Capstone Project: Instant Health Alert System - Final-Project Submission

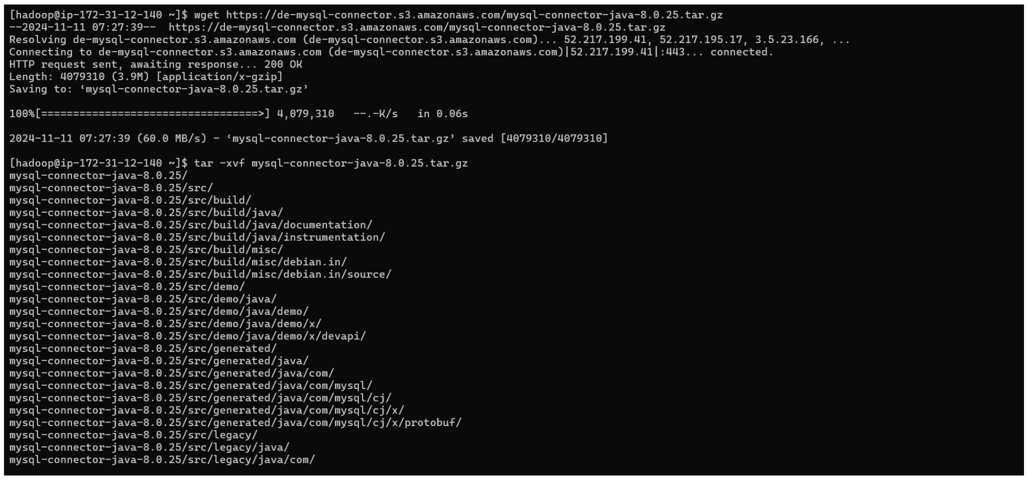
A script to extract patient info using Sqoop into hive table

Introduction:

This document provides an explanation of the sqoop script used to import patient details from remote database

