D2_DB

Building a simple and portable distributed database, Distributed DBMS, and analytics systems

Tasnim Khan(B00882598)
Parth Dhirubhai Gondaliya(B00893694)
Ridham Kathiriya(B00893712)
Siddharth Kharwar(B00897211)
Radhey Rupapara(B00910695)

CSCI 5408 Data Management & Warhousing Analytics

Table of Contents

Table Of Figures

Problem Statement

To build a distributed database management system with two servers that enables multiple read and write operations on data, as well as other functions such as reverse engineering, data modelling, export, and analytics.

Project Overview

In this project, we created a simple distributed database management system (D2_DB) that would run on two Google Cloud Platform Linux virtual instances. To do basic data analytics, our team created an analytics engine. Our database should be able to handle queries from two users (one user for each VM instance). The database management system layer provides a command-line interface and conducts the database management system's many functionalities, which are detailed below.

Team Strength

We believe that by working together on this project, our team displayed excellent communication abilities. Furthermore, our previous experience and background knowledge aided us in finishing this job. The characteristics and abilities contributed to the team's cohesiveness.

Individual Contribution

1. Tasnim Khan(B00882598)
   - Module 1: DB Design
   - Module 2: Query Implementation
   - Module 3: Transaction Processing

2. Parth Dhirubhai Gondaliya(B00893694)
   - Module 4: Log Management
   - Module 5: Data Modelling – Reverse Engineering

3. Ridham Kathiriya(B00893712)
   - Module 7: Analytics
   - Module 8: User Interface and Login Security

4. Siddharth Kharwar(B00897211)
   - Module 2: Query Implementation
   - Module 3: Transaction Processing

5. Radhey Rupapara(B00910695)
   - Module 6: Export Structure & Value
   - Module 5: Data Modelling – Reverse Engineering

Gannt Chart

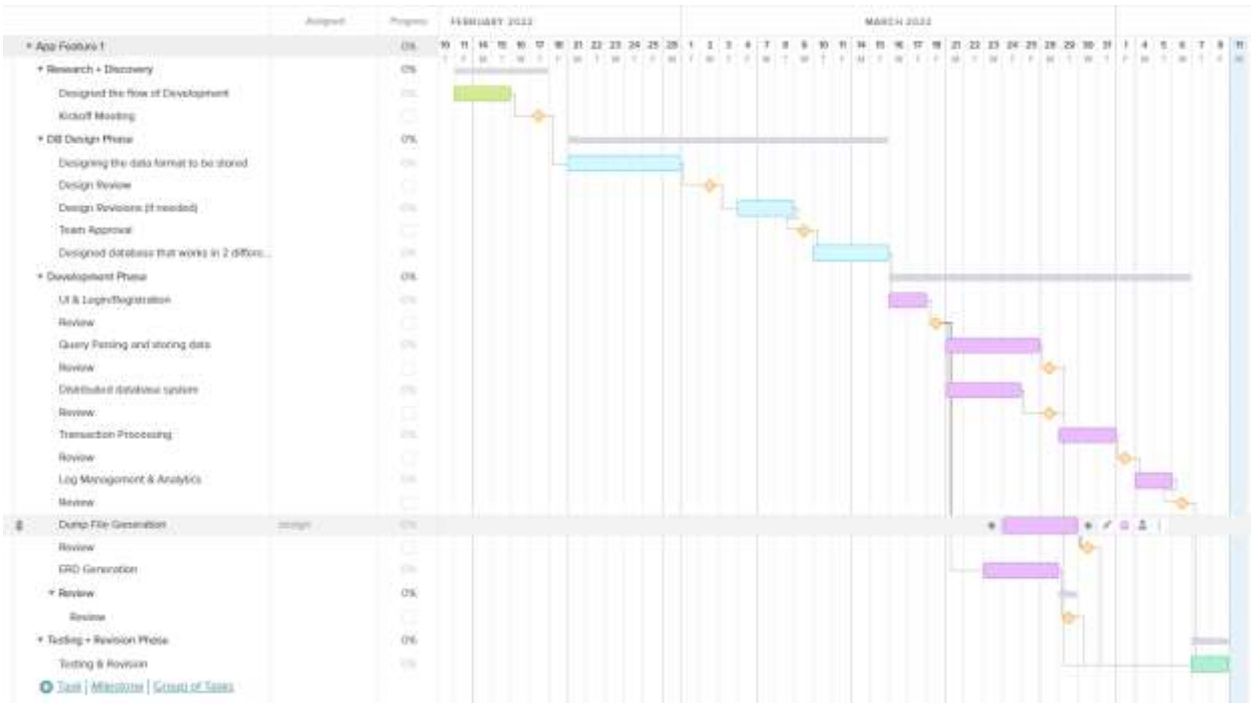
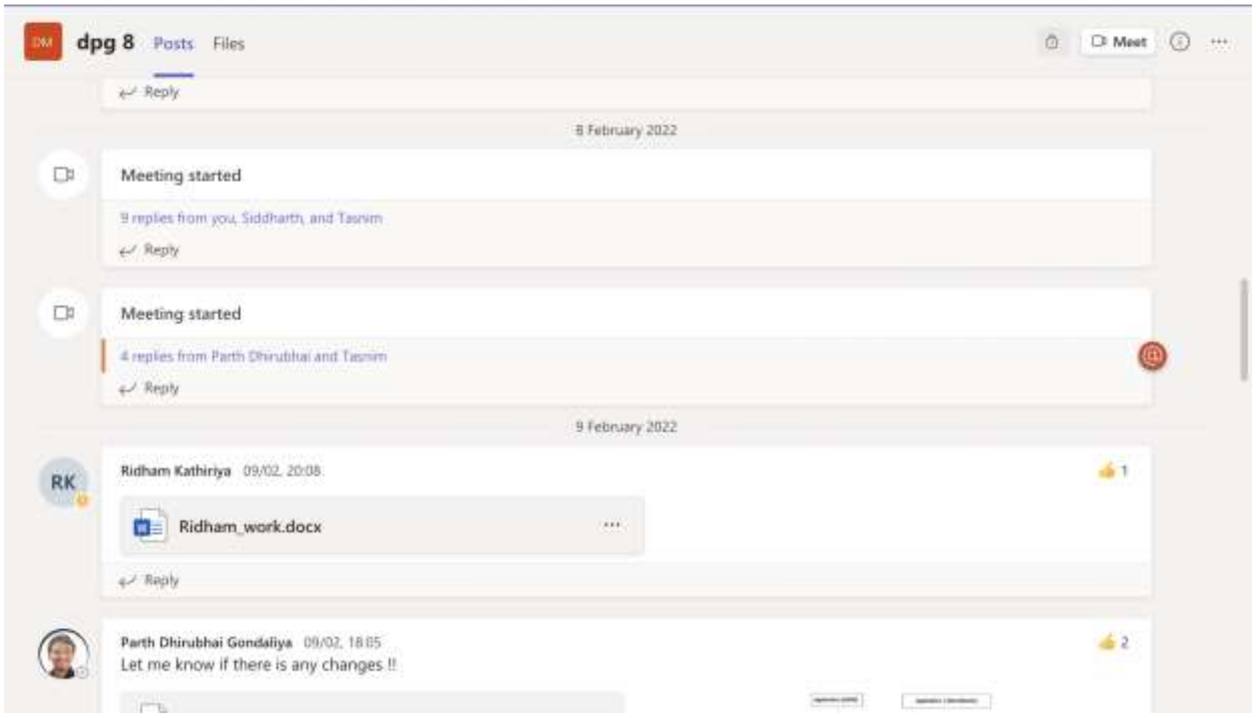


Figure 1 Gantt Chart for the Project Development

Meeting Logs


DM dpg 8 Posts Files

Meet

Reply

8 February 2022

Meeting started

9 replies from you, Siddharth, and Tasnim

Reply

Meeting started

4 replies from Parth Dhirubhai and Tasnim

Reply

9 February 2022

RK Ridham Kathiriya 09/02, 20:08

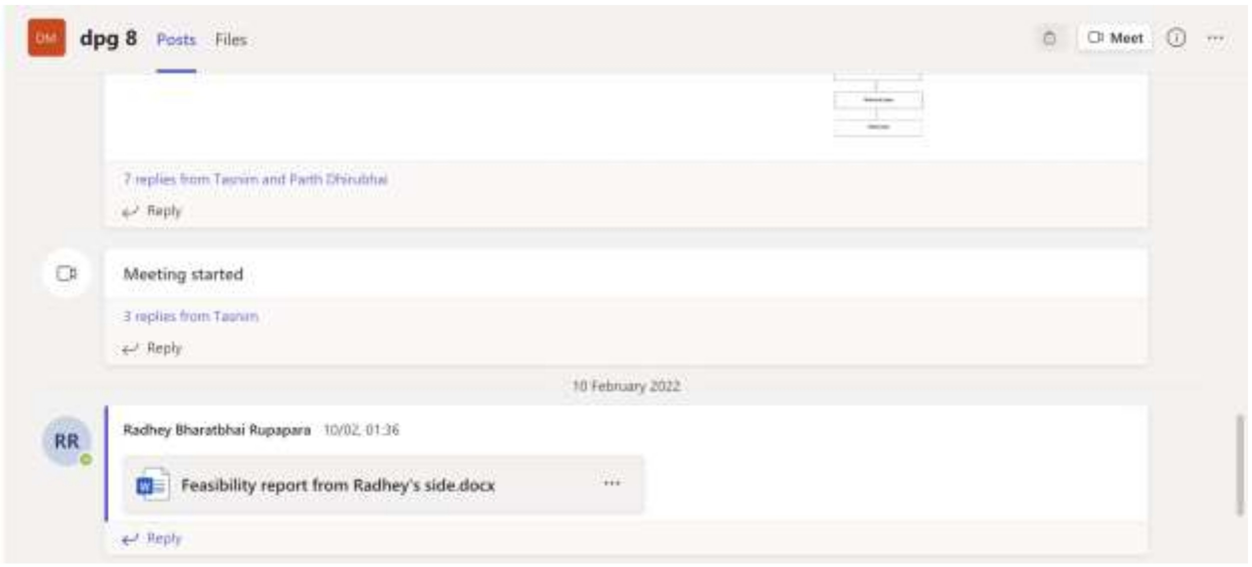
Ridham_work.docx

Reply

Parth Dhirubhai Gondaliya 09/02, 18:05

Let me know if there is any changes !!

