Assignment 6: Report

Ritvika Reddy, Yang Xia

10/26/2017

Implementation:

The code was implemented on our local machines. All the results are present in the output folder.

RESULTS:

The following outputs are for the entire data.

Top 5 loudest songs:

```
##             Track_ID loudness
## 1 TRDZGER12903CD386D    4.318
## 2 TRXFHGZ12903CD2C1D    4.300
## 3 TRZVIPO12903D01BA4    4.231
## 4 TRONJMK12903CFCCC4    4.166
## 5 TRXDEFB128F426EA6A    4.150
```

Top 5 longest songs:

```
##             Track_ID duration
## 1 TRDZTTO12903CF1A2E 3034.906
## 2 TRVFVTA128F421E809 3033.600
## 3 TRSMLIB128F934C0A8 3033.443
## 4 TRPIWVS128F4289D7F 3032.764
## 5 TRPWIUP128F426B47B 3032.581
```

Top 5 hottest songs:

```
##             Track_ID song_hotttness
## 1 TRAALAH128E078234A              1
## 2 TRALLSG128F425A685              1
## 3 TRANKTK128E07921D9              1
## 4 TRAWBHE12903CBC4CB              1
## 5 TRBFNSL128F42776F9              1
```

Top 5 fastest songs:

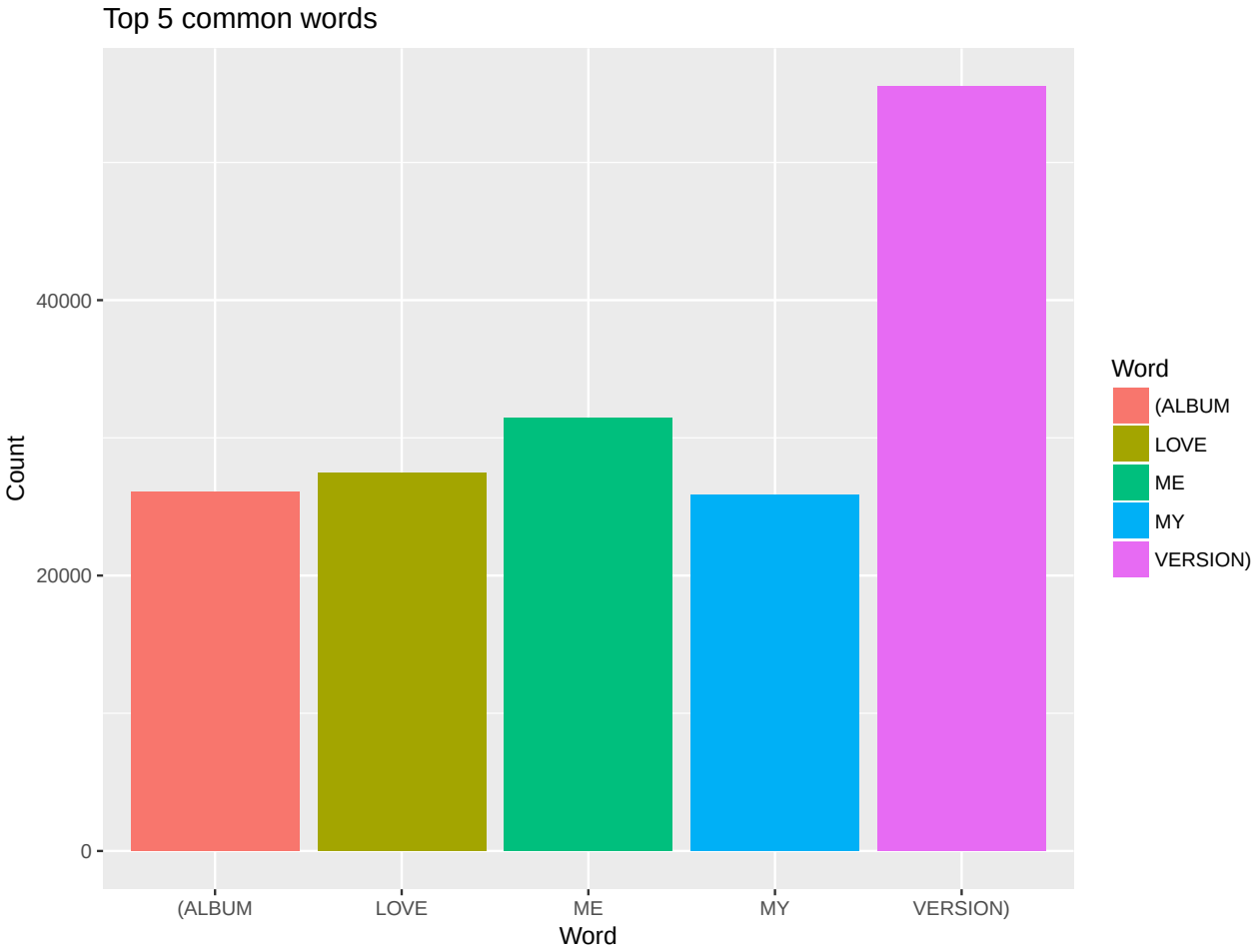
```
##             Track_ID   tempo
## 1 TRPPDKE128F930D9C0 302.300
## 2 TRNPTWJ128F93136D2 296.469
## 3 TRFWRVO128F425C4EF 285.157
## 4 TRBHQUV12903CFAFA9 284.208
## 5 TRLPHPU12903CD8DAA 282.573
```

