BID DATA PROJECT-1

Group-7

Course: BUAN 6346

Term: Fall 2023

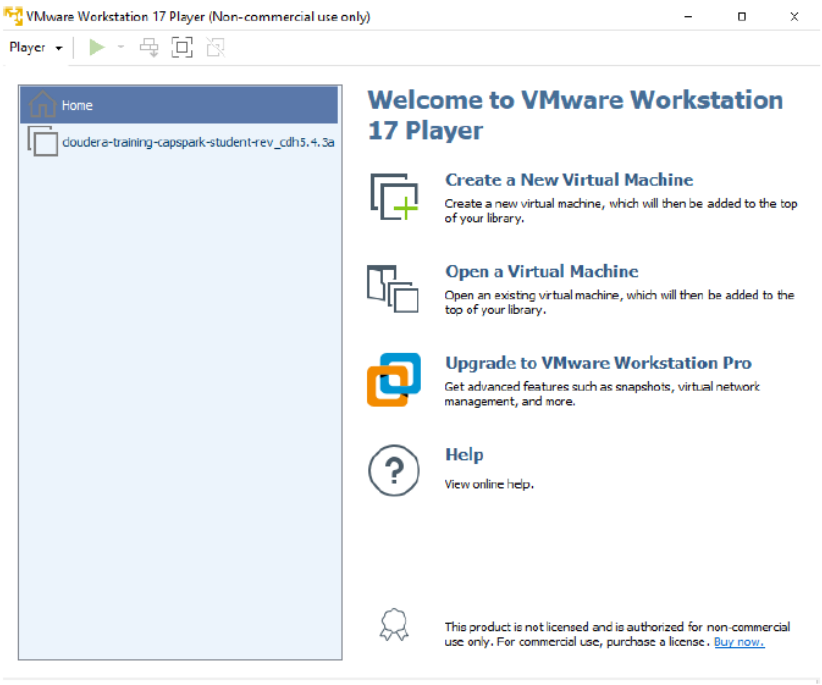
Professor: Farzad Kamalzadeh

Name: Kushagra Rastogi

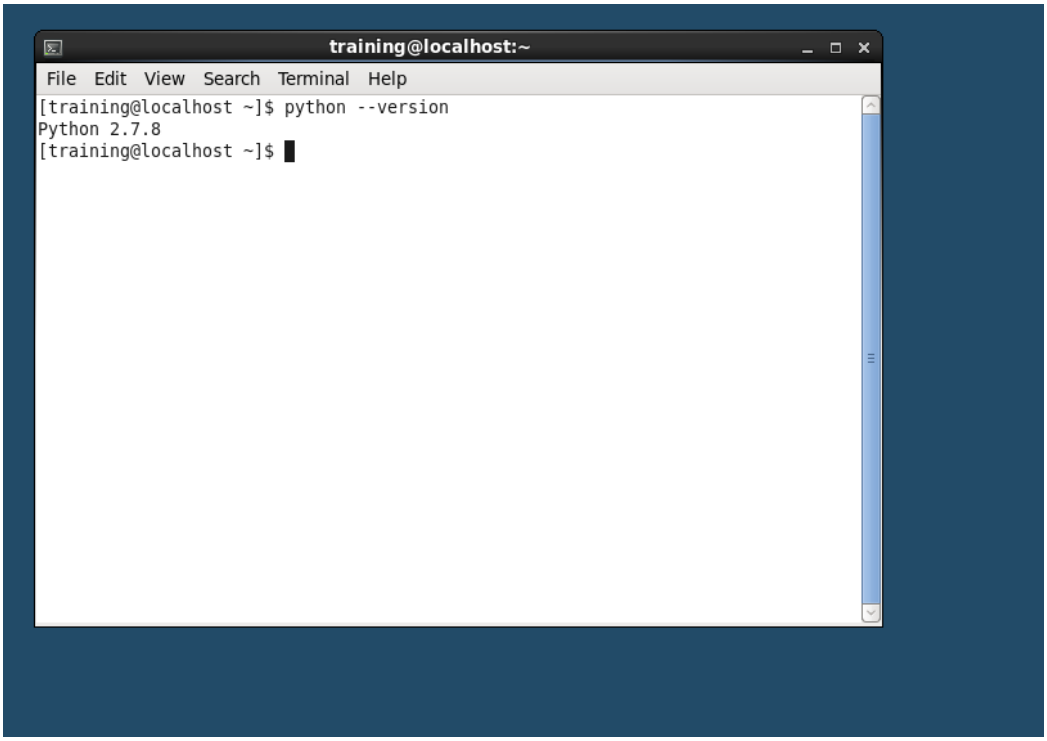
NetID: KXR220031

Step 0: VM Setup

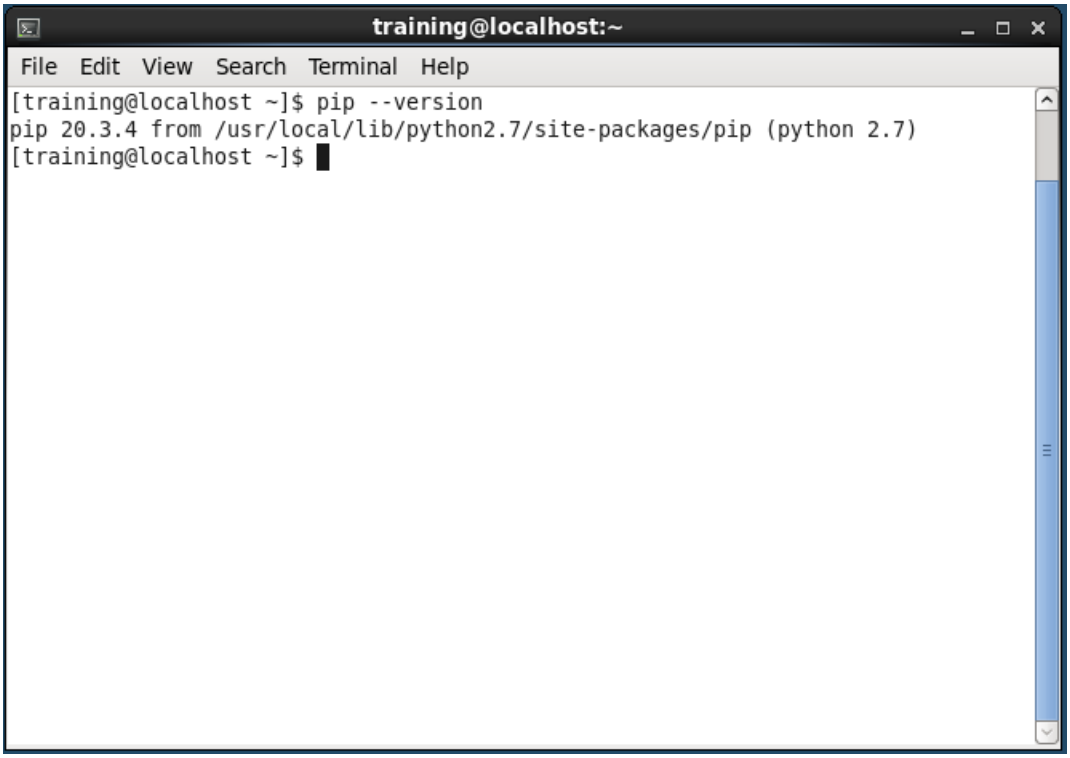
- Create a new virtual machine (VM) from the same image file.



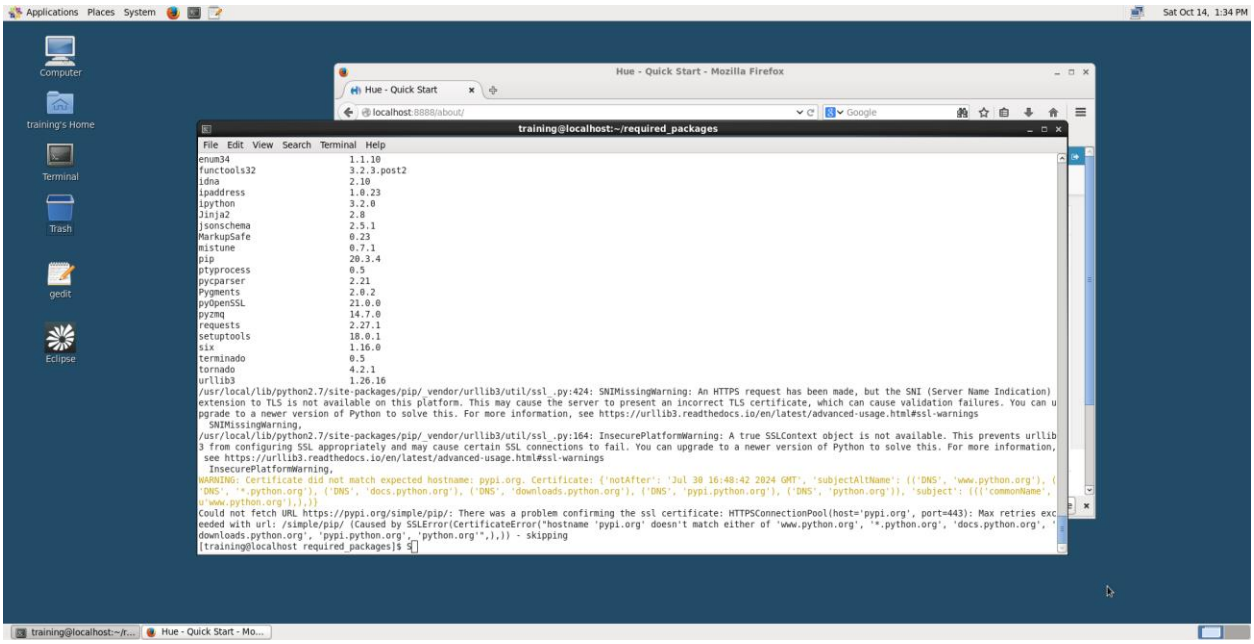
- Followed assignment 1 to enable all the services and set up your Hadoop.
- Change the Python default interpreter to Python2.7 and check version.



- Upgrade the pip and check version.



- Verify that all the packages have been installed by running the following: pip list



Step 1: Data Ingestion

A. Direct File Transfer

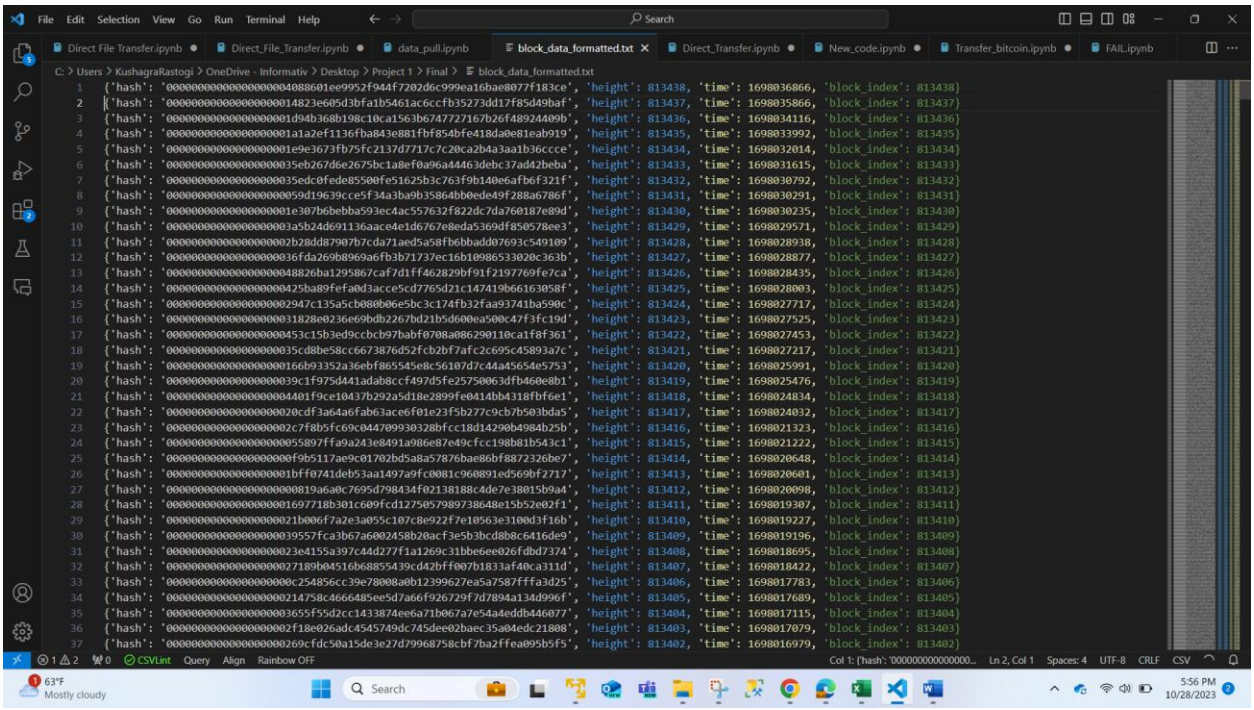
Python code

```python
import requests
from datetime import datetime, timedelta

# Function to pull block data for a specific date
def get_block_data(date):
    timestamp = int(date.timestamp()) * 1000
    url = f'https://blockchain.info/blocks/{timestamp}?format=json'
    response = requests.get(url)
    if response.status_code == 200:
        return response.json()
    else:
        print(f"Failed to fetch data for {date}")
        return None

# Specify the start date
start_date = datetime(2023, 10, 23)

# Create a list to store data for seven days
block_data_list = []

# Loop through seven days
for _ in range(7):
    daily_data = get_block_data(start_date)
    if daily_data:
        block_data_list.append(daily_data)
    start_date -= timedelta(days=1)

# Create a formatted text file
output_filename = "block_data_formatted.txt"

# Write each block's information to the text file
with open(output_filename, "w") as file:
    for daily_data in block_data_list:
        for block_info in daily_data:
            file.write(str(block_info) + "\n")

print(f"Data saved to {output_filename}")
```

