = '#5D6D7E')
                                )

                    )

# trace1 will capture all non-smokers
trace1 = go.Scatter(
                    x = insurance[insurance.smoker=='no'].bmi,
                    y = insurance[insurance.smoker=='no'].charges,
                    mode = 'markers',
                    name = 'Non-Smoker',
                    marker = dict(size = 14, color = '#cbcbcb',symbol = 'circle',
                                line = dict(width = 1,color = '#626567')
                                )
                    )

#Layout Setting
layout = go.Layout(
                    title=dict(text = "Insurance Data",x=0.5,y=0.95),
                    title_font_size=30,
                    title_font_color='#F1C40F',
                    xaxis=dict(
                            showgrid=False, # Hide Gridlines
                            showline=True, # Show X-Axis
                            linecolor='black', # Color of X-axis
                            tickfont_color='black', #Color of ticks
                            showticklabels=True, #Show X Labels
                            dtick=5,
                            ticks='outside',
                            tickcolor='black',
                            ),

                    yaxis=dict(
                            showgrid=False,
                            showline=True,
                            linecolor='black',
                            tickfont_color='black',
                            showticklabels=True,
                            ticks='outside',
                            tickcolor='black',
                            ),
                    legend=dict(
                            font_size=15,
                            yanchor='bottom',
                            xanchor='right',
                            ),
                    paper_bgcolor='white',
                    plot_bgcolor='white',
                    hovermode='closest',
                    width=970,
                    height=800,
                    )

data = [trace0, trace1]
```

```
fig = go.Figure(data=data,layout=layout)
iplot(fig)
```


Insurance Data
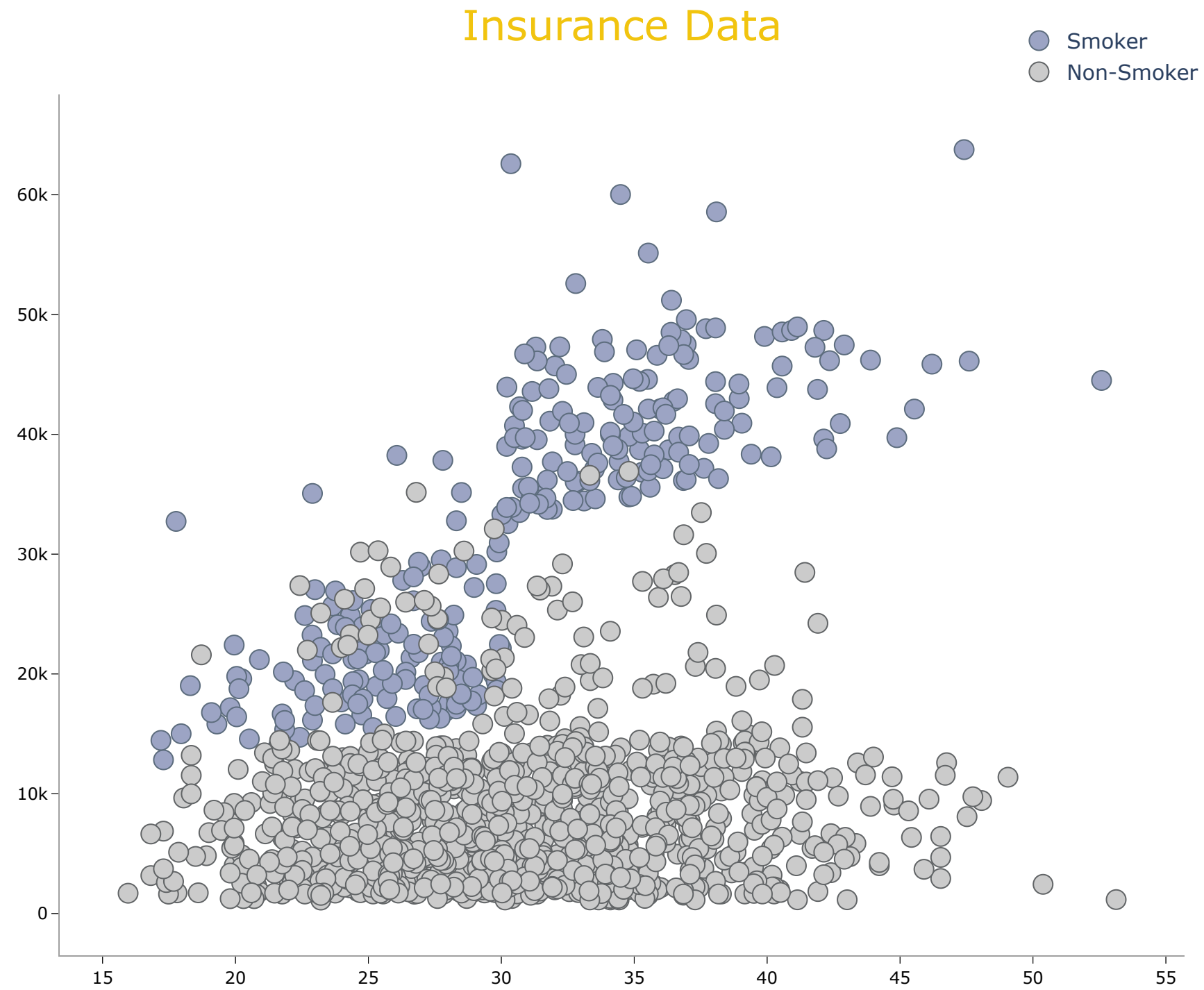
```
In [32]:  # Display multiple Scatter plots in one figure using Subplots

          from plotly.subplots import make_subplots

          #Subplot initialization
          fig = make_subplots(
                            rows=1,
                            cols=2,
                            subplot_titles=("Subplot-1", "Subplot-2")
                          )


          # Subplot - 1 (Add graph object trace to a figure)
          fig.add_trace(go.Scatter
                        (
                          x = x_val,
                          y = y_val,
                          mode = 'markers',
                          marker = dict(size = 10, color = 'crimson',symbol = 'circle',line = dict(width = 1,color = '#0E8700'))
                        ),
                      row=1, col=1
                      )

          # Add graph object trace to a figure (Subplot-2)
          fig.add_trace(go.Scatter
                        (
                          x = x_val,
                          y = y_val,
                          mode = 'markers',
                          marker = dict(size = 10, color = 'gold',symbol = 'circle',line = dict(width = 1))
                        ),
                      row=1, col=2
                      )

          # Hide grid lines
          fig.update_xaxes(showgrid=False)
          fig.update_yaxes(showgrid=False)

          fig.show()
```
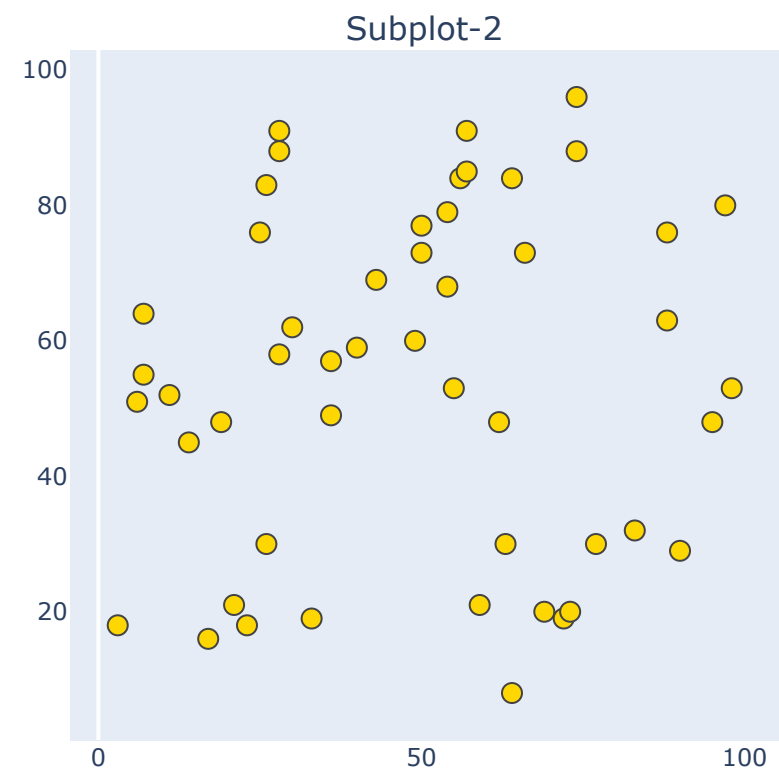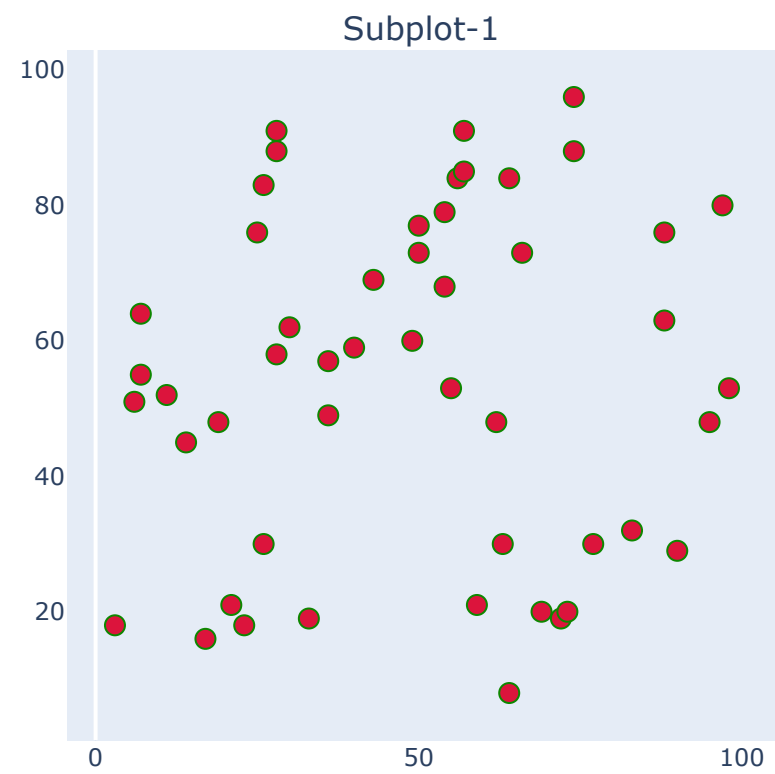
```
In [33]:  # Display multiple Scatter plots in one figure using Subplots

          from plotly.subplots import make_subplots

          #Subplot initialization
          fig = make_subplots(
                          rows=1,
                          cols=2,
                          subplot_titles=("Subplot-1", "Subplot-2")
                        )


          # Subplot - 1 (Add graph object trace to a figure)
          fig.add_trace(go.Scatter
                        (
                          x = x_val,
                          y = y_val,
                          mode = 'markers',
                          marker = dict(size = 10, color = '#4ED700',symbol = 'circle',line = dict(width = 1,color = '#0E8700'))
                        ),
                      row=1, col=1
                      )

          # Add graph object trace to a figure
          fig.add_trace(go.Scatter
                        (
                          x = x_val,
                          y = y_val,
                          mode = 'markers',
                          marker = dict(size = 10, color = '#FFC107',symbol = 'circle',line = dict(width = 1))
                        ),
                      row=1, col=2
                      )

          #Update traces in Suplots
          fig.update_traces(
                          marker=dict(color="#5DADE2"),
                          col=1,
                          row = 1
                          )

          #Update traces in Suplots
          fig.update_traces(
                          marker=dict(color="#E67E22"),
                          col=2,
                          row = 1
                          )
          # Hide grid lines
          fig.update_xaxes(showgrid=False)
          fig.update_yaxes(showgrid=False)

          fig.show()
```
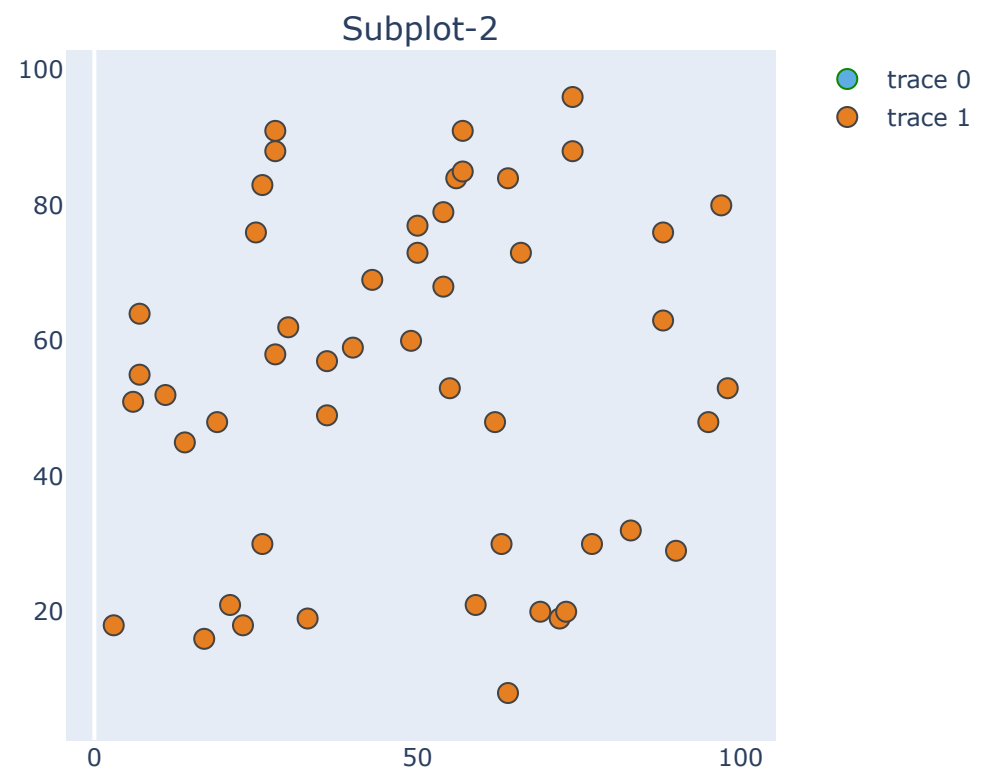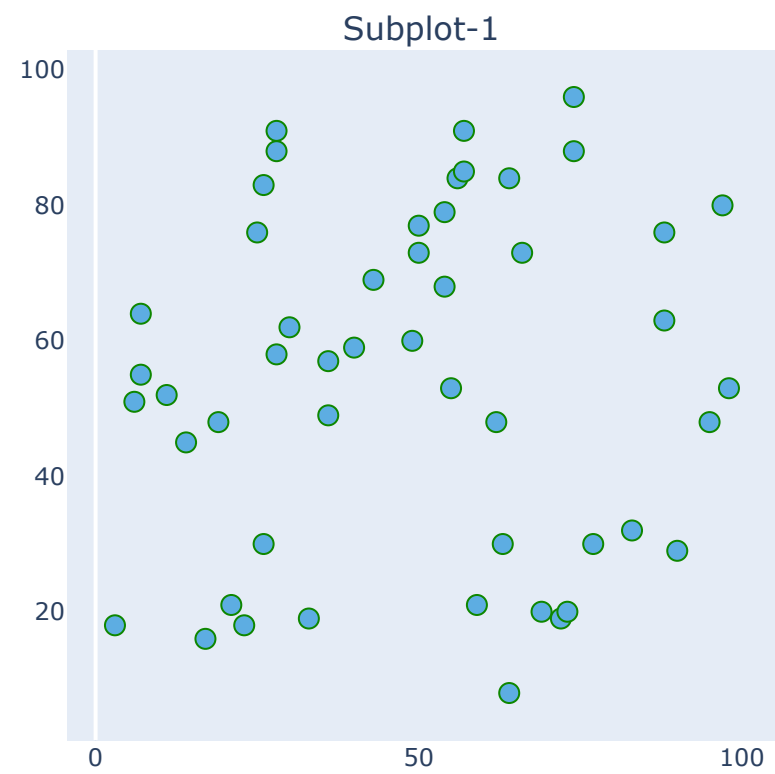
Scatter Plot using Plotly Express

```
In [44]: xval = np.random.normal(0,10,1000)
         yval = np.random.normal(0,10,1000)
         fig = px.scatter(x=xval,y= yval)

         fig.update_layout(
                         xaxis=dict(
                                 showgrid=False, # Hide Gridlines
                                 showline=False, # Hide X-Axis
                                 ),

                         yaxis=dict(
                                 showgrid=False, # Hide Gridlines
                                 showline=False, # Hide X-Axis
                                 ),
                         )

         fig.show()
```
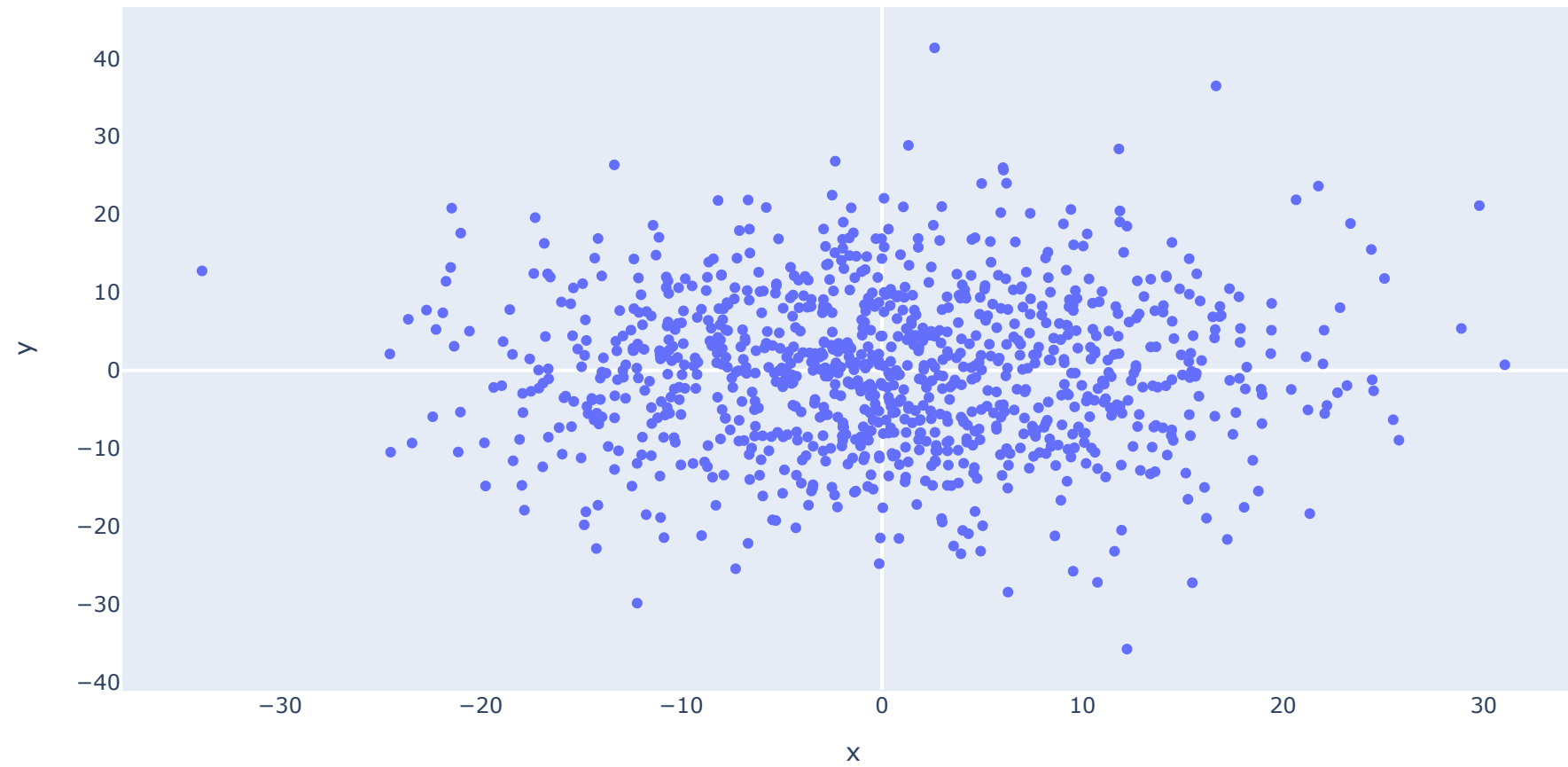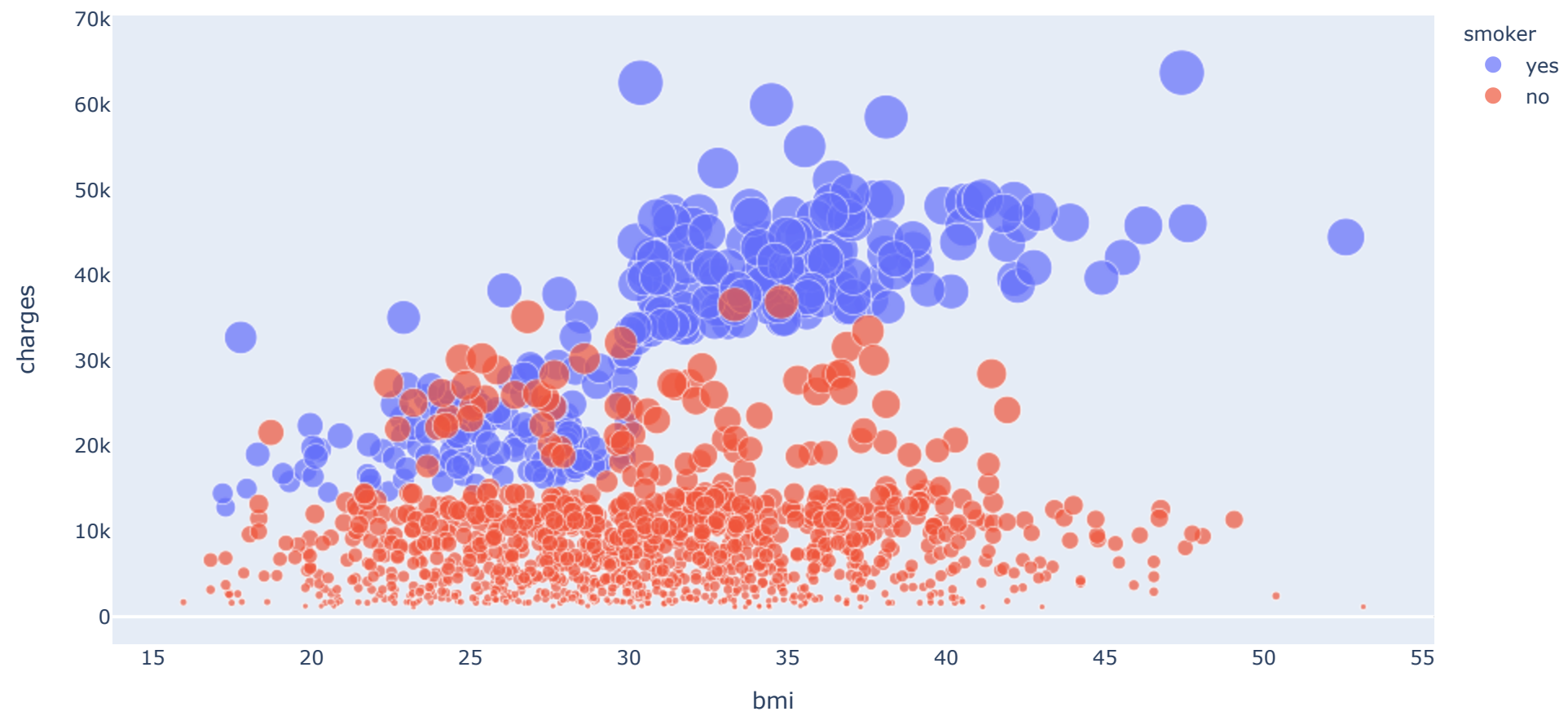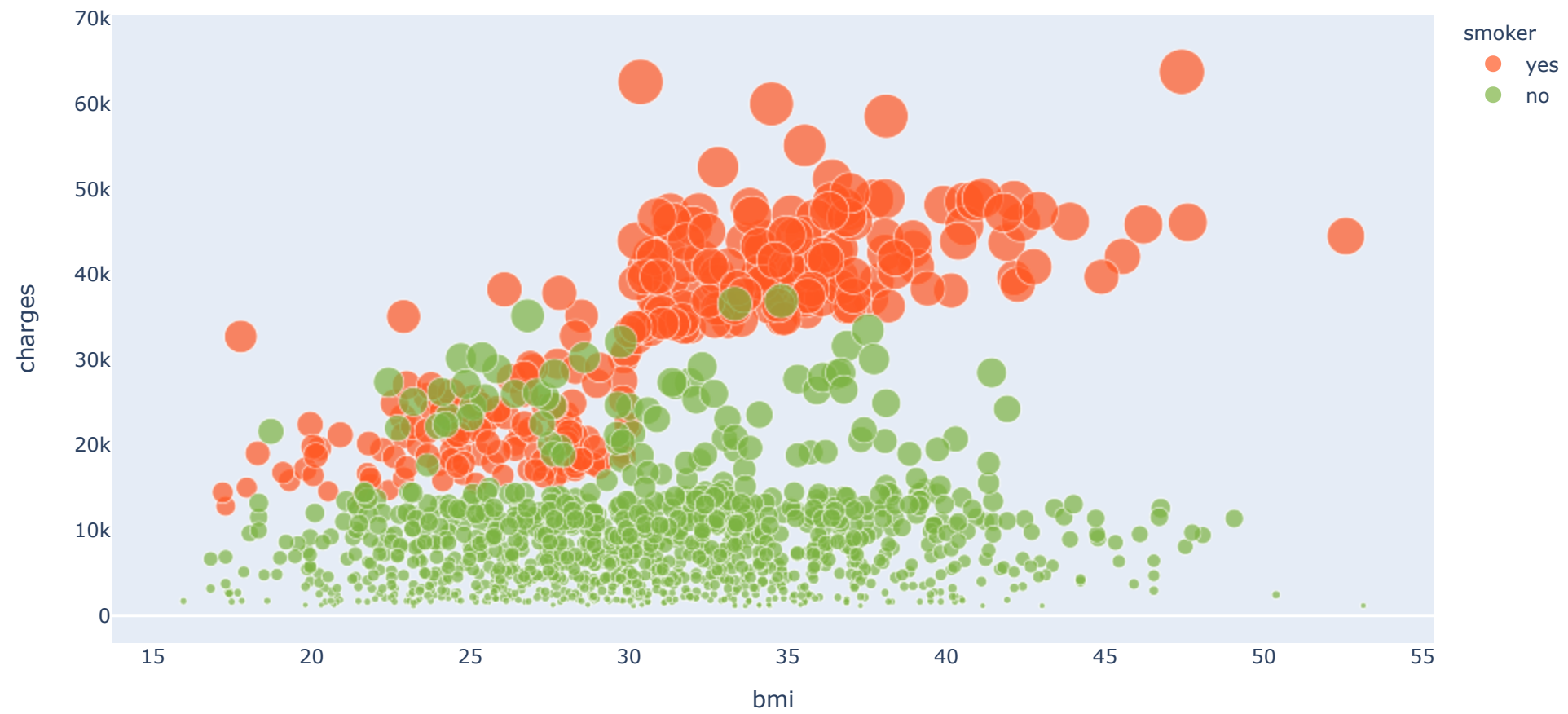
```
In [57]:   fig = px.scatter(
                       insurance,
                       x=insurance.bmi,
                       y= insurance.charges,
                       color="smoker", # Show groups with different colors using "color" parameter
                       size=insurance.charges
                   )

           fig.update_layout(
                       xaxis=dict(
                               showgrid=False, # Hide Gridlines
                               showline=False, # Hide X-Axis
                           ),

                       yaxis=dict(
                               showgrid=False, # Hide Gridlines
                               showline=False, # Hide X-Axis
                           ),
                   )


           fig.show()
```

```
fig = px.scatter(insurance,
                 x=insurance.bmi,
                 y= insurance.charges,
                 color="smoker",  # Show groups with different colors using "color" parameter
                 size=insurance.charges,
                 color_discrete_map={"yes": "#FF5722","no": "#7CB342"} #Map colors to data values
                )

fig.update_layout(
                 xaxis=dict(
                         showgrid=False, # Hide Gridlines
                         showline=False, # Hide X-Axis
                     ),

                 yaxis=dict(
                         showgrid=False, # Hide Gridlines
                         showline=False, # Hide X-Axis
                     ),
                 )


fig.show()
```

```python
# Using facet_col arguments to create Sub plots
fig = px.scatter(insurance,
                 x=insurance.bmi,
                 y=insurance.charges,
                 color=insurance.smoker,    #  Show groups with different colors using "color" parameter
                 facet_col=insurance.region, # Using facet_col arguments to create Sub plots
                 color_discrete_map={"yes": "#FF5722","no": "#7CB342"}, #Map colors to data values
                 title="Insurance Data"
                 )

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```

```python
In [54]:   # Using facet_row and or facet_col arguments to create Sub plots
           fig = px.scatter(insurance,
                            x=insurance.bmi,
                            y=insurance.charges,
                            color=insurance.smoker,
                            facet_col=insurance.region, # Using facet_col argument to create Sub plots
                            facet_row=insurance.sex, # Using facet_row argument to create Sub plots
                            color_discrete_map={"yes": "#FF5722","no": "#7CB342"},
                            width=950,
                            height=800,
                            title="Insurance Data")

           fig.update_layout(
                            plot_bgcolor= "#dcedc1",
                            paper_bgcolor="#FFFDE7",
                            )

           # Hide grid lines
           fig.update_xaxes(showgrid=False)
           fig.update_yaxes(showgrid=False)

           fig.show()
```

Insurance Data
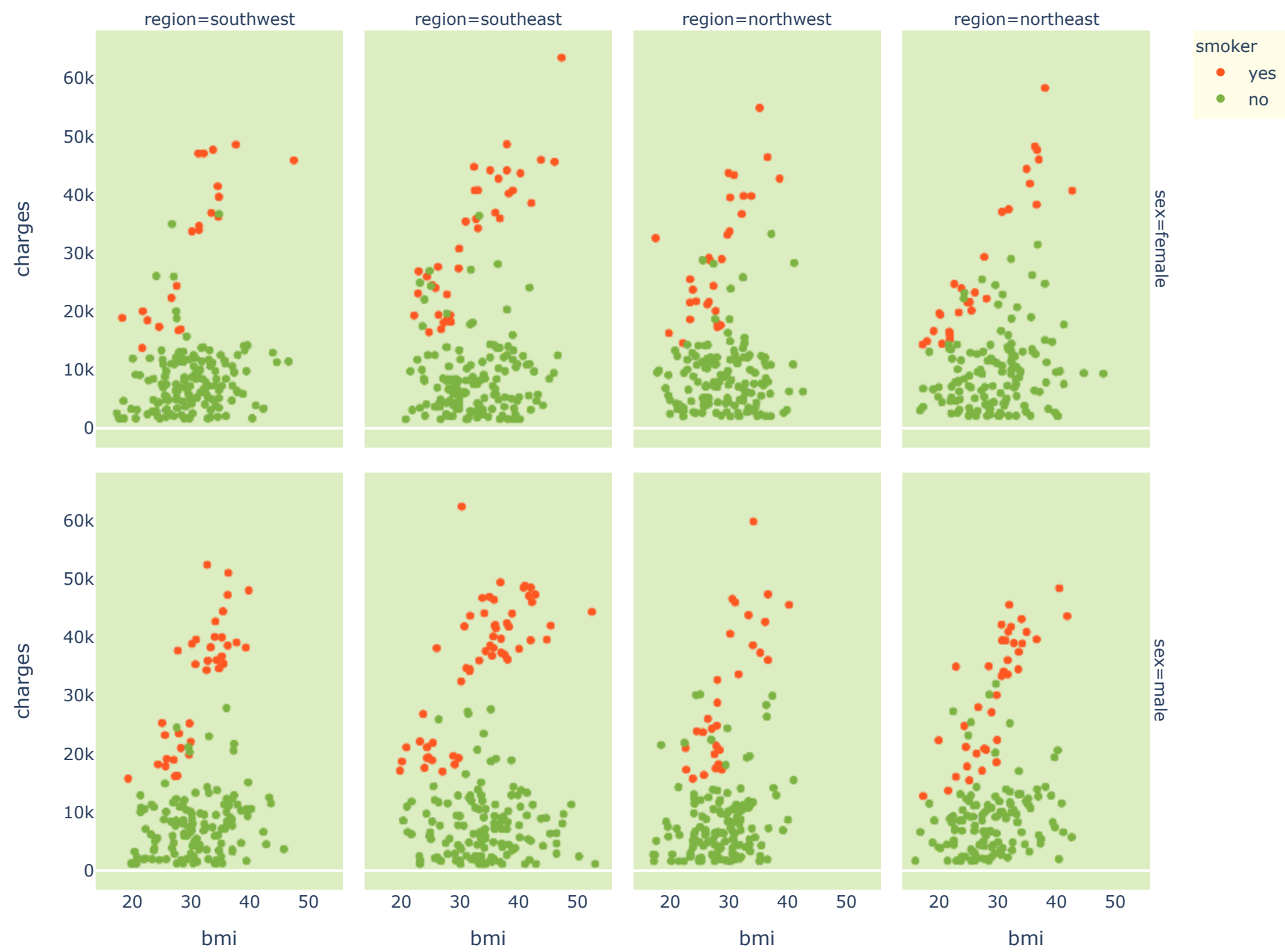
```python
In [55]:  # Using facet_row and or facet_col arguments to create Sub plots
          fig = px.scatter(insurance,
                           x=insurance.bmi,
                           y=insurance.charges,
                           color=insurance.smoker,
                           facet_col=insurance.region,
                           facet_row=insurance.sex,
                           color_discrete_map={"yes": "#FF5722","no": "#7CB342"},
                           width=950,
                           height=800,
                           title="Insurance Data")

          fig.update_layout(
                           plot_bgcolor= "#dcedc1",
                           paper_bgcolor="#FFFDE7",
                           )

          # Updating Traces using "selector" argument
          fig.update_traces(
                           marker_color="#339900",
                           selector=dict(marker_color="#7CB342")
                           )

          # Updating Traces using "selector" argument
          fig.update_traces(
                           marker_color="#FF3333",
                           selector=dict(marker_color="#FF5722")
                           )
          # Hide grid lines
          fig.update_xaxes(showgrid=False)
          fig.update_yaxes(showgrid=False)

          fig.show()
          #po.plot(fig)
```

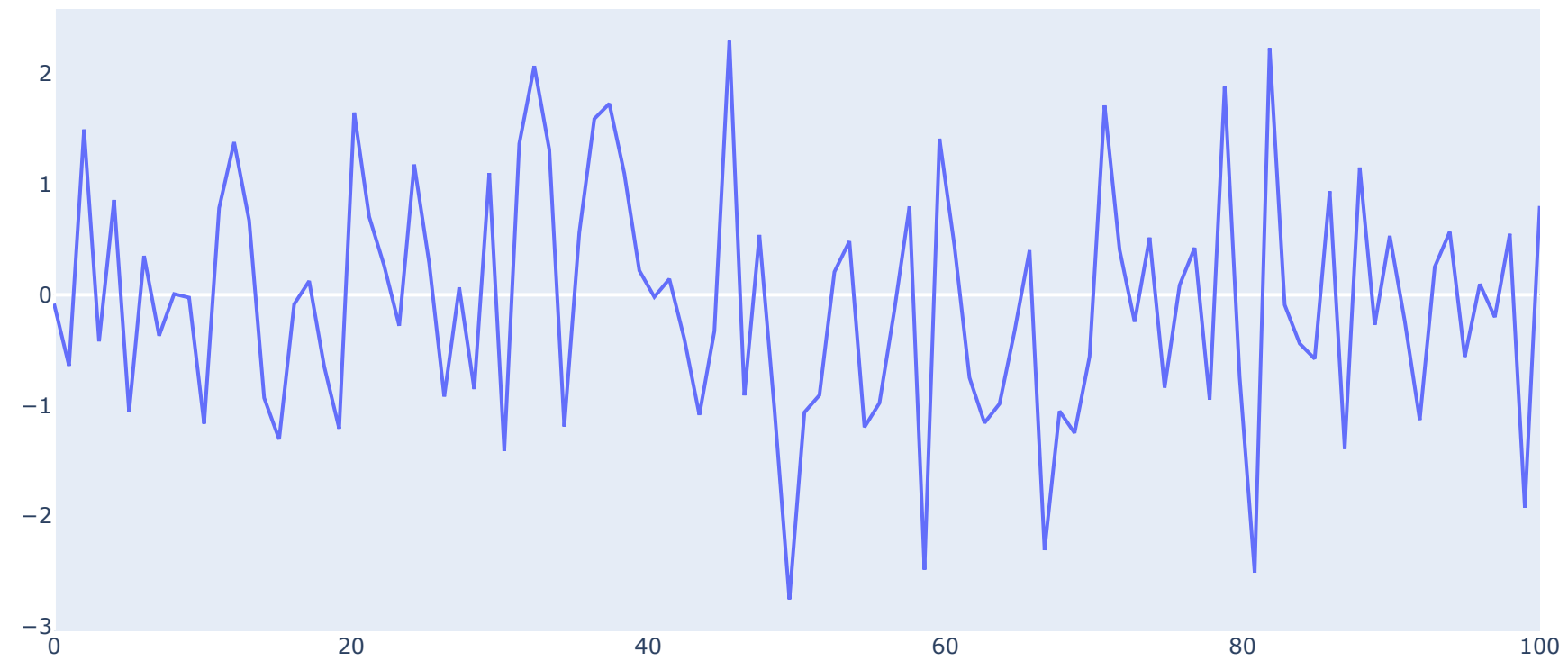Insurance Data
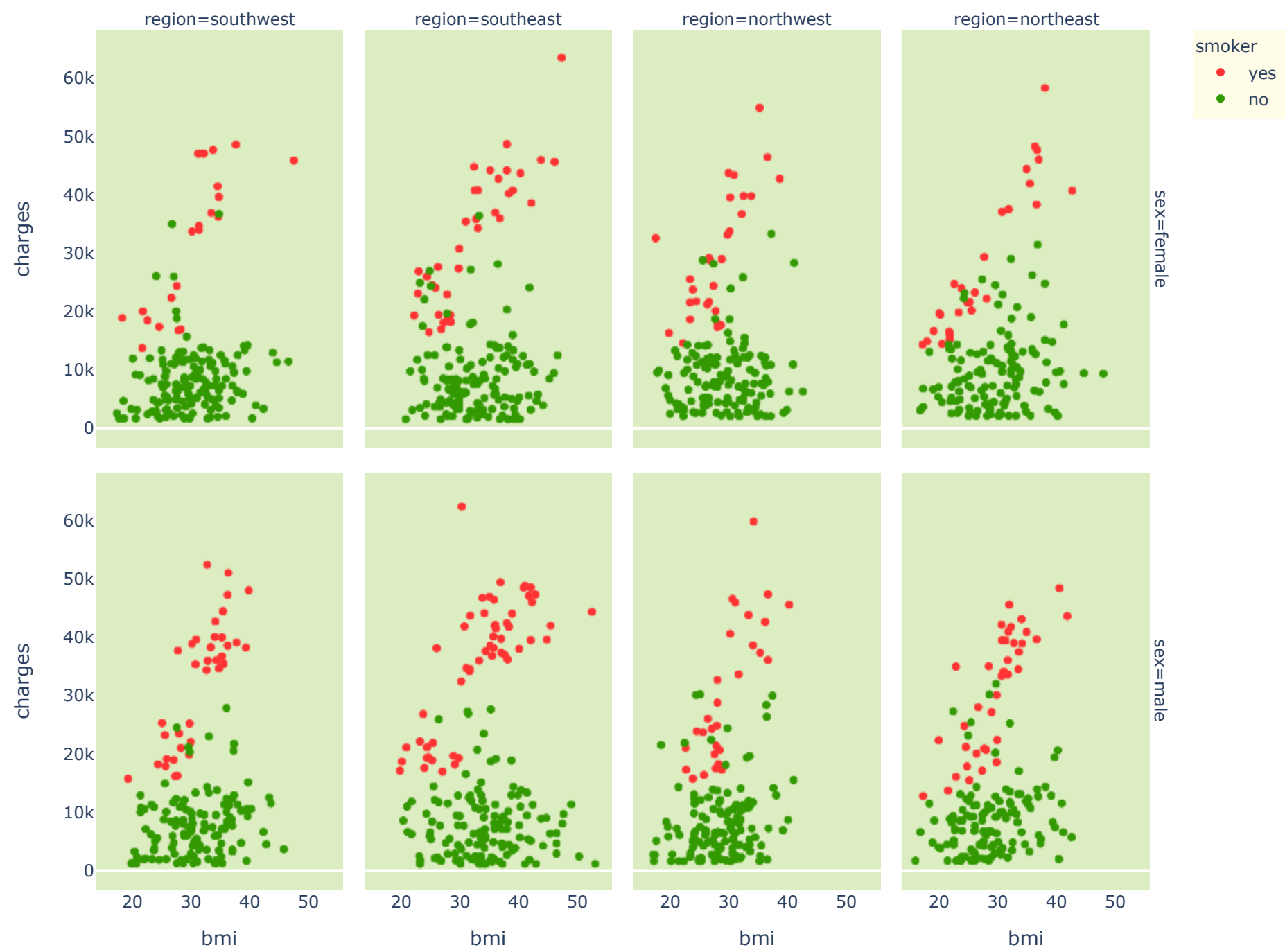
Line Plot

```
In [59]: #Simple Line Plot

x_values = np.linspace(0, 100, 100) # 100 evenly spaced values
y_values = np.random.randn(100)     # 100 random values

# create traces
trace0 = go.Scatter(
                    x = x_values,
                    y = y_values,
                    mode = 'lines',
                    )

layout = go.Layout(
                    xaxis=dict(
                            showgrid=False, # Hide Gridlines
                            showline=False, # Hide X-Axis
                            ),

                    yaxis=dict(
                            showgrid=False, # Hide Gridlines
                            showline=False, # Hide X-Axis
                            ),
                    )

fig = go.Figure(data=trace0,layout=layout)
iplot(fig)
```

```
In [60]:  #Line Styling

          x_values = np.linspace(0, 100, 100) # 100 evenly spaced values
          y_values = np.random.randn(100)   # 100 random values

          trace0 = go.Scatter(
                          x = x_values,
                          y = y_values,
                          mode = 'lines+markers',
                          )
          trace1 = go.Scatter(
                          x = x_values,
                          y = y_values-5,
                          mode = 'lines',
                          )
          data = [trace0, trace1]
          fig = go.Figure(data=data,layout=layout)
          iplot(fig)
```
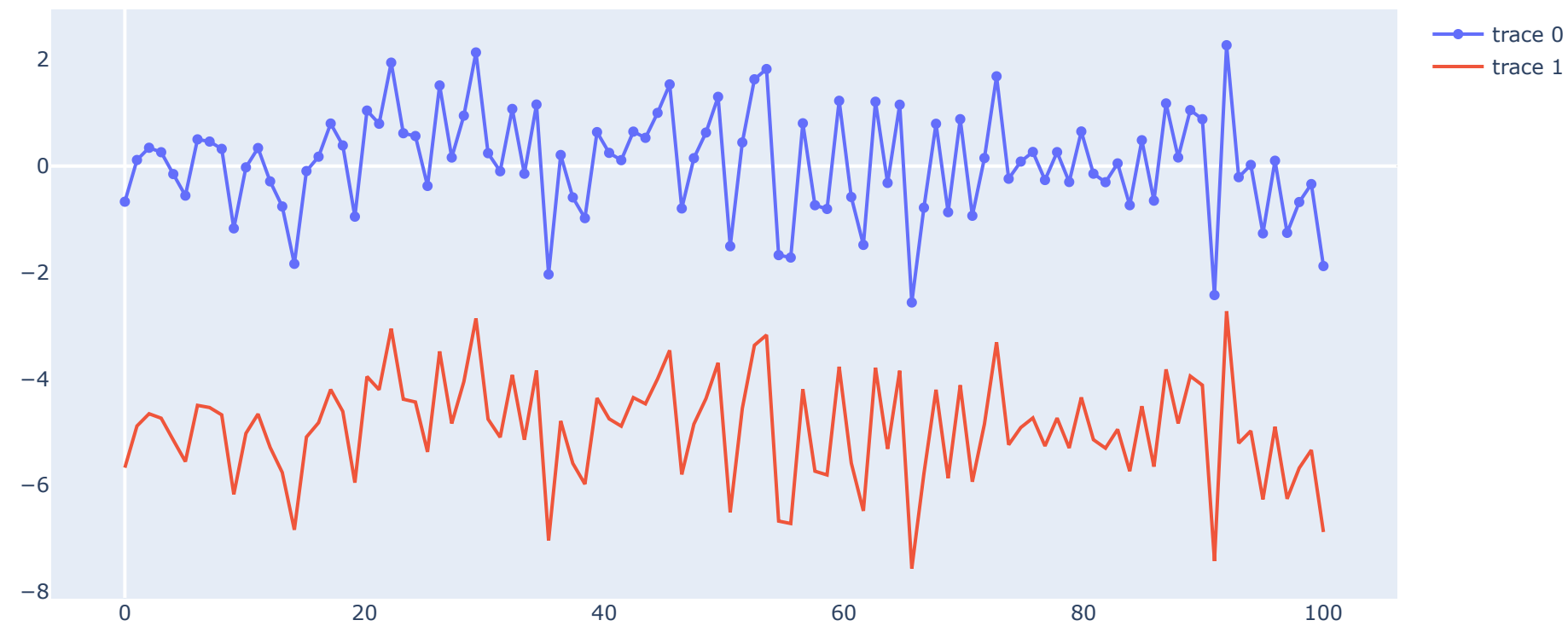
```
In [350]:  canada.head()
```

Out[350]:

	Afghanistan	Albania	Algeria	American Samoa	Andorra	Angola	Antigua and Barbuda	Argentina	Armenia	Australia	...	Uzbekistan	Vanuatu	Venezuela (Bolivarian Republic of)	Viet Nam	Western Sahara	Yemen	Zambia	Zimbabwe	Unknown	Total
1980	16	1	80	0	0	1	0	368	0	702	...	0	0	103	1191	0	1	11	72	44000	143137
1981	39	0	67	1	0	3	0	426	0	639	...	0	0	117	1829	0	2	17	114	18078	128641
1982	39	0	71	0	0	6	0	626	0	484	...	0	0	174	2162	0	1	11	102	16904	121175
1983	47	0	69	0	0	6	0	241	0	317	...	0	0	124	3404	0	6	7	44	13635	89185
1984	71	0	63	0	0	4	42	237	0	317	...	0	0	142	7583	0	0	16	32	14855	88272

