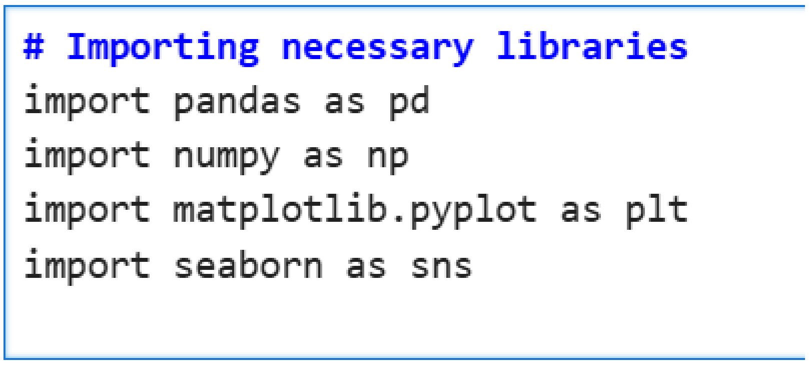
Case Study-2

Dataset: **mudah-apartment-kl-selangor**

Name: **E.R Harish**

Output:

1. **Cell 1:**

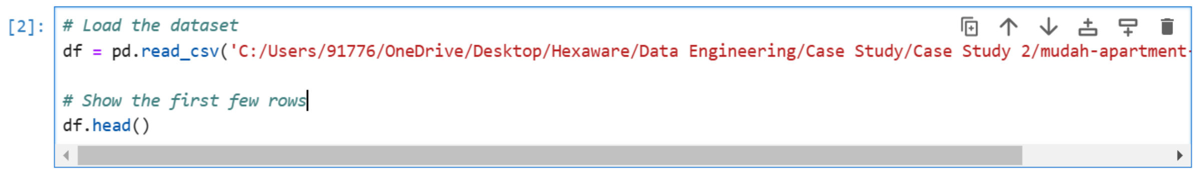


```python
# Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Explanation: This imports the following essential libraries for data analysis and visualization:

- **pandas (as pd)**: Used for data manipulation and analysis, especially working with tabular data.
- **numpy (as np)**: Provides support for numerical operations and arrays.
- **matplotlib.pyplot (as plt)**: A plotting library for creating static, animated, and interactive visualizations.
- **seaborn (as sns)**: Built on matplotlib, used for creating more advanced and attractive statistical plots.

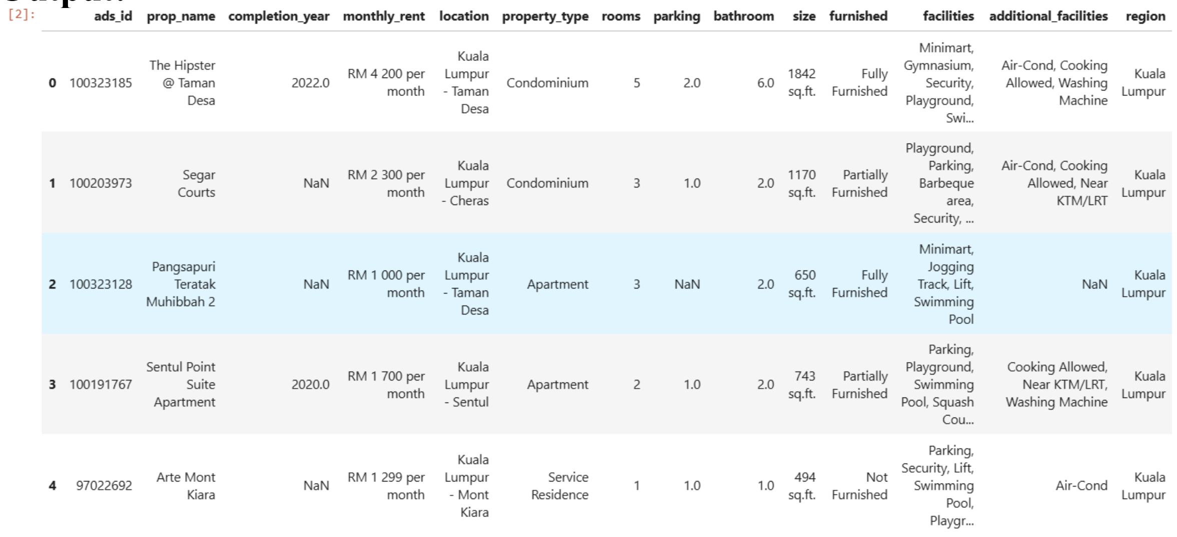
2. Cell 2:



```
[2]: # Load the dataset
     df = pd.read_csv('C:/Users/91776/OneDrive/Desktop/Hexaware/Data Engineering/Case Study/Case Study 2/mudah-apartment

