# Pig Assignment

Using Pig for ETL Processing

### The purpose of this exercise is for you to practice using Pig to explore, correct, and reorder data in files from two different ad networks.

There are two data sets that will be used in this exercise. ***ad\_data1.txt*** and ***ad\_data2.txt***. These represent data from online advertisements used to attract new customers to an ecommerce site. Each of the two ad networks use provides data about the ads they’ve placed. This includes the site where the ad was placed, the date when it was placed, what keywords triggered its display, whether the user clicked the ad, and the per-click cost.

Unfortunately, the data from each network is in a different format. Each file also contains some invalid records. Before we can analyze the data, we must first correct these problems by using Pig to:

* + Filter invalid records
  + Reorder fields
  + Correct inconsistencies
  + Write the corrected data to HDFS

#### **Step 1: Processing Input Data from the First Ad Network**

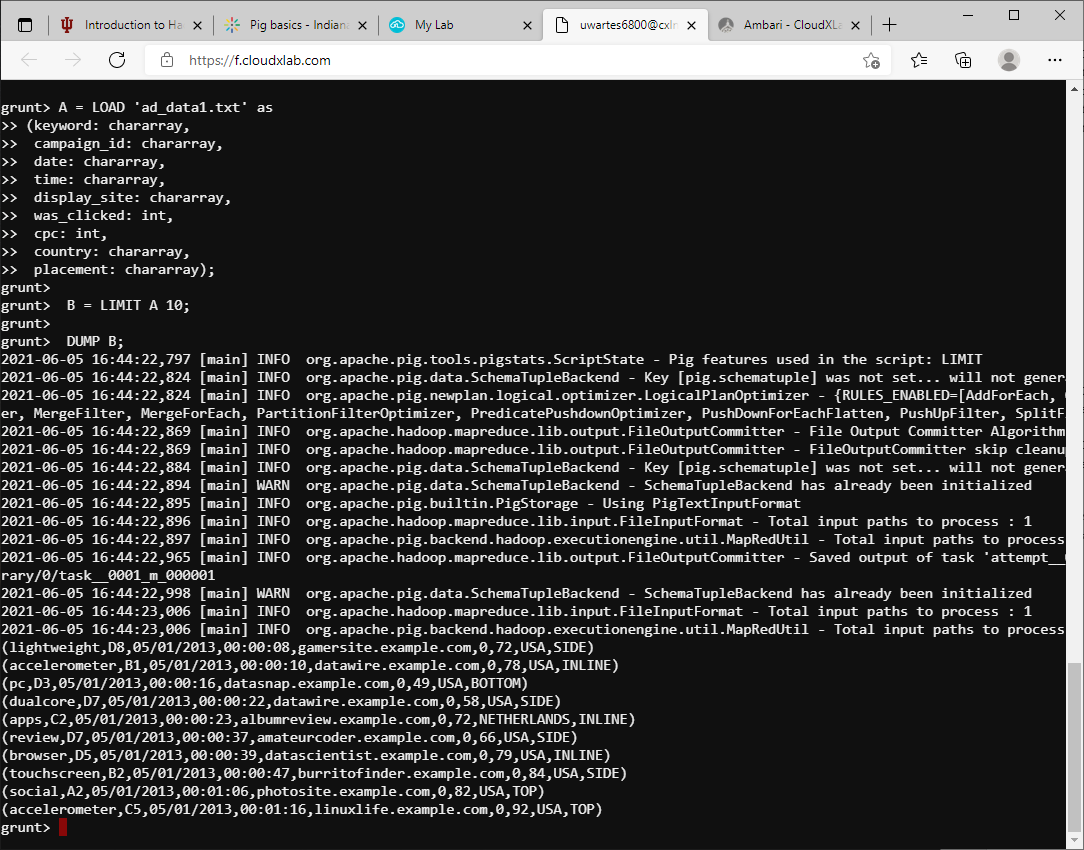
In this step, you will process the input data from the first ad network.

Capture the script in each step in a Word document. Also, capture screen shots to demonstrate that your code did work.

1. Create a pig script called **et11**. You will use the **LOAD** statement and read the data from the **ad\_data1.txt** file. The following table shows the format of the data in the file. For simplicity, you should leave the date and time fields separate, so each will be of type chararray, rather than converting them to a single field of type datetime.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Index | Field | Data Type | Description | Example |
| 0 | keyword | chararray | Keyword that triggered ad | tablet |
| 1 | campaign\_id | chararray | Uniquely identifies our ad | A3 |
| 2 | date | chararray | Date of ad display | 05/29/2013 |
| 3 | time | chararray | Time of ad display | 15:49:21 |
| 4 | display\_site | chararray | Domain where ad shown | [www.example.com](http://www.example.com/) |
| 5 | was\_clicked | int | Whether ad was clicked | 1 |
| 6 | cpc | int | Cost per click, in cents | 106 |
| 7 | country | chararray | Name of country in which ad ran | USA |
| 8 | placement | chararray | Where on page was ad displayed | TOP |

1. Once you have the right LOAD statement , try it out by running your script and ensuring that your schema looks right and that your data (use the **LIMIT** command to display a subset of the data) is indeed loaded correctly. Capture a screen shot of the schema display and the sample data. Capture this script in a txt or word document.



A = LOAD 'ad\_data1.txt' as

(keyword: chararray,

campaign\_id: chararray,

date: chararray,

time: chararray,

display\_site: chararray,

was\_clicked: int,

cpc: int,

country: chararray,

placement: chararray);

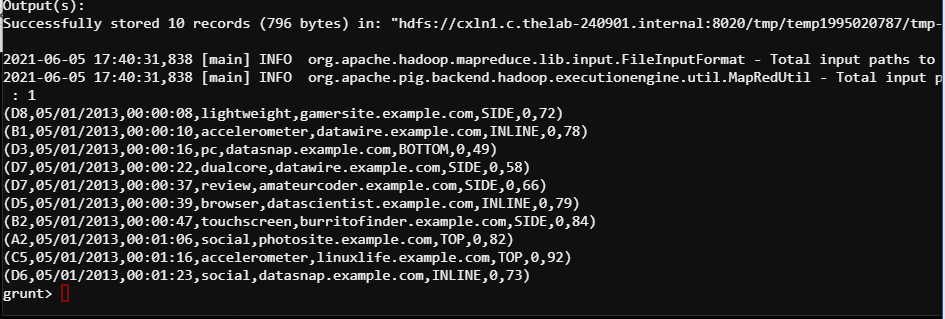
B = LIMIT A 10;

DUMP B;

1. Make each of the following changes, running your script after each one to verify that your change is correct:
   1. Update your script to eliminate all records where the country field does not contain USA.
   2. We need to store the fields in a different order than we received them. Use a FOREACH … GENERATE statement to create a new bag containing the fields in the same order as shown in the following table (NOTE: the country field is not included since all records now have the same value):

|  |  |  |
| --- | --- | --- |
| Index | Field | Description |
| 0 | campaign\_id | Uniquely identifies our ad |
| 1 | date | Date of ad display |
| 2 | time | Time of ad display |
| 3 | keyword | Keyword that triggered ad |
| 4 | display\_site | Domain where ad shown |
| 5 | placement | Where on page was ad displayed |
| 6 | was\_clicked | Whether ad was clicked |
| 7 | cpc | Cost per click, in cents |





