

Análisis de Incidencia Delictiva en México

Profesión de Analista de Datos

Rafael Salinas _ 02-09-2025

Introducción: Contexto del Proyecto

- El objetivo principal de este proyecto es analizar los datos de incidencia delictiva en México para ayudar a una compañía de seguros de automóviles a desarrollar una estrategia de precios competitiva y rentable.
- La incidencia de delitos como el robo de vehículos, la extorsión y la privación ilegal de la libertad es un factor clave en la determinación del riesgo para las pólizas de seguro. Un análisis preciso permite a la empresa ajustar las primas en función de la peligrosidad de cada área geográfica, lo que se traduce en precios justos para los clientes y una mayor rentabilidad para la compañía.

Descarga y Creación de la Base de Datos

- El archivo CSV con el que se realizara el proyecto fue proporcionado desde la pagina oficial de la Secretariado Ejecutivo del Sistema Nacional de Seguridad Pública (SESNSP). Después de la descarga, se cargan los datos en una base de datos relacional para un manejo más eficiente en SQL Server Management Studio.
 - Liga de descarga: <https://www.gob.mx/sesnsp/acciones-y-programas/incidencia-delictiva-del-fuero-comun-nueva-metodologia>
 - **Descarga de los datos de incidencia delictiva del fuero común, *fecha de actualización 15 de agosto de 2025***
 - [Estatat 2015 - 2025](#)
- (descargados en Excel y tranformado en CSV para su manejo.)

Descarga y Creación de la Base de Datos

The screenshot shows a web browser window with the URL www.gob.mx/sesnsp/acciones-y-programas/incidencia-delictiva-del-fuero-comun-nueva-metodologia. The page header includes the Mexican coat of arms, the text "Gobierno de México", and navigation links for "Trámites", "Gobierno", "English", and a search icon. Below the header, there are links for "Protección de datos personales", "Transparencia", and "Datos Abiertos". The main content area features the title "Delitos por cada 100 mil habitantes (formato PDF) *fecha de actualización 15 de agosto de 2025*". Underneath, there are two buttons labeled "Aa+" and "Aa-". The text "Estatat 2015 - 2025" is displayed. A blue highlighted box contains the text "Descarga de los datos de incidencia delictiva del fuero común, *fecha de actualización 15 de agosto de 2025*". Below this, another blue highlighted box shows "Estatat 2015 - 2025". Further down, the text "Municipal 2015 - 2025" is visible. An important note states: "Importante: Leer nota (PDF) al interior del archivo comprimido para instrucciones sobre como abrir los archivos, ya que por el número de filas, la base completa no puede ser cargada en una hoja de cálculo (Excel o Calc)." The Windows taskbar at the bottom shows the search bar, task view, and various application icons, with the system clock indicating 12:52 p.m. on 01/09/2025.

Delitos por cada 100 mil habitantes (formato PDF) *fecha de actualización 15 de agosto de 2025*

Estatat 2015 - 2025

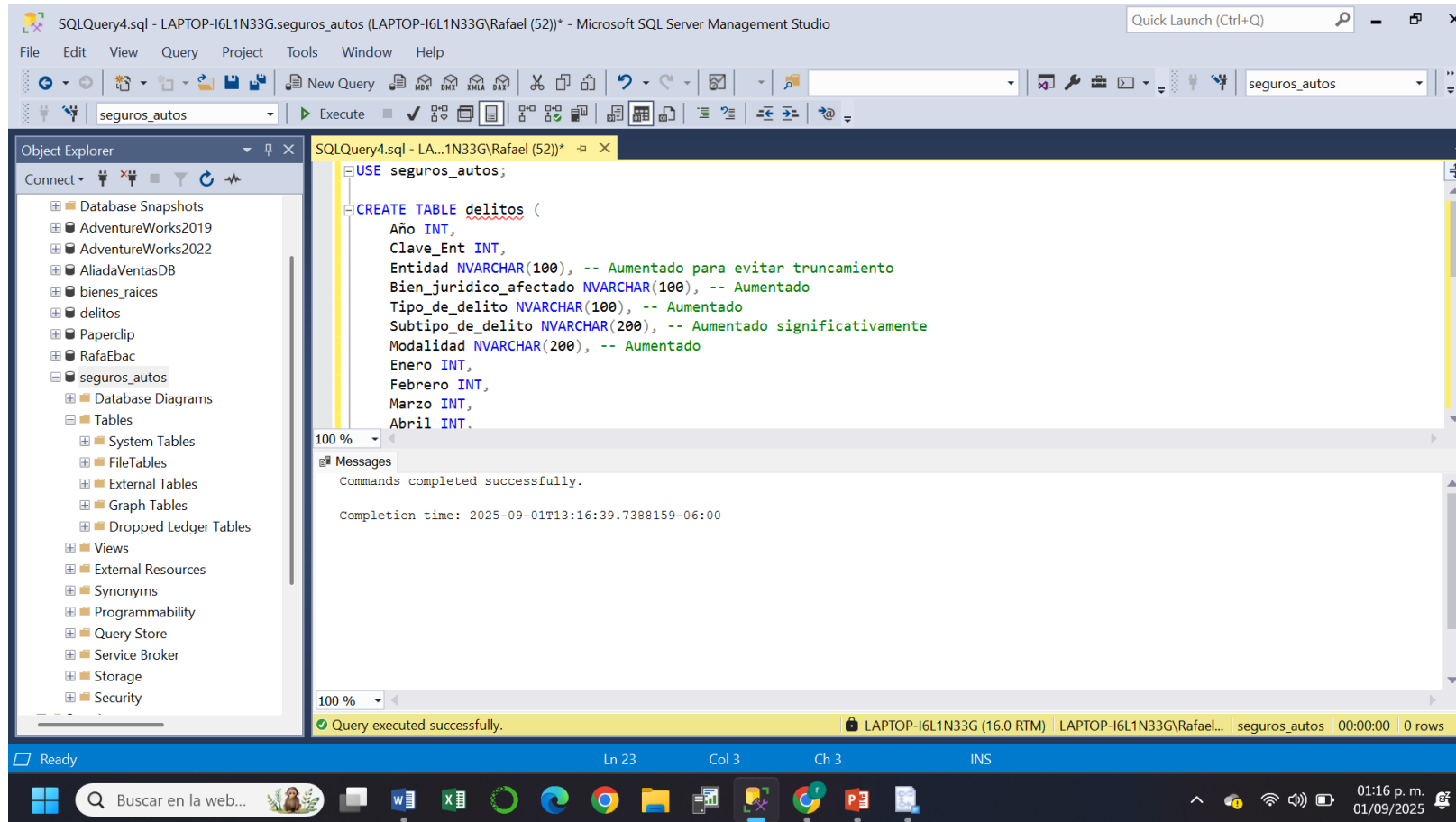
Descarga de los datos de incidencia delictiva del fuero común, *fecha de actualización 15 de agosto de 2025*

Estatat 2015 - 2025

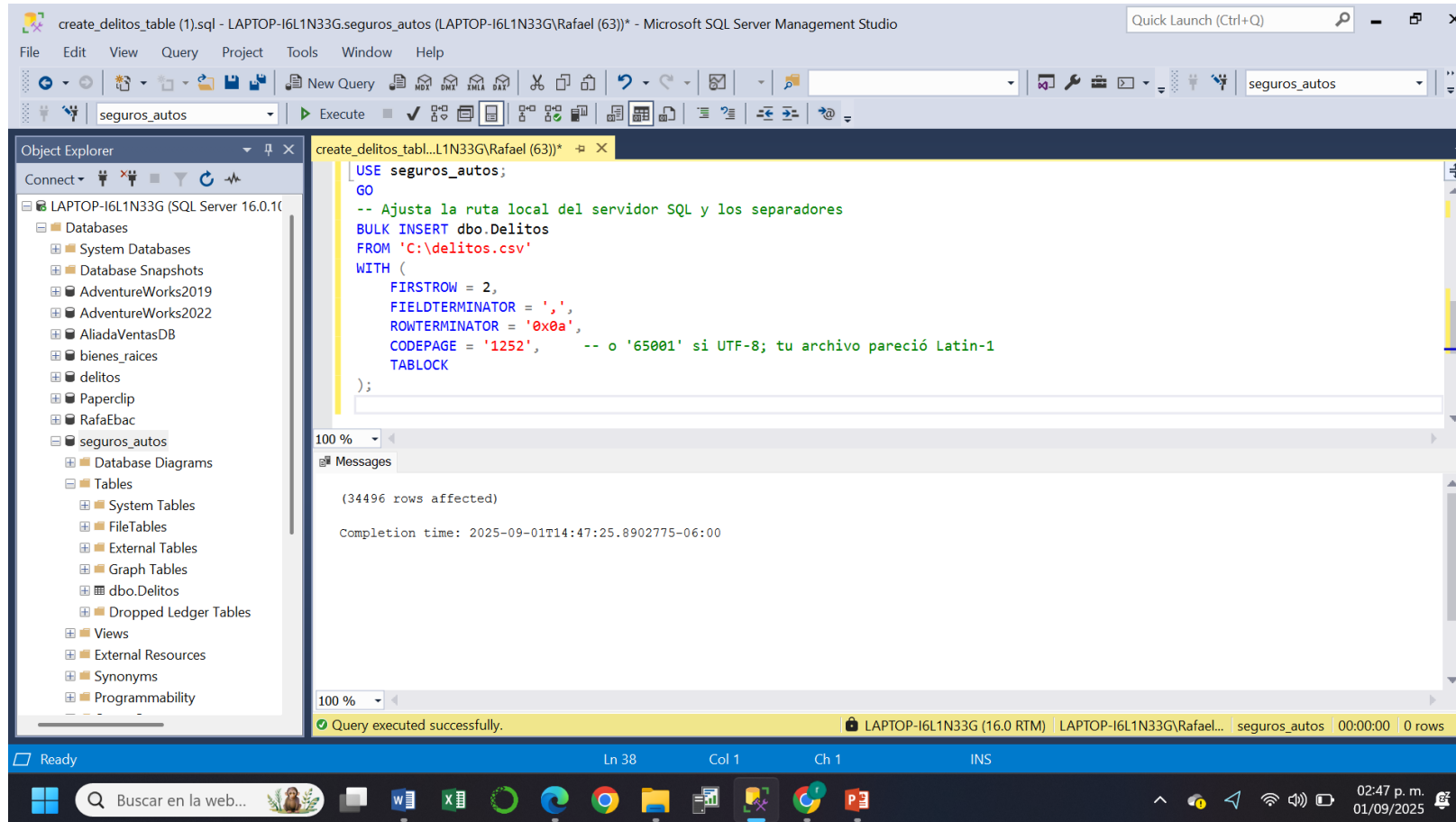
Municipal 2015 - 2025

Importante: Leer nota (PDF) al interior del archivo comprimido para instrucciones sobre como abrir los archivos, ya que por el número de filas, la base completa no puede ser cargada en una hoja de cálculo (Excel o Calc).

