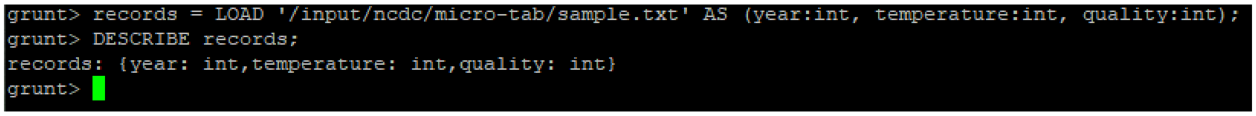
Homework Assignment 4

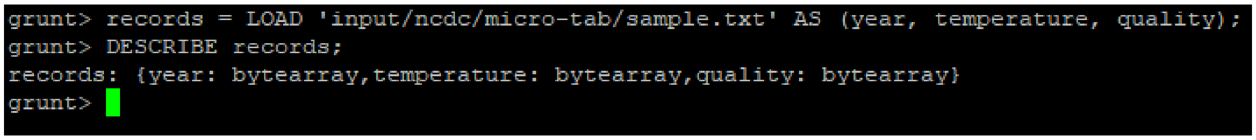
1. Work the five examples on pages 384 and 385, that is, from the title “Schemas” to the subtitle “Validation and nulls.” Print out a screenshot or a couple of screenshots for the results of these five examples in CSUEB Hadoop.

# create records by loading in text data then view the schema of a relation using DESCRIBE
records = LOAD '/input/ncdc/micro-tab/sample.txt' AS (year:int, temperature:int, quality:int);
DESCRIBE records;



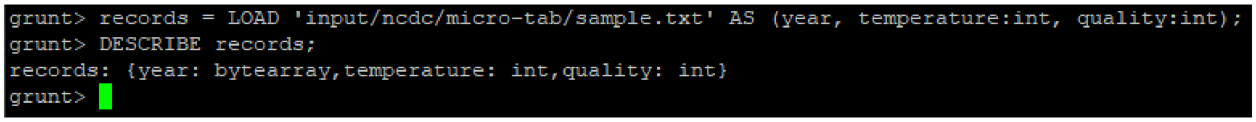
```
grunt> records = LOAD '/input/ncdc/micro-tab/sample.txt' AS (year:int, temperature:int, quality:int);
grunt> DESCRIBE records;
records: {year: int,temperature: int,quality: int}
grunt>
```

# create records by loading in text data then view the schema and omit type declarations completely
records = LOAD 'input/ncdc/micro-tab/sample.txt' AS (year, temperature, quality);
DESCRIBE records;



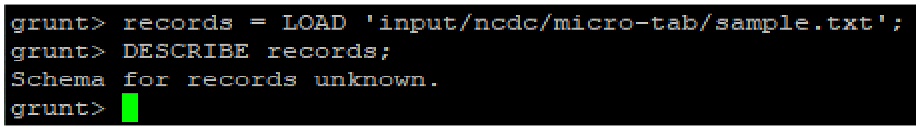
```
grunt> records = LOAD 'input/ncdc/micro-tab/sample.txt' AS (year, temperature, quality);
grunt> DESCRIBE records;
records: {year: bytearray,temperature: bytearray,quality: bytearray}
grunt>
```

# create records by loading in text data then view the schema and leave some to default to byte array
records = LOAD 'input/ncdc/micro-tab/sample.txt' AS (year, temperature:int, quality:int);
DESCRIBE records;



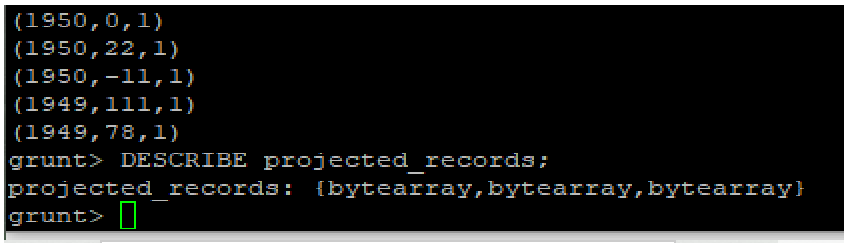
```
grunt> records = LOAD 'input/ncdc/micro-tab/sample.txt' AS (year, temperature:int, quality:int);
grunt> DESCRIBE records;
records: {year: bytearray,temperature: int,quality: int}
grunt>
```