✓ 9.1s

```
Data saved to block_data_formatted_data.txt
```

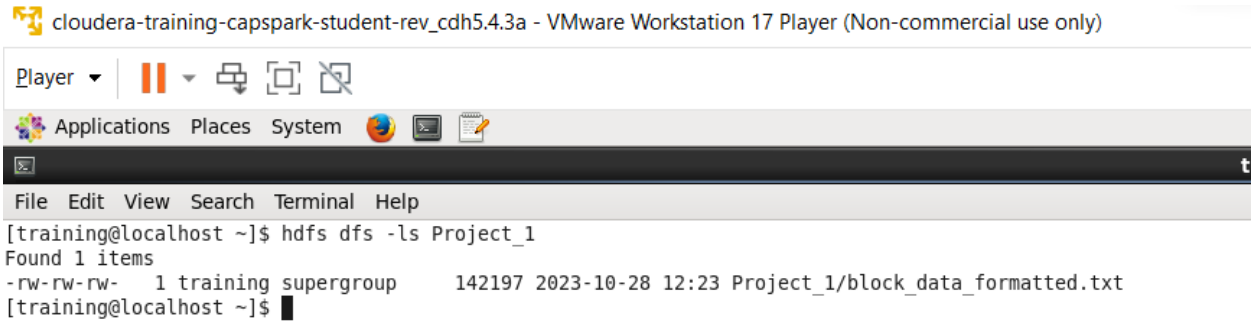
- Block_data_formatted.txt data output



Transfer Data to HDFS

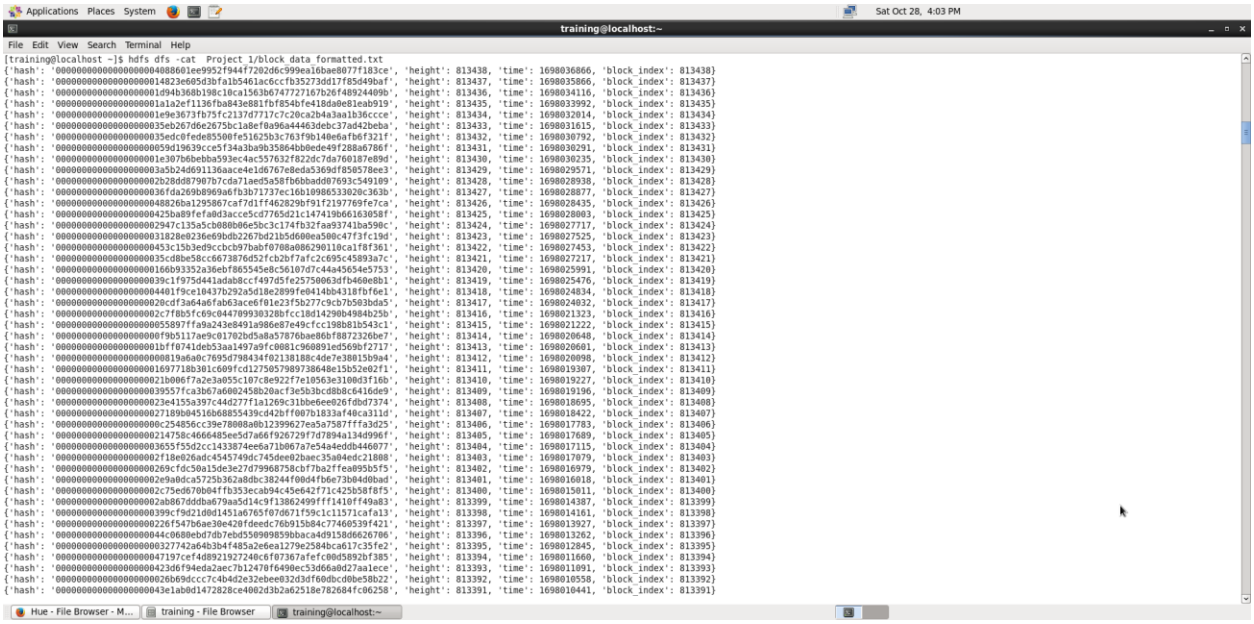
hdfs dfs -copyFromLocal /home/training/ block_data_formatted.txt /user/training/Project_1/

Verify the Copy:
hdfs dfs -ls /user/training/Project_1/

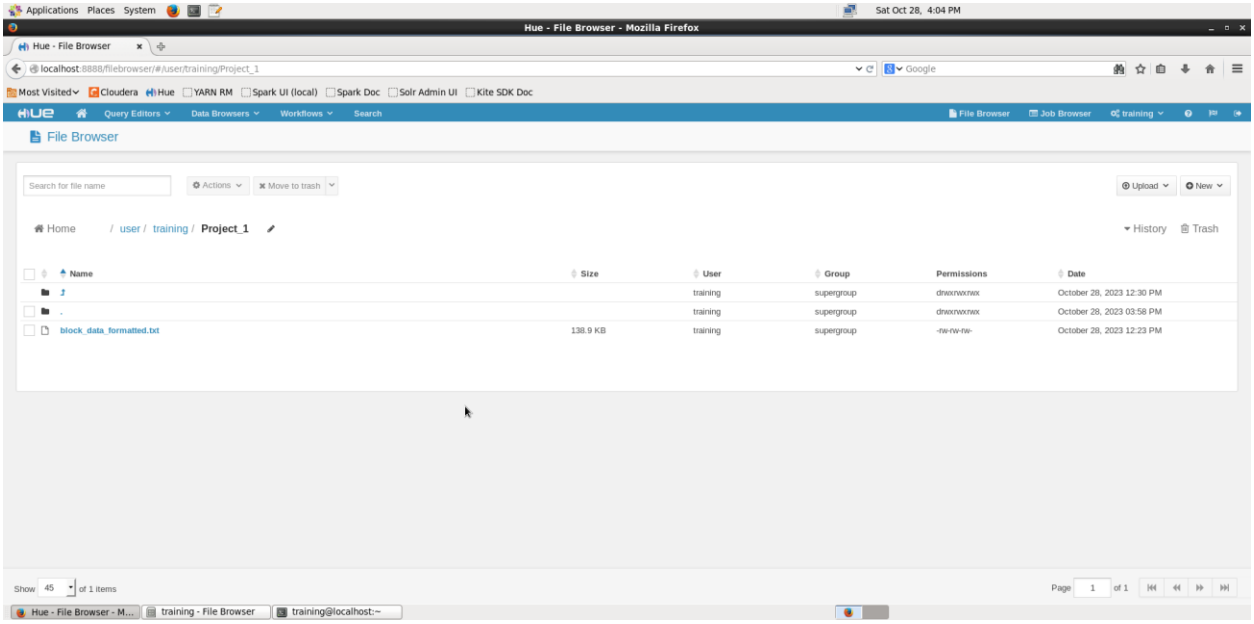


To check the contents of data in HDFS using following command

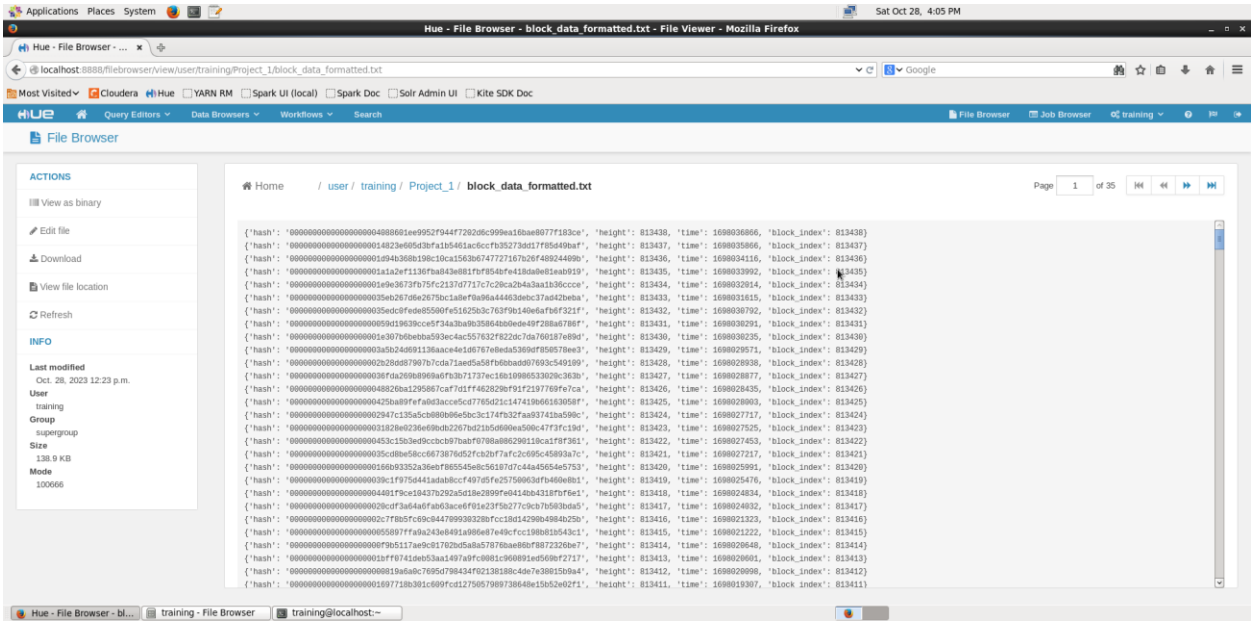
$ hdfs dfs -cat Project_1/block_data_formatted.txt



File In Hue:



Data in Hue:


Hue - File Browser - block_data_formatted.txt - File Viewer - Mozilla Firefox
File Browser
ACTIONS
View as binary
Edit file
Download
View file location
Refresh
INFO
Last modified
Oct. 28, 2023 12:23 p.m.
User
training
Group
supergroup
Size
138.9 KB
Mode
100666
Home / user / training / Project_1 / block_data_formatted.txt

B. Stream Ingestion using Flume.

- API key from Alpha Vantage:

  api_key = "IC61TNV1GKNGZL69"

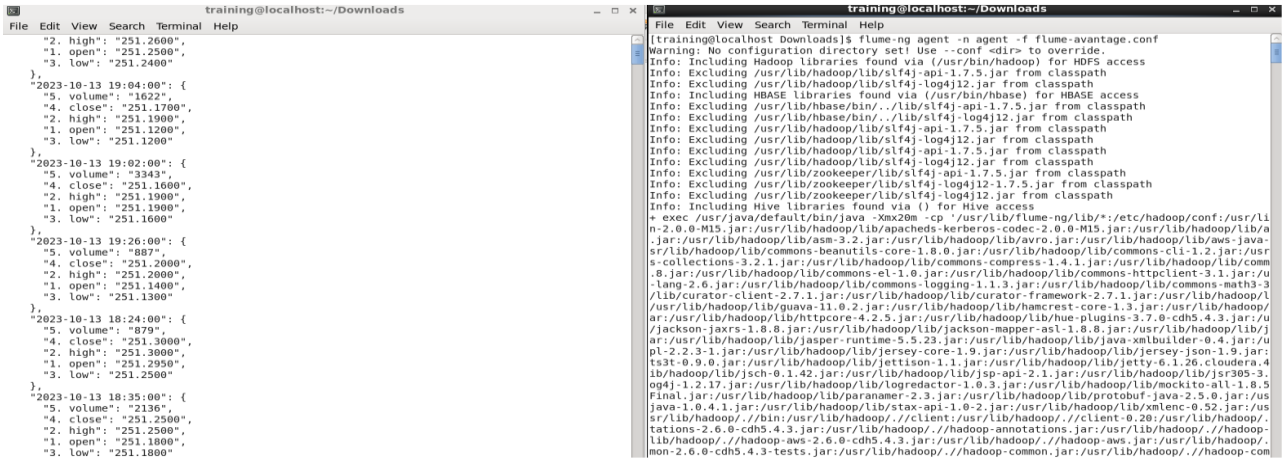
- Python code to pull data from API

```python
import requests
import json
import socket
import time

api_url = "https://www.alphavantage.co/query"
api_key = "IC61TNV1GKNGZL69"
symbols = ["AAPL", "IBM", "GOOGL", "MSFT", "TSLA"]

while True:
    for symbol in symbols:
        params = {
            "function": "TIME_SERIES_INTRADAY",  # Use lowercase "function"
            "symbol": symbol,  # Use the current symbol in the loop
            "interval": "1min",
            "apikey": api_key,
        }
        response = requests.get(api_url, params=params)
        data = response.json()

        print(json.dumps(data, indent=2))

        # Send the stock data to the local server on port 12345
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.connect(("localhost", 12345))
        s.send(json.dumps(data).encode())

        # Wait for 1 minute before downloading the next batch of data
        time.sleep(60)
```