Figure 2 Proof of meeting Logs


DM dpg 8 Posts Files

Meet

7 replies from Tasnim and Parth Dhirubhai

Reply

Meeting started

3 replies from Tasnim

Reply

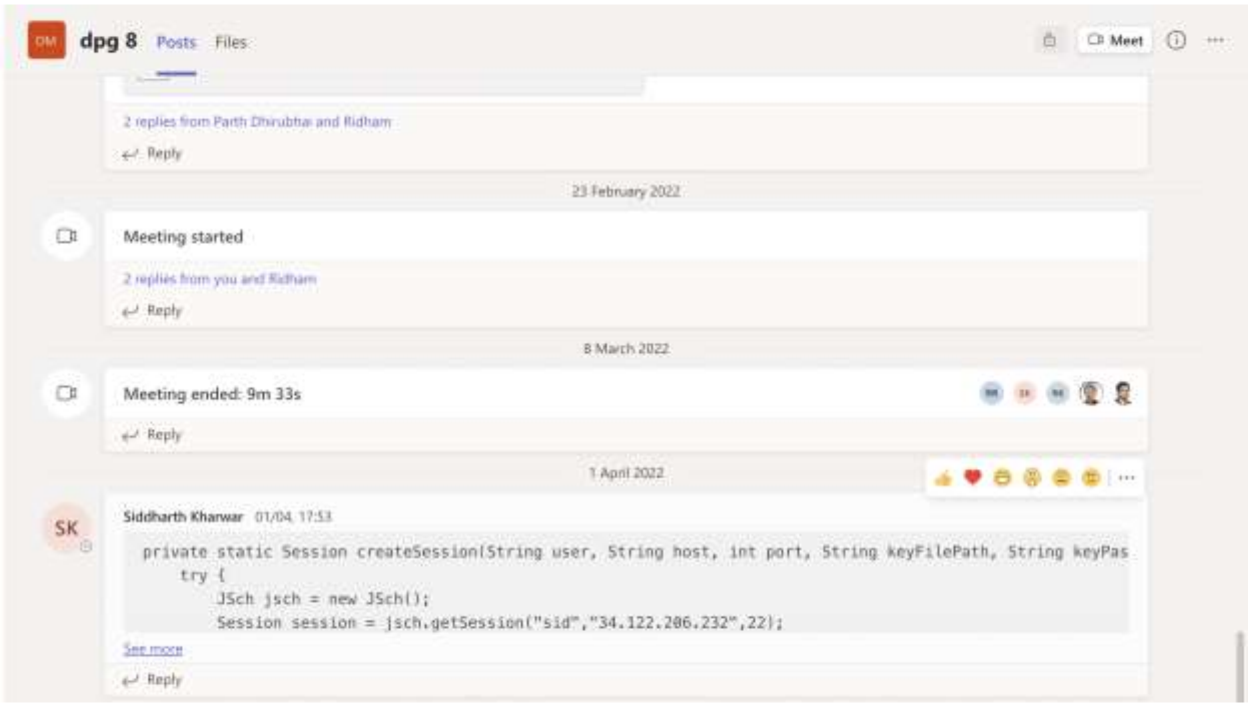
10 February 2022

RR Radhey Bharatbhai Rupapara 10/02, 01:36

Feasibility report from Radhey's side.docx

Reply

Figure 3 Proof of meeting Logs


dpg 8 Posts Files

2 replies from Parth Dhirubhai and Ridham

Reply

23 February 2022

Meeting started

2 replies from you and Ridham

Reply

8 March 2022

Meeting ended: 9m 33s

Reply

1 April 2022

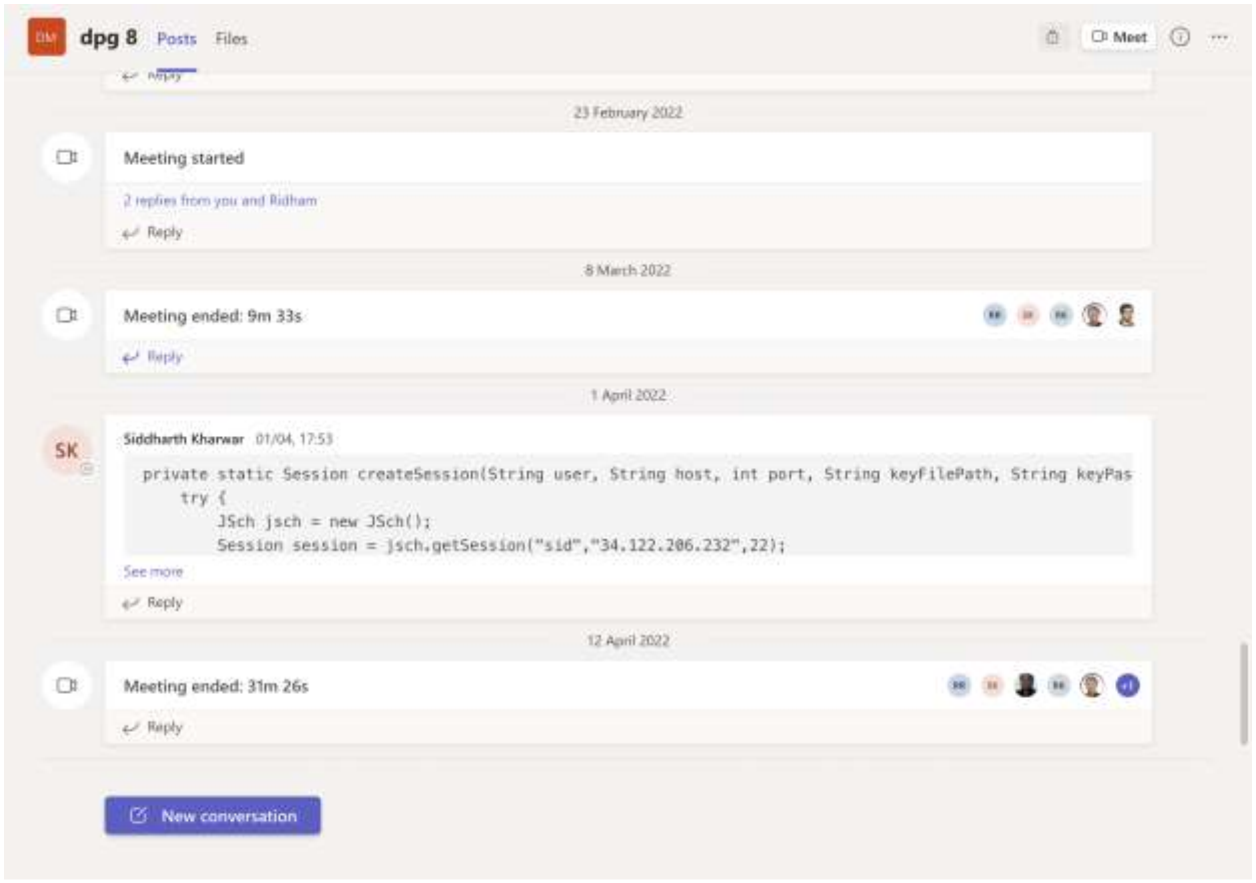
Siddharth Kharwar 01/04, 17:53

```
private static Session createSession(String user, String host, int port, String keyFilePath, String keyPas
    try {
        JSch jsch = new JSch();
        Session session = jsch.getSession("sid","34.122.206.232",22);
```

See more

Reply

Figure 4 Proof of meeting Logs


dpg 8 Posts Files

23 February 2022

Meeting started

2 replies from you and Ridham

Reply

8 March 2022

Meeting ended: 9m 33s

Reply

1 April 2022

Siddharth Kharwar 01/04, 17:53

```
private static Session createSession(String user, String host, int port, String keyFilePath, String keyPas
    try {
        JSch jsch = new JSch();
        Session session = jsch.getSession("sid","34.122.206.232",22);
```

See more

Reply

12 April 2022

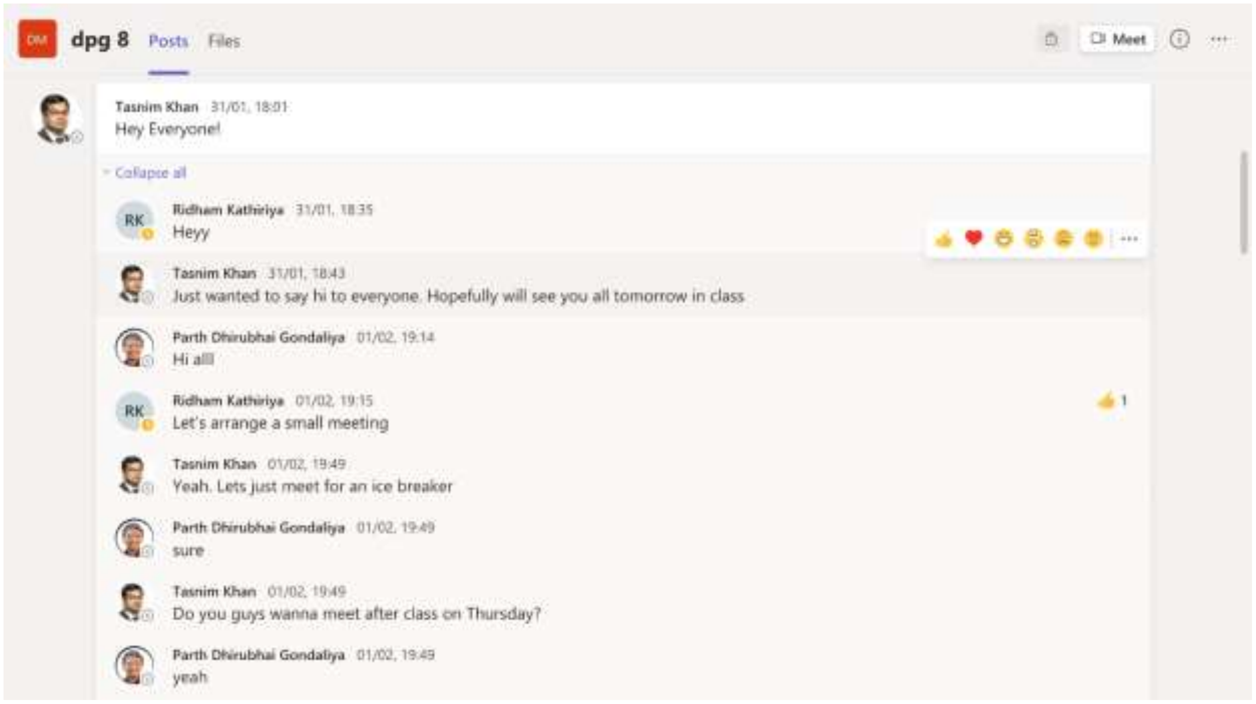
Meeting ended: 31m 26s

Reply

New conversation

Figure 5 Proof of meeting Logs

For the meeting for development, we have met online for a very few times. To Increase the communication properly between the group, we started meeting offline at university promises or at our places after all the team members arrived Halifax.



