Python-Case Study

Submitted By-

Aniroop Gupta
DE Batch1

Dataset: mudah-apartment-kl-selangor

1. Importing necessary libraries:

[3]:
```
# Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Display plots inline (optional, only needed in some Jupyter setups)
%matplotlib inline
```

2. Loading the given Dataset:

[4]:
```
# Load the dataset
df = pd.read_csv('mudah-apartment-kl-selangor.csv')

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

[4]:

	ads_id	prop_name	completion_year	monthly_rent	location	property_type	rooms	parking	bathroom	size	furnished	facilities
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...
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, ...
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
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...
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

Explanation: pd.read_csv() is a function in pandas used to read a CSV (Comma-Separated Values) file. The result is a DataFrame, a 2D structure similar to a table in Excel, with rows and columns.

3. Number of rows and columns, data type information:

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

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

```
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)
```

Explanation: df.shape, returns a tuple representing the DataFrame's dimensions: (number of rows, number of columns).

df.info(), summarizes the DataFrame, including column names, data types, non-null value counts, and memory usage.

4. Checking missing values:

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

Explanation: ‘df.isnull()’ creates a DataFrame of the same shape, marking True for null values and False otherwise. ‘sum()’ calculates the total count of True values (null entries) for each column, providing a summary of missing data.

```
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
```

5. Checking column types, missing values, etc.:

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

```
<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
```

6. Drop missing values rows:

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

Explanation: Above formula removes all rows from the DataFrame that contain any missing (null) values. It is useful for cleaning the dataset by eliminating incomplete records.

7. Summary statistics for numerical columns:

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

[8]:

	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: The ‘df.describe()’ function generates summary statistics for all numerical columns in the DataFrame. This function helps in understanding the distribution, central tendency, and variability of the numerical data in the dataset.

8. Checking unique values in categorical columns:

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

```
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)'
```

```
monthly_rent: ['RM 4 200 per month' 'RM 1 700 per month' 'RM 1 550 per month'
 'RM 1 400 per month' 'RM 2 000 per month' 'RM 1 500 per month'
 'RM 1 300 per month' 'RM 3 200 per month' 'RM 1 099 per month'
 'RM 1 199 per month' 'RM 1 100 per month' 'RM 2 500 per month'
 'RM 2 900 per month' 'RM 4 500 per month' 'RM 1 800 per month'
 'RM 3 000 per month' 'RM 1 600 per month' 'RM 7 000 per month'
 'RM 7 800 per month' 'RM 1 750 per month' 'RM 1 900 per month'
 'RM 2 100 per month' 'RM 2 300 per month' 'RM 1 350 per month'
 'RM 1 650 per month' 'RM 1 200 per month' 'RM 2 599 per month'
 'RM 2 200 per month' 'RM 3 300 per month' 'RM 850 per month'
 'RM 2 499 per month' 'RM 1 948 per month' 'RM 1 488 per month'
 'RM 1 880 per month' 'RM 2 600 per month' 'RM 2 800 per month'
 'RM 5 800 per month' 'RM 1 850 per month' 'RM 1 499 per month'
 'RM 4 800 per month' 'RM 4 600 per month' 'RM 3 500 per month'
 'RM 2 400 per month' 'RM 650 per month' 'RM 2 350 per month'
 'RM 2 250 per month' 'RM 2 700 per month' 'RM 600 per month'
 'RM 1 450 per month' 'RM 1 999 per month' 'RM 1 180 per month'
 'RM 2 450 per month' 'RM 3 600 per month' 'RM 2 650 per month'
 'RM 950 per month' 'RM 1 000 per month' 'RM 5 950 per month'
 'RM 2 899 per month' 'RM 1 799 per month' 'RM 2 480 per month'
```

