Installing plotly externally from Jupyter Notebook

In [ ]:
```
! pip install plotly
```

Packages Loader

In [1]:
```
# importing numpy
import numpy as np

# importing pandas
import pandas as pd

# importing plotly
import plotly
import plotly.express as px
import plotly.graph_objects as go
```

Gapminder Dataset

Country - Factor with 142 levels

Continent - Factor with 5 levels

Year - Ranges from 1952 to 2007 in increments of 5 years

lifeExp - Life expectancy at birth, in years

pop - Population

dgpPercap - GDP per capita

iso_alpha - The 3-digit ISO 3166-1 alpha-3 code - https://en.wikipedia.org/wiki/ISO_3166-1_alpha-3

iso_num - The 3-digit ISO 3166-1 numeric-3 code - https://en.wikipedia.org/wiki/ISO_3166-1_numeric

In [2]:
```
# Loading dataset in a Dataframe

df_gapminder = px.data.gapminder()


# Displaying first few lines of the Dataframe

df_gapminder.head()
```

Out[2]:

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_num
0	Afghanistan	Asia	1952	28.801	8425333	779.445314	AFG	4

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_num
1	Afghanistan	Asia	1957	30.332	9240934	820.853030	AFG	4
2	Afghanistan	Asia	1962	31.997	10267083	853.100710	AFG	4
3	Afghanistan	Asia	1967	34.020	11537966	836.197138	AFG	4
4	Afghanistan	Asia	1972	36.088	13079460	739.981106	AFG	4

In [3]:
```
# Learning about all the unique values in continent column

df_gapminder['continent'].unique()
```

Out[3]:
```
array(['Asia', 'Europe', 'Africa', 'Americas', 'Oceania'], dtype=object)
```

In [4]:
```
# Learning about all the unique values in country column

df_gapminder['country'].unique()
```

Out[4]:
```
array(['Afghanistan', 'Albania', 'Algeria', 'Angola', 'Argentina',
       'Australia', 'Austria', 'Bahrain', 'Bangladesh', 'Belgium',
       'Benin', 'Bolivia', 'Bosnia and Herzegovina', 'Botswana', 'Brazil',
       'Bulgaria', 'Burkina Faso', 'Burundi', 'Cambodia', 'Cameroon',
       'Canada', 'Central African Republic', 'Chad', 'Chile', 'China',
       'Colombia', 'Comoros', 'Congo, Dem. Rep.', 'Congo, Rep.',
       'Costa Rica', "Cote d'Ivoire", 'Croatia', 'Cuba', 'Czech Republic',
       'Denmark', 'Djibouti', 'Dominican Republic', 'Ecuador', 'Egypt',
       'El Salvador', 'Equatorial Guinea', 'Eritrea', 'Ethiopia',
       'Finland', 'France', 'Gabon', 'Gambia', 'Germany', 'Ghana',
       'Greece', 'Guatemala', 'Guinea', 'Guinea-Bissau', 'Haiti',
       'Honduras', 'Hong Kong, China', 'Hungary', 'Iceland', 'India',
       'Indonesia', 'Iran', 'Iraq', 'Ireland', 'Israel', 'Italy',
       'Jamaica', 'Japan', 'Jordan', 'Kenya', 'Korea, Dem. Rep.',
       'Korea, Rep.', 'Kuwait', 'Lebanon', 'Lesotho', 'Liberia', 'Libya',
       'Madagascar', 'Malawi', 'Malaysia', 'Mali', 'Mauritania',
       'Mauritius', 'Mexico', 'Mongolia', 'Montenegro', 'Morocco',
       'Mozambique', 'Myanmar', 'Namibia', 'Nepal', 'Netherlands',
       'New Zealand', 'Nicaragua', 'Niger', 'Nigeria', 'Norway', 'Oman',
       'Pakistan', 'Panama', 'Paraguay', 'Peru', 'Philippines', 'Poland',
       'Portugal', 'Puerto Rico', 'Reunion', 'Romania', 'Rwanda',
       'Sao Tome and Principe', 'Saudi Arabia', 'Senegal', 'Serbia',
       'Sierra Leone', 'Singapore', 'Slovak Republic', 'Slovenia',
       'Somalia', 'South Africa', 'Spain', 'Sri Lanka', 'Sudan',
       'Swaziland', 'Sweden', 'Switzerland', 'Syria', 'Taiwan',
       'Tanzania', 'Thailand', 'Togo', 'Trinidad and Tobago', 'Tunisia',
       'Turkey', 'Uganda', 'United Kingdom', 'United States', 'Uruguay',
       'Venezuela', 'Vietnam', 'West Bank and Gaza', 'Yemen, Rep.',
       'Zambia', 'Zimbabwe'], dtype=object)
```

In [5]:
```
# Learning about all the unique values in iso_alpha column

df_gapminder['iso_alpha'].unique()
```

Out[5]:
```
array(['AFG', 'ALB', 'DZA', 'AGO', 'ARG', 'AUS', 'AUT', 'BHR', 'BGD',
       'BEL', 'BEN', 'BOL', 'BIH', 'BWA', 'BRA', 'BGR', 'BFA', 'BDI',
       'KHM', 'CMR', 'CAN', 'CAF', 'TCD', 'CHL', 'CHN', 'COL', 'COM',
       'COD', 'COG', 'CRI', 'CIV', 'HRV', 'CUB', 'CZE', 'DNK', 'DJI',
       'DOM', 'ECU', 'EGY', 'SLV', 'GNQ', 'ERI', 'ETH', 'FIN', 'FRA',
       'GAB', 'GMB', 'DEU', 'GHA', 'GRC', 'GTM', 'GIN', 'GNB', 'HTI',
       'HND', 'HKG', 'HUN', 'ISL', 'IND', 'IDN', 'IRN', 'IRQ', 'IRL',
       'ISR', 'ITA', 'JAM', 'JPN', 'JOR', 'KEN', 'KOR', 'KWT', 'LBN',
       'LSO', 'LBR', 'LBY', 'MDG', 'MWI', 'MYS', 'MLI', 'MRT', 'MUS',
       'MEX', 'MNG', 'MNE', 'MAR', 'MOZ', 'MMR', 'NAM', 'NPL', 'NLD',
       'NZL', 'NIC', 'NER', 'NGA', 'NOR', 'OMN', 'PAK', 'PAN', 'PRY',
       'PER', 'PHL', 'POL', 'PRT', 'PRI', 'REU', 'ROU', 'RWA', 'STP',
       'SAU', 'SEN', 'SRB', 'SLE', 'SGP', 'SVK', 'SVN', 'SOM', 'ZAF',
       'ESP', 'LKA', 'SDN', 'SWZ', 'SWE', 'CHE', 'SYR', 'TWN', 'TZA',
       'THA', 'TGO', 'TTO', 'TUN', 'TUR', 'UGA', 'GBR', 'USA', 'URY',
       'VEN', 'VNM', 'PSE', 'YEM', 'ZMB', 'ZWE'], dtype=object)
```