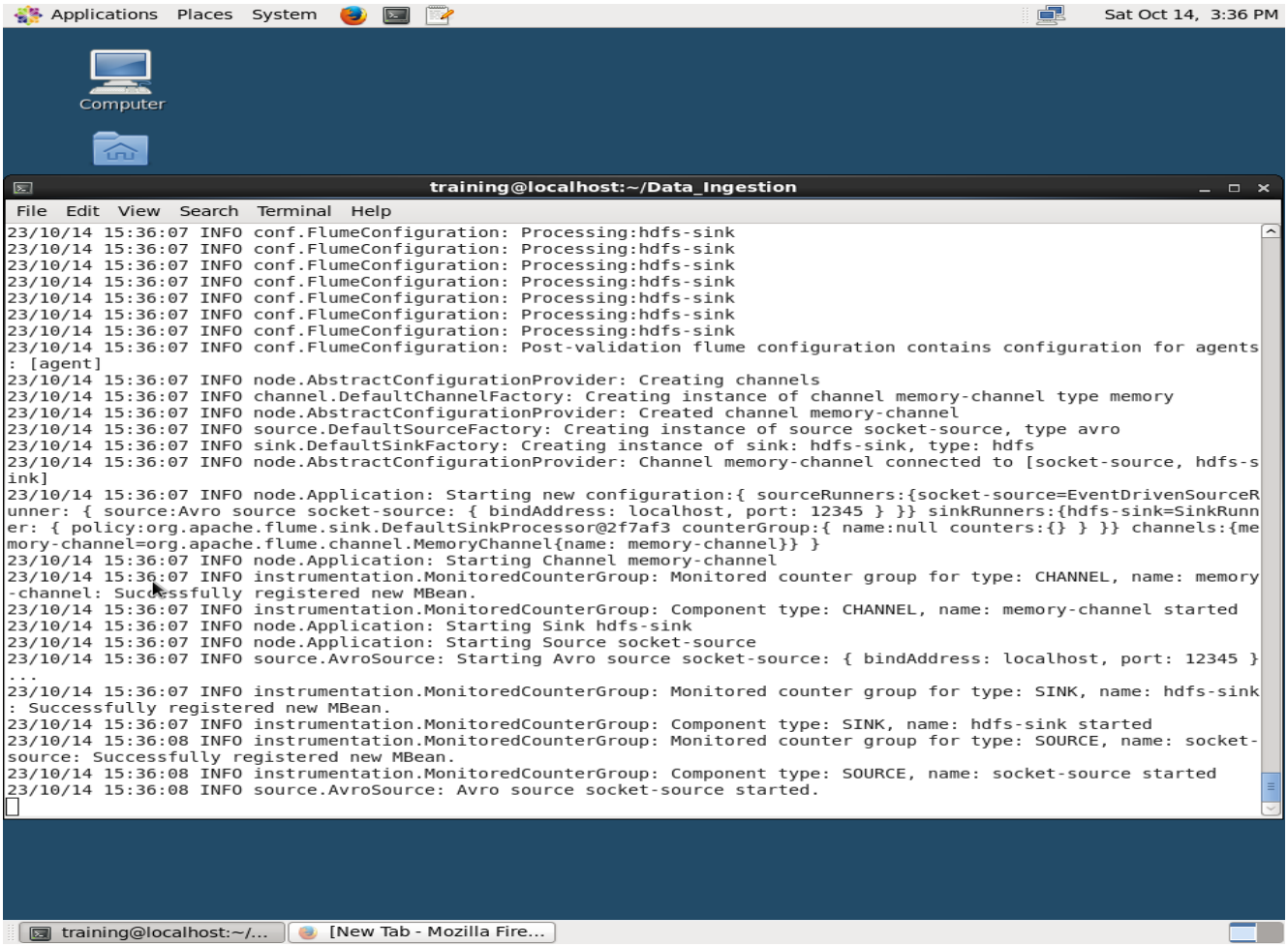
- **Screenshot of console output:**

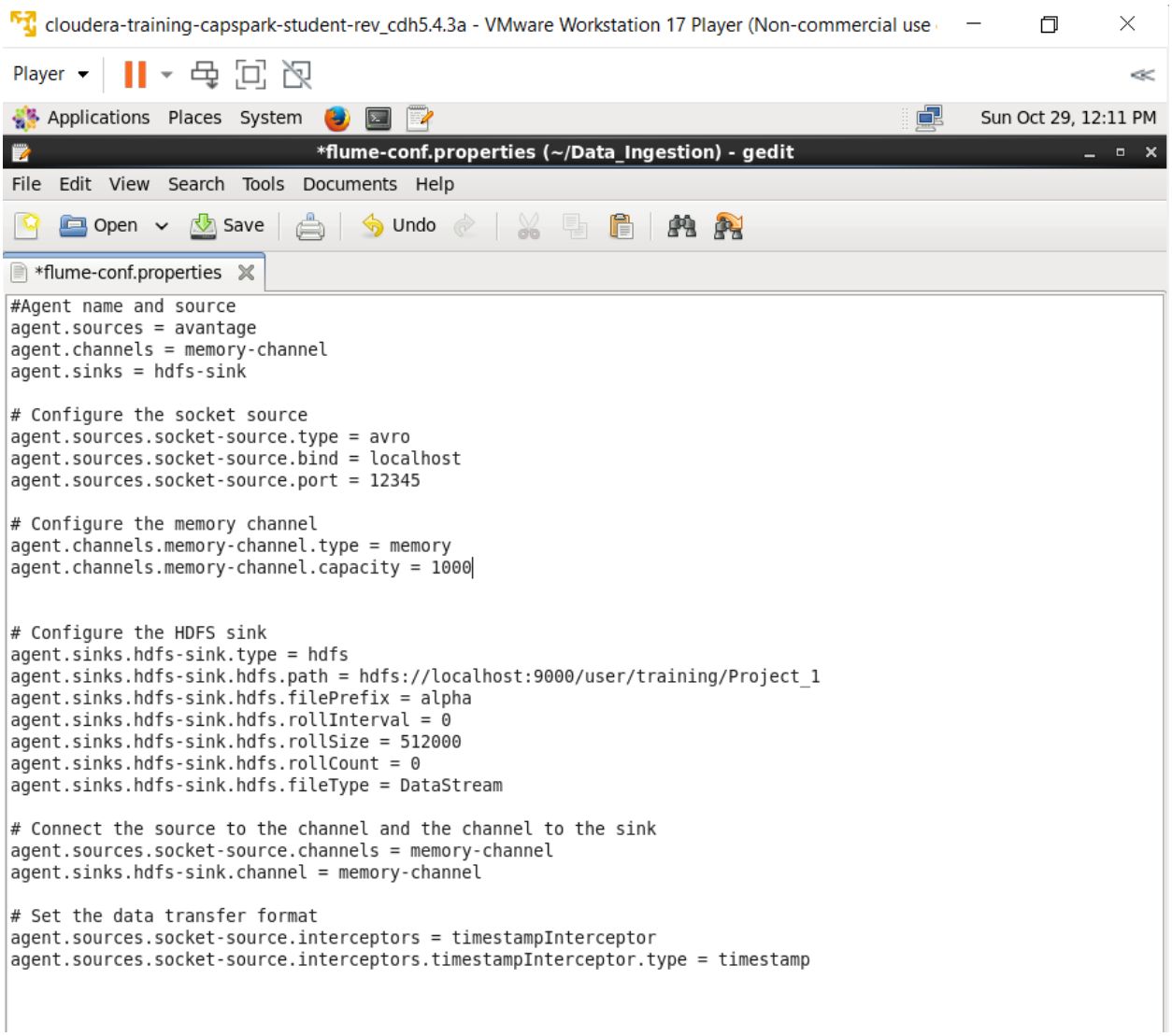

```
{
  "Meta Data": {
    "1. Information": "Intraday (1min) open, high, low, close prices and volume",
    "2. Symbol": "AAPL",
    "3. Last Refreshed": "2023-10-27 19:59:00",
    "4. Interval": "1min",
    "5. Output Size": "Compact",
    "6. Time Zone": "US/Eastern"
  },
  "Time Series (1min)": {
    "2023-10-27 19:59:00": {
      "1. open": "168.0110",
      "2. high": "168.0200",
      "3. low": "168.0100",
      "4. close": "168.0200",
      "5. volume": "432"
    },
    "2023-10-27 19:58:00": {
      "1. open": "168.0250",
      "2. high": "168.0400",
      "3. low": "168.0100",
      "4. close": "168.0200",
      "5. volume": "74"
    },
```

- **Screenshot of both Python code and Flume agent running side by side while the data streams**



- **Flume console output/log**




cloudera-training-capspark-student-rev_cdh5.4.3a - VMware Workstation 17 Player (Non-commercial use
Player
Applications Places System
Sun Oct 29, 12:11 PM
*flume-conf.properties (~/Data_Ingestion) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
*flume-conf.properties
#Agent name and source
agent.sources = avantage
agent.channels = memory-channel
agent.sinks = hdfs-sink
# Configure the socket source
agent.sources.socket-source.type = avro
agent.sources.socket-source.bind = localhost
agent.sources.socket-source.port = 12345
Configure the memory channel
agent.channels.memory-channel.type = memory
agent.channels.memory-channel.capacity = 1000
Configure the HDFS sink
agent.sinks.hdfs-sink.type = hdfs
agent.sinks.hdfs-sink.hdfs.path = hdfs://localhost:9000/user/training/Project_1
agent.sinks.hdfs-sink.hdfs.filePrefix = alpha
agent.sinks.hdfs-sink.hdfs.rollInterval = 0
agent.sinks.hdfs-sink.hdfs.rollSize = 512000
agent.sinks.hdfs-sink.hdfs.rollCount = 0
agent.sinks.hdfs-sink.hdfs.fileType = DataStream
Connect the source to the channel and the channel to the sink
agent.sources.socket-source.channels = memory-channel
agent.sinks.hdfs-sink.channel = memory-channel
Set the data transfer format
agent.sources.socket-source.interceptors = timestampInterceptor
agent.sources.socket-source.interceptors.timestampInterceptor.type = timestamp

C. Data Ingestion using Sqoop.

STEP 2 & 3

Table Uploaded and verified: -

- **Table 1: (**blocks_2023_Sep_10_to_15.csv**):**

```
mysql> LOAD DATA LOCAL INFILE '/home/training/Desktop/Project 1/blocks_2023_Sep_10_to_15.csv'
    -> INTO TABLE blocks_2023_Sep_10_to_15
    -> FIELDS TERMINATED BY ','
    -> LINES TERMINATED BY '\n'
    -> IGNORE 1 LINES;
ERROR 2 (HY000): File '/home/training/Desktop/Project 1/blocks_2023_Sep_10_to_15.csv' not found (Errcode: 2)
mysql> LOAD DATA LOCAL INFILE 'blocks_2023_Sep_10_to_15.csv'
    -> INTO TABLE blocks_2023_Sep_10_to_15
    -> FIELDS TERMINATED BY ','
    -> LINES TERMINATED BY '\n'
    -> IGNORE 1 LINES;
Query OK, 919 rows affected (0.01 sec)
Records: 920  Deleted: 0  Skipped: 1  Warnings: 0
```

Verify the uploads using MYSQL.

```
mysql> SELECT * FROM blocks_2023_Sep_10_to_15 LIMIT 10;
+----+------------------------------------------------------------------+---------------------+-------------+
| id | hash                                                             | time                | block_index |
+----+------------------------------------------------------------------+---------------------+-------------+
|  1 | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 | 2023-09-10 00:00:00 |      806982 |
|  2 | 00000000000000000000372109b9a114633512587c8b074910a4bc02921828b59 | 2023-09-10 00:00:00 |      806980 |
|  3 | 000000000000000000000438def0efe8257f6ff025665074dbcf7e1fda070aff30 | 2023-09-10 00:00:00 |      806979 |
|  4 | 00000000000000000000022e8847528953ebe30fc81867dc9ab70b50ce2660d5df | 2023-09-10 00:00:00 |      806978 |
|  5 | 00000000000000000000051355825a2eee832e8c068a3aa355ac6a28bed2d29472 | 2023-09-10 00:00:00 |      806977 |
|  6 | 00000000000000000000934a50f928c4572cc5d32276ffb55bd8dced508d728d | 2023-09-10 00:00:00 |      806976 |
|  7 | 000000000000000000002914218e992201381ea1fb26cb2fa963a50df844051b6 | 2023-09-10 00:00:00 |      806975 |
|  8 | 00000000000000000000047ea53323c0e8c8609f5dfea221fef380d20a12978d8 | 2023-09-10 00:00:00 |      806974 |
|  9 | 000000000000000000004f5d5af8c8532c5560a2e3f456d9f9fef7a867f3e7c8c | 2023-09-10 00:00:00 |      806973 |
| 10 | 00000000000000000000044e257ee716989fb570589520ead0e9e05fe3458bf155 | 2023-09-10 00:00:00 |      806972 |
+----+------------------------------------------------------------------+---------------------+-------------+
10 rows in set (0.00 sec)
```

SQL CODE:

-- Create a table for File 1

```
CREATE TABLE blocks_2023_Sep_10_to_15 (
    id INT AUTO_INCREMENT PRIMARY KEY,
    hash VARCHAR(64),
    time DATETIME,
    block_index INT
);
```

-- Load data from File 1

LOAD DATA LOCAL INFILE 'blocks_2023_Sep_10_to_15.csv'

INTO TABLE blocks_2023_Sep_10_to_15

FIELDS TERMINATED BY ','

LINES TERMINATED BY '\n'

IGNORE 1 LINES; -- This skips the header line in the CSV

- **Table 2: (**blocks_info_2023_Sep_10_to_15.csv**):**

```
mysql> CREATE TABLE blocks_info_2023_Sep_10_to_15 (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     hash VARCHAR(64),
    ->     ver INT,
    ->     bits INT,
    ->     fee INT,
    ->     nonce INT,
    ->     size INT,
    ->     block_index INT,
    ->     main_chain BOOLEAN,
    ->     height INT,
    ->     weight INT
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> LOAD DATA LOCAL INFILE '/home/training/blocks_info_2023_Sep_10_to_15.csv'
    -> INTO TABLE blocks_info_2023_Sep_10_to_15
    -> FIELDS TERMINATED BY ','
    -> LINES TERMINATED BY '\n'
    -> IGNORE 1 LINES;
Query OK, 309 rows affected, 448 warnings (0.01 sec)
Records: 310  Deleted: 0  Skipped: 1  Warnings: 138
```

