LAPORAN TUGAS PERTEMUAN 10

BIG DATA

Oleh :

MOCHAMMAD ZAKARO AL FAJRI 2241720175

D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2025

Tugas 10 – Data Cleaning dan Transformasi Menggunakan Apache Spark

Persiapan Pengerjakan Praktikum

- Mount data

```
PS C:\Users\KAKA> docker exec -u root -it spark-master bash
root@6766ede032d6:/opt/bitnami/spark# mkdir -p /opt/spark_data
root@6766ede032d6:/opt/bitnami/spark# exit
exit
PS C:\Users\KAKA> docker cp ecommerce_transactions_1000.csv spark-master:/opt/spark_data/
Successfully copied 58.4kB to spark-master:/opt/spark_data/
PS C:\Users\KAKA> ls /opt/spark_data/


    Directory: C:\opt\spark_data


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         4/29/2025   3:27 PM          56482 ecommerce_transactio
                                                 ns_1000.csv
```

Praktikum 1 :

1. Melakukan load data file “ecommerce_transactions_1000.csv”

```
[2]: from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("DataCleaningBigData").getOrCreate()

df = spark.read.csv("/opt/spark_data/ecommerce_transactions_1000.csv", header=True, inferSchema=True)
df.show(5)
```

```
+--------------+-------+--------+--------------------+-------------------+
|transaction_id|user_id|  amount|               email|   transaction_time|
+--------------+-------+--------+--------------------+-------------------+
|         T0001|   U069|    NULL|jeffreyfisher@gma...|2025-04-20 08:00:02|
|         T0002|   U253|70921.08| porteramy@yahoo.com|2025-03-30 21:07:41|
|         T0003|   U222|42313.74|  jerome93@yahoo.com|2025-04-20 10:50:30|
|         T0004|   U187|    NULL|jimeneztamara@sny...|2025-04-05 11:48:29|
|         T0005|   U064|81176.73|   louis64@gmail.com|2025-04-14 08:50:35|
+--------------+-------+--------+--------------------+-------------------+
only showing top 5 rows
```

2. Melakukan inspeksi data dengan beberapa langkah berikut :
   a. Lihat struktur schema:

```
[4]: df.printSchema()
```

```
root
 |-- transaction_id: string (nullable = true)
 |-- user_id: string (nullable = true)
 |-- amount: double (nullable = true)
 |-- email: string (nullable = true)
 |-- transaction_time: timestamp (nullable = true)
```

   b. Menghitung missing values setiap kolom:

```
[6]: from pyspark.sql.functions import col, when, count

df.select([count(when(col(c).isNull(), c)).alias(c) for c in df.columns]).show()
```

```
+--------------+-------+------+-----+----------------+
|transaction_id|user_id|amount|email|transaction_time|
+--------------+-------+------+-----+----------------+
|             0|      0|   316|    0|              50|
+--------------+-------+------+-----+----------------+
```

c. Menghitung jumlah total data:

```
[7]: print("Jumlah baris:", df.count())
```

```
Jumlah baris: 1000
```

3. Melakukan Cleaning Data dengan beberapa langkah berikut :
   a. Handling Missing Values, dengan cara :
      - Drop transaksi yang tidak memiliki **transaction_time**.
      - Isi nilai kosong pada amount dengan 0 .

```
[8]: df = df.dropna(subset=["transaction_time"])
     df = df.fillna({"amount": 0})
```

```
[9]: df.select([count(when(col(c).isNull(), c)).alias(c) for c in df.columns]).show()
```

```
+--------------+-------+------+-----+----------------+
|transaction_id|user_id|amount|email|transaction_time|
+--------------+-------+------+-----+----------------+
|             0|      0|     0|    0|               0|
+--------------+-------+------+-----+----------------+
```

   b. Cleaning Format Email, dengan cara :
      - Buat kolom baru **email_domain** yang berisi domain email
      - Hapus transaksi yang email-nya tidak valid (tidak mengandung '@')

```
[11]: from pyspark.sql.functions import instr, substring_index

      df = df.withColumn("email_domain", substring_index("email", "@", -1))

      df = df.filter(instr(col("email"), "@") > 0)
```

```
[12]: from pyspark.sql.functions import col, when, count

      df.select(
          count(
              when(~col("email").contains("@"), True)
          ).alias("invalid_email_count")
      ).show()
```

```
+-------------------+
|invalid_email_count|
+-------------------+
|                  0|
+-------------------+
```