In [6]:

```
# learning about all the unique values in iso_num column

df_gapminder['iso_num'].unique()
```

Out[6]:

```
array([  4,   8,  12,  24,  32,  36,  40,  48,  50,  56, 204,  68,  70,
        72,  76, 100, 854, 108, 116, 120, 124, 140, 148, 152, 156, 170,
       174, 180, 178, 188, 384, 191, 192, 203, 208, 262, 214, 218, 818,
       222, 226, 232, 231, 246, 250, 266, 270, 276, 288, 300, 320, 324,
       624, 332, 340, 344, 348, 352, 356, 360, 364, 368, 372, 376, 380,
       388, 392, 400, 404, 410, 414, 422, 426, 430, 434, 450, 454, 458,
       466, 478, 480, 484, 496, 499, 504, 508, 104, 516, 524, 528, 554,
       558, 562, 566, 578, 512, 586, 591, 600, 604, 608, 616, 620, 630,
       638, 642, 646, 678, 682, 686, 688, 694, 702, 703, 705, 706, 710,
       724, 144, 736, 748, 752, 756, 760, 158, 834, 764, 768, 780, 788,
       792, 800, 826, 840, 858, 862, 704, 275, 887, 894, 716], dtype=int64)
```

In [7]:

```
# learning about all the unique values in year column

df_gapminder['year'].unique()
```

Out[7]:

```
array([1952, 1957, 1962, 1967, 1972, 1977, 1982, 1987, 1992, 1997, 2002,
       2007], dtype=int64)
```

In [8]:

```
# Statistics across all the columns in the dataframe

print("All Statistics For Gapminder:")
df_gapminder.describe(include='all')
```

```
All Statistics For Gapminder:
```

Out[8]:

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_
count	1704	1704	1704.00000	1704.000000	1.704000e+03	1704.000000	1704	1704.00
unique	142	5	NaN	NaN	NaN	NaN	141	
top	Afghanistan	Africa	NaN	NaN	NaN	NaN	KOR	
freq	12	624	NaN	NaN	NaN	NaN	24	
mean	NaN	NaN	1979.50000	59.474439	2.960121e+07	7215.327081	NaN	425.88
std	NaN	NaN	17.26533	12.917107	1.061579e+08	9857.454543	NaN	248.30
min	NaN	NaN	1952.00000	23.599000	6.001100e+04	241.165876	NaN	4.00

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_
25%	NaN	NaN	1965.75000	48.198000	2.793664e+06	1202.060309	NaN	208.00
50%	NaN	NaN	1979.50000	60.712500	7.023596e+06	3531.846989	NaN	410.00
75%	NaN	NaN	1993.25000	70.845500	1.958522e+07	9325.462346	NaN	638.00
max	NaN	NaN	2007.00000	82.603000	1.318683e+09	113523.132900	NaN	894.00

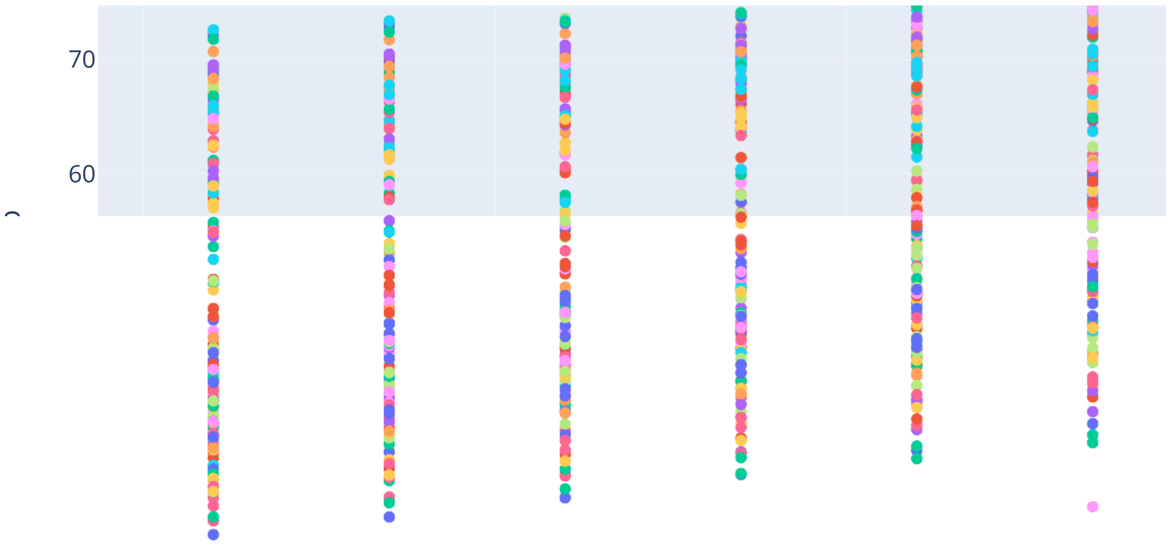
In [9]:

```
# Learning about all the columns, count of rows, data-type of each column, memory usage

df_gapminder.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1704 entries, 0 to 1703
Data columns (total 8 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   country    1704 non-null   object
 1   continent  1704 non-null   object
 2   year       1704 non-null   int64
 3   lifeExp    1704 non-null   float64
 4   pop        1704 non-null   int64
 5   gdpPercap  1704 non-null   float64
 6   iso_alpha  1704 non-null   object
 7   iso_num    1704 non-null   int64
dtypes: float64(2), int64(3), object(3)
memory usage: 106.6+ KB
```