5 rows × 197 columns

```
In [61]:  # Plot Immigrants from China
          china = go.Scatter(
                              x = canada.index.values,
                              y = canada['China'],
                              mode = 'lines',
                              name = 'China'
                             )

          #Plot Immigrants from India
          india = go.Scatter(
                              x = canada.index.values,
                              y = canada['India'],
                              mode = 'lines',
                              name = 'India'
                            )

          #Plot Immigrants from Pakistan
          pakistan = go.Scatter(
                                 x = canada.index.values,
                                 y = canada['Pakistan'],
                                 mode = 'lines',
                                 name = 'Pakistan'
                               )

          # Layout setting
          layout = go.Layout(
                              title = '$Immigrants$', # Title
                              xaxis = dict(title = 'Year',showgrid=False,showline=False), # x-axis label
                              yaxis = dict(title = 'Number of Immigrants',showgrid=False,showline=False), # y-axis label

                             )

          data = [china, india,pakistan]
          fig = go.Figure(data=data,layout=layout)
          iplot(fig)
```
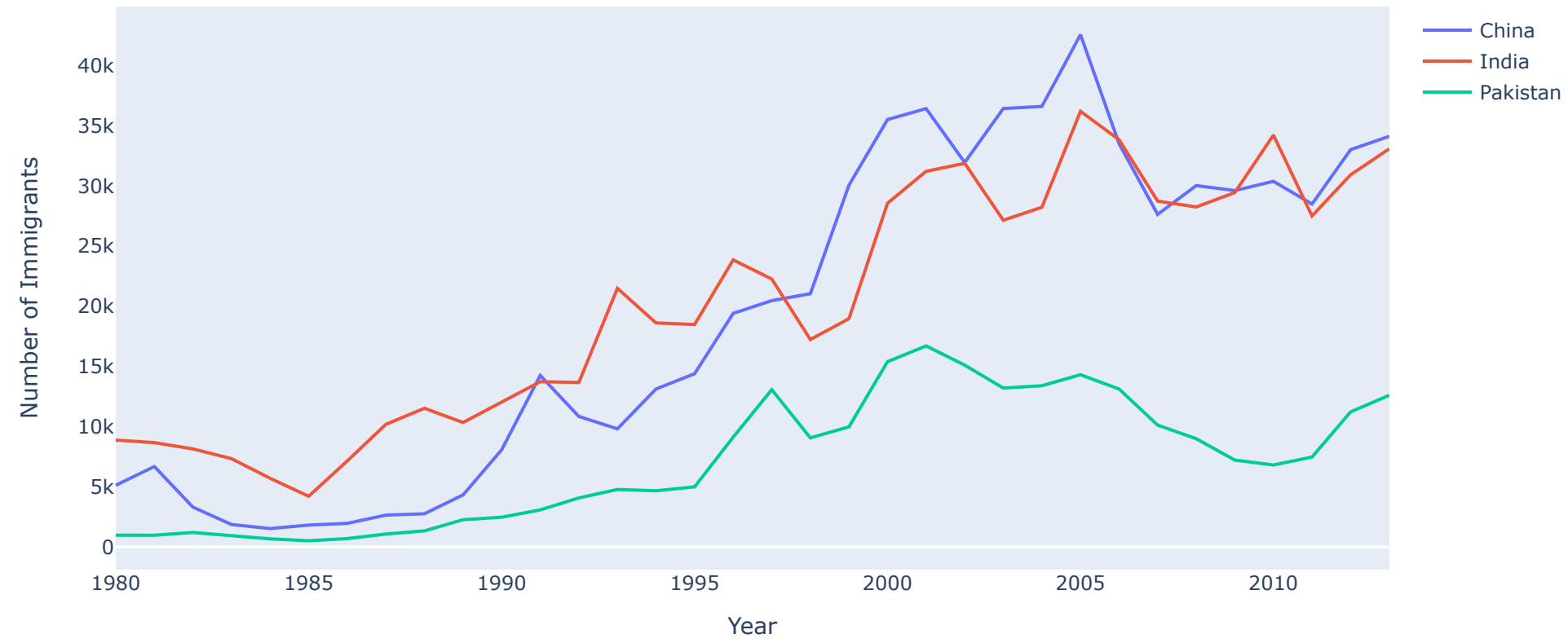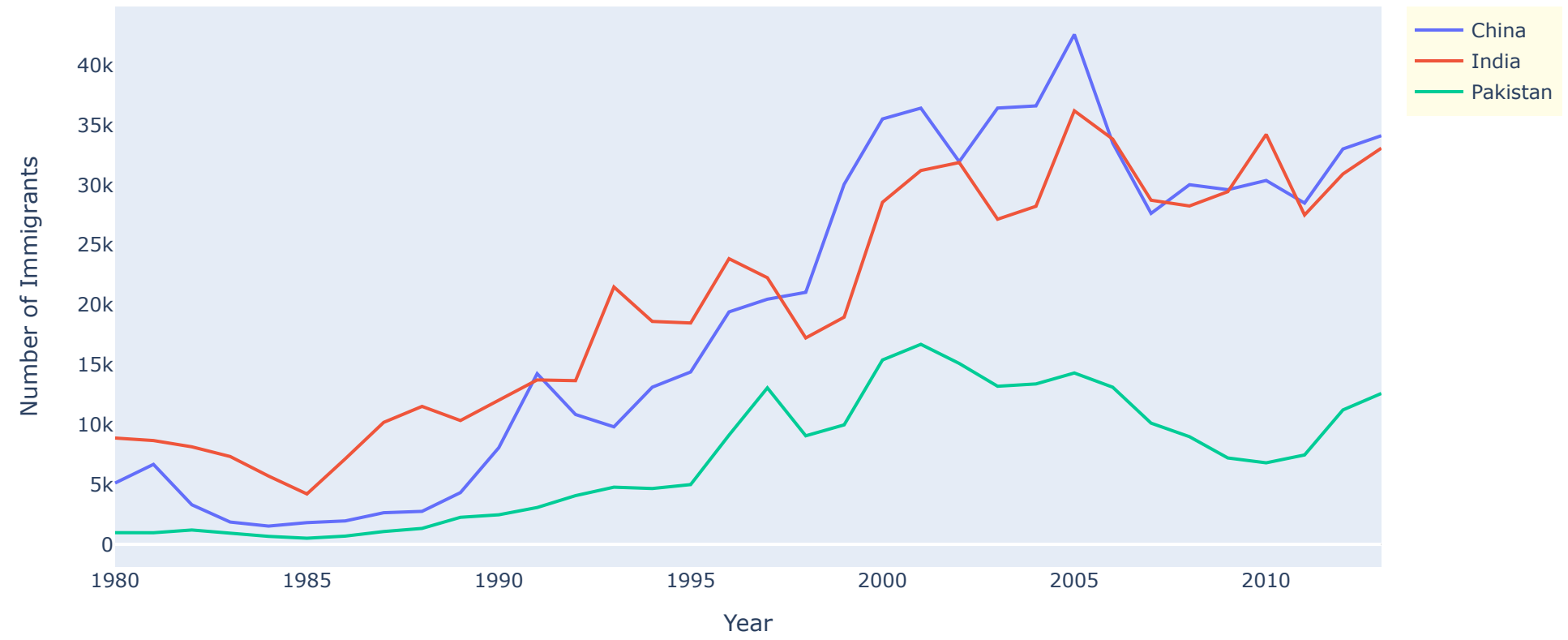
```
In [63]:   # Plot Immigrants from China
           china = go.Scatter(
                              x = canada.index.values,
                              y = canada['China'],
                              mode = 'lines',
                              name = 'China'
                          )

           # Plot Immigrants from India
           india = go.Scatter(
                              x = canada.index.values,
                              y = canada['India'],
                              mode = 'lines',
                              name = 'India'
                          )

           # Plot Immigrants from Pakistan
           pakistan = go.Scatter(
                                 x = canada.index.values,
                                 y = canada['Pakistan'],
                                 mode = 'lines',
                                 name = 'Pakistan'
                              )


           layout = go.Layout(
                              title = '$Immigrants$', # Title
                              xaxis = dict(title = 'Year',showgrid=False,showline=False), # x-axis label
                              yaxis = dict(title = 'Number of Immigrants',showgrid=False,showline=False), # y-axis label
                              paper_bgcolor= '#FFFDE7' # Paper background color

                          )

           data = [china, india,pakistan]
           fig = go.Figure(data=data,layout=layout)
           iplot(fig)
```

Immigrants

```
In [64]:    # Plot Immigrants from China
            china = go.Scatter(
                                x = canada.index.values,
                                y = canada['China'],
                                mode = 'lines',
                                name = 'China'
                              )

            # Plot Immigrants from India
            india = go.Scatter(
                                x = canada.index.values,
                                y = canada['India'],
                                mode = 'lines',
                                name = 'India'
                              )

            # Plot Immigrants from Pakistan
            pakistan = go.Scatter(
                                    x = canada.index.values,
                                    y = canada['Pakistan'],
                                    mode = 'lines',
                                    name = 'Pakistan'
                                 )


            layout = go.Layout(
                                title=dict(text = "Immigration Data",x=0.5,y=0.95),
                                xaxis = dict(title = 'Year',showgrid=False,showline=False), # x-axis label
                                yaxis = dict(title = 'Number of Immigrants',showgrid=False,showline=False), # y-axis label
                                paper_bgcolor= '#FFFDE7',
                                plot_bgcolor= '#FFFDE7'
                              )

            data = [china, india,pakistan]
            fig = go.Figure(data=data,layout=layout)
            iplot(fig)
```

Immigration Data

Number of Immigrants

Year

Legend: China, India, Pakistan
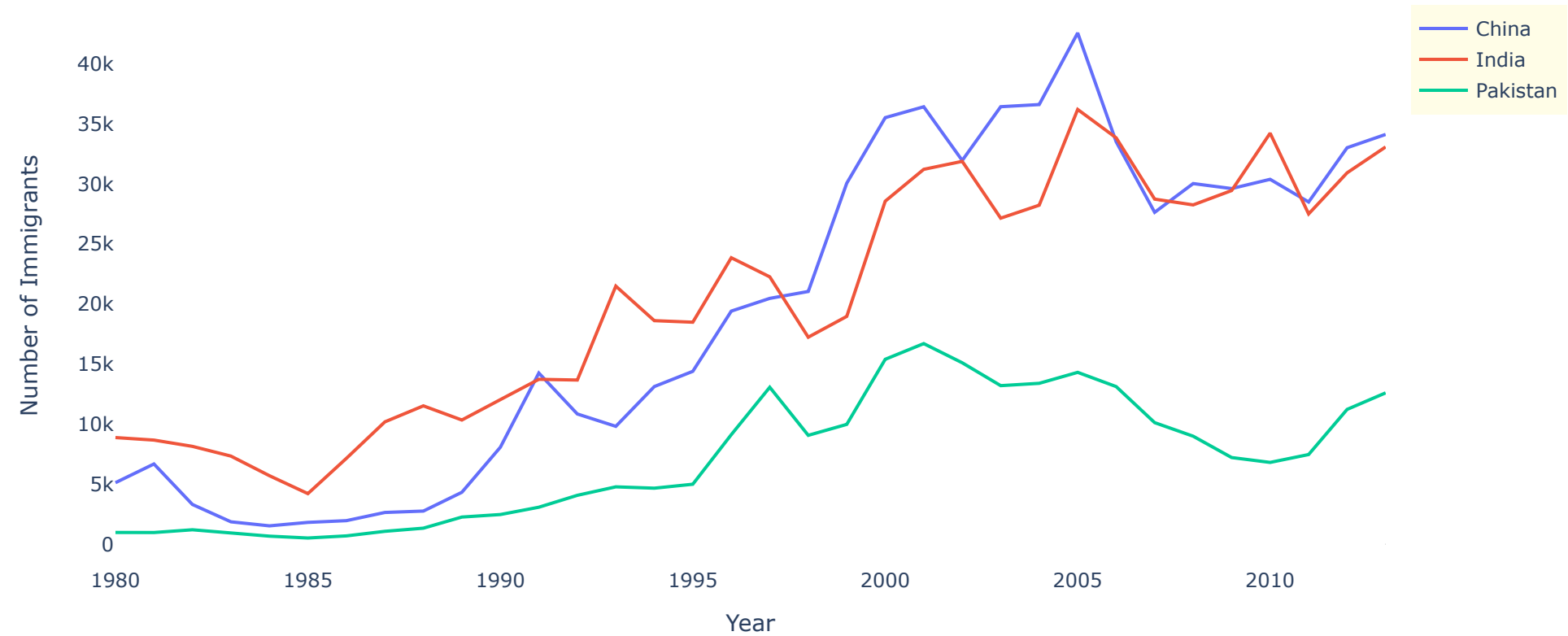
```
In [65]: canada1 = canada.copy()
         del canada1['Unknown']
         del canada1['Total']
         canada1.head()
```

Out[65]:

	Afghanistan	Albania	Algeria	American Samoa	Andorra	Angola	Antigua and Barbuda	Argentina	Armenia	Australia	...	United States of America	Uruguay	Uzbekistan	Vanuatu	Venezuela (Bolivarian Republic of)	Viet Nam	Western Sahara	Yemen	Zambia	Zimbabwe
1980	16	1	80	0	0	1	0	368	0	702	...	9378	128	0	0	103	1191	0	1	11	72
1981	39	0	67	1	0	3	0	426	0	639	...	10030	132	0	0	117	1829	0	2	17	114
1982	39	0	71	0	0	6	0	626	0	484	...	9074	146	0	0	174	2162	0	1	11	102
1983	47	0	69	0	0	6	0	241	0	317	...	7100	105	0	0	124	3404	0	6	7	44
1984	71	0	63	0	0	4	42	237	0	317	...	6661	90	0	0	142	7583	0	0	16	32

5 rows × 195 columns

```
In [66]: canada.head()
```

Out[66]:

	Afghanistan	Albania	Algeria	American Samoa	Andorra	Angola	Antigua and Barbuda	Argentina	Armenia	Australia	...	Uzbekistan	Vanuatu	Venezuela (Bolivarian Republic of)	Viet Nam	Western Sahara	Yemen	Zambia	Zimbabwe	Unknown	Total
1980	16	1	80	0	0	1	0	368	0	702	...	0	0	103	1191	0	1	11	72	44000	143137
1981	39	0	67	1	0	3	0	426	0	639	...	0	0	117	1829	0	2	17	114	18078	128641
1982	39	0	71	0	0	6	0	626	0	484	...	0	0	174	2162	0	1	11	102	16904	121175
1983	47	0	69	0	0	6	0	241	0	317	...	0	0	124	3404	0	6	7	44	13635	89185
1984	71	0	63	0	0	4	42	237	0	317	...	0	0	142	7583	0	0	16	32	14855	88272

5 rows × 197 columns

```
In [68]:  #Immigrants from all countires using for loop
          traces = [] # Initiate trace
          for i in canada1.columns:
              traces.append(
                  go.Scatter(
                              x=canada1.index.values,
                              y=canada1[i],
                              mode='lines',
                              name = i,
                              connectgaps=True,
                          )
                      )

          layout = go.Layout(
                              title = 'Immigrants', # Title
                              title_font=dict(size=20),
                              xaxis = dict(title = 'Year',showgrid=False,showline=False), # x-axis label
                              yaxis = dict(title = 'Number of Immigrants',showgrid=False,showline=False), # y-axis label
                              font=dict(size=10),
                              width=1230,
                              height=650
                              )

          fig = go.Figure(data=traces, layout=layout)
          fig.show()
```

Immigrants

Legend:
- Afghanistan
- Albania
- Algeria
- American Samoa
- Andorra
- Angola
- Antigua and Barbuda
- Argentina
- Armenia
- Australia
- Austria
- Azerbaijan
- Bahamas
- Bahrain
- Bangladesh
- Barbados
- Belarus
- Belgium
- Belize
- Benin
- Bhutan
- Bolivia (Plurinational State of)
- Bosnia and Herzegovina
- Botswana
- Brazil

X-axis: Year (1980–2010+)
Y-axis: Number of Immigrants (0 – 40k)
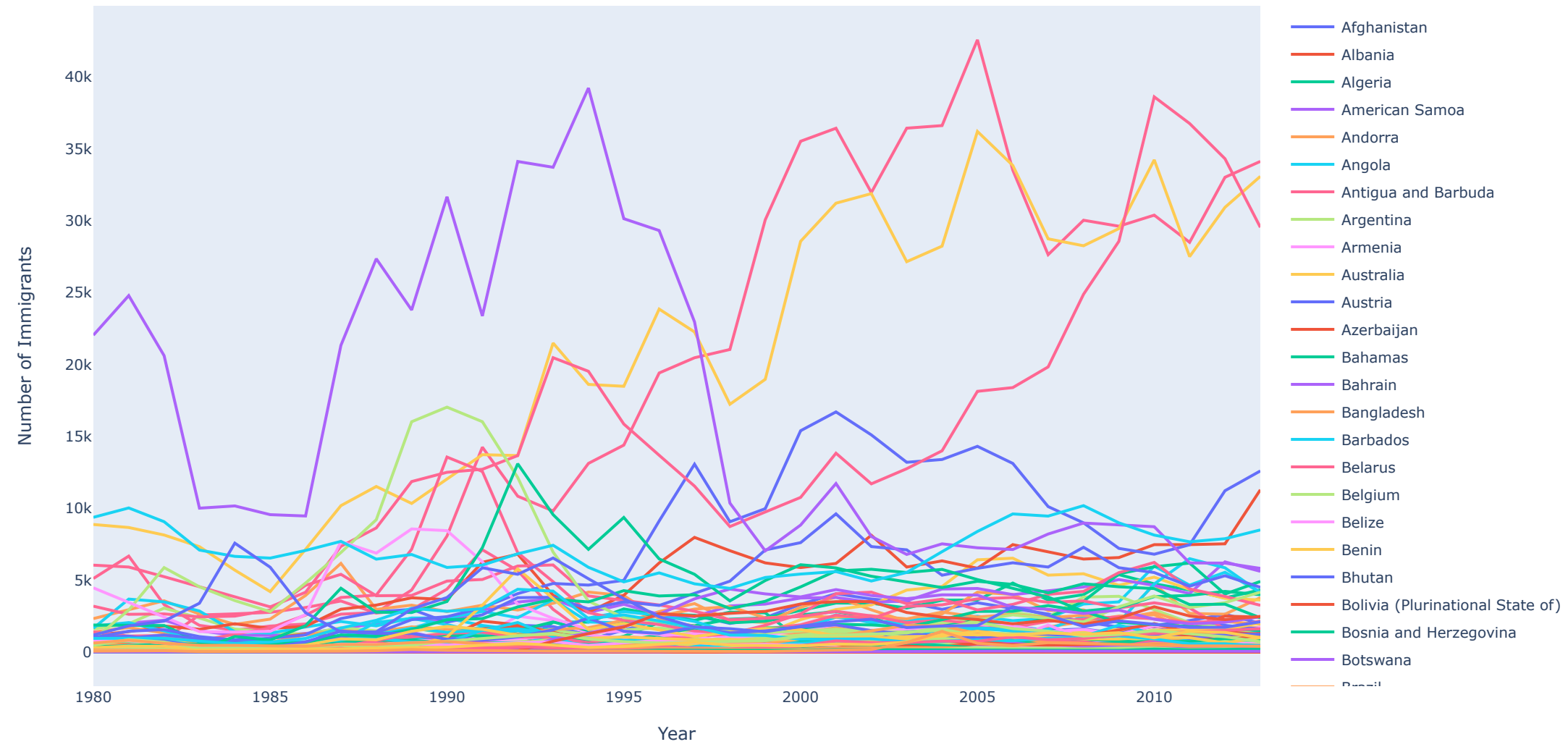
```
In [71]:    from plotly.subplots import make_subplots


            #Subplot initialization
            fig = make_subplots(
                            rows=1,
                            cols=2,
                            subplot_titles=("Immigration Data", "Insurance Data")
                          )


            # Subplot - 1 (Add graph object trace to a figure)
            fig.add_trace(go.Scatter(
                            x = canada.index.values,
                            y = canada['China'],
                            mode = 'lines',
                            name = 'China'
                            ),
                      row=1, col=1
                    )

            # Add graph object trace to a figure
            fig.add_trace(go.Scatter(
                            x = canada.index.values,
                            y = canada['India'],
                            mode = 'lines',
                            name = 'India'
                            ),
                      row=1, col=1
                    )

            # Add graph object trace to a figure
            fig.add_trace(go.Scatter(
                            x = canada.index.values,
                            y = canada['Pakistan'],
                            mode = 'lines',
                            name = 'Pakistan'
                            ),
                      row=1, col=1
                    )


            # Subplot - 2 (Add graph object trace to a figure)
            fig.add_trace(go.Scatter(
                            x = insurance[insurance.smoker=='yes'].bmi,
                            y = insurance[insurance.smoker=='yes'].charges,
                            mode = 'markers',
                            name = 'Smoker',
                            marker = dict(size = 7, color = '#F39C12',symbol = 'circle',line = dict(width = 1))
                            ),
                      row=1, col=2
                    )

            # Add graph object trace to a figure
            fig.add_trace(go.Scatter(
                            x = insurance[insurance.smoker=='no'].bmi,
                            y = insurance[insurance.smoker=='no'].charges,
                            mode = 'markers',
                            name = 'Non-Smoker',
                            marker = dict(size = 7, color = '#8BC34A',symbol = 'circle',line = dict(width = 1))
                            ),
```

```
            row=1, col=2
        )

# Changing X & Y Axis properties
fig.update_xaxes(title_text="Year", row=1, col=1,showgrid=False,showline=False)
fig.update_yaxes(title_text="Number of Immigrants", row=1, col=1,showgrid=False,showline=False)
fig.update_xaxes(title_text="BMI" ,row=1, col=2,showgrid=False,showline=False)
fig.update_yaxes(title_text="Charges", row=1, col=2,showgrid=False,showline=False)


#Changing plot & figure background
fig.update_layout(
                paper_bgcolor= '#FFFDE7',
                plot_bgcolor= '#FFFDE7',
                title=dict(text = "Sub Plots",x=0.5,y=0.95),
                title_font_size=30
            )
fig.show()
```
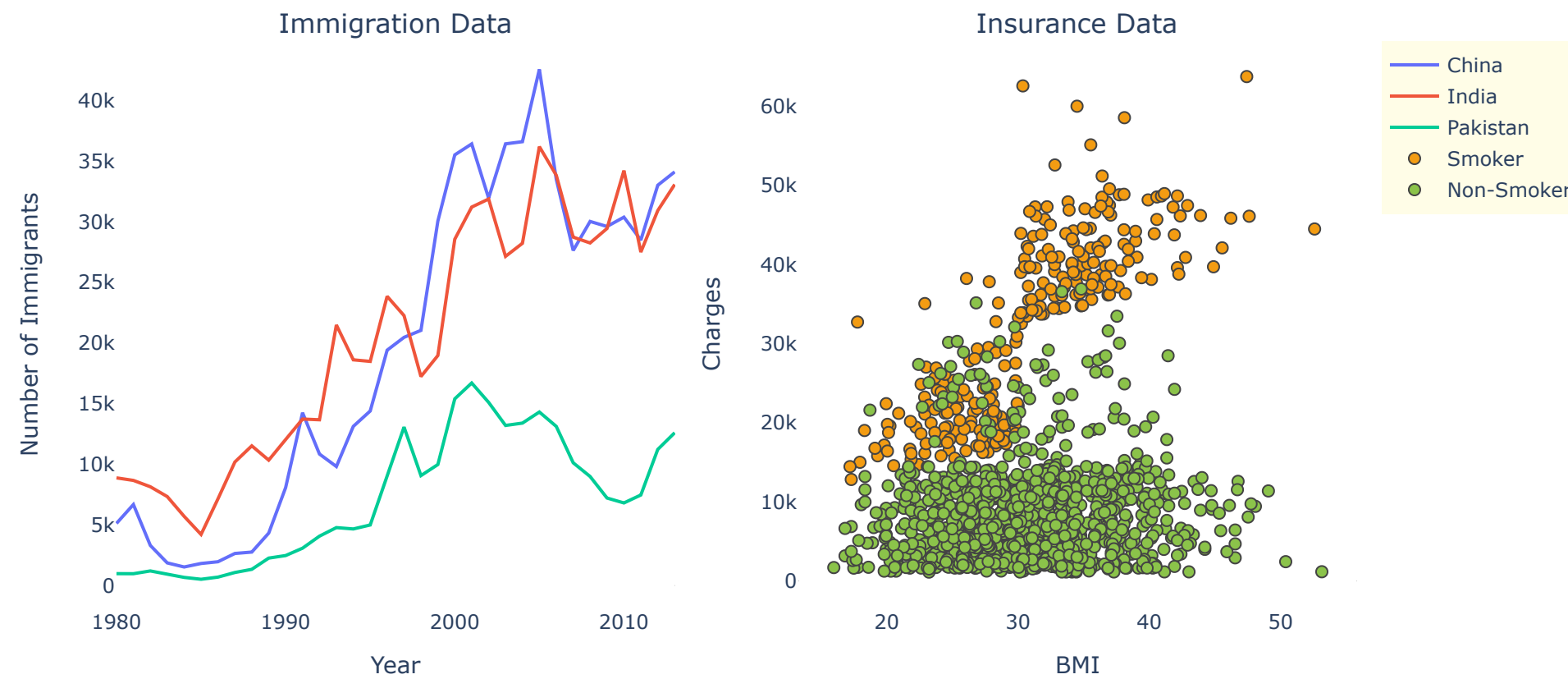

Line Plot using Plotly Express

```
In [72]: col1 = np.linspace(0, 10, 1000)
         col2 = np.sin(col1)
         fig = px.line(
                     x=col1,
                     y= col2,
                     title="Sine Plot"
                     )

         # Hide grid lines
         fig.update_xaxes(showgrid=False)
         fig.update_yaxes(showgrid=False)

         fig.show()
```

Sine Plot
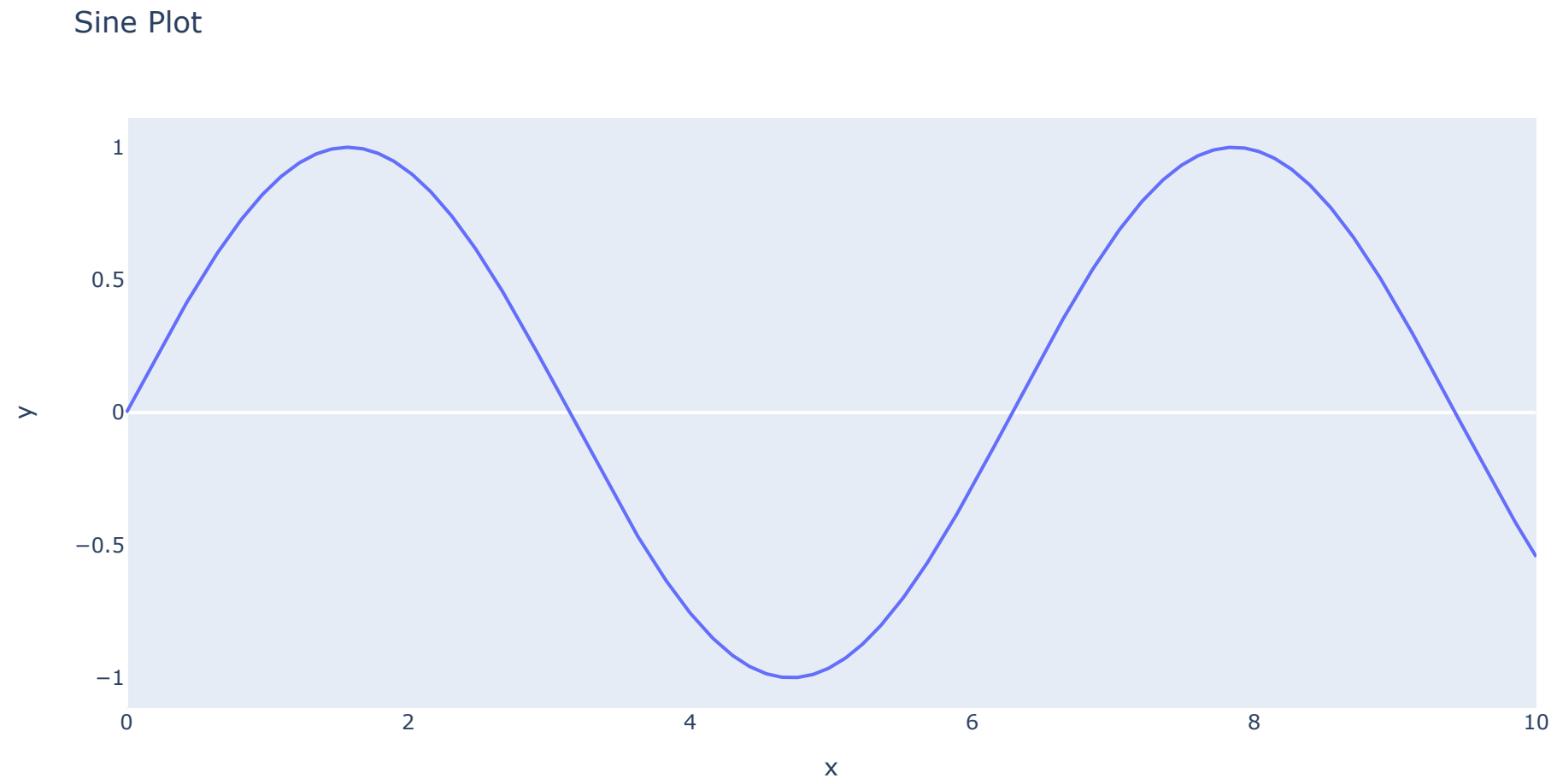
```
In [73]:  fig = px.line(
                        x=col1,
                        y= col2
                        )
          fig.add_scatter(x=col1,
                          y= col2+1
                          )

          # Hide grid lines
          fig.update_xaxes(showgrid=False)
          fig.update_yaxes(showgrid=False)

          fig.show()
```
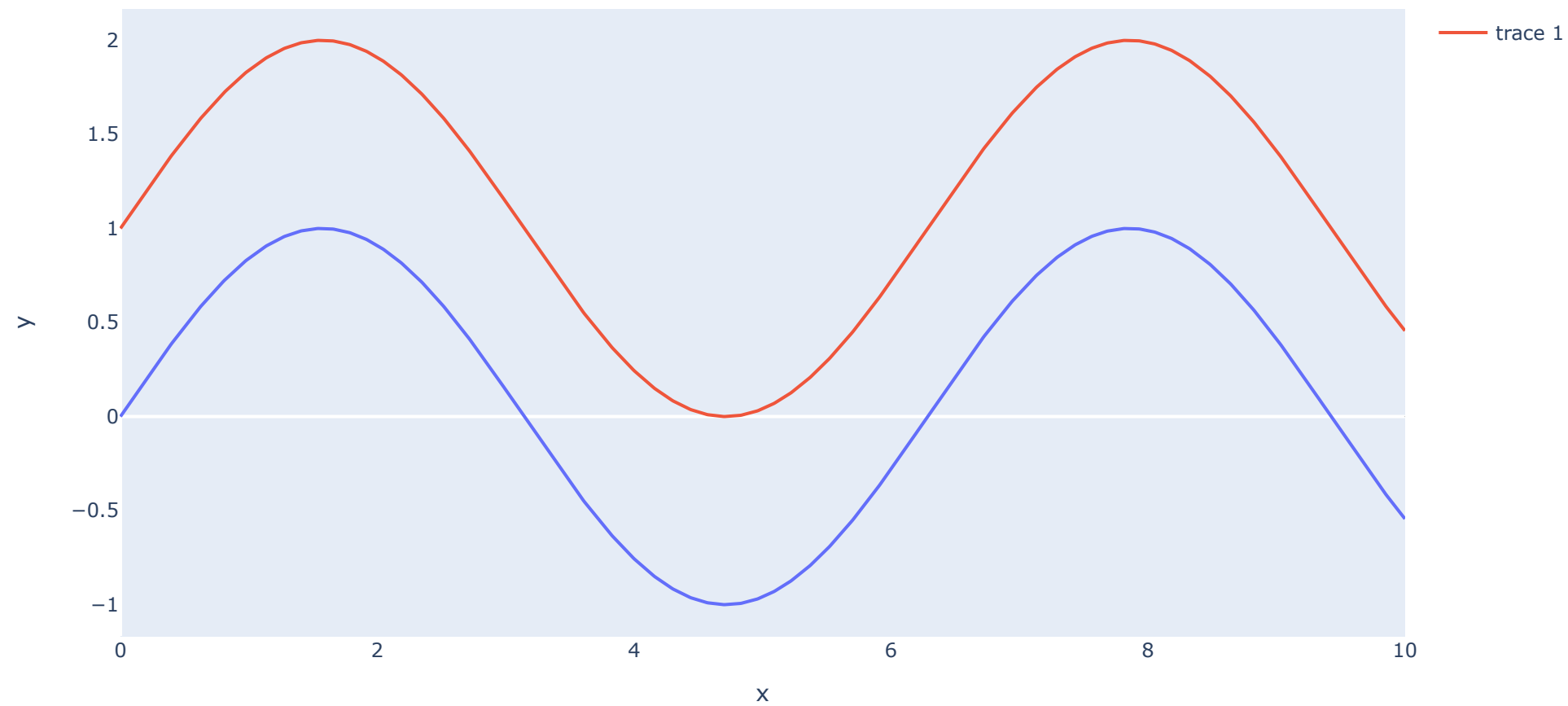
```
In [74]:  # Plot Immigrants from India
          fig = px.line(
                  x=canada1.index.values,
                  y= canada1['India']
                  )

          # Plot Immigrants from Pakistan
          fig.add_scatter(
                  x=canada1.index.values,
                  y= canada1['Pakistan'],
                  name = 'Pakistan',
                  line={'color': 'green'}
                  )

          # Plot Immigrants from Bangladesh
          fig.add_scatter(
                  x=canada1.index.values,
                  y= canada1['Bangladesh'] ,
                  name = 'Bangladesh',
                  line={'color': 'red'}
                  )

          fig.update_layout(title_text='Immigrants')

          # Hide grid lines
          fig.update_xaxes(showgrid=False)
          fig.update_yaxes(showgrid=False)

          fig.show()
```
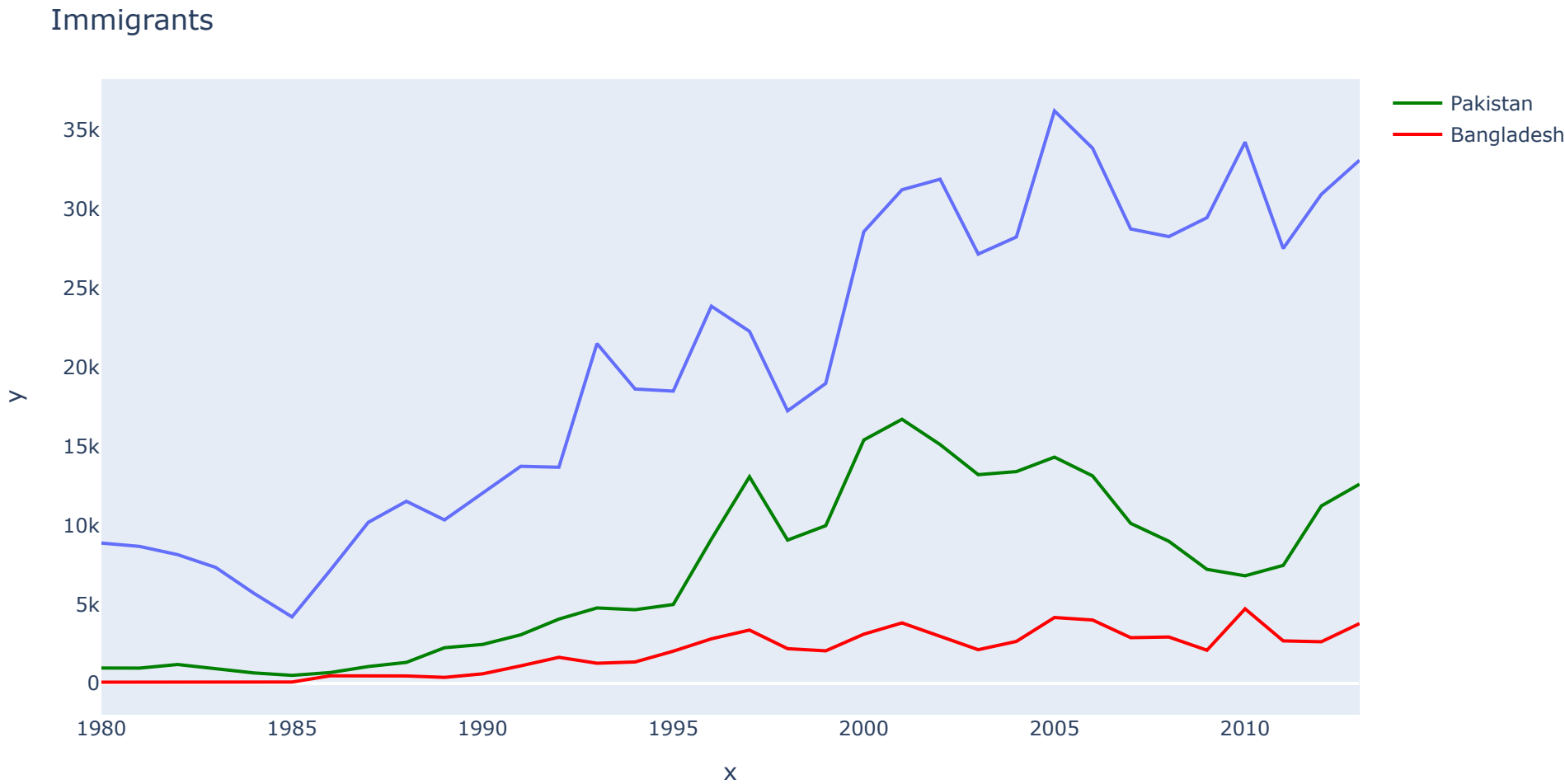
```
In [56]: fig = px.line(
             employment,
             x="Period" ,
             y = "Unemployed",
             color='Gender'
             )

         # Hide grid lines
         fig.update_xaxes(showgrid=False)
         fig.update_yaxes(showgrid=False)

         fig.show()
```
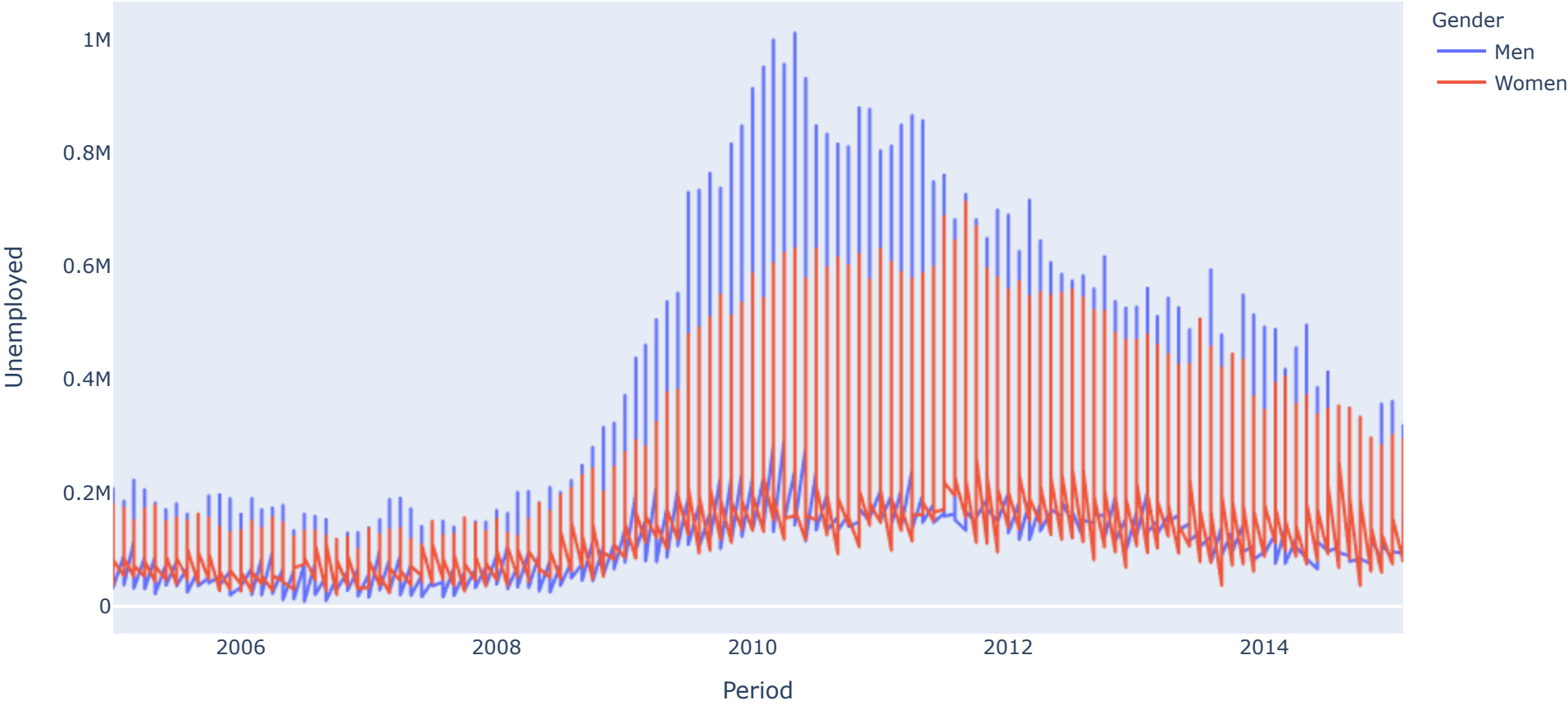

