UNIT 12
PANDAS DAN SEABORN

1 .1 Tujuan:

1.1.1 Mahasiswa dapat melakukan data visualisasi pada Python.
1.1.2 Mahasiswa dapat melakukan manajemen serta memasukkana data dengan menggunakan Pandas.
1.1.3 Mahasiswa dapat melakukan proses import library pada Python
1.1.4 Mahasiswa mampu melakukan analisis data dengan menggunakan metode statistik dengan Seaborn

1 .2 Dasar Teori

Pandas adalah *open-source library* yang dibuat terutama untuk bekerja dengan data relasional atau berlabel baik secara mudah dan intuitif. Pandas menyediakan struktur data yang cepat, fleksibel, dan ekspresif, intuitif dan mudah digunakan. Library Pandas menyediakan fitur dan fungi yang dapat mempermudah pengguna dalam melakukan analisis data dan visualisasi menggunakan Python. Pandas dibangun dari *library* NumPy yang memiliki kinerja cepat dan produktivitas tinggi bagi pengguna..Pandas sangat cocok untuk digunakan dalam berbagai jenis data seperti:

1.2.1 Data tabular heterogen, seperti dalam tabel SQL atau spreadsheet Excel
1.2.2 Data time-series
1.2.3 Data matriks arbitrer dengan label baris dan kolom
1.2.4 Bentuk lain dari kumpulan data observasional/statistik. Data tidak perlu diberi label sama sekali untuk ditempatkan ke dalam struktur data pandas

Seaborn adalah *open-source library* untuk membuat grafik statistik dengan Python. Seaborn dibangun di atas *library* matplotlib dan terintegrasi erat dengan struktur data pandas. Seaborn membantu dalam menganalisis dan memahami data Anda. Fungsi plotnya beroperasi pada *dataframe* dan *array* yang berisi seluruh kumpulan *datasets* dan secara internal melakukan pemetaan semantik dan agregasi statistik yang diperlukan untuk menghasilkan plot yang informatif.

1 .3 Membuat *Dataframe*

Kumpula data yang di representasikan dengan tabel pada Pandas dinamakan *Dataframe, Dataframe* terdiri atas kolom dan baris, kolom pada tabel dinamakan *Series* pada Pandas dan baris dinamakan *row*. Pada proses pembuatan *Dataframe,* data secara manual dimasukkan ke dalam

tabel menggunakan tipe data *Dictionary* dalam membuat kolom dan *List* dalam membuat bagian baris dari kolom.

```
import pandas as pd

# Tabe data pada pandas dinamakan Dataframe (df)
df = pd.DataFrame(
    {
        "Nama" : [
                  "Naufal Firdaus",
                  "Satria Bahureksa",
                  "Sri Sakinah"
        ],
        "NIM" : [
                  2010314069,
                  2010314420,
                  2010314000
        ],
        "Kelamin" : [
                     "Pria",
                     "Pria",
                     "Wanita"
        ]
    }
)
# Melihat isi dari list
df
```

Program diatas merupakan program untuk membuat suatu dataframe dari identitas asistan laboratorium algoritma pemrograman UPN Veteran Jakarta, dengan bagian “Nama”, “NIM”, “Kelamin” merepresentasikan kolom dan bagian di dalam *List* merepresentasikan data dari kolom.

	Nama	NIM	Kelamin
0	Naufal Firdaus	2010314069	Pria
1	Satria Bahureksa	2010314420	Pria
2	Sri Sakinah	2010314000	Wanita

1 .4 Membuat *Series* atau kolom

Dataframe jenis *Series* merupakan *Dataframe* yang terdiri atas satu kolom, fungsi ini digunakan apabila ingin menciptakan suatu tabel data yang memuat satu informasi atau satu kolom.

```
umur_mahasiswa = pd.Series([19,18,17,20,21,18,19,20,22,21,19], name = "umur")
umur_mahasiswa
```

```
0    19
1    18
2    17
3    20
4    21
5    18
6    19
7    20
8    22
9    21
10   19
Name: umur, dtype: int64
```

1 .5 Mengetahui data statistik dari *Dataframe*

Mengetahui data statistik dari suatu data merupakan hal yang penting dalam penelitian, dengan menggunakan *library* Pandas, hal ini bisa dilakukan dengan cepat tanpa harus melakukan kalkulasi secara manual ataupun membuat program secara manual. Data statistik dari suatu kolom dapat diketahui dengan menggunakan dibawah:

```
min()      : Mengetahui nilai terkecil dari data
max()      : Mengetahui nilai terbesar dari data
std()      : Mengetahui standar deviasi dari data
25%        : Mengetahui Q1 dari data
50%        : Mengetahui Q2 dari data
75%        : Mengetahui Q3 dari data
```

```python
umur_mahasiswa = pd.Series([19,18,17,20,21,18,19,20,22,21,19], name = "umur")

# Untuk melihat Data statistik penting pada kolom
print(f"Umur Terkecil Mahasiswa : {umur_mahasiswa.min()}")
print(f"Umur Terbesar Mahasiswa : {umur_mahasiswa.max()}")
print(f"Standar Deviasi Umur Mahasiswa : {umur_mahasiswa.std()}")
print(f"Rata-rata umur Mahasiswa : {umur_mahasiswa.mean()}")

# Melihat keseluruhan data statistik
print("\nData Statistik Kolom Umur Mahasiswa:")
umur_mahasiswa.describe()
```

```
Umur Terkecil Mahasiswa : 17
Umur Terbesar Mahasiswa : 22
Standar Deviasi Umur Mahasiswa : 1.5075567228888183
Rata-rata umur Mahasiswa : 19.454545454545453

Data Statistik Kolom Umur Mahasiswa:
count    11.000000
mean     19.454545
std       1.507557
min      17.000000
25%      18.500000
50%      19.000000
75%      20.500000
max      22.000000
Name: umur, dtype: float64
```

1 .6 Memasukkan Data dengan Panda

Pandas dapat membaca data dari banyak sumber salah satunya yakni csv, excel, json, SQL dan HTML dan mampu mengkonversi ke jenis data lainnya. Pandas akan membaca data ke dalam bentuk *Dataframe* dan dapat mengubah *Dataframe* ke dalam bentuk data yang berbeda.

Masukkan data csv yang disediakan asisten laboratorium ke dalam satu folder yang sama dengan program. Program akan mengidentifikasi *path* dari data csv dan Pandas akan membaca data yang ada di dalam file csv tersebut,

```
import pandas as pd

df_indo_film = pd.read_csv('indonesian_movies.csv')
df_indo_film
```

	title	year	description	genre	rating	users_rating	votes	languages	directors	actors	runtime
0	#FriendButMarried 2	2020	Ayudia (Mawar De Jongh) is not satisfied enoug...	Biography	13+	6.5	120	Indonesian	Rako Prijanto	['Adipati Dolken', 'Mawar Eva de Jongh', 'Vonn...	100 min
1	4 Mantan	2020	Sara, Airin, Rachel, and Amara were accidental...	Thriller	17+	6.4	8	Indonesian	Hanny Saputra	['Ranty Maria', 'Jeff Smith', 'Melanie Berentz...	80 min
2	Aku Tahu Kapan Kamu Mati	2020	After apparent death, Siena is able to see sig...	Horror	13+	5.4	17	Indonesian	Hadrah Daeng Ratu	['Natasha Wilona', 'Ria Ricis', 'Al Ghazali', ...	92 min
3	Anak Garuda	2020	Good Morning Indonesia, a school for poor orph...	Adventure	13+	9.1	27	Indonesian	Faozan Rizal	['Tissa Biani Azzahra', 'Viola Georgie', 'Aji...	129 min
4	Dignitate	2020	Alfi (Al Ghazali) meets Alana (Caitlin Halderm...	Drama	17+	7.6	33	Indonesian	Fajar Nugros	['Al Ghazali', 'Caitlin Halderman', 'Giorgino ...	109 min
...	...	...	...	...	...	...	...	...	...	...	...
1267	The Tiger from Tjampa	1953	Set in the 1930s, and narrated like a ballad f...	Drama	NaN	6.4	30	Indonesian	D. Djajakusuma	['Wahid Chan', 'Bambang Hermanto', 'R.D. Ismai...	97 min
1268	Enam Djam di Djogja	1951	Depicting the celebrated recapture of the town...	Drama	NaN	6.3	9	Indonesian	Usmar Ismail	['R.D. Ismail', 'Del Juzar', 'Aedy Moward', 'A...	116 min
1269	Darah dan Doa	1950	It tells the story of an Indonesian revolution...	Drama	NaN	6.6	27	Indonesian	Usmar Ismail	['Ella Bergen', 'Faridah', 'R.D. Ismail', 'Del...	150 min

untuk dapat melihat isi dari *Dataframe* dapat dilakukan dengan memanggil nama variabel atau dengan menggunakan fungsi :

`head()` : Memanggil lima baris data dari awal

`tail()` : Memanggil lima baris data dari akhir

```
df_indo_film.head()
```

	title	year	description	genre	rating	users_rating	votes	languages	directors	actors	runtime
0	#FriendButMarried 2	2020	Ayudia (Mawar De Jongh) is not satisfied enoug...	Biography	13+	6.5	120	Indonesian	Rako Prijanto	['Adipati Dolken', 'Mawar Eva de Jongh', 'Vonn...	100 min
1	4 Mantan	2020	Sara, Airin, Rachel, and Amara were accidental...	Thriller	17+	6.4	8	Indonesian	Hanny Saputra	['Ranty Maria', 'Jeff Smith', 'Melanie Berentz...	80 min
2	Aku Tahu Kapan Kamu Mati	2020	After apparent death, Siena is able to see sig...	Horror	13+	5.4	17	Indonesian	Hadrah Daeng Ratu	['Natasha Wilona', 'Ria Ricis', 'Al Ghazali', ...	92 min
3	Anak Garuda	2020	Good Morning Indonesia, a school for poor orph...	Adventure	13+	9.1	27	Indonesian	Faozan Rizal	['Tissa Biani Azzahra', 'Viola Georgie', 'Aji...	129 min
4	Dignitate	2020	Alfi (Al Ghazali) meets Alana (Caitlin Halderm...	Drama	17+	7.6	33	Indonesian	Fajar Nugros	['Al Ghazali', 'Caitlin Halderman', 'Giorgino ...	109 min

1 .7 Mengakses Kolom tertentu pada Data

Memilih data pada kolom tertentu yang ingin di observasi secara langu=sung dapat dicari dengan menggunakan metode seperti dibawah. Metode yang digunakan dibawah memiliki tampilan serupa dengan mengakses data dari tipe data *Dictionary* menggunakan *key values* nya

```
df_indo_film["title"]
```

```
0            #FriendButMarried 2
1                        4 Mantan
2       Aku Tahu Kapan Kamu Mati
3                    Anak Garuda
4                      Dignitate
                  ...
1267       The Tiger from Tjampa
1268         Enam Djam di Djogja
1269               Darah dan Doa
1270           Resia Boroboedoer
1271           Loetoeng Kasaroeng
Name: title, Length: 1272, dtype: object
```

Apabila pengguna ingin mengobservasi lebih dari satu kolom maka dapat menggunakan fungsi berikut:

```
df_indo_film[["title","users_rating", "genre"]]
```

	title	users_rating	genre
0	#FriendButMarried 2	6.5	Biography
1	4 Mantan	6.4	Thriller
2	Aku Tahu Kapan Kamu Mati	5.4	Horror
3	Anak Garuda	9.1	Adventure
4	Dignitate	7.6	Drama
...	...	...	...
1267	The Tiger from Tjampa	6.4	Drama
1268	Enam Djam di Djogja	6.3	Drama
1269	Darah dan Doa	6.6	Drama
1270	Resia Boroboedoer	7.0	Adventure
1271	Loetoeng Kasaroeng	7.2	Fantasy

1272 rows × 3 columns

1 .8 Melakukan Filterisasi pada Data

Pandas menyediakan fitur untuk dapat melakukan filter pada data dengan memberikan *conditional expression* seperti (>, ==, !=, <, <=, &, |, ..) yang akan memberikan nilai boolean *True* dan *False*, hanya data yang memiliki nilai *True* yang akan terpilih.

```
diatas_rating_8 = df_indo_film[df_indo_film['users_rating'] > 8.0 ]
diatas_rating_8
```

	title	year	description	genre	rating	users_rating	votes	languages	directors	actors	runtime
3	Anak Garuda	2020	Good Morning Indonesia, a school for poor orph...	Adventure	13+	9.1	27	Indonesian	Faozan Rizal	['Tissa Biani Azzahra', 'Viola Georgie', 'Aji...	129 min
9	Mariposa	2020	Iqbal (Angga Yunanda) is like a Mariposa butte...	Drama	13+	8.5	54	Indonesian	Fajar Bustomi	['Angga Yunanda', 'Adhisty Zara', 'Dannia Sals...	117 min
22	27 Steps of May	2019	Following a horrible experience, May has isola...	Drama	17+	8.2	280	Indonesian	Ravi L. Bharwani	['Raihaanun Soeriaatmadja', 'Lukman Sardi', 'A...	112 min
29	Anak Muda Palsu	2019	Tumming, Abu, Illank, and Darwis, the final se...	Comedy	13+	8.8	23	Indonesian	Ihdar Nur	['Tumming', 'Abu', 'Reo Ramadhan', 'Hisyam Ham...	103 min
	Horas Amang: Tiga Bulan		The story of a family that is not							['Cok Simbara', 'Tanta Ginting',	

Apabila ingin melakukan filter dengan banyak kondisi maka dapat dilakukan dengan menambahkan *conditional expression* pada program.

```
# Film Comdey dengan rating diatas 8
comedy_diatas_rating_8 = df_indo_film.loc[(df_indo_film['users_rating'] > 8.0) & (df_indo_film['genre'] == 'Comedy')]
comedy_diatas_rating_8
```

	title	year	description	genre	rating	users_rating	votes	languages	directors	actors	runtime
29	Anak Muda Palsu	2019	Tumming, Abu, Illank, and Darwis, the final se...	Comedy	13+	8.8	23	Indonesian	Ihdar Nur	['Tumming', 'Abu', 'Reo Ramadhan', 'Hisyam Ham...	103 min
52	Horas Amang: Tiga Bulan Untuk Selamanya	2019	The story of a family that is not harmonious: ...	Comedy	13+	9.4	16	Indonesian	Irham Acho Bahtiar	['Cok Simbara', 'Tanta Ginting', 'Novita Dewi'...	109 min
148	Baco Becce	2018	NaN	Comedy	13+	8.8	23	Indonesian	Syahrir Arsyad Dini	['Mimi Peri', 'Syukri Algazali', 'Mamak Ramlah...	87 min
155	Cinta sama dengan Cindolo Na Tape	2018	Timi (A. Thesar Resandy) falls in love with Ci...	Comedy	13+	8.3	9	Indonesian	Andi Burhamzah	['A. Thesar Resandy', 'Maizura', 'Brillan Rexy...	NaN
227	Yowis Ben	2018	Bayu falls in love with a girl and decided to ...	Comedy	13+	8.4	3,080	Indonesian	Fajar Nugros	['Bayu Skak', 'Cut Meyriska', 'Brandon Salim',...	NaN

1 .9 Membuat Dataframe dari Nilai tertentu

Membuat suatu *Dataframe* berdasarkan nilai fungsi yang telah dibuat menggunakan fungsi, yang di awali dengan memetakan nilai x pad ainterval tertentu dan memasukkan nilai fungsi y terhadap nilai x. Pemetaan fungsi ini menggunakan *library* Numpy dan Matplotlib.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

x = np.linspace(0, 10, 10)
y = x**2 + 10*x + 2
dataframe = {
    "Nilai X" : x,
    "Nilai Y" : y
}
df_y1 = pd.DataFrame(dataframe)
df_y1
```

	Nilai X	Nilai Y
0	0.000000	2.000000
1	1.111111	14.345679
2	2.222222	29.160494
3	3.333333	46.444444
4	4.444444	66.197531
5	5.555556	88.419753
6	6.666667	113.111111
7	7.777778	140.271605
8	8.888889	169.901235

```
df_y1["Nilai Y^2"] = df_y1["Nilai Y"] * df_y1["Nilai Y"]
df_y1["Nilai XY"] = df_y1["Nilai Y"] * df_y1["Nilai X"]
df_y1["Nilai 5y"] = df_y1["Nilai Y"] * 5
df_y1
```

	Nilai X	Nilai Y	Nilai Y^2	Nilai XY	Nilai 5y
0	0.000000	2.000000	4.000000	0.000000	10.000000
1	1.111111	14.345679	205.798506	15.939643	71.728395
2	2.222222	29.160494	850.334400	64.801097	145.802469
3	3.333333	46.444444	2157.086420	154.814815	232.222222
4	4.444444	66.197531	4382.113093	294.211248	330.987654
5	5.555556	88.419753	7818.052736	491.220850	442.098765
6	6.666667	113.111111	12794.123457	754.074074	565.555556
7	7.777778	140.271605	19676.123152	1091.001372	701.358025
8	8.888889	169.901235	28866.429508	1510.233196	849.506173
9	10.000000	202.000000	40804.000000	2020.000000	1010.000000

1 .10 Mengubah Nama dari Kolom

Pandas menyediakan fitur untuk dapat mengubah nama dari kolom secara manual, dengan mendeklarasikan nama kolom awal dan nama kolom akhir dengan tipe data *dictionary*.

```
# Rename Kolom "Nilai X" to "Himpunan X"
df_y1 = df_y1.rename(
    columns = {
        "Nilai X" : "Himpunan X"
    }
)
df_y1
```

	Himpunan X	Nilai Y	Nilai Y^2	Nilai XY	Nilai 5y
0	0.000000	2.000000	4.000000	0.000000	10.000000
1	1.111111	14.345679	205.798506	15.939643	71.728395
2	2.222222	29.160494	850.334400	64.801097	145.802469
3	3.333333	46.444444	2157.086420	154.814815	232.222222
4	4.444444	66.197531	4382.113093	294.211248	330.987654
5	5.555556	88.419753	7818.052736	491.220850	442.098765
6	6.666667	113.111111	12794.123457	754.074074	565.555556
7	7.777778	140.271605	19676.123152	1091.001372	701.358025
8	8.888889	169.901235	28866.429508	1510.233196	849.506173
9	10.000000	202.000000	40804.000000	2020.000000	1010.000000

1 .11 Mengelompokkan Data

Melakukan pengelompokkan data pada kolom tertentu kedalam sub-bagian tertentu. Proses pengelompokkan diawali dengan mengubah memilih kolom tertentu dari *Dataframe*, kemudian dari kolom tersebut akan di kelompokkan kembali ke dalam kategori tertentu dalam bentuk objek, untuk dapat dilakukan proses analisis lebih lanjut seperti mencari nilai standar deviasi, mean, median, kuartil dan min.

Data yang digunakan adalah data populasi negara asia dalam bentuk CSV, yang akan di konversi ke dalam bentuk *Dataframe* dan kemudian menggunakan fungsi `head()` untuk dapat melihat kelima data awalnya.

```
df_asia = pd.read_csv('Data Populasi Negara Asia.csv')
df_asia.head()
```

	country	year	population
0	Indonesia	1952	82052000
1	Indonesia	1957	90124000
2	Indonesia	1962	99028000
3	Indonesia	1967	109343000
4	Indonesia	1972	121282000

Data akan dikelompokkan berdasarkan negara nya untuk kemudian dicari komponen nilai statistik dari dua komponen kolom integer yakni *year* dan *population.* Apabila ingin mengamati data pada kolom tertentu dapat dengan merincikannya seperti gambar dibawah:

```
df_asia.groupby('country').describe()
```

	year								population							
	count	mean	std	min	25%	50%	75%	max	count	mean	std	min	25%	50%	75%	max
country																
Cambodia	12.0	1979.5	18.027756	1952.0	1965.75	1979.5	1993.25	2007.0	12.0	8.510431e+06	3.053246e+06	4693836.0	6.740955e+06	7361545.5	1.055831e+07	14131858.0
Indonesia	12.0	1979.5	18.027756	1952.0	1965.75	1979.5	1993.25	2007.0	12.0	1.483228e+08	4.915754e+07	82052000.0	1.067642e+08	145034000.0	1.884315e+08	223547000.0
Malaysia	12.0	1979.5	18.027756	1952.0	1965.75	1979.5	1993.25	2007.0	12.0	1.457406e+07	5.990940e+06	6748378.0	9.842755e+06	13643648.5	1.885865e+07	24821286.0
Philippines	12.0	1979.5	18.027756	1952.0	1965.75	1979.5	1993.25	2007.0	12.0	5.263663e+07	2.286965e+07	22438691.0	3.409877e+07	50153868.0	6.914257e+07	91077287.0
Singapore	12.0	1979.5	18.027756	1952.0	1965.75	1979.5	1993.25	2007.0	12.0	2.667817e+06	1.090842e+06	1127000.0	1.920750e+06	2488584.5	3.377476e+06	4553009.0

```
df_asia.groupby('country')['population'].describe()
```

country	count	mean	std	min	25%	50%	75%	max
Cambodia	12.0	8.510431e+06	3.053246e+06	4693836.0	6.740955e+06	7361545.5	1.055831e+07	14131858.0
Indonesia	12.0	1.483228e+08	4.915754e+07	82052000.0	1.067642e+08	145034000.0	1.884315e+08	223547000.0
Malaysia	12.0	1.457406e+07	5.990940e+06	6748378.0	9.842755e+06	13643648.5	1.885865e+07	24821286.0
Philippines	12.0	5.263663e+07	2.286965e+07	22438691.0	3.409877e+07	50153868.0	6.914257e+07	91077287.0
Singapore	12.0	2.667817e+06	1.090842e+06	1127000.0	1.920750e+06	2488584.5	3.377476e+06	4553009.0

1 .12 Menggabungkan Kedua Dataframe

Menggabungkan kedua *Dataframe* yang terpisah dari fungsi *Dataframe* y_1 dan fungsi dari *Dataframe* y_2, fungsi y_2 dapat di jabarkan sebagai fungsi berikut:

```
# Menggabungkan dua data tabel
# Menggabungkan data
x2 = np.linspace(0, 10, 10)
y2 = x**3
dataframe2 ={
        "NIlai x ke-2" : x2,
        "Nilai y ke-2" : y2
    }
df_y2 = pd.DataFrame(dataframe2)
df_y2
```

	NIlai x ke-2	Nilai y ke-2
0	0.000000	0.000000
1	1.111111	1.371742
2	2.222222	10.973937
3	3.333333	37.037037
4	4.444444	87.791495
5	5.555556	171.467764
6	6.666667	296.296296
7	7.777778	470.507545
8	8.888889	702.331962
9	10.000000	1000.000000

Setelah menggabungkan kedua *Dataframe* menggunakan fungsi `pd.concat()` dengan memasukkan kedua *Dataframe* ke dalam bentuk tipe data *List.*

```
df_gabung_y1_y2 = pd.concat([df_y1, df_y2], axis = 1)
df_gabung_y1_y2
```

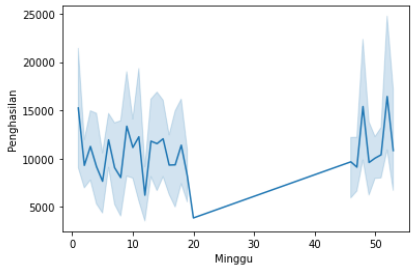
	Himpunan X	Nilai Y	Nilai Y^2	Nilai XY	Nilai 5y	NIlai x ke-2	Nilai y ke-2
0	0.000000	2.000000	4.000000	0.000000	10.000000	0.000000	0.000000
1	1.111111	14.345679	205.798506	15.939643	71.728395	1.111111	1.371742
2	2.222222	29.160494	850.334400	64.801097	145.802469	2.222222	10.973937
3	3.333333	46.444444	2157.086420	154.814815	232.222222	3.333333	37.037037
4	4.444444	66.197531	4382.113093	294.211248	330.987654	4.444444	87.791495
5	5.555556	88.419753	7818.052736	491.220850	442.098765	5.555556	171.467764
6	6.666667	113.111111	12794.123457	754.074074	565.555556	6.666667	296.296296
7	7.777778	140.271605	19676.123152	1091.001372	701.358025	7.777778	470.507545
8	8.888889	169.901235	28866.429508	1510.233196	849.506173	8.888889	702.331962
9	10.000000	202.000000	40804.000000	2020.000000	1010.000000	10.000000	1000.000000

1 .13 Plot Garis dengan Seaborn

Dalam menganilisis data dalam jumlah yang besar akan lebih akurat jika menggunakan seaborn ketimbang menggunakan matplotlib, karena terdapat error dari visualisasi yang dilakukan oleh matplotlib seperti ini:

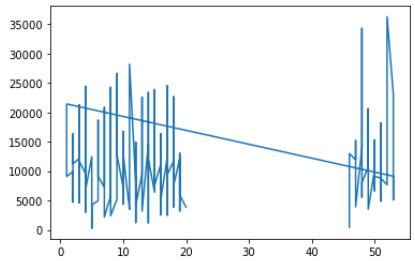
```
import seaborn as sns
sns.lineplot(x='Minggu ', y='Penghasilan', data=pasar)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x29ef3e87fa0>
```




```
import matplotlib.pyplot as plt
plt.plot(pasar['Minggu '], pasar['Penghasilan'])
```

```
[<matplotlib.lines.Line2D at 0x29ef4196b20>]
```

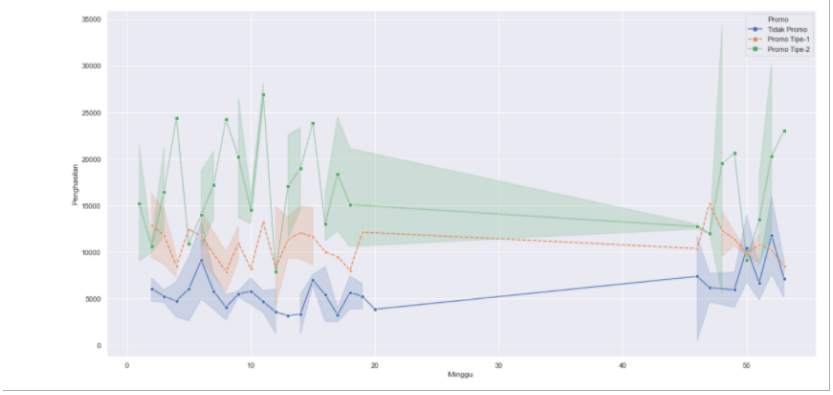


Terdapat error yang dihasilkan dari visualisasi menggunakan matplotlib seperti pada gambar, hal ini membuat seaborn lebih cocok untuk digunakan dalam analisis statistik, selain itu seaborn dilengkapi dengan garis samar-samar yang bertindak sebagai Interval Kepercayaan yang membuat analisis semakin akurat, untuk menggunakan fitur Seaborn diperlukan fungsi awal yakni:

```
Import seaborn as sns
```

Kemudian untuk membuat plot line dengan mendeklarasikan kolom mana yang ingin dijadikan axis-x dan variabel yang mana yang ingin dijadikan axis-y serta sumber data yang ingin digunakan .

```
In [44]: import seaborn as sns
         sns.lineplot(x='Minggu ', y='Penghasilan', data=pasar, style='Promo', markers=True, hue='Promo', sizes=(2,15))
Out[44]: <matplotlib.axes._subplots.AxesSubplot at 0x29ef583d520>
```



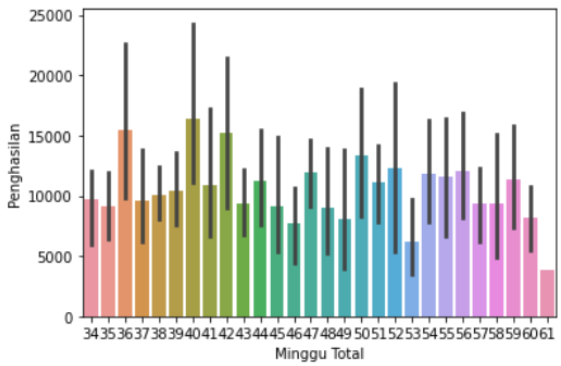
Jika ingin mengklasifikan plot menjadi tiga bagian dapat digunakan fungsi ‘hue’ dan untuk memberikan penanda menggunakan fungsi markers :

```
hue=’variabel’
markers=True
```

1 .14 Plot Histogram dengan Seaborn

Menggunakan data dari data eksternal untuk menghasilkan histogram, garis hitam di tengah menunjukkan nilai interval kepercayaan

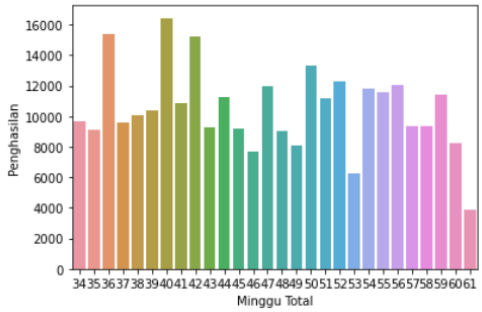
```
ax=sns.barplot(x='Minggu Total', y='Penghasilan', data=pasar)
```



Untuk menghilangkan nilai interval kepercayaan dapat menggunakan fungsi

```
Ci=False
```

```
ax=sns.barplot(x='Minggu Total', y='Penghasilan', data=pasar, ci=False)
```

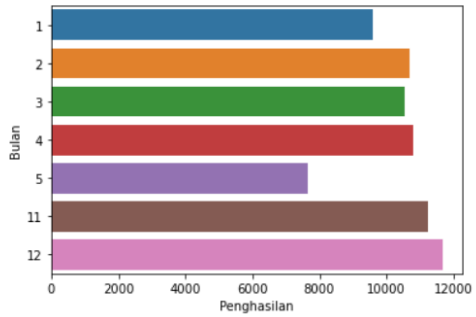


Untuk membuat data dalam bentuk horizontal, dapat mengganti

`Orient='h'`

`Dengan 'h' sebagai horizontal`

```
ax=sns.barplot(x='Penghasilan', y='Bulan', data=pasar, ci=False, orient='h')
```



1 .15 Plot Distribusi dengan Seaborn

Menggunakan grafik distribusi untuk memplot grafik yang diinginkan yang dilengkapi dengan KDE (Kernel Density Estimacy) yang mampu memperkirakan tingkat probabilitas variabel acak pada grafik menggunakan fungsi values untuk mendapatkan nilai di dalam kolom yang diinginkan menjadi suatu array

`pasar['Penghasilan'].values`

```
array_penghasilan=pasar['Penghasilan'].values
print(array_penghasilan)
```

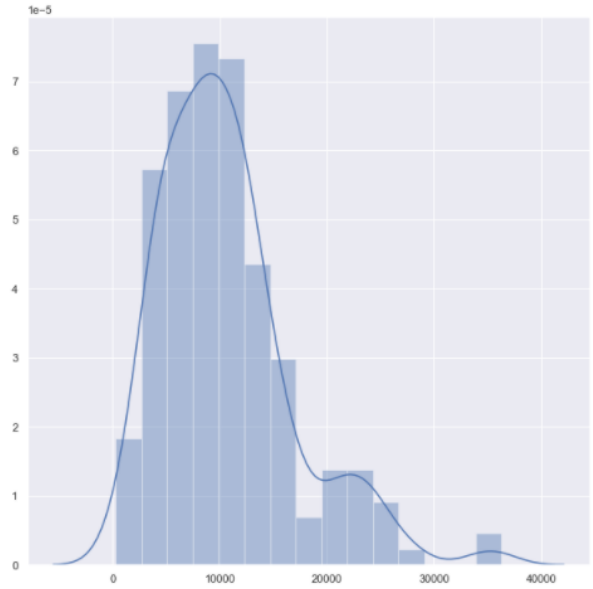
```
[  465 10386 12475 11712 10000 12996 11929  5359 12016  7441  8000 15188
  3926 14012 34278 18650  5574 12425 14760  8091 10647  7290  4587 12154
  8400 20607  3525 10075  9612  6908 12740 15288  6654  9097  8802 11679
 11952  4873  8516 18161  8800  7655 12038 10245 11817 15991 21031 36283
 23014  5100  7793  9180  9125 21428  9062  9952  4714  9418 16325  6254
  7215 11249 12129  4581  8702  5937 14568 21245 11699  9602  8633  7287
  4403  3021 24417  6837 12445 10925   292  4085 11156 10485  4225  4953
 10479 12783 18678 13281 14356  9095  7264  5263 20860 13602  5519  8741
  2184  5772  9987  5966  4910 24247  3025  2409  5171  5847  9135 13719
 20237 26608 12789  7222 13212  4376  8223 16727 15474 12850  3470  3589
  5187 13201 25704  6523 28196  7878  3378  6047  1245 14909  5903  4169
  9486  9209 11494 15190 11524 22587  3181 14587  9262  5478 14957 23368
  1201 12027  6388  8396  8828 16777 12548 23870  7596 11221 10027 11484
 16347  2546  8456  5274 12251  9645  4020  2491 24506  3251  9360 11650
  3884  8046  9540 22655 16542  7445 12111 11273  3215  7284 13021  4587
  5927  3861]
```

Fungsi yang digunakan untuk membuat grafik dari distribusi plot

`sns.displot(variabel)`

```
array_penghasilan=pasar['Penghasilan'].values
sns.distplot(array_penghasilan)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x29ef6505d00>
```

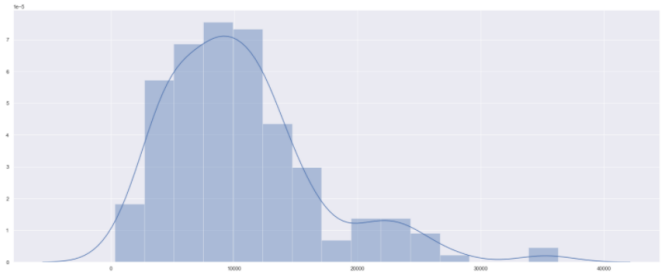


Untuk mengubah ukuran grafik dapat menggunakan fungsi

`sns.set(rc={'figure.figsize':(baris,kolom)})`

```
import seaborn as sns
array_penghasilan=pasar['Penghasilan'].values
sns.set(rc={'figure.figsize':(25,10)})
sns.distplot(array_penghasilan)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x29ef61b2a60>
```



Untuk memberikan nilai median dan mean dapat menggunakan fungsi

`Variabel.mean()`

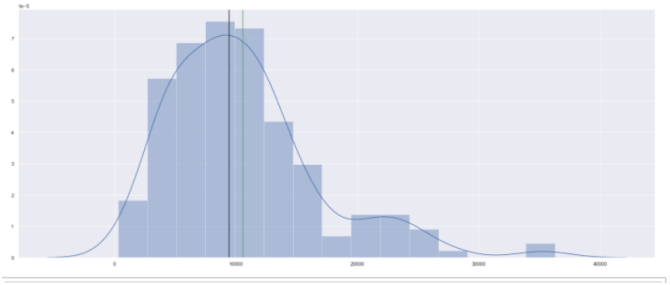
`Variabel.median()`

Diberikan garis vertikal untuk memudahkan proses identifikasi dalam mencari letak median dan mean menggunakan fungsi

`Plot.axvline(variabel)`

```
import seaborn as sns
import numpy as np
array_penghasilan=pasar['Penghasilan'].values
mean=np.mean(array_penghasilan)
median=np.median(array_penghasilan)
sns.set(rc={'figure.figsize':(25,10)})
sns.distplot(array_penghasilan)
plt.axvline(median, color='k')
plt.axvline(mean, color='g')
```

```
<matplotlib.lines.Line2D at 0x29ef6025eb0>
```



1 .16 Plot Box dengan Seaborn

Membuat grafik boxplot yang memuat informasi dari nilai yang sudah di transformasikan ke dalam bentuk array, kemudian menambahkan fungsi median untuk menambah informasi pada grafik.

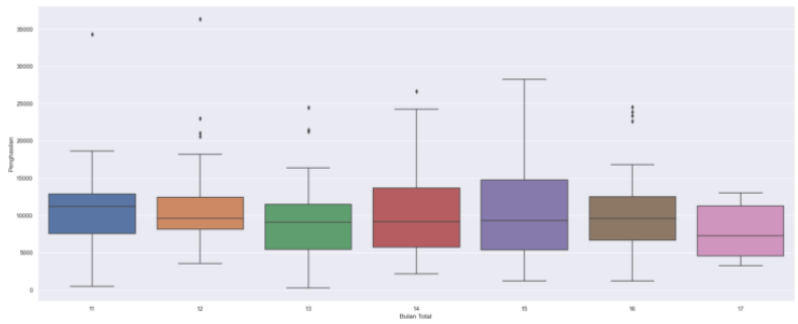
```
x=pasar['Penghasilan'].values
sns.boxplot(x)
median=pasar['Penghasilan'].median()
plt.axvline(median)
print("Median adalah " + str(median))
```

```
Median adalah 9452.0
```

Apabila ingin membuat grafik dengan dua variabel dapat menggunakan fungsi

```
sns.boxplot(x='Bulan Total ', y='Penghasilan', data=pasar)
```

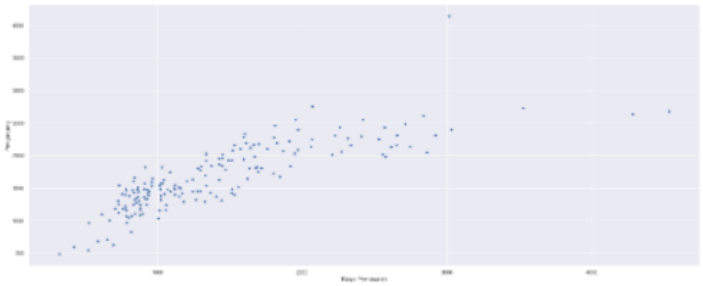
```
<matplotlib.axes._subplots.AxesSubplot at 0x20287387b20>
```



1 .17 Scatterplot dengan Seaborn

Scatterplot merupakan grafik yang memiliki pola titik-titik dalam merepresentasikan atau memvisualisasikan suatu data

```
In [18]: sns.scatterplot(x='Biaya Pemasaran', y='Pengunjung', data=pasar)
Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x168338c9370>
```

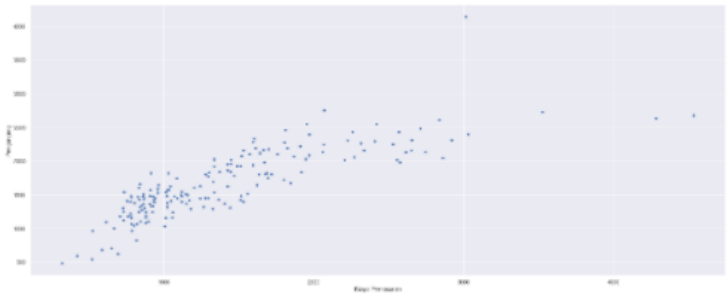


Apabila ingin mengklasifikasikan data menjadi tiga kategori dapat dengan menambahkan fungsi

`Style='String pada Kolom'`

Sehingga dapat dilihat, masing-masing data, memiliki bentuk yang berbeda beda seperti bentuk bulat, silang maupun kotak.

```
In [18]: sns.scatterplot(x='Biaya Pemasaran', y='Pengunjung', data=pasar)
Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x168338c9370>
```



Untuk memberikan warna pada masing-masing kategori dapat menggunakan:

`hue=’String pada Kolom’`

Fungsi Size digunakan untuk memberikan perbedaan pada besar nilai data yang di referensikan, pada contoh dibawah semakin besar Penghasilan didapat maka besar bentuk scatter akan semakin besar terlihat.

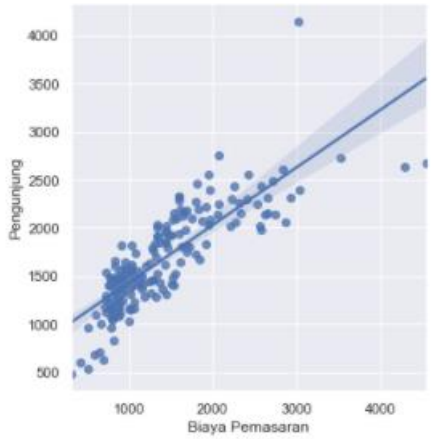
`size=’String pada Kolom’`

```
In [22]: sns.scatterplot(x='Biaya Pemasaran', y='Pengunjung', hue='Promo',
                         style='Promo', size='Penghasilan',data=pasar)
Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x16833ce4a90>
```

1 .18 Linear Regresi dengan Seaborn

Linear Regresi merupakan suatu metode yang digunakan untuk melakukan prediksi terhadap variabel tertentu, yang terdiri atas variabel bebas dan variabel terikat.

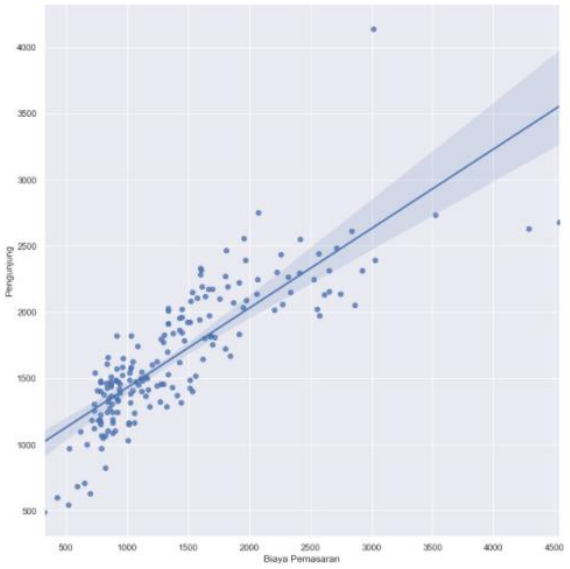
```
In [41]: sns.lmplot(x='Biaya Pemasaran', y='Pengunjung',data=pasar)
Out[41]: <seaborn.axisgrid.FacetGrid at 0x168368d0e20>
```



Untuk memperbesar grafik dapat menggunakan fungsi

`height = Integer`

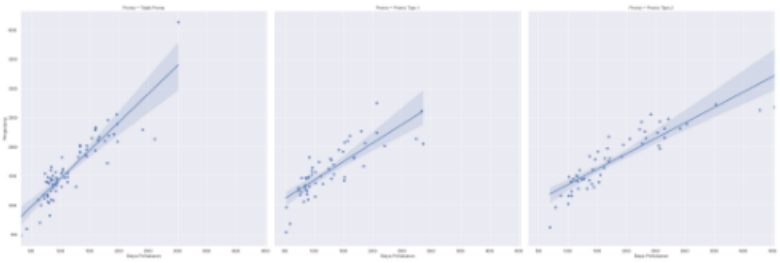
```
In [43]: sns.lmplot(x='Biaya Pemasaran', y='Pengunjung',data=pasar, height=10)
Out[43]: <seaborn.axisgrid.FacetGrid at 0x16836e3daf0>
```



Untuk membagi grafik ke dalam suatu kategori dapat digunakan fungsi

`col=’String dalam kolom’`

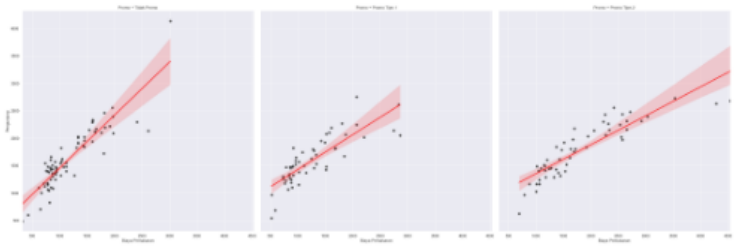
```
In [46]: sns.lmplot(x='Biaya Pemasaran', y='Pengunjung',data=pasar,
                    height=10, col='Promo')
Out[46]: <seaborn.axisgrid.FacetGrid at 0x1683a3804f0>
```



Untuk mengatur agar mengganti warna pada garis Linear Regresi dan Scatter plot tidak sama, maka dapat menggunakan fungsi

```
line_kws={'color':'red'}
scatter_kws={'color':'k'}
```

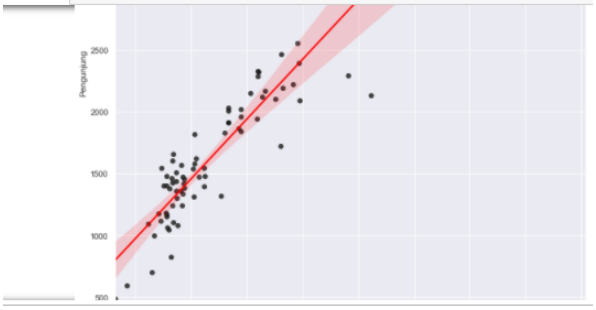
```
In [47]: sns.lmplot(x='Biaya Pemasaran', y='Pengunjung',data=pasar,line_kws={'color':'red
                    height=10, col='Promo')
Out[47]: <seaborn.axisgrid.FacetGrid at 0x1683aeece20>
```



Untuk mengatur agar berapa jumlah grafik yang dapat ditunjukkan pada satu baris dapat menggunakan

`col_wrap=Integer`

```
In [51]: sns.lmplot(x='Biaya Pemasaran', y='Pengunjung',data=pasar,
                    line_kws={'color':'red'},
                    scatter_kws={'color':'k'},
                    height=10, col='Promo',col_wrap=1)
```



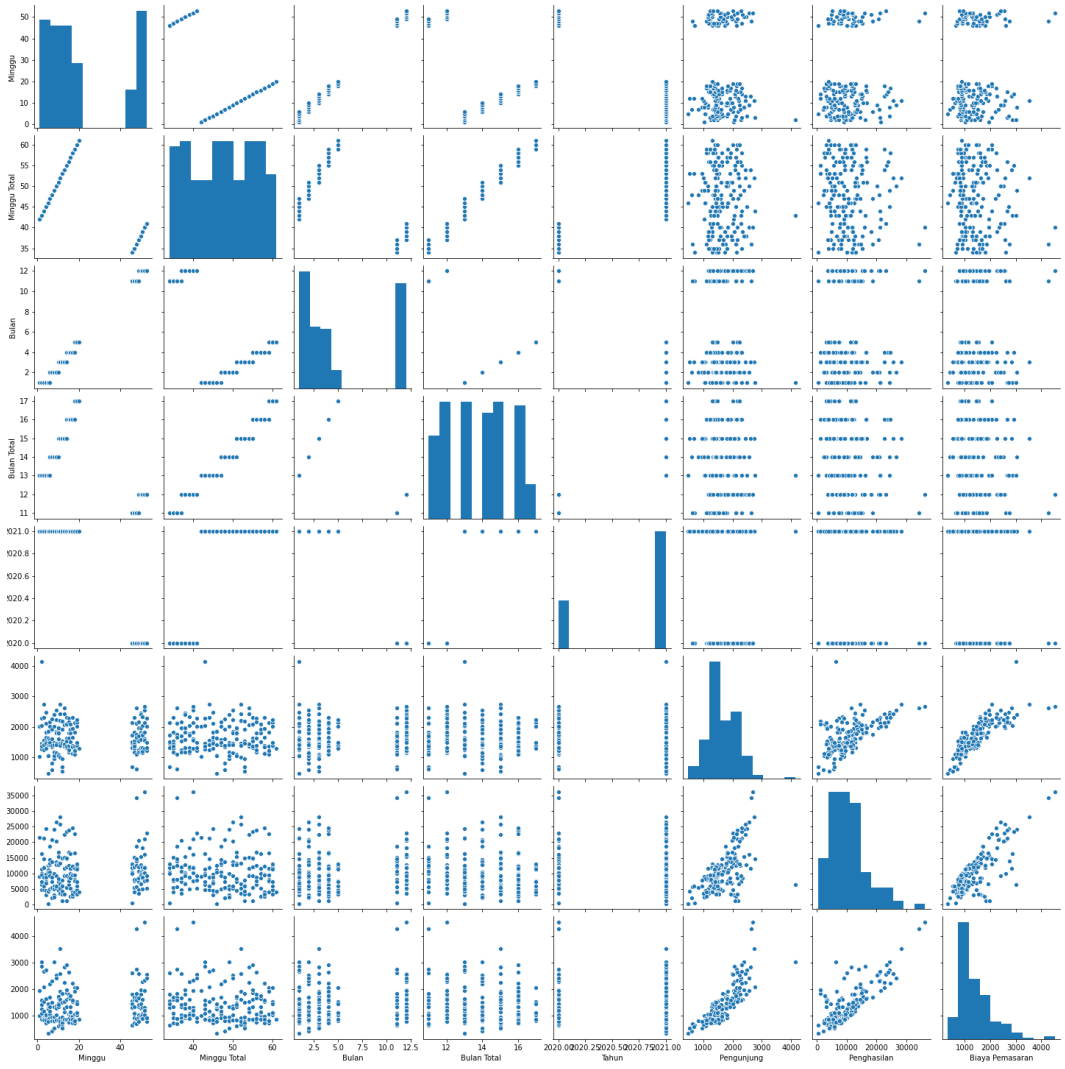
1 .19 Pairplot dengan Seaborn

Fungsi pairplot bertujuan untuk menghasilkan grafik terhadap pada seluruh data dengan waktu yang bersamaan.

```
In [27]: sns.pairplot(pasar)
```

Fungsi tersebut akan menghasilkan grafik yang memvisualisasikan seluruh data dari data yang telah di impor sebelumnya

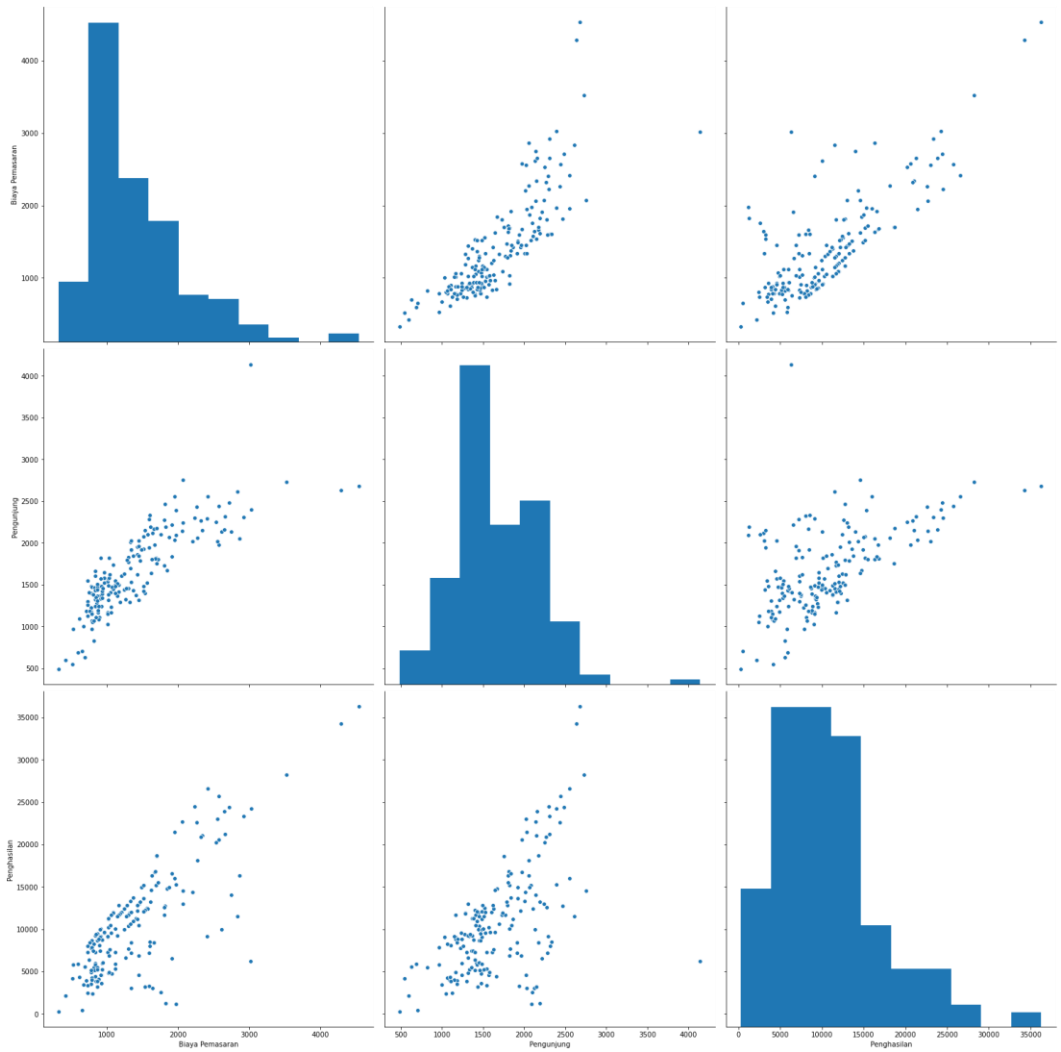
`sns.pairplot(variabel)`



Untuk memilah data tertentu yang ingin di visualisasikan dapat menggunakan fungsi list terhadap data yang ingin dimasukkan
Data_impor[[‘String pada kolom’]]

```
In [52]: sns.pairplot(pasar[['Biaya Pemasaran', 'Pengunjung','Penghasilan']], height=7)
```

Fungsi height digunakan untuk mengubah ukuran dari grafik
`height=integer`

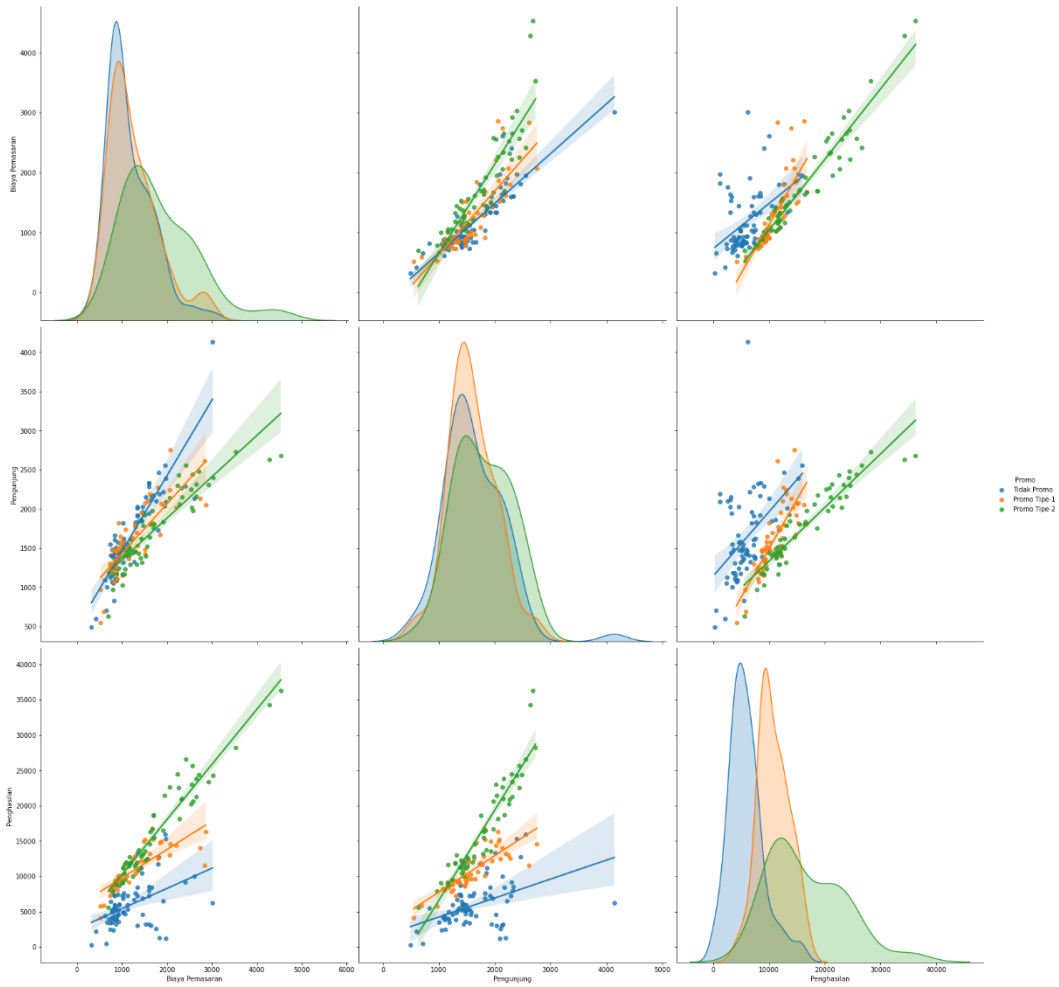


Untuk membagi kedalam beberapa kategori dapat ditambahkan fungsi hue dan menambahkan string yang ingin dimasukkan ke dalam fungsi list, yang akan menampilkan scatterplot pada grafik dua variabel dan grafik distribusi untuk grafik satu variabel

```
In [57]: sns.pairplot(pasar[['Biaya Pemasaran', 'Pengunjung','Penghasilan', 'Promo']], height=7, hue='Promo', kind='reg')
```

`Kind='reg'`

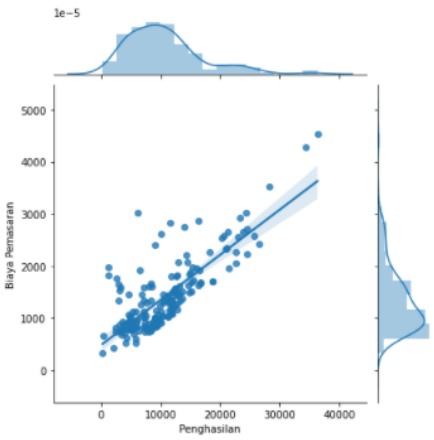
Fungsi diatas digunakan untuk menampilkan ligresi linear pada scatterplot



1 .20 Jointplot pada Seaborn

Fungsi Jointpot merupakan fungsi yang menggabungkan grafik distribusi data terhadap masing-masing variabel dengan grafik scatter sehingga analisis dapat mudah dilakukan.

```
In [63]: sns.jointplot('Penghasilan', 'Biaya Pemasaran', data=pasar, kind='reg')
Out[63]: <seaborn.axisgrid.JointGrid at 0x2cf676e2f10>
```



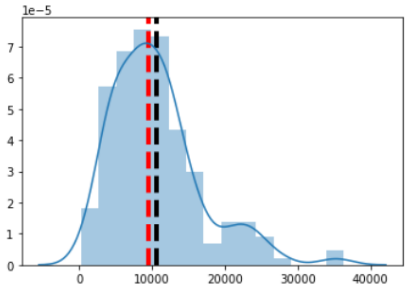
Analisis Praktikum

1. Membuat Dataframe dari fungsi

```
import matplotlib.pyplot as plt
import numpy as np

# Buatlah suatu interval dari [-
2𝝅, 2𝝅] yang dibagi ke dalam 100
x = np.linspace(____, ____, ____)
# Buatlah suatu fungsi sin(2x + 90)
y = __________

# Buatlah suatu dictionary dataframe
dataframe = {
    "Nilai x " : ___,
    "f(x)" : ____
}
# Buatlah suatu dataframe menggunakan pandas
df_fungsi = ______(______)
# Menampilkan tabel dataframe
df_fungsi
```

2. Import File Pandas dan DIstribusi Data

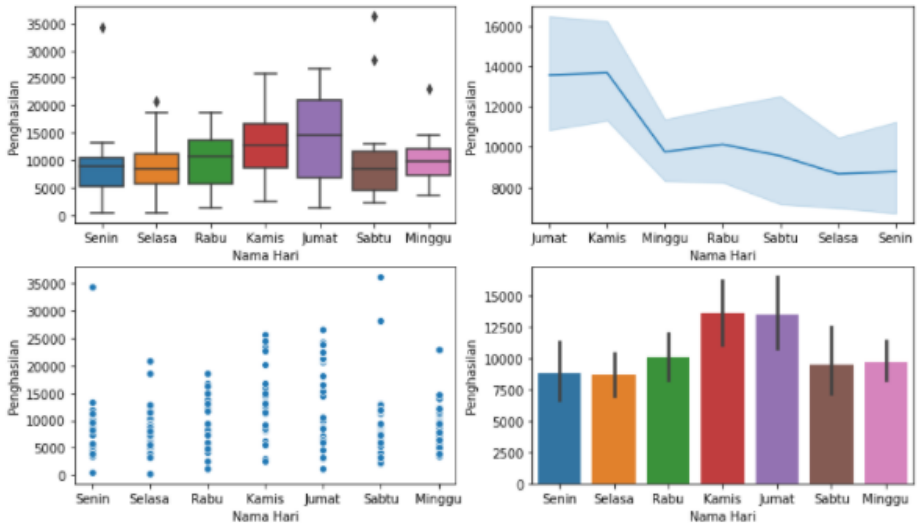
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Bacalah data marketing csv
pasar = pd._____(_____)

# Mengubah data Penghasilan ke dalam bentuk array
x = pasar['Penghasilan'].array

# Carilah fungsi mean dan median dari data kolom Penghasilan
mean = pasar['Penghasilan'].____
median = pasar['Penghasilan'].____

plt.axvline(mean, color = 'k', linewidth = 4, linestyle = '-
-')
plt.axvline(median, color = 'r', linewidth = 4, linestyle =
'--')

# Buatlah tabel distribusi dari data kolom Penghasilan
sns.distplot(__)
plt.show
```



3. Penyusunan Indeks pada Grafik

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

pasar = pd.read_csv('Data Marketing.csv')
# Membuat suatu multi-plot 2x2
fig, axis = plt.subplots(2,2, figsize = (12, 7))

# Buatlah grafik boxplot
sns.___( y = 'Penghasilan', x = 'Nama Hari', data = ___, ax = axis[0,0])
# Buatlah grafik lineplot
sns.___( y = 'Penghasilan', x = 'Nama Hari', data = ___, ax = axis[0,1])
# Buatlah grafik Scatterplot
sns.___( y = 'Penghasilan', x = 'Nama Hari', data = ___, ax = axis[1,0])
# Buatlah grafik Barplot
sns.___( y = 'Penghasilan', x = 'Nama Hari', data = ___, ax = axis[1,1])
```

```
Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x1b4bcb1e550>
```



Tugas Akhir:

(Program beserta gambar keluaran dilampirkan pada saat pengumpulan tugas akhir)

1. Carilah sebuah data dalam format csv di Internet yang memiliki minimal 3 kolom data dan 10 baris data, lalu buatlah plot scatter, plot box dan plot distribusi terhadap satu data yang dipilih pada satu gambar menggunakan indeks pada satu keluaran!

Contoh Output:

```
Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x216e1e30d60>
```

2. Carilah sebuah data dalam format csv di Internet yang memiliki minimal 3 kolom data dan 10 baris data, dan lakukanlah analisis pada setiap kategori dari salah satu kolom, dan carilah nilai mean, median, dan std dari kategori tersebut, kemudian buatlah *Dataframe* dari nilai mean, median, dan std.

Contoh :

Menggunakan Data Marketing.csv

	Tanggal	Minggu	Minggu Total	Bulan	Bulan Total	Tahun	Nama Hari	Pengunjung	Penghasilan	Biaya Pemasaran	Promo
0	09/11/2020	46	34	11	11	2020	Senin	707	465	651.375000	Tidak Promo
1	10/11/2020	46	34	11	11	2020	Selasa	1455	10386	1298.250000	Promo Tipe-1
2	11/11/2020	46	34	11	11	2020	Rabu	1520	12475	1559.375000	Promo Tipe-2
3	12/11/2020	46	34	11	11	2020	Kamis	1726	11712	1801.750000	Tidak Promo
4	13/11/2020	46	34	11	11	2020	Jumat	2134	10000	2614.500000	Tidak Promo
...	...	...	...	...	...	...	...	...	...	...	...
177	05/05/2021	19	60	5	17	2021	Rabu	1400	7284	1119.600000	Tidak Promo
178	06/05/2021	19	60	5	17	2021	Kamis	2244	13021	2067.888889	Promo Tipe-1
179	07/05/2021	19	60	5	17	2021	Jumat	2023	4587	1450.200000	Tidak Promo
180	08/05/2021	19	60	5	17	2021	Sabtu	1483	5927	1121.875000	Tidak Promo
181	09/05/2021	20	61	5	17	2021	Minggu	1303	3861	871.000000	Tidak Promo

182 rows × 11 columns

Nama Hari	Median Pengunjung	Median Penghasilan	Median Biaya Pemasaran	Mean Pengunjung	Mean Penghasilan	Mean Biaya Pemasaran	std Pengunjung	std Penghasilan	std Biaya Pemasaran
Jumat	2026.0	14558.0	1914.725000	1968.423077	13550.576923	1894.765793	390.585527	7638.900157	563.618071
Kamis	2260.5	12784.5	1879.454545	2252.769231	13663.384615	1994.641560	525.839162	6619.879703	561.840965
Minggu	1478.0	9777.0	1047.101010	1493.538462	9751.807692	1202.899213	303.868028	4100.727370	517.717771
Rabu	1589.5	10678.5	1276.800000	1614.884615	10121.846154	1314.621882	461.791843	4967.417290	500.355131
Sabtu	1375.5	8445.5	958.194444	1410.038462	9549.692308	1209.081673	486.792069	7564.858834	881.488033
Selasa	1434.0	8374.0	937.763889	1436.000000	8659.807692	1042.899757	350.615231	4510.280914	393.504376
Senin	1478.5	8921.0	962.850000	1491.461538	8774.038462	1115.586072	315.301155	6129.889014	676.159717