# create records by loading in text data then view the schema and omitted AS clause
records = LOAD 'input/ncdc/micro-tab/sample.txt';
DESCRIBE records;



```
grunt> records = LOAD 'input/ncdc/micro-tab/sample.txt';
grunt> DESCRIBE records;
Schema for records unknown.
grunt>
```

# create records by loading in text data then view the schema and use positional notation
records = LOAD 'hdfs://msba-hadoop-name:9000/home/student29/input/ncdc/micro-tab/sample.txt';
projected_records = FOREACH records GENERATE $0, $1, $2;
DUMP projected_records;
DESCRIBE projected_records;

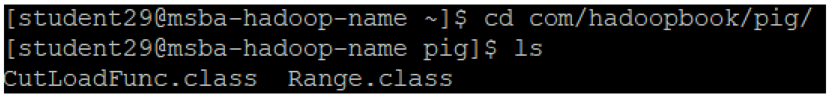


```
(1950,0,1)
(1950,22,1)
(1950,-11,1)
(1949,111,1)
(1949,78,1)
grunt> DESCRIBE projected_records;
projected_records: {bytearray,bytearray,bytearray}
grunt>
```

2. Work “A Load UDF” example on pages 396 and 397. Type out all the commands in each step of the process and print out a screenshot of the final results in CSUEB Hadoop.

# create class files from Range and CutLoadFunc

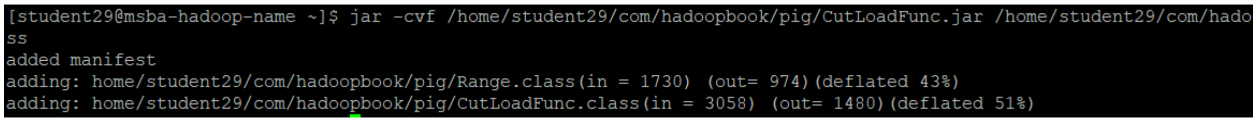
javac -classpath
    /home/student29/hadoop-common-2.6.1.jar:/home/student29/hadoop-mapreduce-client-core-2.6.1.jar:/home/student29/commons-cli-2.0.jar:/home/student29/commons-logging-1.2.jar:/home/student29/pig-0.11.0.jar -d . Range.java CutLoadFunc.java



```
[student29@msba-hadoop-name ~]$ cd com/hadoopbook/pig/
[student29@msba-hadoop-name pig]$ ls
CutLoadFunc.class  Range.class
```

# create jar files

jar -cvf /home/student29/com/hadoopbook/pig/CutLoadFunc.jar
    /home/student29/com/hadoopbook/pig/Range.class
    /home/student29/com/hadoopbook/pig/CutLoadFunc.class



```
[student29@msba-hadoop-name ~]$ jar -cvf /home/student29/com/hadoopbook/pig/CutLoadFunc.jar /home/student29/com/hado
ss
added manifest
adding: home/student29/com/hadoopbook/pig/Range.class(in = 1730) (out= 974)(deflated 43%)
adding: home/student29/com/hadoopbook/pig/CutLoadFunc.class(in = 3058) (out= 1480)(deflated 51%)
```

# start pig

pig

# register jar file in pig

REGISTER /home/student29/com/hadoopbook/pig/CutLoadFunc.jar;

# create record using sample.txt and CutLoadFunc

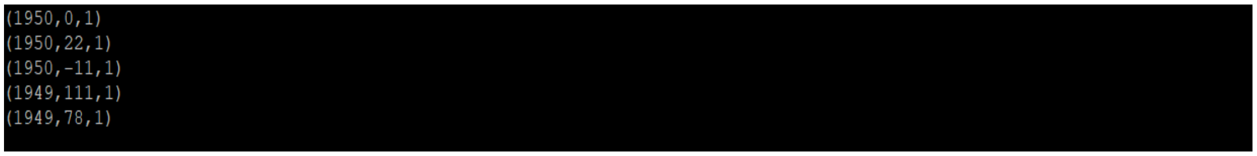
record = LOAD 'sample.txt' USING com.hadoopbook.pig.CutLoadFunc('16-19,88-92,93-93') AS
    (year:int, temperature:int, quality:int);