A = LOAD 'ad\_data1.txt' as

(keyword: chararray,

campaign\_id: chararray,

date: chararray,

time: chararray,

display\_site: chararray,

was\_clicked: int,

cpc: int,

country: chararray,

placement: chararray);

B = FILTER A by (country == 'USA');

C = FOREACH B GENERATE campaign\_id, date, time, keyword, display\_site, placement, was\_clicked, cpc;

D = LIMIT C 10;

DUMP D;

* 1. Update your script to convert the **keyword** field to uppercase and to remove any leading or trailing whitespace (hint: you can nest calls to the two built-in functions inside the **FOREACH … GENERATE** statement from the last statement).

1. Use a **STORE** statement to write the output of step 3 above as tab-delimited records to an **ad\_data1** directory within your <home> directory in HDFS. <home> refers to your home directory.

A = LOAD 'ad\_data1.txt' as

(keyword: chararray,

campaign\_id: chararray,

date: chararray,

time: chararray,

display\_site: chararray,

was\_clicked: int,

cpc: int,

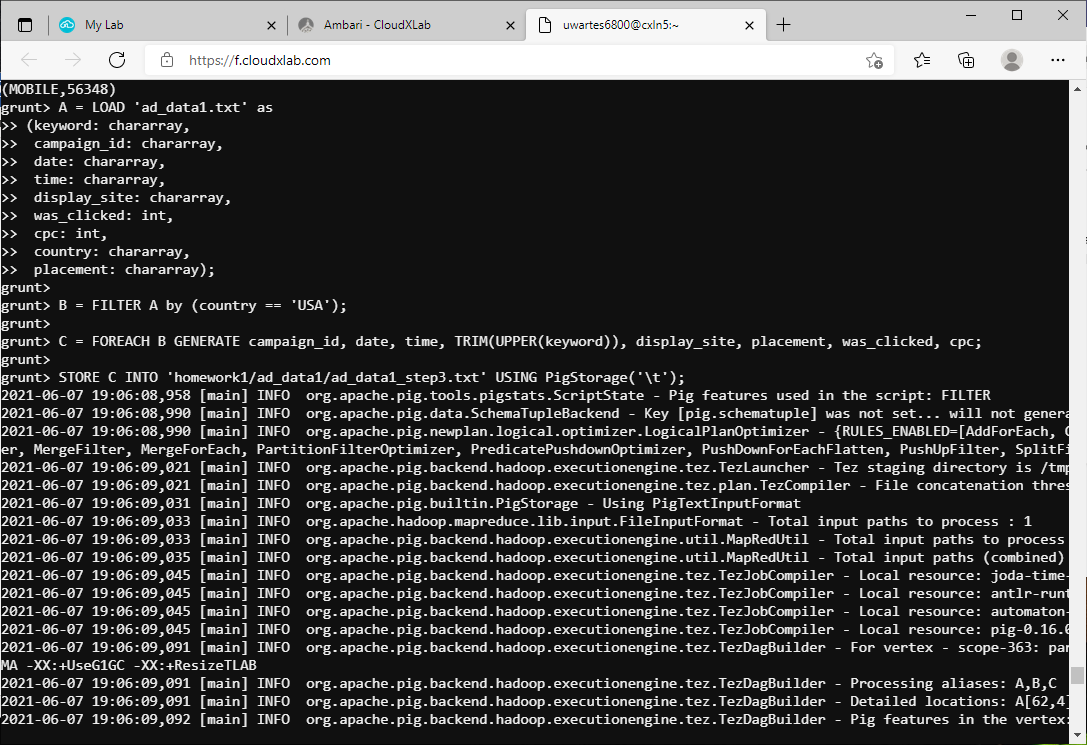
country: chararray,

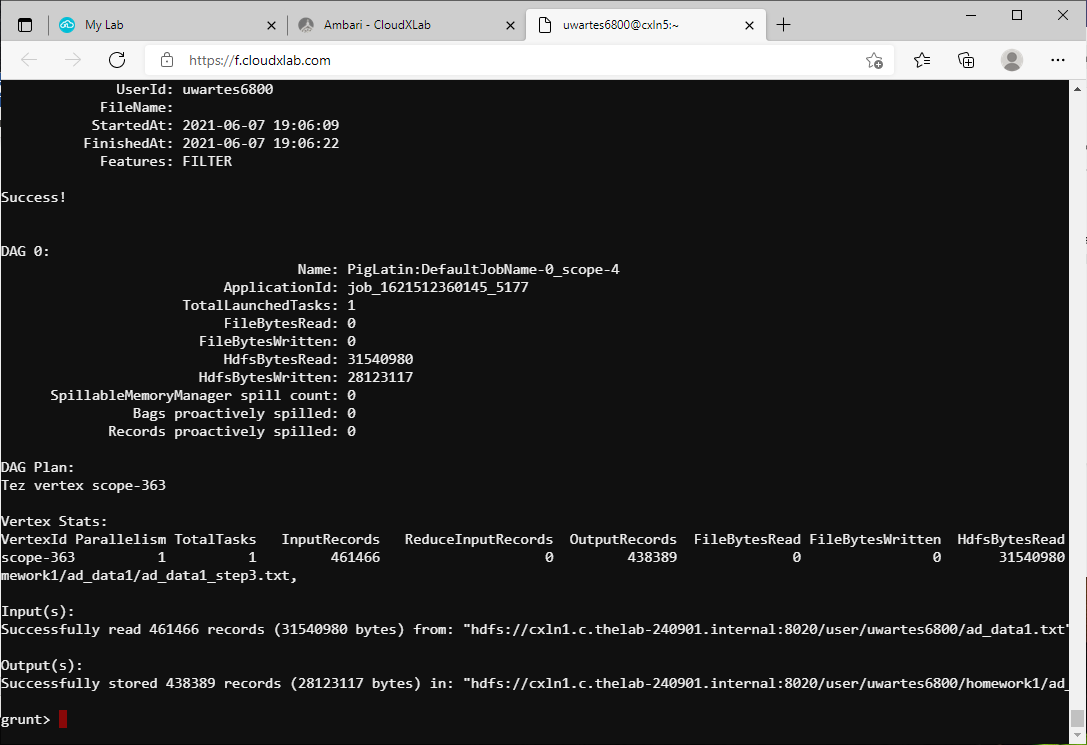
placement: chararray);

B = FILTER A by (country == 'USA');