```
location: ['Kuala Lumpur - Taman Desa' 'Kuala Lumpur - Sentul'
 'Kuala Lumpur - Ampang' 'Kuala Lumpur - Segambut'
 'Kuala Lumpur - Setapak' 'Kuala Lumpur - KL City' 'Kuala Lumpur - Cheras'
 'Kuala Lumpur - Solaris Dutamas' 'Kuala Lumpur - Desa Pandan'
 'Kuala Lumpur - Old Klang Road' 'Kuala Lumpur - Jalan Kuching'
 'Kuala Lumpur - Bukit Jalil' 'Kuala Lumpur - Gombak'
 'Kuala Lumpur - KLCC' 'Kuala Lumpur - Wangsa Maju'
 'Kuala Lumpur - Sungai Besi' 'Kuala Lumpur - Kepong'
 'Kuala Lumpur - Pantai' 'Kuala Lumpur - Sri Petaling'
 'Kuala Lumpur - Brickfields' 'Kuala Lumpur - Jalan Ipoh'
 'Kuala Lumpur - Pandan Jaya' 'Kuala Lumpur - Damansara Heights'
 'Kuala Lumpur - Bandar Damai Perdana' 'Kuala Lumpur - Kuchai Lama'
 'Kuala Lumpur - Titiwangsa' 'Kuala Lumpur - Jalan Sultan Ismail'
 'Kuala Lumpur - Mont Kiara' 'Kuala Lumpur - Pandan Perdana'
 'Kuala Lumpur - Keramat' 'Kuala Lumpur - Desa Petaling'
 'Kuala Lumpur - Puchong' 'Kuala Lumpur - Seputeh'
 'Kuala Lumpur - Bangsar South' 'Kuala Lumpur - Bangsar'
 'Kuala Lumpur - Taman Melawati' 'Kuala Lumpur - Ampang Hilir'
 'Kuala Lumpur - Sri Damansara' 'Kuala Lumpur - Bukit Bintang'
 'Kuala Lumpur - Taman Tun Dr Ismail' 'Kuala Lumpur - Setiawangsa'
 'Kuala Lumpur - Sri Hartamas' 'Kuala Lumpur - Mid Valley City'
 'Kuala Lumpur - City Centre' 'Kuala Lumpur - KL Sentral'
```

```
property_type: ['Condominium' 'Apartment' 'Service Residence' 'Studio' 'Duplex' 'Others'
 'Townhouse Condo' 'Flat']
rooms: ['5' '2' '1' '4' '3' '7' '6' '3.0' '2.0' '1.0' '4.0' '9.0' '6.0' '5.0']
size: ['1842 sq.ft.' '743 sq.ft.' '700 sq.ft.' '750 sq.ft.' '1100 sq.ft.'
 '1009 sq.ft.' '950 sq.ft.' '650 sq.ft.' '819 sq.ft.' '1500 sq.ft.'
 '1719 sq.ft.' '1539 sq.ft.' '1270 sq.ft.' '1200 sq.ft.' '1028 sq.ft.'
 '850 sq.ft.' '1150 sq.ft.' '1076 sq.ft.' '1050 sq.ft.' '1267 sq.ft.'
 '1121 sq.ft.' '872 sq.ft.' '904 sq.ft.' '1132 sq.ft.' '1000 sq.ft.'
 '1142 sq.ft.' '1300 sq.ft.' '800 sq.ft.' '1899 sq.ft.' '521 sq.ft.'
 '468 sq.ft.' '1193 sq.ft.' '1070 sq.ft.' '866 sq.ft.' '925 sq.ft.'
 '570 sq.ft.' '1072 sq.ft.' '990 sq.ft.' '1020 sq.ft.' '450 sq.ft.'
 '931 sq.ft.' '820 sq.ft.' '1018 sq.ft.' '726 sq.ft.' '865 sq.ft.'
 '932 sq.ft.' '1095 sq.ft.' '1290 sq.ft.' '975 sq.ft.' '1118 sq.ft.'
 '805 sq.ft.' '1033 sq.ft.' '1035 sq.ft.' '1010 sq.ft.' '880 sq.ft.'
 '3488 sq.ft.' '680 sq.ft.' '928 sq.ft.' '1717 sq.ft.' '1548 sq.ft.'
 '1125 sq.ft.' '1065 sq.ft.' '1001 sq.ft.' '678 sq.ft.' '1216 sq.ft.'
 '1060 sq.ft.' '1325 sq.ft.' '589 sq.ft.' '890 sq.ft.' '870 sq.ft.'
 '520 sq.ft.' '1400 sq.ft.' '1141 sq.ft.' '1096 sq.ft.' '953 sq.ft.'
 '505 sq.ft.' '1433 sq.ft.' '1170 sq.ft.' '923 sq.ft.' '648 sq.ft.'
 '930 sq.ft.' '558 sq.ft.' '780 sq.ft.' '1272 sq.ft.' '609 sq.ft.'
 '1130 sq.ft.' '679 sq.ft.' '1259 sq.ft.' '935 sq.ft.' '916 sq.ft.'
 '670 sq.ft.' '877 sq.ft.' '1075 sq.ft.' '736 sq.ft.' '807 sq.ft.'
 '1168 sq.ft.' '855 sq.ft.' '1106 sq.ft.' '1047 sq.ft.' '1103 sq.ft.'
```

