

Import Library


```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.linear_model import LinearRegression
```

```
import sys
```

```
if not sys.warnoptions:
    import warnings
    warnings.simplefilter("ignore")
```


✓ Soal 1 : Identifikasi Kebutuhan Data**✓ Memuat dan Mendapatkan Info Umum Data**

```
df = pd.read_csv("Copy of Real_Estate_Sales_2001-2020_GL.csv", delimiter=";")
df
```



	Serial Number	List Year	Date Recorded	Town	Address	Assessed Value	Sale Amount	Sales Ratio	Property Type	Residential Type	Non Use Code	Assessor Remarks
0	2020348.0	2020.0	09/13/2021	Ansonia	230 WAKELEE AVE	150500.0	325000.0	0.463	Commercial	NaN	NaN	
1	20002.0	2020.0	10/02/2020	Ashford	390 TURNPIKE RD	253000.0	430000.0	0.5883	Residential	Single Family	NaN	
2	200212.0	2020.0	03/09/2021	Avon	5 CHESTNUT DRIVE	130400.0	179900.0	0.7248	Residential	Condo	NaN	
3	200243.0	2020.0	04/13/2021	Avon	111 NORTHINGTON DRIVE	619290.0	890000.0	0.6958	Residential	Single Family	NaN	
4	200377.0	2020.0	07/02/2021	Avon	70 FAR HILLS DRIVE	862330.0	1447500.0	0.5957	Residential	Single Family	NaN	
...
996556	190272.0	2019.0	06/24/2020	New London	4 BISHOP CT	60410.0	53100.0	1.137.664.783	Single Family	Single Family	14 - Foreclosure	

```
df.info()
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 996561 entries, 0 to 996560
Data columns (total 14 columns):
#   Column              Non-Null Count  Dtype
---  -
0   Serial Number       996559 non-null float64
1   List Year           996559 non-null float64
2   Date Recorded       996557 non-null object
3   Town                996559 non-null object
4   Address              996511 non-null object
5   Assessed Value      996506 non-null float64
6   Sale Amount         996506 non-null float64
7   Sales Ratio         996506 non-null object
8   Property Type       614271 non-null object
9   Residential Type    608423 non-null object
10  Non Use Code        289433 non-null object
11  Assessor Remarks    149696 non-null object
12  OPM remarks         9917 non-null  object
13  Location            197185 non-null object
dtypes: float64(4), object(10)
memory usage: 106.4+ MB
```

✓ A. Asset Data Organisasi dan Aspek Pengelolaannya

Asset Data Organisasi

- **File Data:** CSV berukuran 106,2 MB
- **Jumlah Data:** 996.561 baris, 14 kolom

Kolom Data:

- `Serial Number (float64)`: ID properti
- `List Year (float64)`: Tahun terdaftar
- `Date Recorded (object)`: Tanggal pencatatan data
- `Town (object)`: Nama kota
- `Address (object)`: Alamat properti
- `Assessed Value (float64)`: Nilai taksiran properti
- `Sale Amount (float64)`: Harga jual properti
- `Sales Ratio (object)`: Rasio antara Assessed Value terhadap Sale Amount
- `Property Type (object)`: Tipe properti
- `Residential Type (object)`: Tipe hunian
- `Non Use Code (object)`: Kode penggunaan non-standar
- `Assessor Remarks (object)`: Catatan dari penilai
- `OPM Remarks (object)`: Catatan tambahan
- `Location (object)`: Lokasi properti

Aspek Pengelolaan Data

Storage

- Data disimpan dalam format file `.csv`.

Data Quality

- Banyak missing value, terutama pada kolom `Property Type`, `Residential Type`, `Non Use Code`, `Assessor Remarks`, `OPM Remarks`, dan `Location`.
- Kolom dengan missing value di atas 60% disarankan untuk dihapus.
- Kesalahan penghitungan pada nilai `Sales Ratio`, perlu dihitung ulang.
- Tipe data belum sepenuhnya sesuai:
 - `Date Recorded` harusnya `datetime`, tapi saat ini `object`.
 - `Sales Ratio` harusnya numerik, tapi bertipe `object`.
 - `Location` seharusnya berupa data point, tetapi masih bertipe `object`.

Data Security

- Karena memuat informasi alamat dan properti, maka perlindungan privasi dan akses data harus diatur.

Governance

- Perlu adanya dokumentasi tentang arti masing-masing kolom (data dictionary).
- Perlu pencatatan perubahan data (audit trail).

Access Control

- Pengaturan hak akses diperlukan (hanya pihak tertentu yang boleh melihat/memproses data).

✓ B. Identifikasi Tipe Model Operasional Data

Model operasional data berarti bagaimana data ini digunakan dalam operasional sehari-hari. Untuk dataset ini:

Tipe Model Operasional

- **Transactional Data**
Data mencatat kejadian nyata, yaitu penjualan properti (`Sale Amount`, `Assessed Value`, `Date Recorded`).
- **Master Data**
Data seperti `Town`, `Address`, `Property Type`, `Residential Type` menggambarkan entitas tetap (properti).

- **Reference Data**

Non Use Code kemungkinan memiliki daftar kode standar (kategori atau penggunaan properti).

- **Analytical Data**

Data digunakan untuk analitik, seperti menghitung Sales Ratio.

✓ Kesimpulan Ringkas

Aspek	Detail
Aset Data	Data properti, transaksi penjualan, atribut properti.
Format Penyimpanan	CSV file, 106,2 MB.
Data Quality Issue	Missing values, tipe data tidak konsisten.
Model Operasional Data	Transactional, Master, Reference, dan Analytical Data.
Risiko / Fokus	Data privacy, data quality improvement, pengelolaan hak akses.

✓ Soal 2 : Mengambil Data

✓ A. Tujuan Teknis Analisis Data

Beberapa tujuan teknis analisis data yang perlu dipertimbangkan adalah:

- Tools untuk memuat dan mengolah data dengan ukuran cukup besar (di atas 100 MB).
- Tools yang bisa digunakan untuk (sekaligus, memudahkan proses pengerjaan tanpa multiplatform):
 - Data cleaning dan data transformation
 - Data visualization
 - Data modelling, khususnya time series modelling

Berdasarkan beberapa pertimbangan tersebut, maka tools yang dipilih adalah bahasa pemrograman **Python** dengan beberapa library berikut:

✓ B. Tools dan Kegunaannya

- **Pandas**

Membaca file .csv, melakukan data cleaning, dan transformasi data (ETL).

- **NumPy**

Melakukan operasi numerik dan manipulasi array, serta mendukung manipulasi data di Pandas.

- **Matplotlib**

Digunakan untuk visualisasi data dasar seperti plotting grafik.

- **Seaborn**

Digunakan untuk visualisasi data tingkat lanjut dan membuat grafik yang lebih menarik.

- **Scikit-learn**

Digunakan untuk pemodelan time series sederhana.

- **Datetime Library**

Membantu dalam mengelola dan membersihkan kolom tanggal (Date Recorded).

✓ Soal 3: Mengintegrasikan Data

Dalam Analisis Data, mengintegrasikan data berarti melakukan penggabungan (merge, join, union) untuk beberapa tabel yang dibutuhkan (jika memiliki data lebih dari satu tabel) atau melakukan pemeriksaan integritas data (jika atau meskipun data hanya dalam satu tabel). Integritas data menjadimn data yang digunakan lengkap, konsisten, valid dan akurasi tinggi sesuai tujuan analisis. Karena data pada kasus ini hanya berupa datu file .csv, maka perlu dilakukan pengecekan integritas data di antaranya:

- Memeriksa kelengkapan data
- Memeriksa adanya kemungkinan duplikasi data
- Standarisasi tipe data

- Memastikan keakuratan nilai
- Memeriksa kemungkinan adanya outlier

✓ Memeriksa kelengkapan data (Cek Missing Value)

Pada bagian ini dicek berapa banyak data yang hilang di setiap kolom data dengan hasil sebagai berikut:

```
# Hitung jumlah missing dan persentase missing
missing_data = pd.DataFrame({
    'Missing Count': df.isnull().sum(),
    'Missing Percentage': (df.isnull().sum() / len(df)) * 100
})
```

missing_data

	Missing Count	Missing Percentage
Serial Number	2	0.000201
List Year	2	0.000201
Date Recorded	4	0.000401
Town	2	0.000201
Address	50	0.005017
Assessed Value	55	0.005519
Sale Amount	55	0.005519
Sales Ratio	55	0.005519
Property Type	382290	38.360923
Residential Type	388138	38.947741
Non Use Code	707128	70.956821
Assessor Remarks	846865	84.978742
OPM remarks	986644	99.004878
Location	799376	80.213454

Berdasarkan hasil ini maka kolom-kolom dengan missing value > 60 % perlu dihapus yaitu Non Use Code, Assessor Remarks, OPM remarks, dan Location

Berikut adalah snippet code menghapus kolom-kolom yang tidak dibutuhkan:

```
df_dropcolumn = df.drop(columns=['Non Use Code', 'Assessor Remarks', 'OPM remarks', 'Location'])
```

Kemudian hapus baris data yang mengandung NaN

```
df_dropna = df_dropcolumn.dropna()
```

```
df_dropna.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 608419 entries, 1 to 996559
Data columns (total 10 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Serial Number         608419 non-null float64
1   List Year              608419 non-null float64
2   Date Recorded         608419 non-null object
3   Town                  608419 non-null object
4   Address               608419 non-null object
5   Assessed Value        608419 non-null float64
6   Sale Amount           608419 non-null float64
7   Sales Ratio           608419 non-null object
8   Property Type         608419 non-null object
9   Residential Type      608419 non-null object
dtypes: float64(4), object(6)
```

memory usage: 51.1+ MB

✓ Memeriksa adanya kemungkinan duplikasi data

Duplikasi data bisa menghasilkan analisis yang tidak valid, perlu dilakukan cek duplikasi data, pada python dapat dilakukan dengan cara yang sederhana, berikut snippet codenya:

```
# Cek apakah ada baris duplikat
print(df_dropna.duplicated().sum())

# Tampilkan baris duplikat
df_dropna[df_dropna.duplicated()]
```

 0

Serial Number	List Year	Date Recorded	Town	Address	Assessed Value	Sale Amount	Sales Ratio	Property Type	Residential Type
---------------	-----------	---------------	------	---------	----------------	-------------	-------------	---------------	------------------

Dari hasil di atas tidak terdapat baris kosong yang dideteksi sebagai duplicate.

```
df_dropduplicate = df_dropna.drop_duplicates()
```

✓ Standarisasi tipe data

Dari data df awal perlu adanya standarisasi tipe data yaitu:

Date Recorded harusnya datetime, tapi bertipe object

List Year harusnya integer, tapi bertipe float

Sales Ratio harusnya numerik, tapi bertipe object

Berikut adalah kode python yang digunakan:

```
# Konvert data datetime
df_dropduplicate['Date Recorded'] = pd.to_datetime(df_dropduplicate['Date Recorded'])

df_dropduplicate['List Year'] = df_dropduplicate['List Year'].astype(int)

# Bersihkan tanda titik di kolom Sales Ratio
df_dropduplicate['Sales Ratio'] = df_dropduplicate['Sales Ratio'].str.replace('.', '', regex=False)

# Baru ubah jadi float
df_dropduplicate['Sales Ratio'] = df_dropduplicate['Sales Ratio'].astype('float')
```

Kemudian dilakukan recalculate untuk Sales Ratio dengan formula

Sales Ratio = Assessed Value / Sale Amount untuk memastikan hasil yang valid

```
df_dropduplicate['Sales Ratio'] = round(df_dropduplicate['Assessed Value']/df_dropduplicate['Sale Amount'],3)
```

Berikut hasil standarisasi tipe data:

```
df_dropduplicate.info()
```



```
<class 'pandas.core.frame.DataFrame'>
Index: 608419 entries, 1 to 996559
Data columns (total 10 columns):
#   Column          Non-Null Count  Dtype
---  -
0   Serial Number    608419 non-null float64
1   List Year        608419 non-null int64
2   Date Recorded    608419 non-null datetime64[ns]
3   Town            608419 non-null object
4   Address         608419 non-null object
5   Assessed Value   608419 non-null float64
```

```

6  Sale Amount      608419 non-null float64
7  Sales Ratio      608419 non-null float64
8  Property Type    608419 non-null object
9  Residential Type  608419 non-null object
dtypes: datetime64[ns](1), float64(4), int64(1), object(4)
memory usage: 51.1+ MB

```

✓ Memastikan keakuratan nilai

Hal ini bertujuan untuk memastikan validitas, konsistensi, dan keandalan data, pada python dapat dilakukan dengan cara yang sederhana, berikut snippet codenya:

```
df_dropduplicate[df_dropduplicate['Sale Amount']<0]
```

```

Serial Number  List Year  Date Recorded  Town  Address  Assessed Value  Sale Amount  Sales Ratio  Property Type  Residential Type

```

```
df_dropduplicate[df_dropduplicate['Assessed Value']<0]
```

```

Serial Number  List Year  Date Recorded  Town  Address  Assessed Value  Sale Amount  Sales Ratio  Property Type  Residential Type

```

```
df_dropduplicate['Date Recorded'].min()
```

```
Timestamp('1970-10-22 00:00:00')
```

```
df_dropduplicate['Date Recorded'].max()
```

```
Timestamp('2021-09-30 00:00:00')
```

Dari hasil di atas

Sale Amount dan Assessed Value tidak ada yang < 0

Data Recorded dimulai pada 22 Oktober 1970 dan berakhir pada 30 September 2021

Hal ini mengindikasikan bahwa tidak ada data yang mencurigakan.

✓ Memeriksa kemungkinan adanya outlier

Outlier merupakan nilai yang jauh dari data kebanyakan, pada penghitungan rata-rata, outlier bisa membuat nilai ini menjadi tidak representatif terhadap data yang sesungguhnya, maka perlu dilakukan pengecekan outlier. Untuk mengecek adakah outlier di dalam data, metode sederhananya adalah dengan membuat Box Plot untuk kolom Sale Amount. Hasilnya sebagai berikut:

```

# Atur ukuran plot
plt.figure(figsize=(10, 5))

# Buat boxplot Sale Amount
sns.boxplot(x=df_dropduplicate['Sale Amount'])

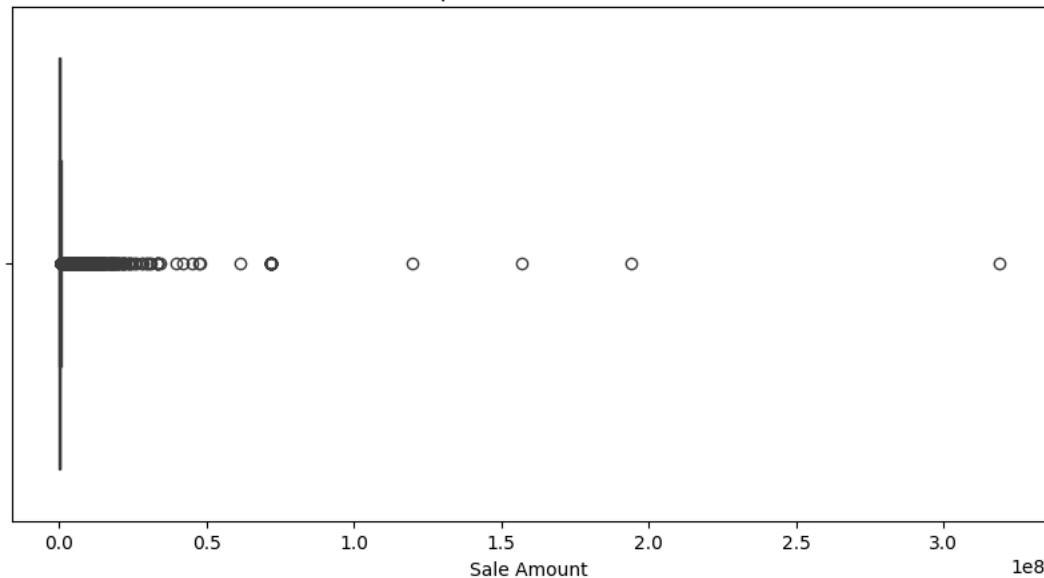
# Tambahkan judul
plt.title('Boxplot of Sale Amount')

# Tampilkan plot
plt.show()

```



Boxplot of Sale Amount



Gambar di atas merupakan boxplot dari Sale Amount yang menunjukkan adanya beberapa nilai outlier. Sebagian besar data terkonsentrasi di nilai yang relatif kecil, terlihat dari kotak dan garis vertikal (whisker) yang sangat rapat di dekat nol. Terdapat beberapa titik di sebelah kanan yang tersebar jauh dari distribusi utama. Hal ini mengindikasikan adanya nilai-nilai penjualan yang sangat tinggi dan tidak umum dibandingkan mayoritas data, yang bisa jadi perlu ditelusuri lebih lanjut apakah merupakan kesalahan data atau transaksi besar yang valid.