Verify the uploads using MYSQL and SQOOP

```
mysql> SELECT * FROM blocks_info_2023_Sep_10_to_15 LIMIT 10;
+----+------------------------------------------------------------------+-----------+-----------+---------+------------+---------+-------------+------------+--------+---------+
| id | hash                                                             | ver       | bits      | fee     | nonce      | size    | block_index | main_chain | height | weight  |
+----+------------------------------------------------------------------+-----------+-----------+---------+------------+---------+-------------+------------+--------+---------+
|  1 | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 | 545259520 | 386216622 | 17380913 | 2147483647 | 1733081 |      806982 |          0 | 806982 | 3993482 |
|  2 | 00000000000000000000372109b9a114633512587c8b074910a4bc02921828b59 | 547356672 | 386216622 | 15397499 | 2147483647 | 1797458 |      806980 |          0 | 806980 | 3993365 |
|  3 | 00000000000000000000438def0efe8257f6ff025665074dbcf7e1fda070aff30 | 887709696 | 386216622 | 14746325 | 2147483647 | 1913649 |      806979 |          0 | 806979 | 3993468 |
|  4 | 00000000000000000000022e8847528953ebe30fc81867dc9ab70b50ce2660d5df | 536870912 | 386216622 | 17157713 |  700034109 | 1695531 |      806978 |          0 | 806978 | 3993549 |
|  5 | 00000000000000000000051355825a2eee832e8c068a3aa355ac6a28bed2d29472 | 536928256 | 386216622 | 14267496 |  690643341 | 1117137 |      806977 |          0 | 806977 | 3992862 |
|  6 | 00000000000000000000000934a50f928c4572cc5d32276ffb55bd8dced508d728d | 587145216 | 386216622 | 16096184 | 1707479123 | 2004003 |      806976 |          0 | 806976 | 3993648 |
|  7 | 00000000000000000000002914218e992201381ea1fb26cb2fa963a50df844051b6 | 536928256 | 386216622 | 16022431 |  739931405 | 1466355 |      806975 |          0 | 806975 | 3993528 |
|  8 | 00000000000000000000000047ea53323c0e8c8609f5dfea221fef380d20a12978d8 | 538157056 | 386216622 | 15046341 | 1066014781 | 1941877 |      806974 |          0 | 806974 | 3993634 |
|  9 | 00000000000000000000004f5d5af8c8532c5560a2e3f456d9f9fef7a867f3e7c8c | 602439680 | 386216622 | 15317484 | 2147483647 | 1594967 |      806973 |          0 | 806973 | 3992672 |
| 10 | 00000000000000000000044e257ee716989fb570589520ead0e9e05fe3458bf155 | 745529344 | 386216622 | 15169136 | 2147483647 | 1885826 |      806972 |          0 | 806972 | 3993056 |
+----+------------------------------------------------------------------+-----------+-----------+---------+------------+---------+-------------+------------+--------+---------+
10 rows in set (0.00 sec)
```

SQL CODE:

-- Create a table for File 2

```
CREATE TABLE blocks_info_2023_Sep_10_to_15 (
   id INT AUTO_INCREMENT PRIMARY KEY,
   hash VARCHAR(64),
   ver INT,
   bits INT,
   fee INT,
   nonce INT,
   size INT,
   block_index INT,
   main_chain BOOLEAN,
   height INT,
   weight INT
);
```

-- Load data from File 2

```
LOAD DATA LOCAL INFILE '/home/training/blocks_info_2023_Sep_10_to_15.csv'
INTO TABLE blocks_info_2023_Sep_10_to_15
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES; -- This skips the header line in the CSV
```

- **Table 3: (**tx_info_2023_Sep_10_to_15.csv**):**

```
mysql> CREATE TABLE tx_info_2023_Sep_10_to_15 (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     tx_hash VARCHAR(64),
    ->     block_hash VARCHAR(64),
    ->     ver INT,
    ->     vin_sz INT,
    ->     vout_sz INT,
    ->     size INT,
    ->     weight INT,
    ->     fee INT,
    ->     relayed_by VARCHAR(255),
    ->     lock_time INT,
    ->     tx_index BIGINT,
    ->     double_spend BOOLEAN,
    ->     time INT,
    ->     block_index INT,
    ->     block_height INT
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql> -- Load data from File 3
mysql> LOAD DATA LOCAL INFILE '/home/training/tx_info_2023_Sep_10_to_15.csv'
    -> INTO TABLE tx_info_2023_Sep_10_to_15
    -> FIELDS TERMINATED BY ','
    -> LINES TERMINATED BY '\n'
    -> IGNORE 1 LINES;
Query OK, 1197103 rows affected, 65535 warnings (3.97 sec)
Records: 1197104  Deleted: 0  Skipped: 1  Warnings: 5
```

Verify the uploads using MYSQL.

```
mysql> SELECT * FROM tx_info_2023_Sep_10_to_15 LIMIT 10;
+----+------------------------------------------------------------------+------------------------------------------------------------------+------+--------+---------+------+--------+------+-------------+-----------+---------------------+-----------------+
| id | tx_hash                                                          | block_hash                                                       | ver  | vin_sz | vout_sz | size | weight | fee  | relayed_by  | lock_time | tx_index            | double_spend    |
| time       | block_index | block_height |
+----+------------------------------------------------------------------+------------------------------------------------------------------+------+--------+---------+------+--------+------+-------------+-----------+---------------------+-----------------+
|  1 | ee6337e93ffc48b7ef212e0069d6ddb19148eac2c52c7642d36c3e417e9ca119 | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    1 |      1 |       4 |  351 |   1296 |    0 | 0.0.0.0     |         0 |  901820990367693 |               0 |
|  2 | 6790aee6b2cc9af0b581a3e7b32bc241ddbb9134a99d5533e00eed34505f1911 | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    1 |      1 |       2 |  380 |    758 | 50000 | 0.0.0.0     |         0 |  601621470289313 |               0 |
|  3 | 4cef857be937250098114e0fc6ab22cc71bf3f3d73e1b627f8fb6c95c2add06c | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    2 |      1 |       1 |  193 |    442 | 24400 | 0.0.0.0     |         0 | 382859277655391 |               0 |
|  4 | 79416846ff1075979648918251 51c5321fca5e688a2b6f16e9c372279bca971a | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    2 |      1 |       1 |  191 |    437 | 20618 | 0.0.0.0     |         0 |  935655729589848 |               0 |
|  5 | 0cdedab0c86b3eaa28a94781bbdaf82eaa4fbf389d862edde7602c4141c09a72 | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    1 |      1 |       2 |  223 |    892 | 26760 | 0.0.0.0     |         0 | 4032287233025420 |               0 |
|  6 | 7c6e3154af81cddc80aea57525d8a5afb89d2f82c84a5ac487bf7cbb891592b7c | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    2 |      1 |       2 |  223 |    892 | 26300 | 0.0.0.0     |         0 | 4368820101126526 |               0 |
|  7 | 658c47fc9a582fd72cdc14fb9a5a825a8db4da0957a8d316cfee1f73f23c3cd1 | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    2 |      1 |       2 |  223 |    892 | 26300 | 0.0.0.0     |         0 | 7361812824482813 |               0 |
|  8 | 3ee9eff6e8122992 95f0e56963218cf7932d902d7eac16f9288497f9f531b7d3 | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    2 |      1 |       2 |  223 |    892 | 26300 | 0.0.0.0     |         0 | 7449080661750512 |               0 |
|  9 | 75a89122b40429999c4d7332c424bde25f7ae299a5f4ef854f6d6e5676638550 | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    2 |      1 |       2 |  224 |    896 | 26300 | 0.0.0.0     |         0 | 2833082546310605 |               0 |
| 10 | b2da0d946cc8c1f5b181a8d9285b6bc9cc5560ca9967351f6f86d337443ddb7e | 00000000000000000000540268ddfc73d8cd7348eb48695fe4a602708c89b2e4 |    2 |      1 |       2 |  224 |    896 | 26300 | 0.0.0.0     |         0 | 4463362906192496 |               0 |
+----+------------------------------------------------------------------+------------------------------------------------------------------+------+--------+---------+------+--------+------+-------------+-----------+---------------------+-----------------+
10 rows in set (0.00 sec)
```

SQL CODE:

-- Create a table for File 3

CREATE TABLE tx_info_2023_Sep_10_to_15 (

```
    id INT AUTO_INCREMENT PRIMARY KEY,
    tx_hash VARCHAR(64),
    block_hash VARCHAR(64),
    ver INT,
    vin_sz INT,
    vout_sz INT,
    size INT,
    weight INT,
    fee INT,
    relayed_by VARCHAR(255),
    lock_time INT,
    tx_index BIGINT,
    double_spend BOOLEAN,
    time INT,
    block_index INT,
    block_height INT
);
```

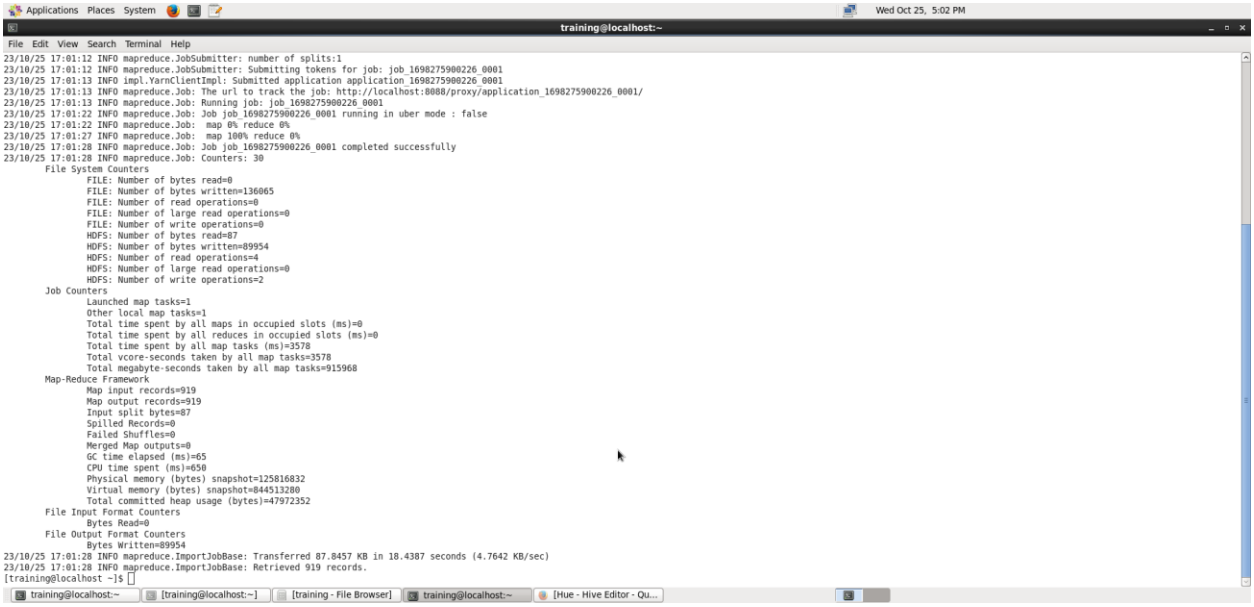
-- Load data from File 3

```
LOAD DATA LOCAL INFILE '/home/training/tx_info_2023_Sep_10_to_15.csv'
INTO TABLE tx_info_2023_Sep_10_to_15
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES; -- This skips the header line in the CSV
```

- **Sqoop command:**

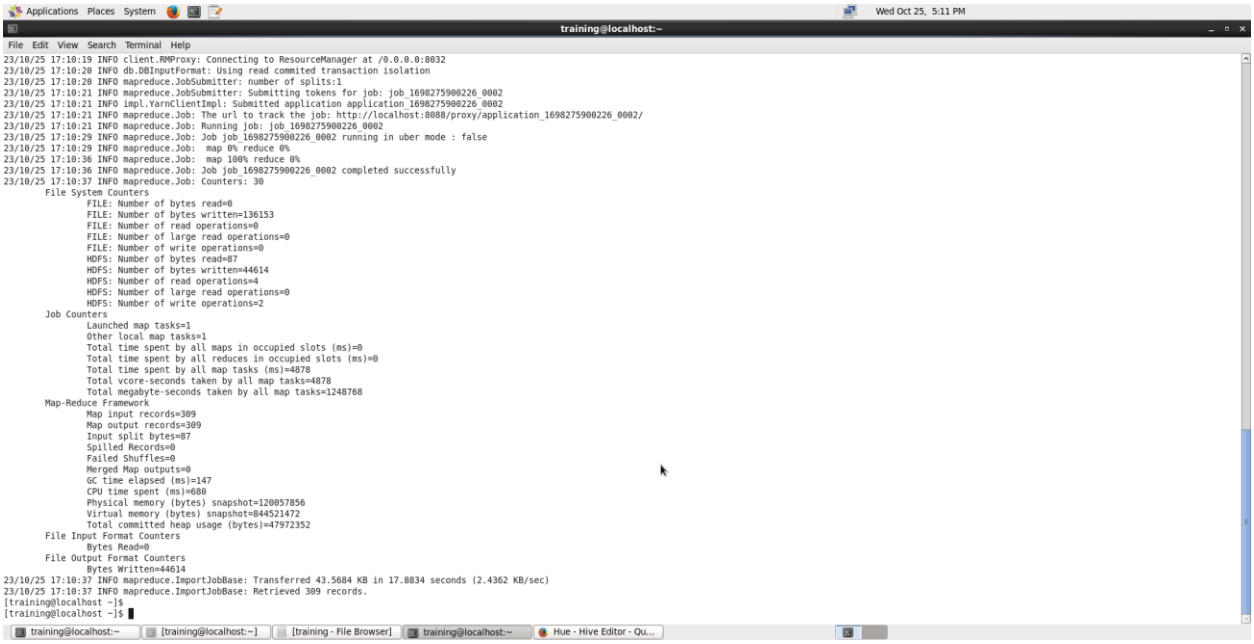
sqoop import --connect jdbc:mysql://localhost:3306/loudacre --username training --password training --table blocks_2023_Sep_10_to_15 --target-dir /user/hive/warehouse/your_hive_database.db/blocks_2023_Sep_10_to_15 --m 1

