**Mata Kuliah Analisis Data Eksploratif**

**Laporan Tugas Membuat Tabel Distribusi Frekuensi di Excell dan SPSS**

**Dosen Pengampu: Wahyu Sri Utami, S.Si., M.Sc.**



**Disusun oleh:**

* **Fadilah Ratu Azzahra (5232811015)**
* **Lathif Ramadhan (5231811022)**

**PROGRAM STUDI SAINS DATA**

**FAKULTAS SAINS DAN TEKNOLOGI**

**UNIVERSITAS TEKNOLOGI YOGYAKARTA**

**YOGYAKARTA**

**2024**

**Pendahuluan**

Dalam dunia bisnis yang kompetitif saat ini, analisis perilaku pelanggan sangat penting untuk menjaga keberlanjutan perusahaan. Salah satu tantangan utama yang dihadapi perusahaan adalah *churn*, yaitu ketika pelanggan memutuskan untuk berhenti menggunakan layanan atau produk. Untuk memahami pola yang mendasari keputusan ini, penting bagi perusahaan untuk melakukan eksplorasi data yang mendalam, termasuk mengidentifikasi *missing values* dan *outliers* yang mungkin memengaruhi hasil analisis.

*Missing values* dalam data adalah catatan yang hilang atau tidak tersedia yang dapat menyebabkan distorsi dalam analisis jika tidak ditangani dengan benar. Sementara itu, *outliers* adalah nilai yang menyimpang secara signifikan dari mayoritas data, yang dapat mempengaruhi kesimpulan statistik. Dengan mendeteksi dan menangani *missing values* serta *outliers*, kita dapat meningkatkan kualitas data dan menghasilkan analisis yang lebih akurat.

Dataset churn yang digunakan dalam tugas ini berisi 64.374 catatan pelanggan yang mencakup berbagai aspek perilaku pelanggan seperti usia, frekuensi penggunaan layanan, durasi langganan, dan status *churn*. Dalam laporan ini, fokus kami adalah pada pengidentifikasian *missing values* dan *outliers* untuk memastikan bahwa data siap untuk dianalisis lebih lanjut. Penggunaan Python dan SPSS akan membantu dalam mendeteksi dan menangani anomali tersebut, sehingga kita dapat memperoleh wawasan yang lebih baik tentang faktor-faktor yang memengaruhi keputusan pelanggan untuk berhenti menggunakan layanan perusahaan.

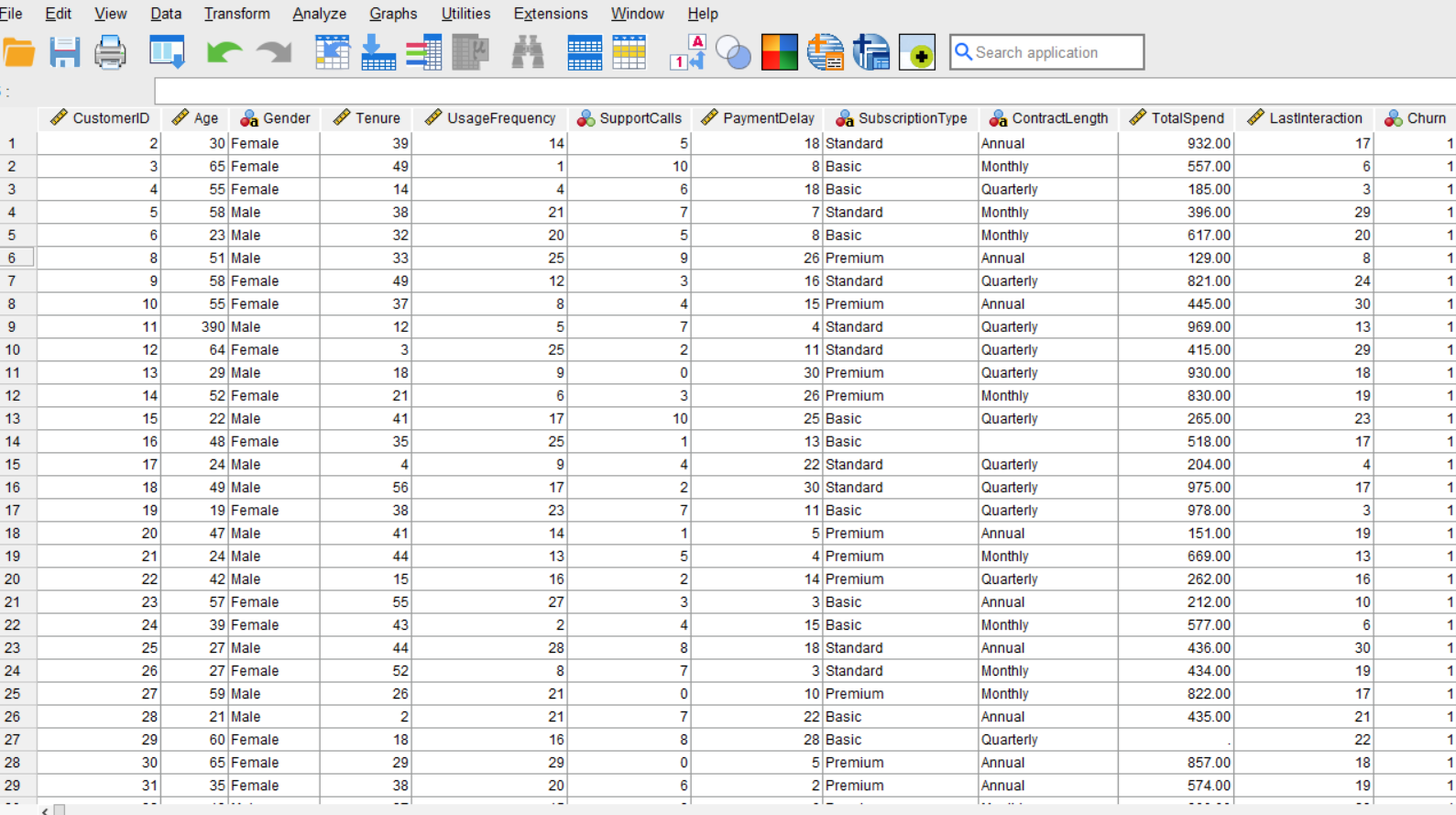
Dengan pemahaman yang lebih mendalam tentang data yang bersih dan konsisten, perusahaan dapat merumuskan strategi retensi yang lebih efektif untuk mencegah *churn* dan mempertahankan pelanggan mereka.

# Menggunakan Tool SPSS

# Missing Value

Missing value adalah informasi yang tidak tersedia untuk sebuah objek (kasus). Missing value terjadi karena informasi untuk sesuatu tentang objek tidak diberikan, sulit dicari, atau memang informasi tersebut tidak ada. Missing value pada dasarnya tidak bermasalah bagi keseluruhan data, apalagi jika jumlahnya hanya sedikit, misal hanya 1 % dari seluruh data. Namun jika persentase data yang hilang tersebut cukup besar, maka perlu dilakukan pengujian apakah data yang mengandung banyak missing tersebut masih layak diproses lebih lanjut ataukah tidak.

