Rakamin Academy X KALBE Nutritionals

Customer Segmentation and Predictive Precision in Inventory using KMeans Clustering and Time-Series Forecasting

Kalbe Nutritionals Data Scientist Project Based Internship Program

Presented by
Favian Sulthan Wafi

Favian Sulthan Wafi

About Myself

I am a third-year undergraduate Statistics student at Universitas Indonesia. I am a hard worker, a fast learner, and a data-driven problem solver who is thrilled to learn new things. I am passionate about using data to uncover insights and solve real-world problems. With a solid foundation in statistical theory and methods, as well as experience with a variety of statistical software packages, I am eager to contribute my skills to real-world challenges.

Rakamin Academy

My Experience

- Himpunan Mahasiswa Departemen Matematika (HMDM) FMIPA UI / Staff of Evaluation Management, Research, and Analysis Bureau
- Lomba Kegiatan Matematika (LOGIKA) UI 2023 / Vice Head of Registration Division
- Delegation of Statistics Essay Competition (SEC) SATRIA DATA 2023. Wrote an essay about “The Impact of ChatGPT on Education in Indonesia: an Analysis using NLP and Sentiment Analysis on Twitter”

Case Study

- **Background Story**
  - Kamu adalah seorang Data Scientist di Kalbe Nutritionals dan sedang mendapatkan project baru dari tim inventory dan tim marketing.
  - Dari tim inventory, kamu diminta untuk dapat membantu memprediksi jumlah penjualan (quantity) dari total keseluruhan product Kalbe
    - Tujuan dari project ini adalah untuk mengetahui perkiraan quantity product yang terjual sehingga tim inventory dapat membuat stock persediaan harian yang cukup.
    - Prediksi yang dilakukan harus harian.
  - Dari tim marketing kamu diminta untuk membuat cluster/segment customer berdasarkan beberapa kriteria.
    - Tujuan dari project ini adalah untuk membuat segment customer.
    - Segment customer ini nantinya akan digunakan oleh tim marketing untuk memberikan personalized promotion dan sales treatment.
  - Tools yang akan kamu gunakan dalam project ini adalah
    - Python
    - Jupyter Notebook
    - Tableau
    - Dbeaver
    - PostgreSQL

Case Study

Rakamin Academy

- **Peserta dapat melakukan data ingestion ke dalam dbeaver**
  - Pilih connect to database di pojok kiri dan pilih postgreSQL
  - Test connection ke postgreSQL
  - Di menu drop down postgreSQL pilih table dan klik kanan (import data)
  - Pilih CSV file dan upload 4 file dari Kalbe https://drive.google.com/drive/folders/1_rQrauVW2OvLIe2zd54Vcwnr2EY-vnnR?usp=sharing
- **Peserta dapat melakukan exploratory data analysis di dbeaver**
  - query 1 : Berapa rata-rata umur customer jika dilihat dari marital statusnya ?
  - query 2 : Berapa rata-rata umur customer jika dilihat dari gender nya ?
  - query 3 : Tentukan nama store dengan total quantity terbanyak!
  - query 4 : Tentukan nama produk terlaris dengan total amount terbanyak!
- **Peserta dapat melakukan data ingestion ke dalam tableau public**
  - Pertama buka tableau public (https://public.tableau.com/app/discover) . Sign in jika sudah punya akun dan sign up jika belum punya akun tableau public.
  - Setelah itu klik create dan pilih web authoring
  - Setelah itu akan di redirect ke halaman baru dan bisa langsung upload 4 data dari Kalbe
- **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.
- **Peserta dapat membuat model prediktif menggunakan regresi dan membuat clustering**
  - Membaca data csv
  - Melakukan data cleansing
  - Menggabungkan semua data menjadi 1 data
  - Membuat model machine learning regression (time series)
  - Membuat model machine learning clustering

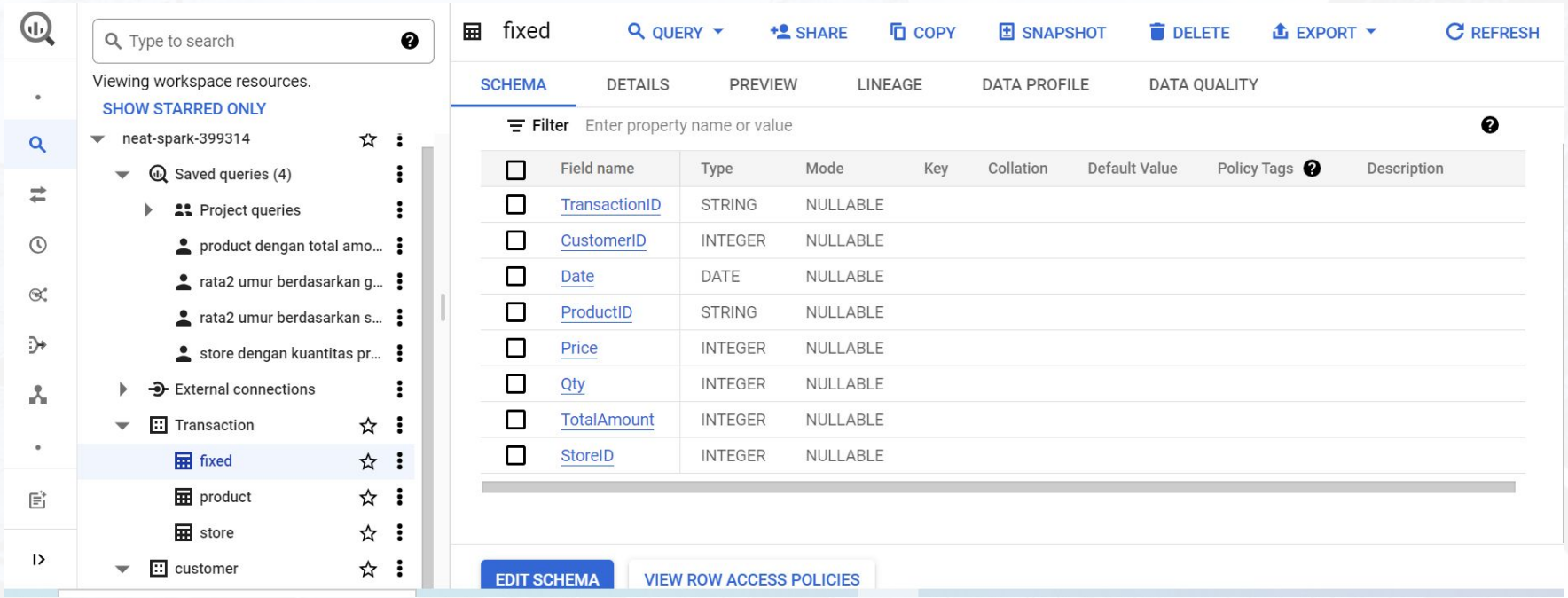
Rakamin Academy

Exploratory Data Analysis using BigQuery

Exploratory Data Analysis

Rakamin Academy

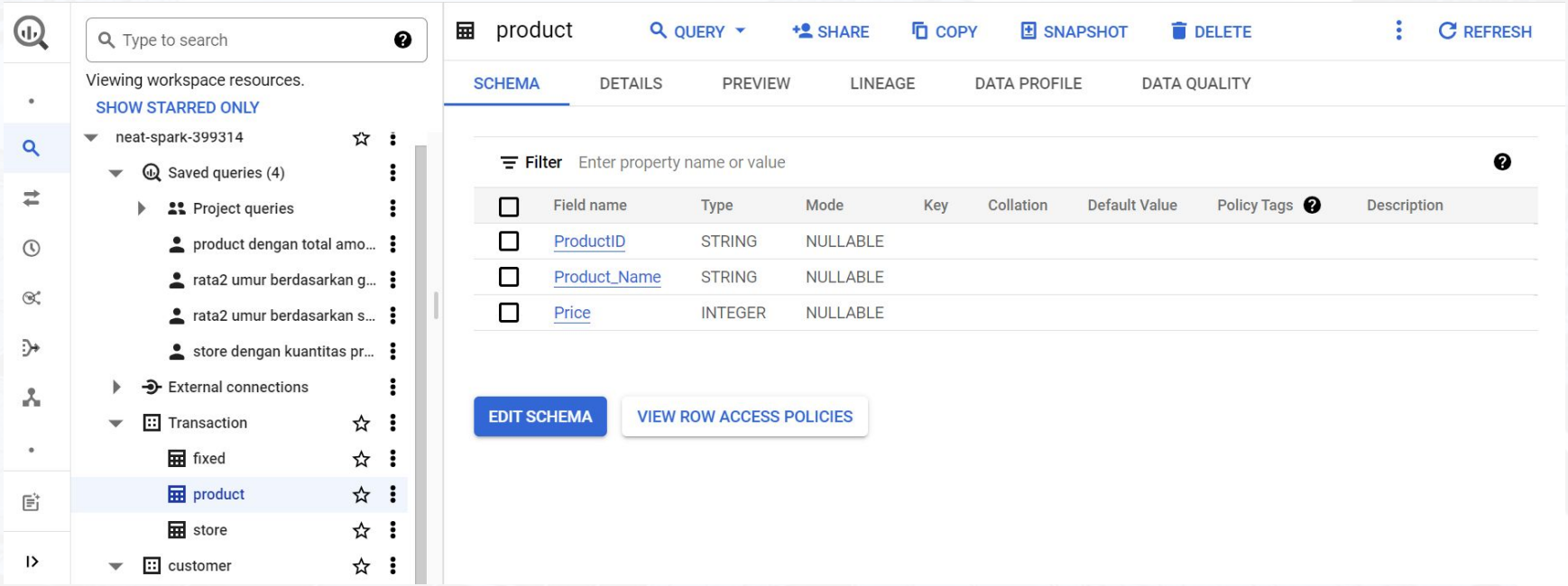
Dataset Transaction



Exploratory Data Analysis

Rakamin Academy

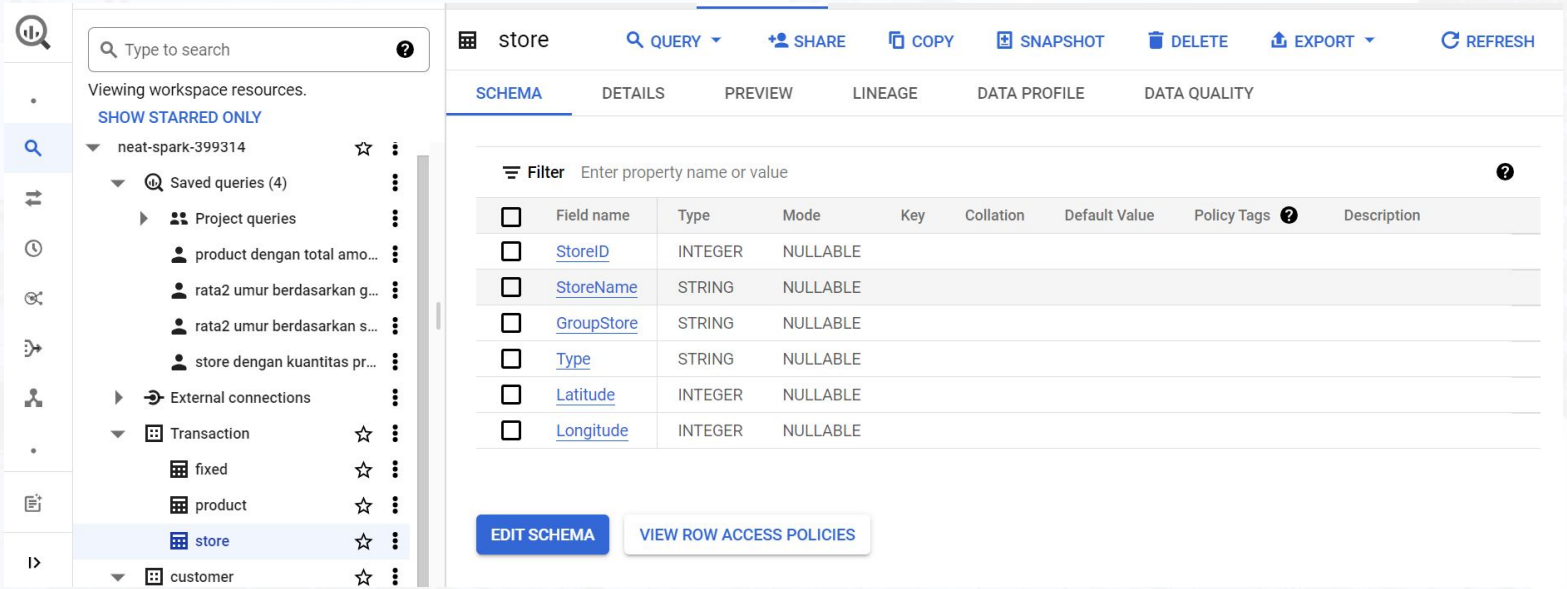
Dataset Product



Exploratory Data Analysis

Rakamin Academy

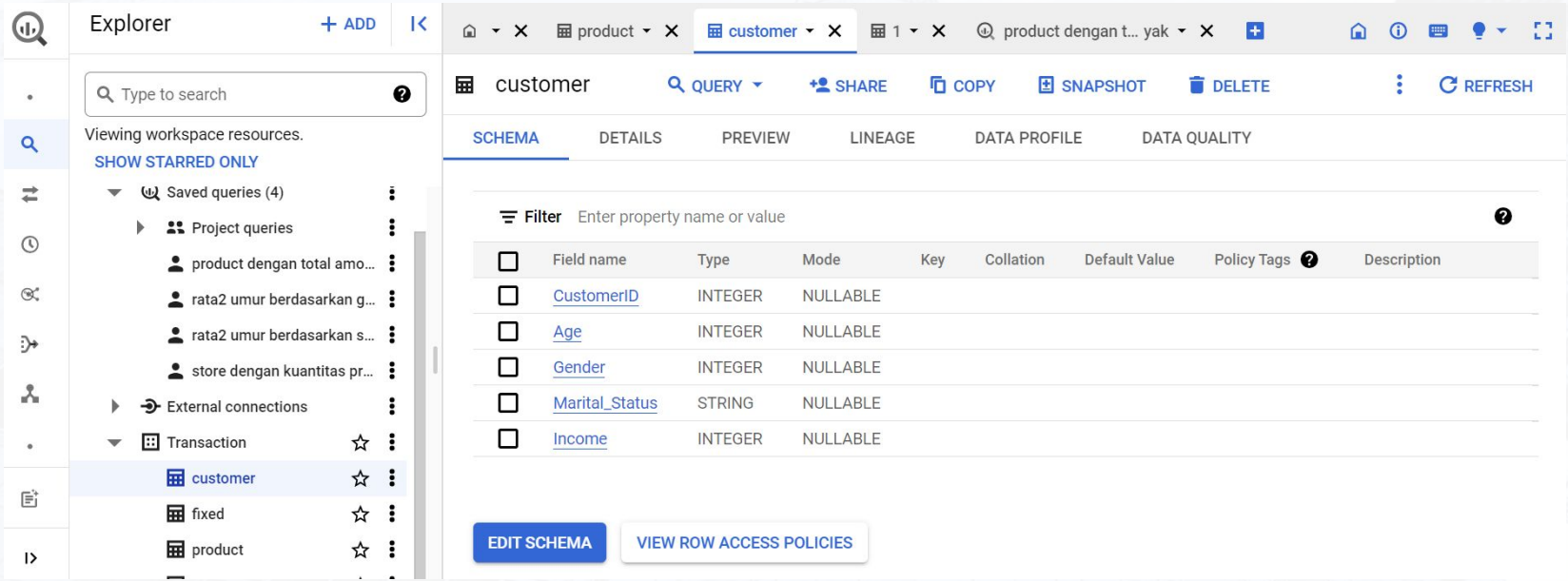
Dataset Store



Exploratory Data Analysis

Rakamin Academy

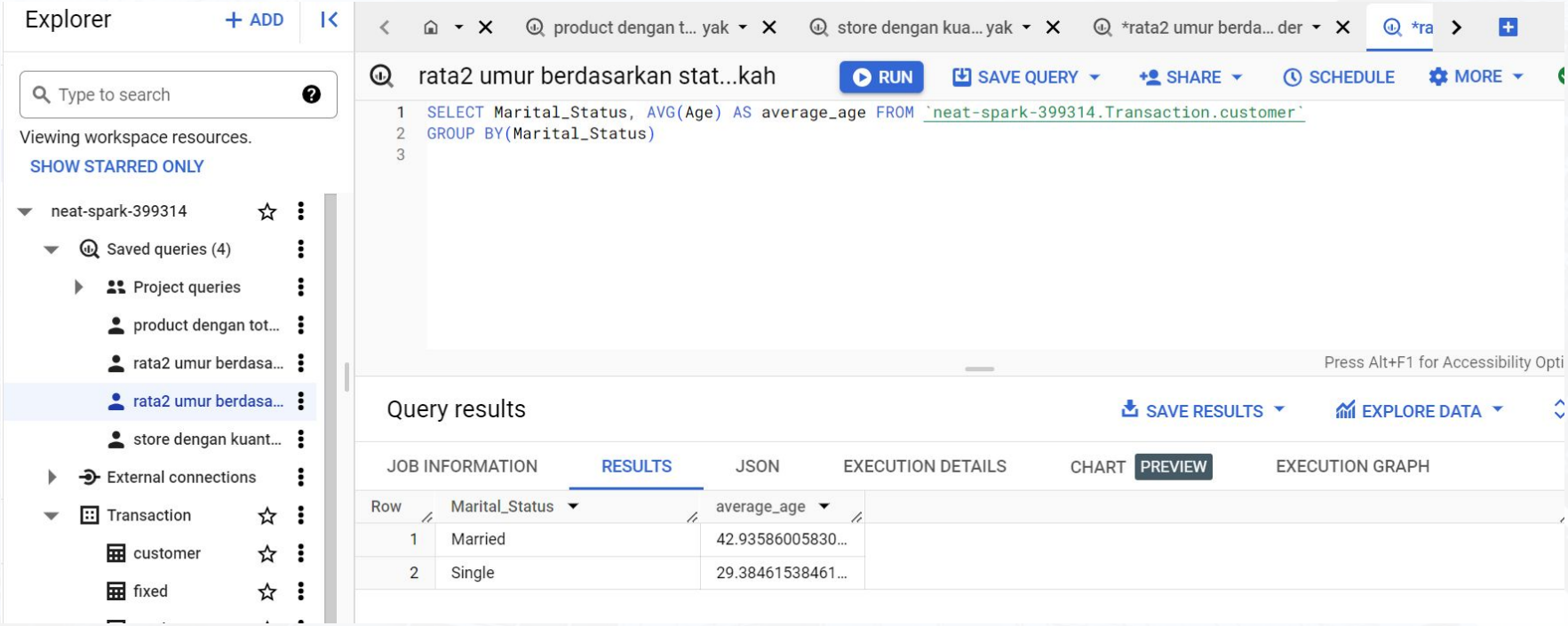
Dataset Customer



Exploratory Data Analysis

Rakamin Academy

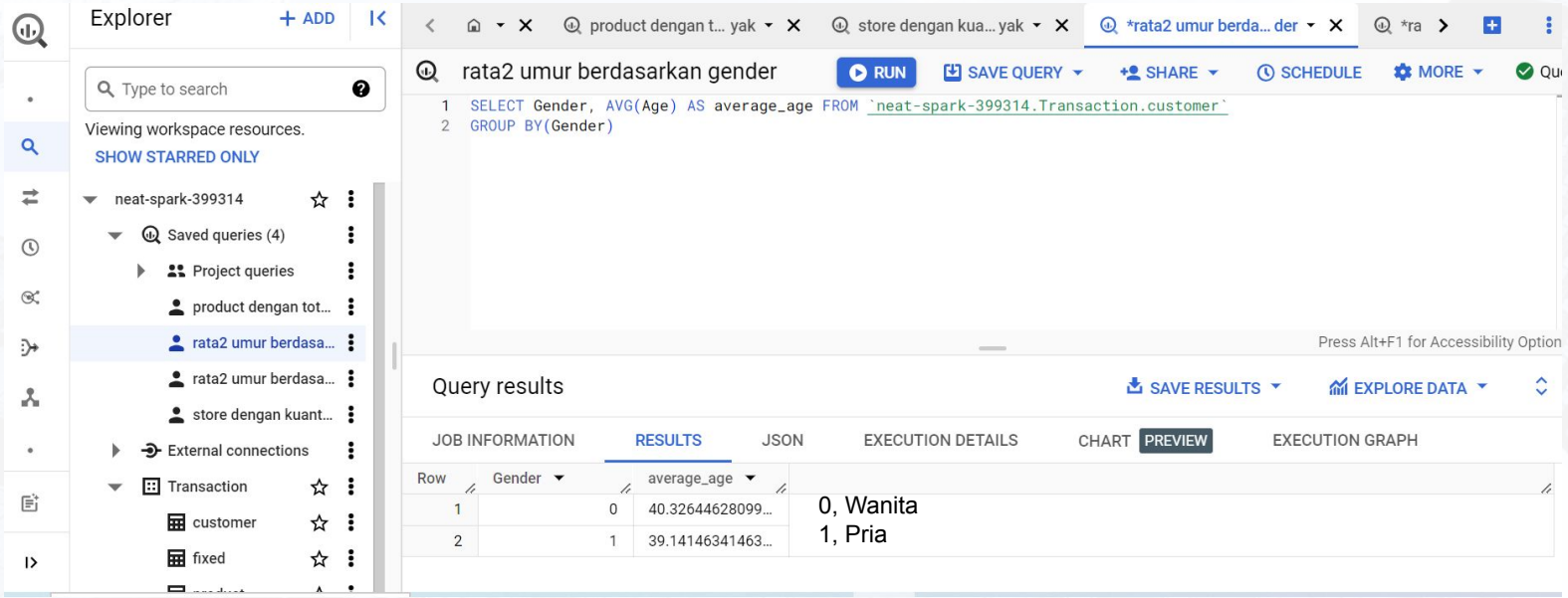
Rata-rata Umur berdasarkan Status Menikah



Exploratory Data Analysis

Rakamin Academy

Rata-rata Umur berdasarkan Jenis Kelamin



Exploratory Data Analysis

Rakamin Academy

Store dengan Kuantitas Produk Terjual Terbanyak

store dengan kuantitas prod...yak RUN

```sql
SELECT
  store.StoreName,
  store.StoreID,
  SUM(ts.Qty) AS total_quantity

FROM
  `neat-spark-399314.Transaction.fixed` AS ts
JOIN
  `neat-spark-399314.Transaction.store` AS store
ON
ts.StoreID = store.StoreID

GROUP BY
store.StoreName,
store.StoreID

ORDER BY
total_quantity DESC
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	StoreName	StoreID	total_quantity
1	Lingga	9	1439
2	Prestasi Utama	12	1395
3	Prima Kota	3	1358
4	Lingga	6	1338
5	Sinar Harapan	11	1331
6	Buana	13	1320
7	Prima Tendean	1	1310
8	Prima Kelapa Dua	2	1296

Exploratory Data Analysis

Rakamin Academy

Produk dengan Total Amount Terbanyak

product denga... RUN

```sql
SELECT
  pd.Product_Name,
  pd.ProductID,
  SUM(ts.TotalAmount) AS total_amount

FROM
  `neat-spark-399314.Transaction.fixed` AS ts
JOIN
  `neat-spark-399314.Transaction.product` AS pd
ON
ts.ProductID = pd.ProductID
GROUP BY
pd.Product_Name,
pd.ProductID
ORDER BY
total_amount DESC
```

Query results SAVE

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART | PREVI

Row	Product_Name	ProductID	total_amount
1	Cheese Stick	P10	27615000
2	Choco Bar	P1	21190400
3	Coffee Candy	P7	19711800
4	Yoghurt	P9	19630000
5	Oat	P8	15440000
6	Crackers	P3	13680000
7	Potato Chip	P4	13104000
8	Thai Tea	P5	11982600
9	Cashew	P6	11286000
10	Ginger Candy	P2	8403200

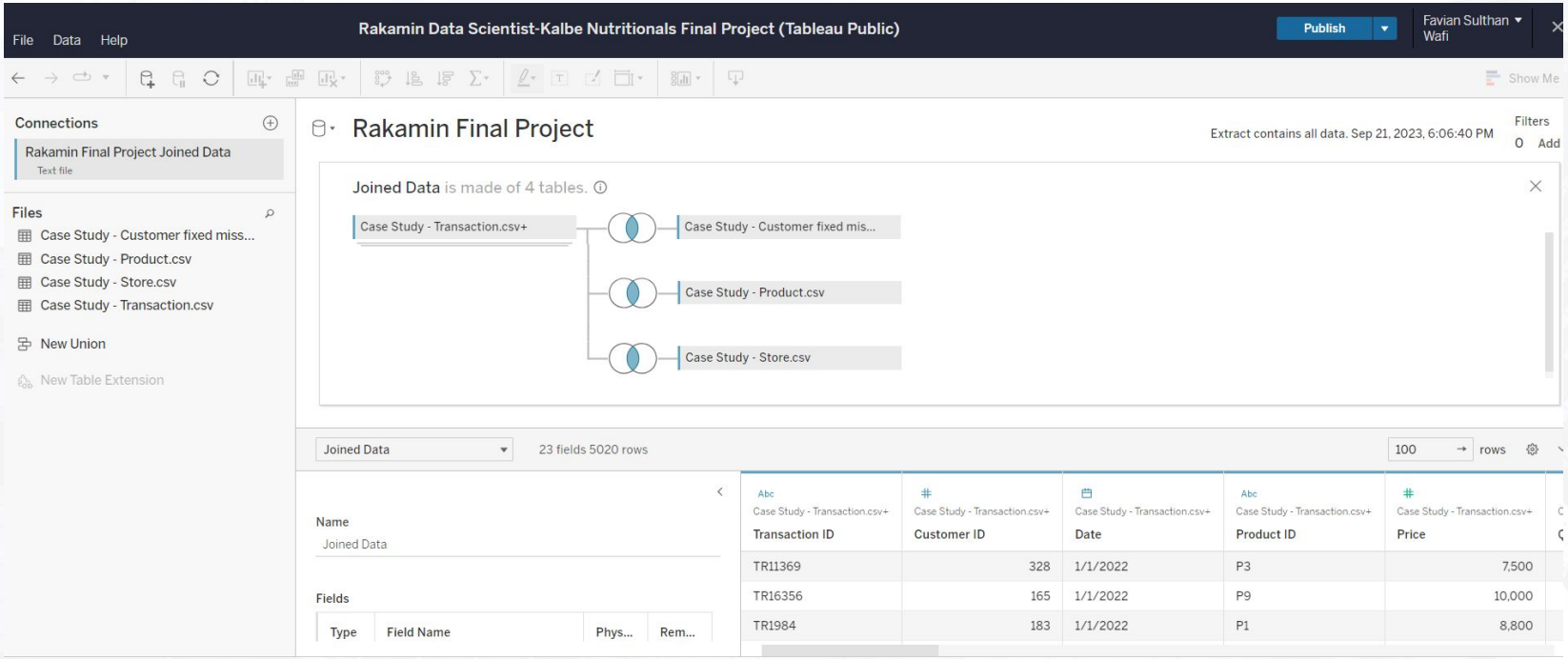
Rakamin Academy

Data Visualization using Tableau

Data Visualization

Rakamin Academy

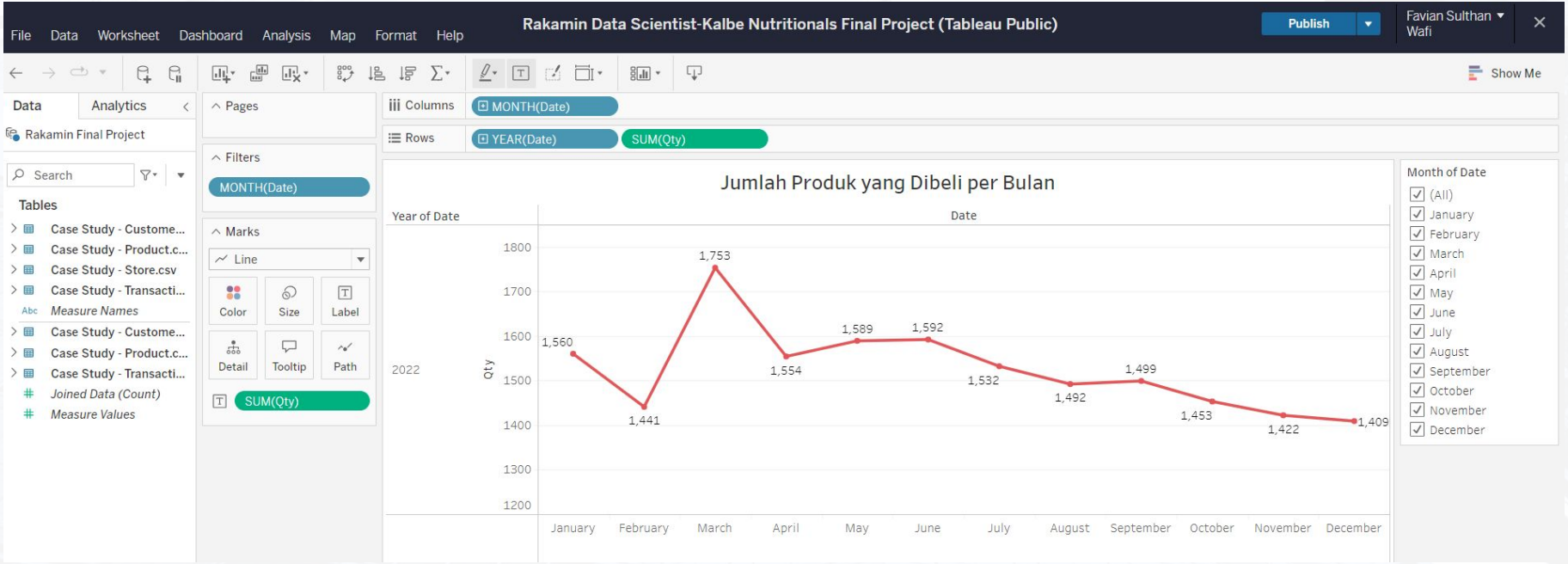
Inserting Dataset



Data Visualization

Rakamin Academy

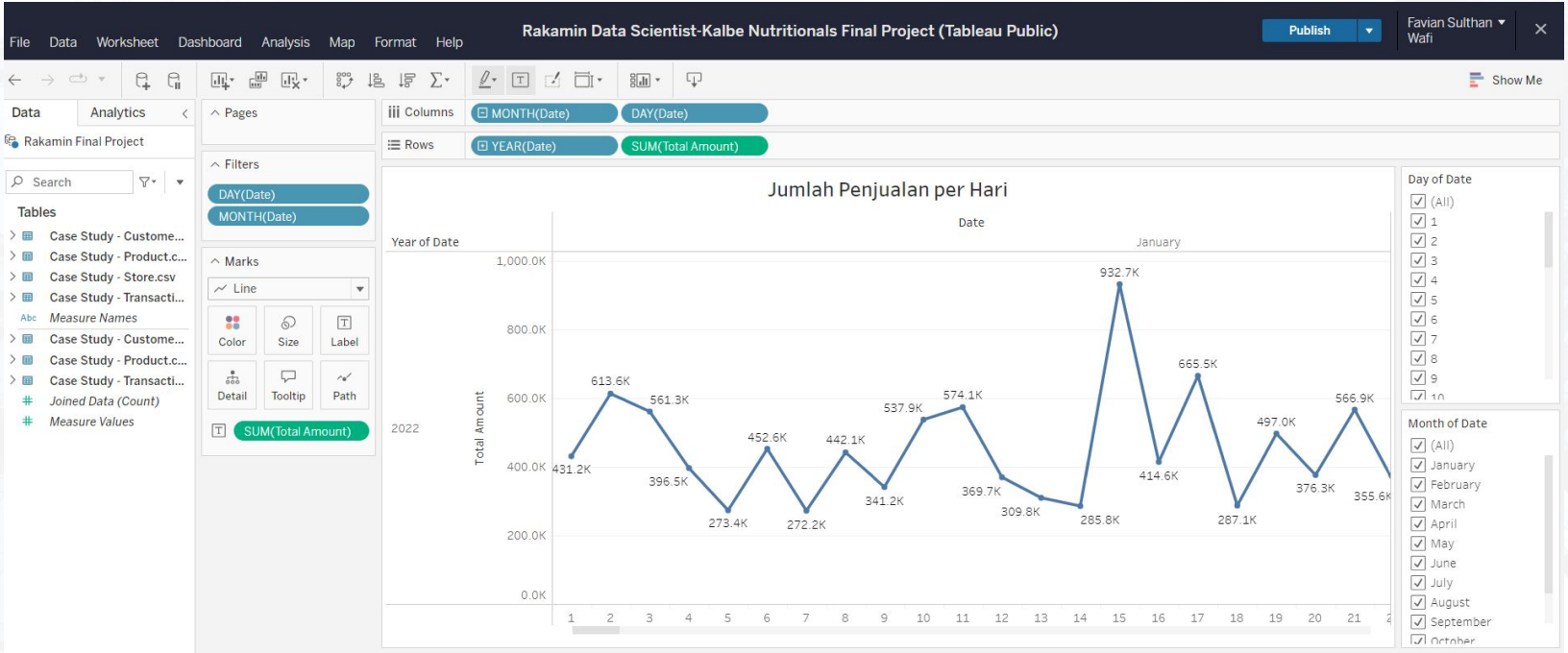
Jumlah Produk yang Dibeli per Bulan



Data Visualization

Rakamin Academy

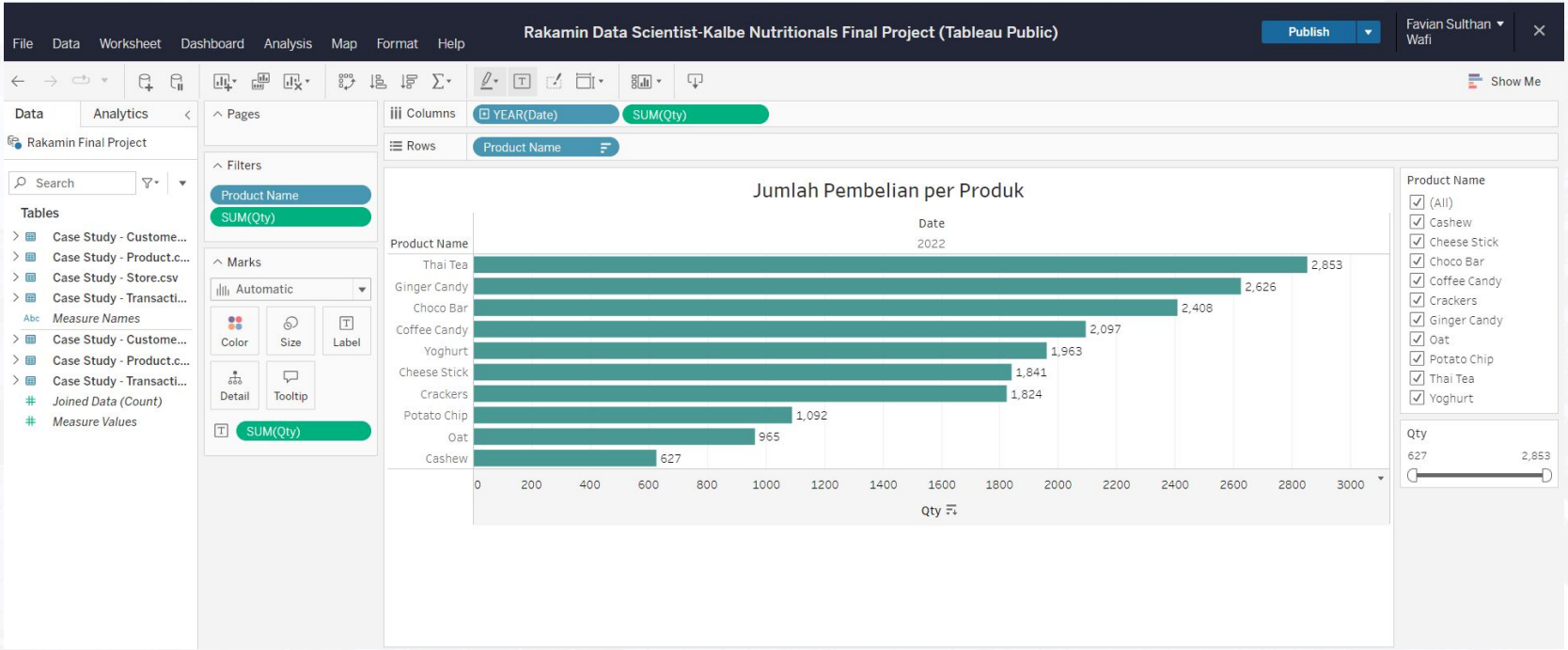
Jumlah Penjualan per Hari



Data Visualization

Rakamin Academy

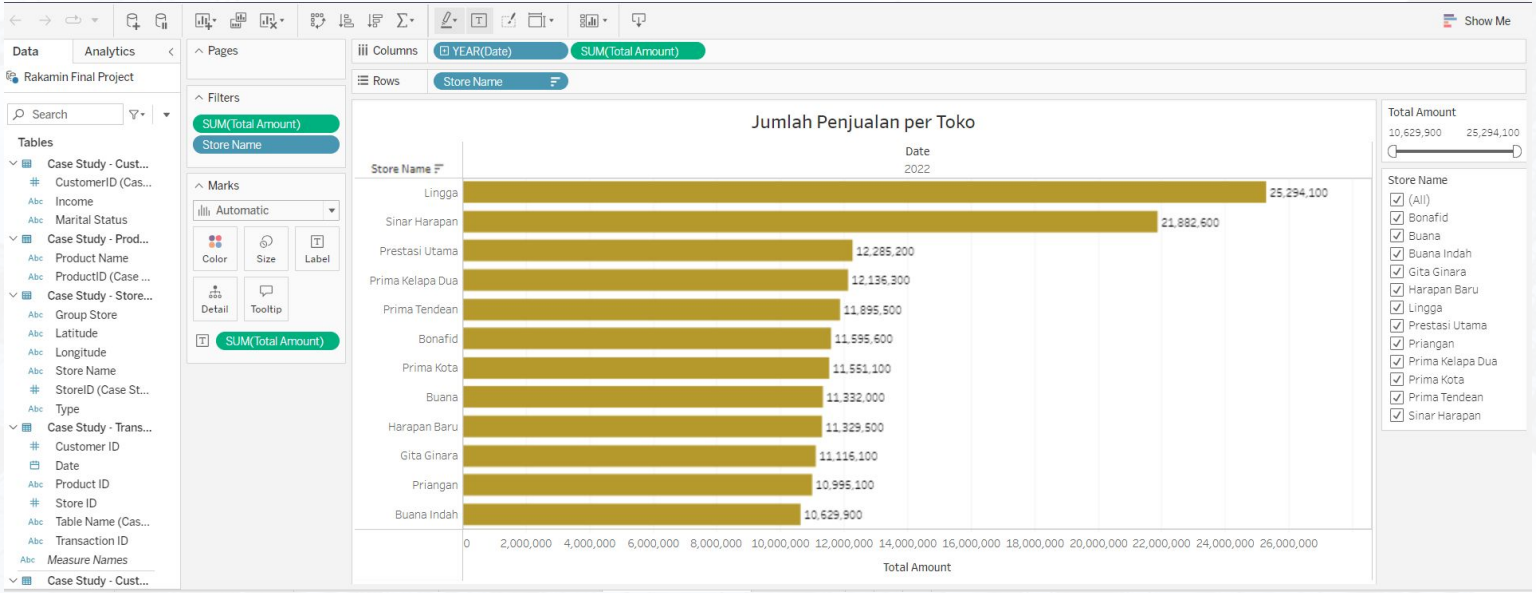
Jumlah Pembelian per Produk



Data Visualization

Rakamin Academy

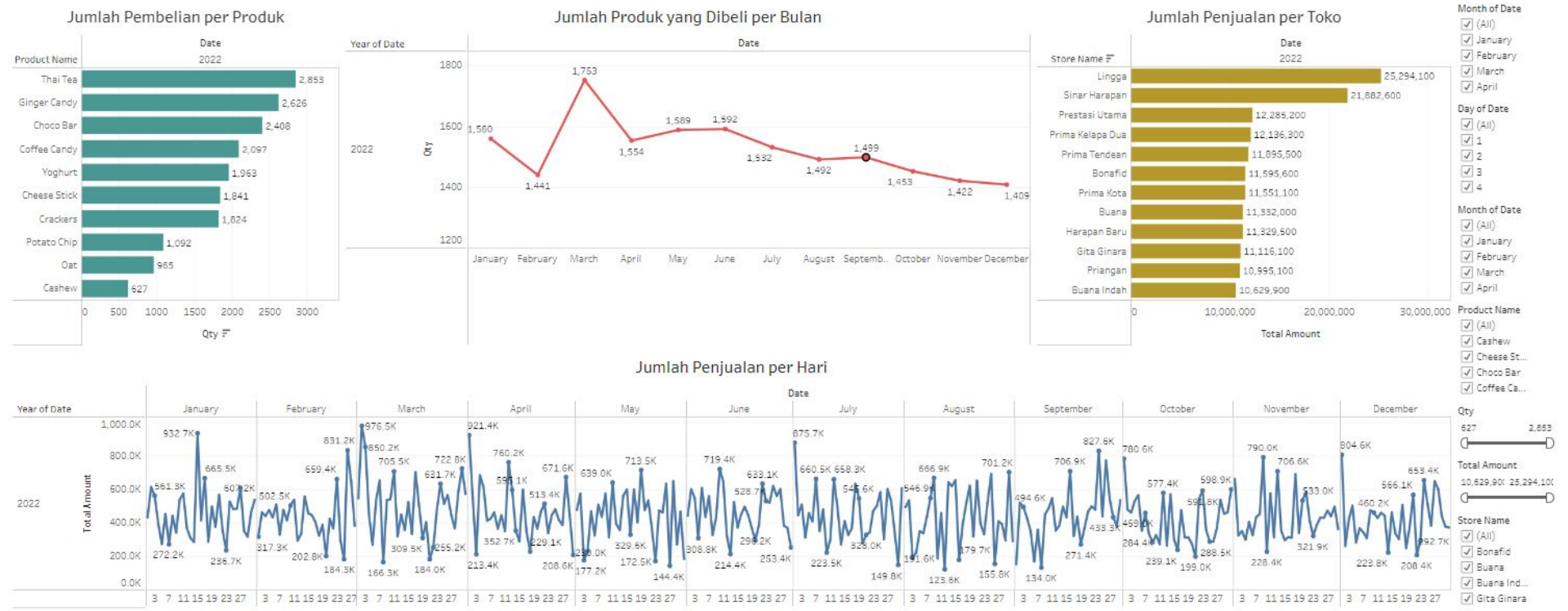
Jumlah Penjualan per Toko



Data Visualization

Rakamin Academy

Dashboard



Rakamin Academy

Data Forecasting and Clustering using Google Colab

Data Preprocessing

Rakamin Academy

Import Modules & Dataset

```
#mengimport modul yang digunakan
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import missingno as msno
```

```
[ ] #mengimport dataset yang digunakan
transaction = pd.read_csv('Case Study - Transaction.csv',sep=";")
store = pd.read_csv('Case Study - Store.csv',sep=";")
customer = pd.read_csv('Case Study - Customer.csv',sep=";")
product = pd.read_csv('Case Study - Product.csv',sep=";")
```

Data Preprocessing

Rakamin Academy

Transaction Data Preprocessing

```
#melihat dataset
transaction
```

	TransactionID	CustomerID	Date	ProductID	Price	Qty	TotalAmount	StoreID
0	TR11369	328	01/01/2022	P3	7500	4	30000	12
1	TR16356	165	01/01/2022	P9	10000	7	70000	1
2	TR1984	183	01/01/2022	P1	8800	4	35200	4
3	TR35256	160	01/01/2022	P1	8800	7	61600	4
4	TR41231	386	01/01/2022	P9	10000	1	10000	4
...	...	...	...	...	...	...	...	...
5015	TR54423	243	31/12/2022	P10	15000	5	75000	3

```
#melihat informasi yang terdapat pada dataset
transaction.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5020 entries, 0 to 5019
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   TransactionID   5020 non-null   object
 1   CustomerID      5020 non-null   int64
 2   Date            5020 non-null   object
 3   ProductID       5020 non-null   object
 4   Price           5020 non-null   int64
 5   Qty             5020 non-null   int64
 6   TotalAmount     5020 non-null   int64
 7   StoreID         5020 non-null   int64
dtypes: int64(5), object(3)
memory usage: 313.9+ KB
```

```
#mengubah tipe data
transaction['Date']= pd.to_datetime(transaction['Date'],format='%d/%m/%Y')
transaction = transaction.astype({"ProductID":"category","StoreID":"category"})
transaction.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5020 entries, 0 to 5019
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   TransactionID   5020 non-null   object
 1   CustomerID      5020 non-null   int64
 2   Date            5020 non-null   datetime64[ns]
 3   ProductID       5020 non-null   category
 4   Price           5020 non-null   int64
 5   Qty             5020 non-null   int64
 6   TotalAmount     5020 non-null   int64
 7   StoreID         5020 non-null   category
dtypes: category(2), datetime64[ns](1), int64(4), object(1)
memory usage: 246.3+ KB
```

```
#mengecek kategori yang terdapat pada variabel kategorik
print(transaction.groupby(['ProductID'])['ProductID'].count())
print("")
print(transaction.groupby(['StoreID'])['StoreID'].count())
```

```
ProductID
P1     397
P10    620
P2     530
P3     519
P4     390
P5     814
P6     255
P7     522
P8     485
P9     488
Name: ProductID, dtype: int64

StoreID
1    354
2    364
3    367
4    350
5    362
```

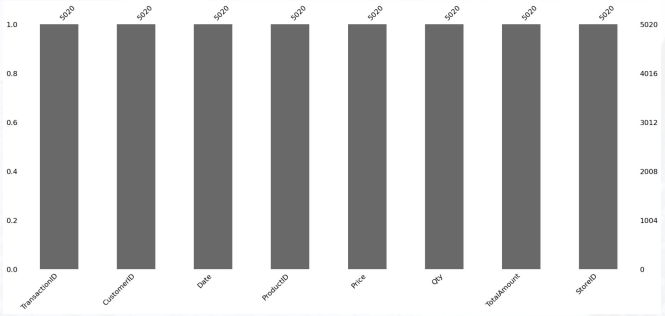
Data Preprocessing

Rakamin Academy

Transaction Data Preprocessing

```
#mengecek missing values
msno.bar(transaction)
plt.show()
transaction.isna().sum()
```

```
TransactionID    0
CustomerID       0
Date             0
ProductID        0
Price            0
Qty              0
TotalAmount      0
StoreID          0
dtype: int64
```




```
#statistika deskriptif
transaction[["Price","Qty","TotalAmount"]].describe()
```

	Price	Qty	TotalAmount
count	5020.000000	5020.000000	5020.000000
mean	9684.800797	3.644622	32279.482072
std	4600.708780	1.855295	19675.462455
min	3200.000000	1.000000	7500.000000
25%	4200.000000	2.000000	16000.000000
50%	9400.000000	3.000000	28200.000000
75%	15000.000000	5.000000	47000.000000
max	18000.000000	10.000000	88000.000000

```
#mengecek duplikasi
transaction.duplicated().value_counts()
```

```
False    5020
dtype: int64
```

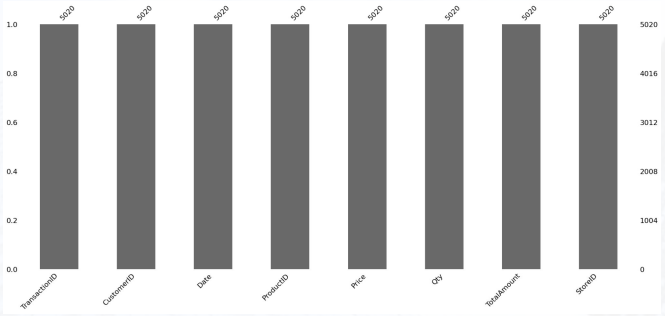
Data Preprocessing

Rakamin Academy

Transaction Data Preprocessing

```
#mengecek missing values
msno.bar(transaction)
plt.show()
transaction.isna().sum()
```

```
TransactionID    0
CustomerID       0
Date             0
ProductID        0
Price            0
Qty              0
TotalAmount      0
StoreID          0
dtype: int64
```




```
#statistika deskriptif
transaction[["Price","Qty","TotalAmount"]].describe()
```

	Price	Qty	TotalAmount
count	5020.000000	5020.000000	5020.000000
mean	9684.800797	3.644622	32279.482072
std	4600.708780	1.855295	19675.462455
min	3200.000000	1.000000	7500.000000
25%	4200.000000	2.000000	16000.000000
50%	9400.000000	3.000000	28200.000000
75%	15000.000000	5.000000	47000.000000
max	18000.000000	10.000000	88000.000000

```
#mengecek duplikasi
transaction.duplicated().value_counts()
```

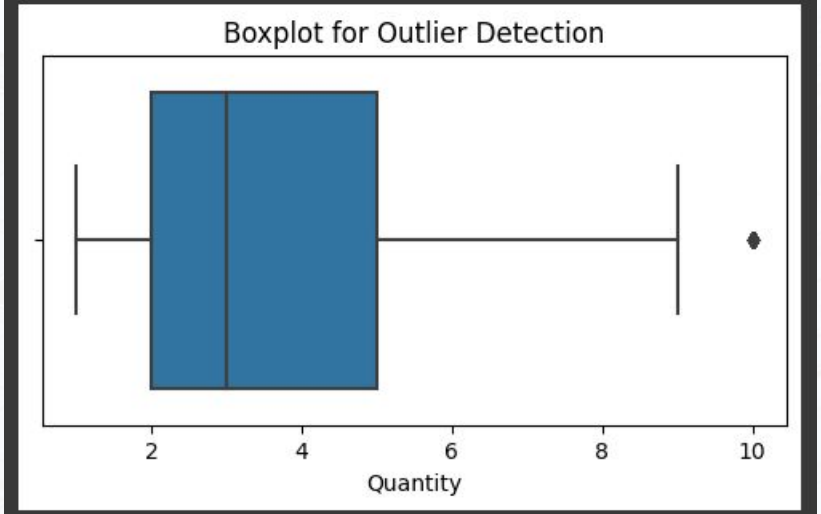
```
False    5020
dtype: int64
```

Data Preprocessing

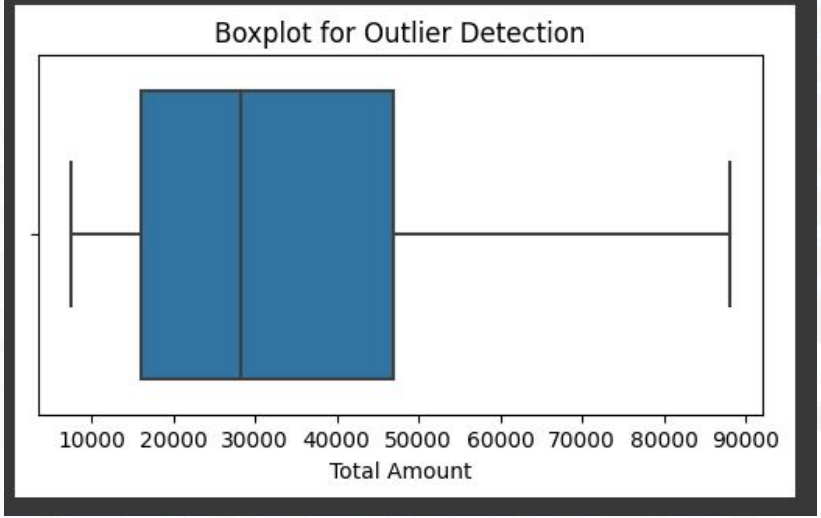
Rakamin Academy

Transaction Data Preprocessing

```
#mengecek outlier pada kolom "Qty"
plt.figure(figsize=(6, 3))
sns.boxplot(x=transaction['Qty'])
plt.title('Boxplot for Outlier Detection')
plt.xlabel('Quantity')
plt.show()
```




```
#mengecek outlier pada kolom "TotalAmount"
plt.figure(figsize=(6, 3))
sns.boxplot(x=transaction['TotalAmount'])
plt.title('Boxplot for Outlier Detection')
plt.xlabel('Total Amount')
plt.show()
```



Data Preprocessing

Rakamin Academy

Store Data Preprocessing

```
#melihat dataset
store
```

	StoreID	StoreName	GroupStore	Type	Latitude	Longitude
0	1	Prima Tendean	Prima	Modern Trade	-6,2	106,816666
1	2	Prima Kelapa Dua	Prima	Modern Trade	-6,914864	107,608238
2	3	Prima Kota	Prima	Modern Trade	-7,797068	110,370529
3	4	Gita Ginara	Gita	General Trade	-6,966667	110,416664
4	5	Bonafid	Gita	General Trade	-7,250445	112,768845
5	6	Lingga	Lingga	Modern Trade	-5,135399	119,42379
6	7	Buana Indah	Buana	General Trade	3,316694	114,590111
7	8	Sinar Harapan	Harapan Baru	General Trade	5,54829	95,323753
8	9	Lingga	Lingga	Modern Trade	-3,654703	128,190643
9	10	Harapan Baru	Harapan Baru	General Trade	3,597031	98,678513
10	11	Sinar Harapan	Prestasi	General Trade	0,533505	101,447403
11	12	Prestasi Utama	Prestasi	General Trade	-2,990934	104,756554
12	13	Buana	Buana	General Trade	-1,26916	116,825264
13	14	Priangan	Priangan	Modern Trade	-5,45	105,26667

```
#mengubah tipe data
store["Latitude"] = store['Latitude'].str.replace(',', '.').astype(float)
store["Longitude"] = store['Longitude'].str.replace(',', '.').astype(float)
store=store.astype({"GroupStore":"category","Type":"category"})
store.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   StoreID     14 non-null     int64
 1   StoreName   14 non-null     object
 2   GroupStore  14 non-null     category
 3   Type        14 non-null     category
 4   Latitude    14 non-null     float64
 5   Longitude   14 non-null     float64
dtypes: category(2), float64(2), int64(1), object(1)
memory usage: 1.1+ KB
```

```
#melihat informasi yang terdapat pada dataset
store.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   StoreID     14 non-null     int64
 1   StoreName   14 non-null     object
 2   GroupStore  14 non-null     object
 3   Type        14 non-null     object
 4   Latitude    14 non-null     object
 5   Longitude   14 non-null     object
dtypes: int64(1), object(5)
memory usage: 800.0+ bytes
```

```
#mengecek kategori yang terdapat pada variabel kategorik
print(store.groupby(['GroupStore'])['GroupStore'].count())
print("")
print(store.groupby(['Type'])['Type'].count())
```

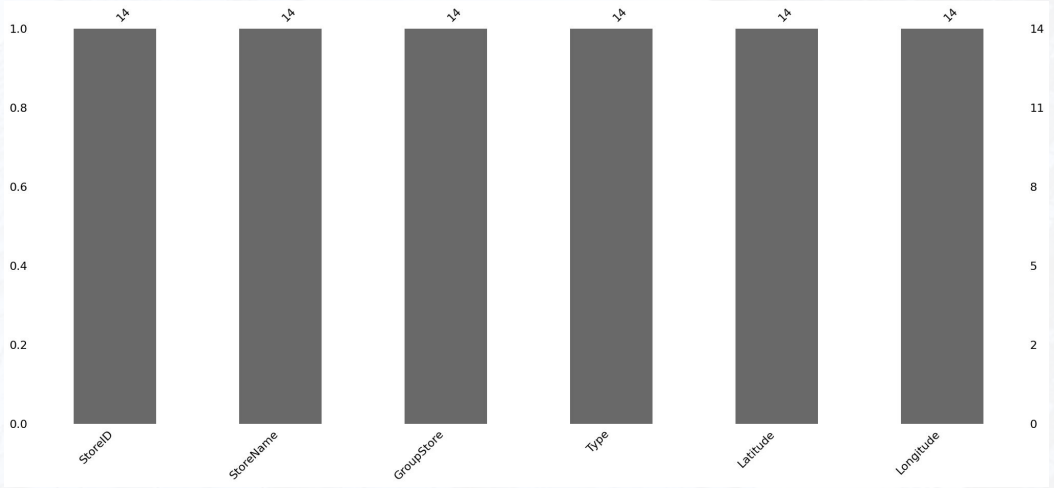
```
GroupStore
Buana           2
Gita            2
Harapan Baru    2
Lingga          2
Prestasi        2
Priangan        1
Prima           3
Name: GroupStore, dtype: int64

Type
General Trade    8
Modern Trade     6
Name: Type, dtype: int64
```

Data Preprocessing

Rakamin Academy

Store Data Preprocessing

```
#mengecek missing values
msno.bar(store)
plt.show()
store.isna().sum()
```




```
#mengecek duplikasi
store.duplicated().value_counts()
```

```
False    14
dtype: int64
```

Data Preprocessing

Rakamin Academy

Customer Data Preprocessing

```
#melihat dataset
customer
```

	CustomerID	Age	Gender	Marital Status	Income
0	1	55	1	Married	5,12
1	2	60	1	Married	6,23
2	3	32	1	Married	9,17
3	4	31	1	Married	4,87
4	5	58	1	Married	3,57
...	...	...	...	...	...
442	443	33	1	NaN	9,28
443	444	53	0	Married	15,31

```
#melihat informasi pada dataset
customer.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 447 entries, 0 to 446
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   CustomerID      447 non-null    int64
 1   Age             447 non-null    int64
 2   Gender          447 non-null    int64
 3   Marital Status  444 non-null    object
 4   Income          447 non-null    object
dtypes: int64(3), object(2)
memory usage: 17.6+ KB
```

```
#mengubah tipe data
customer["Income"] = customer['Income'].str.replace(',', '.').astype(float)
customer=customer.astype({"Gender":"category","Marital Status":"category"})
customer.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 447 entries, 0 to 446
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   CustomerID      447 non-null    int64
 1   Age             447 non-null    int64
 2   Gender          447 non-null    category
 3   Marital Status  444 non-null    category
 4   Income          447 non-null    float64
dtypes: category(2), float64(1), int64(2)
memory usage: 11.7 KB
```

```
#mengecek kategori yang terdapat pada variabel kategorik
print(customer.groupby(['Marital Status'])['Marital Status'].count())
print("")
print(customer.groupby(['Gender'])['Gender'].count())
```

```
Marital Status
Married    340
Single     104
Name: Marital Status, dtype: int64

Gender
0    242
1    205
Name: Gender, dtype: int64
```

Data Preprocessing

Rakamin Academy

Customer Data Preprocessing

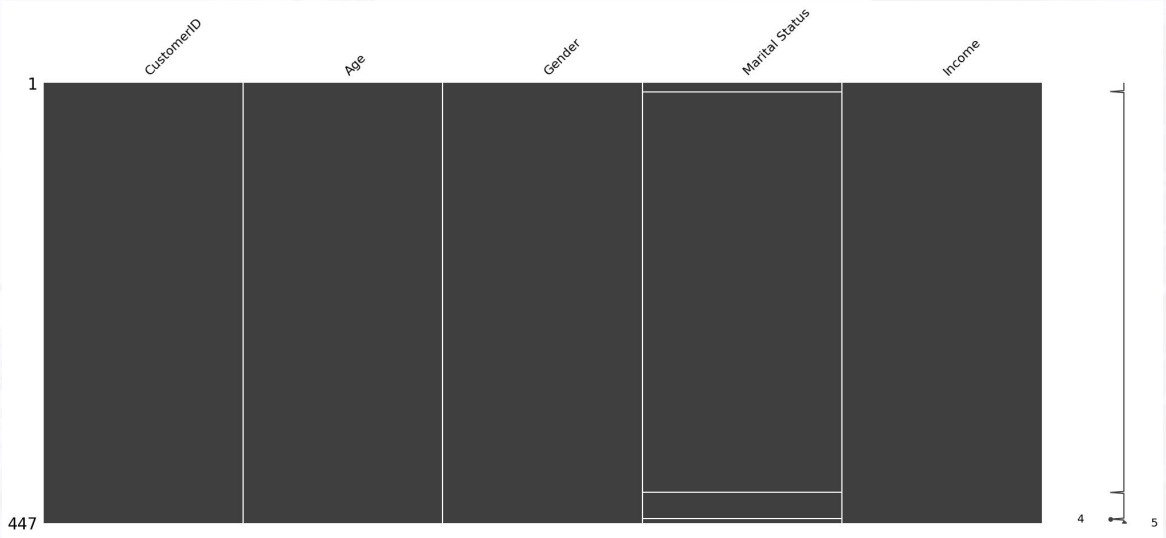
```
#mengecek missing values
msno.matrix(customer)
plt.show()
customer.isna().sum()
```

```
#mengecek rows dengan missing value
customer[customer['Marital Status'].isnull()]
```

	CustomerID	Age	Gender	Marital Status	Income
9	10	34	1	NaN	4.00
415	416	27	1	NaN	3.43
442	443	33	1	NaN	9.28

```
#imputasi data (pake mode karena data kategorik)
customer['Marital Status'].fillna(customer['Marital Status'].mode()[0], inplace=True)
print(customer.isna().sum())
print("")
print(customer.groupby(['Marital Status'])['Marital Status'].count())
```

```
CustomerID       0
Age              0
Gender           0
Marital Status   0
Income           0
dtype: int64

Marital Status
Married    343
Single     104
Name: Marital Status, dtype: int64
```



Data Preprocessing

Rakamin Academy

Customer Data Preprocessing

```
#mengecek duplikasi
customer.duplicated().value_counts()
```

```
False    447
dtype: int64
```

```
#akan dilihat lebih lanjut data dengan nilai "Age" <15
customer[customer['Age']<15]
```

	CustomerID	Age	Gender	Marital Status	Income
11	12	2	1	Married	4.94
73	74	3	1	Married	5.09
127	128	0	1	Married	6.77

```
#statistika deskriptif(aneh ada age yang 0)
customer[["Income","Age"]].describe()
```

	Income	Age
count	447.000000	447.000000
mean	8.592103	39.782998
std	6.607065	12.848719
min	0.000000	0.000000
25%	4.175000	30.000000
50%	7.520000	39.000000
75%	10.810000	50.500000
max	71.300000	72.000000

```
#akan didrop rows dengan anomali tersebut
anomaly = customer['Age']<15
customer = customer[~anomaly]
customer[["Income","Age"]].describe()
```

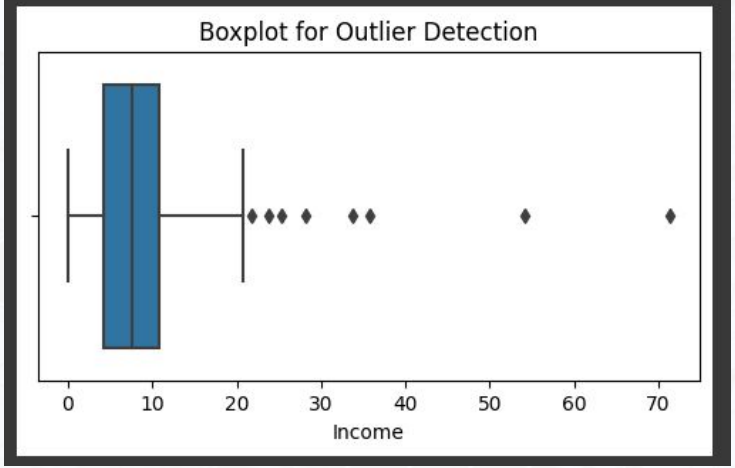
	Income	Age
count	444.000000	444.000000
mean	8.612320	40.040541
std	6.624442	12.501672
min	0.000000	18.000000
25%	4.162500	30.750000
50%	7.545000	39.000000
75%	10.845000	51.000000
max	71.300000	72.000000

Data Preprocessing

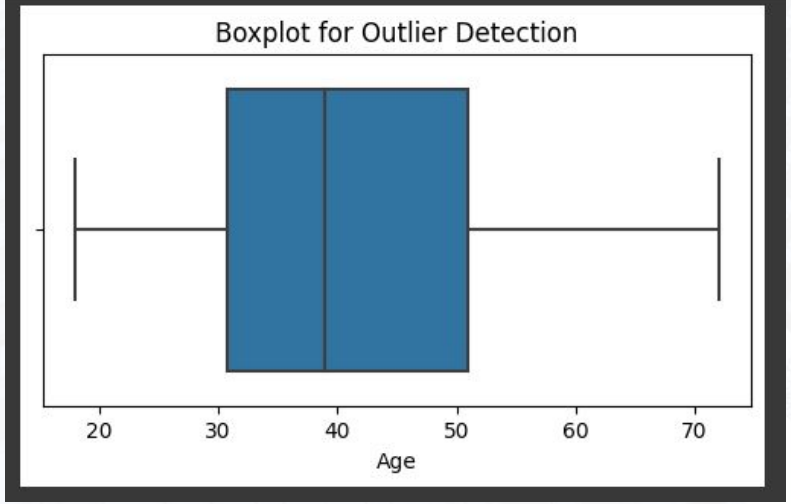
Rakamin Academy

Customer Data Preprocessing

```
#mengecek outlier pada kolom "Income"
plt.figure(figsize=(6, 3))
sns.boxplot(x=customer['Income'])
plt.title('Boxplot for Outlier Detection')
plt.xlabel('Income')
plt.show()
```




```
#mengecek outlier pada kolom "Age"
plt.figure(figsize=(6, 3))
sns.boxplot(x=customer['Age'])
plt.title('Boxplot for Outlier Detection')
plt.xlabel('Age')
plt.show()
```



Data Preprocessing

Rakamin Academy

Product Data Preprocessing

```
#melihat dataset
product
```

	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
5	P6	Cashew	18000
6	P7	Coffee Candy	9400
7	P8	Oat	16000
8	P9	Yoghurt	10000
9	P10	Cheese Stick	15000

```
#melihat informasi yang terdapat pada dataset
product.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   ProductID     10 non-null     object
 1   Product Name  10 non-null     object
 2   Price         10 non-null     int64
dtypes: int64(1), object(2)
memory usage: 368.0+ bytes
```

```
#mengubah tipe data
product=product.astype({"ProductID":"category","Product Name":"category"})
product.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   ProductID     10 non-null     category
 1   Product Name  10 non-null     category
 2   Price         10 non-null     int64
dtypes: category(2), int64(1)
memory usage: 988.0 bytes
```

```
#mengecek kategori yang terdapat pada variabel kategorik
print(product.groupby(['ProductID'])['ProductID'].count())
print("")
print(product.groupby(['Product Name'])['Product Name'].count())
```

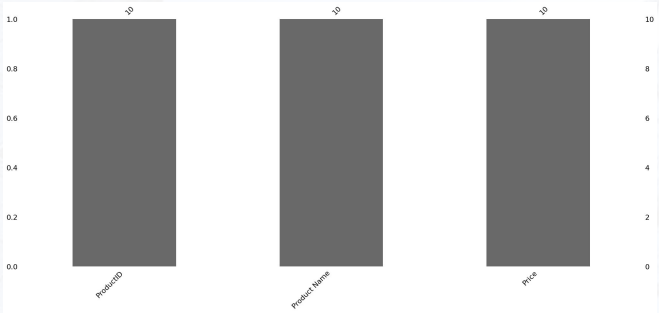
```
ProductID
P1     1
P10    1
P2     1
P3     1
P4     1
P5     1
P6     1
P7     1
P8     1
P9     1
Name: ProductID, dtype: int64

Product Name
Cashew          1
Cheese Stick    1
Choco Bar       1
Coffee Candy    1
Crackers        1
Ginger Candy    1
Oat             1
Potato Chip     1
Thai Tea        1
Yoghurt         1
Name: Product Name, dtype: int64
```

Data Preprocessing

Rakamin Academy

Product Data Preprocessing

```
#mengecek missing values
msno.bar(product)
plt.show()
product.isna().sum()
```




```
#mengecek duplikasi
product.duplicated().value_counts()
```

```
False    10
dtype: int64
```

```
#statistika deskriptif
product.describe()
```

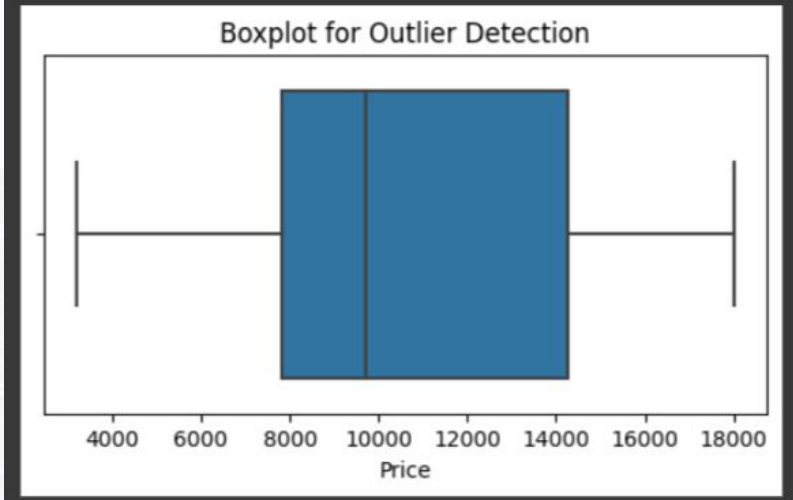
	Price
count	10.000000
mean	10410.000000
std	4890.455557
min	3200.000000
25%	7825.000000
50%	9700.000000
75%	14250.000000
max	18000.000000

Data Preprocessing

Rakamin Academy

Product Data Preprocessing

```
#mengecek outlier pada kolom "Price"
plt.figure(figsize=(6, 3))
sns.boxplot(x=product['Price'])
plt.title('Boxplot for Outlier Detection')
plt.xlabel('Price')
plt.show()
```



Data Forecasting

Rakamin Academy

```
#merge semua data yang telah dipreprocessing
merged_df = pd.merge(transaction, customer, on='CustomerID')
merged_df = pd.merge(merged_df, product, on='ProductID')
merged_df = pd.merge(merged_df, store, on='StoreID')
merged_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4989 entries, 0 to 4988
Data columns (total 19 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   TransactionID   4989 non-null   object
 1   CustomerID      4989 non-null   int64
 2   Date            4989 non-null   datetime64[ns]
 3   ProductID       4989 non-null   category
 4   Price_x         4989 non-null   int64
 5   Qty             4989 non-null   int64
 6   TotalAmount     4989 non-null   int64
 7   StoreID         4989 non-null   int64
 8   Age             4989 non-null   int64
 9   Gender          4989 non-null   category
 10  Marital Status  4989 non-null   category
 11  Income          4989 non-null   float64
 12  Product Name    4989 non-null   category
 13  Price_y         4989 non-null   int64
 14  StoreName       4989 non-null   object
 15  GroupStore      4989 non-null   category
 16  Type            4989 non-null   category
 17  Latitude        4989 non-null   float64
 18  Longitude       4989 non-null   float64
dtypes: category(6), datetime64[ns](1), float64(3), int64(7), object(2)
memory usage: 576.4+ KB
```

```
#bikin dataset time-series baru dengan kolom date dan qty,
#kemudian groupby berdasarkan tanggal order dengan qty dijumlahkan
datexqty = merged_df[["Date","Qty"]]
datexqty.set_index('Date', inplace=True)
ts_df = datexqty.groupby('Date').sum()
ts_df
```

Date	Qty
01/01/2022	49
01/02/2022	50
01/03/2022	76
01/04/2022	98
01/05/2022	67
...	...
31/05/2022	21
31/07/2022	72
31/08/2022	36
31/10/2022	69
31/12/2022	37

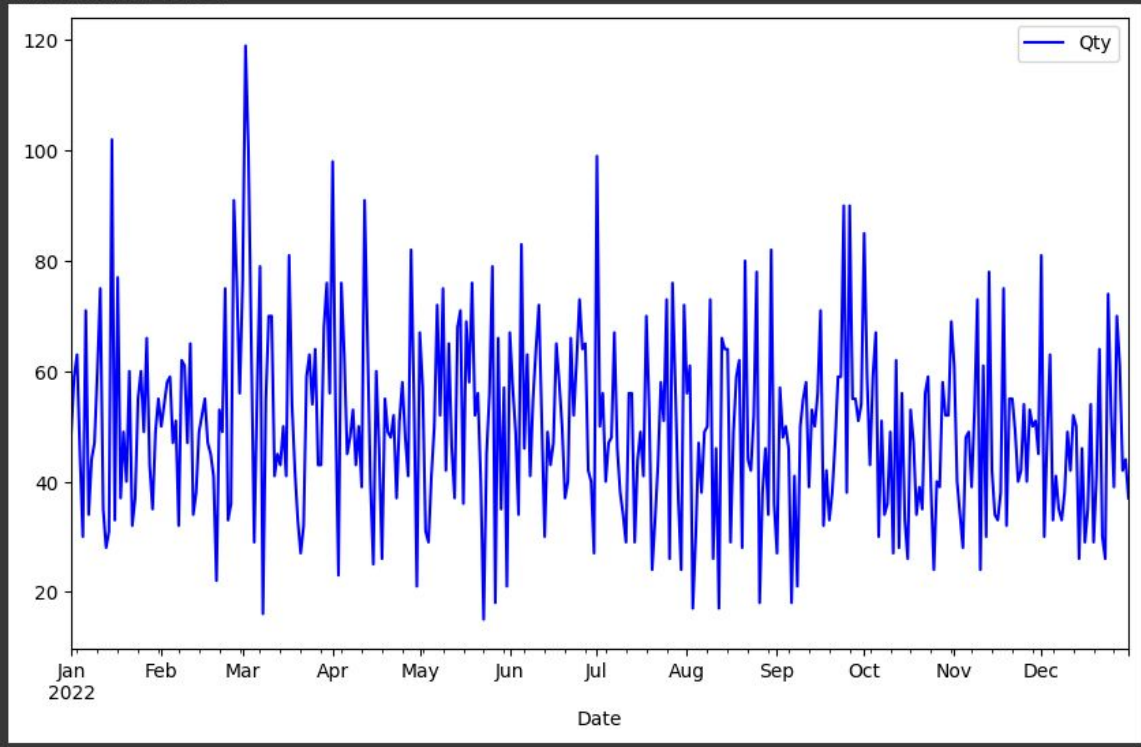
365 rows × 1 columns

Data Forecasting

Rakamin Academy

```
#plot dari data time-series
ts_df.plot(kind='line', color='blue',figsize=(10, 6))
```

```
<Axes: xlabel='Date'>
```



Data Forecasting

Rakamin Academy

H0: data tidak stasioner
H1: data stasioner

```
#tes stasioneritas menggunakan augmented dickey-fuller(ADF) test
from statsmodels.tsa.stattools import adfuller
result = adfuller(ts_df)
print(f'ADF Statistic: {result[0]}')
print(f'p-value: {result[1]}')
#stasioner, tidak perlu didifferencing
```

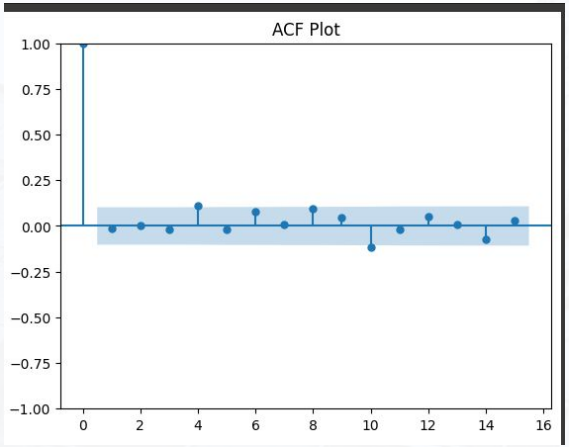
```
ADF Statistic: -19.17337367516702
p-value: 0.0
```

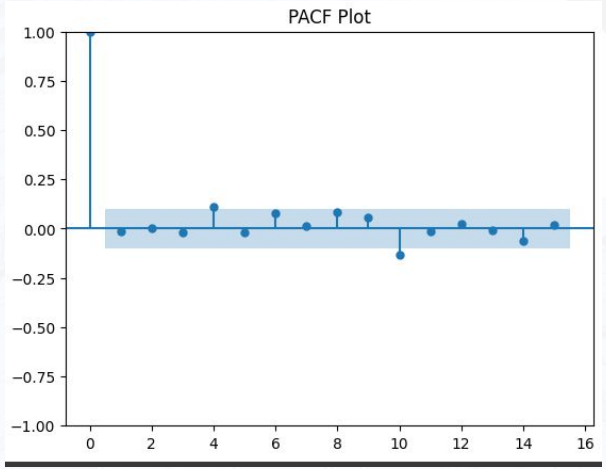
Karena P-value < 0.05 maka H0 ditolak. Sehingga data stasioner

Data Forecasting

Rakamin Academy

```python
#mengecek order dari ARIMA menggunakan plot ACF dan PACF
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf


plot_acf(ts_df, lags=15)
plt.title('ACF Plot')
plt.show()

plot_pacf(ts_df, lags=15)
plt.title('PACF Plot')
plt.show()
```

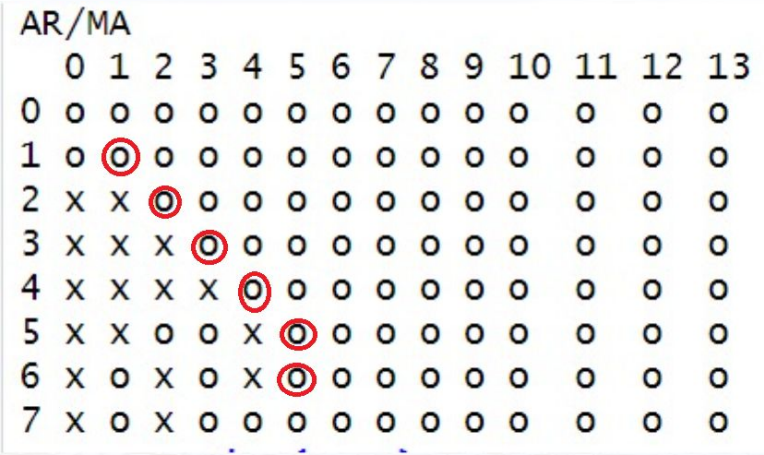




Data Forecasting

Rakamin Academy

Plot EACF



AR/MA	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	o	o	o	o	o	o	o	o	o	o	o	o	o	o
1	o	**o**	o	o	o	o	o	o	o	o	o	o	o	o
2	x	x	**o**	o	o	o	o	o	o	o	o	o	o	o
3	x	x	x	**o**	o	o	o	o	o	o	o	o	o	o
4	x	x	x	x	**o**	o	o	o	o	o	o	o	o	o
5	x	x	o	o	x	**o**	o	o	o	o	o	o	o	o
6	x	o	x	o	x	**o**	o	o	o	o	o	o	o	o
7	x	o	x	o	o	o	o	o	o	o	o	o	o	o

Terdapat beberapa model yang mungkin, yaitu:

1. Model ARMA(1,1)
2. Model ARMA(2,2)
3. Model ARMA(3,3)
4. Model ARMA(4,4)
5. Model ARMA(5,5)
6. Model ARMA(6,5)

Data Forecasting

Rakamin Academy

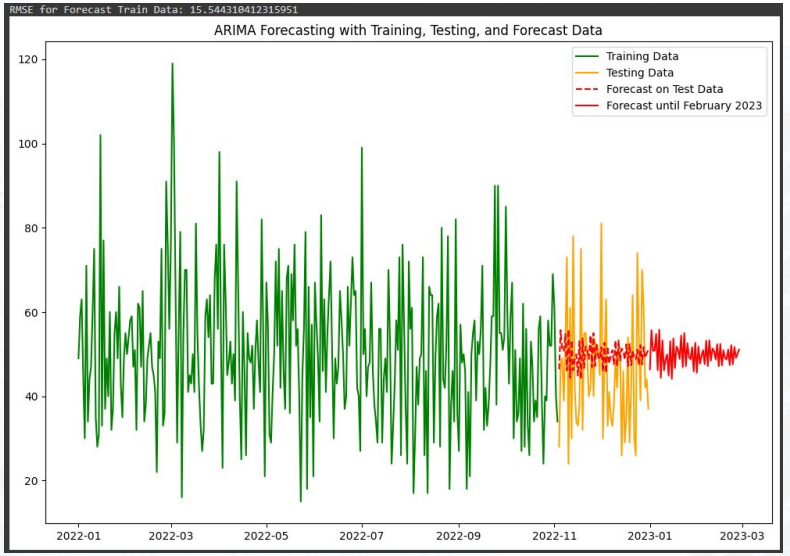
```
#mengecek model terbaik
import statsmodels.api as sm
order1 = (1, 0, 1)
order2 = (2, 0, 2)
order3 = (3, 0, 3)
order3 = (3, 0, 3)
order4 = (4, 0, 4)
order5 = (5, 0, 5)
order6 = (6, 0, 5)


model1 = sm.tsa.arima.ARIMA(ts_df, order=(1, 0, 1))
model2 = sm.tsa.arima.ARIMA(ts_df, order=(2, 0, 2))
model3 = sm.tsa.arima.ARIMA(ts_df, order=(3, 0, 3))
model4 = sm.tsa.arima.ARIMA(ts_df, order=(4, 0, 4))
model5 = sm.tsa.arima.ARIMA(ts_df, order=(5, 0, 5))
model6 = sm.tsa.arima.ARIMA(ts_df, order=(6, 0, 5))

results = [model1.fit(), model2.fit(),model3.fit(),model4.fit(),model5.fit(),model6.fit()]

for i in results:
  print(i.aic)
```

Model	Nilai AIC
ARMA(1,1)	3095.316597751217
ARMA(2,2)	3097.553565472534
ARMA(3,3)	3097.427440927556
ARMA(4,4)	3090.6043573922207
ARMA(5,5)	3100.5158172297697
ARMA(6,5)	3102.895450218636

Data Forecasting

Rakamin Academy

```python
#membagi data train dan test
test_size = 58
train_data, test_data = ts_df[:-test_size], ts_df[-test_size:]

# Fit ARIMA model
model_fit = model4.fit()

# Forecast
forecast_steps = len(test_data)
forecast = model_fit.get_forecast(steps=forecast_steps)
forecast_values = forecast.predicted_mean
time_index_future = pd.date_range(start=ts_df.index[-1], periods=forecast_steps + 1, freq='D')[1:]

# Calculate RMSE
from sklearn.metrics import mean_squared_error
from math import sqrt
rmse1 = sqrt(mean_squared_error(test_data, forecast_values))
print(f"RMSE for Forecast Train Data: {rmse1}")


# Visualize the results
plt.figure(figsize=(12, 8))
plt.plot(train_data.index, train_data, label='Training Data', color='green')
plt.plot(test_data.index, test_data, label='Testing Data', color='orange')
plt.plot(test_data.index, forecast_values, label='Forecast on Test Data', color='red', linestyle='dashed')
plt.plot(time_index_future,forecast_values, color='red', label='Forecast until February 2023')
plt.title('ARIMA Forecasting with Training, Testing, and Forecast Data')
plt.legend()
plt.show()
```

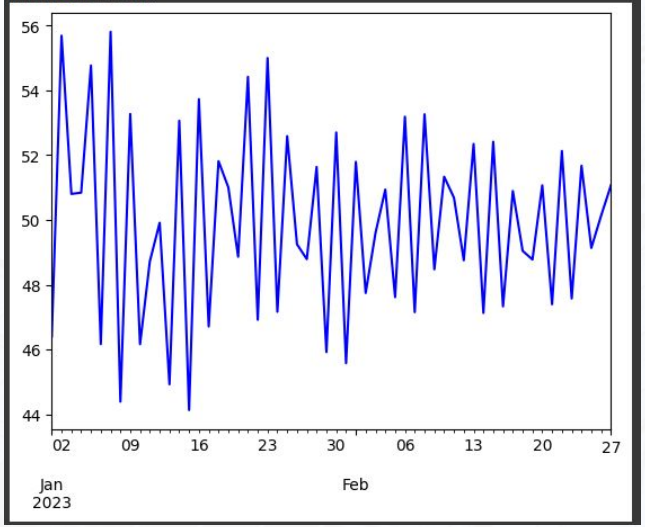


Dari output di atas, terlihat bahwa nilai RMSE cukup besar dan dari plot juga terlihat hasil forecast test data tidak mengikuti garis data test sebenarnya sehingga hasil forecasting kurang akurat. Hal ini sepertinya terjadi karena data yang digunakan terlalu sedikit.

Data Forecasting

```
#Predicted future values plot
forecast_values.plot(kind='line', color='blue')
np.mean(forecast_values)
```

```
49.92697966578447
```



Dari hasil prediksi, terlihat bahwa jumlah penjualan(qty) per hari dari awal januari sampe akhir februari 2023 memiliki rata-rata 49,92 atau 50 produk. Dengan jumlah produk terjual terkecil 44 dan terbesar 56. Mengetahui hal ini, tim inventory bisa mempersiapkan stock harian produk di antara range tersebut. Akan tetapi, perlu diperhatikan pula dari penjelasan sebelumnya, hasil prediksi ini kurang baik akibat dari minimnya jumlah data yang digunakan.

Data Clustering

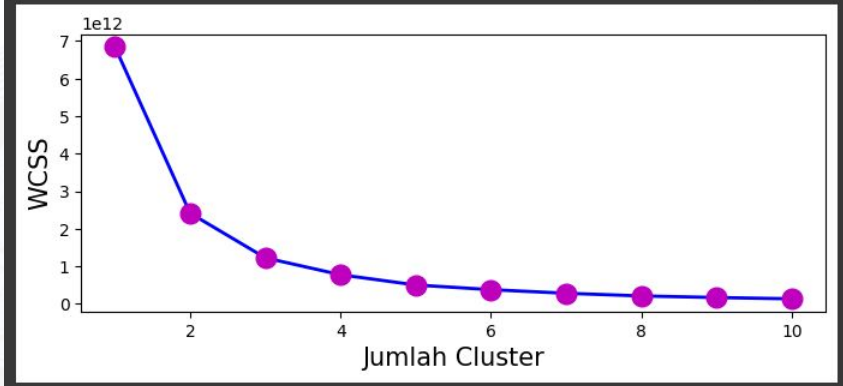
Rakamin Academy

```
cl_df = merged_df.groupby("CustomerID").agg({
    "TransactionID": "count",
    "Qty": "sum",
    "TotalAmount": "sum"
}).reset_index()
cl_df = cl_df.set_index("CustomerID")
cl_df
```

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

444 rows × 3 columns

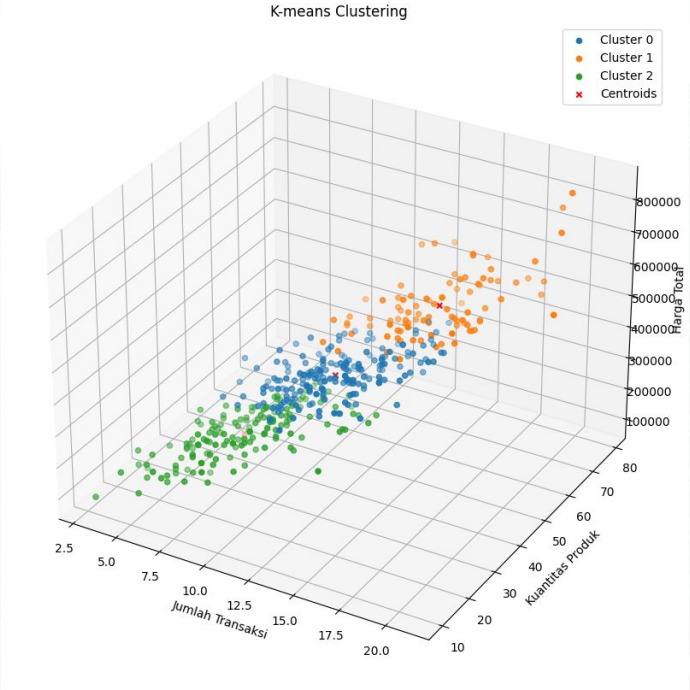
```
#menentukan banyak K kelompok optimal
from sklearn.cluster import KMeans
wcss=[]
for n in range(1,11):
  model=KMeans(n_clusters=n, init="k-means++", n_init=10,max_iter=300,tol=0.0001,random_state=100)
  model.fit(cl_df)
  wcss.append(model.inertia_)

plt.figure(figsize=(8,3))
plt.plot(list(range(1,11)),wcss,color='blue',marker="o",linewidth=2,markersize=12,markerfacecolor='m',markeredgecolor="m")
plt.xlabel("Jumlah Cluster",fontsize=15)
plt.ylabel("WCSS",fontsize=15)
plt.show()
```



Data Clustering

Rakamin Academy

```python
#membuat model k-means clustering dengan k = 3
modelopt = KMeans(n_clusters=3, init="k-means++", n_init=10,max_iter=300,tol=0.0001,random_state=100)
modelopt.fit(cl_df)
labels=modelopt.labels_
centroids=modelopt.cluster_centers_
cl_df["labels"] = labels
#plot data
from mpl_toolkits.mplot3d import Axes3D  # For 3D scatter plot

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

tr =cl_df["TransactionID"]
qty = cl_df["Qty"]
total = cl_df["TotalAmount"]
cluster_labels = cl_df["labels"]


for cluster in set(cluster_labels):
    cluster_data = cl_df[cl_df["labels"] == cluster]
    ax.scatter(cluster_data['TransactionID'], cluster_data['Qty'], cluster_data['TotalAmount'], label=f'Cluster {cluster}')


#plot centroids
ax.scatter(centroids[:,0], centroids[:,1], centroids[:,2], c="red", marker='x', label='Centroids')


ax.set_xlabel('Jumlah Transaksi')
ax.set_ylabel('Kuantitas Produk')
ax.set_zlabel('Harga Total')
ax.legend()
plt.title('K-means Clustering')
plt.show()
```



Data Clustering

Rakamin Academy

```
cl_df.groupby("labels").mean()
```

labels	TransactionID	Qty	TotalAmount
0	11.649215	42.340314	378894.764398
1	15.204301	57.720430	544133.333333
2	8.437500	29.556250	237568.750000

Dari hasil clustering menggunakan KMeans dengan 3 cluster, diperoleh informasi bahwa dari ketiga cluster, cluster kedua(1) memiliki jumlah transaksi yang terbanyak. Sedangkan, untuk cluster ketiga(2) memiliki jumlah transaksi yang paling sedikit. Hal ini dapat menjadi pertimbangan oleh tim marketing untuk membuat strategi baru untuk cluster ketiga, seperti memberikan promo sehingga dapat memperbanyak transaksi yang dilakukan.

Rakamin Academy

Link to Google Drive

Link to files used and presentation video are stored here:

https://drive.google.com/drive/folders/19t4aWQv4F7sURrseL-AI2u3BcjQLVfqs

Thank You