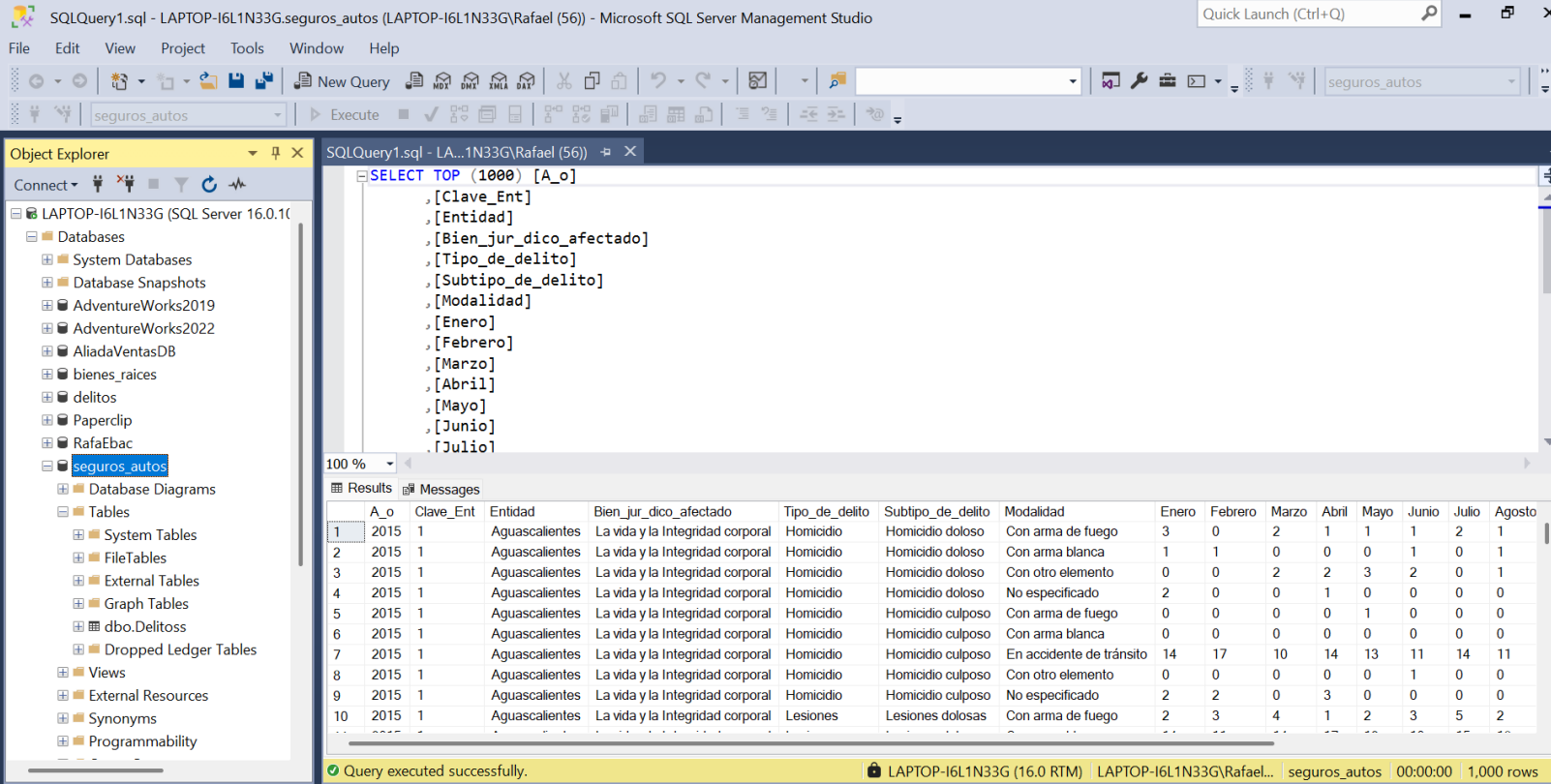
Descarga y Creación de la Base de Datos



Descarga y Creación de la Base de Datos



Descarga y Creación de la Base de Datos



SQLQuery1.sql - LAPTOP-I6L1N33G.seguros_autos (LAPTOP-I6L1N33G/Rafael (56)) - Microsoft SQL Server Management Studio

File Edit View Project Tools Window Help

seguros_autos Execute

Object Explorer

Connect ▼

LAPTOP-I6L1N33G (SQL Server 16.0.10058.1)

Databases

- System Databases
- Database Snapshots
- AdventureWorks2019
- AdventureWorks2022
- AliadaVentasDB
- bienes_raices
- delitos
- Paperclip
- RafaEbac
- seguros_autos**

Database Diagrams

- Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.Delitos
 - Dropped Ledger Tables
- Views
- External Resources
- Synonyms
- Programmability

SQLQuery1.sql - LA...1N33G/Rafael (56)

```
SELECT TOP (1000) [A_o]
,[Clave_Ent]
,[Entidad]
,[Bien_jur_dico_afectado]
,[Tipo_de_delito]
,[Subtipo_de_delito]
,[Modalidad]
,[Enero]
,[Febrero]
,[Marzo]
,[Abril]
,[Mayo]
,[Junio]
,[Julio]
```

100 %

Results Messages

	A_o	Clave_Ent	Entidad	Bien_jur_dico_afectado	Tipo_de_delito	Subtipo_de_delito	Modalidad	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto
1	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio doloso	Con arma de fuego	3	0	2	1	1	1	2	1
2	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio doloso	Con arma blanca	1	1	0	0	0	1	0	1
3	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio doloso	Con otro elemento	0	0	2	2	3	2	0	1
4	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio doloso	No especificado	2	0	0	1	0	0	0	0
5	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio culposo	Con arma de fuego	0	0	0	0	1	0	0	0
6	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio culposo	Con arma blanca	0	0	0	0	0	0	0	0
7	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio culposo	En accidente de tránsito	14	17	10	14	13	11	14	11
8	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio culposo	Con otro elemento	0	0	0	0	0	1	0	0
9	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio culposo	No especificado	2	2	0	3	0	0	0	0
10	2015	1	Aguascalientes	La vida y la Integridad corporal	Lesiones	Lesiones dolosas	Con arma de fuego	2	3	4	1	2	3	5	2

Query executed successfully.

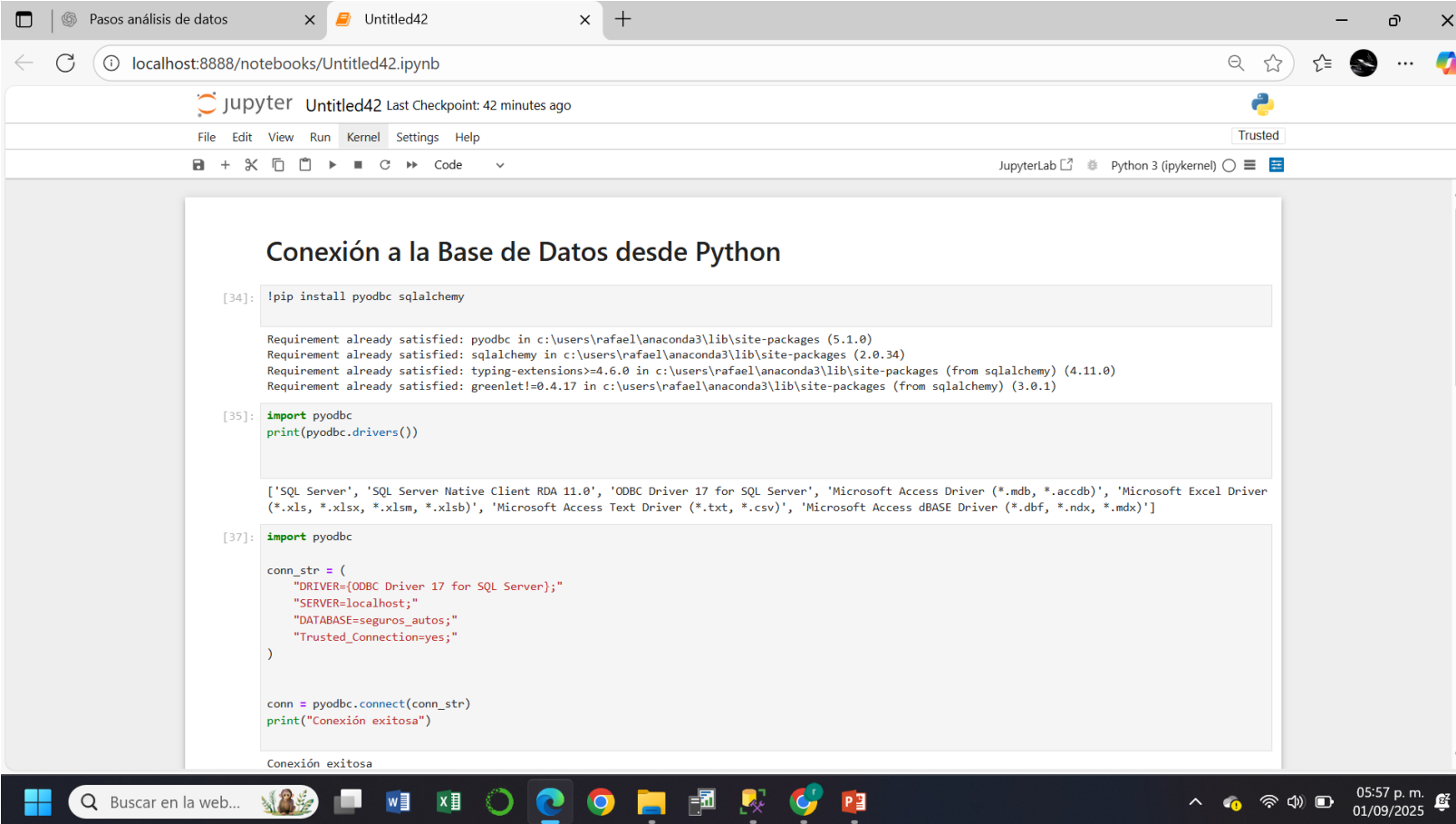
LAPTOP-I6L1N33G (16.0 RTM) LAPTOP-I6L1N33G/Rafael... seguros_autos 00:00:00 1,000 rows

Ready

Buscar en la web...

05:57 p. m.
01/09/2025

Conexión a la Base de Datos desde Python



The screenshot shows a JupyterLab interface in a web browser. The browser's address bar shows 'localhost:8888/notebooks/Untitled42.ipynb'. The JupyterLab header includes the title 'Untitled42' and a 'Last Checkpoint: 42 minutes ago' message. The interface has a menu bar with 'File', 'Edit', 'View', 'Run', 'Kernel', 'Settings', and 'Help'. Below the menu is a toolbar with icons for file operations and code execution. The main area displays a notebook with the title 'Conexión a la Base de Datos desde Python'. The notebook contains three code cells. The first cell, labeled '[34]:', runs the command 'pip install pyodbc sqlalchemy' and shows the output indicating that the requirements are already satisfied. The second cell, labeled '[35]:', imports 'pyodbc' and prints the list of available drivers. The output shows a list of drivers including 'SQL Server', 'SQL Server Native Client RDA 11.0', 'ODBC Driver 17 for SQL Server', and others. The third cell, labeled '[37]:', imports 'pyodbc' and defines a connection string 'conn_str' with the following parameters: 'DRIVER={ODBC Driver 17 for SQL Server};', 'SERVER=localhost;', 'DATABASE=seguros_autos;', and 'Trusted_Connection=yes;'. It then creates a connection object 'conn' using 'pyodbc.connect(conn_str)' and prints 'Conexión exitosa'. The output of the third cell shows 'Conexión exitosa'.

```
[34]: !pip install pyodbc sqlalchemy

Requirement already satisfied: pyodbc in c:\users\rafael\anaconda3\lib\site-packages (5.1.0)
Requirement already satisfied: sqlalchemy in c:\users\rafael\anaconda3\lib\site-packages (2.0.34)
Requirement already satisfied: typing-extensions>=4.6.0 in c:\users\rafael\anaconda3\lib\site-packages (from sqlalchemy) (4.11.0)
Requirement already satisfied: greenlet!=0.4.17 in c:\users\rafael\anaconda3\lib\site-packages (from sqlalchemy) (3.0.1)

[35]: import pyodbc
print(pyodbc.drivers())

['SQL Server', 'SQL Server Native Client RDA 11.0', 'ODBC Driver 17 for SQL Server', 'Microsoft Access Driver (*.mdb, *.accdb)', 'Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *.xlsb)', 'Microsoft Access Text Driver (*.txt, *.csv)', 'Microsoft Access dBASE Driver (*.dbf, *.ndx, *.mdx)']

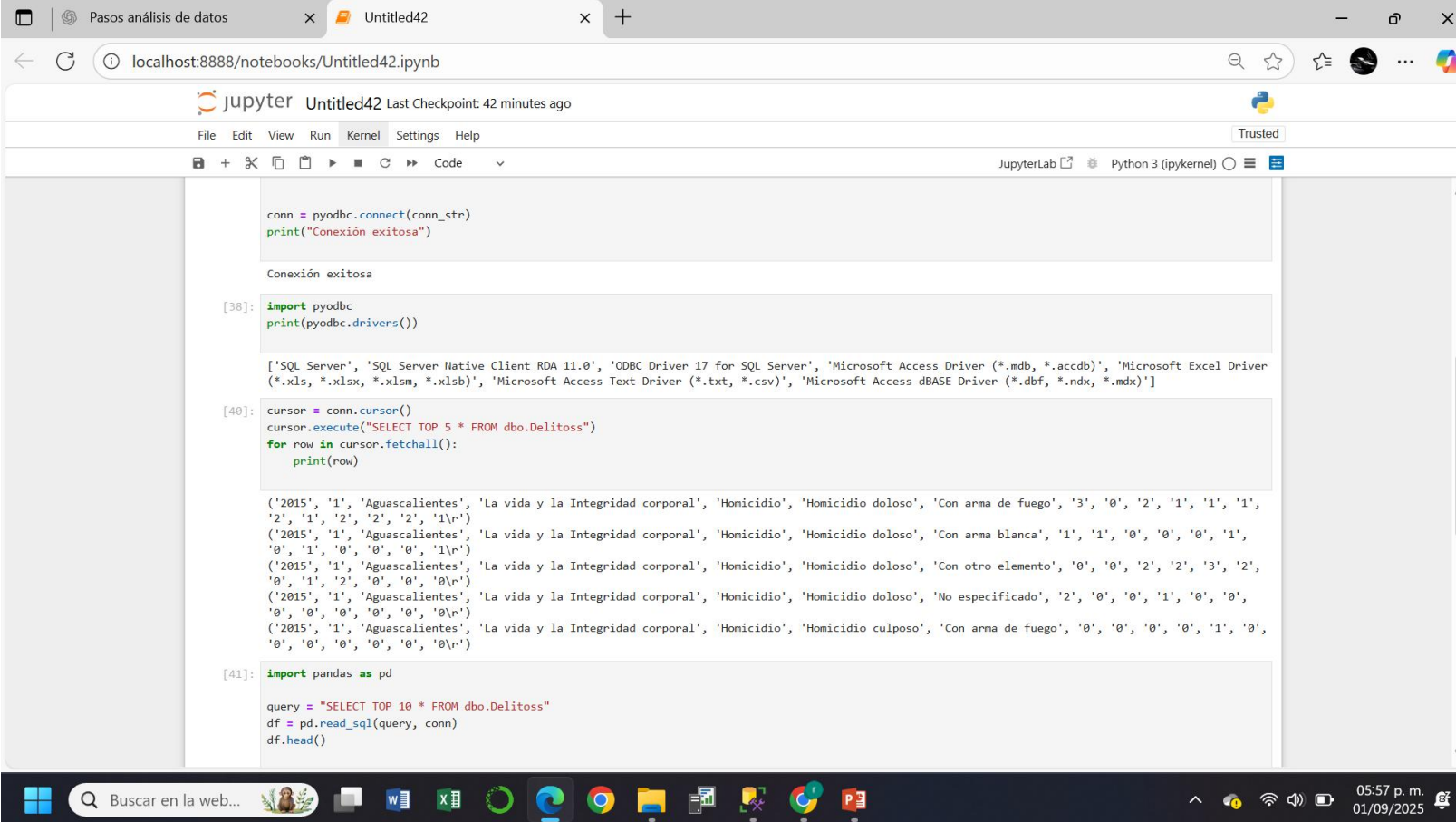
[37]: import pyodbc

conn_str = (
    "DRIVER={ODBC Driver 17 for SQL Server};"
    "SERVER=localhost;"
    "DATABASE=seguros_autos;"
    "Trusted_Connection=yes;"
)

conn = pyodbc.connect(conn_str)
print("Conexión exitosa")

Conexión exitosa
```


