**Minería de Datos**

**Taller Básico 1**

Teniendo como base el archivo csv de crímenes de LA City, desarrollar el siguiente taller, versionar en repositorio del curso bajo el folder /20240909-taller

**1. Cargar el CSV**

Cargar el archivo CSV y revisar las primeras filas para asegurarnos de que se ha cargado correctamente.

import pandas as pd

# Cargar el archivo CSV

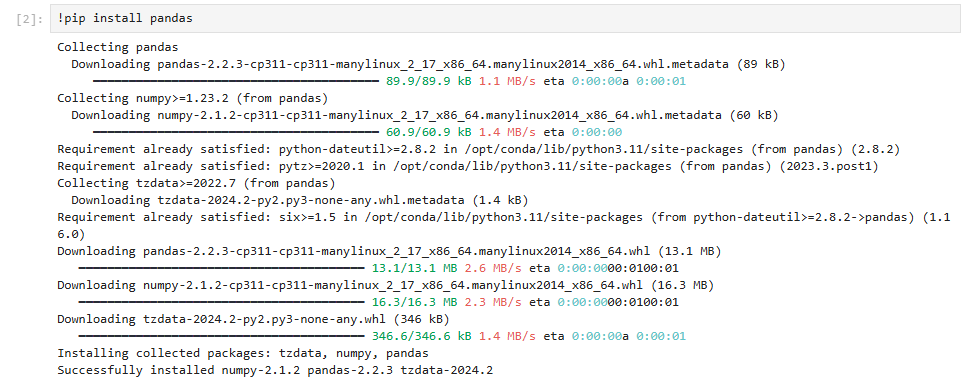
df = pd.read\_csv('lapd\_crime\_data.csv')

# Mostrar las primeras 5 filas del dataframe

print(df.head())

instalamos pandas

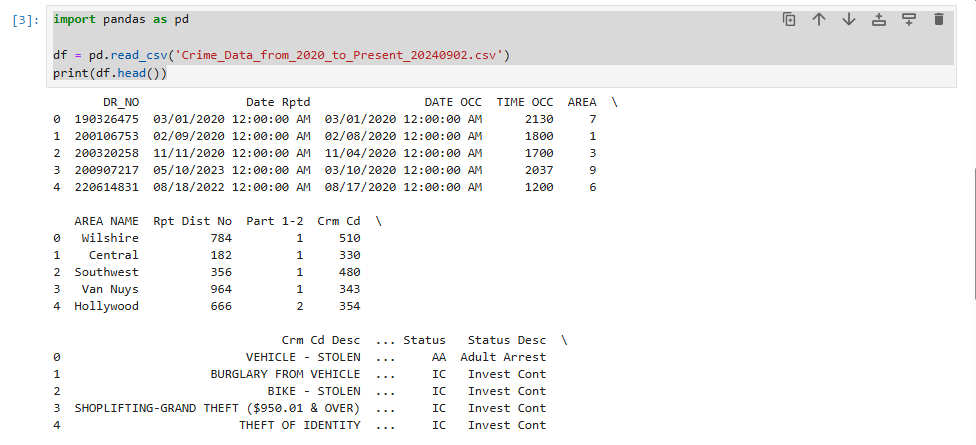
!pip install pandas



import pandas as pd

df = pd.read\_csv('Crime\_Data\_from\_2020\_to\_Present\_20240902.csv')

print(df.head())

 Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

**2. Filtrar los delitos más graves (Crm Cd 1)**

Queremos ver todos los incidentes en los que el delito primario (crm\_cd\_1) es el más grave, por ejemplo, el código 110, que representa “Homicidio criminal”.

