ASSESSMENT 3

- SRUTHY SUJI
- 23AD140
- AI-DS C

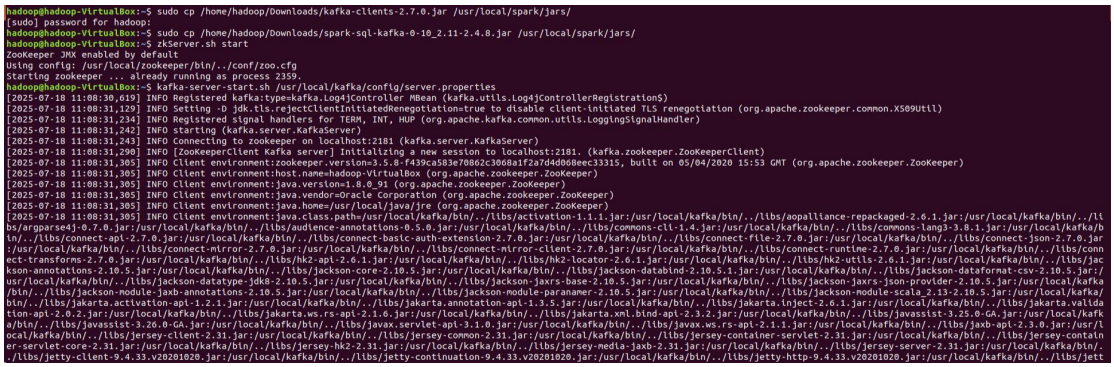
Problem Statement: Real-Time Monitoring and Optimization of FMCG Warehouse Operations Using Kafka and Spark Streaming

Objective:

Design a real-time data processing pipeline using Apache Kafka and Apache Spark Streaming to analyze and monitor operational metrics from multiple FMCG warehouses.

Tasks:

1. Write a Spark Streaming Program for Live alerts for warehouse breakdowns in last 3 months “wh_breakdown_l3m” more than 3 times. [Print Status: Alert or Normal]
2. Write a Spark Streaming Program for Live alerts for transport issues “transport_issue_l1y”, and storage issue “storage_issue_reported_l3m” more than 20. [Print Status: Alert or Normal]