Bar Plot

```
In [75]:  # Use go.Bar to plot Bar charts in Plotly
          x = np.arange(1,10)
          y = np.arange(20,110,10)
          data = go.Bar(
                      x= x,
                      y= y,
                      )

          layout = go.Layout(
                          title = 'Simple Bar Chart',
                          xaxis=dict(
                                    showgrid=False, # Hide Gridlines
                                    showline=False, # Hide X-Axis
                                    ),

                          yaxis=dict(
                                    showgrid=False, # Hide Gridlines
                                    showline=False, # Hide X-Axis
                                    ),
                          )


          fig = go.Figure(data=data,layout=layout)
          fig.show()
```
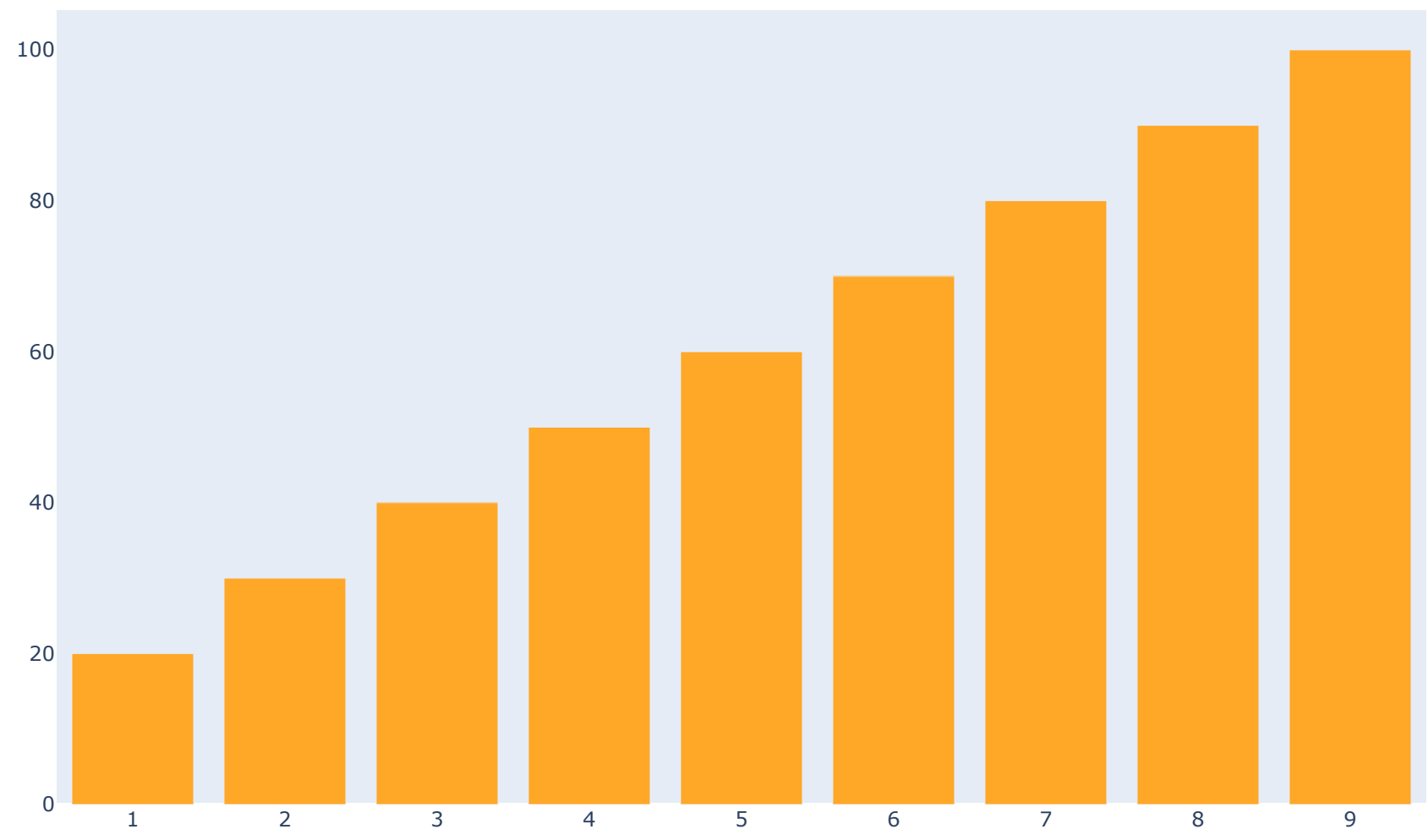
Simple Bar Chart
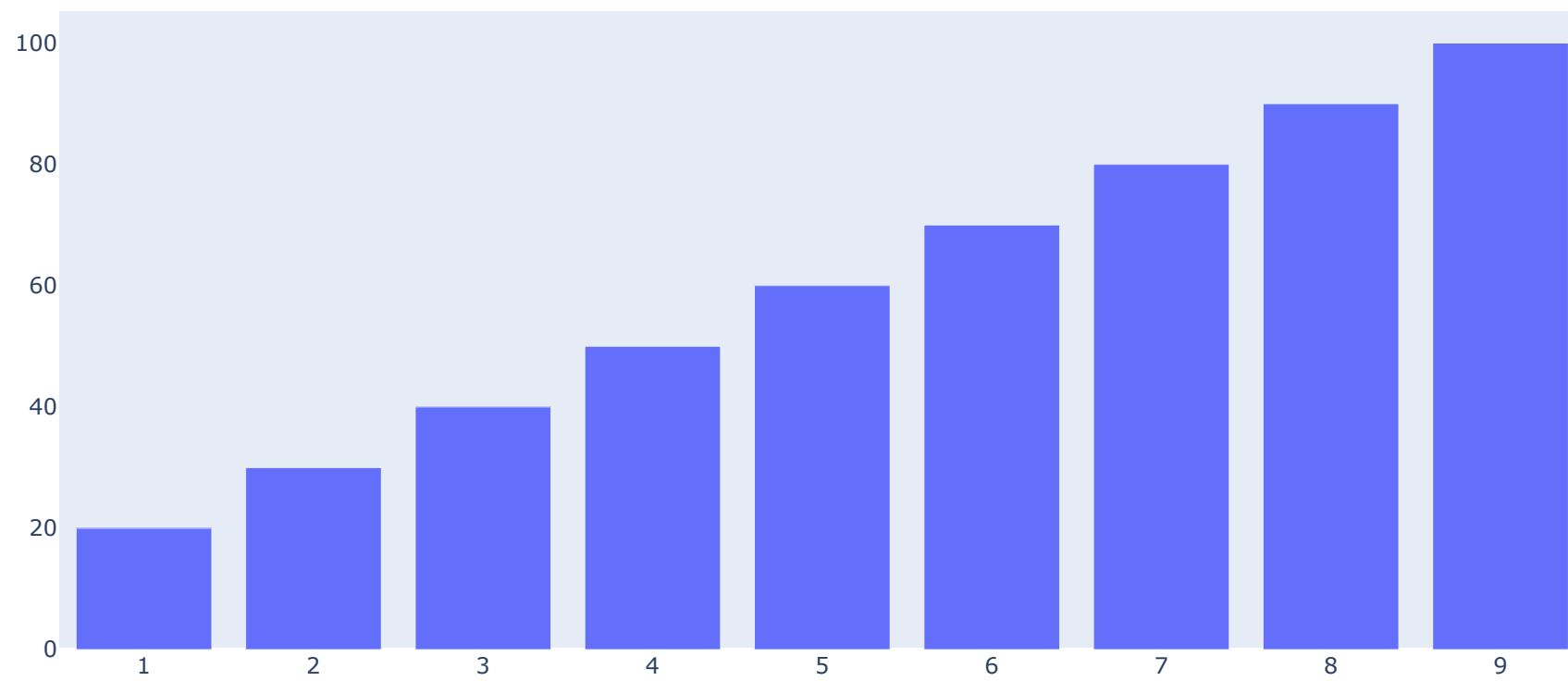
In [76]:
```python
# Use go.Bar to plot Bar charts in Plotly
x1 = [1,2,3,3,3,4,5]
y1 = [10,20,30,40,50,60,65]


data = go.Bar(
            x= x1,
            y= y1,
            )


layout = go.Layout(
                title = 'Simple Bar Chart',
                xaxis=dict(
                            showgrid=False, # Hide Gridlines
                            showline=False, # Hide X-Axis
                            ),

                yaxis=dict(
                            showgrid=False, # Hide Gridlines
                            showline=False, # Hide X-Axis
                            ),

                )

fig = go.Figure(data=data,layout=layout)
fig.show()
```
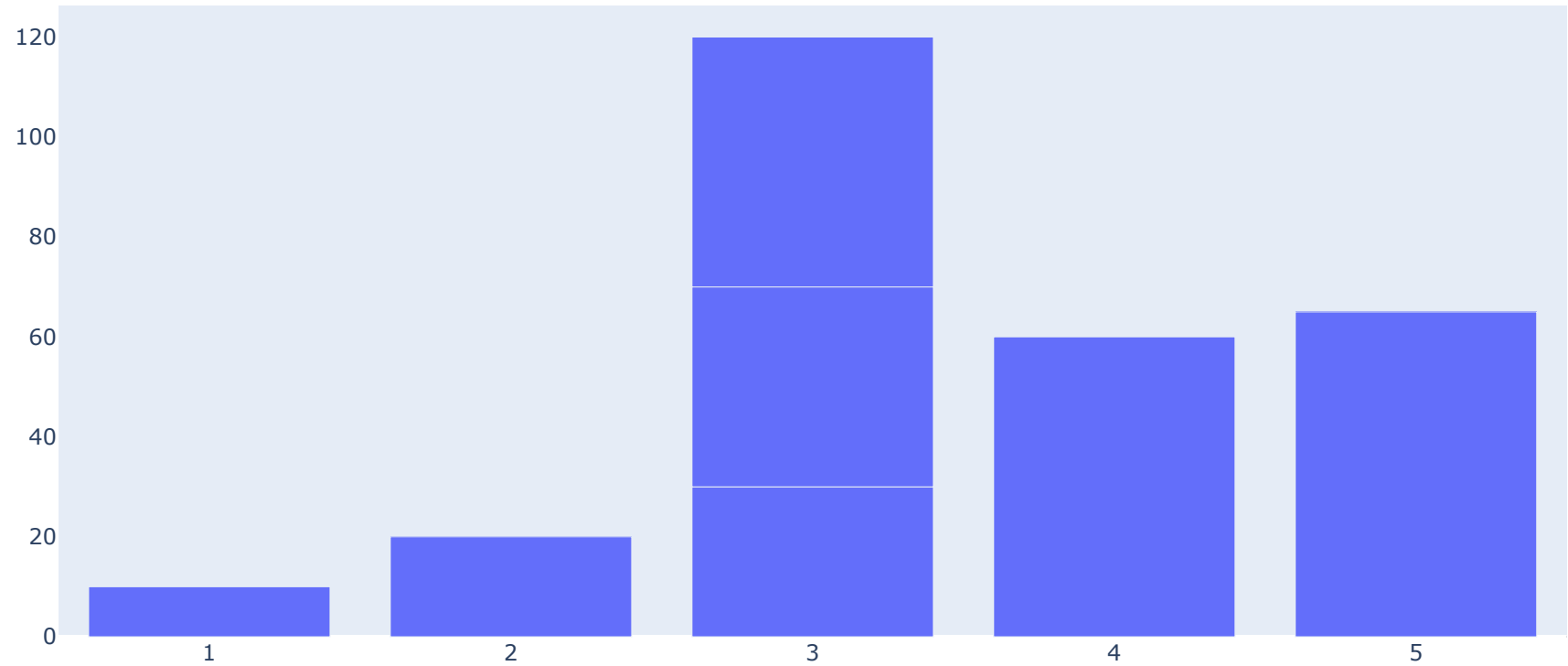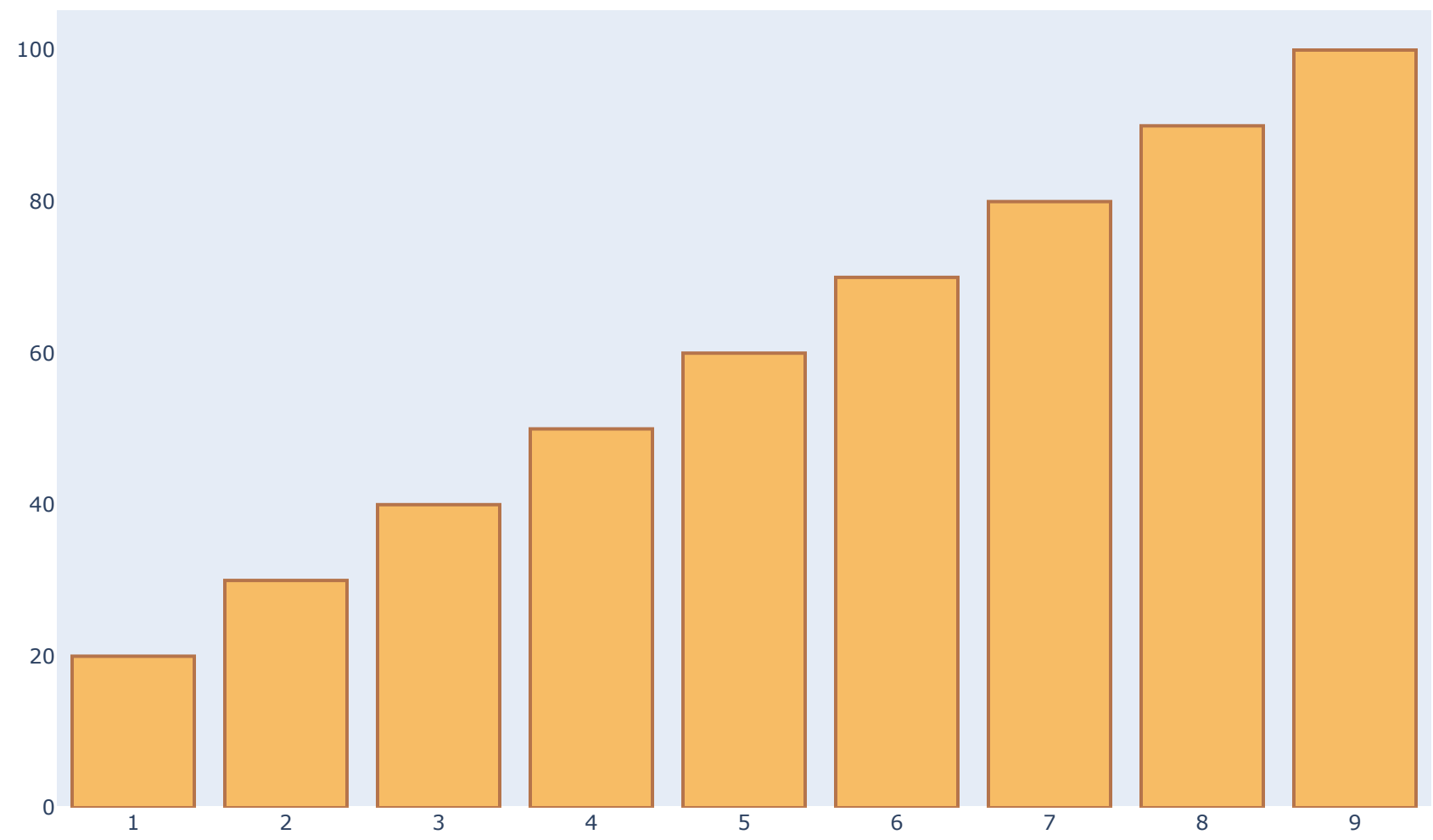
Simple Bar Chart

```
In [77]:  #Changing color of Bar plot
          x = np.arange(1,10)
          y = np.arange(20,130,10)
          data = go.Bar(
                      x= x,
                      y= y,
                      marker={'color' : '#FFA726'} # changing color of bar plot
                      )

          layout = go.Layout(
                          title = 'Simple Bar Chart',
                          width=970,
                          height=650,
                           xaxis=dict(
                                      showgrid=False, # Hide Gridlines
                                      showline=False, # Hide X-Axis
                                     ),

                           yaxis=dict(
                                      showgrid=False, # Hide Gridlines
                                      showline=False, # Hide X-Axis
                                     ),

                          )


          fig = go.Figure(data=data,layout=layout)
          fig.show()
```

Simple Bar Chart

```python
In [78]:  # Changing width of Bar Plot
          x = np.arange(1,10)
          y = np.arange(20,130,10)
          wid = [0.2,]*9
          data = go.Bar(
                      x= x,
                      y= y,
                      marker={'color' : '#FFA726'}, # Changing color of bars
                      width=wid # Changing width of Bars
                      )

          layout = go.Layout(
                          title = 'Simple Bar Chart',
                          width=970,
                          height=650,
                          xaxis=dict(
                                      showgrid=False, # Hide Gridlines
                                      showline=False, # Hide X-Axis
                                      ),

                           yaxis=dict(
                                      showgrid=False, # Hide Gridlines
                                      showline=False, # Hide X-Axis
                                      ),
                          )


          fig = go.Figure(data=data,layout=layout)
          fig.show()
```

Simple Bar Chart
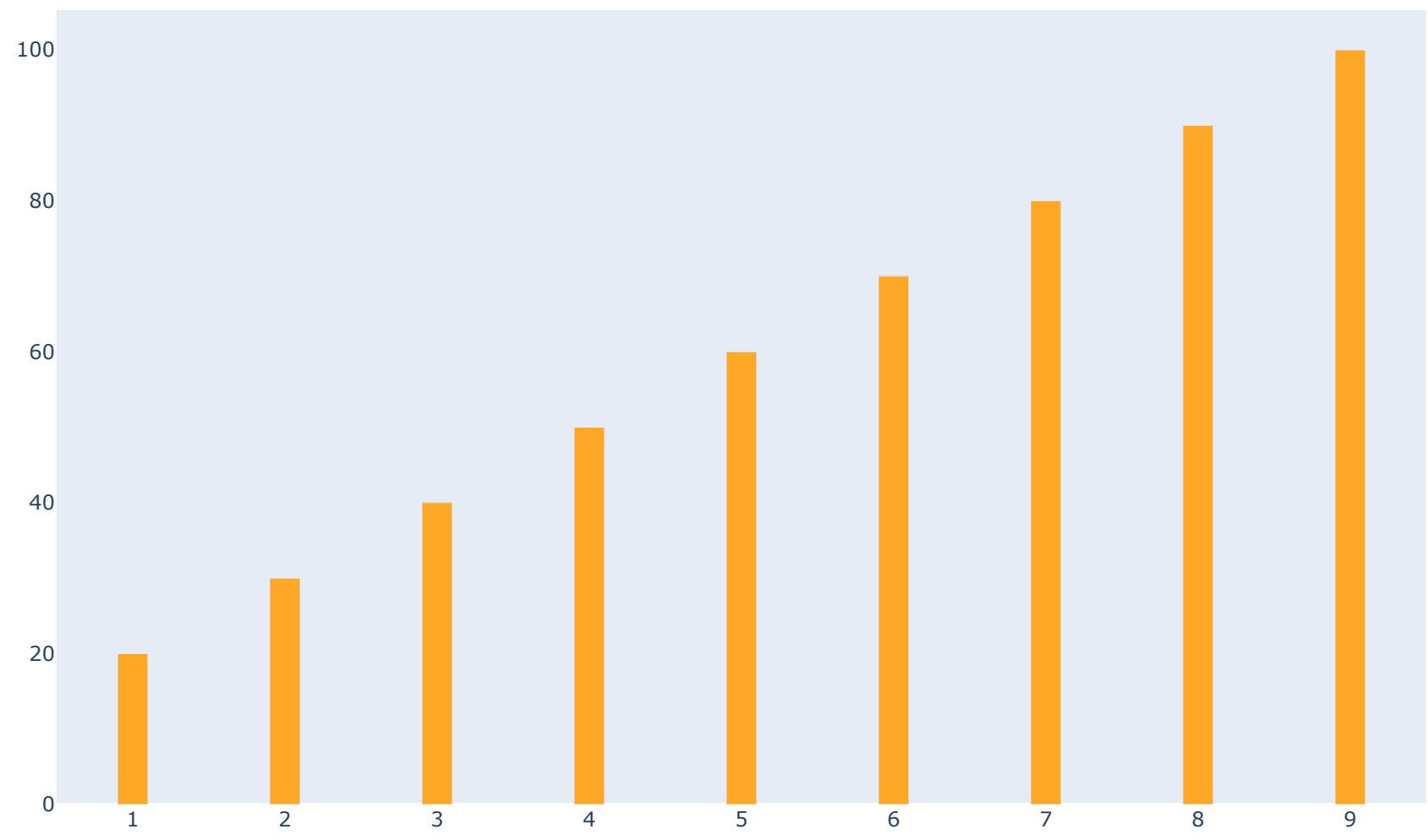
```
In [79]: x = np.arange(1,10)
         y = np.arange(20,130,10)
         data = go.Bar(
                     x= x,
                     y= y,
                     marker_color= '#FFA726', # Changing color of Bars
                     marker_line_color = '#A04000', # Changing color of border
                     marker_line_width = 2, # Changing width of border
                     opacity=0.7 # Changing opacity of Bars
                     )

         layout = go.Layout(
                         title = 'Simple Bar Chart',
                         width=970,
                         height=650,
                          xaxis=dict(
                                     showgrid=False, # Hide Gridlines
                                     showline=False, # Hide X-Axis
                                     ),

                         yaxis=dict(
                                     showgrid=False, # Hide Gridlines
                                     showline=False, # Hide X-Axis
                                     ),
                         )


         fig = go.Figure(data=data,layout=layout)
         fig.show()
```

Simple Bar Chart

```python
# Simple Horizontal Bar Plot (Using orientation='h')
x = np.arange(1,10)
y = np.arange(20,130,10)
data = go.Bar(
              x= y,
              y= x,
              marker={'color' : '#FFA726'},
              orientation='h'
              )

layout = go.Layout(
                  title = 'Simple Bar Chart',
                  width=970,
                  height=650,
                   xaxis=dict(
                              showgrid=False, # Hide Gridlines
                              showline=False, # Hide X-Axis
                             ),

                   yaxis=dict(
                              showgrid=False, # Hide Gridlines
                              showline=False, # Hide X-Axis
                             ),
                  )


fig = go.Figure(data=data,layout=layout)
fig.show()
```

Simple Bar Chart
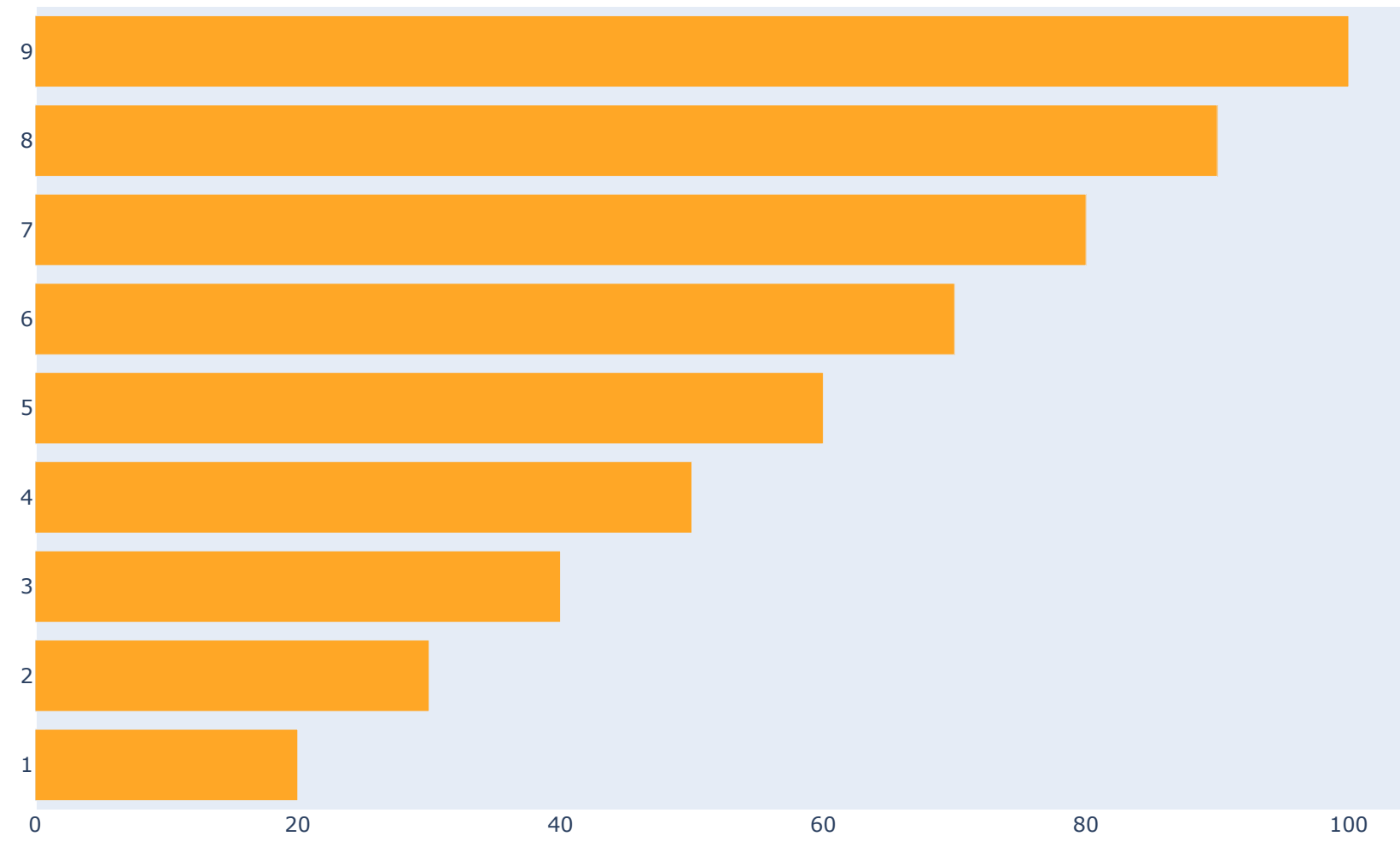
```
In [81]: Assignee = ['Asif', 'Basit', 'John', 'Charles','Vignesh', 'Arun', 'Ashish', 'Varun']
         Tickets_Closed = np.array([52,42,60,35,48,45,28,54])
         per = (Tickets_Closed/sum(Tickets_Closed))*100
         per = np.round(per, decimals=2)

         #Create dataframe
         helpdesk = pd.DataFrame({'Assignee' : Assignee, 'Tickets Closed' :Tickets_Closed , 'Percentage': per})
         helpdesk = helpdesk.sort_values(by='Percentage')

         #Initialize the figure
         fig = go.Figure()

         #Plot Closure percentage using Horizontal Bar plot
         fig.add_trace(go.Bar(
                             x=helpdesk.Percentage,
                             y=helpdesk.Assignee,
                             orientation='h',
                             marker=dict(
                                     color='rgba(70, 171, 100, 0.6)',
                                     line=dict(color='rgba(70, 171, 100, 1.0)',width=1),
                                 ),
                         )
                 )

         #Update layout
         fig.update_layout(
                         title=dict(text = "Ticket Closure Summary",x=0.46,y=0.95,font_size=20),
                         yaxis=dict(
                                 showticklabels=True,showgrid=False,showline=False
                             ),

                         xaxis=dict(
                                 showticklabels=False,
                                 domain=[0, 0.6],
                                 showgrid=False,showline=False
                             ),
                         margin=dict(l=300, r=20, t=70, b=70),
                         paper_bgcolor='rgb(248, 248, 255)',
                         plot_bgcolor='rgb(248, 248, 255)',
                     )



         annotations = [] #Initialize anotation object


         # Labels
         for perc, asg in zip(helpdesk.Percentage, helpdesk.Assignee):

             # Displaying label bar percentage
             annotations.append(dict(xref='x',
                                 yref='y',
                                 y=asg,
                                 x=perc + 2,
                                 text=str(perc) + '%',
                                 font=dict(family='Arial', size=12,color='#004D40'),
                                 showarrow=False))
             # Displaying Footer
             annotations.append(dict(xref='paper',
                                 yref='paper',
                                 x=1,
```

```
                        y=-0.17,
                        text='Help Desk ' +'Closure Statistics, ' +'Year 2020',
                        font=dict(family='Arial', size=11, color='#9E9E9E'),
                        showarrow=False
                    )
                )
            )

fig.update_layout(annotations=annotations)

fig.show()
```

Ticket Closure Summary

Name	Percentage
John	16.48%
Varun	14.84%
Asif	14.29%
Vignesh	13.19%
Arun	12.36%
Basit	11.54%
Charles	9.62%
Ashish	7.69%

Help Desk Closure Statistics, Year 2020
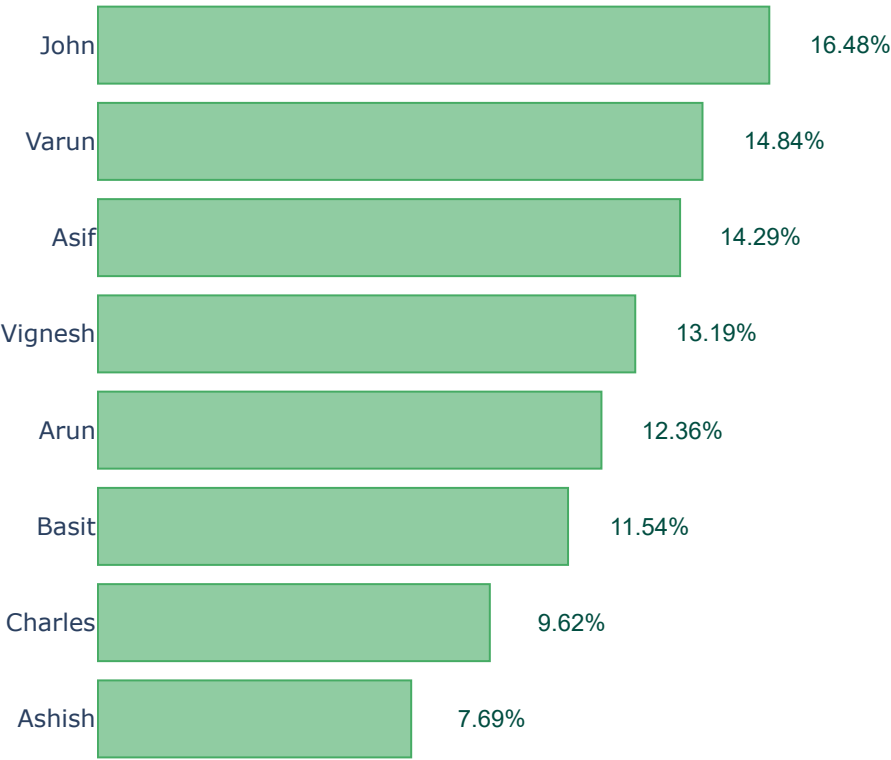
```
In [82]: x1= [1,3,5,7]
         x2=[2,4,6,8]
         y1 = [7,7,7,7]
         y2= [17,18,29,40]

         trace0 = go.Bar(
                     x= x1,
                     y= y1,
                     marker= dict (color ='#FFA726' )

                     )

         trace1 = go.Bar(
                     x= x2,
                     y= y2,
                     marker={'color' : '#94E413'}
                     )


         data = [trace0,trace1]
         fig = go.Figure(data=data)

         # Hide grid lines
         fig.update_xaxes(showgrid=False)
         fig.update_yaxes(showgrid=False)

         fig.show()
```
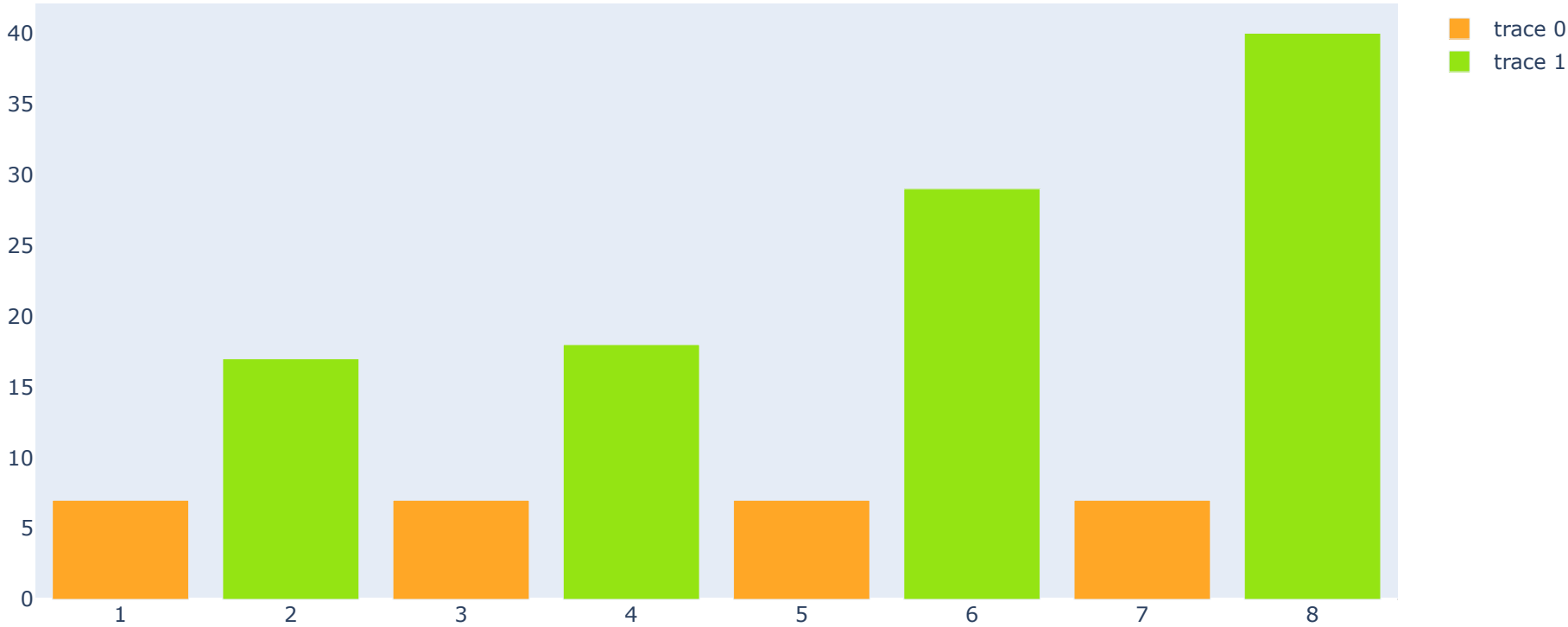
Stacked Vertical Bar

```python
In [83]: x = ['Asif','Basit','John','Batista', 'Gabriel' , 'Paul' , 'Ravi']
         y1 = [17,18,29,33,38,39,42]
         y2 = [20,21,22,23,21,28,25]
         y3 = [5,13,11,12,13,11,16]
         trace0 = go.Bar(
                     x= x,
                     y= y1,
                     marker= dict (color ='#FF6F00' ),
                     name = 'Open Tickets'

                     )

         trace1 = go.Bar(
                     x= x,
                     y= y2,
                     marker={'color' : '#FFB300'},
                      name = 'Closed Tickets'
                      )

         trace2 = go.Bar(
                     x= x,
                     y= y3,
                     marker={'color' : '#F7DC6F'},
                      name = 'Cancelled Tickets'
                      )

         layout = go.Layout(
                         title= 'Open Tickets by Status' ,
                         barmode = 'stack',
                         width=900,
                         height=800
                         )


         data = [trace0,trace1,trace2]
         fig = go.Figure(data=data, layout=layout)

         # Hide grid lines
         fig.update_xaxes(showgrid=False)
         fig.update_yaxes(showgrid=False)

         fig.show()
```
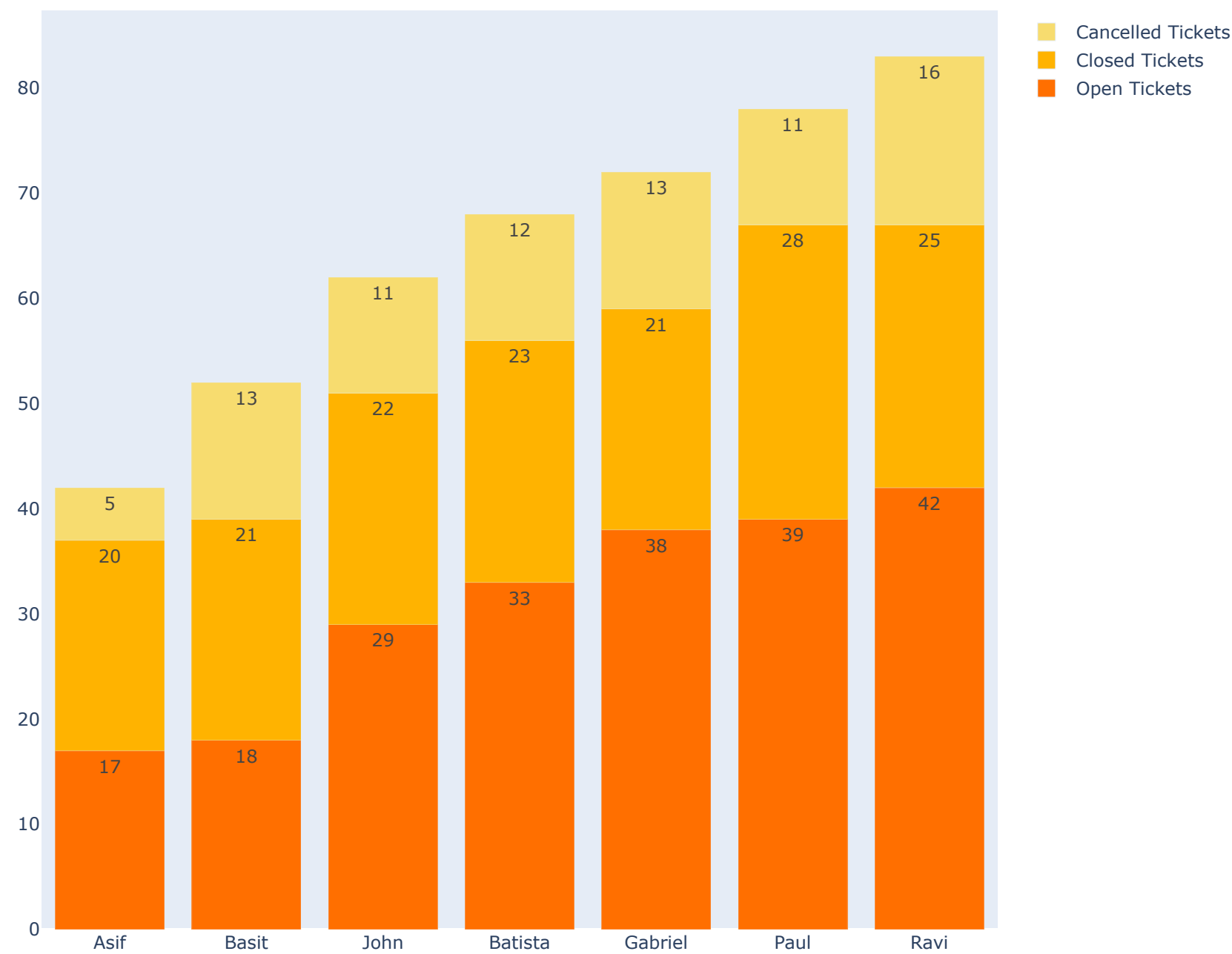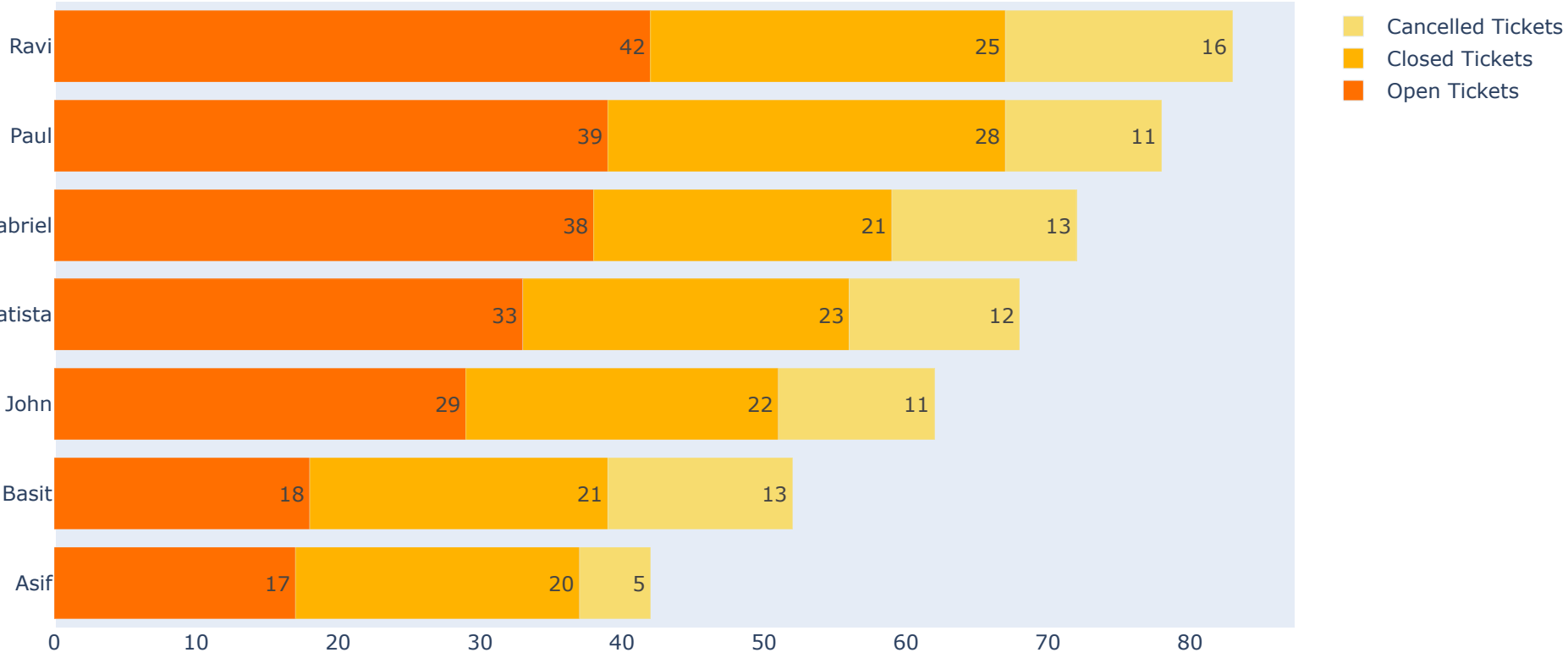
Open Tickets by Status

```
In [84]:  #Displaying values in bar plot using "text" and "textposition" parameter
          x = ['Asif','Basit','John','Batista', 'Gabriel' , 'Paul' , 'Ravi']
          y1 = [17,18,29,33,38,39,42]
          y2 = [20,21,22,23,21,28,25]
          y3 = [5,13,11,12,13,11,16]
          trace0 = go.Bar(
                      x= x,
                      y= y1,
                      marker= dict (color ='#FF6F00' ),
                      name = 'Open Tickets',
                      text=y1,
                      textposition='auto'
                      )

          trace1 = go.Bar(
                      x= x,
                      y= y2,
                      marker={'color' : '#FFB300'},
                      name = 'Closed Tickets',
                      text=y2,
                      textposition='auto'
                       )

          trace2 = go.Bar(
                      x= x,
                      y= y3,
                      marker={'color' : '#F7DC6F'},
                      name = 'Cancelled Tickets',
                      text=y3,
                      textposition='auto'
                       )

          layout = go.Layout(
                          title= 'Open Tickets by Status' ,
                          barmode = 'stack',
                          width=900,
                          height=800
                          )


          data = [trace0,trace1,trace2]
          fig = go.Figure(data=data, layout=layout)

          # Hide grid lines
          fig.update_xaxes(showgrid=False)
          fig.update_yaxes(showgrid=False)

          fig.show()
```

Open Tickets by Status

Stacked Horizontal Bar

```
In [85]: x = ['Asif','Basit','John','Batista', 'Gabriel' , 'Paul' , 'Ravi']
         y1 = [17,18,29,33,38,39,42]
         y2 = [20,21,22,23,21,28,25]
         y3 = [5,13,11,12,13,11,16]

         trace0 = go.Bar(
                      x= y1,
                      y= x,
                      marker= dict (color ='#FF6F00'),
                      name = 'Open Tickets',
                      orientation='h',
                      text=y1,
                      textposition='auto'
                      )

         trace1 = go.Bar(
                      x= y2,
                      y= x,
                      marker={'color' : '#FFB300'},
                      name = 'Closed Tickets',
                      orientation='h',
                      text=y2,
                      textposition='auto'
                       )


         trace2 = go.Bar(
                      x= y3,
                      y= x,
                      marker={'color' : '#F7DC6F'},
                      name = 'Cancelled Tickets',
                      orientation='h',
                      text=y3,
                      textposition='auto'
                       )

         layout = go.Layout(
                          title= 'Open Tickets by Status' ,
                          barmode = 'stack',
                          width=990,
                          height=550
                          )

         data = [trace0,trace1,trace2]
         fig = go.Figure(data=data, layout=layout)

         # Hide grid lines
         fig.update_xaxes(showgrid=False)
         fig.update_yaxes(showgrid=False)

         fig.show()
```

Open Tickets by Status

Name	Open Tickets	Closed Tickets	Cancelled Tickets
Ravi	42	25	16
Paul	39	28	11
Gabriel	38	21	13
Batista	33	23	12
John	29	22	11
Basit	18	21	13
Asif	17	20	5

```
In [86]: x = ['Asif','Basit','John','Batista', 'Gabriel' , 'Paul' , 'Ravi']
         y1 = [17,18,29,33,38,39,42]
         y2 = [20,21,22,23,21,28,25]
         y3 = [5,13,11,12,13,11,16]

         trace0 = go.Bar(
                     x= y1,
                     y= x,
                     marker= dict (color ='#FF9800',line=dict(color='#F4511E',width=2)),
                     name = 'Open Tickets',
                     orientation='h',
                     text=y1,
                     textposition='auto',
                     opacity=0.8,
                     )

         trace1 = go.Bar(
                     x= y2,
                     y= x,
                     marker= dict (color ='#7CB342',line=dict(color='#2E7D32', width=2)),
                     name = 'Closed Tickets',
                     orientation='h',
                     text=y2,
                     textposition='auto',
                     opacity=0.8,
                      )


         trace2 = go.Bar(
                     x= y3,
                     y= x,
                     marker= dict (color ='#1E88E5',line=dict(color='#3F51B5', width=2)),
                     name = 'Cancelled Tickets',
                     orientation='h',
                     text=y3,
                     textposition='auto',
                     opacity=0.7,
                      )

         layout = go.Layout(
                        title= 'Open Tickets by Status' ,
                        barmode = 'stack',
                        width=990,
                        height=600
                        )

         data = [trace0,trace1,trace2]
         fig = go.Figure(data=data, layout=layout)

         # Hide grid lines
         fig.update_xaxes(showgrid=False)
         fig.update_yaxes(showgrid=False)

         fig.show()
```
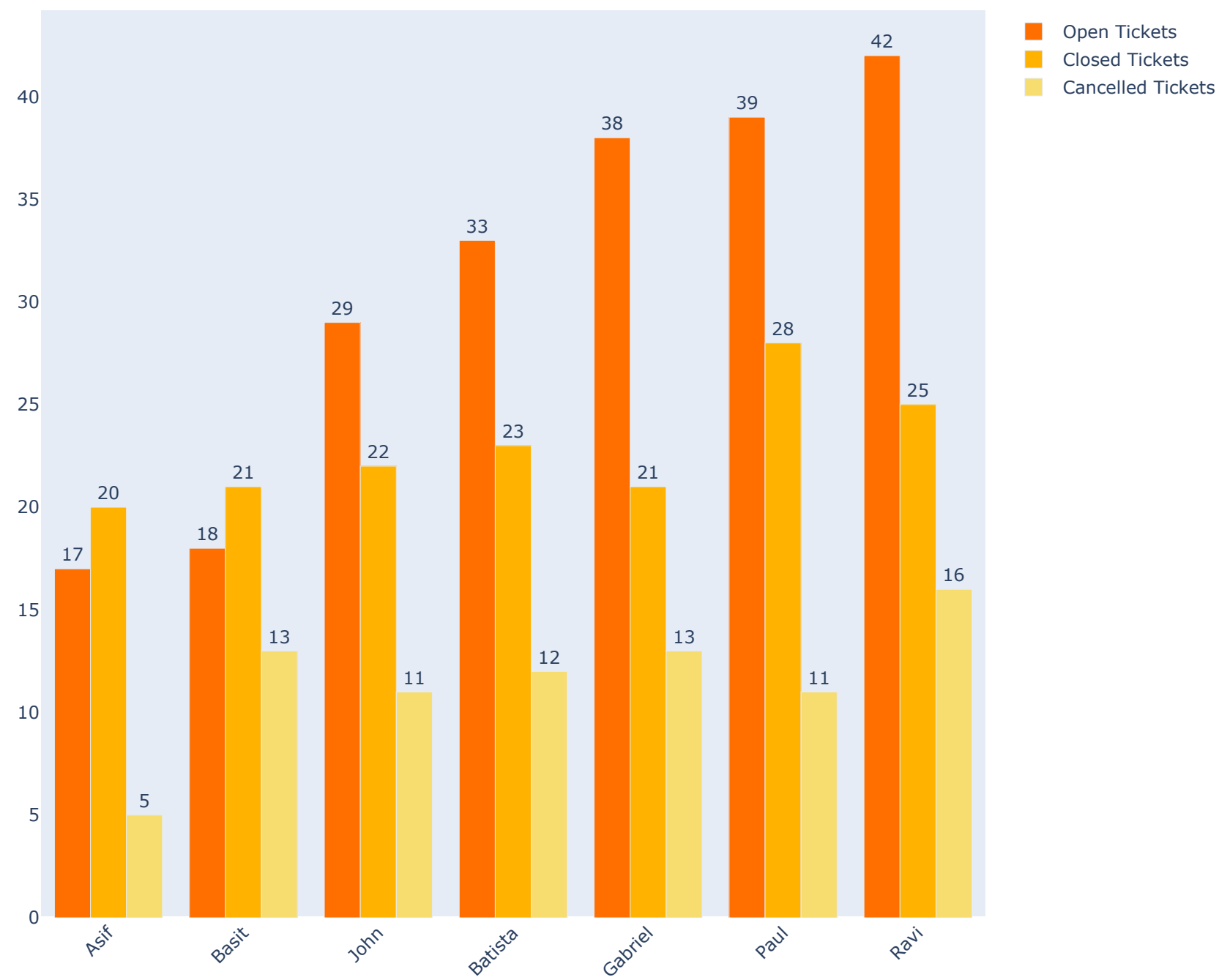
Open Tickets by Status
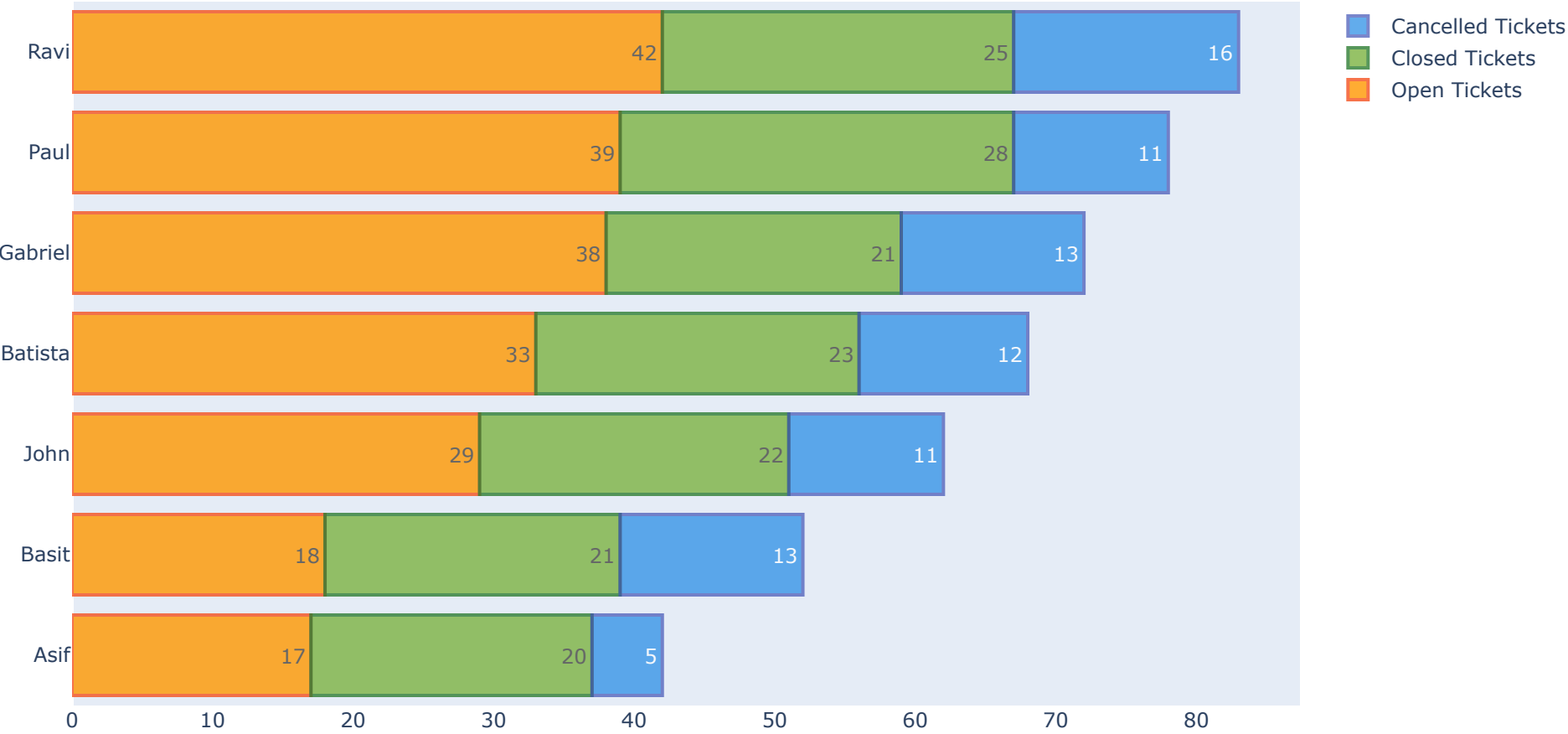
Grouped Bar Chart

```
In [87]: x = ['Asif','Basit','John','Batista', 'Gabriel' , 'Paul' , 'Ravi']
         y1 = [17,18,29,33,38,39,42]
         y2 = [20,21,22,23,21,28,25]
         y3 = [5,13,11,12,13,11,16]
         trace0 = go.Bar(
                     x= x,
                     y= y1,
                     marker= dict (color ='#FF6F00' ),
                     name = 'Open Tickets',
                     )

         trace1 = go.Bar(
                     x= x,
                     y= y2,
                     marker={'color' : '#FFB300'},
                      name = 'Closed Tickets'
                      )

         trace2 = go.Bar(
                     x= x,
                     y= y3,
                     marker={'color' : '#F7DC6F'},
                      name = 'Cancelled Tickets'
                      )

         layout = go.Layout(
                         title= 'Open Tickets by Status' ,
                         width=980,
                         height=800
                         )


         data = [trace0,trace1,trace2]
         fig = go.Figure(data=data, layout=layout)

         # Hide grid lines
         fig.update_xaxes(showgrid=False)
         fig.update_yaxes(showgrid=False)

         fig.show()
```