1. Installation of MySql connector jar file.

```
wget https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
```

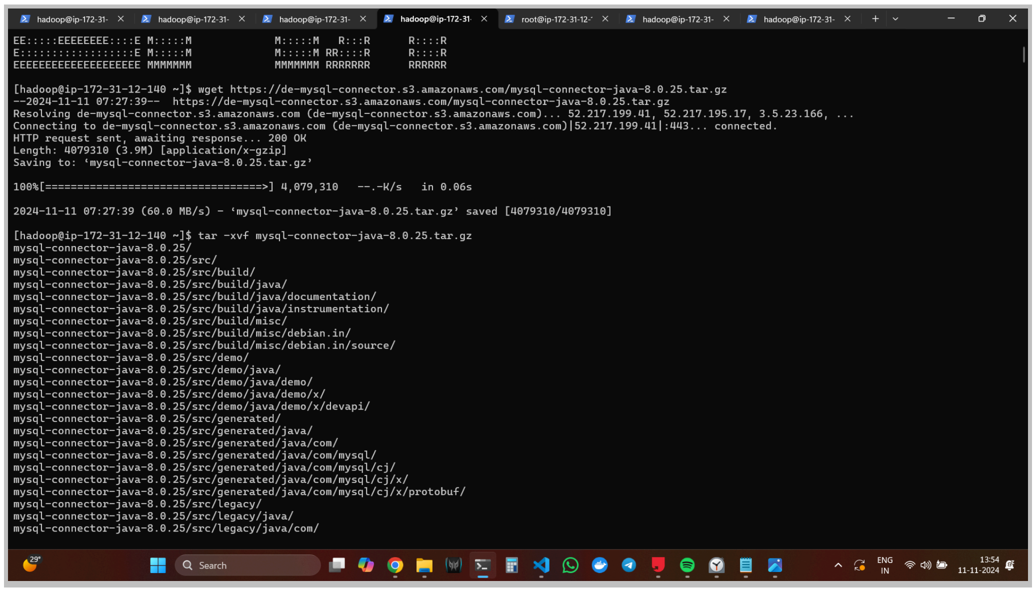


```
[hadoop@ip-172-31-12-140 ~]$ wget https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
--2024-11-11 07:27:39--  https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
Resolving de-mysql-connector.s3.amazonaws.com (de-mysql-connector.s3.amazonaws.com)... 52.217.199.41, 52.217.195.17, 3.5.23.166, ...
Connecting to de-mysql-connector.s3.amazonaws.com (de-mysql-connector.s3.amazonaws.com)|52.217.199.41|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4079310 (3.9M) [application/x-gzip]
Saving to: ‘mysql-connector-java-8.0.25.tar.gz’

100%[===================================>] 4,079,310   --.-K/s   in 0.06s

2024-11-11 07:27:39 (60.0 MB/s) - ‘mysql-connector-java-8.0.25.tar.gz’ saved [4079310/4079310]

[hadoop@ip-172-31-12-140 ~]$ tar -xvf mysql-connector-java-8.0.25.tar.gz
mysql-connector-java-8.0.25/
mysql-connector-java-8.0.25/src/
mysql-connector-java-8.0.25/src/build/
mysql-connector-java-8.0.25/src/build/java/
mysql-connector-java-8.0.25/src/build/java/documentation/
mysql-connector-java-8.0.25/src/build/java/instrumentation/
mysql-connector-java-8.0.25/src/build/misc/
mysql-connector-java-8.0.25/src/build/misc/debian.in/
mysql-connector-java-8.0.25/src/build/misc/debian.in/source/
mysql-connector-java-8.0.25/src/demo/
mysql-connector-java-8.0.25/src/demo/java/
mysql-connector-java-8.0.25/src/demo/java/demo/
mysql-connector-java-8.0.25/src/demo/java/demo/x/
mysql-connector-java-8.0.25/src/demo/java/demo/x/devapi/
mysql-connector-java-8.0.25/src/generated/
mysql-connector-java-8.0.25/src/generated/java/
mysql-connector-java-8.0.25/src/generated/java/com/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/cj/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/cj/x/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/cj/x/protobuf/
mysql-connector-java-8.0.25/src/legacy/
mysql-connector-java-8.0.25/src/legacy/java/
mysql-connector-java-8.0.25/src/legacy/java/com/
```

2. Extract of MySql connector tar file.

This section creates an external Hive table named `threshold_reference_table`. External tables are used when data is stored outside of Hive's own data warehouse; in this case, the table is linked to an HBase table, which facilitates real-time access and modifications to the data.

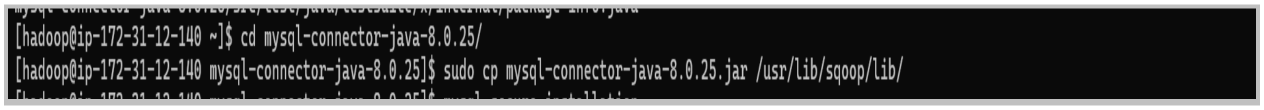
```
tar -xvf mysql-connector-java-8.0.25.tar.gz
```



```
EE:::::EEEEEEEEE::::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-12-140 ~]$ wget https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
--2024-11-11 07:27:39--  https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
Resolving de-mysql-connector.s3.amazonaws.com (de-mysql-connector.s3.amazonaws.com)... 52.217.199.41, 52.217.195.17, 3.5.23.166, ...
Connecting to de-mysql-connector.s3.amazonaws.com (de-mysql-connector.s3.amazonaws.com)|52.217.199.41|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4079310 (3.9M) [application/x-gzip]
Saving to: ‘mysql-connector-java-8.0.25.tar.gz’

100%[===================================>] 4,079,310   --.-K/s   in 0.06s

2024-11-11 07:27:39 (60.0 MB/s) - ‘mysql-connector-java-8.0.25.tar.gz’ saved [4079310/4079310]

[hadoop@ip-172-31-12-140 ~]$ tar -xvf mysql-connector-java-8.0.25.tar.gz
mysql-connector-java-8.0.25/
mysql-connector-java-8.0.25/src/
mysql-connector-java-8.0.25/src/build/
mysql-connector-java-8.0.25/src/build/java/
mysql-connector-java-8.0.25/src/build/java/documentation/
mysql-connector-java-8.0.25/src/build/java/instrumentation/
mysql-connector-java-8.0.25/src/build/misc/
mysql-connector-java-8.0.25/src/build/misc/debian.in/
mysql-connector-java-8.0.25/src/build/misc/debian.in/source/
mysql-connector-java-8.0.25/src/demo/
mysql-connector-java-8.0.25/src/demo/java/
mysql-connector-java-8.0.25/src/demo/java/demo/
mysql-connector-java-8.0.25/src/demo/java/demo/x/
mysql-connector-java-8.0.25/src/demo/java/demo/x/devapi/
mysql-connector-java-8.0.25/src/generated/
mysql-connector-java-8.0.25/src/generated/java/
mysql-connector-java-8.0.25/src/generated/java/com/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/cj/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/cj/x/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/cj/x/protobuf/
mysql-connector-java-8.0.25/src/legacy/
mysql-connector-java-8.0.25/src/legacy/java/
mysql-connector-java-8.0.25/src/legacy/java/com/
```

