FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

Base de Datos

1C2024

Trabajo Práctico: ETL

Indice

Importación de Dataset

Para este trabajo práctico, seleccionamos una base de datos inmobiliaria disponible en kaggle, que consta de más de un millón de filas de datos, en formato csv **"real_estate.csv"**. Este conjunto de datos contiene información sobre las ventas de propiedades, incluidos detalles como la fecha de venta, el precio, la ubicación y el tipo de propiedad.

Para la importación y el preprocesamiento de los datos, trabajamos sobre Google Colab con Python en un GoogleDrive.
Link: BDD-TP.ipynb

Exploracion inicial

En esta etapa, comenzamos analizando el tamaño de nuestro conjunto de datos, que consta de 11 columnas y, como mencionamos previamente, más de un millón de filas.

	Date Recorded	List Year	Town	Address	Assessed Value	Sale Amount	Sales Ratio	Property Type	Residential Type	Longitude	Latitude
0	2021-04-14	2020	Ansonia	323 BEAVER ST	133000.0	248400.0	0.5354	Residential	Single Family	-73.06822	41.35014
1	2021-05-26	2020	Ansonia	152 JACKSON ST	110500.0	239900.0	0.4606	Residential	Three Family	NaN	NaN
2	2021-09-13	2020	Ansonia	230 WAKELEE AVE	150500.0	325000.0	0.4630	Commercial	NaN	NaN	NaN
3	2020-12-14	2020	Ansonia	57 PLATT ST	127400.0	202500.0	0.6291	Residential	Two Family	NaN	NaN
4	2021-09-07	2020	Avon	245 NEW ROAD	217640.0	400000.0	0.5441	Residential	Single Family	NaN	NaN

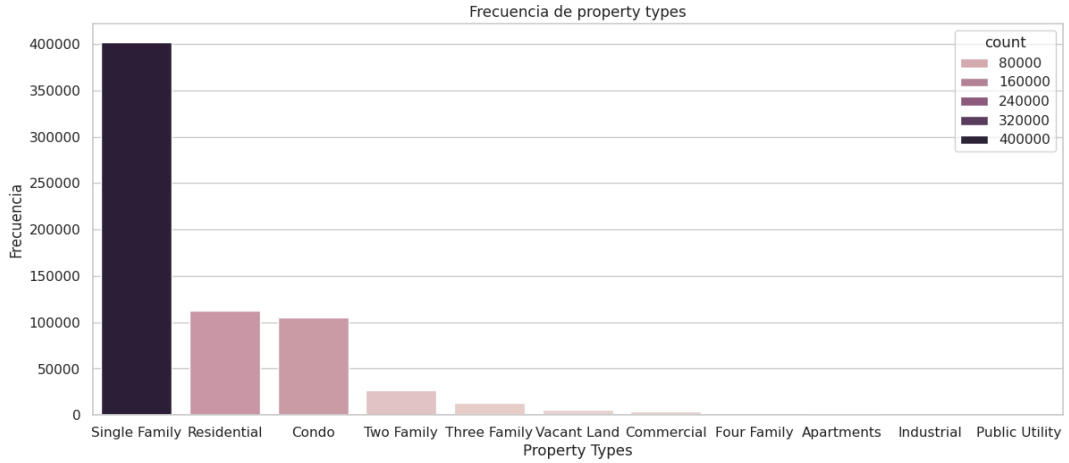
Cada columna indica lo siguiente:

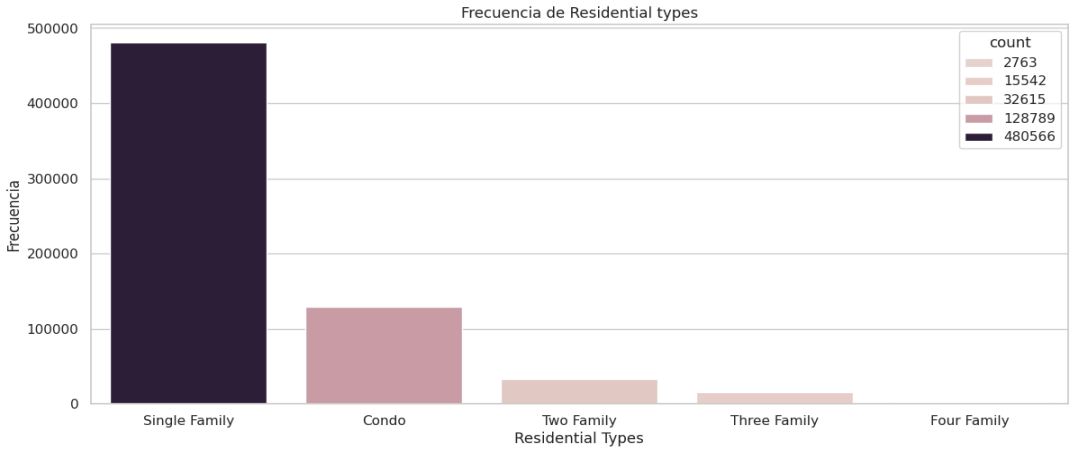
- Date Recorded: La fecha en la que se registró la compra de propiedad
- List Year: El año en que la propiedad fue listada.
- Town: El nombre de la ciudad o pueblo donde se encuentra la propiedad.
- Address: La dirección específica de la propiedad
- Assessed Value: El valor evaluado de la propiedad.
- Sale Amount: El monto por el cual se vendió la propiedad.
- Sales Ratio: La proporción entre el valor de venta y el valor evaluado.
- Property Type: El tipo de propiedad (por ejemplo, comercial, residencial, industrial)
- Residential Type: El tipo de propiedad residencial (por ejemplo, casa unifamiliar, departamento).
- Longitude: La longitud geográfica de la propiedad.
- Latitude: La latitud geográfica de la propiedad.

A continuación, métricas a modo de resumen del dataset.

	List Year	Assessed Value	Sale Amount	Sales Ratio	Longitude	Latitude
count	1.054159e+06	1.054159e+06	1.054159e+06	1.054159e+06	254643.000000	254643.000000
mean	2.010774e+03	2.797416e+05	3.990286e+05	9.953241e+00	-72.878565	41.499377
std	6.540711e+00	1.650117e+06	5.229758e+06	1.838434e+03	0.446531	0.258100
min	2.001000e+03	0.000000e+00	0.000000e+00	0.000000e+00	-121.230910	34.345810
25%	2.004000e+03	8.845000e+04	1.422000e+05	4.816008e-01	-73.198040	41.292266
50%	2.011000e+03	1.395800e+05	2.300000e+05	6.162887e-01	-72.900600	41.504259
75%	2.017000e+03	2.270000e+05	3.700000e+05	7.764000e-01	-72.633226	41.714357
max	2.021000e+03	8.815100e+08	5.000000e+09	1.226420e+06	-71.187550	44.934590

Después de analizar las métricas del dataset, vamos a mostrar la frecuencia de distintos tipos de propiedades y tipos residenciales.

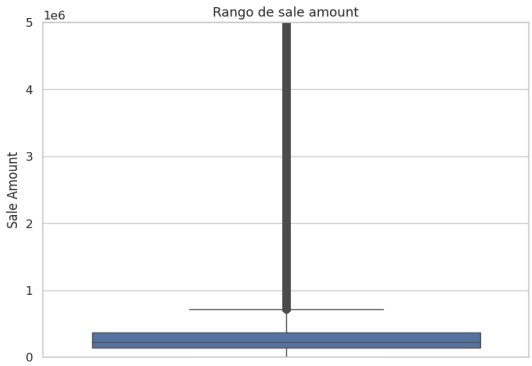




Como muestran las gráficas, el tipo de propiedad más frecuente es "***Single Family***", que también es el tipo residencial más común.

Al analizar las columnas de Town y Address llegamos a estas conclusiones:

1. Hay una gran cantidad de distintos tipos de ciudades (Town) y direcciones (Address).
2. Aunque haya gran cantidad de ciudades y direcciones distintas, ambas columnas contienen datos que se repiten.
3. En la columna de Town, encontramos una sola fila que marcó su ciudad como ***Unknown***, por lo tanto hemos decidido quitarlo de la dataset.