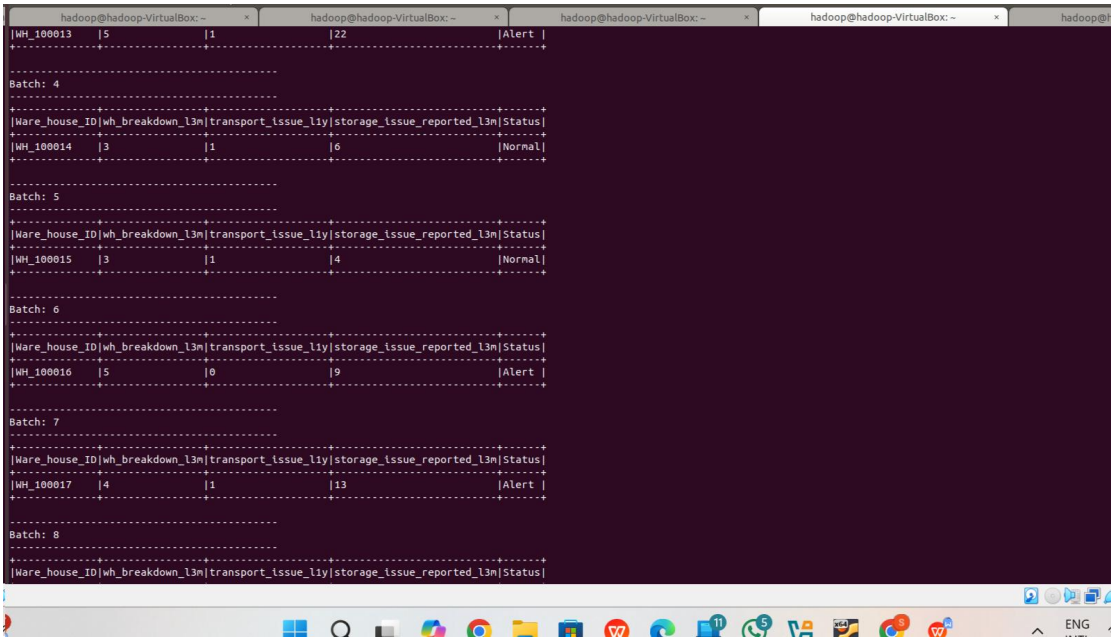
Producer codes:

```
hadoop@hadoop-VirtualBox: ~   hadoop@hadoop-VirtualBox: ~   hadoop@hadoop-VirtualBox: ~
 GNU nano 2.9.3                                              kafka_fmcg_producer.py

from kafka import KafkaProducer
import time
import pandas as pd
import json

# Load the warehouse alert CSV
df = pd.read_csv('/home/hadoop/Downloads/FMCG_data.csv')

# Ensure correct types
df['wh_breakdown_l3m'] = df['wh_breakdown_l3m'].astype(int)
df['transport_issue_l1y'] = df['transport_issue_l1y'].astype(int)
df['storage_issue_reported_l3m'] = df['storage_issue_reported_l3m'].astype(int)

# Create Kafka producer
producer = KafkaProducer(
    bootstrap_servers='localhost:9092',
    value_serializer=lambda v: json.dumps(v).encode('utf-8')
)

# Send each row as JSON to Kafka topic
for _, row in df.iterrows():
    message = {
        "Ware_house_ID": row['Ware_house_ID'],
        "wh_breakdown_l3m": row['wh_breakdown_l3m'],
        "transport_issue_l1y": row['transport_issue_l1y'],
        "storage_issue_reported_l3m": row['storage_issue_reported_l3m']
    }

    # Send message
    producer.send('fmcg_warehouse_data', message)
    print(f"Sent: {message}")
    time.sleep(1)  # simulate delay

# Flush to ensure all messages are sent
producer.flush()
```

```
hadoop@hadoop-VirtualBox:~$ python3 kafka_fmcg_producer.py
Sent: {'Ware_house_ID': 'WH_100000', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 13}
Sent: {'Ware_house_ID': 'WH_100001', 'wh_breakdown_l3m': 3, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 4}
Sent: {'Ware_house_ID': 'WH_100002', 'wh_breakdown_l3m': 6, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 17}
Sent: {'Ware_house_ID': 'WH_100003', 'wh_breakdown_l3m': 3, 'transport_issue_l1y': 4, 'storage_issue_reported_l3m': 17}
Sent: {'Ware_house_ID': 'WH_100004', 'wh_breakdown_l3m': 6, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 18}
Sent: {'Ware_house_ID': 'WH_100005', 'wh_breakdown_l3m': 3, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 23}
Sent: {'Ware_house_ID': 'WH_100006', 'wh_breakdown_l3m': 3, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 24}
Sent: {'Ware_house_ID': 'WH_100007', 'wh_breakdown_l3m': 6, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 18}
Sent: {'Ware_house_ID': 'WH_100008', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 13}
Sent: {'Ware_house_ID': 'WH_100009', 'wh_breakdown_l3m': 6, 'transport_issue_l1y': 3, 'storage_issue_reported_l3m': 6}
Sent: {'Ware_house_ID': 'WH_100010', 'wh_breakdown_l3m': 4, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 17}
Sent: {'Ware_house_ID': 'WH_100011', 'wh_breakdown_l3m': 2, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 11}
Sent: {'Ware_house_ID': 'WH_100012', 'wh_breakdown_l3m': 1, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 4}
Sent: {'Ware_house_ID': 'WH_100013', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 22}
Sent: {'Ware_house_ID': 'WH_100014', 'wh_breakdown_l3m': 3, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 6}
Sent: {'Ware_house_ID': 'WH_100015', 'wh_breakdown_l3m': 3, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 4}
Sent: {'Ware_house_ID': 'WH_100016', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 9}
Sent: {'Ware_house_ID': 'WH_100017', 'wh_breakdown_l3m': 4, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 13}
Sent: {'Ware_house_ID': 'WH_100018', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 29}
Sent: {'Ware_house_ID': 'WH_100019', 'wh_breakdown_l3m': 2, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 22}
Sent: {'Ware_house_ID': 'WH_100020', 'wh_breakdown_l3m': 4, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 19}
Sent: {'Ware_house_ID': 'WH_100021', 'wh_breakdown_l3m': 6, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 14}
Sent: {'Ware_house_ID': 'WH_100022', 'wh_breakdown_l3m': 6, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 28}
Sent: {'Ware_house_ID': 'WH_100023', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 25}
Sent: {'Ware_house_ID': 'WH_100024', 'wh_breakdown_l3m': 2, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 12}
Sent: {'Ware_house_ID': 'WH_100025', 'wh_breakdown_l3m': 3, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 22}
Sent: {'Ware_house_ID': 'WH_100026', 'wh_breakdown_l3m': 6, 'transport_issue_l1y': 3, 'storage_issue_reported_l3m': 8}
Sent: {'Ware_house_ID': 'WH_100027', 'wh_breakdown_l3m': 4, 'transport_issue_l1y': 3, 'storage_issue_reported_l3m': 11}
Sent: {'Ware_house_ID': 'WH_100028', 'wh_breakdown_l3m': 0, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 0}
Sent: {'Ware_house_ID': 'WH_100029', 'wh_breakdown_l3m': 1, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 6}
Sent: {'Ware_house_ID': 'WH_100030', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 12}
Sent: {'Ware_house_ID': 'WH_100031', 'wh_breakdown_l3m': 6, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 18}
Sent: {'Ware_house_ID': 'WH_100032', 'wh_breakdown_l3m': 4, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 34}
Sent: {'Ware_house_ID': 'WH_100033', 'wh_breakdown_l3m': 3, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 14}
Sent: {'Ware_house_ID': 'WH_100034', 'wh_breakdown_l3m': 1, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 16}
Sent: {'Ware_house_ID': 'WH_100035', 'wh_breakdown_l3m': 0, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 0}
Sent: {'Ware_house_ID': 'WH_100036', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 34}
Sent: {'Ware_house_ID': 'WH_100037', 'wh_breakdown_l3m': 4, 'transport_issue_l1y': 1, 'storage_issue_reported_l3m': 17}
Sent: {'Ware_house_ID': 'WH_100038', 'wh_breakdown_l3m': 0, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 0}
Sent: {'Ware_house_ID': 'WH_100039', 'wh_breakdown_l3m': 2, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 38}
Sent: {'Ware_house_ID': 'WH_100040', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 23}
Sent: {'Ware_house_ID': 'WH_100041', 'wh_breakdown_l3m': 5, 'transport_issue_l1y': 0, 'storage_issue_reported_l3m': 25}
Sent: {'Ware_house_ID': 'WH_100042', 'wh_breakdown_l3m': 3, 'transport_issue_l1y': 2, 'storage_issue_reported_l3m': 16}
```