In [10]:

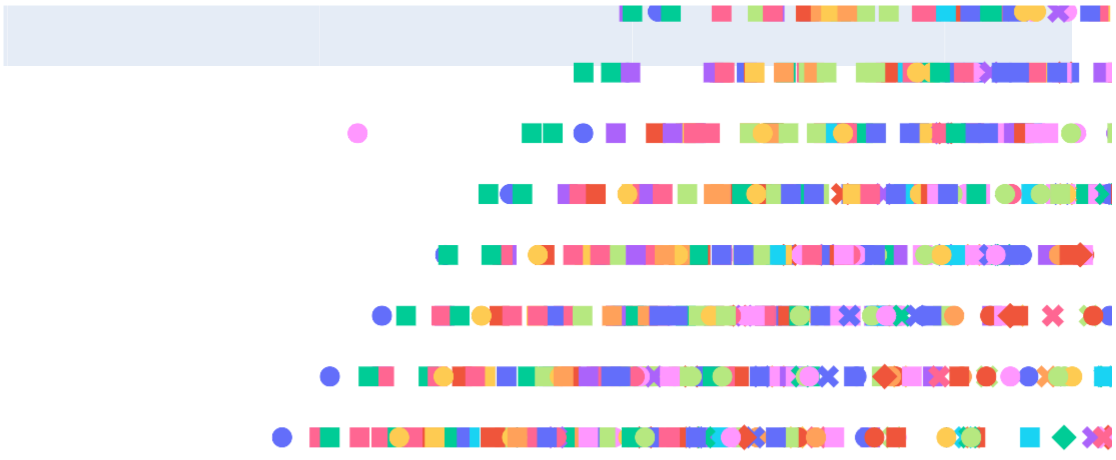
```
# Scatterplot of yearly change in life expectancy w.r.t. the country

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                       custom_data=None, text=None, facet_row=None, facet_col=None, fac
                       facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                       animation_frame=None, animation_group=None, category_orders=None
                       color_discrete_sequence=None, color_discrete_map=None, color_con
                       color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                       marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                       trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                       title=None, template=None, width=None, height=None)
"""

fig = px.scatter(df_gapminder, x="year", y="lifeExp", color='country')
fig.show()
```



In [11]:

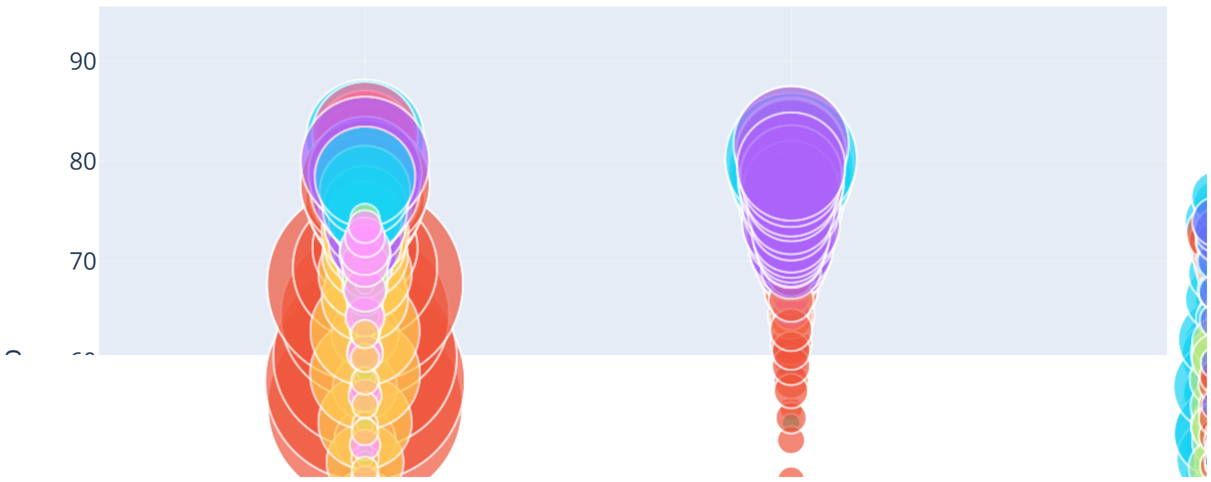
```
# Scatterplot of year vs life expectancy w.r.t. country & continent

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                       custom_data=None, text=None, facet_row=None, facet_col=None, fac
                       facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                       animation_frame=None, animation_group=None, category_orders=None
                       color_discrete_sequence=None, color_discrete_map=None, color_con
                       color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                       marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                       trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                       title=None, template=None, width=None, height=None)
"""

fig = px.scatter(df_gapminder, y="year", x="lifeExp", color="country", symbol="continen

fig.update_traces(marker_size=10)
```