```
furnished: ['Fully Furnished' 'Partially Furnished' 'Not Furnished']
facilities: ['Minimart, Gymnasium, Security, Playground, Swimming Pool, Parking, Lift, Barbeque area, Mu
 'Parking, Playground, Swimming Pool, Squash Court, Security, Minimart, Gymnasium, Lift'
 'Parking, Gymnasium, Playground, Security, Lift, Swimming Pool, Multipurpose hall'
 ...
 'Gymnasium, Swimming Pool, Security, Lift, Barbeque area, Multipurpose hall, Playground, Jogging Track'
 'Parking, Barbeque area, Playground, Swimming Pool, Security, Gymnasium, Lift, Minimart, Multipurpose h
 'Jogging Track, Barbeque area, Playground, Swimming Pool, Tennis Court, Sauna']
additional_facilities: ['Air-Cond, Cooking Allowed, Washing Machine'
 'Cooking Allowed, Near KTM/LRT, Washing Machine'
 'Air-Cond, Cooking Allowed, Near KTM/LRT, Washing Machine'
 'Air-Cond, Cooking Allowed, Washing Machine, Near KTM/LRT'
 'Air-Cond, Cooking Allowed, Near KTM/LRT, Washing Machine, Internet'
 'Air-Cond, Near KTM/LRT, Cooking Allowed'
 'Air-Cond, Cooking Allowed, Near KTM/LRT' 'Air-Cond, Cooking Allowed'
 'Air-Cond, Cooking Allowed, Internet'
 'Air-Cond, Cooking Allowed, Near KTM/LRT, Internet'
 'Air-Cond, Cooking Allowed, Washing Machine, Internet' 'Near KTM/LRT'
 'Air-Cond, Near KTM/LRT, Washing Machine, Cooking Allowed'
 'Cooking Allowed' 'Air-Cond' 'Cooking Allowed, Washing Machine'
 'Cooking Allowed, Air-Cond'
 'Cooking Allowed, Air-Cond, Near KTM/LRT, Washing Machine, Internet'
 'Cooking Allowed, Air-Cond, Near KTM/LRT, Washing Machine'
 'Air-Cond, Washing Machine' 'Air-Cond, Near KTM/LRT'
```

--and so on

Explanation: The above code snippet iterates through all columns in the DataFrame with a data type of object (typically used for categorical or string data). For each column, it prints the column name along with its unique values. This is helpful for exploring categorical data, identifying distinct categories, or detecting anomalies in the dataset.

9. Fill missing values for numeric columns only:

```
[14]: # 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())
```

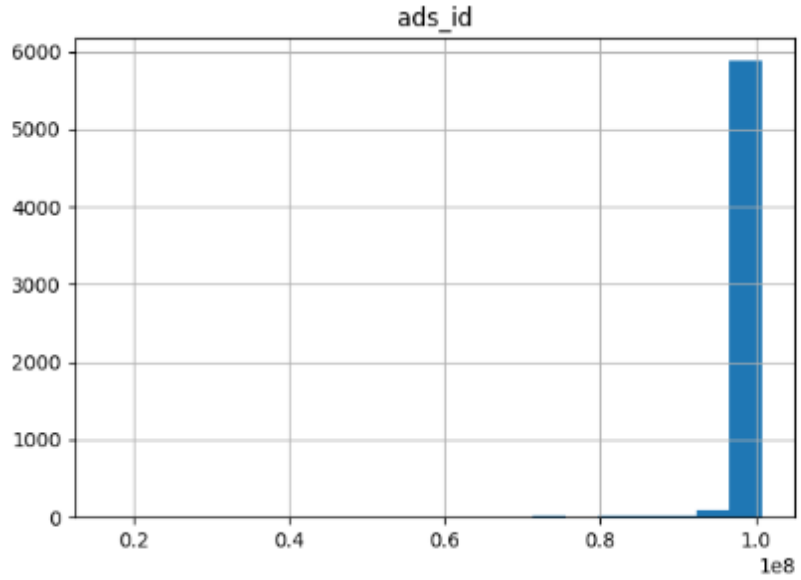
Explanation: This code identifies all numeric columns in the DataFrame and fills any missing (null) values in these columns with their respective mean values. The ‘df.select_dtypes(include=[np.number]).columns’ extracts the names of numeric columns, and ‘df[numeric_columns].fillna(df[numeric_columns].mean())’ replaces null values with column-wise means.

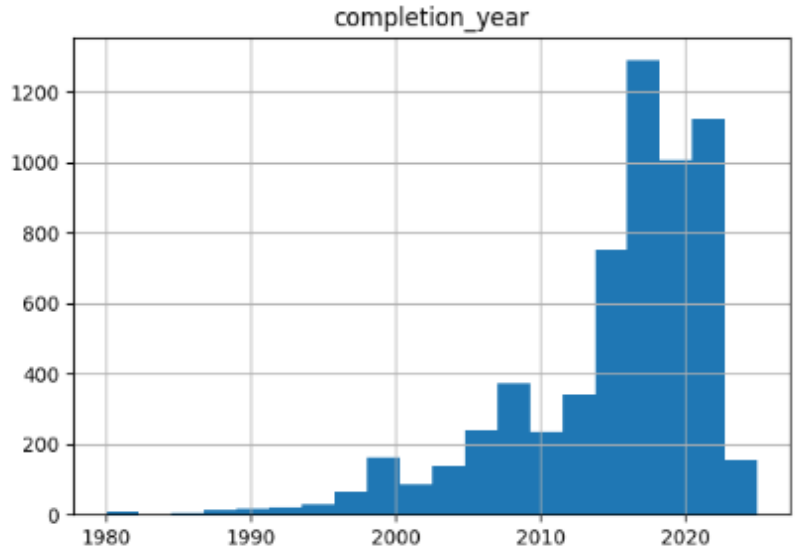
This technique ensures missing data in numerical columns is handled without removing rows.

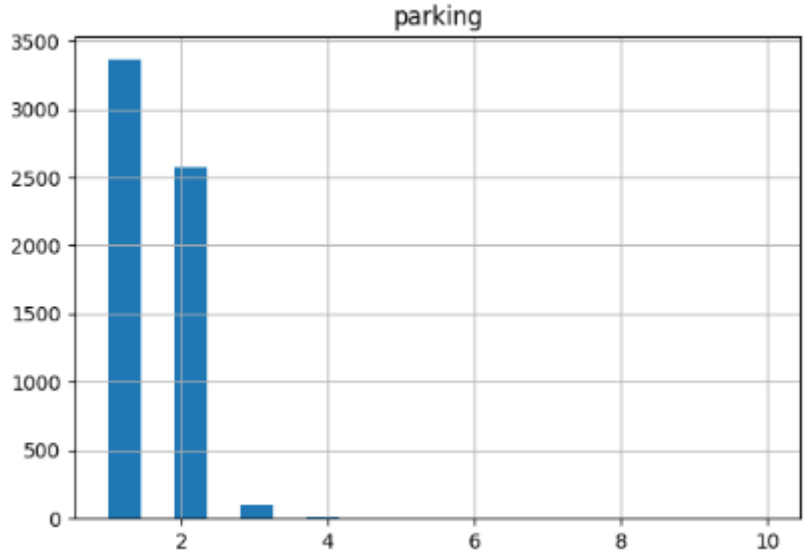
10. Univariate Analysis, plotting histogram for numerical data:

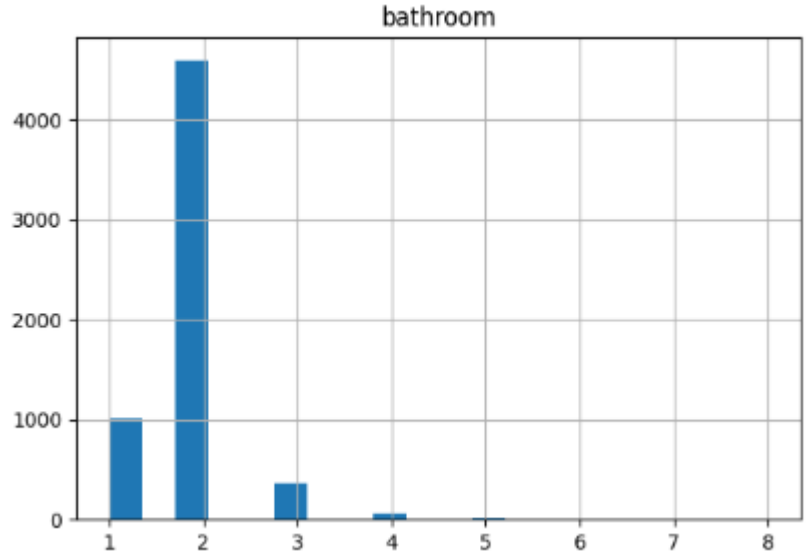
```
[11]: # a) Univariate Analysis
      # Examine individual columns.
      # Plot histogram for numerical data
      df.hist(bins=20, figsize=(15,10))