Hacerlo para 5 codigos mas por parametro

# Filtrar los delitos con Crm Cd 1 igual a 110 (homicidio)

delitos\_graves = df[df['crm\_cd\_1'] == '110']

# Mostrar los primeros 5 registros de delitos graves

print(delitos\_graves.head())

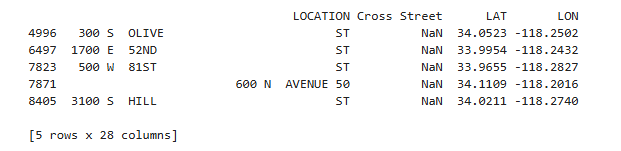
import pandas as pd

delitos\_graves = df[df['Crm Cd 1'] == 110]

print(delitos\_graves.head())

Tabla

Descripción generada automáticamente

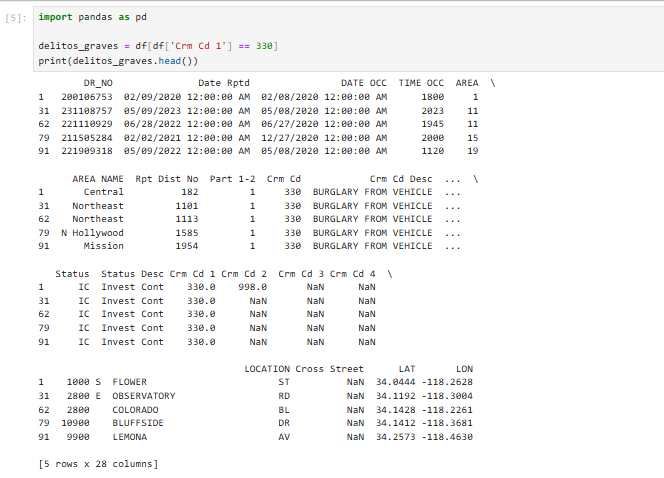


Tabla

Descripción generada automáticamente

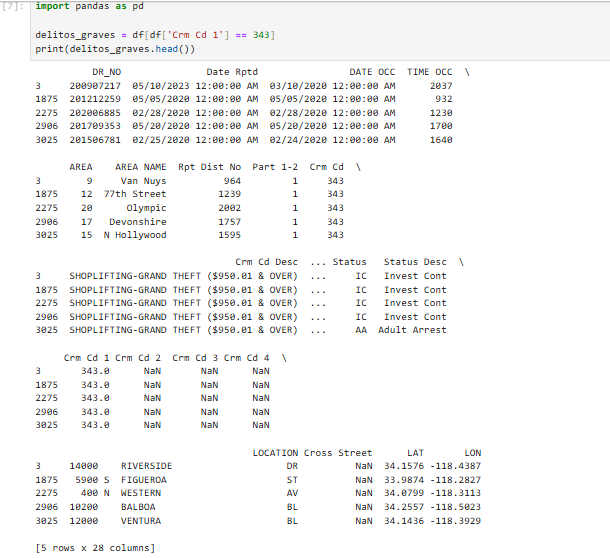
Interfaz de usuario gráfica

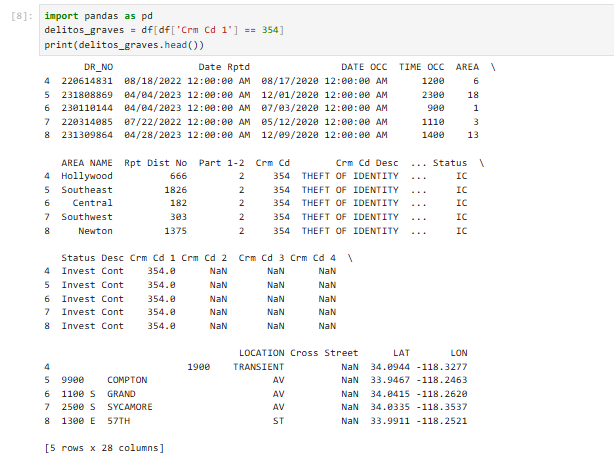
Descripción generada automáticamente



Tabla

Descripción generada automáticamente





**3. Contar el número de incidentes por área (AREA)**

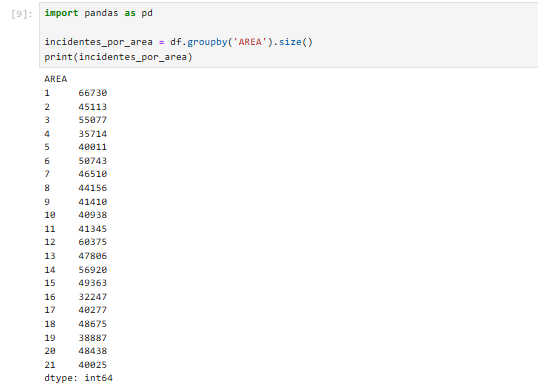
Agrupamos los datos por área (area) para ver cuántos incidentes se reportan en cada una de las áreas geográficas.

# Contar el número de incidentes por área

incidentes\_por\_area = df.groupby('area').size()

# Mostrar los resultados

print(incidentes\_por\_area)



**4. Análisis de delitos por género de la víctima (Vict Sex)**

Realizamos un análisis para ver cuántos incidentes involucran víctimas masculinas, femeninas o desconocidas.