     # Show the first few rows
     df.head()
```

Output:



[2]:

	ads_id	prop_name	completion_year	monthly_rent	location	property_type	rooms	parking	bathroom	size	furnished	facilities	additional_facilities	region
0	100323185	The Hipster @ Taman Desa	2022.0	RM 4 200 per month	Kuala Lumpur - Taman Desa	Condominium	5	2.0	6.0	1842 sq.ft.	Fully Furnished	Minimart, Gymnasium, Security, Playground, Swi...	Air-Cond, Cooking Allowed, Washing Machine	Kuala Lumpur
1	100203973	Segar Courts	NaN	RM 2 300 per month	Kuala Lumpur - Cheras	Condominium	3	1.0	2.0	1170 sq.ft.	Partially Furnished	Playground, Parking, Barbeque area, Security, ...	Air-Cond, Cooking Allowed, Near KTM/LRT	Kuala Lumpur
2	100323128	Pangsapuri Teratak Muhibbah 2	NaN	RM 1 000 per month	Kuala Lumpur - Taman Desa	Apartment	3	NaN	2.0	650 sq.ft.	Fully Furnished	Minimart, Jogging Track, Lift, Swimming Pool	NaN	Kuala Lumpur
3	100191767	Sentul Point Suite Apartment	2020.0	RM 1 700 per month	Kuala Lumpur - Sentul	Apartment	2	1.0	2.0	743 sq.ft.	Partially Furnished	Parking, Playground, Swimming Pool, Squash Cou...	Cooking Allowed, Near KTM/LRT, Washing Machine	Kuala Lumpur
4	97022692	Arte Mont Kiara	NaN	RM 1 299 per month	Kuala Lumpur - Mont Kiara	Service Residence	1	1.0	1.0	494 sq.ft.	Not Furnished	Parking, Security, Lift, Swimming Pool, Playgr...	Air-Cond	Kuala Lumpur

Explanation:

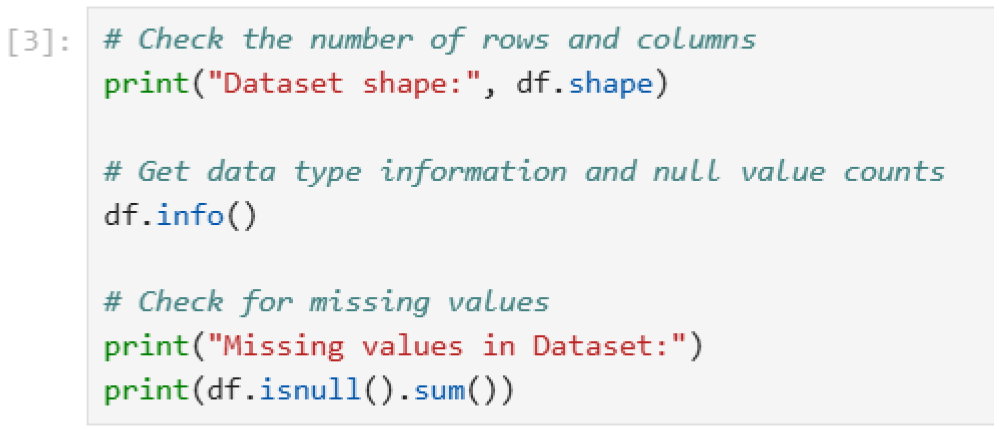
a. Load the dataset:

- **pd.read_csv('path_to_file'):** Reads a CSV file located at the specified path into a pandas DataFrame named df.
- The file contains data related to **apartments in KL Selangor.**

b. Display the first few rows:

- **df.head():** Displays the first 5 rows of the DataFrame by default. This helps to get a quick overview of the dataset's structure and content.

3. **Cell 3:**

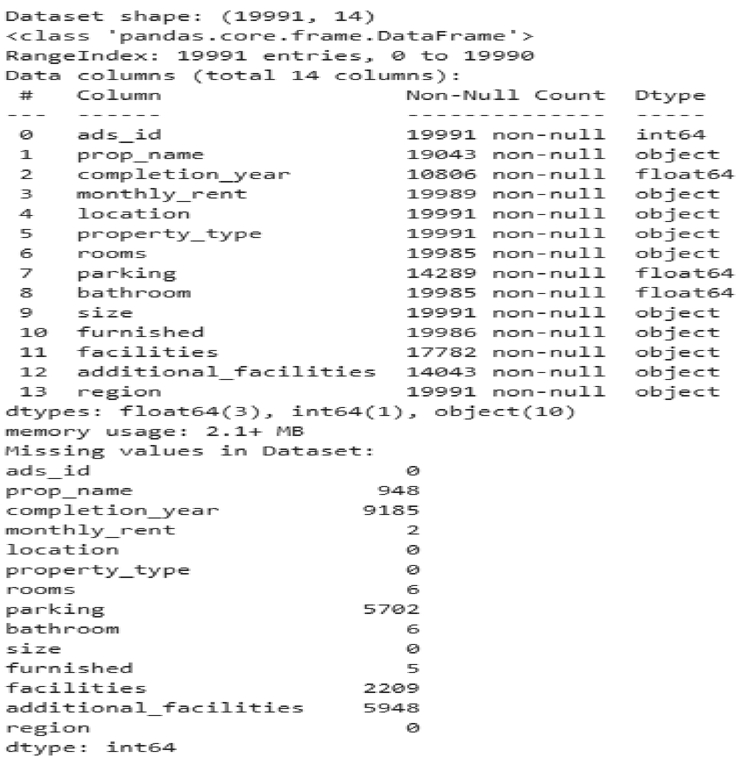


```
[3]: # Check the number of rows and columns
     print("Dataset shape:", df.shape)