Luego, armamos un boxplot para hacer un análisis a simple vista de la columna de ventas (Sales Amount), donde llegamos a la conclusión de que se pueden observar que hay bastantes valores atípicos que abordaremos en dicha sección.

Finalmente, analizamos las columnas de latitud y longitud y concluimos que ambas contienen una gran cantidad de valores nulos. En la etapa de preprocesamiento de datos, veremos cómo tratar estos datos.

Preprocesamiento de los datos

En esta sección, nos dedicaremos al preprocesamiento de datos, analizando valores nulos o faltantes, datos fuera de rango, inconsistencias de formato y datos duplicados. Finalmente, aplicaremos nuestras modificaciones al conjunto de datos para optimizarlo y eliminar redundancias y datos irrelevantes.

Datos Nulos o Faltantes

En esta sección, analizamos la cantidad de datos faltantes en cada columna y llegamos a las siguientes conclusiones:
Las columnas con datos nulos son:

- Date
- Address
- Latitude
- Longitude
- Residential Type
- Property Type

Para las columnas **Date** y **Address**, debido a la escasa cantidad de valores nulos, decidimos eliminar las filas nulas, ya que no afectarán significativamente al conjunto de datos.

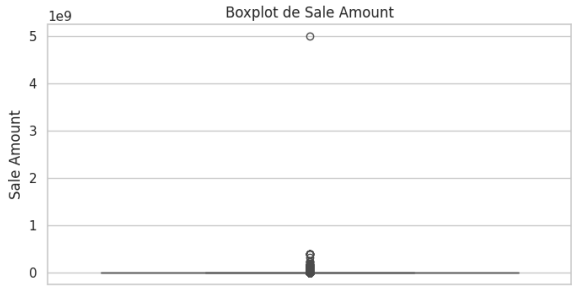
Las columnas **Latitude** y **Longitude** contienen una amplia cantidad de datos nulos, por lo que decidimos descartarlas por completo.

A pesar de la considerable cantidad de datos nulos en **Residential Type** y **Property Type**, decidimos conservar estas columnas y analizar su integridad más adelante.

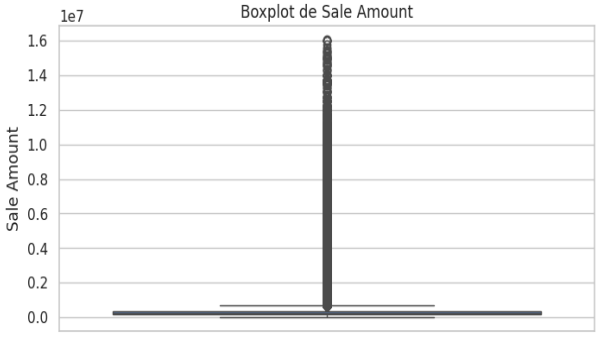
Datos Fuera de Rango

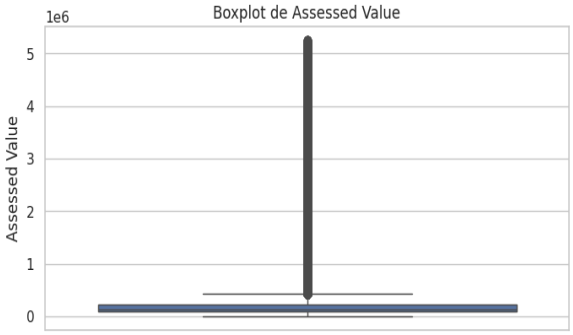
En esta sección de código se implementó las funciones necesarias para aplicar un filtro sobre la información del dataset, donde se quitan registros que se consideran fuera de rango, utilizando la métrica de z-score, y utilizando un umbral de referencia de valor absoluto 3 (que esté entre -3 y +3), además de ciertos datos analizados personalmente que se consideran un prefiltro de la información innecesaria, específicamente que el valor de la propiedad sea mayor a 0.

