**BDT – cs523**

# Lab 4 – Apache Avro Lab

---

- Submit your *own work* on time. No credit will be given if the lab is submitted after the due date.
- Note that the completed lab should be submitted in .zip format only.

---

1. [2] **Avro Word Count**

   Try to run through all the steps and see if they work properly for you.

   Submit the java files, input and output files.

2. [3] **Avro Station-Temperature-Year**

   Submit the java files, input and output files along with the new schema and the screenshot of the JSON produced by avro-tools for the output file.
   *No need to run this question in pseudo-distributed mode.*



3. [5] **Avro Max Temperature**

   Submit all the java files, schema file and the output file along with the screenshot of the JSON produced by avro-tools.
   *No need to run this question in pseudo-distributed mode.*

# 1. Word Count as Avro Data file

The purpose of this question is to give you a feel of how Avro works with MapReduce.

**Make sure that you can run the given *Avro Word Count* programs in Cloudera VM; both locally and in pseudo distributed mode by following these given steps.**

1. Create a new *AvroWordCount* project in Eclipse with the given *WordCountAvroOptput.java* file. (Optionally try out with the given *WordCountTotalAvro.java* file)
2. You need to properly configure the build path of the project by adding external jars from the following locations.

   ```
   File system/usr/lib/hadoop/client-0.20
   File system/usr/lib/hadoop
   File system/usr/lib/hadoop/lib
   File system/usr/lib/avro
   ```

3. Once all the errors are gone, follow the following steps to run the avro word count program in local mode first and then in pseudo distributed mode.
4. Remember to properly take care of some "commented code" and/or add the missing code.

## Local Mode:

- Create a directory in your eclipse project structure as "input" and create a new text file there with some data.
- You'll need to supply runtime arguments to your program as "input" and "output".
- Run the Java program in Eclipse and see that the Avro output data file is created as *"part-r-00000.avro".* This file has data in binary format and schema is also attached to it.
- Next, use the Avro tools (written in Java) to display the contents of *part-r-00000.avro* file.

  The ***tojson*** command converts an Avro datafile to JSON and prints it to the console:

  ```
  avro-tools tojson /home/cloudera/cs523/part-r-00000.avro
  ```

  Or for Pretty JSON, run the following command:

  ```
  avro-tools tojson --pretty /home/cloudera/cs523/part-r-00000.avro
  ```

## Pseudo-distributed Mode:

- For pseudo-distributed mode you need a runnable jar this time. Note that this is different from what we've been doing so far. In runnable jar, you should not give the name of the class in the `hadoop jar` command. (In the Launch configuration at the time of exporting runnable jar, select the class having the driver code. Ignore warnings if there are any!)

- Avro-tools could be used with hdfs file as follows:

  ```
  avro-tools tojson hdfs://localhost/user/cloudera/output/part-r-00000.avro
  ```

## 2. Station-Temperature-Year as Avro Data File

- Use the given *AvroGenericStationTempYear.java* and *NcdcLineReaderUtils.java* to create a project in Eclipse.
- Complete the given *weather.avsc* file so that the schema matches with the output.
- Remember to properly take care of some "commented code" and/or add the missing code.
- A small weather dataset is also given to you. Use that as your input data set.
- Run the program and check the output avro data file using avro-tools.
- Now add the "order" parameter to the schema file so that the final output will have stationId in ascending but the temperature in descending order as shown in the sample output below:

```
011990-99999    10.0    1950
011990-99999    8.6     1901
011990-99999    7.9     1930
……              …       …
012650-99999    12.0    1960
012650-99999    10.5    2015
……
```



**Submit the new schema and the newly generated avro file along with a screenshot of the json output using avro-tools.**

## 3. Avro Max Temperature per Year with Station-Id

- Use the same NCDC small dataset for this problem.
- Now this time the output file should show the maximum temperature per year along with the station that reported it.
- The format should be as follows: Latest year is shown first.

| Year | MaxTemp | StationId |
|---|---|---|
| 2018 | 10.0 | 029720-99999 |
| 1950 | 12.8 | 227070-99999 |
| 1910 | 7.2 | 029720-99999 |

……

For simplicity, if the same MaxTemp is reported by multiple stations then you can just show any one of the station ids. Alternatively, you can show a list of all those station ids.

**Submit the java files, schema file and the newly generated avro output file along with a screenshot of the json output using avro-tools.**