DBWP Team3 Final Report

16102269 Kim Jong Gyu
16102268 Kim Yeong Hyeon
18102073 Kim Genie
18102076 Park Subin

1. This is our team's final report for the term project. We developed the web dashboard with the 'App store' information. We made this web page and graphs to provide useful data for developer who are struggling for developing application. Each paragraph of the report explains each slide of the PPT.

2. The contents are followed:

    1. Data – How to get data & data files

    2. Implementation – Description & result

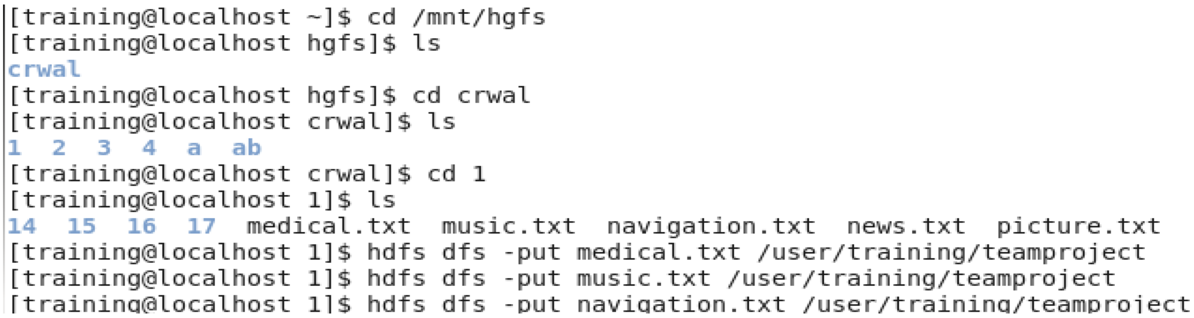
    3. Debugging – Problem & Solution

    4. Work distribution – How we distributed the work

    5. Environment setting

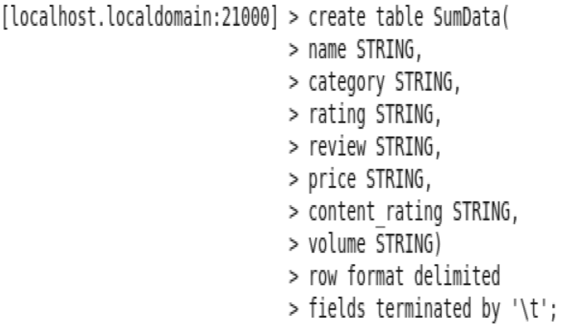
3. We crawled data from the 'App store' web page (https://apps.apple.com/kr/genre/ios/id36). We used python and 'BeautifulSoup' to crawl the data. We submitted the crawling code(app_store_crawl_submit.py) with the report.

4. Data has six columns: name, category, rating, review, price and content rating. Rating is a score rated by the users from zero to five. Review is the number of reviews for the application and it's Int type. Price is 'free' or 'paid', and content rating is the age limitation of the application. Any other columns without rating and review are all String type.
5. In order to use hadoop, we made sharing directory that share file with vmware and local computer. And in vmware, we can get the file in '/mnt/hgfs/'. Using 'hdfs dfs -put' upload the file to hdfs.

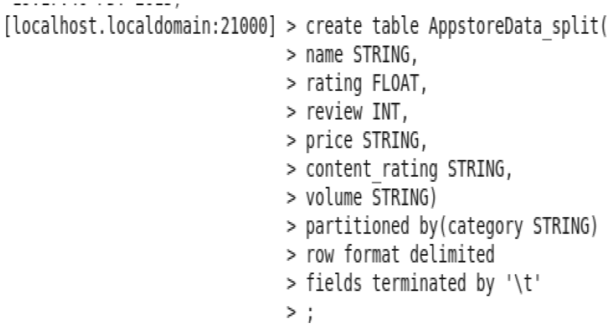


```
[training@localhost ~]$ cd /mnt/hgfs
[training@localhost hgfs]$ ls
crwal
[training@localhost hgfs]$ cd crwal
[training@localhost crwal]$ ls
1  2  3  4  a  ab
[training@localhost crwal]$ cd 1
[training@localhost 1]$ ls
14  15  16  17  medical.txt  music.txt  navigation.txt  news.txt  picture.txt
[training@localhost 1]$ hdfs dfs -put medical.txt /user/training/teamproject
[training@localhost 1]$ hdfs dfs -put music.txt /user/training/teamproject
[training@localhost 1]$ hdfs dfs -put navigation.txt /user/training/teamproject
```

And using query, we create two table. One is that type of all columns are ‘STRING’ because when we crawled, we get all data as string. And some app name has ‘,’ so we set row format delimited fields terminated by ‘/t’.