Kemudian, untuk menghapus outlier menggunakan metode IQR. Metode IQR (Interquartile Range) adalah teknik statistik sederhana untuk mendeteksi outlier dengan menggunakan rentang antara kuartil pertama (Q1) dan kuartil ketiga (Q3) dalam suatu data. IQR dihitung sebagai $IQR = Q3 - Q1$, lalu data dianggap outlier jika nilainya berada di bawah $Q1 - 1.5 \times IQR$ atau di atas $Q3 + 1.5 \times IQR$. Metode ini efektif untuk data yang tidak terdistribusi normal karena tidak bergantung pada rata-rata dan simpangan baku.

```
# # Histogram untuk melihat distribusi
# plt.figure(figsize=(10, 5))
# sns.histplot(df_dropduplicate['Sale Amount'], bins=100, kde=True)
# plt.title('Distribution of Sale Amount')
# plt.show()
```

Menghapus outlier dengan metode IQR berikut snippet kode python yang digunakan:

```
def remove_outliers_iqr(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    print(f"Outlier bounds untuk {column}: [{lower_bound}, {upper_bound}]")
    df_no_outlier = df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]
    print(f"Jumlah data setelah buang outlier: {df_no_outlier.shape[0]}")
    return df_no_outlier
```

```
df_no_outlier = remove_outliers_iqr(df_dropduplicate, 'Sale Amount')
```



```
Outlier bounds untuk Sale Amount: [-180000.0, 700000.0]
Jumlah data setelah buang outlier: 558211
```

```
df_no_outlier
```

	Serial Number	List Year	Date Recorded	Town	Address	Assessed Value	Sale Amount	Sales Ratio	Property Type	Residential Type
1	20002.0	2020	2020-10-02	Ashford	390 TURNPIKE RD	253000.0	430000.0	0.588	Residential	Single Family
2	200212.0	2020	2021-03-09	Avon	5 CHESTNUT DRIVE	130400.0	179900.0	0.725	Residential	Condo
6	2020180.0	2020	2021-03-01	Berlin	1539 FARMINGTON AVE	234200.0	130000.0	1.802	Residential	Two Family
7	2020313.0	2020	2021-07-01	Berlin	216 WATCH HILL RD	412000.0	677500.0	0.608	Residential	Single Family
9	20139.0	2020	2020-12-16	Bethel	16 DEEPWOOD DRIVE	171360.0	335000.0	0.512	Residential	Single Family
...
996554	19921.0	2019	2019-11-18	West Haven	75 CLOVER ST	125230.0	246000.0	0.509	Single Family	Single Family
996556	190272.0	2019	2020-06-24	New London	4 BISHOP CT	60410.0	53100.0	1.138	Single Family	Single Family
996557	190284.0	2019	2019-11-27	Waterbury	126 PERKINS AVE	68280.0	76000.0	0.898	Single Family	Single Family
				Windeor						

Windes

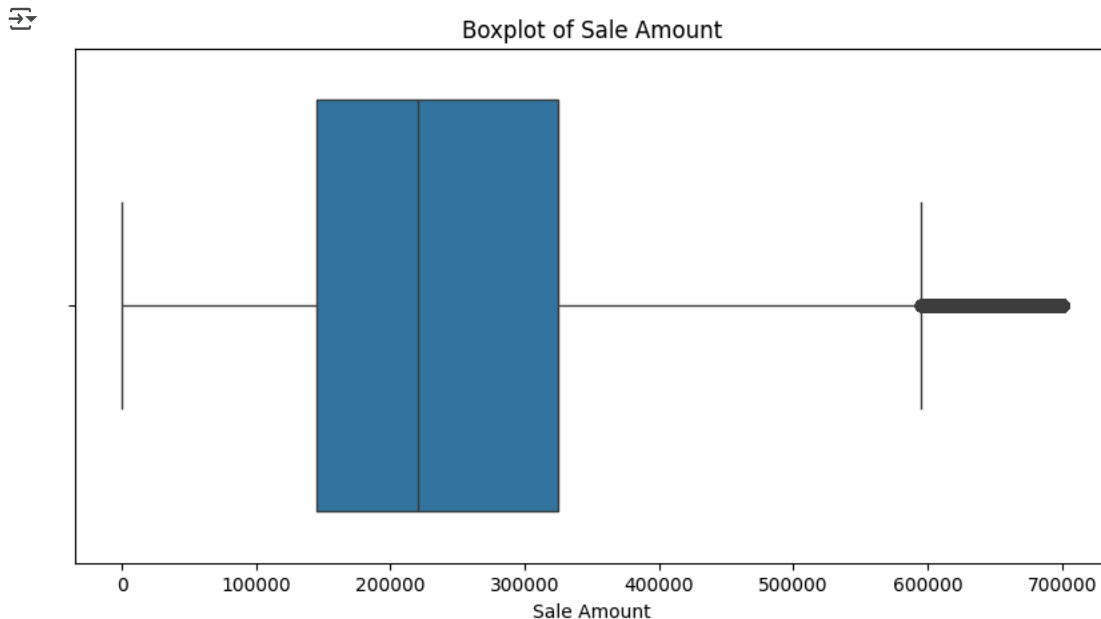
Setelah dihapus, berikut hasil box plot nya:

```
# Atur ukuran plot
plt.figure(figsize=(10, 5))

# Buat boxplot Sale Amount
sns.boxplot(x=df_no_outlier['Sale Amount'])

# Tambahkan judul
plt.title('Boxplot of Sale Amount')

# Tampilkan plot
plt.show()
```



Gambar boxplot terbaru menunjukkan distribusi Sale Amount setelah outlier dihapus. Kini data terlihat lebih seimbang, dengan rentang nilai yang lebih wajar dan tanpa nilai ekstrem yang mengganggu visualisasi. Kotak (IQR) dan whisker mencerminkan sebaran data yang lebih representatif terhadap mayoritas transaksi, serta tidak ada lagi titik-titik outlier yang terpisah jauh. Hal ini mengindikasikan bahwa pembersihan data berhasil menghilangkan nilai-nilai ekstrem yang sebelumnya mendistorsi skala dan interpretasi distribusi penjualan.

✓ Soal 4: Menelaah Data

✓ A. Melakukan Analisis Tipe dan Relasi Data

Berikut Analisis Tipe Data yang diperoleh dari hasil snippet code pyhton:

```
df_dropduplicate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 608419 entries, 1 to 996559
Data columns (total 10 columns):
#   Column              Non-Null Count  Dtype
---  -
0   Serial Number       608419 non-null float64
1   List Year           608419 non-null int64
2   Date Recorded       608419 non-null datetime64[ns]
3   Town                608419 non-null object
4   Address             608419 non-null object
5   Assessed Value      608419 non-null float64
6   Sale Amount         608419 non-null float64
7   Sales Ratio         608419 non-null float64
8   Property Type       608419 non-null object
9   Residential Type    608419 non-null object
dtypes: datetime64[ns](1), float64(4), int64(1), object(4)
memory usage: 51.1+ MB
```

Adapun relasi antar data sebagai berikut :

✓ 1. Analisis Korelasi Numerik

Kolom numerik:

Assessed Value

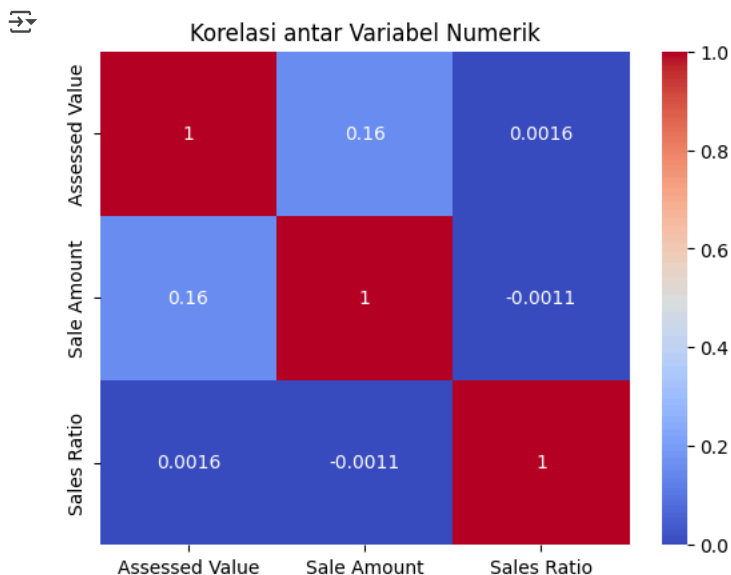
Sale Amount

Sales Ratio (setelah dibersihkan)

```
# Hitung korelasi
corr =df_dropduplicate[['Assessed Value', 'Sale Amount', 'Sales Ratio']].corr()
```

```
# Visualisasi heatmap
import seaborn as sns
import matplotlib.pyplot as plt
```

```
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Korelasi antar Variabel Numerik')
plt.show()
```



Visualisasi di atas adalah heatmap korelasi yang menunjukkan hubungan linear antara tiga variabel numerik: Assessed Value, Sale Amount, dan Sales Ratio. Warna merah cerah dan nilai mendekati 1 menunjukkan korelasi positif yang kuat, warna biru cerah dan nilai mendekati -1 menunjukkan korelasi negatif yang kuat, sedangkan warna di sekitar putih dan nilai mendekati 0 menunjukkan korelasi yang lemah. Terlihat bahwa Assessed Value dan Sale Amount memiliki korelasi positif yang lemah (0.16), mengindikasikan adanya kecenderungan kecil bahwa properti dengan nilai taksiran lebih tinggi juga memiliki harga jual yang lebih tinggi. Sementara itu, korelasi antara Assessed Value dengan Sales Ratio (0.0016) dan antara Sale Amount dengan Sales Ratio (-0.0011) sangat lemah, hampir mendekati nol, menunjukkan tidak adanya hubungan linear yang signifikan antara variabel-variabel tersebut.

2. Analisis Visualisasi Scatter/Boxplot

Scatter Plot Hubungan antar Variabel

a. Sale Amount vs. Assessed Value

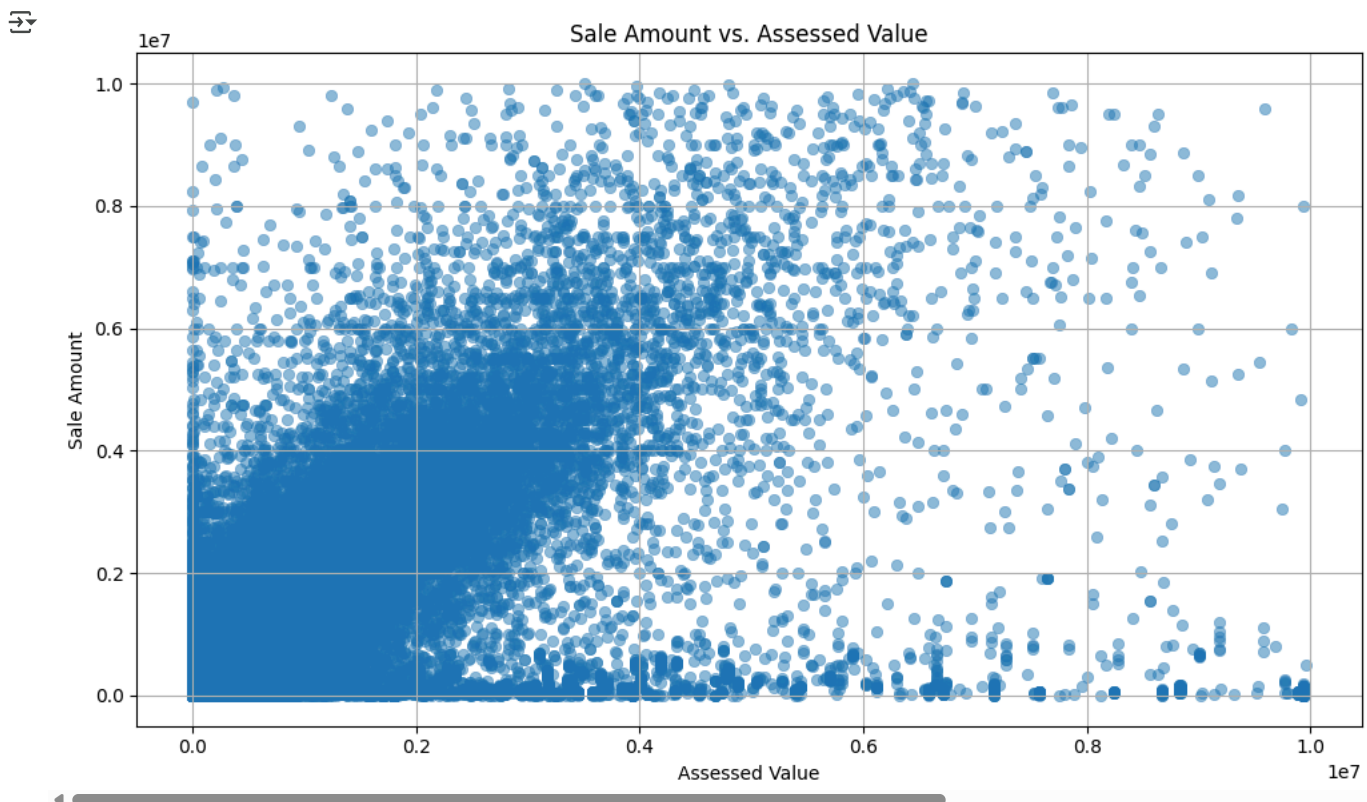
```
df_filtered = df[(df['Sale Amount'] < 1e7) & (df['Assessed Value'] < 1e7)]

# Atur ukuran plot
plt.figure(figsize=(10, 6))

# Scatter plot
sns.scatterplot(
    data=df_filtered,
    x='Assessed Value',
    y='Sale Amount',
    alpha=0.5, # transparansi agar tidak terlalu padat
    edgecolor=None
)

# Tambahkan judul dan label
plt.title('Sale Amount vs. Assessed Value')
plt.xlabel('Assessed Value')
plt.ylabel('Sale Amount')

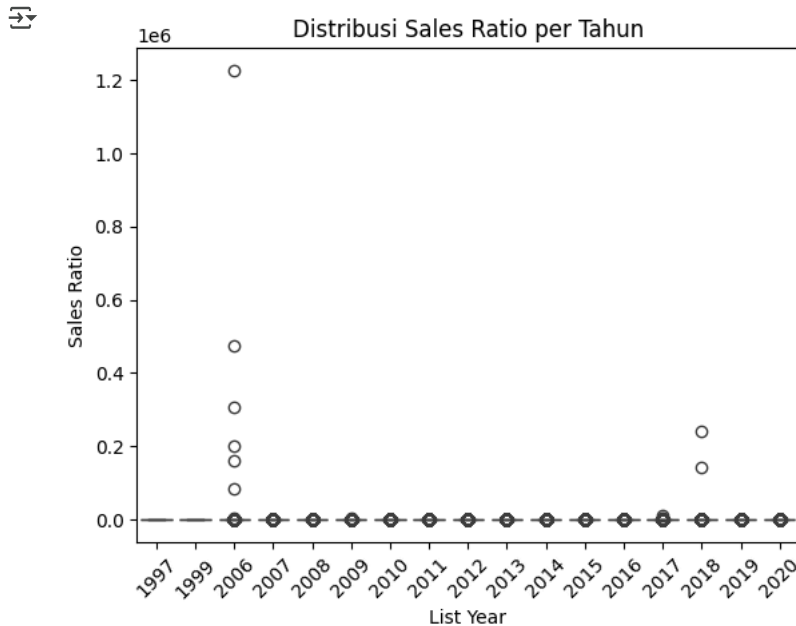
plt.grid(True)
plt.tight_layout()
plt.show()
```



