CSE427S

HOMEWORK 8

M. Neumann

Due: THU 1 NOV 2018 4PM

Getting Started

Update your SVN repository.

When needed, you will find additional materials for *homework x* in the folder `hwx`. So, for the current assignment the folder is `hw8`.

Hint: You can **check your submission** to the SVN repository by viewing `https://svn.seas.wustl.edu/repositories/<yourwustlkey>/cse427s_fl18` in a web browser.

SUBMISSION INSTRUCTIONS

WRITTEN:

- all written work needs to be submitted electronically in *pdf format*[1] via GRADESCOPE
- provide the following information on *every* page of your `pdf` file:
  - name
  - student ID
  - wustlkey (your `wustlkey` indicates the location (SVN repository) for your code submissions; without this information we will not be able to grade your code!)
- start every problem on a *new page*
- **FOR GROUPS**: make a **group submission** on GRADESCOPE and provide the following information for **all group members** on every page of your `pdf` file: names, student IDs, and location of code submissions (**one** student's `wustlkey`)[2]

CODE:

- code needs to be submitted via SVN repository commit (detailed submission instructions are provided whenever code submissions are required)
- make sure to always use the required *file name(s)* and *submission format(s)*
- **comment your code** to receive maximum credit

[1] Please, **type your solutions** or use **clear hand-writing**. If we cannot read your answer, we cannot give you credit nor will we be able to meet any regrade requests concerning your writing.

[2] It is sufficient to commit code to one SVN repository. If you do not specify the repository for your group's code submission clearly, we will only check the first repository, sorting wustlkeys alphabetically.

Good News

In this homework you will help your new employer – *Dualcore Inc.* – to **save money** and **attract new customers** by writing PIG scripts that analyze data from two online ad networks to optimize advertising. If you haven't heard about your new job, find the instructions for **Lab 7: PIG for ETL** on the course webpage and catch up! You will need the data prepared in the lab for the rest of this homework.

Preparation

Complete **Lab 7: PIG for ETL**.

Problem 1: Find Low Cost Sites (25%)

Both ad networks charge a fee only when a user clicks on a *Dualcore* ad. This is ideal for *Dualcore* since their goal is to bring new customers to their site. However, some sites and keywords are more effective than others at attracting people interested in the new tablet being advertised by *Dualcore*. With this in mind, you will begin by identifying which sites have the lowest total cost. The directory for this and the following problem is:

```
~/training_materials/analyst/exercises/analyze_ads
```

(a) Describe 2 different ways – one using pig and one not using pig – to generate sample datasets. It is way faster to test PIG scripts by using a **local subset** of the input data. Describe why this is the case.

(b) Obtain a local subset of the ad data stored in the `dualcore` folder in HDFS. This test data should comprise the first 100 lines of all parts of `ad_data1`. Store this data in `test_ad_data.txt`. Include your command in your written answer. (HINT: this can be best achieved with UNIX commands not using PIG .)

Note: make it a habit and generate the sample datasets for each problem and use those to test your programs during development.

(c) Edit the PIG script `low_cost_sites.pig` to perform the following operations:

- Modify the LOAD statement to read the sample data generated in (b).
- Create a new relation to include only records where `was_clicked` has a value of 1.
- Group this filtered relation by the `display_site` field.
- Create a new relation that includes two fields: the `display_site` and the total cost of all clicks on that site.
- Sort that new relation by cost (in ascending order).
- Display just the first four records to the screen.

Test your script locally against the sample data `test_ad_data.txt`. What gets displayed on the screen?

(d) Run your script against the full data in HDFS. To achieve this comment out (don't delete) the `LOAD` statement from the previous part and add a new `LOAD` statement using the path with a *file glob* loading both ad data sets (`ad_data1` and `ad_data2`) simultaneously. Which four sites have the lowest overall cost?

Submit your answers to (c) and (d) by adding the `low_cost_sites.pig` to the `hw8` folder in your SVN repository. **To add the .pig file to your SVN repo before committing run:**

```
$ svn add low_cost_sites.pig
$ svn commit -m 'hw8 problem 1 submission' .
```

Problem 2: Find High Cost Keywords (10%)

The terms users type when doing searches may prompt the site to display a *Dualcore* advertisement. 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 for *Dualcore* in general. Note that the price of the adds is based on the keywords regradless whether the add gets actually clicked or not.

(a) Write a PIG script called `high_cost_keywords.pig` that groups the ad data by `keyword` and finds the total cost per keyword. Then sort it by the total cost.

(b) Which three keywords have the highest overall cost? Modify `high_cost_keywords.pig` to display these top-3 results on the screen.