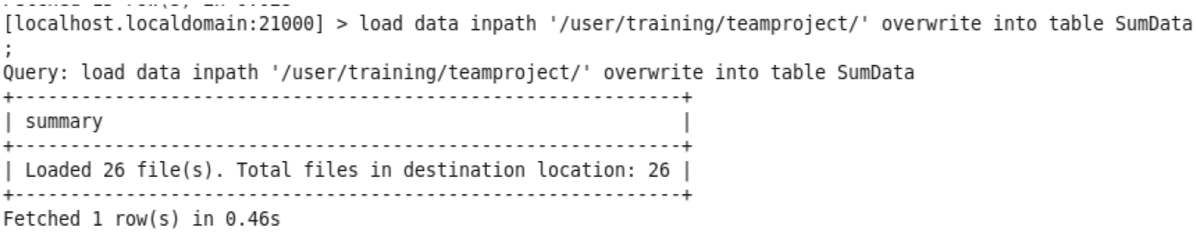


```
[localhost.localdomain:21000] > create table SumData(
                              > name STRING,
                              > category STRING,
                              > rating STRING,
                              > review STRING,
                              > price STRING,
                              > content_rating STRING,
                              > volume STRING)
                              > row format delimited
                              > fields terminated by '\t';
```



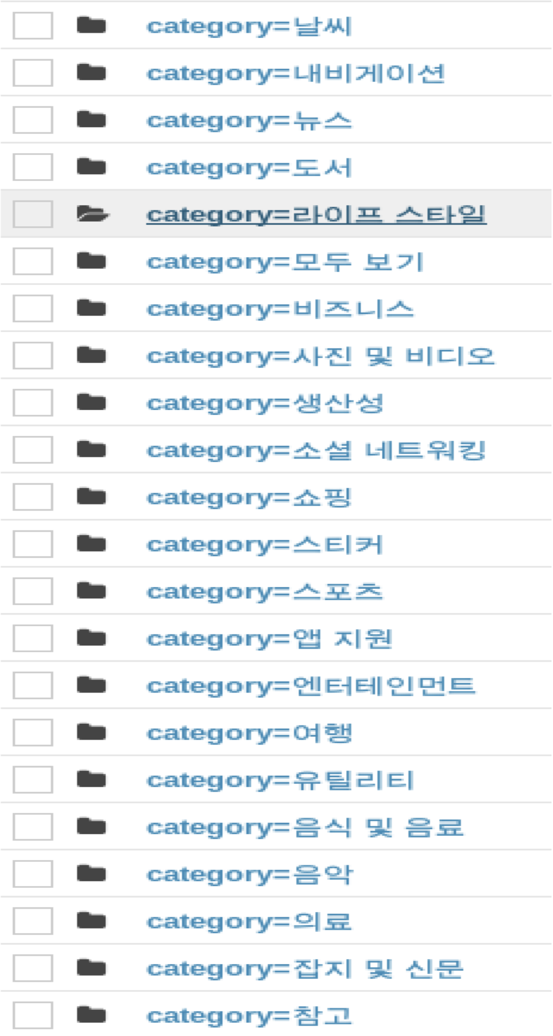
```
[localhost.localdomain:21000] > create table AppstoreData_split(
                              > name STRING,
                              > rating FLOAT,
                              > review INT,
                              > price STRING,
                              > content_rating STRING,
                              > volume STRING)
                              > partitioned by(category STRING)
                              > row format delimited
                              > fields terminated by '\t'
                              > ;
```

In the other table, we add a partition as category for partitioning and set the type of columns as we want. Impala can’t update the column values, so When we insert the data, we used query to change the type of data. And in price column, we compare what the value is ‘free’ or ‘pay’. So we changed the value of price column as if the length of value is greater than 6, change the value as ‘paid’. Because in korean word has 3 length in each words. Ans we get the free data as ‘무료’.



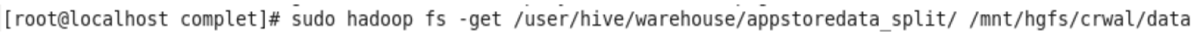
```
[localhost.localdomain:21000] > load data inpath '/user/training/teamproject/' overwrite into table SumData
;
Query: load data inpath '/user/training/teamproject/' overwrite into table SumData
+------------------------------------------------------------+
| summary                                                    |
+------------------------------------------------------------+
| Loaded 26 file(s). Total files in destination location: 26 |
+------------------------------------------------------------+
Fetched 1 row(s) in 0.46s
```

After inserting data into table, we can get the data partitioned for each category.