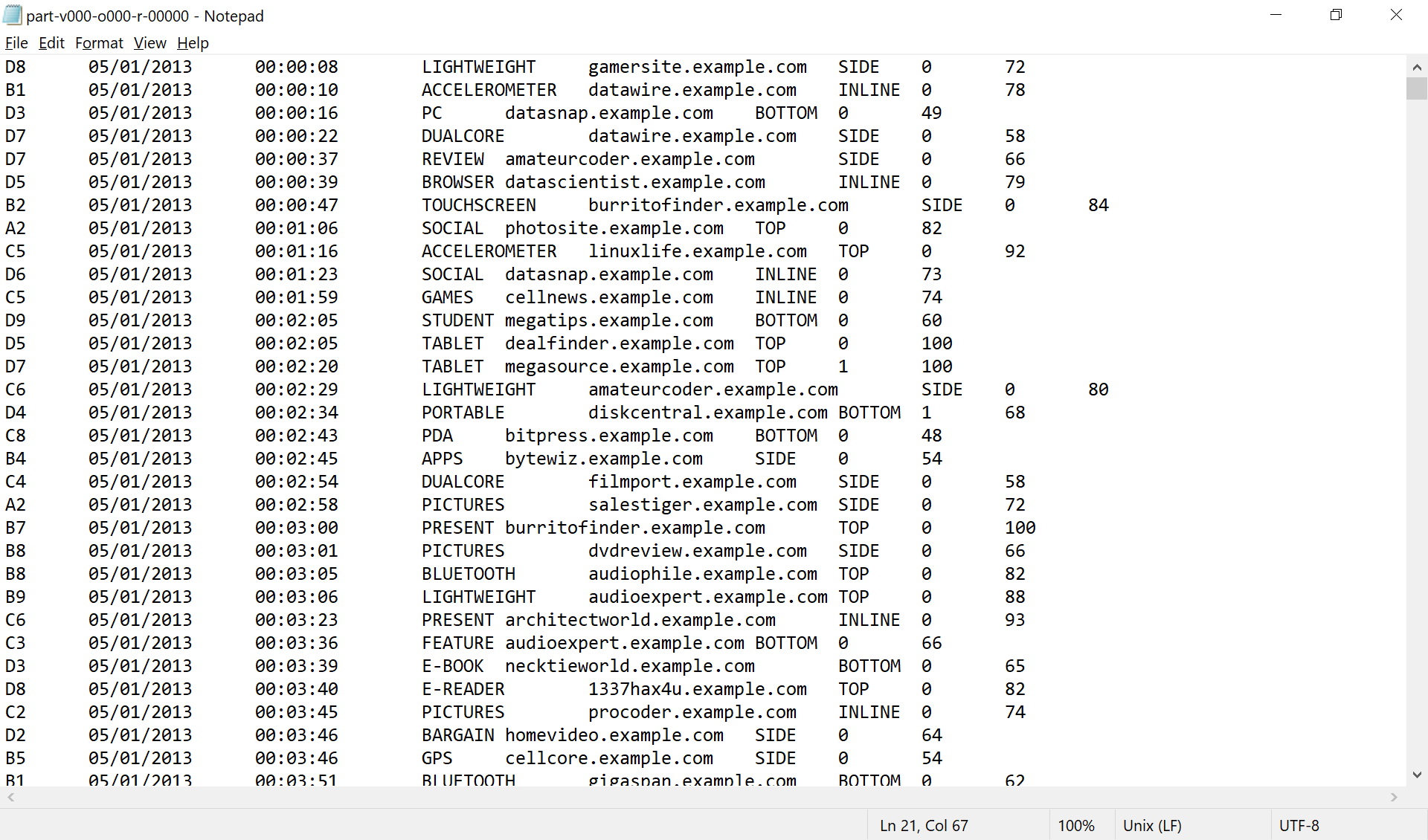
C = FOREACH B GENERATE campaign\_id, date, time, TRIM(UPPER(keyword)), display\_site, placement, was\_clicked, cpc;

STORE C INTO 'homework1/ad\_data1/ad\_data1\_step3.txt' USING PigStorage('\t');





1. Check the first 20 output records that your script wrote to HDFS and ensure they look correct
   1. Are the fields in the correct order?
   2. Are all the keywords now in uppercase?



#### **Step 2: Processing Input Data from the Second Ad Network**

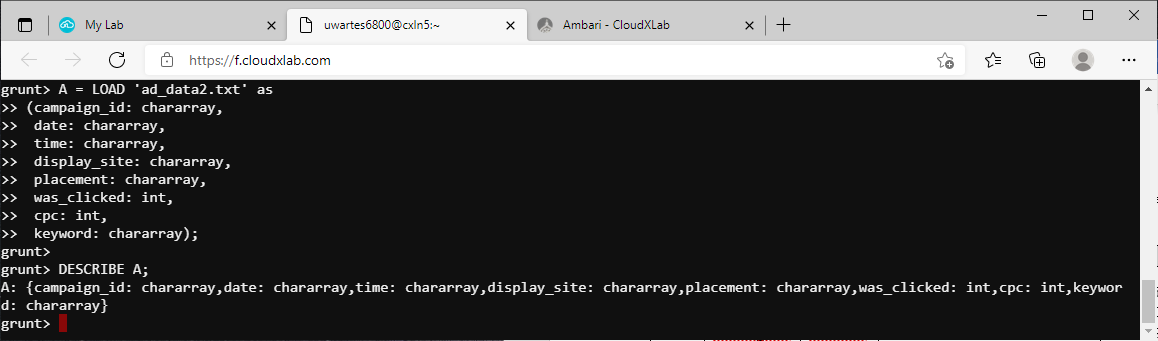
Now that you have successfully processed the data from the first ad network, continue by processing data from the second one.

Capture the script in each step in a word file. Also, capture screen shots to demonstrate that your code did work.

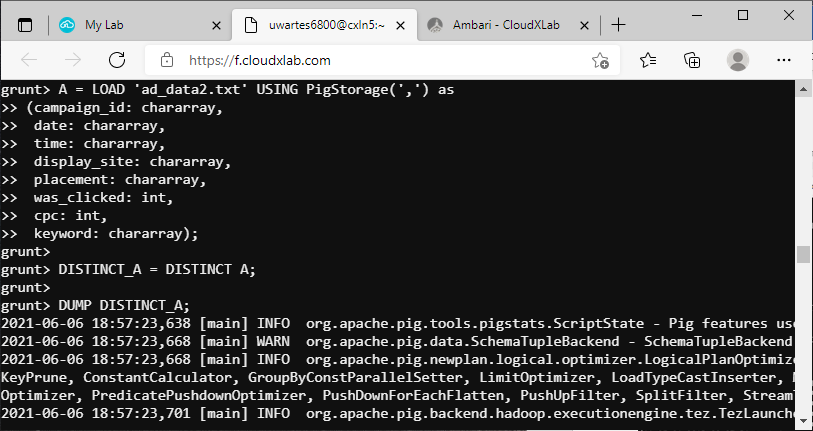
1. Create a script named **etl2** and use the **LOAD** statement and read the data from **ad\_data2.txt** file. The following table shows the order of fields in this file:

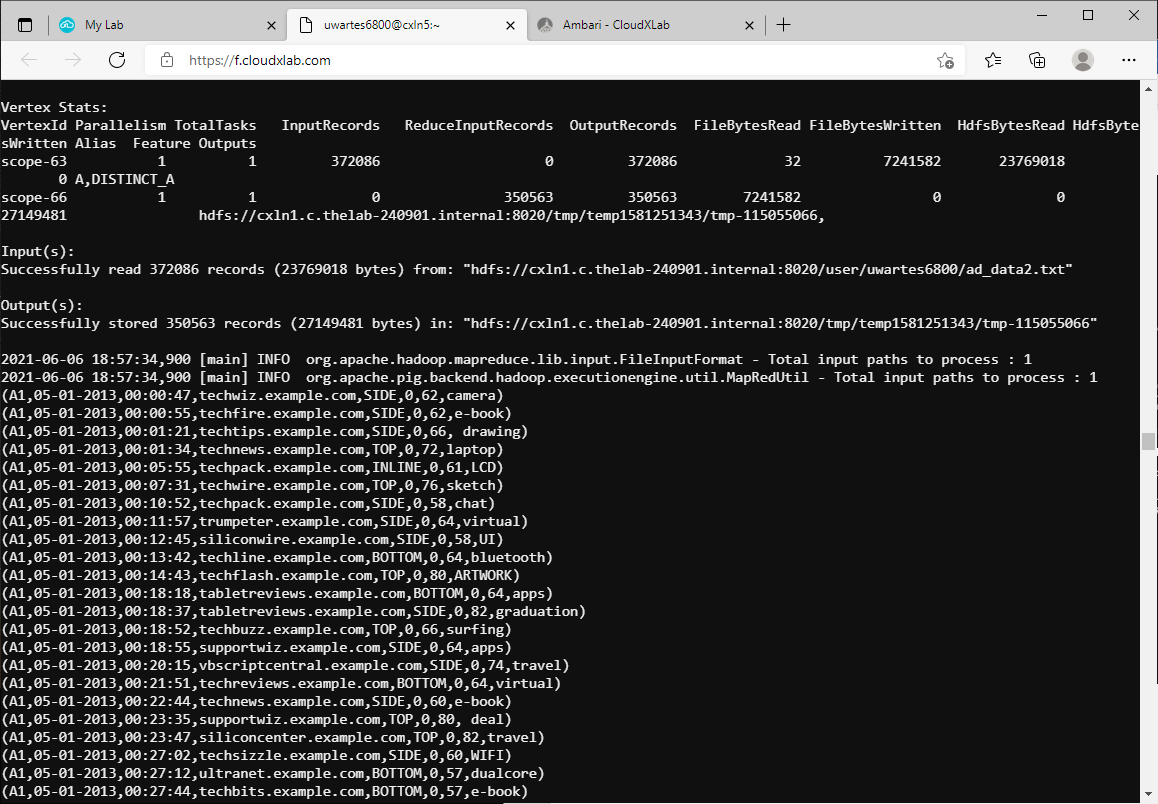
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Index | Field | Data Type | Description | Example |
| 0 | campaign\_id | chararray | Uniquely identifies our ad | A3 |
| 1 | date | chararray | Date of ad display | 05/29/2013 |
| 2 | time | chararray | Time of ad display | 15:49:21 |
| 3 | display\_site | chararray | Domain where ad shown | [www.example.com](http://www.example.com/) |
| 4 | placement | chararray | Where on page was ad displayed | TOP |
| 5 | was\_clicked | int | Whether ad was clicked | Y |
| 6 | cpc | int | Cost per click, in cents | 106 |
| 7 | keyword | chararray | Keyword that triggered ad | tablet |

1. Once you have edited the **LOAD** statement, use the **DESCRIBE** keyword and then run your script to check that the schema matches the table above:



1. Replace **DESCRIBE** with a **DUMP** statement and then make each of the following changes to **etl2**, running this script after each change to verify what you’ve done before you continue with the next step:
   1. This ad network sometimes logs a given record twice. Add a statement to the **etl2** script so that you remove any duplicate records. If you have done this correctly, you should only see one record where the display site field has a value of siliconwire.example.com.



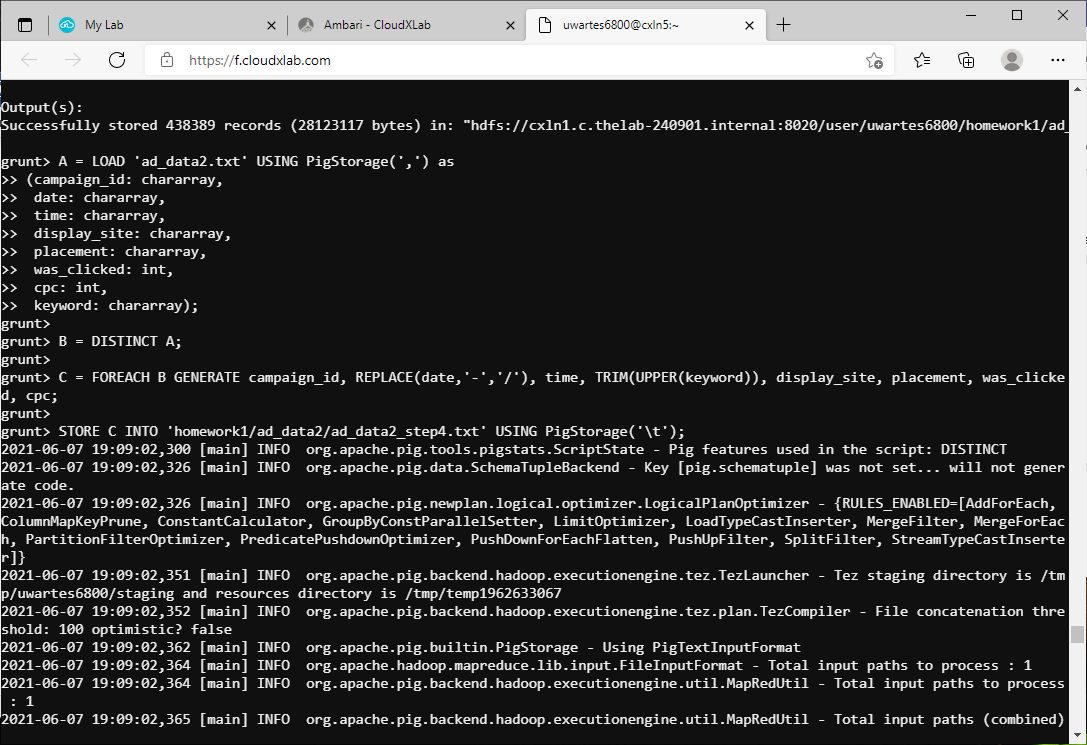


* 1. As before, you need to store the fields in a different order as you received them. Use a **FOREACH … GENERATE** statement to create a new relation containing the fields in the same order you used to write the output from first ad network (shown again in the table below) and also use the **UPPER** and **TRIM** functions to correct the keyword field as you did earlier:

|  |  |  |
| --- | --- | --- |
| Index | Field | Description |
| 0 | campaign\_id | Uniquely identifies our ad |
| 1 | date | Date of ad display |
| 2 | time | Time of ad display |
| 3 | keyword | Keyword that triggered ad |
| 4 | display\_site | Domain where ad shown |
| 5 | placement | Where on page was ad displayed |
| 6 | was\_clicked | Whether ad was clicked |
| 7 | cpc | Cost per click, in cents |

* 1. The date field in this data set is in the format MM-DD-YYYY, while the data you previously wrote is in the format MM/DD/YYYY. Edit the **FOREACH … GENERATE** statement to call the **REPLACE**(date, '-', '/') function to correct this.

