



Universidad Tecnológica de Puebla.

División: Tecnologías de la Información.

Área: Desarrollo y Gestión de Software .

*Materia: Extracción de conocimiento en Bases
de Datos.*

REMEDIAL DEL PRODUCTO 1

Docente: Espinosa Garita José Francisco.

Alumno: Cristian Hernández Bonilla.

Matricula: UTP0154256

Grupo: 9no "C"

Cuatrimestre: Mayo - Agosto.

ÍNDICE.

Tabla de contenido

ÍNDICE.....	1
INTRODUCCIÓN.....	2
JUSTIFICACIÓN DEL PROYECTO.....	2
i. Objetivo	2
ii. Alcance	2
iii. Limitaciones	2
iv. Justificación de los datos.....	3
LIMPIEZA DE DATOS	4
A) Eliminar variables innecesarias.....	4
B) Eliminar observaciones innecesarias	4
C) Cambiar variables numéricas vacías por el valor numérico 0	5
D) Cambiar variables numéricas que contienen texto por 0	5
ESTADÍSTICA DESCRIPTIVA	6
A) Carga de datos con CSV.....	6
B) Carga de datos con sistema gestor de base de datos.....	6
C) Estadística Descriptiva.....	8
CONCLUSIONES.....	20

INTRODUCCIÓN.

La finalidad de este producto es mostrar lo aprendido a lo largo de la unidad 1 de la materia de Extracción de conocimiento de Bases de Datos, en este producto se mostrarán, mediante el uso del lenguaje de programación Python y sus distintas librerías, como leer datos, convertirlos a dataframes, limpiarlos y analizarlos, para posteriormente graficarlos.

Mediante el uso de la base de datos del censo económico correspondiente a la entidad que le fue asignado al alumno, se mostrará el desarrollo de lo dicho anteriormente, poniendo en práctica lo aprendido, y preparándose a su vez para los contenidos posteriores de la materia.

JUSTIFICACIÓN DEL PROYECTO.

i. Objetivo

Analizar datos de niveles escolares entre población femenina y masculina del censo económico 2020 para el estado de Nuevo León mediante las herramientas brindadas por el lenguaje Python y realizará lo que se solicita para el desarrollo del análisis de la información.

ii. Alcance

El presente proyecto contempla estadística descriptiva en base al censo económico del estado de Nuevo León, y en el se deben de cumplir los puntos presentados con antelación a través de las indicaciones del proyecto.

iii. Limitaciones

- * Se trabajará con niveles de escolaridad femenina y masculina.
- * Se trabajará con información del censo económico 2020.
- * Se utilizará información perteneciente al estado de Nuevo León.

iv. Justificación de los datos.

Se eliminaron los datos de “Clave de entidad” y “Nombre de entidad” debido a que no son necesarios para el análisis de los datos, puesto que el contenido entero del censo se refiere desde un inicio a dicha entidad, esto sería necesario únicamente en caso de estar tratando con datos de distintas entidades federativas.

A su vez, se eliminaron los registros cuya mayoría de valores estaban compuestos por los valores 0 o NULL, así como otros valores que no representaban ningún dato, como es el caso de las cadenas “N/A” y “N/D”.

Además, en relación a los datos a utilizar para la estadística descriptiva, se tomarán aquellos pertenecientes a la población femenina y masculina, así como sus niveles educativos, esto con la intención de comparar el grado de educación entre ambos géneros.

LIMPIEZA DE DATOS

A) Eliminar variables innecesarias.

Para la limpieza de variables innecesarias se eliminaron las dos primeras variables del dataset, “Clave de entidad” y “Nombre de la entidad”. Para lograr esto, durante el proceso de análisis y eliminación de observaciones innecesarias, se uso el método pop() el cual elimina el indice asignado de un arreglo, con el cual se eliminaron dichas variables de todas las observaciones.

```
with open(path + 'NLeon.csv', newline='') as File:
    reader = csv.reader(File)
    for row in reader:
        if "Total" not in row[5] and not obtenerRegistrosErroneos(row):
            row.pop(0)
            row.pop(0)
```

B) Eliminar observaciones innecesarias

Para eliminar las observaciones innecesarias se realizo un análisis, en este caso, durante el proceso de lectura de los datos, se pasaba cada fila a una función que analizaba los registros, y si al menos 260 eran datos inservibles, nulos o 0, se eliminaba dicha fila.

```
def obtenerRegistrosErroneos(arr):
    contadorColumnasMalas = 0
    for col in arr:
        if col == '*':
            contadorColumnasMalas +=1
        if col == '0':
            contadorColumnasMalas +=1
        if col == "N/A":
            contadorColumnasMalas +=1
        if col == "N/D":
            contadorColumnasMalas +=1
    return contadorColumnasMalas >= 210
```


C) Cambiar variables numéricas vacías por el valor numérico 0

Posterior al análisis, las filas con menos de 210 registros inservibles veían cambiados sus valores no numéricos, correspondientes a variables con valores numéricos, por 0's, asegurando poder trabajar con dichos datos sin problemas.

```
with open(path + 'NLeon.csv', newline='') as File:
    reader = csv.reader(File)
    for row in reader:
        if "Total" not in row[5] and not obtenerRegistrosErroneos(row):
            row.pop(0)
            row.pop(0)
            for i in range(len(row)):
                if row[i] == '*':
                    row[i] = 0
```

D) Cambiar variables numéricas que contienen texto por 0

A su vez, las variables numéricas con valores de texto también vieron dichos valores reemplazados por 0's-

```
if row[i] == "N/A":
    row[i] = 0
if row[i] == "N/D":
    row[i] = 0
```

Posteriormente se guardaron dichos datos en un nuevo archivo csv, el cuál sera usado para el análisis de estadística descriptiva.

```
▶ csvFiltered = open(path + 'CleanNLeonIter.csv', 'w')
with csvFiltered:
    writer = csv.writer(csvFiltered)
    writer.writerows(csvList)
```

ESTADÍSTICA DESCRIPTIVA

A) Carga de datos con CSV.