# execute record using dump

DUMP record;

# final output

```
[illegible]
```



```
(1950,0,1)
(1950,22,1)
(1950,-11,1)
(1949,111,1)
(1949,78,1)
```

3. Work the four examples on pages 416 and 417, that is, from the title “An Example” to the title “Running Hive.”.

# setup hive commands - preliminary

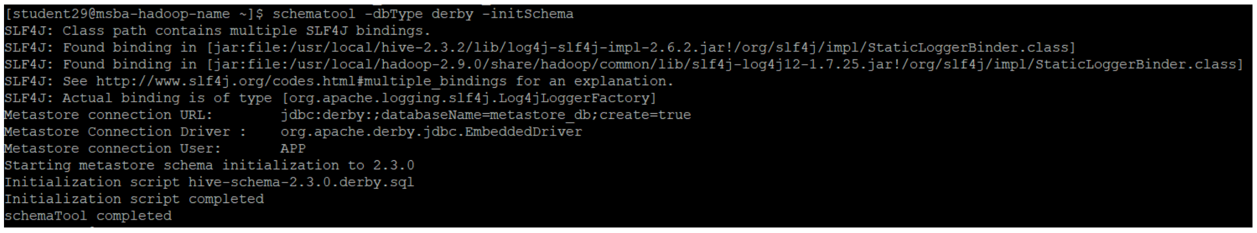
hdfs dfs -mkdir /home/student29/hivedb

hdfs dfs -copyFromLocal sample.txt /home/student29/hivedb/

ls -l | grep meta

mv metastore_db metastore_db.old

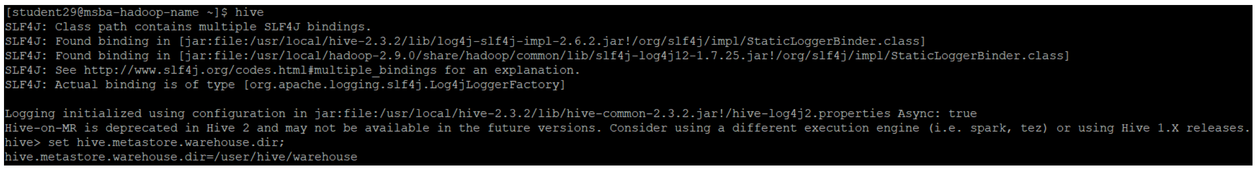
schematool -dbType derby -initSchema



```
[student29@msba-hadoop-name ~]$ schematool -dbType derby -initSchema
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive-2.3.2/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.9.0/share/hadoop/common/lib/slf4j-log4j12-1.7.25.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.apache.logging.slf4j.Log4jLoggerFactory]
Metastore connection URL:        jdbc:derby:;databaseName=metastore_db;create=true
Metastore Connection Driver :    org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User:       APP
Starting metastore schema initialization to 2.3.0
Initialization script hive-schema-2.3.0.derby.sql
Initialization script completed
schemaTool completed
```

hive

set hive.metastore.warehouse.dir;

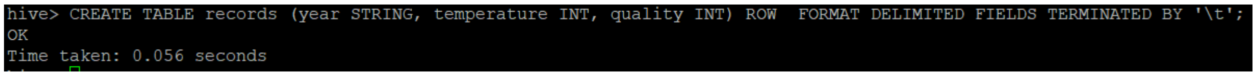


```
[student29@msba-hadoop-name ~]$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive-2.3.2/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.9.0/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/usr/local/hive-2.3.2/lib/hive-common-2.3.2.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
hive> set hive.metastore.warehouse.dir;
hive.metastore.warehouse.dir=/user/hive/warehouse
```

# create table in hive

CREATE TABLE records (year STRING, temperature INT, quality INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';