Open Tickets by Status
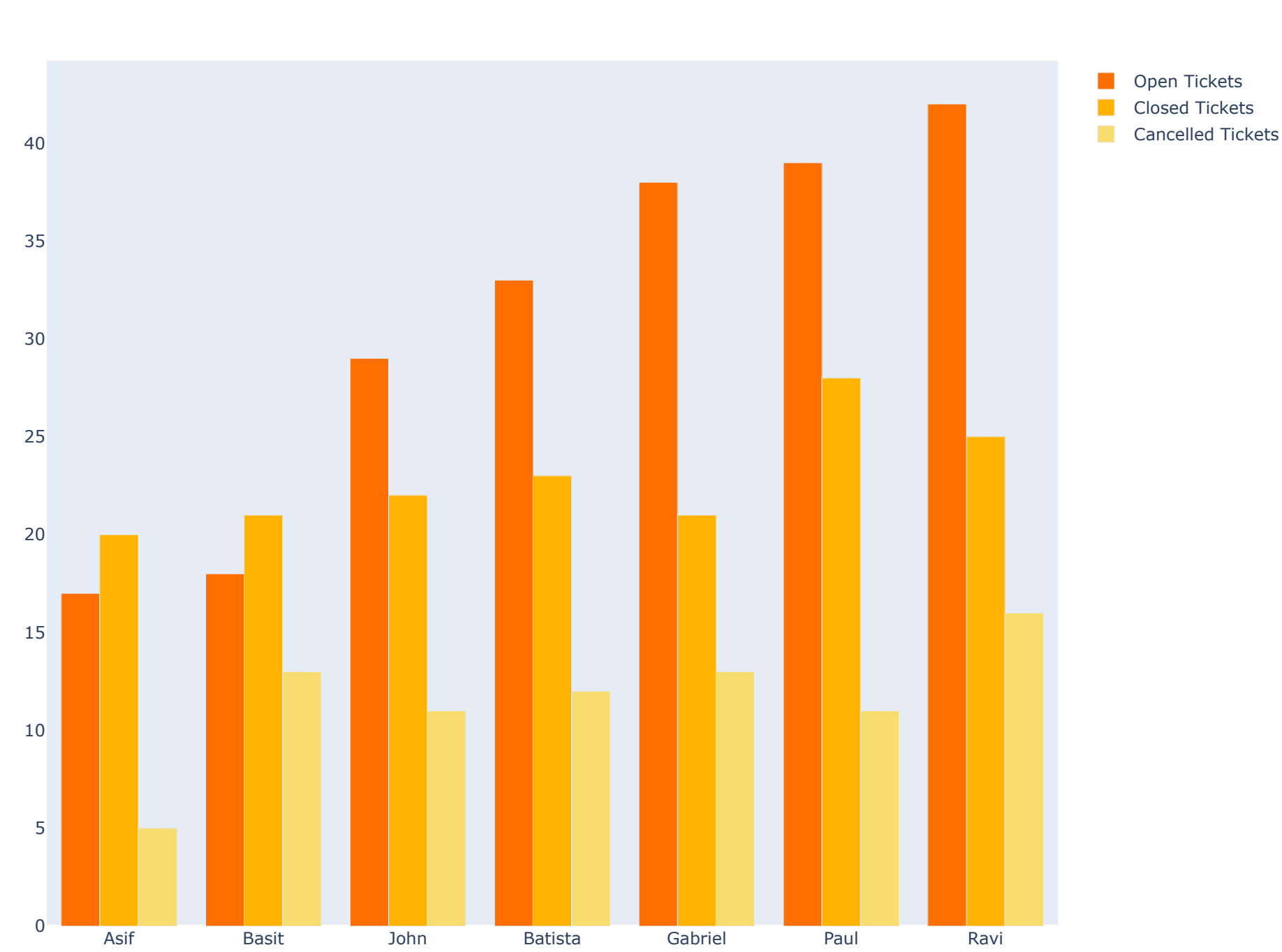
```
In [88]:  # Grouped Bar Chart with values displayed outside the bar (Using  textposition='outside')
          x = ['Asif','Basit','John','Batista', 'Gabriel' , 'Paul' , 'Ravi']
          y1 = [17,18,29,33,38,39,42]
          y2 = [20,21,22,23,21,28,25]
          y3 = [5,13,11,12,13,11,16]
          trace0 = go.Bar(
                      x= x,
                      y= y1,
                      marker= dict (color ='#FF6F00' ),
                      name = 'Open Tickets',
                      text=y1,
                      textposition='outside'
                      )

          trace1 = go.Bar(
                      x= x,
                      y= y2,
                      marker={'color' : '#FFB300'},
                      name = 'Closed Tickets',
                      text=y2,
                      textposition='outside'
                       )

          trace2 = go.Bar(
                      x= x,
                      y= y3,
                      marker={'color' : '#F7DC6F'},
                      name = 'Cancelled Tickets',
                      text=y3,
                      textposition='outside'
                       )

          layout = go.Layout(
                          title= 'Open Tickets by Status' ,
                          barmode = 'group',
                          width=900,
                          height=800,
                          xaxis_tickangle=-45
                          )


          data = [trace0,trace1,trace2]
          fig = go.Figure(data=data, layout=layout)

          # Hide grid lines
          fig.update_xaxes(showgrid=False)
          fig.update_yaxes(showgrid=False)

          fig.show()
```

Open Tickets by Status

Legend:
- Open Tickets
- Closed Tickets
- Cancelled Tickets

Name	Open Tickets	Closed Tickets	Cancelled Tickets
Asif	17	20	5
Basit	18	21	13
John	29	22	11
Batista	33	23	12
Gabriel	38	21	13
Paul	39	28	11
Ravi	42	25	16

```python
In [89]: # Grouped Bar Chart with values displayed inside the bar using "auto" textposition
         x = ['Asif','Basit','John','Batista', 'Gabriel' , 'Paul' , 'Ravi']
         y1 = [17,18,29,33,38,39,42]
         y2 = [20,21,22,23,21,28,25]
         y3 = [5,13,11,12,13,11,16]
         trace0 = go.Bar(
                     x= x,
                     y= y1,
                     marker= dict (color ='#FF6F00' ),
                     name = 'Open Tickets',
                     text=y1,
                     textposition='auto'
                     )

         trace1 = go.Bar(
                     x= x,
                     y= y2,
                     marker={'color' : '#FFB300'},
                     name = 'Closed Tickets',
                     text=y2,
                     textposition='auto'
                      )

         trace2 = go.Bar(
                     x= x,
                     y= y3,
                     marker={'color' : '#F7DC6F'},
                     name = 'Cancelled Tickets',
                     text=y3,
                     textposition='auto'
                      )


         layout = go.Layout(
                         title=dict(text = "Tickets by Status",x=0.5,y=0.95,font_size=25),
                         barmode = 'group',
                         width=980,
                         height=800,
                         xaxis_tickangle=-45,
                         xaxis_tickfont_size=14,
                         yaxis=dict(
                                 title='Number of Tickets',
                                 titlefont_size=16,
                                 tickfont_size=14,
                                 ),

                         xaxis=dict(
                                 title='Assignee',
                                 titlefont_size=16,
                                 tickfont_size=14,
                                 ),
                         legend=dict(
                                 x=0,
                                 y=1,
                                 bgcolor='rgba(255, 255, 255, 0)',
                                 bordercolor='rgba(255, 255, 255, 0)'
                                 ),
                         bargap=0.15, # gap between bars of adjacent location coordinates.
                         bargroupgap=0.08 # gap between bars of the same location coordinate.
                         )
```

```
data = [trace0,trace1,trace2]
fig = go.Figure(data=data, layout=layout)

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```


Tickets by Status
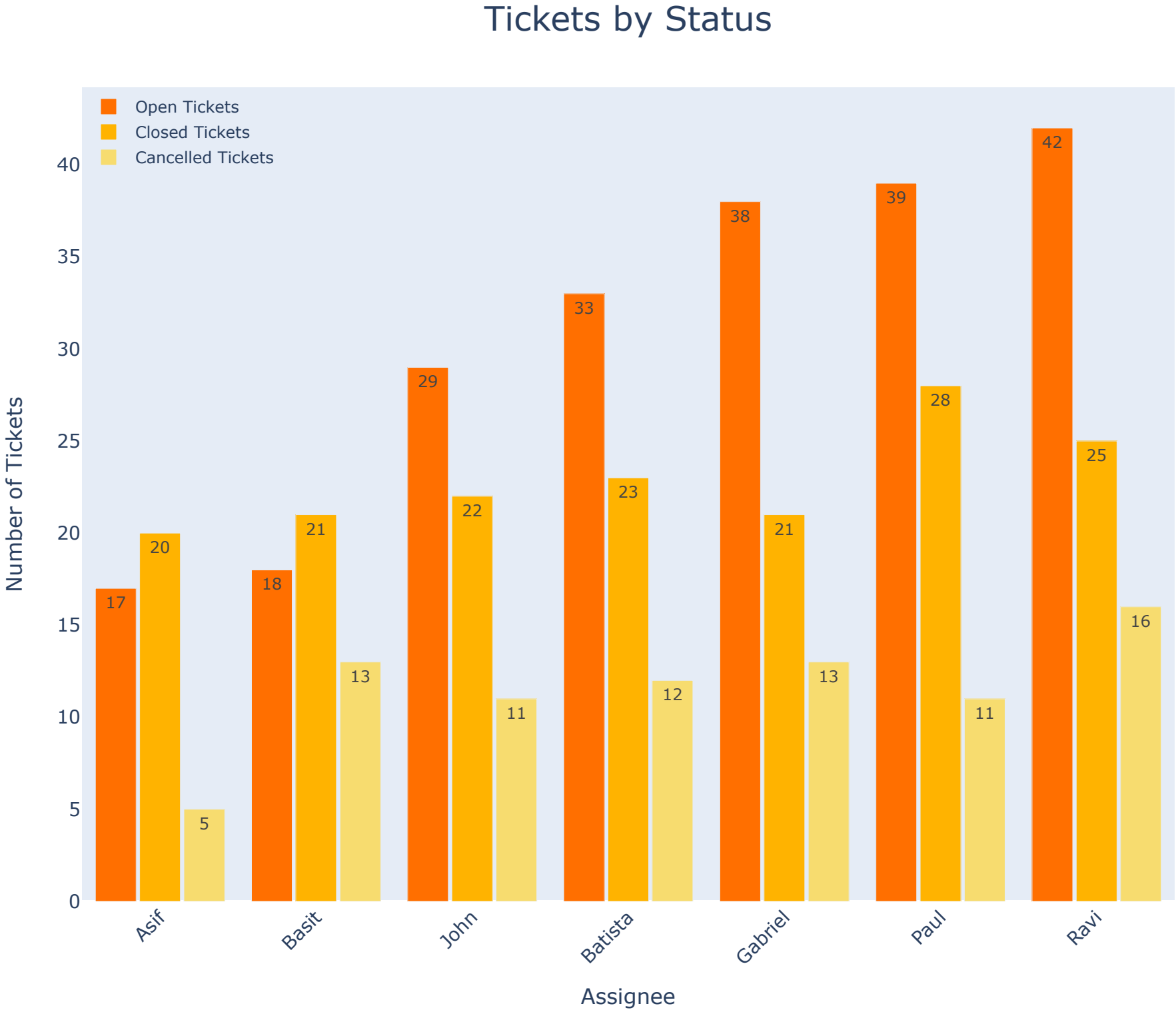
Bar Plot using Plotly Express

In [90]:
```python
#Simple Bar plot using px.bar

x = np.arange(1,10)
y = np.arange(20,110,10)
fig = px.bar(x=x, y=y)

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
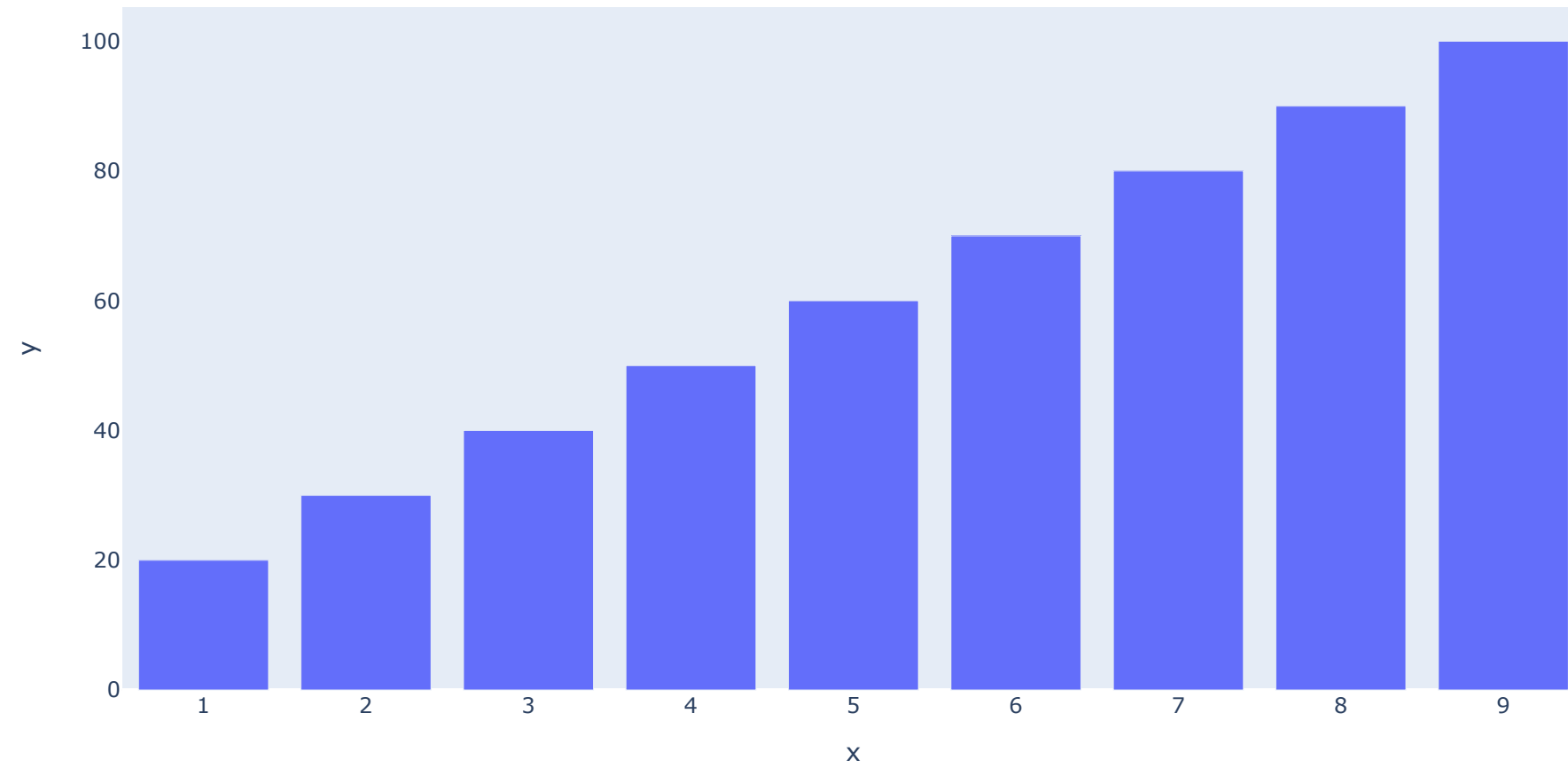
```python
x = np.arange(1,10)
y = np.arange(20,110,10)
fig = px.bar(x=x, y=y,color=y)
fig.layout.title.text = "Bar Plot - Plotly Express"
fig.update_traces(textposition='outside')

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```


Bar Plot - Plotly Express
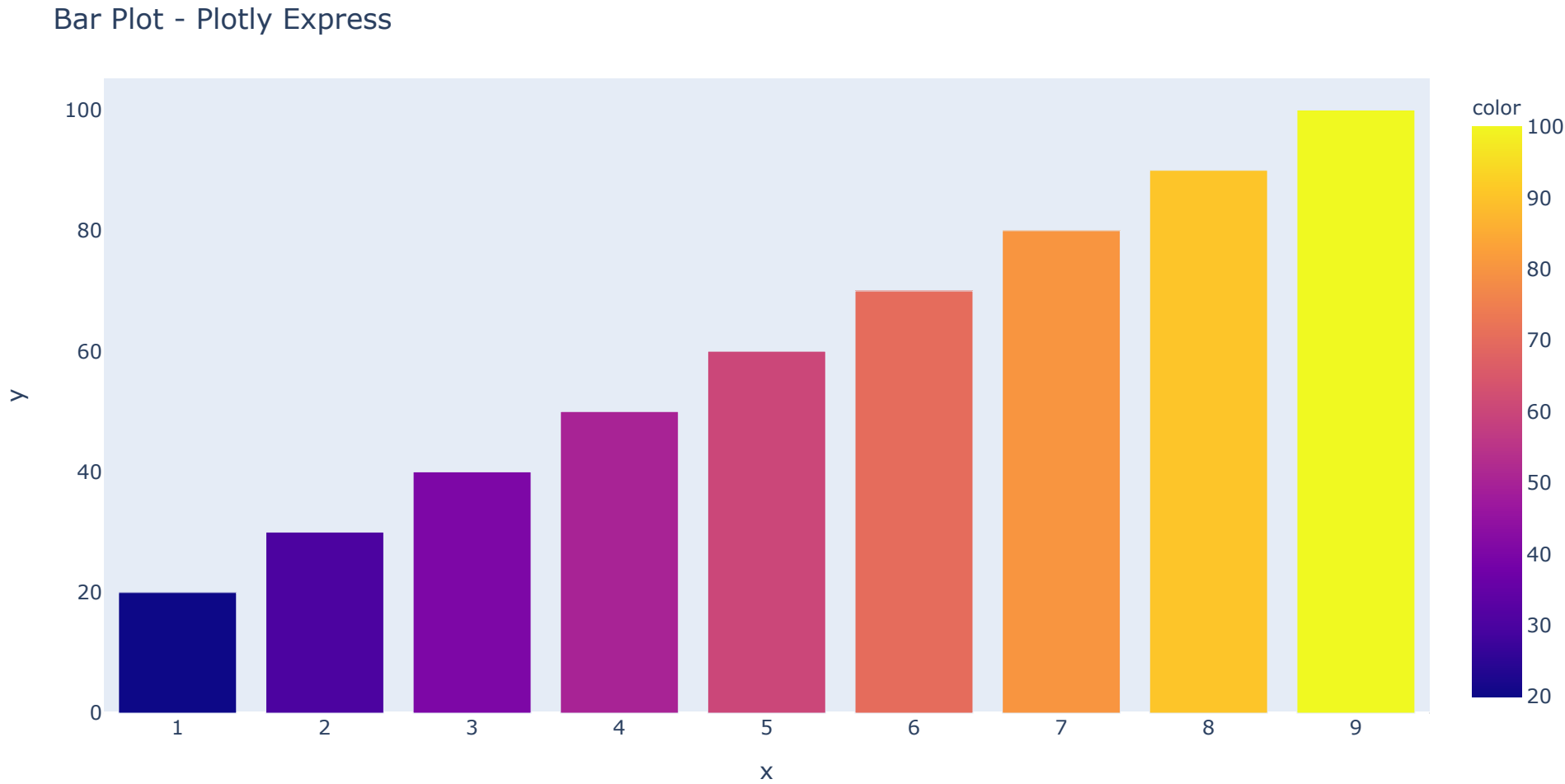
```python
x = np.arange(1,10)
y = np.arange(20,110,10)
fig = px.bar(x=x, y=y,color=y,text=y)
fig.layout.title.text = "Bar Plot - Plotly Express"
fig.update_traces(textposition='outside')

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
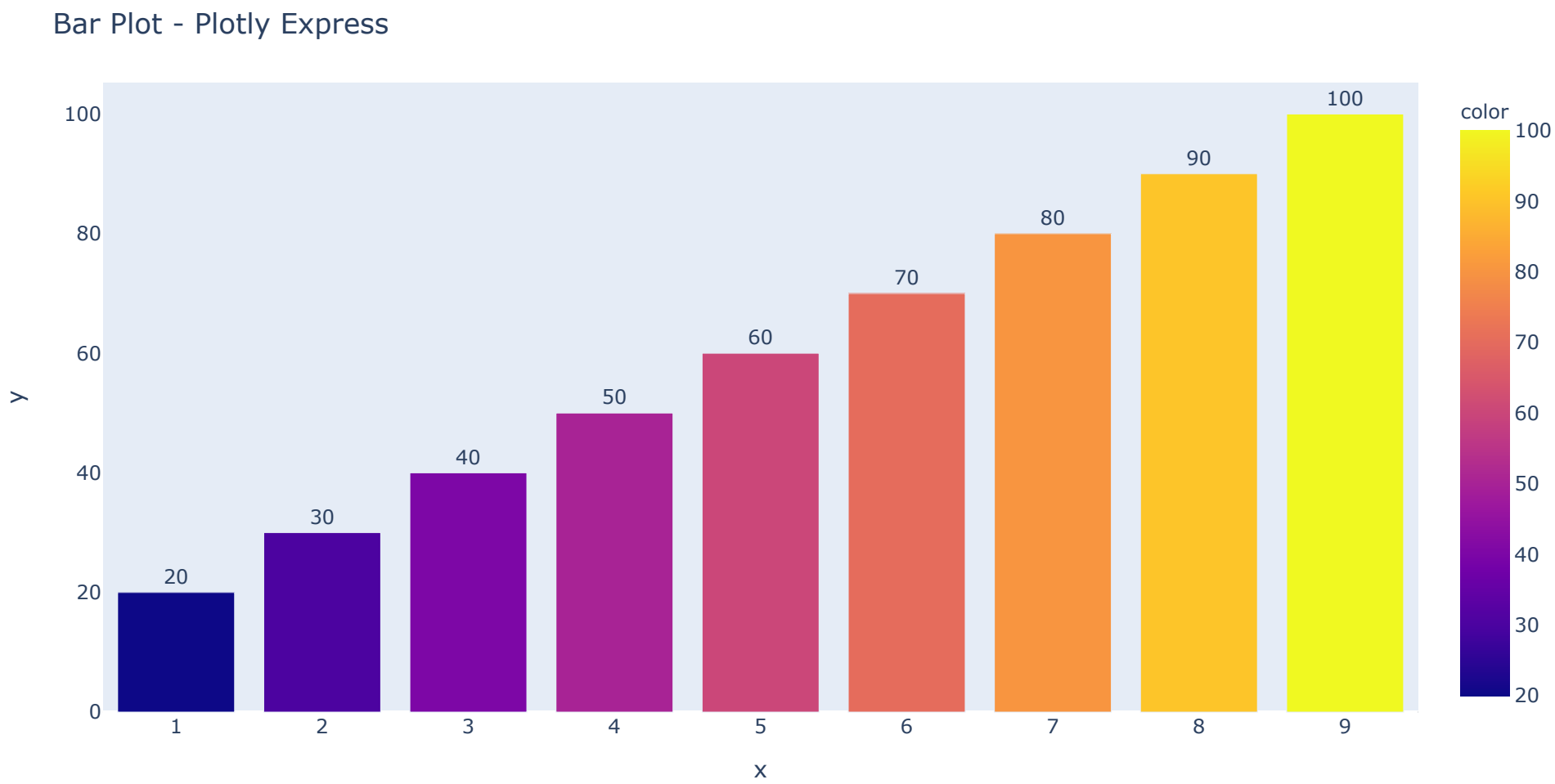

100% Stacked Bar Chart

```
In [93]: col1 = ['Strongly Agree' , 'Agree' , 'Neutral' , 'Disagree' , 'Strongly Disagree',]
         index1 = ['Python' , 'Java' , 'Julia' , 'C++' , 'C']
         review = np.array([[428, 111, 70, 101, 80],
                            [370, 222, 80, 104, 70],
                            [298, 121, 90, 102, 60],
                            [310, 141, 100, 109, 56],
                            [400, 121, 110, 107, 78]]
                           )

         rating = pd.DataFrame(data=review,index=index1,columns=col1)
         rating
```

Out[93]:

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Python	428	111	70	101	80
Java	370	222	80	104	70
Julia	298	121	90	102	60
C++	310	141	100	109	56
C	400	121	110	107	78

```
In [94]: rating['Total'] = rating.sum(axis=1)
         rating
```

Out[94]:

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree	Total
Python	428	111	70	101	80	790
Java	370	222	80	104	70	846
Julia	298	121	90	102	60	671
C++	310	141	100	109	56	716
C	400	121	110	107	78	816

```
In [95]: rating = rating.assign(**{'Strongly Agree (%)': np.nan,
                     'Agree (%)' : np.nan,
                     'Neutral (%)': np.nan ,
                     'Disagree (%)' : np.nan,
                     'Strongly Disagree (%)': np.nan,
                    })
         rating
```

Out[95]:

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree	Total	Strongly Agree (%)	Agree (%)	Neutral (%)	Disagree (%)	Strongly Disagree (%)
Python	428	111	70	101	80	790	NaN	NaN	NaN	NaN	NaN
Java	370	222	80	104	70	846	NaN	NaN	NaN	NaN	NaN
Julia	298	121	90	102	60	671	NaN	NaN	NaN	NaN	NaN
C++	310	141	100	109	56	716	NaN	NaN	NaN	NaN	NaN
C	400	121	110	107	78	816	NaN	NaN	NaN	NaN	NaN

```
In [96]:    for i in range(0,len(rating['Strongly Agree'])):
                k=0
                for j in range(int(len(rating.iloc[0])/2)+1 , len(rating.iloc[0])):
                    rating.iat[i,j] =  np.round((rating.iat[i,k] / rating.iat[i,5])*100 ,decimals=2)
                    k=k+1
            rating
```

Out[96]:

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree	Total	Strongly Agree (%)	Agree (%)	Neutral (%)	Disagree (%)	Strongly Disagree (%)
Python	428	111	70	101	80	790	54.18	14.05	8.86	12.78	10.13
Java	370	222	80	104	70	846	43.74	26.24	9.46	12.29	8.27
Julia	298	121	90	102	60	671	44.41	18.03	13.41	15.20	8.94
C++	310	141	100	109	56	716	43.30	19.69	13.97	15.22	7.82
C	400	121	110	107	78	816	49.02	14.83	13.48	13.11	9.56

```
In [100]: fig = go.Figure()

          # Trace to plot "Strongly Agree" percentage
          fig.add_trace(
                      go.Bar(
                              x= rating['Strongly Agree (%)'],
                              y= rating.index,
                              marker= dict (color ='#2E7D32',line=dict(color='#2E7D32',width=2)),
                              name = 'Strongly Agree (%)',
                              orientation='h',
                              textposition='auto',
                              )
                      )


          # Trace to plot "Agree" percentage
          fig.add_trace(
                      go.Bar(
                              x= rating['Agree (%)'],
                              y= rating.index,
                              marker= dict (color ='#8BC34A',line=dict(color='#8BC34A',width=2)),
                              name = 'Agree (%)',
                              orientation='h',
                              textposition='auto',
                              )
                      )



          # Trace to plot "Neutral" percentage
          fig.add_trace(
                      go.Bar(
                              x= rating['Neutral (%)'],
                              y= rating.index,
                              marker= dict (color ='#D4E157',line=dict(color='#D4E157',width=2)),
                              name = 'Neutral (%)',
                              orientation='h',
                              textposition='auto',
                              )
                      )



          # Trace to plot "Disagree" percentage
          fig.add_trace(
                      go.Bar(
                              x= rating['Disagree (%)'],
                              y= rating.index,
                              marker= dict (color ='#FFB300',line=dict(color='#FFB300',width=2)),
                              name = 'Disagree (%)',
                              orientation='h',
                              textposition='auto',
                              )
                      )


          # Trace to plot "Strongly Disagree" percentage
          fig.add_trace(
                      go.Bar(
```

```python
                    x= rating['Strongly Disagree (%)'],
                    y= rating.index,
                    marker= dict (color ='#FF7043',line=dict(color='#FF7043',width=2)),
                    name = 'Strongly Disagree (%)',
                    orientation='h',
                    textposition='auto',
                )
            )


# Layout setting
fig.update_layout(
                    title=dict(text = "Best Programming Language",x=0.44,y=0.95,font_size=20),
                    barmode = 'stack',
                    width=1000,
                    height=500,
                    margin=dict(l=70, r=0, t=70, b=70),
                    paper_bgcolor='rgb(248, 248, 255)',
                    plot_bgcolor='rgb(248, 248, 255)',
                )

annotations =[]



# Displaying bar percentage label for "Strongly Agree"
for perc, lang in zip(rating.iloc[:,6],rating.index):
    # labeling the bar percentage
    annotations.append(dict(xref='x',
                            yref='y',
                            y=lang,
                            x=perc/2 ,
                            text=str(perc) + ' %',
                            font=dict(family='Arial', size=12,color='white'),
                            showarrow=False))


# Displaying bar percentage label for "Agree"
i=0
for perc, lang in zip(rating.iloc[:,7],rating.index):
    # labeling the bar percentage
    annotations.append(dict(xref='x',
                            yref='y',
                            y=lang,
                            x=perc/2 + rating.iloc[i,6],
                            text=str(perc) + ' %',
                            font=dict(family='Arial', size=12,color='white'),
                            showarrow=False))
    i+=1


# Displaying bar percentage label for "Neutral"
i=0
for perc, lang in zip(rating.iloc[:,8],rating.index):
    # labeling the bar percentage
    annotations.append(dict(xref='x',
                            yref='y',
                            y=lang,
                            x=perc/2 + rating.iloc[i,6]+rating.iloc[i,7],
                            text=str(perc) + ' %',
                            font=dict(family='Arial', size=12,color='black'),
                            showarrow=False))
```
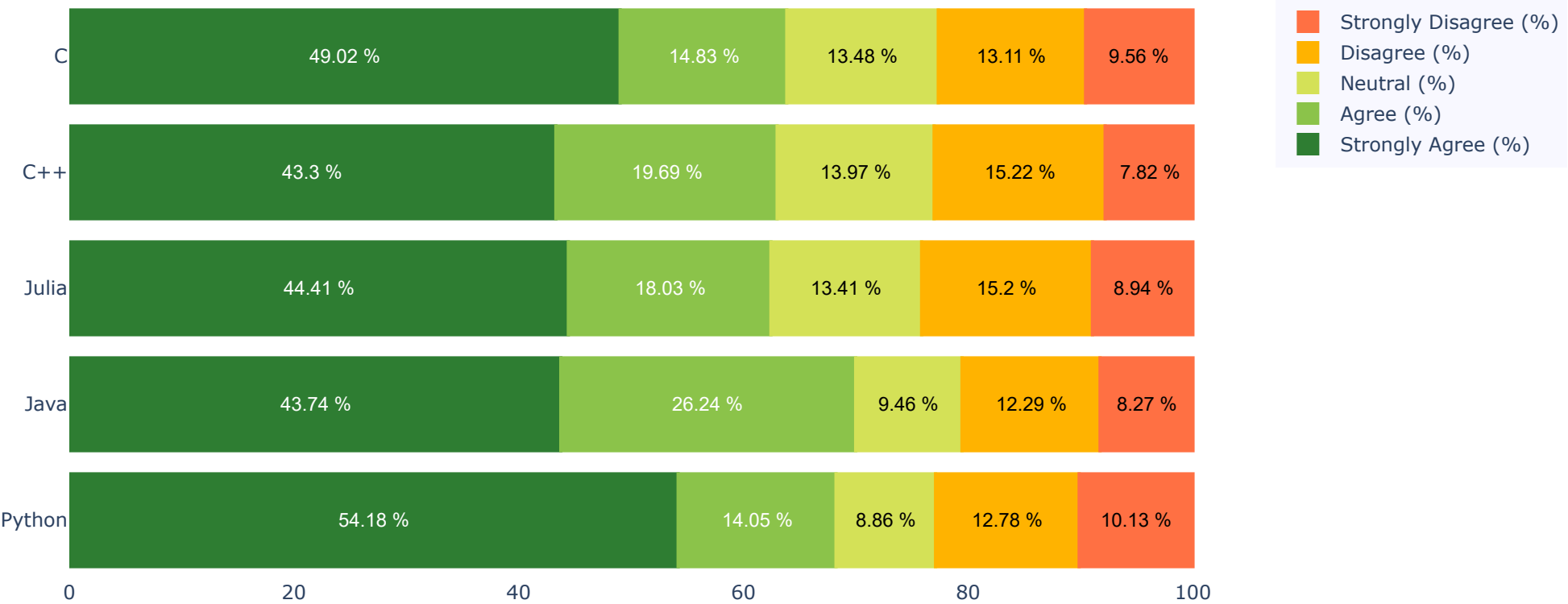
```python
        i+=1


# Displaying bar percentage label for "Disagree"
i=0
for perc, lang in zip(rating.iloc[:,9],rating.index):
    # labeling the bar percentage
    annotations.append(dict(xref='x',
                            yref='y',
                            y=lang,
                            x=perc/2 + rating.iloc[i,6]+rating.iloc[i,7]+rating.iloc[i,8],
                            text=str(perc) + ' %',
                            font=dict(family='Arial', size=12,color='black'),
                            showarrow=False))
    i+=1


# Displaying bar percentage label for "Strongly Disagree"
i=0
for perc, lang in zip(rating.iloc[:,10],rating.index):
    # labeling the bar percentage
    annotations.append(dict(xref='x',
                            yref='y',
                            y=lang,
                            x=perc/2 + rating.iloc[i,6]+rating.iloc[i,7]+rating.iloc[i,8] + rating.iloc[i,9],
                            text=str(perc) + ' %',
                            font=dict(family='Arial', size=12,color='black'),
                            showarrow=False))
    i+=1

fig.update_layout(annotations=annotations)

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```

Best Programming Language

	Strongly Agree (%)	Agree (%)	Neutral (%)	Disagree (%)	Strongly Disagree (%)
C	49.02 %	14.83 %	13.48 %	13.11 %	9.56 %
C++	43.3 %	19.69 %	13.97 %	15.22 %	7.82 %
Julia	44.41 %	18.03 %	13.41 %	15.2 %	8.94 %
Java	43.74 %	26.24 %	9.46 %	12.29 %	8.27 %
Python	54.18 %	14.05 %	8.86 %	12.78 %	10.13 %

```python
In [99]:  #Optimized code for above visualization
          fig = go.Figure()
          cols = ['#2E7D32' , '#8BC34A' , '#D4E157' , '#FFB300' , '#FF7043']

          for i in range (0,5):
              fig.add_trace(
                      go.Bar(
                              x= rating.iloc[:,6+i],
                              y= rating.index,
                              marker= dict (color =cols[i],line=dict(color=cols[i],width=2)),
                              name = 'Strongly Agree (%)',
                              orientation='h',
                              textposition='auto',
                              )
                      )

          fig.update_layout(
                          title=dict(text = "Best Programming Language",x=0.44,y=0.95,font_size=20),
                          barmode = 'stack',
                          width=1000,
                          height=500,
                          margin=dict(l=70, r=0, t=70, b=70),
                          paper_bgcolor='rgb(248, 248, 255)',
                          plot_bgcolor='rgb(248, 248, 255)',
                          )

          annotations =[]


          i=0
          for j in range(1,6):
              if j==1:
                  for perc, lang in zip(rating.iloc[:,5+j],rating.index):
                      # Labeling the bar percentage
                      annotations.append(dict(xref='x',
                                  yref='y',
                                  y=lang,
                                  x=perc/2 ,
                                  text=str(perc) + ' %',
                                  font=dict(family='Arial', size=12,color='white'),
                                  showarrow=False))
                  sum1 = rating.iloc[:,5+j]


              else:
                  i=0
                  for perc, lang in zip(rating.iloc[:,5+j],rating.index):
                      # Labeling the bar percentage
                      annotations.append(dict(xref='x',
                                  yref='y',
                                  y=lang,
                                  x=perc/2 + sum1[i],
                                  text=str(perc) + ' %',
                                  font=dict(family='Arial', size=12,color='black'),
                                  showarrow=False))
                      i+=1
                  sum1 = sum1+ rating.iloc[:,5+j]

          fig.update_layout(annotations=annotations)

          # Hide grid lines
```

```
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```

Best Programming Language
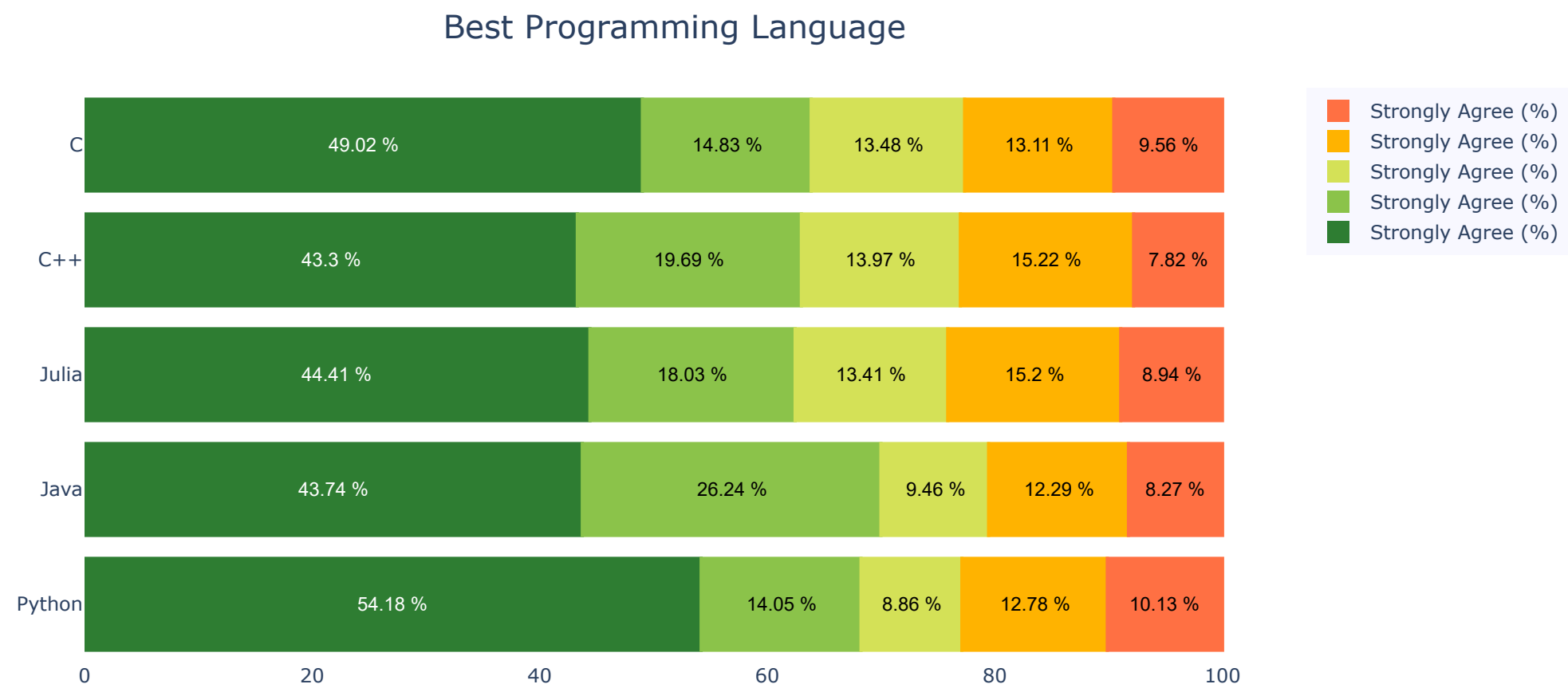
Pie & Donut Chart

```
In [399]: #Simple Pie Chart
          tickets = [48 , 30 , 20 , 15]
          status = ['Low' , 'Medium' , 'High' , 'Critical']
          data = go.Pie(
                          values= tickets,
                          labels= status,
                           )


          layout = go.Layout(
                              title=dict(text = "Tickets by Priority",x=0.46,y=0.95,font_size=20)
                              )

          fig = go.Figure(data=data,layout=layout)

          fig.show()
```