     # Get data type information and null value counts
     df.info()

     # Check for missing values
     print("Missing values in Dataset:")
     print(df.isnull().sum())
```

Output:



```
Dataset shape: (19991, 14)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19991 entries, 0 to 19990
Data columns (total 14 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   ads_id                 19991 non-null  int64  
 1   prop_name              19043 non-null  object 
 2   completion_year        10806 non-null  float64
 3   monthly_rent           19989 non-null  object 
 4   location               19991 non-null  object 
 5   property_type          19991 non-null  object 
 6   rooms                  19985 non-null  object 
 7   parking                14289 non-null  float64
 8   bathroom               19985 non-null  float64
 9   size                   19991 non-null  object 
 10  furnished              19986 non-null  object 
 11  facilities             17782 non-null  object 
 12  additional_facilities  14043 non-null  object 
 13  region                 19991 non-null  object 
dtypes: float64(3), int64(1), object(10)
memory usage: 2.1+ MB
Missing values in Dataset:
ads_id                      0
prop_name                 948
completion_year          9185
monthly_rent                2
location                    0
property_type               0
rooms                       6
parking                  5702
bathroom                    6
size                        0
furnished                   5
facilities               2209
additional_facilities    5948
region                      0
dtype: int64
```

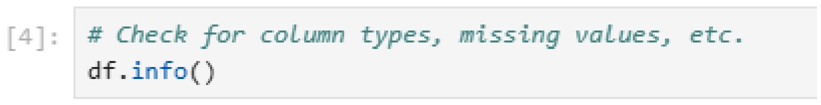
Explanation:

a. **Check the shape of the dataset:**
   - **df.shape:** Returns the number of rows and columns in the dataset.

b. **Get data type information and memory usage:**
   - **df.info():** Displays column names, data types, non-null counts, and memory usage.

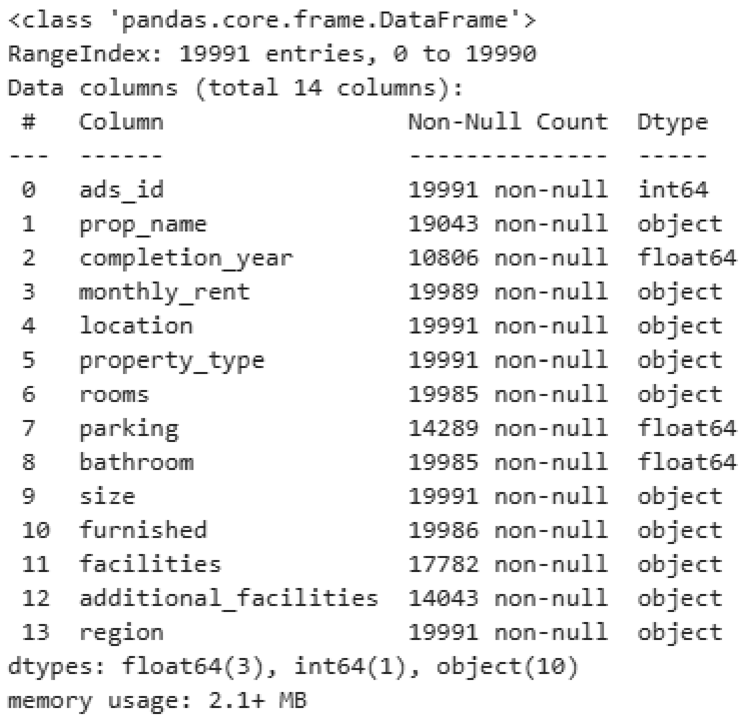
c. **Check for missing values:**
   - **df.isnull().sum():** Computes the total number of missing (NaN) values in each column.

4. **Cell 4:**



```
[4]: # Check for column types, missing values, etc.
     df.info()
```

Output:



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19991 entries, 0 to 19990
Data columns (total 14 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   ads_id                 19991 non-null  int64  
 1   prop_name              19043 non-null  object 
 2   completion_year        10806 non-null  float64
 3   monthly_rent           19989 non-null  object 
 4   location               19991 non-null  object 
 5   property_type          19991 non-null  object 
 6   rooms                  19985 non-null  object 
 7   parking                14289 non-null  float64
 8   bathroom               19985 non-null  float64
 9   size                   19991 non-null  object 
 10  furnished              19986 non-null  object 
 11  facilities             17782 non-null  object 
 12  additional_facilities  14043 non-null  object 
 13  region                 19991 non-null  object 
dtypes: float64(3), int64(1), object(10)
memory usage: 2.1+ MB
```

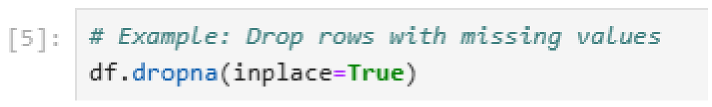
Explanation:

a. **Column names and types:**
   - Lists all columns along with their respective data types (e.g., int64, float64, object).

b. **Non-null counts:**
   - Shows the number of non-missing (non-null) values in each column.
   - Useful for identifying columns with missing data.

c. **Memory usage:**
   - Displays the amount of memory being used by the dataset.

5. **Cell 5:**

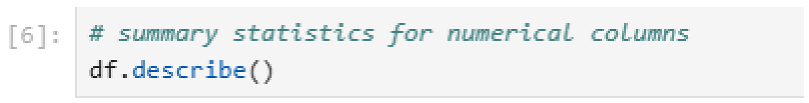


```
[5]: # Example: Drop rows with missing values
     df.dropna(inplace=True)
```

Explanation:

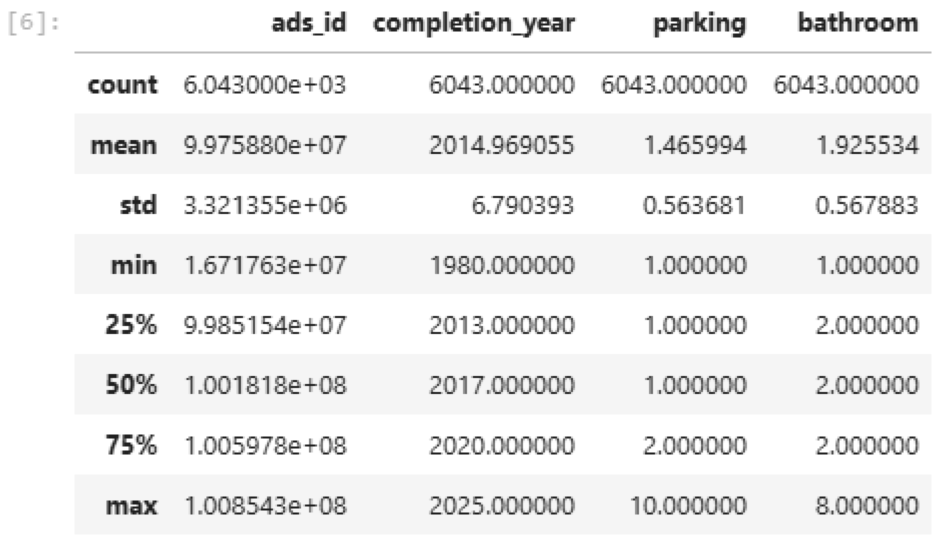
a. **df.dropna(inplace=True):**
   - **dropna():** Drops all rows that contain at least one missing (NaN) value.
   - **inplace=True:** Modifies the original DataFrame directly instead of returning a new one.