Top 5 prolific artists:

```
##            Artist_ID total_songs
## 3 AR6681Y1187FB39B02         208
## 1 ARXPPEY1187FB51DF4         204
## 5 ARH861H1187B9B799E         201
## 4 AR8L6W21187B9AD317         196
## 2 ARLHO5Z1187FB4C861         194
```

Loading the common words found for the `entire data`.

The top 5 words are shown in the graph below.



The top 5 common words along with the number of times they occur in the song titles in the data are shown below.

```
print(commonWords[1:5,])
```

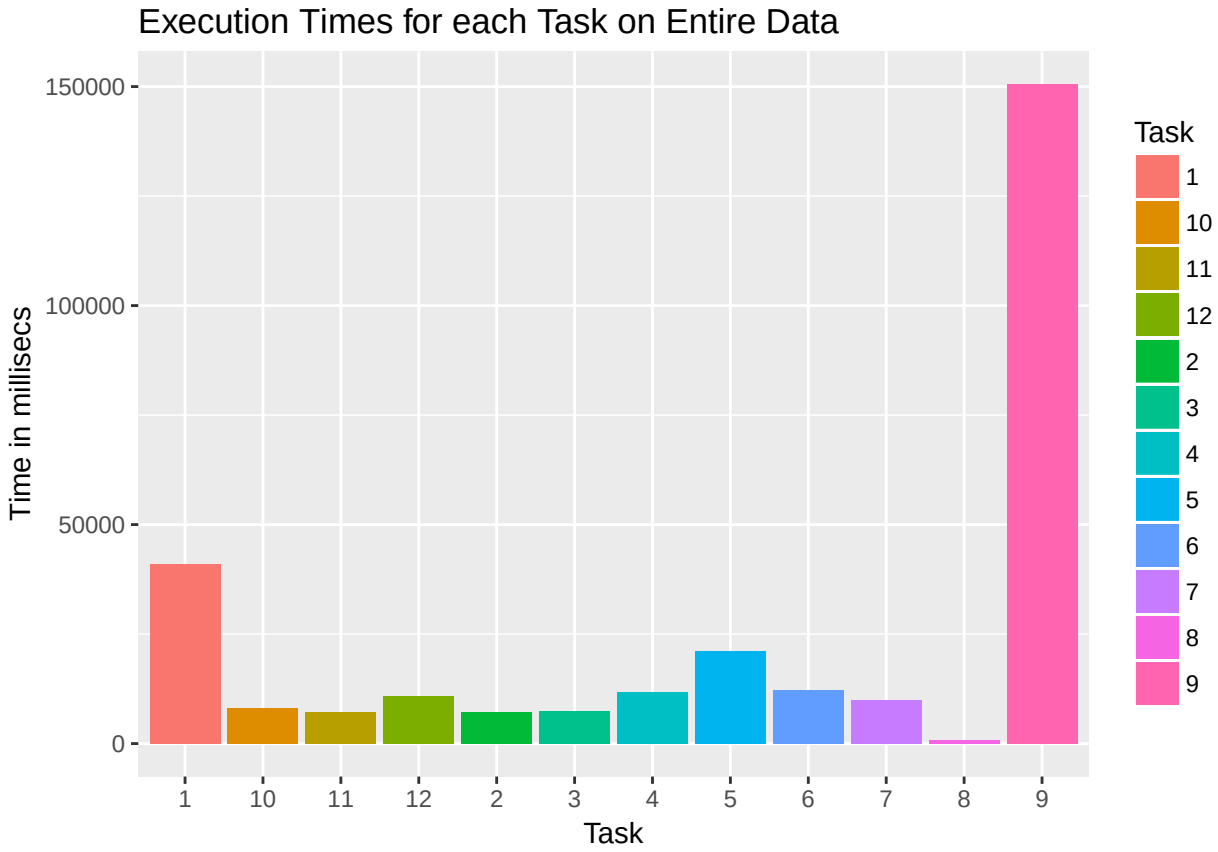
```
##       Word Count
## 1 VERSION) 55507
## 2       ME 31459
## 3     LOVE 27456
## 4   (ALBUM 26075
## 5       MY 25891
```