Para la carga de datos con CSV se hará uso de una función pre-integrada de pandas llamada `read_csv`, la cuál leerá el csv de la dirección que le demos como parámetro y lo transformará en un DataFrame de pandas, el cual usaremos para el análisis de datos.

```
[3] censoNL = pd.read_csv(path + 'CleanNLeonIter.csv')
```

B) Carga de datos con sistema gestor de base de datos.

Para comenzar cargamos el csv limpio en una tabla en nuestra base de datos, puedes usar la herramienta que más se te facilite, en mi caso fue el wizard de MySQL WorkBench, podemos ver que se cargo, al hacer un select a la tabla, esto devolverá los datos que buscamos.

MUN	NOM_MUN	LOC	NOM_LOC	LONGITUD	LATITUD	ALTITUD	POBTOT	POBFEM	POBMAS	P_OA2	P_OA2_F	P_OA2_J
1	Abasolo	1	Abasolo	100°23'59.958" W	25°56'43.215" N	502	1992	1032	960	108	54	54
1	Abasolo	42	Los Diez (La Tripona)	100°25'20.884" W	25°57'15.078" N	511	15	7	8	1	1	0
1	Abasolo	46	Colinas del Fraile	100°25'42.903" W	25°57'11.580" N	545	53	27	26	0	0	0
1	Abasolo	47	Alberto Villarreal	100°24'44.756" W	25°56'41.723" N	507	870	448	422	55	33	22
2	Aguaqueguas	1	Aguaqueguas	99°32'21.801" W	26°18'39.293" N	176	2080	1073	1007	81	44	37
2	Aguaqueguas	9	Tres Hermanos	99°50'01.936" W	26°19'59.295" N	289	25	11	14	0	0	0
2	Aguaqueguas	12	Cieneguitas	99°52'30.250" W	26°20'21.962" N	299	53	32	21	1	1	0
2	Aguaqueguas	17	La Escondida	99°45'43.164" W	26°15'33.852" N	300	149	75	74	4	2	2
2	Aguaqueguas	21	Los Garza	99°46'55.999" W	26°23'02.120" N	231	206	98	108	2	2	0
2	Aguaqueguas	27	Lagunillas (San José Lagunillas)	99°43'02.005" W	26°17'00.081" N	263	61	25	36	1	0	1

Una vez que nuestra tabla esta cargada en MySQL, ahora la convertiremos a un dataframe. Para esto haremos uso de la librería `mysql connector`, la cual debemos instalar mediante `pip` o `conda`, una vez instalada e importada en nuestra Jupyter Notebook, definiremos la variable de conexión, a la cuál asignaremos el host, el nombre de la base de datos, nombre de usuario y contraseña, y ejecutaremos la función `connect` para conectarnos a la base de datos.

```
import pandas as pd
import mysql.connector

connection = mysql.connector.connect(
    host='localhost',
    database='censonleon',
    user='root',
    password=''
)
```

Una vez hecho esto, procederemos a buscar nuestra tabla, definiremos el parámetro de diccionario del cursor como verdadero para que los resultados de las queries sean diccionarios, y solicitaremos los datos de la tabla, con fetchall, los asignaremos a una variable y esta la convertiremos en un DataFrame.

```
if connection.is_connected():

    tabla = 'nleoniter'
    cursor = connection.cursor(dictionary=True)
    cursor.execute(f"SELECT * FROM {tabla}")
    registros = cursor.fetchall()
    df = pd.DataFrame(registros)
    print(f"Tabla '{tabla}' leída correctamente. DataFrame creado:")
    print(df.head())
```

Una vez cargado nuestro DataFrame usaremos la función head para mostrar los primeros valores y verificar que se cargo correctamente.

Tabla 'nleoniter' leída correctamente. DataFrame creado:

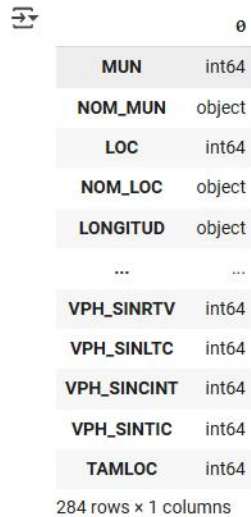
	MUN	NOM_MUN	LOC	NOM_LOC	LONGITUD	\
0	1	Abasolo	1	Abasolo	100°23'59.958" W	
1	1	Abasolo	42	Los Diez (La Tripona)	100°25'20.884" W	
2	1	Abasolo	46	Colinas del Fraile	100°25'42.903" W	
3	1	Abasolo	47	Alberto Villarreal	100°24'44.756" W	
4	2	Aguaaleguas	1	Aguaaleguas	99°32'21.801" W	

	LATITUD	ALTITUD	POBTOT	POBFEM	POBMAS	...	VPH_CEL	VPH_INTER	\
0	25°56'43.215" N	502	1992	1032	960	...	501	259	
1	25°57'15.078" N	511	15	7	8	...	3	0	
2	25°57'11.580" N	545	53	27	26	...	14	4	
3	25°56'41.723" N	507	870	448	422	...	218	84	
4	26°18'39.293" N	176	2080	1073	1007	...	662	378	

C) Estadística Descriptiva.