Tasnim Khan 31/01, 18:01
Hey Everyone!

Ridham Kathiriya 31/01, 18:35
Heyy

Tasnim Khan 31/01, 18:43
Just wanted to say hi to everyone. Hopefully will see you all tomorrow in class

Parth Dhirubhai Gondaliya 01/02, 19:14
Hi alll

Ridham Kathiriya 01/02, 19:15
Let's arrange a small meeting

Tasnim Khan 01/02, 19:49
Yeah. Lets just meet for an ice breaker

Parth Dhirubhai Gondaliya 01/02, 19:49
sure

Tasnim Khan 01/02, 19:49
Do you guys wanna meet after class on Thursday?

Parth Dhirubhai Gondaliya 01/02, 19:49
yeah

Figure 6 Proof of offline meeting Logs

Modules:

Module 1: DB Design

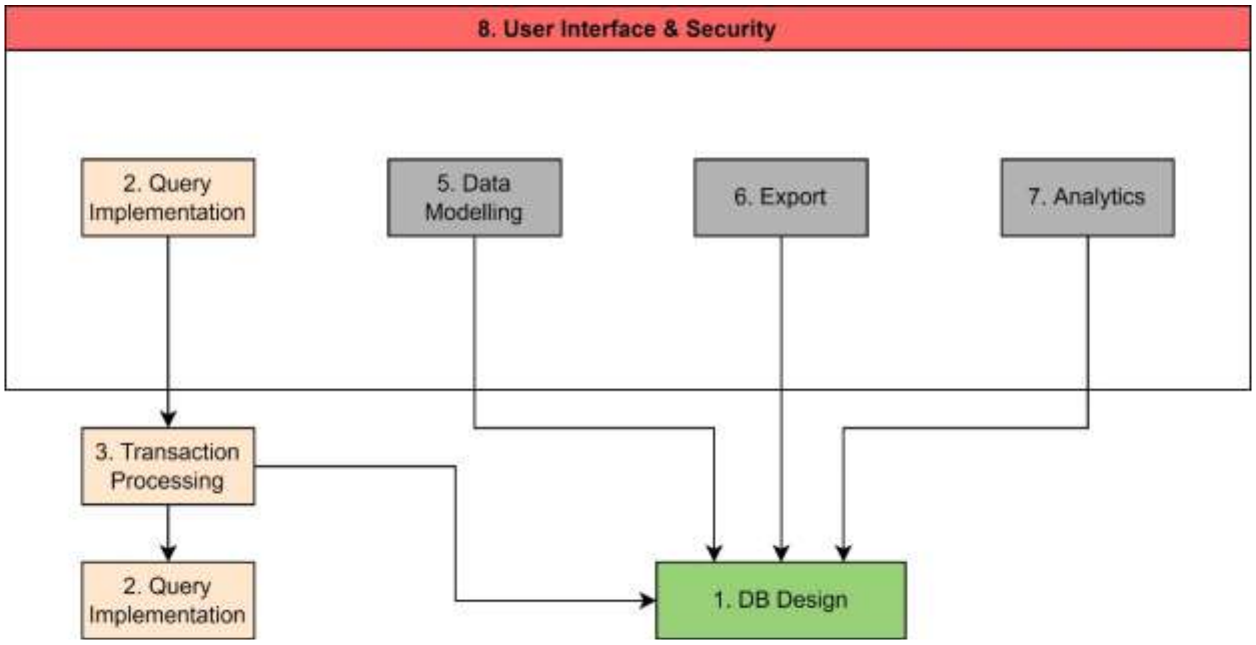
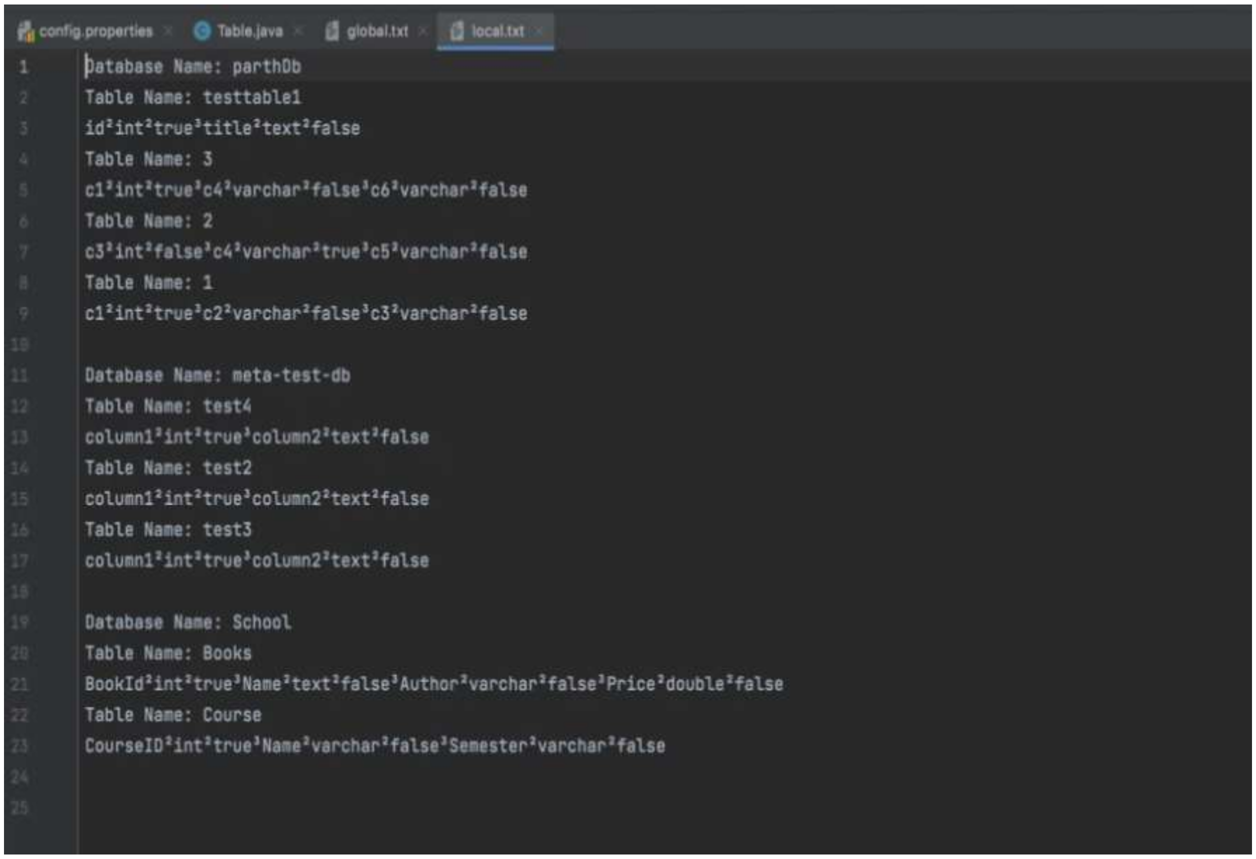


Figure 7 Database Design

Description

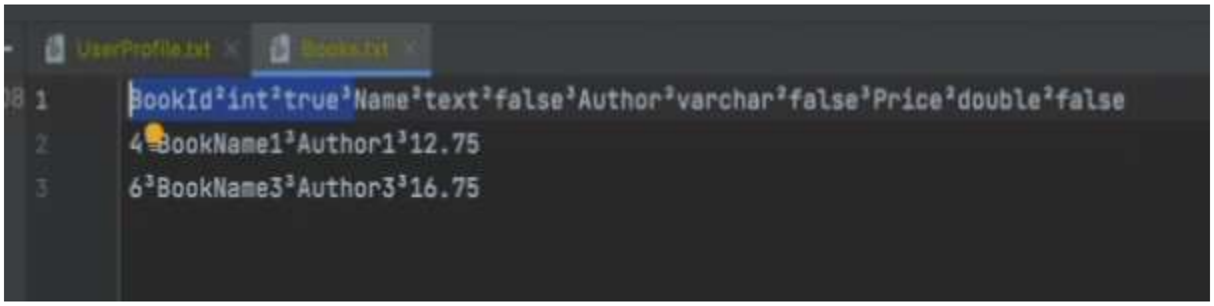
We have Designed a Horizontally Distributed Database Management System. This means that we have different servers at different places. Now, when a user creates a database, then a database will be created on both the servers and same happens with the database. When user creates a database, the database will be generated on both the servers in the selected database. But when user fires any queries to insert, update or delete the rows, then our application will get a combination of both the database, store it locally make the changes in the table and then replace the data again to the respective servers.

Our Database is designed in such a way that, whenever a table database is created, it creates a folder with the name of databases. And all the files inside that is considered as tables. Inside the tables the data is stored in the format in which columns separation are identified using superscript 3(^{3}), and the column type and column primary key separation is identified by superscript 2(^{2}).