Result:



- **Sqoop command:**
  sqoop import --connect jdbc:mysql://localhost:3306/loudacre --username training --password training --table blocks_info_2023_Sep_10_to_15 --target-dir /user/hive/warehouse/your_hive_database.db/blocks_info_2023_Sep_10_to_15 -m 1

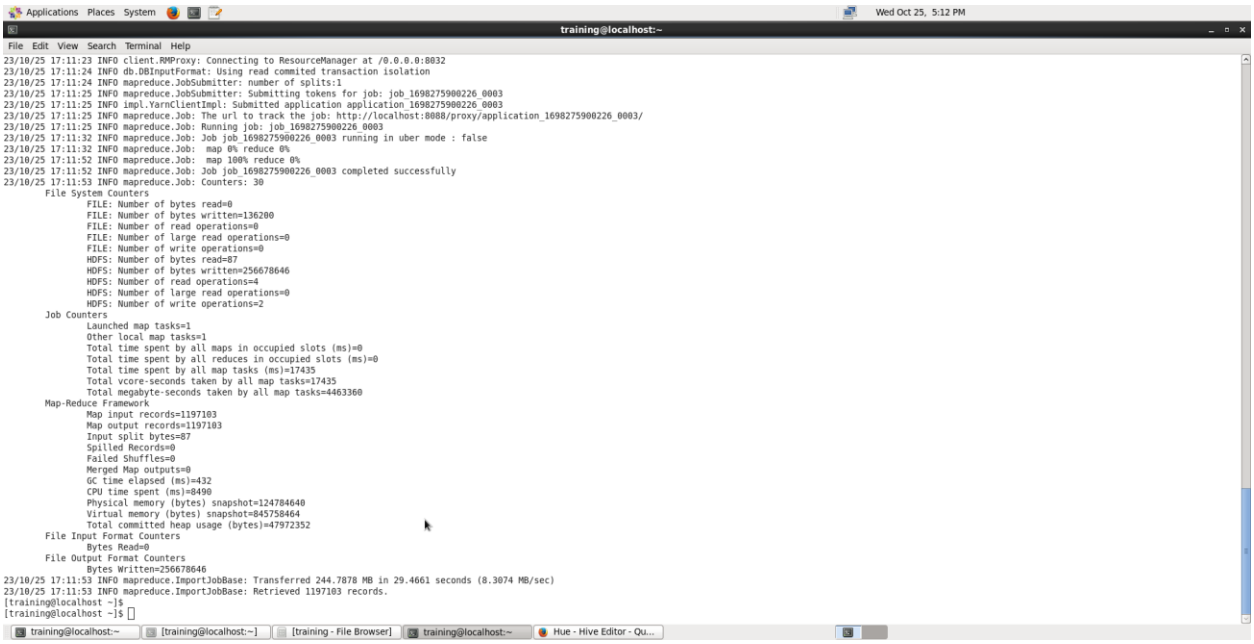
Result:



- **Sqoop command:**

  sqoop import --connect jdbc:mysql://localhost:3306/loudacre --username training --password training --table tx_info_2023_Sep_10_to_15 --target-dir /user/hive/warehouse/your_hive_database.db/tx_info_2023_Sep_10_to_15 -m 1

Result:

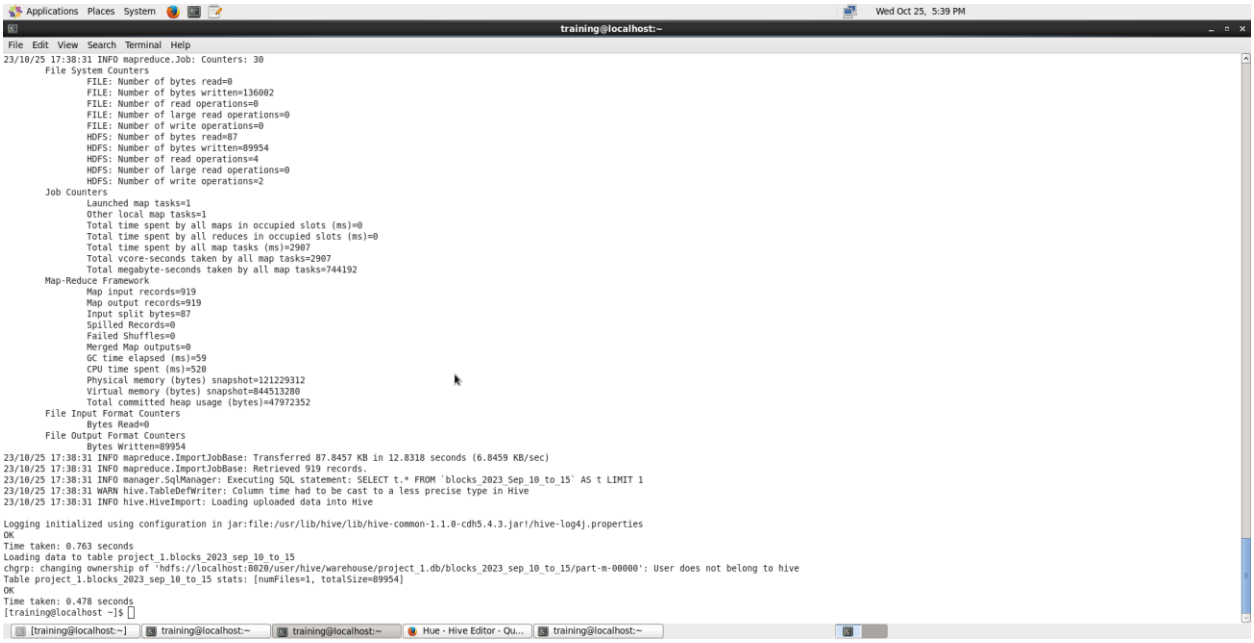


STEP 4

Import each table from MySQL directly into Hive:

- **Sqoop command:**
  sqoop import --connect jdbc:mysql://localhost:3306/loudacre --username training --password training --table blocks_2023_Sep_10_to_15 --hive-import --hive-table project_1.blocks_2023_Sep_10_to_15 --m 1

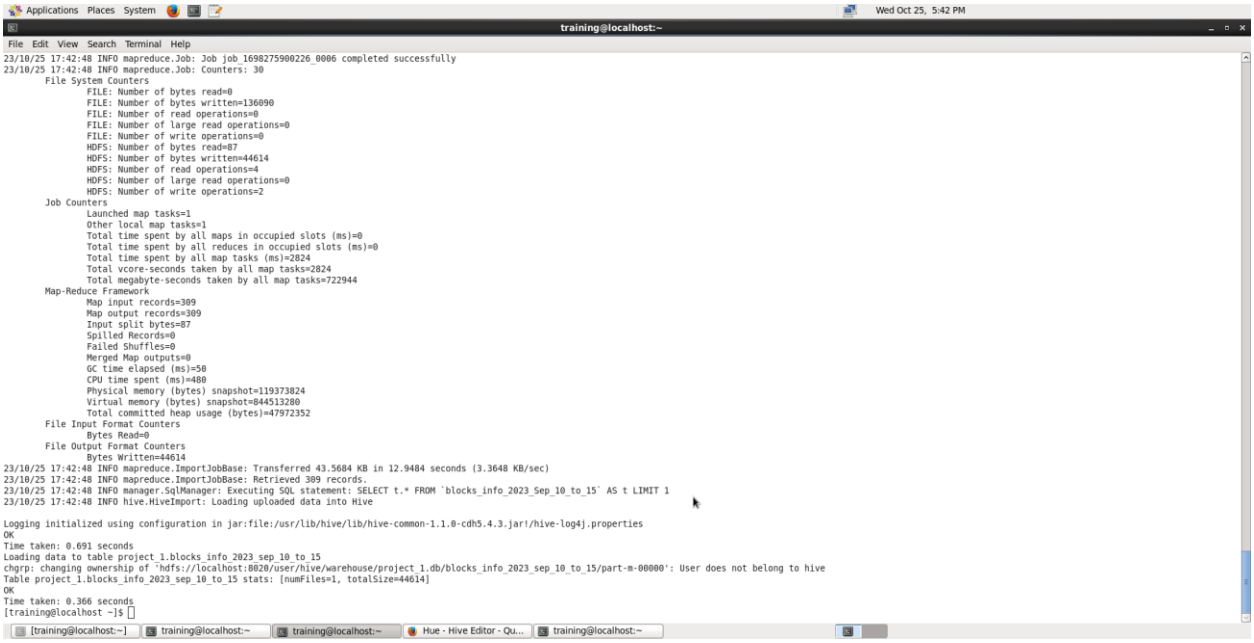
Result:



- **Sqoop command:**

sqoop import --connect jdbc:mysql://localhost:3306/loudacre --username training --password training --table blocks_info_2023_Sep_10_to_15 --hive-import --hive-table project_1.blocks_info_2023_Sep_10_to_15 --m 1

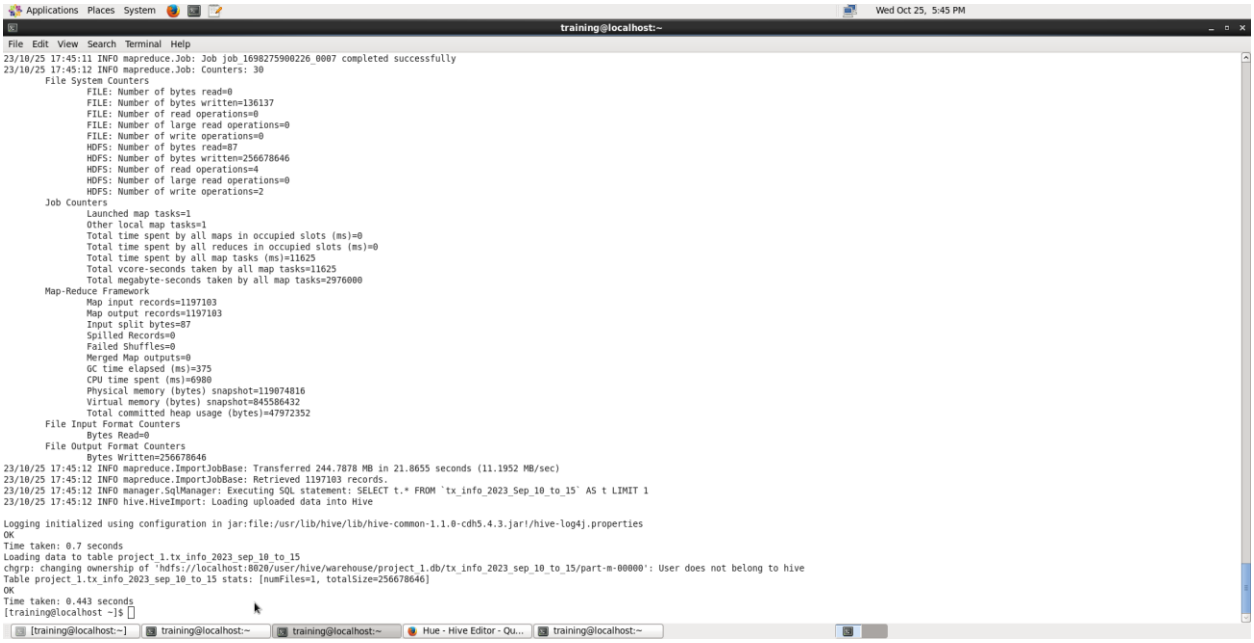
Result:



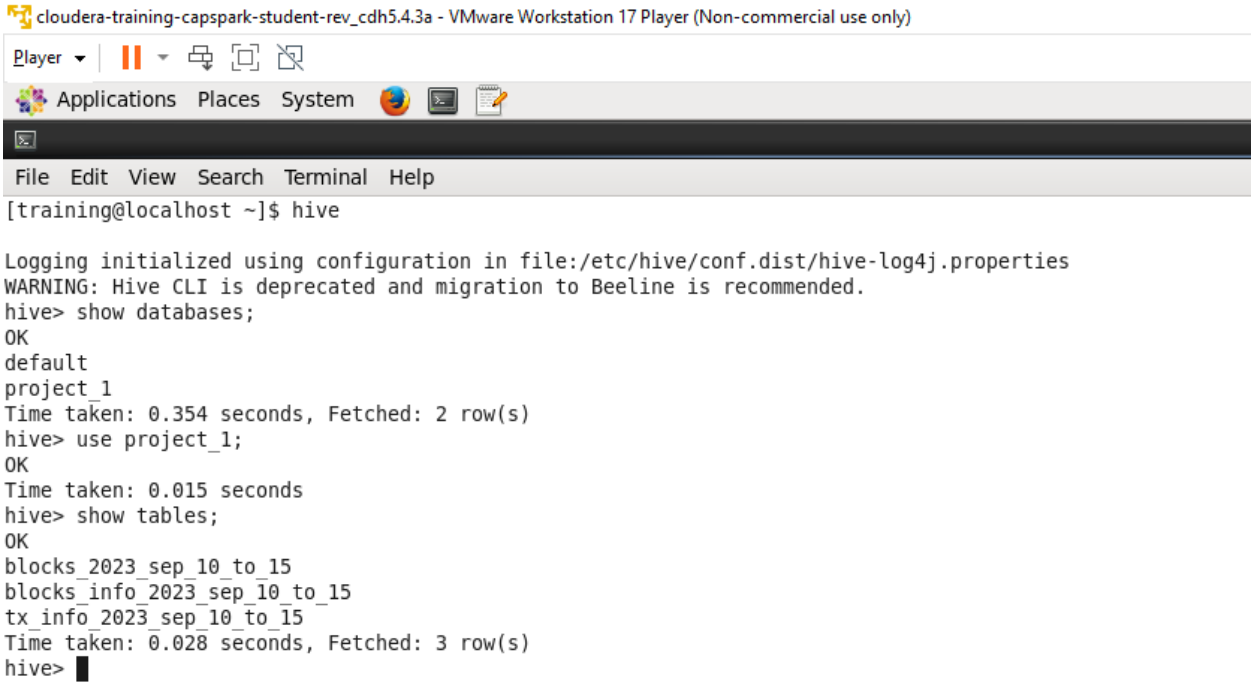
- **Sqoop command:**

sqoop import --connect jdbc:mysql://localhost:3306/loudacre --username training --password training --table table tx_info_2023_Sep_10_to_15 --hive-import --hive-table project_1.table tx_info_2023_Sep_10_to_15 --m 1