- Listar los campos y tipos de datos del dataframe.

Para esto se hará uso de la función `dtype`, la cual muestra los campos y su tipo de dato.

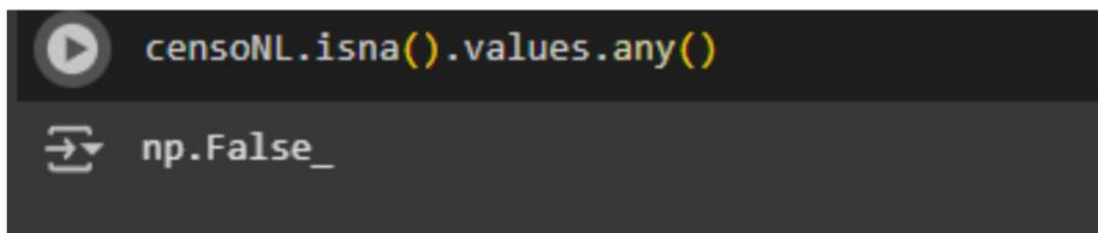


	0
MUN	int64
NOM_MUN	object
LOC	int64
NOM_LOC	object
LONGITUD	object
...	...
VPH_SINRTV	int64
VPH_SINLTC	int64
VPH_SINCINT	int64
VPH_SINTIC	int64
TAMLOC	int64

284 rows x 1 columns

- Verificar si hay datos faltantes.

Para esto usaremos la función `isna`, la cual devolverá `True` en caso de que la variable contenga un valor nulo, y si agregamos su propiedad `values`, junto con la función `any`, nos mostrará `True` si cualquiera de las variables contiene un valor nulo, o `False` si no se encuentra ninguno.

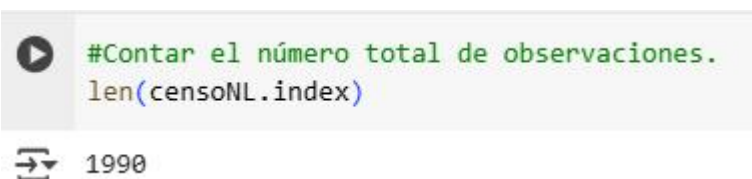


```
censoNL.isna().values.any()

np.False_
```

- Contar el número total de observaciones.

Para esto haremos uso de la propiedad `index`, la cual nos devolverá el total de observaciones registradas en el dataframe.



```
#Contar el número total de observaciones.
len(censoNL.index)

1990
```

- **Obtener el summary (resumen) del dataframe.**

Para esto haremos uso de la función describe, la cuál nos dará un resumen de las variables del dataframe, incluyendo información como cantidad de observaciones, media, desviación estándar, mínimos, máximos y cuartiles.

#Obtener el summary (resumen) del dataframe.
censoNL.describe()

	MUN	LOC	ALTITUD	POBTOT	POBFEM	POBMAS	P_0A2	P_0A2_F	P_0A2_M	P_3YMAS	...	VPH
count	1990.000000	1990.000000	1990.000000	1.990000e+03	1990.000000	1990.000000	1990.000000	1990.000000	1990.000000	1.990000e+03	...	1990.00
mean	24.406030	210.078392	840.358794	2.901014e+03	1451.953266	1449.060302	129.557286	64.240201	65.317085	2.762446e+03	...	771.27
std	13.712041	222.530006	681.436101	3.662315e+04	18440.705413	18183.971937	1461.843021	721.172542	740.751075	3.502501e+04	...	9635.13
min	1.000000	1.000000	79.000000	3.000000e+00	0.000000	2.000000	0.000000	0.000000	0.000000	3.000000e+00	...	0.00
25%	14.000000	46.000000	336.000000	1.400000e+01	6.000000	8.000000	0.000000	0.000000	0.000000	1.300000e+01	...	4.00
50%	22.000000	133.000000	460.500000	4.000000e+01	20.000000	21.000000	2.000000	1.000000	1.000000	3.900000e+01	...	9.00
75%	38.000000	296.750000	1543.750000	1.347500e+02	65.000000	69.750000	6.000000	3.000000	3.000000	1.287500e+02	...	32.00
max	51.000000	1179.000000	2710.000000	1.142952e+06	578172.000000	564780.000000	41910.000000	20520.000000	21390.000000	1.092650e+06	...	298526.00

