# CSCI 5408

# DATA MANAGEMENT AND WAREHOUSING

## Assignment - 1

Banner ID: B00984406

GitLab Assignment Link: https://git.cs.dal.ca/jems/csci5408_s24_b00984406_jems_patel

# Contents

# Problem 1:

## 1.1 Identifying 22 Entity Sets

*Table 1: Entities with the attributes and justification*

| Sr. No. | Entity | Attributes | Justification |
|---|---|---|---|
| 1 | Park | • Park_ID (PK)<br>• Park_Name<br>• Park_Area<br>• Size<br>• Description<br>• Park_Location<br>• Opening_Hours | Provides vital data about each park necessary for management and informative to the visitors. |
| 2 | Site | • Site_ID(PK)<br>• Site_Number<br>• Site_Name<br>• Site_Location<br>• Park_ID(FK)<br>• Capacity<br>• Rate<br>• Allowed_Equipments | Represents specific areas within parks for activities and reservations. |
| 3 | Reservation | • Reservation_ID (PK)<br>• User_ID(FK)<br>• Site_ID(FK)<br>• Duration<br>• Status (Pending or Confirmed) | For Tracking the reservations made by the users and managing the bookings for the park sites. |
| 4 | Amenity | • Amenity_ID (PK)<br>• Amenity_Name<br>• Park_ID (FK)<br>• Type<br>• Description | Brief details about the amenities provided by the park. |
| 5 | Contact us | • Contact_US_ID(PK)<br>• Address<br>• Website<br>• Email<br>• Phone_Number<br>• Social_Media | Provides contact information for public inquiries. |

| | | | |
|---|---|---|---|
| | | • Park_ID(FK)<br>• Lake_ID(FK)<br>• Department_ID(FK) | |
| 6 | Events | • E7vent_ID(PK)<br>• Event_Name<br>• Date<br>• Description<br>• Park_ID(FK) | For representing and tracking the events in parks. |
| 7 | Employment Opportunities | • Opportunity_ID (PK)<br>• Position_Title<br>• Description<br>• Requirements<br>• Location<br>• Salary_Range<br>• Application_Deadline<br>• Park_ID(FK) | Lists all the job openings which are related to the park. |
| 8 | Users | • User_ID(PK)<br>• Name<br>• Email<br>• Password<br>• Mobile_Number<br>• Address<br>• Reservation (FK)<br>• Courses (FK) | Contains information about users who interact with the system particularly in the parks and the education for taking the courses. |
| 9 | FAQ | • FAQ_ID(PK)<br>• Question<br>• Answer<br>• Category<br>• Park_ID(FK) | Provides answers to common questions from the public for better information. |
| 10 | Lake | • Lake_ID (PK)<br>• Lake_Name<br>• Depth | Contains the information about the lakes. |
| 11 | Fishing Area | • Area_ID(PK)<br>• Lake_ID(FK)<br>• Area_Name<br>• Description<br>• Area_Map<br>• Location | Identifies specific areas within lakes for recreational activities. |

| | | | |
|---|---|---|---|
| 12 | Lake CIty | • City_ID(PK)<br>• Name<br>• Area_ID(FK) | Represents the nearby city for the geographical purpose. |
| 13 | Programs | • Program_ID(PK)<br>• Description<br>• Program_Name<br>• Contact_Information<br>• Area_ID(FK)<br>• Department_ID(FK) | For tracking all the programs related to the lakes and the departments. |
| 14 | Publications | • Publication_ID(PK)<br>• Content<br>• Publication_Type<br>• Date<br>• Program_ID(FK)<br>• Department (FK) | For managing the documents like reports, information and all. |
| 15 | Department | • Department_ID(PK)<br>• Department_Name<br>• Description<br>• Contact_Information | Organises all the related information to different managing departments. |
| 16 | Projects | • Project_ID(PK)<br>• Project_Name<br>• Status<br>• Start_Date<br>• End_Date<br>• Description<br>• Department_ID(FK) | Details ongoing and completed projects within departments. |
| 17 | Education | • Education_ID(PK)<br>• Program_Name<br>• Program_Type<br>• Description<br>• Department_ID(FK) | For tracking all the education programs and initiatives for a particular department. |
| 18 | Course | • Course_ID(PK)<br>• Course_Name<br>• Description<br>• Instructor_ID(FK)<br>• Duration<br>• Fees<br>• Prerequisite Course | Provides information on courses offered under educational programs. |

| | | | |
|---|---|---|---|
| 19 | Instructor | • Instructor_ID(PK)<br>• Instructot_Name<br>• Bio<br>• Contact_Information<br>• Expertise<br>• Course_ID(FK) | Details instructors for educational courses. |
| 20 | Field_Offices | • Office_ID(PK)<br>• Office_Name<br>• Address<br>• Phone_Number<br>• Email<br>• Department_ID(FK) | Identifies regional offices for operational management. |
| 21 | News | • News_ID(PK)<br>• Title<br>• Content<br>• Date<br>• Park_ID(FK)<br>• Department_ID(FK) | Manages news and updates related to parks and natural resources. |
| 22 | County | • County_ID (PK)<br>• County_Name<br>• Population<br>• Description<br>• Area(In size) | Provides geographic and administrative details about counties. |
| 23 | Burn_Restriction | • Burn_ID (PK)<br>• Burn_Status<br>• Additional_Restrictions<br>• Burn_Allowed_From<br>• Burn_Allowed_To<br>• County_ID (FK) | For tracking the safety and burning restrictions in different areas. |

## 1.2 Initial ERD:

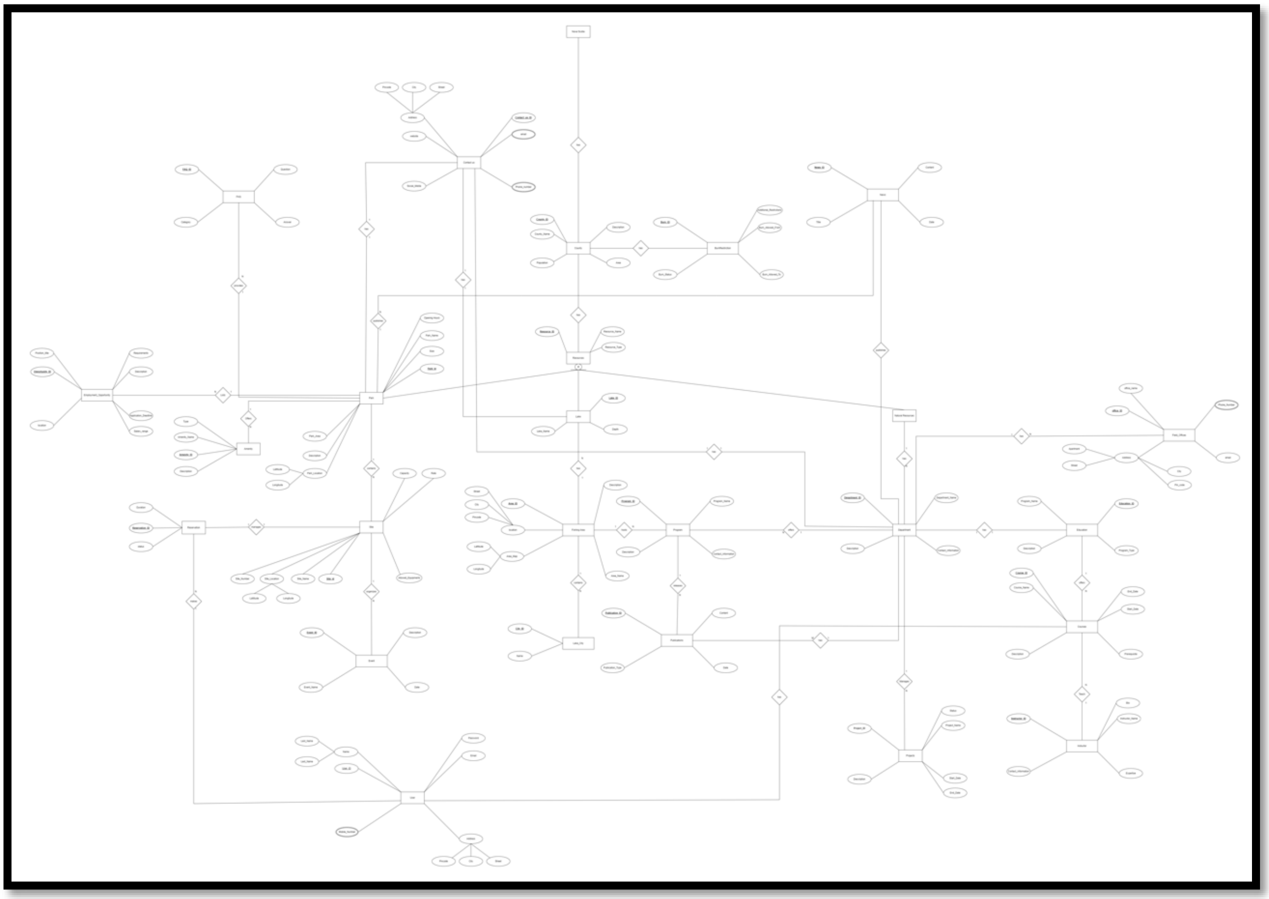


*Figure 1.2: Initial_ERD*

# 1.3 Design Issues:

## Fan Trap:

A fan trap occurs when one entity has "one-to-many" relationships with more than two other entities.