Tickets by Priority
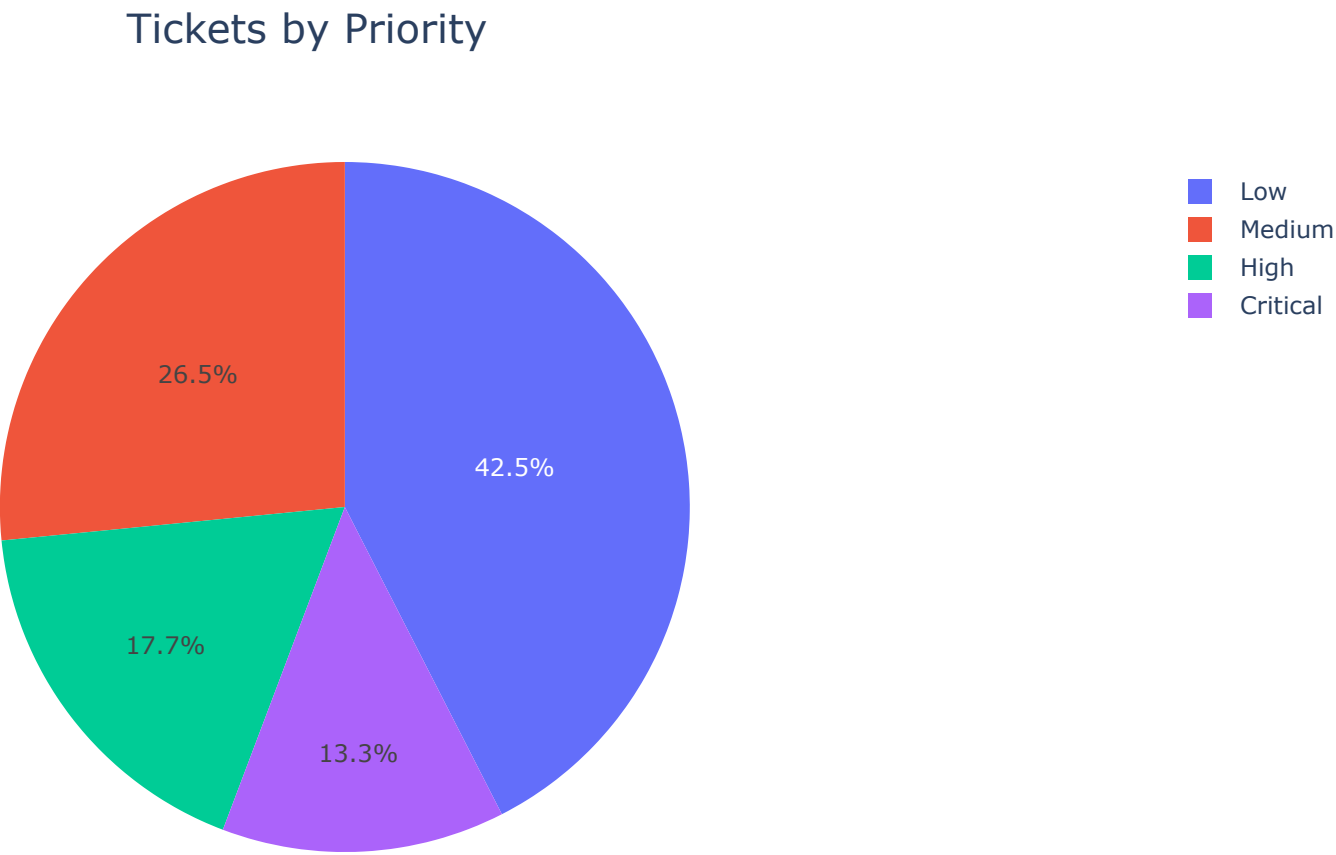
```python
#Changing color of Pie Chart
tickets = [48 , 30 , 20 , 15]
status = ['Low' , 'Medium' , 'High' , 'Critical']
colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']
data = go.Pie(
            values= tickets,
            labels= status,
            marker=dict(colors=colors),
            )


layout = go.Layout(
                title=dict(text = "Tickets by Priority",x=0.46,y=0.95,font_size=20),
                width=800,
                height=650
                )

fig = go.Figure(data=data,layout=layout)

fig.show()
```
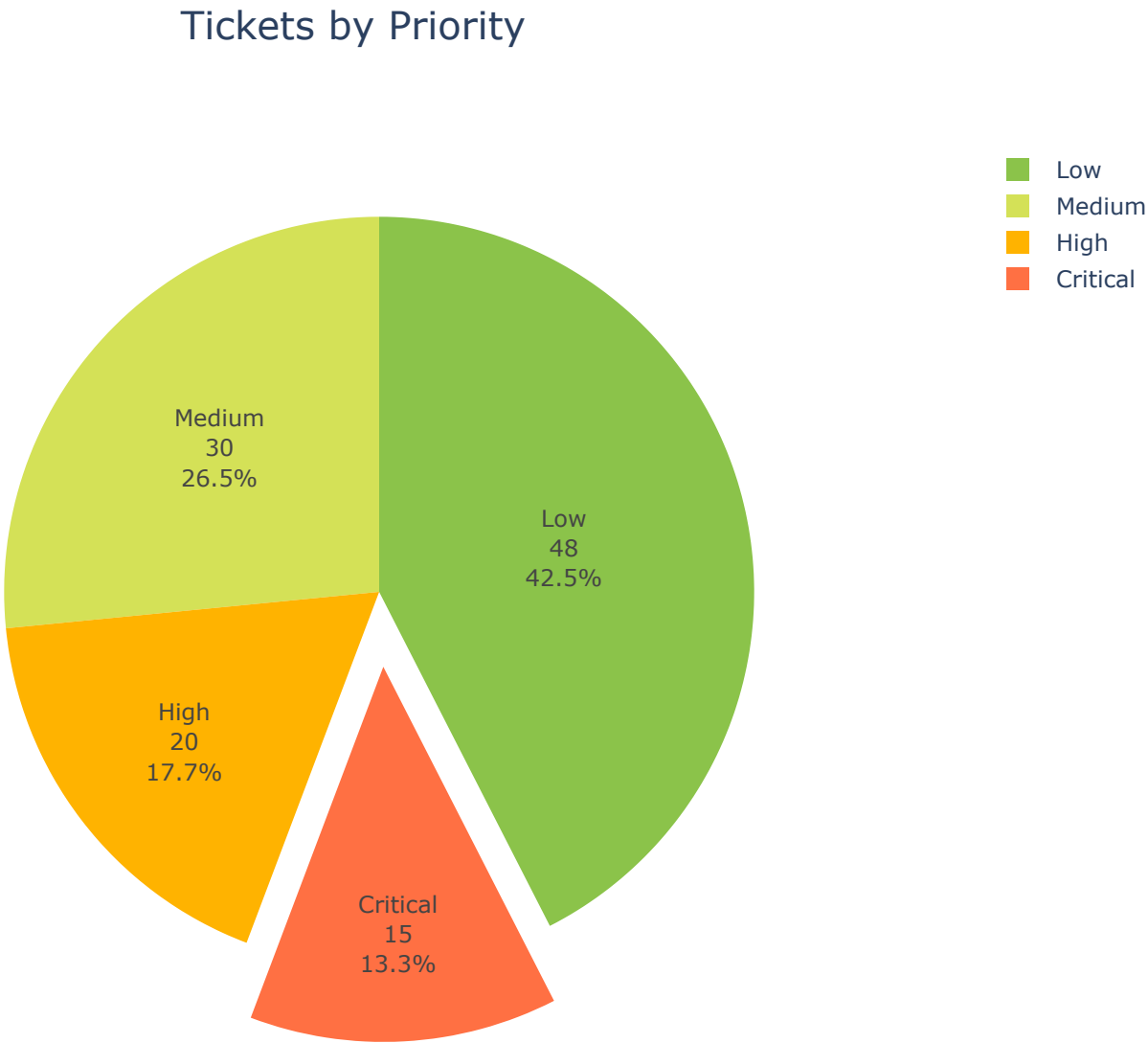
Tickets by Priority
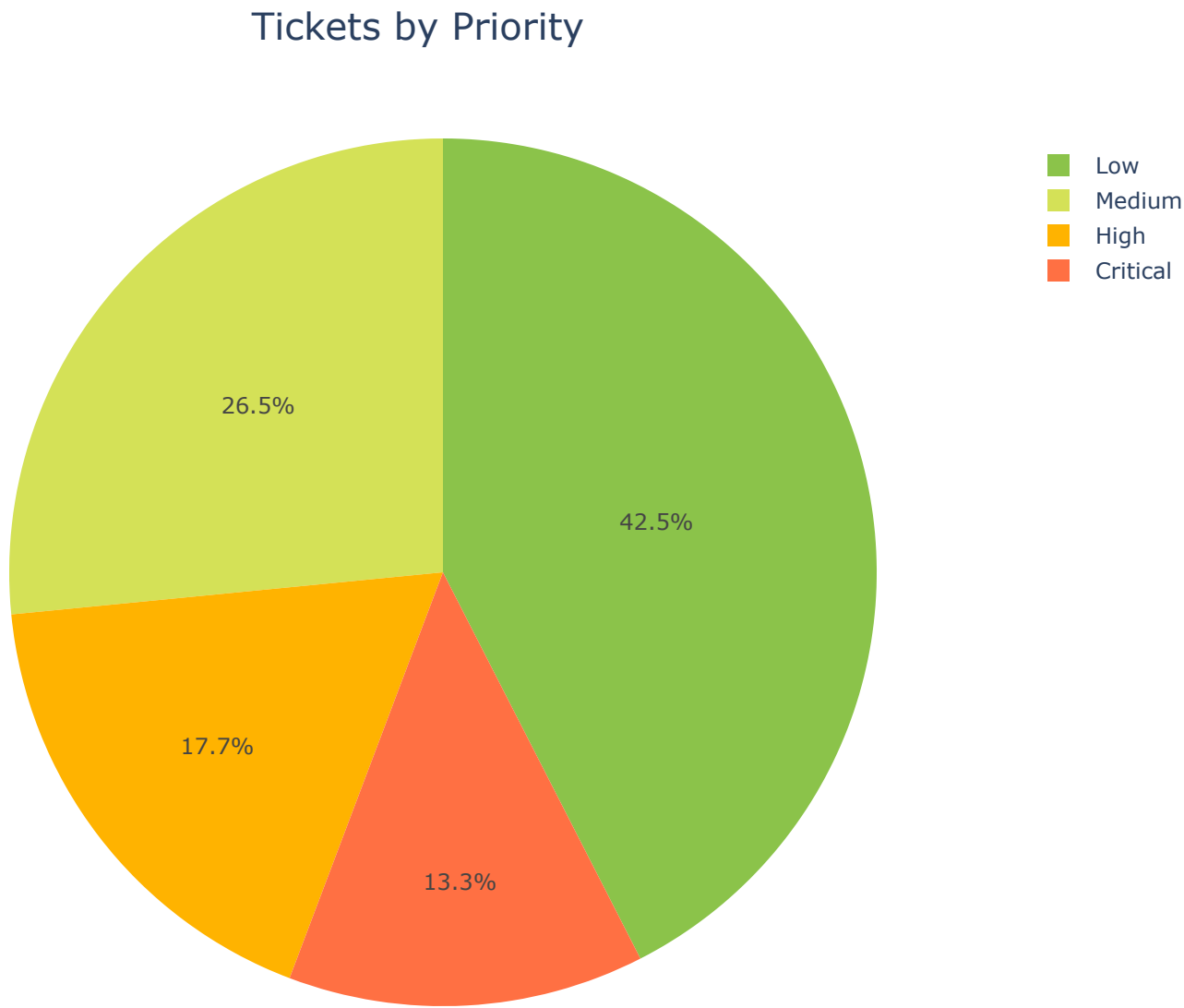
```python
# Displaying Label , Value & percent in Pie Chart using "textinfo" parameter
tickets = [48 , 30 , 20 , 15]
status = ['Low' , 'Medium' , 'High' , 'Critical']
colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']
data = go.Pie(
            values= tickets,
            labels= status,
            marker=dict(colors=colors),
            textinfo='label+value+percent'
            )


layout = go.Layout(
                title=dict(text = "Tickets by Priority",x=0.46,y=0.95,font_size=20),
                width=800,
                height=650
                )

fig = go.Figure(data=data,layout=layout)

fig.show()
```


Tickets by Priority
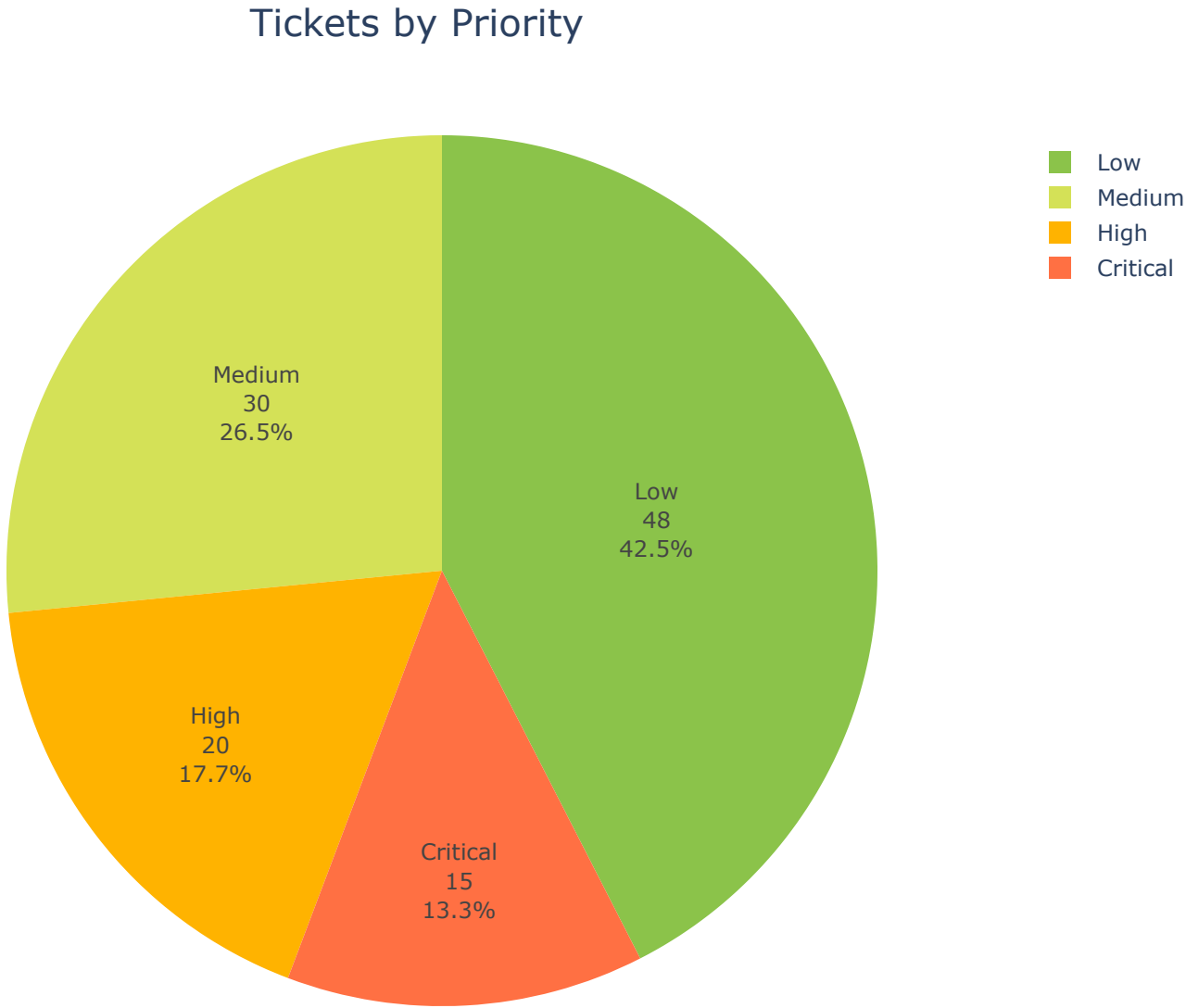
```python
In [233]: # Changing label orientation using "insidetextorientation" paramter
          tickets = [48 , 30 , 20 , 15]
          status = ['Low' , 'Medium' , 'High' , 'Critical']
          colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']
          data = go.Pie(
                        values= tickets,
                        labels= status,
                        marker=dict(colors=colors,line=dict(color='#797D7F', width=1.5)),
                        textinfo='percent',
                        hoverinfo='label+value',
                        textfont_size=15,
                        insidetextorientation='radial'
                        )



          layout = go.Layout(
                            title=dict(text = "Tickets by Priority",x=0.46,y=0.95,font_size=20),
                            width=800,
                            height=650
                            )

          fig = go.Figure(data=data,layout=layout)

          fig.show()
```

Tickets by Priority

Legend:
- Low
- Medium
- High
- Critical

Low: 42.5%
Medium: 26.5%
High: 17.7%
Critical: 13.3%
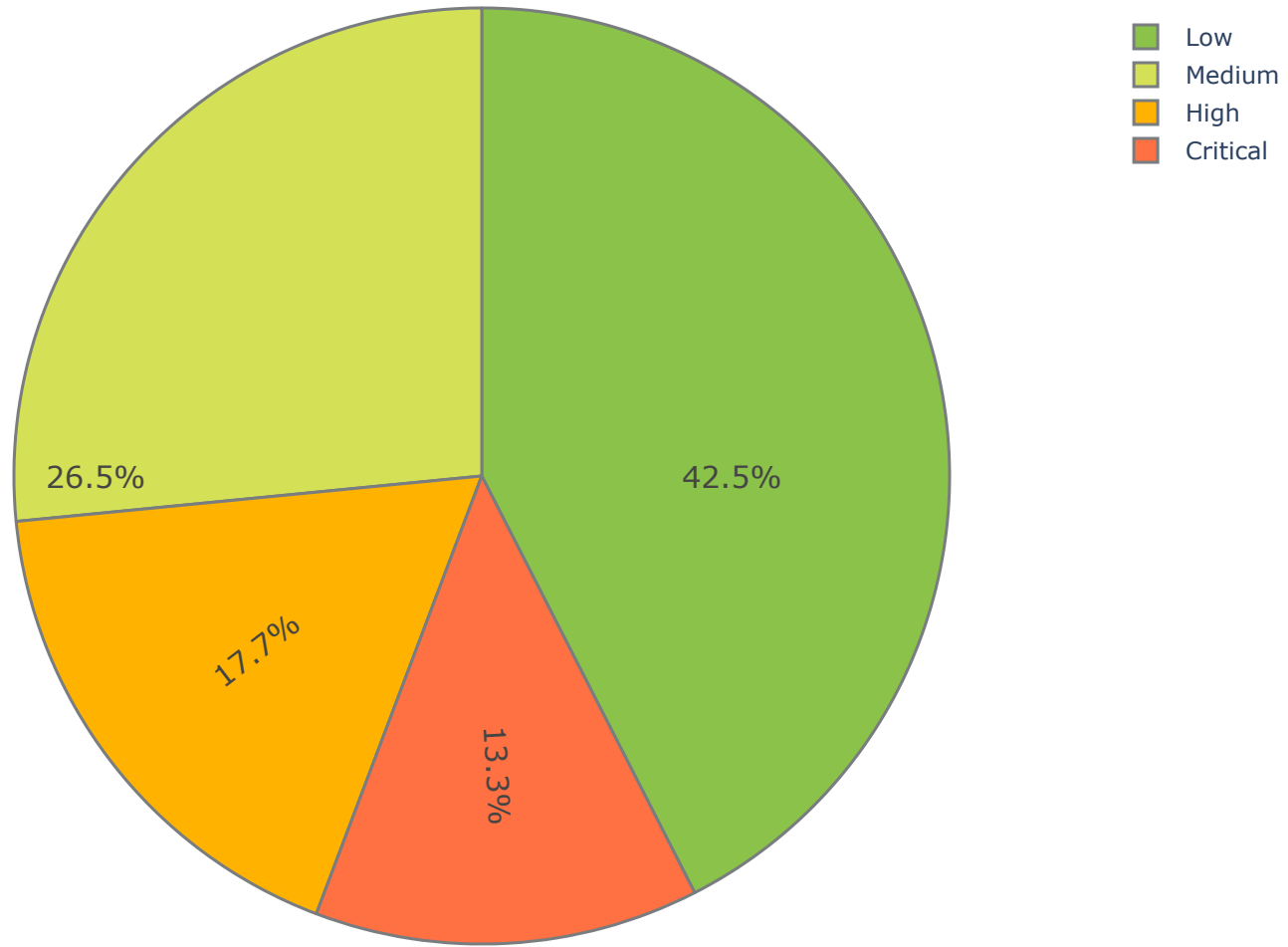
In [234]:
```python
#Explode 4th Slice using "pull" parameter
tickets = [48 , 30 , 20 , 15]
status = ['Low' , 'Medium' , 'High' , 'Critical']
colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']
data = go.Pie(
            values= tickets,
            labels= status,
            marker=dict(colors=colors),
            textinfo='label+value+percent',
            pull=[0, 0, 0, 0.2] #Explode 4th Slice
            )


layout = go.Layout(
            title=dict(text = "Tickets by Priority",x=0.46,y=0.95,font_size=20),
            width=800,
            height=650
            )

fig = go.Figure(data=data,layout=layout)

fig.show()
```


Tickets by Priority

```
In [235]:  #Explode 3rd & 4th Slice using "pull" parameter
           tickets = [48 , 30 , 20 , 15]
           status = ['Low' , 'Medium' , 'High' , 'Critical']
           colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']
           data = go.Pie(
                       values= tickets,
                       labels= status,
                       marker=dict(colors=colors),
                       textinfo='label+value+percent',
                       pull=[0, 0, 0.2, 0.2]
                       )


           layout = go.Layout(
                       title=dict(text = "Tickets by Priority",x=0.46,y=0.95,font_size=20),
                       width=800,
                       height=650
                       )

           fig = go.Figure(data=data,layout=layout)

           fig.show()
```
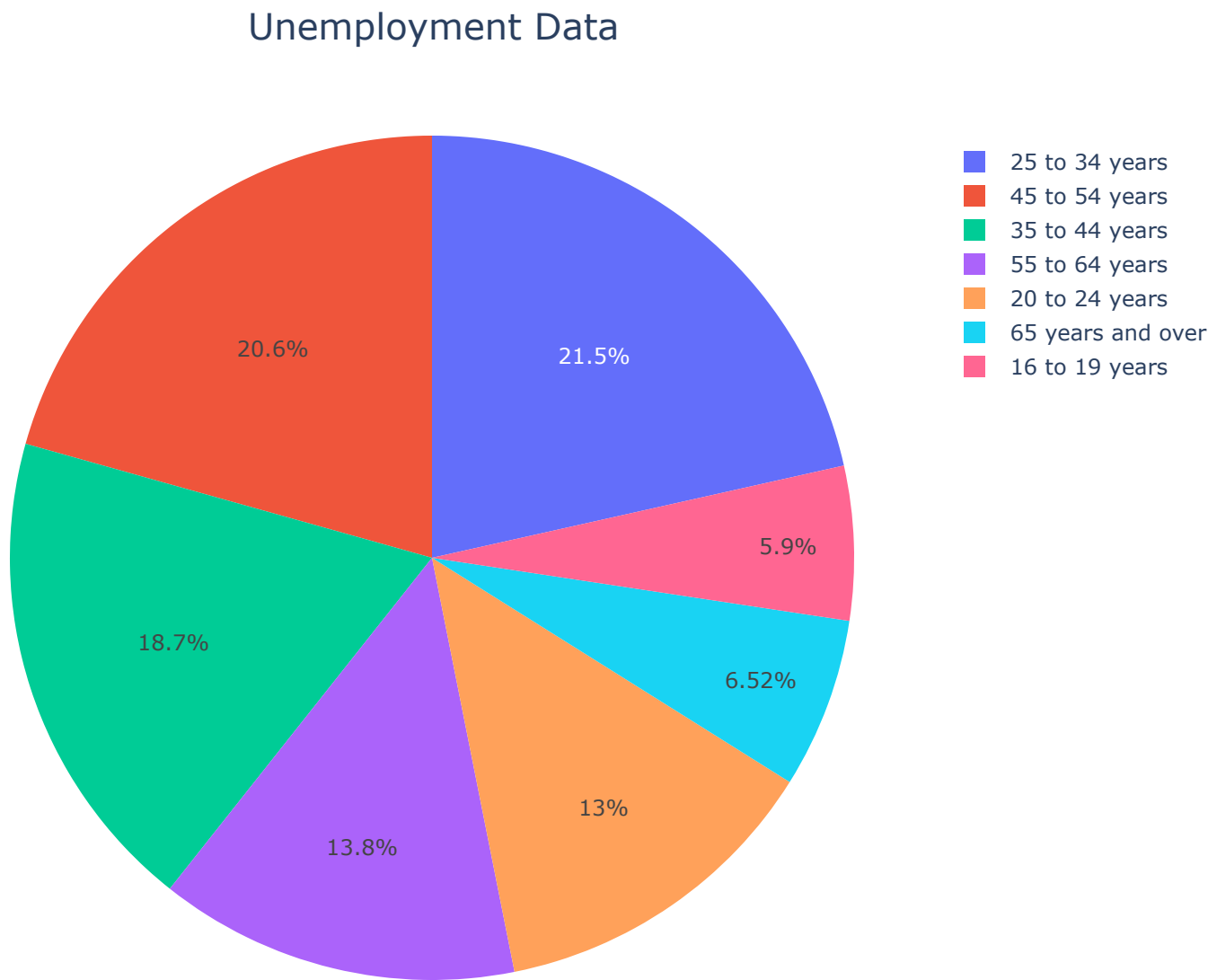

Tickets by Priority
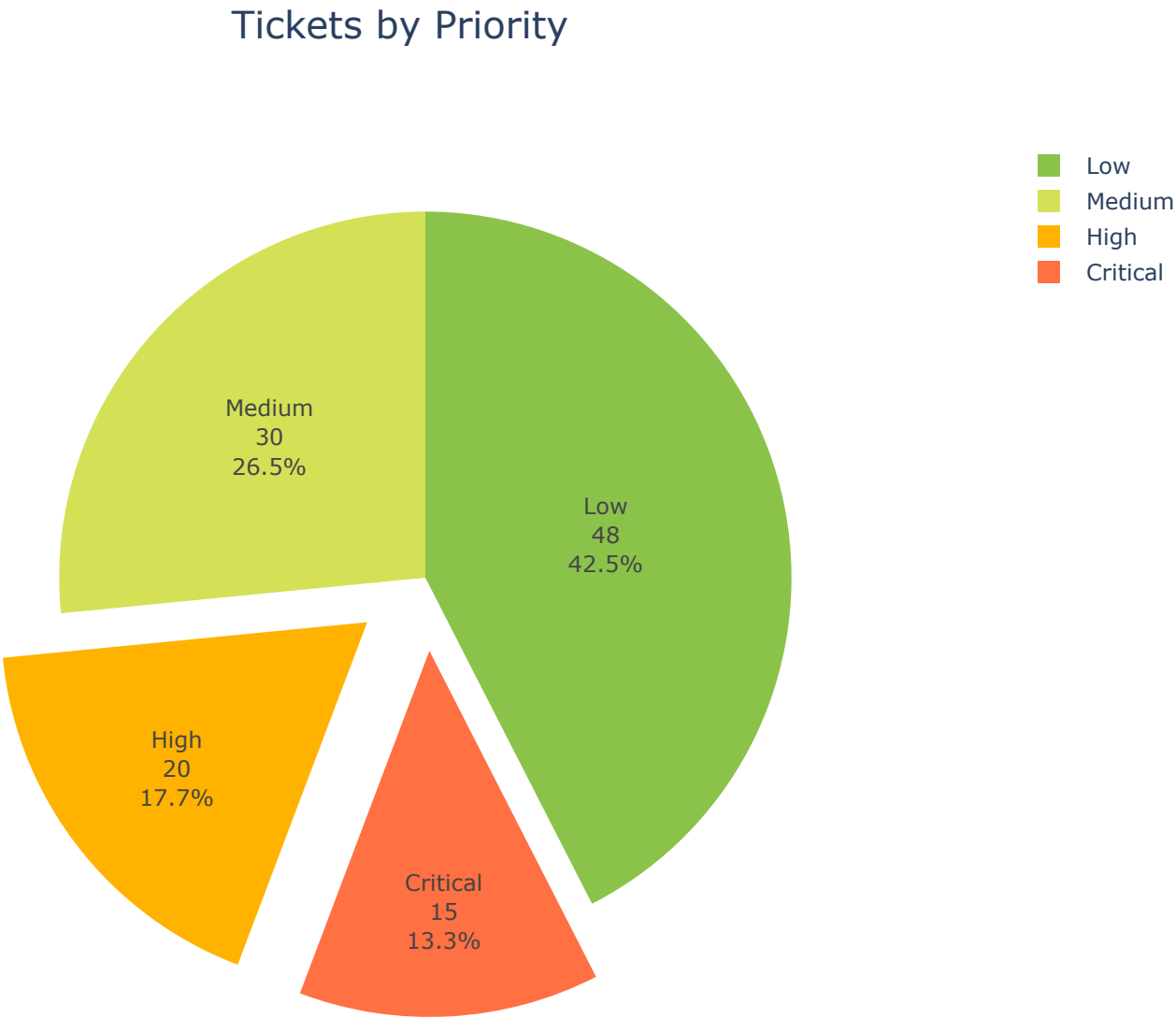
```
In [236]:  # Simple Donut Chart
           tickets = [48 , 30 , 20 , 15]
           status = ['Low' , 'Medium' , 'High' , 'Critical']
           colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']
           data = go.Pie(
                        values= tickets,
                        labels= status,
                        marker=dict(colors=colors),
                        textinfo='label+value+percent',
                        hole=.4
                        )


           layout = go.Layout(
                        title=dict(text = "Tickets by Priority",x=0.46,y=0.95,font_size=20),
                        width=800,
                        height=650
                        )

           fig = go.Figure(data=data,layout=layout)

           fig.show()
```

Tickets by Priority
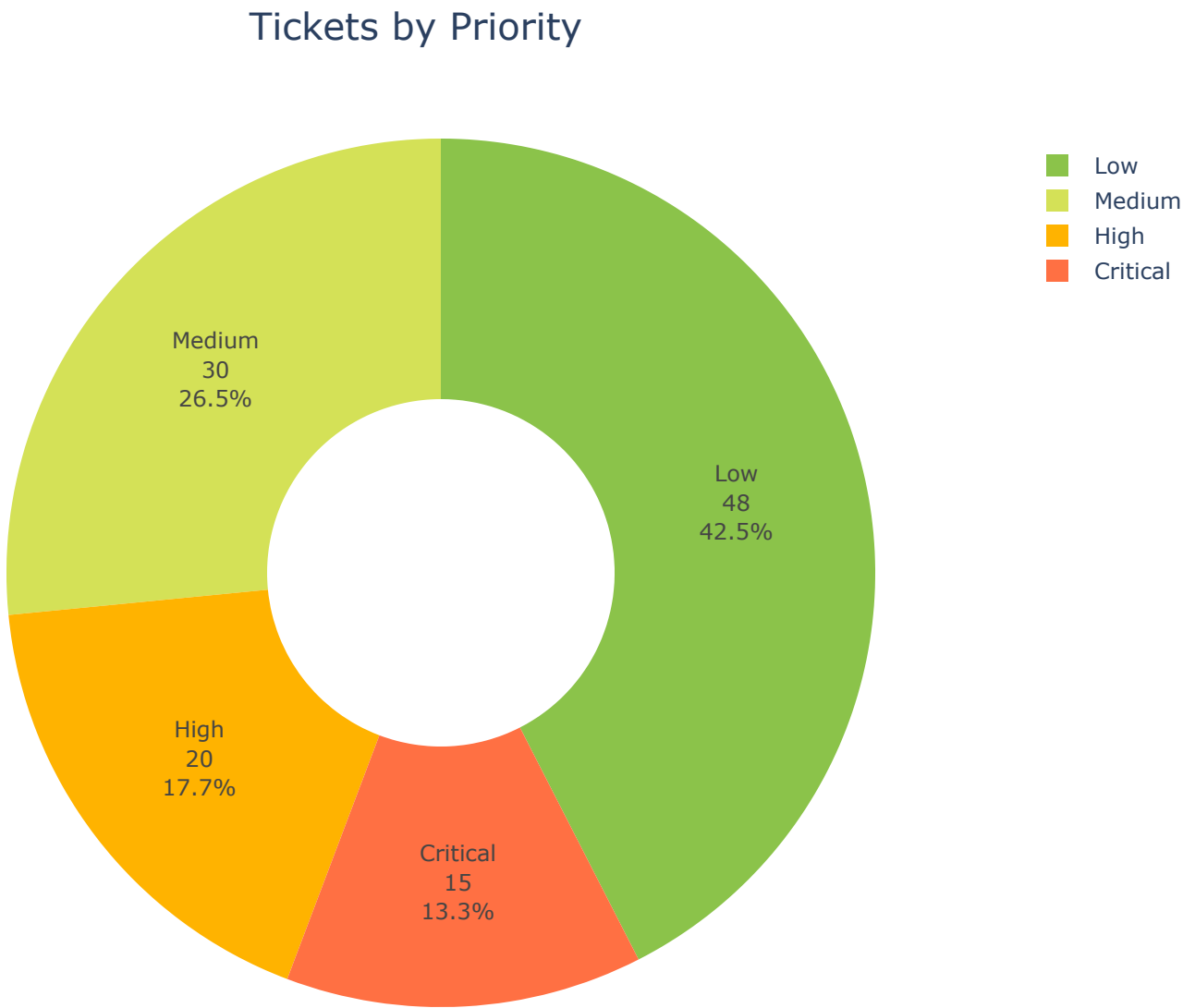
```python
In [237]:  #Explode 4th Slice using "pull" parameter
           tickets = [48 , 30 , 20 , 15]
           status = ['Low' , 'Medium' , 'High' , 'Critical']
           colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']
           data = go.Pie(
                           values= tickets,
                           labels= status,
                           marker=dict(colors=colors),
                           textinfo='label+value+percent',
                           hole=.4,
                           pull=[0, 0, 0, 0.2]
                           )


           layout = go.Layout(
                               title=dict(text = "Tickets by Priority",x=0.46,y=0.95,font_size=20),
                               width=800,
                               height=650
                               )

           fig = go.Figure(data=data,layout=layout)

           fig.show()
```

Tickets by Priority

- Low
- Medium
- High
- Critical

Medium
30
26.5%

Low
48
42.5%

High
20
17.7%

Critical
15
13.3%
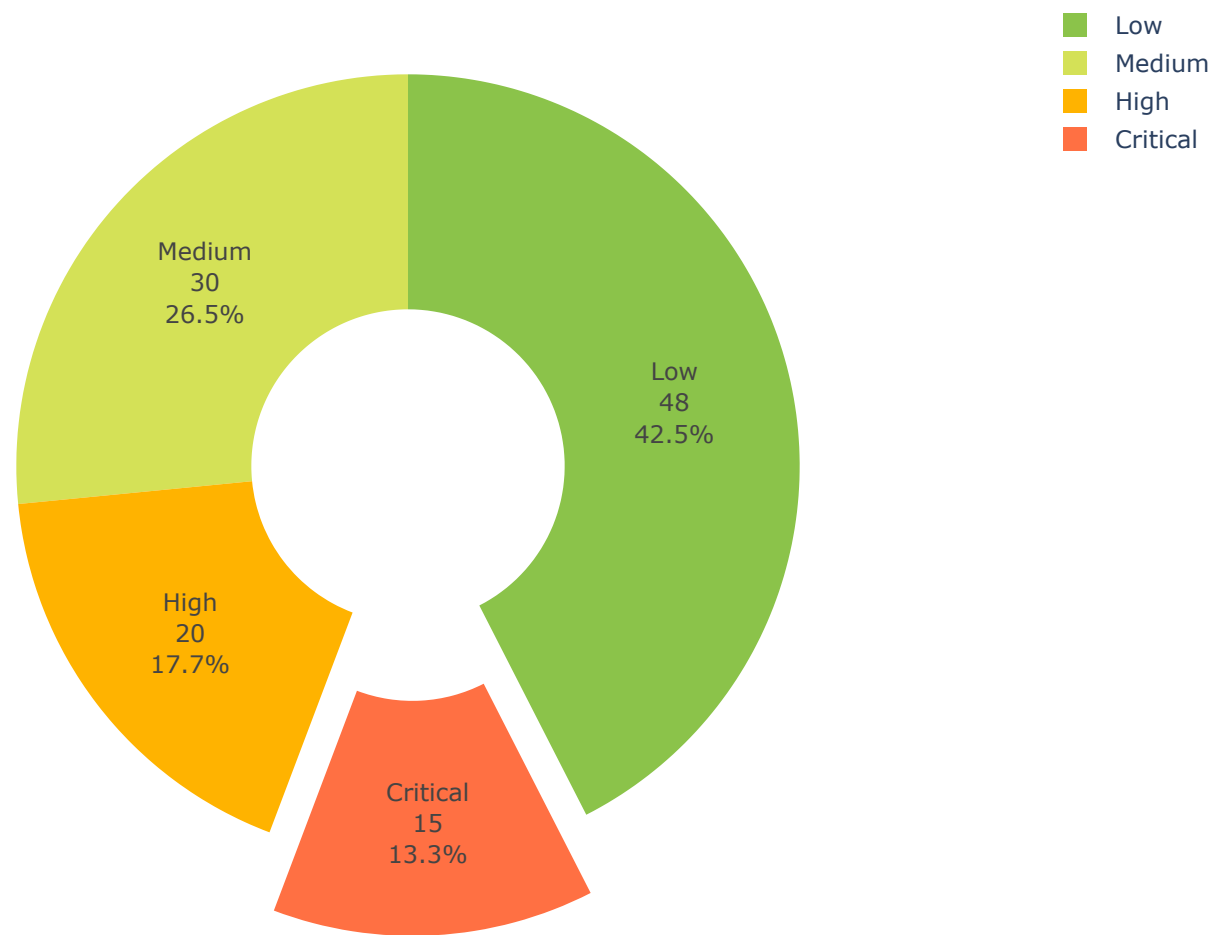
In [401]: `employment.head()`

Out[401]:

	Age	Gender	Period	Unemployed
0	16 to 19 years	Men	2005-01-01	91000
1	20 to 24 years	Men	2005-01-01	175000
2	25 to 34 years	Men	2005-01-01	194000
3	35 to 44 years	Men	2005-01-01	201000
4	45 to 54 years	Men	2005-01-01	207000

```
In [402]: data = go.Pie(
                    values= employment.Unemployed,
                    labels= employment.Age,
                    textinfo='percent',
                    hoverinfo='label+value'
                    )
          layout = go.Layout(
                        title=dict(text = "Unemployment Data",x=0.44,y=0.95,font_size=20),
                        width=800,
                        height=650
                        )

          fig = go.Figure(data=data,layout=layout)

          fig.show()
```

Unemployment Data

Legend:
- 25 to 34 years
- 45 to 54 years
- 35 to 44 years
- 55 to 64 years
- 20 to 24 years
- 65 years and over
- 16 to 19 years

Pie chart values: 21.5%, 20.6%, 18.7%, 13.8%, 13%, 6.52%, 5.9%

```
In [403]: suicide.head()
```

Out[403]:

	country	year	sex	age	suicides_no	population	suicides/100k pop	country-year	HDI for year	gdp_for_year ($)	gdp_per_capita ($)	generation
0	Albania	1987	male	15-24 years	21	312900	6.71	Albania1987	NaN	2,156,624,900	796	Generation X
1	Albania	1987	male	35-54 years	16	308000	5.19	Albania1987	NaN	2,156,624,900	796	Silent
2	Albania	1987	female	15-24 years	14	289700	4.83	Albania1987	NaN	2,156,624,900	796	Generation X
3	Albania	1987	male	75+ years	1	21800	4.59	Albania1987	NaN	2,156,624,900	796	G.I. Generation
4	Albania	1987	male	25-34 years	9	274300	3.28	Albania1987	NaN	2,156,624,900	796	Boomers

```
In [404]: data = go.Pie(
                      values= suicide['suicides_no'],
                      labels= suicide['age'],
                      textinfo='percent',
                      hoverinfo='label+value'
                      )
          layout = go.Layout(
                      title=dict(text = "Suicide Data",x=0.44,y=0.95,font_size=20),
                      width=800,
                      height=650
                      )

          fig = go.Figure(data=data,layout=layout)

          fig.show()
```

Suicide Data
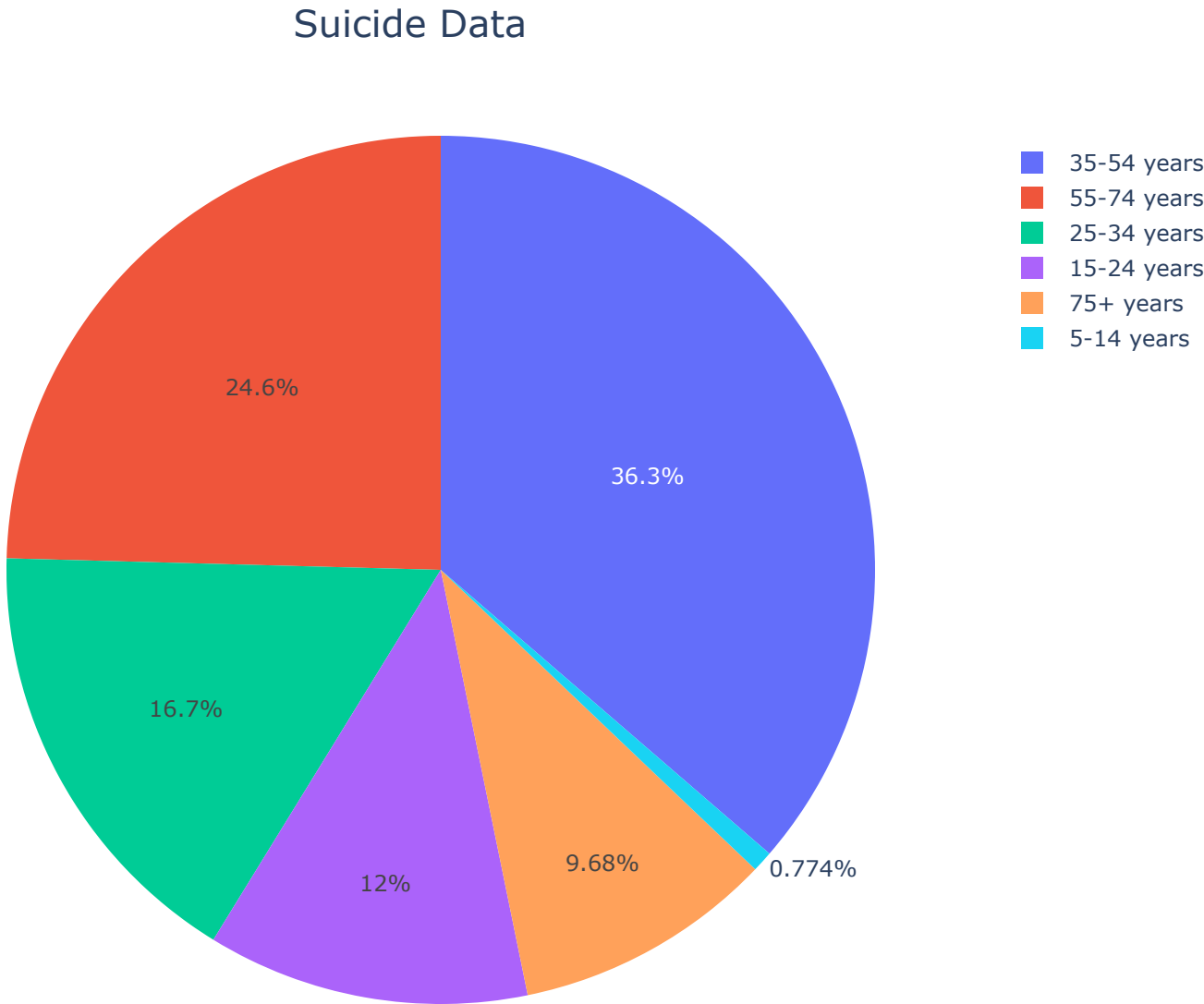
```python
# Display multiple Pie plots in one figure using Subplots

tickets = [48 , 30 , 20 , 15]
status = ['Low' , 'Medium' , 'High' , 'Critical']
colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']

Assignee = ['Asif','Basit','John','Batista', 'Gabriel' , 'Paul' , 'Ravi']
Open = [17,18,29,33,38,39,42]




#Subplot initialization
fig = make_subplots(
                    rows=1,
                    cols=2,
                    subplot_titles=("Tickets by Priority", "Tickets by Assignee"),

                    specs=[[{'type':'domain'}, {'type':'domain'}]]
                )


# Subplot - 1 (Add graph object trace to a figure)
fig.add_trace(go.Pie(
                    values= tickets,
                    labels= status,
                    marker=dict(colors=colors),
                    textinfo='label+value+percent'
                ),
            row=1, col=1
            )


fig.add_trace(go.Pie(
                    values= Open,
                    labels= Assignee,
                    marker=dict(colors=colors),
                    textinfo='label+value+percent'
                ),
            row=1, col=2
            )

fig.update_layout(
                    paper_bgcolor= '#FFFDE7',
                    plot_bgcolor= '#FFFDE7',
                    title=dict(text = "Help Desk",x=0.5,y=0.95),
                    title_font_size=30
                )

fig.show()
```
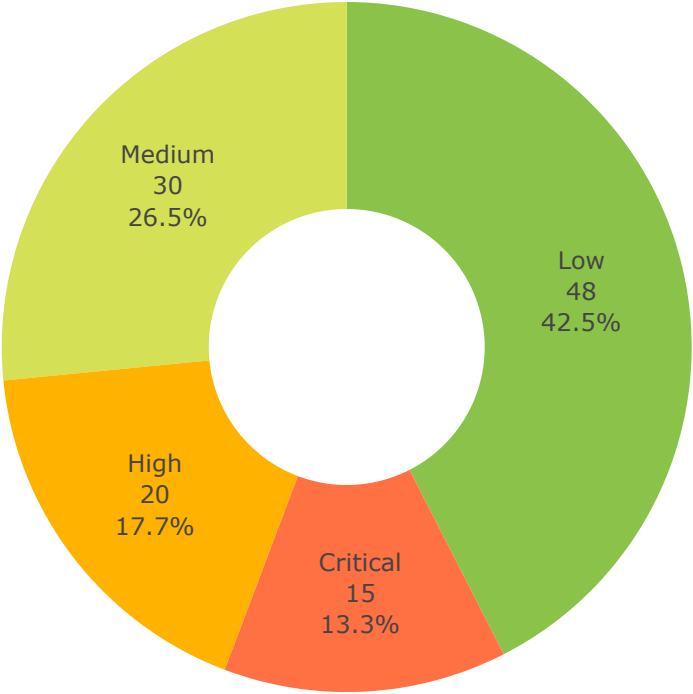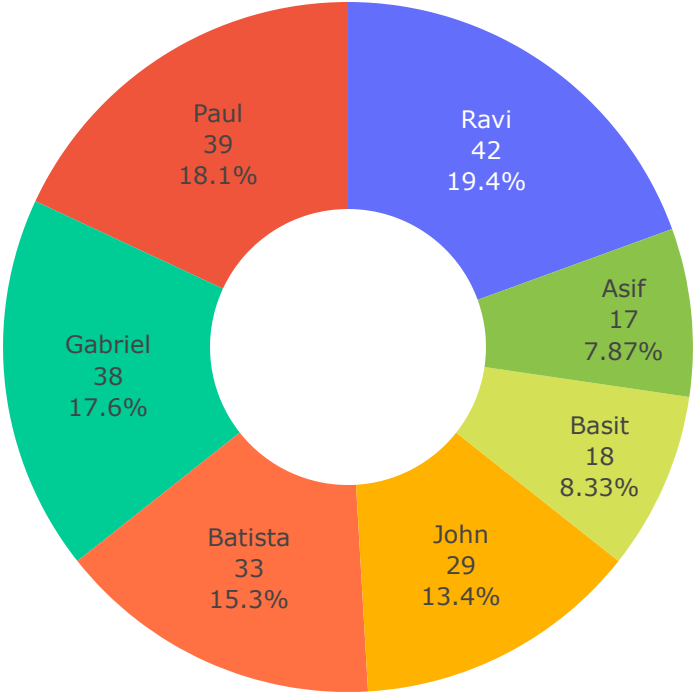
Help Desk