6. **Cell 6:**



```
[6]: # summary statistics for numerical columns
     df.describe()
```

Output:



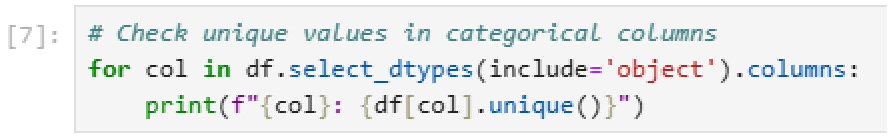
[6]:

	ads_id	completion_year	parking	bathroom
count	6.043000e+03	6043.000000	6043.000000	6043.000000
mean	9.975880e+07	2014.969055	1.465994	1.925534
std	3.321355e+06	6.790393	0.563681	0.567883
min	1.671763e+07	1980.000000	1.000000	1.000000
25%	9.985154e+07	2013.000000	1.000000	2.000000
50%	1.001818e+08	2017.000000	1.000000	2.000000
75%	1.005978e+08	2020.000000	2.000000	2.000000
max	1.008543e+08	2025.000000	10.000000	8.000000

Explanation:

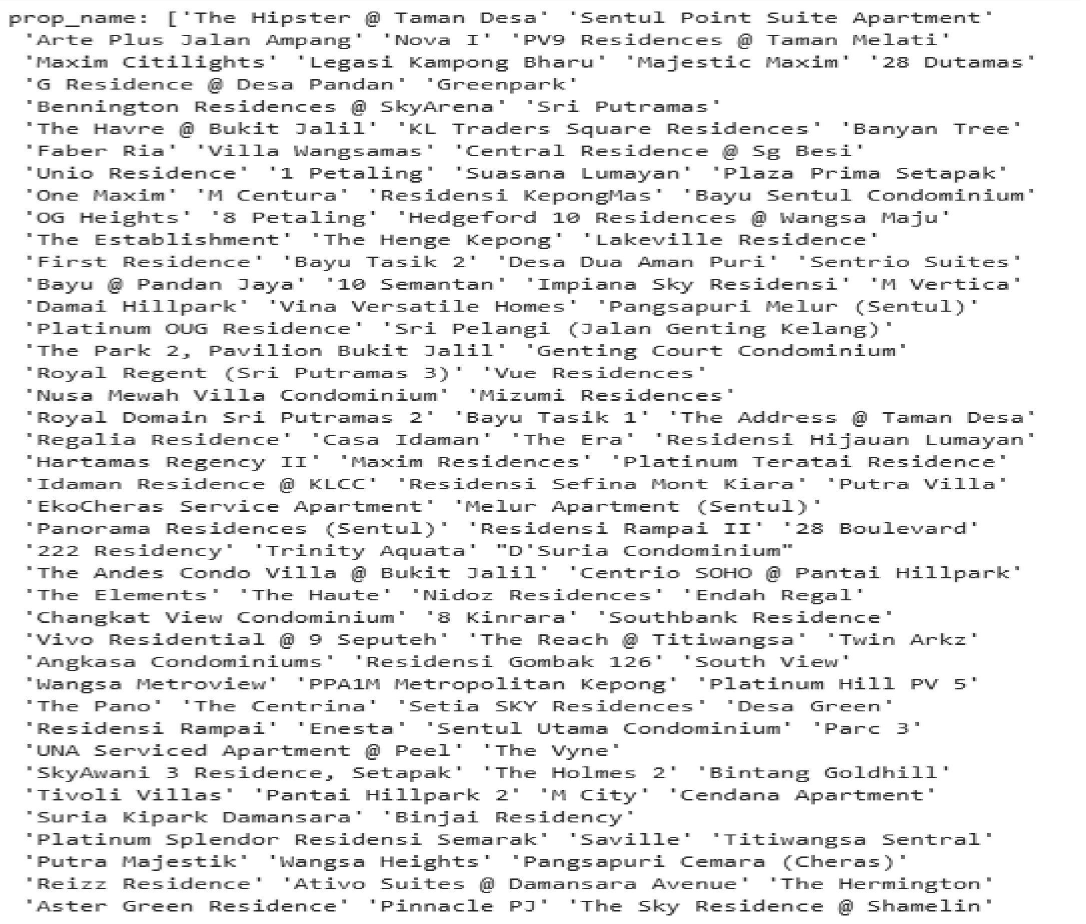
a. **df.describe():**
   - Provides a statistical summary of numerical columns in the DataFrame.

