Rakamin Academy X KALBE Nutritionals

Machine Learning Project

Data Scientist Project Based Internship Program

Presented by
Karina Khairunnisa Putri

Karina Khairunnisa Putri

About Me

I am an enthusiastic, highly motivated and analytical individual. As a 5th semester student majoring in Informatics who's passionate about Data Science. Also, I took some certificates are Introduction to Data Science with R and maximize data science skills. With my educational background in computer science, artificial intelligence and statistics, I have sufficient knowledge in data processing, data analysis, and data modeling. Skilled in operating python, SQL, C++, and R program languages. Looking for an opportunity to develop my skills for an internship in Data Science.

Rakamin Academy

My Experience

Experience 1

Participant of Scientific / Interest Competitions in data science with R language

Experience 2

Project IMPACT (Coding competition event for knowledge in Informatics major) in 2022

2. Case Study

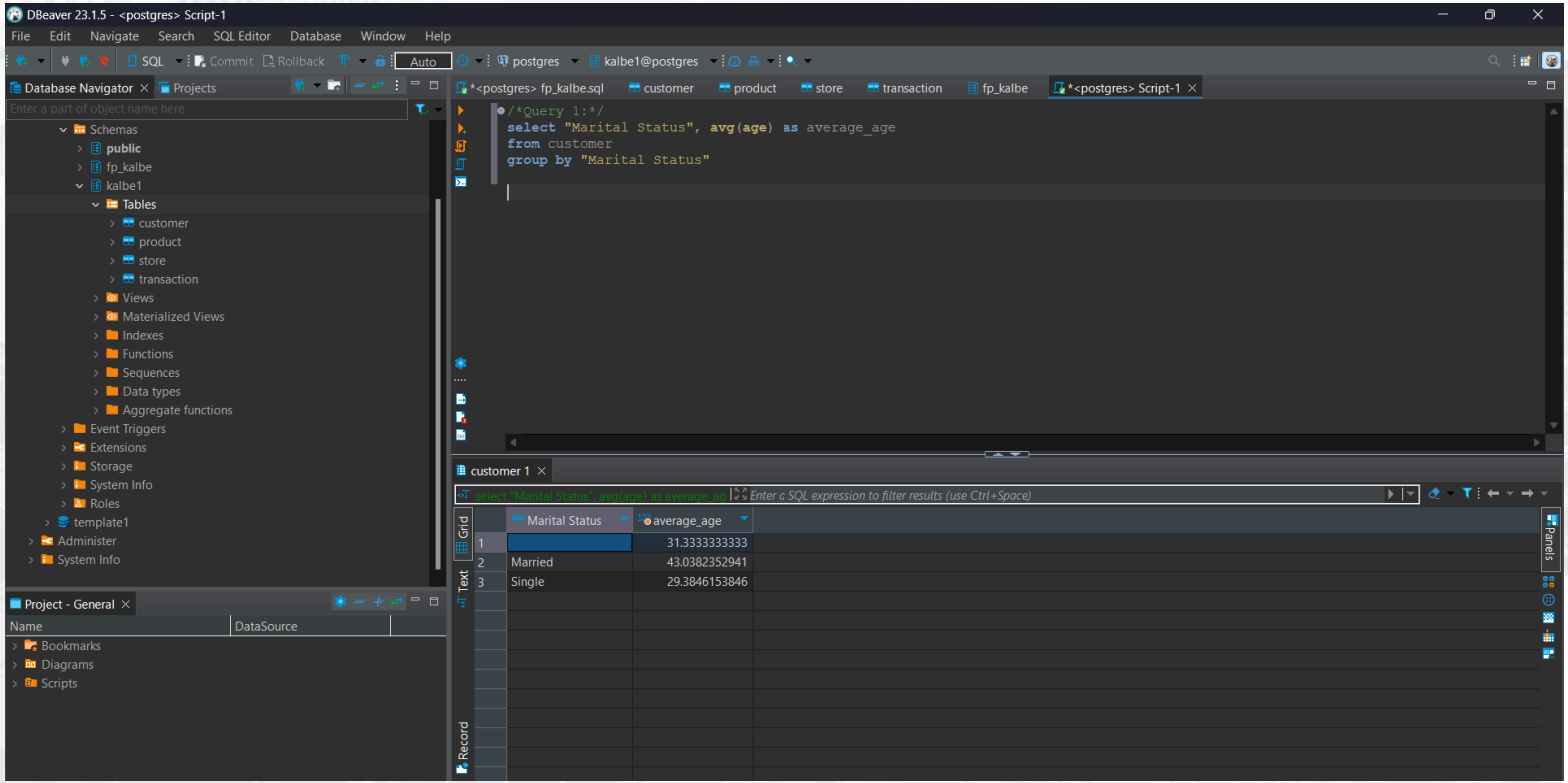
Peserta dapat melakukan exploratory data analysis di dbeaver

Peserta dapat melakukan exploratory data analysis di dbeaver

1. query 1 : Berapa rata-rata umur customer jika dilihat dari marital statusnya ?
2. query 2 : Berapa rata-rata umur customer jika dilihat dari gender nya ?
3. query 3 : Tentukan nama store dengan total quantity terbanyak!
4. query 4 : Tentukan nama produk terlaris dengan total amount terbanyak!

Result Here

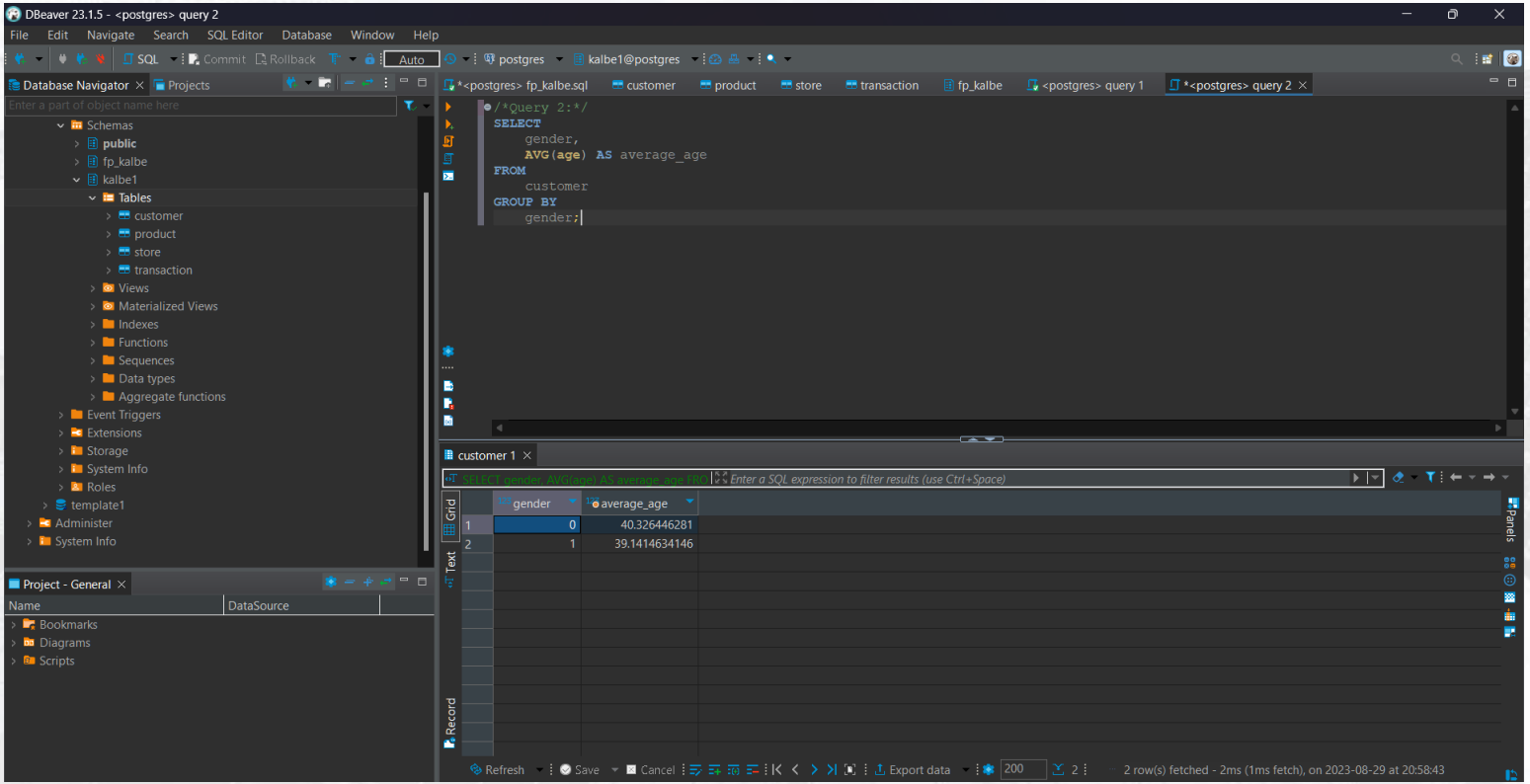
Rata-rata umur customer jika dilihat dari marital statusnya



0 = Wanita, 1 = Pria

Result Here

Rata-rata umur customer jika dilihat dari gender

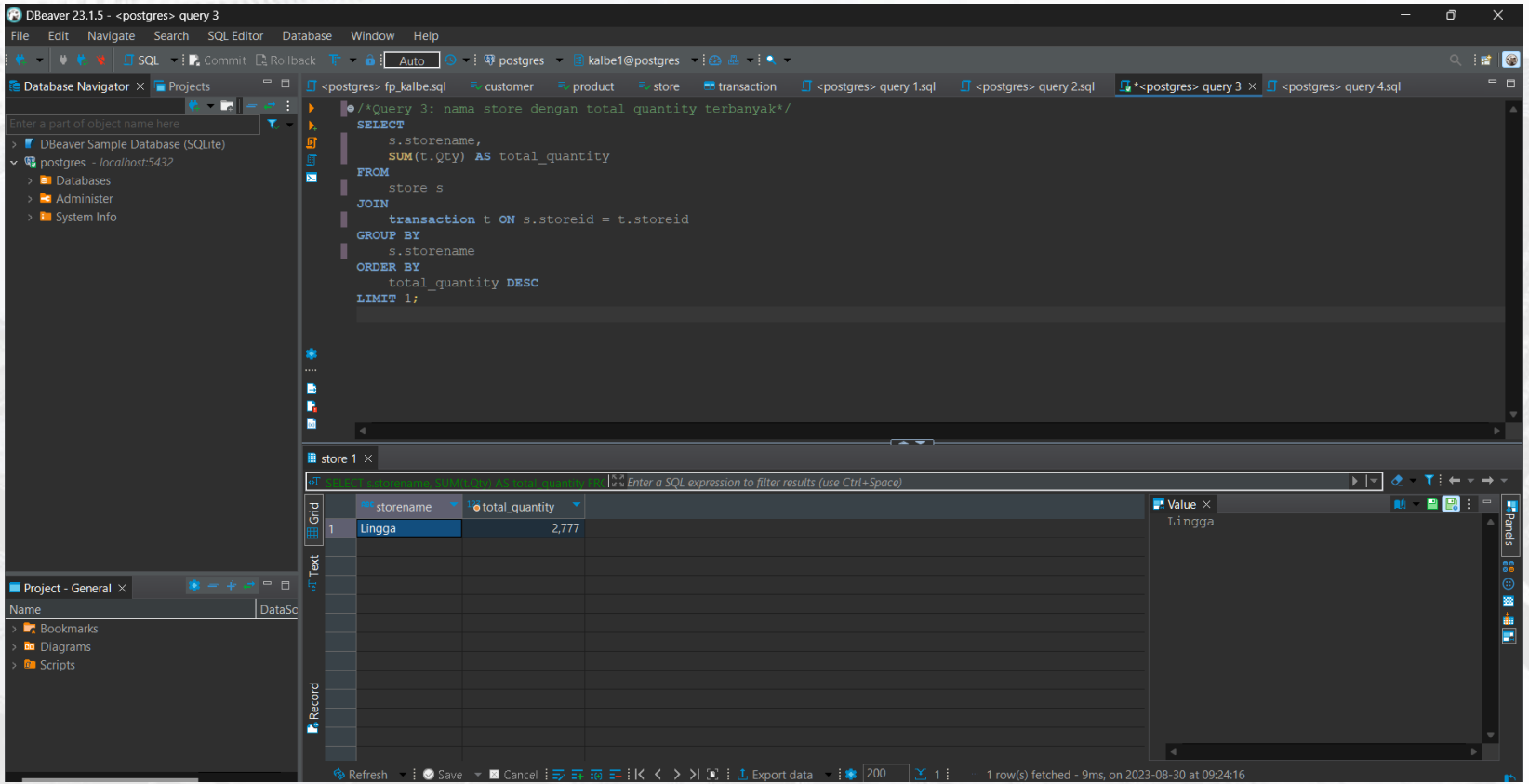


0 = Wanita, 1 = Pria

Result Here

Rakamin Academy

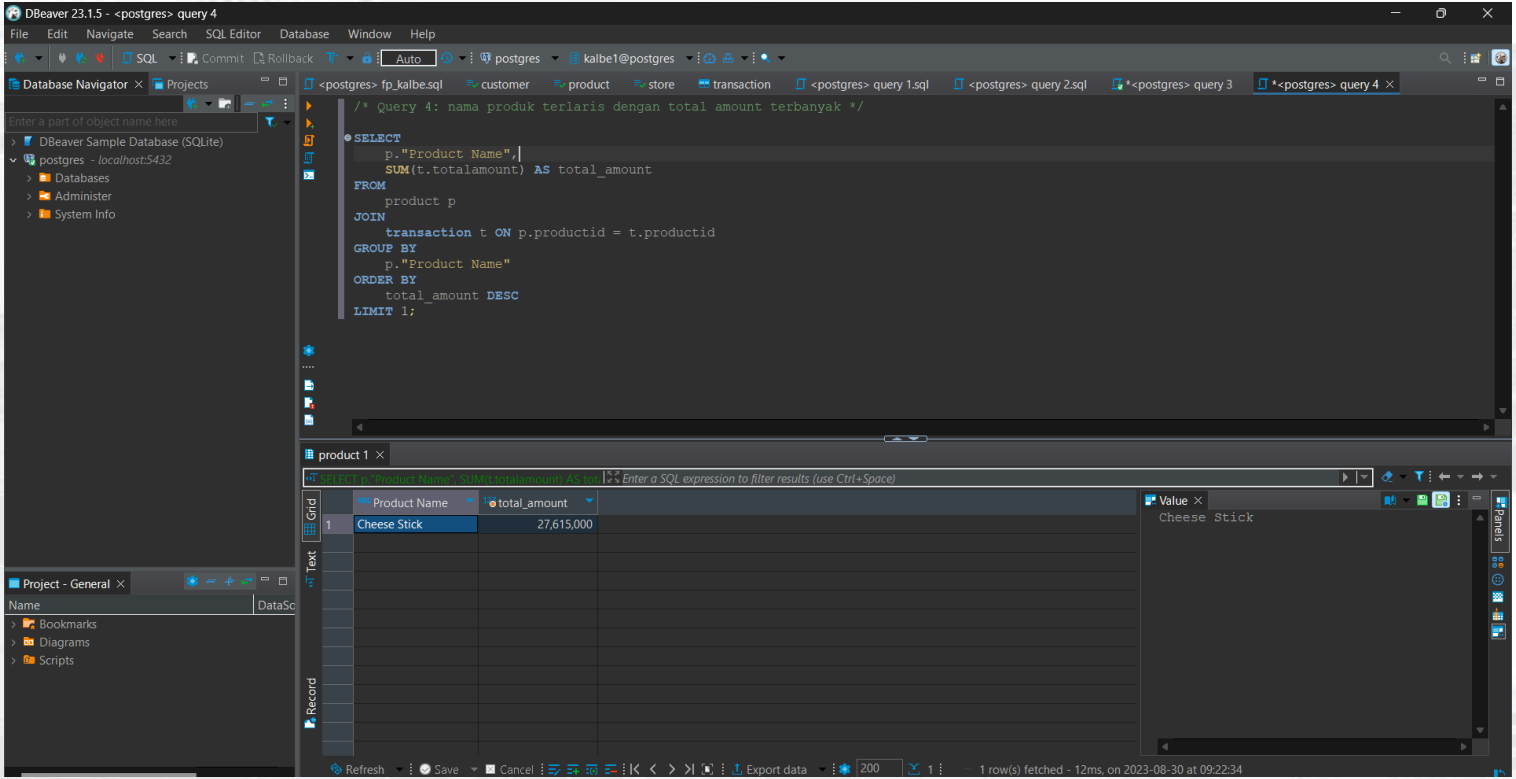
Nama store dengan total quantity terbanyak adalah store Lingga



Result Here

Rakamin Academy

Nama produk terlaris dengan total amount terbanyak adalah cheese stick



1. Case Study

Rakamin Academy

Machine Learning Regression (Time Series)

Tujuan dari pembuatan model machine learning ini adalah untuk dapat memprediksi total quantity harian dari product yang terjual.

- Data cleansing terlebih dahulu, merubah tipe data supaya sesuai
- Data merge untuk menggabungkan semua data
- Membuat data baru untuk regression, yaitu groupby by date lalu yang di aggregasi adalah qty di sum
- Akan ada sekitar 365 rows
- Menggunakan metode time series ARIMA

Result Here

Rakamin Academy

```
#Library import statements
import pandas as pd
import numpy as np

import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn import preprocessing
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.holtwinters import SimpleExpSmoothing, Holt
from statsmodels.tsa.arima.model import ARIMA
from pandas.plotting import autocorrelation_plot

import warnings
```

```
[56] customer_data = pd.read_csv('Case Study - Customer.csv')
     product_data = pd.read_csv('Case Study - Product.csv')
     store_data = pd.read_csv('Case Study - Store.csv')
     transaction_data = pd.read_csv('Case Study - Transaction.csv')
```

Result Here

Rakamin Academy

```
#menampilkan bentuk (shape) dari empat dataframe yang berbeda
customer_data.shape, product_data.shape, store_data.shape, transaction_data.shape
```

```
((447, 5), (10, 3), (14, 6), (5020, 8))
```

```
[7] customer_data.head()
```

	CustomerID	Age	Gender	Marital Status	Income
0	1	55	1	Married	5
1	2	60	1	Married	6
2	3	32	1	Married	9
3	4	31	1	Married	4
4	5	58	1	Married	3

Result Here

Rakamin Academy

```
[8] product_data.head()
```

	ProductID	Product Name	Price
0	P1	Choco Bar	8800
1	P2	Ginger Candy	3200
2	P3	Crackers	7500
3	P4	Potato Chip	12000
4	P5	Thai Tea	4200

```
[9] transaction_data.head()
```

	TransactionID	CustomerID	Date	ProductID	Price	Qty	TotalAmount	StoreID
0	TR11369	328	1/1/2022	P3	7500	4	30000	12
1	TR16356	165	1/1/2022	P9	10000	7	70000	1
2	TR1984	183	1/1/2022	P1	8800	4	35200	4
3	TR35256	160	1/1/2022	P1	8800	7	61600	4
4	TR41231	386	1/1/2022	P9	10000	1	10000	4

```
[10] store_data.head()
```

	StoreID	StoreName	GroupStore	Type	Latitude	Longitude
0	1	Prima Tendean	Prima	Modern Trade	-6	NaN
1	2	Prima Kelapa Dua	Prima	Modern Trade	-6	NaN
2	3	Prima Kota	Prima	Modern Trade	-7	NaN
3	4	Gita Ginara	Gita	General Trade	-6	NaN
4	5	Bonafid	Gita	General Trade	-7	NaN

Result Here

Rakamin Academy

```
[11] #data cleansing customer_data
     customer_data['Income'] = customer_data['Income'].replace('[,]','.', regex = True).astype('float')
```

```
[12] #data cleansing store_data
     store_data['Latitude'] = store_data['Latitude'].replace('[,]','.', regex = True).astype('float')
     customer_data['Longitude'] = store_data['Longitude'].replace('[,]','.', regex = True).astype('float')
```

```
[13] #data cleansing transaction_data
     transaction_data['Date'] = pd.to_datetime(transaction_data['Date'])
```

```
<ipython-input-13-5a407f84c621>:2: UserWarning: Parsing dates in DD/MM/YYYY format when dayfirst=False (the default) was specified. This may le
  transaction_data['Date'] = pd.to_datetime(transaction_data['Date'])
```

```
#menghitung berapa kali setiap nilai yang unik muncul dalam kolom 'TransactionID' dalam dataframe transaction_data
transaction_data['TransactionID'].value_counts()
```

```
TR71313    3
TR42197    2
TR90402    2
TR74019    2
TR49649    2
          ..
TR29792    1
TR23081    1
TR1691     1
TR98388    1
TR85684    1
Name: TransactionID, Length: 4908, dtype: int64
```

Result Here

Rakamin Academy

```
[15] transaction_data [transaction_data['TransactionID'] == 'TR71313']
```

	TransactionID	CustomerID	Date	ProductID	Price	Qty	TotalAmount	StoreID
1982	TR71313	117	2022-05-21	P1	8800	10	88000	8
3336	TR71313	401	2022-08-30	P3	7500	6	45000	11
3722	TR71313	370	2022-09-26	P3	7500	2	15000	3

Gabung Data

```
[16] merge_data = pd.merge(transaction_data, customer_data, on = ['CustomerID'])
     merge_data = pd.merge(merge_data, product_data.drop(columns = ['Price']), on = ['ProductID'])
     merge_data = pd.merge(merge_data, store_data, on = ['StoreID'])
```

Result Here

Rakamin Academy

```
[17] merge_data.head()
```

	TransactionID	CustomerID	Date	ProductID	Price	Qty	TotalAmount	StoreID	Age	Gender	Marital Status	Income	Longitude_x	Product Name	StoreName	GroupStore	Type	Latitude	Longitude_y
0	TR11369	328	2022-01-01	P3	7500	4	30000	12	36	0	Married	10.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN
1	TR89318	183	2022-07-17	P3	7500	1	7500	12	27	1	Single	0.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN
2	TR9106	123	2022-09-26	P3	7500	4	30000	12	34	0	Married	4.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN
3	TR4331	335	2022-08-01	P3	7500	3	22500	12	29	1	Single	4.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN
4	TR6445	181	2022-10-01	P3	7500	4	30000	12	33	1	Married	9.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN

```
[18] #menyimpan data cleansed
     merge_data.to_csv('cleaned_data.csv', index=False)
```

```
[19] cleaned_data = pd.read_csv('cleaned_data.csv')
```

Result Here

Rakamin Academy

Model Machine Learning Regression (Time Series)

```
#melakukan agregasi data
regresi_data = merge_data.groupby(['Date']).agg({
    'Qty' :'sum'
}).reset_index()
```

```
[21] regresi_data
```

	Date	Qty
0	2022-01-01	49
1	2022-01-02	50
2	2022-01-03	76
3	2022-01-04	98
4	2022-01-05	67
...	...	...
360	2022-12-27	70
361	2022-12-28	68
362	2022-12-29	42
363	2022-12-30	44
364	2022-12-31	37

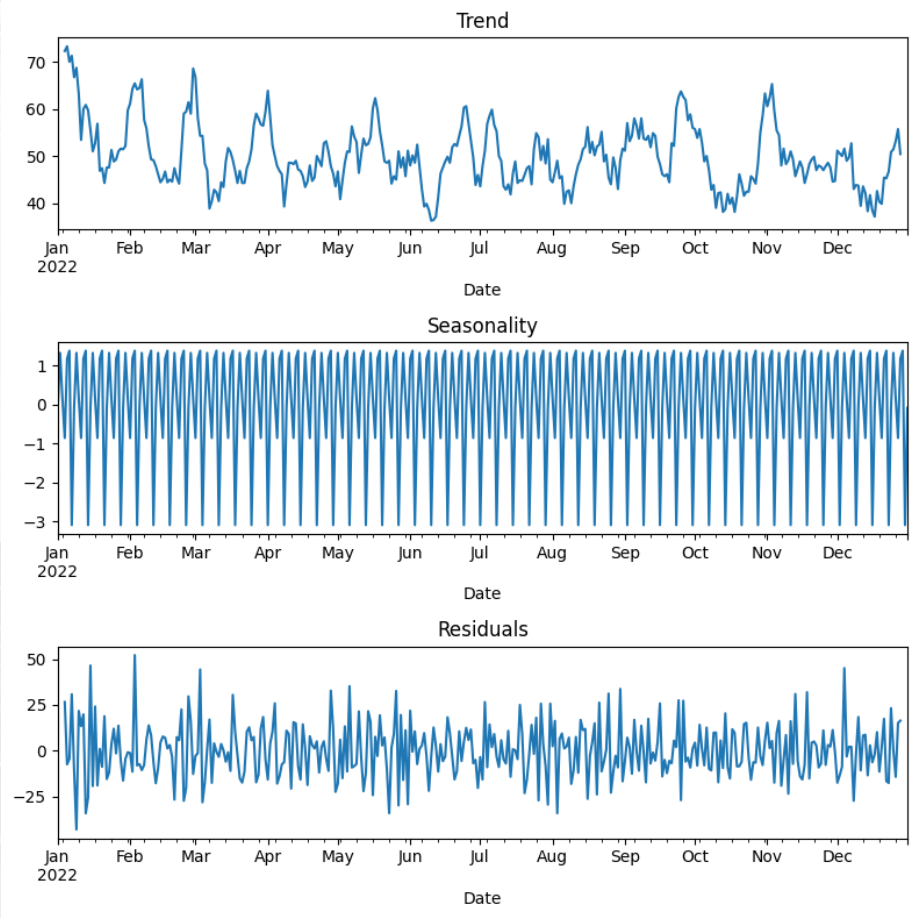
365 rows × 2 columns

Result Here

Rakamin Academy

```
[23] #digunakan untuk menghasilkan tiga subplot dalam satu gambar, masing-masing mewakili komponen utama dari analisis dekomposisi musiman pada data time series
     decomposed = seasonal_decompose(regresi_data.set_index('Date'))

     plt.figure(figsize = (8,8))

     plt.subplot(311)
     decomposed.trend.plot(ax = plt.gca())
     plt.title('Trend')
     plt.subplot(312)
     decomposed.seasonal.plot(ax = plt.gca())
     plt.title('Seasonality')
     plt.subplot(313)
     decomposed.resid.plot(ax = plt.gca())
     plt.title('Residuals')

     plt.tight_layout()
```

Result Here


Trend
70
60
50
40
Jan
2022
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Date
Seasonality
1
0
−1
−2
−3
Jan
2022
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Date
Residuals
50
25
0
−25
Jan
2022
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Date

Result Here

Rakamin Academy

```
[24] #untuk membagi data time series menjadi dua set: set pelatihan (train) dan set pengujian (test)
     cut_off = round(regresi_data.shape[0]*0.8)
     train_df = regresi_data[:cut_off]
     test_df = regresi_data[cut_off:].reset_index(drop=True)
     train_df.shape, test_df.shape

     ((292, 2), (73, 2))
```

[25] train_df

	Date	Qty
0	2022-01-01	49
1	2022-01-02	50
2	2022-01-03	76
3	2022-01-04	98
4	2022-01-05	67
...	...	...
287	2022-10-15	33
288	2022-10-16	26
289	2022-10-17	53
290	2022-10-18	47
291	2022-10-19	34

292 rows × 2 columns

[26] test_df

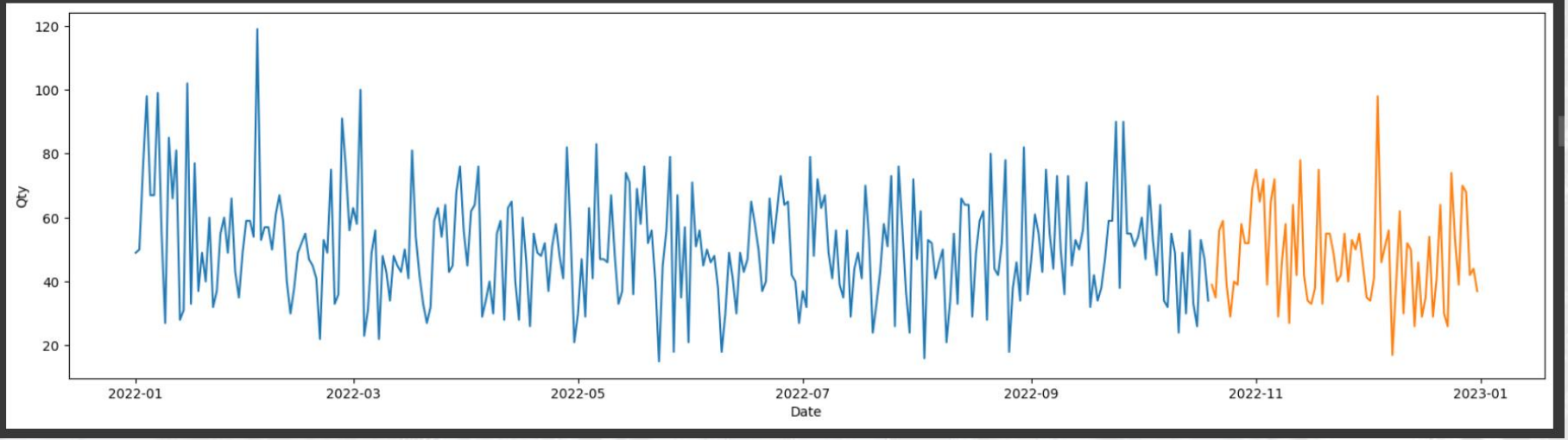
	Date	Qty
0	2022-10-20	39
1	2022-10-21	35
2	2022-10-22	56
3	2022-10-23	59
4	2022-10-24	39
...	...	...
68	2022-12-27	70
69	2022-12-28	68
70	2022-12-29	42
71	2022-12-30	44
72	2022-12-31	37

73 rows × 2 columns

Result Here

Rakamin Academy

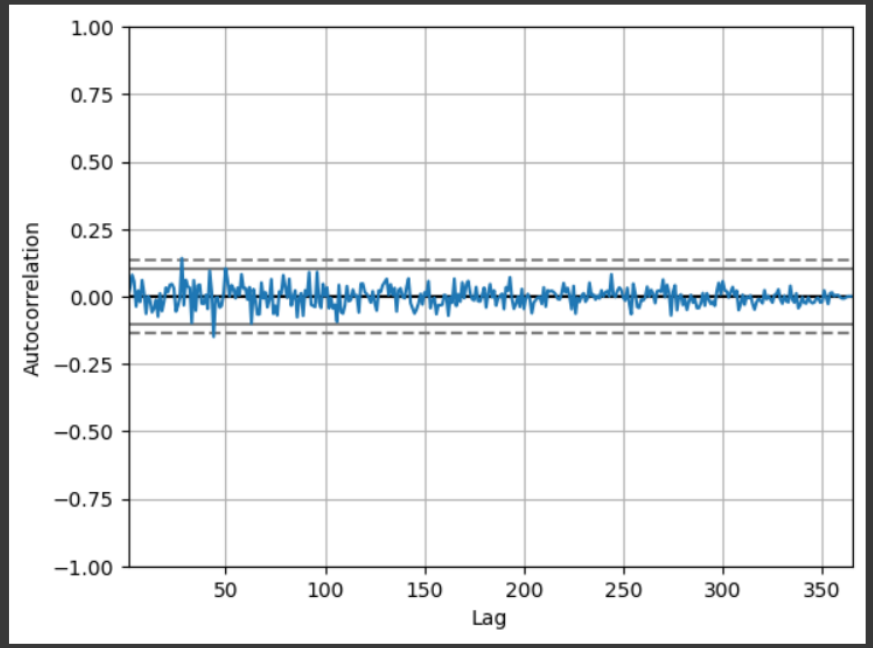
```
[27] #membuat plot garis menggunakan library seaborn untuk memvisualisasikan data dalam set pelatihan (train) dan set pengujian (test) dari time series.
     plt.figure(figsize=(20,5))
     sns.lineplot(data=train_df, x=train_df['Date'], y=train_df['Qty']);
     sns.lineplot(data=test_df, x=test_df['Date'], y=test_df['Qty']);
```



Result Here

Rakamin Academy

```
#mengilustrasikan seberapa berkorelasinya suatu variabel dengan dirinya sendiri pada pergeseran waktu yang berbeda (lag).
autocorrelation_plot(regresi_data['Qty']);
```



Result Here

Rakamin Academy

```
[28] #untuk mengambil baris-baris dari dataframe customer_data di mana kolom 'Marital Status' memiliki nilai yang kosong
     customer_data[customer_data['Marital Status'].isnull()]
```

	CustomerID	Age	Gender	Marital Status	Income	Longitude
9	10	34	1	NaN	4.0	NaN
415	416	27	1	NaN	3.0	NaN
442	443	33	1	NaN	9.0	NaN

```
[29] #mengambil baris-baris dari dataframe merge_data di mana kolom 'Marital Status' memiliki nilai yang kosong
     merge_data[merge_data['Marital Status'].isnull()]
```

	TransactionID	CustomerID	Date	ProductID	Price	Qty	TotalAmount	StoreID	Age	Gender	Marital Status	Income	Longitude_x	Product Name	StoreName	GroupStore	Type	Latitude	Longi
143	TR65931	443	2022-05-19	P1	8800	5	44000	12	33	1	NaN	9.0	NaN	Choco Bar	Prestasi Utama	Prestasi	General Trade	-2.0	
175	TR92340	416	2022-03-14	P7	9400	6	56400	12	27	1	NaN	3.0	NaN	Coffee Candy	Prestasi Utama	Prestasi	General Trade	-2.0	
295	TR28303	10	2022-12-30	P10	15000	4	60000	12	34	1	NaN	4.0	NaN	Cheese Stick	Prestasi Utama	Prestasi	General Trade	-2.0	
387	TR84178	10	2022-04-25	P3	7500	1	7500	13	34	1	NaN	4.0	NaN	Crackers	Buana	Buana	General Trade	-1.0	
458	TR78858	443	2022-09-22	P5	4200	5	21000	13	33	1	NaN	9.0	NaN	Thai Tea	Buana	Buana	General Trade	-1.0	

Result Here

Rakamin Academy

```
[30] #menghitung dan mengevaluasi kinerja model prediksi menggunakan metrik MAE dan RMSE dengan menggunakan contoh data aktual dan prediksi.
     def rmse(y_actual, y_pred):
         mse = mean_squared_error(y_actual, y_pred)
         rmse_value = mse ** 0.5
         return rmse_value


     def eval(y_actual, y_pred):
         mae_value = mean_absolute_error(y_actual, y_pred)
         rmse_value = rmse(y_actual, y_pred)
         print(f'Nilai MAE: {mae_value}')
         print(f'Nilai RMSE: {rmse_value}')

     y_actual = [3, 5, 7, 9]
     y_pred = [2.8, 5.2, 6.6, 9.3]

     eval(y_actual, y_pred)
```

```
Nilai MAE: 0.27500000000000036
Nilai RMSE: 0.2872281323269018
```

Result Here

Rakamin Academy

Metode time series ARIMA

```
[32] #mencetak informasi tentang kolom-kolom dan beberapa baris pertama dari dataframe train_df dan test_df
     print(train_df.columns)
     print(test_df.columns)

     print(train_df.head())
     print(test_df.head())
```

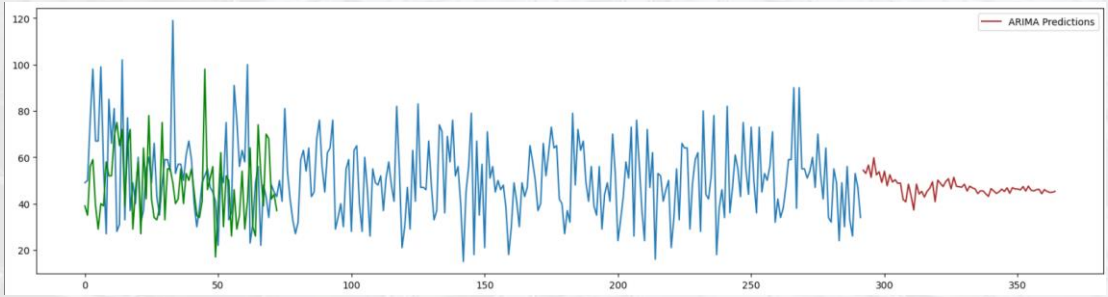
```
Index(['Date', 'Qty'], dtype='object')
Index(['Date', 'Qty'], dtype='object')
        Date  Qty
0 2022-01-01   49
1 2022-01-02   50
2 2022-01-03   76
3 2022-01-04   98
4 2022-01-05   67
        Date  Qty
0 2022-10-20   39
1 2022-10-21   35
2 2022-10-22   56
3 2022-10-23   59
4 2022-10-24   39
```

Result Here

Rakamin Academy

```
[33] #membangun model ARIMA, membuat prediksi, mengevaluasi performa prediksi, dan membuat plot visualisasi dari hasil prediksi

     y = train_df['Qty']

     ARIMAmodel = ARIMA(y, order=(40, 2, 1))
     ARIMAmodel_fit = ARIMAmodel.fit()

     y_pred = ARIMAmodel_fit.get_forecast(steps=len(test_df))

     y_pred_df = y_pred.conf_int()
     y_pred_df['predictions'] = ARIMAmodel_fit.predict(start=y_pred_df.index[0], end=y_pred_df.index[-1])
     y_pred_out = y_pred_df['predictions']
     eval(test_df['Qty'], y_pred_out)

     plt.figure(figsize=(20, 5))
     plt.plot(train_df['Qty'])
     plt.plot(test_df['Qty'], color='green')
     plt.plot(y_pred_out, color='brown', label='ARIMA Predictions')
     plt.legend()
     plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/statespace/sarimax.py:966: UserWarning: Non-stationary starting autoregressive parame
  warn('Non-stationary starting autoregressive parameters'
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/statespace/sarimax.py:978: UserWarning: Non-invertible starting MA parameters found.
  warn('Non-invertible starting MA parameters found.'
/usr/local/lib/python3.10/dist-packages/statsmodels/base/model.py:607: ConvergenceWarning: Maximum Likelihood optimization failed to converge
  warnings.warn("Maximum Likelihood optimization failed to "
Nilai MAE: 12.357079825645519
Nilai RMSE: 15.472386623448042
```



2. Case Study

Rakamin Academy

Machine Learning Clustering

- Tujuan dari pembuatan model machine learning ini adalah untuk dapat membuat cluster customer-customer yang mirip
- Data cleansing terlebih dahulu, merubah tipe data supaya sesuai
- Data merge untuk menggabungkan semua data
- Membuat data baru untuk clustering, yaitu groupby by customerID lalu yang di aggregasi adalah :
  - Transaction id count
  - Qty sum
  - Total amount sum
- Menggunakan metode clustering KMeans
- Untuk proses queri data, dapat memilih alternatif pemrograman yang sesuai preferensi kamu.

Result Here

Rakamin Academy

-- Model Machine Learning Clustering --

```
[34] #menampilkan lima baris pertama dari dataframe merge_data
     merge_data.head()
```

	TransactionID	CustomerID	Date	ProductID	Price	Qty	TotalAmount	StoreID	Age	Gender	Marital Status	Income	Longitude_x	Product Name	StoreName	GroupStore	Type	Latitude	Longitude_y
0	TR11369	328	2022-01-01	P3	7500	4	30000	12	36	0	Married	10.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN
1	TR89318	183	2022-07-17	P3	7500	1	7500	12	27	1	Single	0.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN
2	TR9106	123	2022-09-26	P3	7500	4	30000	12	34	0	Married	4.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN
3	TR4331	335	2022-08-01	P3	7500	3	22500	12	29	1	Single	4.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN
4	TR6445	181	2022-10-01	P3	7500	4	30000	12	33	1	Married	9.0	NaN	Crackers	Prestasi Utama	Prestasi	General Trade	-2.0	NaN

Result Here

Rakamin Academy

```
[35] #menghitung dan menampilkan matriks korelasi antar kolom-kolom dalam dataframe
     merge_data.corr()
```

```
<ipython-input-35-f0b9e936349b>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will
  merge_data.corr()
```

	CustomerID	Price	Qty	TotalAmount	StoreID	Age	Gender	Income	Longitude_x	Latitude	Longitude_y
CustomerID	1.000000	-0.016423	-0.009755	-0.024915	0.004129	-0.025952	-0.009947	0.007537	NaN	0.001060	NaN
Price	-0.016423	1.000000	-0.353640	0.440632	-0.032863	0.014693	0.010705	0.001439	NaN	-0.028502	NaN
Qty	-0.009755	-0.353640	1.000000	0.621129	0.014365	-0.027768	-0.010542	-0.029113	NaN	-0.002765	NaN
TotalAmount	-0.024915	0.440632	0.621129	1.000000	-0.010722	-0.016900	-0.008774	-0.025680	NaN	-0.028421	NaN
StoreID	0.004129	-0.032863	0.014365	-0.010722	1.000000	-0.003872	-0.000189	0.001189	NaN	0.488507	NaN
Age	-0.025952	0.014693	-0.027768	-0.016900	-0.003872	1.000000	-0.033183	0.485788	NaN	0.009051	NaN
Gender	-0.009947	0.010705	-0.010542	-0.008774	-0.000189	-0.033183	1.000000	-0.072268	NaN	-0.005635	NaN
Income	0.007537	0.001439	-0.029113	-0.025680	0.001189	0.485788	-0.072268	1.000000	NaN	0.014446	NaN
Longitude_x	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Latitude	0.001060	-0.028502	-0.002765	-0.028421	0.488507	0.009051	-0.005635	0.014446	NaN	1.000000	NaN
Longitude_y	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Result Here

Rakamin Academy

```
[37] #membuat fitur untuk clustering
     cluster_df = merge_data.groupby('CustomerID').agg({
         'TransactionID': 'count',
         'Qty' : 'sum',
         'TotalAmount' : 'sum'
     }).reset_index()
```

```
[38] cluster_df.head()
```

	CustomerID	TransactionID	Qty	TotalAmount
0	1	17	60	623300
1	2	13	57	392300
2	3	15	56	446200
3	4	10	46	302500
4	5	7	27	268600

```
[39] cluster_df
```

	CustomerID	TransactionID	Qty	TotalAmount
0	1	17	60	623300
1	2	13	57	392300
2	3	15	56	446200
3	4	10	46	302500
4	5	7	27	268600
...	...	...	...	...
442	443	16	59	485100
443	444	18	62	577700
444	445	18	68	587200
445	446	11	42	423300
446	447	13	42	439300

447 rows × 4 columns

Result Here

Rakamin Academy

```
[40] #pra-pemrosesan data sebelum dilakukan analisis klasterisasi (clustering)
     cluster_data = cluster_df.drop(columns = ['CustomerID'])

     cluster_data_normalize = preprocessing.normalize(cluster_data)
```

```
[41] #menampilkan data yang sudah dinormalisasi.
     cluster_data_normalize
```

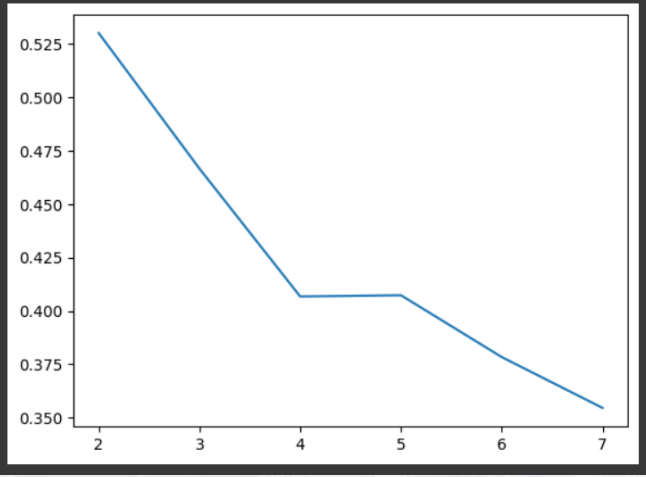
```
array([[2.72741856e-05, 9.62618317e-05, 9.99999995e-01],
       [3.31379043e-05, 1.45296965e-04, 9.99999989e-01],
       [3.36172117e-05, 1.25504257e-04, 9.99999992e-01],
       ...,
       [3.06539507e-05, 1.15803814e-04, 9.99999993e-01],
       [2.59862980e-05, 9.92204105e-05, 9.99999995e-01],
       [2.95925334e-05, 9.56066465e-05, 9.99999995e-01]])
```

Result Here

Rakamin Academy

```
[42] #algoritma K-Means untuk melakukan analisis klasterisasi pada data yang telah dinormalisasi.
     K = range(2,8)
     fits = []
     score = []

     for k in K:
       model = KMeans(n_clusters = k, random_state = 0, n_init = 'auto').fit(cluster_data_normalize)
       fits.append(model)
       score.append(silhouette_score(cluster_data_normalize, model.labels_, metric = 'euclidean'))
```

```
[43] #memvisualisasikan bagaimana nilai silhouette score berubah seiring dengan jumlah klaster yang berbeda
     sns.lineplot(x=K, y=score);
```



Result Here

Rakamin Academy

```
[44] fits[1]
```

KMeans

```
KMeans(n_clusters=3, n_init='auto', random_state=0)
```

```
[45] #menambahkan kolom 'cluster_label' ke dalam dataframe cluster_df yang berisi label klaster hasil dari analisis K-Means
     cluster_df['cluster_label'] = fits[2].labels_
```

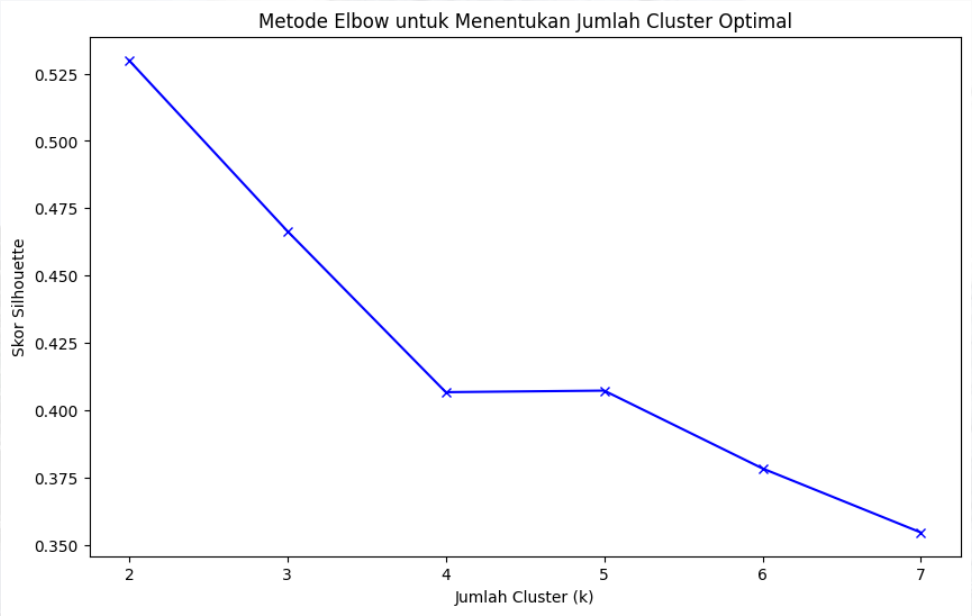
```
[47] #melakukan agregasi statistik pada dataframe cluster_df
     cluster_df.groupby('cluster_label').agg({
         'CustomerID': 'count',
         'TransactionID' : 'mean',
         'Qty' : 'mean'
     })
```

cluster_label	CustomerID	TransactionID	Qty
0	128	11.601562	40.007812
1	28	9.250000	35.142857
2	156	11.628205	42.775641
3	135	10.829630	40.874074

Result Here

Rakamin Academy

```
[48] # Memvisualisasikan hasil siluet untuk menentukan jumlah cluster yang optimal
     plt.figure(figsize=(10, 6))
     plt.plot(K, score, 'bx-')
     plt.xlabel('Jumlah Cluster (k)')
     plt.ylabel('Skor Silhouette')
     plt.title('Metode Elbow untuk Menentukan Jumlah Cluster Optimal')
     plt.show()
```



Result Here

Rakamin Academy

```
[49] # Pilih jumlah cluster berdasarkan hasil visualisasi siluet
     selected_k = 3
     selected_model = fits[selected_k - 2]  # Indeks array dimulai dari 0
```

```
[50] # Tambahkan informasi klaster ke dalam DataFrame cluster_df
     cluster_df['cluster_label'] = selected_model.labels_
```

```
[52] #analisis klasterisasi
     cluster_analysis = cluster_df.groupby('cluster_label').agg({
         'CustomerID': 'count',
         'TransactionID': 'mean',
         'Qty': 'mean',
         'TotalAmount': 'mean'
     })
```

Result Here

Rakamin Academy

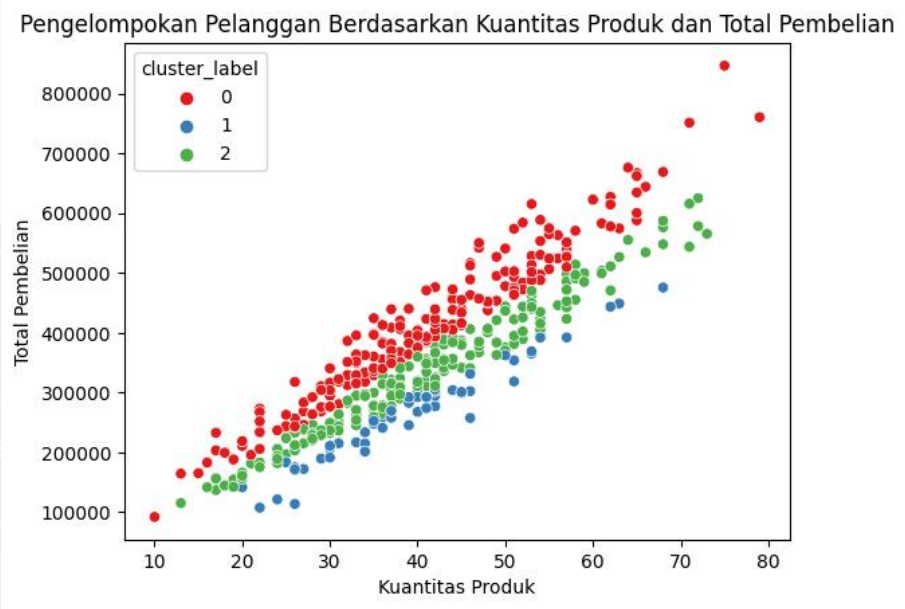
```
[53] # Memberikan rekomendasi berdasarkan analisis klaster
     for cluster_label, data in cluster_analysis.iterrows():
         print(f"Cluster {cluster_label}:")
         print(f"Jumlah Pelanggan: {data['CustomerID']}")
         print(f"Rata-rata Transaksi per Pelanggan: {data['TransactionID']:.2f}")
         print(f"Rata-rata Kuantitas Produk per Pelanggan: {data['Qty']:.2f}")
         print(f"Rata-rata Total Pembelian per Pelanggan: {data['TotalAmount']:.2f}")
         print("\n")
```

```
Cluster 0:
Jumlah Pelanggan: 205.0
Rata-rata Transaksi per Pelanggan: 11.65
Rata-rata Kuantitas Produk per Pelanggan: 40.88
Rata-rata Total Pembelian per Pelanggan: 409866.83


Cluster 1:
Jumlah Pelanggan: 61.0
Rata-rata Transaksi per Pelanggan: 10.34
Rata-rata Kuantitas Produk per Pelanggan: 39.23
Rata-rata Total Pembelian per Pelanggan: 268652.46


Cluster 2:
Jumlah Pelanggan: 181.0
Rata-rata Transaksi per Pelanggan: 11.05
Rata-rata Kuantitas Produk per Pelanggan: 41.56
Rata-rata Total Pembelian per Pelanggan: 340511.05
```

Result Here

Rakamin Academy

```
[54] # Visualisasi klaster dengan scatter plot
     sns.scatterplot(data=cluster_df, x='Qty', y='TotalAmount', hue='cluster_label', palette='Set1')
     plt.xlabel('Kuantitas Produk')
     plt.ylabel('Total Pembelian')
     plt.title('Pengelompokan Pelanggan Berdasarkan Kuantitas Produk dan Total Pembelian')
     plt.show()
```



Case Study

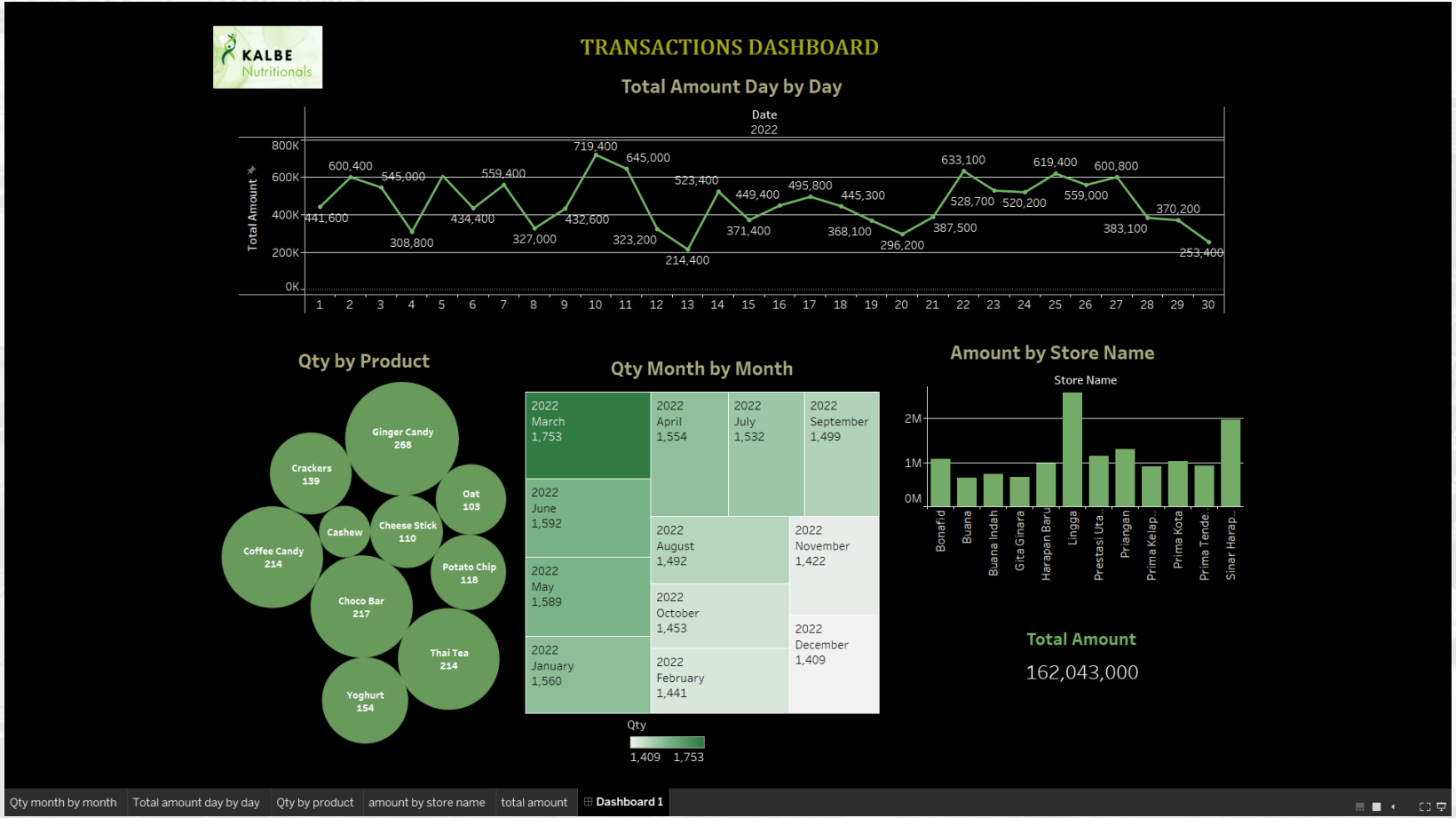
Peserta dapat membuat dashboard di tableau

Sebelum membuat dashboard terlebih dahulu membuat worksheet sebanyak 4.

- Worksheet 1 Jumlah qty dari bulan ke bulan
- Worksheet 2 Jumlah total amount dari hari ke hari
- Worksheet 3 Jumlah penjualan (qty) by product
- Worksheet 4 Jumlah penjualan (total amount) by store name

- Setelah itu bisa membuat dashboard dengan menggabungkan 4 worksheet.

Result Here


KALBE Nutritionals
TRANSACTIONS DASHBOARD
Total Amount Day by Day
Date
2022
Total Amount
800K
600K
400K
200K
0K
441,600
600,400
545,000
308,800
434,400
559,400
327,000
432,600
719,400
323,200
645,000
214,400
523,400
371,400
449,400
495,800
368,100
445,300
296,200
387,500
528,700
633,100
520,200
559,000
619,400
600,800
383,100
370,200
253,400
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
Qty by Product
Ginger Candy 268
Crackers 139
Oat 103
Cashew
Cheese Stick 110
Coffee Candy 214
Potato Chip 118
Choco Bar 217
Thai Tea 214
Yoghurt 154
Qty Month by Month
2022 March 1,753
2022 April 1,554
2022 July 1,532
2022 September 1,499
2022 June 1,592
2022 August 1,492
2022 November 1,422
2022 May 1,589
2022 October 1,453
2022 December 1,409
2022 January 1,560
2022 February 1,441
Qty
1,409 1,753
Amount by Store Name
Store Name
2M
1M
0M
Bonafid
Buana
Buana Indah
Gita Ginara
Harapan Baru
Lingga
Prestasi Uta..
Priangan
Prima Kelap..
Prima Kota
Prima Tende..
Sinar Harap..
Total Amount
162,043,000
Qty month by month
Total amount day by day
Qty by product
amount by store name
total amount
Dashboard 1

Link Tableau

https://public.tableau.com/views/TRANSACTIONSDASHBOARD_KARIN/Dashboard1?:language=en-US&:display_count=n&:origin=viz_share_link

Link Github

https://github.com/KarinaKhPutri/FinalTask_Kalbe_DS_KarinaKhairunnisaPutri

Link Video Presentation

https://drive.google.com/drive/folders/1h0W44zwPoUjrexL-Vx-9_RghuaebhXm0?usp=sharing

Thank You