# Contar los incidentes según el género de la víctima

delitos\_por\_genero = df['vict\_sex'].value\_counts()

# Mostrar los resultados

print(delitos\_por\_genero)

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**5. Visualización: Distribución de incidentes por área**

Vamos a crear un gráfico de barras para visualizar el número de incidentes en cada una de las áreas.

import matplotlib.pyplot as plt

# Agrupar por área y contar el número de incidentes

incidentes\_por\_area = df.groupby('area').size()

# Crear el gráfico de barras

incidentes\_por\_area.plot(kind='bar', figsize=(10, 6))

plt.title('Número de incidentes por área geográfica')

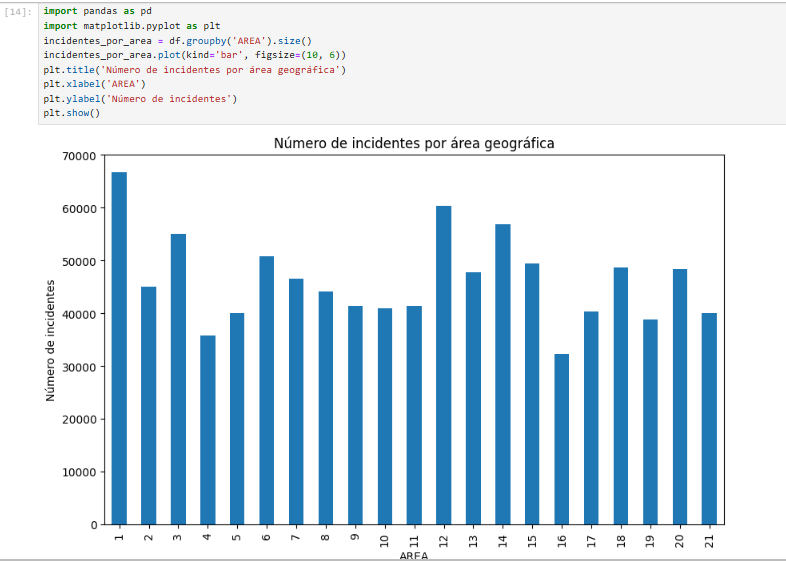
plt.xlabel('Área')

plt.ylabel('Número de incidentes')

plt.show()

primero debemos instalar matplotlib:

!pip install matplotlib



**6. Mapeo de ubicaciones con latitud y longitud**

Usando las columnas de latitud (lat) y longitud (lon), podríamos trazar en un mapa la ubicación de los incidentes. A continuación, te muestro cómo se puede hacer un análisis básico de los datos de ubicación (usando geopandas para visualizar el mapa).

