Advanced Data Management (CMM524)

Laboratory #8: GROUP & Nested FOREACH in Pig Latin

1. Aims

- Using GROUP and nested FOREACH in Pig Latin to perform data analysis.

2. Outcomes

In completing this exercise, you should be able to:
- Group data using the Pig Latin GROUP command.
- Perform aggregation function on a group.
- Use nested FOREACH to apply multiple transformations.

3. The *Dualcore* Domain and Dataset

This lab continues on the *Dualcore* domain from Lab#07.

The company *Dualcore* uses two online advertisement networks to promote its business. The networks provide data on where an advertisement was placed, the date it was placed, the keywords used, whether a user clicked the advertisement, and the cost-per-click they charge *Dualcore*.

3.1. The `ad_data1` Dataset

This labs using the `ad_data1` dataset in Lab#06 section 3. If you have successfully completed the exercise then you should have a dataset "`/dualcore/ad_data1`" in HDFS.

- Check if you have the dataset in "`/dualcore/ad_data1`".
  - The above is an HDFS directory. Inside there should be a `part-m-00000` file which contains the actual data.
  - **If you do, you can jump to section 4.**
  - **If you don't, you need to prepare the dataset. Continue to section 3.2.**

3.2. Option 1: Preparing the ad_data1 Dataset

This section only applies if you do not have the `ad_data1` dataset in HDFS.

Instructions in this section helps you to create the dataset required. If you are too lazy to even do this, jump to section 3.3 to download the dataset.

To prepare the dataset using a script, do the followings:

- In Unix, change into the directory "`~/training_materials/analyst/exercises/pig_etl`".
- Open the Pig script `first_etl.pig` in an editor.
- Modify `first_etl.pig` so that it looks like this:

```
data = LOAD 'ad_data1.txt'
         AS     (keyword:chararray,
                campaign_id:chararray,
                date:chararray,
                time:chararray,
                display_site:chararray,
                was_clicked:int,
                cost_per_click:int,
                country:chararray,
                placement:chararray);

usa_only = FILTER data BY country=='USA';

rearranged = FOREACH usa_only GENERATE campaign_id,
                                       date,
                                       time,
                                       UPPER(TRIM(keyword)) AS keyword:chararray,
                                       display_site,
                                       placement,
                                       was_clicked,
                                       cost_per_click;
STORE rearranged INTO '/dualcore/ad_data1';
```

- This script loads the HDFS file `ad_data1.txt`, processes it, and stores the result into "`/dualcore/ad_data1`" in HDFS.
  - The original data file `ad_data1.txt` is in the Unix directory "`~/training_materials/analyst/data`". Upload this to HDFS before you run the script.
  - If needed, modify the input file path in the Pig script.
  - Run the script in MapReduce mode. It will generate the "`/dualcore/ad_data1`" dataset.

3.3. Option 2: Downloading the Dataset

For those who are too lazy to prepare and run a script, you can download the required data set from Moodle.

- Download the `ad_data1` dataset from Moodle.
- Upload it to the HDFS folder "`/dualcore`". You should have "`/dualcore/ad_data1`" before proceeding.

4. Analysing Ad Campaign Data

4.1. Structure of the `ad_data1` Dataset

Fields of the `ad_data1` dataset are as followed:

Index	Field	Description
0	campaign_id	Uniquely identifies the advertisement
1	date	Date of advertisement displayed
2	time	Time of advertisement displayed
3	keyword	Keyword that triggered the advertisement
4	display_site	Domain where ad was shown
5	placement	Where on page was ad displayed
6	was_clicked	Whether the ad was clicked

7	cost_per_click	Cost-per-click, in cents

4.2. Finding Low Cost Sites

Every time a user clicks an ad of *Dualcore* on a display site, *Dualcore* is being charged a cost by the advertiser. To reduce the cost, *Dualcore* would like to know which display site has the lowest total cost. This requires calculating the sum of all cost-per-click values of a display site.