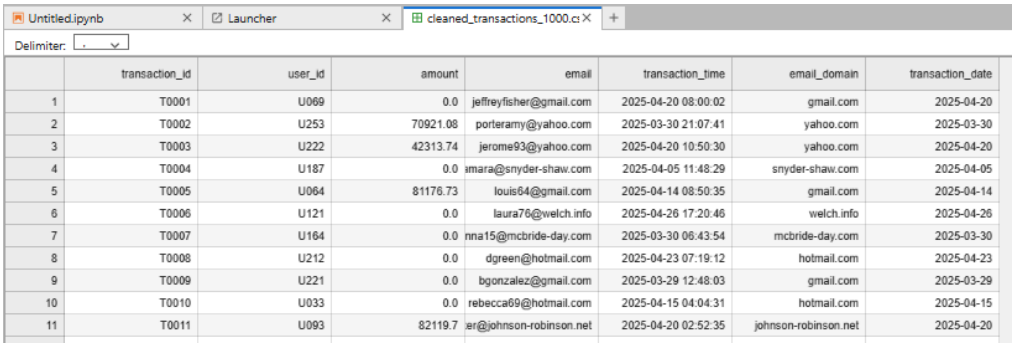
4. Melakukan Transformasi Data dengan beberapa langkah berikut :
   a. Ubah kolom amount menjadi tipe DoubleType.
   b. Tambahkan kolom baru **transaction_date** dari **transaction_time**.

```
[14]: from pyspark.sql.types import DoubleType
      from pyspark.sql.functions import to_date, col

      df = df.withColumn("amount", col("amount").cast(DoubleType()))
      df = df.withColumn("transaction_date", to_date(col("transaction_time")))
```

5. Melakukan Transformasi Data dengan beberapa langkah berikut :
   a. Simpan dataframe hasil cleaning ke file baru.

```
[26]: final = df.toPandas()
      final.to_csv("work/cleaned_transactions_1000.csv", index=False)
```



Untitled.ipynb | Launcher | cleaned_transactions_1000.cs

Delimiter: ,

	transaction_id	user_id	amount	email	transaction_time	email_domain	transaction_date
1	T0001	U069	0.0	jeffreyfisher@gmail.com	2025-04-20 08:00:02	gmail.com	2025-04-20
2	T0002	U253	70921.08	porterarmy@yahoo.com	2025-03-30 21:07:41	yahoo.com	2025-03-30
3	T0003	U222	42313.74	jerome93@yahoo.com	2025-04-20 10:50:30	yahoo.com	2025-04-20
4	T0004	U187	0.0	mara@snyder-shaw.com	2025-04-05 11:48:29	snyder-shaw.com	2025-04-05
5	T0005	U064	81176.73	louis64@gmail.com	2025-04-14 08:50:35	gmail.com	2025-04-14
6	T0006	U121	0.0	laura76@welch.info	2025-04-26 17:20:46	welch.info	2025-04-26
7	T0007	U164	0.0	nna15@mcbride-day.com	2025-03-30 06:43:54	mcbride-day.com	2025-03-30
8	T0008	U212	0.0	dgreen@hotmail.com	2025-04-23 07:19:12	hotmail.com	2025-04-23
9	T0009	U221	0.0	bgonzalez@gmail.com	2025-03-29 12:48:03	gmail.com	2025-03-29
10	T0010	U033	0.0	rebecca69@hotmail.com	2025-04-15 04:04:31	hotmail.com	2025-04-15
11	T0011	U093	82119.7	er@johnson-robinson.net	2025-04-20 02:52:35	johnson-robinson.net	2025-04-20

Pertanyaan dan Jawaban :