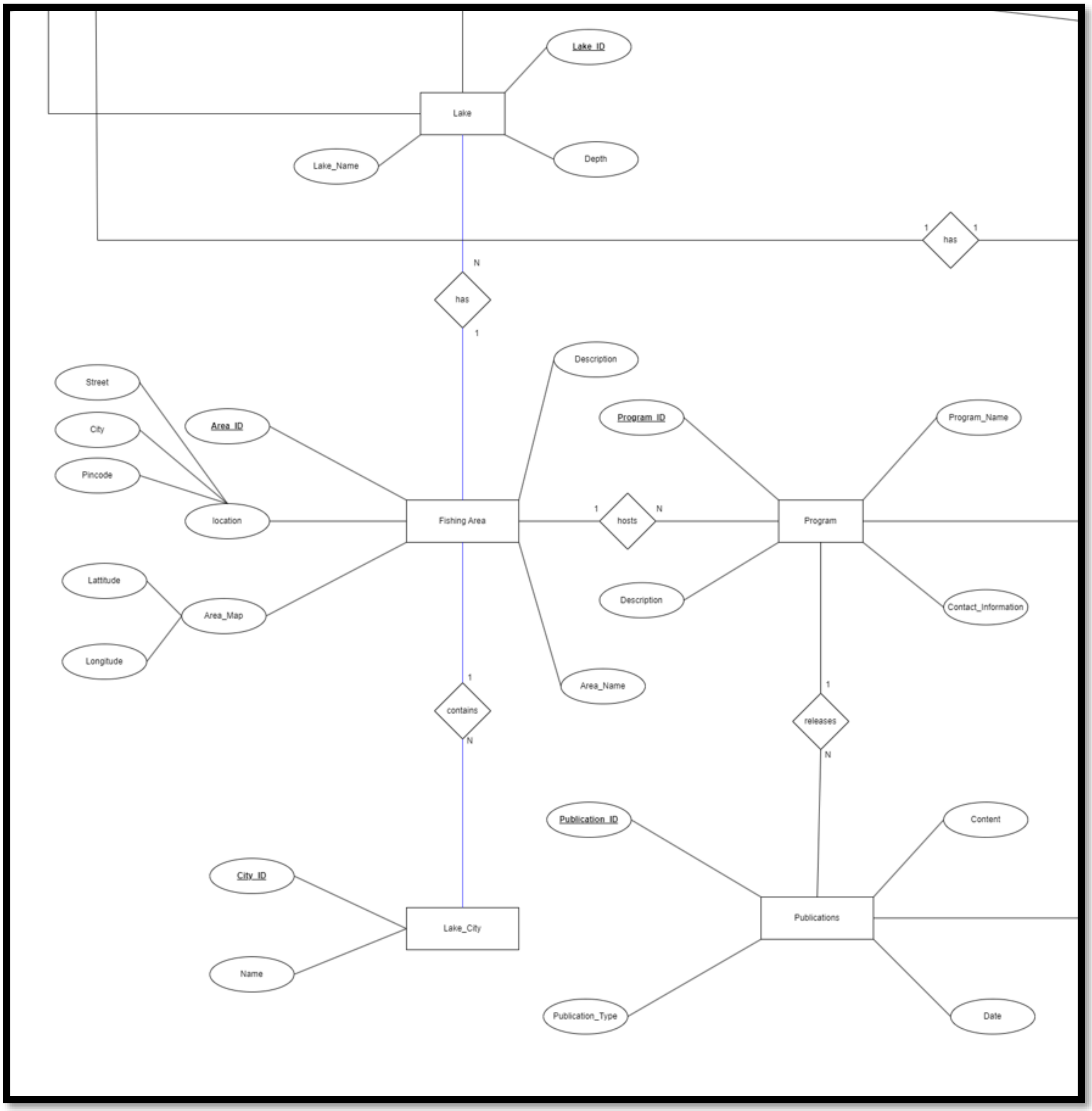


*Figure 1.3.1: Fan trap*

- Here, we can see that the Fishing Area has two one-to-many relationships. If we want to find out in which Lake City a particular Lake is located, we can't directly obtain this information. To solve this problem, we can rearrange the relationships among the three entities as described below.

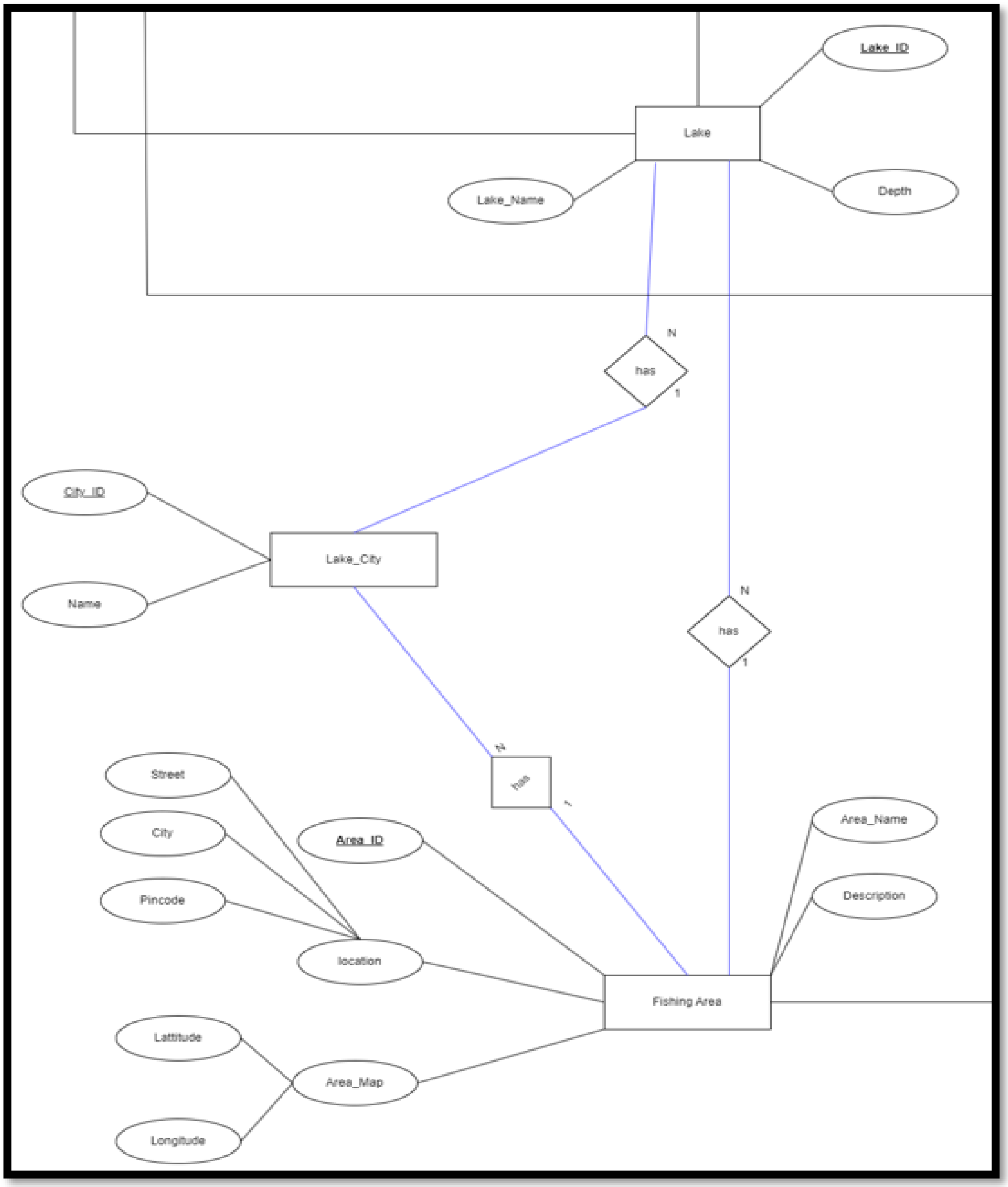


*Figure 1.3.2: After solving fan trap*

- By introducing a one-to-many relationship between Lake City and Fishing Area, we can resolve the fan trap and obtain both the fishing area and the lake city for a particular lake.

- Here, in the above we are not able to get which instructor is joined in which education as instructor is assigned with the courses and if any course is deleted then the instructor automatically deleted which is not true as all the instructors are assigned to the education.

## Chasm Trap:

- A chasm trap occurs when a model indicates a relationship between entity types, but a connection between specific instances is missing.