8 rows x 280 columns

- **¿Identificar cuales son los nombres de las variables?**

Para esto haremos uso de las propiedades columns y values, lo que nos devolverá los valores del header de cada columna o variable, es decir sus nombres.

```
#Identificar cuales son los nombres de las variables.
censoNL.columns.values

array(['MUN', 'NOM_MUN', 'LOC', 'NOM_LOC', 'LONGITUD', 'LATITUD',
      'ALTITUD', 'POBTOT', 'POBFEM', 'POBMAS', 'P_0A2', 'P_0A2_F',
      'P_0A2_M', 'P_3YMAS', 'P_3YMAS_F', 'P_3YMAS_M', 'P_5YMAS',
      'P_5YMAS_F', 'P_5YMAS_M', 'P_12YMAS', 'P_12YMAS_F', 'P_12YMAS_M',
      'P_15YMAS', 'P_15YMAS_F', 'P_15YMAS_M', 'P_18YMAS', 'P_18YMAS_F',
      'P_18YMAS_M', 'P_3A5', 'P_3A5_F', 'P_3A5_M', 'P_6A11', 'P_6A11_F',
      'P_6A11_M', 'P_8A14', 'P_8A14_F', 'P_8A14_M', 'P_12A14',
      'P_12A14_F', 'P_12A14_M', 'P_15A17', 'P_15A17_F', 'P_15A17_M',
      'P_18A24', 'P_18A24_F', 'P_18A24_M', 'P_15A49', 'P_60YMAS',
      'P_60YMAS_F', 'P_60YMAS_M', 'REL_H_M', 'POB0_14', 'POB15_64',
      'POB65_MAS', 'P_0A4', 'P_0A4_F', 'P_0A4_M', 'P_5A9', 'P_5A9_F',
      'P_5A9_M', 'P_10A14', 'P_10A14_F', 'P_10A14_M', 'P_15A19',
      'P_15A19_F', 'P_15A19_M', 'P_20A24', 'P_20A24_F', 'P_20A24_M',
      'P_25A29', 'P_25A29_F', 'P_25A29_M', 'P_30A34', 'P_30A34_F',
      'P_30A34_M', 'P_35A39', 'P_35A39_F', 'P_35A39_M', 'P_40A44',
      'P_40A44_F', 'P_40A44_M', 'P_45A49', 'P_45A49_F', 'P_45A49_M',
      'P_50A54', 'P_50A54_F', 'P_50A54_M', 'P_55A59', 'P_55A59_F',
      'P_55A59_M', 'P_60A64', 'P_60A64_F', 'P_60A64_M', 'P_65A69',
      'P_65A69_F', 'P_65A69_M', 'P_70A74', 'P_70A74_F', 'P_70A74_M',
      'P_75A79', 'P_75A79_F', 'P_75A79_M', 'P_80A84', 'P_80A84_F',
      'P_80A84_M', 'P_85YMAS', 'P_85YMAS_F', 'P_85YMAS_M', 'PROM_HNV',
      'PNACENT', 'PNACENT_F', 'PNACENT_M', 'PNACOE', 'PNACOE_F',
      'PNACOE_M', 'PRES2015', 'PRES2015_F', 'PRES2015_M', 'PRESOE15',
      'PRESOE15_F', 'PRESOE15_M', 'P3YM_HLI', 'P3YM_HLI_F', 'P3YM_HLI_M',
      'P3HLINHE', 'P3HLINHE_F', 'P3HLINHE_M', 'P3HLI_HE', 'P3HLI_HE_F',
      'P3HLI_HE_M', 'P5_HLI', 'P5_HLI_NHE', 'P5_HLI_HE', 'PHOG_IND'],
      dtype=object)
```

- **¿Cuántos municipios hay?**

Para esto haremos uso de la función `len`, para obtener la cantidad de registros del array devuelto por la función `unique` `tolist` sobre la variable `MUN` que registra los municipios, esto nos dará la cantidad de municipios exacta, aún si estos se repiten.

```
len(censoNL['MUN'].unique().tolist())
```

51

- **¿Cuántas localidades hay?**

Se aplica la misma función que en el punto anterior, con la diferencia de que se utiliza sobre la variable `LOC`, para obtener la cantidad de localidades.

```
#¿Cuántas localidades hay?  
len(censoNL['NOM_LOC'].unique().tolist())
```

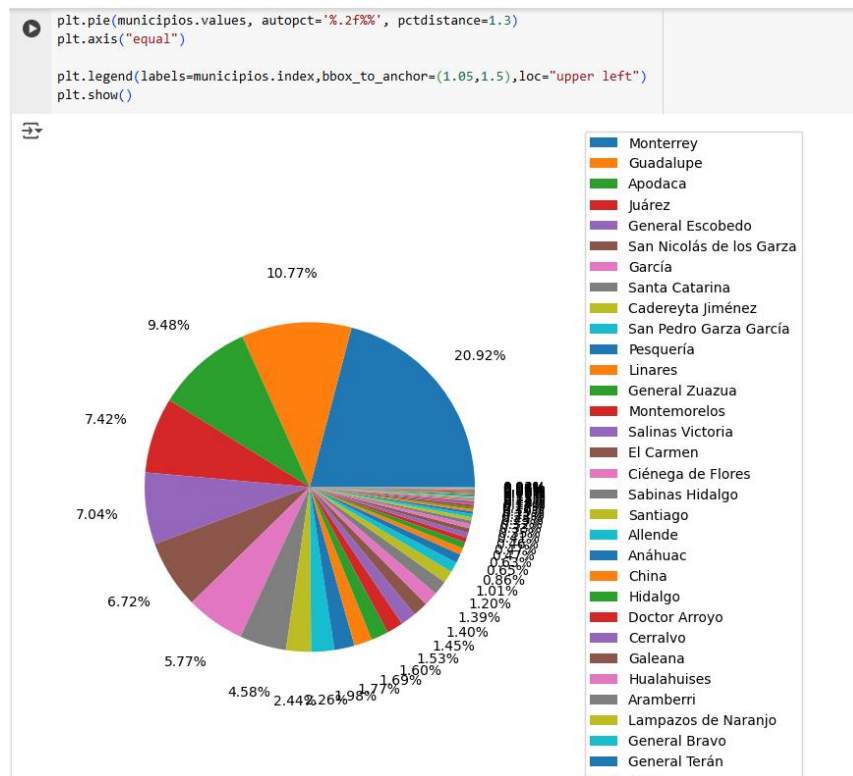
1597

- **¿Cuántas observaciones por municipio hay?**

Para esto hacemos uso de la función `value_counts`, la cuál contará la cantidad de valores y cuanto se repiten, de esta forma obtendremos la cantidad de observaciones por cada municipio, esto lo asignaremos a una nueva variable.

```
municipios = censoNL['NOM_MUN'].value_counts()
```

La cuál después pasaremos a la función `pie` de la librería `matplotlib` para generar un diagrama de pastel con dichos datos.

