This report describes the process of querying Stack Exchange using Apache Spark Data Frames using unmodified JSON files. I examined the JSON structure briefly and ensured that the files used the same structure as the XML files used in the earlier assignment. Hence, it is assumed that Data Frames API will be using a similar schema for files and field names. However, the **record counts may differ** from the previous assignments because the data is not checked for validity.

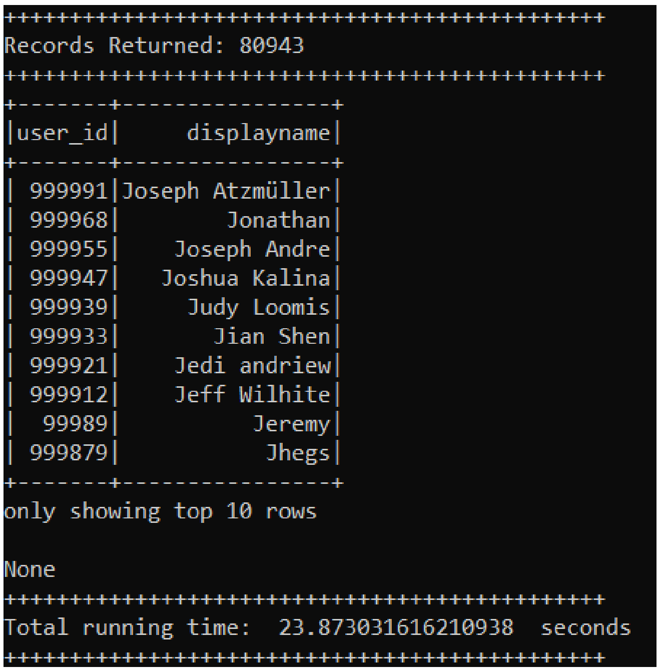
Note: The program code is provided individually for each question in the root folder. The program can be executed by following the README.md instructions. The global settings are provided in "globals.py" inside the root folder.

Query 1: Users whose display name starts with "J" and did not make any posts in 2014.

Execution Time: **23.87** seconds

The execution of the query follows the following process.

1. Get users and posts from storage.
2. Filter users with the display name starting with "J" or "j" (upper and lower case).
3. Convert/ensure the post-creation date is converted to a timestamp for posts (step 1).
4. Filter posts for the year 2014.
5. Select users from filtered posts (step 4) to have users who made posts in 2014.
6. Join (left-anti) filtered users (step 2) and filtered posts with users (step 5) to select users (of step 2) that have not made posts in 2014 (step 5).
7. Select the user ID and the display name as a final result (from step 6). Sorted descending by the user ID (which is still text, and not converted to number).



```
++++++++++++++++++++++++++++++++++++++++++++++
Records Returned: 80943
++++++++++++++++++++++++++++++++++++++++++++++
+-------+----------------+
|user_id|     displayname|
+-------+----------------+
| 999991|Joseph Atzmüller|
| 999968|        Jonathan|
| 999955|    Joseph Andre|
| 999947|   Joshua Kalina|
| 999939|     Judy Loomis|
| 999933|       Jian Shen|
| 999921|    Jedi andriew|
| 999912|    Jeff Wilhite|
|  99989|          Jeremy|
| 999879|           Jhegs|
+-------+----------------+
only showing top 10 rows

None
++++++++++++++++++++++++++++++++++++++++++++++
Total running time:  23.873031616210938  seconds
++++++++++++++++++++++++++++++++++++++++++++++
```