Visualisasi di atas adalah scatter plot yang menggambarkan hubungan antara Nilai yang Dinilai (Assessed Value) dan Jumlah Penjualan (Sale Amount). Terlihat adanya konsentrasi titik yang padat di area nilai yang dinilai rendah hingga sekitar 0.2×10^7 dan jumlah penjualan yang juga relatif rendah hingga sekitar 0.4×10^7 . Seiring dengan peningkatan nilai yang dinilai, sebaran jumlah penjualan cenderung meluas, meskipun masih terdapat banyak transaksi dengan jumlah penjualan yang lebih rendah bahkan pada nilai yang dinilai tinggi. Pola ini mengindikasikan adanya korelasi positif yang lemah hingga sedang antara kedua variabel, di mana properti dengan nilai taksiran yang lebih tinggi cenderung memiliki harga jual yang lebih tinggi, namun terdapat variasi yang signifikan dan tidak semua properti dengan nilai taksiran tinggi terjual dengan harga yang sama tingginya

b. Sales Ratio vs. List Year

```
sns.boxplot(x='List Year', y='Sales Ratio', data=df_dropduplicate)
plt.xticks(rotation=45)
plt.title('Distribusi Sales Ratio per Tahun')
plt.show()
```

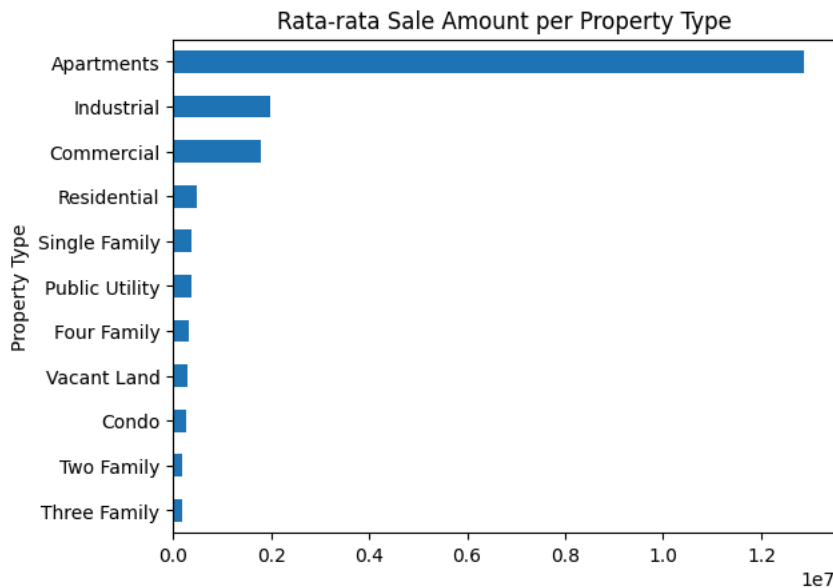


Visualisasi di atas adalah sekumpulan boxplot yang menampilkan distribusi rasio penjualan (Sales Ratio) untuk setiap tahun dari 1997 hingga 2020. Sebagian besar boxplot terlihat sangat terkompresi di dekat nilai nol, mengindikasikan bahwa mayoritas rasio penjualan berada di nilai yang rendah dan stabil di hampir setiap tahun. Namun, terdapat beberapa outlier signifikan yang muncul di beberapa tahun tertentu, terutama pada tahun 2006 dan 2018, di mana terdapat transaksi dengan rasio penjualan yang jauh lebih tinggi dibandingkan dengan transaksi lainnya pada tahun-tahun tersebut. Secara umum, variabilitas rasio penjualan di sekitar median cenderung rendah dari tahun ke tahun, dengan pengecualian adanya outlier sporadis yang menunjukkan adanya transaksi dengan harga jual yang jauh melampaui nilai taksirannya.

3. Analisis Kategori vs Numerik

Rata-rata Sale Amount per Property Type

```
df.groupby('Property Type')['Sale Amount'].mean().sort_values().plot(kind='barh')
plt.title('Rata-rata Sale Amount per Property Type')
plt.show()
```



Visualisasi di atas adalah grafik batang horizontal yang menampilkan rata-rata jumlah penjualan (Rata-rata Sale Amount) untuk berbagai tipe properti. Sumbu vertikal menunjukkan tipe properti, dan sumbu horizontal menunjukkan nilai rata-rata jumlah penjualan. Terlihat bahwa tipe properti "Apartments" memiliki rata-rata jumlah penjualan yang jauh lebih tinggi dibandingkan dengan tipe properti lainnya, mencapai lebih dari 1.2×10^7 . Diikuti oleh "Industrial" dan "Commercial" dengan rata-rata penjualan yang juga signifikan namun jauh di bawah "Apartments". Tipe properti residensial seperti "Residential", "Single Family", "Four Family", "Vacant Land", "Condo", "Two Family", dan "Three Family" memiliki rata-rata jumlah penjualan yang relatif rendah, di bawah 0.2×10^7 .

Analisis menunjukkan adanya disparitas yang besar dalam rata-rata harga jual antar tipe properti. Properti dengan kategori "Apartments", "Industrial", dan "Commercial" secara signifikan lebih mahal dibandingkan dengan properti yang diklasifikasikan sebagai residensial atau tanah kosong. Hal ini mengindikasikan adanya perbedaan nilai pasar yang substansial berdasarkan penggunaan dan jenis properti. Rata-rata harga yang rendah pada kategori residensial yang lebih spesifik seperti "Two Family" dan "Three Family" dibandingkan dengan "Apartments" juga menarik untuk diperhatikan, menunjukkan dinamika pasar yang berbeda untuk berbagai segmen properti.

✓ Ringkasan Tujuan Analisis tipe dan relasi data

Analisis Korelasi Numerik untuk menilai seberapa kuat keterkaitan antar angka

Analisis Visualisasi Scatter/Boxplot untuk menjelajahi pola dan outlier

Analisis Kategori vs Numerik untuk mendeteksi perbedaan antar grup

✓ B. Menentukan Karakteristik data yang terkumpul lalu sajikan dengan visualisasi grafik

Karakteristik Data yang Terkumpul:

DataFrame ini memiliki **608,419 entri** dan **10 kolom dengan tipe data** sebagai berikut:

Numerik:

Serial Number: Bilangan float [Kemungkinan adalah ID unik untuk setiap transaksi]

List Year: Bilangan integer. Tahun listing properti.

Assessed Value: Bilangan float. Nilai properti yang dinilai.

Sale Amount: Bilangan float. Jumlah penjualan properti.

Sales Ratio: Bilangan float. Rasio antara Sale Amount dan Assessed Value.

Waktu:

Date Recorded: Tipe datetime. Tanggal pencatatan transaksi.

Kategorikal (Objek):

Town: Nama kota atau wilayah.

Address: Alamat properti.

Property Type: Jenis properti (misalnya, Residential, Single Family, Condo).

Residential Type: Tipe residensial yang lebih spesifik (kemungkinan subkategori dari Property Type).

Adapun visualisasi grafik untuk menggambarkan karakteristik data:

✓ 1. Distribusi Data Numerik:

Histogram untuk Sale Amount: Menampilkan distribusi harga penjualan. Ini akan menunjukkan rentang harga yang paling umum, kemiringan distribusi, dan adanya outlier.

Histogram untuk Assessed Value: Menampilkan distribusi nilai properti yang dinilai.

Histogram untuk Sales Ratio: Menampilkan distribusi rasio penjualan. Ini penting untuk melihat apakah ada kecenderungan overvaluation atau undervaluation.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

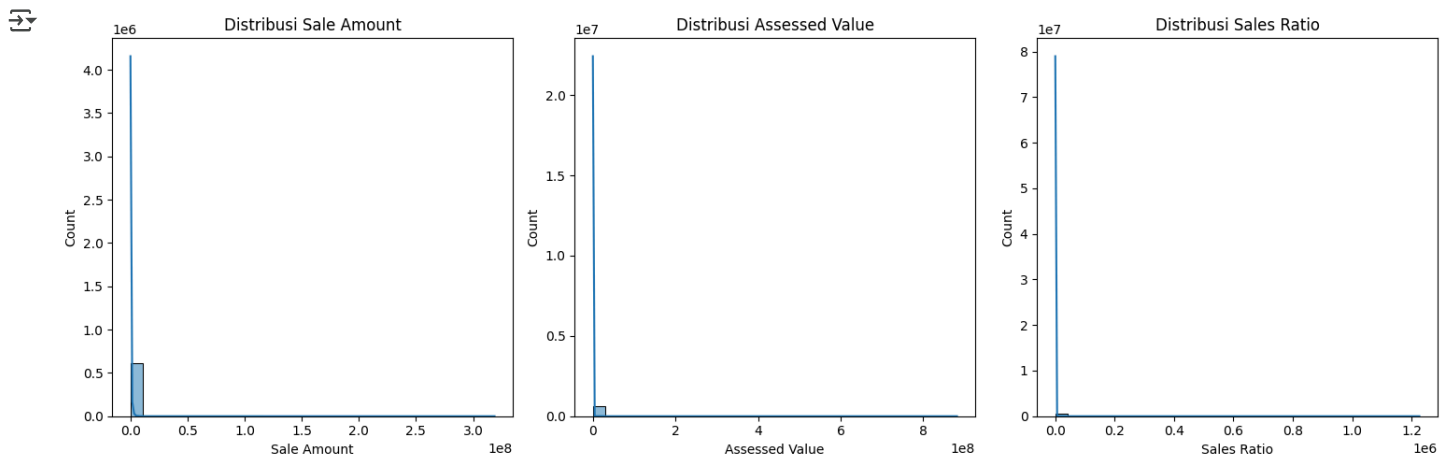
# 1. Distribusi Data Numerik
plt.figure(figsize=(15, 5))

plt.subplot(1, 3, 1)
sns.histplot(df_dropduplicate['Sale Amount'], kde=True, bins=30) # Atur jumlah bins
plt.title('Distribusi Sale Amount')

plt.subplot(1, 3, 2)
sns.histplot(df_dropduplicate['Assessed Value'], kde=True, bins=30) # Atur jumlah bins
plt.title('Distribusi Assessed Value')

plt.subplot(1, 3, 3)
sns.histplot(df_dropduplicate['Sales Ratio'], kde=True, bins=30) # Atur jumlah bins
plt.title('Distribusi Sales Ratio')

plt.tight_layout()
plt.show()
```



Visualisasi di atas menampilkan histogram distribusi untuk tiga variabel numerik: Sale Amount, Assessed Value, dan Sales Ratio. Pada distribusi Sale Amount dan Assessed Value, terlihat bahwa sebagian besar data terkonsentrasi pada nilai-nilai yang relatif rendah, dengan ekor yang memanjang ke kanan (right-skewed), mengindikasikan adanya sejumlah kecil transaksi dengan nilai yang sangat tinggi. Sementara itu, distribusi Sales Ratio juga menunjukkan konsentrasi besar di nilai-nilai rendah, mendekati nol, dengan beberapa outlier yang memiliki

rasio jauh lebih tinggi. Hal ini menunjukkan bahwa sebagian besar properti dijual dengan harga yang tidak jauh berbeda dari nilai taksirannya, namun ada beberapa kasus di mana harga penjualan jauh melebihi nilai taksiran.

Boxplot untuk Sale Amount, Assessed Value, dan Sales Ratio: Memungkinkan kita melihat ringkasan statistik (median, kuartil, outlier) untuk setiap variabel dan membandingkannya jika perlu (misalnya, membandingkan distribusi Sale Amount berdasarkan Property Type).

```
plt.figure(figsize=(18, 6)) # Ukuran plot yang lebih lebar

plt.subplot(1, 3, 1)
sns.boxplot(x='Property Type', y='Sale Amount', data=df_dropduplicate)
plt.title('Boxplot Sale Amount by Property Type')
plt.xticks(rotation=45, ha='right') # Putar label sumbu x

plt.subplot(1, 3, 2)
sns.boxplot(x='Property Type', y='Assessed Value', data=df_dropduplicate)
plt.title('Boxplot Assessed Value by Property Type')
plt.xticks(rotation=45, ha='right')

plt.subplot(1, 3, 3)
sns.boxplot(x='Property Type', y='Sales Ratio', data=df_dropduplicate)
plt.title('Boxplot Sales Ratio by Property Type')
plt.xticks(rotation=45, ha='right')

plt.tight_layout()
plt.show()
```



Visualisasi di atas menampilkan boxplot yang membandingkan distribusi Sale Amount, Assessed Value, dan Sales Ratio berdasarkan tipe properti (Residential, Condo, Two Family, Three Family, Single Family, dan Four Family). Pada boxplot Sale Amount dan Assessed Value, terlihat bahwa properti Single Family dan Four Family cenderung memiliki median dan rentang nilai yang lebih tinggi dibandingkan tipe properti lainnya, dengan Residential juga menunjukkan beberapa nilai penjualan yang sangat tinggi (outlier). Sementara itu, boxplot Sales Ratio menunjukkan bahwa sebagian besar tipe properti memiliki rasio penjualan yang terpusat di nilai rendah, namun terdapat beberapa outlier dengan rasio yang jauh lebih tinggi, terutama terlihat pada tipe Single Family dan Four Family, mengindikasikan adanya transaksi dengan harga jual yang jauh melebihi nilai taksirannya pada tipe properti tersebut.

✓ 2. Distribusi Data Kategorikal:

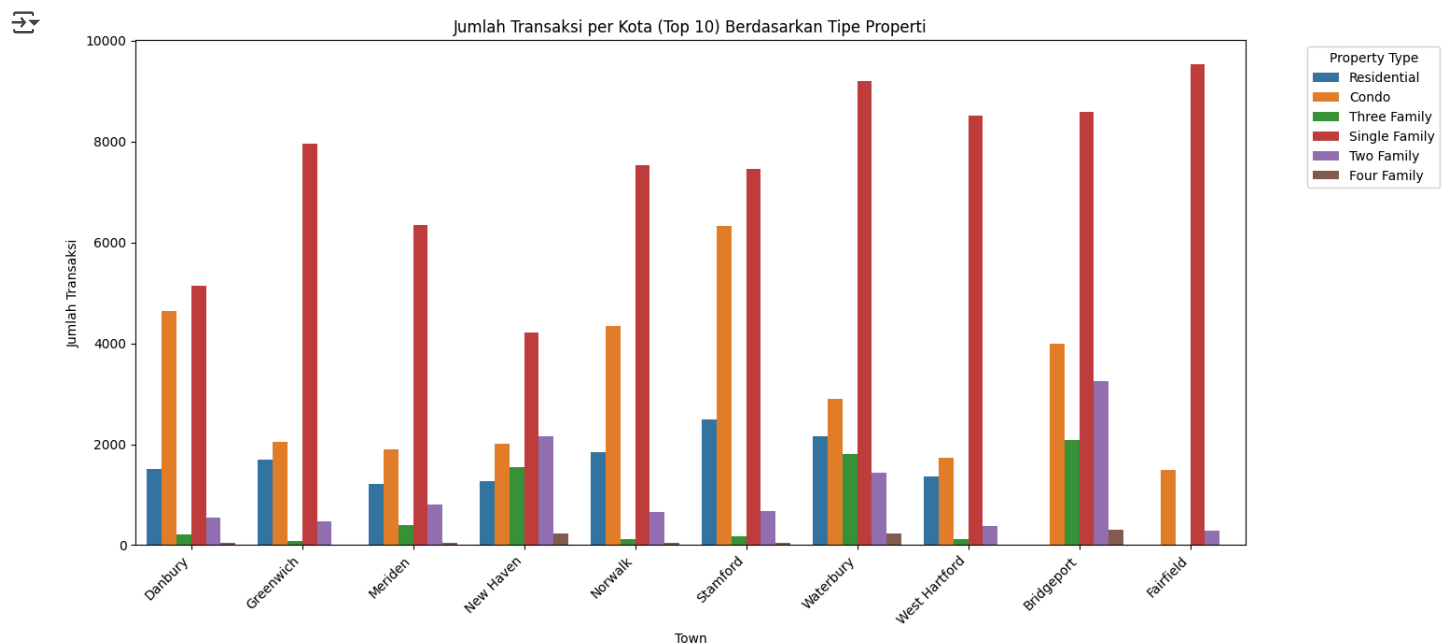
Bar Chart untuk Town: Menampilkan jumlah transaksi di setiap kota. Ini akan menunjukkan kota mana yang memiliki aktivitas penjualan properti tertinggi.

Bar Chart untuk Property Type: Menampilkan jumlah transaksi untuk setiap jenis properti. Ini akan menunjukkan jenis properti mana yang paling banyak diperjualbelikan.

Bar Chart untuk Residential Type: Menampilkan jumlah transaksi untuk setiap tipe residensial yang lebih spesifik.

