Advanced Database Systems (91.673) Fall 2015 Project

Title	Parsing of text files using Hadoop MapReduce and Java
Tools Used	Hadoop MapReduce and Eclipse
Environment	Linux, Pseudo Distributed
Programming Language	Java

Student Information:

Name	Pavankumar Barur Lingaraj
UML ID	01574465
Email	pavankumar_barurlingaraj@student.uml.edu plingara@cs.uml.edu

Project Description:

Main aim of the project is to use MapReduce programming to parse text files and extract required information performing different operations like Selection, Projection, Natural Join and Aggregation. To implement all operations a procedure followed is:

1. Initialize Driver program to instantiate Mapper and Reducer jobs.
2. Input specific file to mapper job.
3. Parse the specific text file line by line, split the fields based on delimiter ‘,’ and store in an array as different columns.
4. Mapper maps the required columns and send <key, value> pair to reducer job.
5. Reducer receives <key, value> pairs from all mapper jobs and perform calculations to output required results.
6. Results are written in to output file.

Computing Selection by MapReduce:

Task: To find cities whose population is larger than 300,000.

Input File: city.txt

Output File: output.txt

Sample input file screenshot:

city.txt

```
1,Kabul,AFG,Kabol,1780000
2,Qandahar,AFG,Qandahar,237500
3,Herat,AFG,Herat,186800
4,Mazar-e-Sharif,AFG,Balkh,127800
5,Amsterdam,NLD,Noord-Holland,731200
```

Sample output file screenshot:

query1

```
City: Abeokuta; Population: 427400
City: Abidjan; Population: 2500000
City: Abu Dhabi; Population: 398695
City: Abu Dhabi; Population: 398695
City: Abu Dhabi; Population: 398695
```

Mapper Job:

1. Lines of city.txt are split based on delimiter ‘,’ and stored in an array as separate columns.
2. Column 2 and 5 are mapped and sent as <key, value> pair to reducer.

Reducer Job:

1. All the <key, value> pairs are received from all mapper jobs.
2. Rows are filtered based on condition population > 300,000.(**Select Operation**)
3. Result rows are written to output file.

Computing Projection by MapReduce:

Task: To find all the name of the cities and corresponding district.

Input File: city.txt

Output File: output.txt

Sample input file screenshot:

city.txt

```
1,Kabul,AFG,Kabol,1780000
2,Qandahar,AFG,Qandahar,237500
3,Herat,AFG,Herat,186800
4,Mazar-e-Sharif,AFG,Balkh,127800
5,Amsterdam,NLD,Noord-Holland,731200
```

Sample output file screenshot:

query2

```
City: Aachen; District: Nordrhein-Westfalen
City: Aalborg; District: Nordjylland
City: Aba; District: Imo & Abia
City: Aba; District: Imo & Abia
City: Aba; District: Imo & Abia
```

Mapper Job:

1. Lines of city.txt are split based on delimiter ‘,’ and stored in an array as separate columns.
2. Column 2 and 4 are mapped and sent as <key, value> pair to reducer.

Reducer Job:

1. All the <key, value> pairs are received from all mapper jobs.
2. Columns are filtered based on required condition.(**Project Operation**)
3. Result columns are written to output file.

Computing Natural Join by MapReduce:

Task: To find all countries whose official language is English.

Input Files: country.txt and countrylanguage.txt

Output File: output.txt

Sample input files screenshot:

countrylanguage.txt

```
1  ABW,Dutch,T,5.3
2  ABW,English,F,9.5
3  ABW,Papiamento,F,76.7
4  ABW,Spanish,F,7.4
5  AFG,Balochi,F,0.9
```

country.txt

```
1  ABW,Aruba,North America,Caribbean,193.00,
2  AFG,Afghanistan,Asia,Southern and Central
3  AGO,Angola,Africa,Central Africa,1246700.
4  AIA,Anguilla,North America,Caribbean,96.0
5  ALB,Albania,Europe,Southern Europe,28748.
```

Sample output file screenshot:

query3

```
1  Country: Aruba; Language: English
2  Country: Anguilla; Language: English
3  Country: Netherlands Antilles; Language: English
4  Country: American Samoa; Language: English
5  Country: Antigua and Barbuda; Language: English
```

Mapper Job:

1. Lines of countrylanguage.txt are split based on delimiter ‘,’ and stored in an array as separate columns.
2. First two columns which represent country code and respective language are mapped and sent as <key, value> pair to reducer.

Reducer Job:

1. All the <key, value> pairs are received from all mapper jobs.
2. Lines of country.txt are split based on delimiter ‘,’ and first two columns which represent country code and respective country name are stored in an array.
3. Join operation performed on received <key, value> pairs and stored columns to extract required results. (**Natural Join Operation**)
4. Results are written to output file.

Computing Aggregation by MapReduce:

Task: To find how many cities each district has.

Input File: city.txt

Output File: output.txt

Sample input file screenshot:

city.txt

```
1,Kabul,AFG,Kabol,1780000
2,Qandahar,AFG,Qandahar,237500
3,Herat,AFG,Herat,186800
4,Mazar-e-Sharif,AFG,Balkh,127800
5,Amsterdam,NLD,Noord-Holland,731200
```

Sample output file screenshot:

query4

```
District : ARMM; Number of Cities : 2
District : Abhasia [Aphazeti]; Number of Cities : 2
District : Abidjan; Number of Cities : 1
District : Abruzzit; Number of Cities : 1
District : Abu Dhabi; Number of Cities : 5
```

Mapper Job:

1. Lines of city.txt are split based on delimiter ‘,’ and stored in an array as separate columns.
2. Column 2 and 4 are mapped and sent as <key, value> pair to reducer.

Reducer Job:

1. All the <key, value> pairs are received from all mapper jobs.
2. Count of number of cities is calculated based on received <key, value> pairs.(**Aggregation Operation**)
3. Results are written to output file.

Problems Faced:

1. Faced many road blocks during installation of Hadoop on Linux environment. Took help of Hadoop related blogs and articles on internet to solve them.
2. Faced difficulties to run few MapReduce functions due to mismatch of Hadoop library versions.

Project Takeaways:

1. Learnt a new tool – Hadoop MapReduce which is a being widely used these days due to handle large volume of data.
2. Invested more time in learning and understanding MapReduce Functions and Libraries which will be useful in long run.
3. Refreshed Linux programming skills and commands.
4. Refreshed Java Programming skills.

Conclusion:

In this project, I am successfully able to implement all the requirements and operations by using Hadoop MapReduce and Java Programming. The project setup details are mentioned in instructions.txt file. Output result file of every query is created in separate folders.