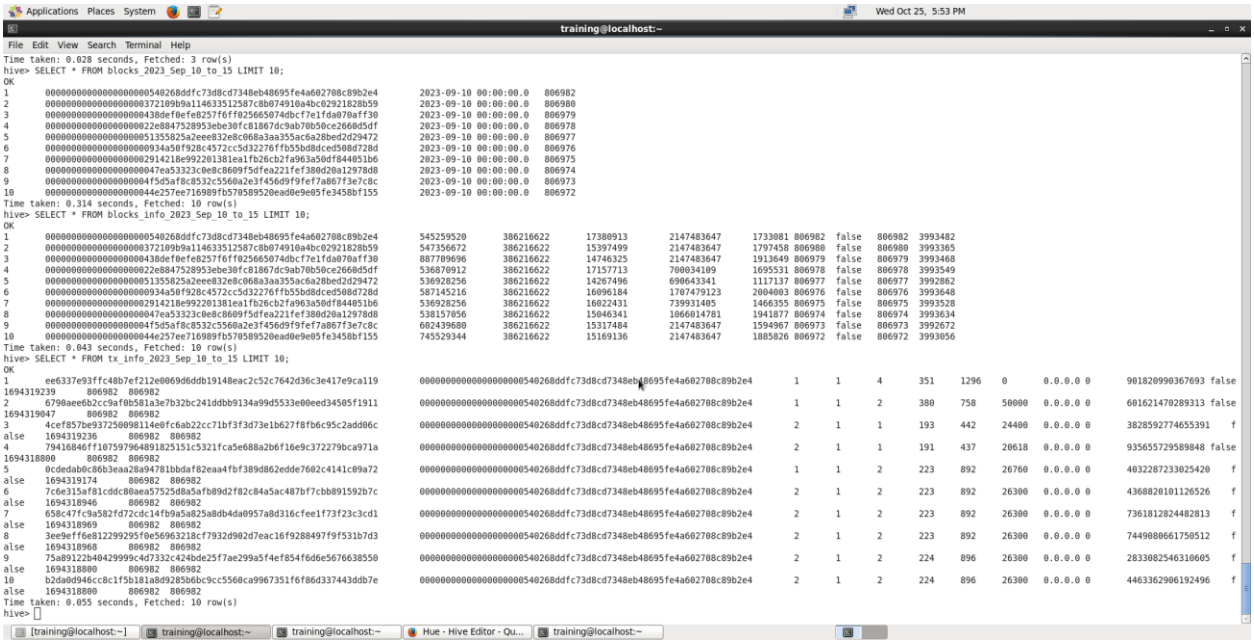
Result:



STEP 5: Verifying Tables on Hive:



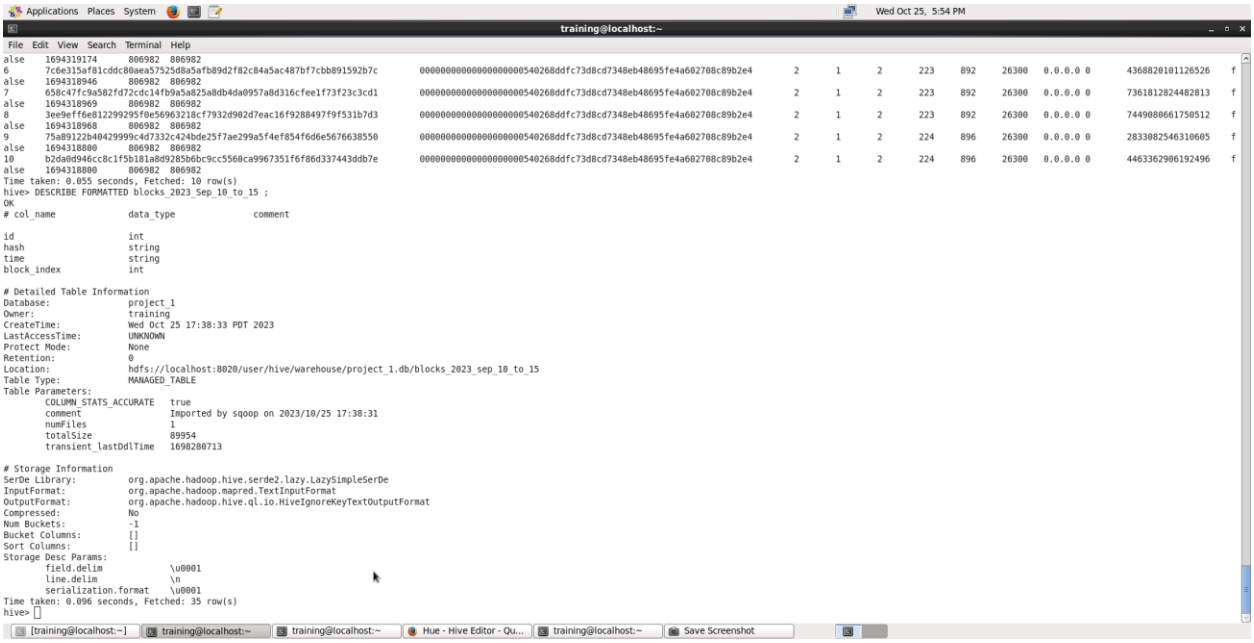
- ***SELECT * FROM blocks_2023_Sep_10_to_15 LIMIT 10;***
- ***SELECT * FROM blocks_info_2023_Sep_10_to_15 LIMIT 10;***
- ***SELECT * FROM tx_info_2023_Sep_10_to_15 LIMIT 10;***



STEP 6: DATA Store in HDFS

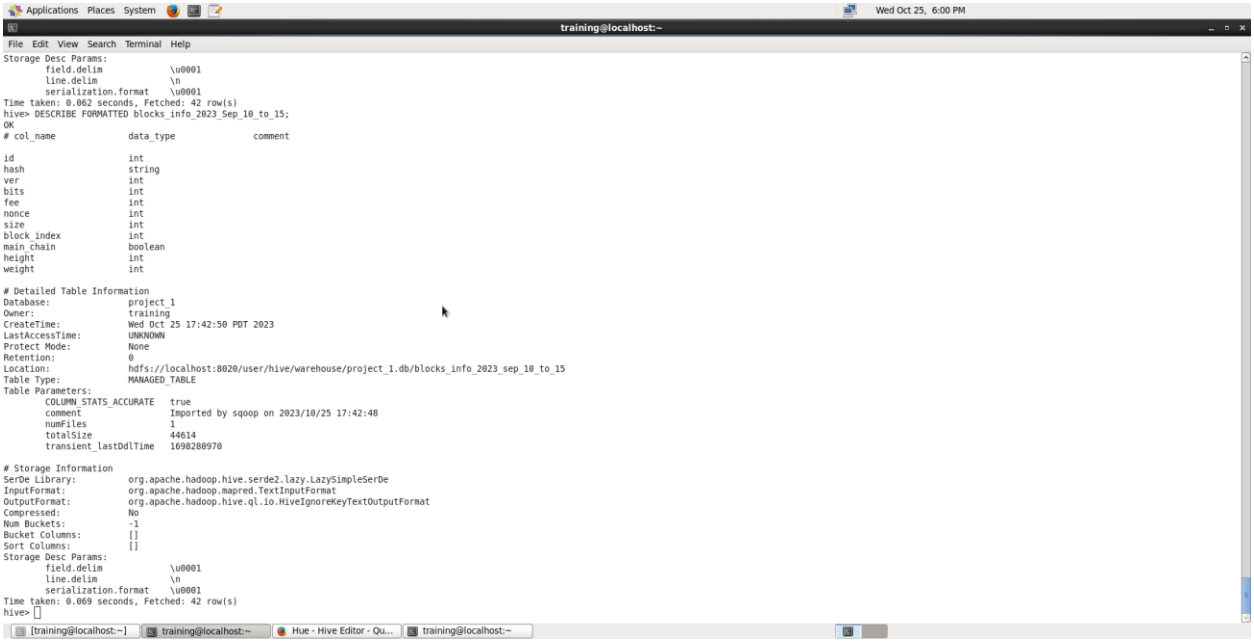
Location for blocks_2023_sep_10_to_15:

hdfs://localhost:8020/user/hive/warehouse/project_1.db/blocks_2023_sep_10_to_15



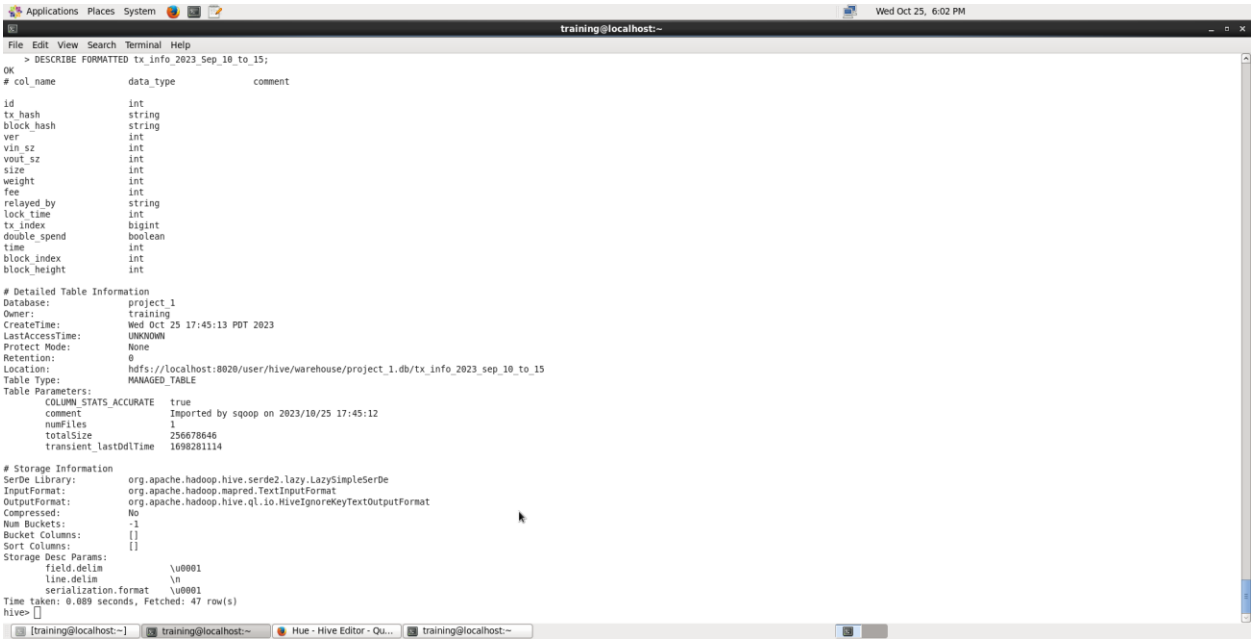
Location for blocks_info_2023_sep_10_to_15:

hdfs://localhost:8020/user/hive/warehouse/project_1.db/blocks_info_2023_sep_10_to_15