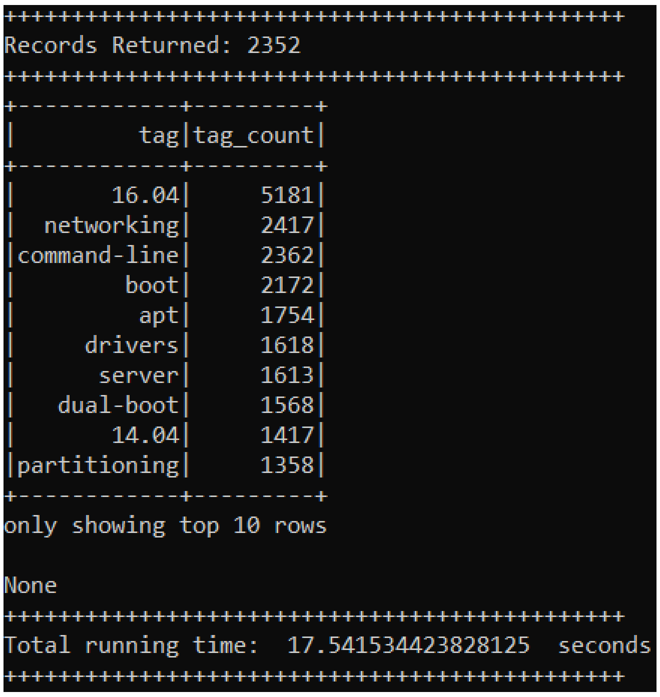
Sample (first 10) records returned by query.

Query 2: The most popular tags on posts created in 2017.

Execution Time: **17.54** seconds

The execution of the query follows the following process.

1. Get posts from storage.
2. Convert/ensure the post-creation date is converted to a timestamp for posts.
3. Filter posts for the year 2017.
4. Explode filtered posts for each tag using the tag array created by splitting tags based on bracket divisions used to separate tags on the posts.
5. Trim space for tags and filter posts with individual tags to remove tags that were created for space. (This is observed during execution).
6. Group filtered posts by tag and report tag and tag count as a final result. Sorted descending by tag count.



```
++++++++++++++++++++++++++++++++++++++++++++++
Records Returned: 2352
++++++++++++++++++++++++++++++++++++++++++++++
+------------+---------+
|         tag|tag_count|
+------------+---------+
|       16.04|     5181|
|  networking|     2417|
|command-line|     2362|
|        boot|     2172|
|         apt|     1754|
|     drivers|     1618|
|      server|     1613|
|   dual-boot|     1568|
|       14.04|     1417|
|partitioning|     1358|
+------------+---------+
only showing top 10 rows

None
++++++++++++++++++++++++++++++++++++++++++++++
Total running time:  17.541534423828125  seconds
++++++++++++++++++++++++++++++++++++++++++++++
```

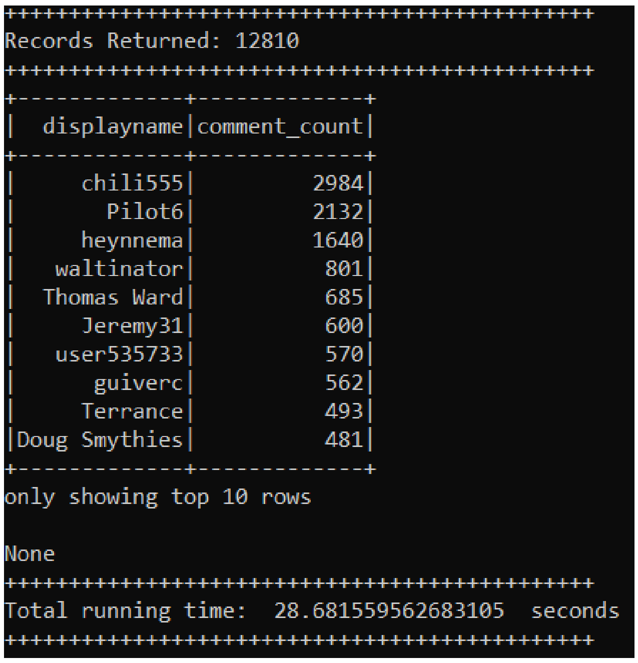
Sample (first 10) records returned by query.

Query 3: The display names of users with the most number of comments on posts with the tag “networking”.

