```
# Importar librerias
import pandas as pd
import datetime

#instalar libreria para conectar a mi MySQL
!pip install mysql-connector-python
import mysql.connector

#ngrok tcp 3306
```

```
Collecting mysql-connector-python
  Downloading mysql_connector_python-8.4.0-cp310-cp310-manylinux_2_17_x86_64.whl (19.4 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 19.4/19.4 MB 46.6 MB/s eta 0:00:00
Installing collected packages: mysql-connector-python
Successfully installed mysql-connector-python-8.4.0
```

```
# Configura los detalles de conexión se requiere instalar ngrok
## Carga de datos desde la bdd MySQL database: personas
config = {
    'user': 'root',
    'password': 'root',
    'host': '0.tcp.sa.ngrok.io',
    'port':'17952',
    'database': 'personas',
    'raise_on_warnings': True
}

# Conectar a la base de datos
cnx = mysql.connector.connect(**config)
cursor = cnx.cursor()

# Realizar una consulta SQL
query = "SELECT * FROM personas.datospersona;"
cursor.execute(query)

# Cargar los datos en un DataFrame
df_partebdd= pd.DataFrame(cursor.fetchall(), columns=[i[0] for i in cursor.description])

# Cerrar la conexión
cursor.close()
cnx.close()

# Mostrar las primeras filas del DataFrame_parteBDD
print(df_partebdd.head())
```

```
       cedula fechaingresoEmpresa        titulo                 empresa  \
0  0100048585          1974-09-14      Doctor/a  Desarrollos Innovadores
1  0100670486          2006-07-11  Licenciado/a       Sistemas Avanzados
2  0101091243          2018-01-16      Doctor/a       Grupo TecnolÃ³gico
3  0102460347          2013-07-28      Doctor/a       Grupo TecnolÃ³gico
4  0106887865          2005-10-12  Arquitecto/a       Grupo TecnolÃ³gico

        tarjetaCredito
0  9015-2383-6543-6701
1  8261-5925-7650-1097
2  2174-3945-6858-8905
3  8024-3820-7982-5983
4  5291-8933-2270-8450
```

```
## Carga de datos
!pip install Faker

from faker import Faker
import random
from datetime import datetime

fake = Faker()

# Lista de cédulas proporcionadas en el df_partebdd
cedulas = df_partebdd['cedula']

# Asegurarse de que la lista tiene 1000 elementos
#cedulas = cedulas * (1000 // len(cedulas) + 1)
#cedulas = cedulas[:1000]

# Generar los demás datos
```

Cuaderno2_Modificado2.ipynb

Raw source

```
1
2
3
4

1
2

1

1  de ejemplo
2
3 e(),'Dirección':fake.city(),'Corr

1
```

```
data = {
    "cedula": cedulas,
    "nombre": [fake.first_name() for _ in range(1000)],
    "apellido": [fake.last_name() for _ in range(1000)],
    "direccion": [fake.address() for _ in range(1000)],
    "correo_electronico": [fake.email() for _ in range(1000)],
    "fecha_nacimiento": [fake.date_between(start_date=datetime(1970, 1, 1), end_date=datetime(1990, 1
}

df_partegenerada = pd.DataFrame(data)

# Mostrar las primeras filas del DataFrame para verificar
print(df_partegenerada.head())
```

```
Requirement already satisfied: Faker in /usr/local/lib/python3.10/dist-packages (25.2.0)
Requirement already satisfied: python-dateutil>=2.4 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python
       cedula     nombre apellido                             direccion  \
0  0100048585  Stephanie   Nelson          Unit 1821 Box 1359\nDPO AE 30606
1  0100670486      Brian     King  85592 Tracy Ranch\nNorth Mason, ID 06369
2  0101091243      Laura   Becker   191 Randy Lodge\nSimmonsshire, NJ 99725
3  0102460347      Paula      Lee  347 Andrew Forges\nJimenezbury, ME 46176
4  0106887865    Carolyn     Diaz  525 Tina Oval\nNorth Grantland, TX 32414

    correo_electronico fecha_nacimiento
0  rachel32@example.net       1975-07-27
1  rachel87@example.com       1972-01-04
2   nancy24@example.com       1978-04-20
3   bobby43@example.org       1970-06-20
4   lhughes@example.net       1971-01-13
```

```
# unir los dos dataframes el uno obtenido de MySQL y el otro generado con Facker
df_final = pd.merge(df_partebdd, df_partegenerada, on='cedula', how='inner')
print("\nInner Join:")
df_final.head()
numeroregistros= len(df_final)
print(numeroregistros)
```

```

Inner Join:
1000
```

```
# Verificar tipos de datos
def verificar_tipos(df_final):
    return df_final.dtypes
print(verificar_tipos(df_final))
```

```