      plt.show()
```

Explanation: This code generates histograms for all numerical columns in the DataFrame. The figsize parameter adjusts the overall size of the plot, and plt.show() displays the generated plots. This is a quick way to explore the data's distribution and detect patterns or outliers.





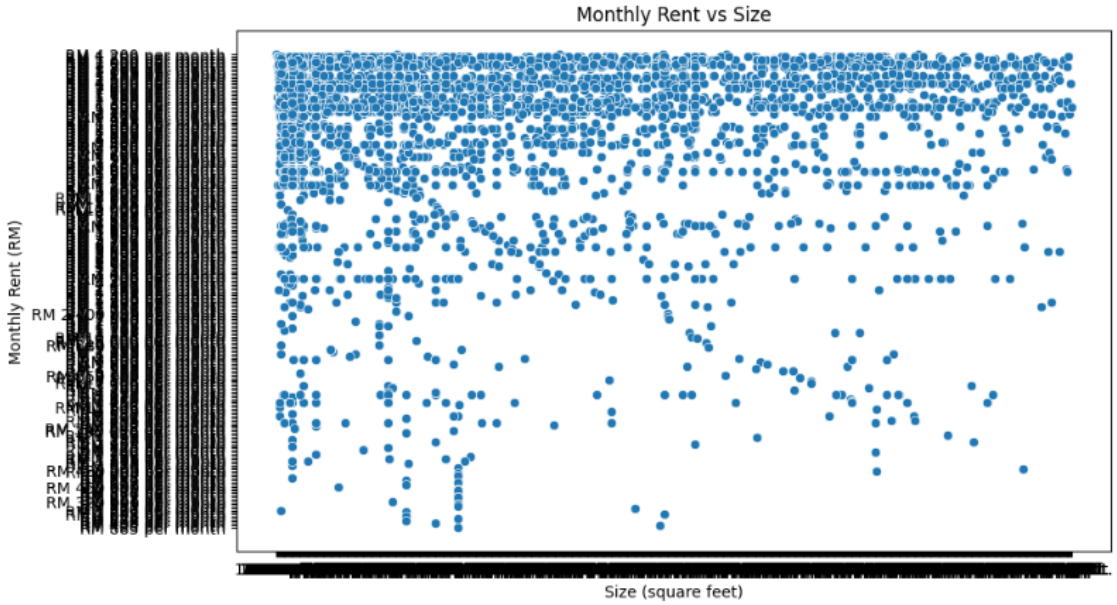




11. Bivariate Analysis, plotting scatter for monthly rent vs. size:

[12]:
```
# 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()
```

Explanation: This code creates a scatter plot to visualize the relationship between the size and monthly_rent columns in the DataFrame. The plt.figure(figsize=(10, 6)) sets the plot's dimensions, while sns.scatterplot() from the seaborn library plots the data points. The plt.title(), plt.xlabel(), and plt.ylabel() functions add a title and labels to the axes for better readability. Finally, plt.show() displays the plot, helping to identify trends or correlations between property size and rent.