3. Copy the MySQL Connector directory to the Sqoop library to complete the installation.

```
cd mysql-connector-java-8.0.25/
sudo cp mysql-connector-java-8.0.25.jar /usr/lib/sqoop/lib/
```

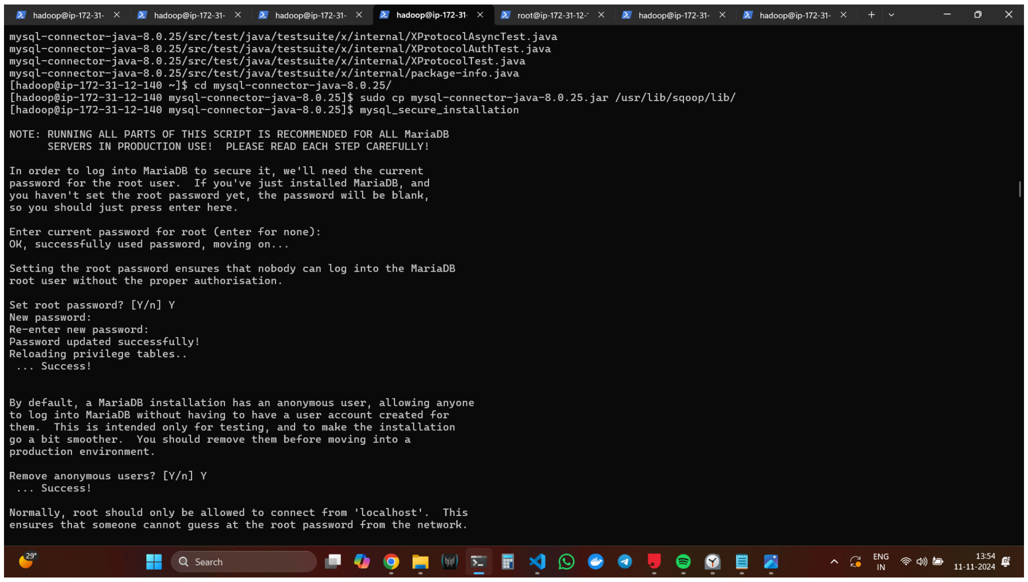


```
[hadoop@ip-172-31-12-140 ~]$ cd mysql-connector-java-8.0.25/
[hadoop@ip-172-31-12-140 mysql-connector-java-8.0.25]$ sudo cp mysql-connector-java-8.0.25.jar /usr/lib/sqoop/lib/
```

4. Install MySql connector in EMR cluster using the extracted folder

```
mysql_secure_installation
```

Enter current password for root (enter for none): ENTER
Set root password [Y/n] Y
New password: 123
Re-enter password: 123
Remove anonymous users [Y/n] Y
Disallow root login remotely [Y/n] n
Remove test database and access to it [Y/n] Y
Reload privilege tables now [Y/n] Y