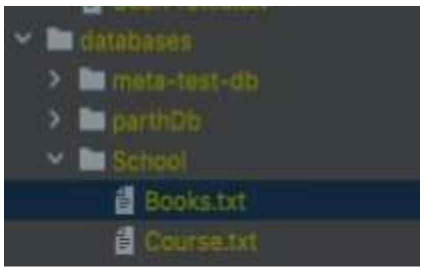
```
Database Name: parthDb
Table Name: testtable1
id²int²true³title²text²false
Table Name: 3
c1²int²true³c4²varchar²false³c6²varchar²false
Table Name: 2
c3²int²false³c4²varchar²true³c5²varchar²false
Table Name: 1
c1²int²true³c2²varchar²false³c3²varchar²false

Database Name: meta-test-db
Table Name: test4
column1²int²true³column2²text²false
Table Name: test2
column1²int²true³column2²text²false
Table Name: test3
column1²int²true³column2²text²false

Database Name: School
Table Name: Books
BookId²int²true³Name²text²false³Author²varchar²false³Price²double²false
Table Name: Course
CourseID²int²true³Name²varchar²false³Semester²varchar²false
```

Figure 8 Metadata of Databases



```
BookId²int²true³Name²text²false³Author²varchar²false³Price²double²false
4³BookName1³Author1³12.75
6³BookName3³Author3³16.75
```

Figure 9 Data Stored as a Table



```
databases
  meta-test-db
  parthDb
  School
    Books.txt
    Course.txt
```

Figure 10 Databases and Tables storage format

Module 2: Query Implementation

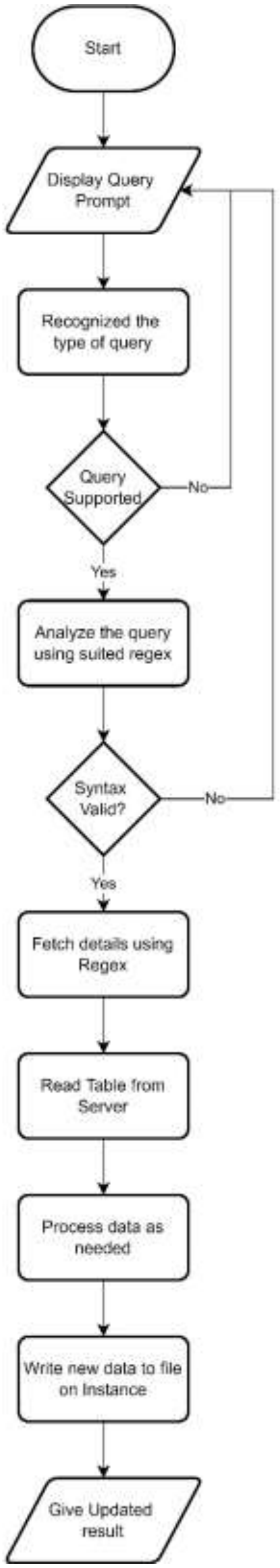


Figure 11 Query Parsing

Description

Regular expressions are used to parse the queries. The programme will first determine the type of query. And once type of query has now been determined, the programme will check the syntax and then use regular expressions to validate the query. The query would be further parsed depending on the input values and the WHERE conditions after successful validation. The necessary table must firstly be loaded into memory. The programme will run each query in memory at first. The new data will be written to the same file for persistence after the query processing is completed.

Pseudo Code

```
parseQuery() {
  switch (operation) {
      case "CREATE":
        parseCreate(query);
        break;
      case "INSERT":
        parseInsert(query);
        break;
      case "SELECT":
        parseSelect(query);
        break;
      case "USE":
        parseUse(query);
        break;
      case "DELETE":
        parseDelete(query);
        break;
      case "UPDATE":
        parseUpdate(query);
        break;
      case "START":
        parseStart(query);
```

```
            break;
        case "COMMIT":
            parseCommit(query);
            break;
        case "ROLLBACK":
            parseRollback(query);
            break;
        default:
            System.out.println("Invalid Operation.");
            break;
    }
}

parseCreate() {
    IF query is Invalid
        Print "Invalid statement"
    IF database is to be created
        createDb()
    ELSE
        parse column names, type, primary key
}

parseUse() {
    IF transaction in progress
        Print "Transaction in progress"
        Throw Exception
    set currentDb
}
```

```
parseSelect() {
   IF database is not selected
      Throw Exception

   IF query is Invalid
      Print "Invalid statement"
      Throw Exception
   parse where clause, columnNames, tableName

   IF table does not exist
      Throw Exception
   fetch local table and remote table
   merge tables
   Print mergedTable
}

parseInsert() {
   IF database is not selected
      Throw Exception

   IF query is Invalid
      Print "Invalid statement"
      Throw Exception
   parse values, tableName

   IF table does not exist
      Throw Exception
   read local and remote table
IF record cannot be inserted
        Throw Exception
    insert record
}

parseUpdate() {
    IF database is not selected
        Throw Exception

    IF query is Invalid
        Print "Invalid statement"
        Throw Exception
    parse toBeUpdated, tableName, whereClause

    IF table does not exist
        Throw Exception
    read local and remote table

    IF record found in local
        update record

    IF record found in remote
        update record
}
```

```
parseDelete() {
    IF database is not selected
        Throw Exception

    IF query is Invalid
        Print "Invalid statement"
        Throw Exception
    parse tableName, whereClause

    IF table does not exist
        Throw Exception
    read local table

    IF record found in local
        delete record

    IF record found in remote
        delete record
}

parseStart() {
    IF database is not selected
        Throw Exception

    IF query is Invalid
        Print "Invalid statement"
        Throw Exception
IF transaction in progress
        Throw Exception

    start transaction
}

parseCommit() {
    if transaction not in progress
        Throw Exception
    push local and remote tables
    delete temp tables
}

parseRollback() {
    if transaction not in progress
        Throw Exception
    delete temp tables
}
```

