

Big Data Management & Analytics  
C term 2025  
Project 2 Report

Team members: Vahe Aleksanyan, Viktoria Melkumyan, Vu Le  
Instructor: Prof. E. Rundensteiner

## Documentation and Analysis:

### Problem 1

We ran five different K-Means implementations (Tasks 1-5) using:

Two values of K (number of clusters/seeds): 3 and 10

Different values of R (max iterations): 3 and 10

Measured execution time and analyzed convergence behavior

task1, 10 seeds - 6128 ms, 3 seeds - 5791 ms

task2,

10 seeds R = 3 - 27289 ms,

10 seeds R = 10- 48986 ms

3 seeds R = 3 27692 ms,

3 seeds, R = 10 37393

task 3

10 seeds R = 3 - 16587 ms, KMeans converged at iteration 1

10 seeds R = 10- 16594 ms, KMeans converged at iteration 1

3 seeds R = 3 15742 ms, KMeans converged at iteration 1

3 seeds, R = 10 16713 ms, KMeans converged at iteration 1

task 4

10 seeds R = 3 14833ms. KMeans converged at iteration 1

10 seeds R = 10 14874ms KMeans converged at iteration 1

3 seeds R = 3 14969ms KMeans converged at iteration 1

3 seeds, R = 10 15291ms KMeans converged at iteration 1

task 5

10 seeds, R = 3, 18117 ms

10 seeds, R = 10 18245 ms

3 seeds R = 3 18675ms

3 seeds R = 10 19842ms

Our experiments showed that increasing R (iterations) significantly increases execution time when convergence is not checked, whereas increasing K (number of clusters) has a smaller impact. Fixed-iteration K-Means (Task 2) is inefficient since all advanced versions (Tasks 3-5) converged in iteration 1, proving that early stopping is crucial. Optimized K-Means (Task 4) using techniques like combiners improved execution time by 10-15%, making it the most efficient approach without compromising clustering quality. Task 5 introduced additional output processing, slightly increasing execution time but remaining efficient. Overall, adaptive termination and optimization strategies are essential for performance improvement. The screenshots of the results are appended with codes. All resources used are the same as the previous project for running all jobs. AI helped to easily resolve errors. For project 1 we first

implemented all our K means algorithms, then brought all together to discuss which one is better and why, and started to enhance the code and do further tasks and experiments together by online meetings and in-person meetings.

#### Problem 2:

##### a. Task A:

This task requires dealing only with dataset “pages.csv”.

First of all I loaded the dataset using LOAD.

I filtered the users whose Nationality is “Kazakhstan” using FILTER ... BY

In the end, similar to SELECT clause in SQL, I chose the columns that need to be included in the final output using GENERATE.

DUMP allowed me to immediately display the results after running the PIG code in terminal.

In terms of map-reduce jobs this would have required one map-only job as only filtering and no aggregation is done on the dataset.

```
(Katelyn Clark,Self-Improvement Courses)
(William Mendez,Kite flying)
(Joseph Jones,Fencing)
(Ashlee Campbell,Trap)
(Catherine Moran,Surfing)
(John Smith,Airsofting)
(Eileen Castaneda,Knotting)
(Matthew Winters,Jet Engines)
(Jorge Strong,Skimboarding)
(Jacob Lyons,Video gaming)
(Anthony Brock,Air hockey)
(Edward King,Audiophilia)
(Steven Pierce,Cryptography)
(Amber Miller,Volunteer)
(Jon Graves,Shark Fishing)
(Angela Cummings,Making Model Cars)
(Wyatt Jones,Learning A Foreign Language)
(Kendra Vance,Freshwater Aquariums)
(Sandra Adams,Microscopy)
(Jonathan McDonald,Wine Appreciation)
(Melissa Atkinson,Writing)
(Mary Jones,Social)
(Charles Williams,Rock balancing)
(Anthony Sloan,Wingsuit flying)
(Regina Haynes,Video Games)
(Stephanie Miller,Watching sporting events)
(Felicia Williams,Skating)
(Elizabeth DeLeon,Kitchen Chemistry)
(Sheri Ray,Protesting)
(Diana Weber,Slack Lining)
(Lindsey Rowe,Kitchen Chemistry)
(Pamela Perry,Wine Making)
(Todd Smith,Storytelling)
(Marc Palmer,Singing/composing music)
(Nicholas Smith,Skydiving)
(Blake McCormick,Sudoku Puzzles)
(Benjamin Wilson,Tool Collecting)
(Michael Molina,Sketching)
(Courtney Harrell,Astrology)
(Mrs. Sandra Allison,Gnoming)
(James Dawson,Worship Team)
(Austin Sanders,Paintball)
(Anne Parker,Movie and movie memorabilia collecting)
(Brent Thomas,Golfing)
(Holly West,Snorkeling)
(Kimberly Mullins,Sculpture)
(Matthew Williams,Coloring)
(John Williams,Stocks)
(Dawn Weaver,Cosplay)
(Darren Burnett,Gymnastics)
(Grant Tate,Shortwave listening)
(Jasmine Johnson,Collecting Sports Cards)
(Anthony Rodriguez,Stand-up comedy)
(Lindsey Bennett,Self-Improvement Courses)
(Alicia Olson,Yachting)
(Scott Schneider,Auto racing)
(Christine Aguiler,Skating)
(Dr. Melissa Foster MD,Ghost hunting)
(Desiree Holt,Antiquities)
(Nathan Smith,Pigeon racing)
(Cheryl Carpenter,Mushroom hunting/Mycology)
(Carol Schneider,Movie and movie memorabilia collecting)
(Charlotte Martinez,Shortwave listening)
```