1. Replace any **DUMP** statements with a **STORE** statement to write your output as tab-delimited records to the **<home>/ad\_data2** directory.



A = LOAD 'ad\_data1.txt' as

(keyword: chararray,

campaign\_id: chararray,

date: chararray,

time: chararray,

display\_site: chararray,

was\_clicked: int,

cpc: int,

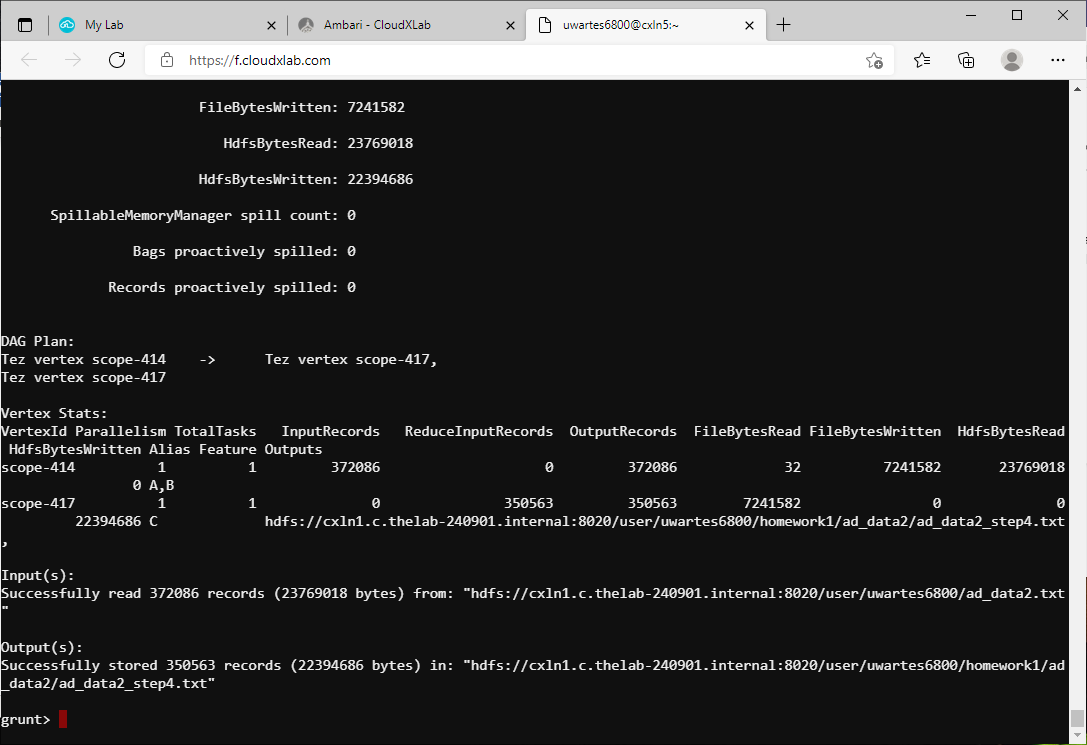
country: chararray,

placement: chararray);

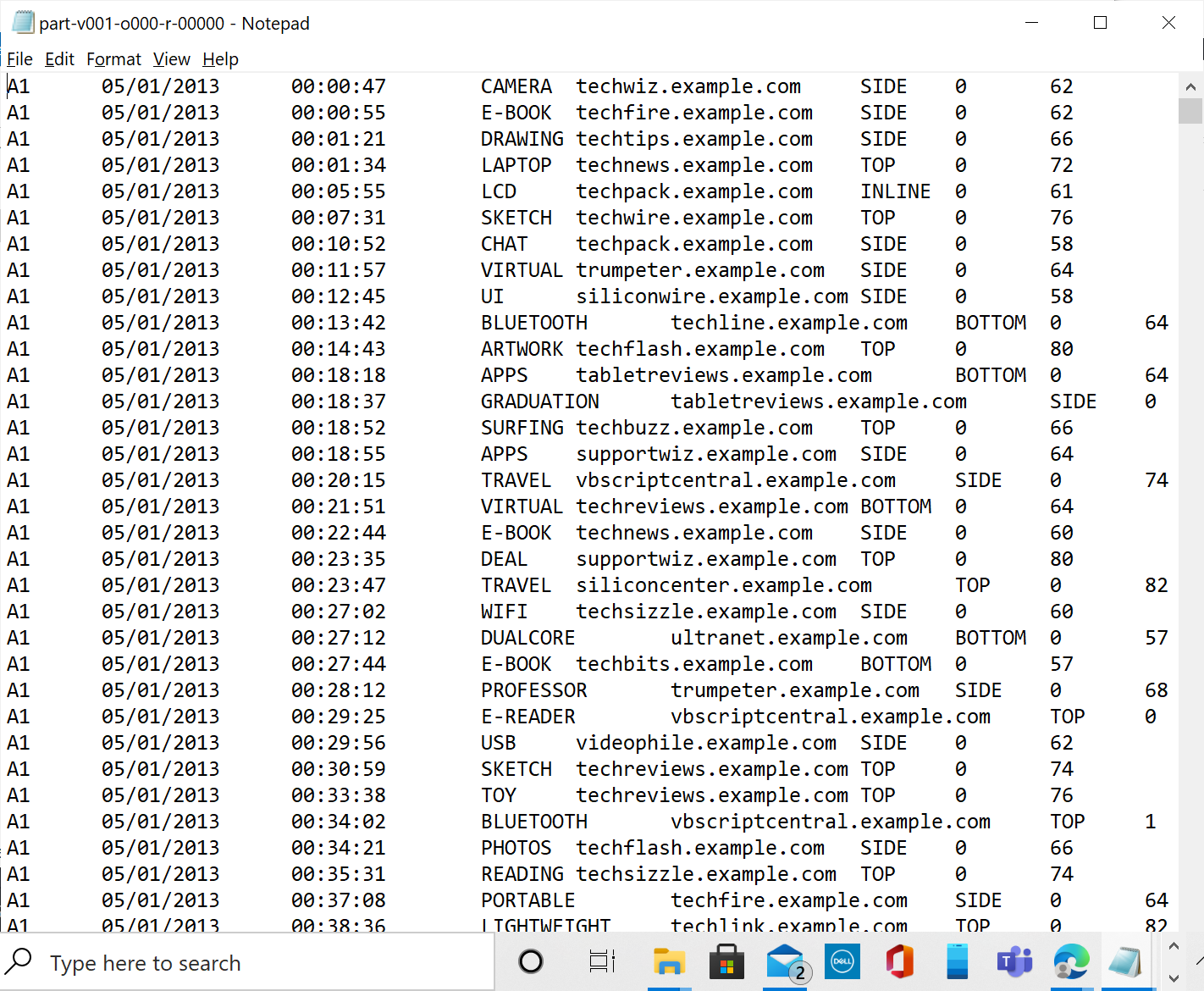
B = FILTER A by (country == 'USA');

