Apache Hadoop –

A course for undergraduates

Homework Labs, Lecture 10

Lab: Analyzing Disparate Data Sets with Pig

In this lab, you will practice combining, joining, and analyzing the product sales data previously exported from Dualcore's MySQL database so you can observe the effects that the recent advertising campaign has had on sales.

IMPORTANT: Since this lab builds on the previous one, it is important that you successfully complete the previous lab before starting this lab.

Step #1: Show Per-Month Sales Before and After Campaign

Before we proceed with more sophisticated analysis, you should first calculate the number of orders Dualcore received each month for the three months before their ad campaign began (February – April, 2013), as well as for the month during which their campaign ran (May, 2013).

1. Change to the directory for this lab:

```
$ cd $ADIR/exercises/disparate_datasets
```

2. Open the `count_orders_by_period.pig` file in your editor. We have provided the `LOAD` statement as well as a `FILTER` statement that uses a regular expression to match the records in the data range you'll analyze. Make the following additional changes:

   a. Following the `FILTER` statement, create a new relation with just one field: the order's year and month (Hint: Use the `SUBSTRING` built-in function to extract the first part of the `order_dtm` field, which contains the month and year).

   b. Count the number of orders in each of the months you extracted in the previous step.

   c. Display the count by month to the screen

3. Once you have made these changes, try running your script against the data in HDFS:

```
$ pig count_orders_by_period.pig
```

Question: Does the data suggest that the advertising campaign we started in May led to a substantial increase in orders?

Step #2: Count Advertised Product Sales by Month

Our analysis from the previous step suggests that sales increased dramatically the same month Dualcore began advertising. Next, you'll compare the sales of the specific product Dualcore advertised (product ID #1274348) during the same period to see whether the increase in sales was actually related to their campaign.

You will be joining two data sets during this portion of the lab. Since this is the first join you have done with Pig, now is a good time to mention a tip that can have a profound effect on the performance of your script. Filtering out unwanted data from each relation *before* you join them, as we've done in our example, means that your script will need to process less data and will finish more quickly. We will discuss several more Pig performance tips later in class, but this one is worth learning now.

4. Edit the `count_tablet_orders_by_period.pig` file and implement the following:

   a. Join the two relations on the `order_id` field they have in common

   b. Create a new relation from the joined data that contains a single field: the order's year and month, similar to what you did previously in the `count_orders_by_period.pig` file.

   c. Group the records by month and then count the records in each group

   d. Display the results to your screen

5. Once you have made these changes, try running your script against the data in HDFS:

```
$ pig count_tablet_orders_by_period.pig
```

Question: Does the data show an increase in sales of the advertised product corresponding to the month in which Dualcore's campaign was active?

Bonus Lab #1: Calculate Average Order Size

It appears that Dualcore's advertising campaign was successful in generating new orders. Since they sell this tablet at a slight loss to attract new customers, let's see if customers who buy this tablet also buy other things. You will write code to calculate the average number of items for all orders that contain the advertised tablet during the campaign period.

1. Change to the `bonus_01` subdirectory of the current lab:

```
$ cd bonus_01
```

2. Edit the `average_order_size.pig` file to calculate the average as described above. While there are multiple ways to achieve this, it is recommended that you implement the following:

   a. Filter the orders by date (using a regular expression) to include only those placed during the campaign period (May 1, 2013 through May 31, 2013)

   b. Exclude any orders which do not contain the advertised product (product ID #1274348)

   c. Create a new relation containing the `order_id` and `product_id` fields for these orders.

   d. Count the total number of products per order

   e. Calculate the average number of products for all orders

3. Once you have made these changes, try running your script against the data in HDFS:

```
$ pig average_order_size.pig
```

Question: Does the data show that the average order contained at least two items in addition to the tablet Dualcore advertised?

Bonus Lab #2: Segment Customers for Loyalty Program

Dualcore is considering starting a loyalty rewards program. This will provide exclusive benefits to their best customers, which will help to retain them. Another advantage is that it will also allow Dualcore to capture even more data about the shopping habits of their customers; for example, Dualcore can easily track their customers' in-store purchases when these customers provide their rewards program number at checkout.

To be considered for the program, a customer must have made at least five purchases from Dualcore during 2012. These customers will be segmented into groups based on the total retail price of all purchases each made during that year:

- **Platinum**: Purchases totaled at least $10,000
- **Gold**: Purchases totaled at least $5,000 but less than $10,000
- **Silver**: Purchases totaled at least $2,500 but less than $5,000

Since we are considering the total sales price of orders in addition to the number of orders a customer has placed, not every customer with at least five orders during 2012 will qualify. In fact, only about one percent of the customers will be eligible for membership in one of these three groups.

During this lab, you will write the code needed to filter the list of orders based on date, group them by customer ID, count the number of orders per customer, and then filter this to exclude any customer who did not have at least five orders. You will then join this information with the order details and products data sets in order to calculate the total sales of those orders for each customer, split them into the groups based on the criteria described above, and then write the data for each group (customer ID and total sales) into a separate directory in HDFS.