Tickets by Priority

Low	48 — 42.5%
Medium	30 — 26.5%
High	20 — 17.7%
Critical	15 — 13.3%

Tickets by Assignee

Ravi	42 — 19.4%
Paul	39 — 18.1%
Gabriel	38 — 17.6%
Batista	33 — 15.3%
John	29 — 13.4%
Basit	18 — 8.33%
Asif	17 — 7.87%
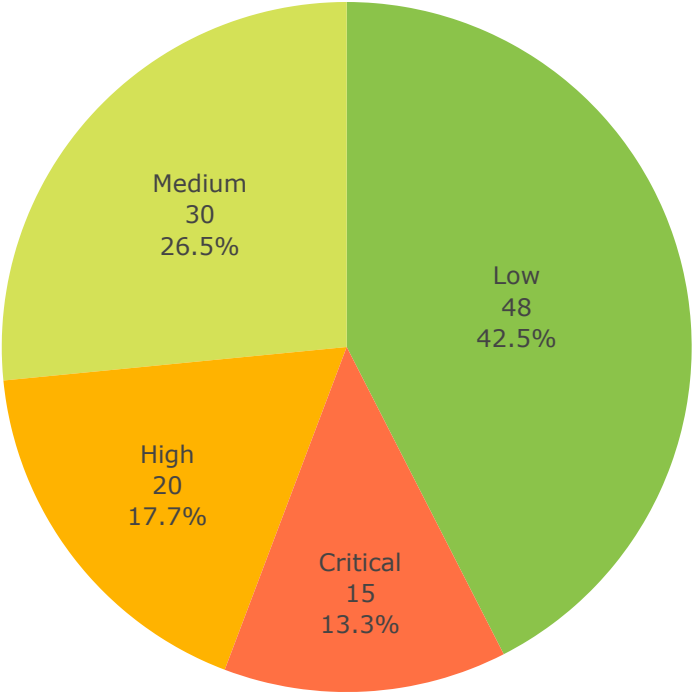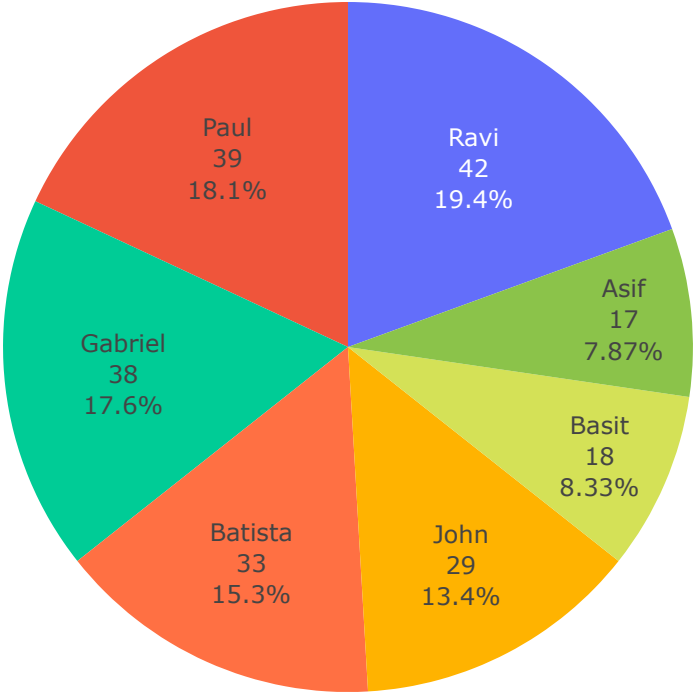
Legend: Low, Medium, High, Critical, Ravi, Paul, Gabriel, Batista, John, Basit, Asif

```
In [243]:  # Display multiple Donut charts in one figure using Subplots

           tickets = [48 , 30 , 20 , 15]
           status = ['Low' , 'Medium' , 'High' , 'Critical']
           colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']

           Assignee = ['Asif','Basit','John','Batista', 'Gabriel' , 'Paul' , 'Ravi']
           Open = [17,18,29,33,38,39,42]




           #Subplot initialization
           fig = make_subplots(
                           rows=1,
                           cols=2,
                           subplot_titles=("Tickets by Priority", "Tickets by Assignee"),
                           specs=[[{'type':'domain'}, {'type':'domain'}]]
                           )


           # Subplot - 1 (Add graph object trace to a figure)
           fig.add_trace(go.Pie(
                           values= tickets,
                           labels= status,
                           hole = .4,
                           marker=dict(colors=colors),
                           textinfo='label+value+percent',
                           hoverinfo='label'
                           ),
                     row=1, col=1
                     )



           fig.add_trace(go.Pie(
                           values= Open,
                           labels= Assignee,
                           hole = .4,
                           marker=dict(colors=colors),
                           textinfo='label+value+percent',
                           hoverinfo='label'
                           ),
                     row=1, col=2
                     )

           fig.update_layout(
                           paper_bgcolor= '#FFFDE7',
                           plot_bgcolor= '#FFFDE7',
                           title=dict(text = "Help Desk",x=0.5,y=0.95),
                           title_font_size=30
                           )

           fig.show()
```

Help Desk

Tickets by Priority

- Low: 48 — 42.5%
- Medium: 30 — 26.5%
- High: 20 — 17.7%
- Critical: 15 — 13.3%

Tickets by Assignee

- Ravi: 42 — 19.4%
- Paul: 39 — 18.1%
- Gabriel: 38 — 17.6%
- Batista: 33 — 15.3%
- John: 29 — 13.4%
- Basit: 18 — 8.33%
- Asif: 17 — 7.87%

Legend: Low, Medium, High, Critical, Ravi, Paul, Gabriel, Batista, John, Basit, Asif

```python
In [407]:   # Display multiple Pie & Donut plots in one figure using Subplots

            tickets = [48 , 30 , 20 , 15]
            priority = ['Low' , 'Medium' , 'High' , 'Critical']
            colors = ['#8BC34A','#D4E157','#FFB300','#FF7043']
            group = ['Service Desk' , 'Problem' , 'Application' , 'Change']
            status = ['Assigned', 'Pending' , 'New' , 'In Progress']
            severity = ['Sev-4' , 'Sev-3' , 'sev-2' , 'sev-1']



            #Subplot initialization
            fig = make_subplots(
                              rows=2,
                              cols=2,
                              subplot_titles=("Tickets by Priority", "Tickets by Severity",
                                              "Tickets by Group", "Tickets by Status"),
                              specs=[[{'type':'domain'}, {'type':'domain'}],[{'type':'domain'}, {'type':'domain'}]]
                          )

            #Change Subplot title font size
            for i in fig['layout']['annotations']:
                i['font']['size'] = 17

            # Subplot - 1 (Add graph object trace to a figure)
            fig.add_trace(go.Pie(
                              values= tickets,
                              labels= priority,
                              marker=dict(colors=colors),
                              textinfo='label+value+percent',
                              hoverinfo='label',
                              ),
                          row=1, col=1
                          )


            # Subplot - 2 (Add graph object trace to a figure)
            fig.add_trace(go.Pie(
                              values= Open,
                              labels= severity,
                              marker=dict(colors=colors),
                              textinfo='label+value+percent',
                              hoverinfo='label',
                              ),
                          row=1, col=2
                          )


            # Subplot - 3 (Add graph object trace to a figure)
            fig.add_trace(go.Pie(
                              values= tickets,
                              labels= group,
                              hole = .4,
                              marker=dict(colors=colors),
                              textinfo='label+value+percent',
                              hoverinfo='label'
                              ),
                          row=2, col=1
                          )
```
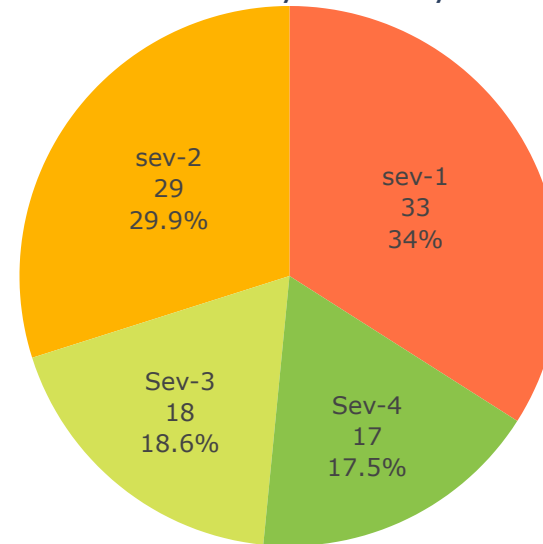
```python
# Subplot - 4 (Add graph object trace to a figure)
fig.add_trace(go.Pie(
                    values= Open,
                    labels= status,
                    hole = .4,
                    marker=dict(colors=colors),
                    textinfo='label+value+percent',
                    hoverinfo='label'
                    ),
             row=2, col=2
             )

fig.update_layout(
                    paper_bgcolor= '#FFFDE7',
                    plot_bgcolor= '#FFFDE7',
                    title=dict(text = "Help Desk",x=0.49,y=0.97,font_size=30),
                    width=950,
                    height=900,
                    showlegend=False
                )

fig.show()
```
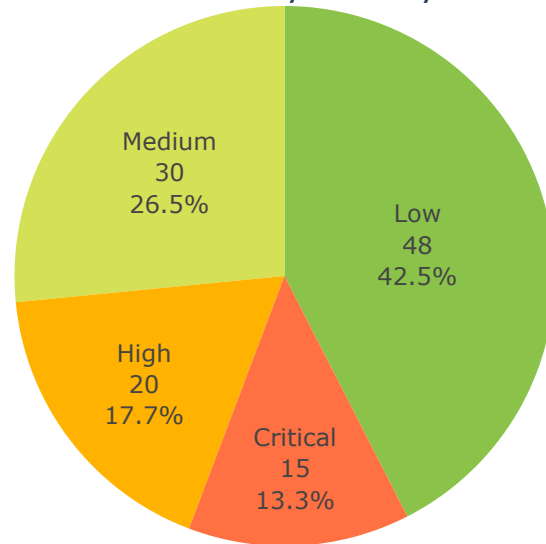
Help Desk

Tickets by Priority

Medium
30
26.5%

Low
48
42.5%

High
20
17.7%

Critical
15
13.3%

Tickets by Severity

sev-2
29
29.9%

sev-1
33
34%

Sev-3
18
18.6%

Sev-4
17
17.5%

Tickets by Group

Problem
30
26.5%

Service Desk
48
42.5%

Application
20
17.7%

Change
15
13.3%

Tickets by Status

New
29
29.9%

In Progress
33
34%

Pending
18
18.6%

Assigned
17
17.5%

Pie Chart using Plotly Express

```
In [408]: suicide.head()
```

Out[408]:

	country	year	sex	age	suicides_no	population	suicides/100k pop	country-year	HDI for year	gdp_for_year ($)	gdp_per_capita ($)	generation
0	Albania	1987	male	15-24 years	21	312900	6.71	Albania1987	NaN	2,156,624,900	796	Generation X
1	Albania	1987	male	35-54 years	16	308000	5.19	Albania1987	NaN	2,156,624,900	796	Silent
2	Albania	1987	female	15-24 years	14	289700	4.83	Albania1987	NaN	2,156,624,900	796	Generation X
3	Albania	1987	male	75+ years	1	21800	4.59	Albania1987	NaN	2,156,624,900	796	G.I. Generation
4	Albania	1987	male	25-34 years	9	274300	3.28	Albania1987	NaN	2,156,624,900	796	Boomers

```
In [409]:  fig = px.pie(suicide, values='suicides_no', names='generation', title='Country')

           fig.update_layout(
                        title=dict(text = "Suicide Data",x=0.44,y=0.95,font_size=20),
                        width=800,
                        height=650
                    )

           fig.update_traces(textposition='outside', textinfo='percent+label')

           fig.show()
```
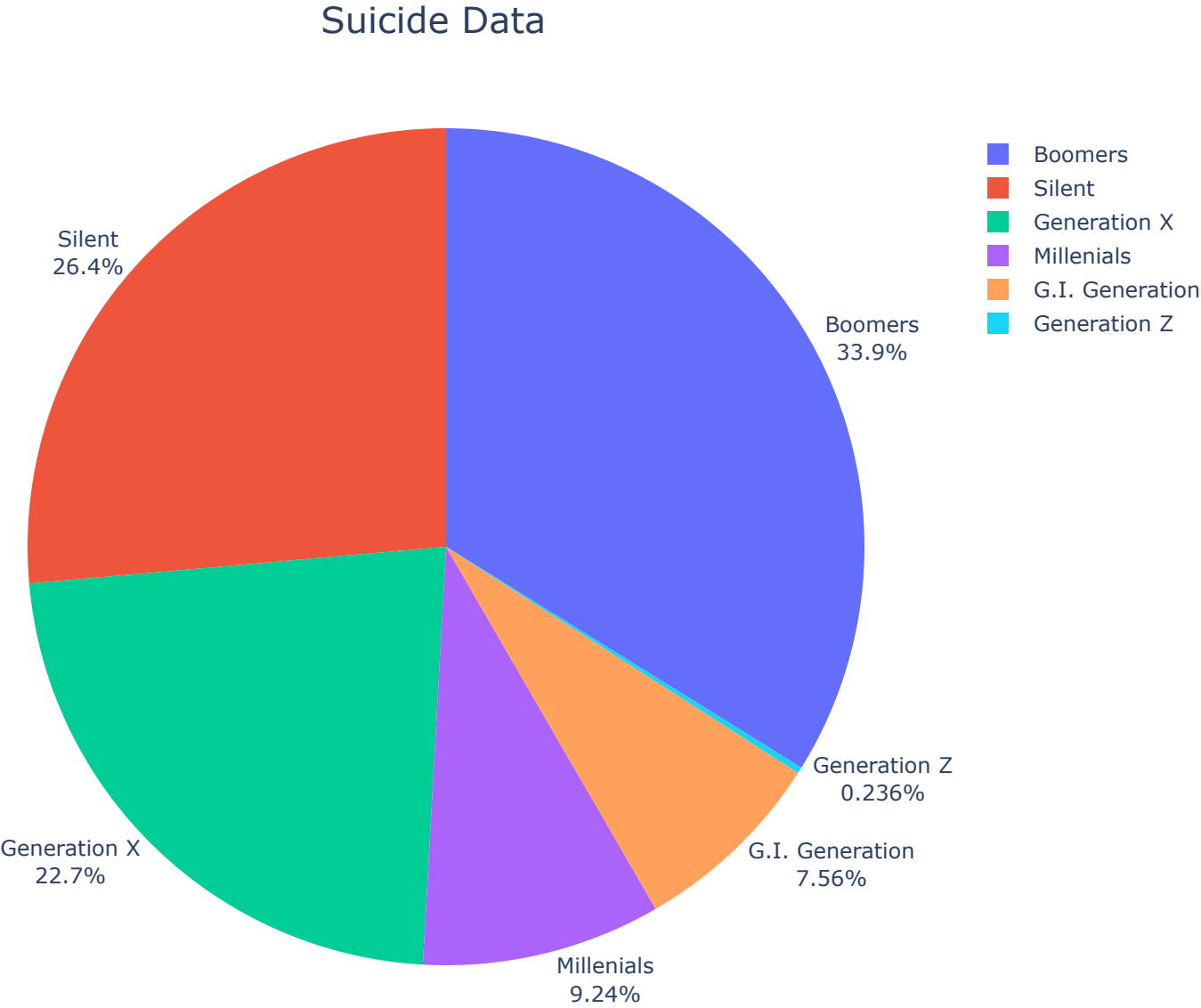

Bubble Chart

In [101]:
```python
#Simple Bubble Chart
x = np.arange(1,10)
y = np.random.randint(1,100,9)
op = np.random.uniform(0.2,1 ,9)
data = go.Scatter(
                x = x,
                y = y,
                mode = 'markers',
                marker = dict(size = y),
                )

fig = go.Figure(data=data)

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
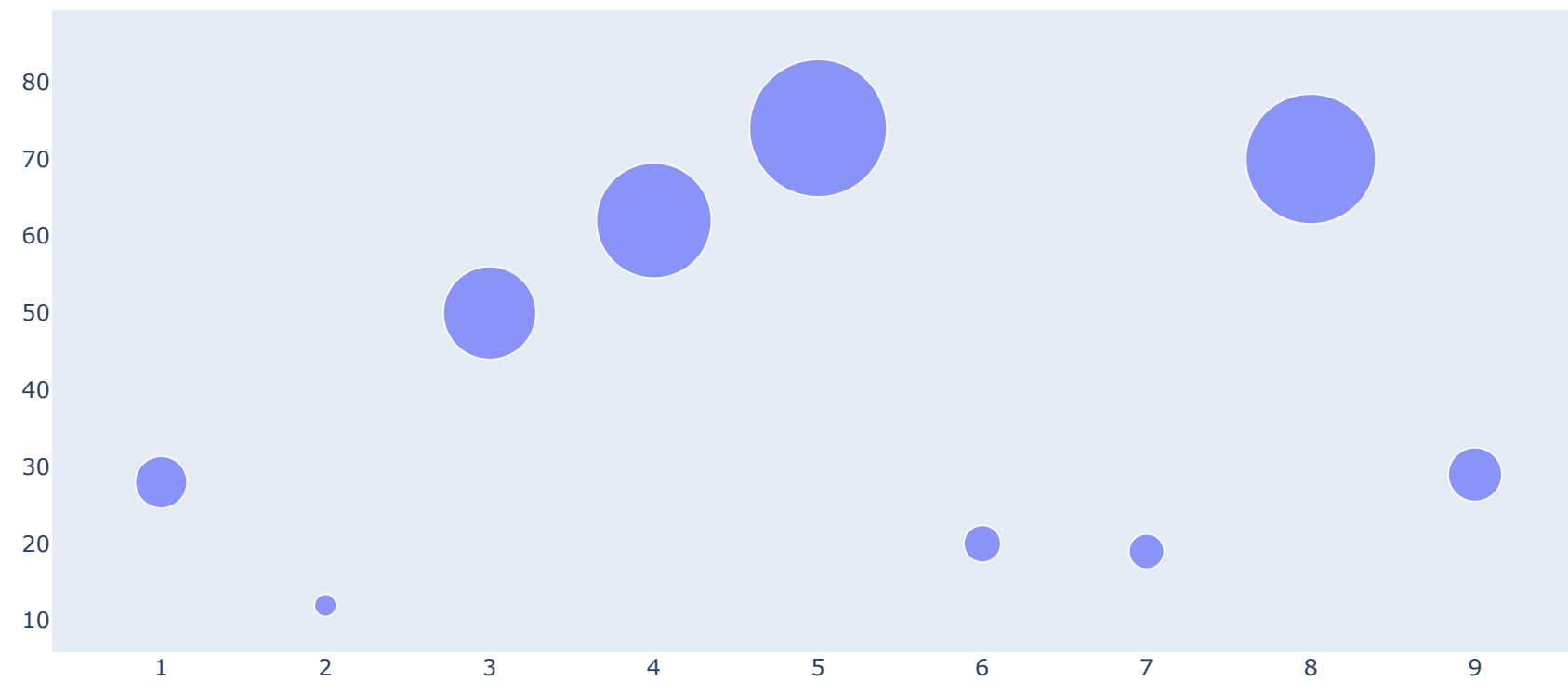
```python
x = np.arange(1,10)
y = np.random.randint(1,100,9)
op = np.random.uniform(0.2,1 ,9)
data = go.Scatter(
                x = x,
                y = y,
                mode = 'markers',
                marker = dict(size = y,opacity = op),   # Changing opacity & size of bubbles
                )

fig = go.Figure(data=data)

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
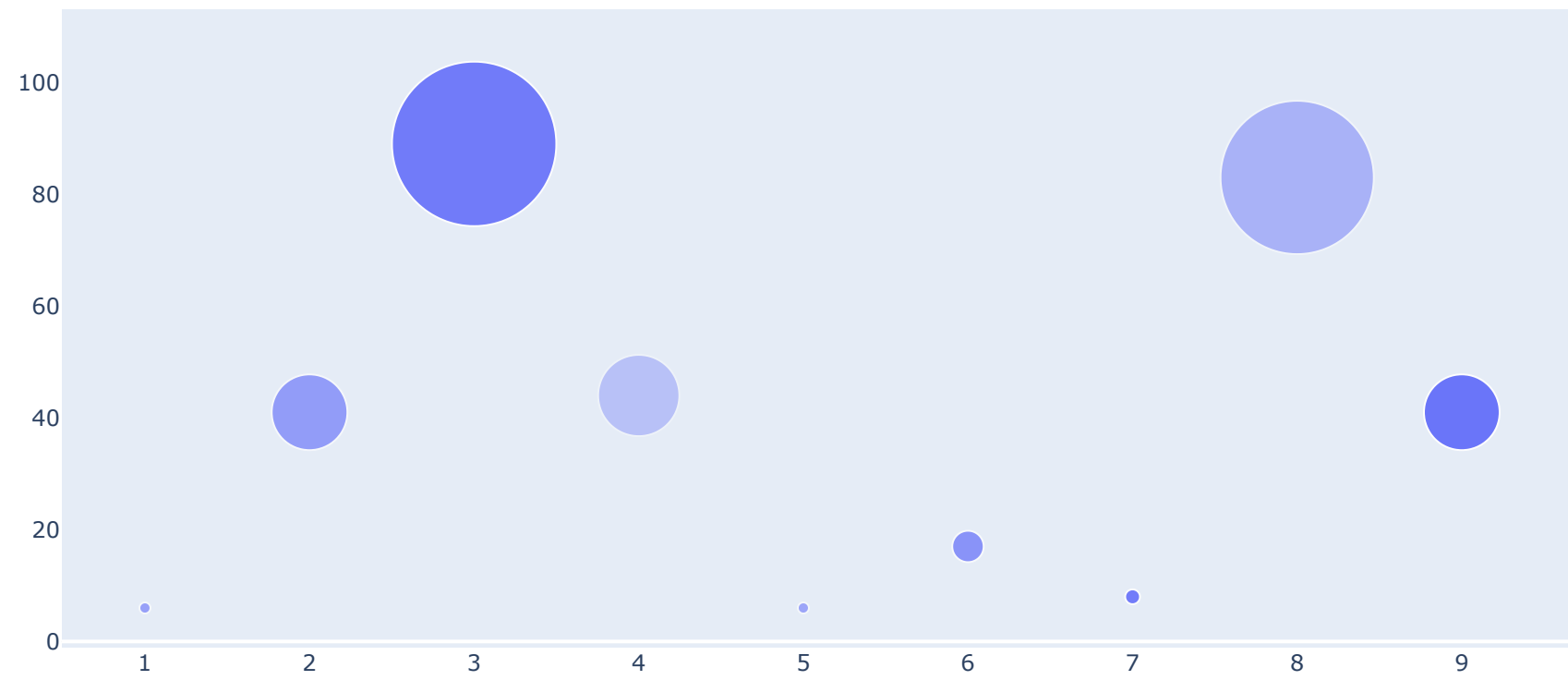
```
In [104]:  x = np.arange(1,10)
           y = np.random.randint(1,100,9)
           op = np.random.uniform(0.2,1 ,9)
           hexval = [hex(x) for x in np.random.randint(0,16777215,10)]
           hexval = ['#' + hexval[i][2:] for i in range(0,10)] #Generate Hex color list
           data = go.Scatter(
                           x = x,
                           y = y,
                           mode = 'markers',
                           marker = dict(size = y,color = hexval) # Changing color & size of bubbles
                           )

           fig = go.Figure(data=data)

           # Hide grid lines
           fig.update_xaxes(showgrid=False)
           fig.update_yaxes(showgrid=False)

           fig.show()
```
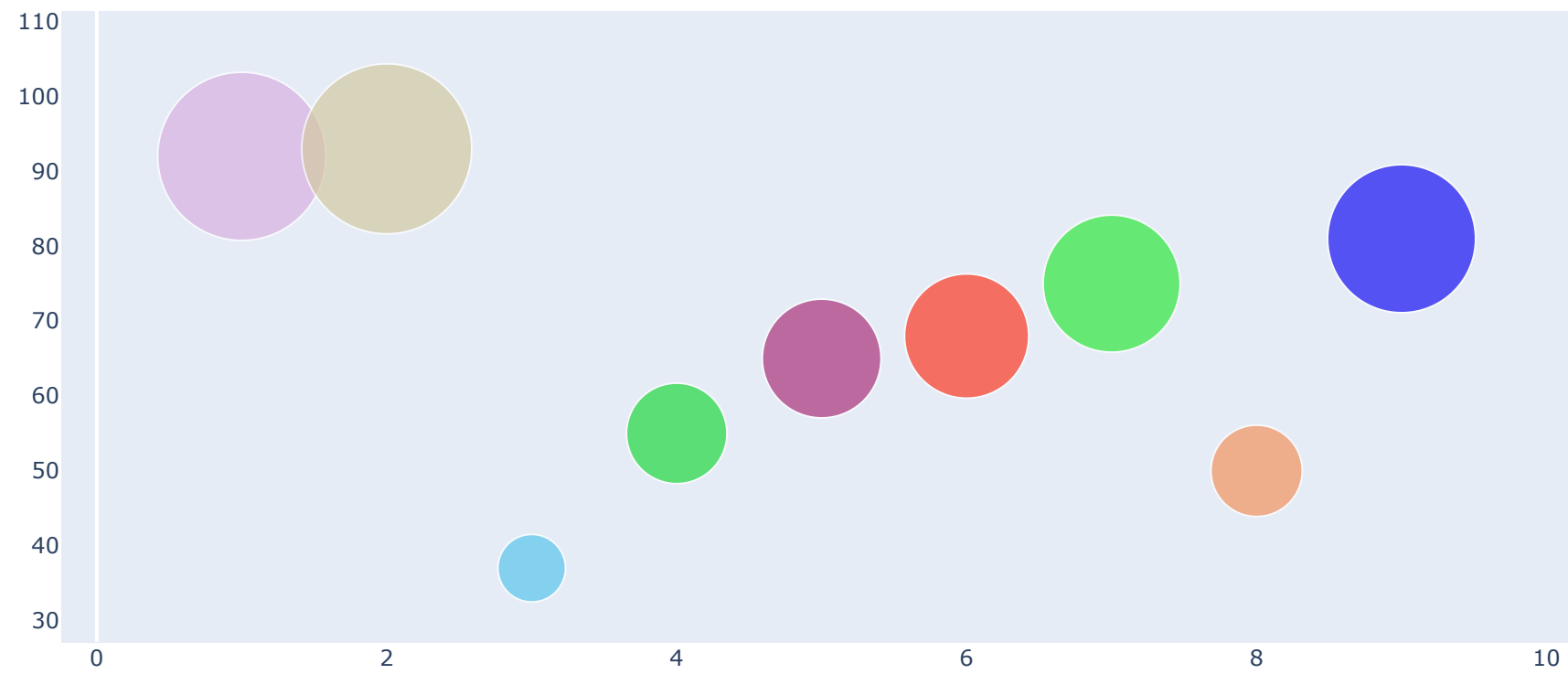
```
In [105]: suicide.head()
```

Out[105]:

	country	year	sex	age	suicides_no	population	suicides/100k pop	country-year	HDI for year	gdp_for_year ($)	gdp_per_capita ($)	generation
0	Albania	1987	male	15-24 years	21	312900	6.71	Albania1987	NaN	2,156,624,900	796	Generation X
1	Albania	1987	male	35-54 years	16	308000	5.19	Albania1987	NaN	2,156,624,900	796	Silent
2	Albania	1987	female	15-24 years	14	289700	4.83	Albania1987	NaN	2,156,624,900	796	Generation X
3	Albania	1987	male	75+ years	1	21800	4.59	Albania1987	NaN	2,156,624,900	796	G.I. Generation
4	Albania	1987	male	25-34 years	9	274300	3.28	Albania1987	NaN	2,156,624,900	796	Boomers

In [106]:
```python
# Bubble Chart using plotly.express
fig = px.scatter(
                suicide.query("year==1987"),
                x="population",
                y="suicides_no",
                size="suicides_no",
                color="age",
                hover_name="country",
                size_max=60
                )


# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
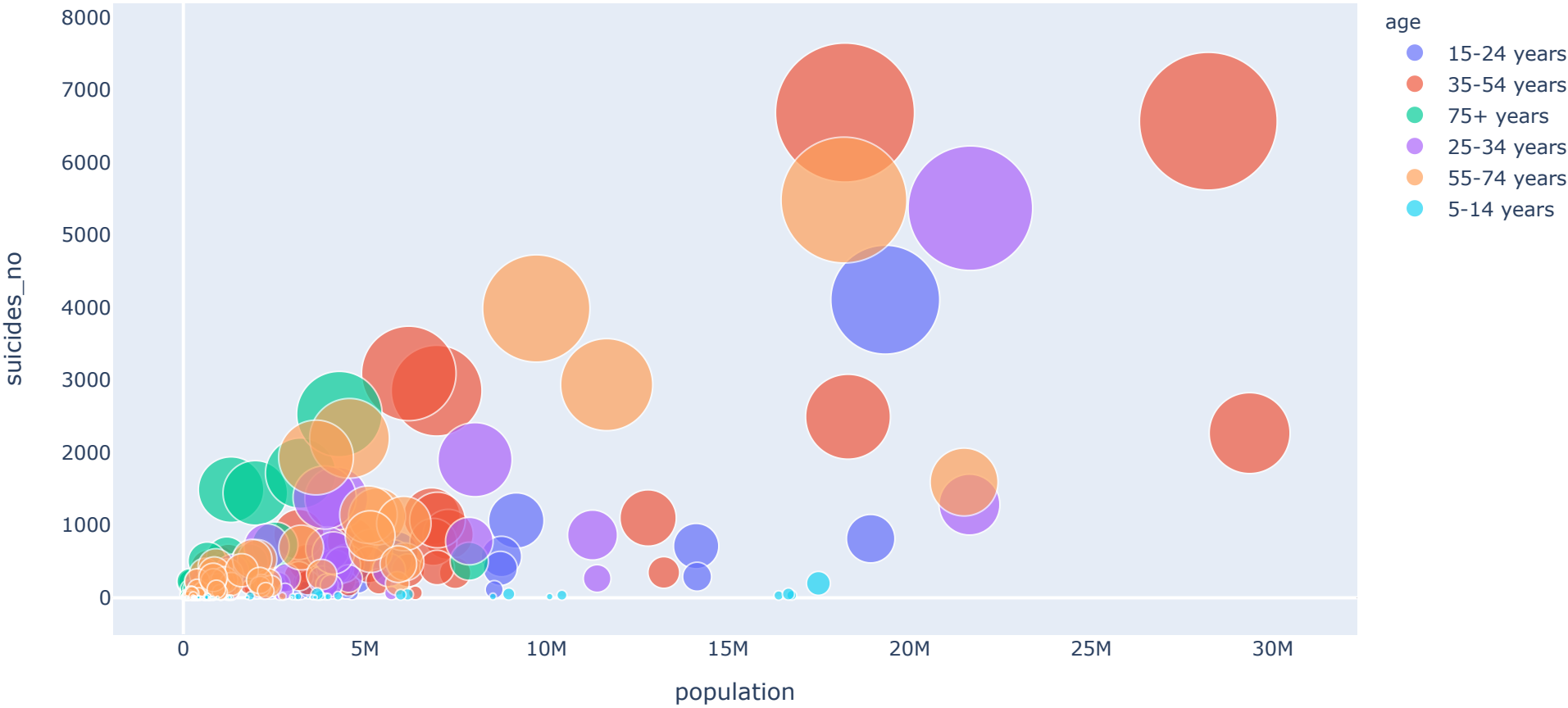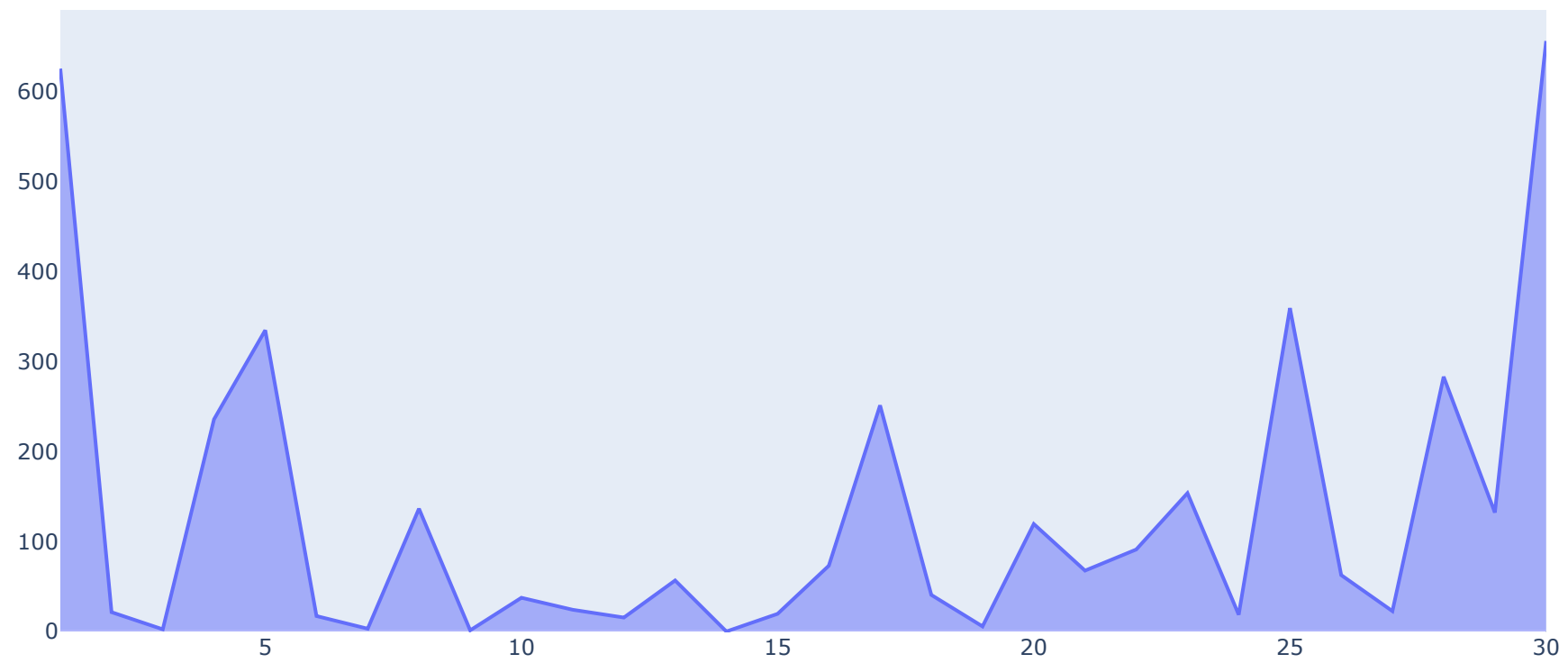

Area Plot

```python
# Simple Area plot
x = np.arange(1,31)
y = np.random.normal(10,11,size=30)
y = np.square(y)
fig = go.Figure()
fig.add_trace(go.Scatter(x=x, y=y, fill='tozeroy')) # fill down to xaxis

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```

```
#Changing color of area plot using marker color
x = np.arange(1,31)
y = np.random.normal(10,11,size=30)
y = np.square(y)
fig = go.Figure()
fig.add_trace(go.Scatter(x=x, y=y, fill='tozeroy',marker = dict(color = 'gold'))) # fill down to xaxis

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
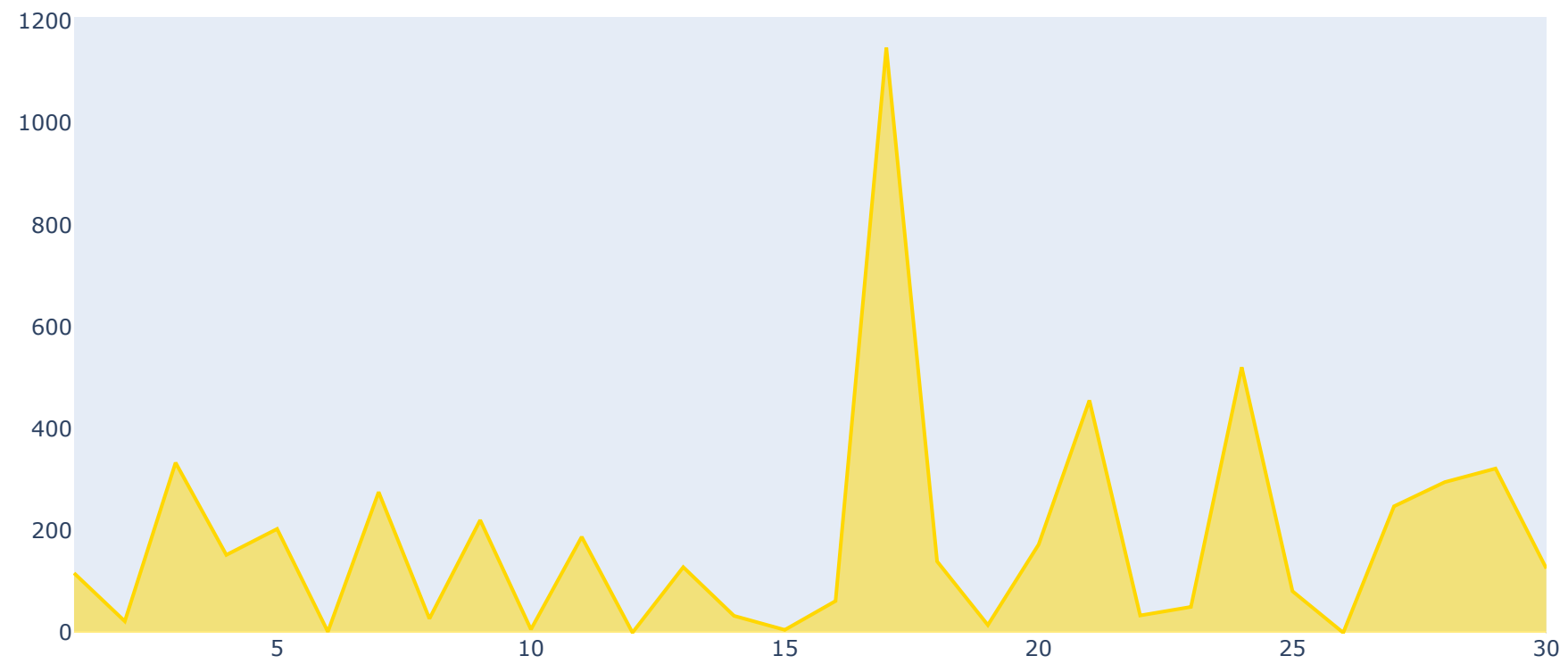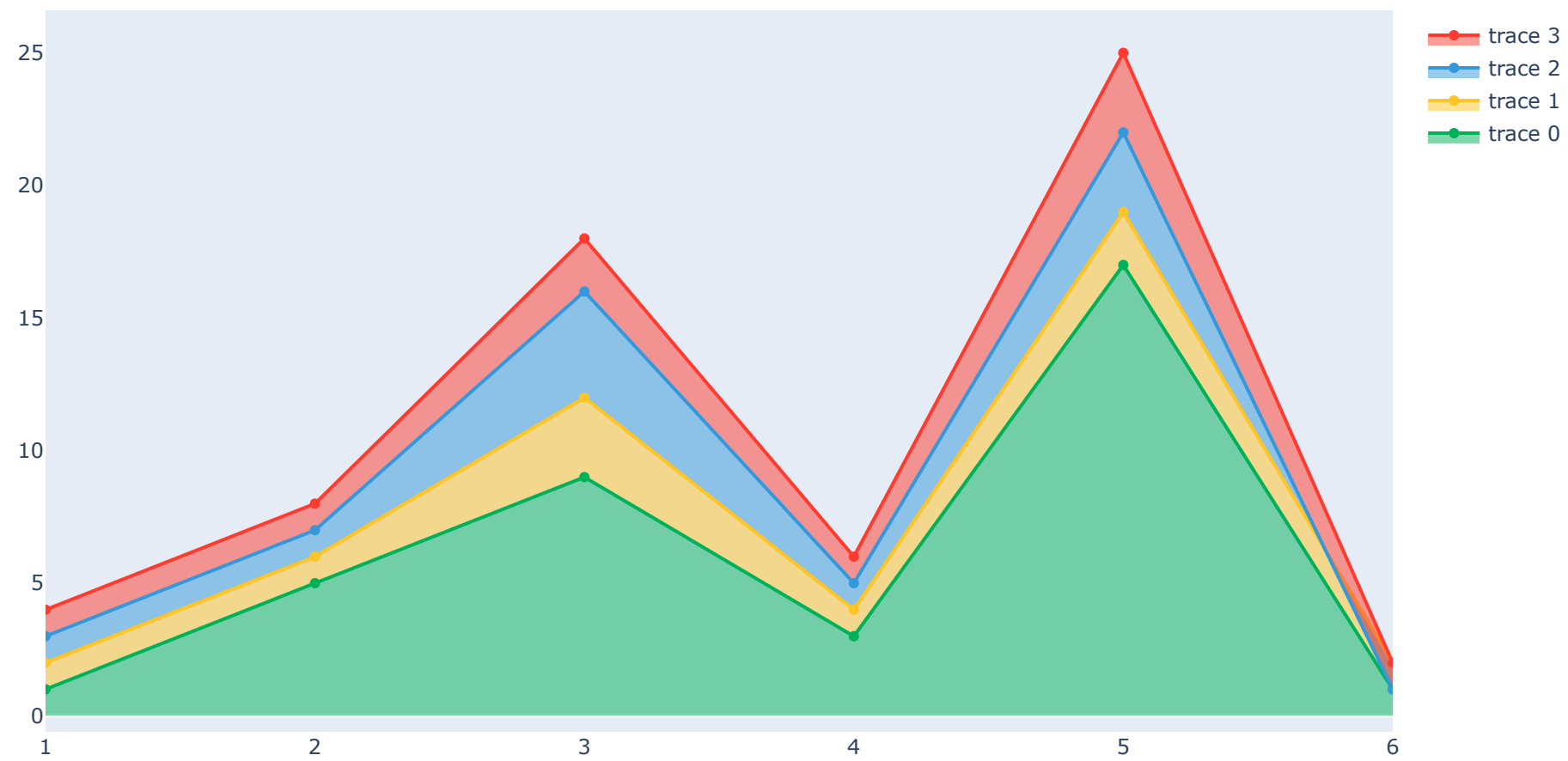
```
In [111]: x=np.arange(1,7)
          y1 =  np.array([1,5,9,3,17,1])
          y2 = np.array([2,6,12,4,19,2])
          y3 = np.array([3,7,16,5,22,1])
          y4 = np.array([4,8,18,6,25,2])


          fig = go.Figure()
          fig.add_trace(go.Scatter(x=x, y=y1, fill='tozeroy',marker = dict(color = '#00b159'))) # fill down to xaxis
          fig.add_trace(go.Scatter(x=x, y=y2, fill='tonexty',marker = dict(color = '#ffc425'))) # fill to trace0 y
          fig.add_trace(go.Scatter(x=x, y=y3, fill='tonexty',marker = dict(color = '#3498DB'))) # fill to trace1 y
          fig.add_trace(go.Scatter(x=x, y=y4, fill='tonexty',marker = dict(color = '#ff3b30'))) # fill to trace2 y
          fig.update_layout(width = 980 , height = 600)

          # Hide grid lines
          fig.update_xaxes(showgrid=False)
          fig.update_yaxes(showgrid=False)

          fig.show()
```

```python
#Stacked Area Chart (Using stackgroup parameter)

x=np.arange(1,7)
y1 =  np.array([1,5,9,3,17,1])
y2 = np.array([2,6,12,4,19,2])
y3 = np.array([3,7,16,5,22,1])
y4 = np.array([4,8,18,6,25,2])


fig = go.Figure()
fig.add_trace(go.Scatter(
                         x=x,
                         y=y1,
                         marker = dict(color = '#00b159'),
                         stackgroup='one' # The stackgroup parameter is used to create a Stacked Area Chart
                        )
             )

fig.add_trace(go.Scatter(
                         x=x,
                         y=y2,
                         marker = dict(color = '#ffc425'),
                         stackgroup='one'
                        )
             )


fig.add_trace(go.Scatter(
                         x=x,
                         y=y3,
                         marker = dict(color = '#3498DB'),
                         stackgroup='one'
                        )
             )


fig.add_trace(go.Scatter(
                         x=x,
                         y=y4,
                         marker = dict(color = '#ff3b30'),
                         stackgroup='one'
                        )
             )

fig.update_layout(width = 980 , height = 600)