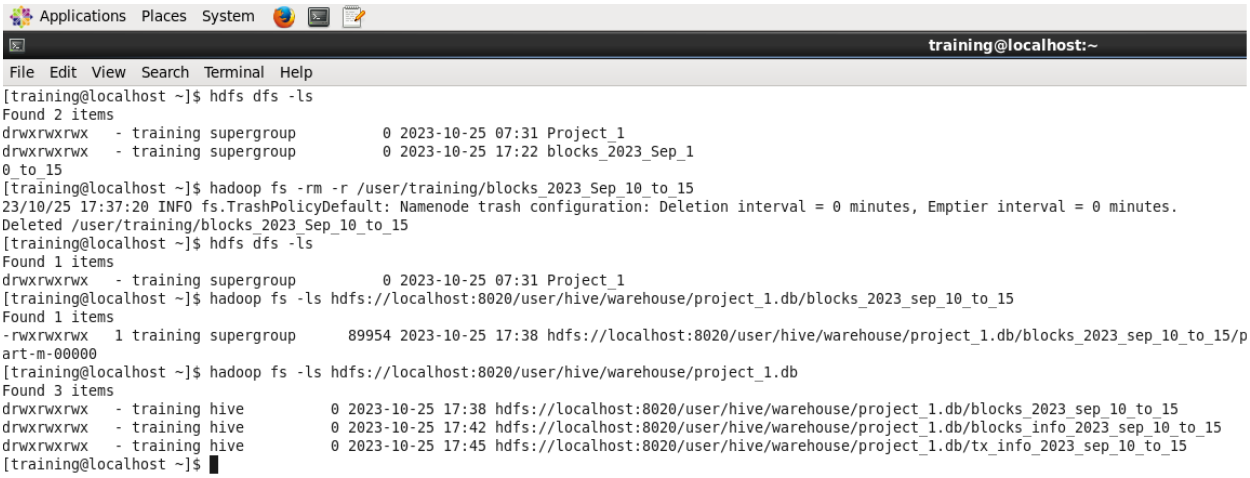


Location of tx_info_2023_sep_10_to_15:

hdfs://localhost:8020/user/hive/warehouse/project_1.db/tx_info_2023_sep_10_to_15



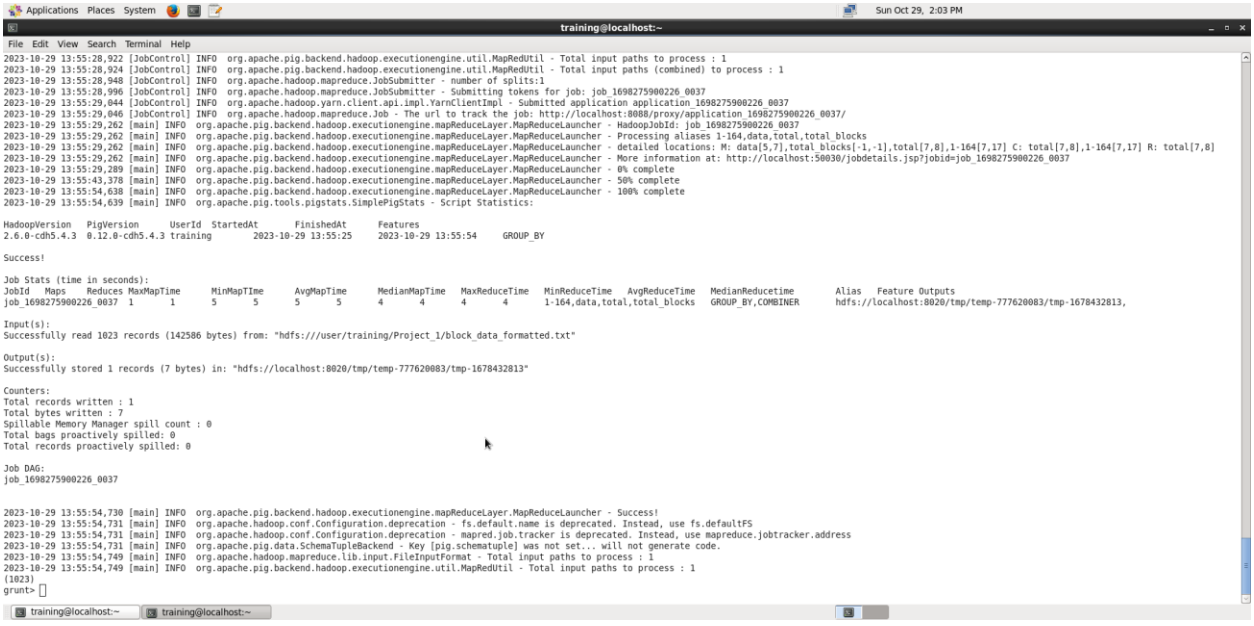
- **Screenshot of location of data stored on HDFS**



Step 2: Data Analysis

A. Data Analysis using Pig.

1. How many total blocks are there in your dataset?
   Answer: There are a total of 1023 blocks in the dataset



PIG Latin code:

-- Load the dataset from HDFS

data = LOAD 'hdfs:///user/training/Project_1/block_data_formatted.txt' USING PigStorage() AS (block: chararray);

-- Count the number of blocks

total_blocks = FOREACH data GENERATE 1 AS count;

total = FOREACH (GROUP total_blocks ALL) GENERATE SUM(total_blocks) AS total_count;

-- Store the total number of blocks in HDFS

STORE total INTO 'hdfs:///user/training/Project_1/total_blocks' USING PigStorage();

-- Display the total number of blocks

DUMP total;

2. What is the largest block height among the blocks in your dataset?

```
HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.4.3  0.12.0-cdh5.4.3 training        2023-10-29 19:30:09     2023-10-29 19:30:44     GROUP_BY

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTIme      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgReduceTime   MedianReducetime        Alias   Feature Outputs
job_1698275900226_0039  1       1       6       6       6       5       5       5       5       1-277,block_heights,data,max_height      GROUP_BY,COMBINER       hdfs://localhost:8020/tmp/temp-777620083/tmp2083940615,

Input(s):
Successfully read 1023 records (142586 bytes) from: "hdfs:///user/training/Project_1/block_data_formatted.txt"

Output(s):
Successfully stored 1 records (5 bytes) in: "hdfs://localhost:8020/tmp/temp-777620083/tmp2083940615"

Counters:
Total records written : 1
Total bytes written : 5
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1698275900226_0039
```

PIG Latin code:

-- Load the dataset from HDFS

data = LOAD 'hdfs:///user/training/Project_1/block_data_formatted.txt' USING PigStorage() AS (block: map[]);

-- Extract block height for each block and find the maximum height

block_heights = FOREACH data GENERATE (int)block#'height' AS height;

max_height = FOREACH (GROUP block_heights ALL) GENERATE MAX(block_heights) AS max_height;

-- Store the maximum block height in HDFS

STORE max_height INTO 'hdfs:///user/training/Project_1/largest_block_height' USING PigStorage();

-- Display the largest block height

DUMP max_height;

3. What is the date and time for that block?

```
HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.4.3  0.12.0-cdh5.4.3 training        2023-10-29 19:42:40     2023-10-29 19:43:28     GROUP_BY,FILTER

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTIme      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgReduceTime   MedianReducetime        Alias   Feature Outputs
job_1698275900226_0042  1       1       6       6       6       6       4       4       4       4       1-898,data,max_height   MULTI_QUERY,COMBINER
job_1698275900226_0043  1       0       3       3       3       3       n/a     n/a     n/a     n/a     max_block       MAP_ONLY        hdfs://localhost:8020/tmp/temp-777620083/tmp315580032,

Input(s):
Successfully read 1023 records (142586 bytes) from: "hdfs:///user/training/Project_1/block_data_formatted.txt"

Output(s):
Successfully stored 0 records in: "hdfs://localhost:8020/tmp/temp-777620083/tmp315580032"

Counters:
Total records written : 0
Total bytes written : 0
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1698275900226_0042  ->      job_1698275900226_0043,
job_1698275900226_0043
```

-- Load the dataset from HDFS (assuming your data contains height and date/time fields)

data = LOAD 'hdfs:///user/training/Project_1/block_data_formatted.txt' USING PigStorage() AS (height: int, date_time: chararray);

-- Find the largest block height

max_height = FOREACH (GROUP data ALL) GENERATE MAX(data.height) AS max_block_height;

-- Find the corresponding date/time for the largest block height

max_block = FILTER data BY height == max_height.max_block_height;

-- Store the largest block and its date/time in HDFS

STORE max_block INTO 'hdfs:///user/training/Project_1/largest_block_date_time' USING PigStorage();

-- Display the largest block height and its date/time

DUMP max_block;

4. What is the highest number of transactions in your blocks?

```
HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.4.3  0.12.0-cdh5.4.3 training        2023-10-29 19:48:34     2023-10-29 19:49:03     GROUP_BY

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTIme      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgReduceTime   MedianReducetime        Alias   Feature Outputs
job_1698275900226_0045  1       1       4       4       4       4       4       4       4       4       1-1122,data,max_transaction_count,transaction_counts    GROUP_BY,COMBINER       hdfs://localhost:8020/tmp/temp-777620083/tmp-15492033
85,

Input(s):
Successfully read 1023 records (142586 bytes) from: "hdfs:///user/training/Project_1/block_data_formatted.txt"

Output(s):
Successfully stored 1 records (5 bytes) in: "hdfs://localhost:8020/tmp/temp-777620083/tmp-1549203385"

Counters:
Total records written : 1
Total bytes written : 5
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1698275900226_0045
```

-- Load the dataset from HDFS

data = LOAD 'hdfs:///user/training/Project_1/block_data_formatted.txt' USING PigStorage() AS (block: map[]);

-- Extract the number of transactions for each block and find the maximum

transaction_counts = FOREACH data GENERATE (int)block#'n_tx' AS transaction_count;

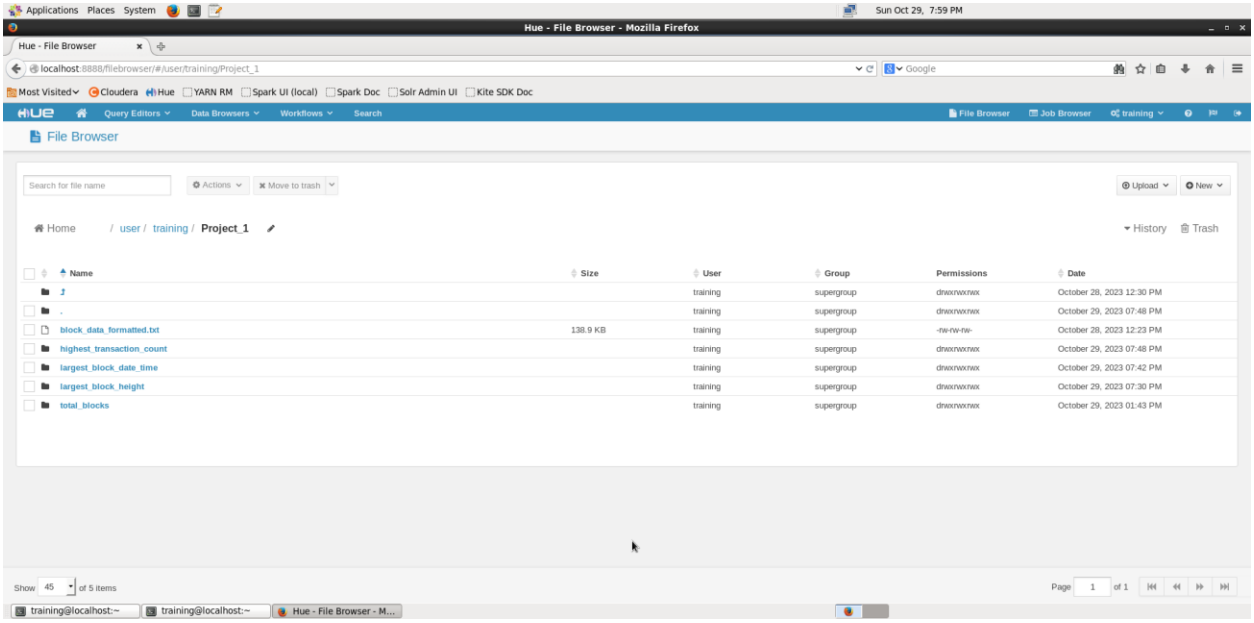
max_transaction_count = FOREACH (GROUP transaction_counts ALL) GENERATE MAX(transaction_counts) AS max_transactions;

-- Store the maximum number of transactions in HDFS

STORE max_transaction_count INTO 'hdfs:///user/training/Project_1/highest_transaction_count' USING PigStorage();

-- Display the highest number of transactions

DUMP max_transaction_count;



B. Data Analysis using Hive – Part 1

1. Create a table in Hive based on the data you stored on HDFS in step 1.B [Deliverable: HiveQL code and result]

hive -e "CREATE DATABASE IF NOT EXISTS blockchain_data; USE blockchain_data; CREATE TABLE IF NOT EXISTS blocks_data (id INT, hash STRING, time TIMESTAMP, block_index INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/data/blockchaindata/blocks_2023_Sep_10_to_15';"

```
training@localhost:~
File  Edit  View  Search  Terminal  Help
[training@localhost ~]$ hive -e "CREATE DATABASE IF NOT EXISTS blockchain_data; USE blockchain_data; CREATE TABLE IF NOT EXISTS blocks_dat
FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/data/blockchaindata/blocks_2023_Sep_10_to_15';"

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
OK
Time taken: 2.822 seconds
OK
Time taken: 0.06 seconds
OK
Time taken: 0.762 seconds
[training@localhost ~]$
```

2. Use HiveQL to query the table

   i. How many records are there in the table?

SELECT COUNT(*) FROM blockchain_data_table;

```
hive> SELECT COUNT(*) FROM blockchain_data_table;
Query ID = training_20231019080606_f96174dd-2126-48a8-877c-2175792c19bd
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1697727319032_0002, Tracking URL = http://localhost:8088/proxy/application_1697727319032_0002/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1697727319032_0002
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2023-10-19 08:06:28,098 Stage-1 map = 0%,  reduce = 0%
```

   ii. How many different days are there in the table?

SELECT COUNT(DISTINCT FROM_UNIXTIME(block_index)) FROM blockchain_data_table;

```
hive> SELECT COUNT(DISTINCT FROM_UNIXTIME(block_index)) FROM blockchain_data_table;
Query ID = training_20231019080707_224d9e85-2651-4a28-9a54-1b794f5bfb7c
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1697727319032_0003, Tracking URL = http://localhost:8088/proxy/application_1697727319032_0003/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1697727319032_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2023-10-19 08:07:35,259 Stage-1 map = 0%,  reduce = 0%
2023-10-19 08:07:45,374 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.2 sec
```

   iii. How many records per each day are there in the table?

SELECT DATE(FROM_UNIXTIME(block_index)) AS day, COUNT(*) AS record_count FROM blockchain_data_table GROUP BY DATE(FROM_UNIXTIME(block_index));

```
hive> SELECT DATE(FROM_UNIXTIME(block_index)) AS day, COUNT(*) AS record_count FROM blockchain_data_table GROUP BY DATE(FROM_UNIXTIME(block_index));
Query ID = training_20231019080909_af3d7505-ce0b-4573-b65b-853c059927fd
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1697727319032_0004, Tracking URL = http://localhost:8088/proxy/application_1697727319032_0004/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1697727319032_0004
```

iv. What are the symbols in the table?