1. Change to the `bonus_02` subdirectory of the current lab:

```
$ cd ../bonus_02
```

2. Edit the `loyalty_program.pig` file and implement the steps described above. The code to load the three data sets you will need is already provided for you.

3. After you have written the code, run it against the data in HDFS:

```
$ pig loyalty_program.pig
```

4. If your script completed successfully, use the `hadoop fs -getmerge` command to create a local text file for each group so you can check your work (note that the name of the directory shown here may not be the same as the one you chose):

```
$ hadoop fs -getmerge /dualcore/loyalty/platinum platinum.txt
$ hadoop fs -getmerge /dualcore/loyalty/gold gold.txt
$ hadoop fs -getmerge /dualcore/loyalty/silver silver.txt
```

5. Use the UNIX `head` and/or `tail` commands to check a few records and ensure that the total sales prices fall into the correct ranges:

```
$ head platinum.txt
$ tail gold.txt
$ head silver.txt
```

6. Finally, count the number of customers in each group:

```
$ wc -l platinum.txt
$ wc -l gold.txt
$ wc -l silver.txt
```

This is the end of the lab.

Lab: Extending Pig with Streaming and UDFs

In this lab you will use the `STREAM` keyword in Pig to analyze metadata from Dualcore's customer service call recordings to identify the cause of a sudden increase in complaints. You will then use this data in conjunction with a user-defined function to propose a solution for resolving the problem.

IMPORTANT: Since this lab builds on the previous one, it is important that you successfully complete the previous lab before starting this lab.

Background Information

Dualcore outsources its call center operations and costs have recently risen due to an increase in the volume of calls handled by these agents. Unfortunately, Dualcore does not have access to the call center's database, but they are provided with recordings of these calls stored in MP3 format. By using Pig's `STREAM` keyword to invoke a provided Python script, you can extract the category and timestamp from the files, and then analyze that data to learn what is causing the recent increase in calls.

Step #1: Extract Call Metadata

Note: Since the Python library we are using for extracting the tags doesn't support HDFS, we run this script in local mode on a small sample of the call recordings. Because you will use Pig's local mode, there will be no need to "ship" the script to the nodes in the cluster.

1. Change to the directory for this lab:

```
$ cd $ADIR/exercises/extending_pig
```

2. A Python script (`readtags.py`) is provided for extracting the metadata from the MP3 files. This script takes the path of a file on the command line and returns a record containing five tab-delimited fields: the file path, call category, agent ID, customer ID, and the timestamp of when the agent answered the call.

   Your first step is to create a text file containing the paths of the files to analyze, with one line for each file. You can easily create the data in the required format by capturing the output of the UNIX `find` command:

```
$ find $ADIR/data/cscalls/ -name '*.mp3' > call_list.txt
```

3. Edit the `extract_metadata.pig` file and make the following changes:

   a. Replace the hardcoded parameter in the `SUBSTRING` function used to filter by month with a parameter named `MONTH` whose value you can assign on the command line. This will make it easy to check the leading call categories for different months without having to edit the script.

   b. Add the code necessary to count calls by category

   c. Display the top three categories (based on number of calls) to the screen.

4. Once you have made these changes, run your script to check the top three categories in the month before Dualcore started the online advertising campaign:

```
$ pig -x local -param MONTH=2013-04 extract_metadata.pig
```