cedula                 object
fechaingresoEmpresa    object
titulo                 object
empresa                object
tarjetaCredito         object
nombre                 object
apellido               object
direccion              object
correo_electronico     object
fecha_nacimiento       object
dtype: object
```

```
# Convertir la columna 'fecha_nacimiento' y 'fechaingresoEmpresa' de objeto a datetime

df_final['fecha_nacimiento'] = pd.to_datetime(df_final['fecha_nacimiento'])
df_final.head()

df_final['fechaingresoEmpresa'] = pd.to_datetime(df_final['fechaingresoEmpresa'])
df_final.head()

print(verificar_tipos(df_final))
```

```
cedula                         object
fechaingresoEmpresa    datetime64[ns]
titulo                         object
empresa                        object
```

```
tarjetaCredito                object
nombre                        object
apellido                      object
direccion                     object
correo_electronico            object
fecha_nacimiento      datetime64[ns]
dtype: object
```

```
#Agregar una columna al datasetfinal  la columna Edad
from datetime import date
# definir una funcion que calcula la edad sabiendo la fecha actual y la fecha de nacimiento
def calcular_edad(nacimiento):
    hoy = date.today()
    return hoy.year - nacimiento.year - ((hoy.month, hoy.day) < (nacimiento.month, nacimiento.day))

# Aplicar la función para crear la columna de edad
df_final['edad'] = df_final['fecha_nacimiento'].apply(lambda x: calcular_edad(x))
df_final.head()
```

	cedula	fechaingresoEmpresa	titulo
0	0100048585	1974-09-14	Doctor/a
1	0100670486	2006-07-11	Licenciado/a
2	0101091243	2018-01-16	Doctor/a
3	0102460347	2013-07-28	Doctor/a
4	0106887865	2005-10-12	Arquitecto/a

Next steps: Generate code with df_final View recommended plots

```
# crear un dicionario que especifique la provincia por la cedula
codigo_provincia = {
    '01': 'Azuay',
    '02': 'Bolivar',
    '03': 'Cañar',
    '04': 'Carchi',
    '05': 'Cotopaxi',
    '06': 'Chimborazo',
    '07': 'El Oro',
    '08': 'Esmeraldas',
    '09': 'Guayas',
    '10': 'Imbabura',
    '11': 'Loja',
    '12': 'Los Ríos',
    '13': 'Manabí',
    '14': 'Morona Santiago',
    '15': 'Napo',
    '16': 'Pastaza',
    '17': 'Pichincha',
    '18': 'Tungurahua',
    '19': 'Zamora Chinchipe',
    '20': 'Galápagos',
    '21': 'Sucumbíos',
    '22': 'Orellana',
    '23': 'Santo Domingo de los Tsáchilas',
    '24': 'Santa Elena'
}

#definir funcion que especifique a que provincia pertenece
def asignar_provincia(cedula):
    # Convertir a string y verificar si los primeros dos caracteres están en el diccionario
    cedula_str = str(cedula)
    codigo = cedula_str[:2]
    return codigo_provincia.get(codigo, 'Fuera del país')
```

```
# Aplicar la función para crear la columna 'provinciaNacimiento'
df_final['provinciaNacimiento'] = df_final['cedula'].apply(asignar_provincia)

# Mostrar el DataFrame resultante
print(df_final)
```

```
         cedula fechaingresoEmpresa       titulo                 empresa  \
0    0100048585          1974-09-14     Doctor/a  Desarrollos Innovadores
1    0100670486          2006-07-11  Licenciado/a       Sistemas Avanzados
2    0101091243          2018-01-16     Doctor/a        Grupo TecnolÃ³gico
3    0102460347          2013-07-28     Doctor/a        Grupo TecnolÃ³gico
4    0106887865          2005-10-12  Arquitecto/a       Grupo TecnolÃ³gico
..          ...                 ...          ...                      ...
995  3089791321          1974-12-20  Licenciado/a  Desarrollos Innovadores
996  3091817693          2002-02-15   Ingeniero/a       Sistemas Avanzados
997  3092409616          1992-01-08     Doctor/a  Desarrollos Innovadores
998  3094147622          2007-09-17  Licenciado/a       Grupo TecnolÃ³gico
999  3098080965          1988-05-23  Licenciado/a    Soluciones Integrales

          tarjetaCredito     nombre apellido  \
0    9015-2383-6543-6701  Stephanie   Nelson
1    8261-5925-7650-1097      Brian     King
2    2174-3945-6858-8905      Laura   Becker
3    8024-3820-7982-5983      Paula      Lee
4    5291-8933-2270-8450    Carolyn     Diaz
..                   ...        ...      ...