Ada Dataset yang berjumlah 440.832 baris dan 12 kolom, seperti gambar di bawah ini



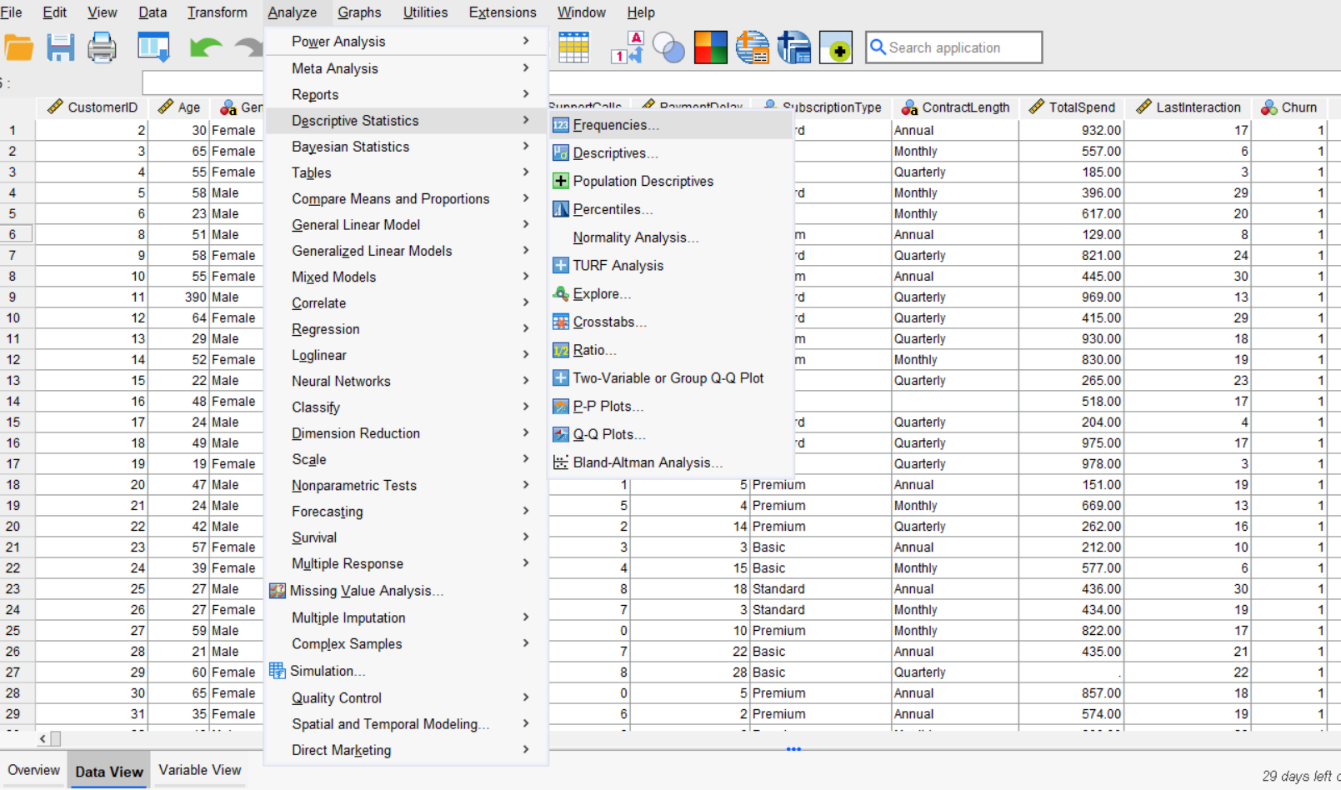
Permasalahan yang muncul dari data di atas yaitu :

a. Bagaimana deskripsi missing value yang terdapat dalam data tersebut ?

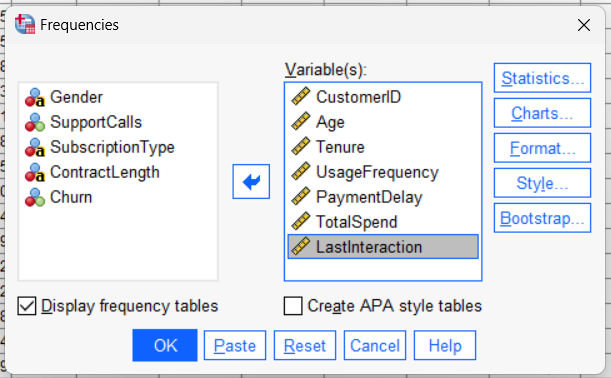
b. Bagaimana memperlakukan objek / kasus yang memiliki missing value ?

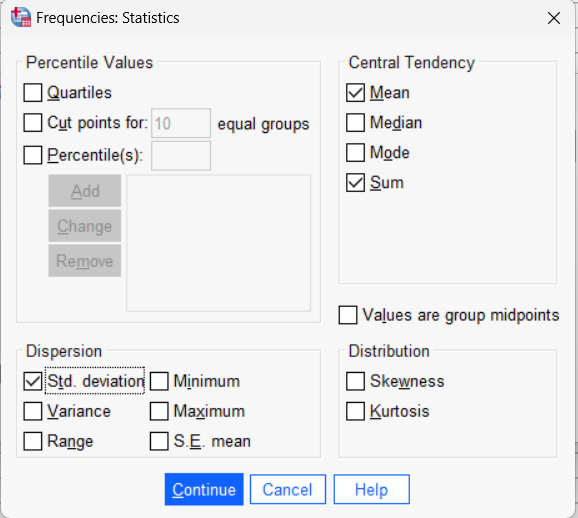
Untuk dapat menjawab permasalahan yang ada, lakukan tahapan pekerjaan dengan menggunakan aplikasi alat SPSS seperti berikut ini.

1. Dari data yang telah dimasukkan, selanjutnya klik menu “**analyze**” dan pilih sub menu “**descriptive** **statistics**”dan kemudian “**frequencies**” seperti tampilan berikut ini :



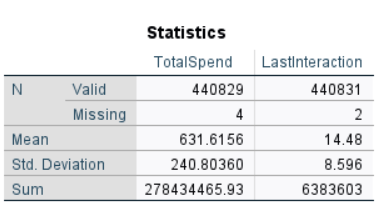
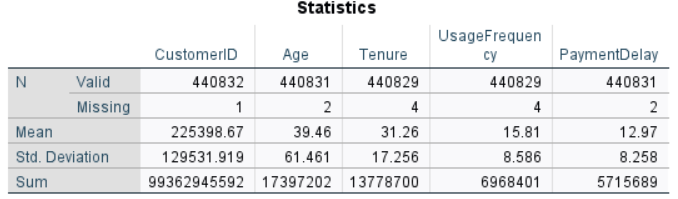
2. Masukkan variabel-variabel yang akan dicari numeriknya (CustomerID, Age, Tenure, UsageFrequency, PaymentDelay, TotalSpend, LastInteraction) pada kotak “**variable(s)**”. Kemudian klik kotak “**staistics**” dan pilih “**sum**” dan “**mean**”pada central tendency, dan “**Std. deviation**” pada Dispersion. Berikutnya klik “**continue**”.





3. Akhiri dengan meng-klik OK untuk menampilkan output.

Untuk menjawab “**permasalahan a**”, tampilan output SPSS berikut ini dapat membantu memberikan deskripsi.



