Assessment - 1

Name **:-** Ashish Gusai
Employee ID **:-** 10200
Mentors **:-** Chetan Khatri
& Babu Prabhakar

=> List of Given tasks:-

1. Install Hortonworks Data Platform (HDP) 2.6.5
2. Analyze UI's on HDP:
    a. Ambari UI
    b. Spark UI
    c. Yarn UI
    d. Spark History Server
3. Install Airflow and work with following Operators:
    a. Spark Submit Operator
    b. Bash Operator
    c. Python Operator
4. Work with Parquet and Avro files
5. Work with configuration files like .yaml and .conf
6. Task / Program:
    a. Read data from hdfs
    b. Write it into the database as a table
    c. Do categorical partition of that table data (eg. Dept_name)
    d. And write it into the parquet file on hdfs
    e. Read the partitioned parquet file from hdfs

- **Install Hortonworks Data Platform (HDP) 2.6.5**
    - **Problem statement :-**
        - Whenever i was trying to install the sandbox on virtual environment it's hang or freeze the system.
    - **Desired solution :-**
        - For smoothly running sandbox it's required 10 GB of RAM for vmware or virtualbox. So i've upgraded RAM to 16GB.
    - Also worked with sandbox shell and ran some spark jobs on it and analyze the changes in spark UI, history server, yarn UI, ect.

- **Analyze UI's on HDP:**
    - Ambari UI (8080)
        - Provides Hadoop management web UI.
    - Spark UI (4040)
        - web interface of a Spark application to monitor and inspect Spark job executions in a web browser.
    - Yarn UI (8088)
        - During the job run, by opening the Yarn UI in web browser, we can monitor the progress of the job.
    - Spark History Server (18080)
        - The Spark History Server displays information about the history of completed Spark applications.

- **Install Airflow and work with following Operators:**
    - **Spark Submit Operator**
        - **Problem statement** :- By default this operator is not come in the package of airflow. So, how to use it ?
        - **Desired Solution** :- by adding some plugins in the ~/airflow/plugin/ directory we can able to use it. Basically in the plugin file the bash operator is used. So we can say that it's based on bash operator.**(spark_operator_plugin.py)**
        - **Example:-** SparkSubmitOperator.py

- **Bash Operator**
  - Execute a Bash script, command or set of commands.
  - **Example:-** BashOperator.py
- **Python Operator**
  - Executes a Python callable
  - **Example:-**PythonOperator.py

- **Work with Parquet, Avro and other file formats**
  - **Parquet**
    - Parquet is a Column based format. If your data consists of lot of columns but you are interested in a subset of columns then you can use Parquet.
    - **Example:-** ParquetSpark.py
  - **Avro**
    - Avro is a Row based format. If you want to retrieve the data as a whole you can use Avro.
    - com.databricks.spark.avro this package name that we've to provide whenever we are submitting spark job.
    - **Example:-** AvroSpark.py
    - (spark-submit --packages org.apache.spark:spark-avro_2.11:2.4.0 avro_spark.py)
  - **Other Formats**
    - **Example:-** Other File Formats

- **Work with configuration files like .yaml and .conf**
  - **.yaml**
    - It's basically a human-readable structured data format.
    - **EXAMPLE:-** yaml_example
  - **.conf**
    - we can use --properties-file which should include parameters with starting keyword **spark** like (spark.driver.memory 5g
      spark.executor.memory 10g)
    - **EXAMPLE:-** Spark_Config.conf

- **Task / Program**
    - **Problem statement :-**
        - Read data from hdfs
        - Write it into the database as a table
        - Do categorical partition of that table data (eg. Dept_name)
        - And write it into the parquet file on hdfs
        - Read the partitioned parquet file from hdfs
    - **Desired Solution :-**
        - **Create a python file for above problem**
        - **Example:-** PartitionBy.py