```
# Ambil 10 kota teratas
top_10_towns = df_dropduplicate['Town'].value_counts().nlargest(10).index
df_top_towns = df_dropduplicate[df_dropduplicate['Town'].isin(top_10_towns)]

# Grouped Bar Chart
plt.figure(figsize=(15, 7))
sns.countplot(data=df_top_towns, x='Town', hue='Property Type')
plt.title('Jumlah Transaksi per Kota (Top 10) Berdasarkan Tipe Properti')
plt.xlabel('Town')
plt.ylabel('Jumlah Transaksi')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Property Type', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```



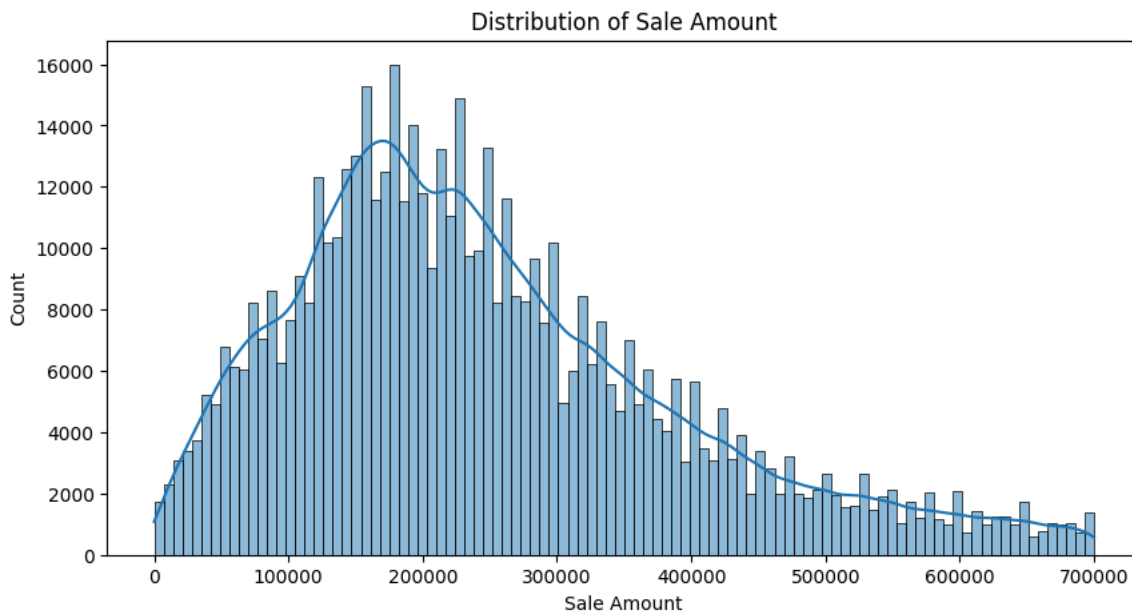
Visualisasi di atas adalah grafik batang kelompok yang menampilkan jumlah transaksi per kota (untuk 10 kota teratas) berdasarkan tipe properti (Residential, Condo, Three Family, Single Family, Two Family, dan Four Family). Setiap kelompok batang mewakili satu kota, dan di dalamnya terdapat batang-batang berwarna yang menunjukkan jumlah transaksi untuk setiap tipe properti di kota tersebut. Terlihat bahwa tipe properti Single Family memiliki jumlah transaksi tertinggi di hampir semua kota, dengan Fairfield dan Stamford menunjukkan jumlah transaksi Single Family yang sangat dominan. Terdapat variasi yang signifikan dalam jumlah transaksi antar tipe properti dan antar kota, mengindikasikan preferensi dan ketersediaan tipe properti yang berbeda di setiap wilayah.

✓ Soal 5: Memvalidasi Data

```
df = df_no_outlier.copy()
```

✓ Histogram Sale Amount

```
# Histogram untuk melihat distribusi
plt.figure(figsize=(10, 5))
sns.histplot(df['Sale Amount'], bins=100, kde=True)
plt.title('Distribution of Sale Amount')
plt.show()
```



Histogram ini menampilkan distribusi jumlah penjualan, memperlihatkan frekuensi berbagai nilai penjualan dalam kumpulan data. Sumbu x mewakili jumlah penjualan, mulai dari 0 hingga 700.000, sementara sumbu y menunjukkan jumlah atau frekuensi penjualan dalam rentang tertentu (bin). Batang-batang tersebut mengilustrasikan bahwa mayoritas jumlah penjualan terkonsentrasi pada nilai yang lebih rendah hingga menengah, dengan puncak di sekitar rentang 150.000 hingga 250.000, dan frekuensi penjualan secara bertahap menurun seiring dengan peningkatan jumlah penjualan, mengindikasikan distribusi yang miring ke kanan di mana lebih sedikit penjualan terjadi pada jumlah yang lebih tinggi.

✓ Bar Chart jumlah properti per kota

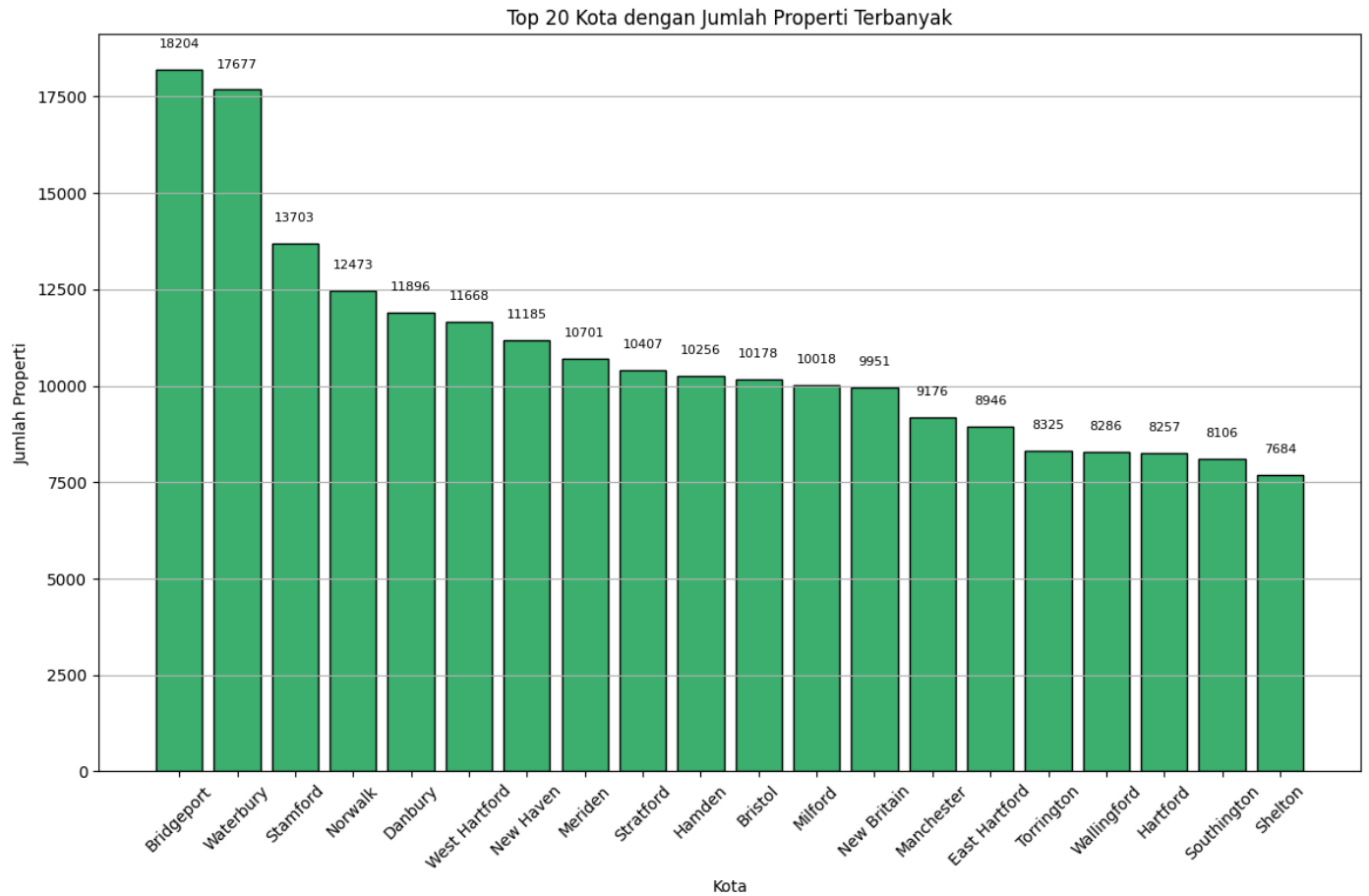
```
# Hitung jumlah properti per kota
jumlah_properti_per_kota = df['Town'].value_counts()

# Pilih top 20 kota (supaya labelnya nggak terlalu rapat)
top20_kota = jumlah_properti_per_kota.head(20)

# Plot bar chart
plt.figure(figsize=(12,8))
bars = plt.bar(top20_kota.index, top20_kota.values, color='mediumseagreen', edgecolor='black')
plt.title('Top 20 Kota dengan Jumlah Properti Terbanyak')
plt.xlabel('Kota')
plt.ylabel('Jumlah Properti')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()

# Tambahkan label jumlah properti di atas bar
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval + 500, int(yval), ha='center', va='bottom', fontsize=8)

plt.show()
```

✓ Pie Chart tipe properti

```
# Hitung jumlah properti per tipe
jumlah_per_tipe = df['Property Type'].value_counts()

# Hitung total untuk dapatkan persentase
total = jumlah_per_tipe.sum()

# Pisahkan yang >=2% dan <2%
proporsi = jumlah_per_tipe / total

# Yang <2% dijadikan 'Others'
data_baru = jumlah_per_tipe[proporsi >= 0.02]
others = jumlah_per_tipe[proporsi < 0.02].sum()

# Gabungkan data
data_final = pd.concat([data_baru, pd.Series({'Others': others})])

# Dictionary warna sesuai tipe properti
color_dict = {
    'Condo': 'skyblue',
    'Four Family': 'lightgreen',
    'Residential': 'lightcoral',
    'Single Family': 'gold',
    'Three Family': 'plum',
    'Two Family': 'blue',
    'Others': 'gray' # Warna default untuk kategori gabungan
}

# Buat list warna sesuai urutan label
colors = [color_dict.get(label, 'gray') for label in data_final.index]
```

```
# Buat pie chart
plt.figure(figsize=(10,10))
plt.pie(data_final, labels=data_final.index, autopct='%1.1f%%', startangle=140, colors=colors)
plt.title('Distribusi Property Type')
plt.axis('equal')
plt.tight_layout()
plt.show()
```

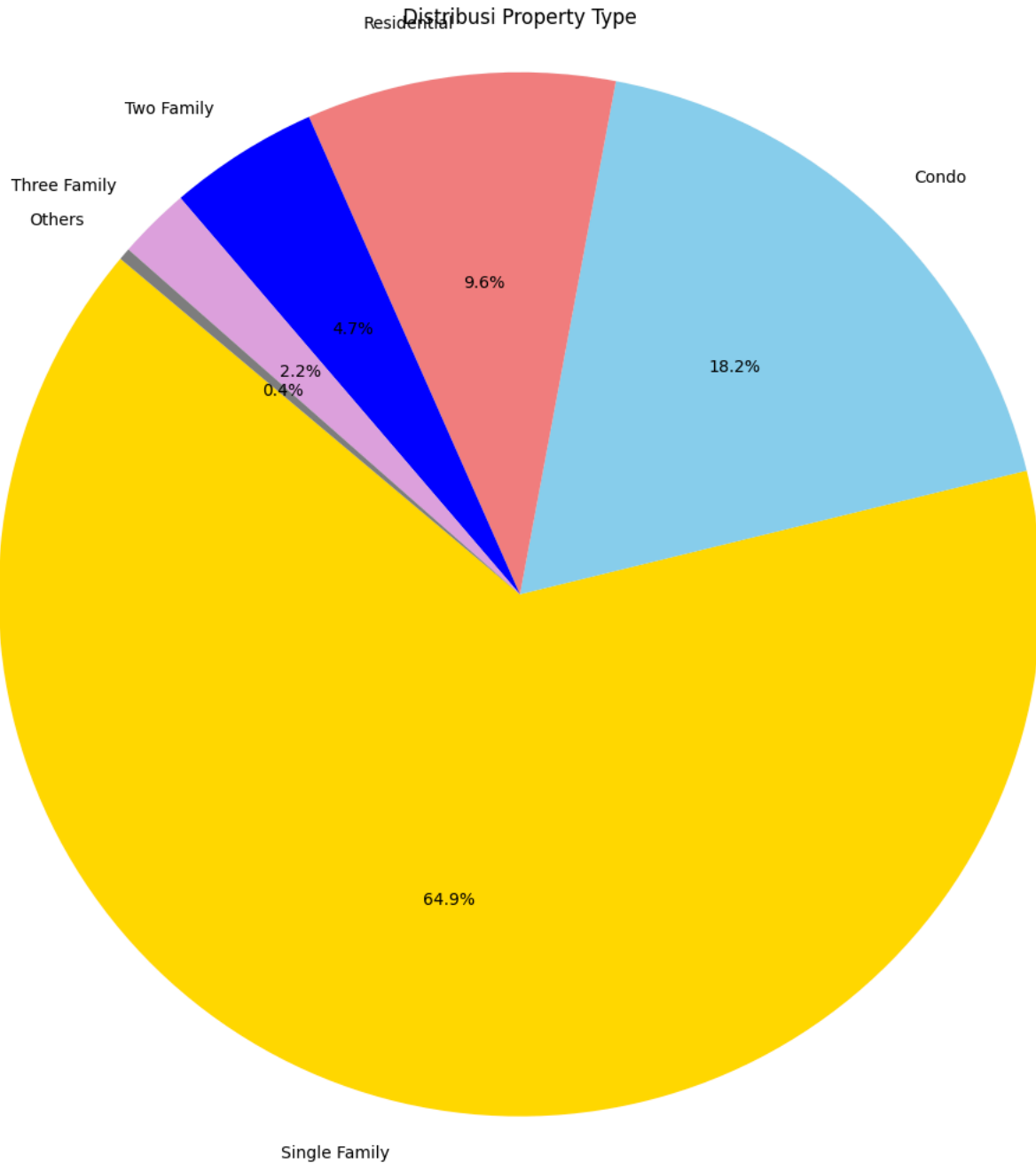


Diagram diatas menunjukkan bagaimana berbagai jenis properti residensial terdistribusi dalam data. Setiap potongan kue mewakili persentase dari total untuk setiap jenis properti. Warna kuning yang mendominasi menunjukkan bahwa properti dengan jenis "Single Family" merupakan bagian terbesar, mencapai 64.9%. Kemudian, diikuti oleh warna biru muda untuk "Condo" sebesar 18.2% dan warna merah muda untuk "Two Family" dengan 9.6%. Jenis properti lain seperti "Three Family" (biru tua) hanya 5.1%, dan kategori "Others" (ungu dan abu-abu) memiliki porsi yang sangat kecil, yaitu 2.2% dan 0.1%. Dari diagram ini, kita bisa melihat bahwa mayoritas properti residensial yang ada adalah Single Family.