995  9813-8550-2171-0282      Julie    Brown
996  8229-3610-1445-9690     Daniel    Burke
997  2423-2910-7213-9761    Jeffrey   Castro
998  9973-1764-1929-5402    Charles  Harrell
999  5000-7893-7803-6988      Mason   Zavala

                                             direccion  \
0                    Unit 1821 Box 1359\nDPO AE 30606
1            85592 Tracy Ranch\nNorth Mason, ID 06369
2             191 Randy Lodge\nSimmonsshire, NJ 99725
3            347 Andrew Forges\nJimenezbury, ME 46176
4            525 Tina Oval\nNorth Grantland, TX 32414
..                                                ...
995  984 Davis Corners Suite 019\nNorth Michael, NY...
996  969 Rodriguez Tunnel Apt. 093\nJaimetown, AL 1...
997  7587 Eric Plains Suite 025\nNorth Theodoreshir...
998  5487 Thompson Prairie Apt. 957\nBreannaland, P...
999  57723 Abigail Trace Apt. 673\nLake Ryan, FL 72439

        correo_electronico fecha_nacimiento  edad provinciaNacimiento
0        rachel32@example.net       1975-07-27    48               Azuay
1        rachel87@example.com       1972-01-04    52               Azuay
2         nancy24@example.com       1978-04-20    46               Azuay
3         bobby43@example.org       1970-06-20    53               Azuay
4         lhughes@example.net       1971-01-13    53               Azuay
..                        ...              ...   ...                 ...
995      laurie55@example.com       1982-08-12    41      Fuera del país
996  reyesterrence@example.com      1982-03-25    42      Fuera del país
997     melissa67@example.net       1974-01-02    50      Fuera del país
998         dbond@example.com       1983-01-30    41      Fuera del país
999  collinssteven@example.org      1982-08-06    41      Fuera del país

[1000 rows x 12 columns]
```

```
# Definicion de variable querealiza el conteo de personas en cada  provincia
conteo_provincias = df_final['provinciaNacimiento'].value_counts()
# Mostrar el conteo para verificar
print(conteo_provincias)

# Definicion de variable que realiza el conteo de personas por edad
conteo_porEdad = df_final['edad'].value_counts()
# Mostrar el conteo para verificar
print(conteo_porEdad)
```

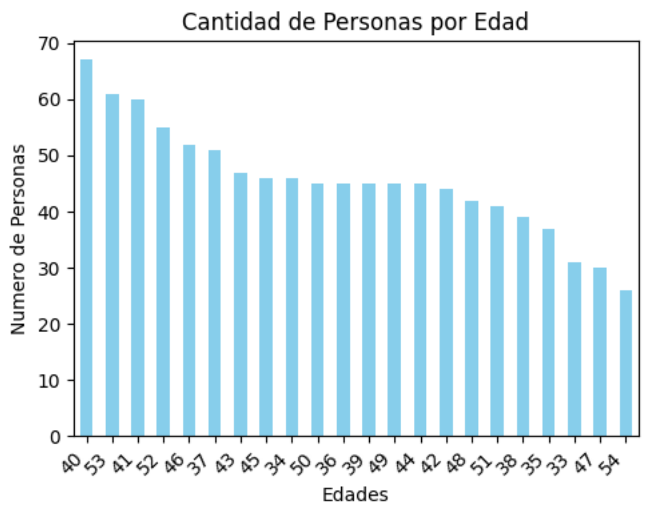
```
provinciaNacimiento
Manabí                           48
Bolivar                          47
Fuera del país                   47
Pichincha                        46
Azuay                            45
Chimborazo                       45
Morona Santiago                  43
El Oro                           43
Tungurahua                       43
Sucumbíos                        42
```

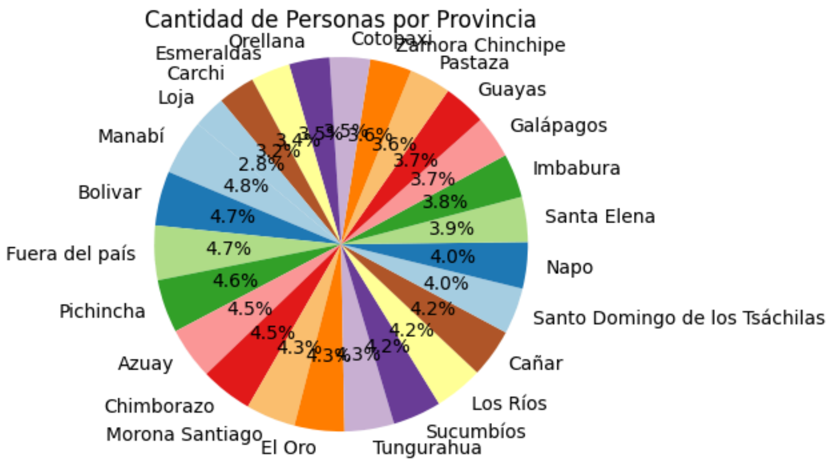
```
Los Ríos                          42
Cañar                             42
Santo Domingo de los Tsáchilas    40
Napo                              40
Santa Elena                       39
