Assignment 16.2

Step 1: Start Hadoop Daemons:

```
$ start-all.sh
$ jps
```

```
[acaddild@localhost dataset]$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
17/08/28 16:35:28 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: namenode running as process 2830. Stop it first.
localhost: datanode running as process 2931. Stop it first.
Starting secondary namenodes [0.0.0.0]
0.0.0.0: secondarynamenode running as process 3086. Stop it first.
17/08/28 16:35:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
starting yarn daemons
resourcemanager running as process 3263. Stop it first.
localhost: nodemanager running as process 3364. Stop it first.
localhost: nodemanager running as process 3364. Stop it first.
2931 DataNode
3364 NodeManager
3380 NameNode
3365 SecondaryNameNode
3263 ResourceManager
4111 Jps
[acadgild@localhost dataset]$

[acadgild@localhost dataset]$

[acadgild@localhost dataset]$

[acadgild@localhost dataset]$
```

Step 2: create a dataset as below where it contains id, name, skills:

\$ cat users.txt

```
[acadgild@localhost assignment16]$ vi users.txt
[acadgild@localhost assignment16]$ cat users.txt
101,Amit,HAD00P:HIVE:SPARK:BIG-DATA
102,Sumit,HIVE:00ZIE:HAD00P:SPARK:STORM
103,Rohit,KAFKA:CASSANDRA:HBASE
[acadgild@localhost assignment16]$ ■
```

Step 3: start hive shell and load data:

```
$ sudo service mysqld start
$ hive
USE default;
CREATE TABLE users
(
id INT,
name STRING,
skills ARRAY<STRING>
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t';
LOAD DATA LOCAL INPATH '/home/acadgild/dataset/assignment16/users.txt'
INTO TABLE users;
SELECT * FROM users;
```

```
hive> USE default;
Time taken: 0.012 seconds
hive> CREATE TABLE users
    > id INT,
    > name STRING,
    > skills ARRAY<STRING>
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ',';
Time taken: 0.058 seconds
hive> LOAD DATA LOCAL INPATH '/home/acadgild/dataset/assignment16/users.txt'
    > INTO TABLE users;
Loading data to table default.users
Table default.users stats: [numFiles=1, totalSize=108]
Time taken: 0.342 seconds
hive> SELECT * FROM users;
0K
101
                  ["HADOOP:HIVE:SPARK:BIG-DATA"]
                 ["HIVE:00ZIE:HAD00P:SPARK:STORM"]
["KAFKA:CASSANDRA:HBASE"]
102
        Sumit
103
        Rohit
Time taken: 0.039 seconds, Fetched: 3 row(s)
hive>
```

Write a hive UDF that implements functionality of string concat_ws(string SEP, array<string>). This UDF will accept two arguments, one string and one array of string. It will return a single string where all the elements of the array are separated by the SEP

Step 4: Write the UDF java program which evaluates adding of SEP as a separator in SKILLS parameter JAR file required: hive-exec-0.8.0.JAR

```
_ _
Package Explorer ⋈
                               package hive;
 Examples
                                 import java.util.ArrayList;
 FestivalPortalR2_Participant
                                  import org.apache.hadoop.hive.ql.exec.UDF;
 Hadoop
 Hadoop5
                                  public class SEPArray extends UDF{
 Hadoop6
 HBase
                                      public String evaluate (String separator, ArrayList<String> array) {
 HeadFirstJava
                                          StringBuffer sBuffer;
HIveudf
                                          if (array == null) {
  return null;
   > 

SEPArray.java
 → JRE System Library [JavaSE-1.8]
                                          sBuffer = new StringBuffer();
 sBuffer.append(array.get(0));
    > in hive-exec-0.8.0.jar - C:\Users\Yc
                                          for (int i=1; i < array.size(); i++) {</pre>
 JDBCConnection
                                              sBuffer.append(separator);
 Piq Piq
                                              sBuffer.append(array.get(i));
 Servers
 Spring Tutorial
 sqljdbc42
                                          return sBuffer.toString();
 TestApp
                                      }
                                  }
```

Step 5: compile program into jar and import it to acadgild sandbox:

```
[acadgild@localhost ~]$ cd dataset/assignment16/
[acadgild@localhost assignment16]$ ll
total 12
-rw-rw-r--. 1 acadgild acadgild 115 Aug 28 20:42 employee.txt
-rw-rw-r--. 1 acadgild acadgild 1569 Aug 28 21:20 hiveudf.jar
-rw-rw-r--. 1 acadgild acadgild 108 Aug 28 21:13 users.txt
[acadgild@localhost assignment16]$ ■
```

Step 6: Add jar and create temp function for usage of evaluation:

ADD JAR /home/acadgild/dataset/assignment16/hiveudf.jar; CREATE TEMPORARY FUNCTION concat_ws AS hive.SEPArray;

```
hive> ADD JAR /home/acadgild/dataset/assignment16/hiveudf.jar;
Added [/home/acadgild/dataset/assignment16/hiveudf.jar] to class path
Added resources: [/home/acadgild/dataset/assignment16/hiveudf.jar]
hive> CREATE TEMPORARY FUNCTION concat_ws AS 'hive.SEPArray';
OK
Time taken: 0.007 seconds
hive>
```

Step 7: perform the said concatenation:

SELECT id, name, concat_ws("SEP",skills) as result from users;

```
hive> SELECT id, name, concat_ws("SEP",skills) as result from users;
0K
101
                  HADOOPSEPHIVESEPSPARKSEPBIG-DATA
         Amit
102
                  HIVESEP00ZIESEPHAD00PSEPSPARKSEPST0RM
         Sumit
103
         Rohit
                  KAFKASEPCASSANDRASEPHBASE
Time taken: 0.056 seconds, Fetched: 3 row(s)
hive> SELECT id, name, skills as result from users;
0K
                  ["HADOOP", "HIVE", "SPARK", "BIG-DATA"]
["HIVE", "OOZIE", "HADOOP", "SPARK", "STORM"]
101
         Amit
102
         Sumit
                  ["KAFKA", "CASSANDRA", "HBASE"]
         Rohit
103
Time taken: 0.055 seconds, Fetched: 3 row(s)
hive>
```