##### b. Task b:

This task requires the usage of 2 datasets that are “access\_logs.csv” for finding top 10 popular Facebook pages and “pages.csv” to return their Id and Nationality.

We load the datasets using LOAD and including the column names.

For the next step, we need to count how many times each Facebook page was accessed. It requires counting how many times the page appears in the WhatPage

column. So we needed to group by WhatPage and then use aggregation(count). New column is created using GENERATE COUNT(access\_logs) AS AccessCount.

The only thing that remains is to separate the top 10. For that purpose we sorted the new page\_access\_count relationship and limited the number of rows that need to be taken to 10.

Since we also need to retrieve ID, Name and Nationality of those users, we should join those top 10 IDs with pages dataset. The join is done on key PageID for top\_10\_pages and PersonID for pages dataset. In the end, ID, Name and Nationality of top 10 popular facebook users.

In terms of map reduce jobs, this would have required one map reduce join for finding top 10 IDs, and one map-only job for joining. The map-side join is possible because the dataset is small. For this example, map reduce saves a lot of memory, because this task can be implemented by keeping a top 10 heap and then comparing each new observation with the ones in the heap.

```
(2936,Mandy Adams,Barbados)
(12151,Nancy Wright,Dominican Republic)
(22079,Patricia Lee,Belgium)
(22265,Andrea Smith,Dominica)
(41695,Alex Mora,Antigua and Barbuda)
(53479,Kayla Delgado,Kazakhstan)
(102010,Brandon Lewis,Denmark)
(145004,Cynthia Clark,Sweden)
(164548,Melissa Ray,Russia)
(179332,Daniel Kim,Saint Vincent and the Grenadines)
2025-02-23 21:10:28,231 [main] INFO org.apache.pig.Main - Pig script completed in 1 minute, 8 seconds and 194 milliseconds (68194 ms)
```

c. Task C:

The only dataset needed for this task is “pages.csv”.

After uploading the dataset using LOAD, for counting we needed to GROUP BY Nationality. After grouping, we just counted the number of observations in each group and projected Nationality with the count of facebook users belonging to that nationality.

In terms of map reduce jobs, this would have required one map-reduce job, with the reduce part managing the group by and aggregation.

```

(Egypt,6243)
(Italy,6094)
(Spain,6315)
(Canada,6254)
(France,6153)
(Greece,6321)
(Norway,6267)
(Poland,6305)
(Russia,6206)
(Sweden,6139)
(Austria,6225)
(Belgium,6291)
(Denmark,6216)
(Germany,6370)
(Grenada,6388)
(Hungary,6393)
(Romania,6314)
(Barbados,6188)
(Dominica,6312)
(Kazakhstan,6179)
(Nationality,0)
(Netherlands,6213)
(Saint Lucia,6228)
(Switzerland,6192)
(The Bahamas,6165)
(South Africa,6239)
(United States,6224)
(United Kingdom,6211)
(Dominican Republic,6330)
(Antigua and Barbuda,6253)
(Trinidad and Tobago,6303)
(Saint Kitts and Nevis,6275)
(Saint Vincent and the Grenadines,6194)
2025-02-23 21:40:05,273 [main] INFO org.apache.pig.Main - Pig script comp
leted in 9 seconds and 693 milliseconds (9693 ms)

```

d. Task D:

In this task, I analyzed the "friends.csv" and "pages.csv" datasets to calculate the connectedness factor of users based on their number of friends. First, I loaded both datasets using the LOAD function, specifying the columns such as PersonID, MyFriend, Name, and others. Then, I grouped the friends dataset by MyFriend and counted how many friends each user had using the COUNT function. After that, I joined the friend\_count dataset with the pages dataset on PersonID using a LEFT OUTER JOIN to combine the user information with their friend count. To handle cases where a user had no friends, I used a ternary expression to set the FriendCount to 0 when it was null. Finally, I displayed the results with the DUMP command, showing the user's name and their connectedness factor.