✓ Soal 6: Menentukan Objek Data

Setelah melakukan proses data analysis hingga tahap validasi data, untuk menghasilkan berbagai analisis terkait tujuan bisnis, maka ditentukan pemilihan kolom data sebagai berikut:

```
df.columns.to_list()

['Serial Number',
 'List Year',
 'Date Recorded',
 'Town',
 'Address',
 'Assessed Value',
 'Sale Amount',
 'Sales Ratio',
 'Property Type',
 'Residential Type']
```

✓ Soal 7: Membuat Business Intelligence

✓ 1. Keterkaitan Sale Amount - Type Property - List Year - Kota

```
# Agregasi total penjualan per kota per tahun
sales_by_town_year = df.groupby(['Town', 'List Year'])['Sale Amount'].sum().reset_index()

# Ambil 5 kota dengan total penjualan tertinggi
top_5_towns = sales_by_town_year.groupby('Town')['Sale Amount'].sum().nlargest(5).index

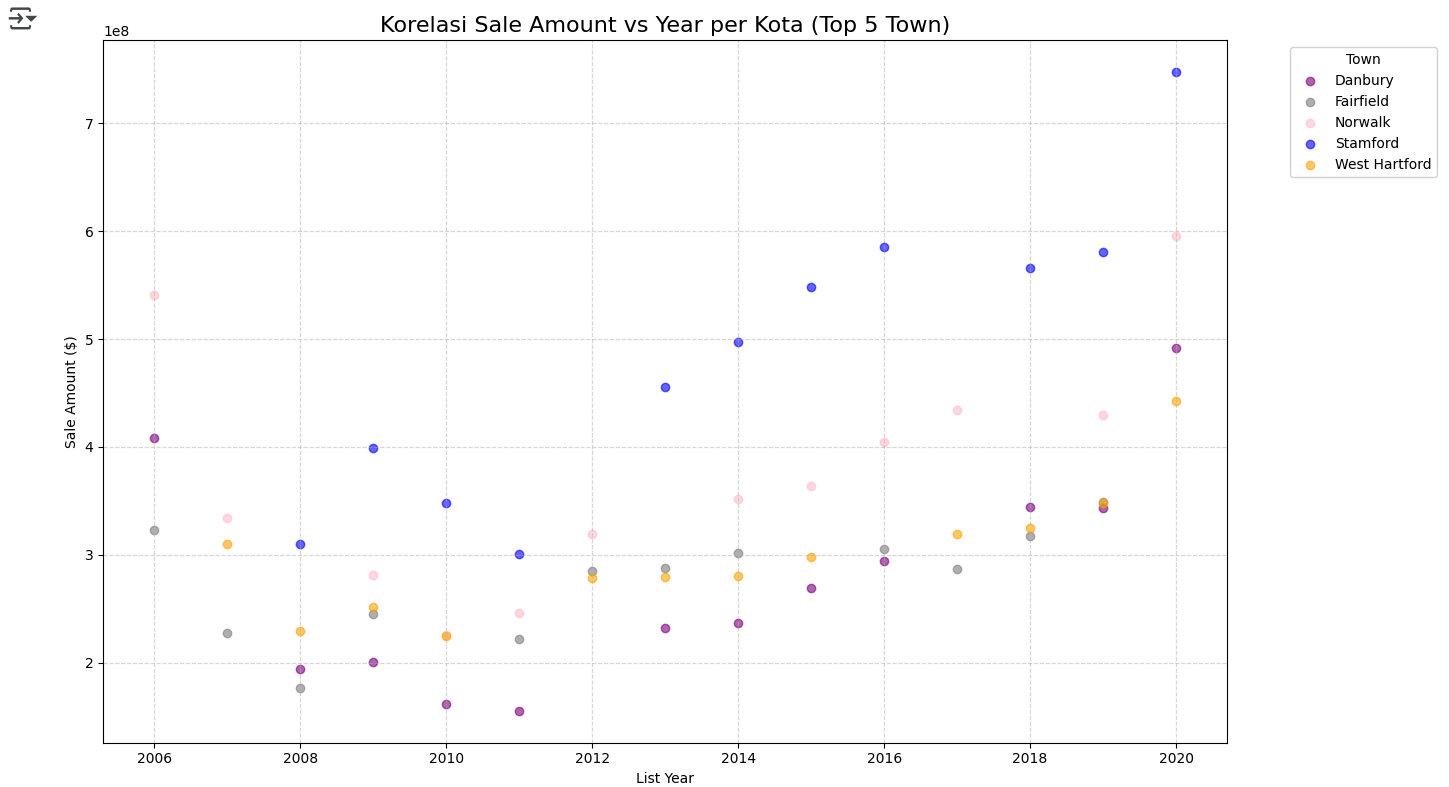
# Filter hanya data dari 5 kota tersebut
df_top_5_towns = sales_by_town_year[sales_by_town_year['Town'].isin(top_5_towns)]

# Gunakan data yang sudah difilter: df_top_5_towns
plt.figure(figsize=(14, 8))

# Warna khusus untuk tiap kota
color_dict = {
    'Bridgeport': 'yellow',
    'Waterbury': 'green',
    'Stamford': 'blue',
    'Norwalk': 'pink',
    'Danbury': 'purple',
    'West Hartford': 'orange'
}

# Plot scatter untuk setiap kota
for town in df_top_5_towns['Town'].unique():
    town_data = df_top_5_towns[df_top_5_towns['Town'] == town]
    plt.scatter(
        town_data['List Year'],
        town_data['Sale Amount'],
        label=town,
        alpha=0.6,
        color=color_dict.get(town, 'gray') # fallback warna abu-abu jika tidak terdaftar
    )

# Tambahkan judul dan label
plt.title('Korelasi Sale Amount vs Year per Kota (Top 5 Town)', fontsize=16)
plt.xlabel('List Year')
plt.ylabel('Sale Amount ($)')
plt.grid(True, linestyle='--', alpha=0.5)
plt.legend(title="Town", bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```



Visualisasi di atas menampilkan korelasi antara tahun listing (List Year) dan jumlah penjualan (Sale Amount) untuk 5 kota teratas berdasarkan jumlah transaksi. Setiap titik pada grafik merepresentasikan rata-rata jumlah penjualan pada tahun tertentu di kota tersebut, dengan warna yang berbeda untuk setiap kota (Danbury, Fairfield, Norwalk, Stamford, dan West Hartford). Terlihat adanya variasi tren harga penjualan antar kota dari tahun 2006 hingga 2020. Beberapa kota menunjukkan kecenderungan peningkatan harga penjualan seiring berjalannya waktu, meskipun dengan fluktuasi di beberapa tahun. Perbedaan pola ini mengindikasikan bahwa dinamika pasar properti dapat berbeda secara signifikan antar wilayah.

Analisis lebih lanjut menunjukkan bahwa tidak ada pola tren yang seragam di antara kelima kota tersebut. Stamford tampak mengalami peningkatan rata-rata harga penjualan yang cukup signifikan menjelang tahun 2020. Sementara itu, kota lain seperti Norwalk dan Fairfield menunjukkan fluktuasi yang lebih besar tanpa tren kenaikan yang sejelas Stamford. Danbury dan West Hartford juga menunjukkan variasi harga dari tahun ke tahun. Perbedaan ini bisa disebabkan oleh berbagai faktor lokal seperti perkembangan ekonomi, infrastruktur, permintaan pasar, dan regulasi properti di masing-masing kota. Untuk pemahaman yang lebih mendalam, perlu dipertimbangkan faktor-faktor spesifik yang mempengaruhi pasar properti di setiap kota tersebut.

```
# Filter data yang valid
df_valid = df[(df['Sale Amount'] > 0) &
              (df['Property Type'].notna()) &
              (df['Town'].notna())]

# Ambil 5 kota teratas berdasarkan total penjualan
top_5_towns = df_valid.groupby('Town')['Sale Amount'].sum().nlargest(5).index
df_top5 = df_valid[df_valid['Town'].isin(top_5_towns)]

# Agregasi rata-rata sale amount per Property Type di tiap kota
grouped = df_top5.groupby(['Town', 'Property Type'])['Sale Amount'].mean().reset_index()

# Tentukan warna untuk masing-masing Property Type
color_dict = {
    'Condo': 'skyblue',
```

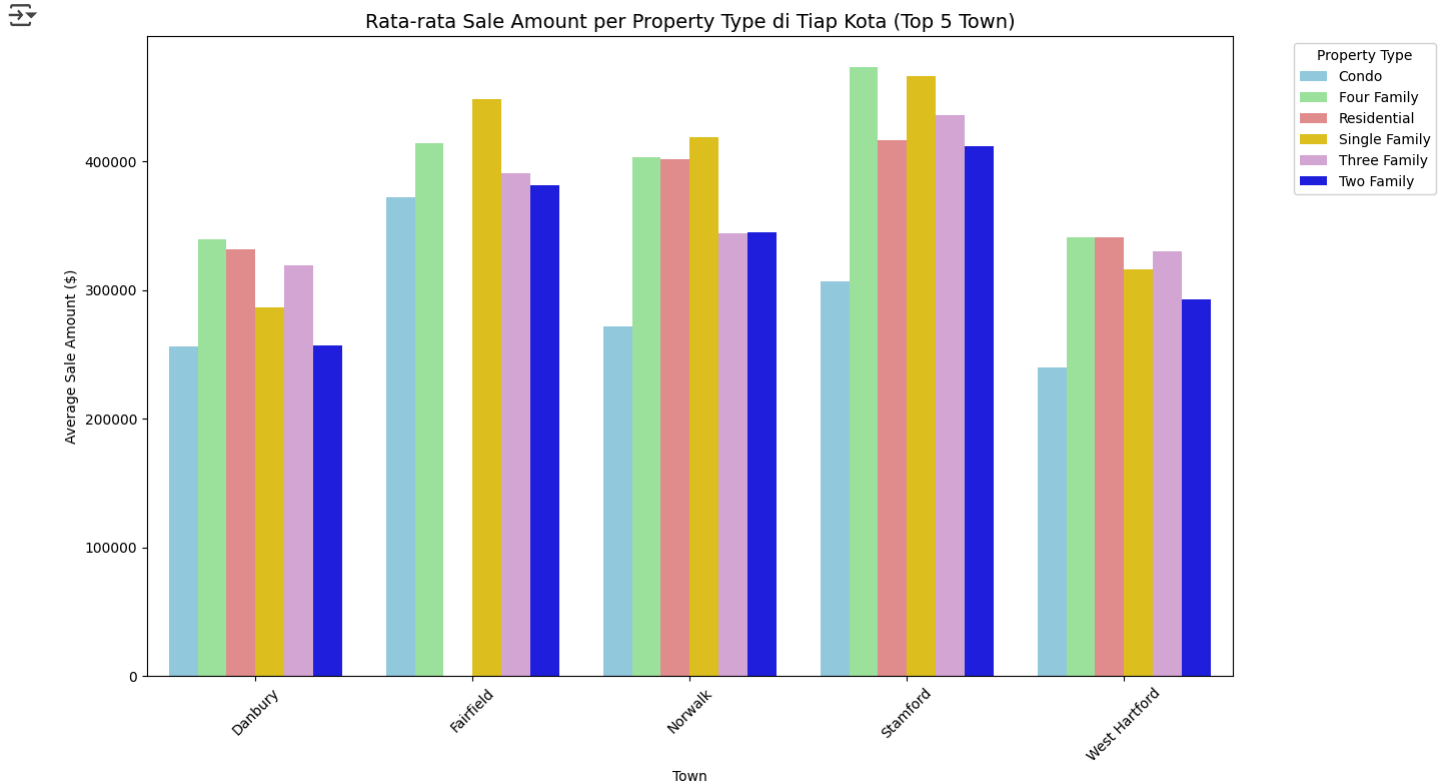
```

'Four Family': 'lightgreen',
'Residential': 'lightcoral',
'Single Family': 'gold',
'Three Family': 'plum',
'Two Family': 'blue' #
}

# Plot bar chart dengan seaborn
plt.figure(figsize=(14, 8))
sns.barplot(data=grouped, x='Town', y='Sale Amount', hue='Property Type', palette=color_dict)

plt.title('Rata-rata Sale Amount per Property Type di Tiap Kota (Top 5 Town)', fontsize=14)
plt.ylabel('Average Sale Amount ($)')
plt.xlabel('Town')
plt.legend(title='Property Type', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

```



Visualisasi di atas menampilkan rata-rata jumlah penjualan (Average Sale Amount) untuk berbagai tipe properti di lima kota teratas berdasarkan jumlah transaksi (Danbury, Fairfield, Norwalk, Stamford, dan West Hartford). Setiap kelompok batang pada grafik mewakili satu kota, dan di dalam setiap kelompok terdapat batang-batang berwarna yang menunjukkan rata-rata harga untuk tipe properti Condo, Four Family, Residential, Single Family, Three Family, dan Two Family. Terlihat adanya perbedaan signifikan dalam rata-rata harga antar tipe properti dan antar kota. Secara umum, properti Single Family cenderung memiliki rata-rata harga tertinggi di sebagian besar kota, diikuti oleh Residential. Tipe properti seperti Condo dan Two Family seringkali memiliki rata-rata harga yang lebih rendah.

Analisis lebih lanjut menunjukkan bahwa urutan rata-rata harga antar tipe properti relatif konsisten di kelima kota, meskipun dengan variasi nilai yang cukup besar. Misalnya, di Stamford, rata-rata harga untuk semua tipe properti cenderung lebih tinggi dibandingkan dengan kota lain. Fairfield juga menunjukkan rata-rata harga yang tinggi untuk Single Family dan Residential. Sementara itu, Danbury dan West Hartford memiliki rata-rata harga yang lebih rendah secara keseluruhan untuk sebagian besar tipe properti. Perbedaan ini mengindikasikan adanya faktor-faktor lokal yang mempengaruhi harga properti di setiap kota, seperti lokasi, fasilitas, dan permintaan pasar untuk jenis properti.

tertentu. Pemahaman lebih mendalam mengenai faktor-faktor ini akan membantu dalam menganalisis dinamika pasar properti di masing-masing kota.

✓ 2. Analisis Tren Penjualan Rumah 2001-2020

```
# Pastikan kolom List Year dan Sale Amount tidak null
df['List Year'] = pd.to_numeric(df['List Year'], errors='coerce')
df['Sale Amount'] = pd.to_numeric(df['Sale Amount'], errors='coerce')