Conexión a la Base de Datos desde Python



The screenshot displays a Jupyter Notebook titled 'Untitled42' running on a local server at localhost:8888. The notebook contains three code cells. The first cell connects to a database using pyodbc and prints 'Conexión exitosa'. The second cell imports pyodbc and prints the list of available drivers. The third cell creates a cursor, executes a SQL query to select the top 5 records from the 'Delitoss' table, and prints the results. The output of the third cell shows a list of tuples containing data from the database. The Windows taskbar at the bottom indicates the date and time as 05:57 p. m. on 01/09/2025.

```
conn = pyodbc.connect(conn_str)
print("Conexión exitosa")

Conexión exitosa

[38]: import pyodbc
print(pyodbc.drivers())

['SQL Server', 'SQL Server Native Client RDA 11.0', 'ODBC Driver 17 for SQL Server', 'Microsoft Access Driver (*.mdb, *.accdb)', 'Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *.xlsb)', 'Microsoft Access Text Driver (*.txt, *.csv)', 'Microsoft Access dBASE Driver (*.dbf, *.ndx, *.mdx)']

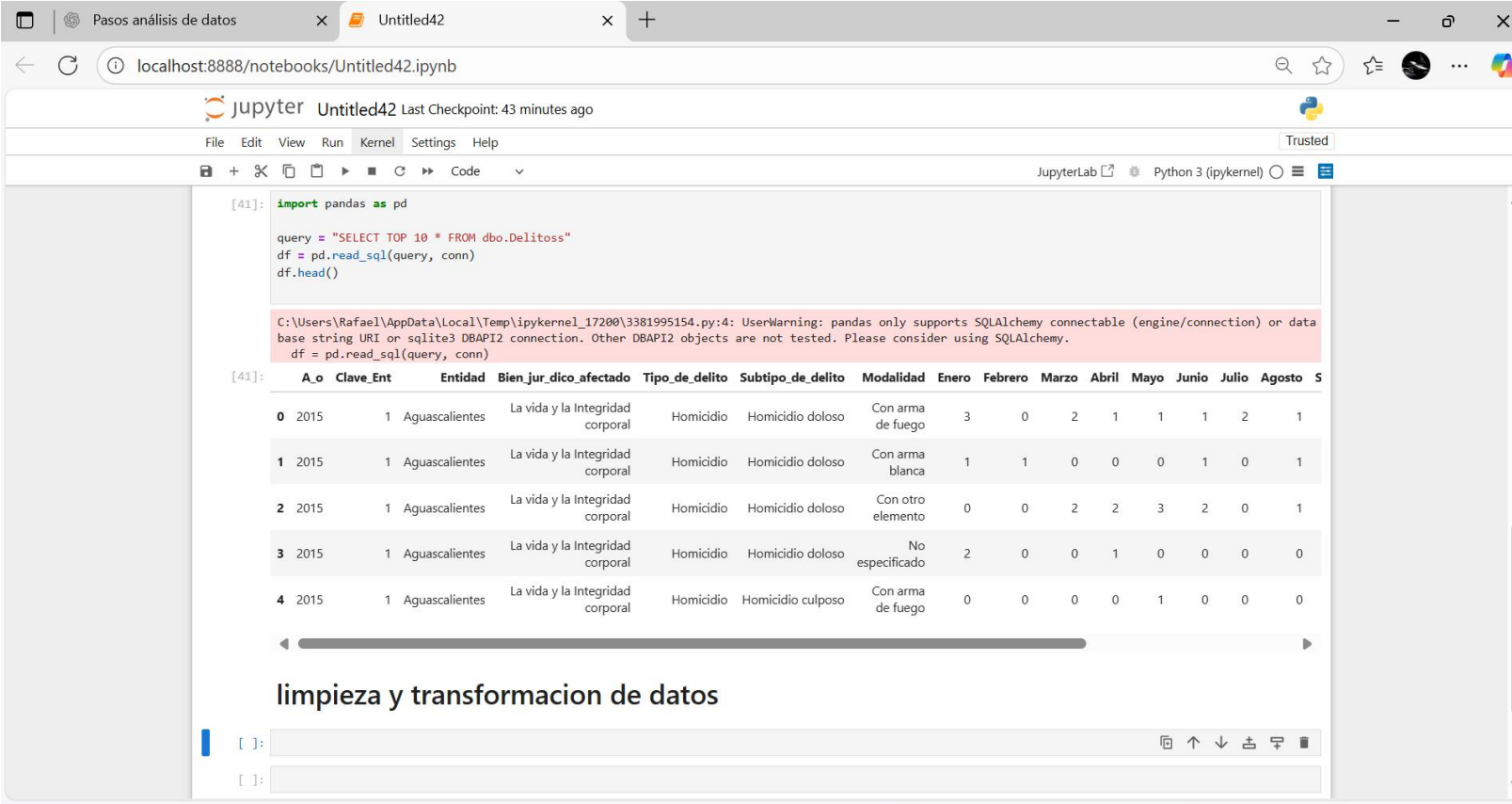
[40]: cursor = conn.cursor()
cursor.execute("SELECT TOP 5 * FROM dbo.Delitoss")
for row in cursor.fetchall():
    print(row)

('2015', '1', 'Aguascalientes', 'La vida y la Integridad corporal', 'Homicidio', 'Homicidio doloso', 'Con arma de fuego', '3', '0', '2', '1', '1', '1', '2', '1', '2', '2', '2', '1\r\n')
('2015', '1', 'Aguascalientes', 'La vida y la Integridad corporal', 'Homicidio', 'Homicidio doloso', 'Con arma blanca', '1', '1', '0', '0', '0', '1', '0', '1', '0', '0', '0', '1\r\n')
('2015', '1', 'Aguascalientes', 'La vida y la Integridad corporal', 'Homicidio', 'Homicidio doloso', 'Con otro elemento', '0', '0', '2', '2', '3', '2', '0', '1', '2', '0', '0', '0\r\n')
('2015', '1', 'Aguascalientes', 'La vida y la Integridad corporal', 'Homicidio', 'Homicidio doloso', 'No especificado', '2', '0', '0', '1', '0', '0', '0', '0', '0', '0', '0', '0\r\n')
('2015', '1', 'Aguascalientes', 'La vida y la Integridad corporal', 'Homicidio', 'Homicidio culposo', 'Con arma de fuego', '0', '0', '0', '0', '1', '0', '0', '0', '0', '0', '0', '0\r\n')

[41]: import pandas as pd

query = "SELECT TOP 10 * FROM dbo.Delitoss"
df = pd.read_sql(query, conn)
df.head()
```

Conexión a la Base de Datos desde Python



The screenshot shows a JupyterLab interface with a notebook titled "Untitled42". The notebook contains a code cell with the following Python code:

```
[41]: import pandas as pd

query = "SELECT TOP 10 * FROM dbo.Delitos"
df = pd.read_sql(query, conn)
df.head()
```

Below the code cell, a warning message is displayed:

C:\Users\Rafael\AppData\Local\Temp\ipykernel_17200\3381995154.py:4: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or data base string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.

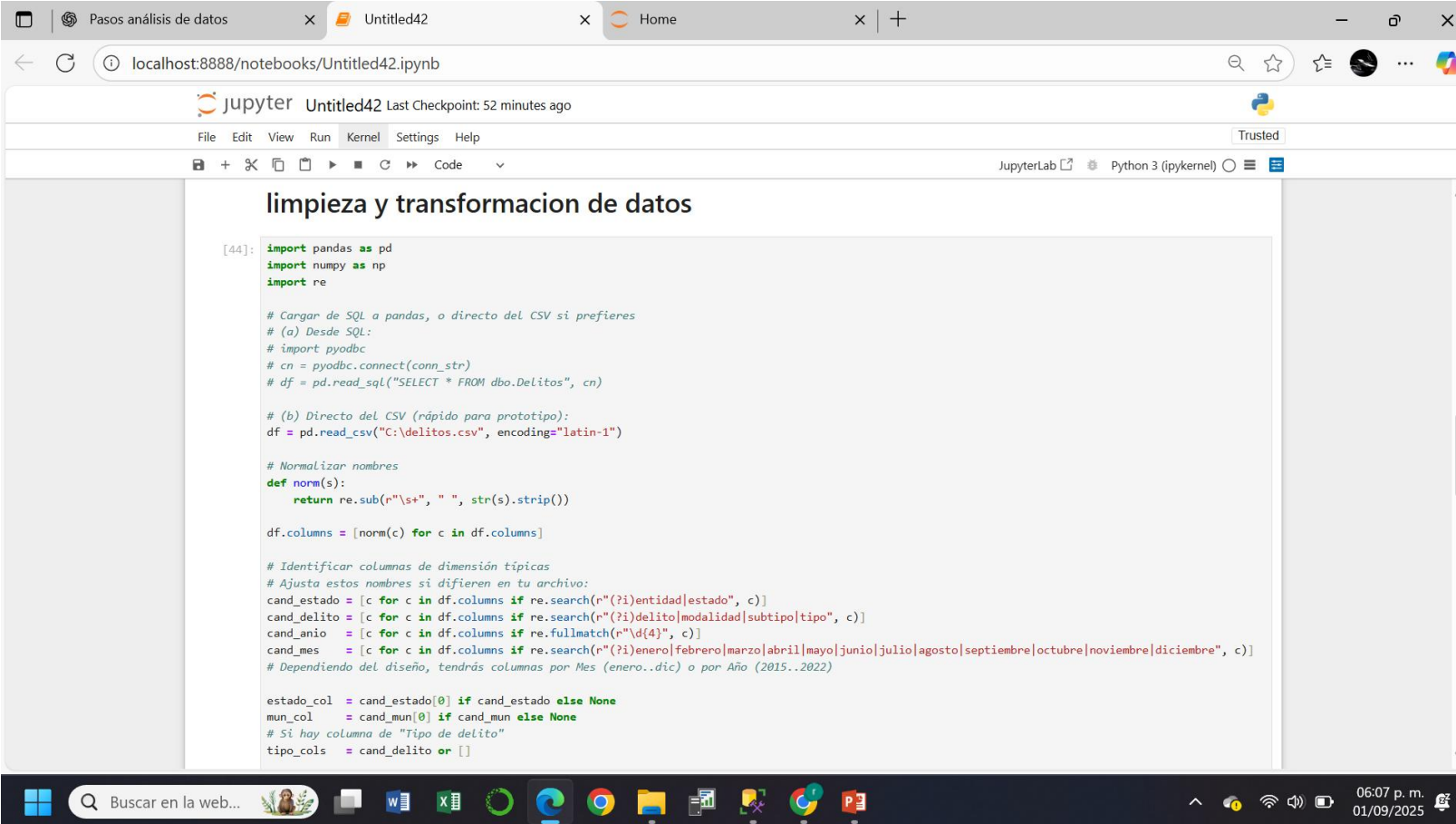
The output of the code cell is a pandas DataFrame with the following columns: A_o, Clave_Ent, Entidad, Bien_jur_dico_afectado, Tipo_de_delito, Subtipo_de_delito, Modalidad, Enero, Febrero, Marzo, Abril, Mayo, Junio, Julio, Agosto, and S. The DataFrame contains 5 rows of data.