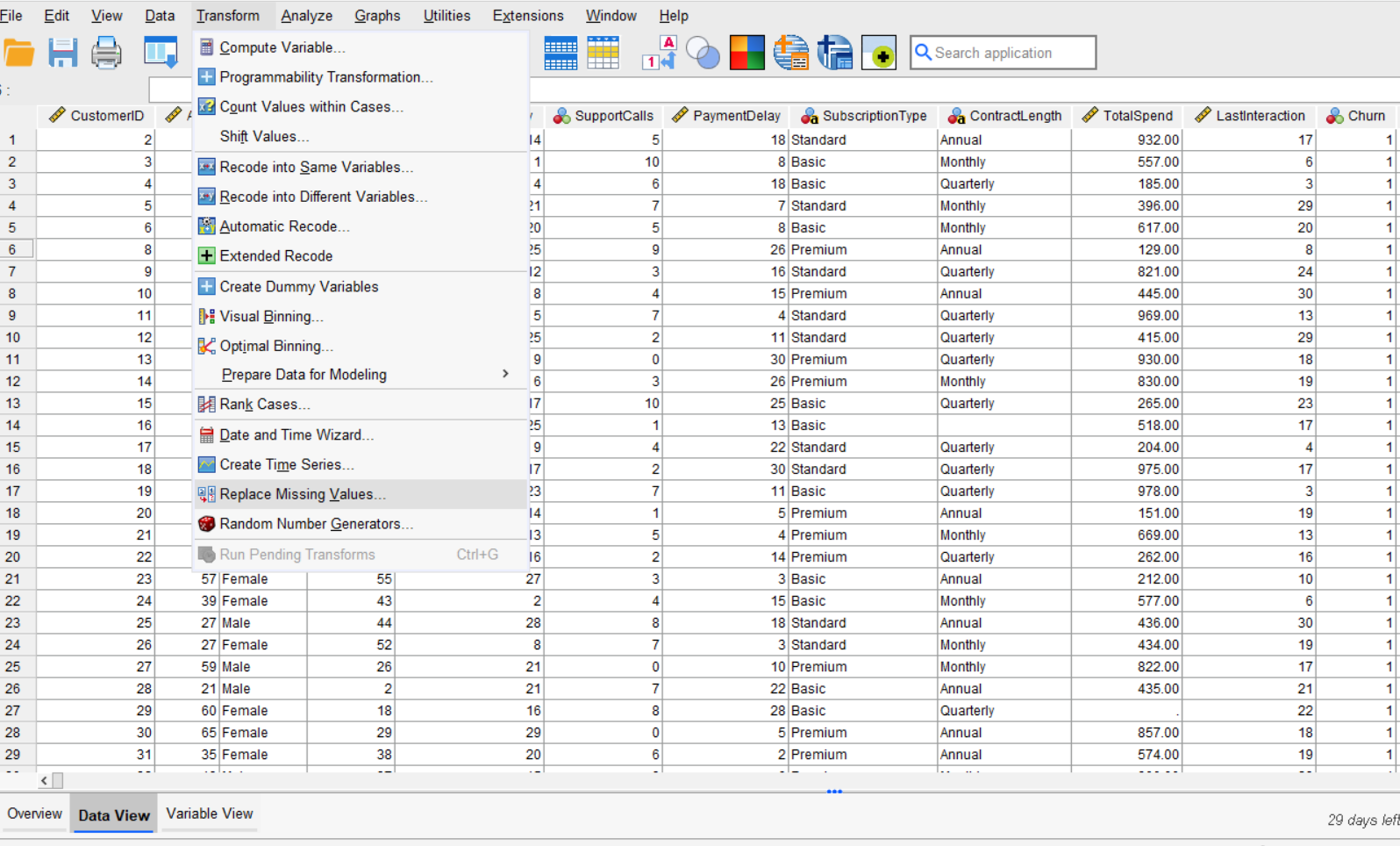
Dari tabel di atas dapat diketahui berapa missing value yang terdapat pada masing-masing variabel data.

* Variabel Customer ID terdapat 1missing value dari 440.833 objek pengamatan, sehingga 440.832 objek yang dianggap valid.
* Variabel Age terdapat 2 missing value dari 440.833 objek pengamatan, sehingga 440.831 yang dianggap valid.
* Variabel Tenure terdapat 4 missing value dari 440.833 objek pengamatan, sehingga hanya 440.829 objek yang dianggap valid.
* Variabel Usage Frequency terdapat 4 missing value dari 440.833 objek pengamatan, sehingga hanya 440.829 objek yang dianggap valid.
* Variabel Payment Delay terdapat 2 missing value dari 440.833 objek pengamatan, sehingga 440.831 objek yang dianggap valid.
* Variabel Total Spent terdapat terdapat 4 missing value dari 440.833 objek pengamatan, sehingga hanya 440.829 objek yang dianggap valid.
* Variabel Last Interaction terdapat 2 missing value dari 440.833 objek pengamatan, sehingga 440.831 objek yang dianggap valid.

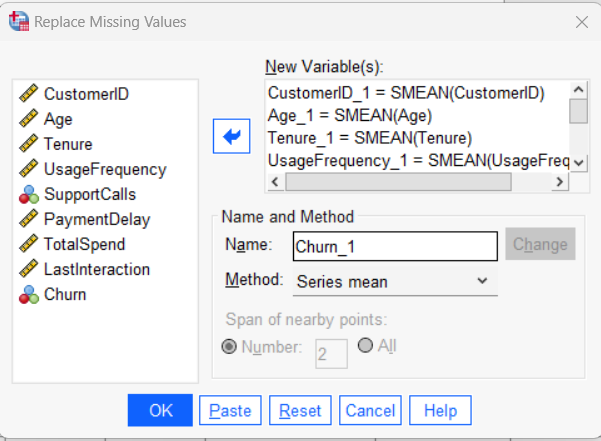
Dengan demikian dapat diketahui berapa persentase validitas data masing-masing variabel

Sedangkan untuk mengetahui bagaimana cara memperlakukan data yang memiliki missing value; berikut ini tahapan yang harus dilakukan dalam aplikasi SPSS.