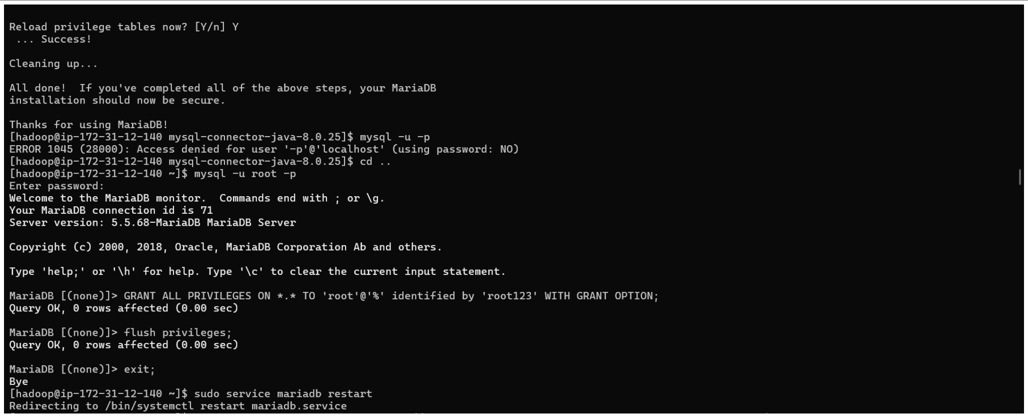


```
mysql-connector-java-8.0.25/src/test/java/testsuite/x/internal/XProtocolAsyncTest.java
mysql-connector-java-8.0.25/src/test/java/testsuite/x/internal/XProtocolAuthTest.java
mysql-connector-java-8.0.25/src/test/java/testsuite/x/internal/XProtocolTest.java
mysql-connector-java-8.0.25/src/test/java/testsuite/x/internal/package-info.java
[hadoop@ip-172-31-12-140 ~]$ cd mysql-connector-java-8.0.25/
[hadoop@ip-172-31-12-140 mysql-connector-java-8.0.25]$ sudo cp mysql-connector-java-8.0.25.jar /usr/lib/sqoop/lib/
[hadoop@ip-172-31-12-140 mysql-connector-java-8.0.25]$ mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] Y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] Y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.
```

5. Accessing the MySql shell

After this a prompt for password will be shown hit 123 and grant root privileges and restart mariaDB

```
mysql -u root -p
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' identified by 'root123' WITH GRANT OPTION;
flush privileges;
exit;
sudo service mariadb restart
```

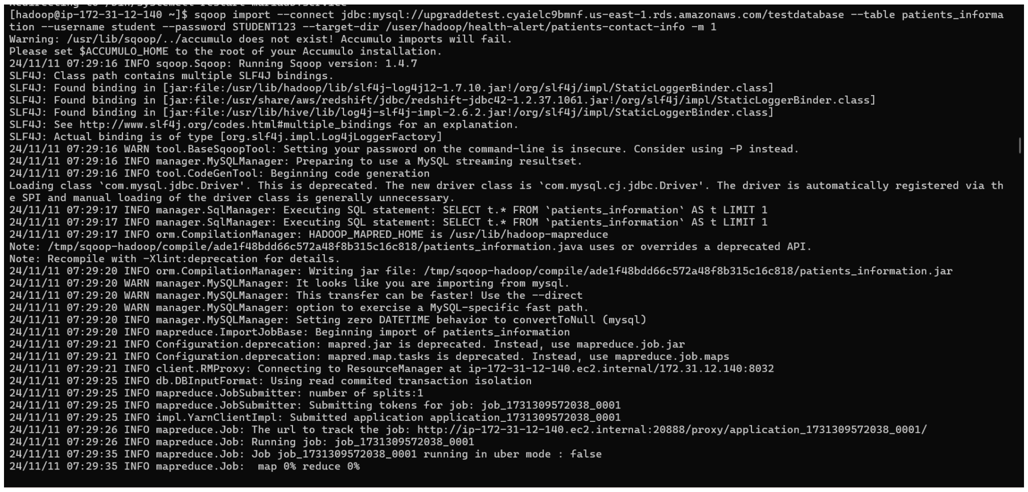


```
Reload privilege tables now? [Y/n] Y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
[hadoop@ip-172-31-12-140 mysql-connector-java-8.0.25]$ mysql -u -p
ERROR 1045 (28000): Access denied for user '-p'@'localhost' (using password: NO)
[hadoop@ip-172-31-12-140 mysql-connector-java-8.0.25]$ cd ..
[hadoop@ip-172-31-12-140 ~]$ mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 71
Server version: 5.5.68-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' identified by 'root123' WITH GRANT OPTION;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> exit;
Bye
[hadoop@ip-172-31-12-140 ~]$ sudo service mariadb restart
Redirecting to /bin/systemctl restart mariadb.service
```