A_o	Clave_Ent	Entidad	Bien_jur_dico_afectado	Tipo_de_delito	Subtipo_de_delito	Modalidad	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	S
0	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio doloso	Con arma de fuego	3	0	2	1	1	1	2	1
1	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio doloso	Con arma blanca	1	1	0	0	0	1	0	1
2	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio doloso	Con otro elemento	0	0	2	2	3	2	0	1
3	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio doloso	No especificado	2	0	0	1	0	0	0	0
4	2015	1	Aguascalientes	La vida y la Integridad corporal	Homicidio	Homicidio culposo	Con arma de fuego	0	0	0	0	1	0	0	0

Below the DataFrame, the text "limpieza y transformacion de datos" is displayed.

The JupyterLab interface includes a menu bar with options: File, Edit, View, Run, Kernel, Settings, Help. The status bar at the bottom shows the time as 05:58 p. m. on 01/09/2025.

Limpieza y Transformación de los Datos



The screenshot displays a JupyterLab environment with a browser window at the top showing the URL `localhost:8888/notebooks/Untitled42.ipynb`. The notebook interface includes a menu bar (File, Edit, View, Run, Kernel, Settings, Help) and a toolbar. The main area contains a code cell with the following Python code:

```
[44]: import pandas as pd
import numpy as np
import re

# Cargar de SQL a pandas, o directo del CSV si prefieres
# (a) Desde SQL:
# import pyodbc
# cn = pyodbc.connect(conn_str)
# df = pd.read_sql("SELECT * FROM dbo.Delitos", cn)

# (b) Directo del CSV (rápido para prototipo):
df = pd.read_csv("C:\\delitos.csv", encoding="latin-1")

# Normalizar nombres
def norm(s):
    return re.sub(r"\s+", " ", str(s).strip())

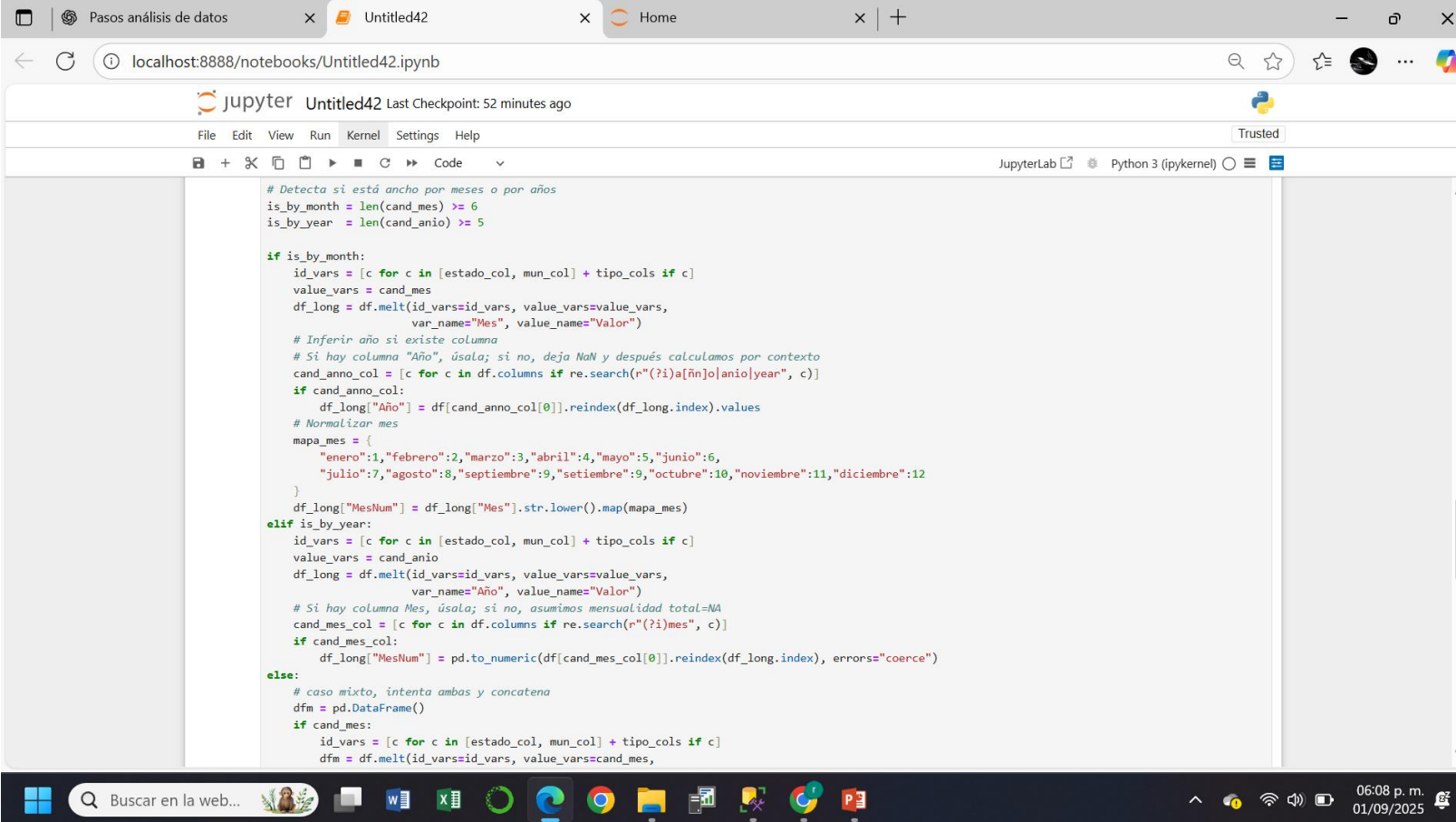
df.columns = [norm(c) for c in df.columns]

# Identificar columnas de dimensión típicas
# Ajusta estos nombres si difieren en tu archivo:
cand_estado = [c for c in df.columns if re.search(r"(?i)entidad|estado", c)]
cand_delito = [c for c in df.columns if re.search(r"(?i)delito|modalidad|subtipo|tipo", c)]
cand_año = [c for c in df.columns if re.fullmatch(r"d{4}", c)]
cand_mes = [c for c in df.columns if re.search(r"(?i)enero|febrero|marzo|abril|mayo|junio|julio|agosto|septiembre|octubre|noviembre|diciembre", c)]
# Dependiendo del diseño, tendrás columnas por Mes (enero..dic) o por Año (2015..2022)

estado_col = cand_estado[0] if cand_estado else None
mun_col = cand_mun[0] if cand_mun else None
# Si hay columna de "Tipo de delito"
tipo_cols = cand_delito or []
```

The Windows taskbar at the bottom shows the search bar, task view, and several application icons. The system clock indicates the time is 06:07 p. m. on 01/09/2025.

Limpieza y Transformación de los Datos



The screenshot displays a JupyterLab environment running in a web browser at localhost:8888/notebooks/Untitled42.ipynb. The interface includes a top navigation bar with tabs for 'Pasos análisis de datos', 'Untitled42', and 'Home'. Below the browser window, the JupyterLab header shows the 'jupyter' logo, the notebook name 'Untitled42', and the last checkpoint time 'Last Checkpoint: 52 minutes ago'. The main area contains a code editor with Python code for data cleaning and transformation. The code includes comments in Spanish and uses pandas and numpy libraries. The code is as follows:

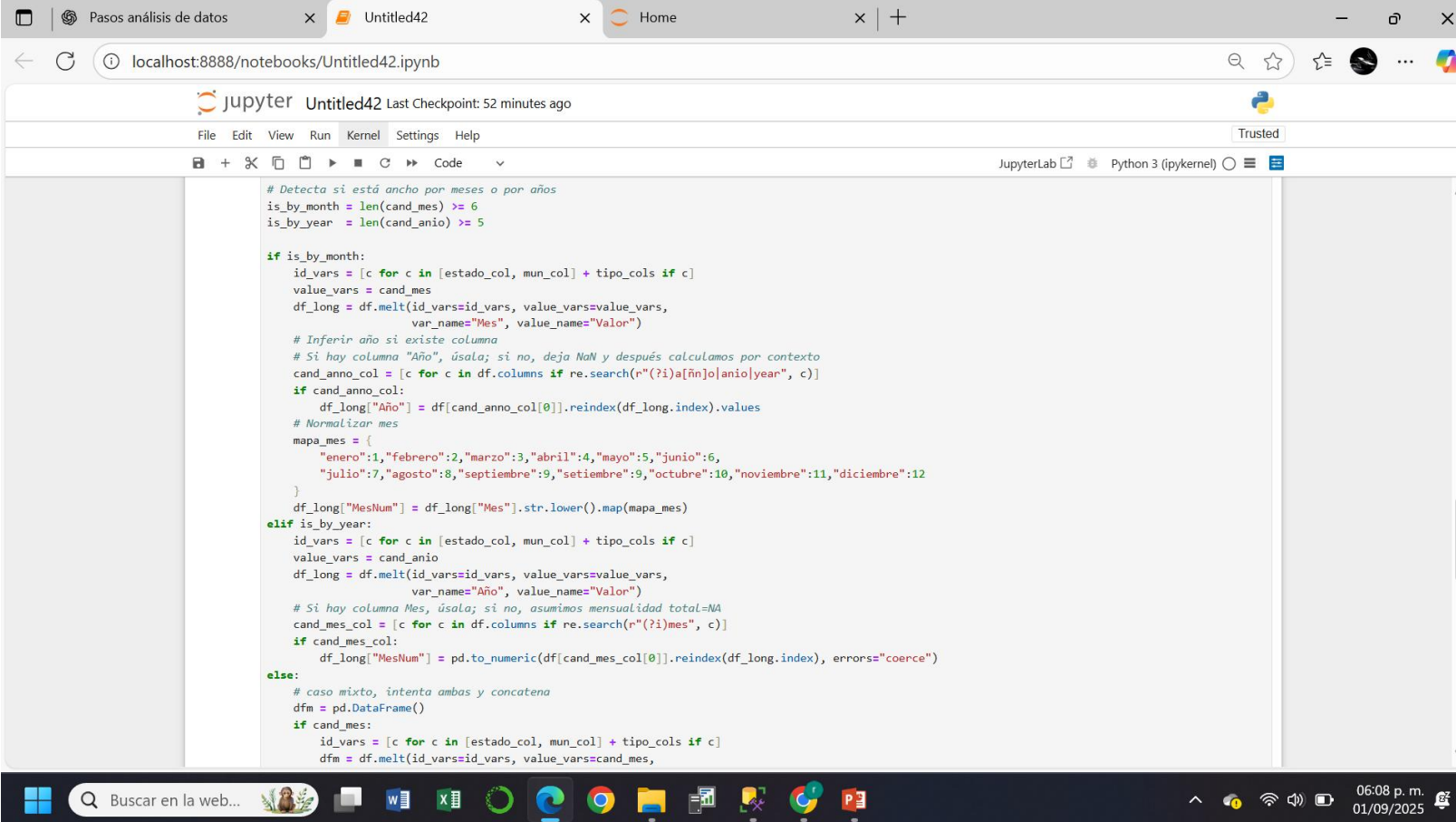
```
# Detecta si está ancho por meses o por años
is_by_month = len(cand_mes) >= 6
is_by_year = len(cand_anio) >= 5

if is_by_month:
    id_vars = [c for c in [estado_col, mun_col] + tipo_cols if c]
    value_vars = cand_mes
    df_long = df.melt(id_vars=id_vars, value_vars=value_vars,
                     var_name="Mes", value_name="Valor")

    # Inferir año si existe columna
    # Si hay columna "Año", úsala; si no, deja NaN y después calculamos por contexto
    cand_anio_col = [c for c in df.columns if re.search(r"(?i)a[ñ]o|year", c)]
    if cand_anio_col:
        df_long["Año"] = df[cand_anio_col[0]].reindex(df_long.index).values
    # Normalizar mes
    mapa_mes = {
        "enero":1,"febrero":2,"marzo":3,"abril":4,"mayo":5,"junio":6,
        "julio":7,"agosto":8,"septiembre":9,"setiembre":9,"octubre":10,"noviembre":11,"diciembre":12
    }
    df_long["MesNum"] = df_long["Mes"].str.lower().map(mapa_mes)
elif is_by_year:
    id_vars = [c for c in [estado_col, mun_col] + tipo_cols if c]
    value_vars = cand_anio
    df_long = df.melt(id_vars=id_vars, value_vars=value_vars,
                     var_name="Año", value_name="Valor")

    # Si hay columna Mes, úsala; si no, asumimos mensualidad total=NA
    cand_mes_col = [c for c in df.columns if re.search(r"(?i)mes", c)]
    if cand_mes_col:
        df_long["MesNum"] = pd.to_numeric(df[cand_mes_col[0]].reindex(df_long.index), errors="coerce")
else:
    # caso mixto, intenta ambas y concatena
    dfm = pd.DataFrame()
    if cand_mes:
        id_vars = [c for c in [estado_col, mun_col] + tipo_cols if c]
        dfm = df.melt(id_vars=id_vars, value_vars=cand_mes,
```