2010
2000
1990



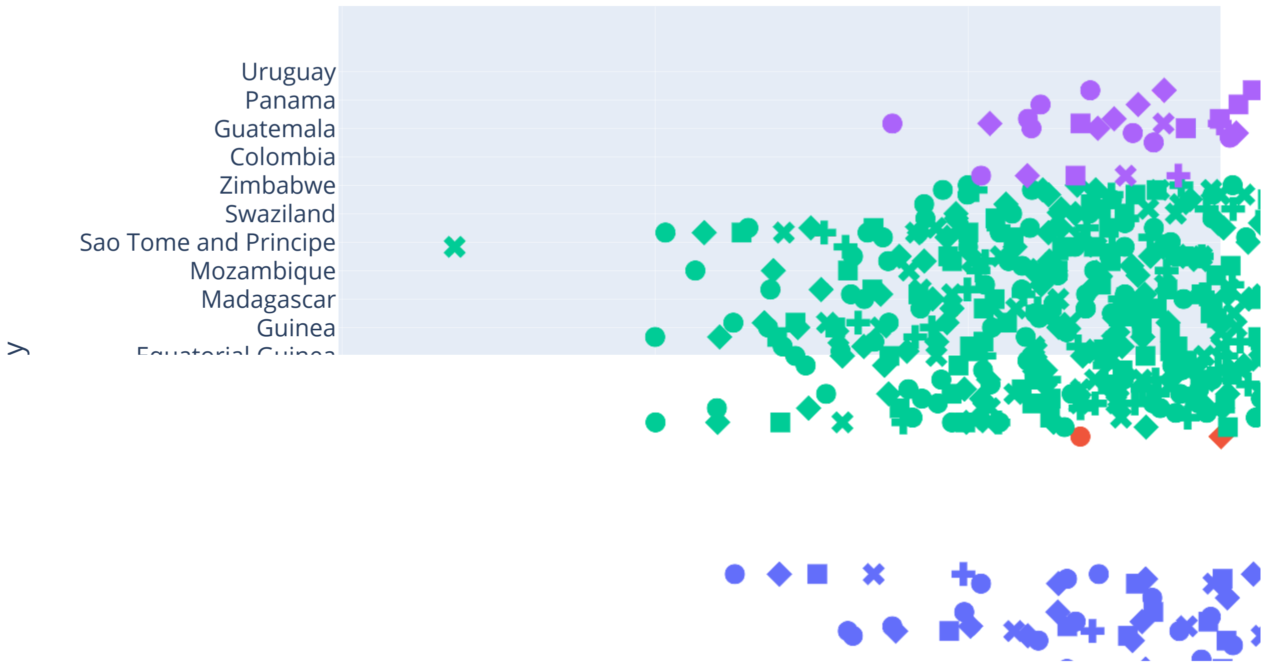
In [12]:

```
# Bubblechart of life expectancy across continents w.r.t. country & gdpPercap

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                       custom_data=None, text=None, facet_row=None, facet_col=None, fac
                       facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                       animation_frame=None, animation_group=None, category_orders=None
                       color_discrete_sequence=None, color_discrete_map=None, color_con
                       color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                       marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                       trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                       title=None, template=None, width=None, height=None)
"""

fig = px.scatter(df_gapminder, x='continent', y='lifeExp', color='country', size='gdpPe
fig.show()
```



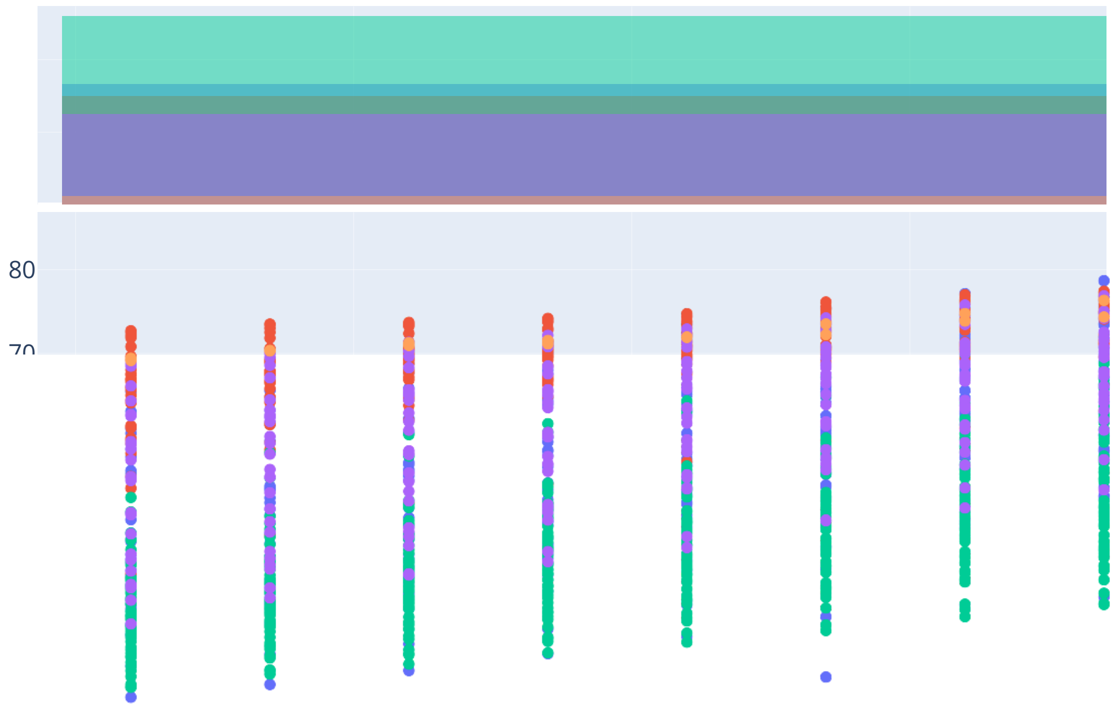
In [13]:

```
# Scatterplot of life expectancy across different country w.r.t. continent and year

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                       custom_data=None, text=None, facet_row=None, facet_col=None, fac
                       facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                       animation_frame=None, animation_group=None, category_orders=None
                       color_discrete_sequence=None, color_discrete_map=None, color_con
                       color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                       marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                       trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                       title=None, template=None, width=None, height=None)
"""

fig = px.scatter(df_gapminder, y="country", x="lifeExp", color="continent", symbol="yea
fig.update_traces(marker_size=10)
```

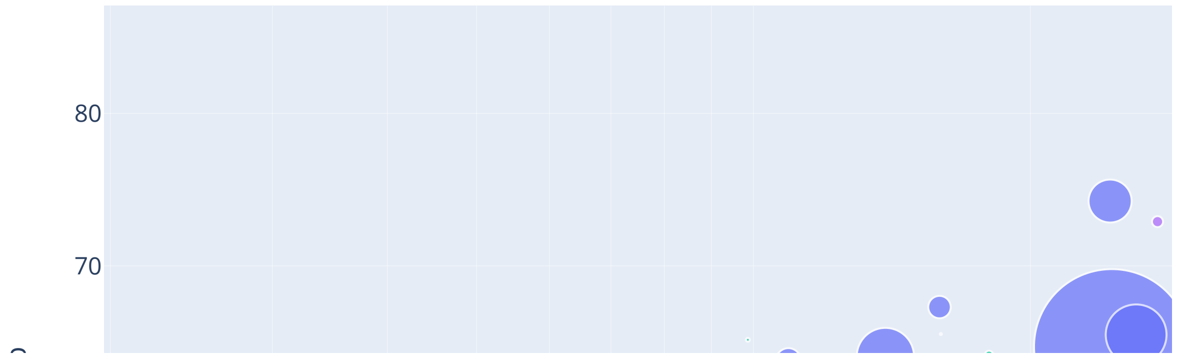


In [14]:

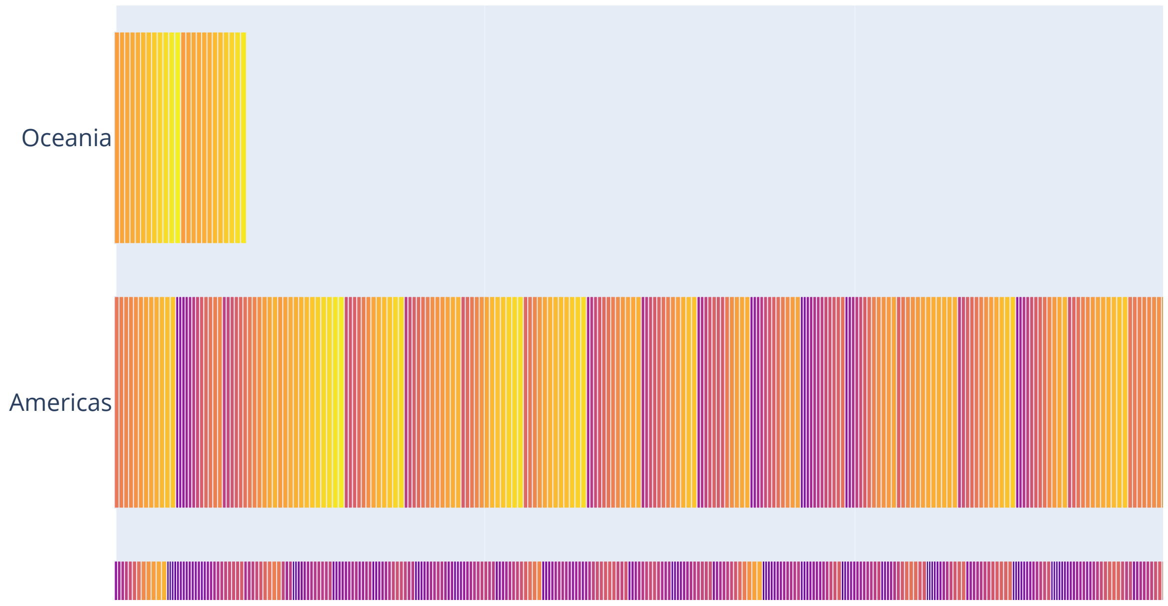
```
# Scatter plot of life expectency across different years represented as histogram. cont

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                       custom_data=None, text=None, facet_row=None, facet_col=None, fac
                       facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                       animation_frame=None, animation_group=None, category_orders=None
                       color_discrete_sequence=None, color_discrete_map=None, color_con
                       color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                       marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                       trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                       title=None, template=None, width=None, height=None)
"""

fig = px.scatter(df_gapminder, x="year", y="lifeExp", marginal_x="histogram", marginal_
fig.show()
```



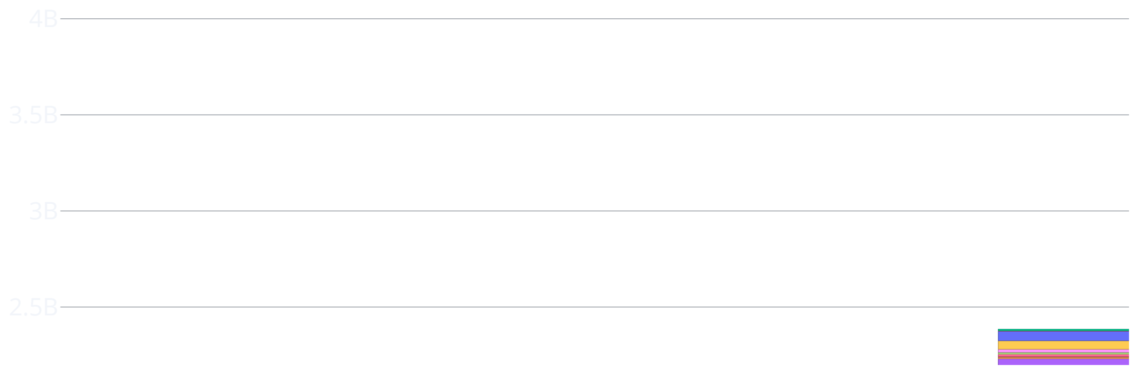
In [15]:
```
# Bubble chart of the year 2007, the life expectancy across different continents w.r.t.

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                       custom_data=None, text=None, facet_row=None, facet_col=None, fac
                       facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                       animation_frame=None, animation_group=None, category_orders=None
                       color_discrete_sequence=None, color_discrete_map=None, color_con
                       color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                       marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                       trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                       title=None, template=None, width=None, height=None)
"""

fig = px.scatter(df_gapminder.query("year==2007"), x="gdpPercap", y="lifeExp", size="po
                 hover_name="country", log_x=True, size_max=60)
fig.show()
```



In [16]:
```
# Bargraph to represent life expectancy across different continents with data hover of

"""
https://plotly.com/python-api-reference/generated/plotly.express.bar

plotly.express.bar(data_frame=None, x=None, y=None, color=None, pattern_shape=None, fac
                   facet_col_wrap=0, facet_row_spacing=None, facet_col_spacing=None, ho
                   custom_data=None, text=None, base=None, error_x=None, error_x_minus=
                   animation_frame=None, animation_group=None, category_orders=None, la
                   color_discrete_map=None, color_continuous_scale=None, pattern_shape_
                   range_color=None, color_continuous_midpoint=None, opacity=None, orie
                   log_x=False, log_y=False, range_x=None, range_y=None, text_auto=Fals
                   width=None, height=None)
"""

fig = px.bar(df_gapminder, x='lifeExp', y='continent', hover_data=['lifeExp', 'gdpPerca
fig.show()
```