- In Unix, change to the directory "`~/training_materials/analyst/exercises/analyze_ads`".
- Get a local subset of the `ad_data1` dataset by the following Unix command (**in 1 single line!**). It will create a file `test_ad_data.txt` in your current Unix directory.
  - If you use the result from Lab 06, or prepared the data using a Pig script (Option 1 in Section 3.2), `ad_data1` will be an HDFS folder containing some `part*` files. Use the following command to get your sample file:

```
hadoop fs -cat /dualcore/ad_data1/part* | head -n 100 > test_ad_data.txt
```

  - If you downloaded the dataset (Option 2 in Section 3.3) instead of prepared it, you will get an error in the above command as `ad_data1` will be a file, not a folder in HDFS. In this case, use the following command instead:

```
hadoop fs -cat /dualcore/ad_data1 | head -n 100 > test_ad_data.txt
```

- Edit the `low_cost_sites.pig` script and make the following changes:
  - Modify the `LOAD` statement to read the sample data in the `test_ad_data.txt` file (as created above).
  - Add a line that creates a new relation to include only records where `was_clicked` has a value of `1`. This leaves us only "clicked" entries.
  - Group this filtered relation (created above) by the `display_site` field. This collect all entries of the same display site into a bag.
  - Create a new relation that includes 2 fields: the `display_site` and the total of `cost_per_click` on that site. This calculates the total cost of the display site.
  - Sort the new relation by cost in ascending order.
  - Display just the first 3 records to the screen.
- Try running your script in local mode with the sample data.
- If the local run is correct, modify the `LOAD` statement to use the result in both `/dualcore/ad_data1`
  - Note: Remember that your processed data are in the `part*` files under the HDFS directory but you only need to specify the HDFS directory name. Pig will read the `part*` files inside automatically.
- Run your script in MapReduce mode using the processed datasets.
  - Which 3 sites have the lowest overall cost?

4.3. Finding Highest Cost Keywords

When a user does search, he/she types some keywords. Based on the keyword, a *Dualcore* advertisement may be displayed. Every display of an advertisement has a cost on *Dualcore*. Some keywords cost more than others. *Dualcore* would like to know the keywords which cost them the most (i.e. having the highest total cost-per-click value). This script to find the most costly keyword is similar to the last `low_cost_sites.pig` script.

- In the Unix prompt, copy the last script as `high_cost_keywords.pig` with the following command (Note: **Of course, do this in Unix, not in Grunt!**):

```
cp low_cost_sites.pig high_cost_keywords.pig
```

- Modify the new `high_cost_keywords.pig` script and make the following changes:
  - Keep only records where the ad was clicked.
  - Instead of grouping by `display_site`, group the tuples by `keyword`. This collects entries according to the keyword.
  - For each keyword, calculate the total cost. This should be the sum of the cost-per-click field.
  - Sort in descending order of total cost.
  - Display the top 5 results to the screen.
- Test your script with the sample data `test_ad_data.txt` in the local Unix file system.
- Once you are happy with the sample data, run your script in MapReduce mode on the `ad_data1` datasets.
  - Which 5 keywords have the highest overall cost?

4.4. Counting Total Ad Clicks

One important statistic is the total number of clicks received over all advertisements. This information helps the marketing directors to plan the next ad campaign budget.

- Change to the `bonus_01` subdirectory in Unix.
  - What is the Unix command to use?
- Edit `total_click_count.pig` to implement the followings:
  - Keep only records where the ad was clicked.
  - Group all records together for the aggregation function.
    - Hint: This should be a `GROUP ALL` which does not depend on the value of any field as we are going to count over all tuples.
  - Find out how many clicks the ads have received over the whole data.
    - Hints:
      - We have already filter out all tuples with no click.
      - You can use the `COUNT` function.
  - Display the result to the screen.
- Test your script on the smaller sample data before running it on the bigger dataset in HDFS.

4.5. Finding the Click-Through Rate of Websites

Dualcore advertisements are displayed on different websites. Some websites are more effective in persuading users to click the ad. Some are less effective. It is important for *Dualcore* to know the *click-through rate (*i.e. number of clicked ads / number of ads displayed) of each website.

- Change to the `bonus_03` subdirectory in Unix.
- Open the `lowest_ctr_by_site.pig` file in an editor.
- The script is already grouping records by the `display_site` field. Modify the file to implement the followings:
  - Within the nested FOREACH, filter to get records where an ad was clicked.
  - Using the filtered relation above, find the number of clicks from this display site.
  - Next count the total number of records in this group. This should include all ads, either clicked or not-clicked.
  - Add another line to calculate the *click-through rate* in this group.
  - After the nested FOREACH, sort the records in ascending order of click-through rate.
  - Show only the top 3 to the screen.