Limpieza y Transformación de los Datos



The screenshot displays a JupyterLab environment with a single notebook titled 'Untitled42'. The interface includes a top bar with tabs for 'Pasos análisis de datos', 'Untitled42', and 'Home'. The notebook's URL is 'localhost:8888/notebooks/Untitled42.ipynb'. The code in the notebook is as follows:

```
# Detecta si está ancho por meses o por años
is_by_month = len(cand_mes) >= 6
is_by_year = len(cand_anio) >= 5

if is_by_month:
    id_vars = [c for c in [estado_col, mun_col] + tipo_cols if c]
    value_vars = cand_mes
    df_long = df.melt(id_vars=id_vars, value_vars=value_vars,
                     var_name="Mes", value_name="Valor")

    # Inferir año si existe columna
    # Si hay columna "Año", úsala; si no, deja NaN y después calculamos por contexto
    cand_anio_col = [c for c in df.columns if re.search(r"(?i)a[ñ]o|year", c)]
    if cand_anio_col:
        df_long["Año"] = df[cand_anio_col[0]].reindex(df_long.index).values
    # Normalizar mes
    mapa_mes = {
        "enero":1,"febrero":2,"marzo":3,"abril":4,"mayo":5,"junio":6,
        "julio":7,"agosto":8,"septiembre":9,"setiembre":9,"octubre":10,"noviembre":11,"diciembre":12
    }
    df_long["MesNum"] = df_long["Mes"].str.lower().map(mapa_mes)
elif is_by_year:
    id_vars = [c for c in [estado_col, mun_col] + tipo_cols if c]
    value_vars = cand_anio
    df_long = df.melt(id_vars=id_vars, value_vars=value_vars,
                     var_name="Año", value_name="Valor")

    # Si hay columna Mes, úsala; si no, asumimos mensualidad total=NA
    cand_mes_col = [c for c in df.columns if re.search(r"(?i)mes", c)]
    if cand_mes_col:
        df_long["MesNum"] = pd.to_numeric(df[cand_mes_col[0]].reindex(df_long.index), errors="coerce")
else:
    # caso mixto, intenta ambas y concatena
    dfm = pd.DataFrame()
    if cand_mes:
        id_vars = [c for c in [estado_col, mun_col] + tipo_cols if c]
        dfm = df.melt(id_vars=id_vars, value_vars=cand_mes,
```

Limpieza y Transformación de los Datos

```
[1]: import pandas as pd
```

```
df = pd.read_csv("delitos.csv", encoding="latin1") # o cp1252 si da error
df.columns = df.columns.str.strip() # eliminar espacios sobrantes
print(df.head())
print(df.columns)
```

	Año	Clave_Ent	Entidad	Bien jurídico afectado \
0	2015	1	Aguascalientes	La vida y la Integridad corporal
1	2015	1	Aguascalientes	La vida y la Integridad corporal
2	2015	1	Aguascalientes	La vida y la Integridad corporal
3	2015	1	Aguascalientes	La vida y la Integridad corporal
4	2015	1	Aguascalientes	La vida y la Integridad corporal

	Tipo de delito	Subtipo de delito	Modalidad	Enero	Febrero	Marzo \
0	Homicidio	Homicidio doloso	Con arma de fuego	3	0	2
1	Homicidio	Homicidio doloso	Con arma blanca	1	1	0
2	Homicidio	Homicidio doloso	Con otro elemento	0	0	2
3	Homicidio	Homicidio doloso	No especificado	2	0	0
4	Homicidio	Homicidio culposo	Con arma de fuego	0	0	0

	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre \
0	1	1	1	2	1.0	2.0	2.0	2.0
1	0	0	1	0	1.0	0.0	0.0	0.0
2	2	3	2	0	1.0	2.0	0.0	0.0
3	1	0	0	0	0.0	0.0	0.0	0.0
4	0	1	0	0	0.0	0.0	0.0	0.0

	Diciembre	Anual
0	1.0	18
1	1.0	5
2	0.0	12
3	0.0	3
4	0.0	1

Limpieza y Transformación de los Datos

```
[5]: df_melted = df.melt(
      id_vars=[
          "AÑO", "CLAVE_ENT", "ENTIDAD",
          "BIEN JURÍDICO AFECTADO", "TIPO DE DELITO",
          "SUBTIPO DE DELITO", "MODALIDAD"
      ],
      value_vars=[
          "ENERO", "FEBRERO", "MARZO", "ABRIL", "MAYO", "JUNIO",
          "JULIO", "AGOSTO", "SEPTIEMBRE", "OCTUBRE", "NOVIEMBRE", "DICIEMBRE"
      ],
      var_name="MES",
      value_name="CASOS"
  )

print(df_melted.head())
```

	AÑO	CLAVE_ENT	ENTIDAD	BIEN JURÍDICO AFECTADO \
0	2015	1	Aguascalientes	La vida y la Integridad corporal
1	2015	1	Aguascalientes	La vida y la Integridad corporal
2	2015	1	Aguascalientes	La vida y la Integridad corporal
3	2015	1	Aguascalientes	La vida y la Integridad corporal
4	2015	1	Aguascalientes	La vida y la Integridad corporal

	TIPO DE DELITO	SUBTIPO DE DELITO	MODALIDAD	MES	CASOS
0	Homicidio	Homicidio doloso	Con arma de fuego	ENERO	3.0
1	Homicidio	Homicidio doloso	Con arma blanca	ENERO	1.0
2	Homicidio	Homicidio doloso	Con otro elemento	ENERO	0.0
3	Homicidio	Homicidio doloso	No especificado	ENERO	2.0
4	Homicidio	Homicidio culposo	Con arma de fuego	ENERO	0.0

```
[72]: df_vehiculos = df_melted[df_melted["TIPO DE DELITO"].str.contains("veh", case=False, na=False)]
print(df_vehiculos.head())
```

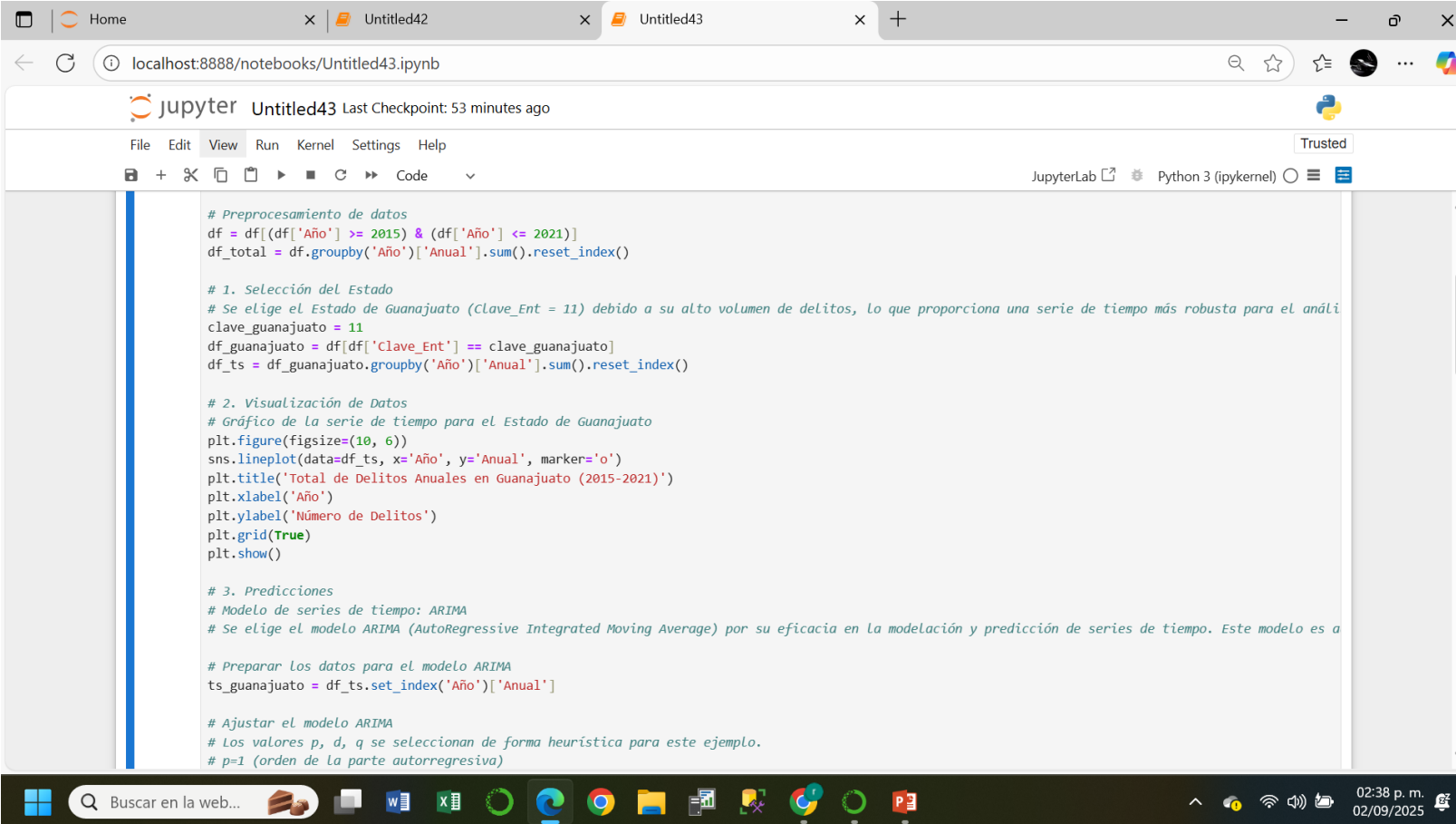
```
Empty DataFrame
Columns: [AÑO, CLAVE_ENT, ENTIDAD, BIEN JURÍDICO AFECTADO, TIPO DE DELITO, SUBTIPO DE DELITO, MODALIDAD, MES, CASOS]
Index: []
```

Análisis de Series de Tiempo

Explicación para Tendencias de Delitos entre 2015 - 2021 y predicción para 2022

- En el análisis de series de tiempo para la predicción de delitos, el proceso se centró en identificar y proyectar los patrones delictivos a lo largo del tiempo.
- Primero, se **seleccionó el estado de Guanajuato** debido a su alto volumen de datos, lo que permite un análisis más robusto. Luego, se visualizó la serie de tiempo para entender la tendencia de los delitos entre 2015 y 2021.
- Finalmente, se utilizó el modelo **ARIMA (AutoRegressive Integrated Moving Average)**, que es ideal para series de tiempo, ya que es capaz de capturar la relación de los datos actuales con sus valores pasados y la dependencia entre los errores de la predicción, lo que resultó en una predicción para el año 2022.