1. Berapa banyak data yang dibuang karena transaction_time kosong?
   Jawab : terdapat 50 data yang dibuang
   - Sebelum dilakukan pembersihan, terdapat 50 data transaction_time kosong

```
from pyspark.sql.functions import col, when, count

df.select([count(when(col(c).isNull(), c)).alias(c) for c in df.columns]).show()
```

```
+--------------+-------+------+-----+----------------+
|transaction_id|user_id|amount|email|transaction_time|
+--------------+-------+------+-----+----------------+
|             0|      0|   316|    0|              50|
+--------------+-------+------+-----+----------------+
```

   - Kemudian setelah melakukan pembersihan, 50 data transaction_time kosong berhasil dibuang

```
df.select([count(when(col(c).isNull(), c)).alias(c) for c in df.columns]).show()
```

```
+--------------+-------+------+-----+----------------+
|transaction_id|user_id|amount|email|transaction_time|
+--------------+-------+------+-----+----------------+
|             0|      0|     0|    0|               0|
+--------------+-------+------+-----+----------------+
```

2. Apakah semua data amount sudah bertipe numerik setelah cleaning?
   Jawab : Iya semua data amount sudah bertipe numerik
   - Sebelum cleaning, ada beberapa nilai amount yang tidak bertipe numerik

8	T0007	U164		deanna15@mcbride	2025-03-30 6:43:54
9	T0008	U212	NaN	dgreen@hotmail.cor	2025-04-23 7:19:12
10	T0009	U221		bgonzalez@gmail.c	2025-03-29 12:48:03
11	T0010	U033	NaN	rebecca69@hotmail	2025-04-15 4:04:31

   - Namun setelah dilakukan cleaning data, semua data amount bernilai numerik

8	T0008	U212	0.0	dgreen@hotmail.com
9	T0009	U221	0.0	bgonzalez@gmail.com
10	T0010	U033	0.0	rebecca69@hotmail.com

3. Kenapa lebih baik memperbaiki email invalid sebelum menganalisis data transaksi?
   Jawab :
   Memperbaiki email yang tidak valid penting dilakukan agar analisis data transaksi lebih akurat. Email berfungsi sebagai identitas pelanggan, sehingga jika tidak valid, dapat mengganggu proses segmentasi, pelacakan transaksi, dan integrasi data. Dengan data email yang bersih, hasil analisis menjadi lebih andal dan dapat digunakan secara efektif untuk mendukung keputusan bisnis.

Praktikum 2 :

1. Melakukan load data :

```
[1]: from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("OutlierDetection").getOrCreate()

df = spark.read.csv("/opt/spark_data/ecommerce_transactions_1000.csv", header=True, inferSchema=True)
df = df.fillna({"amount":0})
df = df.withColumn("amount", df["amount"].cast("double"))
```

2. Hitung Statistik Dasar
   Kita butuh:
   - Q1 (25th percentile)
   - Q3 (75th percentile)
   - IQR (Interquartile Range)

```
[2]: quantiles = df.approxQuantile("amount", [0.25, 0.75], 0.05)
Q1, Q3 = quantiles
IQR = Q3 - Q1

lower_bound = Q1 -  1.5 * IQR
upper_bound = Q3 +  1.5 * IQR

print(f"Q1 = {Q1}, Q3 = {Q3}, IQR = {IQR}")
print(f"Lower Bound = {lower_bound}, Upper Bound = {upper_bound}")
```

```
Q1 = 0.0, Q3 = 24763.06, IQR = 24763.06
Lower Bound = -37144.590000000004, Upper Bound = 61907.65000000001
```