# Filter tahun 2001 sampai 2020 saja
df_filtered = df[(df['List Year'] >= 2001) & (df['List Year'] <= 2020)]

# Hitung rata-rata harga properti per List Year
rata2_per_tahun = df_filtered.groupby('List Year')['Sale Amount'].mean().sort_index()

# Plot bar chart
plt.figure(figsize=(14,8))
plt.bar(rata2_per_tahun.index.astype(str), rata2_per_tahun.values, color='skyblue', edgecolor='black')

# Menambahkan judul dan label
plt.title('Harga Rata-Rata Properti per Tahun (2001 - 2020)', fontsize=16)
plt.xlabel('Tahun', fontsize=12)
plt.ylabel('Harga Rata-Rata Properti (USD)', fontsize=12)

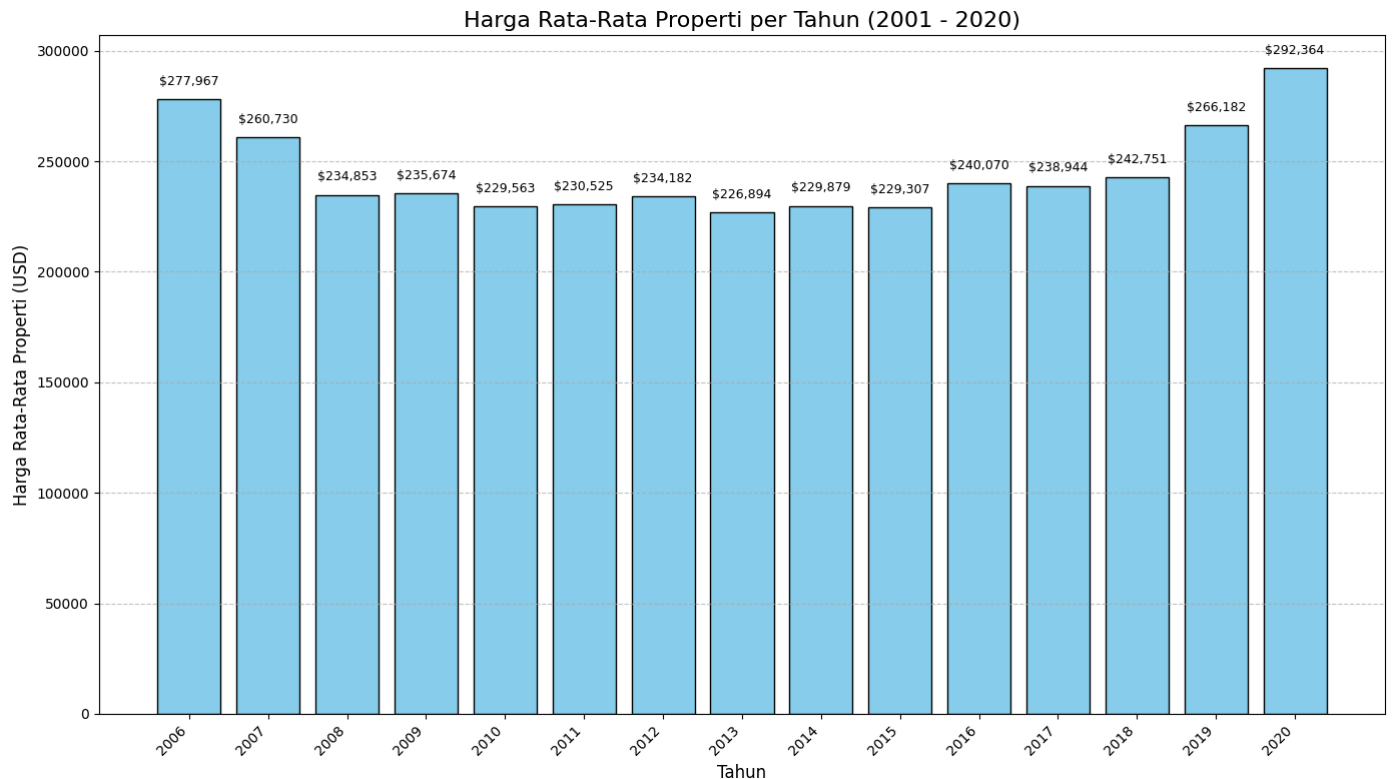
# Rotasi label x untuk kejelasan
plt.xticks(rotation=45, ha='right', fontsize=10)
plt.yticks(fontsize=10)

# Menambahkan grid horizontal
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Menambahkan label di atas bar
for i, value in enumerate(rata2_per_tahun.values):
    plt.text(i, value + 5000, f"${value:,.0f}", ha='center', va='bottom', fontsize=9)

# Layout yang rapih
plt.tight_layout()

plt.show()
```



Visualisasi di atas adalah grafik batang yang menampilkan harga rata-rata properti per tahun dari tahun 2006 hingga 2020. Sumbu horizontal menunjukkan tahun, sedangkan sumbu vertikal menunjukkan harga rata-rata properti dalam USD. Setiap batang merepresentasikan harga rata-rata properti pada tahun tertentu, dengan nilai eksaknya tertera di atas setiap batang. Terlihat adanya fluktuasi harga rata-rata properti selama periode waktu tersebut. Harga cenderung menurun setelah tahun 2006, mencapai titik terendah di sekitar tahun 2013-2014, sebelum kemudian menunjukkan tren peningkatan hingga tahun 2020.

Analisis lebih lanjut menunjukkan bahwa harga rata-rata properti mengalami penurunan yang signifikan dari puncaknya di tahun 2006 (277,867) hingga titik terendahnya di tahun 2014 (226,894). Setelah tahun 2014, pasar properti mulai menunjukkan pemulihan dan pertumbuhan, dengan harga rata-rata secara bertahap meningkat hingga mencapai nilai tertinggi kedua di tahun 2020 (\$292,364). Penurunan setelah tahun 2006 kemungkinan dipengaruhi oleh krisis keuangan global, sementara pemulihan setelah tahun 2014 bisa disebabkan oleh berbagai faktor ekonomi dan demografi. Lonjakan harga di tahun 2020 menarik untuk dicermati lebih lanjut, mengingat konteks waktu tersebut.

✓ 3. Analisis Tren Penjualan Rumah 2001-2020 Per Tipe Properti

```
# Pastikan kolom valid
df['List Year'] = pd.to_numeric(df['List Year'], errors='coerce')
df['Sale Amount'] = pd.to_numeric(df['Sale Amount'], errors='coerce')

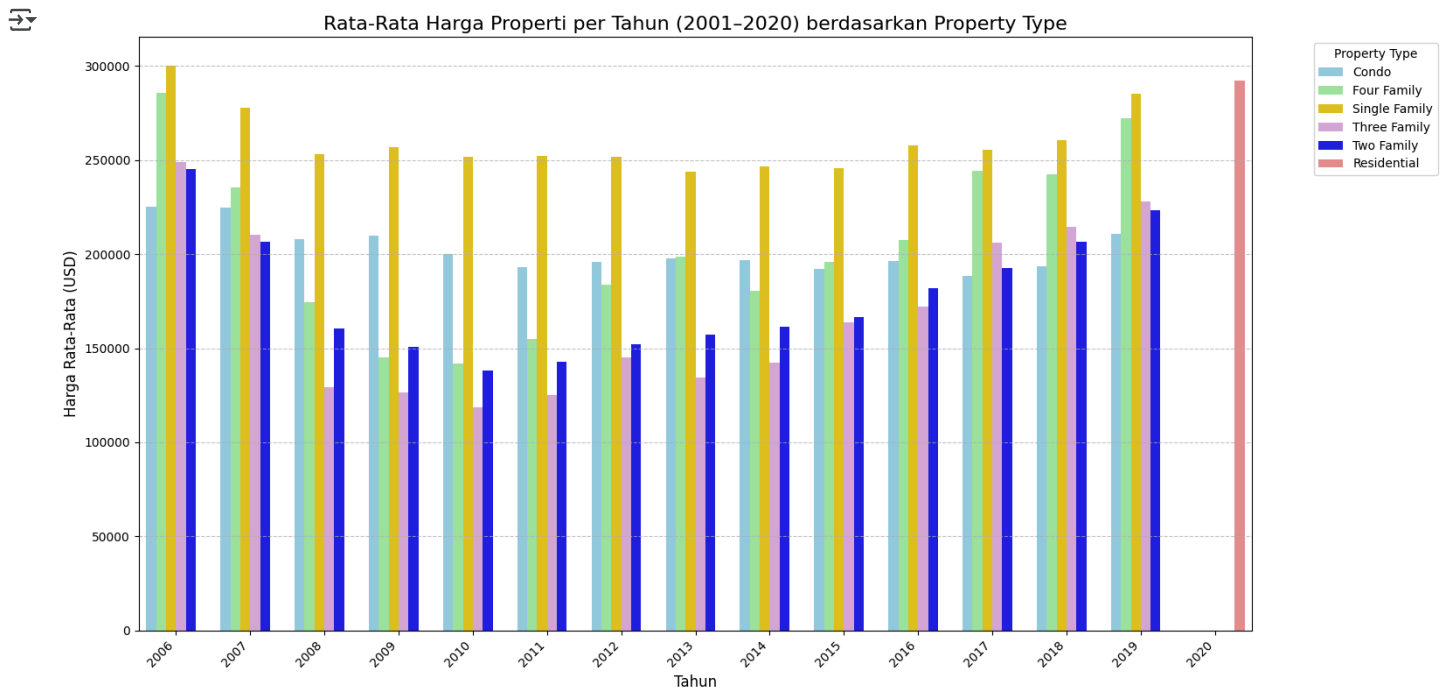
# Filter tahun dan pastikan property type tidak null
df_filtered = df[(df['List Year'] >= 2001) &
                 (df['List Year'] <= 2020) &
                 (df['Sale Amount'] > 0) &
                 (df['Property Type'].notna())]

# Hitung rata-rata Sale Amount per tahun dan Property Type
rata2_per_tahun_ptype = df_filtered.groupby(['List Year', 'Property Type'])['Sale Amount'].mean().reset_index()
```

```
# Definisi warna untuk setiap property type
warna_properti = {
    'Condo' : 'skyblue',
    'Four Family' : 'lightgreen',
    'Residential' : 'lightcoral',
    'Single Family' : 'gold',
    'Three Family' : 'Plum',
    'Two Family' : 'blue'
}

# Plot menggunakan seaborn dengan palette warna yang didefinisikan
plt.figure(figsize=(16, 8))
sns.barplot(data=rata2_per_tahun_ptype,
            x='List Year',
            y='Sale Amount',
            hue='Property Type',
            palette=warna_properti)

plt.title('Rata-Rata Harga Properti per Tahun (2001-2020) berdasarkan Property Type', fontsize=16)
plt.xlabel('Tahun', fontsize=12)
plt.ylabel('Harga Rata-Rata (USD)', fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.legend(title='Property Type', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



Visualisasi di atas adalah grafik batang kelompok yang menampilkan rata-rata harga properti per tahun dari 2006 hingga 2020, dibedakan berdasarkan tipe properti (Condo, Four Family, Single Family, Three Family, Two Family, dan Residential). Setiap kelompok batang untuk setiap tahun menunjukkan rata-rata harga untuk masing-masing tipe properti. Terlihat adanya variasi harga rata-rata antar tipe properti dan bagaimana harga-harga ini berubah dari tahun ke tahun. Secara umum, properti Single Family cenderung memiliki rata-rata harga tertinggi di

sebagian besar tahun, diikuti oleh Residential. Tipe properti seperti Condo dan Two Family biasanya memiliki rata-rata harga yang lebih rendah.

Analisis lebih lanjut mengungkapkan bahwa tren harga dari tahun 2006 hingga 2020 tidak seragam di semua tipe properti. Setelah penurunan harga secara umum di beberapa tahun awal periode, sebagian besar tipe properti menunjukkan tren peningkatan harga menjelang tahun 2020. Namun, tingkat pemulihan dan pertumbuhan harga bervariasi antar tipe properti. Misalnya, Single Family dan Residential seringkali menunjukkan nilai rata-rata yang lebih tinggi dan perubahan yang lebih signifikan dibandingkan dengan tipe properti dengan unit ganda atau tiga keluarga. Fluktuasi pasar properti tampaknya mempengaruhi setiap segmen dengan cara yang berbeda, yang mungkin disebabkan oleh dinamika permintaan dan penawaran yang spesifik untuk setiap jenis properti.

✓ 4.Pemodelan Forecasting Sale Amount 2021 Untuk Setiap Kota

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 558211 entries, 1 to 996559
Data columns (total 10 columns):
#   Column              Non-Null Count  Dtype
---  -
0   Serial Number       558211 non-null float64
1   List Year           558211 non-null int64
2   Date Recorded       558211 non-null datetime64[ns]
3   Town                558211 non-null object
4   Address             558211 non-null object
5   Assessed Value      558211 non-null float64
6   Sale Amount         558211 non-null float64
7   Sales Ratio         558211 non-null float64
8   Property Type       558211 non-null object
9   Residential Type    558211 non-null object
dtypes: datetime64[ns](1), float64(4), int64(1), object(4)
memory usage: 46.8+ MB
```

```
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
import itertools
import warnings

warnings.filterwarnings("ignore")

# Filter data untuk kota Bridgeport
bridgeport_df = df[df['Town'] == 'Bridgeport']

# Agregasi rata-rata Sale Amount per List Year
annual_sales = bridgeport_df.groupby('List Year')['Sale Amount'].mean().sort_index()

# Gunakan hanya data sampai 2020 (untuk prediksi 2021)
ts = annual_sales[annual_sales.index <= 2020]

# Grid search untuk parameter ARIMA (p, d, q)
p = d = q = range(0, 3)
pdq_combinations = list(itertools.product(p, d, q))

# Cari kombinasi terbaik berdasarkan AIC
best_aic = float('inf')
best_order = None
for order in pdq_combinations:
    try:
        model = ARIMA(ts, order=order)
        results = model.fit()
        if results.aic < best_aic:
            best_aic = results.aic
            best_order = order
    except:
        continue

print(f"Best ARIMA order: {best_order} with AIC: {best_aic:.2f}")

# Fit model dengan parameter terbaik
model = ARIMA(ts, order=best_order)
results = model.fit()

# Forecast 1 tahun (2021)
forecast = results.forecast(steps=1)
```

```

forecast_2021 = forecast.iloc[0] # Perbaiki di sini

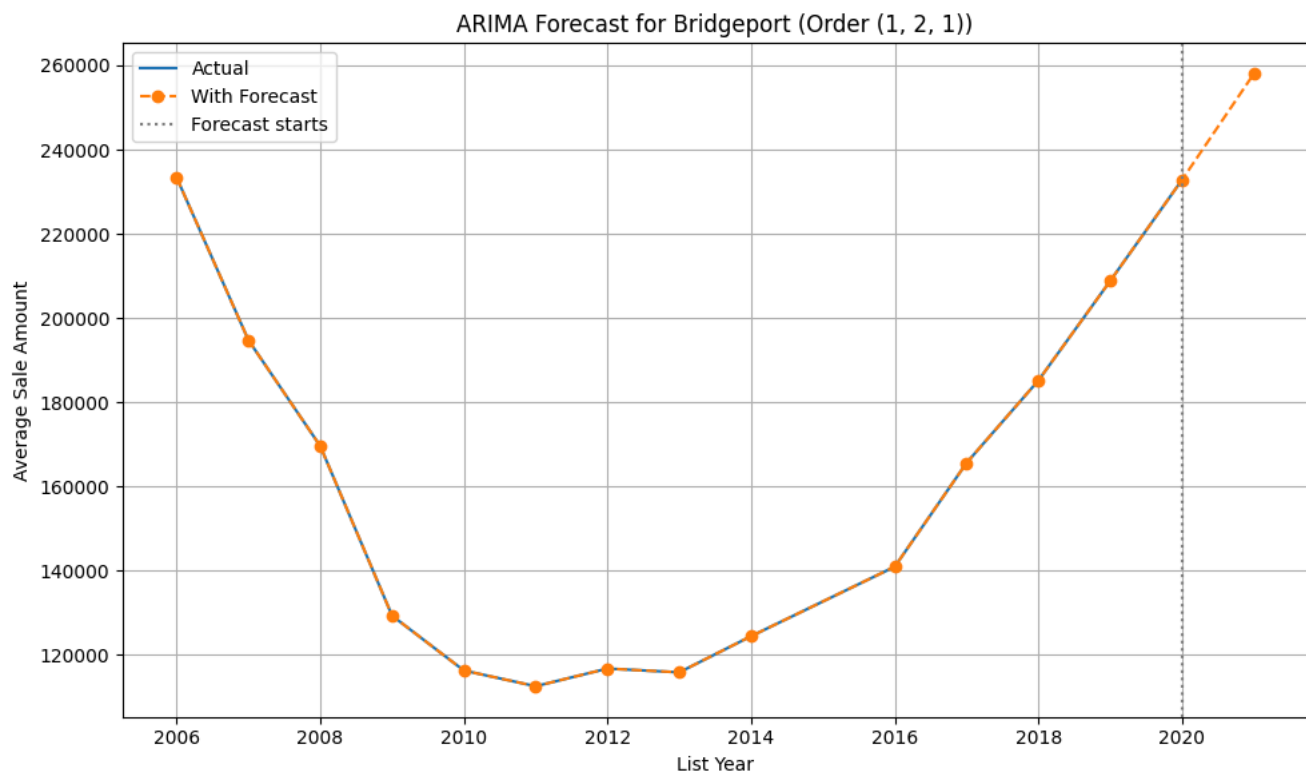
# Tambahkan hasil forecast ke data asli
ts_with_forecast = ts.copy()
ts_with_forecast.loc[2021] = forecast_2021