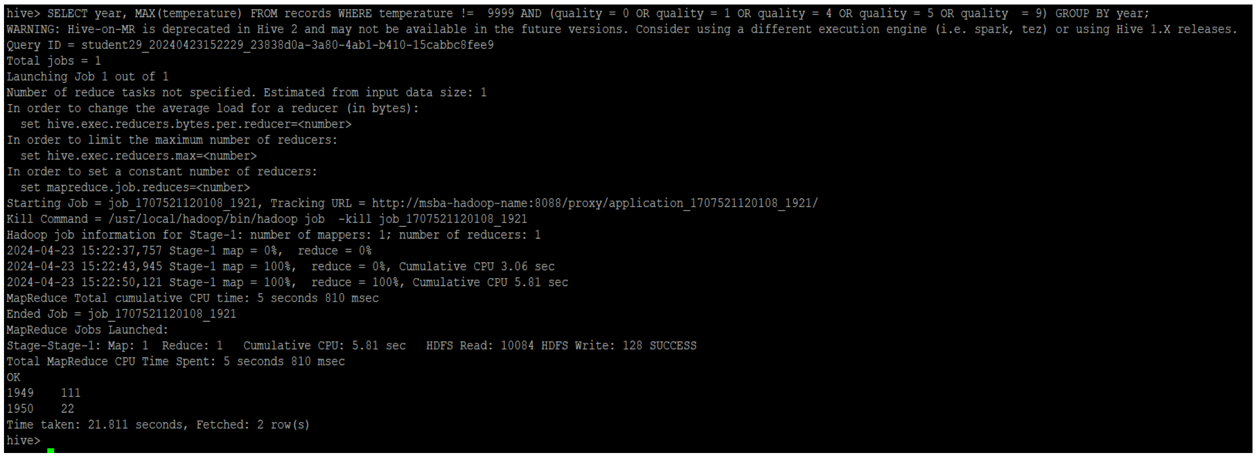
```
hive> CREATE TABLE records (year STRING, temperature INT, quality INT) ROW  FORMAT DELIMITED FIELDS TERMINATED BY '\t';
OK
Time taken: 0.056 seconds
```

# load data and overwrite records table

LOAD DATA LOCAL INPATH 'sample.txt' OVERWRITE INTO TABLE records;

# perform filtering query on the updated records table - final output

SELECT year, MAX(temperature) FROM records WHERE temperature != 9999 AND (quality = 0 OR quality = 1 OR quality = 4 OR quality = 5 OR quality = 9) GROUP BY year;

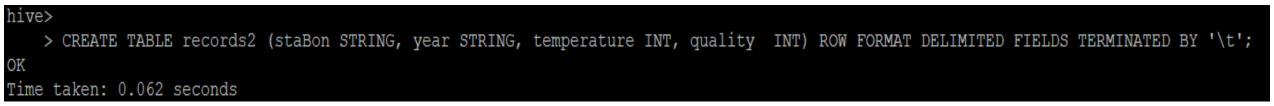


```
hive> SELECT year, MAX(temperature) FROM records WHERE temperature !=  9999 AND (quality = 0 OR quality = 1 OR quality = 4 OR quality = 5 OR quality  = 9) GROUP BY year;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = student29_20240423152229_23838d0a-3a80-4ab1-b410-15cabbc8fee9
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_1707521120108_1921, Tracking URL = http://msba-hadoop-name:8088/proxy/application_1707521120108_1921/
Kill Command = /usr/local/hadoop/bin/hadoop job  -kill job_1707521120108_1921
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2024-04-23 15:22:37,757 Stage-1 map = 0%,  reduce = 0%
2024-04-23 15:22:43,945 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.06 sec
2024-04-23 15:22:50,121 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.81 sec
MapReduce Total cumulative CPU time: 5 seconds 810 msec
Ended Job = job_1707521120108_1921
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.81 sec   HDFS Read: 10084 HDFS Write: 128 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 810 msec
OK
1949    111
1950    22
Time taken: 21.811 seconds, Fetched: 2 row(s)
hive>
```

4. Work the only example on page 444.

   # create table in hive

   CREATE TABLE records2 (staBon STRING, year STRING, temperature INT, quality INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';



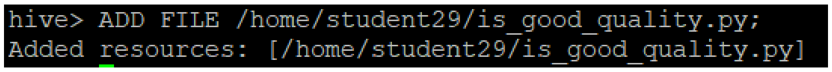
```
hive>
    > CREATE TABLE records2 (staBon STRING, year STRING, temperature INT, quality  INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';
OK
Time taken: 0.062 seconds
```

   # overwrite table with data from sample2.txt

   LOAD DATA LOCAL INPATH '/home/student29/sample2.txt' OVERWRITE INTO TABLE records2;