□ category=날씨
□ category=내비게이션
□ category=뉴스
□ category=도서
□ category=라이프 스타일
□ category=모두 보기
□ category=비즈니스
□ category=사진 및 비디오
□ category=생산성
□ category=소셜 네트워킹
□ category=쇼핑
□ category=스티커
□ category=스포츠
□ category=앱 지원
□ category=엔터테인먼트
□ category=여행
□ category=유틸리티
□ category=음식 및 음료
□ category=음악
□ category=의료
□ category=잡지 및 신문
□ category=참고

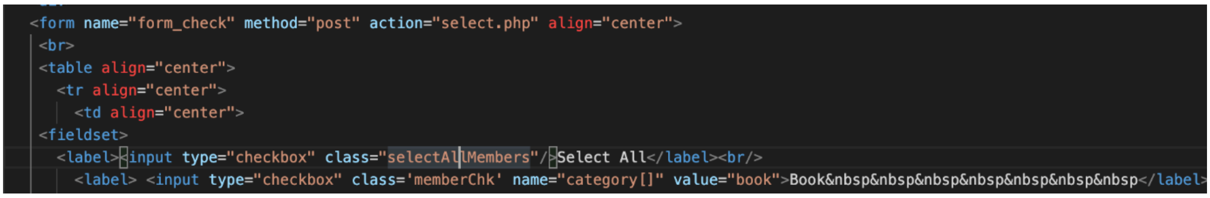
And in txt file, deleted the category value. We can reduce the size of the data file. After that, using 'hadoop fs -get' we can put the partitioned data into local computer.



```
[root@localhost complet]# sudo hadoop fs -get /user/hive/warehouse/appstoredata_split/ /mnt/hgfs/crwal/data
```

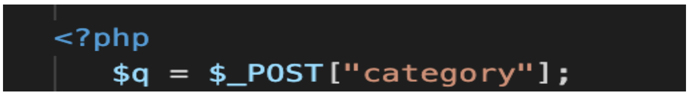
When we crawled the data, we get the same data twice, so distinct function is needed.

6. we make the select option. Using this select category option, user can see the 4 chart according to selecting category. This option is made by link together between html and php. Looking the under the php code,



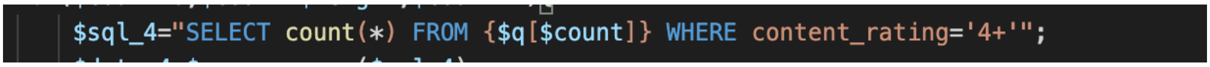
```
<form name="form_check" method="post" action="select.php" align="center">
 <br>
 <table align="center">
   <tr align="center">
     <td align="center">
 <fieldset>
   <label><input type="checkbox" class="selectAllMembers"/>Select All</label><br/>
     <label> <input type="checkbox" class='memberChk' name="category[]" value="book">Book&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp</label
```

We use form type and action for sending selecting category data to php file. And action link is own file name for we show 4 chart in one page. So as user to be able multi selecting, we make checkbox and variable is same to the partioning table name.



```
<?php
    $q = $_POST["category"];
```

This code is to receive the selecting option in php and this structure is array. So using this code we can make 4 different charts according to user selecting.

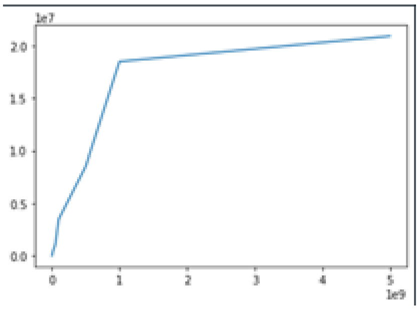


```
    $sql_4="SELECT count(*) FROM {$q[$count]} WHERE content_rating='4+'";
```

Once, Because we make table portioning, we should make variable for selected category about table name. Looking above code from next variable, this variable is to function about selected table appropriately selected category.

7. This is the result web page.

*) Prerequisites



Once there is a prerequisite that review treat to similarly the number of downloads. Before changing data from play store application to apple app store application, we can calculate to graph relationship the number of reviews and the number of download user. So I wanted to get a relationship the number of downloads and reviews in Apple Store, but I couldn't get the number of downloads from Apple Store because I couldn't crawl, so we determine this factor to prerequisites.