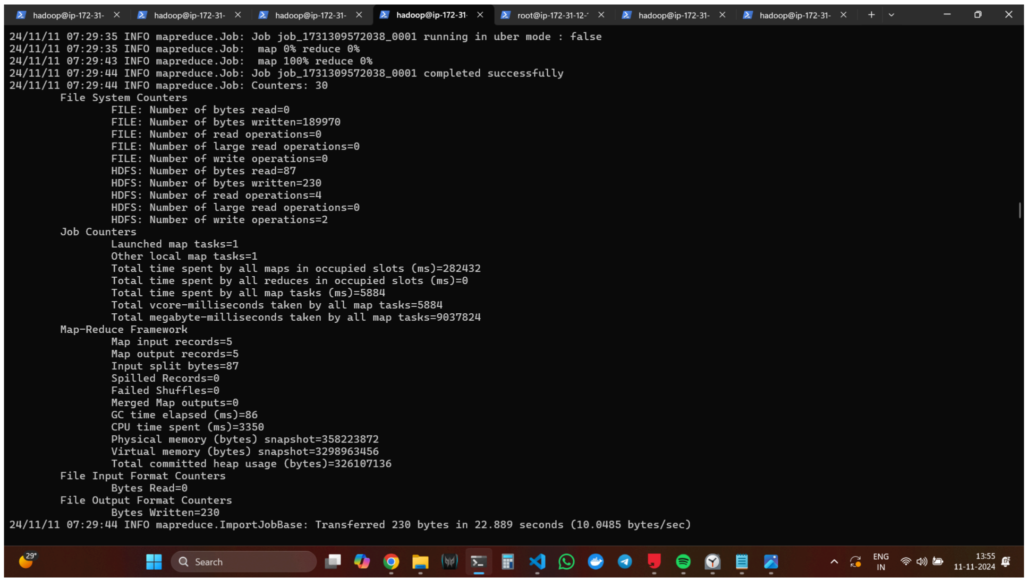
Sqoop Commands

1. Importing patients information into HDFS

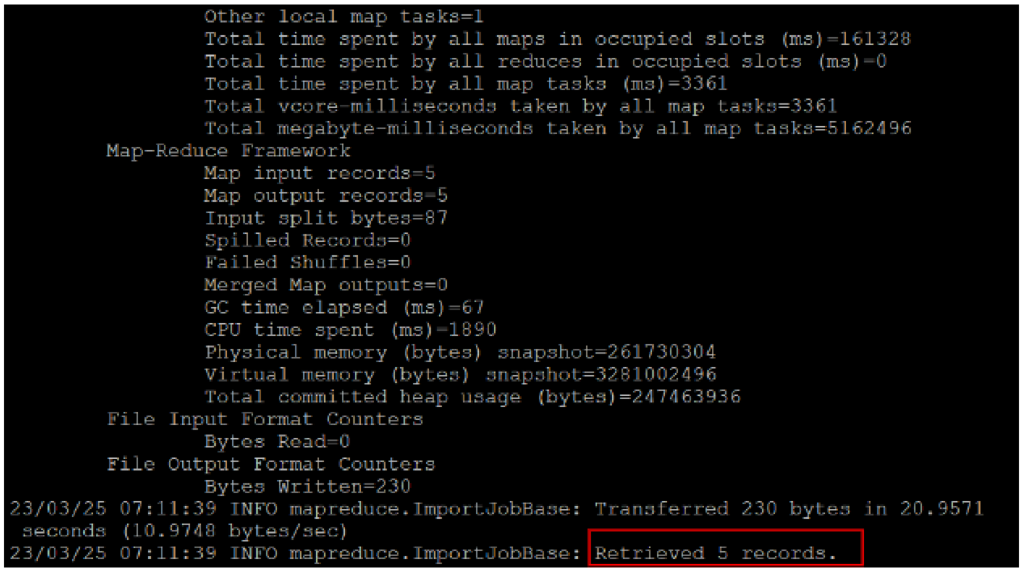
```
sqoop import --connect
jdbc:mysql://upgraddetest.cyaielc9bmnf.us-east-1.rds.amazonaws.com/testdatabase --table
patients_information --username student --password STUDENT123 --target-dir
/user/hadoop/health-alert/patients-contact-info -m 1
```



```
[hadoop@ip-172-31-12-140 ~]$ sqoop import --connect jdbc:mysql://upgraddetest.cyaielc9bmnf.us-east-1.rds.amazonaws.com/testdatabase --table patients_informa
tion --username student --password STUDENT123 --target-dir /user/hadoop/health-alert/patients-contact-info -m 1
Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
24/11/11 07:29:16 INFO sqoop.Sqoop: Running Sqoop version: 1.4.7
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/share/aws/redshift/jdbc/redshift-jdbc42-1.2.37.1061.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
24/11/11 07:29:16 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
24/11/11 07:29:16 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
24/11/11 07:29:16 INFO tool.CodeGenTool: Beginning code generation
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered via th
e SPI and manual loading of the driver class is generally unnecessary.
24/11/11 07:29:17 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `patients_information` AS t LIMIT 1
24/11/11 07:29:17 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `patients_information` AS t LIMIT 1
24/11/11 07:29:17 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /usr/lib/hadoop-mapreduce
Note: /tmp/sqoop-hadoop/compile/ade1f48bdd66c572a48f8b315c16c818/patients_information.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
24/11/11 07:29:20 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-hadoop/compile/ade1f48bdd66c572a48f8b315c16c818/patients_information.jar
24/11/11 07:29:20 WARN manager.MySQLManager: It looks like you are importing from mysql.
24/11/11 07:29:20 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
24/11/11 07:29:20 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
24/11/11 07:29:20 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
24/11/11 07:29:20 INFO mapreduce.ImportJobBase: Beginning import of patients_information
24/11/11 07:29:21 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