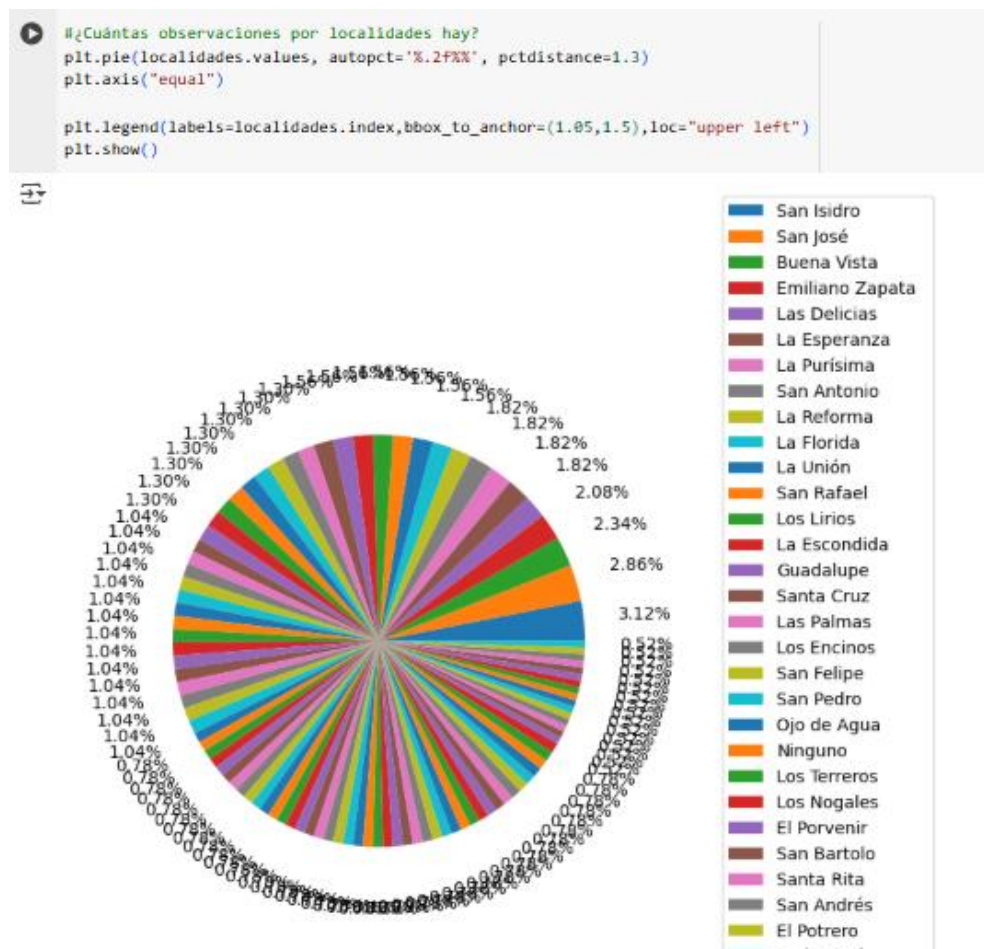


● ¿Cuántas observaciones por localidades hay?

Se hará uso de las mismas funciones que en el punto anterior, únicamente cambiará el nombre de la variable que en este caso será NOM_LOC, para obtener la cantidad de observaciones por localidad, así como el nombre de dicha localidad.

```
#¿Cuántas observaciones por localidades hay?
localidades = censoNL['NOM_LOC'].value_counts().iloc[:100]
```

Dicho resultado se utilizará para generar la gráfica de pastel, debido a que la cantidad es muy grande, pues se cuenta con 1597 localidades, se mostrará la información referente a las primeras 100 con mayor cantidad de observaciones..



- **5 totales de las variables que desee.**

Para esto usaremos la función `sum`, la cuál sumará los valores de cada observación, de esta forma obtendremos los totales.

1. Población total de mujeres, obtenida al sumar todas las observaciones de la variable de población femenina.

```

print(f"Poblacion Total de Mujeres: {censoNL['POBFEM'].sum()}")

```

Poblacion Total de Mujeres: 2774996

2. Población total de hombres, obtenida al sumar todas las observaciones de la variable de población masculina.

```

print(f"Poblacion Total de Hombres: {censoNL['POBMAS'].sum()}")

```

Poblacion Total de Hombres: 2762298

3. Población total de personas de entre 12 y 14 años de edad que no asisten a la escuela.

```
print(f"Poblacion Total de Personas entre 12 y 14 años de edad que no asisten a la escuela: {censoNL['P12A14NOA'].sum()}")
```

Poblacion Total de Personas entre 12 y 14 años de edad que no asisten a la escuela: 18122

4. Población total personas de entre 8 y 14 años de edad que no saben leer ni escribir.

```
print(f"Poblacion Total de 8 a 14 años que no sabe leer ni escribir: {censoNL['P12A14NOA'].sum()}")
```