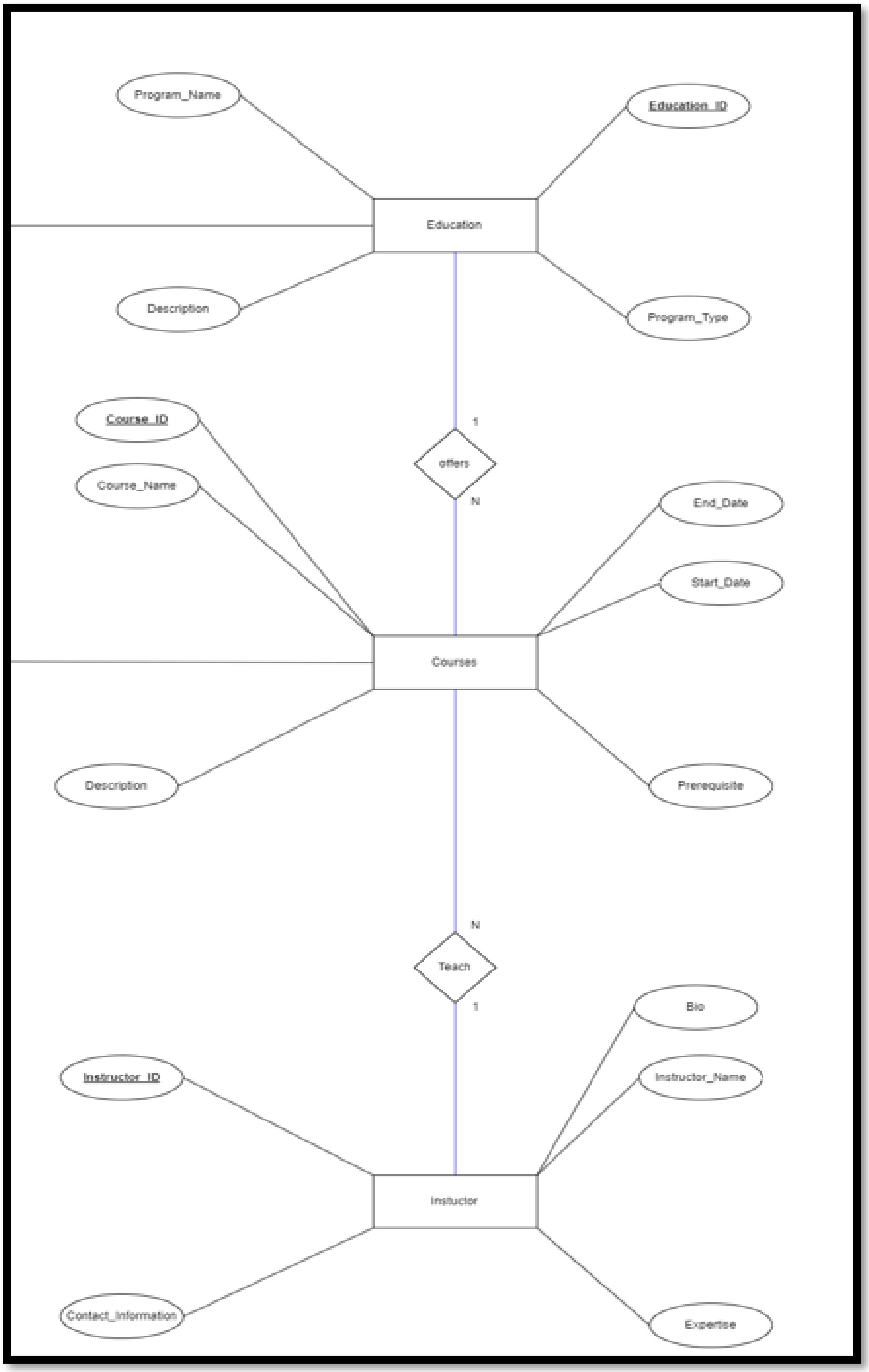


*Figure 1.3.3: Chasm Trap*

- Here, the model implies a path between Education and Instructor, with the instructor depending on the Courses. If a Course is deleted, the associated Instructor is also deleted. To solve this issue, we need to establish a direct relationship between Instructor and Education.

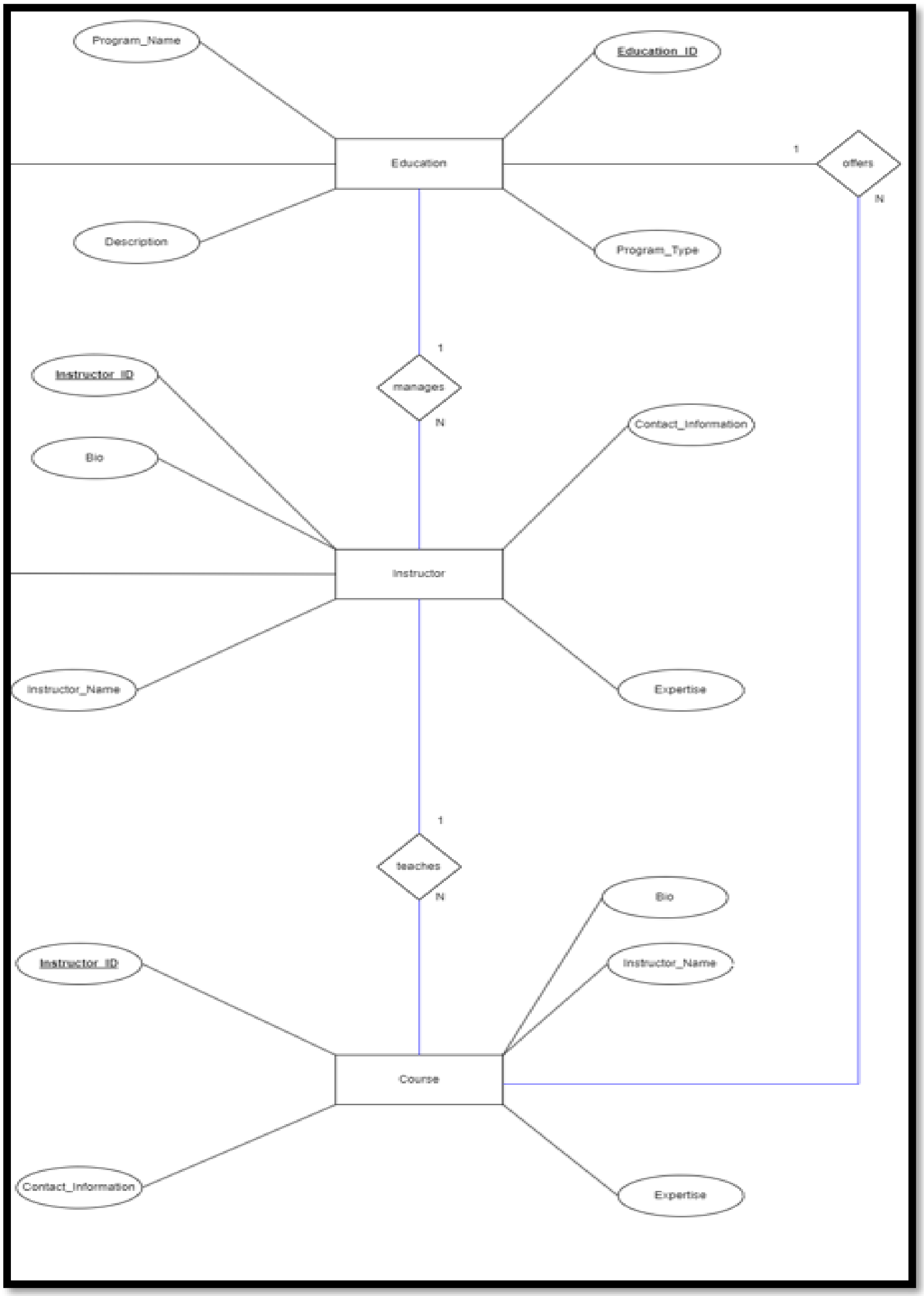


*Figure 1.3.4: After solving chasm trap*

- By introducing the new one to many relations between the education and instructor, we will solve this problem.

