Tugas Praktik Advanced Statistics

1. Hitunglah semua Measures of Central Tendency untuk kolom payAmountuntuk seluruh transaksi berbayar (tidak gratis)!

   Berikut script dan hasilnya :

```
# Filter hanya transaksi berbayar (tidak gratis)
transaksi_berbayar = df[df['payAmount'] > 0]

# Menghitung Measures of Central Tendency
mean_payAmount = transaksi_berbayar['payAmount'].mean()
median_payAmount = transaksi_berbayar['payAmount'].median()
mode_payAmount = transaksi_berbayar['payAmount'].mode().values[0]

# Menampilkan hasil
print("Measures of Central Tendency untuk kolom 'payAmount' pada seluruh transaksi berbayar:")
print("Mean (Rata-rata):", mean_payAmount)
print("Median (Tengah):", median_payAmount)
print("Mode (Modus):", mode_payAmount)
```

```
Measures of Central Tendency untuk kolom 'payAmount' pada seluruh transaksi berbayar:
Mean (Rata-rata): 4919.757964929612
Median (Tengah): 3500.0
Mode (Modus): 3500.0
```

2. Buatlah satu kolom baru umur yang berisi umur penumpang di tahun 2024 ini. Dengan menggunakan groupby & aggregate function atau pivot table, hitunglah mean dan median payAmount dan umur berdasarkan kelompok payCardBank yang digunakan!

   Berikut script dan hasilnya :

```
# Membuat kolom baru
df['umur'] = 2024 - df['payCardBirthDate']

# Groupby payCardBank dan menghitung mean dan median dari payAmount dan umur
result = df.groupby('payCardBank').agg({'payAmount': ['mean', 'median'], 'umur': ['mean', 'median']})

# Cetak hasil
result
```

	payAmount		umur	
	mean	median	mean	median
payCardBank				
bni	2552.278820	3500.0	31.634550	34.0
brizzi	3507.548845	3500.0	34.010479	34.0
dki	2444.656281	3500.0	34.413648	34.0
emoney	3051.308745	3500.0	32.621177	32.0
flazz	2854.880414	3500.0	34.598330	34.0
online	2514.296188	0.0	34.920878	35.0

Next steps: View recommended plots

3. Buatlah beberapa kolom baru:

   - num_stop untuk menghitung jumlah pemberhentian (stopEndSeq - stopStartSeq)
   - duration untuk menghitung lama perjalanan (tapInTime-tapOutTime)

   kemudian gunakan method corr() pada dataframe (dokumentasi dan contoh penggunaan dapat dilihat pada link berikut) untuk menghitung pearson correlation antara num_stop dan payAmount kemudian berikan penjelasan dari hasil nilai correlation tersebut.

   Berikut script dan hasilnya :

```
# Mengubah kolom tapInTime dan tapOutTime menjadi tipe datetime
df['tapInTime'] = pd.to_datetime(df['tapInTime'])
df['tapOutTime'] = pd.to_datetime(df['tapOutTime'])

# Membuat dan menghitung kolom baru
df['num_stop'] = df['stopEndSeq'] - df['stopStartSeq']
df['duration'] = df['tapInTime'] - df['tapOutTime']

# Menghitung korelasi antara num_stop dan payAmount
correlation = df['num_stop'].corr(df['payAmount'], method='pearson')

print("Pearson Correlation between num_stop and payAmount:", correlation)
```

```
Pearson Correlation between num_stop and payAmount: -0.18104482091150012
```

Jika nilai korelasi mendekati 1, itu menunjukkan korelasi positif yang kuat, yang berarti semakin tinggi nilai 'num_stop', semakin tinggi juga nilai 'payAmount', dan jika nilai korelasi mendekati -1, itu menunjukkan korelasi negatif yang kuat, yang berarti semakin tinggi nilai 'num_stop', semakin rendah nilai 'payAmount'. Namun jika nilai korelasi mendekati 0, itu menunjukkan bahwa tidak ada korelasi linier antara kedua variabel tersebut. Dan dengan hasih di atas yang bernilai -0.18104482091150012 maka dapat disimpulkan bahwasanya tidak terdapat korelasi linear yang signifikan antaran num_stop dengan payAmount.

4. Buatlah 2 dataframe baru yang berisi 600 sample data dengan kriteria berikut:

   A. Sampling menggunakan random sample method sample()

   B. Sampling menggunakan stratified sampling untuk kelompok payCardBank, hint : dapat menggunakan kombinasi groupby dan lambda

   Kemudian hitung distribusi banyaknya jumlah transaksi (dalam percentage) di setiap kelompok payCardBank untuk dataframe original (sebelum di sampling), hasil random sampling, dan hasil stratified sampling. Dan tuliskan penjelasan apa yang dapat kamu simpulkan dari hasil tersebut.

   Berikut script dan hasilnya :

```
# Random sampling
df_random_sample = df.sample(n=600)

# Hitung distribusi banyaknya jumlah transaksi (dalam percentage) di setiap kelompok payCardBank untuk dataframe original
original_distribution = df['payCardBank'].value_counts(normalize=True) * 100

# Hitung distribusi banyaknya jumlah transaksi (dalam persentase) di setiap kelompok payCardBank di hasil random sampling
random_sample_distribution = df_random_sample['payCardBank'].value_counts(normalize=True) * 100

# Stratified sampling
stratified_sample = df.groupby('payCardBank', group_keys=False).apply(lambda x: x.sample(frac=600/len(df[df['payCardBank'] == x.name])))

# Hitung distribusi banyaknya jumlah transaksi (dalam persentase) di setiap kelompok payCardBank di hasil stratified sampling
stratified_sample_distribution = stratified_sample['payCardBank'].value_counts(normalize=True) * 100

# Menampilkan hasil
print("\nOriginal Distribution")
print(original_distribution)
print("\nRandom Sampling Distribution:")
print(random_sample_distribution)
print("\nStratified Sampling Distribution:")
print(stratified_sample_distribution)
```

```
Original Distribution
dki       49.453826
emoney    18.116095
brizzi     9.316623
flazz      8.532982
online     7.569921
bni        7.010554
Name: payCardBank, dtype: float64

Random Sampling Distribution:
dki       47.833333
emoney    18.166667
brizzi     9.500000
flazz      9.166667
online     8.666667
bni        6.666667
Name: payCardBank, dtype: float64

Stratified Sampling Distribution:
bni       16.666667
brizzi    16.666667
dki       16.666667
emoney    16.666667
flazz     16.666667
online    16.666667
Name: payCardBank, dtype: float64
```

Dari hasil diatas, dapat kita simpulkan bahwasanya stratified sampling lebih baik daripada random sampling dalam mempertahankan representasi proporsi setiap kelompok payCardBank dalam sampel. Ini penting karena memastikan bahwa sampel mencerminkan populasi dengan baik dapat menghasilkan hasil yang lebih akurat dan dapat dipercaya dalam analisis data.