Consumer codes:

```
hadoop@hadoop-VirtualBox: ~   hadoop@hadoop-VirtualBox: ~   hadoop@hadoop-VirtualBox: ~   hadoop@hadoop-Virtu
 GNU nano 2.9.3                                              kafka_fmcg_consumer.py

from pyspark.sql import SparkSession
from pyspark.sql.functions import from_json, col
from pyspark.sql.types import StructType, StringType, IntegerType

# Define schema based on the JSON structure you're sending
schema = StructType() \
    .add("Ware_house_ID", StringType()) \
    .add("wh_breakdown_l3m", IntegerType()) \
    .add("transport_issue_l1y", IntegerType()) \
    .add("storage_issue_reported_l3m", IntegerType()) \
    .add("Status", StringType())

# Create Spark Session
spark = SparkSession.builder \
    .appName("WarehouseKafkaConsumer") \
    .master("local[*]") \
    .getOrCreate()

spark.sparkContext.setLogLevel("WARN")

# Read stream from Kafka
raw_df = spark.readStream \
    .format("kafka") \
    .option("kafka.bootstrap.servers", "localhost:9092") \
    .option("subscribe", "fmcg_data-topic") \
    .option("startingOffsets", "latest") \
    .load()

# Parse JSON messages
parsed_df = raw_df.selectExpr("CAST(value AS STRING) as json_value") \
    .select(from_json(col("json_value"), schema).alias("data")) \
    .select("data.*")

# Write to console
query = parsed_df.writeStream \
    .outputMode("append") \
    .format("console") \
    .option("truncate", "false") \
    .start()
```



