**Steps to create hive udf and execute it.**

1. Create an sbt project in IDE.
2. Create a class in src/main/java as follows:

*import org.apache.hadoop.hive.ql.exec.UDF;*

*import org.apache.hadoop.io.Text;*

*public class ToUpper extends UDF {*

*public Text evaluate(Text s) {*

*Text to\_value = new Text("");*

*if (s != null) {*

*try {*

*to\_value.set(s.toString().toUpperCase());*

*} catch (Exception e) {*

*to\_value = new Text(s);*

*}*

*}*

*return to\_value;*

*}*

*}*

1. Add the following dependencies to the build.sbt

*libraryDependencies += "org.apache.hadoop" % "hadoop-core" % "0.20.2" % "provided"*

*libraryDependencies += "org.apache.hive" % "hive-exec" % "0.8.1" % "provided"*

1. Now compile and build the project as follows:
   1. sbt compile
   2. sbt package
2. Copy the jar using winscp or FileZilla
3. Copy the file in hdfs as follows:

*hadoop fs -put hiveudf\_2.12-0.1.jar /example/jars/*

1. Give the necessary permissions as follows:

*hadoop fs -chmod 777 /example/jars/hiveudf\_2.12-0.1.jar*

1. Login to beeline:

*beeline -u jdbc:hive2://headnodehost:10001/default?hive.server2.transport.mode=http*

1. Add the jar to hive

*beeline> add jar adl://yetiadls.azuredatalakestore.net/clusters/yeti-dpe-3600/example/jars/hiveudf\_2.12-0.1.jar;*

1. Create temporary function as follows:

*hive> create temporary function function\_name as ‘ClassName’;*

*e.g. create temporary function to\_upper as ‘ToUpper’;*

1. Use the function created as follows:

*hive> select function\_name(column\_name) from table\_name;*

*e.g. select to\_upper(currency) from test limit 5;*

*samle output:*

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

*| \_c0 |*

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

*| USD |*

*| USD |*

*| USD |*

*| USD |*

*| USD |*

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