README:

Datasets:

My own dataset is about wage which is an auxiliary dataset

The main dataset of our team is about perm

wage\_industries.csv is the raw dataset

wage\_clean is the final clean dataset

perm.csv is the clean dataset from my teammate Yue Tan. I used this dataset for analyzing all applicants

perm\_status is also the clean dataset from my teammate Yue Tan but includes the status of the perm application: certified, denied, withdrawn. I used this dataset for analyzing the certified applicants. This dataset is more important that perm.csv

Important Columns we used:

Wage:

Occ\_title: occupation title

Tot\_emp: total employment

a\_mean: mean of the annually wage for all US workers in each occupation

a\_median: median of the annually wage for all US workers in each occupation

h\_mean: mean of the hourly wage for all US workers in each occupation

h\_median: median of the hourly wage for all US workers in each occupation

Perm:

Status: whether the perm application is certified, denied, or withdrawn

Title: occupation title

Wage: annually wage for each applicant

Data Ingestion:

I uploaded wage\_industries.csv, wage\_clean, perm.csv, perm\_status.csv from my computer to hdfs.

I used scp to transfer each of the dataset from my computer to peel

Then, I used hdfs dfs -put to transfer each of the dataset from peel to hdfs

Data Cleaning:

I cleaned the data twice, so I have two separate folders for cleaning code.

Firstly, by mapreduce, I extracted the possible useful columns which are occ\_title, tot\_emp, h\_mean, a\_mean, h\_median, and a\_median.

Secondly, by mapreduce, I got rid of the rows containing missing values.

For the extra 0 columns, you can drop them in excel or use the same format of the cleaning code 2 which got rid of the zero in cleaning code 1.

Data Profiling:

I uploaded the wage dataset into the hive and created a hive table for it.

I showed the first 10 row of the dataset to inspect the data, so I can know what the dataset looks like.

Select \* from wage limit 10;

I counted the number of the rows by the hive command

select count(occ\_title) from wage;

I calculated the average for each numeric columns by the hive command.

select avg(h\_mean) from wage

select avg(a\_mean) from wage

select avg(h\_median) from wage

select avg(a\_median) from wage

Analysis:

Create hive table for wage and perm status:

CREATE EXTERNAL TABLE wage(occ\_title STRING, tot\_emp BIGINT, h\_mean DOUBLE, a\_mean BIGINT, h\_median DOUBLE, a\_median BIGINT) COMMENT 'Bigdata-wage' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/zl2878/476\_big\_data/hw9/input\_wage';

CREATE EXTERNAL TABLE perm\_status(status STRING, title STRING, skill STRING, wage BIGINT, citizenship STRING, education STRING) COMMENT 'Bigdata-perm' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/zl2878/476\_big\_data/hw9/input\_perm\_status';

Make a directories for grouping:

Logged to beeline and grouped my wage\_clean.csv by occ\_title by the hive command. I saved the query result to the hdfs as wage\_group.csv

insert overwrite directory '/user/zl2878/476\_big\_data/final\_code/hive\_group' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' SELECT occ\_title, avg(a\_mean) FROM wage group by occ\_title;

Second, I grouped my perm\_status.csv by title by the hive command and only select the certified applicants. I saved the query result to the hdfs as perm\_status\_group.csv

insert overwrite directory '/user/zl2878/476\_big\_data/final\_code/perm\_status\_group' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' SELECT title, avg(wage) FROM perm\_status where status="Certified" group by title;

Third, I created hive tables for both wage\_group.csv, and perm\_status\_group.csv.

CREATE EXTERNAL TABLE wage\_group(occ\_title STRING, a\_mean BIGINT) COMMENT 'Bigdata-wage' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/zl2878/476\_big\_data/final\_code/hive\_group';

CREATE EXTERNAL TABLE perm\_status\_group(title STRING, wage BIGINT) COMMENT 'Bigdata-perm' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/zl2878/476\_big\_data/final\_code/perm\_status\_group';

Fourth, I merged wage\_group.csv, and perm\_status\_group.csv by occ\_title by hive command. I saved the query result to the hdfs as wage\_perm\_certified.csv

insert overwrite directory '/user/zl2878/476\_big\_data/final\_code/merge\_status' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' select occ\_title, a\_mean, wage from wage\_group inner join perm\_status\_group on occ\_title =title;

Fifth, I created a hive table for wage\_perm\_certified.csv

CREATE EXTERNAL TABLE wage\_perm\_certified(occ\_title STRING, a\_mean BIGINT, wage BIGINT) COMMENT 'Bigdata-merge\_certified' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/zl2878/476\_big\_data/final\_code/merge\_status';

Sixth, I renamed the column a\_mean to a\_mean\_national, and then I renamed the column wage to a\_mean\_perm.

ALTER TABLE wage\_perm\_certified CHANGE a\_mean a\_mean\_national BIGINT;

ALTER TABLE wage\_perm\_certified CHANGE wage a\_mean\_perm\_certified BIGINT;

Seventh, I counted the number of rows in wage\_perm\_certified hive table where a\_mean\_national is larger than a\_mean\_perm.

select count(occ\_title) from wage\_perm\_certified where a\_mean\_national > a\_mean\_perm\_certified;

Eighth, I counted the number of rows in wage\_perm\_certified hive table where a\_mean\_national is smaller than a\_mean\_perm.

select count(occ\_title) from wage\_perm\_certified where a\_mean\_national < a\_mean\_perm\_certified;

194 and 58 are my results.

To analyze all applicants, you should redo the steps above but replace all “perm\_status” with “perm” and replace all “wage\_perm\_certified” with “wage\_perm”.