Imbabura                          38
Galápagos                         37
Guayas                            37
Pastaza                           36
Zamora Chinchipe                  36
Cotopaxi                          35
Orellana                          35
Esmeraldas                        34
Carchi                            32
Loja                              28
Name: count, dtype: int64
edad
40    67
53    61
41    60
52    55
46    52
37    51
43    47
45    46
34    46
50    45
36    45
39    45
49    45
44    45
42    44
48    42
51    41
38    39
35    37
33    31
47    30
54    26
Name: count, dtype: int64
```

```
#instalara libreria matplotlib para realizar las dos visualizaciones con esta libreria
!pip install matplotlib
import matplotlib.pyplot as plt
```

```
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (fro
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from ma
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages (from mat
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from m
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (fro
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python
```

```
# Crear gráfico UNO de barras en la libreria matplotlib
plt.figure(figsize=(5, 4))  # Configura el tamaño del gráfico
conteo_porEdad.plot(kind='bar', color='skyblue')  # Gráfico de barras
plt.title('Cantidad de Personas por Edad')  # Título del gráfico
plt.xlabel('Edades')  # Etiqueta del eje X
plt.ylabel('Numero de Personas')  # Etiqueta del eje Y
plt.xticks(rotation=45, ha='right')  # Rotar las etiquetas del eje X para mejor lectura
plt.tight_layout()  # Ajusta automáticamente los parámetros del subplot para que el subplot se ajuste

# Mostrar el gráfico
plt.show()

# Crear gráfico UNO de PASTEL en la libreria matplotlib
labels = conteo_provincias.index  # Las etiquetas son los nombres de las provincias
sizes = conteo_provincias.values  # Los tamaños son el conteo de cédulas en cada provincia

# Crear el gráfico de pastel
plt.figure(figsize=(5, 4))  # Configurar el tamaño de la figura
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors)  # Crea
plt.title('Cantidad de Personas por Provincia')  # Título del gráfico
plt.axis('equal')  # Esto asegura que el pastel se dibuje como un círculo.

# Mostrar el gráfico
# Mostrar el grafico
plt.show()
```



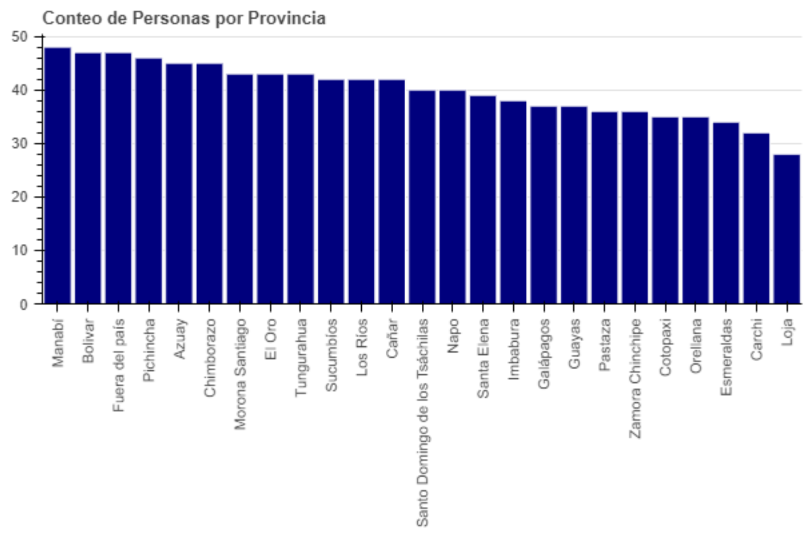


```
#instalara libreria bokeh para realizar las dos visualizaciones con esta libreria
!pip install bokeh
```

```
Requirement already satisfied: bokeh in /usr/local/lib/python3.10/dist-packages (3.3.4)
Requirement already satisfied: Jinja2>=2.9 in /usr/local/lib/python3.10/dist-packages (from bok
Requirement already satisfied: contourpy>=1 in /usr/local/lib/python3.10/dist-packages (from bo
