Name: **Talha Asif**

ID: **24280053**

Assignment 2: Building a Batch Analytics Pipeline on HDFS & Hive

1) **Raw Tables in Hive:**

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS raw_user_logs (
    > user_id INT,
    > content_id INT,
    > action STRING,
    > event_time STRING,
    > device STRING,
    > region STRING,
    > session_id STRING )
    > PARTITIONED BY (year INT, month INT, day INT)
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE
    > LOCATION '/raw/logs/'
    > TBLPROPERTIES ("skip.header.line.count"="1");
OK
Time taken: 0.711 seconds
```

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS raw_content_metadata (
    > content_id INT,
    > title STRING,
    > category STRING,
    > length INT,
    > artist STRING )
    > PARTITIONED BY (year INT, month INT, day INT)
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE
    > LOCATION '/raw/metadata/'
    > TBLPROPERTIES ("skip.header.line.count"="1");
OK
Time taken: 0.105 seconds
```

2) Data Modeling: Star Schema

Dimension Table

```
hive> CREATE TABLE IF NOT EXISTS dim_content (
    > content_id INT,
    > title STRING,
    > category STRING,
    > length INT,
    > artist STRING,
    > event_date TIMESTAMP,
    > is_current BOOLEAN )
    > STORED AS PARQUET;
OK
Time taken: 0.632 seconds
```

Fact Table

```
hive> CREATE TABLE IF NOT EXISTS fact_user_actions (
    > action_id BIGINT,
    > user_id INT,
    > content_id INT,
    > action STRING,
    > event_timestamp TIMESTAMP,
    > device STRING,
    > region STRING,
    > session_id STRING )
    > PARTITIONED BY (year INT, month INT, day INT)
    > STORED AS PARQUET;
OK
Time taken: 0.112 seconds
```

3) Data Transformation

Use Hive SQL (INSERT OVERWRITE) to move data from the raw_content_metadata to the dimension table dim_content

```
hive> INSERT OVERWRITE TABLE dim_content
    > SELECT
    > m.content_id,
    > m.title,
    > m.category,
    > m.length,
    > m.artist,
    > CAST(CONCAT_WS('-', CAST(m.year AS STRING), LPAD(CAST(m.month AS STRING), 2, '0'), LPAD(CAST(m.day AS STRING), 2, '
    > TRUE AS is_current
    > FROM raw_content_metadata m;
Query ID = hadoop_20250311014440_211f6897-88eb-4954-ae92-31a1789151a4
Total jobs = 3
Launching Job 1 out of 3
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>
Job running in-process (local Hadoop)
2025-03-11 01:44:47,496 Stage-1 map = 0%,  reduce = 0%
2025-03-11 01:44:50,561 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local1757465627_0001
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to directory hdfs://localhost:9000/user/hive/warehouse/mediaco.db/dim_content/.hive-staging_hive_2025-03-11_0
Loading data to table mediaco.dim_content
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 1086 HDFS Write: 3720 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 10.8 seconds
```

Use Hive SQL (INSERT OVERWRITE) to move data from the raw_user_logs to the Fact table fact_user_actions

```
hive>  INSERT OVERWRITE TABLE fact_user_actions PARTITION (year, month, day)
    > SELECT
    > CASE
    > WHEN l.action = 'play' THEN 1
    > WHEN l.action = 'skip' THEN 2
    > WHEN l.action = 'pause' THEN 3
    > WHEN l.action = 'forward' THEN 4
    > ELSE NULL  END AS action_id,
    > l.user_id,l.content_id, l.action,CAST(l.event_time AS TIMESTAMP) AS event_timestamp,l.device,l.region,l.session
    > YEAR(CAST(l.event_time AS TIMESTAMP)) AS year,
    > MONTH(CAST(l.event_time AS TIMESTAMP)) AS month,
    > DAY(CAST(l.event_time AS TIMESTAMP)) AS day
    > FROM raw_user_logs l;
Query ID = hadoop_20250311030006_bca638ce-f7bc-4d5f-b57d-ee027d94a573
Total jobs = 3
Launching Job 1 out of 3
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>
Job running in-process (local Hadoop)
2025-03-11 03:00:10,994 Stage-1 map = 0%,  reduce = 0%
2025-03-11 03:00:14,332 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local1955101712_0001
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to directory hdfs://localhost:9000/user/hive/warehouse/mediaco.db/fact_user_actions/.hive-staging_hive_20
000
Loading data to table mediaco.fact_user_actions partition (year=null, month=null, day=null)


        Time taken to load dynamic partitions: 1.045 seconds
        Time taken for adding to write entity : 0.003 seconds
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 2912 HDFS Write: 4558 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 9.934 seconds
```