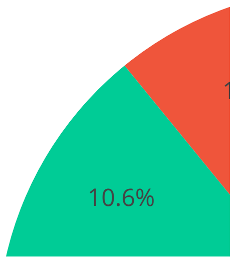


In [17]:
```
# Bargraph with dark background to represent year vs population of each country in Asia

"""
https://plotly.com/python-api-reference/generated/plotly.express.bar

plotly.express.bar(data_frame=None, x=None, y=None, color=None, pattern_shape=None, fac
                   facet_col_wrap=0, facet_row_spacing=None, facet_col_spacing=None, ho
                   custom_data=None, text=None, base=None, error_x=None, error_x_minus=
                   animation_frame=None, animation_group=None, category_orders=None, la
                   color_discrete_map=None, color_continuous_scale=None, pattern_shape_
                   range_color=None, color_continuous_midpoint=None, opacity=None, orie
                   log_x=False, log_y=False, range_x=None, range_y=None, text_auto=Fals
                   width=None, height=None)
"""

fig = px.bar(df_gapminder.query("continent == 'Asia'"), x='year', y='pop',barmode='stac
             template='plotly_dark')
fig.show()
```



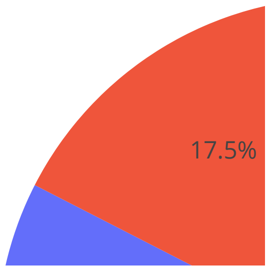
In [18]:
```
# Piechart to represent the population of Europe across different country

"""
https://plotly.com/python-api-reference/generated/plotly.express.pie.html

plotly.express.pie(data_frame=None, names=None, values=None, color=None, facet_row=None
                   facet_row_spacing=None, facet_col_spacing=None, color_discrete_seque
                   hover_name=None, hover_data=None, custom_data=None, category_orders=
                   template=None, width=None, height=None, opacity=None, hole=None)
"""

fig = px.pie(df_gapminder.query("continent == 'Europe'"), values='pop', names='country'
             title='Country-wise Population in Europe')
fig.show()
```

Country-wise Population in Europe



In [19]:
```
# Piechart to represent the population of Oceania across different country

"""
https://plotly.com/python-api-reference/generated/plotly.express.pie.html

plotly.express.pie(data_frame=None, names=None, values=None, color=None, facet_row=None
                   facet_row_spacing=None, facet_col_spacing=None, color_discrete_seque
                   hover_name=None, hover_data=None, custom_data=None, category_orders=
                   template=None, width=None, height=None, opacity=None, hole=None)
"""

fig = px.pie(df_gapminder.query("continent == 'Oceania'"), values='pop', names='country
             title='Country-wise Population in Oceania')
fig.show()
```

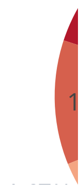
Country-wise Population in Oceania



In [20]:

```
# Piechart to represent the population of Americas across different country

"""
https://plotly.com/python-api-reference/generated/plotly.express.pie.html

plotly.express.pie(data_frame=None, names=None, values=None, color=None, facet_row=None
                   facet_row_spacing=None, facet_col_spacing=None, color_discrete_seque
                   hover_name=None, hover_data=None, custom_data=None, category_orders=
                   template=None, width=None, height=None, opacity=None, hole=None)
"""

fig = px.pie(df_gapminder.query("continent == 'Americas'"), values='pop', names='countr
             title='Country-wise Population in Americas', color_discrete_sequence=px.co
fig.show()
```

Country-wise Population in Americas



In [21]:

```
# Choropleth to represent 2007 with location as iso-alpha, life expectancy as shade

fig = px.choropleth(df_gapminder.query("year == 2007"), locations='iso_alpha', color='l
                    animation_frame='year', color_continuous_scale=px.colors.sequential
fig.show()
```

In [22]:

```
# Choropleth to represent 2007 with location as iso-alpha, gdp per capita as shade

fig = px.choropleth(df_gapminder.query("year == 2007"), locations='iso_alpha', color='g
                    animation_frame='year', color_continuous_scale=px.colors.sequential
fig.show()
```

In [23]:

```
# Choropleth to represent 2007 with location as iso-alpha, continent as projections sha

"""
https://plotly.com/python-api-reference/generated/plotly.express.line_geo.html

plotly.express.line_geo(data_frame=None, lat=None, lon=None, locations=None, locationmo
                        featureidkey=None, color=None, line_dash=None, text=None, facet
                        facet_col_wrap=0, facet_row_spacing=None, facet_col_spacing=Non
                        custom_data=None, line_group=None, symbol=None, animation_frame
                        category_orders=None, labels=None, color_discrete_sequence=None
                        line_dash_sequence=None, line_dash_map=None, symbol_sequence=No
                        projection=None, scope=None, center=None, fitbounds=None, basem
                        template=None, width=None, height=None)
"""

fig = px.line_geo(df_gapminder.query('year == 2007'), locations='iso_alpha', color='con
                  projection='winkel tripel')
fig.show()
```

Tips Dataset

In [24]:

```
# Loading the tips data in a dataframe

df_tips = px.data.tips()


# Displaying the head of the tips dataframe

df_tips.head()
```

Out[24]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2

	total_bill	tip	sex	smoker	day	time	size
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

In [25]:
```
# Descriptive statistics summary using describe function of the dataframe

df_tips.describe()
```

Out[25]:

	total_bill	tip	size
count	244.000000	244.000000	244.000000
mean	19.785943	2.998279	2.569672
std	8.902412	1.383638	0.951100
min	3.070000	1.000000	1.000000
25%	13.347500	2.000000	2.000000
50%	17.795000	2.900000	2.000000
75%	24.127500	3.562500	3.000000
max	50.810000	10.000000	6.000000

In [26]:
```
# All statistics summary using describe all function of the dataframe

print("All Statistics For Tips:\n")
df_tips.describe(include='all')
```

```
All Statistics For Tips:
```

