NM1112 - Problem Statements

1. PROBLEM STATEMENT - 1

QUESTION :

The pandemic Covid has badly impacted everyone's life across the globe in the year 2020. Assessing the available data related to patients , treatments , post Covid prognosis , recovery rate and many other such information will help hospitals and health organizations to evaluate what care approaches are most effective. This can also help in understanding what is the effect of medication on patients with history of other illness such as cardiac problems, diabetics, cancer etc.

All data related to Covid pandemic have continuously been monitored and analyzed to find the intensity of its spread. A sample of such data that has been captured from different locations on daily basis. You are required to get some useful insights by processing this data using the Big Data platform Hadoop and its ecosystem components.
Listed below are few reports expected from analysis :

• What is the number of people who are infected globally ?
• How many cases are reported in a continent ?
• Which country has recorded maximum number of deaths due to Covid?
• How many people are vaccinated so far ?

AIM :
To utilize Hadoop ecosystem tools for efficient storage, processing, and analytical querying of large-scale Covid-19 datasets. The primary objectives include:

- Quantifying the total number of Covid-19 infections worldwide.
- Analyzing the distribution of cases across different continents.
- Identifying the most severely affected country in terms of total deaths.
- Calculating the aggregate number of vaccinations globally.
- Evaluating which Hadoop component (e.g., HDFS, Hive, Spark) provides the most suitable environment
  for big data analysis in healthcare use cases.

This analysis will assist health organizations and policymakers in decision-making and tracking pandemic response effectiveness with scalable data-driven approaches.

A. HADOOP INITIALIZATION

- Ensure Hadoop (HDFS and YARN) is properly installed and running.
- Load Covid data (CSV format) into HDFS:

Command:
hdfs dfs -put covid_data.csv /user/hadoop/covid/

- Use Hive for analytics:
Start Hive shell using:
hive

- Create Hive table and load data:
-- Create table
CREATE TABLE covid_data (
    date STRING,
    country STRING,
    continent STRING,
    total_cases INT,
    total_deaths INT,
    total_vaccinations INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

-- Load CSV data into table
LOAD DATA INPATH '/user/hadoop/covid/covid_data.csv' INTO TABLE covid_data;

B. PROGRAM

```
-- 1. Find the number of people infected globally
-- Procedure: Sum up the total_cases column globally.
SELECT SUM(total_cases) AS total_infected_globally
FROM covid_data;


-- 2. Count number of cases reported by continent
-- Procedure: Group by continent and calculate sum of cases.
SELECT continent, SUM(total_cases) AS cases_by_continent
FROM covid_data
GROUP BY continent;


-- 3. Determine the country with the maximum number of deaths
-- Procedure: Order countries by deaths in descending order and get the top.
SELECT country, total_deaths
FROM covid_data
ORDER BY total_deaths DESC
LIMIT 1;
```

```
-- 4. Identify how many people have been vaccinated so far
-- Procedure: Sum up the total_vaccinations column.
SELECT SUM(total_vaccinations) AS global_vaccinations
FROM covid_data;


-- 5. Best suited Hadoop component and why
COMPONENT: Hive
REASON: Hive supports SQL-like querying (HiveQL) on large datasets stored in HDFS.
It is ideal for analytical tasks like aggregations, filters, groupings over large Covid datasets.
It provides scalability, distributed processing, and is easier for users familiar with SQL.
```

C. SAMPLE OUTPUT

Query	**Answer**
Total infected globally	250000000
Cases by continent	Asia: 120M, Europe: 70M, Americas: 50M, ...
Country with most deaths	USA 500000
Total vaccinations so far	4500000000

---

2. TASK 1 - PROBLEM STATEMENT

QUESTION :

Task 1: Ingesting data from MySQL to HDFS using Sqoop and
working on HDFS commands.
Data preparation steps :
[LOAD DATA LOCAL INFILE '/home/user/CovidGlobalData.csv'
INTO TABLE CovidData FIELDS TERMINATED BY ',' ENCLOSED BY
'"'

LINES TERMINATED BY '\n';]
1. Create a new HDFS directory called CovidHDFS.
2. Move the data from the MySQL table CovidData into HDFS
directory named CovidHDFS using Sqoop import with a single mapper
argument.
3. Copy the files from CovidHDFS directory to CovidData_Backup
4. Display first 1 KB of data using appropriate HDFS command (Hint :
head)
5. Display last 1 KB of data using appropriate HDFS command (Hint :
tail)
6. Transfer the data from the table 'CovidData' in MySQL to HDFS
directory SqoopCovidAsiaData using Sqoop. Output file should contain

only Asia related Covid details.
7. Transfer the data from the table 'CovidData' in MySQL to HDFS
directory SqoopNonVaccinated using Sqoop. Ouput file should contain
only details of people not been vaccinated

AIM :
We are tasked with ingesting Covid-related data from a MySQL database into HDFS using Apache Sqoop.
The data consists of global Covid data stored in a MySQL table named `CovidData`. The objective is to:

- Move data from MySQL to HDFS.
- Create backups.
- Filter specific records using Sqoop.
- Work with standard HDFS commands.