24/11/11 07:29:21 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
24/11/11 07:29:21 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-12-140.ec2.internal/172.31.12.140:8032
24/11/11 07:29:25 INFO db.DBInputFormat: Using read commited transaction isolation
24/11/11 07:29:25 INFO mapreduce.JobSubmitter: number of splits:1
24/11/11 07:29:25 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1731309572038_0001
24/11/11 07:29:25 INFO impl.YarnClientImpl: Submitted application application_1731309572038_0001
24/11/11 07:29:26 INFO mapreduce.Job: The url to track the job: http://ip-172-31-12-140.ec2.internal:20888/proxy/application_1731309572038_0001/
24/11/11 07:29:26 INFO mapreduce.Job: Running job: job_1731309572038_0001
24/11/11 07:29:35 INFO mapreduce.Job: Job job_1731309572038_0001 running in uber mode : false
24/11/11 07:29:35 INFO mapreduce.Job:  map 0% reduce 0%
```



```
24/11/11 07:29:35 INFO mapreduce.Job: Job job_1731309572038_0001 running in uber mode : false
24/11/11 07:29:35 INFO mapreduce.Job:  map 0% reduce 0%
24/11/11 07:29:43 INFO mapreduce.Job:  map 100% reduce 0%
24/11/11 07:29:44 INFO mapreduce.Job: Job job_1731309572038_0001 completed successfully
24/11/11 07:29:44 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=189970
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=87
                HDFS: Number of bytes written=230
                HDFS: Number of read operations=4
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Other local map tasks=1
                Total time spent by all maps in occupied slots (ms)=282432
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=5884
                Total vcore-milliseconds taken by all map tasks=5884
                Total megabyte-milliseconds taken by all map tasks=9037824
        Map-Reduce Framework
                Map input records=5
                Map output records=5
                Input split bytes=87
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=86
                CPU time spent (ms)=3350
                Physical memory (bytes) snapshot=358223872
                Virtual memory (bytes) snapshot=3298963456
                Total committed heap usage (bytes)=326107136
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=230
24/11/11 07:29:44 INFO mapreduce.ImportJobBase: Transferred 230 bytes in 22.889 seconds (10.0485 bytes/sec)
```