Análisis de Series de Tiempo



```
# Preprocesamiento de datos
df = df[(df['Año'] >= 2015) & (df['Año'] <= 2021)]
df_total = df.groupby('Año')['Anual'].sum().reset_index()

# 1. Selección del Estado
# Se elige el Estado de Guanajuato (Clave_Ent = 11) debido a su alto volumen de delitos, lo que proporciona una serie de tiempo más robusta para el análisis
clave_guanajuato = 11
df_guanajuato = df[df['Clave_Ent'] == clave_guanajuato]
df_ts = df_guanajuato.groupby('Año')['Anual'].sum().reset_index()

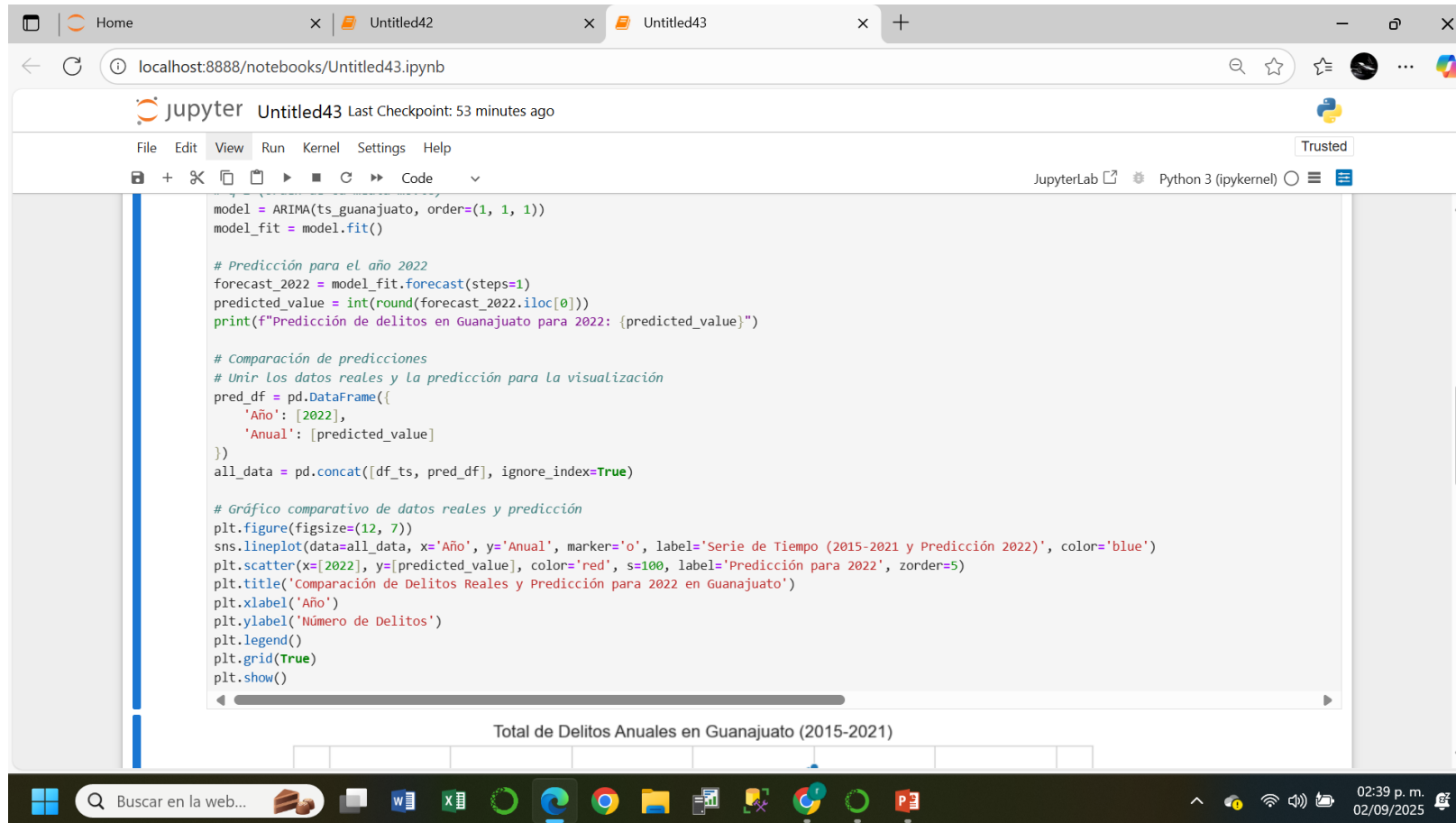
# 2. Visualización de Datos
# Gráfico de la serie de tiempo para el Estado de Guanajuato
plt.figure(figsize=(10, 6))
sns.lineplot(data=df_ts, x='Año', y='Anual', marker='o')
plt.title('Total de Delitos Anuales en Guanajuato (2015-2021)')
plt.xlabel('Año')
plt.ylabel('Número de Delitos')
plt.grid(True)
plt.show()

# 3. Predicciones
# Modelo de series de tiempo: ARIMA
# Se elige el modelo ARIMA (AutoRegressive Integrated Moving Average) por su eficacia en la modelación y predicción de series de tiempo. Este modelo es adecuado para series de tiempo que exhiben una tendencia y/o estacionalidad.

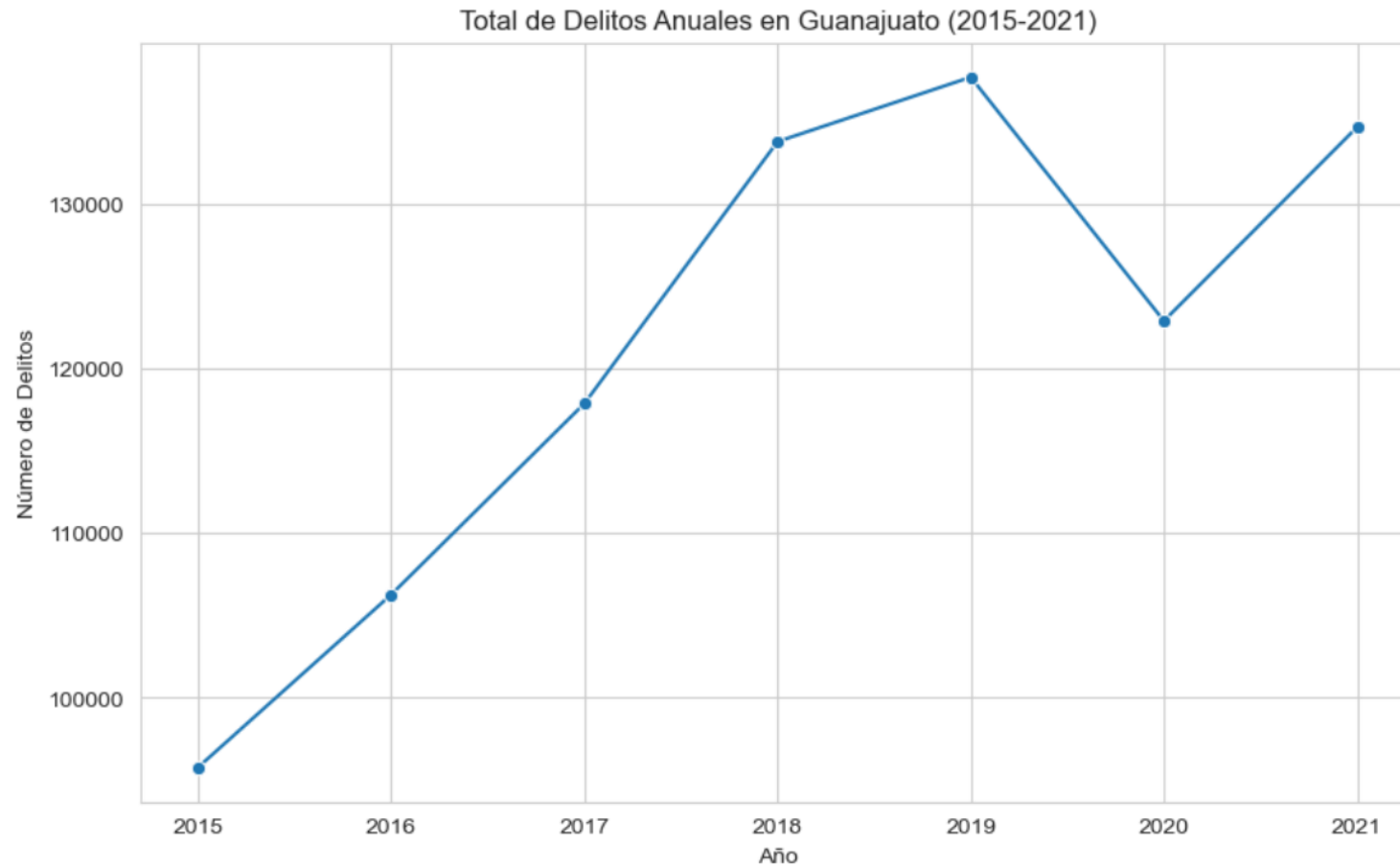
# Preparar los datos para el modelo ARIMA
ts_guanajuato = df_ts.set_index('Año')['Anual']

# Ajustar el modelo ARIMA
# Los valores p, d, q se seleccionan de forma heurística para este ejemplo.
# p=1 (orden de la parte autorregresiva)
```

Análisis de Series de Tiempo

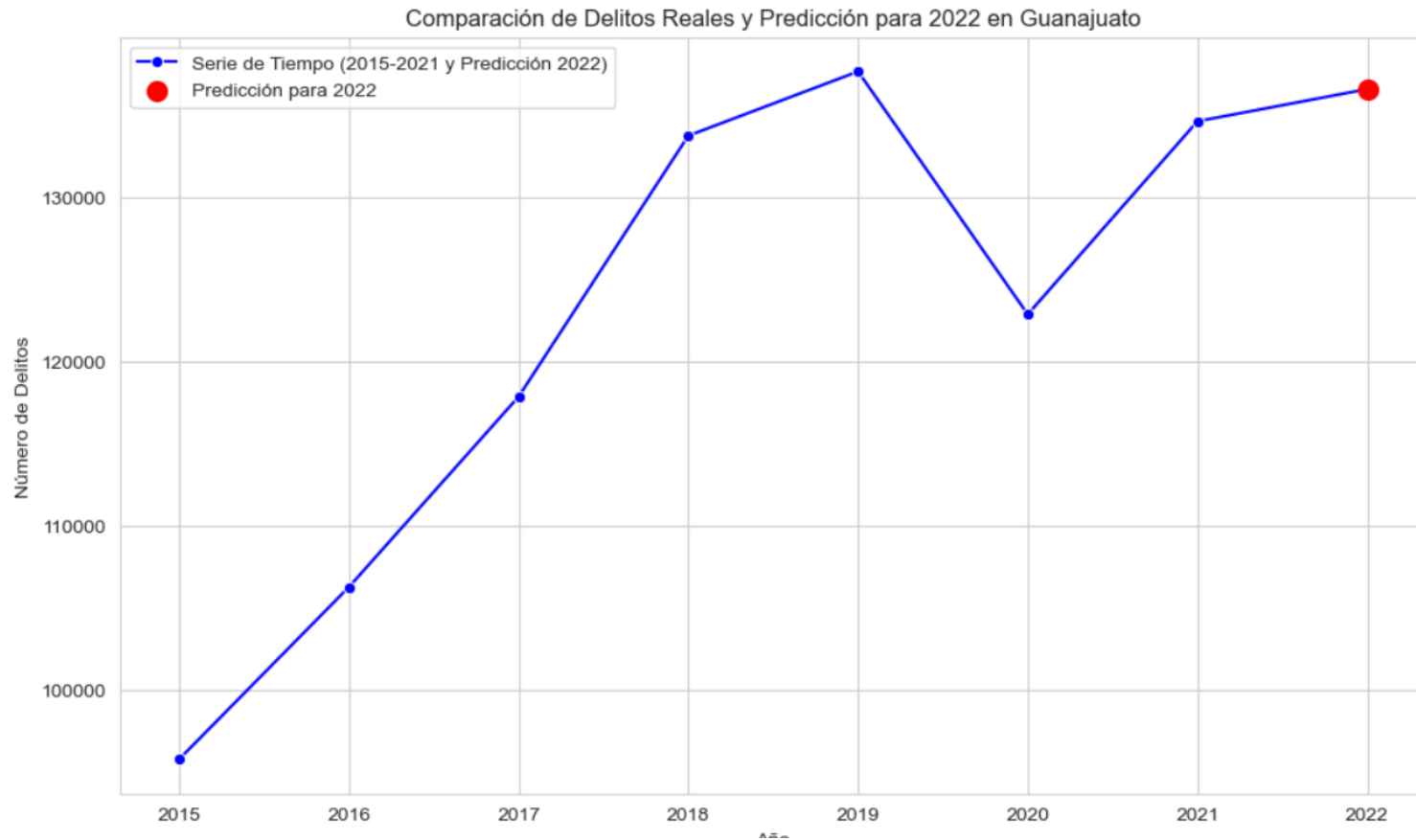


Análisis de Series de Tiempo



Análisis de Series de Tiempo

Predicción de delitos en Guanajuato para 2022: 136582



Clasificación de Estados por Peligrosidad

1. Normalización de Datos

- Para el análisis, se filtraron los datos del archivo delitos.csv para incluir solo el año 2021.
- Luego, se agruparon los delitos por cada estado para obtener el total anual. El paso crucial fue la normalización de los datos usando StandardScaler.
- Esto transformó la columna Anual a una escala estandarizada, con una media de 0 y una desviación estándar de 1, lo que previene que la magnitud de los valores absolutos de los estados con más delitos sesgue los resultados del clustering.