3. Deteksi Outliers
   Cari data amount yang lebih kecil dari lower bound atau lebih besar dari upper bound.

```
[6]: outliers = df.filter((df.amount < lower_bound) | (df.amount > upper_bound))
outliers.show()
```

```
+--------------+-------+--------+--------------------+-------------------+
|transaction_id|user_id|  amount|               email|   transaction_time|
+--------------+-------+--------+--------------------+-------------------+
|         T0002|   U253|70921.08| porteramy@yahoo.com|2025-03-30 21:07:41|
|         T0005|   U064|81176.73|   louis64@gmail.com|2025-04-14 08:50:35|
|         T0011|   U093| 82119.7|roberttucker@john...|2025-04-20 02:52:35|
|         T0012|   U279| 63515.6|brucesmith@gmail.com|2025-04-20 09:58:53|
|         T0035|   U180|74468.55|michaelcarey@gmai...|2025-04-01 16:09:24|
|         T0036|   U066|88464.76|stephanie50@yahoo...|2025-04-11 05:50:57|
|         T0049|   U050|93898.14|carlsonjames@gard...|2025-04-05 03:12:16|
|         T0052|   U088|70959.19|jessica48@hotmail...|2025-04-25 00:09:15|
|         T0060|   U265|80521.08|      kaitlynsalazar|2025-04-10 17:07:00|
|         T0063|   U098|87681.99|         rachelhayes|2025-04-13 16:25:19|
|         T0066|   U108|80296.12|    jill11@gmail.com|2025-04-03 09:51:20|
|         T0067|   U183|98103.36|danielramirez@hot...|2025-04-19 08:54:15|
|         T0075|   U131|89574.63|jonesgeorge@yahoo...|2025-04-14 00:16:53|
|         T0076|   U199|95746.19|              eric18|2025-03-29 22:51:17|
|         T0081|   U209|63408.75|    tara00@gmail.com|2025-04-22 15:38:34|
|         T0090|   U043|73488.49|   scott49@gmail.com|2025-04-08 18:42:41|
|         T0095|   U031|72250.11|    ryan82@brown.com|2025-04-06 08:18:57|
|         T0097|   U065|82322.29|    kaustin@soto.com|2025-04-18 15:16:49|
|         T0099|   U108|95527.61|walterelliott@yah...|2025-04-07 10:00:41|
|         T0100|   U044|64732.73|              ayoung|2025-04-10 10:08:57|
+--------------+-------+--------+--------------------+-------------------+
only showing top 20 rows
```

4. Hitung Berapa Banyak Outliers

```
[7]: print("Jumlah Outliers: ", outliers.count())
```

```
Jumlah Outliers:  158
```

Tugas Praktikum : Memakai file ecommerce_transactions_1000.csv (Belum di cleaning)

1. Tampilkan top 5 transaksi dengan amount terbesar ?
   Jawab : Top 5 transaksi dengan amount terbesar

```
[11]: from pyspark.sql.functions import col

amount5 = df.orderBy(col("amount").desc()).limit(5)
print("Top 5 transaksi dengan amount terbesar")
amount5.show()
```

```
Top 5 transaksi dengan amount terbesar
+--------------+-------+--------+--------------------+-------------------+
|transaction_id|user_id|  amount|               email|   transaction_time|
+--------------+-------+--------+--------------------+-------------------+
|         T0437|   U233|99830.84|franklincraig@gma...|2025-03-31 01:07:47|
|         T0175|   U224|99410.65|natalie63@hotmail...|2025-04-10 14:15:20|
|         T0320|   U046|99399.22|bonniemack@yahoo.com|2025-04-05 21:15:08|
|         T0115|   U148|98589.66|          hillsophia|2025-03-29 20:30:24|
|         T0451|   U293|98343.68|  sean46@walters.com|2025-04-17 14:27:35|
+--------------+-------+--------+--------------------+-------------------+
```

2. Hitung **jumlah total transaksi** ?
   Jawab :
   - Jumlah total transaksi yang pernah dilakukan :

```
[12]: total_transaksi = df.count()
print(f"Jumlah total transaksi: {total_transaksi}")
```

```
Jumlah total transaksi: 1000
```

   - Jumlah total pendapatan/ amount dari seluruh transaksi

```
[13]: from pyspark.sql.functions import sum as spark_sum

total_pendapatan = df.select(spark_sum("amount")).collect()[0][0]
print(f"Total pendapatan dari seluruh transaksi: {total_pendapatan}")
```

```
Total pendapatan dari seluruh transaksi: 19644994.95
```