Out[26]:

	total_bill	tip	sex	smoker	day	time	size
count	244.000000	244.000000	244	244	244	244	244.000000
unique	NaN	NaN	2	2	4	2	NaN
top	NaN	NaN	Male	No	Sat	Dinner	NaN
freq	NaN	NaN	157	151	87	176	NaN
mean	19.785943	2.998279	NaN	NaN	NaN	NaN	2.569672
std	8.902412	1.383638	NaN	NaN	NaN	NaN	0.951100
min	3.070000	1.000000	NaN	NaN	NaN	NaN	1.000000
25%	13.347500	2.000000	NaN	NaN	NaN	NaN	2.000000
50%	17.795000	2.900000	NaN	NaN	NaN	NaN	2.000000
75%	24.127500	3.562500	NaN	NaN	NaN	NaN	3.000000
max	50.810000	10.000000	NaN	NaN	NaN	NaN	6.000000

In [27]:
```
# Evaluating the missing values, non-null values, column datatypes in the dataframe

df_tips.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   total_bill  244 non-null    float64
 1   tip         244 non-null    float64
 2   sex         244 non-null    object 
 3   smoker      244 non-null    object 
 4   day         244 non-null    object 
 5   time        244 non-null    object 
 6   size        244 non-null    int64  
dtypes: float64(2), int64(1), object(4)
memory usage: 13.5+ KB
```

In [28]:
```
# Learning about all the unique values in day column

df_tips['day'].unique()
```

Out[28]:
```
array(['Sun', 'Sat', 'Thur', 'Fri'], dtype=object)
```

In [29]:
```
# Learning about all the unique values in time column

df_tips['time'].unique()
```

Out[29]:
```
array(['Dinner', 'Lunch'], dtype=object)
```

In [30]:
```
# Learning about all the unique values in sex column

df_tips['sex'].unique()
```

Out[30]:
```
array(['Female', 'Male'], dtype=object)
```

In [31]:
```
# Learning about all the unique values in smoker column

df_tips['smoker'].unique()
```

Out[31]:
```
array(['No', 'Yes'], dtype=object)
```

In [32]:
```
# Learning about all the unique values in size column

df_tips['size'].unique()
```

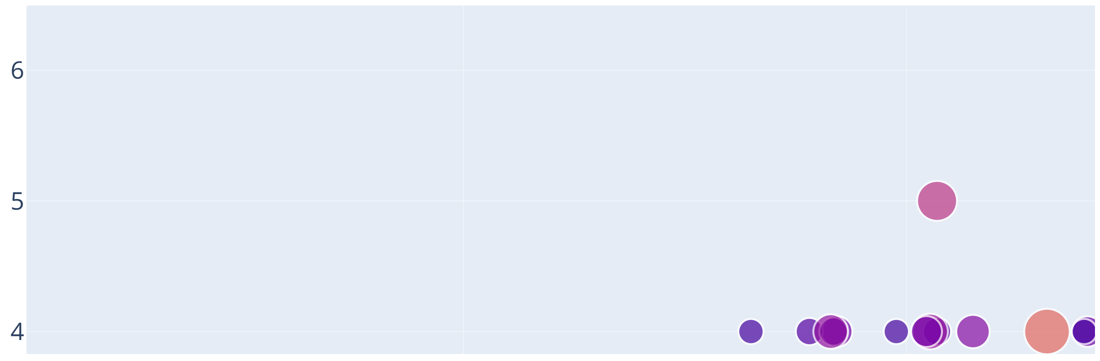
Out[32]:
```
array([2, 3, 4, 1, 6, 5], dtype=int64)
```

In [33]:
```
# Scatter plotting to represent total bill vs tips and the trendline
```

```
"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                       custom_data=None, text=None, facet_row=None, facet_col=None, fac
                       facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                       animation_frame=None, animation_group=None, category_orders=None
                       color_discrete_sequence=None, color_discrete_map=None, color_con
                       color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                       marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                       trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                       title=None, template=None, width=None, height=None)
"""

fig = px.scatter(df_tips, x='total_bill', y="tip", trendline="ols")
fig.show()
```

In [34]:
```
# Scatterplot to represent totalbill to the size, tip

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                       custom_data=None, text=None, facet_row=None, facet_col=None, fac
                       facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                       animation_frame=None, animation_group=None, category_orders=None
```

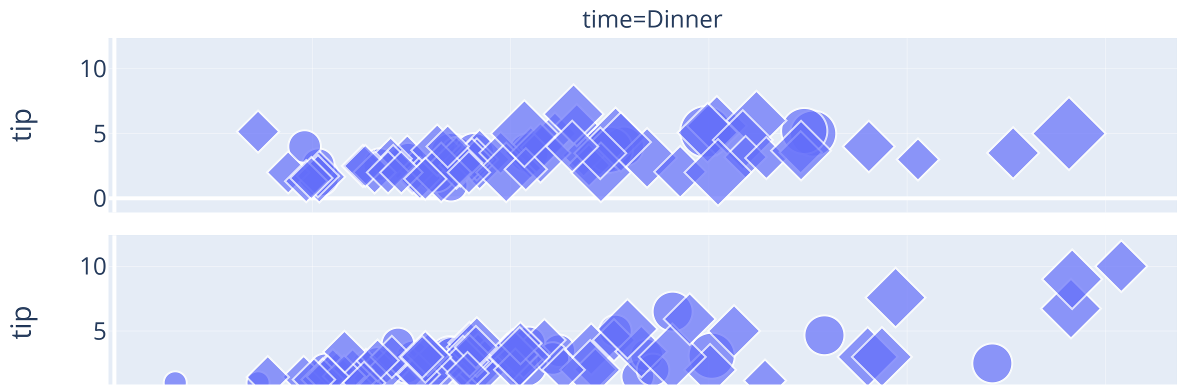
```
                           color_discrete_sequence=None, color_discrete_map=None, color_con
                           color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                           marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                           trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                           title=None, template=None, width=None, height=None)
"""

fig = px.scatter(df_tips, x="total_bill", y="size", size="tip", color="tip", size_max=2
fig.show()
```