Times taken for different operations on entire data with persist:

```
##    Task              Query performed
## 1     1          Find distinct songs
## 2     2        Find distinct artists
## 3     3         Find distinct albums
## 4     4     Find top 5 loudest songs
## 5     5     Find top 5 longest songs
## 6     6     Find top 5 fastest songs
## 7     7     Find top 5 hottest songs
## 8     8  Find top 5 familiar artists
## 9     9   Find top 5 hottest artists
## 10   10    Find top 5 hottest Genres
## 11   11 Find top 5 most popular keys
## 12   12  Find top 5 prolific artists
## 13   13      Find top 5 common words
```

```
##          Task   Time
## 1           1  40937
## 2           2   7198
## 3           3   7441
## 4           4  11823
## 5           5  10645
## 6           5  10473
## 7           6  12137
## 8           7   9880
## 9           8    789
## 10          9 150557
## 11         10   8159
## 12         11   7170
## 13         12  10905
## 14 total time 289531
```

```
## [1] "Total time taken to execute all queries = 289531"
```



####Times taken for different operations on the **entire data without persist**:

```
##    Task              Query performed
## 1     1          Find distinct songs
## 2     2        Find distinct artists
## 3     3         Find distinct albums
## 4     4     Find top 5 loudest songs
## 5     5     Find top 5 longest songs
## 6     6     Find top 5 fastest songs
## 7     7     Find top 5 hottest songs
## 8     8  Find top 5 familiar artists
## 9     9   Find top 5 hottest artists
## 10   10    Find top 5 hottest Genres
## 11   11 Find top 5 most popular keys
## 12   12  Find top 5 prolific artists
## 13   13      Find top 5 common words
```

```
##          Task   Time
## 1           1  16153
## 2           2   7688
## 3           3   7944
## 4           4  11038
## 5           5   8342
## 6           5   8500
## 7           6   9725
## 8           7   8152
## 9           8   2250
## 10          9 159262
```

```
## 11          10   5512
## 12          11   7424
## 13          12  10212
## 14 total time 263635
```

```
## [1] "Total time taken to execute all queries = 263635"
```