Primero, instalamos geopandas:

pip install geopandas

Luego, cargamos los datos y los visualizamos en un mapa.

import geopandas as gpd

from shapely.geometry import Point

# Crear una geometría de puntos a partir de latitudes y longitudes

df['geometry'] = df.apply(lambda row: Point(row['lon'], row['lat']), axis=1)

# Convertir el dataframe de pandas en un GeoDataFrame

gdf = gpd.GeoDataFrame(df, geometry='geometry')

# Cargar el mapa base de Los Ángeles (puedes utilizar un shapefile o un mapa base desde GeoPandas)

world = gpd.read\_file(gpd.datasets.get\_path('naturalearth\_lowres'))

# Visualizar los puntos en el mapa

ax = world[world.name == 'United States'].plot(color='white', edgecolor='black')

gdf.plot(ax=ax, marker='o', color='red', markersize=5)

plt.show()

primero instalamos geopandas

!pip install geopandas

**7. Filtrar incidentes por rango de fechas**

Queremos filtrar los incidentes que ocurrieron en un rango específico de fechas, por ejemplo, entre el 1 de enero de 2023 y el 31 de marzo de 2023.

# Convertir las columnas de fechas a formato datetime

df['date\_occ'] = pd.to\_datetime(df['date\_occ'], format='%m/%d/%Y')

# Filtrar incidentes ocurridos entre el 01/01/2023 y el 31/03/2023

rango\_fechas = (df['date\_occ'] >= '2023-01-01') & (df['date\_occ'] <= '2023-03-31')

incidentes\_en\_rango = df[rango\_fechas]

# Mostrar los resultados

print(incidentes\_en\_rango.head())

**8. Frecuencia de delitos por arma utilizada (Weapon Used Cd)**

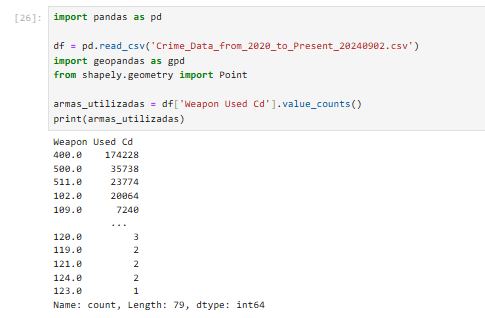
Queremos saber con qué frecuencia se utilizaron distintos tipos de armas en los incidentes reportados.

# Contar los incidentes según el código de arma utilizada

armas\_utilizadas = df['weapon\_used\_cd'].value\_counts()

# Mostrar los resultados

print(armas\_utilizadas)



**9. Identificación de los delitos más comunes (Crm Cd)**

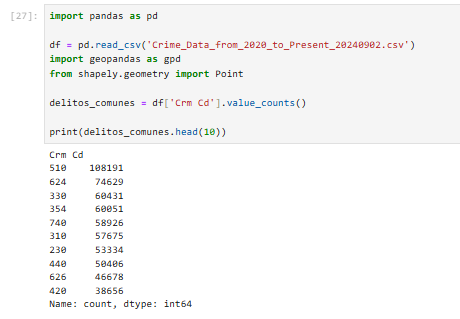
Queremos identificar los delitos más comunes reportados en la base de datos.

# Contar la frecuencia de cada tipo de delito según Crm Cd

delitos\_comunes = df['crm\_cd'].value\_counts()

# Mostrar los 10 delitos más comunes

print(delitos\_comunes.head(10))



**10. Análisis de edades de las víctimas (Vict Age)**

Analizamos la distribución de las edades de las víctimas.

# Convertir Vict Age a numérico, ignorando valores que no se pueden convertir

df['vict\_age'] = pd.to\_numeric(df['vict\_age'], errors='coerce')

# Visualizar la distribución de las edades con un histograma

df['vict\_age'].plot(kind='hist', bins=20, figsize=(10, 6))

plt.title('Distribución de edades de las víctimas')

plt.xlabel('Edad')

plt.ylabel('Frecuencia')

plt.show()

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Gráfico, Histograma

Descripción generada automáticamente

**Conclusión**

Estos ejemplos ilustran cómo puedes trabajar con la estructura de datos que proporcionaste utilizando herramientas como pandas, matplotlib, y geopandas para análisis y visualización.