Primero se aplica un análisis de datos sin filtrar, se espera que sea una visualización mediante boxplots de manera desprolija e inentendible (debido a la presencia de valores atípicos), a diferencia de la información ya filtrada próximamente:



Ahora se aplica el filtro a la base de datos explicado previamente en las declaraciones de las funciones.





Inconsistencias de Formatos

En esta sección, abordamos las inconsistencias de formato en nuestro conjunto de datos. Utilizamos los siguientes criterios para estandarizar los formatos de las columnas relevantes:

Convirtiendo 'Date Recorded' al formato de fecha:
Transformamos la columna Date Recorded al tipo de dato fecha y hora en el formato YYYY-MM-DD HH:MM:SS para garantizar una representación consistente y facilitar el análisis temporal.

Transformar 'Town', 'Property Type', y 'Residential Type' de object a string:
Convertimos estas columnas al tipo de dato string para asegurar que todos los valores se traten uniformemente como cadenas de texto, eliminando posibles inconsistencias derivadas del tipo de dato object.

Transformar 'List Year' a tipo int:
Convertimos la columna List Year al tipo de dato int (entero) para asegurar que los años se manejen como valores numéricos, permitiendo operaciones matemáticas y comparaciones precisas.

Estas transformaciones nos permiten estandarizar el formato de las columnas y asegurarnos de que los datos sean consistentes y adecuados para el análisis posterior.

Datos duplicados

En esta sección, analizamos la cantidad de filas duplicadas en nuestro conjunto de datos. Dado que cada venta debe ser única, no tiene sentido que haya valores duplicados; para que dos filas sean idénticas, deben coincidir en cada columna. Es bastante improbable que haya datos duplicados para cada venta.

En caso de encontrar duplicados, decidimos descartarlos del conjunto de datos. Además, concluimos que la cantidad de filas duplicadas es inferior al 2% de nuestro conjunto de datos. Por lo tanto, eliminar estos duplicados no afectará negativamente la integridad de la base de datos.

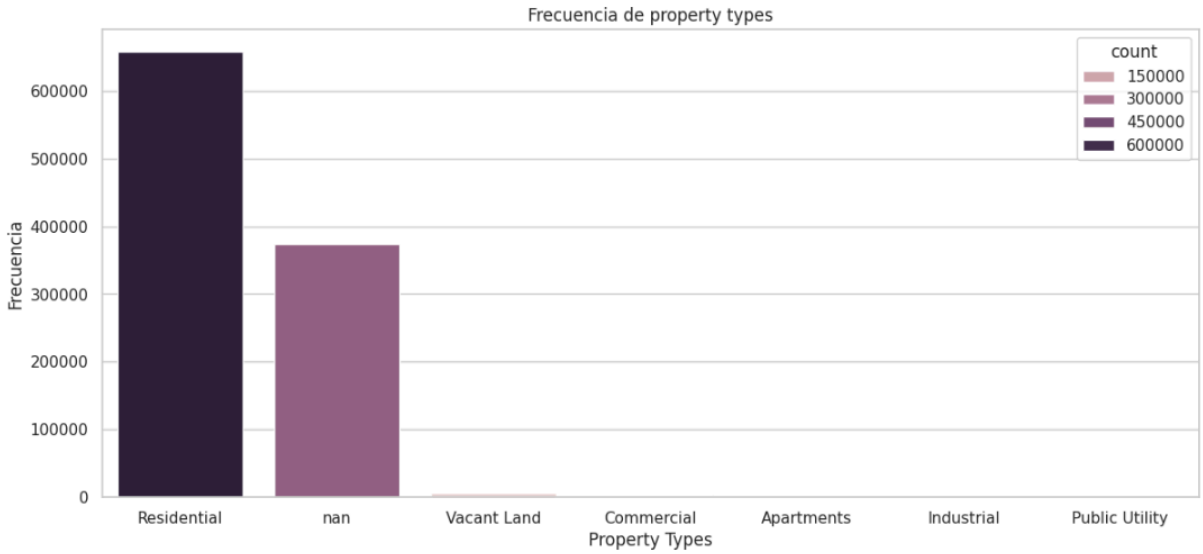
Transformación del Dataset

Previo a realizar la transformación del dataset teniendo en cuenta los analizado en el preprocesamiento de datos, haremos el manejo de la incongruencia entre property type y residential type mencionada en la exploración inicial.