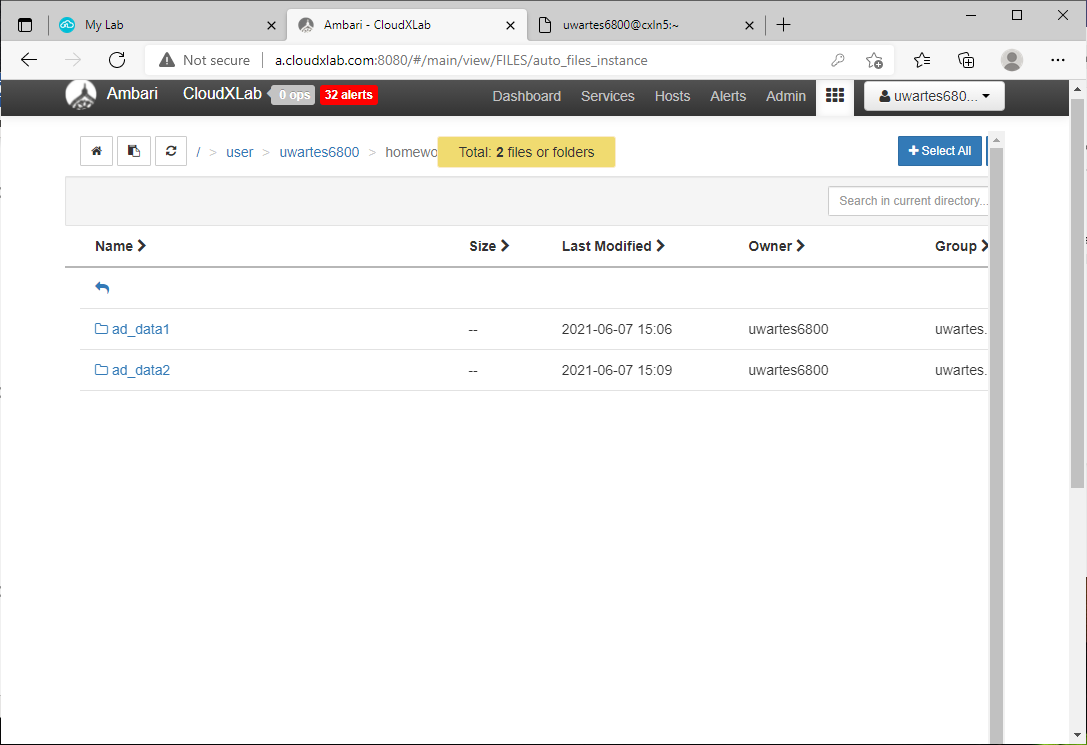
C = FOREACH B GENERATE campaign\_id, date, time, TRIM(UPPER(keyword)), display\_site, placement, was\_clicked, cpc;

STORE C INTO 'homework1/ad\_data1/ad\_data1\_step3.txt' USING PigStorage('\t');



1. Check the first 15 output records written in HDFS by your script:
   1. Do you see any duplicate records?
2. Are the fields in the correct order?
3. Are all the keywords in uppercase?
4. Is the date field in the correct (MM/DD/YYYY) format?





#### **Step 3: Find Low Cost Sites**

Both ad networks charge only when a user clicks on our ad. However, some sites and keywords are more effective than others at attracting people interested in a new tablet that has been advertised. With this in mind, you will begin by identifying which sites have the lowest total cost.

1. Open the **low\_cost\_sites.pig** file (from the PigScripts zip file you can find on canvas), copy into the pig script window, and then make the following changes:
   1. Modify the LOAD statement to read the data from both **ad\_data1** and **ad\_data2** directories in your home path.
   2. Add a line that creates a new relation to include only records where

**was\_clicked** has a value of 1.

* 1. Group this filtered relation by the **display\_site** field.
  2. Create a new relation that includes two fields: the **display\_site** and the total cost of all clicks on that site.
  3. Sort that new relation by **cost** (in ascending order)
  4. Display just the **first three records** to the screen

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

Capture the final script and the output in your word document.

data = LOAD 'homework1' AS (campaign\_id:chararray,

date:chararray, time:chararray,

keyword:chararray, display\_site:chararray,

placement:chararray, was\_clicked:int, cpc:int);

-- B: Includes only records where the ad was clicked

filter\_data = FILTER data by (was\_clicked == 1);

-- C: Groups the data by the 'display\_site' field

group\_data = GROUP filter\_data BY display\_site;

-- D: Creates relation with 'display\_site' and sum of click cost

sum\_clicked = FOREACH group\_data GENERATE group, SUM(filter\_data.cpc) AS sum\_cpc;

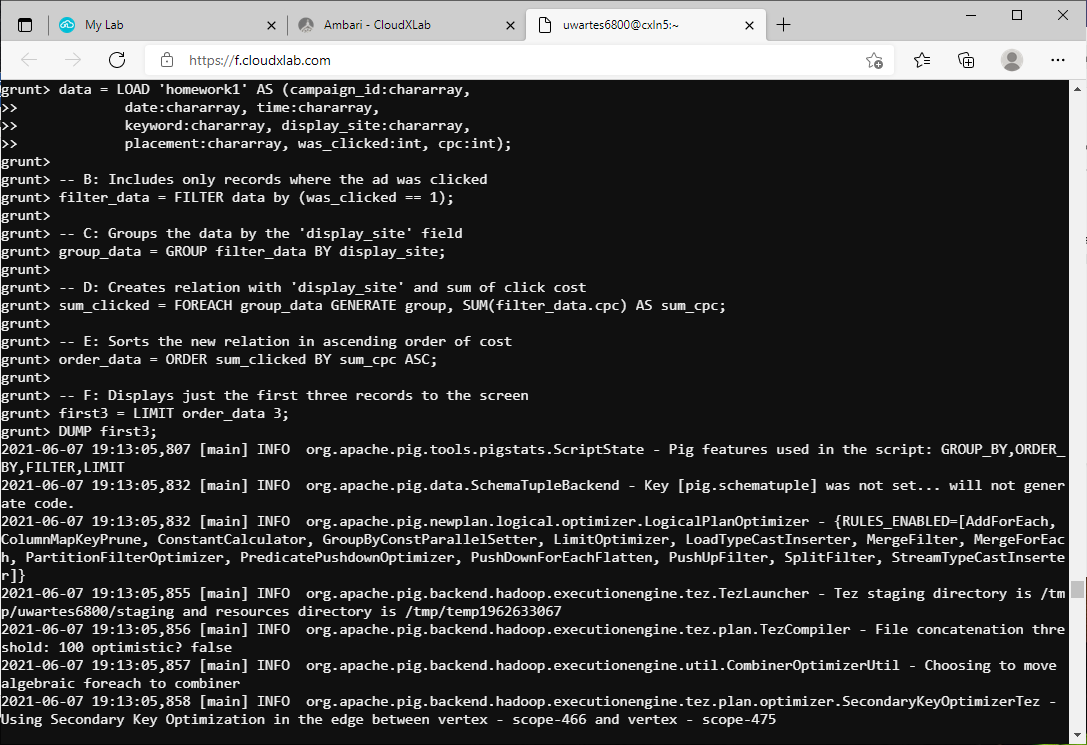
-- E: Sorts the new relation in ascending order of cost