Module 3: Transaction Processing

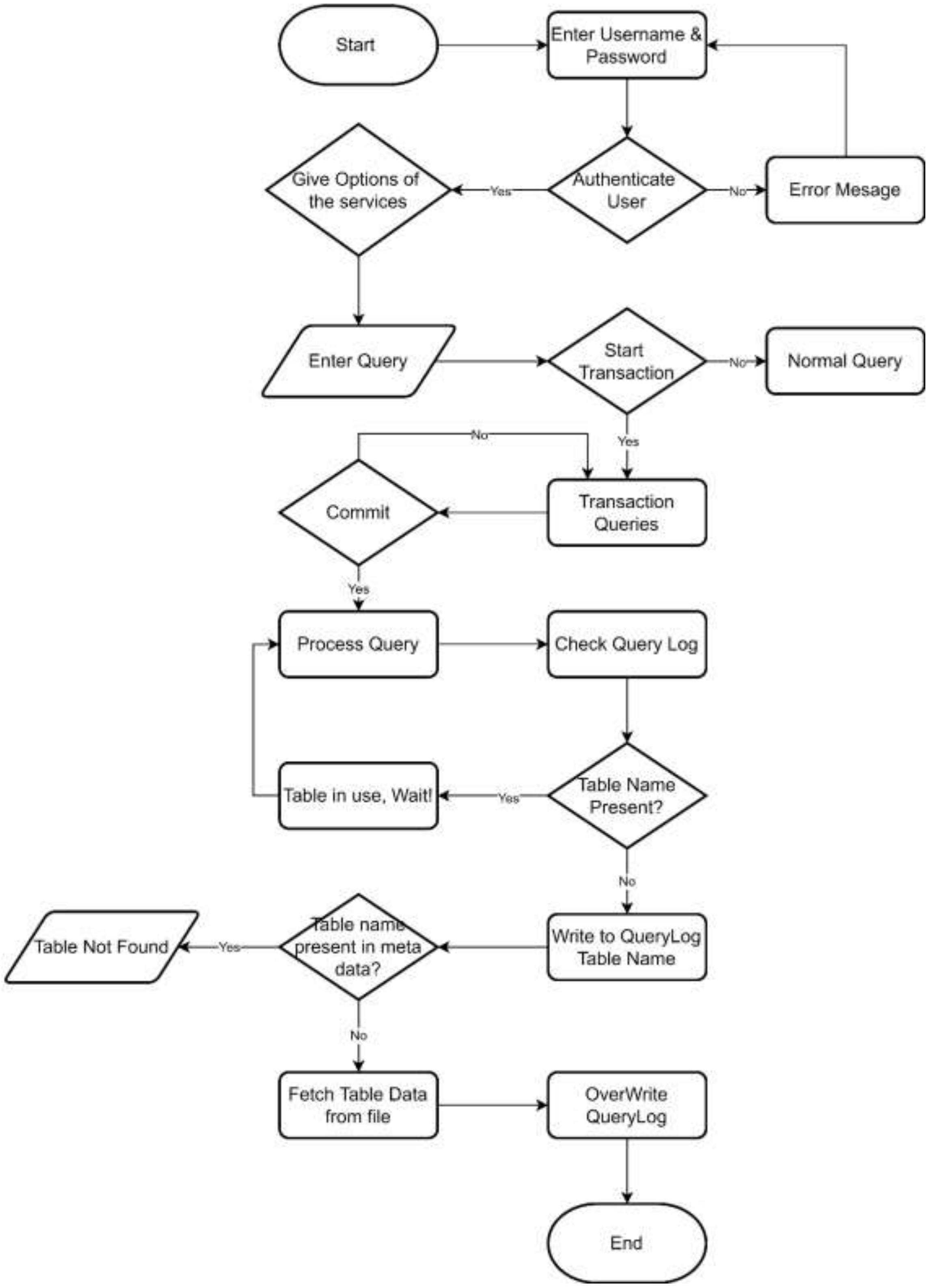


Figure 12 Transaction Processing

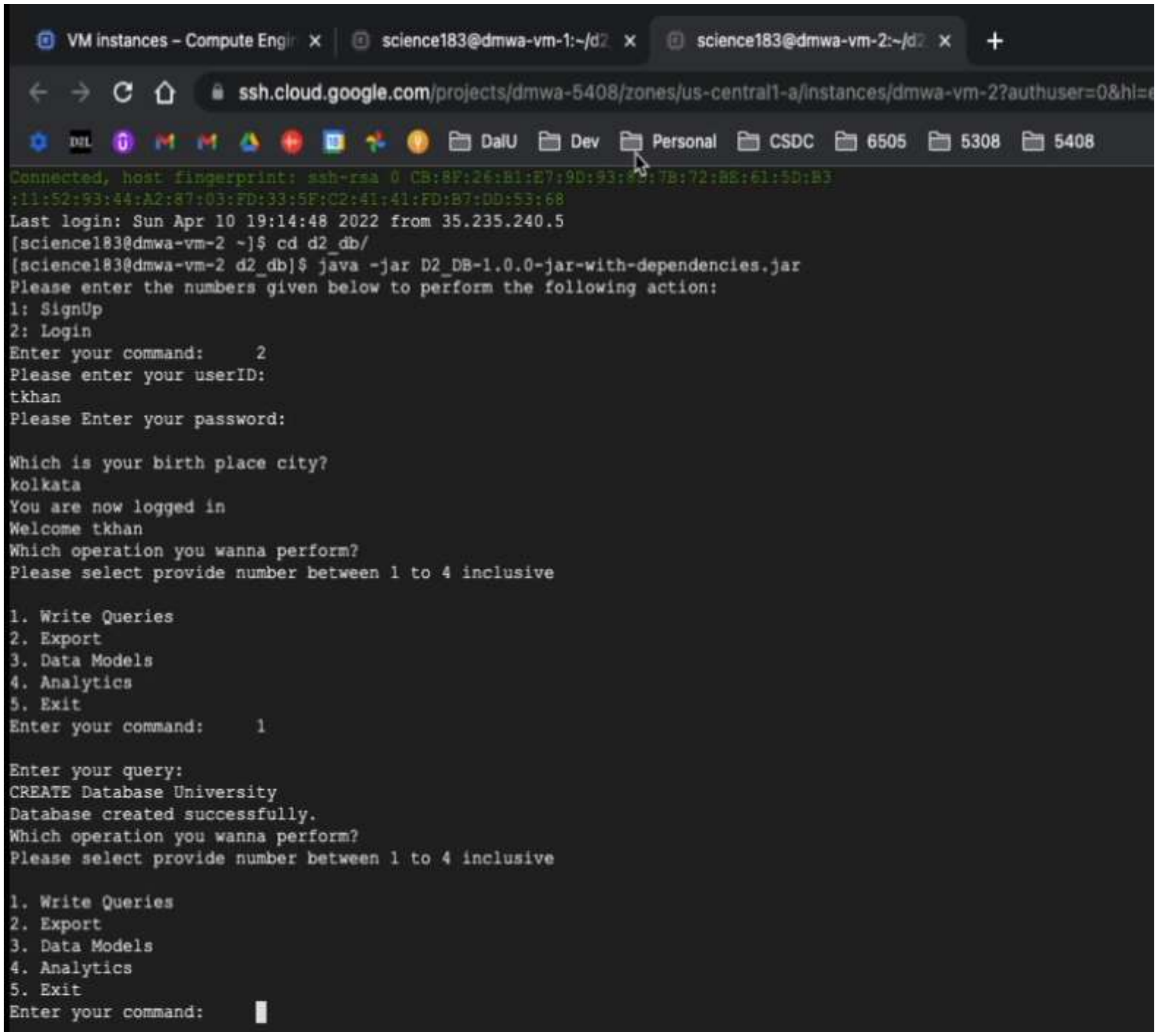
Description

Our project can distinguish between a regular query and a transaction, as indicated in the issue statement. If a user wishes to begin the transaction, he must first log in using proper credentials. Then he must select the write query option that is shown to him; as soon as he does, a new prompt with a query will appear. If a user wishes to initiate a transaction, he or she must first use the syntax "Begin Transaction;" A new prompt with transaction will appear as soon as he writes the syntax.

Until the system encounters commit, statement, the user will be free to write queries. Following the commit statement, all of the user's requests are saved in an array string and separated using a regular expression.

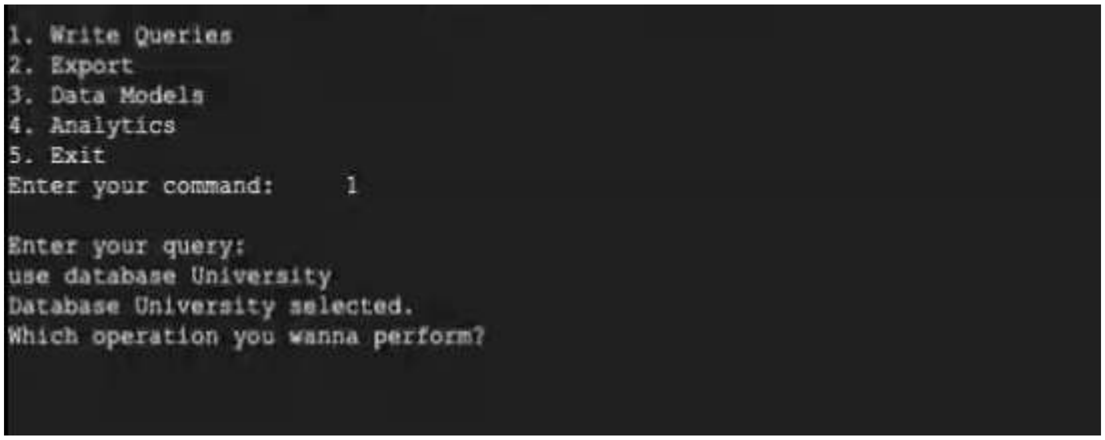
Our project adheres to the ACID characteristics, which are required for data consistency and accuracy. For the record of tables that are now in use, a log file is kept. That is, before completing every transaction, our project will check to see if the table the user desires is already in use. If another user is using the table, the table in use wait will be printed. As a result, no more than one user may utilise the same table at the same time.

Test Cases



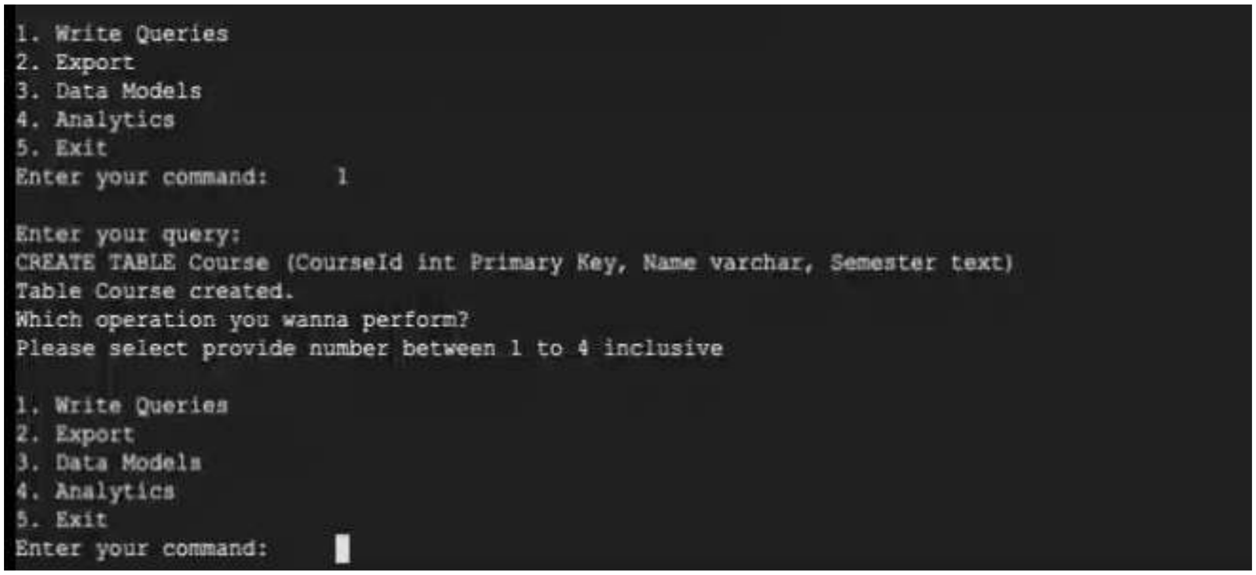
```
Connected, host fingerprint: ssh-rsa 0 CB:8F:26:B1:E7:9D:93:9B:7B:72:BE:61:5D:B3
:11:52:93:44:A2:87:03:FD:33:5F:C2:41:41:FD:B7:DD:53:68
Last login: Sun Apr 10 19:14:48 2022 from 35.235.240.5
[science183@dmwa-vm-2 ~]$ cd d2_db/
[science183@dmwa-vm-2 d2_db]$ java -jar D2_DB-1.0.0-jar-with-dependencies.jar
Please enter the numbers given below to perform the following action:
1: SignUp
2: Login
Enter your command:     2
Please enter your userID:
tkhan
Please Enter your password:

Which is your birth place city?
kolkata
You are now logged in
Welcome tkhan
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
CREATE Database University
Database created successfully.
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:
```

Figure 13 Create Database



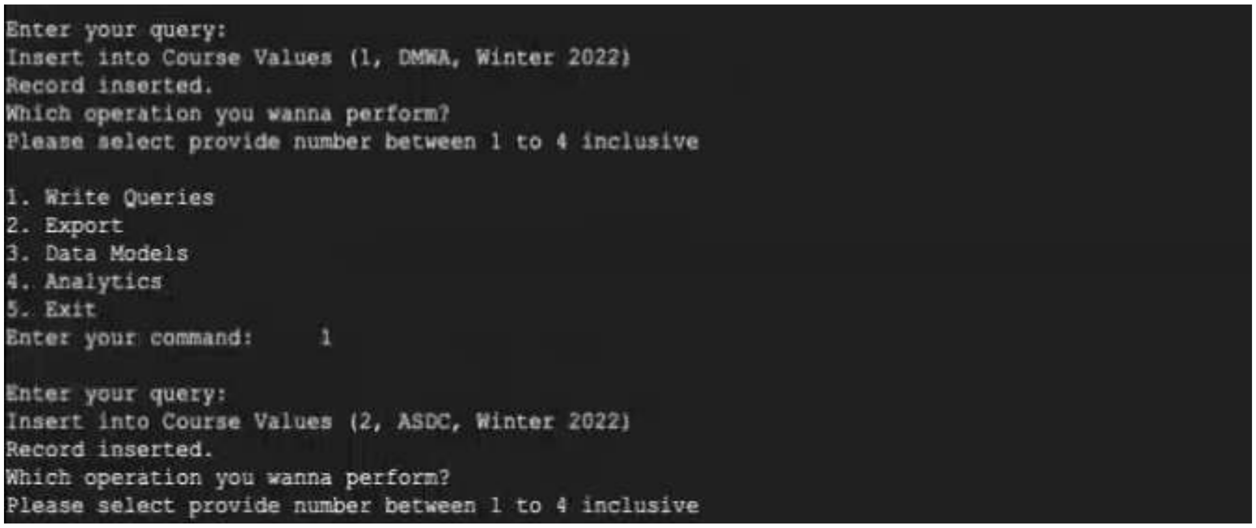
```
1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
use database University
Database University selected.
Which operation you wanna perform?
```