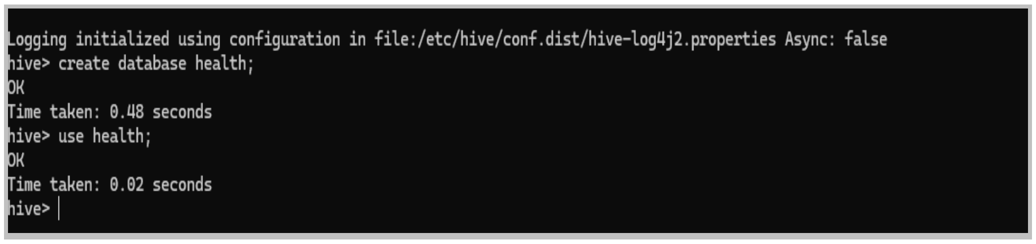


```
                Other local map tasks=1
                Total time spent by all maps in occupied slots (ms)=161328
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=3361
                Total vcore-milliseconds taken by all map tasks=3361
                Total megabyte-milliseconds taken by all map tasks=5162496
        Map-Reduce Framework
                Map input records=5
                Map output records=5
                Input split bytes=87
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=67
                CPU time spent (ms)=1890
                Physical memory (bytes) snapshot=261730304
                Virtual memory (bytes) snapshot=3281002496
                Total committed heap usage (bytes)=247463936
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=230
23/03/25 07:11:39 INFO mapreduce.ImportJobBase: Transferred 230 bytes in 20.9571
 seconds (10.9748 bytes/sec)
23/03/25 07:11:39 INFO mapreduce.ImportJobBase: Retrieved 5 records.
```

2. Creating Hive Table `patients_contact_info` for the imported data

2.1 Create a database health

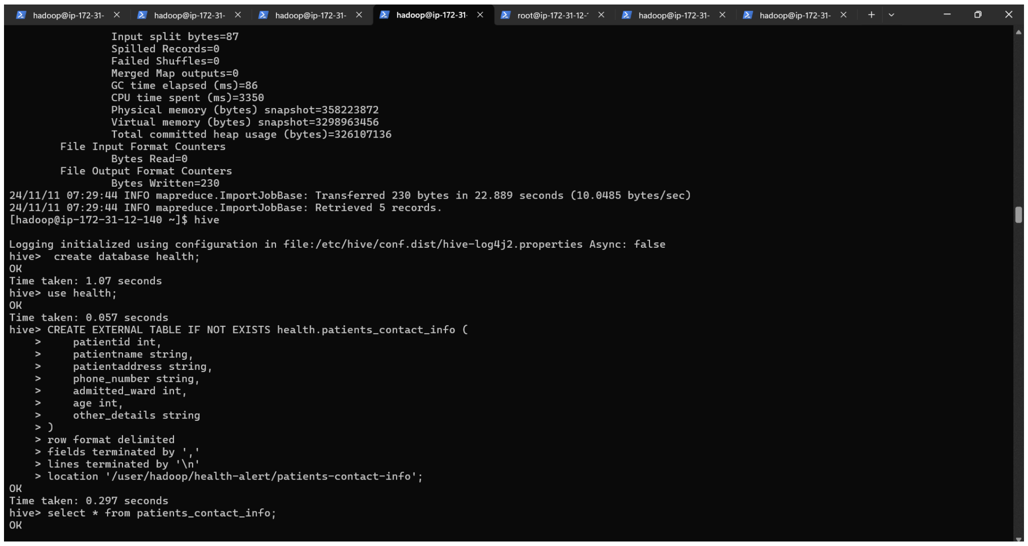
```
create database health;
use health;
```



```
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> create database health;
OK
Time taken: 0.48 seconds
hive> use health;
OK
Time taken: 0.02 seconds
hive> |
```

