RAKAMIN
DATA SCIENCE

Agustus 2025

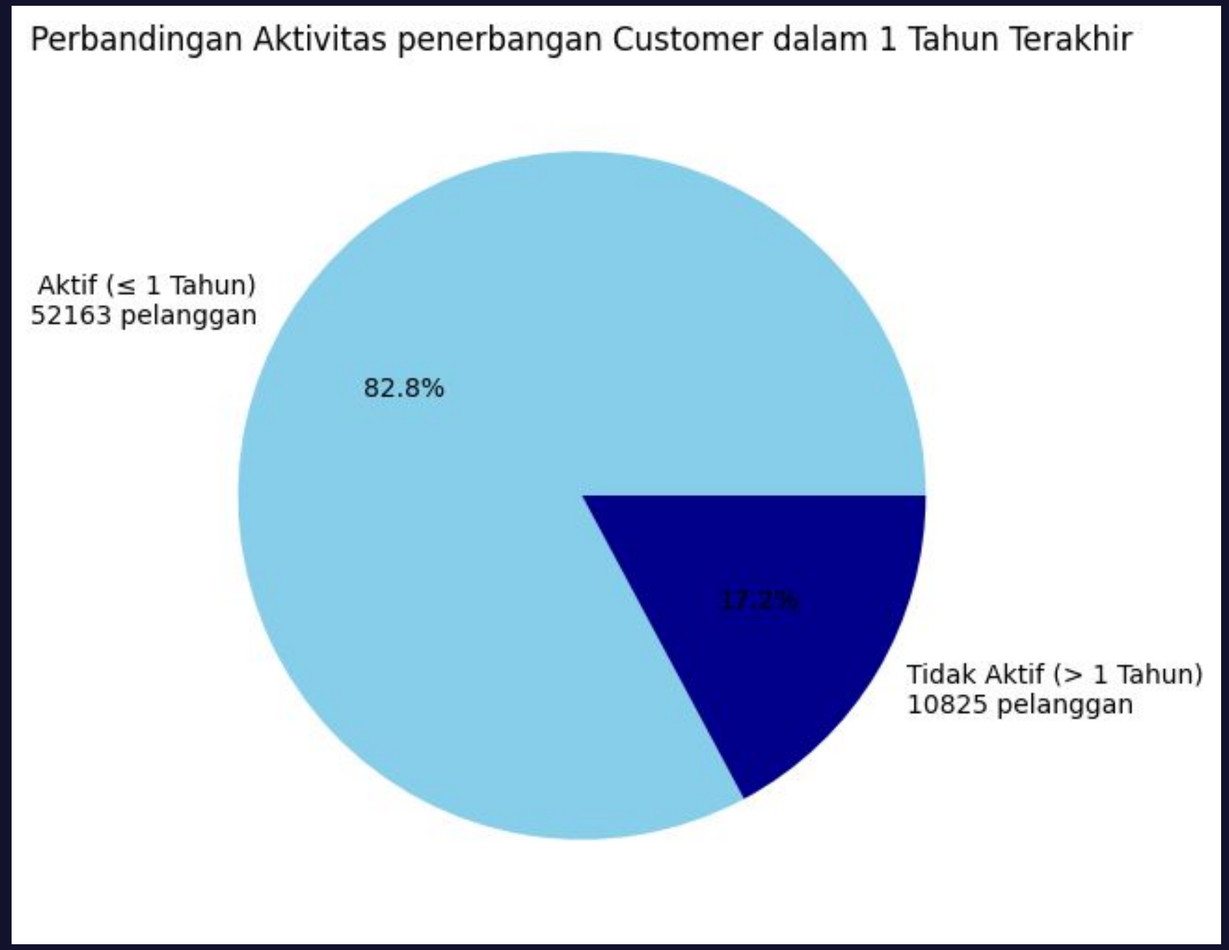
Airline Customer Value Analysis Case

Agi Rahmawandi_Batch57

EDA

exploratory data analysis

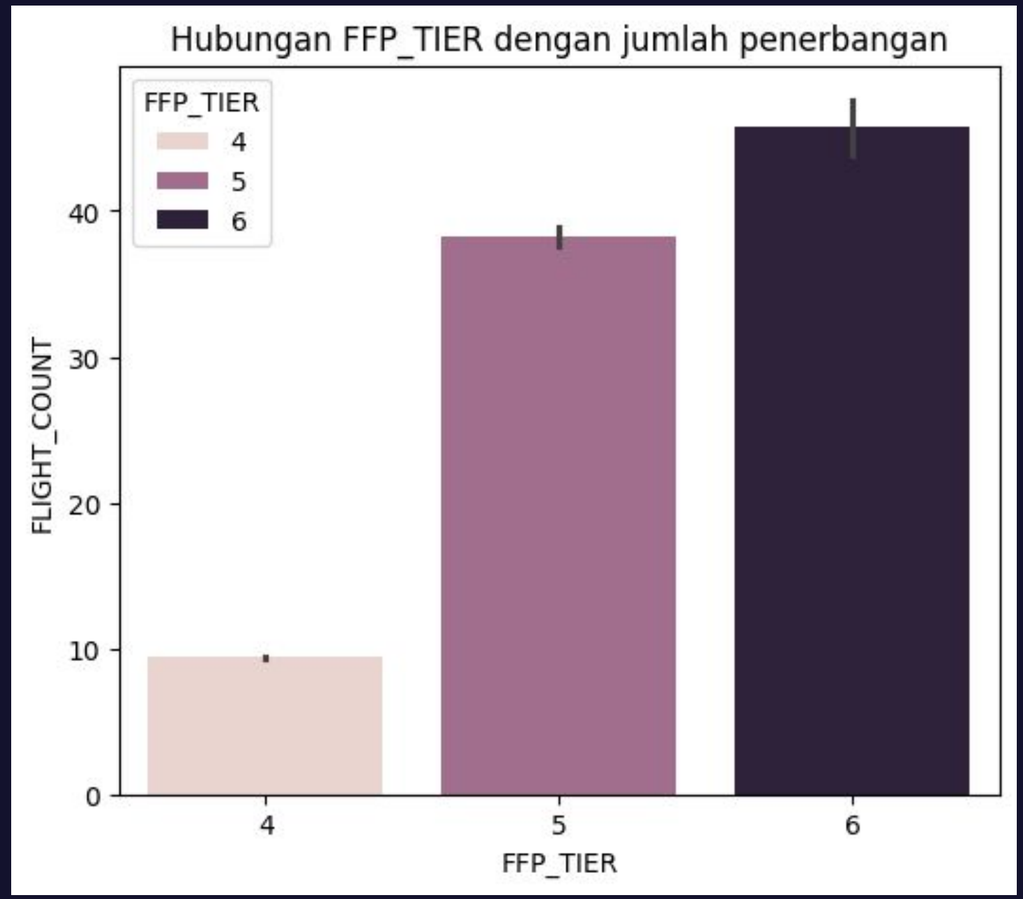
Berapa banyak customer yang melakukan penerbangan dalam setahun terkahir?



EDA

exploratory data analysis

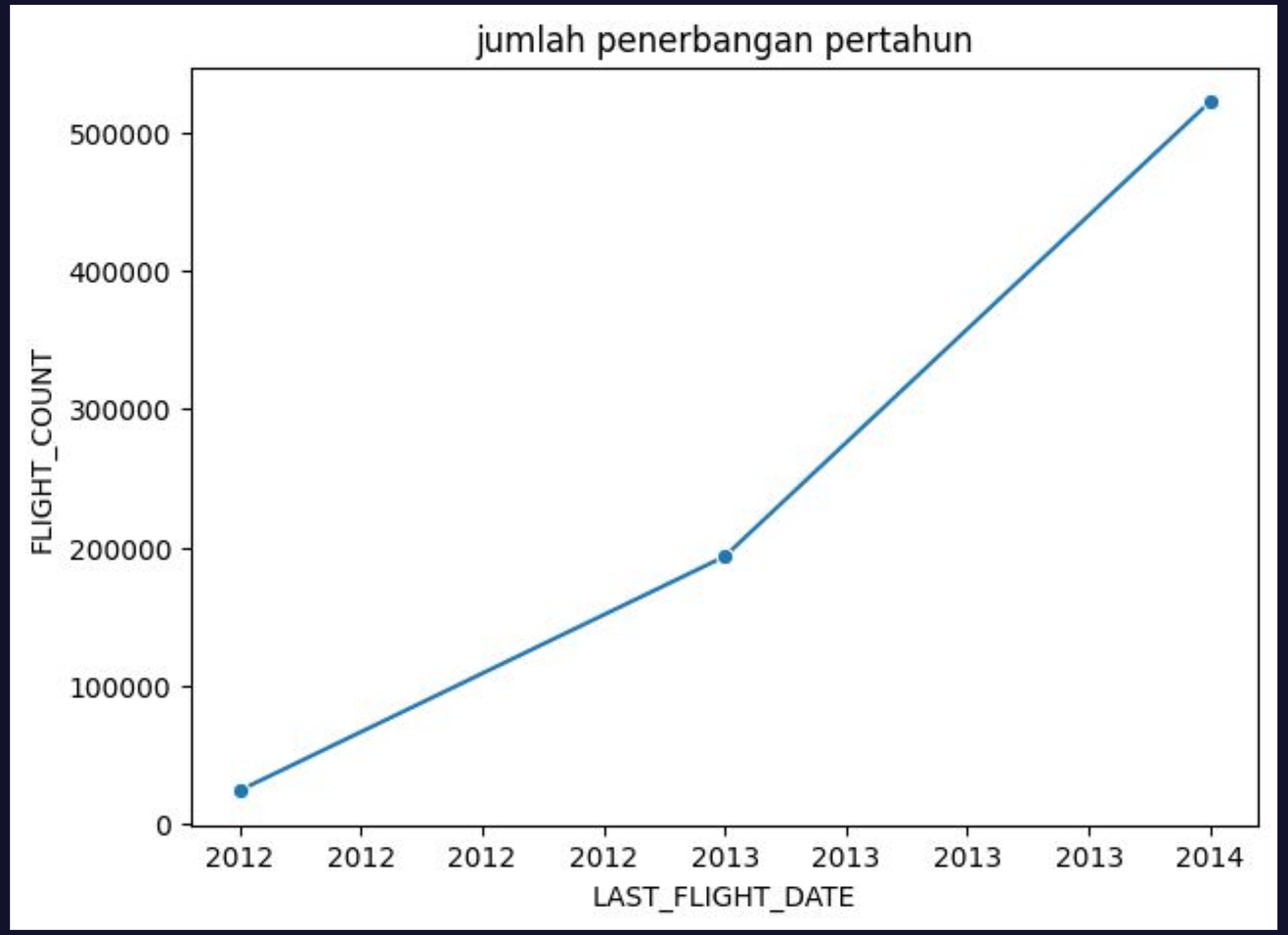
Customer dengan Tier/peringkat apa yang sering melakukan penerbangan ?



EDA

exploratory data analysis

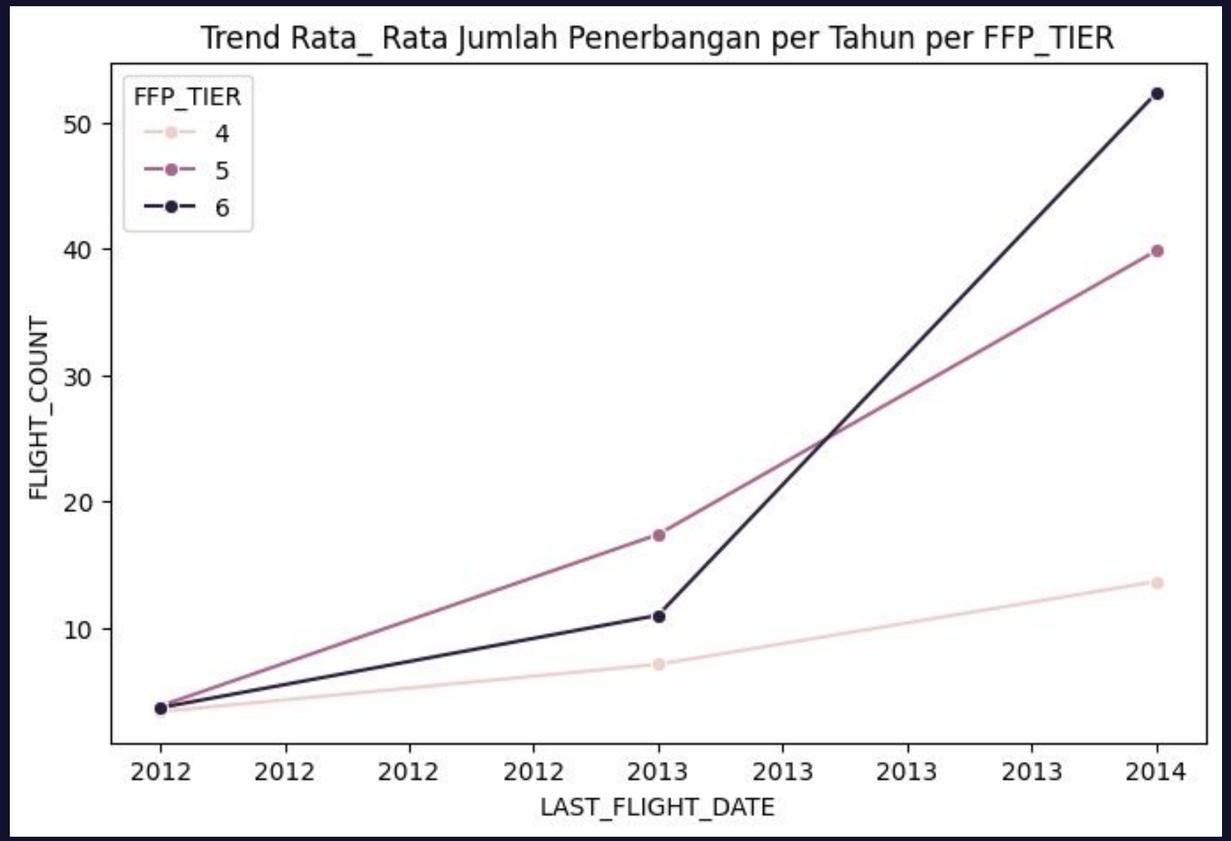
Apakah setiap tahun mengalami penigkatan jumlah penerbangan ?



EDA

exploratory data analysis

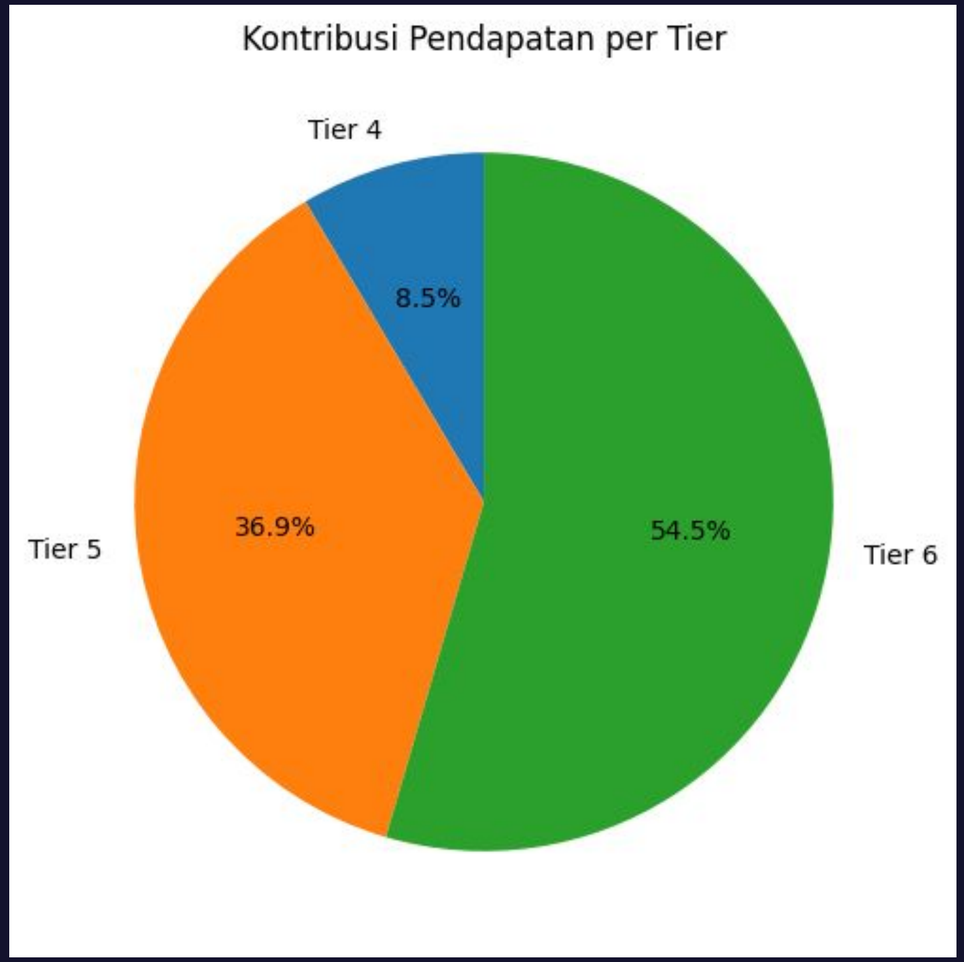
Berapa jumlah penerbangan untuk setiap Tiernya ?



EDA

exploratory data analysis

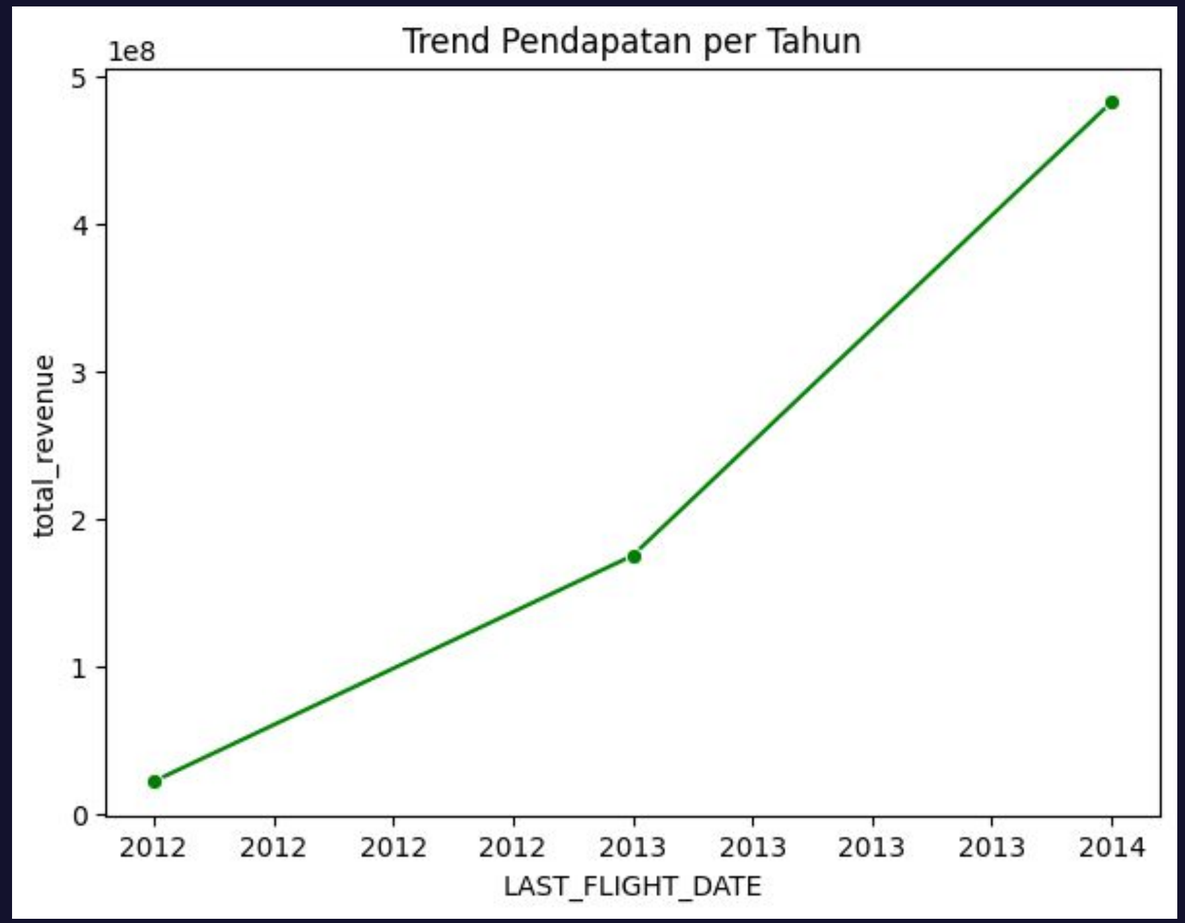
Berapa nilai rata - rata pedapatan pada setiap Tiernya?



EDA

exploratory data analysis

Bagaiamana pendapatan pertahunnya apakah mengalami kenaikan ?



EDA

descriptive statistics

ada beberapa kolom yang memiliki baris yang kosong null, dan ada beberapa kolon dengan nilai standar deviasinya besar yang menandakan ada nilai outlier yang cukup tinggi di kolom tersebut.

```
df.describe().T
```

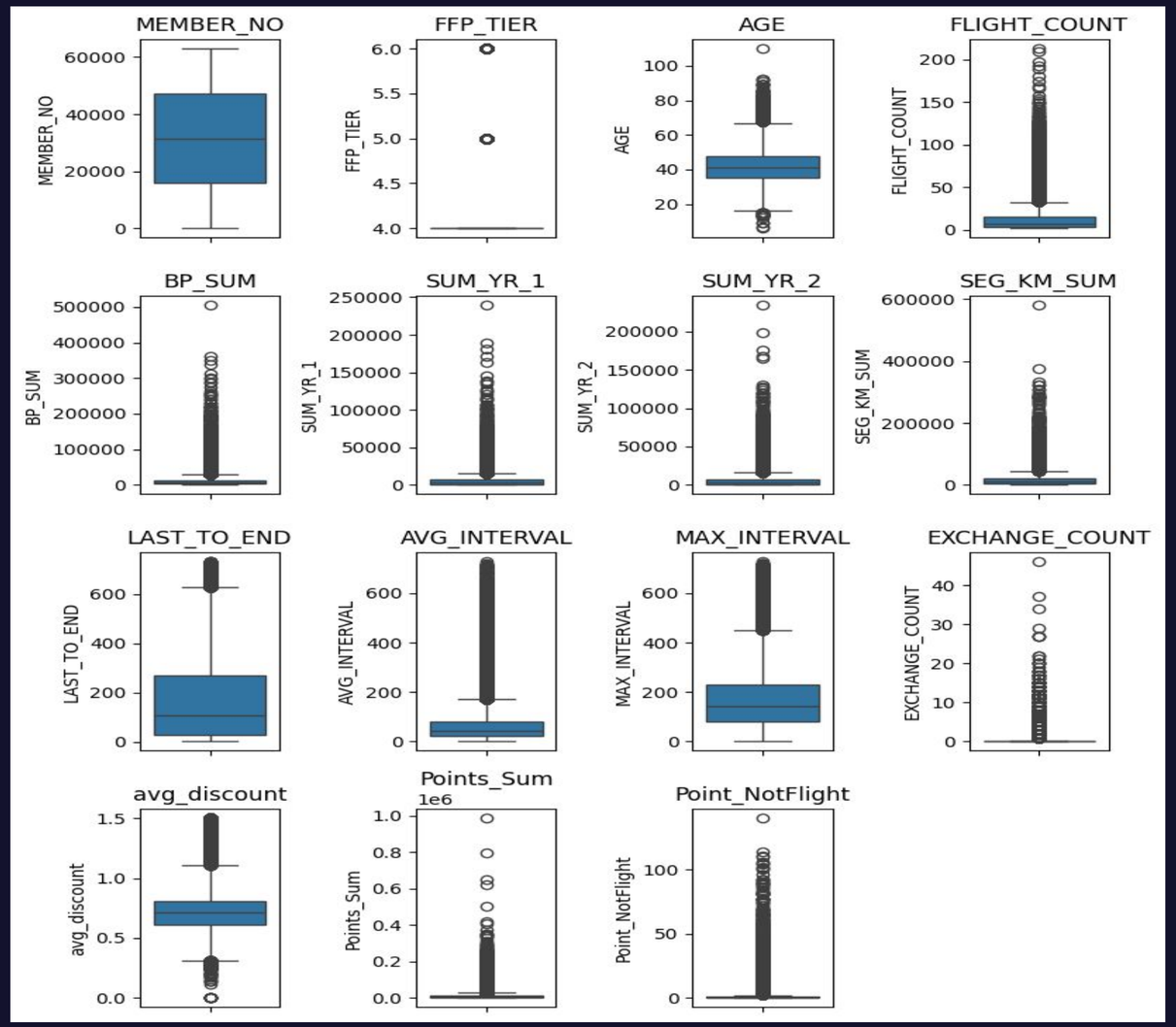
✓ 0.2s

	count	mean	std	min	25%	50%	75%	max
MEMBER_NO	62988.0	31494.500000	18183.213715	1.0	15747.750000	31494.500000	47241.250000	62988.0
FFP_TIER	62988.0	4.102162	0.373856	4.0	4.000000	4.000000	4.000000	6.0
AGE	62568.0	42.476346	9.885915	6.0	35.000000	41.000000	48.000000	110.0
FLIGHT_COUNT	62988.0	11.839414	14.049471	2.0	3.000000	7.000000	15.000000	213.0
BP_SUM	62988.0	10925.081254	16339.486151	0.0	2518.000000	5700.000000	12831.000000	505308.0
SUM_YR_1	62437.0	5355.376064	8109.450147	0.0	1003.000000	2800.000000	6574.000000	239560.0
SUM_YR_2	62850.0	5604.026014	8703.364247	0.0	780.000000	2773.000000	6845.750000	234188.0
SEG_KM_SUM	62988.0	17123.878691	20960.844623	368.0	4747.000000	9994.000000	21271.250000	580717.0
LAST_TO_END	62988.0	176.120102	183.822223	1.0	29.000000	108.000000	268.000000	731.0
AVG_INTERVAL	62988.0	67.749788	77.517866	0.0	23.370370	44.666667	82.000000	728.0
MAX_INTERVAL	62988.0	166.033895	123.397180	0.0	79.000000	143.000000	228.000000	728.0
EXCHANGE_COUNT	62988.0	0.319775	1.136004	0.0	0.000000	0.000000	0.000000	46.0
avg_discount	62988.0	0.721558	0.185427	0.0	0.611997	0.711856	0.809476	1.5
Points_Sum	62988.0	12545.777100	20507.816700	0.0	2775.000000	6328.500000	14302.500000	985572.0
Point_NotFlight	62988.0	2.728155	7.364164	0.0	0.000000	0.000000	1.000000	140.0

EDA

Univaiate analisis

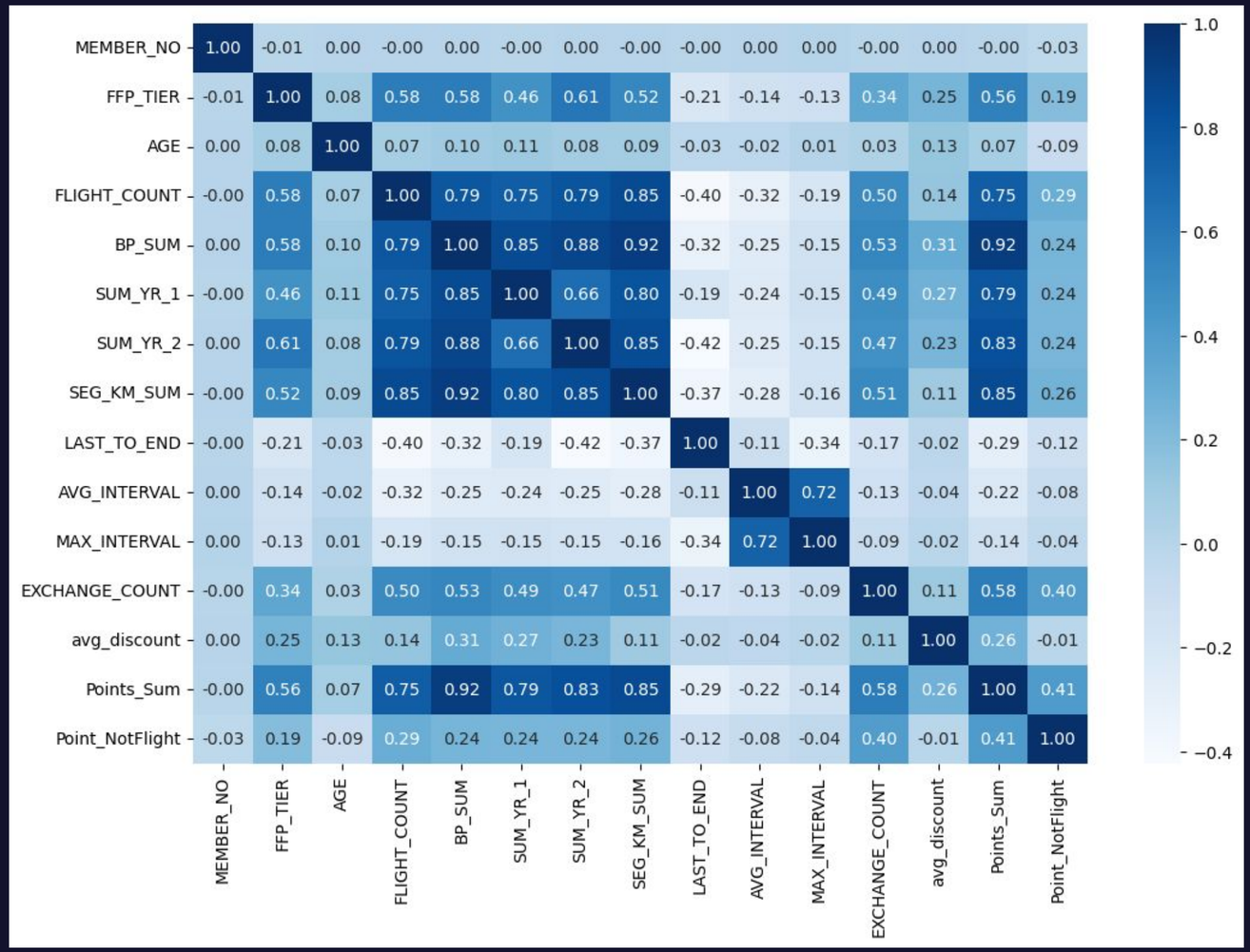
ada 10 kolom yang memiliki nilai outlier yang cukup besar, pada waktu preprocessing akan dilakukan pemberisan outlier dengan z-score



EDA

Multivariate Analysis

ada beberapa kolom yang berkoreasi sangat kuat(redundan), jadi dipilih salah satu kolom yang nilai korelasi nya mewakilkan kolom redundan tersebut



PREPOCESSING

Handling MIssing Value

```
null_info = (
    df.isnull().sum()
    .to_frame(name='Jumlah Null')
    .assign(Persentase_Null=lambda x: (x['Jumlah Null'] / len(df)) * 100)
    .sort_values(by='Persentase_Null', ascending=False)
)

print(null_info)
```

✓ 0.2s

	Jumlah Null	Persentase_Null
WORK_PROVINCE	3248	5.156538
WORK_CITY	2269	3.602273
SUM_YR_1	551	0.874770
AGE	420	0.666794
SUM_YR_2	138	0.219089
WORK_COUNTRY	26	0.041278
GENDER	3	0.004763
MEMBER_NO	0	0.000000
LAST_FLIGHT_DATE	0	0.000000
Points_Sum	0	0.000000
avg_discount	0	0.000000
EXCHANGE_COUNT	0	0.000000
MAX_INTERVAL	0	0.000000
AVG_INTERVAL	0	0.000000
LAST_TO_END	0	0.000000
BP_SUM	0	0.000000
SEG_KM_SUM	0	0.000000
FFP_DATE	0	0.000000
FLIGHT_COUNT	0	0.000000
LOAD_TIME	0	0.000000
FFP_TIER	0	0.000000
FIRST_FLIGHT_DATE	0	0.000000
Point_NotFlight	0	0.000000

```
#deleting rows with missing value
df_cleaned= df_cleaned.drop(columns=['WORK_PROVINCE']).dropna()
```

✓ 0.1s

```
df_cleaned.isnull().sum()
```

✓ 0.1s

```
MEMBER_NO           0
FFP_DATE            0
FIRST_FLIGHT_DATE   0
GENDER              0
FFP_TIER            0
WORK_CITY           0
WORK_COUNTRY        0
AGE                 0
LOAD_TIME           0
FLIGHT_COUNT        0
BP_SUM              0
SUM_YR_1            0
SUM_YR_2            0
SEG_KM_SUM          0
LAST_FLIGHT_DATE    0
LAST_TO_END         0
AVG_INTERVAL        0
MAX_INTERVAL        0
EXCHANGE_COUNT      0
avg_discount        0
Points_Sum          0
Point_NotFlight     0
dtype: int64
```

Akan dilakukan drop kolom WORK_PROVINCE, banyak sekali nilai kosong dikolom tersebut dan melakukan drop baris yang berisi nilai null

PREPOCESSING

Handling data duplicate & data adjusment

Handling duplicate data

```
df_cleaned.duplicated().sum()
```

✓ 0.0s

```
np.int64(0)
```

Tidak ada kolom duplicate

Data Adjusment

Menyesuaikan type data kolom

```
df_cleaned["FIRST_FLIGHT_DATE"] = pd.to_datetime(df_cleaned["FIRST_FLIGHT_DATE"], errors="coerce")
df_cleaned["LAST_FLIGHT_DATE"] = pd.to_datetime(df_cleaned["LAST_FLIGHT_DATE"], errors="coerce")
df_cleaned["LOAD_TIME"] = pd.to_datetime(df_cleaned["LOAD_TIME"], errors="coerce")
df_cleaned["FFP_DATE"] = pd.to_datetime(df_cleaned["FFP_DATE"], errors="coerce")
```

✓ 0.0s

Setelah dicek tidak ada kolom duplicat.

Kemudian dilakukan penyesuaian type data, dari 4 kolom ini **FIRST_FLIGHT_DATE,LAST_FLIGHT_DATE,LOAD_TIME,FFP_DATE,** sebelumnya bertype data object dan dirubah menjadi datetime

Feature Enginering

Feature Extraction

Feature extraction

```
df_cleaned["total_revenue"] = df_cleaned[["SUM_YR_1", "SUM_YR_2"]].sum(axis=1, skipna=True)
df_cleaned["Membership_dur"] = ((df_cleaned["LOAD_TIME"] - df_cleaned["FFP_DATE"]).dt.days)
df_cleaned[["total_revenue","Membership_dur"]].head()
```

✓ 0.0s

	total_revenue	Membership_dur
111	55717.0	1230
114	50990.0	662
147	46650.0	1828
188	44040.0	1329
201	45294.0	3092

Membuat feature baru dari kolom :

- SUM_YR_1, SUM_YR_2 mengahsilkan kolom **total_revenue,** yaitu kolom untuk menghitung total semua permbayaran dari masing masing customer

- LOAD_TIME - FFP_DATE mengasilkan kolom Membership_dur, yaitu untuk menghitung seberapa lama customer telah menjadi member maskapai ini

Feature Enginering

```
feats = ['LAST_TO_END','FLIGHT_COUNT','total_revenue','avg_discount','Membership_dur']
```
✓ 0.0s

```
df_cleaned[feats].head()
```
✓ 0.0s

	LAST_TO_END	FLIGHT_COUNT	total_revenue	avg_discount	Membership_dur
111	44	31	55717.0	0.723434	1230
114	27	16	50990.0	0.753750	662
147	18	19	46650.0	0.681332	1828
188	179	15	44040.0	0.681333	1329
201	30	16	45294.0	0.665449	3092

Feature Selection

Untuk Feature selection disini menggunakan acuan dari teknik analisi RFM, Recency, Frequency dan Monetary.

Feature yang dipilih

- **LAST_TO_END** (Recency), Melihat sebarapa jauh jarak penerbangan terakhir dilakukan hingga data yang terakhir diambil(dalam hari)
- **FLIGHT_COUNT** (Frequency), Menunjukkan frekuensi penerbangan, indikator untuk perilaku perjalanan customer.
- **Total_revenue** (Monetary), untuk melihat jumlah total pembayaran dari setiap customer
- **Avg_Discount,** untuk melihat repon customer terhadap discount
- **Membership_dur**, untuk melihat seberapa lama pelanggan telah menjadi member

Feature Enginering

Handling Outlier

```
### Handling Outlier
from scipy import stats

print(f'jumlah baris sebelum memfilter outlier: {len(df_cleaned)}')

filtered_enteries = np.array([True] * len(df_cleaned))

for col in feats:
    zscore = abs(stats.zscore(df_cleaned[col]))
    filtered_enteries = (zscore < 3) & filtered_enteries

df_cleaned = df_cleaned[filtered_enteries]

print(f'jumlah baris sesudah memfilter outlier: {len(df_cleaned)}')
```

✓ 0.0s

```
jumlah baris sebelum memfilter outlier: 59701
jumlah baris sesudah memfilter outlier: 56739
```

Karena data ini memiliki outlier makan akan dilakukan penangan outlier dengan menghapus baris pada feature yang diangap outlier menggunakan metode Z-Score, karena metode dinilai tidak begitu banyak mereduksi data, sehingga data yang terhapus tidak begitu banyak.
Jumlah baris sebelum diproses **59701** dan sesudah diproses menjadi **56739**

Feature Enginering

Scaling

Scaling

```
from sklearn.preprocessing import StandardScaler

X = df_cleaned[feats].copy()

# Buat objek scaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Untuk hasilnya kembali jadi DataFrame
X_scaled_df = pd.DataFrame(X_scaled, columns=X.columns)
X_scaled_df.describe().T
```

✓ 0.0s

	count	mean	std	min	25%	50%	75%	max
LAST_TO_END	56739.0	9.617667e-17	1.000009	-0.970486	-0.797853	-0.363484	0.496900	3.000089
FLIGHT_COUNT	56739.0	0.000000e+00	1.000009	-0.845516	-0.744816	-0.342013	0.362890	4.390912
total_revenue	56739.0	-8.014723e-17	1.000009	-0.975560	-0.685646	-0.365146	0.318716	5.155719
avg_discount	56739.0	-5.610306e-16	1.000009	-3.416074	-0.623961	0.008955	0.612673	3.649284
Membership_dur	56739.0	6.411778e-17	1.000009	-1.298498	-0.878759	-0.266591	0.822443	2.396161

Agar data memiliki skala yang sama digunakan metode scaling dengan StandardScaler. dan hasilnya akan di simpan pada variable **X_scaled** yang bertipe Numpy array, yang akan digunakan saat pemodelan

MODELING

```python
# Range jumlah cluster yang akan diuji
range_n_clusters = range(2, 11)

# Untuk menyimpan hasil
wcss = []  # inertia_
silhouette_avg = []

for n_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters=n_clusters, random_state=0, n_init='auto')
    cluster_labels = kmeans.fit_predict(X_scaled)

    # Simpan WCSS
    wcss.append(kmeans.inertia_)

    # Hitung silhouette score
    silhouette_avg.append(silhouette_score(X_scaled, cluster_labels))

# Plot Elbow Method
plt.figure(figsize=(12,5))

plt.subplot(1, 2, 1)
plt.plot(range_n_clusters, wcss, marker='o')
plt.title('Elbow Method')
plt.xlabel('Jumlah Cluster (k)')
plt.ylabel('WCSS')
plt.grid(True)

# Plot Silhouette Score
plt.subplot(1, 2, 2)
plt.plot(range_n_clusters, silhouette_avg, marker='o', color='orange')
plt.title('Silhouette Score')
plt.xlabel('Jumlah Cluster (k)')
plt.ylabel('Silhouette Score')
plt.grid(True)

plt.show()
```

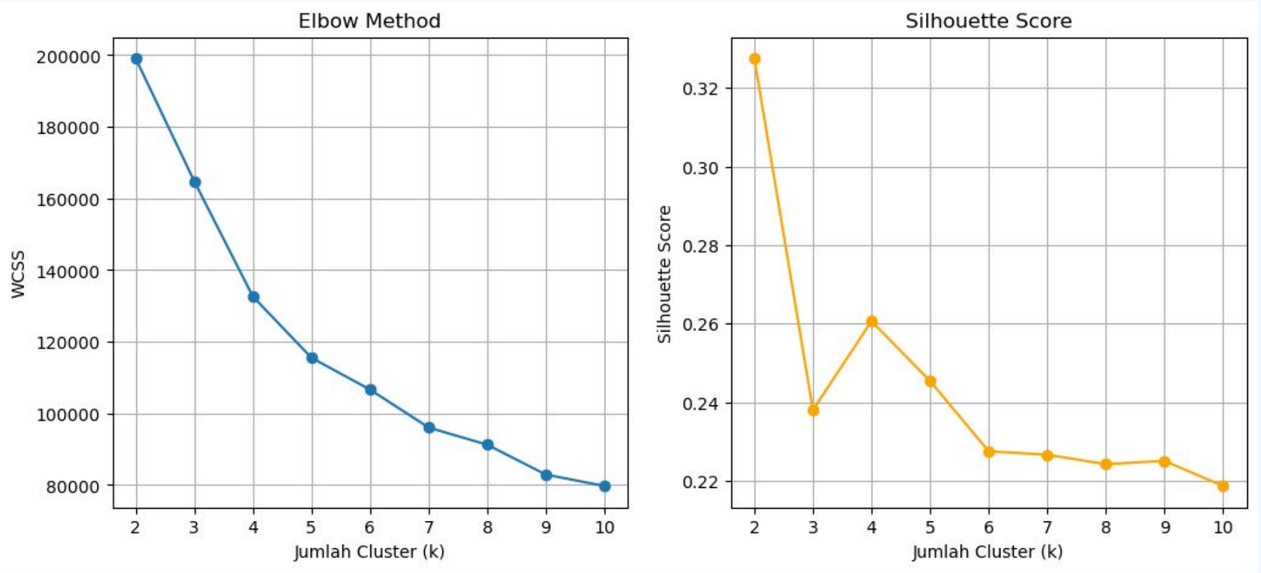
K-Means

Mencari Nilai Cluster Optimal

Untuk mencari nilai cluster yang optimal digunakan Elbow Methode dan Silhoutte Score

MODELING

K-Means



	k	WCSS	Silhouette Score
0	2	199068.995078	0.327493
1	3	164772.998497	0.238033
2	4	132627.307248	0.260636
3	5	115529.669912	0.245410
4	6	106649.977431	0.227516
5	7	96062.906932	0.226655
6	8	91266.456450	0.224237
7	9	82935.938077	0.225101
8	10	79745.543483	0.218687

Mencari Nilai Cluster Optimal

Dari Elbow method :
Dari elbow method terlihat penurunan tajam sampai K=4 dan setelah itu mulai melandai menunjukan cluseter optimal adalah 4 cluster

Dari Silhoutte Score :
dari shilhouette score nilai tertinggi adalah 2, namun terlalu sederhana apabila hanya ada 2 cluster, dan mencoba lihat nilai selanjutnya yang mengalami kenaikan pada K=4 di angka **0.26**

Kesimpulan :
Jumlah Cluster terbaik adalah pada 4 cluster

MODELING

K-Means

K-Means Dengan 4 cluster

```
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
kmeans_mod= KMeans(n_clusters=4, random_state=0,n_init=10)
labels =kmeans_mod.fit_predict(X_scaled)
df_kmeans = pd.DataFrame(X_scaled, columns=feats)
df_kmeans['Cluster'] = labels
df_kmeans.head()
```

✓ 0.2s

	LAST_TO_END	FLIGHT_COUNT	total_revenue	avg_discount	Membership_dur	Cluster
0	-0.731027	2.074800	5.024870	0.121872	-0.258172	3
1	-0.825697	0.564292	4.515797	0.316202	-0.941299	3
2	-0.875816	0.866393	4.048402	-0.148011	0.461036	3
3	0.020765	0.463591	3.767318	-0.148002	-0.139106	3
4	-0.808990	0.564292	3.902368	-0.249826	1.981234	3

```
#Analisis cluster
print("Jumlah anggota tiap cluster:")
df_kmeans['Cluster'].value_counts()
```

✓ 0.0s

```
Jumlah anggota tiap cluster:

Cluster
0    22715
2    14055
1    11637
3     8332
Name: count, dtype: int64
```

Didapatkan anggota tiap cluster :

Cluster 0 : 22715
Cluster 2 : 14055
Cluster 1 : 11637
Cluster 3 : 8332

Cluster 0 mempunya anggota terbanyak dan Cluster 3 menjadi anggota paling sedikit

MODELING

PCA

```python
from sklearn.decomposition import PCA

# --- PCA untuk visualisasi ---
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
df_pca_plot = pd.DataFrame(X_pca, columns=['PC1', 'PC2'])
df_pca_plot['Cluster'] = labels
df_pca_plot.head()
```

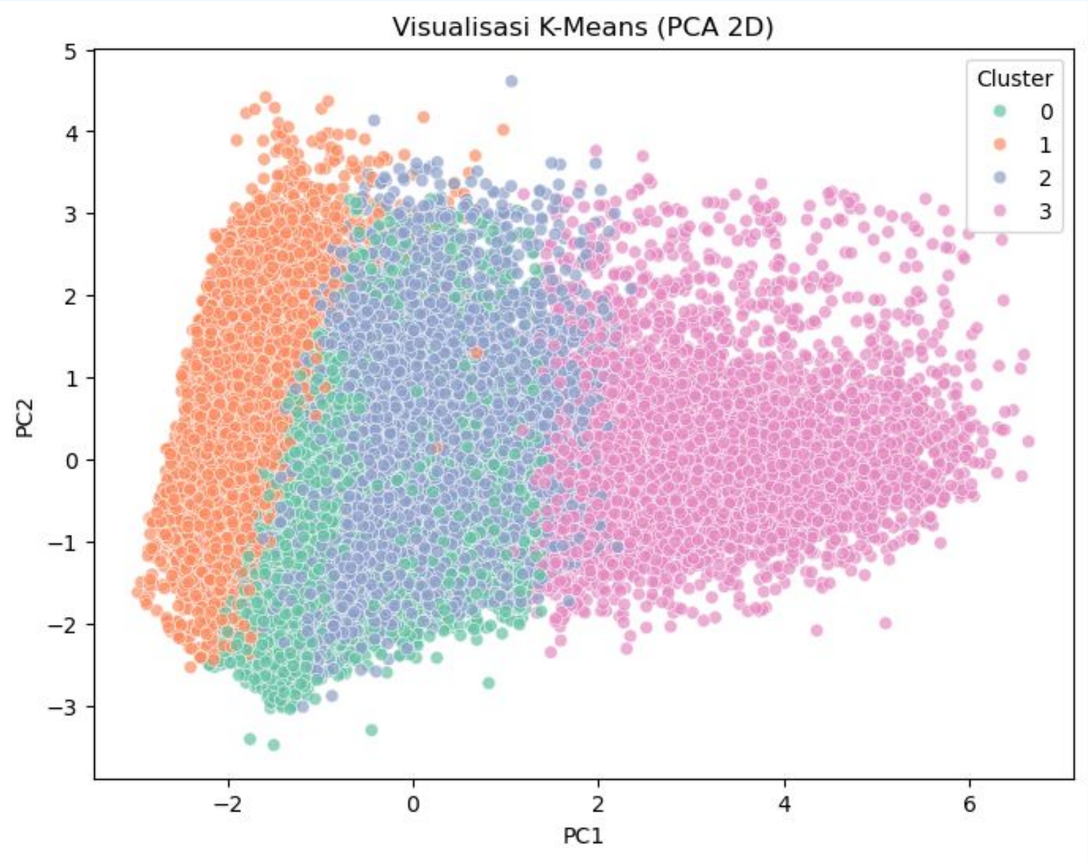
✓ 0.0s

	PC1	PC2	Cluster
0	4.618051	-0.190445	3
1	3.313634	-0.060356	3
2	3.445772	-0.292272	3
3	2.523521	-0.029935	3
4	3.441958	-0.063453	3

Untuk melihat cluster yang telah dibuat datanya maka direduksi dulu dimensinya dengan PCA agar mudah untuk melihat pembagian clusternya

MODELING

Visualisasi Cluster



Hasil dari PCA divisualisasikan 2D dengan scaterplot didapat bentuk sebaran datanya dari tiap cluster

INTERPRETASI

	LAST_TO_END	FLIGHT_COUNT	total_revenue	avg_discount	Membership_dur
	mean	mean	mean	mean	mean
Cluster					
3	44.294647	29.548008	27021.746399	0.757948	1710.449712
2	118.538883	8.462042	7150.919886	0.709568	2427.776805
0	105.597887	7.958750	6533.005767	0.680267	874.270922
1	473.569391	3.778207	3430.885709	0.707032	1180.353442

```
df_cleaned['Cluster'].value_counts()
✓ 0.0s

Cluster
0    22715
2    14055
1    11637
3     8332
Name: count, dtype: int64
```

Interpretasi Cluster

Setelah didapat 4 cluster berikut interpretasinya :

- Cluster 3 :

Jumlahnya sedikit, paling sering melakukan pernerbangan, revenue paling besar, **High-Value Customers**

- Cluster 2 :

Lumayan sering melakukan penerbangan, revenue sedang, membership paling lama, **Loyal Customers**

- Cluster 0 :

Jumlahnya paling banyak, flight count & revenue kecil, **Mass Customers (biasa)**

- Cluster 1 :

Flight count paling rendah,sudah lama tidak melakukan penerbangan, revenue paling kecil, **Inactive Customer**

REKOMENDASI BISNIS

- Fokus utama: pertahankan cluster 3 (**High-Value Customers**) & dorong cluster 2 (**Loyal Customers**) menjadi **High-Value Customers**, inilah engine utama untuk revenue perusahaan.
- Cluster 0 (**Mass Customers**) bisa dinaikkan value dengan promo massal , walau kontribusi kecil, volume mereka besar.
- Cluster 1 (**Inactive Customer**) perlu program khusus reaktivasi. didorong kembali dengan campaign atau promo agar melakukan lagi penerbangan dengan maskapai ini.

TERIMAKASIH