NAMA : **MARLAN MUNAJI**
NIM : **310700012420010**
PROGRAM STUDI : **INFORMATIKA**
BATCH : **8**
MATA KULIAH : **INTRODUCTION TO DATA SCIENCE**
TUGAS : **UTS (PERTEMUAN 8)**

1. **Dua Pertanyaan Bisnis (Pertanyaan Analisis)**
   - Bagaimana hubungan antara waktu pengiriman (delivery time) dan skor ulasan pelanggan (review_score)?
     Menganalisis apakah keterlambatan pengiriman menyebabkan pelanggan memberikan rating rendah.
   - Kategori produk apa yang menghasilkan total penjualan terbesar berdasarkan data transaksi?
     Mengetahui kategori produk paling menguntungkan sehingga dapat menjadi fokus bisnis.

2. **Penggabungan dan Pembersihan Data**
   Pada tahap ini dilakukan proses penggabungan dan pembersihan data menggunakan Python (Google Colab).

[1]
```
#import library

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

[2]
```
#Membaca Semua Dataset

base_path = "/content/ETS/"

pesanan = pd.read_csv(base_path + "orders_dataset.csv", dtype=str)
item_pesanan = pd.read_csv(base_path + "order_items_dataset.csv", dtype=str)
pembayaran = pd.read_csv(base_path + "order_payments_dataset.csv", dtype=str)
ulasan = pd.read_csv(base_path + "order_reviews_dataset.csv", dtype=str)
pelanggan = pd.read_csv(base_path + "customers_dataset.csv", dtype=str)
geolokasi = pd.read_csv(base_path + "geolocation_dataset.csv", dtype=str)
produk = pd.read_csv(base_path + "products_dataset.csv", dtype=str)
kategori = pd.read_csv(base_path + "product_category_name_translation.csv", dtype=str)
penjual = pd.read_csv(base_path + "sellers_dataset.csv", dtype=str)
```

Dua dataset yang digabungkan adalah:
- orders_dataset.csv (data pesanan)
- customers_dataset.csv (data pelanggan)

Penggabungan dilakukan berdasarkan kolom customer_id dengan metode inner join, sehingga hanya pesanan yang memiliki pelanggan valid yang akan digunakan untuk analisis.

a. Jumlah Baris Awal Tiap Dataset
   Sebelum dilakukan penggabungan, jumlah baris dan kolom dari setiap dataset adalah sebagai berikut:

[3]

```
#Menampilkan Jumlah Baris Setiap Dataset

print("Jumlah pesanan:", pesanan.shape)
print("Jumlah item pesanan:", item_pesanan.shape)
print("Jumlah pembayaran:", pembayaran.shape)
print("Jumlah ulasan:", ulasan.shape)
print("Jumlah pelanggan:", pelanggan.shape)
print("Jumlah geolokasi:", geolokasi.shape)
print("Jumlah produk:", produk.shape)
print("Jumlah kategori:", kategori.shape)
print("Jumlah penjual:", penjual.shape)
```

```
Jumlah pesanan: (99441, 8)
Jumlah item pesanan: (112650, 7)
Jumlah pembayaran: (103886, 5)
Jumlah ulasan: (99224, 7)
Jumlah pelanggan: (99441, 5)
Jumlah geolokasi: (1000163, 5)
Jumlah produk: (32951, 9)
Jumlah kategori: (71, 2)
Jumlah penjual: (3095, 4)
```

- Jumlah pesanan: (99.441 baris, 8 kolom)
- Jumlah item pesanan: (112.650 baris, 7 kolom)
- Jumlah pembayaran: (103.886 baris, 5 kolom)
- Jumlah ulasan: (99.224 baris, 7 kolom)
- Jumlah pelanggan: (99.441 baris, 5 kolom)
- Jumlah geolokasi: (1.000.163 baris, 5 kolom)
- Jumlah produk: (32.951 baris, 9 kolom)
- Jumlah kategori produk: (71 baris, 2 kolom)
- Jumlah penjual: (3.095 baris, 4 kolom)

Dari sini terlihat bahwa masing-masing dataset memiliki ukuran yang berbeda tergantung konteks datanya.

b. Penggabungan Data

Dataset orders dan customers digabung menggunakan:

merged_data = pd.merge(orders, customers, on='customer_id', how='inner')

[4]