Poblacion Total de 8 a 14 años que no sabe leer ni escribir: 18122

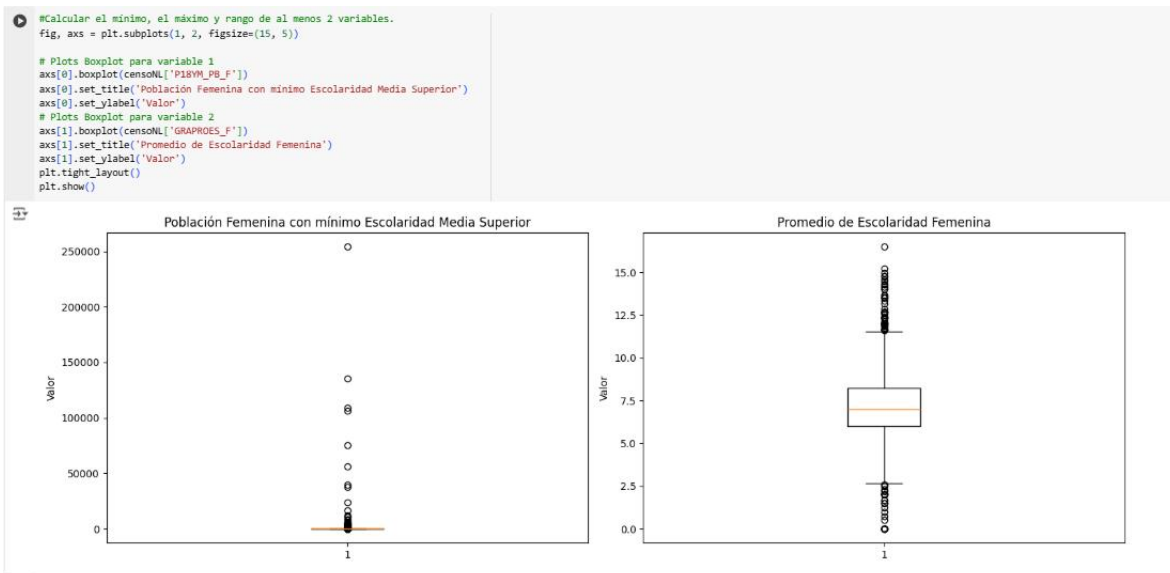
5. Población total de 15 años o más que no sabe leer ni escribir

```
print(f"Poblacion Total de 15 o más años que no sabe leer ni escribir: {censoNL['P15YM_AN'].sum()}")
```

Poblacion Total de 15 o más años que no sabe leer ni escribir: 63368

- **Calcular el mínimo, el máximo y rango de al menos 2 variables.**

Para esto haremos uso de la función boxplot de matplotlib, a la cuál le pasaremos las variables que queremos calcular y graficar, en este caso el grado promedio de femenino y la población femenina que cuenta con una escolaridad mínima de educación Media Superior, esto con el fin de comparar los valores de ambas variables.



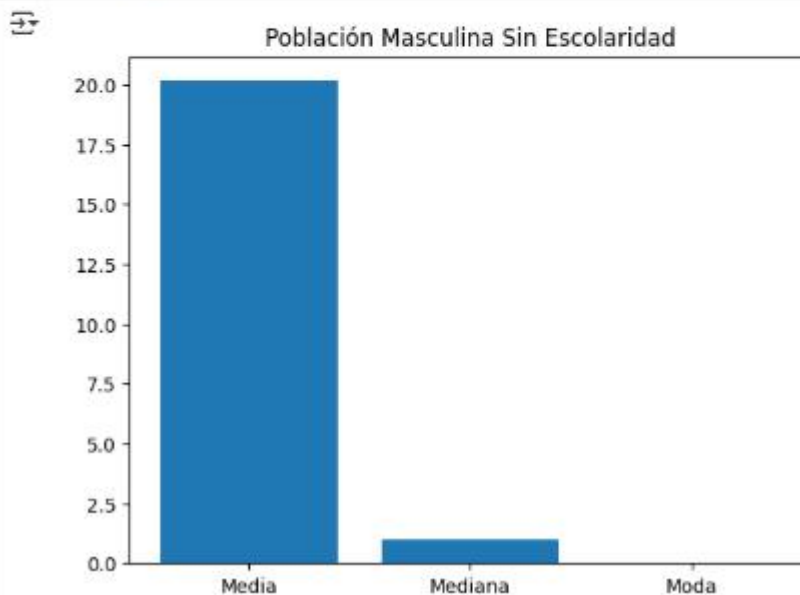
- **Calcular la media, la media y la moda de al menos 2 variables.**

Para esto haremos uso de las funciones mean, median y mode, las cuáles nos darán la media, mediana y moda respectivamente, de las variables en las que se realicen, datos los cuáles guardaremos en una lista que pasaremos a la función de matplotlib, bar, la cuál nos devolverá dichos datos en una gráfica de barras.