SELECT DISTINCT symbol FROM blockchain_data_table;

```
Ended Job = job_1697727319032_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.48 sec   HDFS Read: 8364 HDFS Write: 5 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 480 msec
OK
NULL    1
Time taken: 32.565 seconds, Fetched: 1 row(s)
```

v. What is the highest price for each symbol?

SELECT symbol, MAX(price) FROM blockchain_data_table GROUP BY symbol;

```
Ended Job = job_1697727319032_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.48 sec   HDFS Read: 8364 HDFS Write: 5 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 480 msec
OK
NULL    1
Time taken: 32.565 seconds, Fetched: 1 row(s)
```

vi. What is the lowest price for each symbol?

SELECT symbol, MIN(price) FROM blockchain_data_table GROUP BY symbol;

```
Ended Job = job_1697727319032_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.48 sec   HDFS Read: 8364 HDFS Write: 5 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 480 msec
OK
NULL    1
Time taken: 32.565 seconds, Fetched: 1 row(s)
```

vii. What is the average price for each symbol?

SELECT symbol, AVG(price) FROM blockchain_data_table GROUP BY symbol;

```
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1697727319032_0005, Tracking URL = http://localhost:8088/proxy/application_1697727319032_0005/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1697727319032_0005
```

viii. What is the range of price for each symbol?

SELECT symbol, MIN(price), MAX(price) FROM blockchain_data_table GROUP BY symbol;

ix. What is the date on which each symbol experienced the highest price?

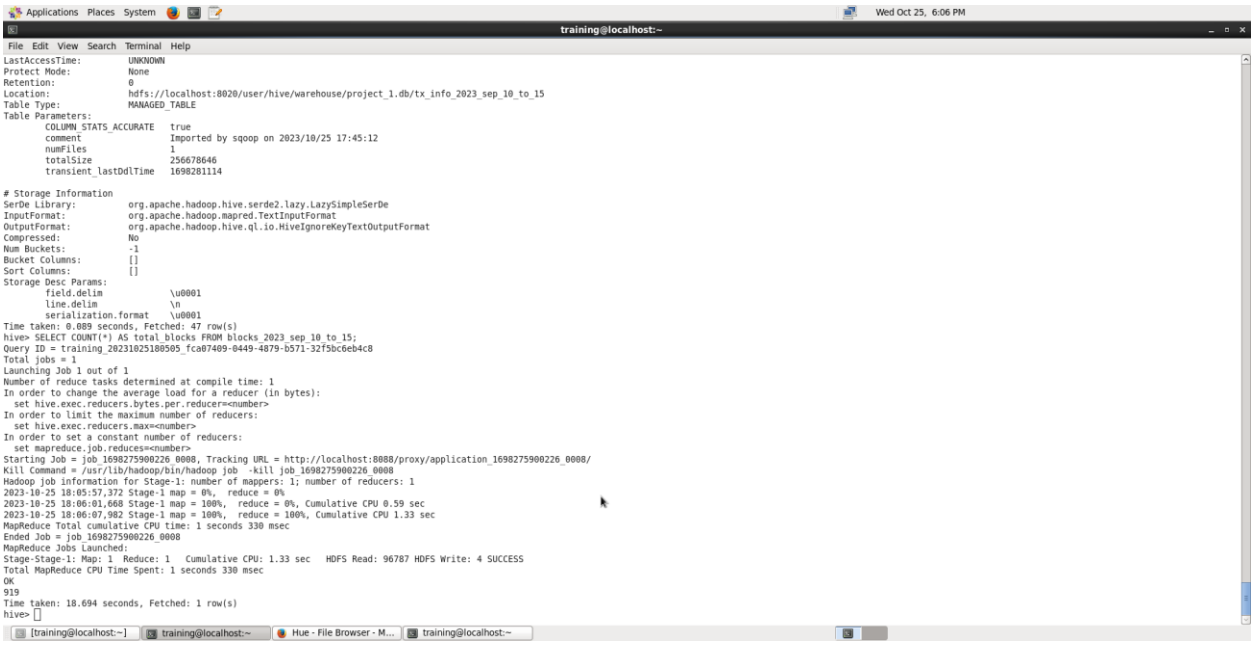
```
SELECT symbol, DATE(max_price_timestamp), MAX(price) FROM (
SELECT symbol, price, block_index, max(price) OVER (PARTITION BY symbol) AS max_price
FROM blockchain_data_table
) x
WHERE price = max_price;
```

```
Query ID = training_20231019081414_9c1e28da-e13e-4cd7-88d4-b38e0b387194
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1697727319032_0006, Tracking URL = http://localhost:8088/proxy/application_1697727319032_0006/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1697727319032_0006
```

C. Data Analysis using Hive – Part 2

1. How many total blocks are there in your blocks table?

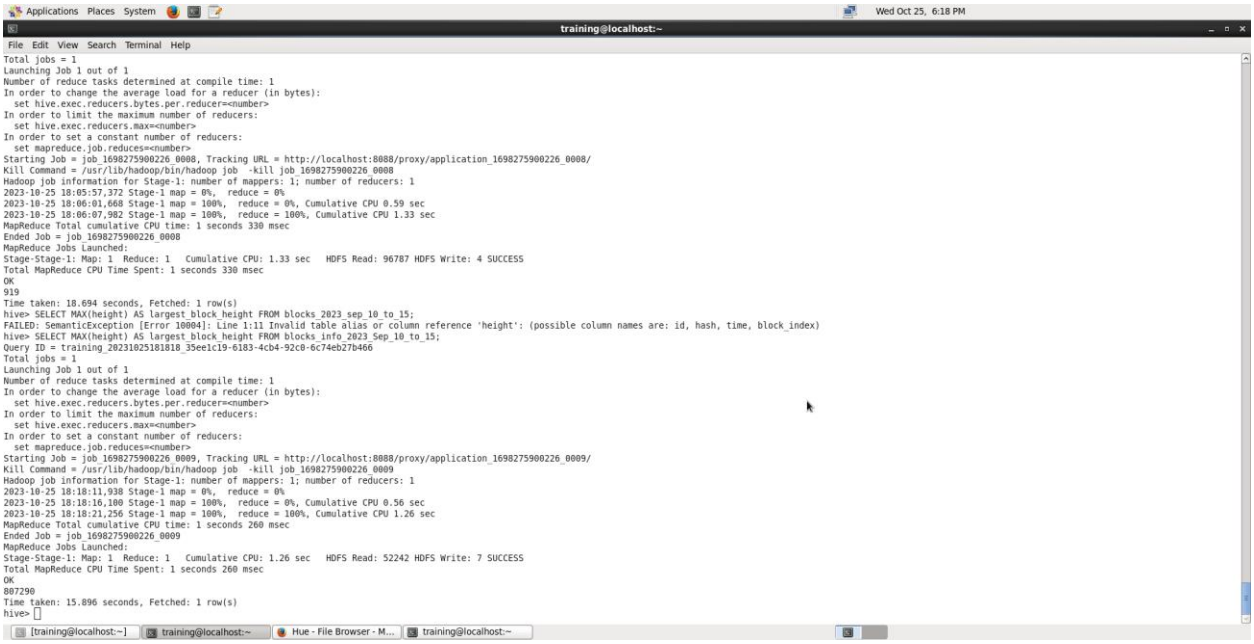
ANS: There is a total of 919 blocks.



HiveQL code: *SELECT COUNT(*) AS total_blocks FROM blocks_2023_sep_10_to_15;*

2. What is the largest block height among the blocks in your blocks table?

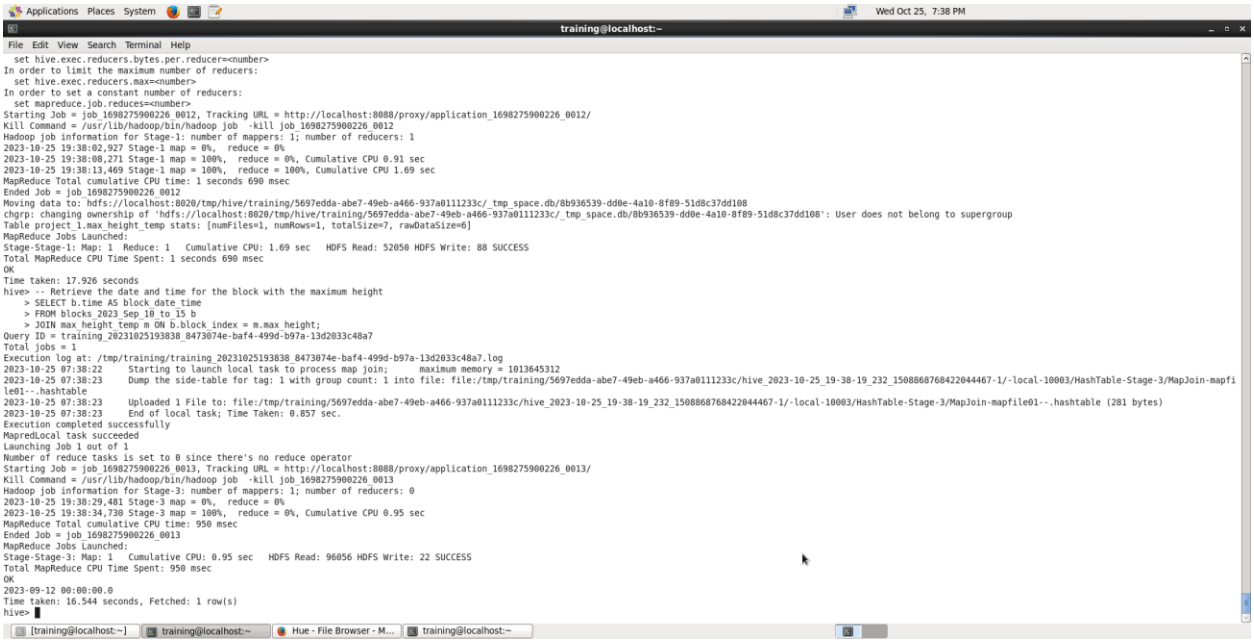
ANS: The largest block height is 807,290.



HiveQL code: *SELECT MAX(block_height) AS largest_block_height FROM tx_info_2023_Sep_10_to_15;*

3. What is the date and time for that block?

ANS. Date and time of the block is 2023-09-12 00:00:00.0.



HiveQL code:

-- Create a temporary table to store the maximum height

CREATE TEMPORARY TABLE max_height_temp AS

SELECT MAX(height) AS max_height

FROM blocks_info_2023_Sep_10_to_15;

-- Retrieve the date and time for the block with the maximum height

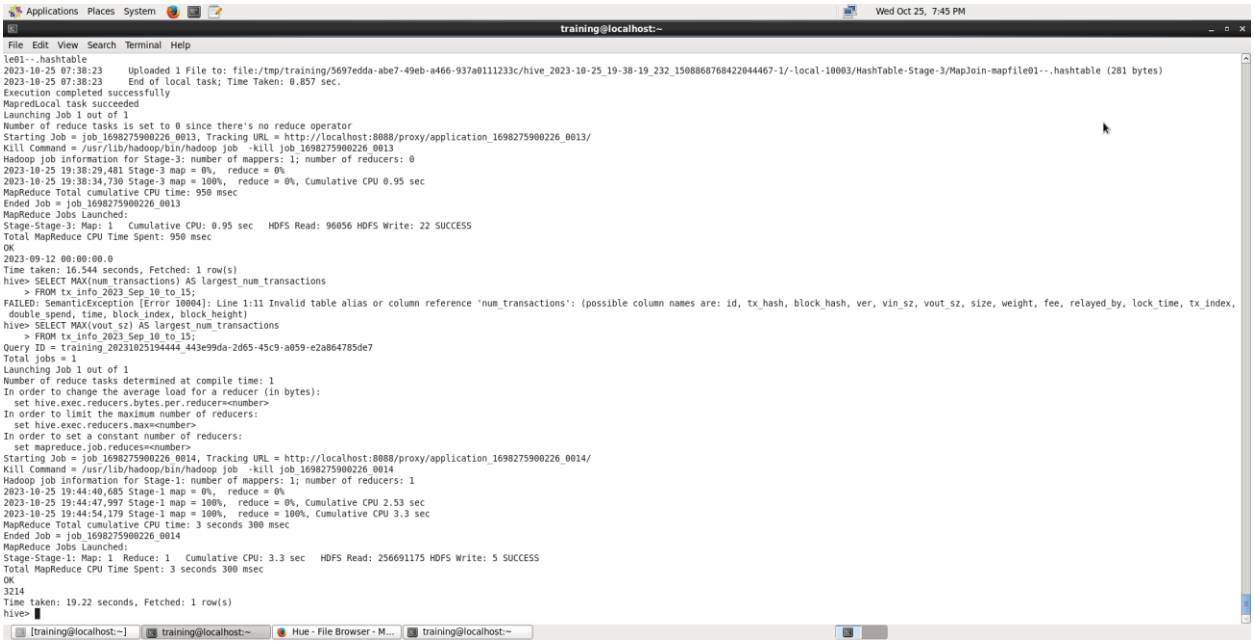
SELECT b.time AS block_date_time

FROM blocks_2023_Sep_10_to_15 b

JOIN max_height_temp m ON b.block_index = m.max_height;

4. What is the largest number of transactions in your blocks?

ANS: The largest number of transactions is 3,214.



HiveQL code: *SELECT MAX(vout_sz) AS largest_num_transactions FROM tx_info_2023_Sep_10_to_15;*