Execution Time: **28.68** seconds

The execution of the query follows the following process.

1. Get users, posts, and comments from storage.
2. Convert tags to the array by splitting tags based on bracket divisions used to separate tags on the posts.
3. Filter posts with a tag array for the tag “networking”.
4. Join comments (step 1) with filtered posts with a “networking” tag (step 3).
5. Get comment users by joining comments (step 4), the counts of comments for each user grouping by user ID to match further with users.
6. Join users with comments count (step 5) with users (step 1) to get filtered users (networking comments users with comment counts).
7. Select the user display name and the comments count as a final result (from step 6). Sorted descending by the comments count.



```
++++++++++++++++++++++++++++++++++++++++++++++
Records Returned: 12810
++++++++++++++++++++++++++++++++++++++++++++++
+-------------+-------------+
|  displayname|comment_count|
+-------------+-------------+
|     chili555|         2984|
|       Pilot6|         2132|
|     heynnema|         1640|
|   waltinator|          801|
|  Thomas Ward|          685|
|     Jeremy31|          600|
|   user535733|          570|
|      guiverc|          562|
|     Terrance|          493|
|Doug Smythies|          481|
+-------------+-------------+
only showing top 10 rows

None
++++++++++++++++++++++++++++++++++++++++++++++
Total running time:  28.681559562683105  seconds
++++++++++++++++++++++++++++++++++++++++++++++
```

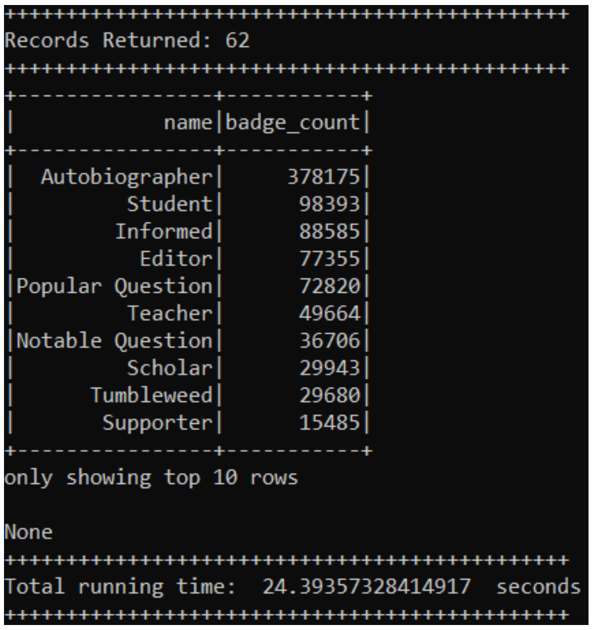
Sample (first 10) records returned by query.

Query 4: The most popular badges earned by users with less than 100 reputations.

Execution Time: **24.39** seconds

The execution of the query follows the following process.

1. Get users and badges from storage.
2. Filter users with a reputation of less than 100.
3. Join badges (step 1) with filtered users having a reputation of less than 100.
4. Get badge users, and count the counts of badges for each user grouping by badge name.



```
++++++++++++++++++++++++++++++++++++++++++++++
Records Returned: 62
++++++++++++++++++++++++++++++++++++++++++++++
+----------------+-----------+
|            name|badge_count|
+----------------+-----------+
|  Autobiographer|     378175|
|         Student|      98393|
|        Informed|      88585|
|          Editor|      77355|
|Popular Question|      72820|
|         Teacher|      49664|
|Notable Question|      36706|
|         Scholar|      29943|
|      Tumbleweed|      29680|
|       Supporter|      15485|
+----------------+-----------+
only showing top 10 rows

None
++++++++++++++++++++++++++++++++++++++++++++++
Total running time:  24.39357328414917  seconds
++++++++++++++++++++++++++++++++++++++++++++++
```

Sample (first 10) records returned by query.

Note: Instructions to run the code are provided in the README.md file in the root folder.