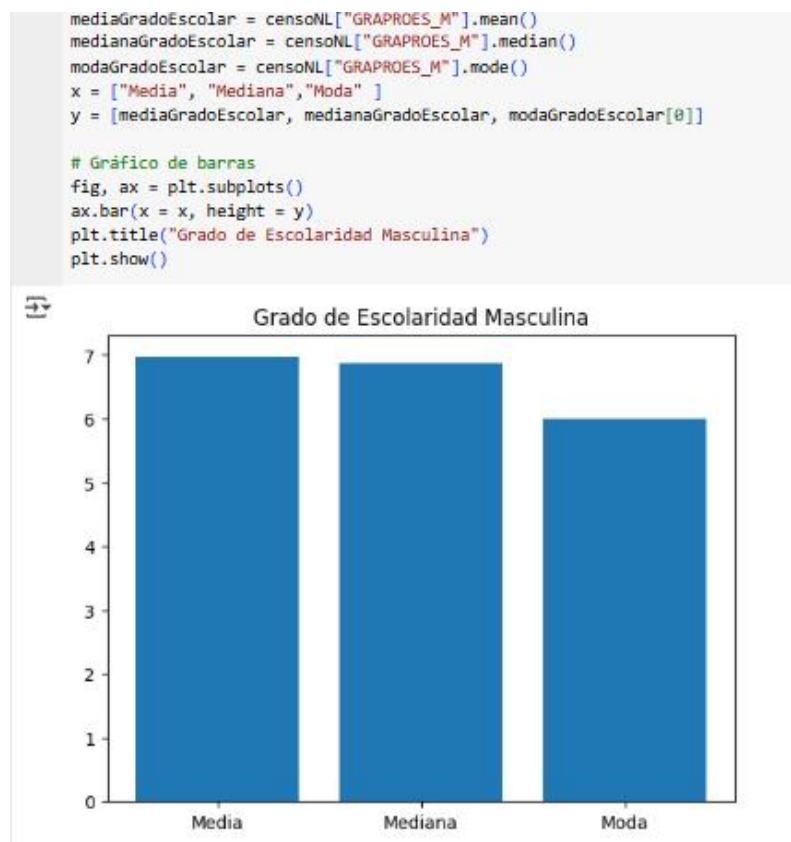
Primero con la variable P15YM_SE_M, la cuál nos dará la cantidad de habitantes masculinos mayores a 15 años, que no cuentan con algún grado de escolaridad.

```
#Calcular la media, la media y la moda de al menos 2 variables.
mediaEscMasculina = censoNL["P15YM_SE_M"].mean()
medianaEscMasculina = censoNL["P15YM_SE_M"].median()
modaEscMasculina = censoNL["P15YM_SE_M"].mode()
x = ["Media", "Mediana", "Moda" ]
y = [mediaEscMasculina, medianaEscMasculina, modaEscMasculina[0]]

# Gráfico de barras
fig, ax = plt.subplots()
ax.bar(x = x, height = y)
plt.title("Población Masculina Sin Escolaridad")
plt.show()
```



Posteriormente se realizará con la variable GRAPROES_M, la cuál nos dará el Grado de Escolaridad Masculina



- **Calcular la desviación estándar y la varianza de al menos 2 variables.**

Para realizar esto haremos uso de las funciones std y var, que nos devolverán la desviación estándar y la varianza respectivamente, y con dicha información haremos lo mismo que en el punto anterior para desarrollar una gráfica de barras con dichos datos.

Primero se realizará con la variable P12A14NOAF, la cuál devolverá la cantidad de mujeres de 12 a 14 años que no asiste a la escuela.

```
#Calcular la desviación estándar y la varianza de al menos 2 variables.
desviacionMujeresNoEscararizadas = censoNL["P12A14NOAF"].std()
varianzaMujeresNoEscararizadas= censoNL["P12A14NOAF"].var()
x = ["Desviación Estándar", "Varianza" ]
y = [desviacionMujeresNoEscararizadas, varianzaMujeresNoEscararizadas]

# Gráfico de barras
fig, ax = plt.subplots()
plt.title("Población femenina de 12 a 14 años que no asiste a la escuela")
ax.bar(x = x, height = y)
plt.show()
```



Y posteriormente se realizará con la variable P12A14NOAFM, la cuál devolverá la cantidad de hombres de 12 a 14 años que no asiste a la escuela.



- **Calcular los cuartiles (25%, 50%, 75%) para al menos 2 variables.**

Para esto pasaremos los datos a la función boxplot y haremos uso de la función txt del boxplot para asignar las etiquetas a los valores donde se encuentran los cuartiles, estos se obtendrán mediante la función de numpy, percentile, a la cuál debemos pasar la variable a calcular, así como el número de percentil que deseamos calcular, en este caso los percentiles 25, 50 y 75.

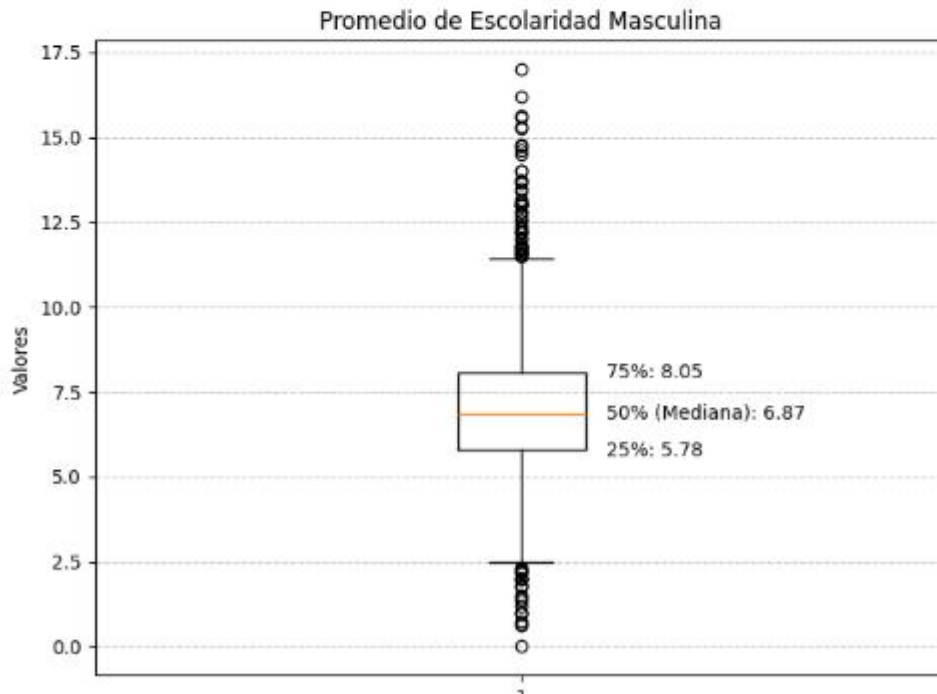
Primero se realizará con la variable GRAPROES_M, con la cuál visualizaremos la escolaridad promedio masculina.


```
#Calcular los cuantiles (25%, 50%, 75%) para al menos 2 variables.
plt.figure(figsize=(8, 6))
plt.boxplot(censoNL["GRAPROES_M"], vert=True)

plt.title("Promedio de Escolaridad Masculina")
plt.ylabel("Valores")
plt.grid(axis='y', linestyle='--', alpha=0.7)

quartiles = np.percentile(censoNL["GRAPROES_M"], [25, 50, 75])
plt.text(1.1, quartiles[0], f'25%: {quartiles[0]:.2f}', va='center')
plt.text(1.1, quartiles[1], f'50% (Mediana): {quartiles[1]:.2f}', va='center')
plt.text(1.1, quartiles[2], f'75%: {quartiles[2]:.2f}', va='center')

Text(1.1, 8.05, '75%: 8.05')
```