Submit your answers to (a) and (b) by adding the `high_cost_keywords.pig` to the `hw8` folder in your SVN repository. **To add the .pig file to your SVN repo before committing run:**

```
$ svn add high_cost_keywords.pig
$ svn commit -m 'hw8 problem 2 submission' .
```

Problem 3: Calculate Click-Through Rate (15%)

So far, we got a rough idea about the success of the ad campaign at *Dualcore*, but this analysis didn't account for the fact that some sites display *Dualcore's* ads more than others. This makes it difficult to determine how effective their ads were by simply counting the number of clicks on one site and comparing it to the number of clicks on another site. One metric that would allow Dualcore to better make such comparisons is the Click-Through Rate (`https://en.wikipedia.org/wiki/Click-through_rate`), commonly abbreviated as CTR. This value is simply the percentage of ads shown that users actually clicked, and can be calculated by dividing the number of clicks by the total number of ads shown. The directory for this problem is:

```
~/training_materials/analyst/exercises/analyze_ads/bonus_03
```

(a) Edit the `lowest_ctr_by_site.pig` file and implement the following:

- Within a nested `FOREACH`, filter the records to include only records where the ad was clicked.

- Create a new relation on the line that follows the `FILTER` statement which counts the number of records within the current group.
- Add another line below that to calculate the CTR in a new field named `ctr`.
- After the nested `FOREACH`, sort the records in ascending order of CTR and display the first four to the screen.

(b) Once you have made these changes, try running your script against the data in HDFS. Which four sites have the lowest click through rate?

Submit your answers to (a) by adding the `lowest_ctr_by_site.pig` to the `hwM` folder in your SVN repository. **To add the .pig file to your SVN repo and commit your work run:**

```
$ svn add lowest_ctr_by_site.pig
$ svn commit -m 'hw8 problem 3 submission' .
```

More Good News

Another hypothetical company needs your help! Loudacre Mobile is a (fictional) fast-growing wireless carrier that provides mobile service to customers throughout western USA. Loudacre just hired you as a data analyst. Congrats and let's get to work! Your first task is to migrate their existing infrastructure to Hadoop and perform some ETL processing. The size and velocity of their data has exceeded their ability to process and analyze their data without a cloud solution. Loudacre has several data sources:

- MySQL database – customer account data (name, address, phone numbers, devices)
- Apache web server logs from Customer Service site, such as HTML files and Knowledge base articles
- XML files such as device activation records
- Real-time device status logs
- Base stations (cell tower locations)

Problem 4: ETL with SPARK (40%)

In this problem you will parse a set of activation records in XML format to extract the account numbers and model names.
SPARK is commonly used for ETL (Extract/Transform/Load) operations. Sometimes data is stored in line-oriented records, like the webserver logs we analyzed previously, but sometimes the data is in a multi-line format that must be processed as a *whole file*. In this exercise you will practice working with **file-based input format** instead of line-based input formats and use **named functions**.

Preparation:

- Understand how to use **file-based** input formats and **named functions** in SPARK → the SP1 lecture slides should be helpful!
- There are three **named functions** provided in `ActivationModels.pyspark/scalaspark` in the `hw8` folder in your SVN repository.

- `getactivations` takes an XML string, parses it, and returns a collection of activation record objects.
- `getmodel` takes an individual activation record object and finds the device model.
- `getaccount` takes an individual activation record object and finds the account number.

To use those copy and paste them into the spark shell.

- The data for this problem can be found in your local file system under:

```
~/training_materials/dev1/data/activations
```

Review the `activations` data. We will be interested in the following information: account-number and model. Each XML file contains data for all the devices activated by customers during a specific month. Put the data into the `loudacre` folder in HDFS .

- Consult the API documentation for RDD operations on the Spark API page you bookmarked in **Lab 8**. **Review the list of available methods of the RDD class.**

Go back to these steps whenever you get stuck!

Use the **spark shell** for this problem and include all required commands in your written answers.

(a) Create an RDD from the activations dataset, where the entire content of each XML file is a single RDD element. What information is stored in the first value, what in the second value of the tuples in the resulting RDD?

(b) Which SPARK operation can be used to map the information in the second value to separate RDD elements? Create a new RDD of separate activation records. You can use the named function `getactivations` in `ActivationModels.pyspark/scalaspark`.

(c) Extract the account number and device model for each activation record, and save the list to a text file formatted as `account_number:model`. Store the file in `/loudacre/account-models` in HDFS. You can use the provided `getaccount` and `getmodel` functions described above to find the account number and the device model.

Include **all required pyspark or Scala commands** in your written answer in `hw8.pdf`.

Problem 5: SPARK Job Execution (10%)

(a) Describe what *pipelining* means in the context of a SPARK job execution. What is its benefit?

(b) Give an example of two operations that can be pipelined together.

(c) Give an example of two operations that cannot be pipelined together.

Bonus Problem (5% up to a max. of 100%) - no group work!

Write a review for this homework and store it in the file `hw8_review.txt` provided in your SVN repository (and commit your changes). This file should only include the review, **no other information** such as name, wustlkey, etc. Remember that you are not graded for the content of your review, solely it's completion.

Provide the star rating for `hw8` via this link: `https://wustl.az1.qualtrics.com/jfe/form/SV_aaAc0ILmBleyKzj`.
You can only earn bonus points if you write a meaningful review of **at least 50 words** and provide the corresponding star rating. Bonus points are given to the **owner of the repository only** (no group work!).

Submit your review in the file **`hw8_review.txt`** provided in the `hw8` folder in your SVN repository. **To commit the file run:**

```
$ svn commit -m 'hw8 review submission' .
```

Take 1 minute to provide a star rating for this homework – your star rating should match