# Visualisasi hasil
plt.figure(figsize=(10,6))
plt.plot(ts.index, ts.values, label='Actual')
plt.plot(ts_with_forecast.index, ts_with_forecast.values, label='With Forecast', linestyle='--', marker='o')
plt.axvline(x=2020, color='gray', linestyle=':', label='Forecast starts')
plt.title(f"ARIMA Forecast for Bridgeport (Order {best_order})")
plt.xlabel('List Year')
plt.ylabel('Average Sale Amount')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# Tampilkan hasil prediksi
print(f"Forecasted Sale Amount for Bridgeport in 2021: ${forecast_2021:,.2f}")

```

Best ARIMA order: (1, 2, 1) with AIC: 259.02



Forecasted Sale Amount for Bridgeport in 2021: \$258,005.15

Visualisasi di atas menampilkan hasil peramalan (forecast) rata-rata jumlah penjualan (Average Sale Amount) untuk kota Bridgeport menggunakan model ARIMA (Autoregressive Integrated Moving Average) dengan orde (1, 2, 1). Garis biru menunjukkan data rata-rata penjualan aktual dari tahun 2006 hingga 2020. Garis berwarna oranye dengan titik-titik menunjukkan hasil peramalan model ARIMA yang sesuai dengan data aktual hingga tahun 2020, dan kemudian diperpanjang sebagai proyeksi ke tahun 2021. Garis putus-putus vertikal menandai dimulainya periode peramalan di luar data aktual. Model ARIMA ini dipilih sebagai model terbaik dengan nilai AIC (Akaike Information Criterion) sebesar 259.02.

Berdasarkan hasil peramalan, terlihat bahwa setelah tren peningkatan yang signifikan dari sekitar tahun 2012 hingga 2020, model ARIMA memprediksi adanya kenaikan rata-rata jumlah penjualan yang terus berlanjut di tahun 2021 untuk kota Bridgeport. Meskipun model ini telah dipilih berdasarkan kriteria AIC, penting untuk diingat bahwa peramalan selalu mengandung ketidakpastian. Akurasi peramalan di masa depan akan bergantung pada stabilitas pola data historis dan tidak adanya kejadian tak terduga yang dapat mempengaruhi pasar properti. Oleh karena itu, hasil peramalan ini sebaiknya dipertimbangkan sebagai indikasi potensi tren, bukan sebagai prediksi yang pasti.

✓ Soal 8: Menyusun Laporan Hasil Analisis

LAPORAN HASIL ANALISIS DATA

Judul Analisis: Analisis Data Penjualan Properti Tahun 2001 - 2020

Disusun oleh: Erykka Yustari

Tanggal: 29 April 2025

1. Latar Belakang

Data properti merupakan salah satu sumber informasi penting dalam memahami dinamika pasar perumahan dan properti secara umum. Melalui data tersebut, kita dapat menganalisis bagaimana nilai properti berkembang dari waktu ke waktu, mengevaluasi kesenjangan antara nilai taksiran dengan harga jual, serta mengidentifikasi karakteristik wilayah dengan aktivitas jual beli properti yang tinggi.

Dataset yang digunakan dalam analisis ini berisi 996.561 baris data properti dengan 14 variabel, termasuk informasi seperti tahun listing, tanggal pencatatan, nilai taksiran properti (*Assessed Value*), harga jual (*Sale Amount*), rasio penjualan (*Sales Ratio*), dan klasifikasi properti berdasarkan tipe hunian maupun komersial.

Dengan volume data yang besar dan cakupan informasi yang luas, analisis ini bertujuan untuk mengungkap pola-pola penting dalam transaksi properti. Hal ini mencakup identifikasi tren harga jual, kota dengan aktivitas transaksi tertinggi, distribusi tipe properti yang dominan, serta evaluasi akurasi penilaian properti melalui perbandingan nilai taksiran dan harga jual.

Pemahaman yang lebih dalam terhadap data properti ini diharapkan dapat memberikan wawasan yang berguna bagi pengambil keputusan di bidang perencanaan tata ruang, investasi properti, serta pengembangan wilayah.

2. Tujuan

Analisis data properti ini dilakukan dengan tujuan untuk:

1. Menganalisis hubungan antara harga jual properti (*Sale Amount*), tipe properti (*Property Type*), tahun terdaftar (*List Year*), dan lokasi properti (*Town*) untuk memahami pola distribusi harga jual berdasarkan faktor-faktor tersebut.
 2. Melakukan analisis tren penjualan rumah antara tahun 2001 hingga 2020 berdasarkan harga jual dan tipe rumah untuk mengidentifikasi perubahan pasar dari waktu ke waktu.
 3. Melakukan prediksi (forecast) harga jual properti (*Sale Amount*) untuk tahun 2021 di setiap kota, dengan tujuan memberikan gambaran potensi pasar properti di masa mendatang dan membantu pengambilan keputusan dalam pengelolaan dan investasi properti.
-

3. Ruang Lingkup

Analisis ini mencakup data properti yang tercatat antara tahun 2001 hingga 2020, dengan fokus pada variabel-variabel berikut:

1. **Sale Amount (Harga Jual):** Menganalisis harga jual properti berdasarkan lokasi, tipe properti, dan tahun terdaftar.
2. **Property Type (Tipe Properti):** Mengelompokkan properti berdasarkan kategori tipe, seperti rumah, apartemen, atau komersial, dan menganalisis hubungannya dengan harga jual dan lokasi.
3. **List Year (Tahun Terdaftar):** Memeriksa tren penjualan properti berdasarkan tahun pendaftaran, dari 2001 hingga 2020, serta hubungan dengan harga jual dan tipe properti.
4. **Town (Kota):** Menganalisis distribusi harga jual dan tren properti di berbagai kota yang tercatat dalam dataset.
5. **Forecasting (Prediksi):** Melakukan prediksi harga jual properti untuk tahun 2021 di setiap kota, menggunakan data historis harga jual yang tersedia untuk memproyeksikan nilai masa depan.

Analisis ini akan menggunakan metode statistik dan model prediksi untuk menggali informasi tentang tren penjualan properti dan faktor-faktor yang mempengaruhi harga jual di berbagai lokasi.

4. Metodologi

Dalam analisis ini, digunakan beberapa metode dan pustaka Python untuk memproses data, melakukan visualisasi, dan membangun model prediksi. Berikut adalah rincian metodologi yang diterapkan:

1. Pengolahan Data (Data Processing)

- Data yang digunakan berasal dari file CSV yang berukuran 106,2 MB dengan 996.561 baris dan 14 kolom.
- Pustaka **Pandas** digunakan untuk mengimpor, membersihkan, dan memproses data. Proses ini mencakup:

- Penghilangan nilai yang hilang (missing values) atau duplikat.
- Pengubahan tipe data jika diperlukan (misalnya, mengonversi kolom tanggal menjadi tipe `datetime`).
- Penyaringan data berdasarkan tahun terdaftar (List Year) dan lokasi (Town) yang relevan.

2. Visualisasi Data (Data Visualization)

- **Matplotlib** digunakan untuk menghasilkan grafik yang menggambarkan hubungan antar variabel dan tren data, termasuk:
 - Scatter plot untuk melihat keterkaitan antara `Sale Amount`, `Property Type`, dan `Town`.
 - Grafik garis untuk menganalisis tren penjualan berdasarkan tahun.
 - Box plot atau histogram untuk visualisasi distribusi harga jual dan tipe properti.

3. Analisis Tren Penjualan (Trend Analysis)

- Tren harga jual (`Sale Amount`) dan tipe properti akan dianalisis untuk periode 2001 hingga 2020 menggunakan agregasi data berdasarkan tahun.
- **Matplotlib** digunakan untuk memplot grafik tren harga dan tipe properti seiring waktu, untuk memahami perubahan pasar properti.

4. Model Prediksi (Forecasting)

- Untuk memprediksi harga jual properti (`Sale Amount`) di tahun 2021, model ARIMA (AutoRegressive Integrated Moving Average) digunakan, yang merupakan salah satu model peramalan berbasis statistik yang populer.
- **Statsmodels** digunakan untuk membangun model ARIMA, di mana langkah-langkah berikut dilakukan:
 - Pemilihan parameter ARIMA (p, d, q) menggunakan teknik seperti ACF (AutoCorrelation Function) dan PACF (Partial AutoCorrelation Function).
 - Pelatihan model ARIMA pada data penjualan dari 2001 hingga 2020.
 - Prediksi harga jual properti di setiap kota untuk tahun 2021 berdasarkan model yang telah dibangun.

Proses ini memastikan bahwa hasil analisis dan prediksi berbasis data yang valid dan dapat memberikan wawasan yang relevan mengenai pasar properti.

5. Hasil Analisis

✓ 1. Keterkaitan Sale Amount - Type Property - List Year - Kota

```
# Agregasi total penjualan per kota per tahun
sales_by_town_year = df.groupby(['Town', 'List Year'])['Sale Amount'].sum().reset_index()

# Ambil 5 kota dengan total penjualan tertinggi
top_5_towns = sales_by_town_year.groupby('Town')['Sale Amount'].sum().nlargest(5).index

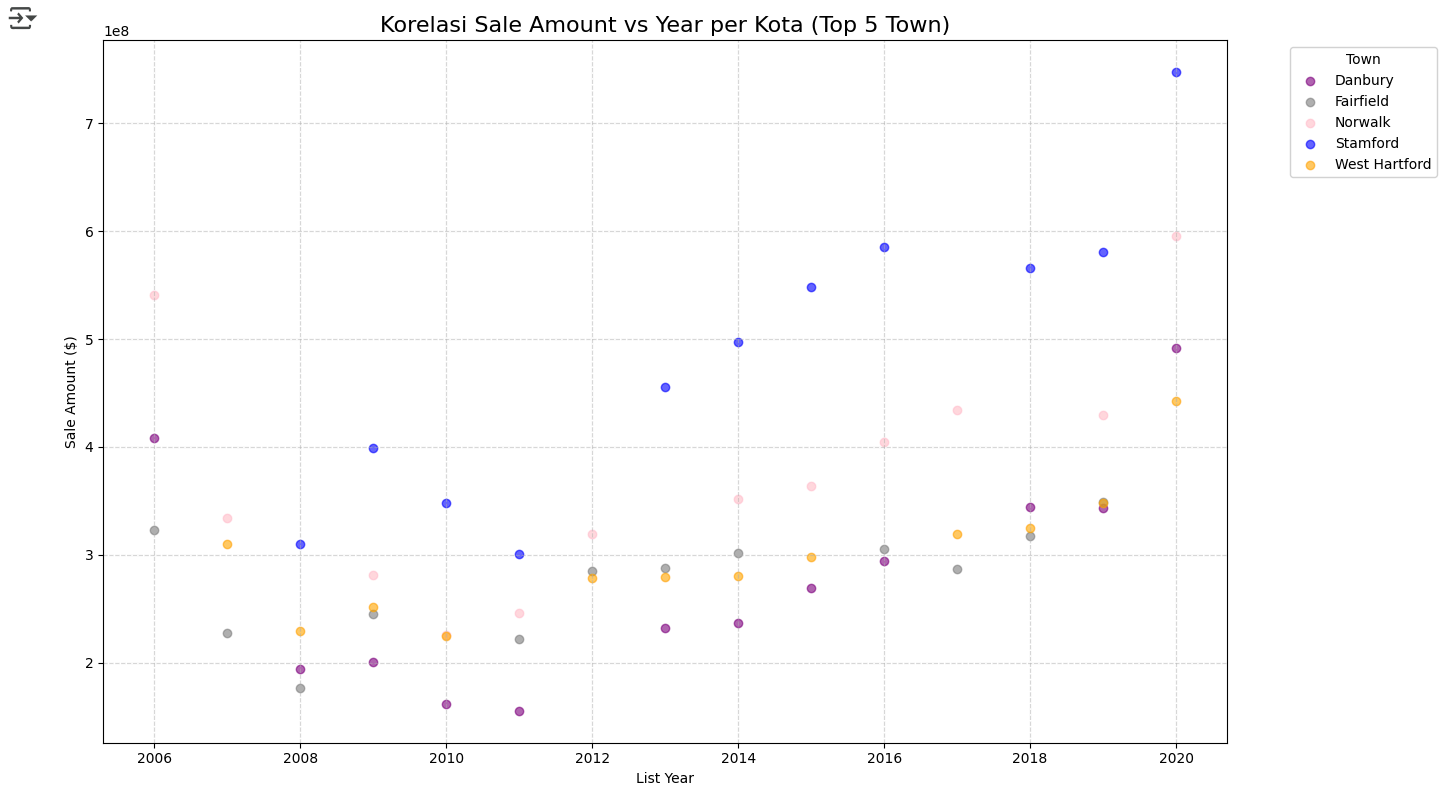
# Filter hanya data dari 5 kota tersebut
df_top_5_towns = sales_by_town_year[sales_by_town_year['Town'].isin(top_5_towns)]

# Gunakan data yang sudah difilter: df_top_5_towns
plt.figure(figsize=(14, 8))

# Warna khusus untuk tiap kota
color_dict = {
    'Bridgeport': 'yellow',
    'Waterbury': 'green',
    'Stamford': 'blue',
    'Norwalk': 'pink',
    'Danbury': 'purple',
    'West Hartford': 'orange'
}

# Plot scatter untuk setiap kota
for town in df_top_5_towns['Town'].unique():
    town_data = df_top_5_towns[df_top_5_towns['Town'] == town]
    plt.scatter(
        town_data['List Year'],
        town_data['Sale Amount'],
        label=town,
        alpha=0.6,
        color=color_dict.get(town, 'gray') # fallback warna abu-abu jika tidak terdaftar
    )
```

```
# Tambahkan judul dan label
plt.title('Korelasi Sale Amount vs Year per Kota (Top 5 Town)', fontsize=16)
plt.xlabel('List Year')
plt.ylabel('Sale Amount ($)')
plt.grid(True, linestyle='--', alpha=0.5)
plt.legend(title="Town", bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```



Visualisasi di atas menampilkan korelasi antara tahun listing (List Year) dan jumlah penjualan (Sale Amount) untuk 5 kota teratas berdasarkan jumlah transaksi. Setiap titik pada grafik merepresentasikan rata-rata jumlah penjualan pada tahun tertentu di kota tersebut, dengan warna yang berbeda untuk setiap kota (Danbury, Fairfield, Norwalk, Stamford, dan West Hartford). Terlihat adanya variasi tren harga penjualan antar kota dari tahun 2006 hingga 2020. Beberapa kota menunjukkan kecenderungan peningkatan harga penjualan seiring berjalannya waktu, meskipun dengan fluktuasi di beberapa tahun. Perbedaan pola ini mengindikasikan bahwa dinamika pasar properti dapat berbeda secara signifikan antar wilayah.