## 1.4 Final ERD:

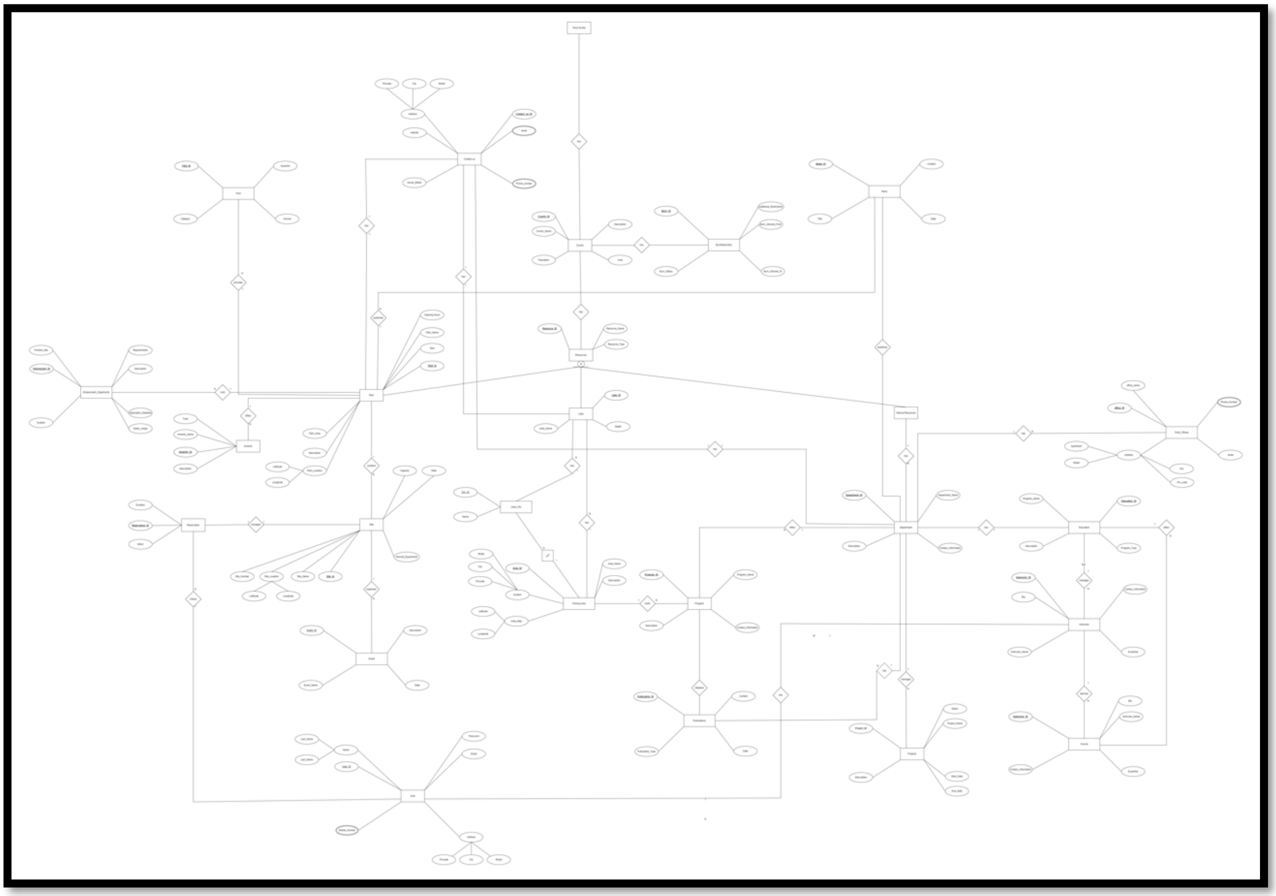


*Figure 1.4: Final_ERD*

# Tool that I use for creating the ERD:

Draw. io is a powerful and interactive web-based diagramming tool which I used while making the Enhanced Entity-Relationship (EER) diagrams. It has an extremely simple layout, which is supplemented by a vast number of shapes, connectors, and other settings, so this tool is perfect for visualizing database structures. This reduces the difficulties associated with the addition of entities, attributes and the connectivity of relationships as well as the capacity for the integration of team comments and feedback. Additionally, Draw. I have realized that by exporting the diagrams in different IO format ensures that besides my EER diagrams that can easily be shared with other people as well as can easily be incorporated in other documentations or be presented.

# Problem 2:

## 2.1 Fragmentation:

### Vertical Fragmentation:

Vertical fragmentation subdivides a table into several tables that are more specialized than the original table based on a column. This strategy makes further access patterns rationalized, duplicate work eliminated, and the specific queries performed quicker.

Upon analyzing the dataset, I identified two primary types of data: The characteristics of the two are tweet data characteristics and user characteristics. As for vertical fragmentation, it was used for improving data management and query performances. This involved dividing the original table into two distinct tables: “Tweets” and “Users” Leaking some of the columns pertinent to tweets and users into different tables in order form a proper database design, I separated tweets and users into different tables to enhance access patterns of the data type in question and correspondingly query the pertinent tables with more ease.

The splitting of data into “Tweets” and “Users” was done on a single machine, which guaranteed the proper and timely partition of information relative to tweets and information concerning users. This was a fragmentation strategy adopted to enhance the retrieval process and to eliminate any duplication by bringing out the nature of data handled by every table.

Certainly! Here's the structured content for the report section on vertical fragmentation of the `users` and `tweets` tables:

Users Table Fragmentation:

- User Id
- Name
- Username
- User Bio
- Verified or non-verified
- Profile URL
- Protected or non-protected
- Followers Count
- Following Count
- Account Creation Date

Tweets Table Fragmentation:

- Tweet Id
- Tweet URL
- Tweet Posted Time (UTC)
- Tweet Content
- Tweet Type
- Client
- Retweets Received
- Likes Received
- Tweet Location
- Tweet Language
- User Id (for user association)
- Impressions

## Table Relationship and Integration:

User Id Relationship:

- Both users and tweets tables maintain a relationship via the User Id column.
- Enables seamless joins and efficient queries for linking tweets to their respective users.

To perform the fragmentation, I have created a java program in which I read the given CSV file and perform fragmentation by splitting the sample.csv file into two files ‘users’ and ‘tweets’. I used the “opencsv” library for reading data from the csv file. After running this program I got two files.

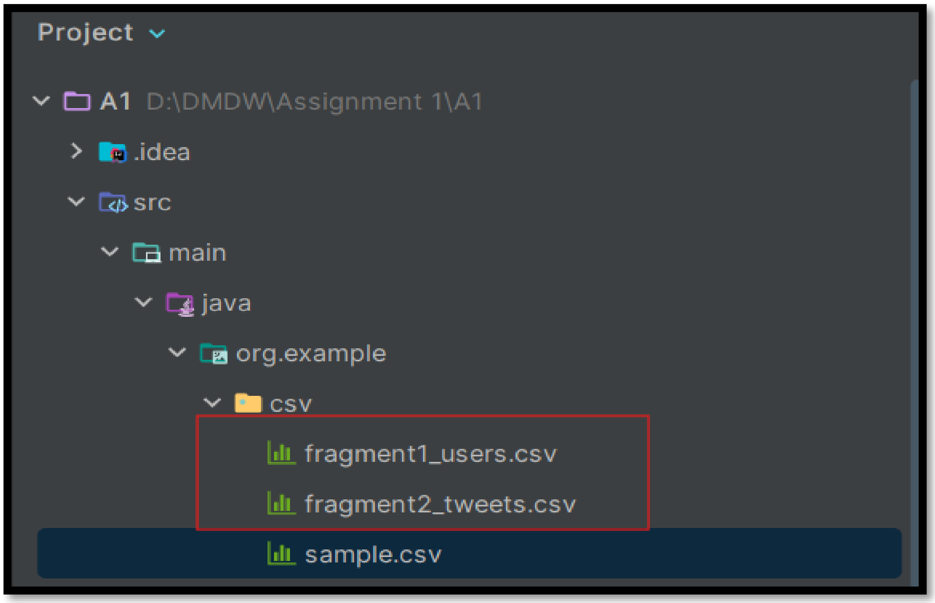
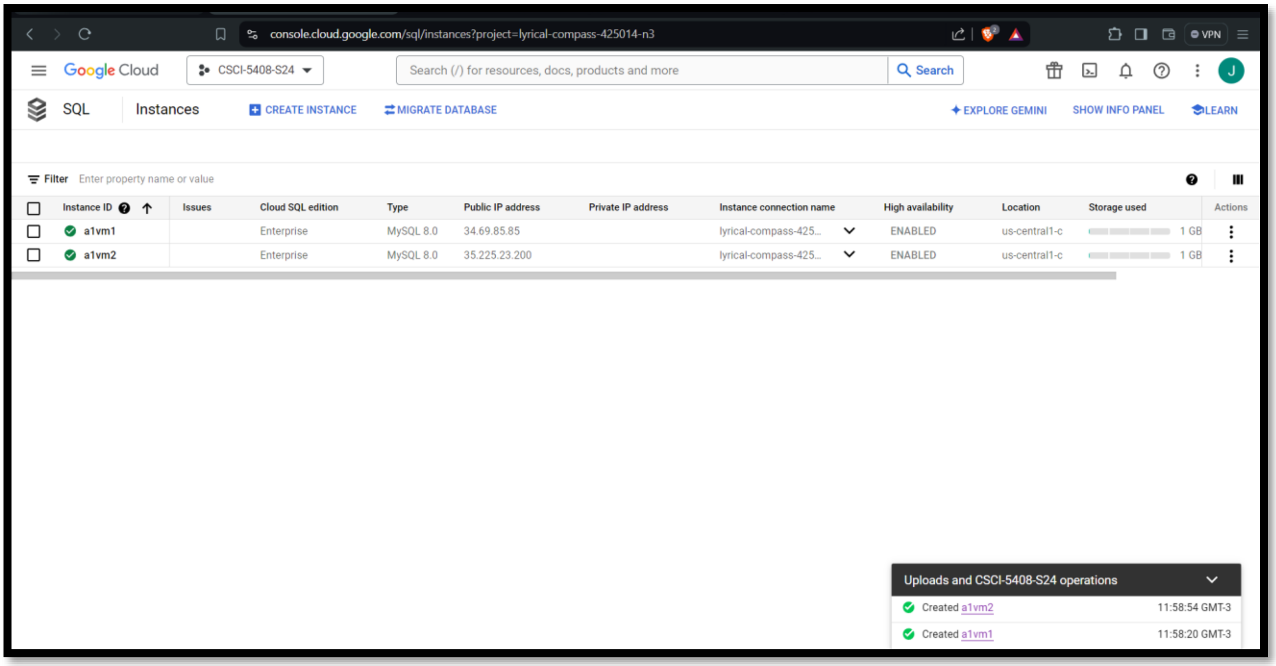


*Figure 2.1: Fragmentation files for the given sample*

## 2.2 Cloud VM Setup:

I have created a two VM instances on the google cloud as below as I learned previously in the lab.