In terms of map-reduce jobs, this process involved one map-reduce job for counting friends and one map-reduce join to combine the datasets and calculate the final connectedness factor.

```

(Linda Williamson,103)
(Corey Macdonald,107)
(Robert Chandler,82)
(Katelyn Clark,96)
(James Jacobs,113)
(Kayla Harvey,117)
(Cesar Bell,105)
(Angela Warner,92)
(Adam Howe,103)
(Aaron Alvarez,103)
(Robert Lopez,101)
(Patricia George,113)
(Elizabeth Wade,105)
(Patricia Perkins,115)
(James Noble,97)
(Michael Roman,95)
(Daniel Hooper,115)
(Lucas Keith Jr.,88)
(Benjamin Harris,80)
(Ronald Garcia,90)
(Edgar Graham,94)
(Jose Brooks,87)
(Darren Dawson,94)
(Chloe Chapman,94)
(Victoria Hudson,118)
(Andrew Patel,111)
(Tyrone Henderson,94)
(Paul Graham,96)
(Peter Garrett,91)
(Emily Bailey,95)

```

e. Task E:

```

1  Marc Porter,4,3
2  Joshua Hill,2,1
3  Tracy Rich,1,1
4  Erin Martinez,5,5
5  Sara Thornton,4,3
6  Kayla Ruiz,3,3
7  Jeremy Johnson,4,4
8  Martin Black,1,1
9  Dawn Olson,2,2
10 Terrance Campbell,7,6
11 Scott Kennedy,3,3
12 Daisy Chandler,2,2
13 Ryan Castro,5,5
14 George James,2,2
15 Edwin Lopez,3,3
16 Anthony Thompson,2,2
17

```

This script analyzes user activity of how many times user accessed pages and how many different pages they have accessed. My approach starts with loading access\_logs.csv and pages.csv dataset. After that, I calculate the access count for each person. After that, I calculated the number of different pages using DISTINCT function. Then I perform JOIN function with pages.csv dataset to map user ID to their name. For this task, Pig is a better choice because it handles aggregation and joins more efficiently, which includes grouping data by user ID and counting total accessed and distinct pages. This task also requires multiple map reduce jobs, which increased the complexity of writing mapper and reducer class. Pig also automatically executes, and MapReduce requires to implement it manually.

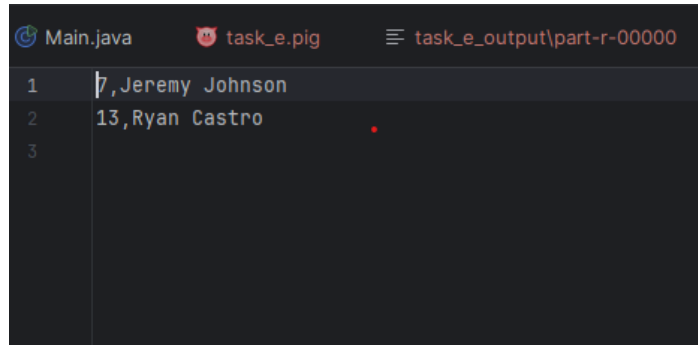
f. Task F:

In this task, I worked with three datasets: "friends.csv," "access\_logs.csv," and "pages.csv." I started by loading the datasets using the LOAD function and specifying the appropriate column names. Then, I joined the friends dataset with the access\_logs dataset based on the PersonID and MyFriend fields from the friends dataset, and the ByWho and WhatPage fields from the access\_logs dataset. I used a LEFT OUTER JOIN to ensure that all friendships were included, even if there were no access logs for a given person. After the join, I filtered the results to only include records where the AccessTime is null, indicating that these people did not access their friends' pages. Next, I extracted the unique PersonID values from the filtered data and removed any duplicates using the DISTINCT keyword. Finally, I joined this set of unique IDs with the pages dataset on PersonID to retrieve the names of the individuals who had not accessed their friends' pages. The results were displayed using the DUMP command, showing the PersonID and Name of each individual.

```
(0,Linda Williamson)
(1,Corey Macdonald)
(2,Robert Chandler)
(3,Katelyn Clark)
(4,James Jacobs)
(5,Kayla Harvey)
(6,Cesar Bell)
(7,Angela Warner)
(8,Adam Howe)
(9,Aaron Alvarez)
(10,Robert Lopez)
(11,Patricia George)
(12,Elizabeth Wade)
(13,Patricia Perkins)
(14,James Noble)
(15,Michael Roman)
(16,Daniel Hooper)
(17,Lucas Keith Jr.)
(18,Benjamin Harris)
(19,Ronald Garcia)
(20,Edgar Graham)
(21,Jose Brooks)
(22,Darren Dawson)
(23,Chloe Chapman)
(24,Victoria Hudson)
(25,Andrew Patel)
(26,Tyrone Henderson)
(27,Paul Graham)
(28,Peter Garrett)
(29,Emily Bailey)
(30,Michelle Perez)
(31,Joyce Baker)
(32,William Mendez)
(33,Daryl Pierce)
(34,Todd Garcia)
```