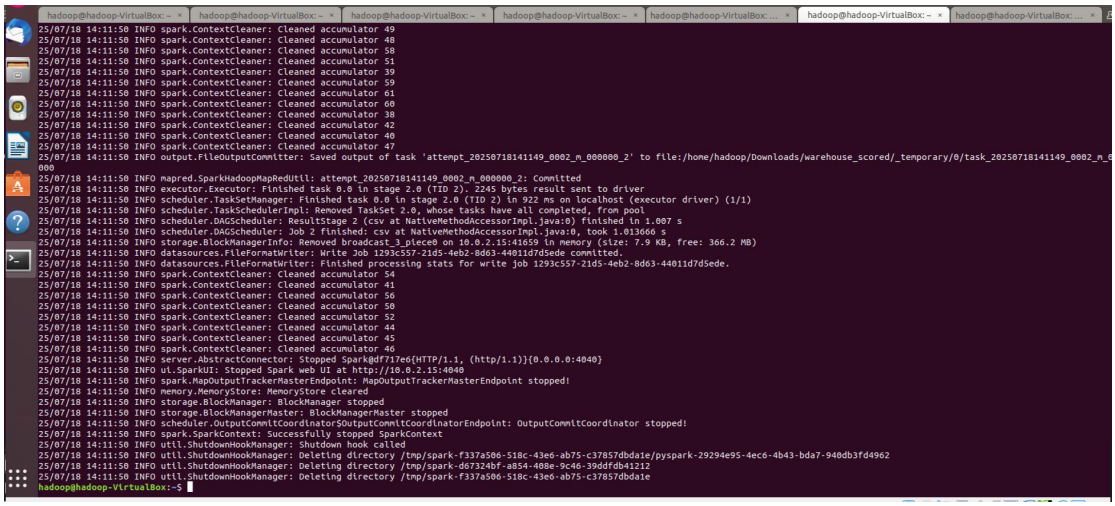
Objective: Design and implement a data analytics pipeline using Apache Spark for large-scale processing and Apache Airflow for scheduling and orchestration. The pipeline will:

- Ingest and clean warehouse data.
- Perform statistical and rule-based analysis on warehouse performance.
- Generate insights for operations, risk management, and compliance reporting.
- Automate periodic reporting.

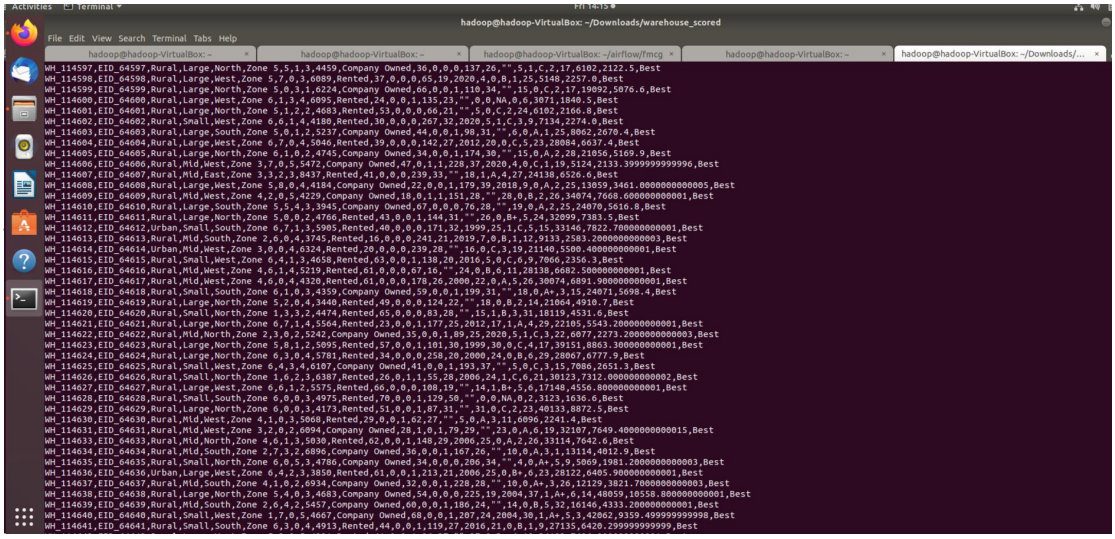
Tasks:

1. Warehouse Performance Scoring Use Fields: num_refill_req_l3m, product_wg_ton, workers_num, retail_shop_num, distributor_num • Task: Compute warehouse category based on demand and operational capacity i.e.
2. Use all five fields to find Best, Worst and Medium Category warehouse. [Use Condition to Categorize WH as Best, Medium and Worst]
3. Use Fields: num_refill_req_l3m, product_wg_ton, workers_num, retail_shop_num, distributor_num • Create two Spark Task files: 1) Data cleaning 2) Analysis File: Scoring formula application using DataFrame operations.
4. Airflow DAG: Run this daily, store results in local FS – “/home/Hadoop/Downloads/warehouse

Program has executed



Output has been saved in /home/hadoop/Downloads/warehouse_scored



DAG :