Requirement already satisfied: numpy>=1.16 in /usr/local/lib/python3.10/dist-packages (from bok
Requirement already satisfied: packaging>=16.8 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages (from bok
Requirement already satisfied: pillow>=7.1.0 in /usr/local/lib/python3.10/dist-packages (from b
Requirement already satisfied: PyYAML>=3.10 in /usr/local/lib/python3.10/dist-packages (from bo
Requirement already satisfied: tornado>=5.1 in /usr/local/lib/python3.10/dist-packages (from bo
Requirement already satisfied: xyzservices>=2021.09.1 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pa
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python
```

```
# Crear grafico mediante bokeh
# Convertir a DataFrame para facilitar el manejo con Bokeh de # de personas en cada Povincia
conteo_df = conteo_provincias.reset_index()
conteo_df.columns = ['Provincia', 'Conteo']

from bokeh.plotting import figure, show, output_notebook
from bokeh.models import ColumnDataSource
from bokeh.palettes import Category20  # Importar una paleta de colores adecuada


output_notebook()  # Para mostrar el gráfico dentro de Colab
print (conteo_df)
# Crear ColumnDataSource
# Crear ColumnDataSource
source = ColumnDataSource(data=conteo_df)

# Crear la figura
p = figure(x_range=conteo_df['Provincia'], height=400, title="Conteo de Personas por Provincia",
           toolbar_location=None, tools="")

# Agregar las barras verticales
p.vbar(x='Provincia', top='Conteo', width=0.9, source=source,
       line_color='white', fill_color='navy')

# Personalizar el gráfico
p.xgrid.grid_line_color = None
p.y_range.start = 0
p.xaxis.major_label_orientation = 1.57  # Rotar las etiquetas para mejor visibilidad
p.outline_line_color = None

# Mostrar el gráfico
show(p)
```

```
                     Provincia  Conteo
0                       Manabí      48
1                      Bolivar      47
2               Fuera del país      47
3                    Pichincha      46
4                        Azuay      45
5                   Chimborazo      45
6              Morona Santiago      43
7                       El Oro      43
8                   Tungurahua      43
9                    Sucumbíos      42
10                    Los Ríos      42
11                       Cañar      42
12  Santo Domingo de los Tsáchilas      40
13                        Napo      40
14                 Santa Elena      39
15                    Imbabura      38
16                   Galápagos      37
17                      Guayas      37
18                     Pastaza      36
19            Zamora Chinchipe      36
20                    Cotopaxi      35
21                    Orellana      35
22                  Esmeraldas      34
23                      Carchi      32
24                        Loja      28
```



```
#definiendo un dataframe que solo tenga los campos Edades y el total de personas en cada edad