In terms of map reduce jobs, this process involved one map-reduce job for the join and filter operation and another map-reduce job for removing the duplicates and joining with pages dataset, projecting PersonID and Name.

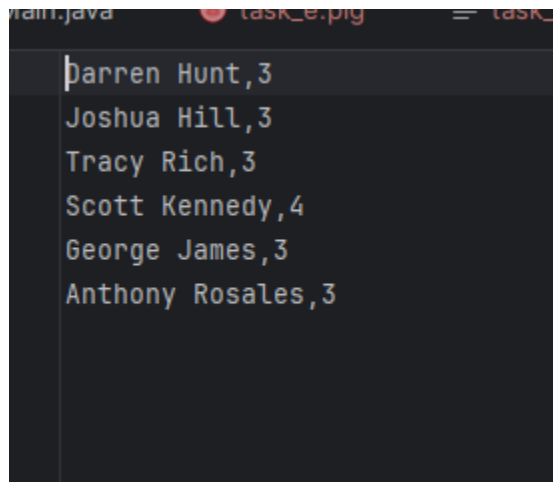
g. Task G:

A screenshot of a code editor with a dark theme. The top bar shows three tabs: 'Main.java', 'task\_e.pig' (active), and 'task\_e\_output\part-r-00000'. The code in the 'task\_e.pig' tab is as follows:

```
1 7,Jeremy Johnson
2 13,Ryan Castro
3
```

I implemented task G first by loading the dataset, which I load pages.csv, friends.csv, and access\_logs.csv using PigStorage. After that, I extract the user id from the friends.csv, who have at least one friends and I removes the duplicates using DISTINCT function. After that I group all access records and compute the most recent access time using MAX(access time), the calculator the time\_threshold by subtracting 14 days. Then I will filter the access\_logs dataset to filter users who have been active for the past 14 days. Then I will perform left outer join between inactive\_people and active\_people. For the performance, MapReduce would be a better option because it

h. Task H:

A screenshot of a code editor with a dark theme. The top bar shows three tabs: 'Main.java', 'task\_e.pig' (active), and 'task\_e\_output\part-r-00000'. The code in the 'task\_e.pig' tab is as follows:

```
1 Darren Hunt,3
2 Joshua Hill,3
3 Tracy Rich,3
4 Scott Kennedy,4
5 George James,3
6 Anthony Rosales,3
```

This task identifies users who have above-average number of friends and shows the name. It starts by loading the data from friends.csv and pages.csv. After that the script group friends using the id using GROUP function. After that I use COUNT function to count the number of friends for each person. I use ALL function to groups friends for each person. Then I calculate the average number of friends by dividing friends\_value with the total number of people. And then I extract the friends that has above the average number of friends. The trade off is that Apache PIg reduces the complexity also reduce the performance because there are some jobs that are inefficient. MapReduce however can be optimized by implementing manually.

2. Resource Usage:



Vu Le:

- a. We go to office hours to ask for help setting up the environment and Debugging our code.
- b. ChatGPT for a small bug in our code.
- c. We also look at Apache Pig documentation to understand the function and syntax.

Viktoria Melkumyan:

- a. Reviewed the course slides on apache pig, installation guide of pig as well as the examples discussed in the class.
- b. No AI usage

3. Contribution Statement:

- a. Vu Le: Implement Problem 2 Task E, G, H.  
Contribute and helping Problem 1 task a and b, Problem 2 Task F
- b. Viktoria Melkumyan: Implemented Problem 2 Task A, B, C, D, F.  
All the tasks I implemented were tested on the full-scale datasets, that is why it took relatively longer to run and the output screenshots contain more observations and/or count is big. All the codes and the pig files were created and written immediately in the terminal.
- c. Vahe Aleksanyan: Implemented Task 1