Data has already been inserted into MySQL using:
LOAD DATA LOCAL INFILE '/home/user/CovidGlobalData.csv'
INTO TABLE CovidData FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n';

Tasks include creating directories in HDFS, importing data using Sqoop, copying data, displaying file contents, and filtering based on region and vaccination status.

A. HADOOP / SQOOP INITIALIZATION

- Ensure Hadoop cluster (HDFS, YARN) is up and running.
- Sqoop should be installed and configured to connect to MySQL.
- MySQL server should contain the `CovidData` table.

MySQL credentials (example):
- Host: localhost
- Port: 3306
- Database: CovidDB
- User: root
- Password: root

B. PROGRAM

```
# 1. Create a new HDFS directory called CovidHDFS
hdfs dfs -mkdir /CovidHDFS


# 2. Import CovidData table from MySQL into HDFS directory CovidHDFS using Sqoop (single mapper)
sqoop import \
--connect jdbc:mysql://localhost/CovidDB \
--username root --password root \
```

```
--table CovidData \
--target-dir /CovidHDFS \
--m 1


# 3. Copy the files from CovidHDFS directory to CovidData_Backup
hdfs dfs -cp /CovidHDFS /CovidData_Backup


# 4. Display first 1 KB of data
hdfs dfs -cat /CovidHDFS/part-m-00000 | head -c 1024


# 5. Display last 1 KB of data
hdfs dfs -cat /CovidHDFS/part-m-00000 | tail -c 1024


# 6. Import only Asia-related Covid records into SqoopCovidAsiaData HDFS directory
sqoop import \
--connect jdbc:mysql://localhost/CovidDB \
--username root --password root \
--table CovidData \
--target-dir /SqoopCovidAsiaData \
--m 1 \
--where "continent = 'Asia'"


# 7. Import only non-vaccinated people into SqoopNonVaccinated HDFS directory
sqoop import \
--connect jdbc:mysql://localhost/CovidDB \
--username root --password root \
--table CovidData \
--target-dir /SqoopNonVaccinated \
--m 1 \
--where "total_vaccinations = 0"
```

C. SAMPLE OUTPUT

```
# After step 1:
$ hdfs dfs -ls /
drwxr-xr-x   - hadoop hadoop          0 2025-11-01 11:00 /CovidHDFS

# After step 2:
$ hdfs dfs -ls /CovidHDFS
-rw-r--r--   1 hadoop hadoop     123456 2025-11-01 11:02 /CovidHDFS/part-m-00000

# After step 3:
$ hdfs dfs -ls /CovidData_Backup
```

-rw-r--r-- 1 hadoop hadoop 123456 2025-11-01 11:03 /CovidData_Backup/part-m-00000

After step 4 (first 1KB):
(First 1024 bytes of part-m-00000 printed...)

After step 5 (last 1KB):
(Last 1024 bytes of part-m-00000 printed...)

# After step 6:
$ hdfs dfs -ls /SqoopCovidAsiaData
-rw-r--r-- 1 hadoop hadoop 54321 2025-11-01 11:05 /SqoopCovidAsiaData/part-m-00000

# After step 7:
$ hdfs dfs -ls /SqoopNonVaccinated
-rw-r--r-- 1 hadoop hadoop 67890 2025-11-01 11:07 /SqoopNonVaccinated/part-m-00000

---

2. TASK 2 - PROBLEM STATEMENT

QUESTION :

Task 2: Data Analysis using Hive
1. Create a Hive internal table(CovidDatawarehouse) for the dataset (CovidGlobalData.csv), provide schema accordingly and load the data into the table. [date_current may be created as String and later converted to Date]
2. Count the number of vaccinations available in each location using a Hive Query
3. List the number of vaccinations available on location names starting with 'United.*'
4. Partition the CovidDatawarehouse table based on the values of the column 'continent'
5. Count the number of continents and distribute the CovidDatawarehouse table data randomly into 4 number of buckets as per the number of continents
6. Find the maximum, minimum and average number of people infected with Covid in each bucket
7. Create a Hive query to find the number of deaths in each continent
8. Create a Hive query to find the average diabetes prevalence for the country ‘Israel’

AIM :

We are required to analyze global Covid-19 data using Apache Hive. The data is stored in a CSV file named `CovidGlobalData.csv` and contains fields like date, location, continent, total_cases, total_deaths, total_vaccinations, diabetes_prevalence, etc.