Figure 14 Use Database Query



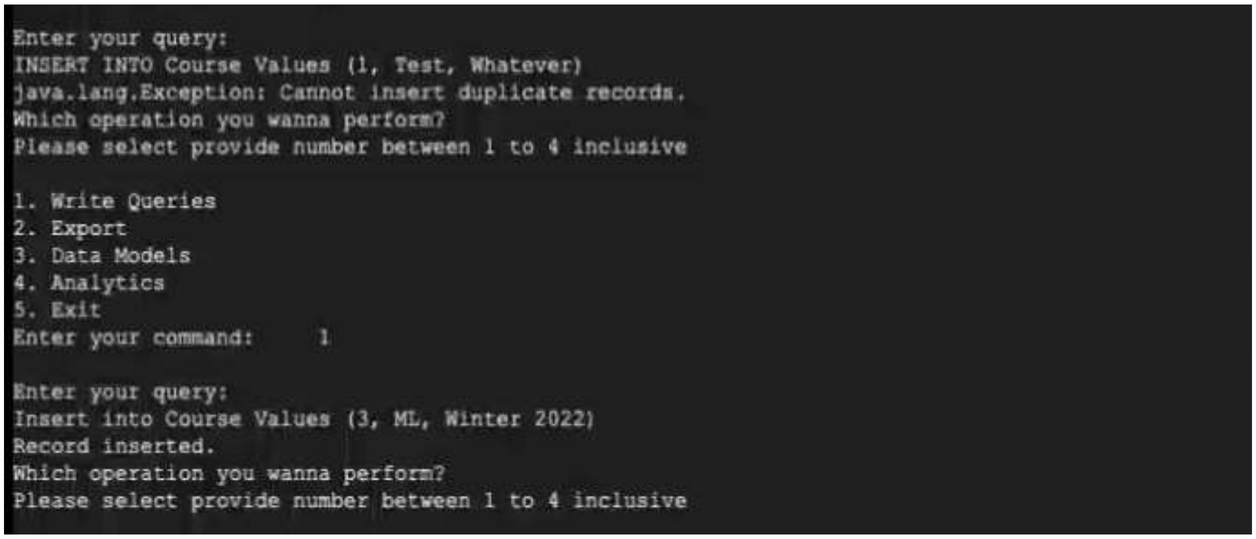
```
1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
CREATE TABLE Course (CourseId int Primary Key, Name varchar, Semester text)
Table Course created.
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:
```

Figure 15 Create Table Query



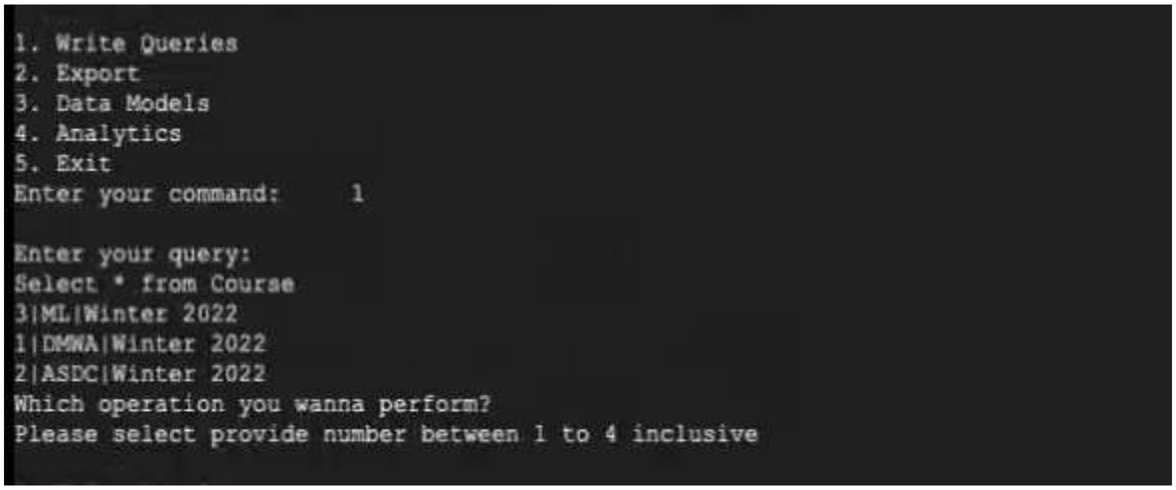
```
Enter your query:
Insert into Course Values (1, DMWA, Winter 2022)
Record inserted.
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Insert into Course Values (2, ASDC, Winter 2022)
Record inserted.
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 16 Insert Row Query in VM2



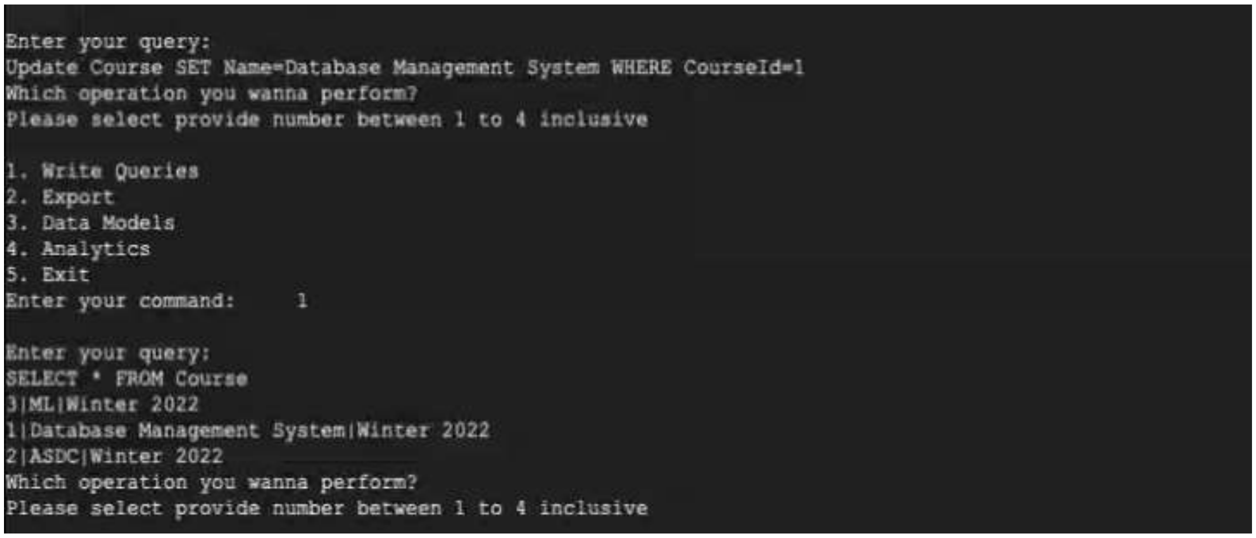
```
Enter your query:
INSERT INTO Course Values (1, Test, Whatever)
java.lang.Exception: Cannot insert duplicate records.
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Insert into Course Values (3, ML, Winter 2022)
Record inserted.
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 17 Insert Query in VM1



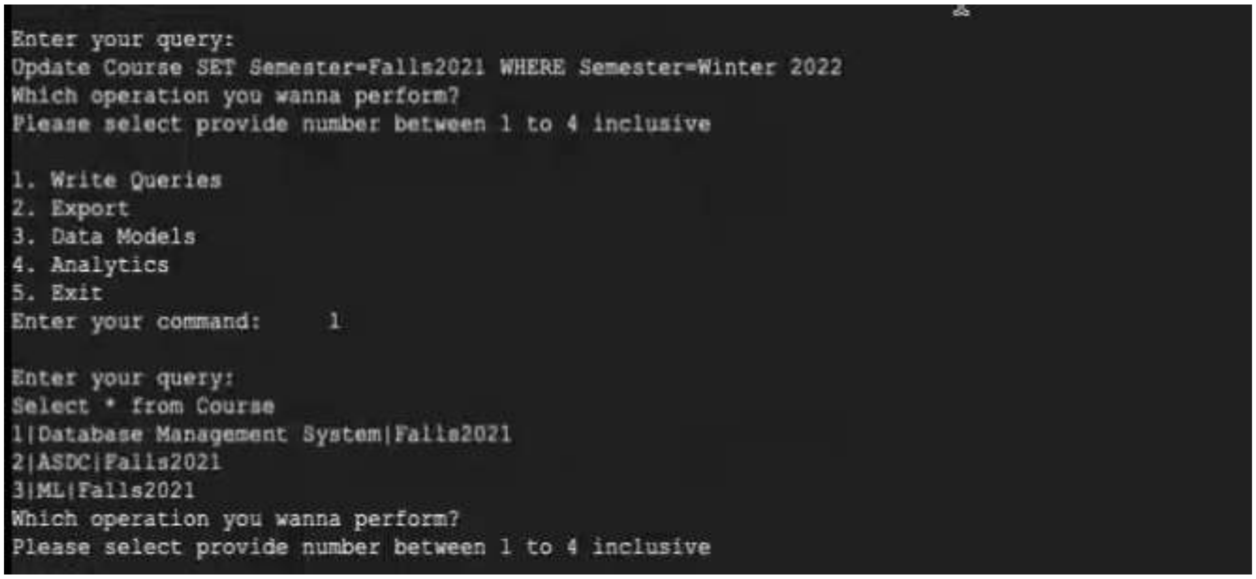
```
1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Select * from Course
3|ML|Winter 2022
1|DMWA|Winter 2022
2|ASDC|Winter 2022
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 18 Select Query in VM1



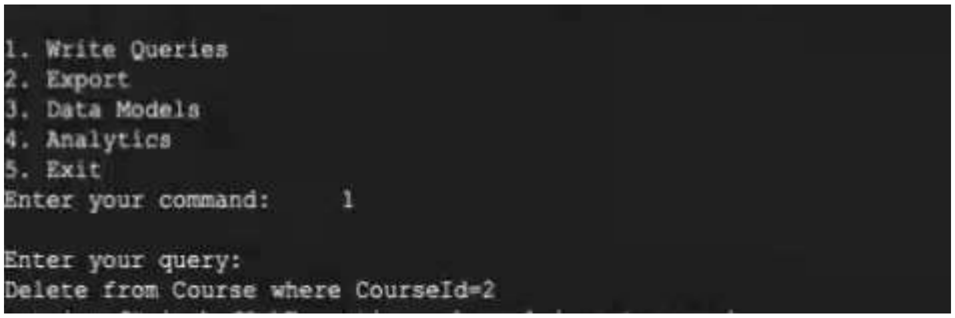
```
Enter your query:
Update Course SET Name=Database Management System WHERE CourseId=1
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
SELECT * FROM Course
3|ML|Winter 2022
1|Database Management System|Winter 2022
2|ASDC|Winter 2022
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 19 Update Query in VM1