Analisis lebih lanjut menunjukkan bahwa tidak ada pola tren yang seragam di antara kelima kota tersebut. Stamford tampak mengalami peningkatan rata-rata harga penjualan yang cukup signifikan menjelang tahun 2020. Sementara itu, kota lain seperti Norwalk dan Fairfield menunjukkan fluktuasi yang lebih besar tanpa tren kenaikan yang sejelas Stamford. Danbury dan West Hartford juga menunjukkan variasi harga dari tahun ke tahun. Perbedaan ini bisa disebabkan oleh berbagai faktor lokal seperti perkembangan ekonomi, infrastruktur, permintaan pasar, dan regulasi properti di masing-masing kota. Untuk pemahaman yang lebih mendalam, perlu dipertimbangkan faktor-faktor spesifik yang mempengaruhi pasar properti di setiap kota tersebut.

2. Analisis Tren Penjualan Rumah 2001-2020

```
# Pastikan kolom List Year dan Sale Amount tidak null
df['List Year'] = pd.to_numeric(df['List Year'], errors='coerce')
```

```

df['Sale Amount'] = pd.to_numeric(df['Sale Amount'], errors='coerce')

# Filter tahun 2001 sampai 2020 saja
df_filtered = df[(df['List Year'] >= 2001) & (df['List Year'] <= 2020)]

# Hitung rata-rata harga properti per List Year
rata2_per_tahun = df_filtered.groupby('List Year')['Sale Amount'].mean().sort_index()

# Plot bar chart
plt.figure(figsize=(14,8))
plt.bar(rata2_per_tahun.index.astype(str), rata2_per_tahun.values, color='skyblue', edgecolor='black')

# Menambahkan judul dan label
plt.title('Harga Rata-Rata Properti per Tahun (2001 - 2020)', fontsize=16)
plt.xlabel('Tahun', fontsize=12)
plt.ylabel('Harga Rata-Rata Properti (USD)', fontsize=12)

# Rotasi label x untuk kejelasan
plt.xticks(rotation=45, ha='right', fontsize=10)
plt.yticks(fontsize=10)

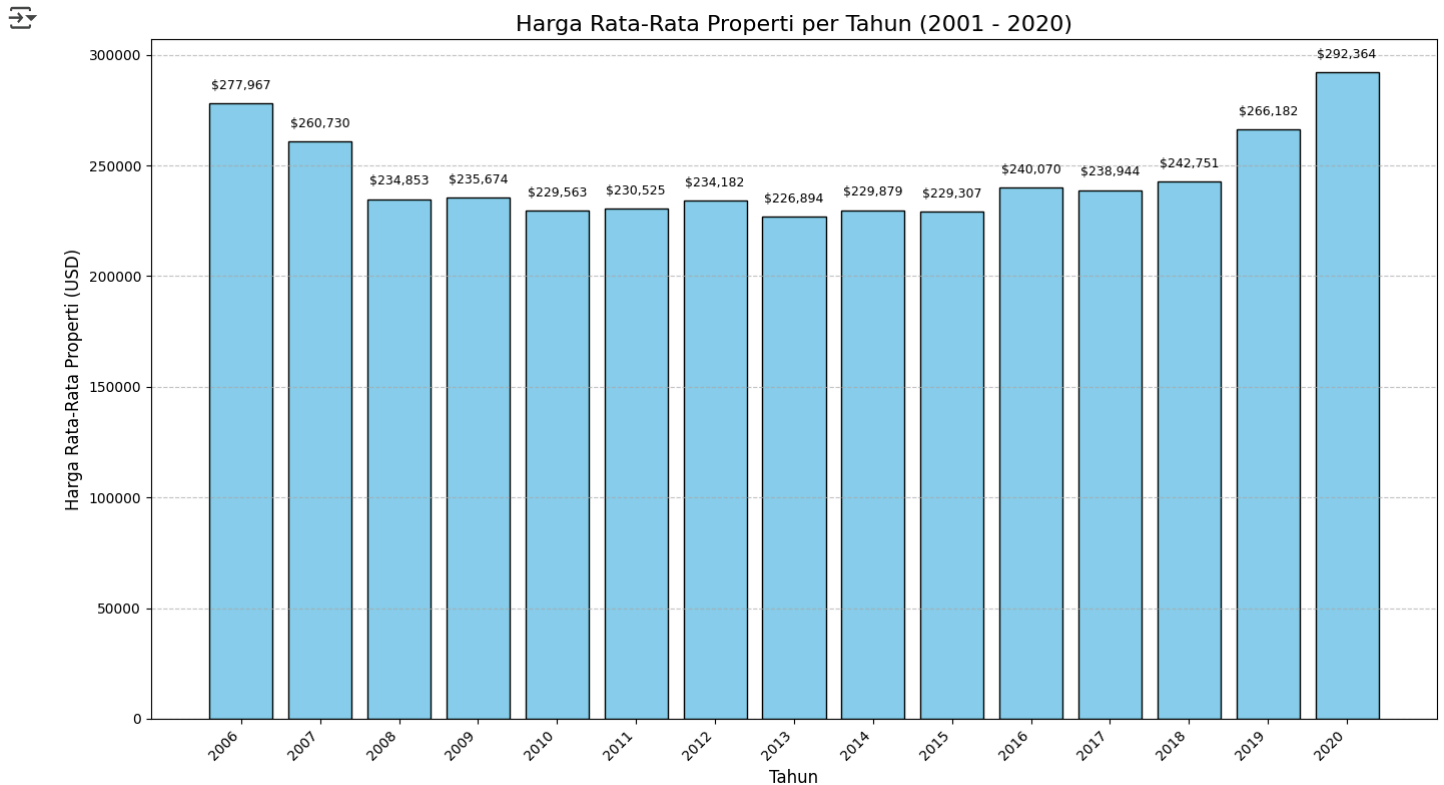
# Menambahkan grid horizontal
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Menambahkan label di atas bar
for i, value in enumerate(rata2_per_tahun.values):
    plt.text(i, value + 5000, f"${value:,.0f}", ha='center', va='bottom', fontsize=9)

# Layout yang rapih
plt.tight_layout()

plt.show()

```



Visualisasi di atas adalah grafik batang yang menampilkan harga rata-rata properti per tahun dari tahun 2006 hingga 2020. Sumbu horizontal menunjukkan tahun, sedangkan sumbu vertikal menunjukkan harga rata-rata properti dalam USD. Setiap batang merepresentasikan harga rata-rata properti pada tahun tertentu, dengan nilai eksaknya tertera di atas setiap batang. Terlihat adanya fluktuasi harga rata-rata properti selama periode waktu tersebut. Harga cenderung menurun setelah tahun 2006, mencapai titik terendah di sekitar tahun 2013-2014, sebelum kemudian menunjukkan tren peningkatan hingga tahun 2020.

Analisis lebih lanjut menunjukkan bahwa harga rata-rata properti mengalami penurunan yang signifikan dari puncaknya di tahun 2006 (277,867) hingga titik terendahnya di tahun 2014 (226,894). Setelah tahun 2014, pasar properti mulai menunjukkan pemulihan dan pertumbuhan, dengan harga rata-rata secara bertahap meningkat hingga mencapai nilai tertinggi kedua di tahun 2020 (\$292,364). Penurunan setelah tahun 2006 kemungkinan dipengaruhi oleh krisis keuangan global, sementara pemulihan setelah tahun 2014 bisa disebabkan oleh berbagai faktor ekonomi dan demografi. Lonjakan harga di tahun 2020 menarik untuk dicermati lebih lanjut, mengingat konteks waktu tersebut.

3. Analisis Tren Penjualan Rumah 2001-2020 Per Tipe Properti

```
# Pastikan kolom valid
df['List Year'] = pd.to_numeric(df['List Year'], errors='coerce')
df['Sale Amount'] = pd.to_numeric(df['Sale Amount'], errors='coerce')

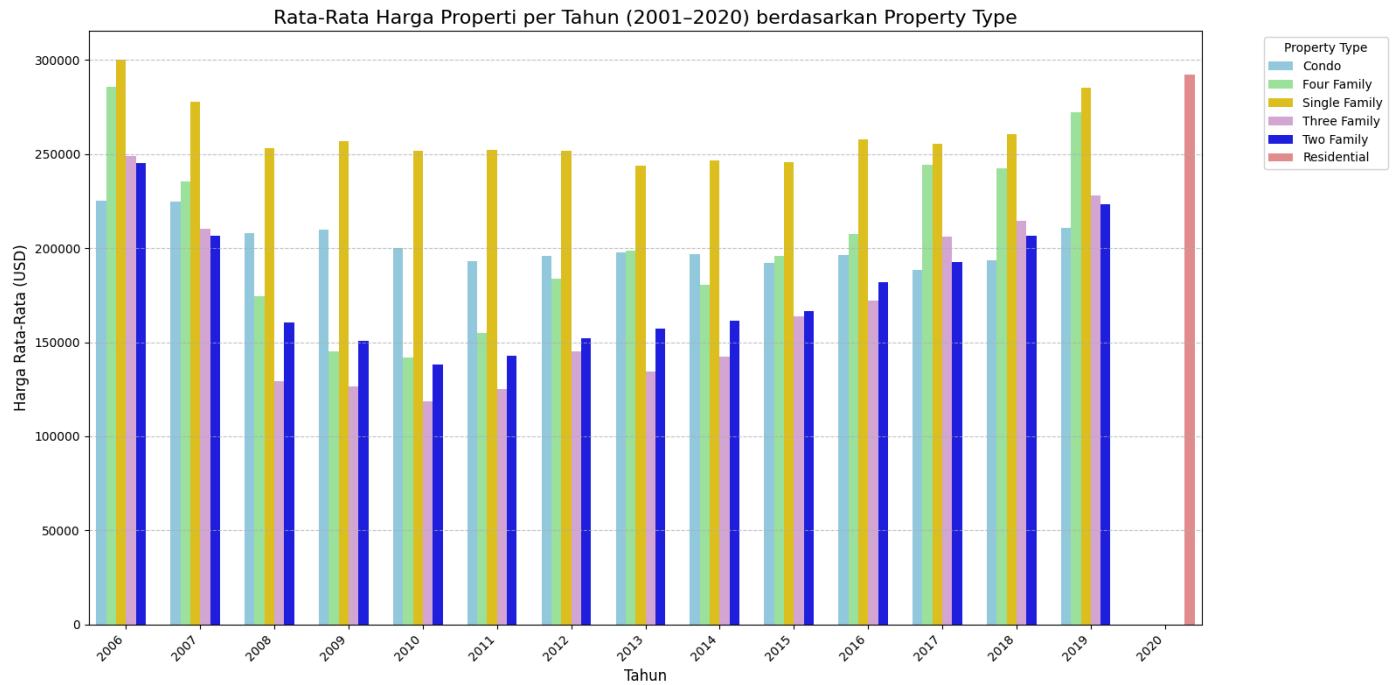
# Filter tahun dan pastikan property type tidak null
df_filtered = df[(df['List Year'] >= 2001) &
                 (df['List Year'] <= 2020) &
                 (df['Sale Amount'] > 0) &
                 (df['Property Type'].notna())]

# Hitung rata-rata Sale Amount per tahun dan Property Type
rata2_per_tahun_ptype = df_filtered.groupby(['List Year', 'Property Type'])['Sale Amount'].mean().reset_index()

# Definisi warna untuk setiap property type
warna_properti = {
    'Condo' : 'skyblue',
    'Four Family' : 'lightgreen',
    'Residential' : 'lightcoral',
    'Single Family' : 'gold',
    'Three Family' : 'Plum',
    'Two Family' : 'blue'
}

# Plot menggunakan seaborn dengan palette warna yang didefinisikan
plt.figure(figsize=(16, 8))
sns.barplot(data=rata2_per_tahun_ptype,
            x='List Year',
            y='Sale Amount',
            hue='Property Type',
            palette=warna_properti)

plt.title('Rata-Rata Harga Properti per Tahun (2001-2020) berdasarkan Property Type', fontsize=16)
plt.xlabel('Tahun', fontsize=12)
plt.ylabel('Harga Rata-Rata (USD)', fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.legend(title='Property Type', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



Visualisasi di atas adalah grafik batang kelompok yang menampilkan rata-rata harga properti per tahun dari 2006 hingga 2020, dibedakan berdasarkan tipe properti (Condo, Four Family, Single Family, Three Family, Two Family, dan Residential). Setiap kelompok batang untuk setiap tahun menunjukkan rata-rata harga untuk masing-masing tipe properti. Terlihat adanya variasi harga rata-rata antar tipe properti dan bagaimana harga-harga ini berubah dari tahun ke tahun. Secara umum, properti Single Family cenderung memiliki rata-rata harga tertinggi di sebagian besar tahun, diikuti oleh Residential. Tipe properti seperti Condo dan Two Family biasanya memiliki rata-rata harga yang lebih rendah.

Analisis lebih lanjut mengungkapkan bahwa tren harga dari tahun 2006 hingga 2020 tidak seragam di semua tipe properti. Setelah penurunan harga secara umum di beberapa tahun awal periode, sebagian besar tipe properti menunjukkan tren peningkatan harga menjelang tahun 2020. Namun, tingkat pemulihan dan pertumbuhan harga bervariasi antar tipe properti. Misalnya, Single Family dan Residential seringkali menunjukkan nilai rata-rata yang lebih tinggi dan perubahan yang lebih signifikan dibandingkan dengan tipe properti dengan unit ganda atau tiga keluarga. Fluktuasi pasar properti tampaknya mempengaruhi setiap segmen dengan cara yang berbeda, yang mungkin disebabkan oleh dinamika permintaan dan penawaran yang spesifik untuk setiap jenis properti.

✓ 4. Pemodelan Forecasting Sale Amount 2021 Untuk Setiap Kota

```
df.info()
```



```
<class 'pandas.core.frame.DataFrame'>
Index: 558211 entries, 1 to 996559
Data columns (total 10 columns):
#   Column          Non-Null Count  Dtype
---  -
0   Serial Number    558211 non-null float64
1   List Year        558211 non-null int64
2   Date Recorded    558211 non-null datetime64[ns]
3   Town             558211 non-null object
4   Address          558211 non-null object
```



```
5   Assessed Value    558211 non-null float64
6   Sale Amount       558211 non-null float64
7   Sales Ratio       558211 non-null float64
8   Property Type     558211 non-null object
9   Residential Type   558211 non-null object
dtypes: datetime64[ns](1), float64(4), int64(1), object(4)
memory usage: 46.8+ MB
```

```
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
import itertools
import warnings
```

```
warnings.filterwarnings("ignore")
```

```
# Filter data untuk kota Bridgeport
bridgeport_df = df[df['Town'] == 'Bridgeport']
```

```
# Agregasi rata-rata Sale Amount per List Year
annual_sales = bridgeport_df.groupby('List Year')['Sale Amount'].mean().sort_index()
```

```
# Gunakan hanya data sampai 2020 (untuk prediksi 2021)
ts = annual_sales[annual_sales.index <= 2020]
```

```
# Grid search untuk parameter ARIMA (p, d, q)
p = d = q = range(0, 3)
pdq_combinations = list(itertools.product(p, d, q))
```