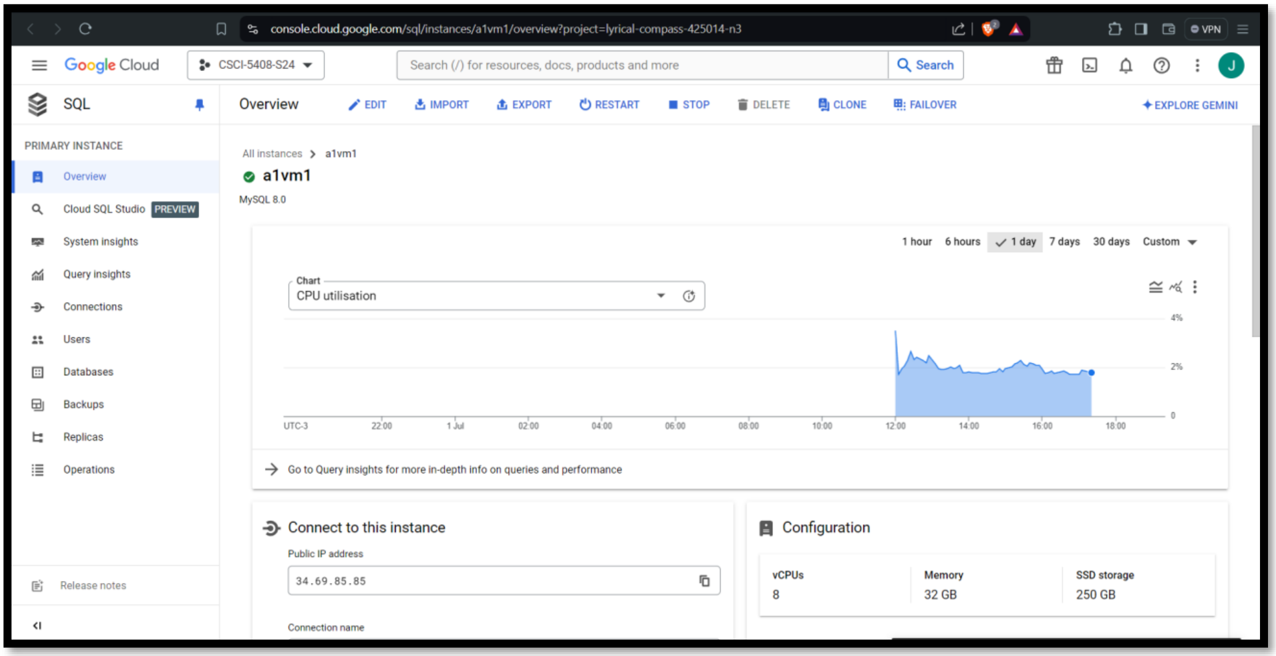

console.cloud.google.com/sql/instances?project=lyrical-compass-425014-n3
Google Cloud
CSCI-5408-S24
Search (/) for resources, docs, products and more
Search
SQL
Instances
CREATE INSTANCE
MIGRATE DATABASE
EXPLORE GEMINI
SHOW INFO PANEL
LEARN
Filter
Enter property name or value

| ☐ | Instance ID | Issues | Cloud SQL edition | Type | Public IP address | Private IP address | Instance connection name | High availability | Location | Storage used | Actions |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ☐ | ✓ a1vm1 | | Enterprise | MySQL 8.0 | 34.69.85.85 | | lyrical-compass-425... | ENABLED | us-central1-c | 1 GB | ⋮ |
| ☐ | ✓ a1vm2 | | Enterprise | MySQL 8.0 | 35.225.23.200 | | lyrical-compass-425... | ENABLED | us-central1-c | 1 GB | ⋮ |

Uploads and CSCI-5408-S24 operations

| | |
|---|---|
| ✓ Created a1vm2 | 11:58:54 GMT-3 |
| ✓ Created a1vm1 | 11:58:20 GMT-3 |

*Figure 2.2.1: Creation of the VM instances*



✓ a1vm1

MySQL 8.0

Connect to this instance

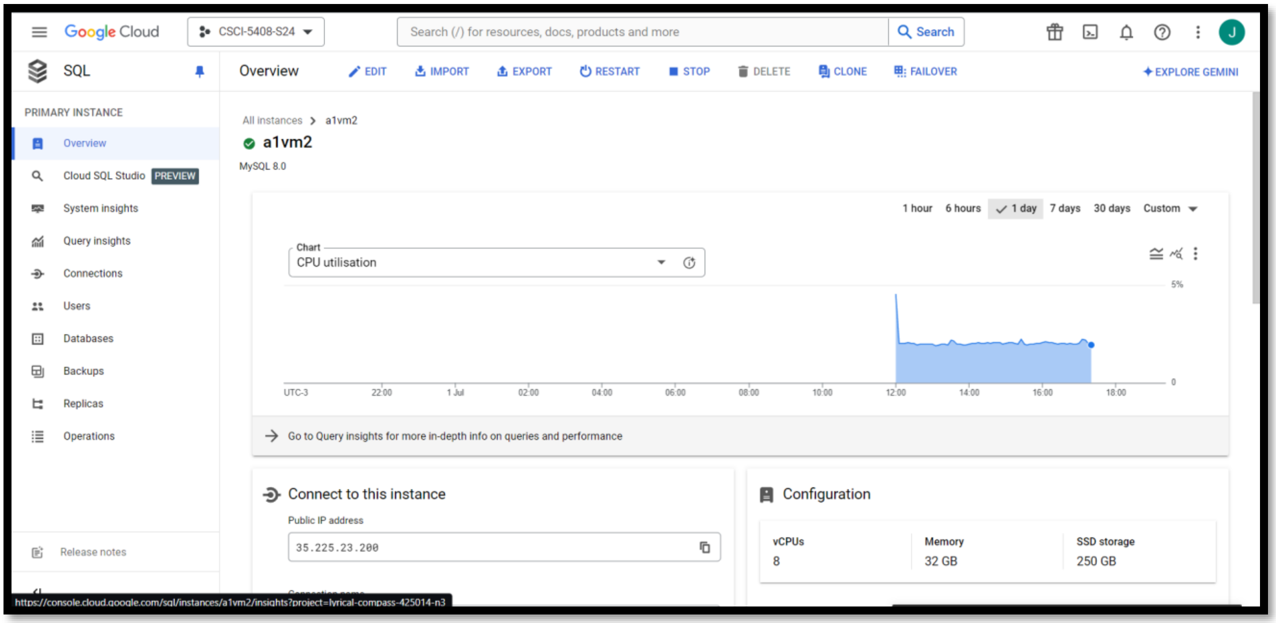
Public IP address: 34.69.85.85

Connection name

Configuration

| vCPUs | Memory | SSD storage |
|---|---|---|
| 8 | 32 GB | 250 GB |

*Figure 2.2.2: Configuration of the VM1*



All instances > a1vm2

a1vm2

MySQL 8.0

Chart: CPU utilisation

Connect to this instance

Public IP address: 35.225.23.200

Configuration

| vCPUs | Memory | SSD storage |
|---|---|---|
| 8 | 32 GB | 250 GB |

https://console.cloud.google.com/sql/instances/a1vm2/insights?project=lyrical-compass-425014-n3

*Figure 2.2.3: Configuration of the VM2*

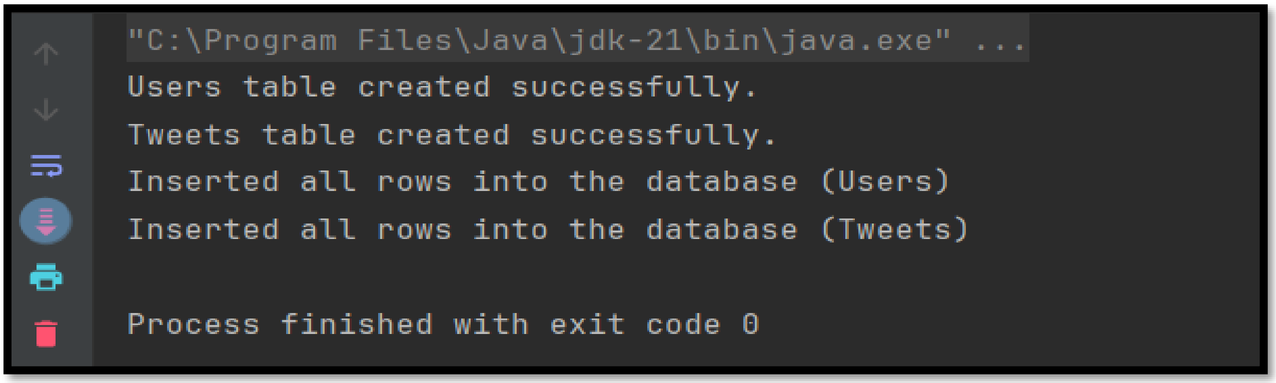
## 2.3 Importing the data into database:

I have created java code for importing the CSV files into two different databases in two different VMs. First, I tried hard to import it by simply using the UI of the workbench, but I was not able to do that way, so I made one java code and import the csv file into the two different VM’s.

