**SELECT \* FROM students;**

**/\***

**+------------+------------+------------+------------+**

**| student\_id | first\_name | last\_name | advisor\_id |**

**+------------+------------+------------+------------+**

**| 1 | John | Blake | 2 |**

**| 2 | Jess | Goldsmith | NULL |**

**| 3 | Tim | Wu | 3 |**

**| 4 | Alvin | Grand | 1 |**

**| 5 | Fox | Zimmermann | 2 |**

**+------------+------------+------------+------------+**

**5 rows in set (0.00 sec)**

**\*/**

**SELECT \* FROM advisors;**

**/\***

**+------------+------------+-----------+**

**| advisor\_id | first\_name | last\_name |**

**+------------+------------+-----------+**

**| 1 | James | Francis |**

**| 2 | Amy | Cheng |**

**| 3 | Lamar | Alexander |**

**| 4 | Anita | Woods |**

**+------------+------------+-----------+**

**4 rows in set (0.00 sec)**

1. Join the above tables to get the students table with advisor details using Spark.

**import org.apache.spark.SparkContext**

**import org.apache.spark.SparkContext.\_**

**import org.apache.spark.SparkConf**

**import org.apache.spark.sql.SQLContext**

**import org.apache.spark.sql.hive.HiveContext**

**object StudentApp {**

**def main(args: Array[String]) {**

**val conf = new SparkConf().setAppName("Student\_Advisor Application")**

**val sc = new SparkContext(conf)**

**val hiveContext=new HiveContext(sc)**

**val student\_advisor\_DF=hiveContext.sql(“select s.student\_id,s.first\_name,s.last\_name,s.advisor\_id,a.first\_name,b.last\_name from students s left outer join advisors a where s.advisor\_id = a.advisor\_id;”)**

**student\_advisor\_DF.show()**

**}**

**}**

1. You have a employee.csv file size of 10G, at the end of the file it contains a row as a footer which is a total of the rows in the csv. Ex: **Total rows is 1M.**

By using spark, Remove the footer of the csv file and make it a single csv file.

**import org.apache.spark.SparkContext**

**import org.apache.spark.SparkContext.\_**

**import org.apache.spark.SparkConf**

**object EmpApp {**

**def main(args: Array[String]) {**

**val conf = new SparkConf().setAppName("Emp Application")**

**val sc = new SparkContext(conf)**

**val empRDD = sc.textFile("hdfs://namenode:9000/user/emp/employee.csv")**

**val numPartitions=empRDD.partitions.size**

**val empNoTrailerRDD = empRDD.mapPartitionsWithIndex{ (idx, iter) =>**

**val size = iter.size();**

**if (idx == numPartitions) {**

**iter.take(size - 1)**

**}**

**else iter**

**}**

**empNoTrailerRDD.coalesce(1).saveAsTextFile("hdfs://namenode:9000/user/emp\_out")**

**}**

1. For a given integer, Find the number of ways to represent it as a sum of two or more consecutive positive integers

Ex: if num = 15 there are 3 such ways (**1+2+3+4+5**, **4+5+6**,**7+8**) **Ans = 3**

If num = 10 there is one way (**1+2+3+4**) **Ans = 1**

Write a method to find the above (take a parameter as integer)choose your known language

**def countConsecutive(N):**

**count = 0**

**L = 1**

**while( L \* (L + 1) < 2 \* N):**

**a = (1.0 \* N - (L \* (L + 1) ) / 2) / (L + 1)**

**if (a - int(a) == 0.0):**

**count += 1**

**L += 1**

**return count**

**N = 15**

**print countConsecutive(N)**

**N = 10**

**print countConsecutive(N)**

1. You have log file like below. Like below many rows

DBName, Table name, rows, date

===========================

bigdata, emp, 10000,01-01-2010

bigdata,finance,3000,03-09-2010

bugdata,mktg,2000,10-10-2010

--------

Now, write a program/script, to check these log file table rows are equal to Hive tables or not. If the table rows are equal print “Table rows are equal” else “Table rows are not equal” Assume that you have Hive and table names are same.

**#!/bin/bash**

**while IFS="," read f1 f2 f3 f4**

**do**

**HIVE\_COUNT=0**

**HIVE\_COUNT=hive -e "select count(1) from ${f1}.${f2}"**

**if [[ $f3 == $HIVE\_COUNT ]]; then**

**echo "${f2} Rows are equal"**

**else**

**echo "${f2} Rows are not equal"**

**fi**

**done <log\_file.csv**