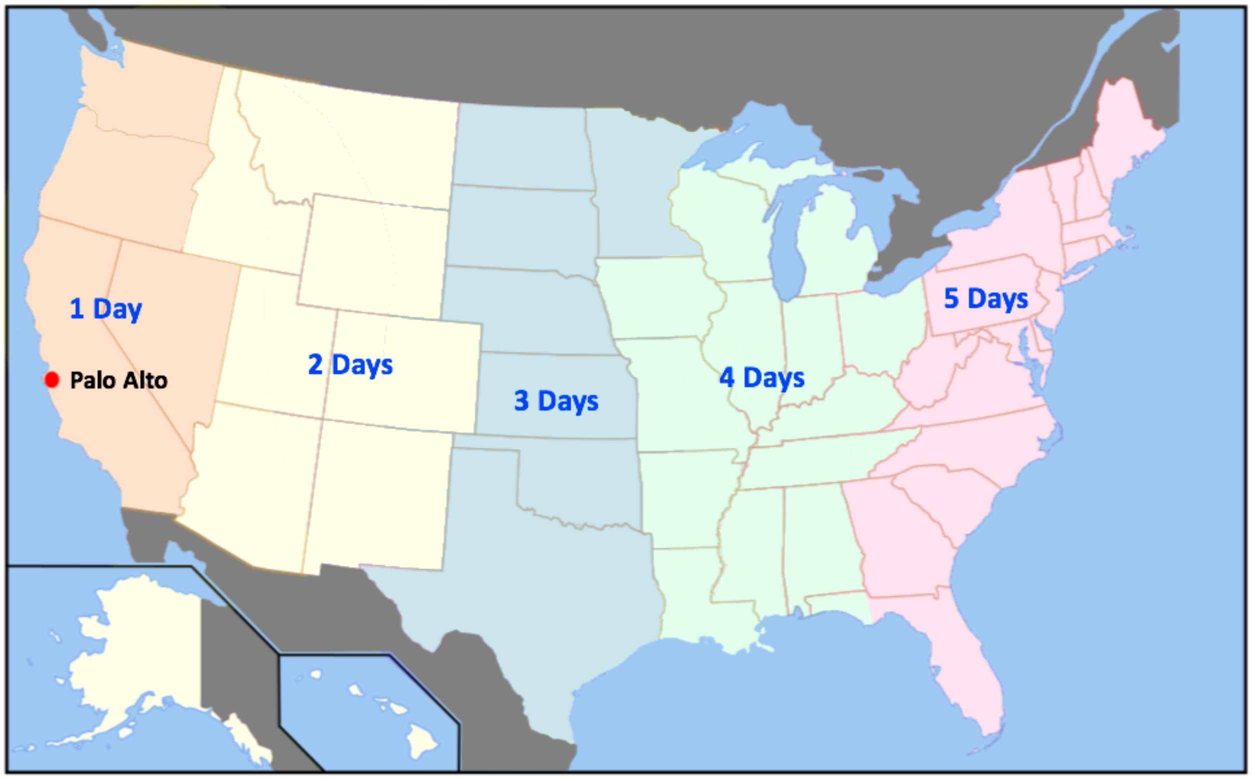
5. Now run the script again, this time specifying the parameter for May:

```
$ pig -x local -param MONTH=2013-05 extract_metadata.pig
```

   The output should confirm that not only is call volume substantially higher in May, the `SHIPPING_DELAY` category has more than twice the amount of calls as the other two.

Step #2: Choose Best Location for Distribution Center

The analysis you just completed uncovered a problem. Dualcore's Vice President of Operations launched an investigation based on your findings and has now confirmed the cause: their online advertising campaign is indeed attracting many new customers, but many of them live far from Dualcore's only distribution center in Palo Alto, California. All shipments are transported by truck, so an order can take up to five days to deliver depending on the customer's location.



To solve this problem, Dualcore will open a new distribution center to improve shipping times.

The ZIP codes for the three proposed sites are 02118, 63139, and 78237. You will look up the latitude and longitude of these ZIP codes, as well as the ZIP codes of customers who have recently ordered, using a supplied data set. Once you have the coordinates, you will invoke the use the `HaversineDistInMiles` UDF distributed with DataFu to determine how far each customer is from the three data centers. You will then calculate the average distance for all customers to each of these data centers in order to propose the one that will benefit the most customers.

1. Add the tab-delimited file mapping ZIP codes to latitude/longitude points to HDFS:

```
$ hadoop fs -mkdir /dualcore/distribution
$ hadoop fs -put $ADIR/data/latlon.tsv \
/dualcore/distribution
```

2. A script (`create_cust_location_data.pig`) has been provided to find the ZIP codes for customers who placed orders during the period of the ad campaign. It also excludes the ones who are already close to the current facility, as well as customers in the remote states of Alaska and Hawaii (where orders are shipped by airplane). The Pig Latin code joins these customers' ZIP codes with the latitude/longitude data set uploaded in the previous step, then writes those three columns (ZIP code, latitude, and longitude) as the result. Examine the script to see how it works, and then run it to create the customer location data in HDFS:

```
$ pig create_cust_location_data.pig
```

3. You will use the `HaversineDistInMiles` function to calculate the distance from each customer to each of the three proposed warehouse locations. This function requires us to supply the latitude and longitude of both the customer and the warehouse. While the script you just executed created the latitude and longitude for each customer, you must create a data set containing the ZIP code, latitude, and longitude for these warehouses. Do this by running the following UNIX command:

```
$ egrep '^02118|^63139|^78237' \
   $ADIR/data/latlon.tsv > warehouses.tsv
```

4. Next, add this file to HDFS:

```
$ hadoop fs -put warehouses.tsv /dualcore/distribution
```

5. Edit the `calc_average_distances.pig` file. The UDF is already registered and an alias for this function named `DIST` is defined at the top of the script, just before the two data sets you will use are loaded. You need to complete the rest of this script:

a. Create a record for every combination of customer and proposed distribution center location

b. Use the function to calculate the distance from the customer to the warehouse

c. Calculate the average distance for all customers to each warehouse

d. Display the result to the screen

6. After you have finished implementing the Pig Latin code described above, run the script:

```
$ pig calc_average_distances.pig
```

Question: Which of these three proposed ZIP codes has the lowest average mileage to Dualcore's customers?

This is the end of the lab.