```
#Mengonversi Kolom Tanggal

kolom_tanggal = [
    "order_purchase_timestamp",
    "order_approved_at",
    "order_delivered_carrier_date",
    "order_delivered_customer_date",
    "order_estimated_delivery_date"
]

for kolom in kolom_tanggal:
    pesanan[kolom] = pd.to_datetime(pesanan[kolom], errors="coerce")
```

[5]

```
#Cek 5 baris awa kolom_tanggal

pesanan[kolom_tanggal].head()
```

	order_purchase_timestamp	order_approved_at	order_delivered_carrier_date	order_delivered_custc
0	2017-10-02 10:56:33	2017-10-02 11:07:15	2017-10-04 19:55:00	2017-10-1
1	2018-07-24 20:41:37	2018-07-26 03:24:27	2018-07-26 14:31:00	2018-08-0
2	2018-08-08 08:38:49	2018-08-08 08:55:23	2018-08-08 13:50:00	2018-08-1
3	2017-11-18 19:28:06	2017-11-18 19:45:59	2017-11-22 13:39:59	2017-12-0
4	2018-02-13 21:18:39	2018-02-13 22:20:29	2018-02-14 19:46:34	2018-02-1

[6]
✓ 0 d

```
#gabungkan dataset pesanan dan pelanggan lalu tampil head (5 teratas)

gabungan = pd.merge(pesanan, pelanggan, on="customer_id", how="inner")
gabungan.head()
```

	order_id	customer_id	order_status	order_purchase_
0	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	2017-10-
1	53cdb2fc8bc7dce0b6741e2150273451	b0830fb4747a6c6d20dea0b8c802d7ef	delivered	2018-07-
2	47770eb9100c2d0c44946d9cf07ec65d	41ce2a54c0b03bf3443c3d931a367089	delivered	2018-08-
3	949d5b44dbf5de918fe9c16f97b45f8a	f88197465ea7920adcdbec7375364d82	delivered	2017-11-
4	ad21c59c0840e6cb83a9ceb5573f8159	8ab97904e6daea8866dbdbc4fb7aad2c	delivered	2018-02-

Hasil penggabungan menghasilkan:

- Jumlah baris hasil merge: 99.441 baris
- Jumlah kolom: 12 kolom

Ini menunjukkan bahwa seluruh data pesanan memiliki customer_id yang valid.

c. Pengecekan Missing Value

Hasil pengecekan missing value pada data gabungan adalah:

[7]
✓ 0 d

```
#cek Missing Values Sebelum Cleaning

print("Jumlah missing value setiap kolom:\n")
print(gabungan.isnull().sum())
```

```
Jumlah missing value setiap kolom:

order_id                           0
customer_id                        0
order_status                       0
order_purchase_timestamp           0
order_approved_at                160
order_delivered_carrier_date    1783
order_delivered_customer_date   2965
order_estimated_delivery_date      0
customer_unique_id                 0
customer_zip_code_prefix           0
customer_city                      0
customer_state                     0
dtype: int64
```

- order_id 0
- customer_id 0
- order_status 0
- order_purchase_timestamp 0
- order_approved_at 160
- order_delivered_carrier_date 1783
- order_delivered_customer_date 2965
- order_estimated_delivery_date 0
- customer_unique_id 0
- customer_zip_code_prefix 0
- customer_city 0
- customer_state 0

Missing value hanya muncul pada kolom tanggal proses pengiriman.

d. Pembersihan Data

```
[8]
#drop NA (value salah) & drop data duplikat

# Hapus baris yang ada missing value
gabungan = gabungan.dropna()

# Hapus data duplikat
gabungan = gabungan.drop_duplicates()

# Cek jumlah baris akhir
print("Jumlah baris setelah cleaning:", gabungan.shape)
```

```
Jumlah baris setelah cleaning: (96461, 12)
```

Langkah pembersihan yang dilakukan:

1. Menghapus baris yang berisi missing value: merged_data = merged_data.dropna()
2. Menghapus data duplikat: merged_data = merged_data.drop_duplicates()

e. Jumlah Baris Akhir Setelah Cleaning

Setelah dilakukan pembersihan:

Jumlah baris akhir: (96.461 baris, 12 kolom)

Artinya sebanyak 2.980 baris dihapus karena memiliki missing value.

Data akhir yang digunakan untuk perhitungan korelasi, mean, dan varians adalah **96.461** baris.