7. Cell 7:



```
[7]: # Check unique values in categorical columns
     for col in df.select_dtypes(include='object').columns:
         print(f"{col}: {df[col].unique()}")
```

Output:



```
prop_name: ['The Hipster @ Taman Desa' 'Sentul Point Suite Apartment'
 'Arte Plus Jalan Ampang' 'Nova I' 'PV9 Residences @ Taman Melati'
 'Maxim Citilights' 'Legasi Kampong Bharu' 'Majestic Maxim' '28 Dutamas'
 'G Residence @ Desa Pandan' 'Greenpark'
 'Bennington Residences @ SkyArena' 'Sri Putramas'
 'The Havre @ Bukit Jalil' 'KL Traders Square Residences' 'Banyan Tree'
 'Faber Ria' 'Villa Wangsamas' 'Central Residence @ Sg Besi'
 'Unio Residence' '1 Petaling' 'Suasana Lumayan' 'Plaza Prima Setapak'
 'One Maxim' 'M Centura' 'Residensi KepongMas' 'Bayu Sentul Condominium'
 'OG Heights' '8 Petaling' 'Hedgeford 10 Residences @ Wangsa Maju'
 'The Establishment' 'The Henge Kepong' 'Lakeville Residence'
 'First Residence' 'Bayu Tasik 2' 'Desa Dua Aman Puri' 'Sentrio Suites'
 'Bayu @ Pandan Jaya' '10 Semantan' 'Impiana Sky Residensi' 'M Vertica'
 'Damai Hillpark' 'Vina Versatile Homes' 'Pangsapuri Melur (Sentul)'
 'Platinum OUG Residence' 'Sri Pelangi (Jalan Genting Kelang)'
 'The Park 2, Pavilion Bukit Jalil' 'Genting Court Condominium'
 'Royal Regent (Sri Putramas 3)' 'Vue Residences'
 'Nusa Mewah Villa Condominium' 'Mizumi Residences'
 'Royal Domain Sri Putramas 2' 'Bayu Tasik 1' 'The Address @ Taman Desa'
 'Regalia Residence' 'Casa Idaman' 'The Era' 'Residensi Hijauan Lumayan'
 'Hartamas Regency II' 'Maxim Residences' 'Platinum Teratai Residence'
 'Idaman Residence @ KLCC' 'Residensi Sefina Mont Kiara' 'Putra Villa'
 'EkoCheras Service Apartment' 'Melur Apartment (Sentul)'
 'Panorama Residences (Sentul)' 'Residensi Rampai II' '28 Boulevard'
 '222 Residency' 'Trinity Aquata' "D'Suria Condominium"
 'The Andes Condo Villa @ Bukit Jalil' 'Centrio SOHO @ Pantai Hillpark'
 'The Elements' 'The Haute' 'Nidoz Residences' 'Endah Regal'
 'Changkat View Condominium' '8 Kinrara' 'Southbank Residence'
 'Vivo Residential @ 9 Seputeh' 'The Reach @ Titiwangsa' 'Twin Arkz'
 'Angkasa Condominiums' 'Residensi Gombak 126' 'South View'
 'Wangsa Metroview' 'PPA1M Metropolitan Kepong' 'Platinum Hill PV 5'
 'The Pano' 'The Centrina' 'Setia SKY Residences' 'Desa Green'
 'Residensi Rampai' 'Enesta' 'Sentul Utama Condominium' 'Parc 3'
 'UNA Serviced Apartment @ Peel' 'The Vyne'
 'SkyAwani 3 Residence, Setapak' 'The Holmes 2' 'Bintang Goldhill'
 'Tivoli Villas' 'Pantai Hillpark 2' 'M City' 'Cendana Apartment'
 'Suria Kipark Damansara' 'Binjai Residency'
 'Platinum Splendor Residensi Semarak' 'Saville' 'Titiwangsa Sentral'
 'Putra Majestik' 'Wangsa Heights' 'Pangsapuri Cemara (Cheras)'
 'Reizz Residence' 'Ativo Suites @ Damansara Avenue' 'The Hermington'
 'Aster Green Residence' 'Pinnacle PJ' 'The Sky Residence @ Shamelin'
```

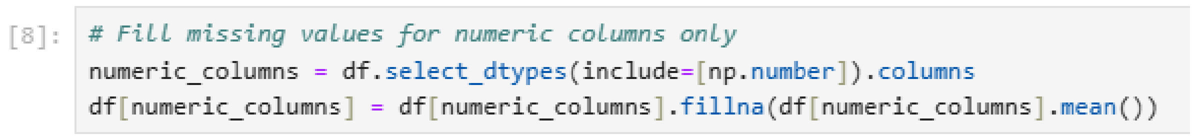
Explanation:

a. **Select categorical columns:**
   - **df.select_dtypes(include='object'):** Filters columns with data type object (typically used for categorical or string data).

b. **Iterate over the categorical columns:**
   - Loops through each column in the filtered DataFrame.

c. **Print unique values:**
   - **df[col].unique():** Retrieves and prints all unique values in the current column.

8. **Cell 8:**

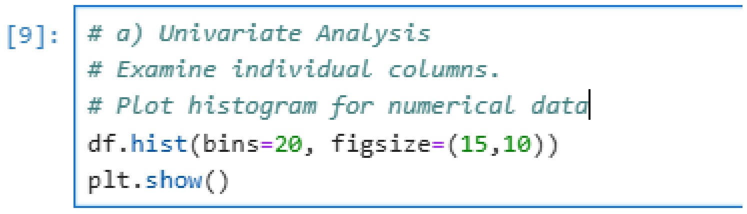


```
[8]: # Fill missing values for numeric columns only
     numeric_columns = df.select_dtypes(include=[np.number]).columns
     df[numeric_columns] = df[numeric_columns].fillna(df[numeric_columns].mean())
```

Explanation:

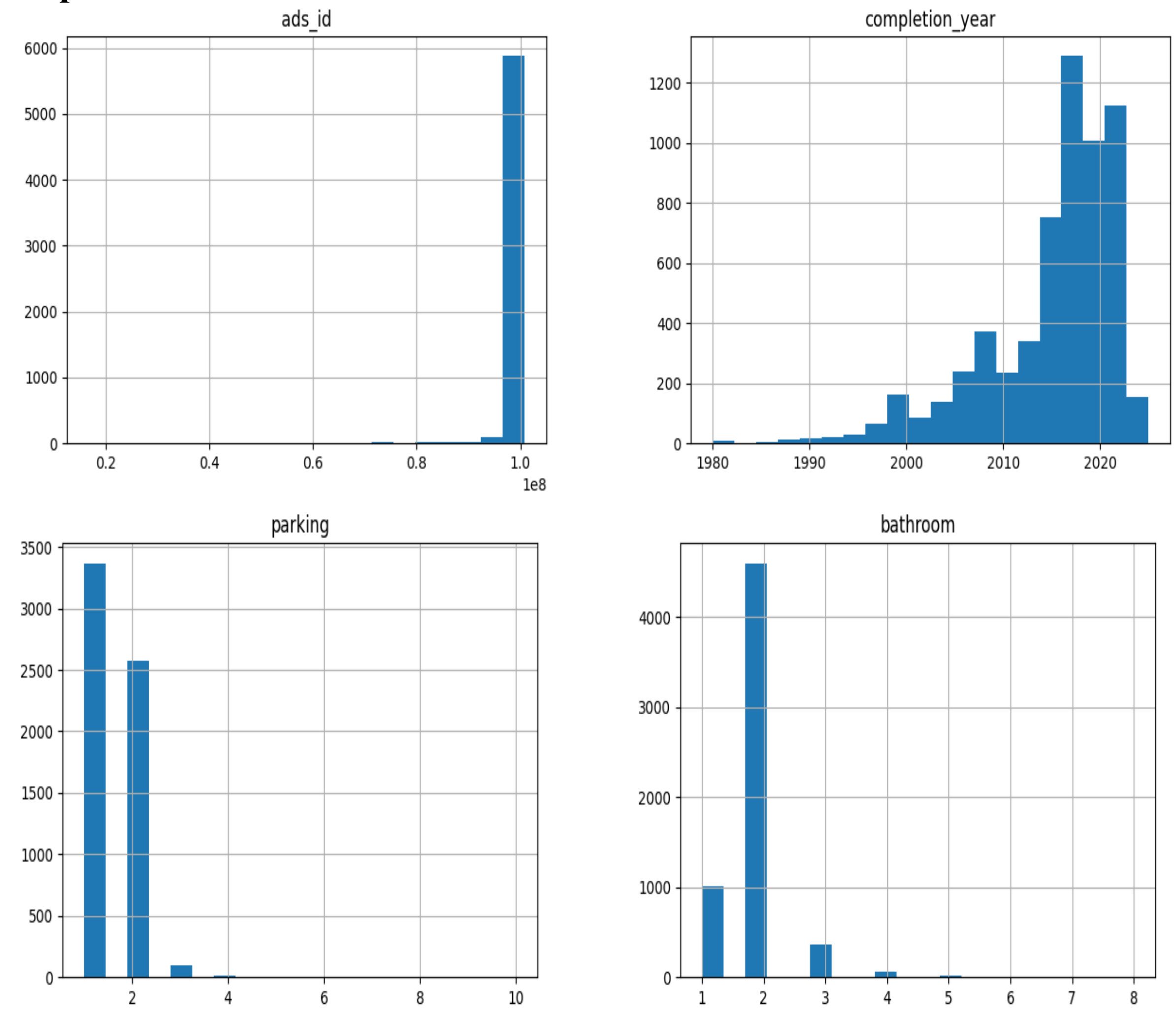
a. **df.select_dtypes(include=[np.number]).columns:** Selects columns with numeric data types (integers or floats) in the DataFrame.
b. **df[numeric_columns]:** Selects the numeric columns from the DataFrame.
c. **.fillna(df[numeric_columns].mean()):** Fills missing values (NaN) in these numeric columns with the mean of the respective column.

9. **Cell 9:**



```
[9]: # a) Univariate Analysis
     # Examine individual columns.
     # Plot histogram for numerical data
     df.hist(bins=20, figsize=(15,10))
     plt.show()
```

Output:

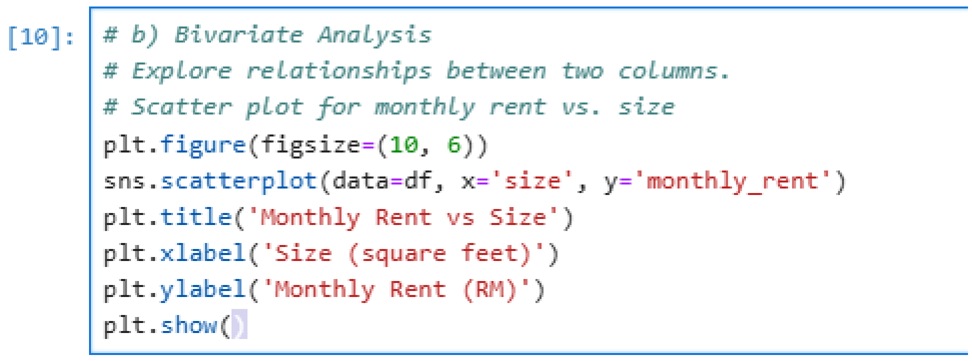


Explanation:

a. **df.hist(bins=20, figsize=(15,10)):** This creates histograms for each numeric column in the DataFrame. The bins=20 parameter specifies that the data will be divided into 20 bins for each histogram. The figsize=(15,10) adjusts the overall size of the plot for better visualization.