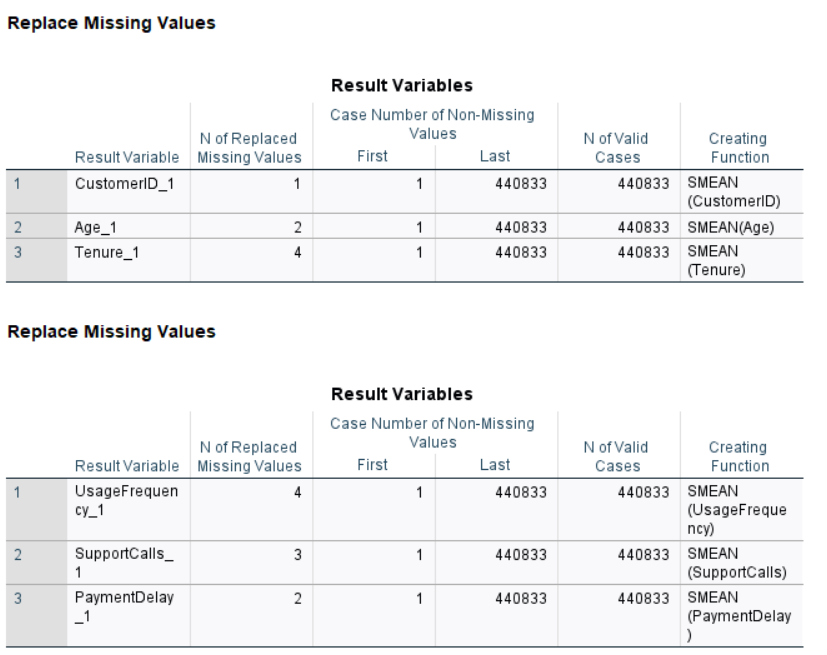
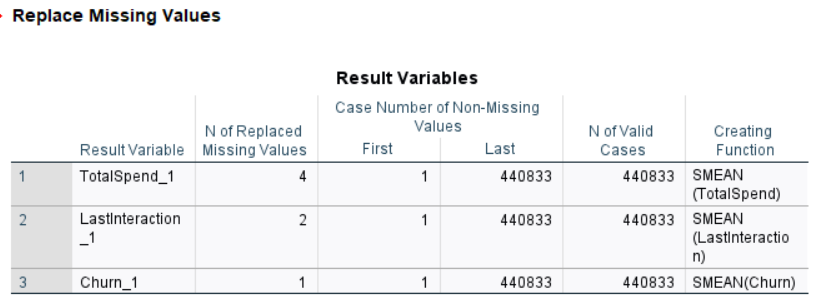
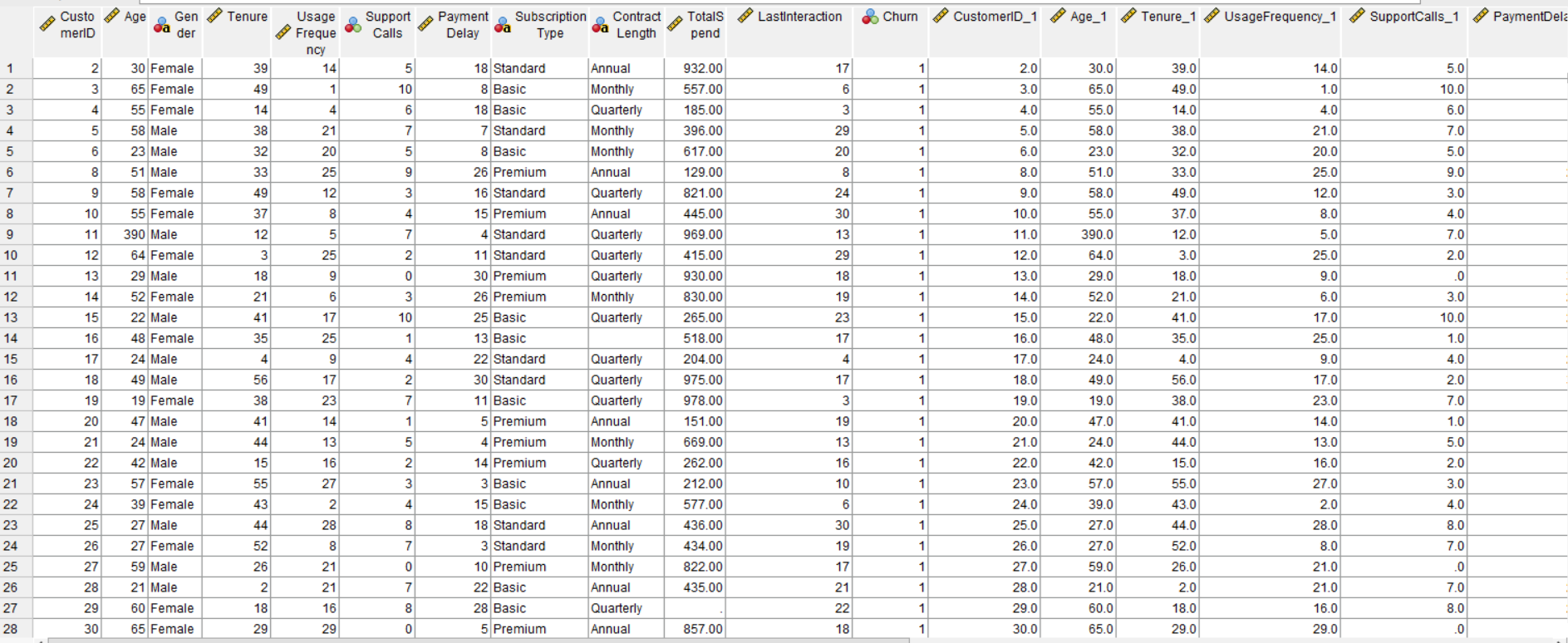
1. Dengan kembali pada tampilan data yang telah dientry, selanjutnya dari menu transform, pilih submenu “**Replace missing value**”.



2. Masukkan variabel-variabel yang memiliki missing value ke bagian New Variable. Perhatikan isi di bagian “**name and method**”



3. Akhiri dengan mengklik OK untuk menampilkan output dari aplikasi SPSS seperti berikut ini.

# B. Outlier

Data outlier (pencilan) adalah data yang secara nyata berbeda dengan data-data yang lain. Sebagai contoh, data dari 40 mahasiswa Jurusan Teknik Planologi yang mengikuti matakuliah MAP diperoleh rata-rata nilainya 60, sedangkan ada seorang mahasiswa yang mempunyai nilai MAP 100. Jelas dalam hal ini berarti seorang mahasiswa yang memiliki nilai MAP 100 tersebut merupakan data outlier.

Beberapa hal yang mempengaruhi munculnya data outlier antara lain :

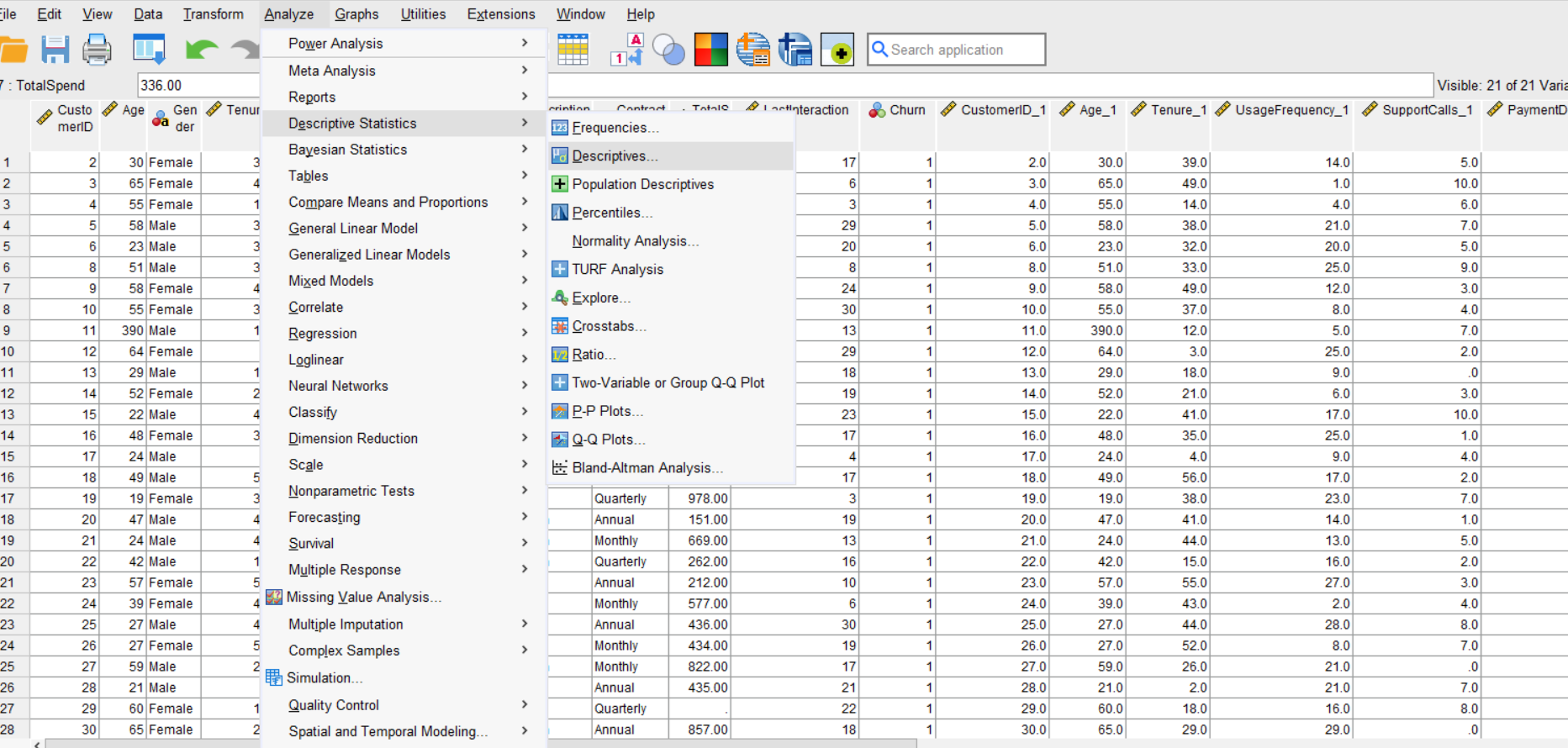
1. Kesalahan dalam pemasukan data.