conteo_df = conteo_porEdad.reset_index()
conteo_df.columns = ['Edades', 'Total']
print(conteo_df)
print(len(conteo_df))
```

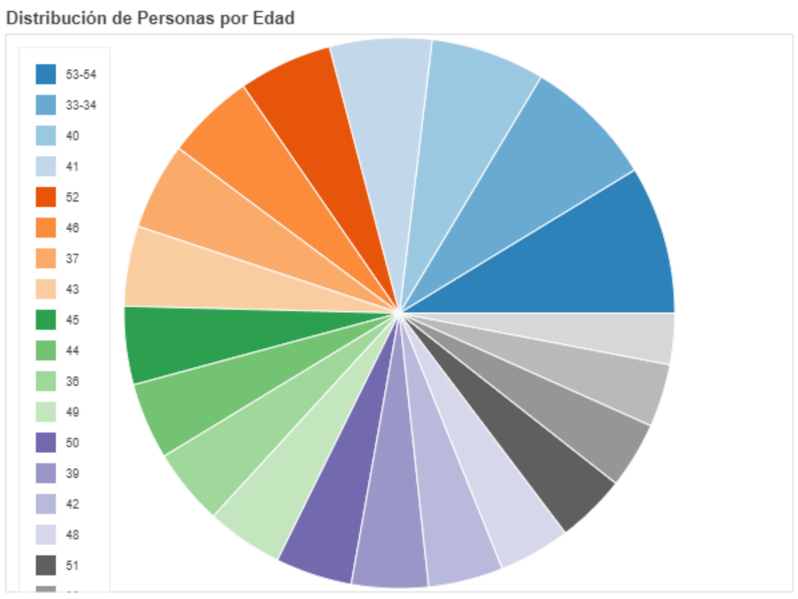
```
    Edades  Total
0       40     67
1       53     61
2       41     60
3       52     55
4       46     52
```

```
     5      37     51
     6      43     47
     7      45     46
     8      34     46
     9      50     45
     10     36     45
     11     39     45
     12     49     45
     13     44     45
     14     42     44
     15     48     42
     16     51     41
     17     38     39
     18     35     37
     19     33     31
     20     47     30
     21     54     26
     22
```

```
# juntar registros comunes para solo tener un dataframe de 20 registros
# Ordenar el DataFrame por 'Edades'
conteo_df.sort_values('Edades', inplace=True)

# Combinar los dos últimos registros
new_row = pd.DataFrame({
    'Edades': [f"{conteo_df.iloc[-2, 0]}-{conteo_df.iloc[-1, 0]}"],
    'Total': [conteo_df.iloc[-2, 1] + conteo_df.iloc[-1, 1]]
})

# Eliminar los dos últimos registros y añadir el nuevo usando pd.concat
conteo_df = pd.concat([conteo_df.iloc[:-2], new_row]).reset_index(drop=True)

# Ejemplo de combinar otro par de registros para alcanzar 20 registros total
new_row_2 = pd.DataFrame({
    'Edades': [f"{conteo_df.iloc[0, 0]}-{conteo_df.iloc[1, 0]}"],
    'Total': [conteo_df.iloc[0, 1] + conteo_df.iloc[1, 1]]
})

conteo_df = pd.concat([conteo_df.iloc[2:], new_row_2]).sort_values('Total', ascending=False).reset_

print(conteo_df)
```

```
       Edades  Total
    0   53-54     87
    1   33-34     77
    2      40     67
    3      41     60
    4      52     55
    5      46     52
    6      37     51
    7      43     47
    8      45     46
    9      44     45
    10     36     45
    11     49     45
    12     50     45
    13     39     45
    14     42     44
    15     48     42
    16     51     41
    17     38     39
    18     35     37
    19     47     30
```

```
from math import pi
from bokeh.io import show, output_notebook
from bokeh.plotting import figure
from bokeh.transform import cumsum
from bokeh.palettes import Category20c  # Esta paleta tiene 20 colores, necesitaremos más colores
from bokeh.models import ColumnDataSource

output_notebook()
df = pd.DataFrame(conteo_df)

# Asegurarte de que las edades están en string si son necesarias como etiquetas
df['Edades'] = df['Edades'].astype(str)

# Calcular los ángulos y porcentajes para el gráfico de pastel
df['angle'] = df['Total']/df['Total'].sum() * 2*pi
df['color'] = Category20c[20] * (len(df) // 20 )  # Repetir la paleta para tener suficientes colore
df['percentage'] = df['Total']/df['Total'].sum() * 100

source = ColumnDataSource(df)
p = figure(height=450, title="Distribución de Personas por Edad", toolbar_location=None,