2. Aplicación de Algoritmos (K-means)

- Se eligió el algoritmo K-means debido a su simplicidad y eficiencia para agrupar conjuntos de datos.
- Para determinar el número óptimo de clusters (k), se utilizó el método del codo.
- El gráfico mostró un punto de inflexión claro en $k=3$, lo que indica que agrupar los estados en tres categorías (menos peligrosos, peligrosidad media y más peligrosos) es la división más natural y significativa.

Clasificación de Estados por Peligrosidad

```
# 1. Preparación de Datos
# Filtrar Los datos para el año 2021
df_2021 = df[df['Año'] == 2021]

# Agrupar Los datos por estado y sumar Los delitos anuales
df_estado = df_2021.groupby('Entidad')['Anual'].sum().reset_index()

# Normalizar Los datos
# Se utiliza StandardScaler para escalar Los datos, asegurando que cada característica (en este caso, 'Anual') tenga una media de 0 y una desviación e
scaler = StandardScaler()
df_estado['Anual_Normalizado'] = scaler.fit_transform(df_estado[['Anual']])

# Preparar el DataFrame para K-means
X = df_estado[['Anual_Normalizado']]

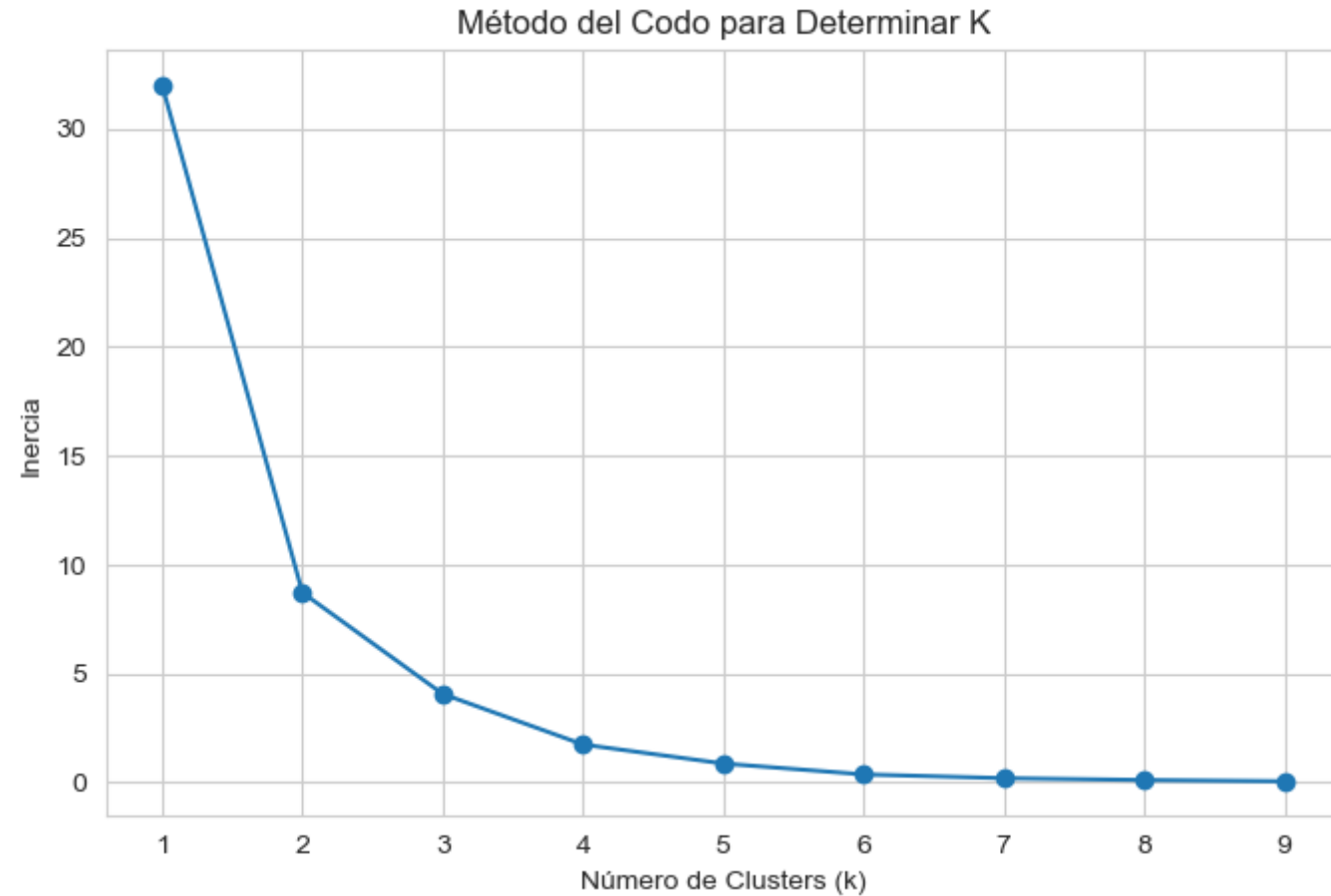
# 2. Aplicación de Algoritmos (K-means)
# Determinación del número de clusters (k) usando el método del codo
# El método del codo busca el valor de 'k' donde La disminución en La inercia (suma de Las distancias cuadradas de Las muestras a su centro de clúster
inercia = []
for i in range(1, 10):
    kmeans = KMeans(n_clusters=i, random_state=42, n_init=10)
    kmeans.fit(X)
    inercia.append(kmeans.inertia_)

plt.figure(figsize=(8, 5))
plt.plot(range(1, 10), inercia, marker='o')
plt.title('Método del Codo para Determinar K')
plt.xlabel('Número de Clusters (k)')
plt.ylabel('Inercia')
plt.grid(True)
plt.show()

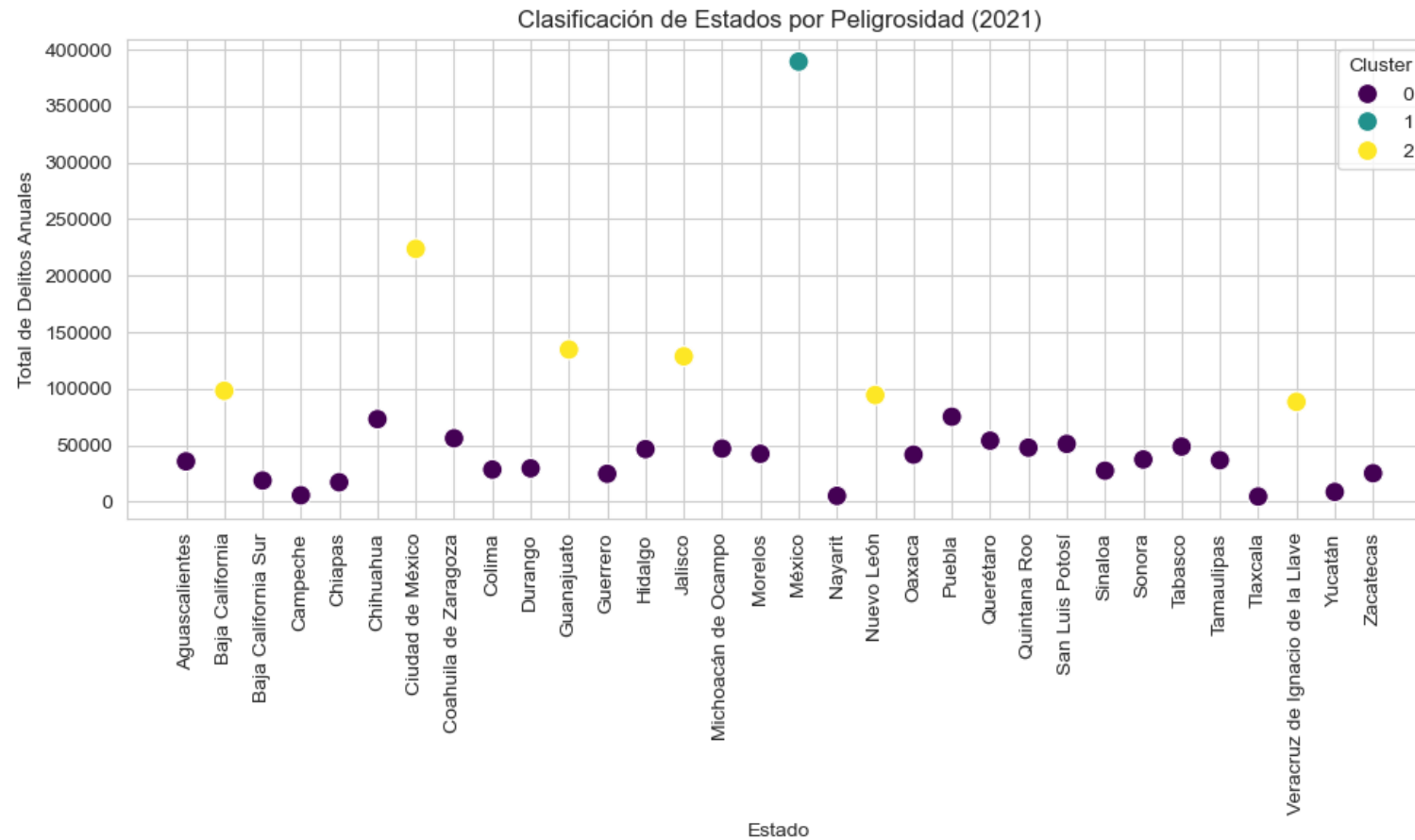
# Basado en el gráfico, el "codo" se encuentra en k=3.
k = 3
kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
df_estado['Cluster'] = kmeans.fit_predict(X)

# Gráfico de dispersión de Los clusters
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Entidad', y='Anual', hue='Cluster', data=df_estado, palette='viridis', s=100)
plt.title('Clasificación de Estados por Peligrosidad (2021)')
plt.xlabel('Estado')
plt.ylabel('Total de Delitos Anuales')
plt.xticks(rotation=90)
plt.grid(True)
plt.tight_layout()
plt.show()
```

Clasificación de Estados por Peligrosidad



Clasificación de Estados por Peligrosidad



Clasificación de Estados por Peligrosidad

3. Interpretación de Resultados

El análisis de K-means clasificó a los estados en tres clusters, cada uno representando un nivel de peligrosidad.

- Cluster 0 (Menos Peligrosos): Este grupo incluye a los estados con los promedios de delitos más bajos. Por ejemplo, Tlaxcala y Campeche se encuentran en este grupo.
- Cluster 1 (Peligrosidad Media): Aquí se encuentran los estados con un número de delitos moderado, como Aguascalientes y Querétaro.
- Cluster 2 (Más Peligrosos): Este cluster está compuesto por los estados con los mayores volúmenes de delitos, como el Estado de México y la Ciudad de México.