Cambiaremos todos los property types que tengan alguno de los siguientes valores:

- Single Family
- Condo
- Two Family
- Three Family
- Four Family

a **Residential** y le asignaremos lo que tenía antes a su residential type y terminamos con lo siguiente:



Como casi todas las property types que están definidas son residential procedemos a dropear dicha columna y solo quedarnos con residential type,

Otra observación clave del análisis del dataset fue que la columna Sales Ratio se compone de Assessed Value / Sale Amount. Dado que Sales Ratio depende de estas columnas y es fácilmente calculable, decidimos descartarla.

Finalmente, nuestra dataset se verá de este tamaño:

- Cantidad de Filas: **1041780**
- Cantidad de Columnas: **7**

Columnas:

```
date_recorded        object
list_year             int64
town                 object
address              object
assessed_value      float64
sale_amount         float64
residential_type     object
```

Validación de Integridad de los Datos

Antes del análisis, realizamos una exploración inicial para revisar la información relevante del conjunto de datos, incluyendo la cantidad de columnas y filas, los tipos de columnas y la identificación de la información más importante para almacenar en la base de datos.

Luego de la exploración inicial, analizamos lo siguiente:

1. **Análisis de datos nulos**:
    - Eliminamos las columnas de ***Latitud y Longitud***, ya que eran casi 100% nulas.
    - Eliminamos los registros con fechas y direcciones nulas debido a su baja cantidad.
2. **Análisis de datos fuera de rango (outliers)**:
    - Filtramos valores negativos o cero en las propiedades.
    - Utilizamos el Z-score para eliminar valores atípicos, considerando un valor absoluto mayor que 3 como umbral.
3. **Inconsistencia de formatos**:
    - Convertimos ***Date Recorded*** a formato de fecha.
    - Aseguramos que ***Town, Property Type y Residential Type*** sean de tipo string.
4. **Datos duplicados**:
    - Eliminamos el 2% de los valores duplicados, ya que su eliminación no afecta negativamente la base de datos.
    - Observamos que ***Sales Ratio*** se puede calcular como ***Assessed Value / Sale Amount***, por lo que no es necesario almacenarlo.

Creación de Base de Datos

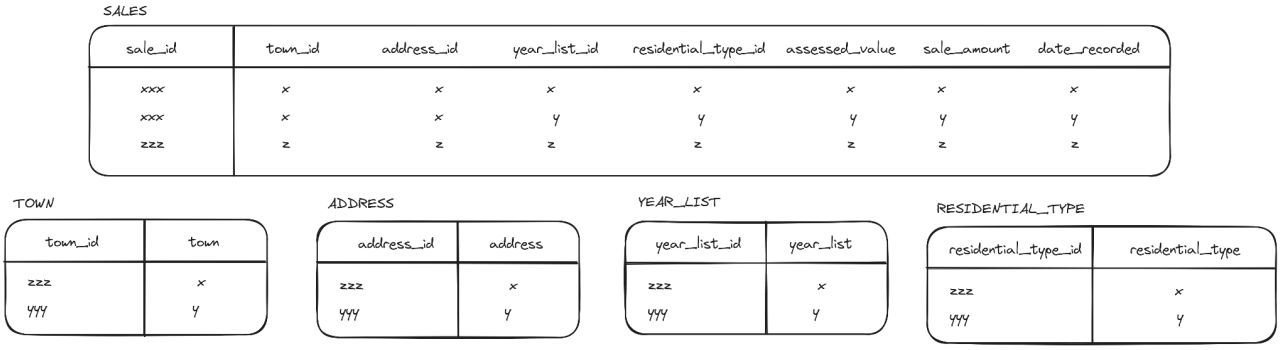
Finalmente, con el uso de SQLITE3, vamos a crear la base de datos sin normalizar, cuyo resultado se verá así:

	date_recorded	list_year	town	address	assessed_value	sale_amount	residential_type
0	2021-04-14 00:00:00	2020	Ansonia	323 BEAVER ST	133000.0	248400.0	Single Family
1	2021-05-26 00:00:00	2020	Ansonia	152 JACKSON ST	110500.0	239900.0	Three Family
2	2021-09-13 00:00:00	2020	Ansonia	230 WAKELEE AVE	150500.0	325000.0	nan
3	2020-12-14 00:00:00	2020	Ansonia	57 PLATT ST	127400.0	202500.0	Two Family
4	2021-09-07 00:00:00	2020	Avon	245 NEW ROAD	217640.0	400000.0	Single Family
...	...	...	...	...	...	...	...
1041775	2021-11-16 00:00:00	2021	Watertown	50 SUMMIT RIDGE	263100.0	430000.0	Single Family
1041776	2022-09-20 00:00:00	2021	Woodbury	89 TAMARACK LANR UNIT 89A	79810.0	200000.0	Condo
1041777	2022-05-06 00:00:00	2021	Woodbury	69 BACON POND ROAD	79590.0	360000.0	nan
1041778	2022-06-29 00:00:00	2021	West Haven	114 TUTHILL ST	117600.0	275000.0	Single Family
1041779	2022-04-26 00:00:00	2021	Windsor	200 BLOOMFIELD AVE	130690.0	190000.0	nan

1041780 rows × 7 columns

Normalizacion de Base de Datos

La necesidad de normalizar surge porque nuestra base de datos sin normalizar contiene una gran cantidad de valores repetidos, lo que la hace difícil de entender y de consultar. Procedemos a normalizar la base de datos para evitar información redundante, facilitar las relaciones y simplificar las consultas.



A continuación se puede ver un diagrama de cómo debería ser la base de datos una vez normalizada.

Esta forma de normalización es a través de la forma Normal de Boyce-Codd (BCNF) ya que cumple con los siguientes requisitos:

- Cada determinante no trivial debe ser una superclave, lo que significa que si un atributo determina otro atributo, entonces debe ser una clave candidata. Además, debe cumplir con las formas normales 1, 2, y 3

- *Debe cumplir con la primera forma normal (1FN):* Todos los atributos de una tabla deben ser atómicos, es decir, no deben tener valores repetidos ni multivaluados.

- *Debe cumplir con la segunda forma normal (2FN):* Debe cumplir con la 1FN y además, todos los atributos no clave deben depender completamente de la clave primaria.

- *Debe cumplir con la tercera forma normal (3FN):* Debe cumplir con la 2FN y además, no debe haber dependencias transitivas.

Verificación de Base de Datos

Verificaremos que los datos han sido insertados correctamente y que la base de datos está en un estado consistente.

Ejecutamos la siguiente consulta para obtener un dataframe que coincida con el estado inicial del dataset al comienzo de esta sección

```
query = '''
  SELECT
    town.name AS town,
    address.address AS address,
    list_year.year AS list_year,
    residential_type.r_type AS residential_type,
    sales.sale_amount,
    sales.assessed_value,
    sales.date_recorded
  FROM sales
  LEFT JOIN town ON sales.town_id = town.id
  LEFT JOIN address ON sales.address_id = address.id
  LEFT JOIN list_year ON sales.list_year_id = list_year.id
  LEFT JOIN residential_type ON sales.residential_type_id = residential_type.id
  WHERE town_id = 1 AND address_id = 1
'''
verif_df = pd.read_sql_query(query, conn)
verif_df
```

El resultado final quedo asi: [esto es la muestra de una sola fila]

	town	address	list_year	residential_type	sale_amount	assessed_value	date_recorded
0	Ansonia	323 BEAVER ST	2020	Single Family	248400.0	133000.0	2021-04-14 00:00:00

Que coincide con como tuvimos el dataset previo a cargar los datos.