3. Hitung **jumlah outlier** ?
   Jawab :

```
[14]: print("Jumlah Outliers: ", outliers.count())
```

```
Jumlah Outliers:  158
```

4. Hitung p**ersentase outlier** terhadap seluruh transaksi ?
   Jawab :

```
[17]: outliers_count = outliers.count()

total_transaksi = df.count()

persentase_outlier = (outliers_count / total_transaksi) * 100

print(f"Persentase outlier terhadap seluruh transaksi: {persentase_outlier:.2f}%")
```

```
Persentase outlier terhadap seluruh transaksi: 15.80%
```

Tugas Praktikum : Memakai file cleaned_transactions_1000.csv (Sesudah di cleaning)

1. Tampilkan top 5 transaksi dengan amount terbesar ?
   Jawab : Top 5 transaksi dengan amount terbesar

```
[52]: from pyspark.sql.functions import col

amount5 = df.orderBy(col("amount").desc()).limit(5)
print("Top 5 transaksi dengan amount terbesar")
amount5.show()
```

```
Top 5 transaksi dengan amount terbesar
+--------------+-------+--------+--------------------+-------------------+------------+----------------+
|transaction_id|user_id|  amount|               email|   transaction_time|email_domain|transaction_date|
+--------------+-------+--------+--------------------+-------------------+------------+----------------+
|         T0437|   U233|99830.84|franklincraig@gma...|2025-03-31 01:07:47|   gmail.com|      2025-03-31|
|         T0175|   U224|99410.65|natalie63@hotmail...|2025-04-10 14:15:20| hotmail.com|      2025-04-10|
|         T0320|   U046|99399.22|bonniemack@yahoo.com|2025-04-05 21:15:08|   yahoo.com|      2025-04-05|
|         T0451|   U293|98343.68|  sean46@walters.com|2025-04-17 14:27:35| walters.com|      2025-04-17|
|         T0067|   U183|98103.36|danielramirez@hot...|2025-04-19 08:54:15| hotmail.com|      2025-04-19|
+--------------+-------+--------+--------------------+-------------------+------------+----------------+
```

2. Hitung **jumlah total transaksi** ?
   Jawab :
   - Jumlah total transaksi yang pernah dilakukan :

```
[53]: total_transaksi = df.count()
print(f"Jumlah total transaksi: {total_transaksi}")
```

```
Jumlah total transaksi: 867
```

   - Jumlah total pendapatan/ amount dari seluruh transaksi

```
[54]: from pyspark.sql.functions import sum as spark_sum

total_pendapatan = df.select(spark_sum("amount")).collect()[0][0]
print(f"Total pendapatan dari seluruh transaksi: {total_pendapatan}")
```

```
Total pendapatan dari seluruh transaksi: 16922579.86999999
```

3. Hitung **jumlah outlier** ?
   Jawab :

```
[61]: quantiles = df.approxQuantile("amount", [0.25, 0.75], 0.05)
Q1, Q3 = quantiles
IQR = Q3 - Q1

lower_bound = Q1 -  1.5 * IQR
upper_bound = Q3 +  1.5 * IQR

print(f"Q1 = {Q1}, Q3 = {Q3}, IQR = {IQR}")
print(f"Lower Bound = {lower_bound}, Upper Bound = {upper_bound}")
outliers = df.filter((df.amount < lower_bound) | (df.amount > upper_bound))
print("Jumlah Outliers : ", outliers.count())
```

```
Q1 = 0.0, Q3 = 30029.83, IQR = 30029.83
Lower Bound = -45044.745, Upper Bound = 75074.57500000001
Jumlah Outliers :  86
```

4. Hitung p**ersentase outlier** terhadap seluruh transaksi ?
   Jawab :

```
[62]: outliers_count = outliers.count()

total_transaksi = df.count()

persentase_outlier = (outliers_count / total_transaksi) * 100
print(f"Persentase outlier terhadap seluruh transaksi: {persentase_outlier:.2f}%")
```

```
Persentase outlier terhadap seluruh transaksi: 9.92%
```