Tasks include:
- Creating an internal Hive table and loading data into it.
- Performing queries for vaccination statistics.
- Partitioning and bucketing the table.
- Performing analytical queries like min/max/avg infections and deaths by continent or bucket.

A. HIVE INITIALIZATION

- Start Hive shell:
  hive

- Ensure the CovidGlobalData.csv file is available in HDFS or local file system.
- For simplicity, we'll load the file from local file system using `LOAD DATA LOCAL INPATH`.

B. PROGRAM

```
-- 1. Create Hive internal table and load data
CREATE TABLE CovidDatawarehouse (
    date_current STRING,
    location STRING,
    continent STRING,
    total_cases INT,
    total_deaths INT,
    total_vaccinations INT,
    diabetes_prevalence FLOAT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

-- Load data
LOAD DATA LOCAL INPATH '/home/user/CovidGlobalData.csv'
INTO TABLE CovidDatawarehouse;

-- Convert date_current to date format (optional step)
ALTER TABLE CovidDatawarehouse ADD COLUMNS (date_formatted DATE);
UPDATE CovidDatawarehouse SET date_formatted = TO_DATE(date_current);


-- 2. Count number of vaccinations available in each location
SELECT location, SUM(total_vaccinations) AS total_vaccinations
FROM CovidDatawarehouse
GROUP BY location;


-- 3. List the number of vaccinations for locations starting with 'United'
SELECT location, SUM(total_vaccinations) AS total_vaccinations
```

```
FROM CovidDatawarehouse
WHERE location RLIKE '^United.*'
GROUP BY location;


-- 4. Partition the table based on continent
CREATE TABLE CovidDatawarehouse_partitioned (
    date_current STRING,
    location STRING,
    total_cases INT,
    total_deaths INT,
    total_vaccinations INT,
    diabetes_prevalence FLOAT
)
PARTITIONED BY (continent STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

-- Load data into partitioned table
INSERT OVERWRITE TABLE CovidDatawarehouse_partitioned
PARTITION (continent)
SELECT date_current, location, total_cases, total_deaths, total_vaccinations, diabetes_prevalence,
continent
FROM CovidDatawarehouse;


-- 5. Bucket the table into 4 buckets
-- First, get the number of unique continents
SELECT COUNT(DISTINCT continent) AS number_of_continents
FROM CovidDatawarehouse;

-- Create bucketed table (distributed randomly)
CREATE TABLE CovidDatawarehouse_bucketed (
    date_current STRING,
    location STRING,
    continent STRING,
    total_cases INT,
    total_deaths INT,
    total_vaccinations INT,
    diabetes_prevalence FLOAT
)
CLUSTERED BY (continent) INTO 4 BUCKETS
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

-- Load data into bucketed table
INSERT OVERWRITE TABLE CovidDatawarehouse_bucketed
SELECT * FROM CovidDatawarehouse;
```

```
-- 6. Max, Min, Avg infected in each bucket
SELECT
   bucket_number() AS bucket_id,
   MAX(total_cases) AS max_cases,
   MIN(total_cases) AS min_cases,
   AVG(total_cases) AS avg_cases
FROM CovidDatawarehouse_bucketed
GROUP BY bucket_number();


-- 7. Number of deaths in each continent
SELECT continent, SUM(total_deaths) AS total_deaths
FROM CovidDatawarehouse
GROUP BY continent;


-- 8. Average diabetes prevalence for the country 'Israel'
SELECT AVG(diabetes_prevalence) AS avg_diabetes_prevalence
FROM CovidDatawarehouse
WHERE location = 'Israel';
```

C. SAMPLE OUTPUT

Query 2: Vaccinations per location

Location	**Total Vaccinations**
India	1500000000
USA	1350000000
Brazil	900000000

Query 3: Locations starting with 'United'

Location	**Total Vaccinations**
United States	1350000000
United Kingdom	900000000
United Arab Emirates	700000000

Query 6: Bucket-wise stats

Bucket ID	**Max Cases**	**Min Cases**	**Avg Cases**
0	50000000	10000	1000000
1	70000000	20000	2000000

Query 7: Deaths by continent

Continent	Total Deaths
Asia	1500000
Europe	1800000
Africa	500000

Query 8: Average diabetes prevalence for Israel

Avg Diabetes Prevalence
9.5

---

2. TASK 3 - PROBLEM STATEMENT

QUESTION :

Task 3: Data processing with Spark
Data pre-processing: Create RDDs and Data Frames needed to perform the below operations.