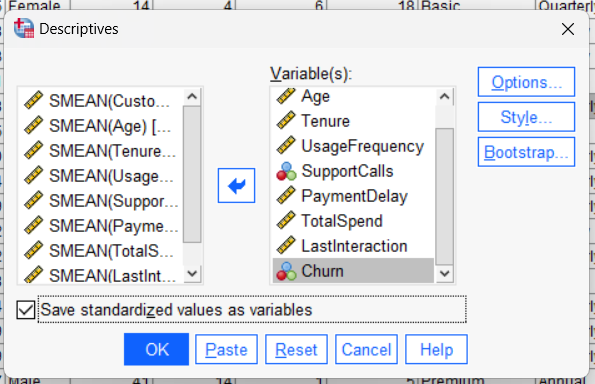
2. Kesalahan dalam pengambilan sampel.

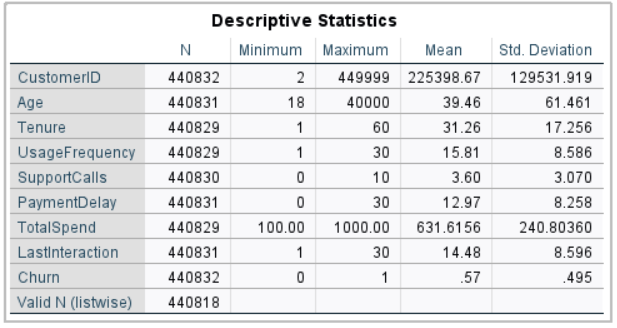
3. Memang ada data-data ekstrim yang tidak dapat dihindarkan keberadaannya.

Untuk lebih jelasnya, kasus berikut ini akan menjelaskan pengujian apakah suatu data mengandung data outlier, serta bagaimana cara penanganan data outlier tersebut.

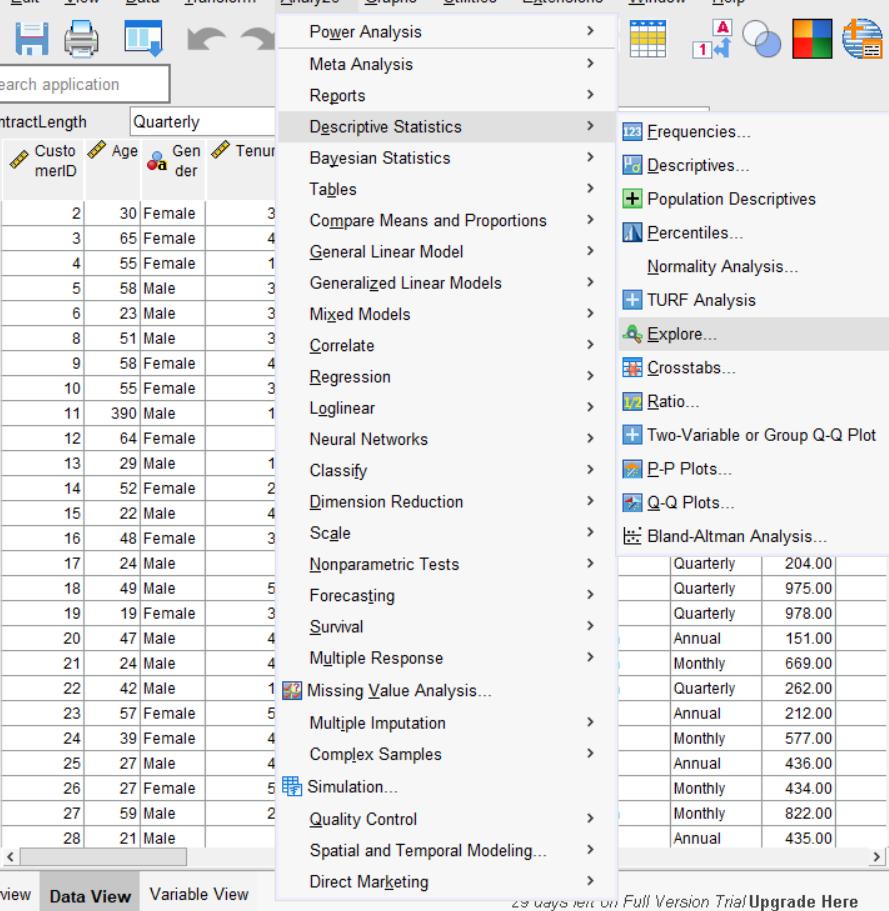
Setelah dilakukan survey pada churn atau kebiasaan pengguna berisi 440.833 catatan pelanggan yang mencakup berbagai aspek perilaku pelanggan seperti usia, frekuensi penggunaan layanan, durasi langganan, dan status *churn* (12 kolom) diperoleh data sebagai berikut :

