In [88]:
```
import plotly.express as px
'''plotly.express templates :

'plotly', 'plotly_white', 'plotly_dark',
'ggplot2', 'seaborn', 'simple_white',
'none', 'presentation', 'xgridoff', 'ygridoff',
'gridon'
'''
from itables.dash import ITable
#from itables import show
from itables import init_notebook_mode
init_notebook_mode(connected=True)
```

PANDAS AND SQL

In [90]:
```
!pip install psycopg2
from sqlalchemy import create_engine ,inspect, MetaData, Table
engine = create_engine("postgresql+psycopg2://postgres:PASSWORD@LOCALHOST:PORT/World")
engine.connect()
inspector = inspect(engine)
# get the tables in the schema public
inspector.get_table_names(schema="public")
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: psycopg2 in c:\users\cj ingene\appdata\roaming\python\python312\site-packages (2.9.10)
```

Out[90]:
```
['city', 'country', 'countrylanguage']
```

In [91]:
```
# get the schemas names
inspector.get_schema_names()
```

Out[91]:
```
['information_schema', 'public']
```

In [92]:
```
# get the columns of the table and their information
inspector.get_columns("city")
```

Out[92]:
```
[{'name': 'id',
  'type': INTEGER(),
  'nullable': False,
  'default': None,
  'autoincrement': False,
  'comment': None},
 {'name': 'name',
  'type': TEXT(),
  'nullable': False,
  'default': None,
  'autoincrement': False,
  'comment': None},
 {'name': 'countrycode',
  'type': CHAR(length=3),
  'nullable': False,
  'default': None,
  'autoincrement': False,
  'comment': None},
 {'name': 'district',
  'type': TEXT(),
  'nullable': False,
  'default': None,
  'autoincrement': False,
  'comment': None},
 {'name': 'population',
  'type': INTEGER(),
  'nullable': False,
  'default': None,
  'autoincrement': False,
  'comment': None}]
```

In [93]:
```
 # get the structure of the table
metadata = MetaData()
metadata.reflect(bind=engine)
table = metadata.tables['city']
for col in table.columns:
    print(f"{col.name} - {col.type}-{col.nullable} - {col.default} - {col.primary_key} - {col.foreign_keys}")
```

```
id - INTEGER-False - None - True - set()
name - TEXT-False - None - False - set()
countrycode - CHAR(3)-False - None - False - set()
district - TEXT-False - None - False - set()
population - INTEGER-False - None - False - set()
```

In [94]:
```
import pandas as pd
query = "SELECT * FROM city"
city = pd.read_sql_query(query, con=engine)
city.head(5)
```

Out[94]:

id	name	countrycode	district	population
1	Kabul	AFG	Kabol	1780000
2	Qandahar	AFG	Qandahar	237500
3	Herat	AFG	Herat	186800
4	Mazar-e-Sharif	AFG	Balkh	127800
5	Amsterdam	NLD	Noord-Holland	731200

In [95]:
```
import pandas as pd
query= "SELECT * FROM country"
country = pd.read_sql_query(query, con=engine)
country.head(5)
```

Out[95]:

code	name	continent	region	surfacearea	indepyear	population	lifeexpectancy	gnp	gnpold	localname	governmentform	headofstate	capital	code2
AFG	Afghanistan	Asia	Southern and Central Asia	652090	1919	22720000	45.9	5976	NaN	Afganistan/Afqanestan	Islamic Emirate	Mohammad Omar	1	AF
NLD	Netherlands	Europe	Western Europe	41526	1581	15864000	78.3	371362	360478	Nederland	Constitutional Monarchy	Beatrix	5	NL
ANT	Netherlands Antilles	North America	Caribbean	800	NaN	217000	74.7	1941	NaN	Nederlandse Antillen	Nonmetropolitan Territory of The Netherlands	Beatrix	33	AN
ALB	Albania	Europe	Southern Europe	28748	1912	3401200	71.6	3205	2500	Shqipëria	Republic	Rexhep Mejdani	34	AL
DZA	Algeria	Africa	Northern Africa	2381741	1962	31471000	69.7	49982	46966	Al-Jaza□ir/Algérie	Republic	Abdelaziz Bouteflika	35	DZ