2.2 Create a table **patients_contact_info**

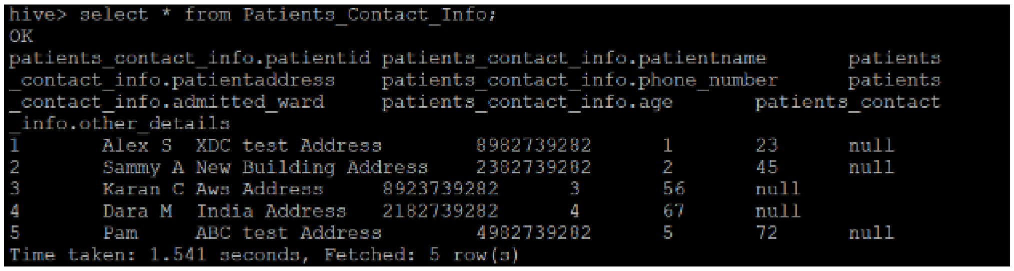
```
CREATE EXTERNAL TABLE IF NOT EXISTS health.patients_contact_info (
    patientid int,
    patientname string,
    patientaddress string,
    phone_number string,
    admitted_ward int,
    age int,
    other_details string
)
row format delimited
fields terminated by ','
lines terminated by '\n'
location '/user/hadoop/health-alert/patients-contact-info';
```



```
                Input split bytes=87
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=86
                CPU time spent (ms)=3350
                Physical memory (bytes) snapshot=358223872
                Virtual memory (bytes) snapshot=3298963456
                Total committed heap usage (bytes)=326107136
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=230
24/11/11 07:29:44 INFO mapreduce.ImportJobBase: Transferred 230 bytes in 22.889 seconds (10.0485 bytes/sec)
24/11/11 07:29:44 INFO mapreduce.ImportJobBase: Retrieved 5 records.
[hadoop@ip-172-31-12-140 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive>  create database health;
OK
Time taken: 1.07 seconds
hive> use health;
OK
Time taken: 0.057 seconds
hive> CREATE EXTERNAL TABLE IF NOT EXISTS health.patients_contact_info (
    >     patientid int,
    >     patientname string,
    >     patientaddress string,
    >     phone_number string,
    >     admitted_ward int,
    >     age int,
    >     other_details string
    > )
    > row format delimited
    > fields terminated by ','
    > lines terminated by '\n'
    > location '/user/hadoop/health-alert/patients-contact-info';
OK
Time taken: 0.297 seconds
hive> select * from patients_contact_info;
OK
```

2.3 View the imported data

```
Select * from  patients_contact_info
```



```
hive> select * from Patients_Contact_Info;
OK
patients_contact_info.patientid patients_contact_info.patientname       patients
_contact_info.patientaddress    patients_contact_info.phone_number      patients
_contact_info.admitted_ward     patients_contact_info.age       patients_contact
_info.other_details
1       Alex S  XDC test Address        8982739282      1       23      null
2       Sammy A New Building Address    2382739282      2       45      null
3       Karan C Aws Address     8923739282      3       56      null
4       Dara M  India Address   2182739282      4       67      null
5       Pam     ABC test Address        4982739282      5       72      null
Time taken: 1.541 seconds, Fetched: 5 row(s)
```