4) Analytical Queries

Monthly Active Users by Region:

This query counts distinct users by region for each month

```
hive> SELECT year,month,region,
    >  COUNT(DISTINCT user_id) AS monthly_active_users
    > FROM fact_user_actions
    > GROUP BY year, month, region
    > ORDER BY year, month, region;
Query ID = hadoop_20250311031329_2e15ae17-f4ff-49d6-bf4b-b3bc1049a9a9
Total jobs = 2
Launching Job 1 out of 2
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>
Job running in-process (local Hadoop)
2025-03-11 03:13:31,986 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local60589707_0002
Launching Job 2 out of 2
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>
Job running in-process (local Hadoop)
2025-03-11 03:13:34,035 Stage-2 map = 100%,  reduce = 100%
Ended Job = job_local1156819567_0003
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 18636 HDFS Write: 4558 SUCCESS
Stage-Stage-2:  HDFS Read: 18636 HDFS Write: 4558 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
2023    9       APAC    11
2023    9       EU      6
2023    9       US      8
Time taken: 4.451 seconds, Fetched: 3 row(s)
```

Top Categories by Play Count

This query joins fact and dimension tables to find the most popular content categories

```
hive> SELECT d.category, COUNT(*) AS play_count
    > FROM fact_user_actions f
    > JOIN dim_content d ON f.content_id = d.content_id
    > WHERE f.action = 'play' AND f.year = 2023
    > GROUP BY d.category
    > ORDER BY play_count DESC
    > LIMIT 5;
Query ID = hadoop_20250311033024_11af7cc7-cec5-45ab-9047-5149d861310c
Total jobs = 2
SLF4J: Found binding in [jar:file:/home/hadoop/hive/lib/log4j-slf4j-impl-2.10.0.
2025-03-11 03:34:15     Starting to launch local task to process map join;

MapredLocal task succeeded
Launching Job 1 out of 2
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>
Job running in-process (local Hadoop)
Ended Job = job_local1878612104_0004
Launching Job 2 out of 2
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>
Job running in-process (local Hadoop)
2025-03-11 03:36:12,922 Stage-3 map = 100%,  reduce = 100%
Ended Job = job_local302296201_0005
MapReduce Jobs Launched:
Stage-Stage-2:  HDFS Read: 22686 HDFS Write: 4558 SUCCESS
Stage-Stage-3:  HDFS Read: 22686 HDFS Write: 4558 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Jazz    2
Hip-Hop 2
Rock    1
Classical       1
Time taken: 348.25 seconds, Fetched: 4 row(s)
```

Average Session Length in seconds by Device Type

We used CTE to find the session duration. First calculate MAX (event_timestamp) - MIN (event_timestamp) for each device and then compute the average session length

```
hive> WITH session_durations AS (
    > SELECT device, session_id,
    > MAX(UNIX_TIMESTAMP(event_timestamp)) - MIN(UNIX_TIMESTAMP(event_timestamp)) AS session_length
    > FROM fact_user_actions
    > GROUP BY device, session_id )
    > SELECT device,COUNT(DISTINCT session_id) AS total_sessions,
    > AVG(session_length) AS avg_session_length
    > FROM session_durations
    > GROUP BY device
    > ORDER BY avg_session_length DESC;
Query ID = hadoop_20250311041247_af3b621c-92d5-4fb6-ad6f-39024c7f6498
Total jobs = 2
Launching Job 1 out of 2
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>
Job running in-process (local Hadoop)
2025-03-11 04:12:49,538 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local187432306_0006
Launching Job 2 out of 2
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>
Job running in-process (local Hadoop)
2025-03-11 04:12:51,565 Stage-2 map = 100%,  reduce = 100%
Ended Job = job_local292023071_0007
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 27698 HDFS Write: 4558 SUCCESS
Stage-Stage-2:  HDFS Read: 27698 HDFS Write: 4558 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
desktop 5       11098.0
mobile  11      530.6363636363636
tablet  7       0.0
Time taken: 4.342 seconds, Fetched: 3 row(s)
```

5) Write-up

Design considerations:

- Used external tables for raw data to maintain the original files

- Implemented partitioning by year, month, and day for efficient querying
- Used Parquet for star schema tables for better compression and columnar storage

Performance Optimization:

Running queries in hive, as visible from the time taken in above queries, is slow even for small datasets because Hive is designed for batch processing on large-scale distributed data rather than low-latency queries. We can improve our performance by:

- Even for small data, Hive translates queries into MapReduce. Tez is much faster than MapReduce for Hive queries
- If the table is partitioned, always filter by partition columns to reduce data scanning
- Merge small files into bigger ones to reduce HDFS overhead
- Enable Vectorized Query Execution which processes multiple rows in batches