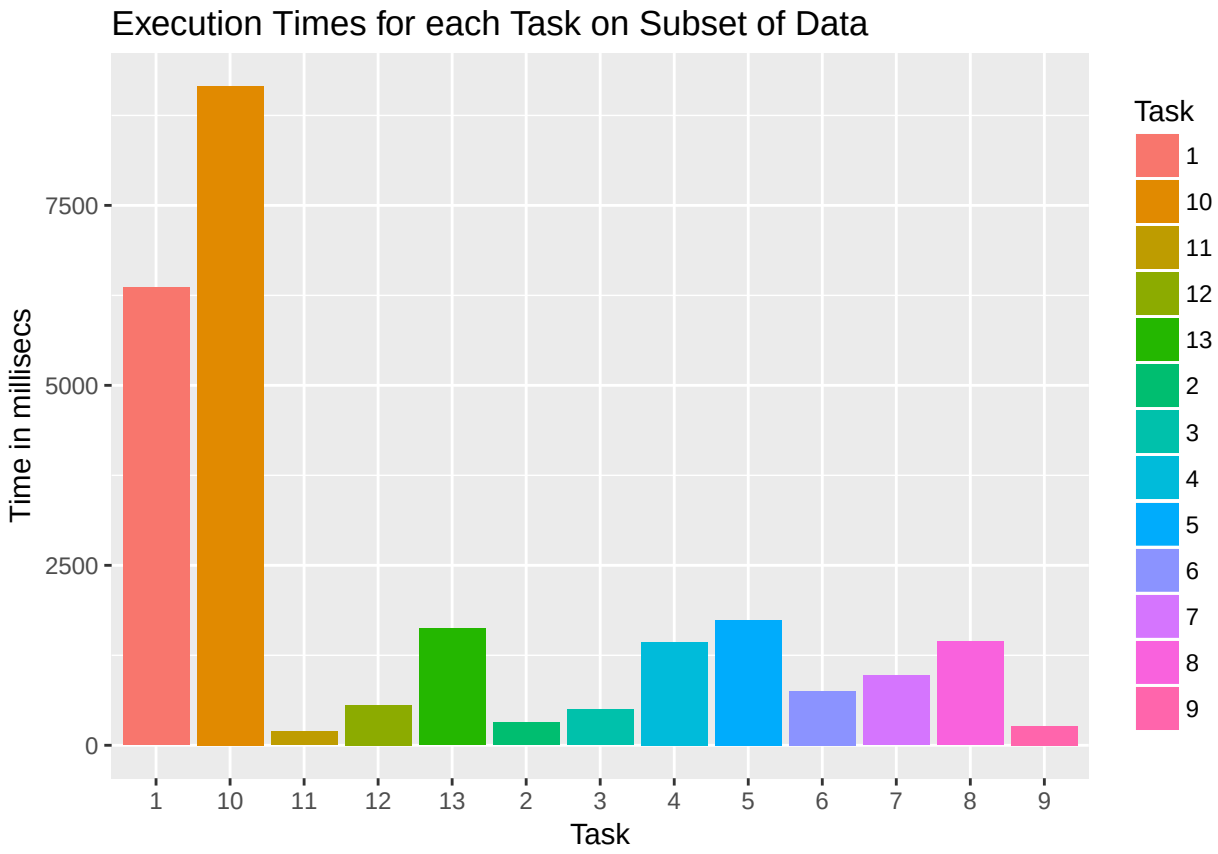
Times taken for different operations on the subset of the data with persist:

```
##    Task              Query performed
## 1     1          Find distinct songs
## 2     2        Find distinct artists
## 3     3         Find distinct albums
## 4     4     Find top 5 loudest songs
## 5     5     Find top 5 longest songs
## 6     6     Find top 5 fastest songs
## 7     7     Find top 5 hottest songs
## 8     8  Find top 5 familiar artists
## 9     9   Find top 5 hottest artists
## 10   10    Find top 5 hottest Genres
## 11   11 Find top 5 most popular keys
## 12   12  Find top 5 prolific artists
## 13   13      Find top 5 common words
```

```
##          Task  Time
## 1           1  6357
## 2           2   316
## 3           3   496
```

```
## 4            4  1437
## 5            5  1740
## 6            6   757
## 7            7   972
## 8            8  1441
## 9            9   265
## 10          10  9159
## 11          11   190
## 12          12   563
## 13          13  1623
## 14 total time 28525
```

```
## [1] "Total time taken to execute all queries = 28525"
```



Conclusions:

We also ran the code on the subset as well as the entire dataset. We observed that the time varies almost linearly. The code took 4.825 minutes to run on the entire dataset and 0.13 minutes to run on the subset with persisting the RDD. Without using the persist for the entire data, the code took 4.39 minutes.

We can see that for majority of the tasks, the execution time is faster when we do not persist the RDD and keep generating it on the fly. This may not matter as we are running our code in a standalone mode and not in a distributed mode.