   # add .py resource

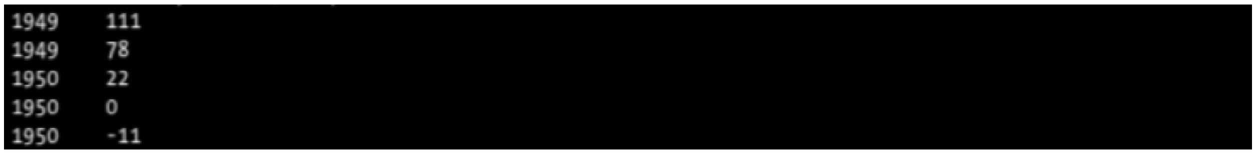
   ADD FILE /home/student29/is_good_quality.py;



```
hive> ADD FILE /home/student29/is_good_quality.py;
Added resources: [/home/student29/is_good_quality.py]
```

   # use the .py script - output

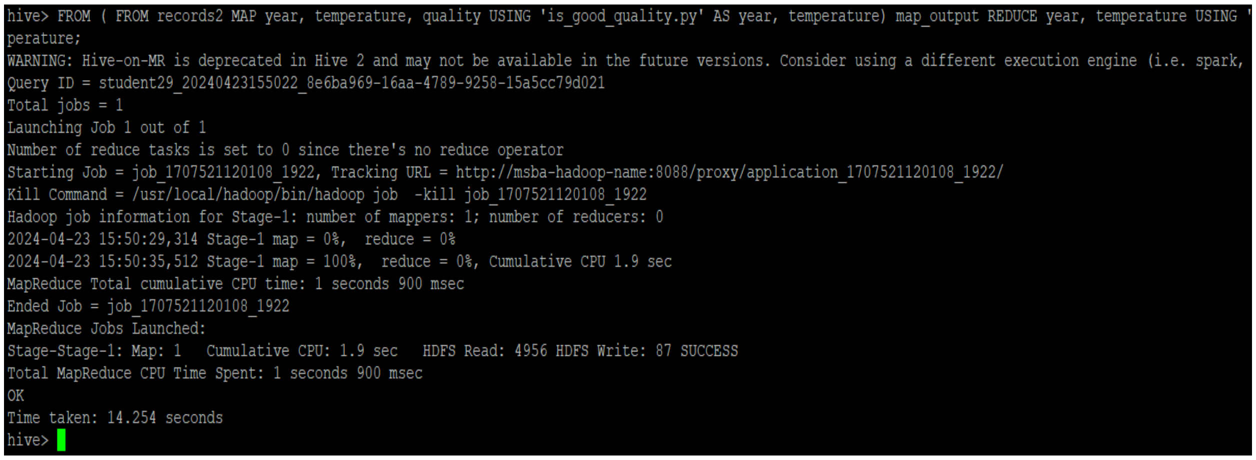
   FROM records2 SELECT TRANSFORM(year, temperature, quality) USING 'is_good_quality.py'
   AS year, temperature;



```
1949    111
1949    78
1950    22
1950    0
1950    -11
```

   # use .py script with mapper and reducer - output

   FROM (FROM records2 MAP year, temperature, quality USING 'is_good_quality.py' AS year, temperature) map_output REDUCE year, temperature USING 'max_temperature_reduce.py' AS year, temperature;



```
hive> FROM ( FROM records2 MAP year, temperature, quality USING 'is_good_quality.py' AS year, temperature) map_output REDUCE year, temperature USING '
perature;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark,
Query ID = student29_20240423155022_8e6ba969-16aa-4789-9258-15a5cc79d021
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1707521120108_1922, Tracking URL = http://msba-hadoop-name:8088/proxy/application_1707521120108_1922/
Kill Command = /usr/local/hadoop/bin/hadoop job  -kill job_1707521120108_1922
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2024-04-23 15:50:29,314 Stage-1 map = 0%,  reduce = 0%
2024-04-23 15:50:35,512 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.9 sec
MapReduce Total cumulative CPU time: 1 seconds 900 msec
Ended Job = job_1707521120108_1922
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 1.9 sec   HDFS Read: 4956 HDFS Write: 87 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 900 msec
OK
Time taken: 14.254 seconds
hive>
```