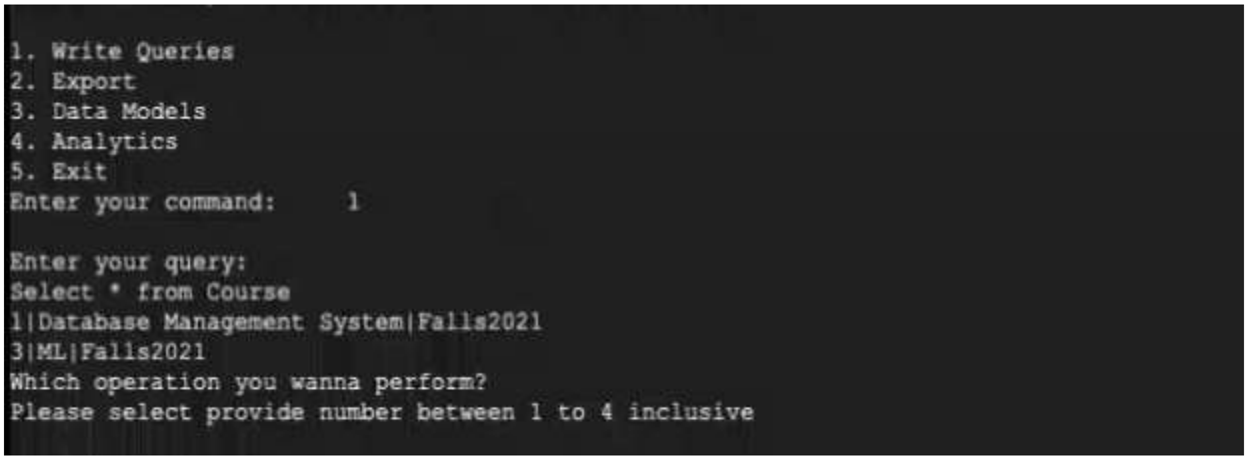
```
Enter your query:
Update Course SET Semester=Falls2021 WHERE Semester=Winter 2022
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Select * from Course
1|Database Management System|Falls2021
2|ASDC|Falls2021
3|ML|Falls2021
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 20 Update Query in VM2



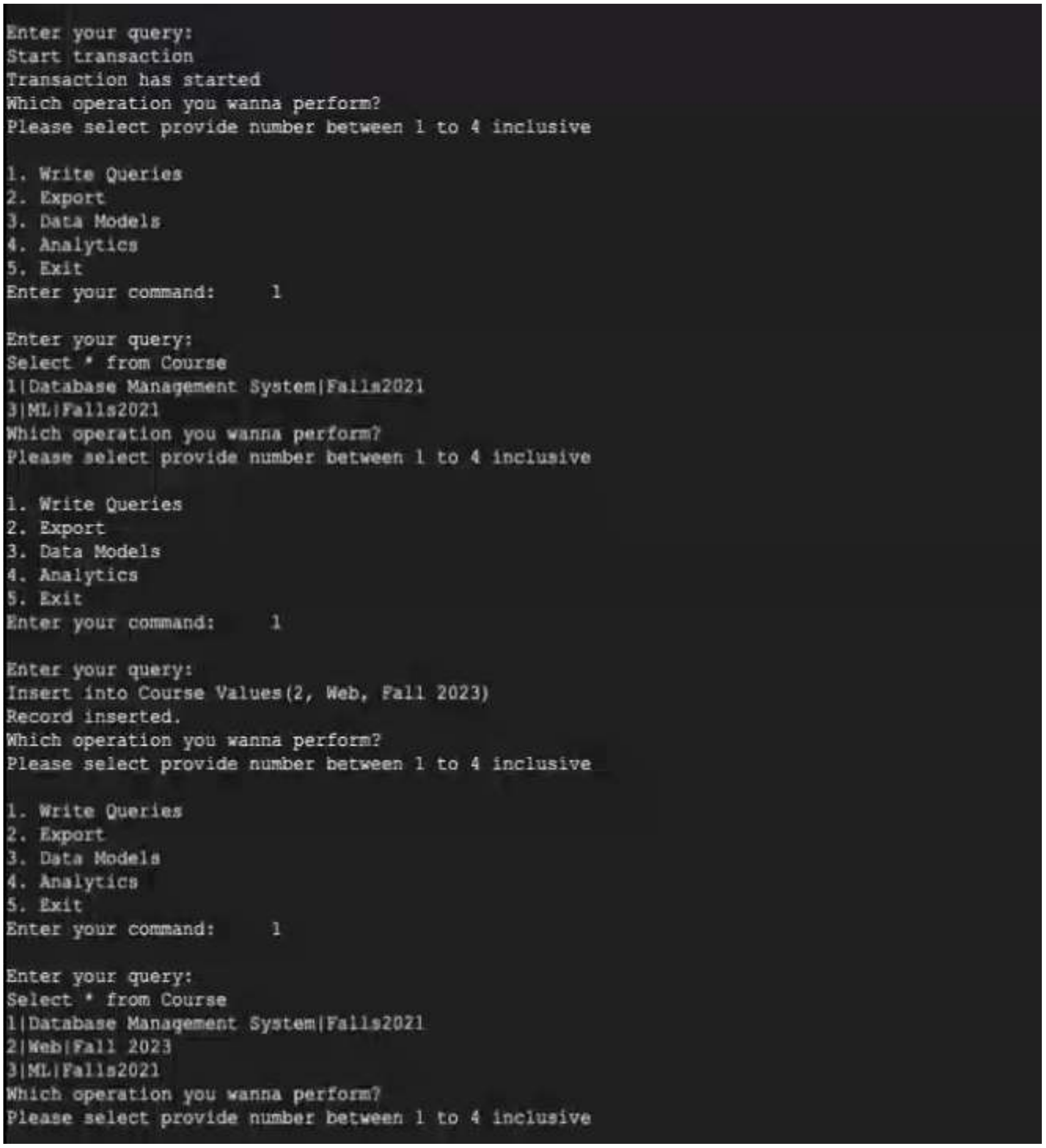
```
1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Delete from Course where CourseId=2
```

Figure 21 Delete Query in VM2



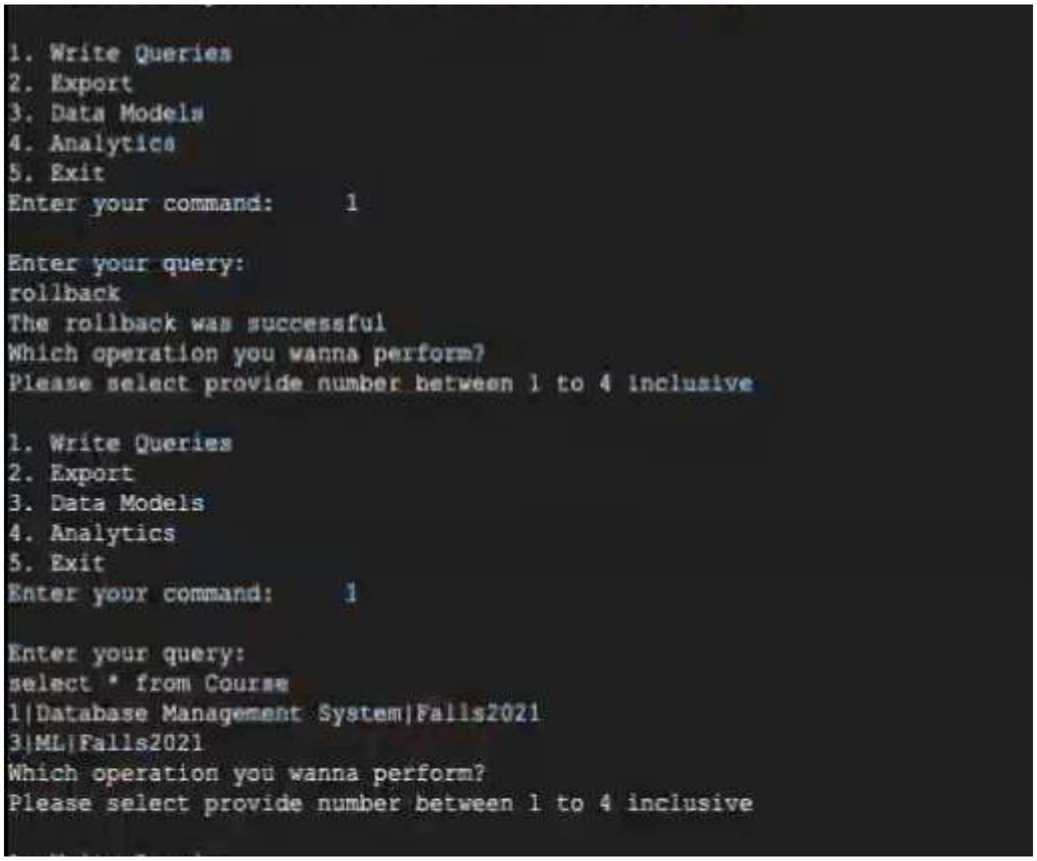
```
1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Select * from Course
1|Database Management System|Falls2021
3|ML|Falls2021
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 22 Result of Delete Query