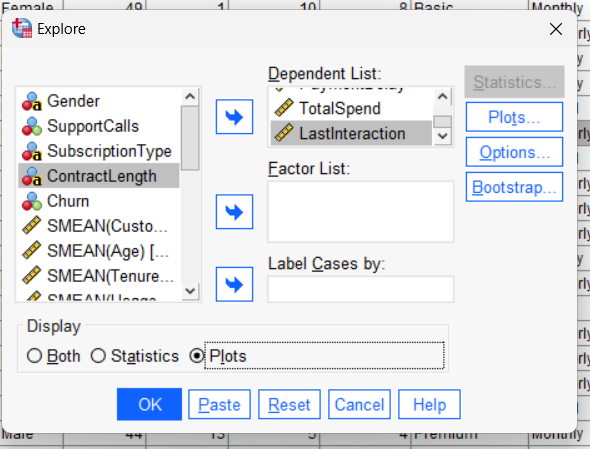


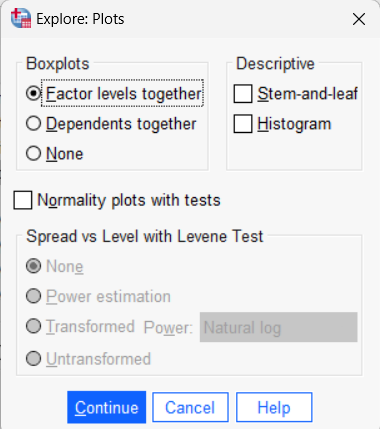


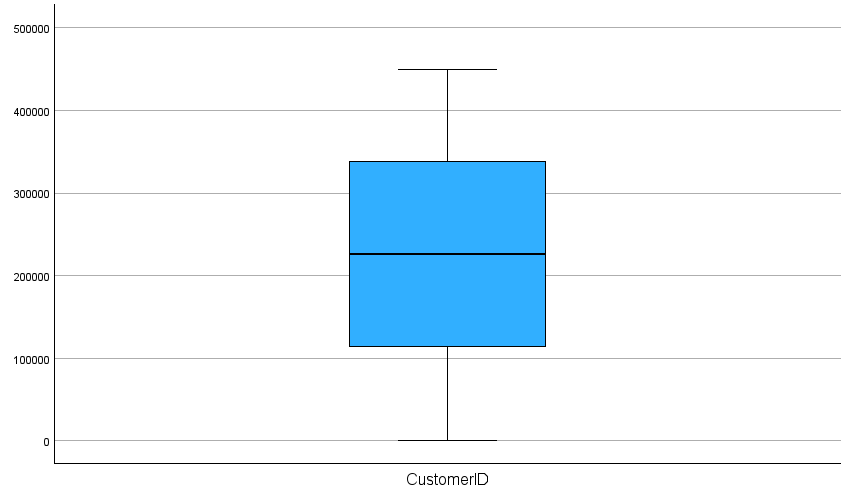


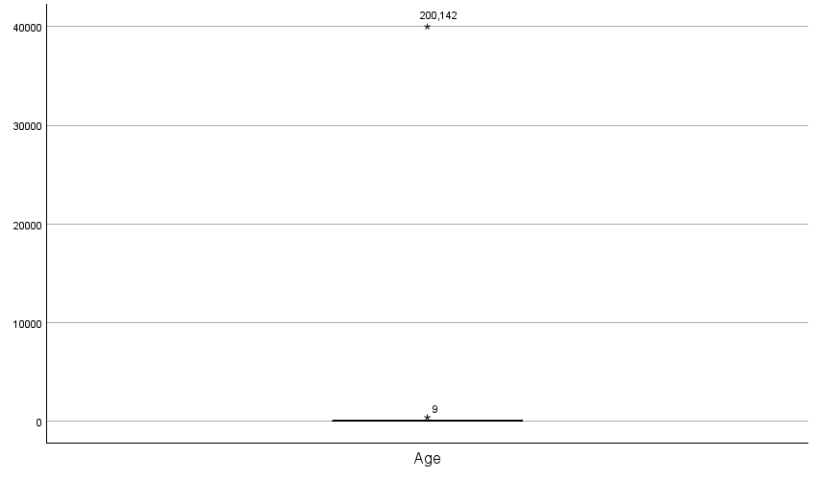
Menangani outlier

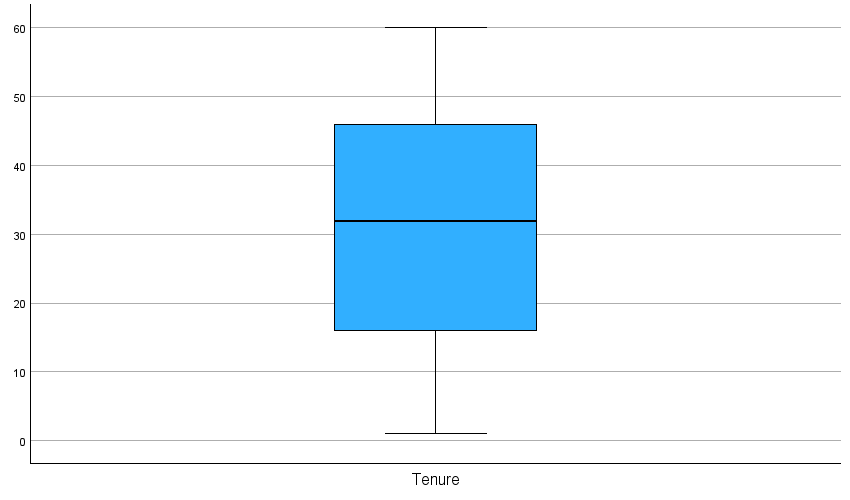


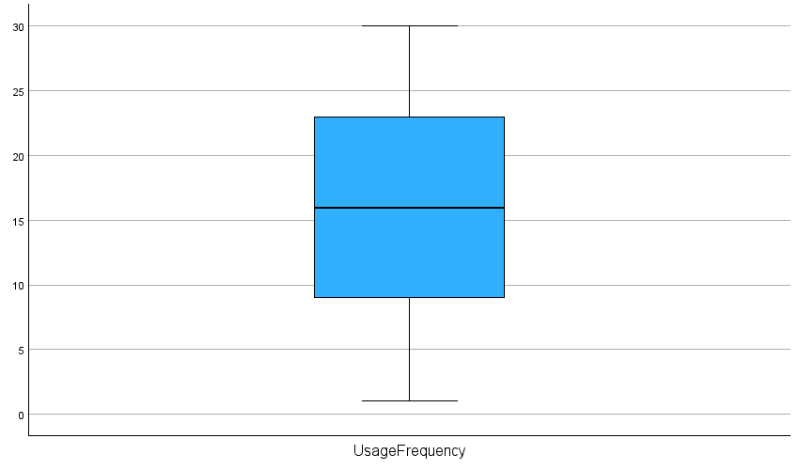
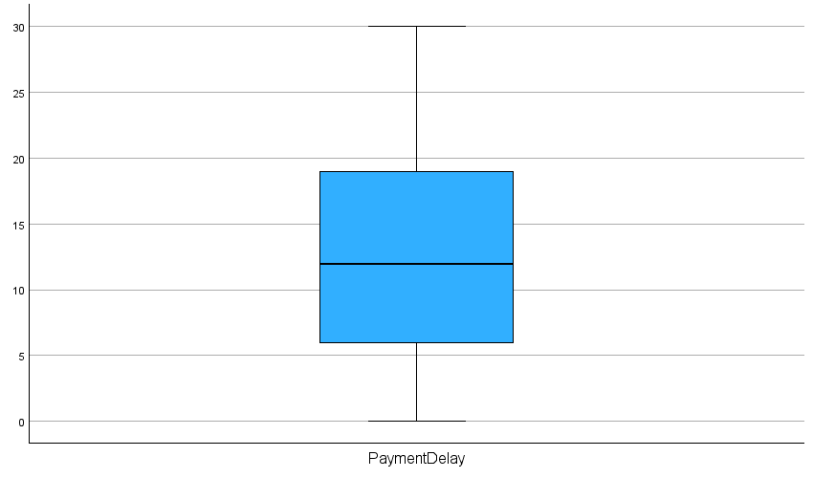
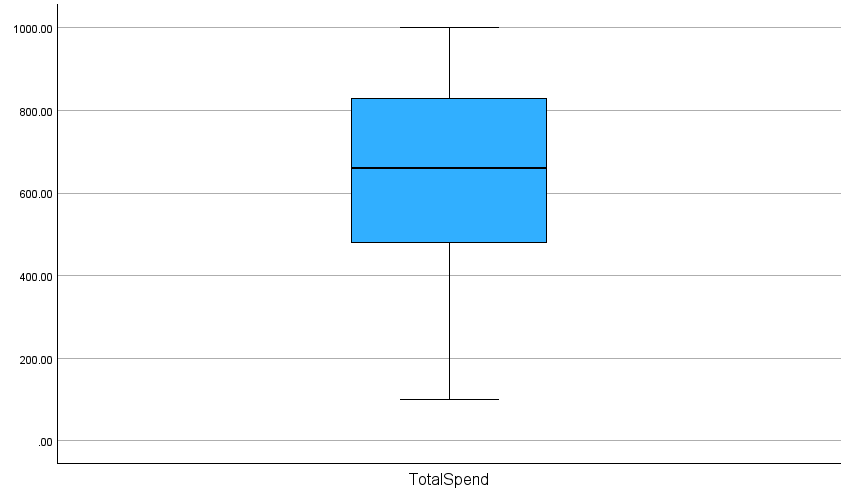
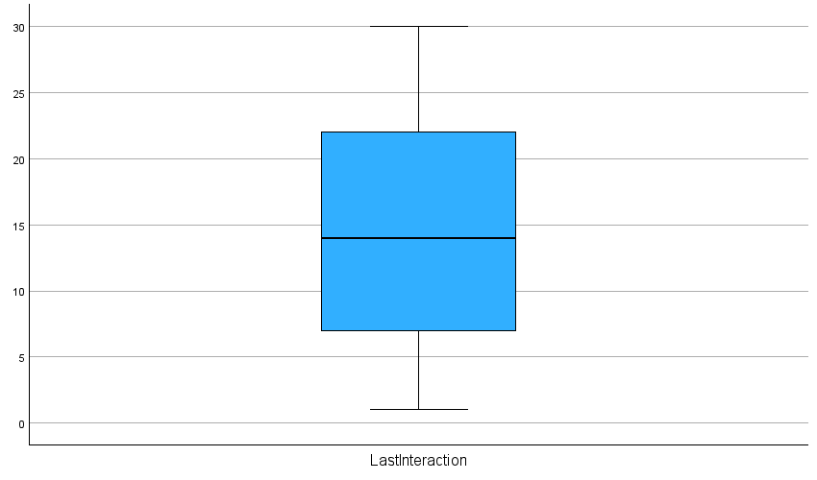