In [35]:

```
# Scatterplot to represent totalbill to the tip, and colored as lunch or dinner

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                           custom_data=None, text=None, facet_row=None, facet_col=None, fac
                           facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                           animation_frame=None, animation_group=None, category_orders=None
                           color_discrete_sequence=None, color_discrete_map=None, color_con
                           color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                           marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                           trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                           title=None, template=None, width=None, height=None)
"""
```

```
fig = px.scatter(df_tips, x='total_bill', y="tip", color='time', symbol='sex', size='si
fig.show()
```



In [36]:

```
# Tips to total bill, smoker colored

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter.html

plotly.express.scatter(data_frame=None, x=None, y=None, color=None, symbol=None, size=N
                       custom_data=None, text=None, facet_row=None, facet_col=None, fac
                       facet_col_spacing=None, error_x=None, error_x_minus=None, error_
                       animation_frame=None, animation_group=None, category_orders=None
                       color_discrete_sequence=None, color_discrete_map=None, color_con
                       color_continuous_midpoint=None, symbol_sequence=None, symbol_map
                       marginal_x=None, marginal_y=None, trendline=None, trendline_opti
                       trendline_scope='trace', log_x=False, log_y=False, range_x=None,
                       title=None, template=None, width=None, height=None)
"""

fig = px.scatter(df_tips, x="total_bill", y="tip", color="smoker", facet_col="sex")
fig.show()
```

In [37]:

```
# Total bill to sex, tip as the third axis, colored for day of the week

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter_3d

plotly.express.scatter_3d(data_frame=None, x=None, y=None, z=None, color=None, symbol=N
                          hover_name=None, hover_data=None, custom_data=None, error_x=N
                          error_y_minus=None, error_z=None, error_z_minus=None, animati
                          category_orders=None, labels=None, size_max=None, color_discr
                          color_discrete_map=None, color_continuous_scale=None, range_c
                          color_continuous_midpoint=None, symbol_sequence=None, symbol_
                          log_x=False, log_y=False, log_z=False, range_x=None, range_y=
                          template=None, width=None, height=None)
"""

fig = px.scatter_3d(df_tips, x="total_bill", y="sex", z="tip", color = "day")
fig.show()
```

In [38]:

```
# plotting the figure of total bill to gender and z axis as tip, colored w.r.t. day and

"""
https://plotly.com/python-api-reference/generated/plotly.express.scatter_3d

plotly.express.scatter_3d(data_frame=None, x=None, y=None, z=None, color=None, symbol=N
                          hover_name=None, hover_data=None, custom_data=None, error_x=N
                          error_y_minus=None, error_z=None, error_z_minus=None, animati
                          category_orders=None, labels=None, size_max=None, color_discr
                          color_discrete_map=None, color_continuous_scale=None, range_c
                          color_continuous_midpoint=None, symbol_sequence=None, symbol_
                          log_x=False, log_y=False, log_z=False, range_x=None, range_y=
                          template=None, width=None, height=None)
"""

fig = px.scatter_3d(df_tips, x="total_bill", y="sex", z="tip", color='day', size='total
fig.show()
```

In [39]:

```
# plotting the violin w.r.t. day vs tip vs gender and lunch or dinner as facet plot

"""
https://plotly.com/python-api-reference/generated/plotly.express.violin.html

lotly.express.violin(data_frame=None, x=None, y=None, color=None, facet_row=None, facet
                     facet_row_spacing=None, facet_col_spacing=None, hover_name=None, h
                     animation_frame=None, animation_group=None, category_orders=None,
                     color_discrete_sequence=None, color_discrete_map=None, orientation
                     log_x=False, log_y=False, range_x=None, range_y=None, points=None,
                     template=None, width=None, height=None)
"""

fig = px.violin(df_tips, x="day", y="tip", color='sex', facet_row='time', box=True)
fig.show()
```

In [40]:
```
# plotting the histogram of total bill

"""
https://plotly.com/python-api-reference/generated/plotly.express.histogram.html

plotly.express.histogram(data_frame=None, x=None, y=None, color=None, pattern_shape=Non
                         facet_col_wrap=0, facet_row_spacing=None, facet_col_spacing=No
                         animation_frame=None, animation_group=None, category_orders=No
                         color_discrete_sequence=None, color_discrete_map=None, pattern
                         pattern_shape_map=None, marginal=None, opacity=None, orientati
                         barnorm=None, histnorm=None, log_x=False, log_y=False, range_x
                         cumulative=None, nbins=None, text_auto=False, title=None, temp
"""

fig = px.histogram(df_tips, x="total_bill")
fig.show()
```

In [41]:
```
# plotting the histogram with sex, tip, smoker as colored separation, marginal type as

"""
https://plotly.com/python-api-reference/generated/plotly.express.histogram.html
```

```
plotly.express.histogram(data_frame=None, x=None, y=None, color=None, pattern_shape=Non
                         facet_col_wrap=0, facet_row_spacing=None, facet_col_spacing=No
                         animation_frame=None, animation_group=None, category_orders=No
                         color_discrete_sequence=None, color_discrete_map=None, pattern
                         pattern_shape_map=None, marginal=None, opacity=None, orientati
                         barnorm=None, histnorm=None, log_x=False, log_y=False, range_x
                         cumulative=None, nbins=None, text_auto=False, title=None, temp
"""

fig = px.histogram(df_tips, x="sex", y="tip", color="smoker", marginal="box", hover_dat
fig.show()
```

In [42]:
```
# plotting the histogram with day, tip, sex, marginal as box

"""
https://plotly.com/python-api-reference/generated/plotly.express.histogram.html

plotly.express.histogram(data_frame=None, x=None, y=None, color=None, pattern_shape=Non
                         facet_col_wrap=0, facet_row_spacing=None, facet_col_spacing=No
                         animation_frame=None, animation_group=None, category_orders=No
                         color_discrete_sequence=None, color_discrete_map=None, pattern
                         pattern_shape_map=None, marginal=None, opacity=None, orientati
                         barnorm=None, histnorm=None, log_x=False, log_y=False, range_x
                         cumulative=None, nbins=None, text_auto=False, title=None, temp
"""
```

```
fig = px.histogram(df_tips, x="day", y="tip", color="sex", marginal="box", hover_data=d
fig.show()
```