           tools="hover", tooltips="@Edades: @Total (@percentage{0.2f}%)", x_range=(-1, 1))

p.wedge(x=0, y=1, radius=0.7,
        start_angle=cumsum('angle', include_zero=True), end_angle=cumsum('angle'),
        line_color="white", fill_color='color', legend_field='Edades', source=source)

p.axis.axis_label = None
p.axis.visible = False
p.grid.grid_line_color = None

# Mover la leyenda fuera del gráfico
p.legend.location = "top_left"
p.legend.label_text_font_size = "9px"

show(p)
```

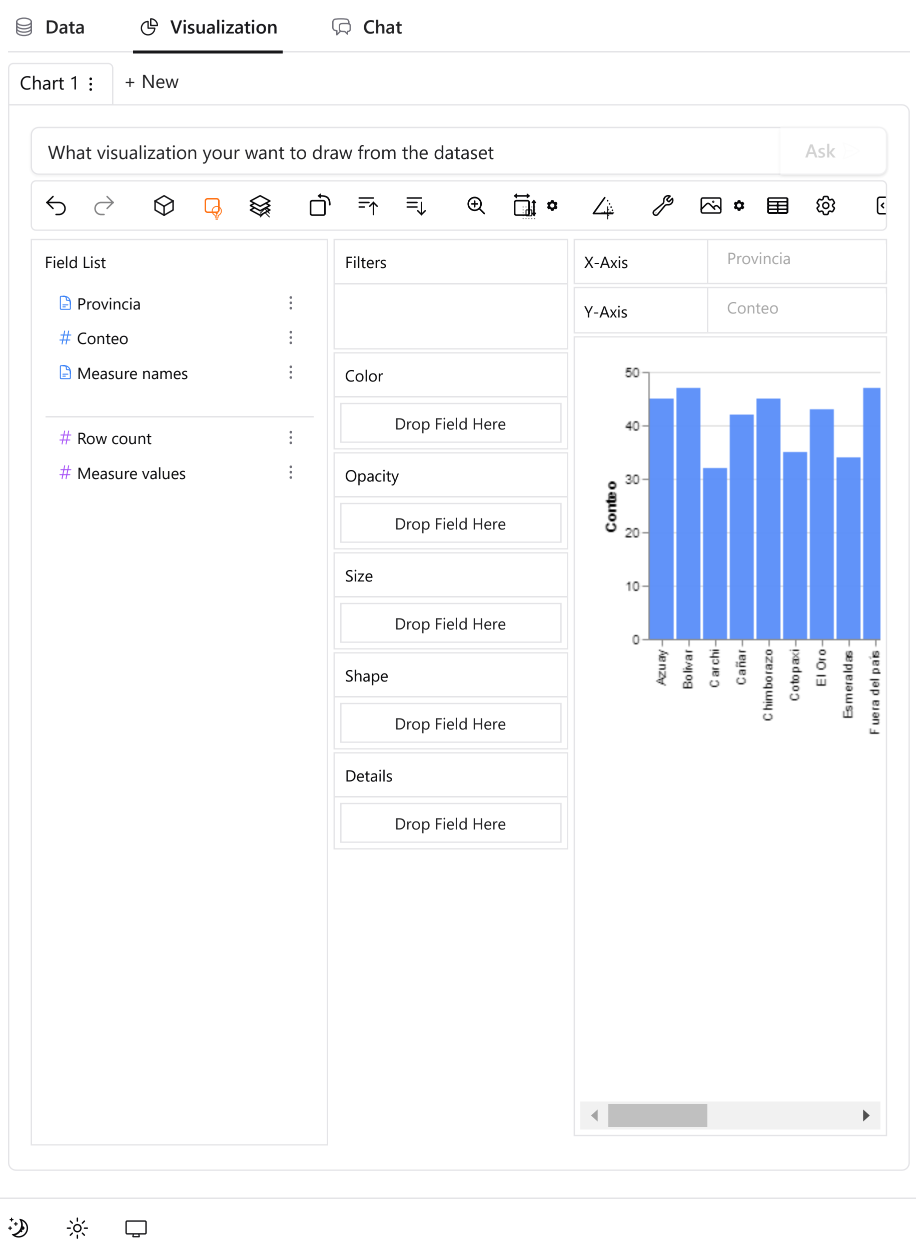


```
#instalar la libreria pygwalker
!pip install pygwalker
```

```
    Requirement already satisfied: tornado>=4.2 in /usr/local/lib/python3.10/dist-packages (from
    Requirement already satisfied: parso<0.9.0,>=0.8.3 in /usr/local/lib/python3.10/dist-package
    Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.10/dist-packages (f
    Requirement already satisfied: wcwidth in /usr/local/lib/python3.10/dist-packages (from prom
    Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from pyt
    Requirement already satisfied: notebook>=4.4.1 in /usr/local/lib/python3.10/dist-packages (f
    Requirement already satisfied: pyzmq<25,>=17 in /usr/local/lib/python3.10/dist-packages (fro
    Requirement already satisfied: argon2-cffi in /usr/local/lib/python3.10/dist-packages (from
    Requirement already satisfied: jupyter-core>=4.6.1 in /usr/local/lib/python3.10/dist-package
    Requirement already satisfied: nbformat in /usr/local/lib/python3.10/dist-packages (from not
    Requirement already satisfied: nbconvert>=5 in /usr/local/lib/python3.10/dist-packages (from
    Requirement already satisfied: nest-asyncio>=1.5 in /usr/local/lib/python3.10/dist-packages
    Requirement already satisfied: Send2Trash>=1.8.0 in /usr/local/lib/python3.10/dist-packages
    Requirement already satisfied: terminado>=0.8.3 in /usr/local/lib/python3.10/dist-packages (
    Requirement already satisfied: prometheus-client in /usr/local/lib/python3.10/dist-packages
    Requirement already satisfied: nbclassic>=0.4.7 in /usr/local/lib/python3.10/dist-packages (
    Requirement already satisfied: platformdirs>=2.5 in /usr/local/lib/python3.10/dist-packages
    Requirement already satisfied: jupyter-server>=1.8 in /usr/local/lib/python3.10/dist-package
    Requirement already satisfied: notebook-shim>=0.2.3 in /usr/local/lib/python3.10/dist-packag
    Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-packages (from nbconve
    Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (fr
    Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages (from nbcon
    Requirement already satisfied: defusedxml in /usr/local/lib/python3.10/dist-packages (from n
    Requirement already satisfied: entrypoints>=0.2.2 in /usr/local/lib/python3.10/dist-packages