# Menggunakan Bahasa Pemrograman Python

Nah, untuk tool kedua yaitu menggunakan bahasa pemrograman python untuk mencari missing value dan outlier, serta menangani keduanya. Untuk *IDE*-nya, saya mempercayakan kepada Google Colab. Berikut merupakan langkah-langkahnya disertai kode program, output dan penjelasannya:

## 1. Mengimpor semua library yang diperlukan

**Kode program:**

**import pandas as pd**

**from google.colab import files**

**import seaborn as sns**

**import matplotlib.pyplot as plt**

**from scipy import stats**

**import numpy as np**

**Penjelasan:**

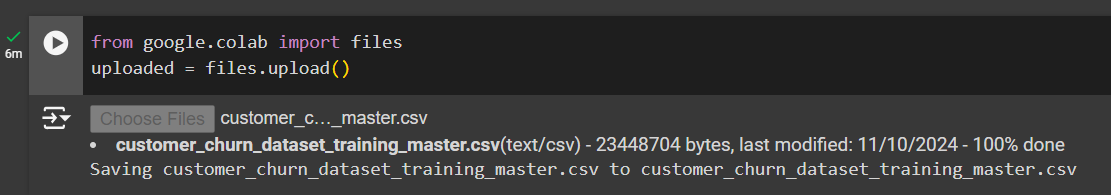
* pandas: Digunakan untuk manipulasi dan analisis data dalam bentuk data frame.
* files: Dari google.colab untuk mengupload file ke Colab.
* seaborn: Visualisasi data, seperti boxplot dan histogram.
* matplotlib.pyplot: Untuk membuat grafik dan visualisasi dasar.
* scipy.stats: Untuk melakukan perhitungan statistik, seperti Z-score untuk deteksi outlier.
* numpy: Perhitungan numerik, seperti penanganan array dan matriks.

## 2. Mengupload dataset ke colab

**Kode program:**

**from google.colab import files**

**uploaded = files.upload()**



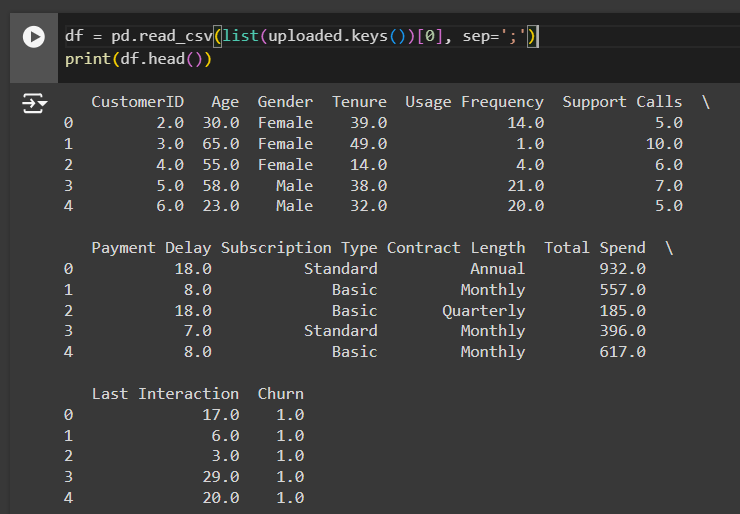
Dengan fungsi files.upload() dari library **google.colab** kita dapat mengunggah file dataset dari komputer lokal ke Colab. Ketika file diunggah, nama file akan ditampilkan di output.

## 3. Membaca file yang di-upload

**Kode program:**

**df = pd.read\_csv(list(uploaded.keys())[0], sep=';')**

**print(df.head())**



Data yang telah diupload dibaca menggunakan pd.read\_csv(). Di sini, list(uploaded.keys())[0] adalah cara untuk mengambil nama file yang diunggah. Menampilkan 5 baris pertama dataset. Ini membantu melihat struktur dataset.

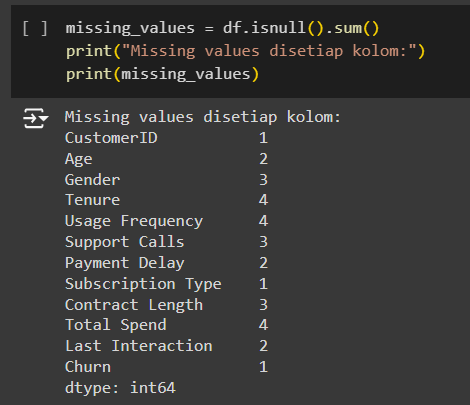
## 4. Mencari Missing Value

**Kode program:**

**missing\_values = df.isnull().sum()**

**print("Missing values disetiap kolom:")**

**print(missing\_values)**



isnull() mendeteksi missing values di setiap kolom dan sum() menghitung jumlahnya. Kode diatas menampilkan jumlah missing values pada setiap kolom.

## 5. Mengatasi Missing Value

### a. Mengisi missing values untuk kolom numerik dengan mean

**Kode program:**

**missing\_values = df.isnull().sum()**

**print("Missing values disetiap kolom:")**

**print(missing\_values)**

Data numerik yang kosong diisi dengan rata-rata (mean) kolom tersebut menggunakan fillna(). Mengisi kolom numerik dengan rata-rata, menghilangkan missing values pada kolom ini.

### b. Mengisi missing values untuk kolom kategorikal dengan mode

**Kode program:**

**categorical\_columns = df.select\_dtypes(include=['object']).columns**

**for col in categorical\_columns:**

**df[col].fillna(df[col].mode()[0], inplace=True)**

Data kategorikal yang kosong diisi dengan nilai mode (nilai yang paling sering muncul) menggunakan fillna(). Ini menghilangkan missing values pada kolom kategorikal.

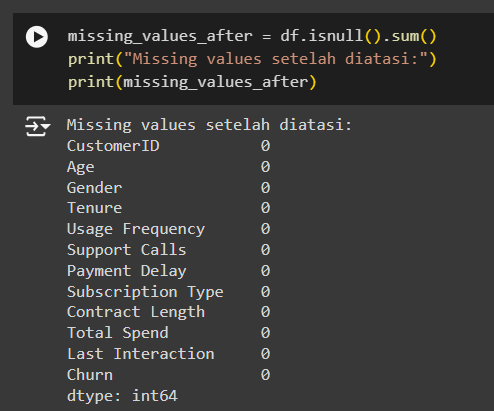
### c. Menampilkan jumlah missing values setelah diisi

**Kode program:**

**missing\_values\_after = df.isnull().sum()**

**print("Missing values setelah diatasi:")**

**print(missing\_values\_after)**



Untuk memastikan bahwa semua missing values telah diisi dengan menampilkan jumlah 0 untuk semua kolom.