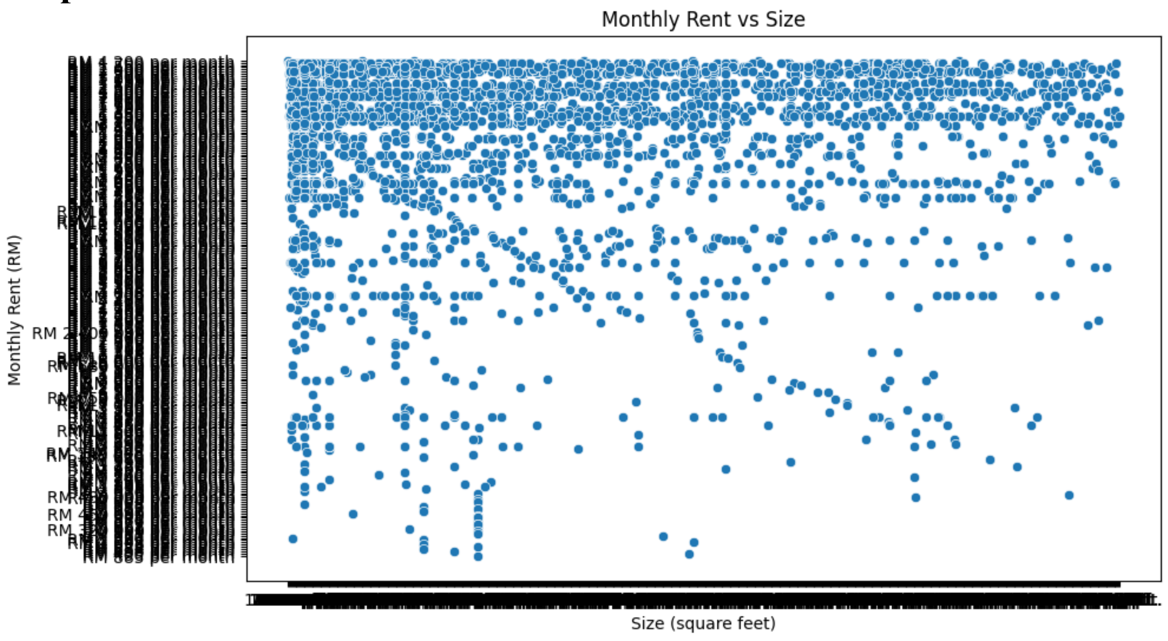
b. **plt.show():** Displays the histograms.

10. Cell 10:



```
[10]: # b) Bivariate Analysis
      # Explore relationships between two columns.
      # Scatter plot for monthly rent vs. size
      plt.figure(figsize=(10, 6))
      sns.scatterplot(data=df, x='size', y='monthly_rent')
      plt.title('Monthly Rent vs Size')
      plt.xlabel('Size (square feet)')
      plt.ylabel('Monthly Rent (RM)')
      plt.show()
```

Output:



Explanation:

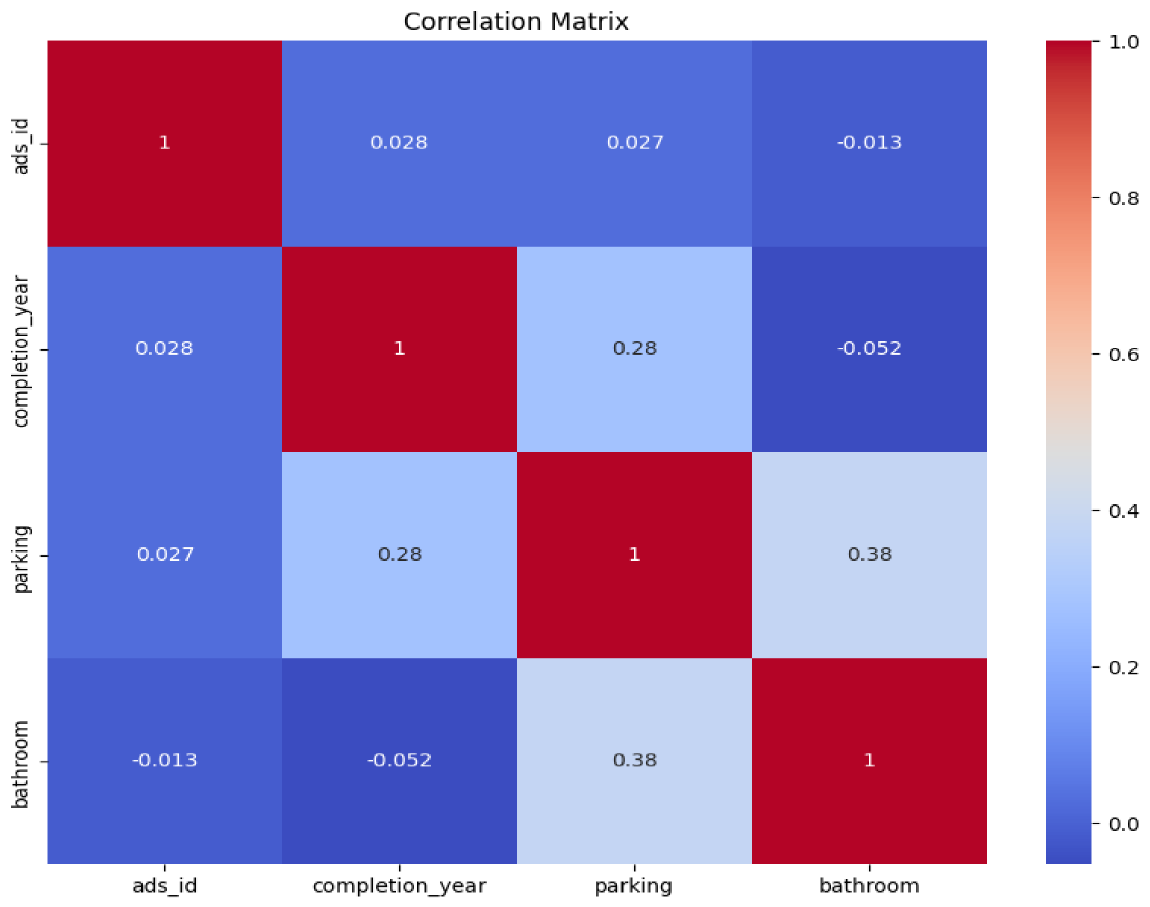
a. **plt.figure(figsize=(10, 6)):** Sets the figure size for the plot.
b. **sns.scatterplot(data=df, x='size', y='monthly_rent'):** Creates a scatter plot with size on the x-axis and monthly_rent on the y-axis using Seaborn's scatterplot function. It visualizes how changes in size are related to changes in monthly rent.
c. **plt.title('Monthly Rent vs Size'):** Adds a title to the plot.
d. **plt.xlabel('Size (square feet)'):** Labels the x-axis as "Size (square feet)".
e. **plt.ylabel('Monthly Rent (RM)'):** Labels the y-axis as "Monthly Rent (RM)".
f. **plt.show():** Displays the plot.