1) Average rating and the Number of Reviews

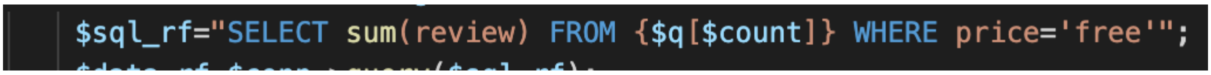
For the line graph, we got average value of 'rating' and sum value of 'review' from the each category. At this step, we used php to connect mysql. To get meaningful average value of rating, we had to excluded unrated application which's rating is zero. Then, added each value to the php array, and at javascript part, we used CavnasJS to draw the graph. For the dataPoints, used php echo to get php array to javascript. By doing this, we were able to draw the graph on the web by query result. Not to display empty category for the condition on the web, we add 'if' statement before appending to php array. This line graph is 'Average Rating and the Number of Reviews'. The pink line and left y-axis are for average rating of the category, and the blue line and right y-axis are for the total number of reviews. This graph shows that relation of rating and reviews. We can learn that many reviews do not always mean high rating.

2) Percentage of Content Rating

For the stack chart, we made 4 array cause of content rating separate by 4 section such as 4+,9+,12+,17+. We determine 4 count sql query for each 4 factors. And then push 4 array corresponding category. So send these 4 arrays to Script and show stack chart by standard 100 percent about the number of reviews for paid and free application. The reason that why use stack chart instead pie chart is that we make selecting options so we show multi stack bars corresponding content rating proportion according to option.

Using this chart, Prospective developers or people who are interested in apps know the trend of what age groups are popular by category. So when they develop the application such as shopping app , they can set the development direction with focusing target primary customer layer.

3) The Number of Reviews for Paid and Free App



```
$sql_rf="SELECT sum(review) FROM {$q[$count]} WHERE price='free'";
```

For the Number of Reviews for Paid and Free App bar chart, we made 2 arrays for integrating. First looking above the code, we use sum function for knowing the free application reviews. And similarly this code, we know the paid application reviews. Push this data corresponding selecting category into pre-made 2 arrays. And then using CanvasJS show the bar chart. Reason that show this bar charts is that developer know that relationship between the number of download users according to the price of the application. It can be seen that the price is of great importance. That means you might think that the free app is right for the current trend.

4) The Number of Paid and Free App

For the bar graph, we made two array for collecting the number of price about each category whether is free or paid. At this step, we use query statement with count function and where option and then calculate the number of free and the number of paid corresponding to category. And this data push the two array. And then these two arrays send to javascript and show the bar chat using these array.

Through this chart, preparatory developer can find out where there are many free or paid apps for each category and if the developer make application about specific category, they can select free or paid application knowing the trend by this graph.

8. When we made crawling code, we think that the more data is better. Thus, we get the volume of data. Although we get the size of app data as separated by ‘KB’,’MB’,’GB’, we can’t find some relationship between data attribute that we get. So, we decided to delete the column of volume and the speed of web page is little bit faster.

9. There was a big difference between the number of free and paid apps, therefore the bar for paid app was too small. Thus, we used quantile function to find appropriate shrink point. We used ‘0.85’ quartile and ‘0.95’ quartile for start and end value of the shrink. After shrinking, it was easy to recognize the difference between paid app of two category.

10. When send made some arrays in php for showing chart sent from php to javascript, we can not check problems in code using debug so I solved this problem by developer tool using webpage console window. We can see the error message about Javascript code and how the variables sent from php have changed in script. But this receipt is not perfect cause of php. We cannot see error about php.

11. Next slide is about our limitation. First limitation is data set changed. The reason why we changed the data set is google play store data were quite small and old. So, we changed the data set in app store. However, app store has some missing information such as the number of app downloaded.
    Next limitation is we couldn’t fully maximize use of Hadoop. We failed to connect the APACHE Hadoop with web. So, instead of using web with Hadoop, we used the mySQL.

12. Next slide is about our work distribution. Yeong Hyeon Kim did a crawling the app information about categories “medical”~”reference”. Also, he used the Hadoop environment to separate all collected data file. Jong Gyu Kim did a crawling the app information about categories “shopping”~”weather”. Then, he made the PHP file about the bar graphs and implemented the checkbox. Subin Park did a crawling the app information about categories “game”~”magazine”. Then, she made the PHP file about the pie chart and line graph. After that, she integrated the al graphs. Genie Kim did a crawling the app information about categories “book”~”finance”. Then made the overall html design and powerpoint file.

13. Next slide is about the environment setting. Firstly, create the database that named “projectDB”. Secondly, create tables for each category like schema. The detail table names are in the #13 slide. Thirdly, Load each text file to the table. Then, check the php file name.

It should be “team3.php”.

14. Using the searching function, move to ‘setting’ part and change the hostname, username, and password as yours. Finally, open ‘hostname/team3.php’ on your browser.
15. We couldn't meet in person because of Corona, but we were able to do the project using with zoom or other communication programs. Thank you for reading.