## 6. Mencari Outlier

### a. Menggunakan Boxplot untuk Semua Kolom Numerik

**Kode program:**

**# Mengambil kolom numerik dari dataframe**

**numeric\_columns = df.select\_dtypes(include=['float64', 'int64']).columns**

**# Membuat boxplot untuk setiap kolom numerik**

**plt.figure(figsize=(15, 10))**

**for i, col in enumerate(numeric\_columns):**

**plt.subplot(len(numeric\_columns)//3 + 1, 3, i + 1)**

**sns.boxplot(x=df[col])**

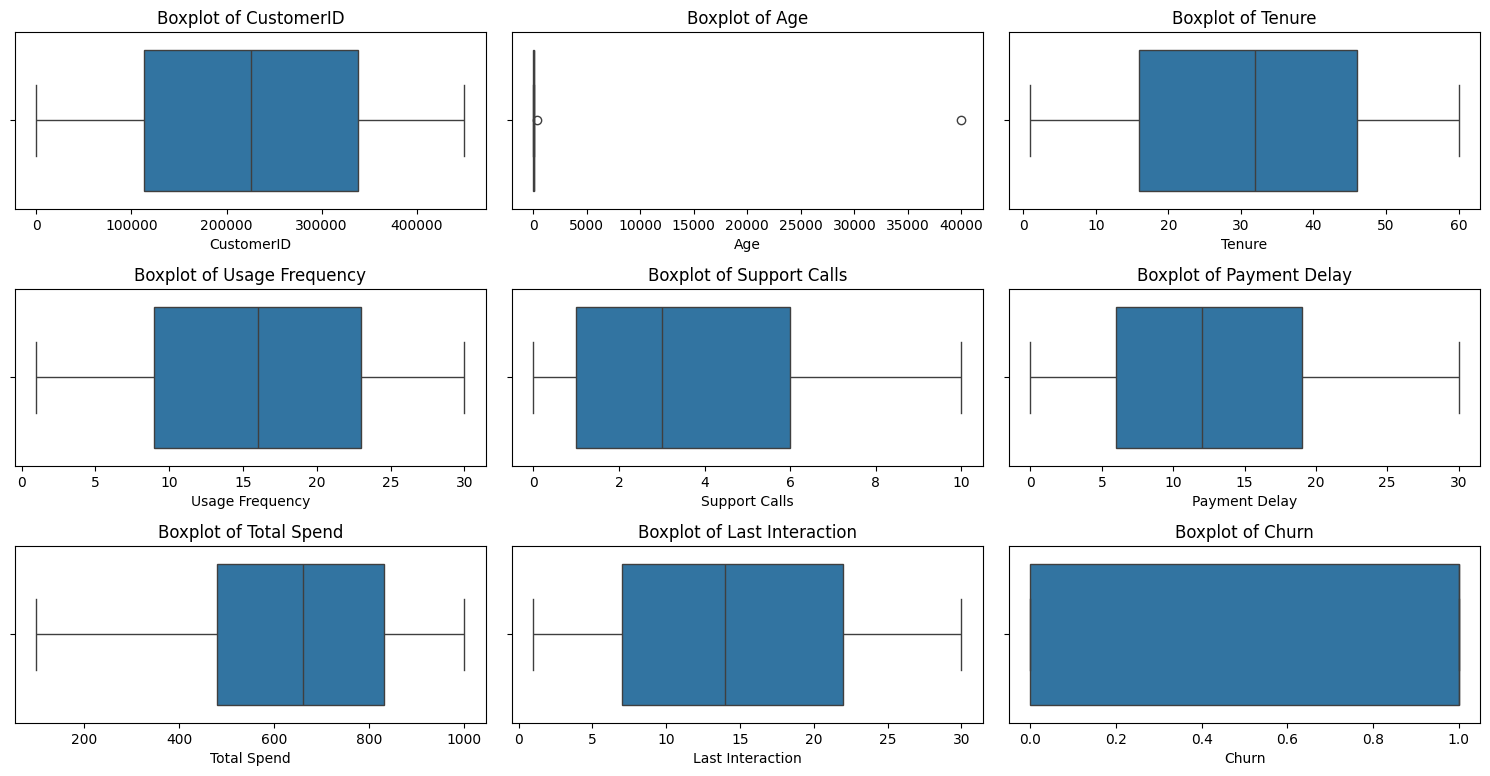
**plt.title(f'Boxplot of {col}')**

**plt.tight\_layout()**

**plt.show()**

Membuat visualisasi boxplot untuk setiap kolom numerik. Boxplot menunjukkan distribusi data dan mendeteksi outlier (ditampilkan sebagai titik di luar whiskers).

**Output:** Tampilan boxplot untuk setiap kolom numerik.



### b. Menggunakan Z-Score untuk Mendeteksi Outlier di Semua Kolom

**Kode program:**

**outliers\_dict = {}**

**# Mendeteksi outlier menggunakan z-score untuk setiap kolom numerik**

**for col in numeric\_columns:**

**z\_scores = np.abs(stats.zscore(df[col]))**

**threshold = 3 # Z-score threshold**

**outliers = df[z\_scores > threshold]**

**outliers\_dict[col] = outliers**

**print(f"Outliers in {col}:")**

**print(outliers)**

**# menampilkan outlier di setiap kolom.**

**for col, outliers in outliers\_dict.items():**

**print(f"Jumlah outliers di {col}: {len(outliers)}")**

Z-score digunakan untuk mendeteksi outlier. Z-score mengukur seberapa jauh nilai data dari rata-rata dalam satuan standar deviasi. Nilai dengan Z-score > 3 dianggap outlier.

## 7. Menangani Outlier menggunakan IQR (Interquartile Range) untuk deteksi dan penggantian

**Kode program:**

**# Menggunakan IQR untuk mendeteksi dan menangani outlier**

**Q1 = df['Age'].quantile(0.25)**

**Q3 = df['Age'].quantile(0.75)**

**IQR = Q3 - Q1**

**# Batas untuk mendeteksi outlier**

**lower\_bound = Q1 - 1.5 \* IQR**

**upper\_bound = Q3 + 1.5 \* IQR**

**# Mengganti outlier dengan batas**

**df['Age'] = np.where(df['Age'] > upper\_bound, upper\_bound, df['Age'])**

**df['Age'] = np.where(df['Age'] < lower\_bound, lower\_bound, df['Age'])**

QR digunakan untuk mendeteksi outlier pada kolom 'Age'. Nilai yang lebih rendah dari lower\_bound atau lebih tinggi dari upper\_bound dianggap outlier, dan diubah ke batas bawah atau atas.

## 8. Membuat histogram untuk melihat distribusi data

**Kode program:**

**plt.figure(figsize=(8, 6))**

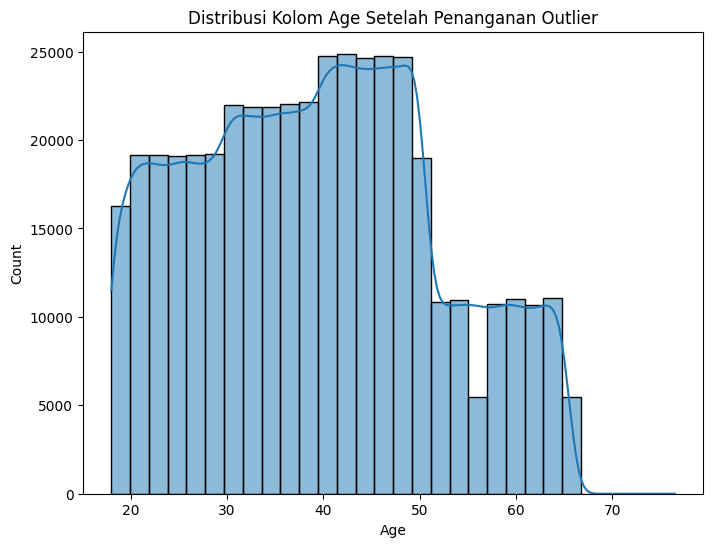
**sns.histplot(df['Age'], bins=30, kde=True)**

**plt.title('Distribusi Kolom Age Setelah Penanganan Outlier')**

**plt.show()**

Membuat histogram untuk melihat distribusi kolom 'Age' setelah penanganan outlier.

**Output:** Histogram menunjukkan distribusi 'Age' yang lebih terkontrol setelah penanganan outlier



## 9. Menyimpan dan mendownload file/data yang sudah dibersihkan

**Kode program:**

**# Menyimpan dataset yang sudah bersih ke file CSV baru**

**df.to\_csv('cleaned\_churn\_dataset.csv', index=False)**

**# Mendownload file CSV yang sudah dibersihkan**

**files.download('cleaned\_churn\_dataset.csv')**