In [96]:
```
number = country["region"].value_counts()
number.columns=["region", "numbers"]
number
# Number of countries per region
px.bar(number,
       x=number.index,
       y=country["region"].value_counts(),
       title='Number of countries per region',
       labels={'x':'Continent', 'y':'Number of '},
       template='plotly',
       #color_discrete_sequence=['#080070'],
       color=number.index,
       ).show()
```

```
In [97]: #numbers of countries per continent
         continent = country.groupby('continent').agg({'code':'count'})
         continent
```

Out[97]:

	code
continent	
Africa	58
Antarctica	5
Asia	51
Europe	46
North America	37
Oceania	28
South America	14

```
In [98]: # mean of life expectancy per continent
         meanoflifeexpectancy = country.groupby('continent').agg({'lifeexpectancy':'mean'})
         meanoflifeexpectancy
         px.bar(meanoflifeexpectancy,
                x=continent.index,
                y='lifeexpectancy',
                title='mean of life expectancy per continent',
                labels={'x':'Continent', 'y':'Number of countries'},
                template='plotly',
                color_discrete_sequence=['#0F5030'],
                ).show()
```

In [99]:
```
px.bar(continent,
       x=continent.index,
       y='code',
       title='Number of countries per continent',
       labels={'x':'Continent', 'y':'Number of countries'},
       template='gridon',
       color=continent.index
       ).show()
```

In [100…
```
query = "SELECT * FROM countrylanguage"
countrylanguage= pd.read_sql_query(query, con=engine)
countrylanguage.head(5)
```

Out[100…

countrycode	language	isofficial	percentage
AFG	Pashto	true	52.4
NLD	Dutch	true	95.6
ANT	Papiamento	true	86.2
ALB	Albaniana	true	97.9
DZA	Arabic	true	86

In [101…

```python
px.bar(countrylanguage,
       x=countrylanguage['countrycode'],
       y='percentage',
       title='Percentage of languages per country',
       labels={'x':'Country', 'y':'Percentage'},
       template='simple_white',
       color=countrylanguage.index
       ).show()
```

In [102…

```python
import pandas as pd
query = "SELECT * FROM city INNER JOIN countrylanguage ON city.countrycode = countrylanguage.countrycode;"
combine = pd.read_sql_query(query, con=engine)
combine.head(5)
```

Out[102…

id	name	countrycode	district	population	countrycode	language	isofficial	percentage
4	Mazar-e-Sharif	AFG	Balkh	127800	AFG	Pashto	true	52.4
3	Herat	AFG	Herat	186800	AFG	Pashto	true	52.4
2	Qandahar	AFG	Qandahar	237500	AFG	Pashto	true	52.4
1	Kabul	AFG	Kabol	1780000	AFG	Pashto	true	52.4
32	Alkmaar	NLD	Noord-Holland	92713	NLD	Dutch	true	95.6

In [103…

```python
import dask.dataframe as dd

df = dd.read_sql_table(
    table_name='orders', # table name in the database
    con='postgresql://postgres:PASSWORD@LOCALHOST/Store', # connection string
    index_col='orderid',# index column in the table
    npartitions=10  # number of partitions to create
)
df.head(10)
```

Out[103…

	orderdate	customerid	netamount	tax	totalamount
orderid					
1	2004-01-27	7888	313.24	25.84	339.08
2	2004-01-01	4858	54.9	4.53	59.43
3	2004-01-17	15399	160.1	13.21	173.31
4	2004-01-28	17019	106.67	8.8	115.47
5	2004-01-09	14771	256	21.12	277.12
6	2004-01-11	13734	382.59	31.56	414.15
7	2004-01-05	17622	256.44	21.16	277.6
8	2004-01-18	8331	67.85	5.6	73.45
9	2004-01-06	14902	29.82	2.46	32.28
10	2004-01-18	15112	20.78	1.71	22.49

SQLITE

In [105…

```
import sqlite3 # import the sqlite3 module
conn = sqlite3.connect("chinook.db") # create a connection to the database
cursor = conn.cursor() # create a cursor object to execute SQL commands
cursor.execute("SELECT name FROM sqlite_master WHERE type='table';")
tables = cursor.fetchall()
# show the names of the tables in the database
for table in tables:
    print(table[0])
cursor.close()
```

```
albums
sqlite_sequence
artists
customers
employees
genres
invoices
invoice_items
media_types
playlists
playlist_track
tracks
sqlite_stat1
```

In [108…

```
query = "SELECT * FROM albums" # SQL query to select all data from the employees table
d = pd.read_sql_query(query, conn) # read the data into a pandas DataFrame
d.head(10) # show the first 10 rows of the DataFrame
cursor.close()
```

In [ ]: