MODEL PEMBELAJARAN DAN LAPORAN AKHIR
PROJECT-BASED LEARNING
MATA KULIAH DATA WRANGLING
KELAS D

"PROSES PENERAPAN DATA WRANGLING UNTUK MENGANALISIS UKURAN KELUARGA PELANGGAN DALAM MEMPENGARUHI KEBIASAAN BELANJA"

DISUSUN OLEH KELOMPOK "VI" :

1. ADINDA PUTRI RHAINA (22083010002)
2. REZA PUTRI ANGGA (22083010006)
3. MUCHAMAD RISQI (22083010029)
4. ANNITA FADHILAH APRILIA (22083010033)
5. MUHAMMAD AZKIYA' AKMAL (22083010084)

DOSEN PENGAMPU:
KARTIKA MAULIDA HINDRAYANI, S.KOM, M.KOM (199209092022032009)

PROGRAM STUDI SAINS DATA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"
JAWA TIMUR
2023

Deskripsi Dataset dan Metadata

Data Wrangling merupakan proses dalam sains data. Di mulai dengan proses pengumpulan, pengolahan, dan penyajian informasi yang bermanfaat bagi pihak yang berkepentingan. Melalui proses transformasi dan pembersihan data mentah menjadi format yang dapat di pahami dan dapat di pergunakan untuk melakukan analisis lebih lanjut.

Pada penugasan ini, di lakukan serangkaian proses penerapan data wrangling pada suatu dataset bernama “Shop Customer Data” atau “Belanja Data Pelanggan”. Berisi mengenai data belanja pada toko. Tujuan dari rangkaian proses penerapan data wrangling ini, yakni melakukan analisis bagaimana ukuran keluarga pelanggan mempengaruhi kebiasaan belanja, sehingga toko dapat melakukan proses segmentasi pasar yang sesuai pada pelanggan.

Pemilik toko mendapatkan informasi mengenai pelanggan melalui kartu keanggotaan pelanggan. Dengan studi kasus, misalnya di perlukan untuk melihat apakah pelanggan dengan ukuran keluarga yang lebih besar cenderung memiliki pengeluaran yang lebih tinggi atau memilih produk tertentu yang sesuai untuk keluarga besar.

Untuk penjelasan mengenai metadata dalam dataset tersebut dapat di jelaskan lebih lanjut sebagai berikut :

1. Nama Dataset : Shop Customer Data
2. Sumber Dataset : Kaggle
   https://www.kaggle.com/datasets/datascientistanna/customers-dataset
3. Deskripsi Dataset :Berisi informasi mengenai data data belanja pelanggan pada suatu toko, dengan informasi yang di peroleh melalui kartu keanggotaan pelanggan selama satu tahun dengan tujuan agar mengetahui bagaimana ukuran keluarga pelanggan mempengaruhi kebiasaan belanja, sehingga toko dapat melakukan proses segmentasi pasar yang sesuai pada pelanggan.
4. Ukuran Dataset : Terdiri atas 200 baris dan 8 kolom
5. Format Dataset : Csv (comma-seperated-values)
6. Variabel Tipe Data : Terdiri atas beberapa variabel, di antaranya yakni

a) CustomerID : Berisi id pelanggan dalam toko dengan tipe data integer (numerik)
b) Gender : Berisi jenis kelamin pelanggan dengan tipe data string (karakter)
c) Age : Berisi usia pelanggan dengan tipe data integer (numerik)
d) Annual Income ($) : Berisi pendapatan tahunan pelanggan dengan tipe data integer (numerik)
e) Spending Score (1-100) :Berisi mengenai skor yang di peroleh pelanggan berdasarkan transaksi yang di lakukan dengan tipe data integer (numerik)
f) Profession : Berisi mengenai profesi pelanggan dengan tipe data string (karakter)
g) Work Experience : Berisi mengenai pengalaman kerja pelanggan dengan tipe data string (karakter)
h) Family Size : Berisi mengenai jumlah keanggotaan keluarga pelanggan dengan tipe data integer (numerik).

Proses penerapan data wrangling ini di mulai dari proses penemuan data, pemformatan atau perapian data, pembersihan data, transformasi data, visualisasi, sql dan python. Sehingga dari serangkaian proses data wrangling ini dapat di hasilkan suatu output yang berguna bagi pihak yang berkepentingan.

Proses Penerapan Data Wrangling

Untuk proses penerapan data wrangling pada dataset Shop Customer Data ini, dapat dilakukan serangkaian proses di mulai dengan penemuan data, pemformatan atau perapian data, pembersihan data, transformasi data, visualisasi data, dan menyambungkan dataset yang telah diolah ke dalam python dan sql.

Mengenai langkah secara detail, dapat dijelaskan lebih lanjut sebagai berikut :

A. Penemuan Data

Tahapan pertama dalam proses data wrangling yakni melakukan penemuan data. Penemuan data ini bertujuan untuk mengetahui dataset yang akan di proses dan bagaimana insight (output) yang akan dihasilkan dari dataset yang telah diperoleh.

A.1 Library Yang Di Perlukan

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.ensemble import IsolationForest
from sklearn.preprocessing import StandardScaler
```

Dilakukan import untuk penggunaan 4 (empat) library, yakni pandas, numpy, matplotlib, dan sklearn. Masing-masing library tersebut memiliki kegunaan yang berbeda dalam proses data wrangling ini.

Library pandas yang dimisalkan sebagai pd dipergunakan untuk meload dan melakukan pengolahan pada dataset, library numpy yang dimisalkan sebagai np dipergunakan untuk melakukan perhitungan komputasi dengan mendeteksi jumlah outliers, library matplotlib yang dimisalkan sebagai plt dipergunakan untuk melakukan visualisasi plot dan histogram dari hubungan kolom-kolom, dan library sklearn dipergunakan untuk instance class untuk melatih model dan menampilkan outliers.

A.2 Load Dataset

```
print("Di Tampilkan Dataset Shop Customer Data : ")

dfs = pd.read_csv("Shop_Customer_Data.csv")
dfs
```

```
Di Tampilkan Dataset Shop Customer Data :
```

	CustomerID	Gender	Age	Annual Income ($)	Spending Score (1-100)	Profession	Work Experience	Family Size
0	1	Male	19	15000	39	Healthcare	1	4
1	2	Male	21	35000	81	Engineer	3	3
2	3	Female	20	86000	6	Engineer	1	1
3	4	Female	23	59000	77	Lawyer	0	2
4	5	Female	31	38000	40	Entertainment	2	6
...	...	...	...	...	...	...	...	...
1995	1996	Female	71	184387	40	Artist	8	7
1996	1997	Female	91	73158	32	Doctor	7	7
1997	1998	Male	87	90961	14	Healthcare	9	2
1998	1999	Male	77	182109	4	Executive	7	2
1999	2000	Male	90	110610	52	Entertainment	5	2

2000 rows × 8 columns

Dilakukan load dataset bernama Shop Customer Data yang berada di directory yang sama dengan file kode script yang dijalankan. Dataset tersebut berisi mengenai sejumlah data pelanggan yang akan di proses yang dimasukkan ke dalam variabel dfs.

Proses load dataset ini menggunakan library pandas dan di tampilkan data 5 (lima) terbatas dan 5 (lima) terbawah yang akan dilakukan dalam proses penerapan data wrangling.

Dari proses ini di peroleh dapat di tentukan informasi yang relevan menganai dataset dan di tentukan bahwa output dari pengolahan dataset ini adalah menentukan bagaimana ukuran keluarga pelanggan mempengaruhi kebiasaan belanja, dengan menerapkan berbagai proses yang akan di lakukan selanjutnya.

B. Pemformatan Atau Perapian Data

Setelah mendapatkan data yang diperlukan, tahapan kedua adalah melakukan pemformatan atau perapian data. Pada tahapan ini, sangat penting untuk memastikan bahwa data tersebut memiliki format yang sesuai dan mudah diproses untuk tahapan analisis selanjutnya. Hal ini bertujuan untuk integrasi data, pengubahan nama kolom, penyesuaian tipe data untuk memastikan bahwa data untuk di eksplorasi, dianalisis, dan di manfaatkan dalam tahap selanjutnya dalam pembersihan data untuk kepentingan analisis data dan pembuatan model.

B.1 Pengubahan Nama Kolom

```
print("Di Tampilkan Pengubahan Nama Kolom pada Dataset Shop Customer Data : ")

dfs = dfs.rename(columns={'Annual Income ($)': 'Annual_Income'})
dfs = dfs.rename(columns={'Spending Score (1-100)': 'Spending_Score'})
dfs = dfs.rename(columns={'Work Experience': 'Work_Experience'})
dfs = dfs.rename(columns={'Family Size': 'Size_Family'})

print(dfs.columns)
```

```
Di Tampilkan Pengubahan Nama Kolom pada Dataset Shop Customer Data :
Index(['CustomerID', 'Gender', 'Age', 'Annual_Income', 'Spending_Score',
       'Profession', 'Work_Experience', 'Size_Family'],
      dtype='object')
```

Di lakukan pengubahan nama kolom dari dataset Shop Customer Data pada masing-masing kolom. Dengan masing-masing nama kolom yang awalnya Annual Income ($), Spending Score (1-100), Work Experience, dan Family Size, dilakukan perubahan nama (rename) pada masing-masing kolom tersebut, yakni menjadi Annual_Income, Spending_Score, Work_Experience, Size_Family.

Dengan tujuan agar lebih mudah dipanggil dan dimanfaatkan dalam proses data wrangling ini.

B.2 Penampilan Jumlah Baris Dan Kolom

```
print("Di Tampilkan Jumlah Kolom dan Baris pada Dataset Shop Customer Data : ")

dfs.shape
```

```
Di Tampilkan Jumlah Kolom dan Baris pada Dataset Shop Customer Data :

(2000, 8)
```

Di lakukan pengecekan ukuran jumlah kolom dan baris dari dataset Shop Customer Data dengan menggunakan dfs.shape untuk mengetahui berapa banyak data dan kolom yang tersedia. Dan di tampilkan jumlah data dari dataset sebanyak 2000 baris dan 8 kolom.

B.3 Penampilan Tipe Data Pada Masing-Masing Kolom

```
print("Di Tampilkan Pengecekan Tipe Dataset Shop Customer Data : ")

dfs.dtypes
```

```
Di Tampilkan Pengecekan Tipe Dataset Shop Customer Data :

CustomerID          int64
Gender             object
Age                 int64
Annual_Income       int64
Spending_Score      int64
Profession         object
Work_Experience     int64
Size_Family         int64
dtype: object
```

Dilakukan pengecekan tipe data dari dataset Shop Customer Data dengan menggunakan dfs.dtypes. Dengan tujuan agar dapat menentukan kolom mana yang ingin di proses dan kolom mana yang tidak di butuhkan dalam proses ini dan bisa di hilangkan.

B.4 Pengubahan Tipe Data

```
print("Di Tampilkan Perubahan Tipe Dataset Shop Customer Data : ")

dfs['Age'] = dfs['Age'].astype(float)
dfs.dtypes
```

```
Di Tampilkan Perubahan Tipe Dataset Shop Customer Data :

CustomerID           int64
Gender              object
Age                float64
Annual_Income        int64
Spending_Score       int64
Profession          object
Work_Experience      int64
Size_Family          int64
dtype: object
```

Di lakukan pengubahan tipe data dari dataset Shop Customer Data di kolom age dengan menggunakan astype. Kolom age yang mulanya bertipe data integer diubah menjadi float. Hal ini bertujuan agar umur dari pelanggan dapat di tuliskan secara detail.

Dari proses ini dapat di peroleh informasi yang relevan mengenai penamaan kolom terkini yang akan di proses, tipe data yang terdapat, dan mengubahan tipe data agar dapat di tuliskan secara mendetail untuk keperluan proses selanjutnya, yakni pembersihan data.

C. Pembersihan Data

Data yang sebelumnya ditemukan kemungkinan besar mengandung kesalahan, duplikasi, atau outlier yang dapat berdampak pada hasil analisis yang akurat. Oleh karena itu, tahapan ketiga dalam proses data wrangling yakni melakukan pembersihan data untuk memastikan keakuratan data yang digunakan. Hal ini bertujuan untuk mengecek apakah di dalam dataset terdapat kecacatan data seperti data yang kurang lengkap, kemudian mengecek apakah terdapat data yang duplikat di dalam dataset ataupun menghapus variabel atau kolom yang tidak diperlukan di dalam dataset yang nantinya akan dipergunakan untuk perhitungan guna dilakukan analisis.

C.1 Penemuan Missing Value

```
print("Di Tampilkan Missing Values Pada Dataset Shop Customer Data : ")

dfs.isnull()
```

```
Di Tampilkan Missing Values Pada Dataset Shop Customer Data :
```

	CustomerID	Gender	Age	Annual_Income	Spending_Score	Profession	Work_Experience	Size_Family
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
1995	False	False	False	False	False	False	False	False
1996	False	False	False	False	False	False	False	False
1997	False	False	False	False	False	False	False	False
1998	False	False	False	False	False	False	False	False
1999	False	False	False	False	False	False	False	False

2000 rows × 8 columns

Dilakukan penampilan menemukan missing values dalam bentuk data frame pada dataset Shop Customer Data dengan kode dfs.isnull() sebagai identifikasi missing values dalam dataset dan menampilakan Data Frame yang berisi nilai boolean. Dengan True apabila terapat nilai missing value dan false jika tidak terdapat nilai missing value.

C.2 Perhitungan Jumlah Missing Value

```
print("Di Tampilkan Perhitungan Jumlah Missing Values Pada Dataset Shop Customer Data : ")

dfs.isnull().sum()
```

```
Di Tampilkan Perhitungan Jumlah Missing Values Pada Dataset Shop Customer Data :

CustomerID          0
Gender              0
Age                 0
Annual_Income       0
Spending_Score      0
Profession         35
Work_Experience     0
Size_Family         0
dtype: int64
```

Dilakukan perhitungan jumlah missing values dari dataset Shop Customer Data pada setiap kolom dengan kode dfs.isnull().sum() untuk mengidentifikasi nilai yang hilang (missing) dan menjumlahkan nilai missing value pada setiap kolom, yang mewakili jumlah missing values pada kolom tersebut.

Dan di tampilkan bahwa kolom profession memiliki 35 missing value yang akan di proses lebih lanjut.

C.3 Penggantian Nilai Missing Value Dengan Undefined

```
print("Di Tampilkan Penggantian Nilai Missing Value Dengan Undefined Pada Dataset Shop Customer Data :")

dfs['Profession'].fillna('Undefined', inplace=True)
print(dfs.isnull().sum())
```

```
Di Tampilkan Penggantian Nilai Missing Value Dengan Undefined Pada Dataset Shop Customer Data :
CustomerID         0
Gender             0
Age                0
Annual_Income      0
Spending_Score     0
Profession         0
Work_Experience    0
Size_Family        0
dtype: int64
```

Di lakukan penggantian missing values dengan undefined dari dataset Shop Customer Data di kolom profession menggunakan fungsi fillna untuk mengisi nilai yang hilang dengan Undefined. Hal ini di pergunakan untuk menjaga kekonsistenan data.

Dan di tampilkan kolom dari profession yang memiliki nilai missing value telah di lakukan perubahan, sehingga semua kolom dari dataset tidak memiliki nilai missing value.

C.4 Pengecekan Data Duplikat

```
print("Di Tampilkan Pengecekan Data Duplikat Pada Dataset Shop Customer Data : ")

duplikat = dfs.duplicated()
duplikat
```

```
Di Tampilkan Pengecekan Data Duplikat Pada Dataset Shop Customer Data :

0       False
1       False
2       False
3       False
4       False
        ...
1995    False
1996    False
1997    False
1998    False
1999    False
Length: 2000, dtype: bool
```

Dilakukan pengecekan data duplikat dari dataset Shop Customer Data dengan menggunakan fungsi `duplicated()` dengan True jika baris data merupakan duplikat dari baris sebelumnya, dan False jika tidak.

Ditampilkan series yang menunjukkan keberadaan data duplikat pada setiap baris dalam dataset hal ini di pergunakan untuk memastikan bahwa analisis dan pemodelan yang akan di lakukan berdasarkan data yang bersih dan terpercaya.

C.5 Penghapusan Kolom Bertipe String

```
print("Di Tampilkan Hasil Penghapusan Kolom Pada Dataset Shop Customer Data :")

#menghapus kolom yang bertipe string
customers = dfs.drop('Gender', axis=1)
customers = customers.drop('Profession', axis=1)

#Buat instance dari kelas IsolationForest dan atur parameter-parameter yang sesuai.
isolation_forest = IsolationForest(n_estimators=100, contamination=0.05, random_state=42)
isolation_forest.fit(customers)
outlier_predictions = isolation_forest.predict(customers)
outlier_indices = np.where(outlier_predictions == -1)[0]

customers['Outlier'] = outlier_predictions == -1
customers.head(10)
```

```
Di Tampilkan Hasil Penghapusan Kolom Pada Dataset Shop Customer Data :
```

	CustomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier
0	1	19.0	15000	39	1	4	False
1	2	21.0	35000	81	3	3	False
2	3	20.0	86000	6	1	1	False
3	4	23.0	59000	77	0	2	False
4	5	31.0	38000	40	2	6	False
5	6	22.0	58000	76	0	2	False
6	7	35.0	31000	6	1	3	False
7	8	23.0	84000	94	1	3	False
8	9	64.0	97000	3	0	3	False
9	10	30.0	98000	72	1	4	False

Dilakukan penghapusan kolom 'Gender' dan 'Profession' pada Dataset Shop Customer Data menggunakan fungsi drop, karena kedua kolom tersebut tidak memiliki peran penting. Dilakukan penggunaan IsolationForest dari sklearn.ensemble untuk pembuatan instance dari kelas untuk dilakukan prediksi outlier dengan menghasilkan nilai prediksi untuk setiap baris dalam dataset.

Jika nilai prediksi adalah -1, baris tersebut dianggap sebagai outlier, dengan pencarian indeks baris yang diprediksi sebagai outlier.

C.6 Penampilan Jumlah Outliers

```
print("Di Tampilkan Jumlah Outliers Pada Dataset Shop Customer Data :")

value_counts = customers['Outlier'].value_counts()
print(value_counts)
```

```
Di Tampilkan Jumlah Outliers Pada Dataset Shop Customer Data :
False    1900
True      100
Name: Outlier, dtype: int64
```

Dilakukan perhitungan dan penampilan jumlah outliners pada Dataset Shop Customer Data dengan menggunakan fungsi value_counts() untuk mengembalikan objek series yang berisi jumlah kemunculan setiap nilai unik dalam kolom. Apabila

dalam kolom 'Outliers' terdapat 100 nilai True (outlier) dan 1900 nilai False (non-outlier), maka proporsi outlier dalam data adalah sekitar 5% dari total data. Ini dapat dihitung dengan membagi jumlah outlier dengan jumlah total data sebagai berikut :

Dengan Proporsi Outlier = Jumlah Outlier / Jumlah Total Data = 100/(100+ 1900) = 0.05 atau 5%. Dari sekitar 5% data tersebut dianggap sebagai outlier. Untuk data yang sebanyak 1900 data (non-outlier) dianggap normal atau tidak terklasifikasi sebagai outlier oleh model Isolation Forest.

Kesimpulanya sebesar 95% data dianggap normal atau tidak mengandung nilai yang dianggap sebagai outlier oleh model.

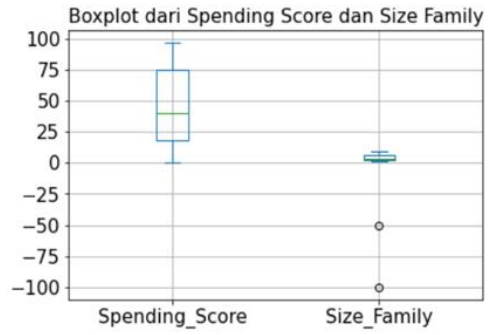
C.7 Boxplot Outlier Dari Spending Score Dan Size Family

```
print("Di Tampilkan Boxplot Dari Spending Score Dan Size Family Pada Dataset Shop Customer Data : ")

df_sample = dfs[["Spending_Score", "Size_Family"]].sample(n=50).copy()
df_sample.loc[5, 'Size_Family'] = -100.0
df_sample.loc[15, 'Size_Family'] = -50.0
df_sample.plot.box()

plt.title("Boxplot dari Spending Score dan Size Family", fontsize=15)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.grid(True)
plt.show()
```

```
Di Tampilkan Boxplot Dari Spending Score Dan Size Family Pada Dataset Shop Customer Data :
```



Dilakukan penampilan boxplot dari spending score dan size family pada dataset Shop Customer Data dengan menggunakan 50 sampel acak dari kolom "Spending_Score" dan "Size Family dataset. Dengan menjalankan kode di atas, kita dapat melihat boxplot dari kolom "Spending_Score" dan "Size_Family" dan memberikan informasi tentang distribusi, nilai-nilai ekstrim, serta adanya outlier dalam kedua kolom tersebut.

Dari proses ini dapat di temukan informasi yang relevan mengenai pengecekan dan penanganan nilai missing value, pengecekan data duplikat dan perhitungan dan penangannan jumlah outlier sehingga kolom-kolom tersebut sudah siap untuk di

analisis dan di lakukan perhitungan matematika secara lebih lanjut, di tahapan transformasi data.

D. Transformasi Data

Tahapan keempat dalam proses data wrangling yakni transformasi data, guna mengubahnya menjadi bentuk yang lebih sesuai untuk analisis yang akan dilakukan. Hal ini bertujuan untuk menggambarkan informasi tambahan, menyederhanakan data, dan membuat data lebih jelas dan mudah dipahami. Proses transformasi ini penting dalam mempersiapkan data untuk pembuatan visualisasi, karena data yang telah diubah atau diolah dengan benar dapat mempermudah dalam menciptakan visualisasi dan keputusan informasi yang efektif dan informatif mengenai dataset.

D.1 Kategori Keluarga Berdasarkan Size Family

```
print("Di Tampilkan Kategori Keluarga Berdasarkan Size Family Pada Dataset Shop Customer Data :")

def categorize_family_size(size):
    if size <= 3:
        return 'Kecil'
    elif size <= 6:
        return 'Sedang'
    else:
        return 'Besar'

customers['Family_Size_Category'] = customers['Size_Family'].apply(categorize_family_size)
customers['Family_Size_Category']
```

```
Di Tampilkan Kategori Keluarga Berdasarkan Size Family Pada Dataset Shop Customer Data :

0       Sedang
1        Kecil
2        Kecil
3        Kecil
4       Sedang
         ...
1995     Besar
1996     Besar
1997     Kecil
1998     Kecil
1999     Kecil
Name: Family_Size_Category, Length: 2000, dtype: object
```

Dilakukan proses untuk mengkategorikan ukuran keluarga berdasarkan nilai pada kolom Size Family. Jika nilai ukuran kurang dari atau sama dengan 3 maka keluarga akan dikategorikan sebagai “kecil”, jika nilai ukuran kurang dari atau sama dengan 6 maka akan dikategorikan sebagai “sedang”, dan jika nilai ukuran lebih dari 6 maka akan dikategorikan sebagai “besar”.

Hasilnya akan ditambahkan pada kolom Family_Size_Category berupa nilai yang merepresentasikan ukuran keluarga hal ini di pergunakan untuk pendukung penarikan hasil yang di peroleh dari dataset.

D.2 Penambahan Kolom Expenditure Income Rasio

```
print("Di Tampilkan Penambahan Kolom Expenditure Income Ratio Pada Dataset Shop Customer Data :")

customers['Expenditure_Income_Ratio'] = customers['Size_Family'] / customers['Annual_Income']
customers.head(10)
```

```
Di Tampilkan Penambahan Kolom Expenditure Income Ratio Pada Dataset Shop Customer Data :
```

	CustomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio
0	1	19.0	15000	39	1	4	False	Sedang	0.000267
1	2	21.0	35000	81	3	3	False	Kecil	0.000086
2	3	20.0	86000	6	1	1	False	Kecil	0.000012
3	4	23.0	59000	77	0	2	False	Kecil	0.000034
4	5	31.0	38000	40	2	6	False	Sedang	0.000158
5	6	22.0	58000	76	0	2	False	Kecil	0.000034
6	7	35.0	31000	6	1	3	False	Kecil	0.000097
7	8	23.0	84000	94	1	3	False	Kecil	0.000036
8	9	64.0	97000	3	0	3	False	Kecil	0.000031
9	10	30.0	98000	72	1	4	False	Sedang	0.000041

Dilakukan proses untuk penambahan kolom dengan melakukan operasi pembagian. Hasilnya akan menampilkan 10 baris pertama dari dataset dengan penambahan kolom 'Expenditure_Income_Ratio' yang berisi nilai rasio pengeluaran terhadap pendapatan yang dihitung berdasarkan hasil pembagian dengan nilai dalam kolom 'Size_Family' dibagi dengan nilai dalam kolom 'Annual_Income' yang di pergunakan untuk melakukan pengelompokkan tingkat pengeluaran.

D.3 Penambahan Kolom Size Family Scaled

```
print("Di Tampilkan Penambahan Kolom Size Family Scaled Pada Dataset Shop Customer Data :")

scaler = StandardScaler()
customers['Size_Family_Scaled'] = scaler.fit_transform(customers[['Size_Family']])

customers.head()
```

```
Di Tampilkan Penambahan Kolom Size Family Scaled Pada Dataset Shop Customer Data :
```

tomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio	Size_Family_Scaled
1	19.0	15000	39	1	4	False	Sedang	0.000267	0.117497
2	21.0	35000	81	3	3	False	Kecil	0.000086	-0.390051
3	20.0	86000	6	1	1	False	Kecil	0.000012	-1.405148
4	23.0	59000	77	0	2	False	Kecil	0.000034	-0.897599
5	31.0	38000	40	2	6	False	Sedang	0.000158	1.132594

Dilakukan penambahan kolom dengan mengoperasikan data dalam dataset untuk melakukan penskalaan data pada kolom Size_Family menggunakan metode StandardScaler. Hasilnya akan ditampilkan kolom baru yakni Size_Family_Scaled yang di pergunakan untuk membandingkan, menganalisis, membuat model, dan memvisualisasikan data yang terkait dengan kolom Size_Family yang telah dinormalisasi. Dengan melakukan normalisasi, data pada kolom Size_Family diubah

menjadi skala yang seragam sehingga lebih mudah digunakan dalam berbagai analisis dan pemodelan.

D.4 Penampilan Nama Kolom Setelah Proses Penghapusan Dan Penambahan

```
print("Di Tampilkan Nama Kolom Setelah Proses Penghapusan Dan Penambahan Pada Dataset Shop Customer Data :")

print(customers.columns)
```

```
Di Tampilkan Nama Kolom Setelah Proses Penghapusan Dan Penambahan Pada Dataset Shop Customer Data :
Index(['CustomerID', 'Age', 'Annual_Income', 'Spending_Score',
       'Work_Experience', 'Size_Family', 'Outlier', 'Family_Size_Category',
       'Expenditure_Income_Ratio', 'Size_Family_Scaled'],
      dtype='object')
```

Dilakukan proses untuk menampilkan nama-nama kolom setelah dilakukan penghapusan kolom dari Gender dan Profession. Selain itu, dilakukan penambahan beberapa kolom baru yaitu, Family Size Category, dan Expenditure Income Ratio.

Jadi terdapat kolom CustomerID,Age,Annual_Income,Spending_Score, Work_Experience,Size_Family,Outlier,Family_Size_Category,Expenditure_Income_ Ratio,dan Size_Family_Scaled.

D.5 Penampilan Rata-Rata Gaji Tahunan, Maksimal, dan Minimal

```
ai = customers['Annual_Income']

print("Di Tampilkan Rata-Rata Gaji: ", ai.mean())
print("Di Tampilkan Gaji Tertinggi: ", ai.max())
print("Di Tampilkan Gaji Terendah: ", ai.min())
```

```
Di Tampilkan Rata-Rata Gaji:  110731.8215
Di Tampilkan Gaji Tertinggi:  189974
Di Tampilkan Gaji Terendah:  0
```

Di lakukan proses penampilan gaji rata-rata, maksimal, dan minimal dari kolom annual income sebagai statistik yang berguna untuk mendapatkan informasi mengenai keterkaitannya dengan output dari dataset yang di hasilkan.

D.6 Penampilan Jumlah Keluarga Berdasarkan Kategori Ukuran Keluarga

```
print("Di Tampilkan Jumlah Keluarga Berdasarkan Kategori Ukuran Keluarga Pada Dataset Shop Customer Data :")

Family_Size_Counts = customers['Family_Size_Category'].value_counts()

print(Family_Size_Counts)
```

```
Di Tampilkan Jumlah Keluarga Berdasarkan Kategori Ukuran Keluarga Pada Dataset Shop Customer Data :
Kecil     971
Sedang    790
Besar     239
Name: Family_Size_Category, dtype: int64
```

Di lakukan proses untuk menampilkan jumlah keluarga dengan menggunakan value_counts() untuk menghitung jumlah keluarga dalam setiap kategori ukuran

keluarga yang terdapat pada kolom Family Size Category. Dengan hasil terdapat 971 keluarga dengan kategori kecil, 790 keluarga dengan kategori sedang, dan 239 keluarga dengan kategori besar.

D.7 Pengelompokkan Tingkat Pengeluaran

```
print("DI Tampilkan Pengelompokkan Tingkatan Pengeluaran Pada Dataset Shop Customer Data :")

# Membuat fungsi untuk mengelompokkan tingkatan pengeluaran
def expenditure_category(row):
    if row['Family_Size_Category'] == 'Kecil':
        if row['Expenditure_Income_Ratio'] < 0.2:
            return 'Rendah'
        elif row['Expenditure_Income_Ratio'] < 0.4:
            return 'Sedang'
        else:
            return 'Tinggi'
    elif row['Family_Size_Category'] == 'Sedang':
        if row['Expenditure_Income_Ratio'] < 0.3:
            return 'Rendah'
        elif row['Expenditure_Income_Ratio'] < 0.6:
            return 'Sedang'
        else:
            return 'Tinggi'
    elif row['Family_Size_Category'] == 'Besar':
        if row['Expenditure_Income_Ratio'] < 0.4:
            return 'Rendah'
        elif row['Expenditure_Income_Ratio'] < 0.7:
            return 'Sedang'
        else:
            return 'Tinggi'

# Menerapkan fungsi pada setiap baris data
customers['Expenditure_Category'] = customers.apply(expenditure_category, axis=1)

Expenditure_Category_Counts = customers['Expenditure_Category'].value_counts()

print(Expenditure_Category_Counts)
```

```
DI Tampilkan Pengelompokkan Tingkatan Pengeluaran Pada Dataset Shop Customer Data :
Rendah    1998
Tinggi       2
Name: Expenditure_Category, dtype: int64
```

Dilakukan proses untuk mengelompokkan tingkatan pengeluaran berdasarkan kategori ukuran keluarga dan rasio pengeluaran terhadap pendapatan. Terdapat beberapa kondisi untuk menentukan tingkatan pengeluaran berdasarkan nilai pada kolom Family Size dan Expenditure Income Ratio.

Pada kategori ukuran keluarga “kecil” akan dilakukan pengujian, jika nilai kurang dari 0.2 maka akan dikategorikan sebagai “rendah”, jika nilai kurang dari 0.4 maka akan dikategorikan sebagai “sedang”, dan jika tidak memenuhi kedua kondisi maka akan dikategorikan sebagai “tinggi”. Pada kategori ukuran keluarga “sedang” akan dilakukan pengujian, jika nilai kurang dari 0.3 maka akan dikategorikan sebagai “rendah”, jika nilai kurang dari 0.6 maka akan dikategorikan sebagai “sedang”, dan jika tidak memenuhi kedua kondisi maka akan dikategorikan sebagai “tinggi”. Pada kategori ukuran keluarga “kecil” akan dilakukan pengujian, jika nilai kurang dari 0.4

maka akan dikategorikan sebagai “rendah”, jika nilai kurang dari 0.7 maka akan dikategorikan sebagai “sedang”, dan jika tidak memenuhi kedua kondisi maka akan dikategorikan sebagai “tinggi”.

Hasilnya akan ditampilkan berupa deretan tingkatan pengeluaran dan jumlah kemunculan setiap tingkatan serta hasil dari kategori tersebut ditambahkan dalam kolom Expenditure Category dengan 1998 memiliki pengeluaran rendah dan 2 memiliki pengeluaran tinggi.

D.8 Penampilan DataFrame Dari Dataset Yang Telah Di Proses

```
customers
```

	CustomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio
0	1	19.0	15000	39	1	4	False	Sedang	0.000267
1	2	21.0	35000	81	3	3	False	Kecil	0.000086
2	3	20.0	86000	6	1	1	False	Kecil	0.000012
3	4	23.0	59000	77	0	2	False	Kecil	0.000034
4	5	31.0	38000	40	2	6	False	Sedang	0.000158
...	...	...	...	...	...	...	...	...	...
1995	1996	71.0	184387	40	8	7	True	Besar	0.000038
1996	1997	91.0	73158	32	7	7	False	Besar	0.000096
1997	1998	87.0	90961	14	9	2	False	Kecil	0.000022
1998	1999	77.0	182109	4	7	2	True	Kecil	0.000011
1999	2000	90.0	110610	52	5	2	False	Kecil	0.000018

2000 rows × 11 columns

```
customers
```

ncome	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio	Size_Family_Scaled	Expenditure_Category
15000	39	1	4	False	Sedang	0.000267	0.117497	Rendah
35000	81	3	3	False	Kecil	0.000086	-0.390051	Rendah
86000	6	1	1	False	Kecil	0.000012	-1.405148	Rendah
59000	77	0	2	False	Kecil	0.000034	-0.897599	Rendah
38000	40	2	6	False	Sedang	0.000158	1.132594	Rendah
...	...	...	...	...	...	...	...	...
184387	40	8	7	True	Besar	0.000038	1.640142	Rendah
73158	32	7	7	False	Besar	0.000096	1.640142	Rendah
90961	14	9	2	False	Kecil	0.000022	-0.897599	Rendah
182109	4	7	2	True	Kecil	0.000011	-0.897599	Rendah
110610	52	5	2	False	Kecil	0.000018	-0.897599	Rendah

Dilakukan proses untuk menampilkan data frame dari dataset yang telah diproses sebelumnya. Kolom-kolom yang ditampilkan yakni CustomerID, Age, Annual Income, Spending Score, Work Experience, Size Family, Outlier, Total Members, Family Size Category, Expenditure Income Ratio, Size Family Scaled, dan Expenditure category.

Dari proses ini di peroleh informasi yang relevan mengenai jumlah kategori keluarga, nilai expenditure rasio yang akan di kelompokkan pada tingkat pengeluaran,

informasi statistik yang berguna dari kolom annual income, dan di peroleh dataset yang telah di proses untuk di lakukan visualisasi data.

E. Visualisasi Data

Tahapan kelima dalam proses data wrangling yakni memvisualisasikan data guna memahami pola dan hubungan yang mungkin terdapat di dalamnya. Visualisasi data memungkinkan kita untuk menciptakan grafik, diagram, atau bagan yang memberikan kemudahan dalam menganalisis dan menyajikan informasi.

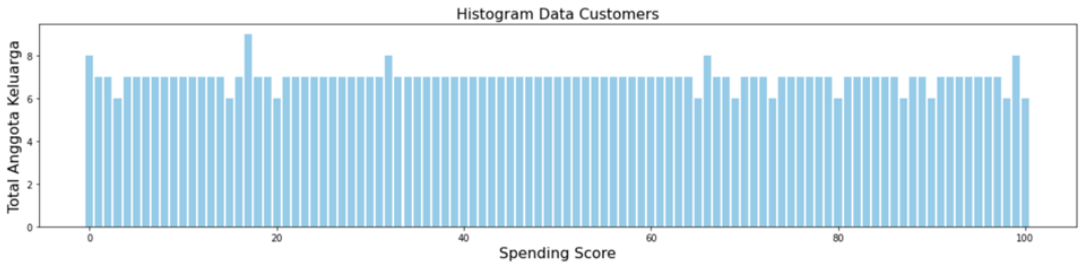
Melalui penggunaan grafik atau bagan, kita dapat dengan jelas melihat perbandingan antara data, melihat sebaran nilai dalam dataset, serta memvisualisasikan pola yang muncul. Hal ini bertujuan untuk mengubah data yang telah melalui proses pengolahan menjadi bentuk visual yang mudah dibaca dan dipahami. Dengan menggunakan visualisasi yang tepat, kita dapat menyampaikan informasi hasil analisis data dengan lebih efektif.

E.1 Penampilan Histogram Data Customers

```
print("Di Tampilkan Histogram Data Customers Pada Dataset Shop Customers Data :")

plt.figure(figsize=(20, 4))
plt.title("Histogram Data Customers", fontsize=16)
plt.bar(customers['Spending_Score'], height=customers['Size_Family'], color='skyblue')
plt.xlabel("Spending Score", fontsize=16)
plt.ylabel("Total Anggota Keluarga", fontsize=16)
plt.show()
```

```
Di Tampilkan Histogram Data Customers Pada Dataset Shop Customers Data :
```



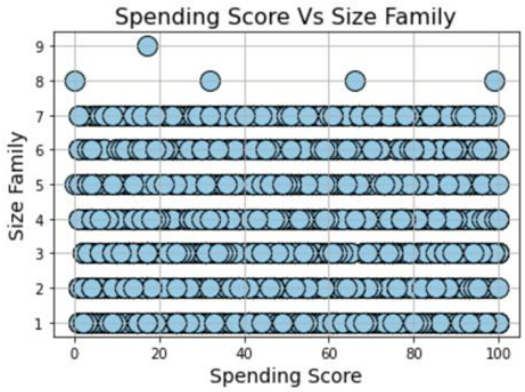
Dilakukan proses untuk menampilkan histogram dari data customers untuk melihat pola dan distribusi data pada kolom Spending Score dan Size Family. Histogram ini menampilkan distribusi pada sumbu x (Spending Score) dan jumlah anggota keluarga pada sumbu y (Size Family). Setiap batang dari histogram menunjukkan frekuensi atau jumlah data pada rentang tertentu.

E.2 Penampilan Scatter Plot Dari Spending Score Vs Size Family

```
print("Di Tampilkan Scatter Plot Dari Spending Score Vs Size Family :")

customers.plot.scatter('Spending_Score', 'Size_Family', s=200,\
                       c='skyblue',edgecolor='k')
plt.grid(True)
plt.title('Spending Score Vs Size Family',fontsize=16)
plt.xlabel('Spending Score',fontsize=14)
plt.ylabel('Size Family',fontsize=14)
plt.show()
```

```
Di Tampilkan Scatter Plot Dari Spending Score Vs Size Family :
```



Dilakukan proses untuk menampilkan scatter plot untuk melihat pola atau hubungan dari kolom Spending Score dan Size Family. Setiap titik yang berwarna biru merepresentasikan data pada baris tertentu dalam dataset. Posisi titik pada sumbu x dan sumbu y menunjukkan nilai dari Spending Score dan Size Family dari masing-masing data.

Dari proses ini di peroleh informasi mengenai visualisasi histogram dan scatter plot antara kolom spending score dengan kolom size family dan dapat di peroleh hasil bahwa dalam dataset yang dianalisis, tidak terdapat hubungan yang konsisten antara ukuran keluarga (Size Family) dengan skor pengeluaran (Spending Score). Meskipun ukuran keluarga dapat berbeda-beda, tidak selalu berarti bahwa keluarga dengan ukuran yang lebih besar akan memiliki skor pengeluaran yang lebih tinggi. Hal ini menunjukkan bahwa faktor-faktor lain juga mempengaruhi tingkat pengeluaran keluarga.

F. Python Dan SQL

Tahapan keenam dalam proses data wrangling, di perlukan proses penghubungan atau pembuatan koneksi antara dataset yang telah diproses ke database yang disimpan dalam SQL. Hal ini bertujuan untuk mengetahui hasil dataset yang telah

diproses dan mencoba untuk melakukan manipulasi data berupa CRUD (Create, Read, Update, Delete) di dalam python yang terhubung dengan SQL.

Dengan menghubungkan dataset yang telah diproses ke dalam database SQL, kita dapat memanfaatkan fleksibilitas SQL dalam mengelola dan menganalisis data secara efisien. Melalui operasi CRUD, dapat membuat entry baru (Create), membaca data yang ada (Read), memperbarui data yang ada (Update), atau menghapus data yang tidak diperlukan (Delete) melalui Python.

F.1 Proses Create Database

```
import sqlite3

data_entry = pd.DataFrame(customers)

#Membuat koneksi SQL ke database SQLite
con = sqlite3.connect("customers_data.db", timeout=10)

#Menyimpan DataFrame ke tabel 'df_fit' dalam database
data_entry.to_sql('customers', con, if_exists='replace', index=False)

print("Database dengan nama customers_data.db berhasil di buat")

#Menutup koneksi
con.close()
```

```
Database dengan nama customers_data.db berhasil di buat
```

Dilakukan proses create (pembuatan) database dari dataset customers yang telah diproses sebelumnya dengan database bernama customers_data.db. Di lakukan penampilan dataset yang telah di buat di dalam SQL yang selanjutnya akan dilakukan proses CRUD.

	CustomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio	Size_Family_Scaled	Expenditure_Category
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1975	1975	14.0	153145	59	8	6	0	Sedang	3.91785562702014e-05	1.13259407642617	Rendah
1976	1976	41.0	128960	43	9	4	0	Sedang	3.10173697270471e-05	0.117497436115912	Rendah
1977	1977	60.0	127438	82	7	2	0	Kecil	1.56939060562783e-05	-0.897599204194342	Rendah
1978	1978	60.0	125968	100	8	2	0	Kecil	1.587704813921e-05	-0.897599204194342	Rendah
1979	1979	84.0	104589	85	10	2	0	Kecil	1.91224698582069e-05	-0.897599204194342	Rendah
1980	1980	0.0	165321	93	8	1	1	Kecil	6.04883832060053e-06	-1.40514752434947	Rendah
1981	1981	10.0	86925	76	7	2	0	Kecil	2.30083405234397e-05	-0.897599204194342	Rendah
1982	1982	62.0	149797	19	5	6	0	Sedang	4.00542066930579e-05	1.13259407642617	Rendah
1983	1983	33.0	137094	68	4	1	0	Kecil	7.2942652486615e-06	-1.40514752434947	Rendah
1984	1984	52.0	55395	41	10	1	0	Kecil	1.80521707735355e-05	-1.40514752434947	Rendah
1985	1985	2.0	153622	51	6	6	0	Sedang	3.90569059119137e-05	1.13259407642617	Rendah
1986	1986	27.0	74050	44	8	1	0	Kecil	1.35043889264011e-05	-1.40514752434947	Rendah
1987	1987	4.0	68094	61	4	7	1	Besar	0.000102799071871237	1.64014239658129	Rendah
1988	1988	63.0	59244	80	7	1	0	Kecil	1.68793464317062e-05	-1.40514752434947	Rendah
1989	1989	54.0	118944	77	4	4	0	Sedang	3.36292709174065e-05	0.117497436115912	Rendah
1990	1990	47.0	75293	55	6	7	0	Besar	9.29701300253676e-05	1.64014239658129	Rendah
1991	1991	30.0	166983	69	7	3	0	Kecil	1.79659007204326e-05	-0.390050884039215	Rendah
1992	1992	97.0	129444	96	5	6	1	Sedang	4.63520904792806e-05	1.13259407642617	Rendah
1993	1993	94.0	181183	24	9	3	0	Kecil	1.65578448309168e-05	-0.390050884039215	Rendah
1994	1994	64.0	175254	100	9	5	1	Sedang	2.853001928629 3e-05	0.625045756271039	Rendah
1995	1995	19.0	54121	89	6	3	0	Kecil	5.54313482751612e-05	-0.390050884039215	Rendah
1996	1996	71.0	184387	40	8	7	1	Besar	3.79636308416537e-05	1.64014239658129	Rendah
1997	1997	91.0	73158	32	7	7	0	Besar	9.56833155635747e-05	1.64014239658129	Rendah
1998	1998	87.0	90961	14	9	2	0	Kecil	2.19874451688086e-05	-0.897599204194342	Rendah
1999	1999	77.0	182109	4	7	2	1	Kecil	1.09824335974609e-05	-0.897599204194342	Rendah
2000	2000	90.0	110610	52	5	2	0	Kecil	1.808154778049e-05	-0.897599204194342	Rendah

F.2 Proses Penambahan Data

```
import sqlite3

con = sqlite3.connect("customers_data.db", timeout = 10)
cur = con.cursor()

for row in cur.execute("INSERT INTO customers VALUES (2001, 46.0, 114435, 34, 8, 3, 0, 'Kecil', 2.62157556691572e-05,\
                       -0.390050884039215, 'Rendah')"):
    print(row)
con.commit()

print("1 data berhasil di tambahkan")

con.close()
```

```
1 data berhasil di tambahkan
```

Dilakukan proses penambahan data menggunakan bahasa pemrograman python yang terhubung di SQL, di tambahkan data pada baris ke 2001. Dan dilakukan penampilan hasil data yang telah ditambahkan. Awalnya dataset memiliki data sebanyak 2000 baris dan setelah dilakukan penambahan dataset menjadi 2001 baris.

	CustomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio	Size_Family_Scaled	Expenditure_Category
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1976	1976	41.0	128960	43	9	4	0	Sedang	3.10173697270471e-05	0.117497436115912	Rendah
1977	1977	60.0	127438	82	7	2	0	Kecil	1.56939060562783e-05	-0.89759920419434 2	Rendah
1978	1978	60.0	125968	100	8	2	0	Kecil	1.587704813921e-05	-0.89759920419434 2	Rendah
1979	1979	84.0	104589	85	10	2	0	Kecil	1.91224698582069e-05	-0.89759920419434 2	Rendah
1980	1980	0.0	165321	93	8	1	1	Kecil	6.04883832060053e-06	-1.40514752434947	Rendah
1981	1981	10.0	86925	76	7	2	0	Kecil	2.30083405234397e-05	-0.89759920419434 2	Rendah
1982	1982	62.0	149797	19	5	6	0	Sedang	4.00542066930579e-05	1.13259407642617	Rendah
1983	1983	33.0	137094	68	4	1	0	Kecil	7.2942652486615e-06	-1.40514752434947	Rendah
1984	1984	52.0	55395	41	10	1	0	Kecil	1.80521707735355e-05	-1.40514752434947	Rendah
1985	1985	2.0	153622	51	6	6	0	Sedang	3.90569059119137e-05	1.13259407642617	Rendah
1986	1986	27.0	74050	44	8	1	0	Kecil	1.35043889264011e-05	-1.40514752434947	Rendah
1987	1987	4.0	68094	61	4	7	1	Besar	0.00010279907 1871237	1.64014239658129	Rendah
1988	1988	63.0	59244	80	7	1	0	Kecil	1.68793464317062e-05	-1.40514752434947	Rendah
1989	1989	54.0	118944	77	4	4	0	Sedang	3.36292709174065e-05	0.117497436115912	Rendah
1990	1990	47.0	75293	55	6	7	0	Besar	9.29701300253676e-05	1.64014239658129	Rendah
1991	1991	30.0	166983	69	7	3	0	Kecil	1.79659007204326e-05	-0.390050884039215	Rendah
1992	1992	97.0	129444	96	5	6	1	Sedang	4.63520904792806e-05	1.13259407642617	Rendah
1993	1993	94.0	181183	24	9	3	0	Kecil	1.65578448309168e-05	-0.390050884039215	Rendah
1994	1994	64.0	175254	100	9	5	1	Sedang	2.8530019286293e-05	0.625045756271039	Rendah
1995	1995	19.0	54121	89	6	3	0	Kecil	5.54313482751612e-05	-0.390050884039215	Rendah
1996	1996	71.0	184387	40	8	7	1	Besar	3.79636308416537e-05	1.64014239658129	Rendah
1997	1997	91.0	73158	32	7	7	0	Besar	9.56833155635747e-05	1.64014239658129	Rendah
1998	1998	87.0	90961	14	9	2	0	Kecil	2.19874451688086e-05	-0.89759920419434 2	Rendah
1999	1999	77.0	182109	4	7	2	1	Kecil	1.09824335974609e-05	-0.89759920419434 2	Rendah
2000	2000	90.0	110610	52	5	2	0	Kecil	1.808154778049e-05	-0.89759920419434 2	Rendah
2001	2001	46.0	114435	34	8	3	0	Kecil	2.62157556691572e-05	-0.390050884039215	Rendah

F.3 Proses Update Data

```
con = sqlite3.connect("customers_data.db", timeout=10)

cur = con.cursor()

for row in cur.execute("UPDATE customers SET Age = 25.0 WHERE CustomerID = 2001"):
    print(row)
con.commit()

print("1 data pada kolom Age dengan CustomerID 2001 berhasil di update")

con.close()
```

```
1 data pada kolom Age dengan CustomerID 2001 berhasil di update
```

Dilakukan proses update (pembaruan) data menggunakan bahasa pemrograman python yang terhubung di SQL pada umur pelanggan yang memiliki CustomerID 2001.

Dan di tampilkan hasil update (pembaruan) umur pelanggan menjadi 25.0 di CustomerID 2001.

```
print("Di Tampilkan Data Dari Dataset Pada SQL :")

con = sqlite3.connect("customers_data.db", timeout=10)
cur = con.cursor()

for row in cur.execute("SELECT * FROM customers"):
    print(row)
con.commit()

con.close()
```

```
(1983, 33.0, 137094, 68, 4, 1, 0, 'Kecil', 7.294265248661502e-06, -1.4051475243494693, 'Rendah')
(1984, 52.0, 55395, 41, 10, 1, 0, 'Kecil', 1.8052170773535517e-05, -1.4051475243494693, 'Rendah')
(1985, 2.0, 153622, 51, 6, 6, 0, 'Sedang', 3.905690591191366e-05, 1.1325940764261662, 'Rendah')
(1986, 27.0, 74050, 44, 8, 1, 0, 'Kecil', 1.350438892640108e-05, -1.4051475243494693, 'Rendah')
(1987, 4.0, 68094, 61, 4, 7, 1, 'Besar', 0.00010279907187123682, 1.6401423965812933, 'Rendah')
(1988, 63.0, 59244, 80, 7, 1, 0, 'Kecil', 1.6879346431706163e-05, -1.4051475243494693, 'Rendah')
(1989, 54.0, 118944, 77, 4, 4, 0, 'Sedang', 3.362927091740651e-05, 0.11749743611591194, 'Rendah')
(1990, 47.0, 75293, 55, 6, 7, 0, 'Besar', 9.297013002536756e-05, 1.6401423965812933, 'Rendah')
(1991, 30.0, 166983, 69, 7, 3, 0, 'Kecil', 1.7965900720432618e-05, -0.3900508840392152, 'Rendah')
(1992, 97.0, 129444, 96, 5, 6, 1, 'Sedang', 4.6352090479280616e-05, 1.1325940764261662, 'Rendah')
(1993, 94.0, 181183, 24, 9, 3, 0, 'Kecil', 1.6557844830916807e-05, -0.3900508840392152, 'Rendah')
(1994, 64.0, 175254, 100, 9, 5, 1, 'Sedang', 2.8530019286293038e-05, 0.6250457562710391, 'Rendah')
(1995, 19.0, 54121, 89, 6, 3, 0, 'Kecil', 5.543134827516121e-05, -0.3900508840392152, 'Rendah')
(1996, 71.0, 184387, 40, 8, 7, 1, 'Besar', 3.79636308416537e-05, 1.6401423965812933, 'Rendah')
(1997, 91.0, 73158, 32, 7, 7, 0, 'Besar', 9.568331556357473e-05, 1.6401423965812933, 'Rendah')
(1998, 87.0, 90961, 14, 9, 2, 0, 'Kecil', 2.198744516880861e-05, -0.8975992041943422, 'Rendah')
(1999, 77.0, 182109, 4, 7, 2, 1, 'Kecil', 1.0982433597460861e-05, -0.8975992041943422, 'Rendah')
(2000, 90.0, 110610, 52, 5, 2, 0, 'Kecil', 1.808154778049001e-05, -0.8975992041943422, 'Rendah')
(2001, 25.0, 114435, 34, 8, 3, 0, 'Kecil', 2.62157556691572e-05, -0.390050884039215, 'Rendah')
```

F.4 Proses Delete Data

```
import sqlite3

con = sqlite3.connect("customers_data.db", timeout=10)
cur = con.cursor()

cur.execute("DELETE FROM customers WHERE CustomerID = 2001")
con.commit()

print("1 data dengan CustomerID 2001 berhasil dihapus")

con.close()
```

```
1 data dengan CustomerID 2001 berhasil dihapus
```

Dilakukan proses delete (penghapusan) data menggunakan bahasa pemrograman python yang terhubung di SQL di data yang memiliki CustomerID 2001 sebagai kata kunci. Dan di tampilkan data yang mulanya 2001 menjadi 2000.

	CustomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio	Size_Family_Scaled	Expenditure_Category
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1975	1975	14.0	153145	59	8	6	0	Sedang	3.91785562702014e-05	1.13259407642617	Rendah
1976	1976	41.0	128960	43	9	4	0	Sedang	3.10173697270471e-05	0.117497436115912	Rendah
1977	1977	60.0	127438	82	7	2	0	Kecil	1.56939060562783e-05	-0.897599204194342	Rendah
1978	1978	60.0	125968	100	8	2	0	Kecil	1.587704813921e-05	-0.897599204194342	Rendah
1979	1979	84.0	104589	85	10	2	0	Kecil	1.91224698582069e-05	-0.897599204194342	Rendah
1980	1980	0.0	165321	93	8	1	1	Kecil	6.04883832060053e-06	-1.40514752434947	Rendah
1981	1981	10.0	86925	76	7	2	0	Kecil	2.30083405234397e-05	-0.897599204194342	Rendah
1982	1982	62.0	149797	19	5	6	0	Sedang	4.00542066930579e-05	1.13259407642617	Rendah
1983	1983	33.0	137094	68	4	1	0	Kecil	7.2942652486615e-06	-1.40514752434947	Rendah
1984	1984	52.0	55395	41	10	1	0	Kecil	1.80521707735355e-05	-1.40514752434947	Rendah
1985	1985	2.0	153622	51	6	6	0	Sedang	3.90569059119137e-05	1.13259407642617	Rendah
1986	1986	27.0	74050	44	8	1	0	Kecil	1.35043889264011e-05	-1.40514752434947	Rendah
1987	1987	4.0	68094	61	4	7	1	Besar	0.000102799071871237	1.64014239658129	Rendah
1988	1988	63.0	59244	80	7	1	0	Kecil	1.68793464317062e-05	-1.40514752434947	Rendah
1989	1989	54.0	118944	77	4	4	0	Sedang	3.36292709174065e-05	0.117497436115912	Rendah
1990	1990	47.0	75293	55	6	7	0	Besar	9.29701300253676e-05	1.64014239658129	Rendah
1991	1991	30.0	166983	69	7	3	0	Kecil	1.79659007204326e-05	-0.390050884039215	Rendah
1992	1992	97.0	129444	96	5	6	1	Sedang	4.63520904792806e-05	1.13259407642617	Rendah
1993	1993	94.0	181183	24	9	3	0	Kecil	1.65578448309168e-05	-0.390050884039215	Rendah
1994	1994	64.0	175254	100	9	5	1	Sedang	2.8530019286293e-05	0.625045756271039	Rendah
1995	1995	19.0	54121	89	6	3	0	Kecil	5.54313482751612e-05	-0.390050884039215	Rendah
1996	1996	71.0	184387	40	8	7	1	Besar	3.79636308416537e-05	1.64014239658129	Rendah
1997	1997	91.0	73158	32	7	7	0	Besar	9.56833155635747e-05	1.64014239658129	Rendah
1998	1998	87.0	90961	14	9	2	0	Kecil	2.19874451688086e-05	-0.897599204194342	Rendah
1999	1999	77.0	182109	4	7	2	1	Kecil	1.09824335974609e-05	-0.897599204194342	Rendah
2000	2000	90.0	110610	52	5	2	0	Kecil	1.808154778049e-05	-0.897599204194342	Rendah

F.5 Pengelompokan Berdasarkan Ukuran Keluarga

```
import sqlite3

print("Di tampilkan hasil pengelompokkan data berdasarkan Family Size Category : ")

con = sqlite3.connect("customers_data.db", timeout = 10)
cur = con.cursor()

for row in cur.execute("SELECT COUNT(*), Family_Size_Category FROM customers Group BY Family_Size_Category"):
    print(row)
con.commit()

con.close()
```

```
Di tampilkan hasil pengelompokkan data berdasarkan Family Size Category :
(239, 'Besar')
(971, 'Kecil')
(790, 'Sedang')
```

Dilakukan proses pengelompokan pada dataset berdasarkan ukuran keluarga. Dan di tampilkan bahwa 239 memiliki ukuran keluarga besar, 971 memiliki ukuran keluarga kecil, dan 790 memiliki ukuran keluarga sedang.

F.6 Pengelompokan Berdasarkan Kategori Pengeluaran

```
import sqlite3

print("Di tampilkan hasil pengelompokkan data berdasarkan Expenditure Category : ")

con = sqlite3.connect("customers_data.db", timeout = 10)
cur = con.cursor()

for row in cur.execute("SELECT COUNT(*), Expenditure_Category FROM customers Group BY Expenditure_Category"):
    print(row)
con.commit()

con.close()
```

```
Di tampilkan hasil pengelompokkan data berdasarkan Expenditure Category :
(1998, 'Rendah')
(2, 'Tinggi')
```

Dilakukan proses pengelompokan pada dataset berdasarkan kategori pengeluaran. Dan di tampilkan bahwa 1998 memiliki ukuran kategori pengeluaran rendah, 2 memiliki kategori pengeluaran tinggi.

Dari proses ini di peroleh informasi bahwa metode CRUD (Create, Read, Update, Delete) untuk proses manipulasi data telah berhasil di lakukan dan dengan hasil akhir terdapat sebanyak 2000 data dengan 11 kolom.

Hasil dan Pembahasan

Pada proses penerapan data wrangling ini, di gunakan dataset bernama “Shop Customer Data” dengan data awal 2000 baris dan 8 kolom. Dataset ini telah di analisis dengan tujuan untuk mengetahui bagaimana ukuran pelanggan mempengaruhi kebiasaan belanja. Misalnya, akan di ketahui apakah pelanggan dengan keluarga yang lebih besar cenderung memiliki pengeluaran yang lebih tinggi atau memilih produk tertentu yang sesuai untuk kelurga besar. Untuk dataset awal dapat di perlihatkan sebagai berikut.

```
print("Di Tampilkan Dataset Shop Customer Data : ")

dfs = pd.read_csv("Shop_Customer_Data.csv")
dfs
```

```
Di Tampilkan Dataset Shop Customer Data :
```

	CustomerID	Gender	Age	Annual Income ($)	Spending Score (1-100)	Profession	Work Experience	Family Size
0	1	Male	19	15000	39	Healthcare	1	4
1	2	Male	21	35000	81	Engineer	3	3
2	3	Female	20	86000	6	Engineer	1	1
3	4	Female	23	59000	77	Lawyer	0	2
4	5	Female	31	38000	40	Entertainment	2	6
...	...	...	...	...	...	...	...	...
1995	1996	Female	71	184387	40	Artist	8	7
1996	1997	Female	91	73158	32	Doctor	7	7
1997	1998	Male	87	90961	14	Healthcare	9	2
1998	1999	Male	77	182109	4	Executive	7	2
1999	2000	Male	90	110610	52	Entertainment	5	2

2000 rows × 8 columns

Dari dataset tersebut di peroleh informasi mengenai kolom apa saja yang terdapat di dalam dataset dan informasi awal mengenai dataset. Kemudian akan di lakukan serangkaian proses data wrangling yang telah memiliki hasil pertahapannya dan akan di jelaskan secara detail, di antaranya yakni :

A. Penemuan Data

Pada tahapan ini di lakukan load dataset Shop Customer Data dengan tujuan untuk mengetahui dataset apa yang akan di gunakan beserta kolom-kolom yang tersedia pada dataset tersebut sehingga bisa di peroleh informasi awal mengenai dataset dan informasi apa yang akan di hasilkan dari dataset tersebut. Berdasarkan dataset tersebut, di harapkan mendapatkan informasi mengenai bagaimana ukuran keluarga pelanggan mempengaruhi kebiasaan belanja.

B. Pemformatan Atau Perapian Data

Pada tahapan ini di lakukan rename (pengubahan nama) pada kolom-kolom yang tersedia dari dataset dengan tujuan agar lebih mudah di panggil dan di manfaatkan dalam proses ini dan di lakukan pengubahan tipe data di kolom age agar dapat di tuliskan umur pelanggan secara spesifik dan mendetail, sehingga pada tahap ini di peroleh tampilan dataset sebagai berikut :

```
print("Di Tampilkan Dataset Shop Customer Data : ")

dfs
```

```
Di Tampilkan Dataset Shop Customer Data :
```

	CustomerID	Gender	Age	Annual_Income	Spending_Score	Profession	Work_Experience	Size_Family
0	1	Male	19.0	15000	39	Healthcare	1	4
1	2	Male	21.0	35000	81	Engineer	3	3
2	3	Female	20.0	86000	6	Engineer	1	1
3	4	Female	23.0	59000	77	Lawyer	0	2
4	5	Female	31.0	38000	40	Entertainment	2	6
...	...	...	...	...	...	...	...	...
1995	1996	Female	71.0	184387	40	Artist	8	7
1996	1997	Female	91.0	73158	32	Doctor	7	7
1997	1998	Male	87.0	90961	14	Healthcare	9	2
1998	1999	Male	77.0	182109	4	Executive	7	2
1999	2000	Male	90.0	110610	52	Entertainment	5	2

2000 rows × 8 columns

```
print("Di Tampilkan Tipe Data Pada Dataset Shop Customer Data : ")

dfs.dtypes
```

```
Di Tampilkan Tipe Data Pada Dataset Shop Customer Data :

CustomerID          int64
Gender             object
Age               float64
Annual_Income       int64
Spending_Score      int64
Profession         object
Work_Experience     int64
Size_Family         int64
dtype: object
```

Dan di dapatkan informasi bahwa dataset ini memiliki 8 kolom yang telah dengan nama-nama kolom yang telah di lakukan perubahan dengan ukuran dataset terdiri atas 2000 baris dan 8 kolom dengan tipe-tipe data yang sesuai dengan kebutuhan untuk analisis lebih lanjut.

C. Pembersihan Data

Pada tahapan ini di lakukan pengecekan nilai missing value untuk mengetahui apakah terdapat kolom yang memiliki nilai missing value, penggantian nilai missing value dengan “undifined” dan pengecekan data duplikat untuk menjaga kekonsistenan data, serta di lakukan penghapusan kolom Gender dan Profession, dan di lakukan perhitungan outlier serta boxplot outlier dari Spending_Score dan Size_Family.

Di lakukan penghapusan pada kolom Gender dan Profession karena kedua kolom tersebut di anggap tidak memiliki peran kontribusi yang penting pada proses tahapan data wrangling ini. Dan di lakukan perhitungan dan penambahan kolom, Outlier. Berdasarkan kolom Outlier dapat di representasikan bahwa sekitar 95% dari data tersebut di anggap normal atau tidak mengandung nilai yang di anggap sebagai outlier oleh model outlier yang telah di tentukan, sehingga sekitar 5% dari data tersebut di anggap sebagai outlier, sehingga pada tahap ini di peroleh tampilan dataset sebagai berikut :

	CustomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier
0	1	19.0	15000	39	1	4	False
1	2	21.0	35000	81	3	3	False
2	3	20.0	86000	6	1	1	False
3	4	23.0	59000	77	0	2	False
4	5	31.0	38000	40	2	6	False
5	6	22.0	58000	76	0	2	False
6	7	35.0	31000	6	1	3	False
7	8	23.0	84000	94	1	3	False
8	9	64.0	97000	3	0	3	False
9	10	30.0	98000	72	1	4	False

Dengan jumlah penampilan outlier yang di peroleh sebagai berikut :

```
print("Di Tampilkan Jumlah Outliers Pada Dataset Shop Customer Data :")

value_counts = customers['Outlier'].value_counts()
print(value_counts)
```

```
Di Tampilkan Jumlah Outliers Pada Dataset Shop Customer Data :
False    1900
True      100
Name: Outlier, dtype: int64
```

D. Transformasi Data

Pada tahapan ini di lakukan pengkategorian Family_Size_Category, penambahan kolom Expenditure_Income_Ratio, penambahan kolom Size_Family_Scaled, penampilan nama kolom setelah di lakukan proses penghapusan

dan penambahan, penampilan rata-rata gaji tahunan, minimal, dan maksimal, serta di lakukan pengelompokkan tingkat pengeluaran berdasarkan kolom Family_Size_Category dan Expenditure_Income_Ratio, sehingga pada tahap ini di peroleh tampilan dataset yang telah di proses di simpan dengan nama customers sebagai berikut :

```
customers
```

	CustomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio
0	1	19.0	15000	39	1	4	False	Sedang	0.000267
1	2	21.0	35000	81	3	3	False	Kecil	0.000086
2	3	20.0	86000	6	1	1	False	Kecil	0.000012
3	4	23.0	59000	77	0	2	False	Kecil	0.000034
4	5	31.0	38000	40	2	6	False	Sedang	0.000158
...	...	...	...	...	...	...	...	...	...
1995	1996	71.0	184387	40	8	7	True	Besar	0.000038
1996	1997	91.0	73158	32	7	7	False	Besar	0.000096
1997	1998	87.0	90961	14	9	2	False	Kecil	0.000022
1998	1999	77.0	182109	4	7	2	True	Kecil	0.000011
1999	2000	90.0	110610	52	5	2	False	Kecil	0.000018

2000 rows × 11 columns

```
customers
```

ncome	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio	Size_Family_Scaled	Expenditure_Category
15000	39	1	4	False	Sedang	0.000267	0.117497	Rendah
35000	81	3	3	False	Kecil	0.000086	-0.390051	Rendah
86000	6	1	1	False	Kecil	0.000012	-1.405148	Rendah
59000	77	0	2	False	Kecil	0.000034	-0.897599	Rendah
38000	40	2	6	False	Sedang	0.000158	1.132594	Rendah
...	...	...	...	...	...	...	...	...
184387	40	8	7	True	Besar	0.000038	1.640142	Rendah
73158	32	7	7	False	Besar	0.000096	1.640142	Rendah
90961	14	9	2	False	Kecil	0.000022	-0.897599	Rendah
182109	4	7	2	True	Kecil	0.000011	-0.897599	Rendah
110610	52	5	2	False	Kecil	0.000018	-0.897599	Rendah

Dengan nama-nama kolom yang terdapat pada dataset sebagai berikut :

```
print("Di Tampilkan Nama Kolom Setelah Proses Penghapusan Dan Penambahan Pada Dataset Shop Customer Data :")

print(customers.columns)
```

```
Di Tampilkan Nama Kolom Setelah Proses Penghapusan Dan Penambahan Pada Dataset Shop Customer Data :
Index(['CustomerID', 'Age', 'Annual_Income', 'Spending_Score',
       'Work_Experience', 'Size_Family', 'Outlier', 'Family_Size_Category',
       'Expenditure_Income_Ratio', 'Size_Family_Scaled'],
      dtype='object')
```

Dengan pengkategorian keluarga sebagai berikut :

```
print("Di Tampilkan Jumlah Keluarga Berdasarkan Kategori Ukuran Keluarga Pada Dataset Shop Customer Data :")

Family_Size_Counts = customers['Family_Size_Category'].value_counts()

print(Family_Size_Counts)
```

```
Di Tampilkan Jumlah Keluarga Berdasarkan Kategori Ukuran Keluarga Pada Dataset Shop Customer Data :
Kecil     971
Sedang    790
Besar     239
Name: Family_Size_Category, dtype: int64
```

Dengan pengkategorian tingkat pengeluaran sebagai berikut :

```
print("DI Tampilkan Pengelompokkan Tingkatan Pengeluaran Pada Dataset Shop Customer Data :")

# Membuat fungsi untuk mengelompokkan tingkatan pengeluaran
def expenditure_category(row):
    if row['Family_Size_Category'] == 'Kecil':
        if row['Expenditure_Income_Ratio'] < 0.2:
            return 'Rendah'
        elif row['Expenditure_Income_Ratio'] < 0.4:
            return 'Sedang'
        else:
            return 'Tinggi'
    elif row['Family_Size_Category'] == 'Sedang':
        if row['Expenditure_Income_Ratio'] < 0.3:
            return 'Rendah'
        elif row['Expenditure_Income_Ratio'] < 0.6:
            return 'Sedang'
        else:
            return 'Tinggi'
    elif row['Family_Size_Category'] == 'Besar':
        if row['Expenditure_Income_Ratio'] < 0.4:
            return 'Rendah'
        elif row['Expenditure_Income_Ratio'] < 0.7:
            return 'Sedang'
        else:
            return 'Tinggi'

# Menerapkan fungsi pada setiap baris data
customers['Expenditure_Category'] = customers.apply(expenditure_category, axis=1)

Expenditure_Category_Counts = customers['Expenditure_Category'].value_counts()

print(Expenditure_Category_Counts)
```

```
DI Tampilkan Pengelompokkan Tingkatan Pengeluaran Pada Dataset Shop Customer Data :
Rendah    1998
Tinggi       2
Name: Expenditure_Category, dtype: int64
```

E. Visualisasi Data

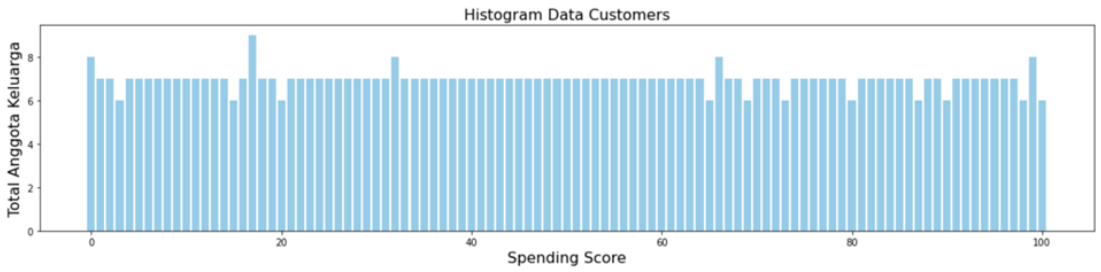
Pada tahapan ini di lakukan proses visualisasi histogram dan scatter plot untuk mengetahui hubunga antara kolom Spending Score dan Size Family. Berdasarkan hasil visualisasi ini di peroleh informasi bahwa tidak terdapat hubungan konsisten antara ukuran keluarga (Size_Family) dengan skor pengeluaran (Spending_Score). Dengan adanya ukuran keluarga yang berbeda-beda tidak selalu berarti bahwa keluarga dengan ukuran besar memiliki skor pengeluaran yang lebih tinggi.

Dengan visualisasi histogram sebagai berikut :

```
print("Di Tampilkan Histogram Data Customers Pada Dataset Shop Customers Data :")

plt.figure(figsize=(20, 4))
plt.title("Histogram Data Customers", fontsize=16)
plt.bar(customers['Spending_Score'], height=customers['Size_Family'], color='skyblue')
plt.xlabel("Spending Score", fontsize=16)
plt.ylabel("Total Anggota Keluarga", fontsize=16)
plt.show()
```

```
Di Tampilkan Histogram Data Customers Pada Dataset Shop Customers Data :
```

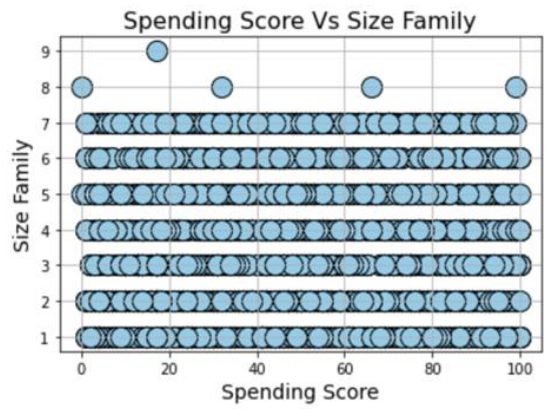


Dengan visualisasi scatter plot sebagai berikut :

```
print("Di Tampilkan Scatter Plot Dari Spending Score Vs Size Family :")

customers.plot.scatter('Spending_Score', 'Size_Family', s=200,\
                       c='skyblue',edgecolor='k')
plt.grid(True)
plt.title('Spending Score Vs Size Family',fontsize=16)
plt.xlabel('Spending Score',fontsize=14)
plt.ylabel('Size Family',fontsize=14)
plt.show()
```

```
Di Tampilkan Scatter Plot Dari Spending Score Vs Size Family :
```



F. Python Dan SQL

Pada tahapan ini di lakukan proses python dan SQL untuk pembuatan koneksi antara dataset yang telah di proses ke dalam database yang di simpan dalam SQL. Dan di lakukan serangkaian proses CRUD (Create, Read, Update, Delete), penampilan kategori ukuran keluarga, dan penampilan kategori tingkat pengeluaran di dalam python yang terhubung ke SQL, hal ini bertuuan untuk proses memanipulasi data, sehingga akan di tampilkan dataset yang telah di simpan ke dalam SQL dengan tampilan akhir sebagai berikut :

	CustomerID	Age	Annual_Income	Spending_Score	Work_Experience	Size_Family	Outlier	Family_Size_Category	Expenditure_Income_Ratio	Size_Family_Scaled	Expenditure_Category
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1975	1975	14.0	153145	59	8	6	0	Sedang	3.91785562702014e-05	1.13259407642617	Rendah
1976	1976	41.0	128960	43	9	4	0	Sedang	3.10173697270471e-05	0.117497436115912	Rendah
1977	1977	60.0	127438	82	7	2	0	Kecil	1.56939060562783e-05	-0.89759920419434	2 Rendah
1978	1978	60.0	125968	100	8	2	0	Kecil	1.587704813921e-05	-0.89759920419434	2 Rendah
1979	1979	84.0	104589	85	10	2	0	Kecil	1.91224698582069e-05	-0.89759920419434	2 Rendah
1980	1980	0.0	165321	93	8	1	1	Kecil	6.04883832060053e-06	-1.40514752434947	Rendah
1981	1981	10.0	86925	76	7	2	0	Kecil	2.30083405234397e-05	-0.89759920419434	2 Rendah
1982	1982	62.0	149797	19	5	6	0	Sedang	4.00542066930579e-05	1.13259407642617	Rendah
1983	1983	33.0	137094	68	4	1	0	Kecil	7.2942652486615e-06	-1.40514752434947	Rendah
1984	1984	52.0	55395	41	10	1	0	Kecil	1.80521707735355e-05	-1.40514752434947	Rendah
1985	1985	2.0	153622	51	6	6	0	Sedang	3.90569059119137e-05	1.13259407642617	Rendah
1986	1986	27.0	74050	44	8	1	0	Kecil	1.35043889264011e-05	-1.40514752434947	Rendah
1987	1987	4.0	68094	61	4	7	1	Besar	0.000102799071871237	1.64014239658129	Rendah
1988	1988	63.0	59244	80	7	1	0	Kecil	1.68793464317062e-05	-1.40514752434947	Rendah
1989	1989	54.0	118944	77	4	4	0	Sedang	3.36292709174065e-05	0.117497436115912	Rendah
1990	1990	47.0	75293	55	6	7	0	Besar	9.29701300253676e-05	1.64014239658129	Rendah
1991	1991	30.0	166983	69	7	3	0	Kecil	1.79659007204326e-05	-0.390050884039215	Rendah
1992	1992	97.0	129444	96	5	6	1	Sedang	4.63520904792806e-05	1.13259407642617	Rendah
1993	1993	94.0	181183	24	9	3	0	Kecil	1.65578448309168e-05	-0.390050884039215	Rendah
1994	1994	64.0	175254	100	9	5	1	Sedang	2.8530019286293e-05	0.625045756271039	Rendah
1995	1995	19.0	54121	89	6	3	0	Kecil	5.54313482751612e-05	-0.390050884039215	Rendah
1996	1996	71.0	184387	40	8	7	1	Besar	3.79636308416537e-05	1.64014239658129	Rendah
1997	1997	91.0	73158	32	7	7	0	Besar	9.56833155635747e-05	1.64014239658129	Rendah
1998	1998	87.0	90961	14	9	2	0	Kecil	2.19874451688086e-05	-0.89759920419434	2 Rendah
1999	1999	77.0	182109	4	7	2	1	Kecil	1.09824335974609e-05	-0.89759920419434	2 Rendah
2000	2000	90.0	110610	52	5	2	0	Kecil	1.80815478049e-05	-0.89759920419434	2 Rendah

Berasarkan dari serangkaian langkah-langkah di atas dapat di tarik hasil bahwa dari sebanyak 2000 pelanggan yang tercatat di peroleh informasi bahwa pelanggan tersebut kebanyakan memiliki pengeluaran yang rendah, hal ini di buktikan dengan sebanyak 1998 pelanggan memiliki tingkatan pengeluaran rendah dengan 2 pelanggan memiliki tingkat pengeluaran tinggi. Padahal, di ketahui sebanyak 239 keluarga memiliki kategori besar.

Jumlah ini di kelompokkan ke dalam kategori kecil, sedang, dan besar, jika di lihat dari rata-rata anual income (pendapatan tahunan) pelanggan di peroleh angka sebesar 110731.82 membuktikan bahwa keluarga yang memiliki pendapatan besar lebih memilih menghemat pengeluaran. Hal ini di buktikan dengan adanya pengelompokan bedasarkan kategori pengeluaran, dan tercatat bahwa hanya 2 keluarga yang memiliki pengeluaran tinggi. Berdasarkan pembahasan tersebut dapat di peroleh hasil bahwa ukuran keluarga pelanggan tidak mempengaruhi kebiasaan belanja.

Kesimpulan

Proses data wrangling pada dataset "Shop Customer Data" dengan awal 2000 baris dan 8 kolom, serta hasil akhir 2000 baris dan 11 kolom. Tahapan data wrangling melibatkan penemuan data, pemformatan data, pembersihan data, transformasi data, visualisasi data, dan penggunaan Python dan SQL.

Dalam proses tersebut, kolom Gender dan Profession dihapus karena dianggap tidak berkontribusi secara signifikan. Ditambahkan juga kolom seperti Outlier, Expenditure_Income_Ratio, Size_Family_Scaled, dan Expenditure_Category untuk mendapatkan hasil dan kesimpulan yang lebih baik.

Informasi dari kolom Outlier menunjukkan sekitar 95% data normal dan 5% data dianggap sebagai outlier. Kolom Expenditure_Income_Ratio digunakan untuk mengelompokkan pengeluaran berdasarkan kategori ukuran keluarga. Kolom Size_Family_Scaled digunakan untuk melakukan transformasi skala, dan Expenditure_Category digunakan untuk menggambarkan tingkat pengeluaran berdasarkan Size_Family_Category dan Expenditure_Income_Ratio.

Dilakukan juga penampilan statistik dari kolom Annual_Income untuk mendukung hasil analisis. Visualisasi kolom Spending_Score dan Size_Family menunjukkan bahwa tidak ada hubungan konsisten antara ukuran keluarga (size family) dengan skor pengeluaran (spending score). Penggunaan Python dan SQL digunakan dalam manipulasi data menggunakan proses CRUD.

Berdasarkan serangkaian tahapan dalam proses data wrangling yang telah di terapkan dalam proses dapat di simpulkan bahwa sebagian besar dari 2000 pelanggan memiliki pengeluaran rendah, hal ini menunjukkan ukuran pelanggan tidak begitu mempengaruhi kebiasaan belanja. Hanya 2 pelanggan dengan pengeluaran tinggi, meskipun ada 239 keluarga dengan ukuran besar. Hal ini menunjukkan bahwa keluarga dengan ukuran besar dan tingkat pendapatan tinggi cenderung menghemat pengeluaran. Pengelompokan berdasarkan kategori pengeluaran juga mengkonfirmasi bahwa hanya 2 keluarga dengan pengeluaran tinggi.

Lampiran

Link lampiran poster :
https://drive.google.com/drive/folders/1U-Qb60k5cRxix4yk4ljKTXF_Z01tekKc?usp=sharing