In this code, I also wrote the logic for creating the table first into the VM instance databases. Here, I provide you with the query and format of the database that I used to create it.

```
CREATE TABLE users (
 user_id BIGINT,
 name VARCHAR(100),
 username VARCHAR(100),
 user_bio TEXT,
 verified_or_non_verified VARCHAR(50),
 profile_url VARCHAR(255),
 protected_or_non_protected VARCHAR(50),
 user_followers INT,
 user_following INT,
 user_account_creation_date TIMESTAMP,
 impressions INT,
 PRIMARY KEY (user_id)
);

CREATE TABLE tweets (
 tweet_id BIGINT,
 tweet_url VARCHAR(255),
 tweet_posted_time_utc TIMESTAMP,
 tweet_content TEXT,
 tweet_type VARCHAR(50),
 client VARCHAR(100),
 retweets_received INT,
 likes_received INT,
 tweet_location TEXT,
 tweet_language VARCHAR(50),
 user_id BIGINT,
 PRIMARY KEY (tweet_id, user_id)
);
```



```
"C:\Program Files\Java\jdk-21\bin\java.exe" ...
Users table created successfully.
Tweets table created successfully.
Inserted all rows into the database (Users)
Inserted all rows into the database (Tweets)

Process finished with exit code 0
```

*Figure 2.3.1: Output of the CSVToMYSQL java program*

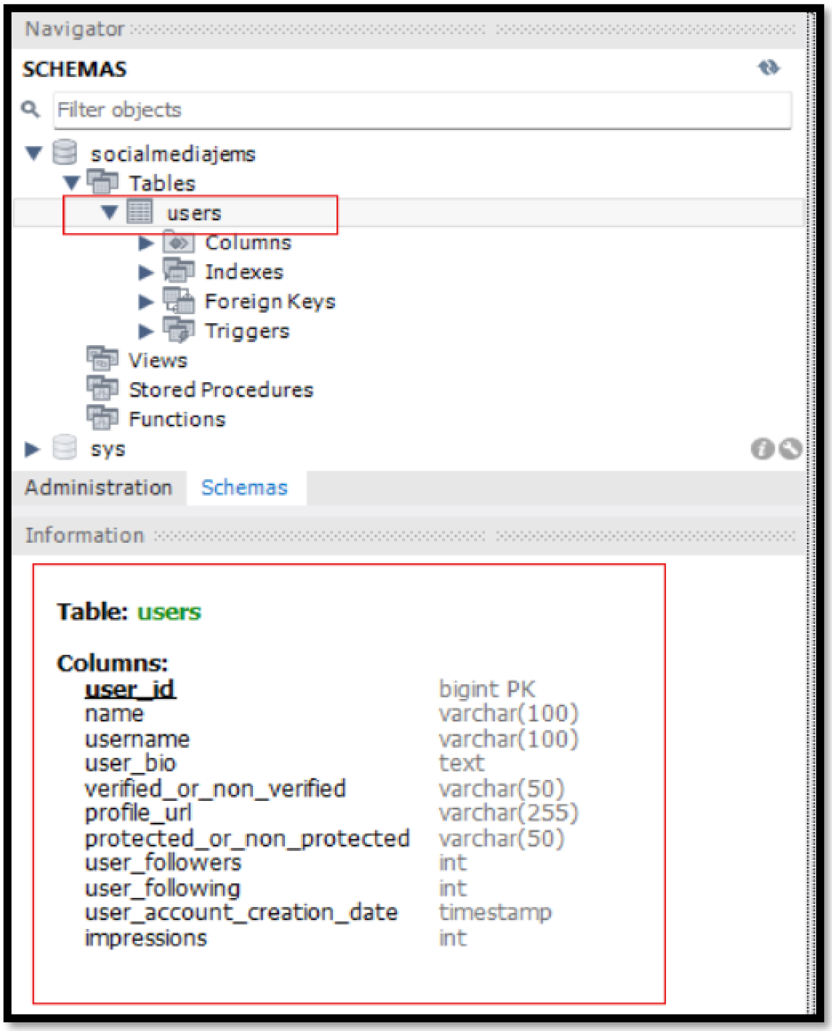


**Table: users**

**Columns:**

| | |
|---|---|
| **user_id** | bigint PK |
| name | varchar(100) |
| username | varchar(100) |
| user_bio | text |
| verified_or_non_verified | varchar(50) |
| profile_url | varchar(255) |
| protected_or_non_protected | varchar(50) |
| user_followers | int |
| user_following | int |
| user_account_creation_date | timestamp |
| impressions | int |

*Figure 2.3.2: Creation of the users table*

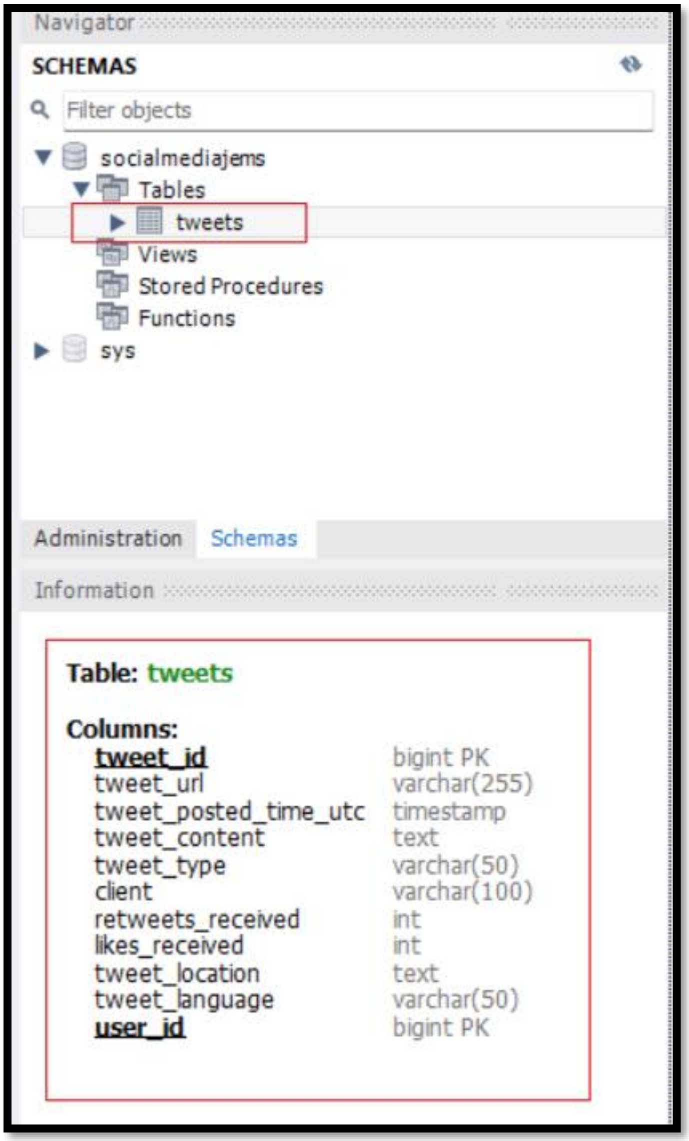
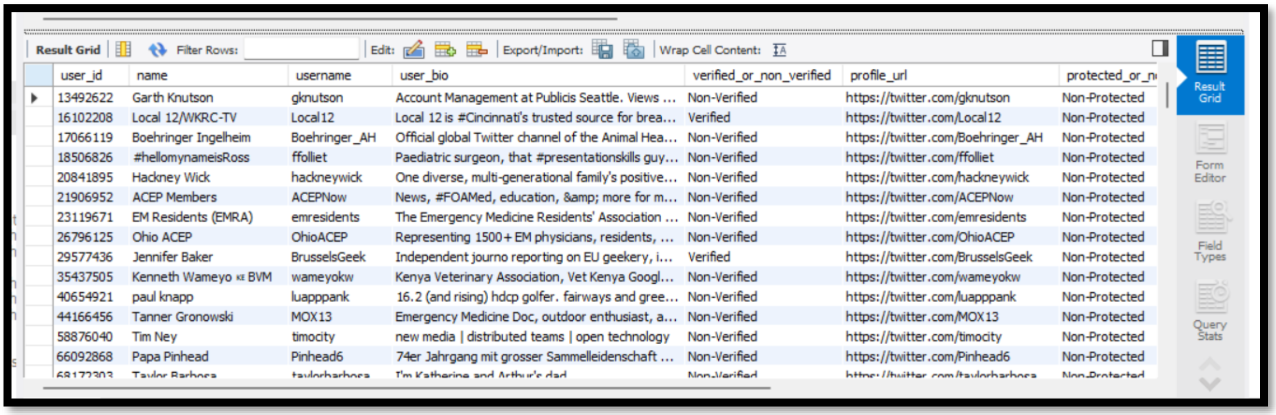


Table: tweets

Columns:

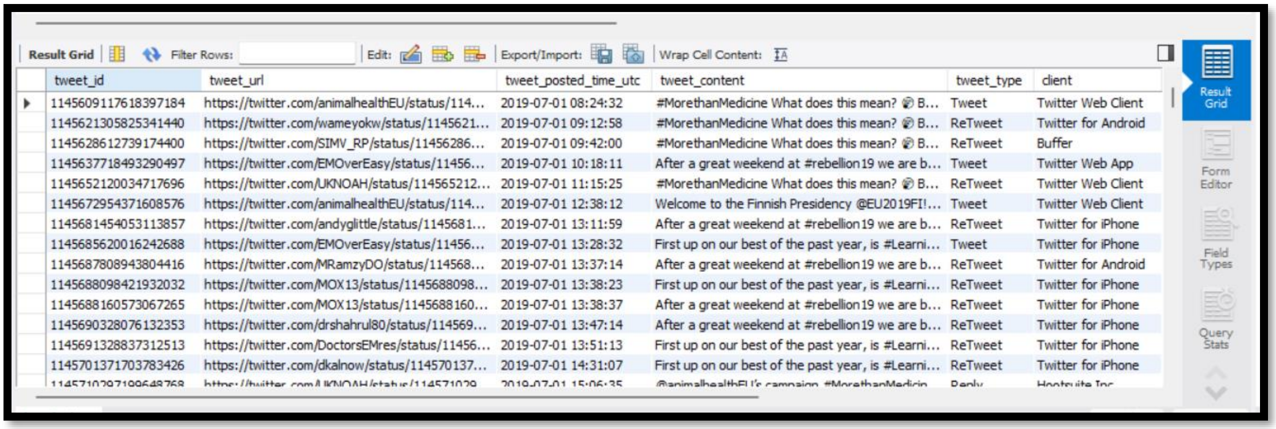
| | |
|---|---|
| **tweet_id** | bigint PK |
| tweet_url | varchar(255) |
| tweet_posted_time_utc | timestamp |
| tweet_content | text |
| tweet_type | varchar(50) |
| client | varchar(100) |
| retweets_received | int |
| likes_received | int |
| tweet_location | text |
| tweet_language | varchar(50) |
| **user_id** | bigint PK |

*Figure 2.3.3: Creation of the tweets table*



| user_id | name | username | user_bio | verified_or_non_verified | profile_url | protected_or_n |
|---|---|---|---|---|---|---|
| 13492622 | Garth Knutson | gknutson | Account Management at Publicis Seattle. Views ... | Non-Verified | https://twitter.com/gknutson | Non-Protected |
| 16102208 | Local 12/WKRC-TV | Local12 | Local 12 is #Cincinnati's trusted source for brea... | Verified | https://twitter.com/Local12 | Non-Protected |
| 17066119 | Boehringer Ingelheim | Boehringer_AH | Official global Twitter channel of the Animal Hea... | Non-Verified | https://twitter.com/Boehringer_AH | Non-Protected |
| 18506826 | #hellomynameisRoss | ffolliet | Paediatric surgeon, that #presentationskills guy... | Non-Verified | https://twitter.com/ffolliet | Non-Protected |
| 20841895 | Hackney Wick | hackneywick | One diverse, multi-generational family's positive... | Non-Verified | https://twitter.com/hackneywick | Non-Protected |
| 21906952 | ACEP Members | ACEPNow | News, #FOAMed, education, &amp; more for m... | Non-Verified | https://twitter.com/ACEPNow | Non-Protected |
| 23119671 | EM Residents (EMRA) | emresidents | The Emergency Medicine Residents' Association ... | Non-Verified | https://twitter.com/emresidents | Non-Protected |
| 26796125 | Ohio ACEP | OhioACEP | Representing 1500+ EM physicians, residents, ... | Non-Verified | https://twitter.com/OhioACEP | Non-Protected |
| 29577436 | Jennifer Baker | BrusselsGeek | Independent journo reporting on EU geekery, i... | Verified | https://twitter.com/BrusselsGeek | Non-Protected |
| 35437505 | Kenneth Wameyo 🇰🇪 BVM | wameyokw | Kenya Veterinary Association, Vet Kenya Googl... | Non-Verified | https://twitter.com/wameyokw | Non-Protected |
| 40654921 | paul knapp | luapppank | 16.2 (and rising) hdcp golfer. fairways and gree... | Non-Verified | https://twitter.com/luapppank | Non-Protected |
| 44166456 | Tanner Gronowski | MOX13 | Emergency Medicine Doc, outdoor enthusiast, a... | Non-Verified | https://twitter.com/MOX13 | Non-Protected |
| 58876040 | Tim Ney | timocity | new media \| distributed teams \| open technology | Non-Verified | https://twitter.com/timocity | Non-Protected |
| 66092868 | Papa Pinhead | Pinhead6 | 74er Jahrgang mit grosser Sammelleidenschaft ... | Non-Verified | https://twitter.com/Pinhead6 | Non-Protected |
| 68172303 | Taylor Barbosa | taylorbarbosa | I'm Katherine and Arthur's dad | Non-Verified | https://twitter.com/taylorbarbosa | Non-Protected |

*Figure 2.3.4: Inserted users data on the first VM*

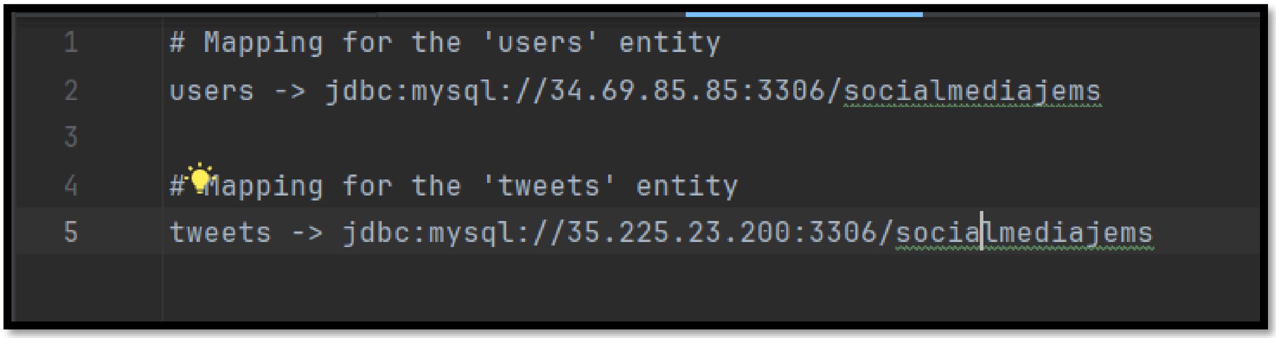


| tweet_id | tweet_url | tweet_posted_time_utc | tweet_content | tweet_type | client |
|---|---|---|---|---|---|
| 1145609117618397184 | https://twitter.com/animalhealthEU/status/114... | 2019-07-01 08:24:32 | #MorethanMedicine What does this mean? 🌍 B... | Tweet | Twitter Web Client |
| 1145621305825341440 | https://twitter.com/wameyokw/status/1145621... | 2019-07-01 09:12:58 | #MorethanMedicine What does this mean? 🌍 B... | ReTweet | Twitter for Android |
| 1145628612739174400 | https://twitter.com/SIMV_RP/status/11456286... | 2019-07-01 09:42:00 | #MorethanMedicine What does this mean? 🌍 B... | ReTweet | Buffer |
| 1145637718493290497 | https://twitter.com/EMOverEasy/status/11456... | 2019-07-01 10:18:11 | After a great weekend at #rebellion19 we are b... | Tweet | Twitter Web App |
| 1145652120034717696 | https://twitter.com/UKNOAH/status/114565212... | 2019-07-01 11:15:25 | #MorethanMedicine What does this mean? 🌍 B... | ReTweet | Twitter Web Client |
| 1145672954371608576 | https://twitter.com/animalhealthEU/status/114... | 2019-07-01 12:38:12 | Welcome to the Finnish Presidency @EU2019FI!... | Tweet | Twitter Web Client |
| 1145681454053113857 | https://twitter.com/andyglittle/status/1145681... | 2019-07-01 13:11:59 | After a great weekend at #rebellion19 we are b... | ReTweet | Twitter for iPhone |
| 1145685620016242688 | https://twitter.com/EMOverEasy/status/11456... | 2019-07-01 13:28:32 | First up on our best of the past year, is #Learni... | Tweet | Twitter for iPhone |
| 1145687808943804416 | https://twitter.com/MRamzyDO/status/114568... | 2019-07-01 13:37:14 | After a great weekend at #rebellion19 we are b... | ReTweet | Twitter for Android |
| 1145688098421932032 | https://twitter.com/MOX13/status/1145688098... | 2019-07-01 13:38:23 | First up on our best of the past year, is #Learni... | ReTweet | Twitter for iPhone |
| 1145688160573067265 | https://twitter.com/MOX13/status/1145688160... | 2019-07-01 13:38:37 | After a great weekend at #rebellion19 we are b... | ReTweet | Twitter for iPhone |
| 1145690328076132353 | https://twitter.com/drshahrul80/status/114569... | 2019-07-01 13:47:14 | After a great weekend at #rebellion19 we are b... | ReTweet | Twitter for iPhone |
| 1145691328837312513 | https://twitter.com/DoctorsEMres/status/11456... | 2019-07-01 13:51:13 | First up on our best of the past year, is #Learni... | ReTweet | Twitter for iPhone |
| 1145701371703783426 | https://twitter.com/dkalnow/status/114570137... | 2019-07-01 14:31:07 | First up on our best of the past year, is #Learni... | ReTweet | Twitter for iPhone |
| 1145710297199648768 | https://twitter.com/UKNOAH/status/114571029 | 2019-07-01 15:06:35 | @animalhealthEU's campaign #MorethanMedicin | Reply | Hootsuite Inc |

Figure 2.3.5: Inserted tweets data on the second VM

## 2.4 GDC Creation:

Certainly! The chosen mapping format for the 'users' and 'tweets' entities utilizes specific JDBC URLs to streamline database operations within the social media application. By linking 'users' to `jdbc:mysql://34.69.85.85:3306/socialmediajems` and 'tweets' to `jdbc:mysql://35.225.23.200:3306/socialmediajems`, the format ensures efficient data access and management. This structured approach enables focused retrieval of user profiles and interactions from one database instance, while tweet content and engagement metrics are accessed from another. The separation enhances query performance, simplifies maintenance, and supports scalability as data volumes grow, aligning database architecture with optimized data handling practices for sustained application performance.