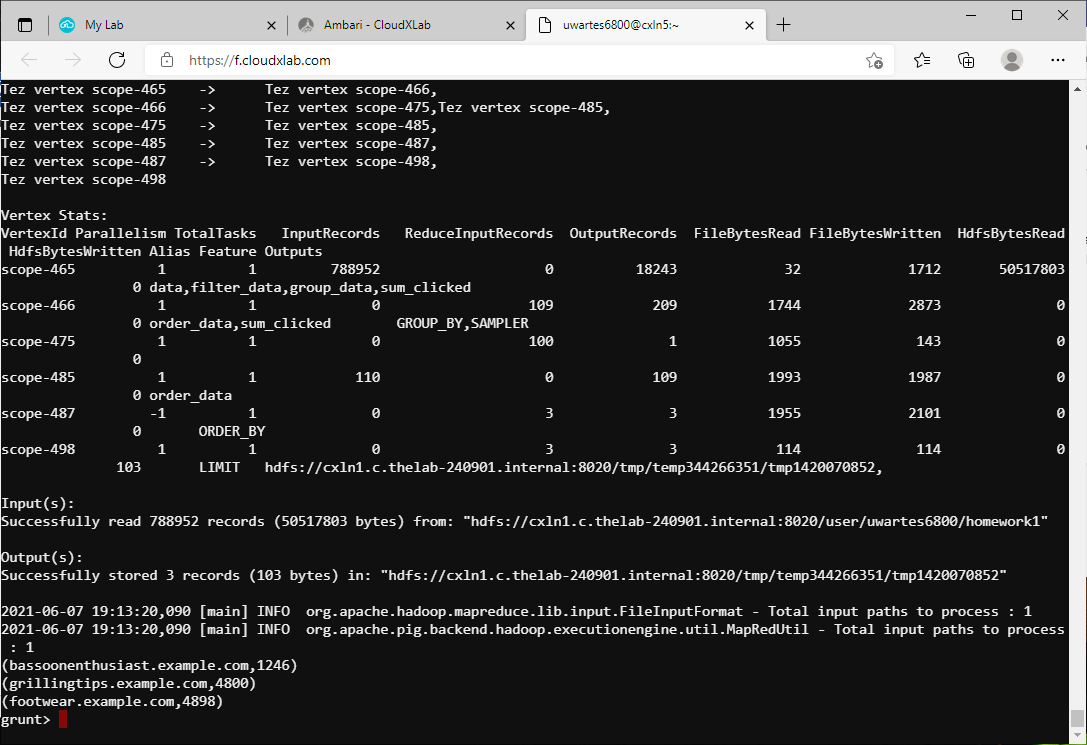
order\_data = ORDER sum\_clicked BY sum\_cpc ASC;

-- F: Displays just the first three records to the screen

first3 = LIMIT order\_data 3;

DUMP first3;





#### **Step 4: Find High Cost Keywords**

The terms users type when doing searches may prompt the site to display an advertisement for a specific company. Since online advertisers compete for the same set of keywords, some of them cost more than others. You will now write some Pig Latin to determine which keywords are the most expensive overall.

1. Since this will be a slight variation on the code you have just written, copy that script into a new script labeled as **high\_cost\_keywords.pig**
2. Edit the **high\_cost\_keywords** script and make the following three changes:
   1. Group by the **keyword** field instead of **display\_site**
   2. Sort in descending order of **cost**
   3. Display the **top five results** to the screen instead of the top three as before
3. Once you have made these changes, try running your script against the data in HDFS.

Capture the final script and the output in your word document.

data = LOAD 'homework1' AS (campaign\_id:chararray,

date:chararray, time:chararray,

keyword:chararray, display\_site:chararray,

placement:chararray, was\_clicked:int, cpc:int);

-- B: Includes only records where the ad was clicked

filter\_data = FILTER data by (was\_clicked == 1);

-- C: Groups the data by the 'keyword' field

group\_data = GROUP filter\_data BY keyword;

-- D: Creates relation with 'keyword' and sum of click cost

sum\_clicked = FOREACH group\_data GENERATE group, SUM(filter\_data.cpc) AS sum\_cpc;

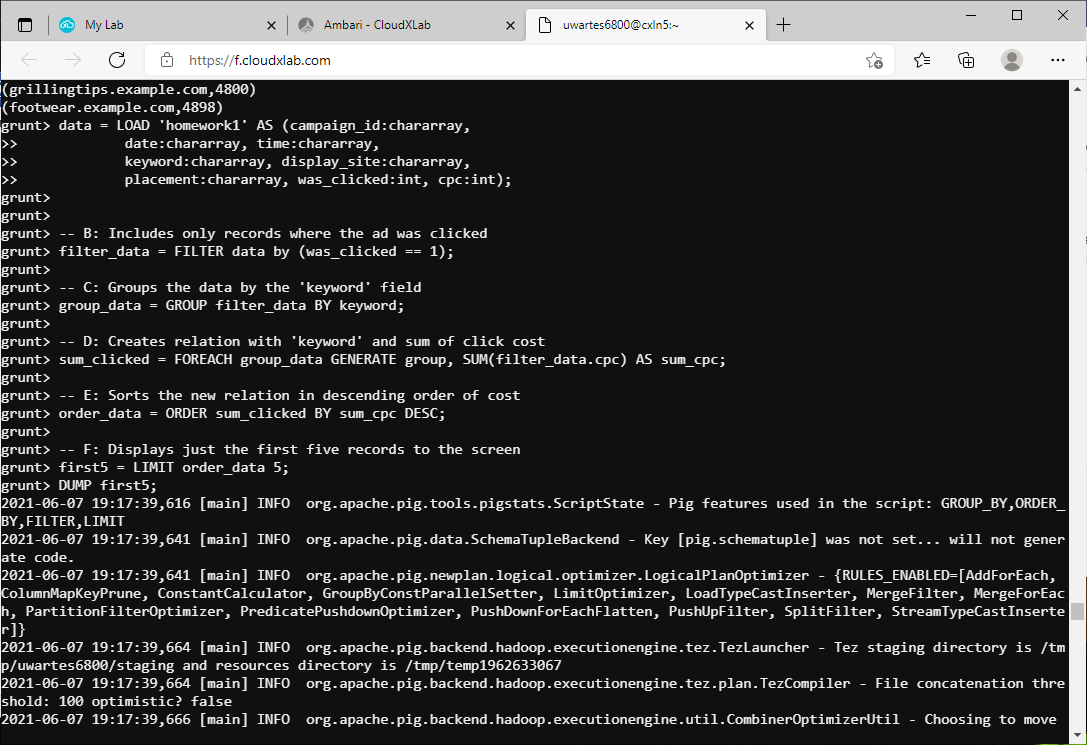
-- E: Sorts the new relation in descending order of cost