Find the total number of cases in each continent
Find the total number of deaths in each location
Compute the maximum deaths at specific locations like 'Europe' and 'Asia'
Find the total number of people vaccinated at each continent
Find the count of country wise vaccination for the month "January 2021"
What is the average number of total cases across all locations?
Which continent has the highest total number of vaccinations?
Extract the year, month, and day from the date_current column and create separate columns for each

AIM :

Perform data processing and analysis on a Covid-19 dataset using Apache Spark. Use Spark RDDs and DataFrames to answer queries related to total cases, deaths, vaccinations, and date manipulations on the dataset containing fields such as:
- date_current
- location
- continent
- total_cases
- total_deaths
- total_vaccinations

A. SPARK INITIALIZATION

```
# Start Spark Session (Python)
from pyspark.sql import SparkSession

spark = SparkSession.builder \
    .appName("CovidDataAnalysis") \
    .getOrCreate()

# Load data into a DataFrame
df = spark.read.csv("/path/to/CovidGlobalData.csv", header=True, inferSchema=True)

# Create RDD from DataFrame
rdd = df.rdd
```

B. PROGRAM

```
# 1. Create RDD and DataFrame
df = spark.read.csv("/path/to/CovidGlobalData.csv", header=True, inferSchema=True)
rdd = df.rdd


# 2. Total number of cases in each continent
df.groupBy("continent") \
  .sum("total_cases") \
  .withColumnRenamed("sum(total_cases)", "total_cases_each_continent") \
  .show()


# 3. Total number of deaths in each location
df.groupBy("location") \
  .sum("total_deaths") \
  .withColumnRenamed("sum(total_deaths)", "total_deaths_each_location") \
  .show()


# 4. Maximum deaths in Europe and Asia
df.filter(df.continent.isin("Europe", "Asia")) \
  .groupBy("continent") \
  .max("total_deaths") \
  .withColumnRenamed("max(total_deaths)", "max_deaths") \
  .show()


# 5. Total number of vaccinated people per continent
df.groupBy("continent") \
  .sum("total_vaccinations") \
  .withColumnRenamed("sum(total_vaccinations)", "total_vaccinations_each_continent") \
```

```
  .show()


# 6. Count of country-wise vaccinations for January 2021
from pyspark.sql.functions import month, year, col

df_jan_2021 = df.filter((month("date_current") == 1) & (year("date_current") == 2021)) \
    .groupBy("location") \
    .sum("total_vaccinations") \
    .withColumnRenamed("sum(total_vaccinations)", "vaccinations_jan_2021")
df_jan_2021.show()


# 7. Average number of total cases across all locations
df.agg({"total_cases": "avg"}).withColumnRenamed("avg(total_cases)", "avg_total_cases").show()


# 8. Continent with the highest total number of vaccinations
from pyspark.sql.functions import desc

df.groupBy("continent") \
  .sum("total_vaccinations") \
  .withColumnRenamed("sum(total_vaccinations)", "total_vaccinations") \
  .orderBy(desc("total_vaccinations")) \
  .show(1)


# 9. Extract year, month, and day from date_current
from pyspark.sql.functions import split

df = df.withColumn("year", split(df["date_current"], "-")[0]) \
       .withColumn("month", split(df["date_current"], "-")[1]) \
       .withColumn("day", split(df["date_current"], "-")[2])

df.select("date_current", "year", "month", "day").show(5)
```

C. SAMPLE OUTPUT

Query	**Result**		
Total cases per continent	Asia: 120M	Europe: 70M	Americas: 50M
Total deaths per location	USA: 500K	India: 300K	Brazil: 400K
Max deaths in Europe and Asia	Europe: 500K	Asia: 400K	
Vaccinations per continent	Asia: 2B	Europe: 1.5B	Americas: 1.2B
Vaccinations in Jan 2021	USA: 100M	India: 80M	Brazil: 60M
Average total cases	1000000		
Highest vaccinated continent	Asia		
Date extraction example	2021	01	15