Task No: Use Case 4 Date:29/10/25	Messaging using PySpark and Hive	CO4 K3

AIM: To Build a Data Pipeline based on Messaging using PySpark and Hive.

PROCEDURE:

Step 1: Create a Spark session

Step 2: Ingest data from a source (e.g., CSV file)

Step 3: Data Transformation and Cleaning (e.g., filtering)

Step 4: Write data to HDFS (e.g., Parquet format)

Step 5: Create a Hive External Table

Step 6: Query and analyze data with Hive

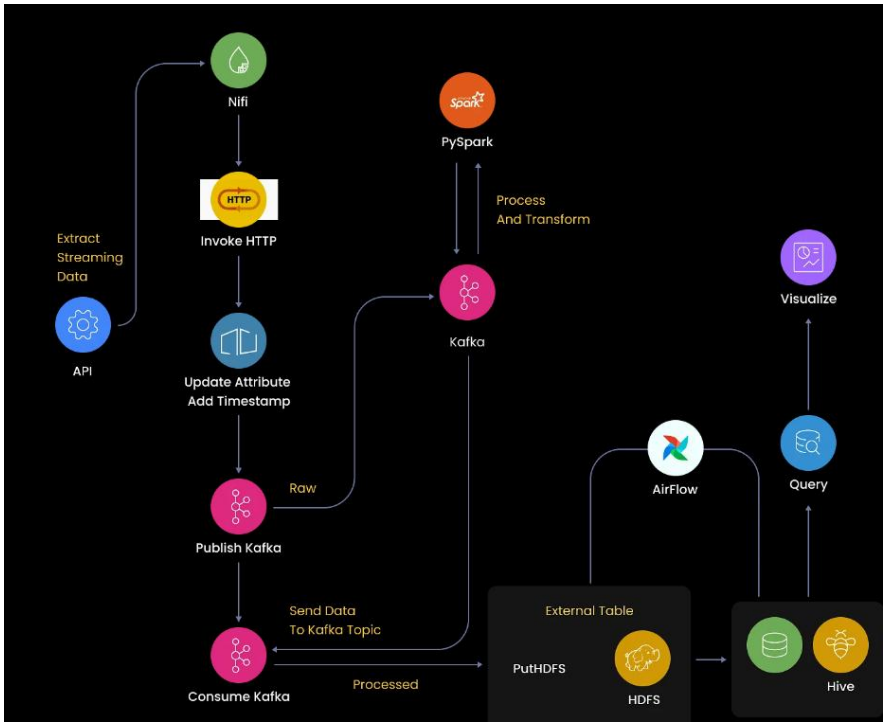
Step 7: Stop the Spark session

CODE:

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("DataPipeline").getOrCreate()
source_data = spark.read.csv("source_data.csv", header=True, inferSchema=True)
cleaned_data = source_data.filter(source_data["column_name"] > 10)
cleaned_data.write.mode("overwrite").parquet("hdfs:///user/hive/warehouse/cleaned_data")
spark.sql("""
   CREATE EXTERNAL TABLE cleaned_data (
      col1 STRING,
      col2 INT
   )
   ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
   STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.SequenceFileInputFormat'
   OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveSequenceFileOutputFormat'
   LOCATION 'hdfs:///user/hive/warehouse/cleaned_data'
""")

# You can now use Hive to run SQL queries on the data in the "cleaned_data" external table.
spark.stop()
```

IMPLEMENTATION:



RESULT:Thus to build a Data Pipeline based on Messaging using PySpark and Hive was executed successfully.