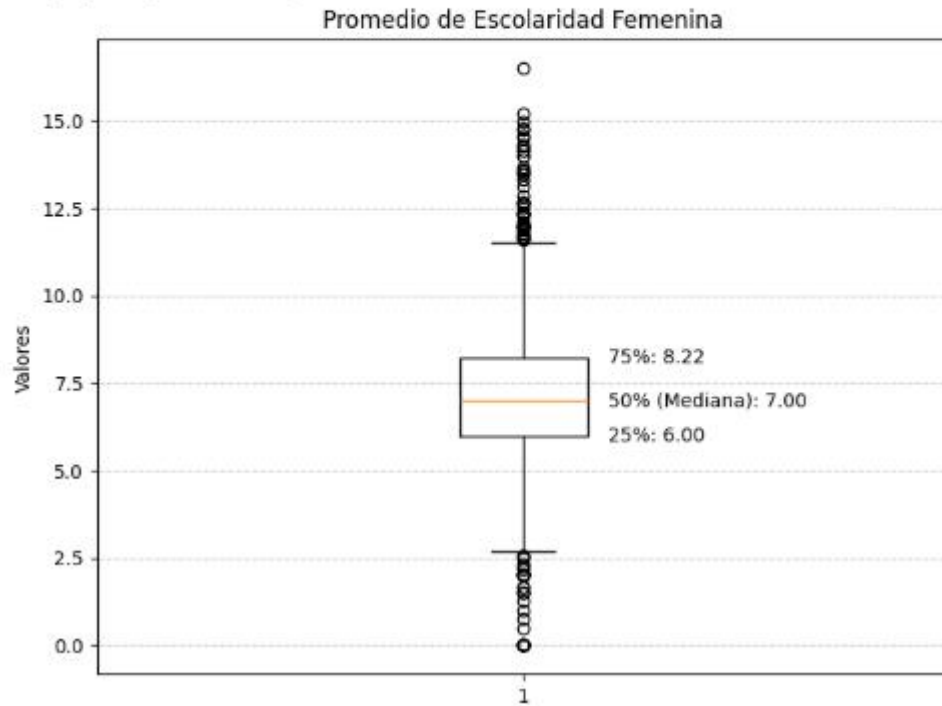
Posteriormente se realizará con la variable GRAPROES_F, con la cuál visualizaremos la escolaridad promedio femenina, esto con el fin de comparar el grado de escolaridad general entre hombres y mujeres en el estado de Nuevo León.

```
plt.figure(figsize=(8, 6))
plt.boxplot(censoNL["GRAPROES_F"], vert=True)

plt.title("Promedio de Escolaridad Femenina")
plt.ylabel("Valores")
plt.grid(axis='y', linestyle='--', alpha=0.7)

quartiles = np.percentile(censoNL["GRAPROES_F"], [25, 50, 75])
plt.text(1.1, quartiles[0], f'25%: {quartiles[0]:.2f}', va='center')
plt.text(1.1, quartiles[1], f'50% (Mediana): {quartiles[1]:.2f}', va='center')
plt.text(1.1, quartiles[2], f'75%: {quartiles[2]:.2f}', va='center')

Text(1.1, 8.22, '75%: 8.22')
```



CONCLUSIONES.

En este producto pudimos poner en práctica lo visto a través de la unidad, lo cuál fue el análisis de datos con las herramientas que encontramos en el lenguaje Python, gracias a este proyecto se pudo poner a prueba de forma extendida lo visto a lo largo de la unidad, así como hacer un análisis en grandes cantidades de datos, esto nos ayuda a reforzar el tema visto y nos puede ser de gran ayuda en nuestra formación laboral, pues el análisis de datos es un área con bastante demanda y bien remunerada.

En el caso de aquellos que deseen profundizar más en este tema, este producto será de gran ayuda para sentar las bases de su desarrollo, además el desarrollo de estos ejercicios será de gran ayuda para lo que se verá durante las siguientes unidades de la materia, en las que seguiremos analizando datos de una manera más profunda.

BIBLIOGRAFÍA

- *pandas.read_csv* — *pandas 2.3.0 documentation.* (s. f.). https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html#pandas.read_csv
- *GeeksforGeeks.* (2025, 17 marzo). *How to Create Boxplot from Pandas DataFrame?* *GeeksforGeeks.* <https://www.geeksforgeeks.org/python/how-to-create-boxplot-from-pandas-dataframe/>
- *Gráficos de barras.* (s. f.). <https://cursosinformatica.ucm.es/trial/dataviz/>
- *W3Schools- Ploting with Pandas and Matplotlib Python* (s. f.). https://www.w3schools.com/python/pandas/pandas_plotting.asp
- *6 formas de contar las filas del marco de datos de Pandas.* (s. f.). <https://python.19633.com/es/python-tag-2/Pandas-2/1002003676.html>