```
[9]
✓ 0 d
#saya tampilkan hasil data sampel sebanyak 50

gabungan.head(50).to_csv("/content/ETS/gabungan_sample50.csv", index=False)
print("berhasil disimpan.")
```

```
berhasil disimpan.
```

3. Mencari Nilai Korelasi pada Data yang Telah Dibersihkan

Pada tahap ini saya menghitung korelasi antar kolom waktu (timestamp) pada dataset gabungan antara pesanan dan pelanggan (hasil nomor 2). Sebelum menghitung korelasi, seluruh kolom

tanggal saya konversi menjadi tipe data numerik (integer timestamp), karena korelasi hanya dapat dihitung pada data numerik.

[11] ✓ 0 d

```
#Ubah kolom tanggal menjadi angka
kolom_tanggal = [
    "order_purchase_timestamp",
    "order_approved_at",
    "order_delivered_carrier_date",
    "order_delivered_customer_date",
    "order_estimated_delivery_date"
]

for col in kolom_tanggal:
    gabungan[col] = pd.to_datetime(gabungan[col], errors="coerce")
    gabungan[col] = gabungan[col].astype("int64")   # ubah ke angka
```

[12]

```
#Pilih hanya kolom numerik

kolom_angka = gabungan.select_dtypes(include="number")
kolom_angka.head()
```

	order_purchase_timestamp	order_approved_at	order_delivered_carrier_date	order_delivered_cus
0	1506941793000000000	1506942435000000000	1507146900000000000	150767071
1	1532464897000000000	1532575467000000000	1532615460000000000	153365566
2	1533717529000000000	1533718523000000000	1533736200000000000	153452918
3	1511033286000000000	1511034359000000000	1511357999000000000	151217452
4	1518556719000000000	1518560429000000000	1518637594000000000	151880502

Hasil Korelasi:

[13] ✓ 0 d

```
#hitung korelasi
korelasi = kolom_angka.corr()
korelasi
```

	order_purchase_timestamp	order_approved_at	order_delivered_carrier_
order_purchase_timestamp	1.000000	0.999984	0.99
order_approved_at	0.999984	1.000000	0.99
order_delivered_carrier_date	0.999724	0.999733	1.00
order_delivered_customer_date	0.998050	0.998062	0.99
order_estimated_delivery_date	0.998402	0.998401	0.99

	order_purchase_timestamp	order_approved_at	order_delivered_carrier_date	order_delivered_customer_date	order_estimated_delivery_date
order_purchase_timestamp	1.000000	0.999984	0.999724	0.998050	0.998402
order_approved_at	0.999984	1.000000	0.999733	0.998062	0.998401

order_delivered_carrier_date	0.999724	0.999733	1.000000	0.998350	0.998382
order_delivered_customer_date	0.998050	0.998062	0.998350	1.000000	0.997770
order_estimated_delivery_date	0.998402	0.998401	0.998382	0.997770	1.000000

Penjelasan Korelasi

1. Korelasi sangat kuat antar semua variabel waktu (0.997 – 1.000)
   Ini menunjukkan bahwa seluruh proses dalam siklus pemesanan e-commerce berlangsung secara berurutan dan saling terkait. Misalnya:
   - Pesanan dilakukan -> segera disetujui
   - Pesanan disetujui -> segera diserahkan ke kurir
   - Kurir mengambil paket -> kemudian paket sampai ke pelanggan
   - Perkiraan tanggal pengiriman sangat dekat dengan tanggal aktual

2. Korelasi tertinggi (0.999984)
   order_purchase_timestamp <-> order_approved_at
   Artinya: pesanan hampir selalu disetujui sangat cepat setelah dibuat.

3. Korelasi yang juga sangat kuat (0.998 – 0.9997)
   order_delivered_carrier_date berhubungan kuat dengan
   - purchase time
   - approved time
   - delivery time

   Semakin cepat proses awal dilakukan, semakin cepat pula pesanan dikirim dan diterima pelanggan.

4. Tidak ada korelasi negatif
   Semua bernilai positif: menandakan pola linier yang konsisten dalam alur pembelian.

4. Mean dan Varian

a. Mean
   Berikut adalah rata-rata waktu dari setiap tahapan pesanan berdasarkan hasil perhitungan:

```
[ ]
#perhitungan mean
# Daftar kolom tanggal yang sudah menjadi angka
kolom_tanggal = [
    "order_purchase_timestamp",
    "order_approved_at",
    "order_delivered_carrier_date",
    "order_delivered_customer_date",
    "order_estimated_delivery_date"
]

# Hitung mean
mean_waktu = gabungan[kolom_tanggal].mean()

# Hitung variance
var_waktu = gabungan[kolom_tanggal].var()

mean_waktu, var_waktu

mean_waktu_datetime = pd.to_datetime(mean_waktu)
mean_waktu_datetime
```

	0
order_purchase_timestamp	2018-01-01 23:53:26.642249216
order_approved_at	2018-01-02 10:10:06.480142336
order_delivered_carrier_date	2018-01-05 05:21:04.508827392
order_delivered_customer_date	2018-01-14 13:17:13.228102400
order_estimated_delivery_date	2018-01-25 17:33:14.236012544

Tahapan	Mean (Rata-rata Waktu)
Rata-rata waktu pembelian	**2018-01-01 23:53:26**
Rata-rata waktu persetujuan pembelian	**2018-01-02 10:10:06**
Rata-rata waktu pengiriman ke kurir	**2018-01-05 05:21:04**
Rata-rata pesanan diterima pelanggan	**2018-01-14 13:17:13**
Rata-rata estimasi pengiriman	**2018-01-25 17:33:14**

Interpretasi Mean:

- Rata-rata pesanan dilakukan pada awal Januari 2018.
- Rata-rata pesanan disetujui kurang dari 1 hari setelah dibuat.
- Rata-rata pesanan tiba di kurir sekitar 3 hari setelah pembelian.
- Rata-rata pelanggan menerima barang sekitar 13 hari setelah pembelian.
- Estimasi pengiriman rata-rata diset ke 25 Januari, artinya sistem memberi estimasi sekitar 24 hari dari tanggal pembelian.

b. Varian

Setelah data pada kolom waktu dibersihkan dan dikonversi ke format numerik (int64), dilakukan perhitungan variance untuk mengetahui tingkat penyebaran data terhadap nilai rata-ratanya. Adapun hasil perhitungan variance pada masing-masing kolom adalah sebagai berikut:

```
#varian
var_waktu = gabungan[kolom_tanggal].var()
var_waktu

print("=== VARIANCE WAKTU (SETIAP KOLOM) ===")
print(var_waktu)
```

```
=== VARIANCE WAKTU (SETIAP KOLOM) ===
order_purchase_timestamp         1.743573e+32
order_approved_at                1.743812e+32
order_delivered_carrier_date     1.737551e+32
order_delivered_customer_date    1.729173e+32
order_estimated_delivery_date    1.704524e+32
dtype: float64
```

- order_purchase_timestamp: 1.743573 x 10^{32}
- order_approved_at: 1.743812 x 10^{32}
- order_delivered_carrier_date: 1.737551 x 10^{32}
- order_delivered_customer_date: 1.729173 x 10^{32}
- order_estimated_delivery_date: 1.704524 x 10^{32}

Nilai variance yang sangat besar ini wajar terjadi karena kolom timestamp telah dikonversi menjadi bilangan integer 64-bit dalam satuan nanodetik sejak epoch, sehingga skala angkanya memang besar. Variance yang lebih tinggi menunjukkan bahwa data waktu pada kolom tersebut memiliki rentang atau variasi yang lebih luas, sementara variance yang lebih rendah menunjukkan penyebaran data waktu yang lebih konsisten.

5. Konsep Berpikir Komputasi

Pada proses analisis data di soal nomor 1 sampai 4, saya menggunakan beberapa teknik dalam berpikir komputasi, yaitu:

- Dekomposisi
  Memecah pekerjaan menjadi langkah-langkah kecil, seperti membaca data, menggabungkan dataset, membersihkan data, dan menghitung statistik.
- Pengenalan Pola (Pattern Recognition)
  Mengidentifikasi pola seperti data yang hilang, data duplikat, serta melihat pola hubungan antar variabel melalui korelasi.
- Abstraksi
  Memilih hanya data yang relevan untuk dianalisis, misalnya hanya memakai kolom numerik dan timestamp yang dibutuhkan untuk korelasi, mean, dan variance.
- Berpikir Algoritmik
  Menyusun langkah-langkah terurut dalam analisis, seperti proses merge, drop missing value, konversi tipe data, lalu menghitung korelasi, mean, dan variance secara sistematis.