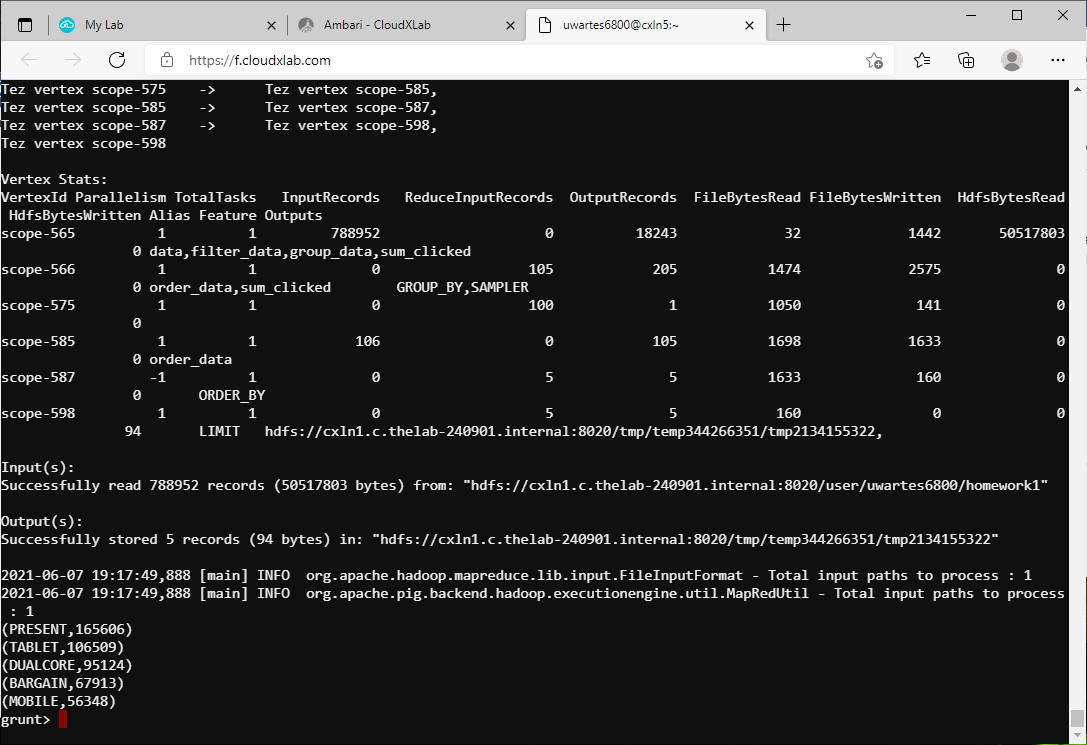
order\_data = ORDER sum\_clicked BY sum\_cpc DESC;

-- F: Displays just the first five records to the screen

first5 = LIMIT order\_data 5;

DUMP first5;





**Etl1.pig**

A = LOAD 'ad\_data1.txt' as

(keyword: chararray,

campaign\_id: chararray,

date: chararray,

time: chararray,

display\_site: chararray,

was\_clicked: int,

cpc: int,

country: chararray,

placement: chararray);

B = FILTER A by (country == 'USA');

C = FOREACH B GENERATE campaign\_id, date, time, TRIM(UPPER(keyword)), display\_site, placement, was\_clicked, cpc;

STORE C INTO 'homework1/ad\_data1/ad\_data1\_step3.txt' USING PigStorage('\t');

**Etl2.pig**

A = LOAD 'ad\_data2.txt' USING PigStorage(',') as

(campaign\_id: chararray,

date: chararray,

time: chararray,

display\_site: chararray,

placement: chararray,

was\_clicked: int,

cpc: int,

keyword: chararray);

B = DISTINCT A;

C = FOREACH B GENERATE campaign\_id, REPLACE(date,'-','/'), time, TRIM(UPPER(keyword)), display\_site, placement, was\_clicked, cpc;

STORE C INTO 'homework1/ad\_data2/ad\_data2\_step4.txt' USING PigStorage('\t');

**High\_cost\_keywords.pig**

-- A: This glob loads only the ad\_data1 and ad\_data2 directories

data = LOAD 'homework1' AS (campaign\_id:chararray,

date:chararray, time:chararray,

keyword:chararray, display\_site:chararray,

placement:chararray, was\_clicked:int, cpc:int);

-- B: Includes only records where the ad was clicked

filter\_data = FILTER data by (was\_clicked == 1);

-- C: Groups the data by the 'keyword' field

group\_data = GROUP filter\_data BY keyword;

-- D: Creates relation with 'keyword' and sum of click cost

sum\_clicked = FOREACH group\_data GENERATE group, SUM(filter\_data.cpc) AS sum\_cpc;

-- E: Sorts the new relation in descending order of cost

order\_data = ORDER sum\_clicked BY sum\_cpc DESC;

-- F: Displays just the first five records to the screen

first5 = LIMIT order\_data 5;

DUMP first5;

**Low\_cost\_sites.pig**

-- A: This glob loads only the ad\_data1 and ad\_data2 directories

data = LOAD 'homework1' AS (campaign\_id:chararray,

date:chararray, time:chararray,

keyword:chararray, display\_site:chararray,

placement:chararray, was\_clicked:int, cpc:int);

-- B: Includes only records where the ad was clicked

filter\_data = FILTER data by (was\_clicked == 1);

-- C: Groups the data by the 'display\_site' field

group\_data = GROUP filter\_data BY display\_site;

-- D: Creates relation with 'display\_site' and sum of click cost

sum\_clicked = FOREACH group\_data GENERATE group, SUM(filter\_data.cpc) AS sum\_cpc;

-- E: Sorts the new relation in ascending order of cost

order\_data = ORDER sum\_clicked BY sum\_cpc ASC;

-- F: Displays just the first three records to the screen

first3 = LIMIT order\_data 3;

DUMP first3;