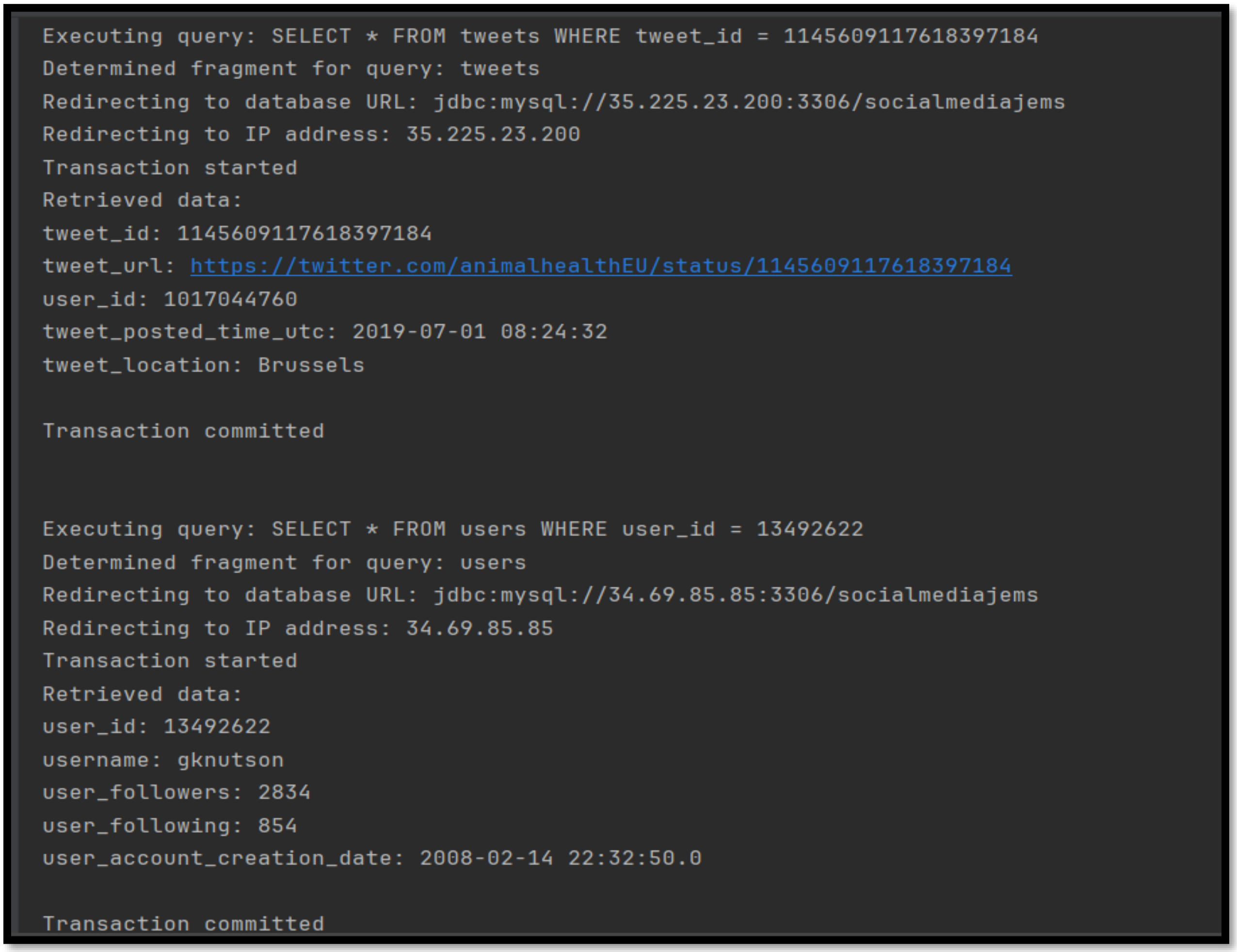
```
# Mapping for the 'users' entity
users -> jdbc:mysql://34.69.85.85:3306/socialmediajems

# Mapping for the 'tweets' entity
tweets -> jdbc:mysql://35.225.23.200:3306/socialmediajems
```

*Figure 2.4.1: GDC File for the determining the selection of the query*

## 2.5 Client Driver Program:

"After completing the setup process, I developed a ClientDriver.java application following the specified instructions to handle multiple queries efficiently. To showcase the functionality of vertical fragmentation and the GDC file, I executed two simple SELECT queries based on tweet_id and user_id. According to the fragmentation strategy, operations related to users are directed to the first VM at IP address 34.69.85.85, while operations concerning tweets are routed to the second VM at IP address 35.225.23.200, as defined in the GDC file mappings provided earlier. This setup ensures optimized data retrieval and management. Specifically, the SELECT query for tweets successfully accessed data from the second VM, demonstrating the effective distribution of tweet-related information. Similarly, the SELECT query for users accessed relevant columns from the first VM. This approach not only enhances query performance but also supports scalability and efficient data handling practices in the social media application environment."



```
Executing query: SELECT * FROM tweets WHERE tweet_id = 1145609117618397184
Determined fragment for query: tweets
Redirecting to database URL: jdbc:mysql://35.225.23.200:3306/socialmediajems
Redirecting to IP address: 35.225.23.200
Transaction started
Retrieved data:
tweet_id: 1145609117618397184
tweet_url: https://twitter.com/animalhealthEU/status/1145609117618397184
user_id: 1017044760
tweet_posted_time_utc: 2019-07-01 08:24:32
tweet_location: Brussels

Transaction committed


Executing query: SELECT * FROM users WHERE user_id = 13492622
Determined fragment for query: users
Redirecting to database URL: jdbc:mysql://34.69.85.85:3306/socialmediajems
Redirecting to IP address: 34.69.85.85
Transaction started
Retrieved data:
user_id: 13492622
username: gknutson
user_followers: 2834
user_following: 854
user_account_creation_date: 2008-02-14 22:32:50.0

Transaction committed
```

*Figure 2.5.1: Output of the client driver program in order to show the fragmentation*

# References:

[1] Parks Nova Scotia. (n.d.). [Online]. Available: https://parks.novascotia.ca/. [Accessed: 1-Jul-2024].

[2] Parks Nova Scotia - Nova Scotia Region. (n.d.). [Online]. Available: https://parks.novascotia.ca/region/nova-scotia [Accessed: 1-Jul-2024].

[3] Parks Nova Scotia - News and Events. (n.d.). [Online]. Available: https://parks.novascotia.ca/news-and-events. [Accessed: 1-Jul-2024].

[4] Nova Scotia Jobs. (n.d.). [Online]. Available: https://jobs.novascotia.ca/. [Accessed: 1-Jul-2024].

[5] Nova Scotia - Sportfishing Lakes. (n.d.). [Online]. Available: https://novascotia.ca/fish/sportfishing/our-lakes/. [Accessed: 1-Jul-2024].

[6] Nova Scotia - Lake Survey Program. (n.d.). [Online]. Available: https://novascotia.ca/nse/surface.water/lakesurveyprogram.asp. [Accessed: 1-Jul-2024].

[7] Parks Nova Scotia. (n.d.). [Online]. Available: https://parks.novascotia.ca/. [Accessed: 1-Jul- 2024].

[8] Nova Scotia Department of Natural Resources - Education Programs. (n.d.). [Online]. Available: https://novascotia.ca/natr/Education/. [Accessed: 1-Jul-2024].

[9] Stack Overflow. (2017). "Java code to split CSV file into different CSV files and extracting a single column," Stack Overflow. [Online]. Available: https://stackoverflow.com/questions/45026140/java-code-to-split-csv-file-into-different-csv-files-and-extracting-a-single-col. [Accessed: 1-Jul-2024].

[10] Stack Overflow. (2019). "How to import a large CSV into a MySQL with Java using prepared statements and batching," Stack Overflow. [Online]. Available: https://stackoverflow.com/questions/55233640/how-to-import-a-large-csv-into-a-mysql-with-java-using-prepared-statements-and-b. [Accessed: 1-Jul-2024].

[11] CodeJava.net. (n.d.). "Java code example to insert data from CSV to database," CodeJava.net. [Online]. Available: https://www.codejava.net/coding/java-code-example-to-insert-data-from-csv-to-database. [Accessed: 1-Jul-2024].

[12] GeeksforGeeks. (n.d.). "Fragmentation in Distributed DBMS," GeeksforGeeks. [Online]. Available: https://www.geeksforgeeks.org/fragmentation-in-distributed-dbms/. [Accessed: 1-Jul-2024].