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
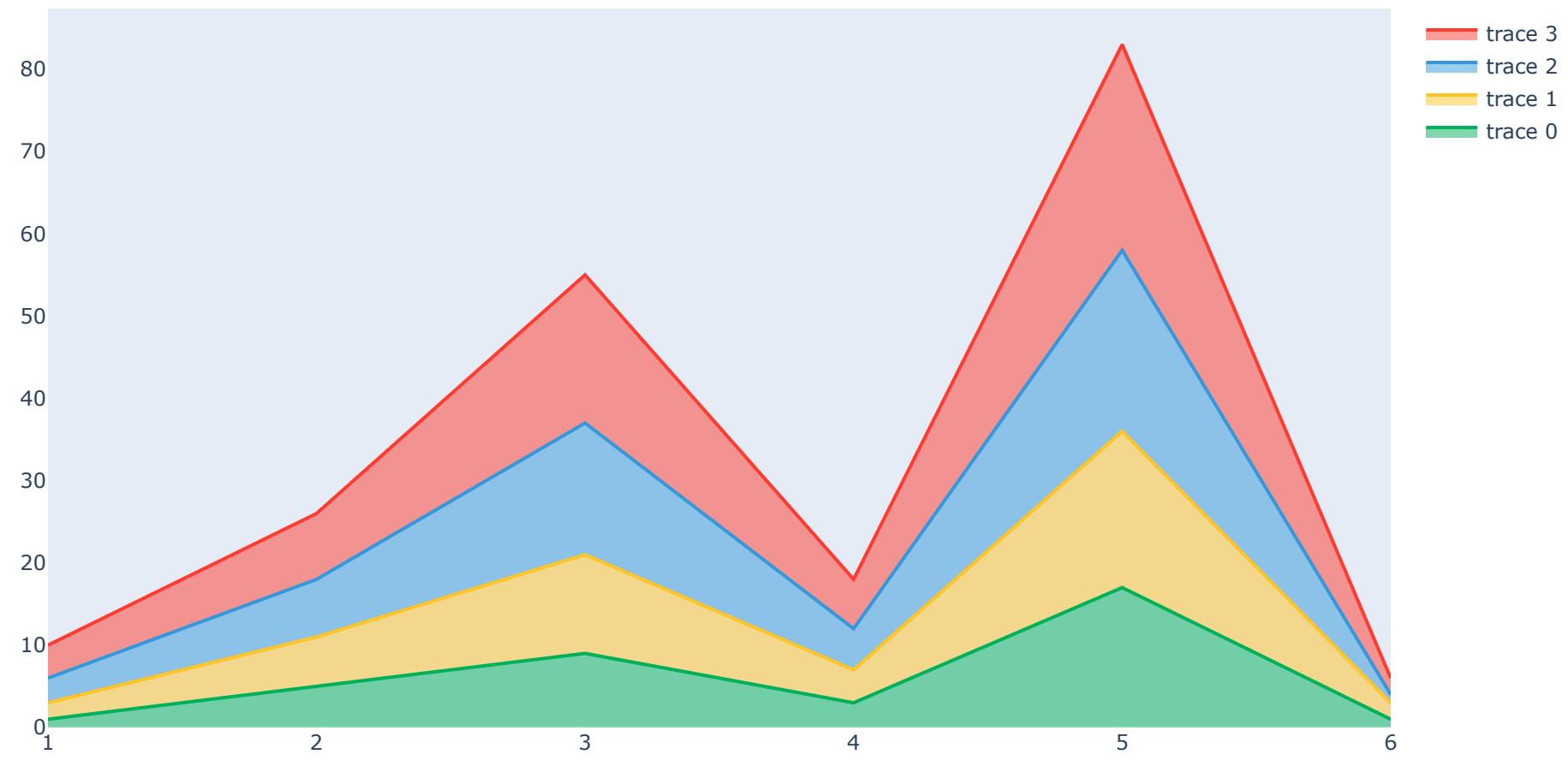
In [113]:
```python
#Stacked Area Chart

x=['A','B','C','D','E']
y1 =  np.array([30,10,40,20,60])
y2 = np.array([10,20,10,20,10])
y3 = np.array([5,10,5,10,5])
y4 = np.array([10,5,10,5,10])


fig = go.Figure()
fig.add_trace(go.Scatter(
                        x=x,
                        y=y1,
                        marker = dict(color = '#00b159'), # Color of trace0
                        stackgroup='one' # The stackgroup parameter is used to create a Stacked Area Chart
                        )
             )

fig.add_trace(go.Scatter(
                        x=x,
                        y=y2,
                        marker = dict(color = '#ffc425'),
                        stackgroup='one'
                        )
             )


fig.add_trace(go.Scatter(
                        x=x,
                        y=y3,
                        marker = dict(color = '#3498DB'),
                        stackgroup='one'
                        )
             )


fig.add_trace(go.Scatter(
                        x=x,
                        y=y4,
                        marker = dict(color = '#ff3b30'),
                        stackgroup='one'
                        )
             )

fig.update_layout(width = 980 , height = 600)

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
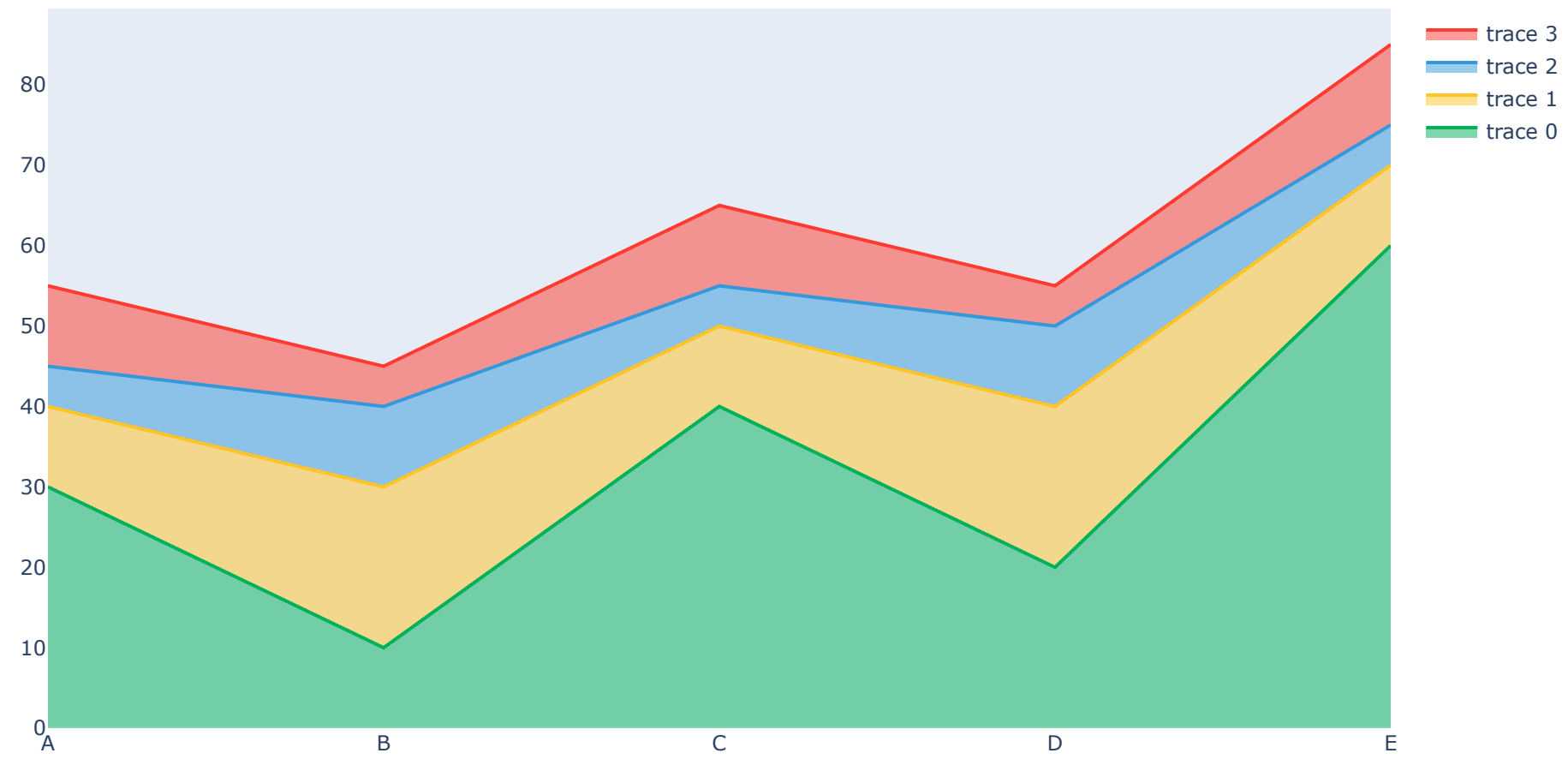
```python
#Stacked Area Chart Without Boundary Lines (Using mode = 'none')

x=['A','B','C','D','E']
y1 =  np.array([30,10,40,20,60])
y2 = np.array([10,20,10,20,10])
y3 = np.array([5,10,5,10,5])
y4 = np.array([10,5,10,5,10])


fig = go.Figure()
fig.add_trace(go.Scatter(
                        x=x,
                        y=y1,
                        mode='none',
                        stackgroup='one'
                        )
            )

fig.add_trace(go.Scatter(
                        x=x,
                        y=y2,
                        mode='none',
                        stackgroup='one'
                        )
            )


fig.add_trace(go.Scatter(
                        x=x,
                        y=y3,
                        mode='none',
                        stackgroup='one'
                        )
            )


fig.add_trace(go.Scatter(
                        x=x,
                        y=y4,
                        mode='none',
                        stackgroup='one'
                        )
            )

fig.update_layout(width = 980 , height = 600)

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
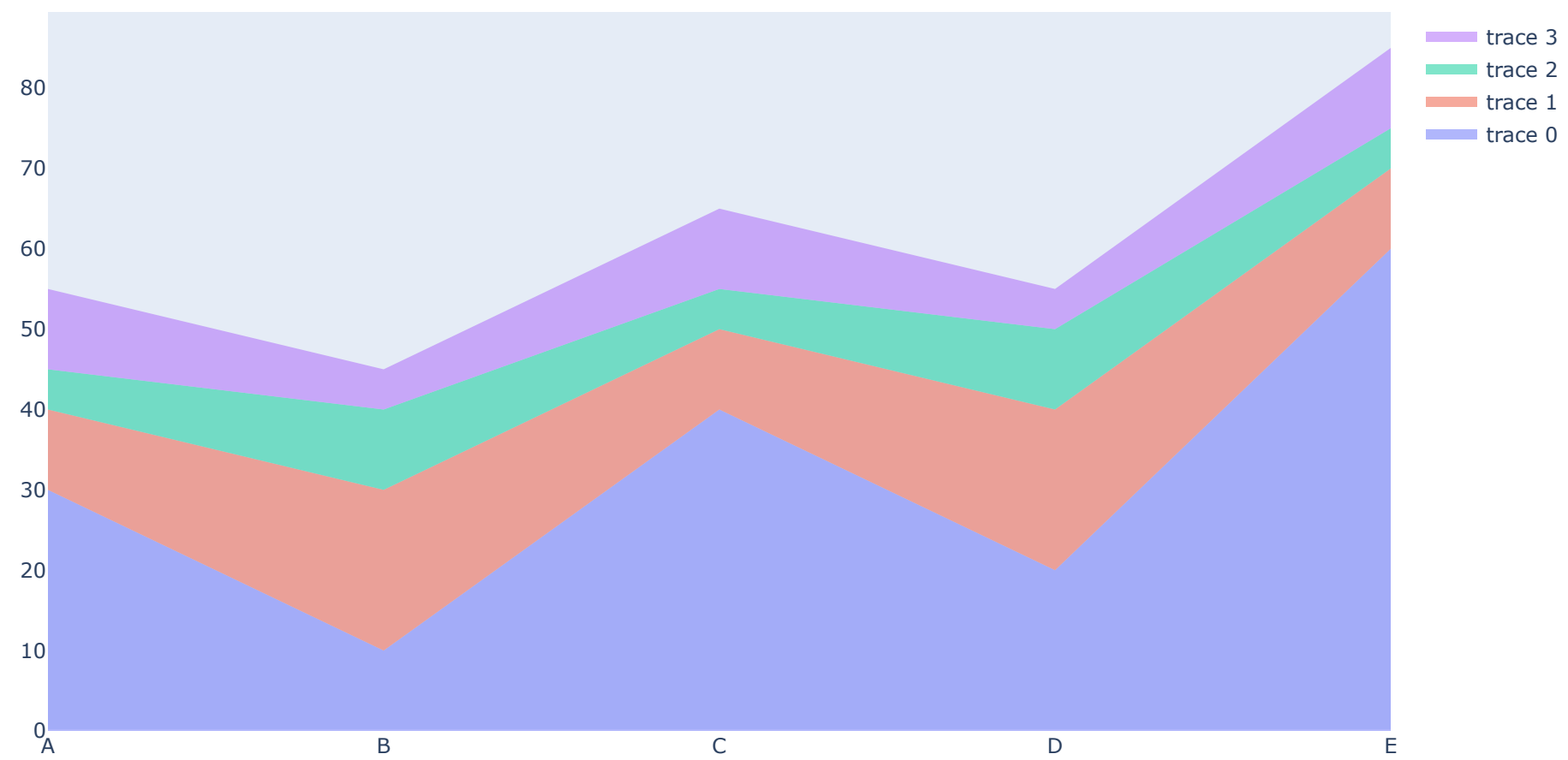
```
In [115]:  #100 Percent Stacked Area Chart (Using groupnorm parameter)


           x=['A','B','C','D','E']
           y1 =  np.array([30,10,40,20,60])
           y2 = np.array([10,20,10,20,10])
           y3 = np.array([5,10,5,10,5])
           y4 = np.array([10,5,10,5,10])


           fig = go.Figure()
           fig.add_trace(go.Scatter(
                               x=x,
                               y=y1,
                               marker = dict(color = '#00b159'),
                               stackgroup='one',
                               groupnorm='percent' #normalization for the sum of the stackgroup
                           )
                   )

           fig.add_trace(go.Scatter(
                               x=x,
                               y=y2,
                               marker = dict(color = '#ffc425'),
                               stackgroup='one',
                               groupnorm='percent'
                           )
                   )


           fig.add_trace(go.Scatter(
                               x=x,
                               y=y3,
                               marker = dict(color = '#3498DB'),
                               stackgroup='one',
                               groupnorm='percent'
                           )
                   )


           fig.add_trace(go.Scatter(
                               x=x,
                               y=y4,
                               marker = dict(color = '#ff3b30'),
                               stackgroup='one',
                               groupnorm='percent'
                           )
                   )

           fig.update_layout(
                           width = 990,
                           height = 650,
                           xaxis_type='category',
                           yaxis=dict(
                                   range=[1, 100],
                                   ticksuffix='%'
                               )
                       )


           # Hide grid lines
```
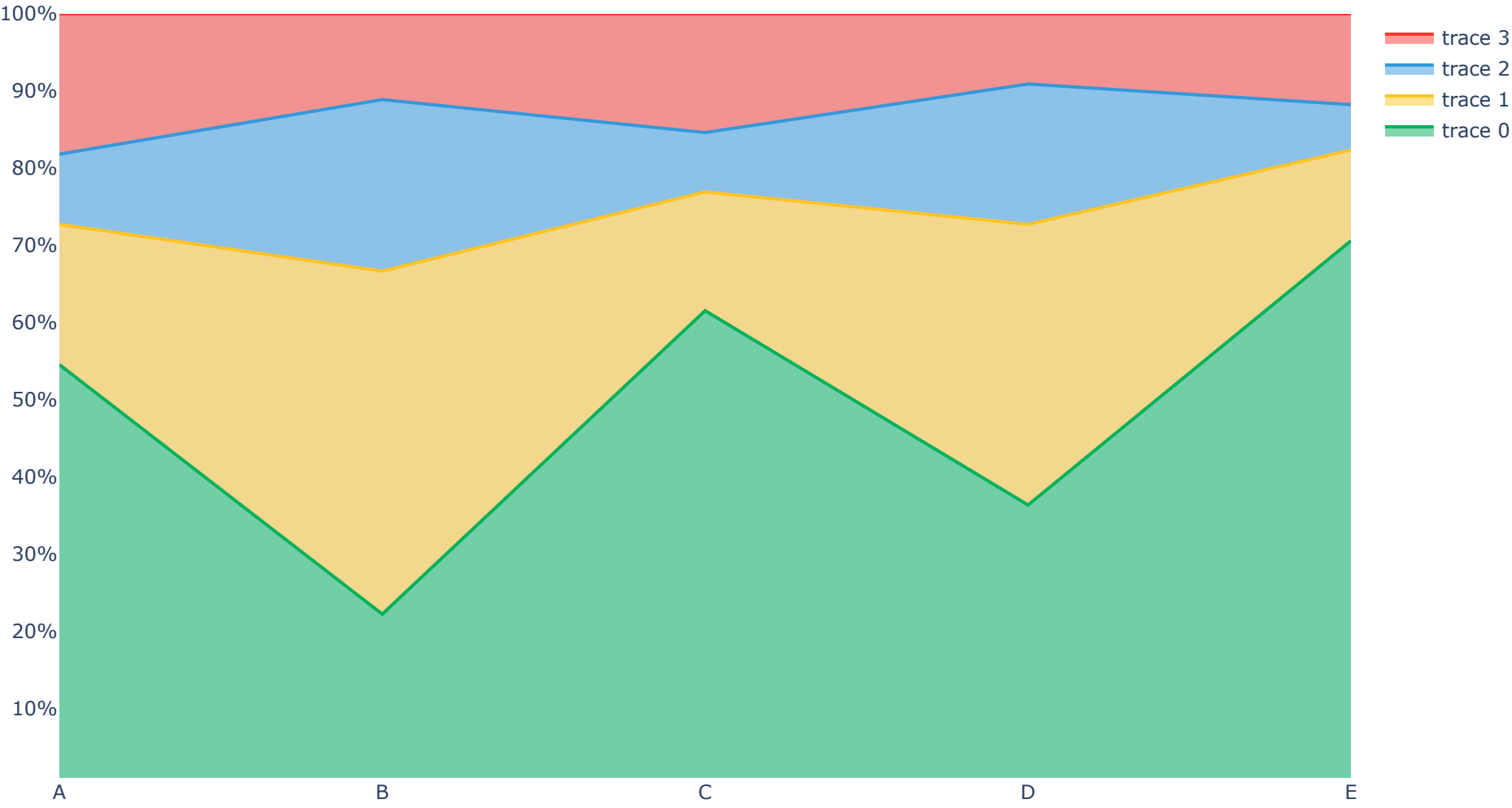
```
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```

```python
# Area Chart with interioir filling using  fill='tonexty'

x=['A','B','C','D','E']
y1 =  np.array([30,10,40,20,60])
y2 = np.array([10,20,10,20,10])


fig = go.Figure()
fig.add_trace(go.Scatter(
                        x=x,
                        y=y1,
                        marker = dict(color = '#00b159'),
                        fill = None
                        )
            )

fig.add_trace(go.Scatter(
                        x=x,
                        y=y2,
                        fill='tonexty', # fill to trace0 y
                        marker = dict(color = '#ffc425'),
                        )
            )


fig.update_layout(width = 980 , height = 600)

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
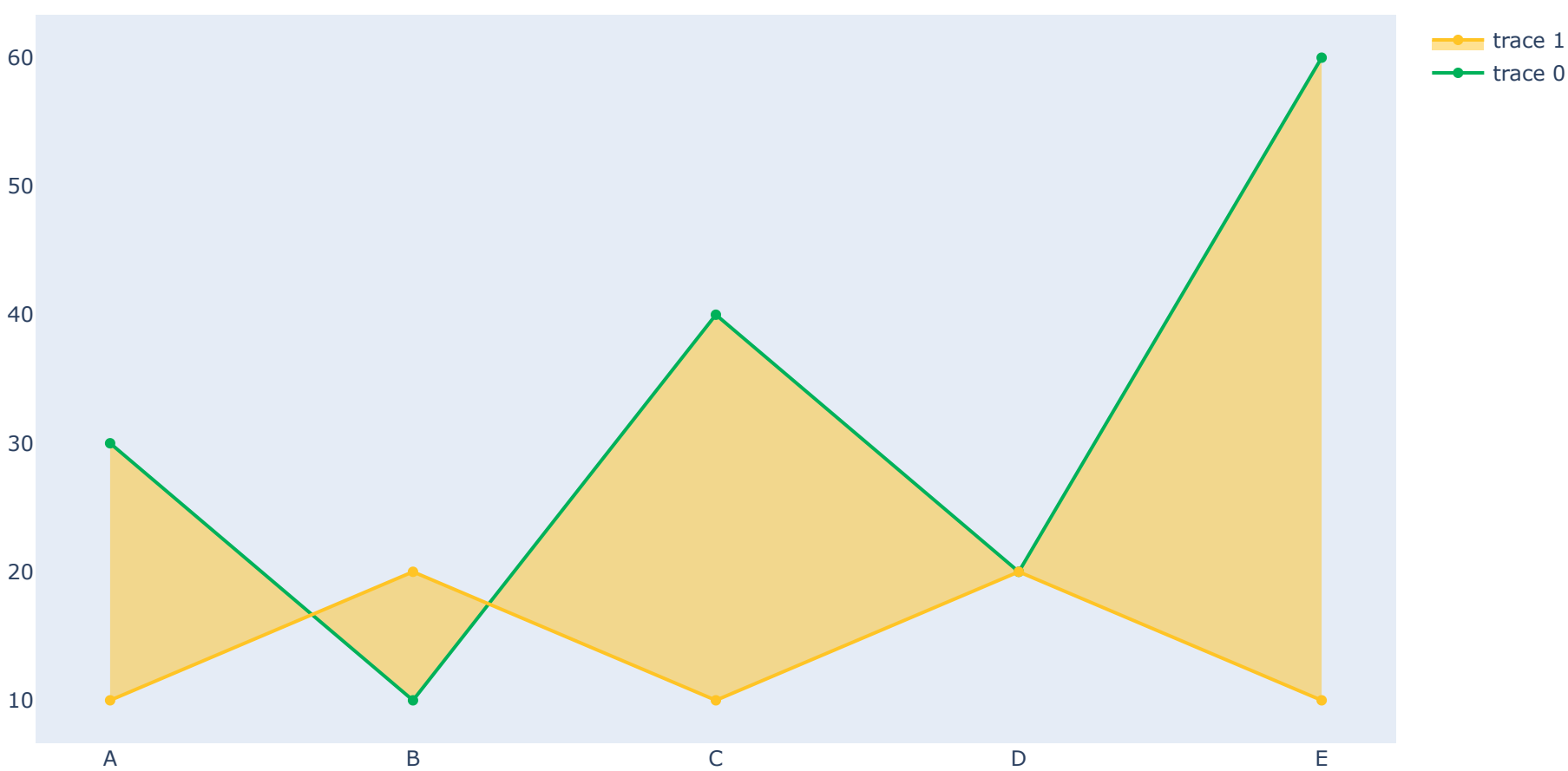
Tables & Figure Factory Tables

`rating`

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree	Total	Strongly Agree (%)	Agree (%)	Neutral (%)	Disagree (%)	Strongly Disagree (%)
Python	428	111	70	101	80	790	54.18	14.05	8.86	12.78	10.13
Java	370	222	80	104	70	846	43.74	26.24	9.46	12.29	8.27
Julia	298	121	90	102	60	671	44.41	18.03	13.41	15.20	8.94
C++	310	141	100	109	56	716	43.30	19.69	13.97	15.22	7.82
C	400	121	110	107	78	816	49.02	14.83	13.48	13.11	9.56

In [118]: `#Basic table in Plotly`

```python
fig = go.Figure(data=[go.Table(
                    header=dict(
                            values=list(rating.columns),
                        ),
                    cells=dict(values=[
                                rating['Strongly Agree'] ,
                                rating['Agree'] ,
                                rating['Neutral'] ,
                                rating['Disagree'] ,
                                rating['Strongly Disagree'],
                                rating['Total'],
                                rating['Strongly Agree (%)'],
                                rating['Agree (%)'],
                                rating['Neutral (%)'],
                                rating['Disagree (%)'],
                                rating['Strongly Disagree (%)']
                                ],
                        )
                    )
                ]
            )

fig.show()
```

Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree	Total	Strongly Agree (%)	Agree (%)	Neutral (%)	Disagree (%)	Strongly Disagree (%)
428	111	70	101	80	790	54.18	14.05	8.86	12.78	10.13
370	222	80	104	70	846	43.74	26.24	9.46	12.29	8.27
298	121	90	102	60	671	44.41	18.03	13.41	15.2	8.94
310	141	100	109	56	716	43.3	19.69	13.97	15.22	7.82
400	121	110	107	78	816	49.02	14.83	13.48	13.11	9.56

```
In [119]:    # Styled Table in Plotly
             fig = go.Figure(data=[go.Table(
                                    header=dict(
                                            values=list(rating.columns),
                                            fill_color='paleturquoise',
                                            align='left'
                                            ),
                                    cells=dict(values=[
                                            rating['Strongly Agree'] ,
                                            rating['Agree'] ,
                                            rating['Neutral'] ,
                                            rating['Disagree'] ,
                                            rating['Strongly Disagree'],
                                            rating['Total'],
                                            rating['Strongly Agree (%)'],
                                            rating['Agree (%)'],
                                            rating['Neutral (%)'],
                                            rating['Disagree (%)'],
                                            rating['Strongly Disagree (%)']
                                            ],
                                    fill_color='lavender',
                                    align='center'))
                            ]
                    )
             fig.update_layout(width=990, height=350)
             fig.show()
```

Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree	Total	Strongly Agree (%)	Agree (%)	Neutral (%)	Disagree (%)	Strongly Disagree (%)
428	111	70	101	80	790	54.18	14.05	8.86	12.78	10.13
370	222	80	104	70	846	43.74	26.24	9.46	12.29	8.27
298	121	90	102	60	671	44.41	18.03	13.41	15.2	8.94
310	141	100	109	56	716	43.3	19.69	13.97	15.22	7.82
400	121	110	107	78	816	49.02	14.83	13.48	13.11	9.56

```python
In [120]:  # Styled Table in Plotly

fig = go.Figure(data=[go.Table(
                        header=dict(
                                values=list(insurance.columns), # Header values
                                line_color='black', # Line Color of header
                                fill_color='orange', # background color of header
                                align='center', # Align header at center
                                height=40, # Height of Header
                                font=dict(color='white', size=18), # Font size & color of header text
                                ),
                        cells=dict(values=[
                                    insurance.age , # Column values
                                    insurance.sex,
                                    insurance.bmi,
                                    insurance.children,
                                    insurance.smoker,
                                    insurance.region,
                                    insurance.charges
                                    ],
                                line_color='darkgrey', # Line color of the cell
                                fill_color='lightcyan', # Color of the cell
                                align='left'  # Align text to left in cell
                                )
                        )
                ]
            )

fig.show()
```

age	sex	bmi	children	smoker	region	charges
19	female	27.9	0	yes	southwest	16884.924
18	male	33.77	1	no	southeast	1725.5523
28	male	33	3	no	southeast	4449.462
33	male	22.705	0	no	northwest	21984.47061
32	male	28.88	0	no	northwest	3866.8552
31	female	25.74	0	no	southeast	3756.6216
46	female	33.44	1	no	southeast	8240.5896
37	female	27.74	3	no	northwest	7281.5056
37	male	29.83	2	no	northeast	6406.4107
60	female	25.84	0	no	northwest	28923.136919999
25	male	26.22	0	no	northeast	2721.3208
62	female	26.29	0	yes	southeast	27808.7251
23	male	34.4	0	no	southwest	1826.8429999999
56	female	39.82	0	no	southeast	11090.7178
27	male	42.13	0	yes	southeast	39611.7577

```
In [143]:  # Styled Table in Plotly

           rowEvenColor = 'lightgrey'
           rowOddColor = 'white'
           fig = go.Figure(data=[go.Table(  columnwidth = [80,80,80,80,80,80,120,80,80,80,120],
                                   header=dict(
                                           values=[
                                                   '<b>Strongly Agree</b>',
                                                   '<b>Agree</b>',
                                                   '<b>Neutral</b>',
                                                   '<b>Disagree</b>',
                                                   '<b>Strongly Disagree</b>',
                                                   '<b>Total</b>',
                                                   '<b>Strongly Agree (%)</b>',
                                                   '<b>Agree (%)</b>',
                                                   '<b>Neutral (%)</b>',
                                                   '<b>Disagree (%)</b>',
                                                   '<b>Strongly Disagree (%)</b>'
                                                   ],
                                           fill_color='#8BC34A',
                                           line = dict(color = '#689F38' , width = 4),
                                           align='center',
                                           font_size=12,
                                           font_color = 'white'
                                       ),
                                   cells=dict(values=[
                                                   rating['Strongly Agree'] ,
                                                   rating['Agree'] ,
                                                   rating['Neutral'] ,
                                                   rating['Disagree'] ,
                                                   rating['Strongly Disagree'],
                                                   rating['Total'],
                                                   rating['Strongly Agree (%)'],
                                                   rating['Agree (%)'],
                                                   rating['Neutral (%)'],
                                                   rating['Disagree (%)'],
                                                   rating['Strongly Disagree (%)']
                                                   ],
                                           fill_color = [[rowOddColor,rowEvenColor]*5],
                                           line = dict(color = 'lightgreen' , width = 4),
                                           align ='center',
                                           font_size=12,
                                           font = dict(color = 'darkslategray', size = 11),
                                           height=40
                                       )
                                   )
                               ]
                           )
           fig.update_layout(width=990, height=500)
           fig.show()
```

Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree	Total	Strongly Agree (%)	Agree (%)	Neutral (%)	Disagree (%)	Strongly Disagree (%)
428	111	70	101	80	790	54.18	14.05	8.86	12.78	10.13
370	222	80	104	70	846	43.74	26.24	9.46	12.29	8.27
298	121	90	102	60	671	44.41	18.03	13.41	15.2	8.94
310	141	100	109	56	716	43.3	19.69	13.97	15.22	7.82
400	121	110	107	78	816	49.02	14.83	13.48	13.11	9.56

In [31]:
```python
# Create simple table using create_table function

fig = ff.create_table(insurance.tail(5))
fig.show()
```

age	sex	bmi	children	smoker	region	charges
50	male	30.97	3	no	northwest	10600.5483
18	female	31.92	0	no	northeast	2205.9808
18	female	36.85	0	no	southeast	1629.8335
21	female	25.8	0	no	southwest	2007.945
61	female	29.07	0	yes	northwest	29141.3603

```
In [266]:  # Create simple table using create_table function

           fig = ff.create_table(insurance.tail(5),height_constant=50)
           fig.show()
```

age	sex	bmi	children	smoker	region	charges
50	male	30.97	3	no	northwest	10600.5483
18	female	31.92	0	no	northeast	2205.9808
18	female	36.85	0	no	southeast	1629.8335
21	female	25.8	0	no	southwest	2007.945
61	female	29.07	0	yes	northwest	29141.3603

```
In [47]:  # Using color scales in table

          colorscale = [[0, 'red'],[.5, '#DCE775'],[1, '#C0CA33']]
          fig = ff.create_table(insurance.tail(5),height_constant=50,colorscale=colorscale)
          fig.show()
```

age	sex	bmi	children	smoker	region	charges
50	male	30.97	3	no	northwest	10600.5483
18	female	31.92	0	no	northeast	2205.9808
18	female	36.85	0	no	southeast	1629.8335
21	female	25.8	0	no	southwest	2007.945
61	female	29.07	0	yes	northwest	29141.3603

```
In [50]:  # Changing font color

          colorscale = [[0, 'red'],[.5, '#DCE775'],[1, '#C0CA33']]
          font=['white', '#212121' , 'red']
          fig = ff.create_table(insurance.tail(5),height_constant=50,colorscale=colorscale,font_colors=font)
          fig.show()
```

age	sex	bmi	children	smoker	region	charges
50	male	30.97	3	no	northwest	10600.5483
18	female	31.92	0	no	northeast	2205.9808
18	female	36.85	0	no	southeast	1629.8335
21	female	25.8	0	no	southwest	2007.945
61	female	29.07	0	yes	northwest	29141.3603

```
In [269]:  # Changing font size using "fig.layout.annotations[i].font.size"

           colorscale = [[0, 'red'],[.5, '#DCE775'],[1, '#C0CA33']]
           font=['white', '#212121' , 'red']
           fig = ff.create_table(insurance.tail(5),height_constant=50,colorscale=colorscale,font_colors=font)
           for i in range(len(fig.layout.annotations)):
               fig.layout.annotations[i].font.size = 17
           fig.show()
```

age	sex	bmi	children	smoker	region	charges
50	male	30.97	3	no	northwest	10600.5483
18	female	31.92	0	no	northeast	2205.9808
18	female	36.85	0	no	southeast	1629.8335
21	female	25.8	0	no	southwest	2007.945
61	female	29.07	0	yes	northwest	29141.3603

```
In [44]: colorscale = [[0, 'red'],[.5, '#DCE775'],[1, '#C0CA33']]
         font=['white', '#212121' , 'red']
         fig = ff.create_table(insurance.tail(5),height_constant=50,colorscale=colorscale,font_colors=font)
         for i in range(len(fig.layout.annotations)):
             fig.layout.annotations[i].font.size = 12
         fig.show()
```

age	sex	bmi	children	smoker	region	charges
50	male	30.97	3	no	northwest	10600.5483
18	female	31.92	0	no	northeast	2205.9808
18	female	36.85	0	no	southeast	1629.8335
21	female	25.8	0	no	southwest	2007.945
61	female	29.07	0	yes	northwest	29141.3603

```
In [52]: canada.loc[:, ['India','Pakistan','China']].head(6)
```

Out[52]:

	India	Pakistan	China
1980	8880	978	5123
1981	8670	972	6682
1982	8147	1201	3308
1983	7338	900	1863
1984	5704	668	1527
1985	4211	514	1816

```
In [121]:  # Displaying tables along with graphs

           import plotly.graph_objs as go
           import plotly.figure_factory as ff

           # Add table data
           table_data = canada.loc[:, ['India','Pakistan','China']].head(6)


           # Initialize a figure with ff.create_table(table_data)
           fig = ff.create_table(table_data, height_constant=60)


           # Make traces for graph
           fig.add_trace( go.Scatter(
                           x = canada.index.values,
                           y = canada['China'],
                           mode = 'lines',
                           name = 'China',
                           xaxis='x2', yaxis='y2'
                           )
                       )


           # Make traces for graph
           fig.add_trace( go.Scatter(
                           x = canada.index.values,
                           y = canada['India'],
                           mode = 'lines',
                           name = 'India',
                           xaxis='x2', yaxis='y2'
                           )
                       )

           # Make traces for graph
           fig.add_trace( go.Scatter(
                           x = canada.index.values,
                           y = canada['Pakistan'],
                           mode = 'lines',
                           name = 'Pakistan',
                           xaxis='x2', yaxis='y2'
                           )
                       )



           fig.update_layout(
                           title=dict(text = "Immigration Data",x=0.5,y=0.98), # Figure title along with Alignment values
                           paper_bgcolor= '#dbdbdb', # Figure background
                           margin = {'t':50, 'b':100},
                           xaxis = {'domain': [0, .5] , 'title' : 'Migrants'},
                           xaxis2 = {'domain': [0.6, 1.] , 'title' : 'Year'},
                           yaxis2 = {'anchor': 'x2', 'title': 'Count'},
                           width = 990,
                           height = 600
                       )

           # Hide grid lines
           fig.update_xaxes(showgrid=False)
           fig.update_yaxes(showgrid=False)
```

Immigration Data

India	Pakistan	China
8880	978	5123
8670	972	6682
8147	1201	3308
7338	900	1863
5704	668	1527
4211	514	1816

Migrants
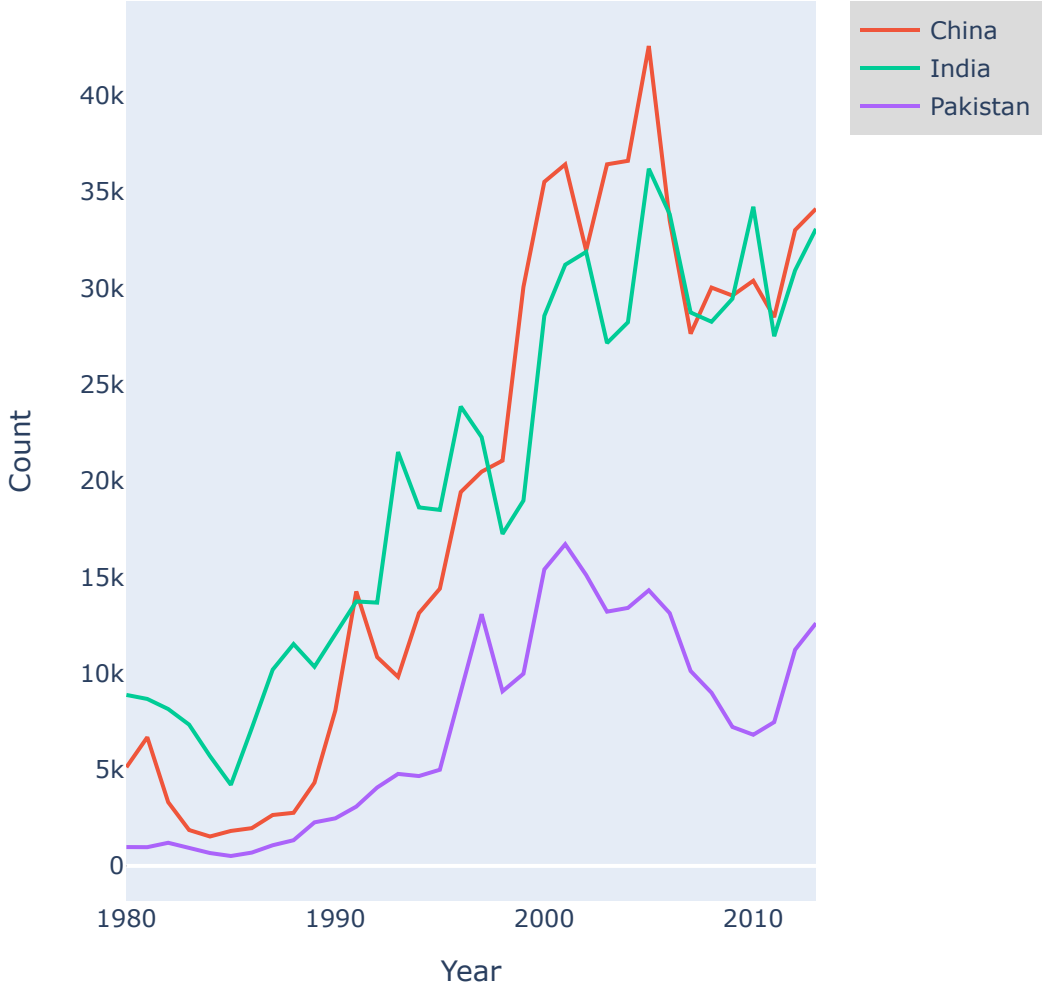
```
In [122]:   # Displaying tables along with graphs

            import plotly.graph_objs as go
            import plotly.figure_factory as ff

            # Add table data
            table_data = canada.loc[:, ['India','Pakistan','China' , 'Australia' , 'Germany' , 'Austria']].head(6)


            # Initialize a figure with ff.create_table(table_data)
            fig = ff.create_table(table_data, height_constant=60)


            # Make traces for graph
            fig.add_trace( go.Scatter(
                            x = canada.index.values,
                            y = canada['China'],
                            mode = 'lines',
                            name = 'China',
                            xaxis='x2', yaxis='y2'
                            )
                        )

            fig.add_trace( go.Scatter(
                            x = canada.index.values,
                            y = canada['India'],
                            mode = 'lines',
                            name = 'India',
                            xaxis='x2', yaxis='y2'
                            )
                        )

            fig.add_trace( go.Scatter(
                              x = canada.index.values,
                              y = canada['Pakistan'],
                              mode = 'lines',
                              name = 'Pakistan',
                              xaxis='x2', yaxis='y2'
                              )
                        )



            fig.update_layout(
                title_text = '2016 Hockey Stats',
                height = 800,
                margin = {'t':50, 'l':20},
                yaxis = {'domain': [0, .3]},
                yaxis2 = {'domain': [.4, 1], 'anchor': 'x2', 'title': 'Goals'},
                xaxis2 = {'anchor': 'y2'},
            )

            # Hide grid lines
            fig.update_xaxes(showgrid=False)
            fig.update_yaxes(showgrid=False)

            fig.show()
```

2016 Hockey Stats
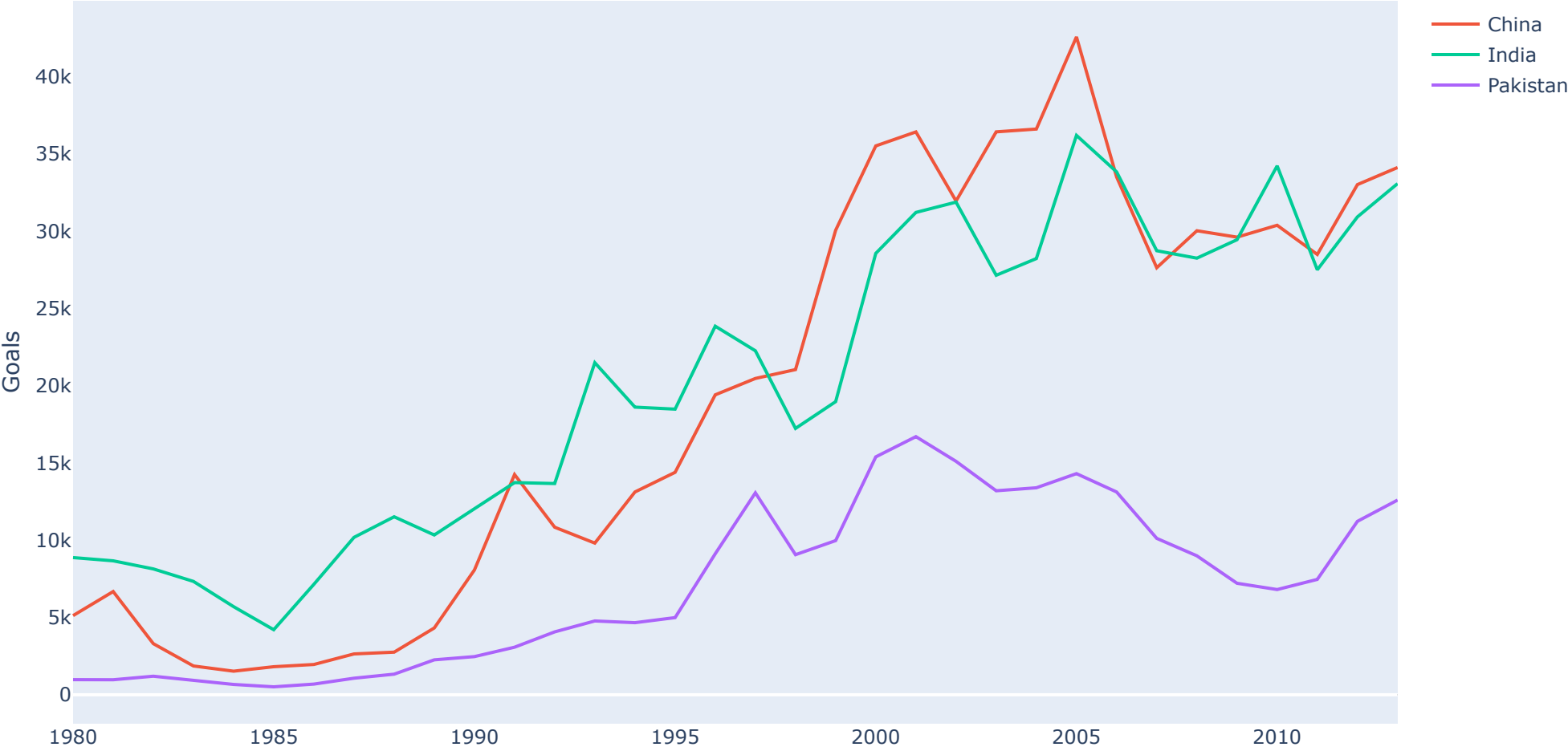
India	Pakistan	China	Australia	Germany	Austria
8880	978	5123	702	1626	234
8670	972	6682	639	1977	238
8147	1201	3308	484	3062	201
7338	900	1863	317	2376	117
5704	668	1527	317	1610	127
4211	514	1816	319	1441	165