Clasificación de Estados por Peligrosidad

```
# 3. Interpretación de Resultados
# Características de cada cluster
cluster_caracteristicas = df_estado.groupby('Cluster')['Anual'].agg(['mean', 'count']).sort_values(by='mean').reset_index()
cluster_caracteristicas.columns = ['Cluster', 'Delitos Promedio', 'Número de Estados']

print("Características de cada cluster:")
print(cluster_caracteristicas)

# Identificación de los estados más y menos peligrosos
# Los clusters se ordenan de menor a mayor peligrosidad (según el promedio de delitos).
# Cluster 0: Menos peligrosos
# Cluster 1: Peligrosidad media
# Cluster 2: Más peligrosos

cluster_0 = df_estado[df_estado['Cluster'] == cluster_caracteristicas.iloc[0]['Cluster']].sort_values(by='Anual')
cluster_1 = df_estado[df_estado['Cluster'] == cluster_caracteristicas.iloc[1]['Cluster']].sort_values(by='Anual')
cluster_2 = df_estado[df_estado['Cluster'] == cluster_caracteristicas.iloc[2]['Cluster']].sort_values(by='Anual')

print("\nEstados menos peligrosos (Cluster 0):")
print(cluster_0[['Entidad', 'Anual']])

print("\nEstados de peligrosidad media (Cluster 1):")
print(cluster_1[['Entidad', 'Anual']])

print("\nEstados más peligrosos (Cluster 2):")
print(cluster_2[['Entidad', 'Anual']])
```

Clasificación de Estados por Peligrosidad

Características de cada cluster:

Cluster	Delitos Promedio	Número de Estados
0	35483.6	25
1	127945.5	6
2	389492.0	1

Estados menos peligrosos (Cluster 0):

	Entidad	Anual
28	Tlaxcala	4527
17	Nayarit	5072
3	Campeche	5611
30	Yucatán	8565
4	Chiapas	17130
2	Baja California Sur	18677
11	Guerrero	24629
31	Zacatecas	25110
24	Sinaloa	27386
8	Colima	28368
9	Durango	29479
0	Aguascalientes	35645
27	Tamaulipas	36636
25	Sonora	37301
19	Oaxaca	41590
15	Morelos	42301
12	Hidalgo	46464
14	Michoacán de Ocampo	46925
22	Quintana Roo	47753
26	Tabasco	48715
23	San Luis Potosí	51070
21	Querétaro	53944
7	Coahuila de Zaragoza	56045
5	Chihuahua	73006
20	Puebla	75141

Estados de peligrosidad media (Cluster 1):

	Entidad	Anual
29	Veracruz de Ignacio de la Llave	88306
18	Nuevo León	94321
1	Baja California	98090
13	Jalisco	128588
10	Guanajuato	134626
6	Ciudad de México	223742

Estados más peligrosos (Cluster 2):

	Entidad	Anual
16	México	389492

Enlace y Capturas de Pantalla del Tablero de Google Data Studio

Mayor Cantidad de Delitos cometidos entre 2015-2025 en México

	Entidad	Tipo de delito	Subtipo de delito	Modalidad	Anual ▾
1.	México	Otros delitos del Fuero Común	Otros delitos del Fuero Común	Otros delitos del Fuero Común	
2.	México	Lesiones	Lesiones dolosas	Con otro elemento	
3.	Ciudad de México	Violencia familiar	Violencia familiar	Violencia familiar	
4.	México	Robo	Otros robos	Sin violencia	
5.	Ciudad de México	Robo	Otros robos	Sin violencia	
6.	Guanajuato	Otros delitos del Fuero Común	Otros delitos del Fuero Común	Otros delitos del Fuero Común	
7.	Nuevo León	Violencia familiar	Violencia familiar	Violencia familiar	
8.	México	Robo	Robo de vehículo automotor	Robo de coche de 4 ruedas Con violencia	
9.	México	Robo	Robo a transeúnte en vía públ...	Con violencia	
10.	Ciudad de México	Fraude	Fraude	Fraude	
11.	México	Violencia familiar	Violencia familiar	Violencia familiar	
12.	México	Robo	Robo de vehículo automotor	Robo de coche de 4 ruedas Sin violencia	
13.	Guanajuato	Robo	Otros robos	Sin violencia	
14.	Ciudad de México	Amenazas	Amenazas	Amenazas	
15.	Guanajuato	Narcomenudeo	Narcomenudeo	Narcomenudeo	
16.	México	Daño a la propiedad	Daño a la propiedad	Daño a la propiedad	
17.	Chihuahua	Violencia familiar	Violencia familiar	Violencia familiar	
18.	Guanajuato	Violencia familiar	Violencia familiar	Violencia familiar	
19.	Jalisco	Robo	Otros robos	Sin violencia	

1 - 100 / 3138 < >

Gráfico de Tendência Anual de Delitos

- Este gráfico de líneas es la pieza central del análisis de series de tiempo. Muestra la tendencia de los delitos a lo largo de los años (2015-2021)

Enlace y Capturas de Pantalla del Tablero de Google Data Studio

Visualizar la Evolución de los Delitos

Gráfico de Series de Temporal de Delitos a Nivel Nacional

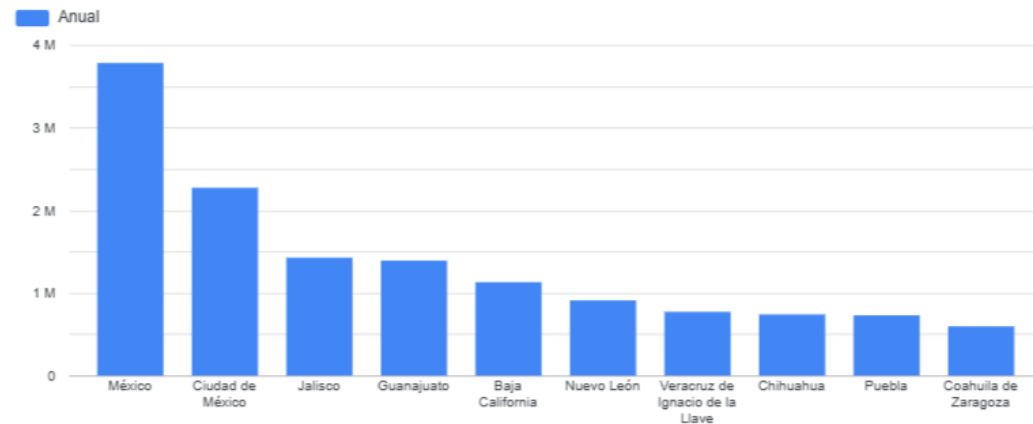


- Una herramienta de visualización de datos como Looker Studio es **fundamental para la toma de decisiones basada en datos.**

Enlace y Capturas de Pantalla del Tablero de Google Data Studio

Nivel de Peligrosidad

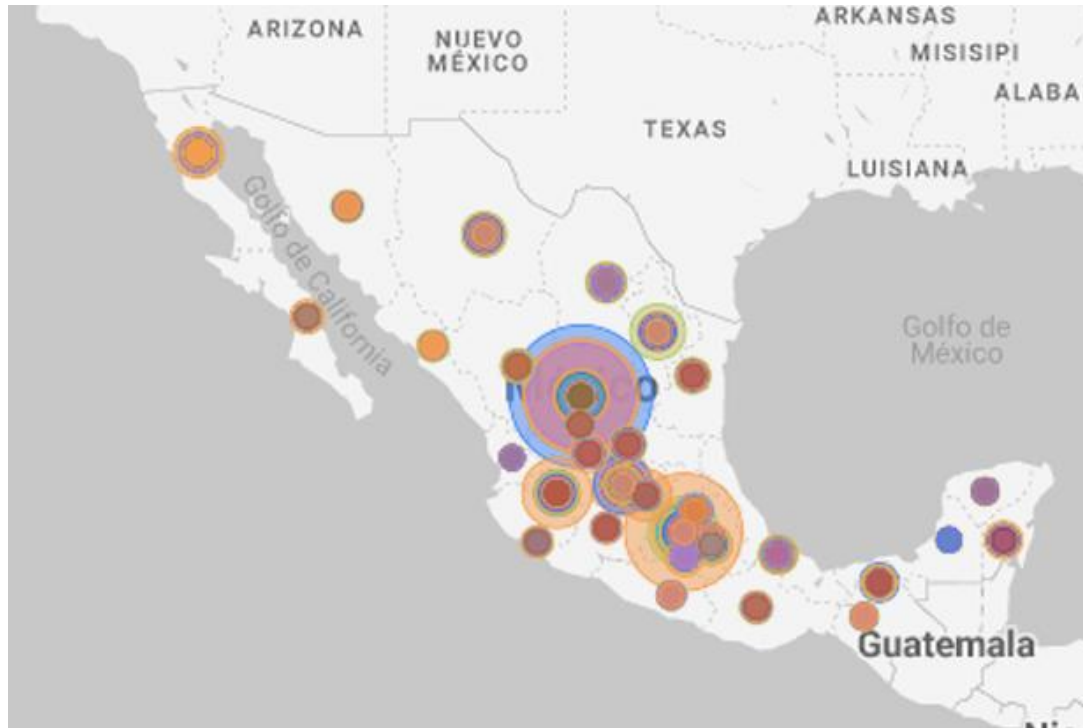
Gráfico de Barras para Clasificación de Estados



- Transforma datos crudos y complejos en gráficos y tablas fáciles de entender. Esto democratiza el acceso a la información, permitiendo que cualquier persona en la empresa.
- Monitoreo permite conectar fuentes de datos en vivo, como bases de datos, para que los tableros se actualicen automáticamente. En el caso del análisis de delitos, se podría monitorear la situación de seguridad en tiempo real.

Enlace y Capturas de Pantalla del Tablero de Google Data Studio

La Distribución de Delitos por Estado



- Identificación de Patrones y Tendencias en un tablero de control con gráficos interactivos hace que sea mucho más sencillo identificar patrones, tendencias y valores atípicos.
- Ayuda a comunicar hallazgos de manera clara y concisa a diferentes audiencias a los equipos operativos o a stakeholders externos.

Enlace y Capturas de Pantalla del Tablero de Google Data Studio

Enlace Google Data Studio

https://lookerstudio.google.com/s/gdwgW_rD7hc

Conclusiones y Recomendaciones

- Durante el análisis del archivo delitos entre 2015 a 2025 se han transformaron los datos de formato ancho a largo, se limpiaron nulos, se estandarizaron los nombres de los meses y se creó una columna de fecha para facilitar el análisis temporal.
- Esto permitió visualizar la evolución mensual y anual de los delitos, particularmente aquellos relacionados con vehículos, que son clave para la evaluación de riesgos en aseguradoras.
- Se identificaron tendencias estacionales, con picos en meses específicos, y concentraciones geográficas claras, lo que indica que el riesgo no está distribuido de forma uniforme en todo el país.

Conclusiones y Recomendaciones

- El análisis de series de tiempo reveló que algunos estados mantienen tasas altas estables, mientras que otros experimentan aumentos o descensos graduales. Esto sugiere que existen factores locales, económicos, sociales o de seguridad pública que terminan impactando directamente la frecuencia de los delitos.
- También se recomienda implementar modelos predictivos (ARIMA, Prophet o machine learning) para anticipar cambios y detectar estados emergentes con riesgo creciente.
- Para mejorar el análisis, se propone integrar fuentes externas como indicadores socioeconómicos o datos de densidad vehicular, lo que permitiría explicar mejor las tendencias.

Conclusiones y Recomendaciones

- Sugiere automatizar el pipeline de análisis mediante dashboards en Google Data Studio y generar reportes ejecutivos con mapas de calor y gráficas interactivas. Esto facilitaría la actualización mensual y permitiría a los tomadores de decisiones identificar rápidamente cambios significativos.
- Profundizar en la segmentación por tipo de delito para poder separar claramente robo de auto estacionado, robo con violencia, fraude vehicular, etc. De esta forma detectar delitos que tengan mayor impacto financiero directo para aseguradoras.

Conclusiones y Recomendaciones

- Buscar Automatizar el Análisis de datos a través de Construir notebooks o dashboards automáticos que actualicen los datos mensualmente. Generar reportes ejecutivos que destaquen alertas y cambios significativos por estado.
- Con el fin de mejorar a visualización hay que tomar en cuenta incluir mapas de calor y gráficos interactivos para ubicar de forma visual los focos de riesgo y paneles comparativos que permitan a un tomador de decisiones ver qué estados mejoran o empeoran cada mes.