***JAVA Code for UDF :-***

**import** java.io.IOException;

**import** java.util.regex.Matcher;

**import** java.util.regex.Pattern;

**import** org.apache.pig.EvalFunc;

**import** org.apache.pig.data.DataType;

**import** org.apache.pig.data.Tuple;

**import** org.apache.pig.data.TupleFactory;

**import** org.apache.pig.impl.logicalLayer.schema.Schema;

**import** org.apache.pig.impl.logicalLayer.schema.Schema.FieldSchema;

**public** **class** LogParsing **extends** EvalFunc<Tuple> {

String rex ="^(\\S+) (\\S) (\\S) \\[(\\d{2})\\/([a-zA-Z]{3})\\/(\\d{4}):\\d{2}:\\d{2}:\\d{2} -\\d{4}] \"(\\S+ \\S+\\s\*\\S\*\\s\*)\" (\\d{3}) (\\S+) \"(\\S+)\" \"(.+)\"";

@Override

**public** Tuple exec(Tuple input) **throws** IOException {

Tuple returnTuple = TupleFactory.*getInstance*().newTuple(7);

**if**(input!= **null** && input.size() !=0) {

String logLine = (String) input.get(0);

Pattern pattern = Pattern.*compile*(rex);

Matcher matcher = pattern.matcher(logLine);

**while** (matcher.find()) {

returnTuple.set(0,matcher.group(1));

returnTuple.set(1,matcher.group(4));

returnTuple.set(2,matcher.group(5));

returnTuple.set(3,matcher.group(6));

returnTuple.set(4,matcher.group(7));

returnTuple.set(5,matcher.group(8));

returnTuple.set(6,matcher.group(9));

System.***out***.println("group 1: " + matcher.group(1));

System.***out***.println("group 2: " + matcher.group(2));

System.***out***.println("group 3: " + matcher.group(3));

System.***out***.println("group 4: " + matcher.group(4));

}

}

**return** returnTuple;

}

**public** Schema outputSchema(Schema input) {

**try**{

Schema tupleSchema = **new** Schema();

tupleSchema.add(**new** FieldSchema("ip", DataType.***CHARARRAY***));

tupleSchema.add(**new** FieldSchema("day",DataType.***CHARARRAY***));

tupleSchema.add(**new** FieldSchema("month",DataType.***CHARARRAY***));

tupleSchema.add(**new** FieldSchema("year",DataType.***CHARARRAY***));

tupleSchema.add(**new** FieldSchema("resReq",DataType.***CHARARRAY***));

tupleSchema.add(**new** FieldSchema("res",DataType.***CHARARRAY***));

tupleSchema.add(**new** FieldSchema("dataByte",DataType.***CHARARRAY***));

Schema schema = **new** Schema(**new** FieldSchema(**null**, tupleSchema));

**return** schema;

}**catch** (Exception e){

**return** **null**;

}

}

}

Pig Script :-

REGISTER '/home/manas/pigudf.jar'

DEFINE LogParsing LogParsing();

logLine = LOAD '$inputpath' as (logLine :chararray);

dataTuple = FOREACH logLine GENERATE FLATTEN (LogParsing(logLine));

STORE dataTuple INTO '$outPath' USING PigStorage (',');

***Command to run PIG :-***

Pig -param inputpath='hdfs://user/manas/logAnalysisData/' -param outPath='outPath' LogParsing.pig

***Hive table Creation :***

create External table if not exists logAnalysis

( ip string ,

day string,

month string,

year string,

resReq string,

status int,

dataByte string

)

ROW format delimited fields terminated by ','

Location '/user/manas/outPath/';

***Partition creation by year and month:-***

create table if not exists logAnalysis\_partition

(

ip string ,

day string,

resReq string,

status int,

dataByte string

)

PARTITIONED by (year string,month string)

ROW format delimited fields terminated by ',';

***insert data in to table :-***

insert into logAnalysis\_partition partition (year,month)

select ip,day,resReq,status,dataByte,year,month from logAnalysis

Query :-

select data.ip,data.month,data.status,data.cnt,data.rank from (

select rk.ip,rk.month,rk.status,rk.cnt,rank() over (partition by month ORDER BY month, cnt desc) as rank from (

select month,ip,count(ip) as cnt ,status from loganalysis\_partition where status>= 400 group by ip, month,status order by month,cnt desc) rk

) data where data.rank<6;