Color scales in Plotly Express

```
#Sequential Color scales
fig = px.colors.sequential.swatches()
fig.update_layout(width = 990 , height = 1760)
fig.show()
```

plotly.colors.sequential
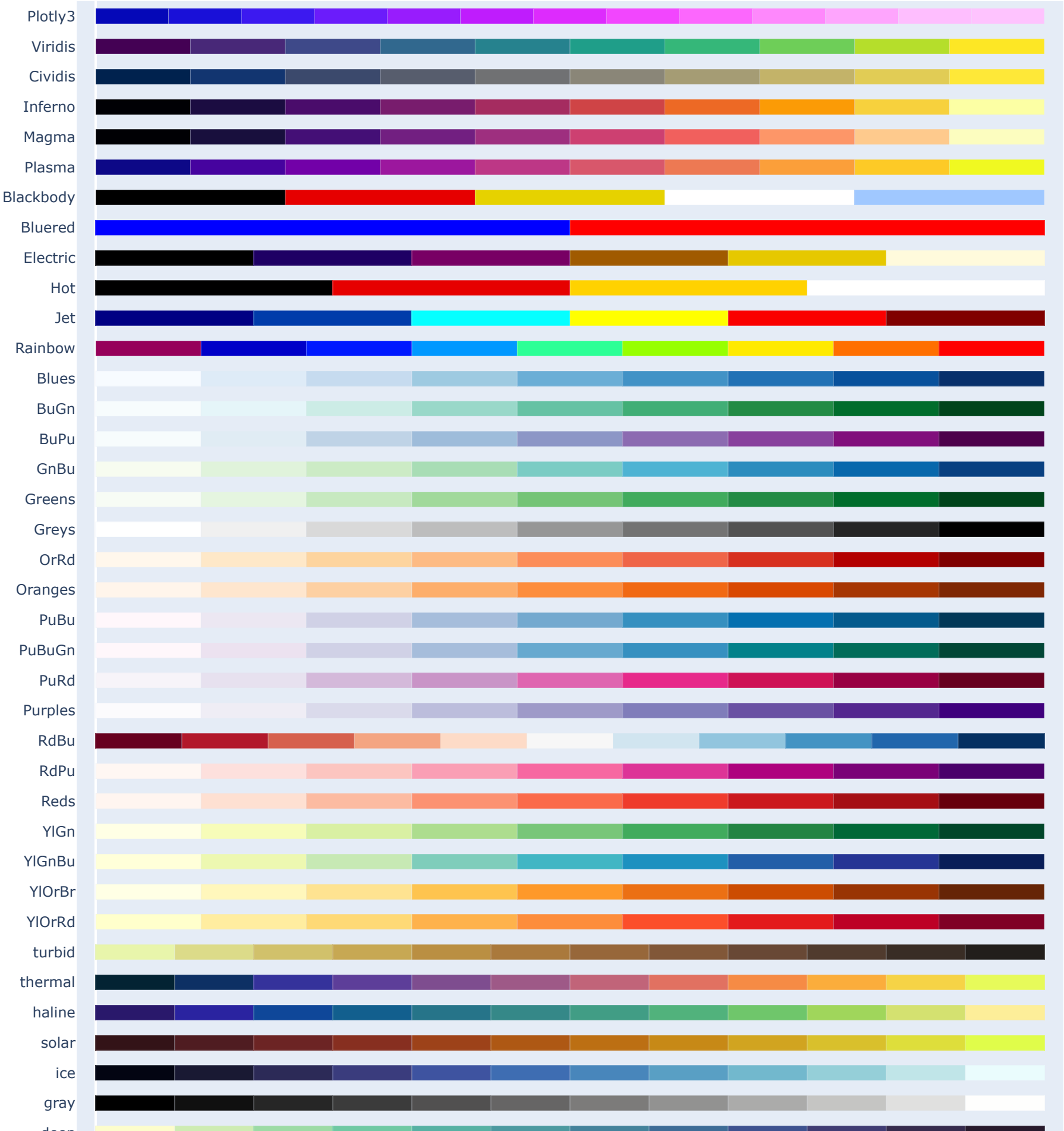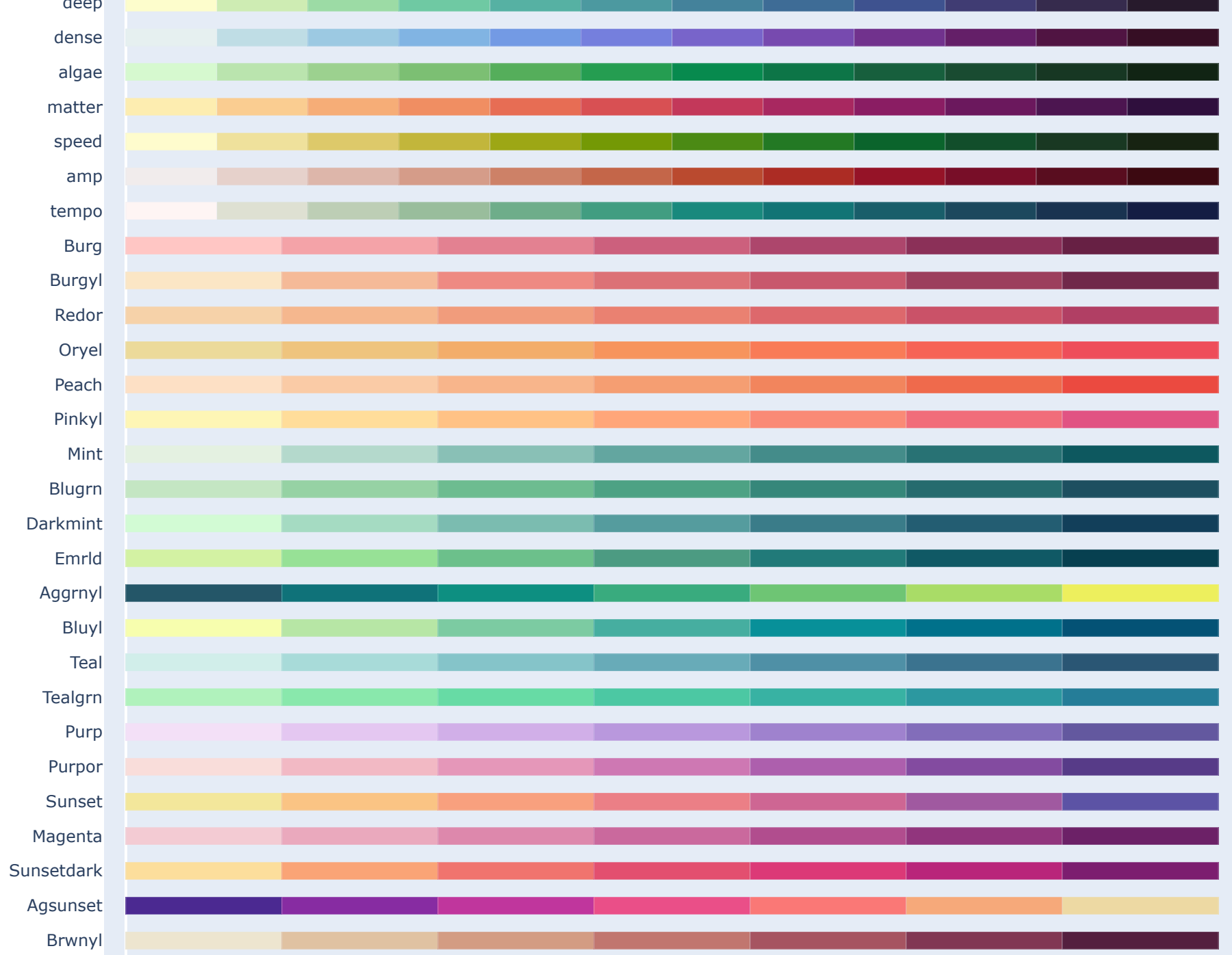
```
#Cyclical Color scales
fig = px.colors.cyclical.swatches_cyclical()
fig.show()
```

plotly.colors.cyclical
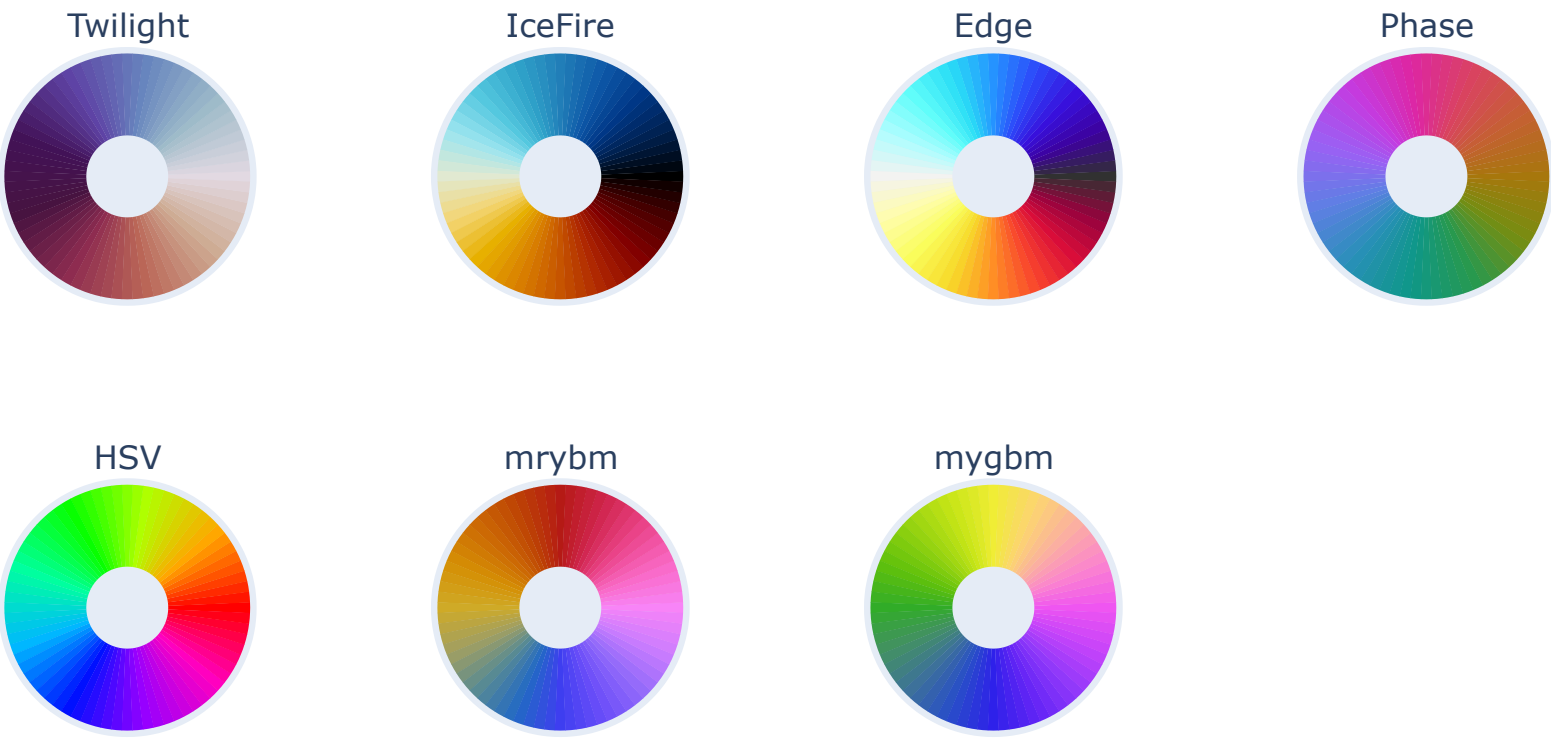
```
In [314]:  #Cyclical Color scales
           fig = px.colors.cyclical.swatches()
           fig.show()
```

plotly.colors.cyclical
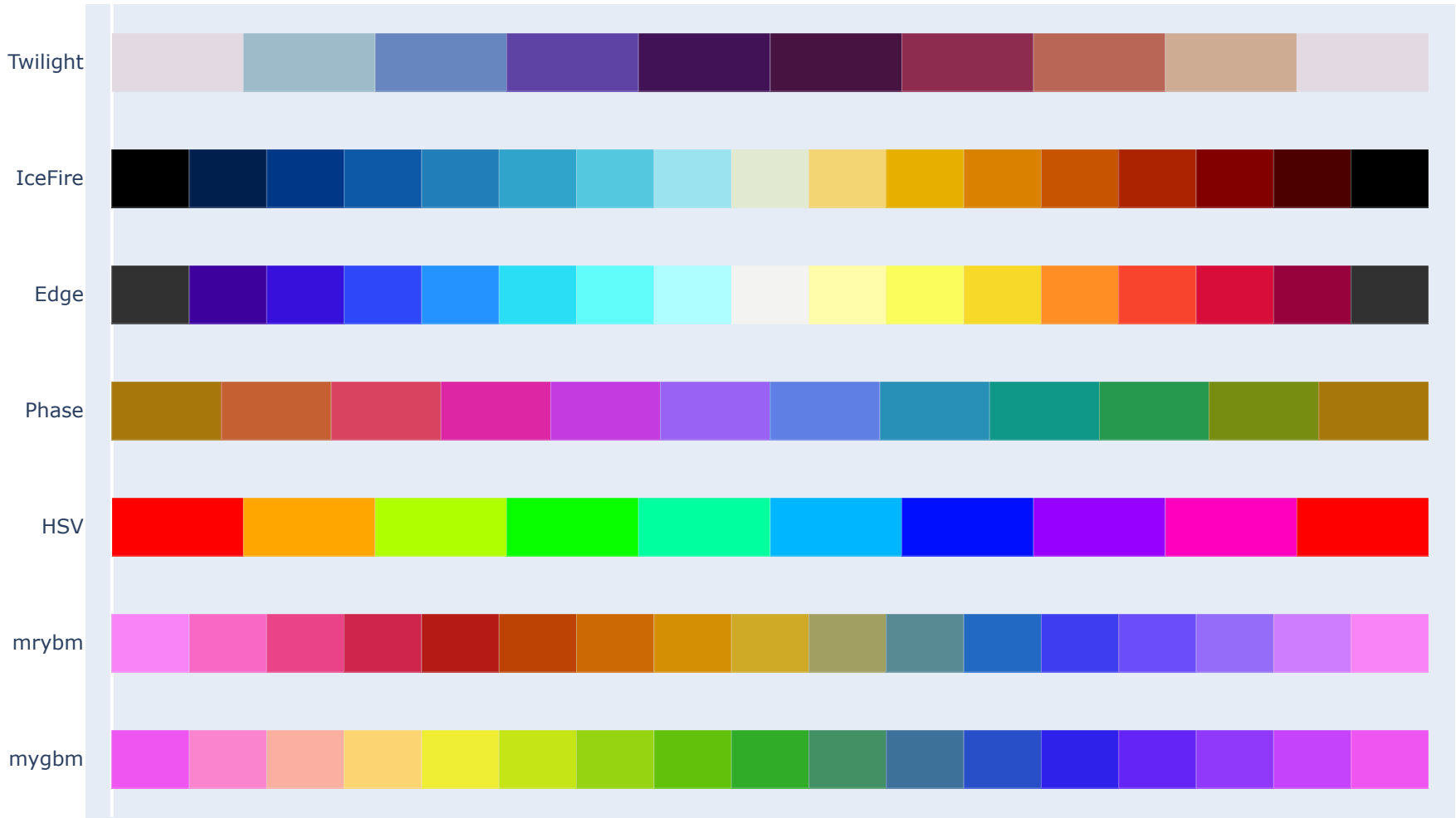
```
#Diverging Color scales
fig = px.colors.diverging.swatches().update_layout(margin_b=10)
fig.show()
```

plotly.colors.diverging
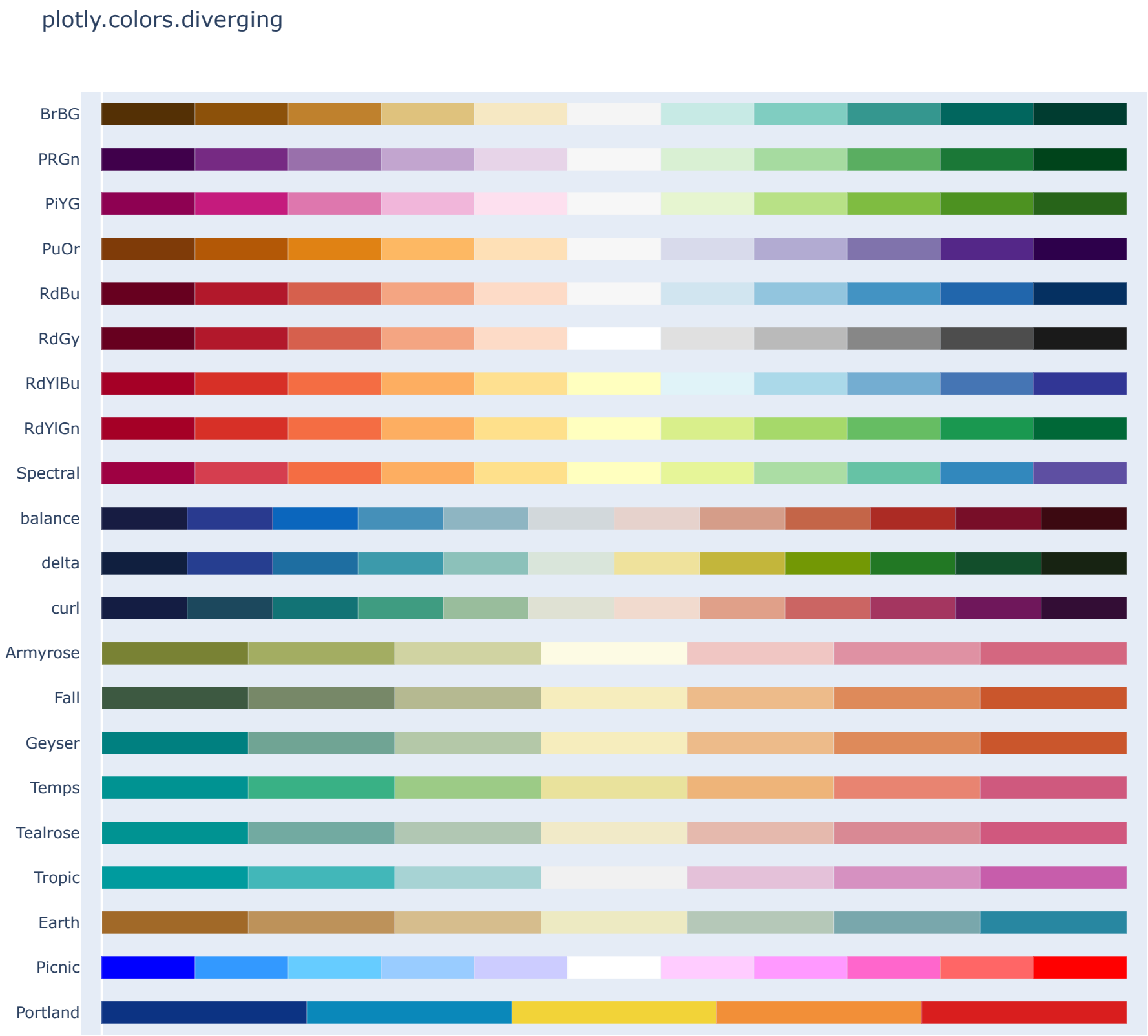
```
#Qualitative Color scales
fig = px.colors.qualitative.swatches()
fig.show()
```

plotly.colors.qualitative

Sunburst Chart

```
In [273]: insurance = pd.read_csv('C:/Users/DELL/Documents/GitHub/Data-Visualization/insurance.csv')
          insurance.head(10)
```

Out[273]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
5	31	female	25.740	0	no	southeast	3756.62160
6	46	female	33.440	1	no	southeast	8240.58960
7	37	female	27.740	3	no	northwest	7281.50560
8	37	male	29.830	2	no	northeast	6406.41070
9	60	female	25.840	0	no	northwest	28923.13692

```
# Simple Sunburst Chart

fig = px.sunburst(insurance, path=['region', 'smoker' , 'sex'], values='charges')
fig.update_layout (height = 800 , width = 800)
fig.show()
```
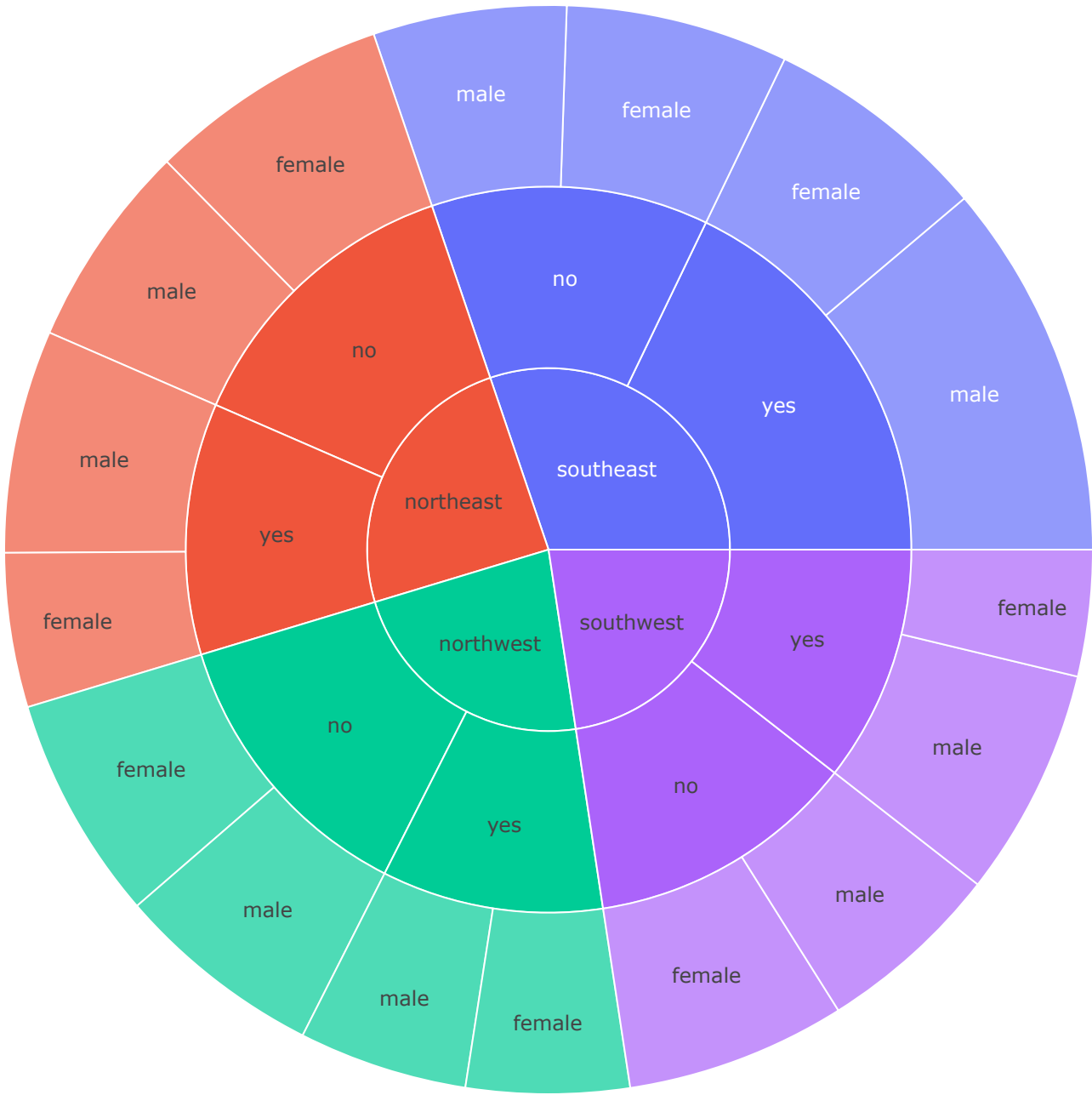
```python
fig = px.sunburst(insurance, path=['region', 'smoker' , 'sex'], values='charges' , color= 'charges')
fig.update_layout (height = 800 , width = 800)
fig.show()
```
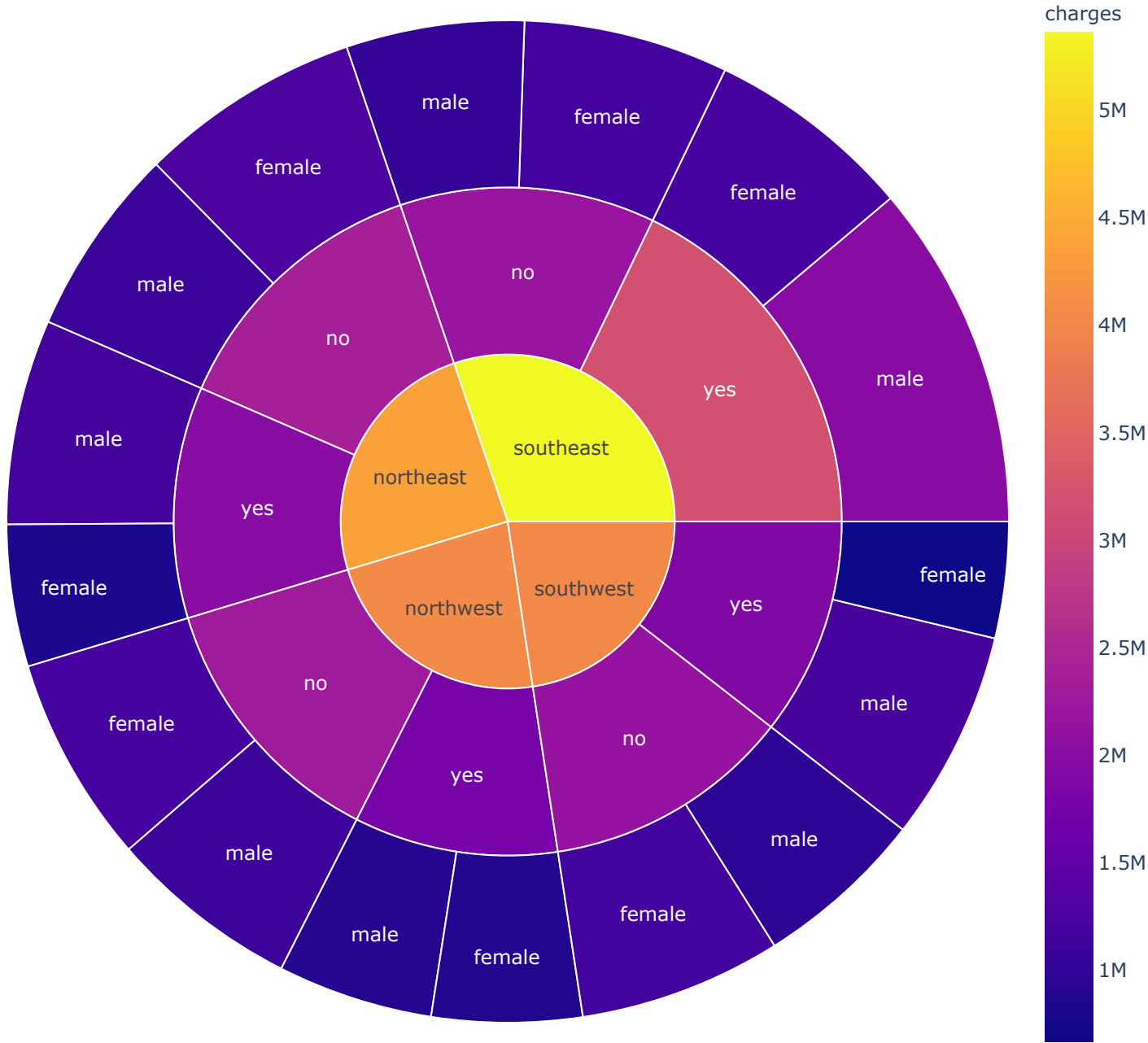
In [308]:
```python
# Using inbuilt color scales in Sunburst Chart

fig = px.sunburst(
                insurance,
                path=['region', 'smoker' , 'sex'],
                values='charges' , color= 'charges' ,
                color_continuous_scale=px.colors.sequential.Aggrnyl
                )

fig.update_layout (height = 800 , width = 800)
fig.show()
```
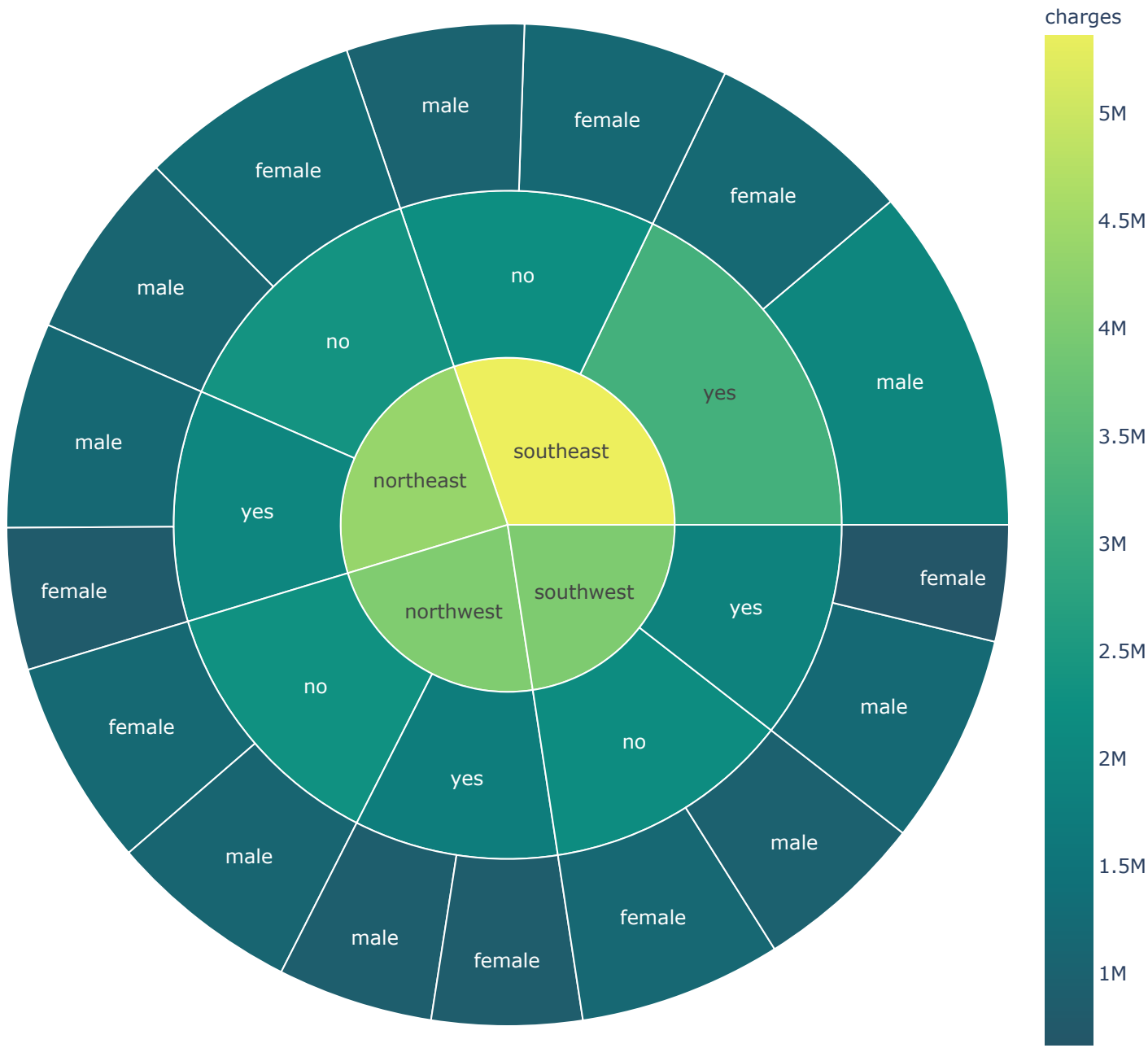
```
In [295]:   # Using color scales in Sunburst Chart

            fig = px.sunburst(
                        insurance,
                        path=['region', 'smoker' , 'sex'],
                        values='charges' ,
                        color= 'charges' ,
                        color_continuous_scale=["#8BC34A", "#FF6F00"] #Explicitly Constructing a Color Sequence
                        )

            fig.update_layout (height = 800 , width = 800)
            fig.show()
```

```python
# Using color scales in Sunburst Chart

fig = px.sunburst(
                insurance,
                path=['region', 'smoker' , 'sex'],
                values='charges' , color= 'charges' ,
                color_continuous_scale=["#689F38","#F9A825", "#FF6F00"] #Explicitly Constructing a Color Sequence
                )

#Controlling text fontsize with uniformtext
fig.update_layout (height = 800 , width = 800 , uniformtext=dict(minsize=14, mode='hide'))
fig.show()
```

```
In [65]:  suicide = pd.read_csv("suicide.csv")
          suicide.head(10)
```

Out[65]:

	country	year	sex	age	suicides_no	population	suicides/100k pop	country-year	HDI for year	gdp_for_year ($)	gdp_per_capita ($)	generation
0	Albania	1987	male	15-24 years	21	312900	6.71	Albania1987	NaN	2,156,624,900	796	Generation X
1	Albania	1987	male	35-54 years	16	308000	5.19	Albania1987	NaN	2,156,624,900	796	Silent
2	Albania	1987	female	15-24 years	14	289700	4.83	Albania1987	NaN	2,156,624,900	796	Generation X
3	Albania	1987	male	75+ years	1	21800	4.59	Albania1987	NaN	2,156,624,900	796	G.I. Generation
4	Albania	1987	male	25-34 years	9	274300	3.28	Albania1987	NaN	2,156,624,900	796	Boomers
5	Albania	1987	female	75+ years	1	35600	2.81	Albania1987	NaN	2,156,624,900	796	G.I. Generation
6	Albania	1987	female	35-54 years	6	278800	2.15	Albania1987	NaN	2,156,624,900	796	Silent
7	Albania	1987	female	25-34 years	4	257200	1.56	Albania1987	NaN	2,156,624,900	796	Boomers
8	Albania	1987	male	55-74 years	1	137500	0.73	Albania1987	NaN	2,156,624,900	796	G.I. Generation
9	Albania	1987	female	5-14 years	0	311000	0.00	Albania1987	NaN	2,156,624,900	796	Generation X

```
In [66]:  fig = px.sunburst(suicide, path=['sex', 'age' , 'generation'], values='suicides_no')
          fig.update_layout (height = 900 , width = 900)
          fig.show()
```
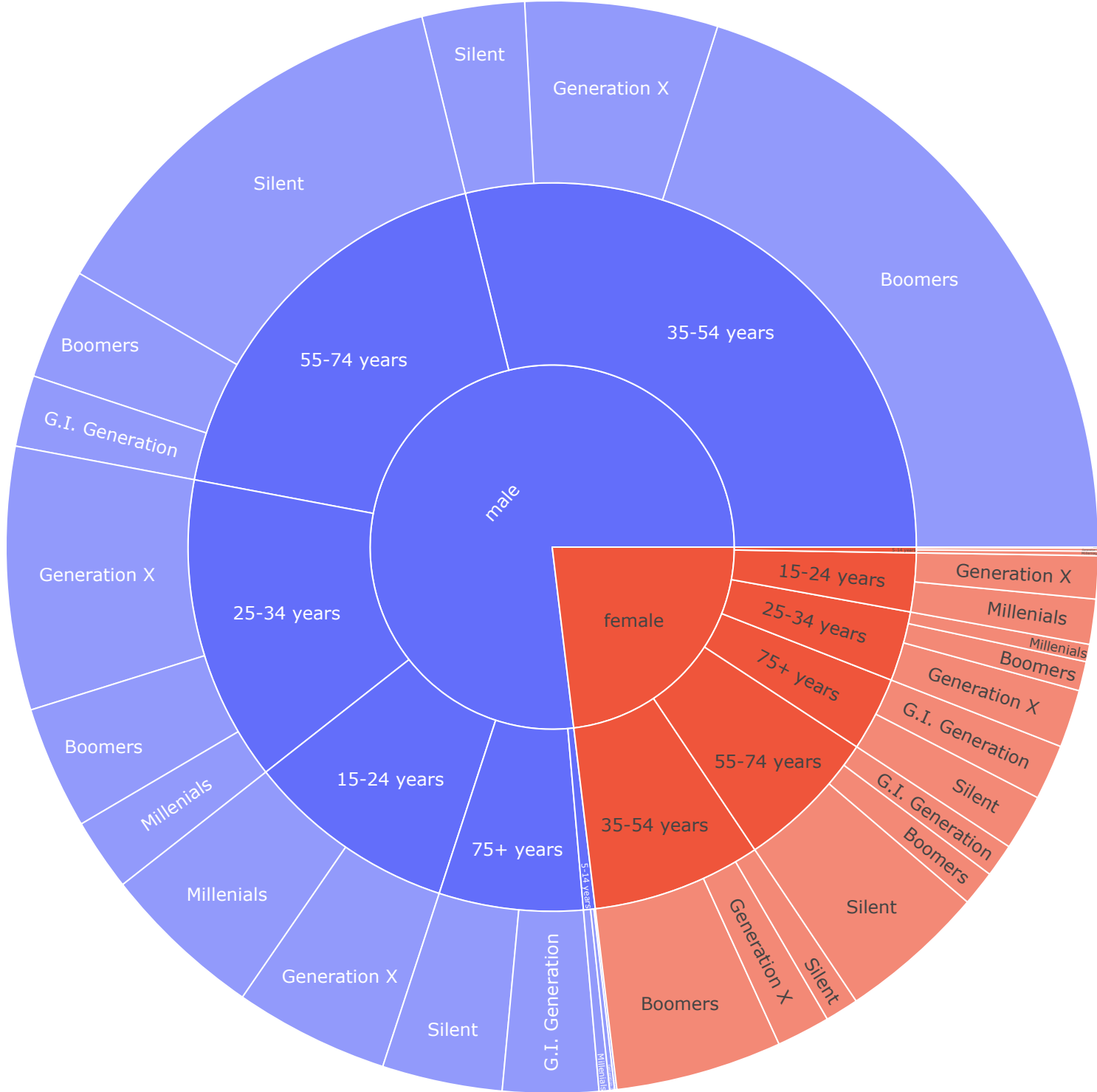
```
In [67]:  fig = px.sunburst(
                         suicide, path=['sex', 'age' , 'generation'],
                         values='suicides_no' ,
                         color= 'suicides_no' ,
                         color_continuous_scale=["#8BC34A","#FF6F00"]
                    )

          fig.update_layout (height = 900 , width = 900)
          fig.show()
```
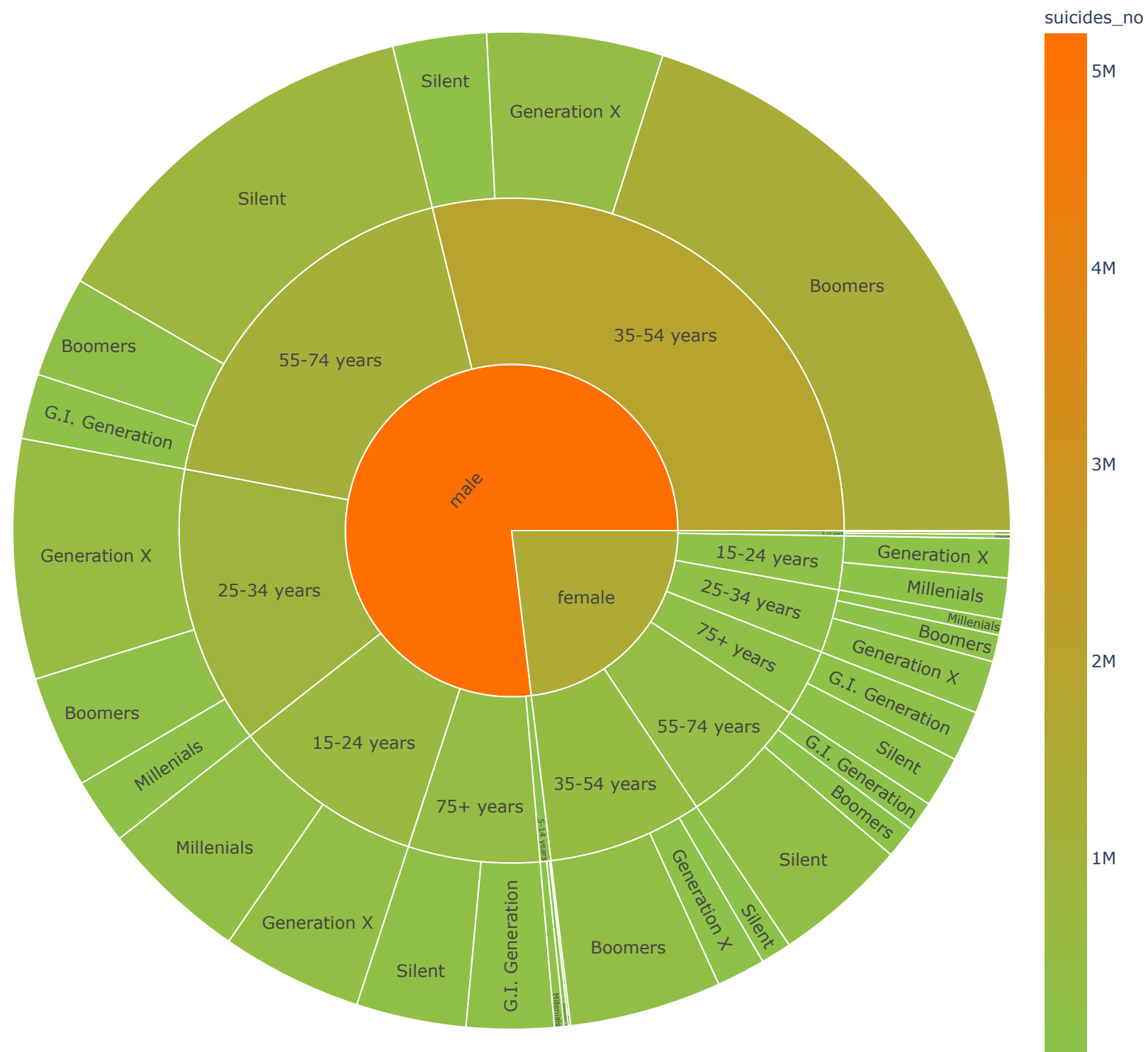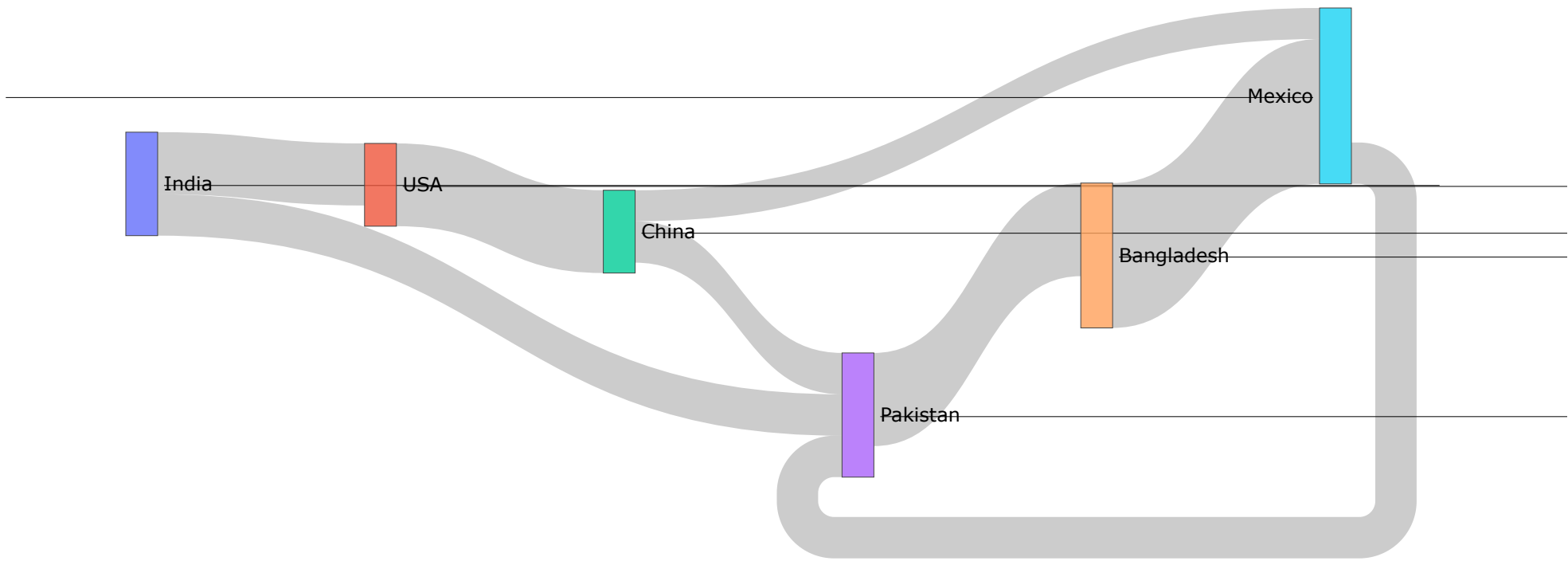
Sankey Diagram

```
In [123]:  #Simple Sankey Diagram

           fig = go.Figure(
                       go.Sankey(
                               node = {
                                       "label": ["India", "USA", "China", "Pakistan", "Bangladesh", "Mexico"],
                               },
                               link = {
                                       "source": [0, 1, 2, 3, 4, 0, 2, 5],
                                       "target": [1, 2, 3, 4, 5, 3, 5, 3],
                                       "value": [300, 400, 200, 450, 700, 200,150, 200]
                               }
                       )
               )

           # Hide grid lines
           fig.update_xaxes(showgrid=False)
           fig.update_yaxes(showgrid=False)

           fig.show()
```
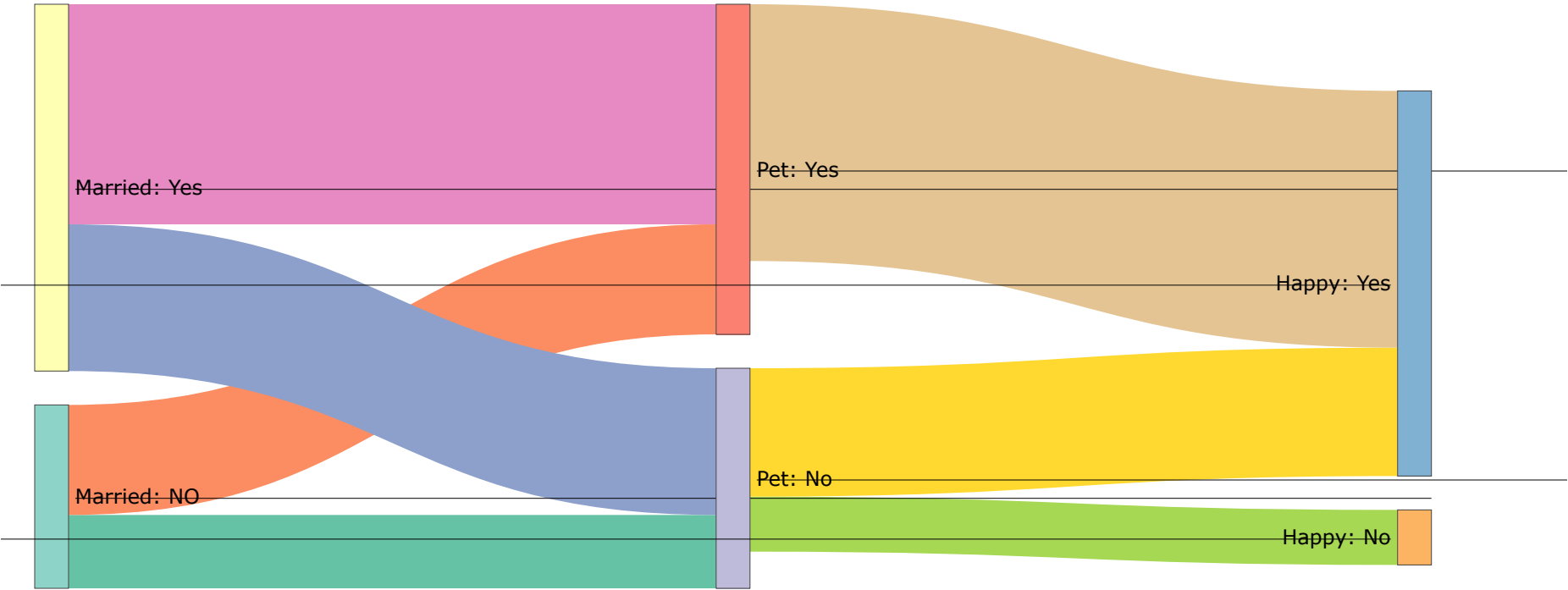
```python
In [124]: #Simple Sankey Diagram

fig = go.Figure(
            go.Sankey(
                node = dict(
                        thickness = 40, # Changing thickness of nodes
                        color = "lightgreen", # Changing color of the node
                        line = dict(color = "red", width = 0.5), # Changing line color
                        label = ["India", "USA", "China", "Pakistan", "Bangladesh", "Mexico"],

                ),
                link = {
                    "source": [0, 1, 2, 3, 4, 0, 2, 5],
                    "target": [1, 2, 3, 4, 5, 3, 5, 3],
                    "value": [300, 400, 200, 450, 550, 200,150, 200]
                }
            )
        )

# Hide grid lines
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)

fig.show()
```
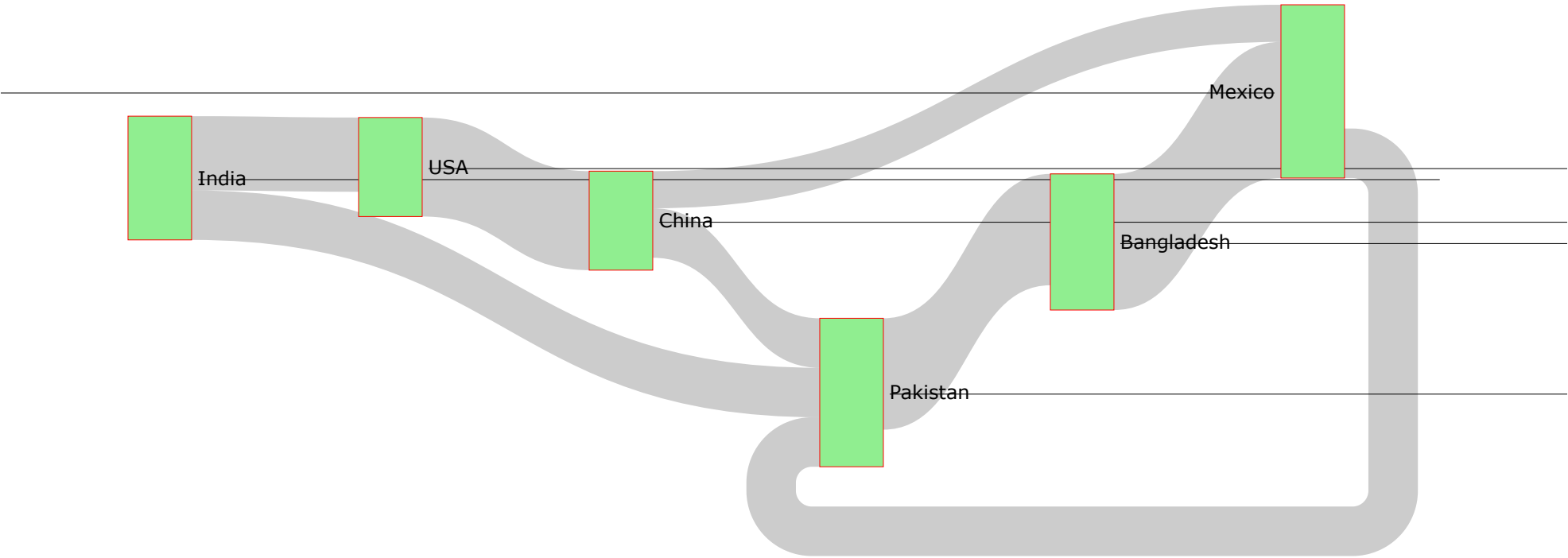
```
In [125]: #Simple Sankey Diagram

          fig = go.Figure(
                      go.Sankey(
                              node = {
                                      "label": ["Married: NO", "Married: Yes",
                                              "Pet: No", "Pet: Yes",
                                              "Happy: Yes", "Happy: No"],
                                      "color" : px.colors.qualitative.Set3 # Node color
                              },
                              link = dict(
                                      source = [0, 0, 1, 1, 2, 2, 3, 5],
                                      target = [2, 3, 2, 3, 5, 4, 4, 3],
                                      value = [200, 300, 400, 600, 150, 350,700],
                                      color = px.colors.qualitative.Set2 # Color of links
                                      )
                              )
                      )

          # Hide grid lines
          fig.update_xaxes(showgrid=False)
          fig.update_yaxes(showgrid=False)

          fig.show()
```



```
In [ ]:
```

In [ ]:

END