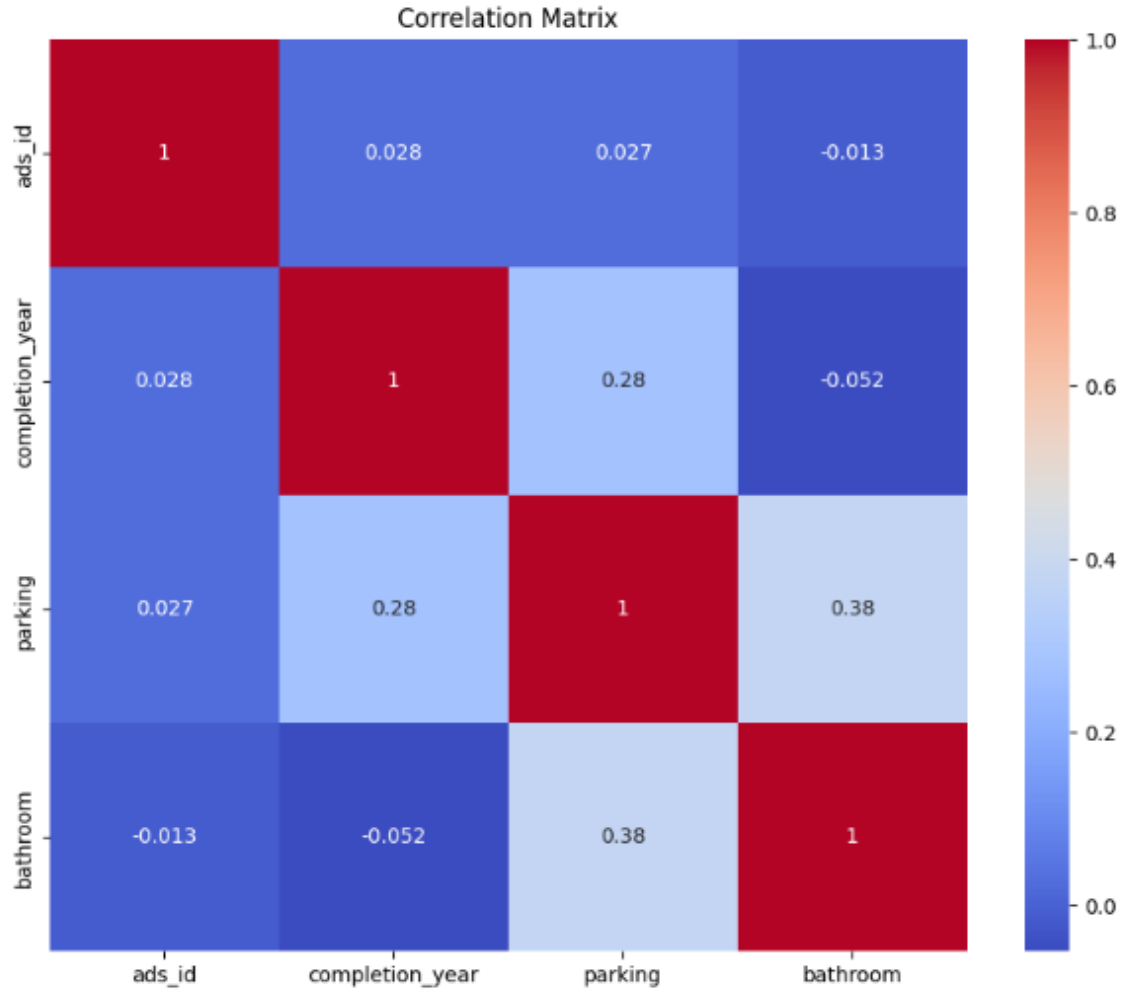
12. Correlation Analysis, checking the correlation between numerical columns:

[13]:
```
# 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()
```

Explanation: This code computes and visualizes the correlation matrix for all numeric columns in the DataFrame. The `numeric_df = df.select_dtypes(include=[np.number])` filters the DataFrame to include only numeric columns. The `numeric_df.corr()` calculates the pairwise correlation coefficients. The `sns.heatmap()` creates a heatmap to visualize these correlations, with annotations (`annot=True`) showing the correlation values and a color gradient

(`cmap='coolwarm'`) for better differentiation. This plot helps identify strong positive or negative relationships between numerical variables.



--Thank You!