The dataset I used from data.gov is "Hospital Charge Data" which can be found here:

https://data.cms.gov/Medicare/Inpatient-Prospective-Payment-System-IPPS-Provider/97k6-zzx3

A CSV file was generated by clicking "Export – Download as CSV."

This database contains charges from various hospitals for the 100 most common inpatient services and 30 most common outpatient services across the country in 2011. I wanted to find out 2 things: (1) What are the top 5 states with the average highest cost, and (2) What are the 5 procedures that cost on the average the highest?

There are 3 cost fields at the end – "covered charges", "total payments", and "medicare payments." Since "total payments" represents the out of pocket cost paid by the patient, that column was used.

Part 1 - Hive

For Part 1, Hive was used to make the query. To set up, the dataset ("inpatient.csv") was copied to /home/cloudera/hive/inpatient. The Hive commands used can be found in "hive_final_assignment.q".

There were 2 obstacles that were run into when using this data. The first was that the cost values are strings with "$" and embedded commas (for example: "$5,777.24"). Initially I tried reading these as floats but this resulted in all costs being "NULL." This was overcome by initially reading costs as a string, and then making a second table ("inpatient_edited") that contained just the 3 main columns of interest (procedure name, state, and total charge), using a regular expression to remove the dollar sign and commas and converting the result to a float. This can be seen in the INSERT OVERWRITE TABLE command used.

The second obstacle was sometimes the address would contain a comma in it. Those rows would have quotation marks around the address. When I initially tried using ROW FORMAT DELIMITED it would take those embedded commas as a separator, causing wrong values later for the state and total cost for that row. To get around this, I used a CSV serde jar file publicly available by clicking on the "csv-serde-0.9.1.jar" link at:

http://illyayalovyy.github.io/csv-serde/

This was copied to the /home/cloudera/hive directory and included with the command

```
ADD JAR hive/csv-serde-0.9.1.jar
```

and then in CREATE EXTERNAL TABLE adding the following lines:

```
ROW FORMAT SERDE 'com.bizo.hive.serde.csv.CSVSerde'
WITH serdeproperties(
     "separatorChar" = "\,",
     "quoteChar" = "\"")
```

After doing this, the data could be processed. Using the inquiries:

```
SELECT state, avg(total_chargeF) AS highest_avg_cost FROM
   inpatient_edited GROUP BY state SORT BY highest_avg_cost DESC

SELECT procedure, avg(total_chargeF) AS highest_avg_cost FROM
   inpatient_edited GROUP BY procedure SORT BY highest_avg_cost DESC
```

We see that the 5 most expensive states are Alaska, Washington DC, Hawaii, California, and Maryland:

AK	14572.391745298972
DC	12998.029389880952
HI	12775.739504162542
CA	12629.668472019062
MD	12608.947661015507

and the 5 most expensive procedures are Septicemia, Infectious & Parasitic Diseases, Respiratory System Diagnosis w/ Ventilator Support 96+ Hours, Major Small & Large Bowel Procedures, and Spinal Fusion:

870 – SEPTICEMIA OR SEVERE SEPSIS W MV 96+ HOURS	44259.48536716254
853 – INFECTIOUS & PARASITIC DISEASES W O.R. PROCEDURE	40315.961397392806
207 – RESPIRATORY SYSTEM DIAGNOSIS W VENTILATOR SUPPORT	38588.92100171969
329 – MAJOR SMALL & LARGE BOWEL PROCEDURES W MCC	37765.59428459519
460 – SPINAL FUSION EXCEPT CERVICAL W/O MCC	27778.67117410379

Part 2 - Pig

The same command was made using Pig. The commands used can be seen in the file "finalassignment.pig." The first line shows the DFS command used to copy the data into the part of HDFS used for pig work. The jar file "piggybank", which comes standard with the standard Cloudera 5.4.2 setup, was used.

The same 2 obstacles of needing to remove commas embedded within quotes and removing "$" and "," were addressed. To address removing embedded commas, "CSVLoader" from piggybank was used rather than the standard "PigStorage" with a "," delimiter was used. REPLACE and REGEX_EXTRACT were used to get rid of the embedded "$" and "," within the total_charge field (the "BF" and "CF" queries).

The "A = FILTER L BY state != 'Provider State'" line was a simple way to filter out the header line. Finally, the last 3 lines (X, AC, S) are the commands to do the actual query by state. The 3 lines used to do the query by procedure are also shown.

The results from the queries can be seen at the very end of the file. The results are identical to the Hive result.

Part 3 – MapReduce

MapReduce jobs were created to generate the average total cost per procedure and average total cost per state. Since the generated outputs were relatively small files (100 lines for the procedures query and 52 lines for the states query), python scripts were used to do the final sorting by cost rather than doing another MapReduce job.

CSV processing was handled by using an external jar file. The file was downloaded from:

http://www.java2s.com/Code/Jar/o/Downloadopencsv22jar.htm

The resulting .jar file was copied to the lib directory of the Eclipse project and an "Add External Jar" was done for the project to point to this jar file. In the .java file, the following line was added:

```
import au.com.bytecode.opencsv.CSVParser
```

and in the code itself, the following lines call this CSV code so that commas within quotes are properly handled:

```
CSVParser parser = new CSVParser();
String[] parts = parser.parseLine(value.toString());
```

The removal of "$" and "," and conversion to float for the total_charge field are handled using standard java substring and replace commands for the string class.

The source files "InPatient.java" and "InPatientState.java" show the MapReduce code to list average total cost by procedure and by state, respectively. The files "output_by_procedure.txt" and "output_by_state.txt" show the resulting outputs of the MapReduce jobs. For example, in "output_by_procedure.txt", one can see the most expensive procedure entry:

870 - SEPTICEMIA OR SEVERE SEPSIS W MV 96+ HOURS 44259.496

and in "output_by_state.txt" one can see the most expensive state entry:

AK 14572.392

The files "sortProcedure.py" and "sortState.py" are simple python scripts that take "output_by_procedure.txt" and "output_by_state.txt" and generate "output_by_procedure.txt.sorted" and "output_by_state.txt.sorted", respectively. Looking at these files, the top 5 procedures can be seen as:

870 - SEPTICEMIA OR SEVERE SEPSIS W MV 96+ HOURS	44259.496
853 - INFECTIOUS & PARASITIC DISEASES W O.R. PROCEDURE W MCC	40315.984
207 - RESPIRATORY SYSTEM DIAGNOSIS W VENTILATOR SUPPORT 96+ HOURS	38588.934
329 - MAJOR SMALL & LARGE BOWEL PROCEDURES W MCC	37765.605
460 - SPINAL FUSION EXCEPT CERVICAL W/O MCC	27778.666

and the top 5 states as:

AK 14572.392
DC 12998.031
HI 12775.744
CA 12629.705
MD 12608.943

One can see that these match the results from Pig / Hive.

For reference, zip files of the full Eclipse work directories for “InPatient” and “InPatientState” are included.