    Requirement already satisfied: jupyterlab-pygments in /usr/local/lib/python3.10/dist-package
    Requirement already satisfied: mistune<2,>=0.8.1 in /usr/local/lib/python3.10/dist-packages
    Requirement already satisfied: nbclient>=0.5.0 in /usr/local/lib/python3.10/dist-packages (f
    Requirement already satisfied: pandocfilters>=1.4.1 in /usr/local/lib/python3.10/dist-packag
    Requirement already satisfied: tinycss2 in /usr/local/lib/python3.10/dist-packages (from nbc
    Requirement already satisfied: fastjsonschema>=2.15 in /usr/local/lib/python3.10/dist-packag
    Requirement already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.10/dist-packages (f
    Requirement already satisfied: argon2-cffi-bindings in /usr/local/lib/python3.10/dist-packag
    Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/dist-packages (fro
    Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python
    Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.10/dist-package
    Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.10/dist-packages (fr
    Requirement already satisfied: anyio<4,>=3.1.0 in /usr/local/lib/python3.10/dist-packages (f
    Requirement already satisfied: websocket-client in /usr/local/lib/python3.10/dist-packages (
    Requirement already satisfied: cffi>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from
    Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (fro
    Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from
    Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.10/dist-packages (from
    Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-packages (fr
    Requirement already satisfied: pycparser in /usr/local/lib/python3.10/dist-packages (from cf
```

```python
#definiendo un dataframe que solo tenga los campos Provincia y el total de personas en cada provincia
conteo_df = conteo_provincias.reset_index()
conteo_df.columns = ['Provincia', 'Conteo']
```

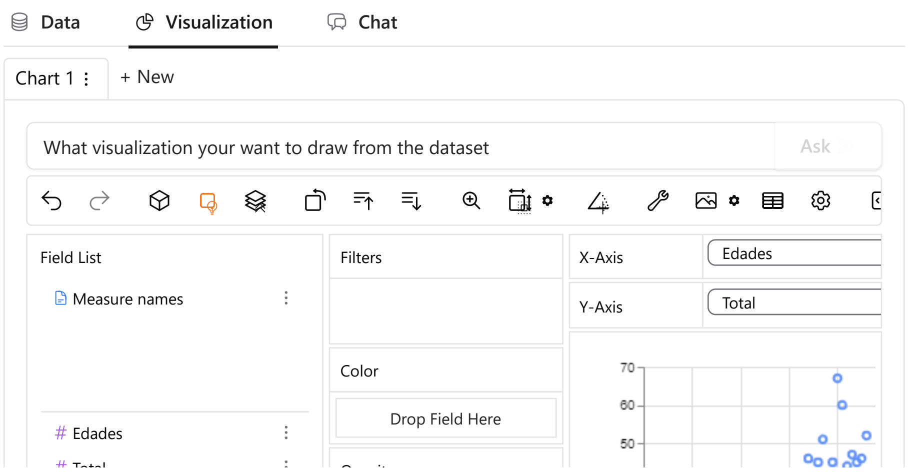
```python
#importando la libreria y llamando al objeto visual
import pygwalker as pyg
pyg.walk(conteo_df)
```



```
<pygwalker.api.pygwalker.PygWalker at 0x7a5243e41ea0>
```

```
#definiendo un dataframe que solo tenga los campos Edades y el total de personas en cada edad
conteo_df = conteo_porEdad.reset_index()
conteo_df.columns = ['Edades', 'Total']
print(conteo_df)
pyg.walk(conteo_df)
```

```
    Edades  Total
0       40     67
1       53     61
2       41     60
3       52     55
4       46     52
5       37     51
6       43     47
7       45     46
8       34     46
9       50     45
10      36     45
11      39     45
12      49     45
13      44     45
14      42     44
15      48     42
16      51     41
17      38     39
18      35     37
19      33     31
20      47     30
21      54     26
```



X-Axis: Edades
Y-Axis: Total