BDT – cs523

Lab 4 - Avro Lab

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

---

1. 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. 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.

3. 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.

1) Producing Word Count as Avro Data file

The purpose of this lab 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.

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".

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 binary compressed data and schema is also attached to it.
- Next, we'll 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:

- Now you are an expert, I don't need to give you instructions for running this program in pseudo-distributed mode. Try it out yourself! 😉

2) Station-Temperature-Year as Avro Data File

1. Use the given *AvroGenericStationTempYear.java* and *NcdcLineReaderUtils.java* to create a project in Eclipse.
2. Modify the given *weather.avsc* file to define your schema.
3. A small weather dataset is also given to you. Use that as your input data set.
4. Run the program and check the output avro data file using avro-tools.
5. Now add the "order" parameter to the schema file so that the final output will have stationId in ascending and temperature in descending order.

```
011990-99999    100     1950
011990-99999    80      1901
011990-99999    70      1930
……              …       …
012650-99999    120     1960
012650-99999    100     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

1. Use the same NCDC small dataset for this problem.
2. Now this time the output file should show the maximum temperature per year reported by a station in the following format:
3. Sort the file showing the latest year first.

Year	MaxTemp	StationId
2018	100	029720-99999
1950	120	227070-99999
1910	70	029720-99999

……

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