11. Cell 11:

```
[12]: # c) Correlation Analysis
      # Check correlation between numerical columns.
      # Select only numeric columns for correlation analysis
      numeric_df = df.select_dtypes(include=[np.number])

      # Display correlation matrix for numeric columns
      plt.figure(figsize=(10, 8))
      sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm')
      plt.title('Correlation Matrix')
      plt.show()
```

Output:



Explanation:

a. **numeric_df = df.select_dtypes(include=[np.number]):** This line selects only the numeric columns from the DataFrame (df) for correlation analysis.

b. numeric_df.corr(): Computes the correlation matrix, which shows the pairwise correlation between the numeric columns. The values range from -1 (perfect negative correlation) to 1 (perfect positive correlation).

c. plt.figure(figsize=(10, 8)): Adjusts the figure size of the plot.

d. **sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm'):** Creates a heatmap of the correlation matrix using Seaborn's heatmap function. The annot=True argument displays the correlation values in the heatmap cells, and cmap='coolwarm' specifies the color palette for the heatmap (blue for negative correlations and red for positive correlations).

e. plt.title('Correlation Matrix'): Adds a title to the heatmap**.**

f. plt.show(): Displays the heatmap.

1. **Summary and Conclusion:**
   - **Summary of Findings:**

     Throughout this exploratory data analysis (EDA), several key insights emerged regarding the apartment rental market in Malaysia, specifically in Kuala Lumpur and Selangor. Here are the main observations:

     - Rental Prices and Apartment Size: A positive correlation between apartment size (square footage) and monthly rental prices was observed, meaning larger apartments generally command higher rent. However, the scatter plot also showed outliers, which could indicate high-demand areas or luxury apartments with higher-than-average rents for a given size.
     - Location Impact: The data showed that location significantly impacts rental prices. Apartments in premium locations in Kuala Lumpur, such as the city center, tend to have higher rental rates than those in surrounding areas. This suggests that location, alongside apartment size, is a major factor in determining rental prices.
     - Property Type and Amenities: Properties with more amenities or additional facilities, such as parking, security, and furnished options, showed a tendency for higher rental prices. This is in line with expectations, as additional features enhance property value and attractiveness.
     - Room and Bathroom Count: A moderate correlation was observed between the number of rooms and bathrooms with rental prices, likely because larger apartments with more rooms are more suitable for families or groups, leading to higher demand.

   - **Conclusion:**

   This analysis provided insights into the factors influencing apartment rental prices in Malaysia, highlighting the importance of size, location, and amenities. By understanding these factors, renters and real estate investors can make more informed decisions, and property managers can better tailor their offerings to meet market demand. The EDA also emphasized the importance of data cleaning, as it affects the quality and reliability of the analysis.

2. **Interesting Insights and Graphs**
    - Scatter Plot of Monthly Rent vs. Size: This graph demonstrated the positive relationship between apartment size and rental price, allowing for easy identification of outliers.
    - Heatmap of Correlation Matrix: The correlation matrix visually displayed the relationships between numerical variables, highlighting which features are most associated with higher rental prices.
    - Bar Plot of Average Rent by Location: This plot provided a clear view of how average rental prices vary across different regions, with central areas showing higher averages.

3. **Ideas for Future Work**
    - **Incorporate Additional Datasets:** To deepen the analysis, we could incorporate additional data, such as:
        - **Economic indicators** (e.g., average income in different regions) to study affordability.
        - **Public transportation proximity** to explore its impact on rental prices.
        - **Crime rates by neighborhood** to assess the influence of safety on rental values.

    - **Time Series Analysis:** If rental data were available across different years or months, a time series analysis could reveal rental price trends over time, showing how prices have evolved and predicting future trends.

    - **Sentiment Analysis on Property Descriptions:** Analyzing the sentiment or keyword frequencies in property descriptions could uncover additional factors influencing rental decisions.
    - **Machine Learning Models:** Using this cleaned dataset, machine learning models such as linear regression or decision trees could predict rental prices based on property features, providing valuable insights for property valuation.

4. Resources:

Here are some resources that were helpful during this analysis:

- Seaborn Documentation: For creating statistical visualizations and heatmaps: Seaborn Documentation
- Pandas Documentation: To explore and manipulate data: Pandas Documentation
- NumPy Documentation: For handling numerical operations efficiently: NumPy Documentation
- Matplotlib Documentation: For plotting and data visualization: Matplotlib Documentation