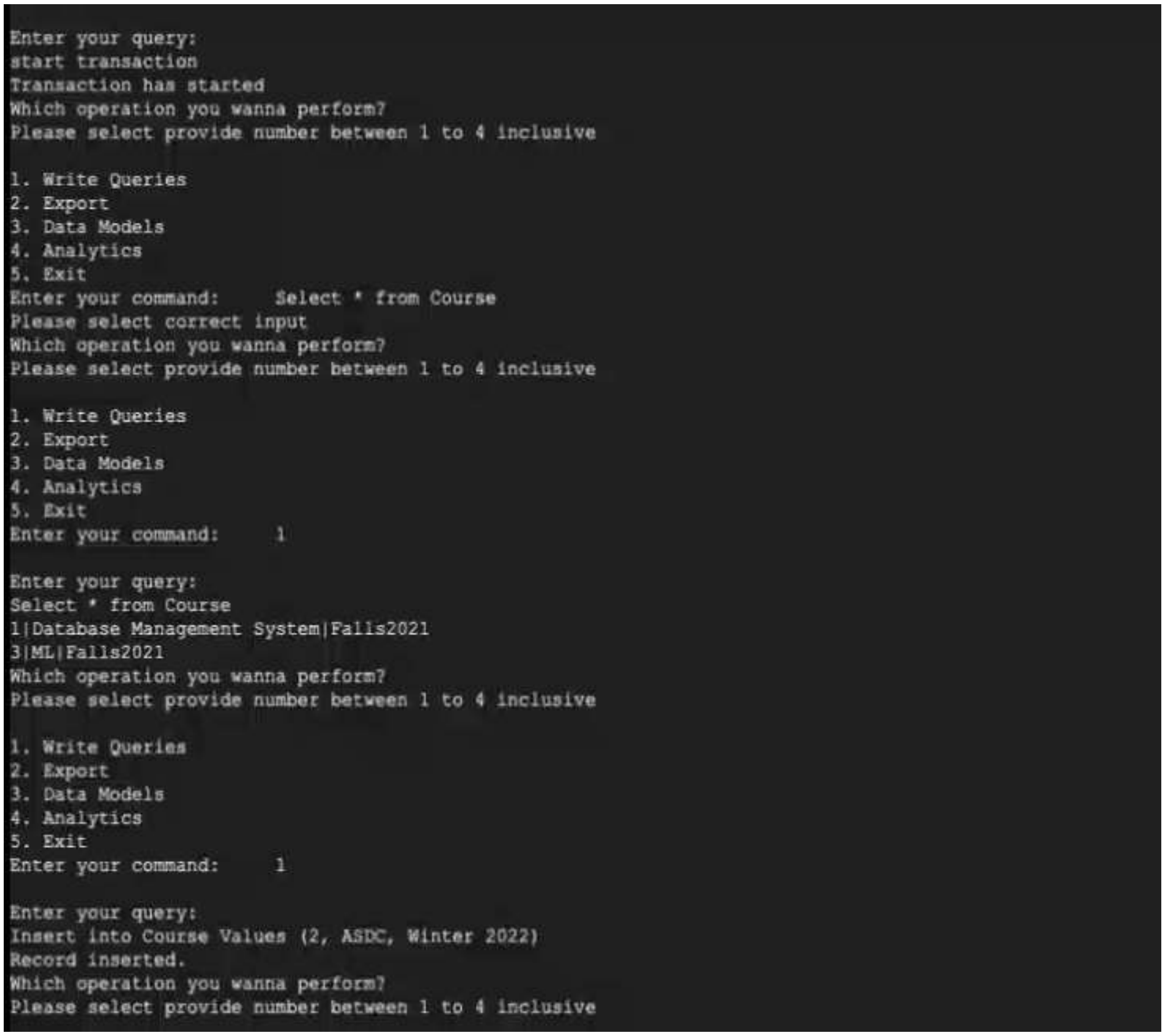
```
Enter your query:
Start transaction
Transaction has started
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Select * from Course
1|Database Management System|Falls2021
3|ML|Falls2021
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Insert into Course Values(2, Web, Fall 2023)
Record inserted.
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Select * from Course
1|Database Management System|Falls2021
2|Web|Fall 2023
3|ML|Falls2021
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 23 Transaction (Rollback)



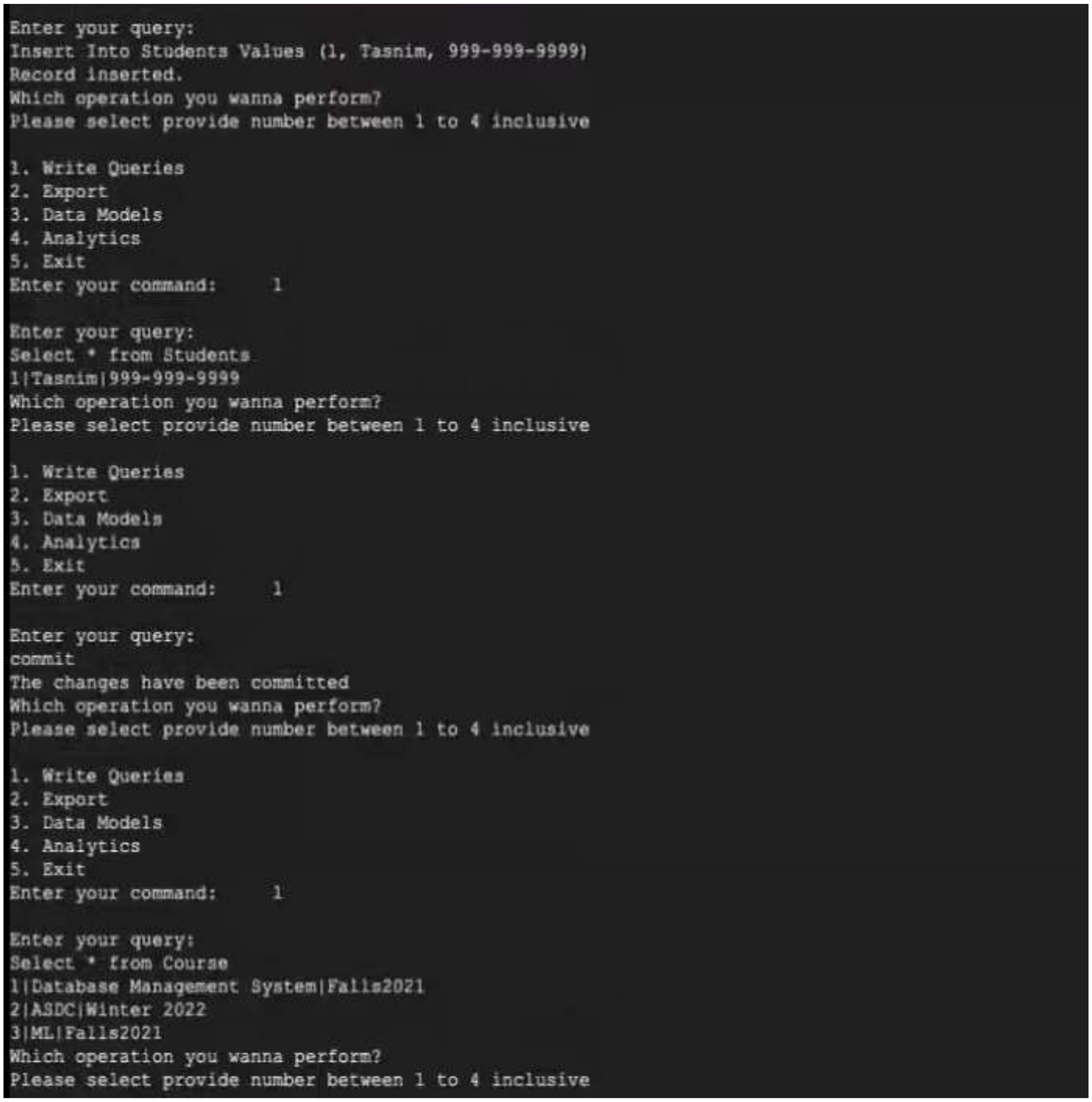
```
1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
rollback
The rollback was successful
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
select * from Course
1|Database Management System|Falls2021
3|ML|Falls2021
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 24 Transaction (Rollback)



```
Enter your query:
start transaction
Transaction has started
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     Select * from Course
Please select correct input
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Select * from Course
1|Database Management System|Falls2021
3|ML|Falls2021
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Insert into Course Values (2, ASDC, Winter 2022)
Record inserted.
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 25 Transaction (Committed)



```
Enter your query:
Insert Into Students Values (1, Tasnim, 999-999-9999)
Record inserted.
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Select * from Students
1|Tasnim|999-999-9999
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
commit
The changes have been committed
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     1

Enter your query:
Select * from Course
1|Database Management System|Falls2021
2|ASDC|Winter 2022
3|ML|Falls2021
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 26 Transaction (Committed)

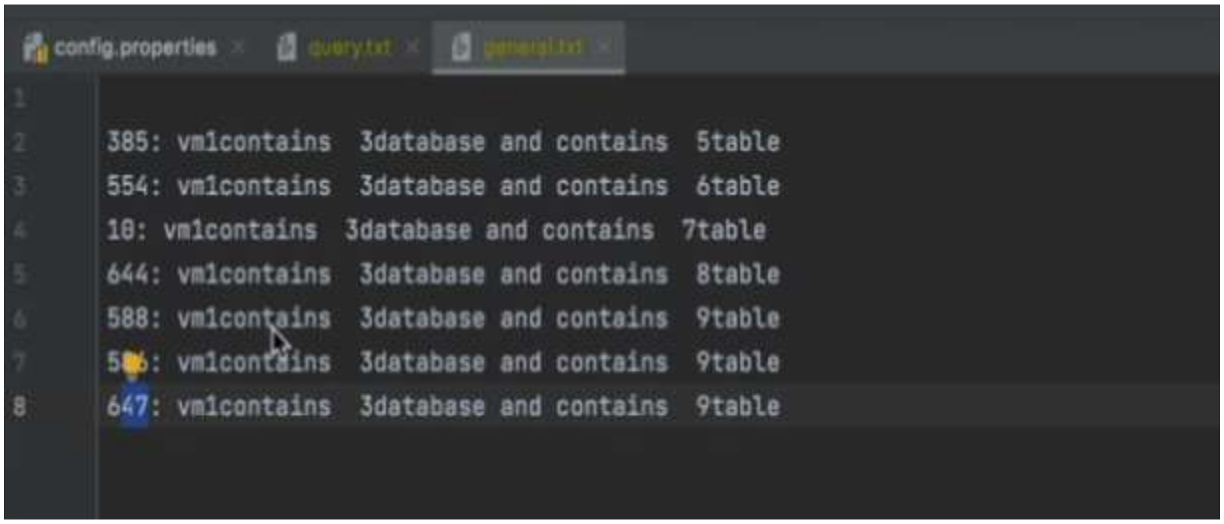
Module 4: Log Management

Description

All of the aforementioned processes create logs, which are saved in text format. General logs, event logs, and query logs are the three sorts of logs that are generated.

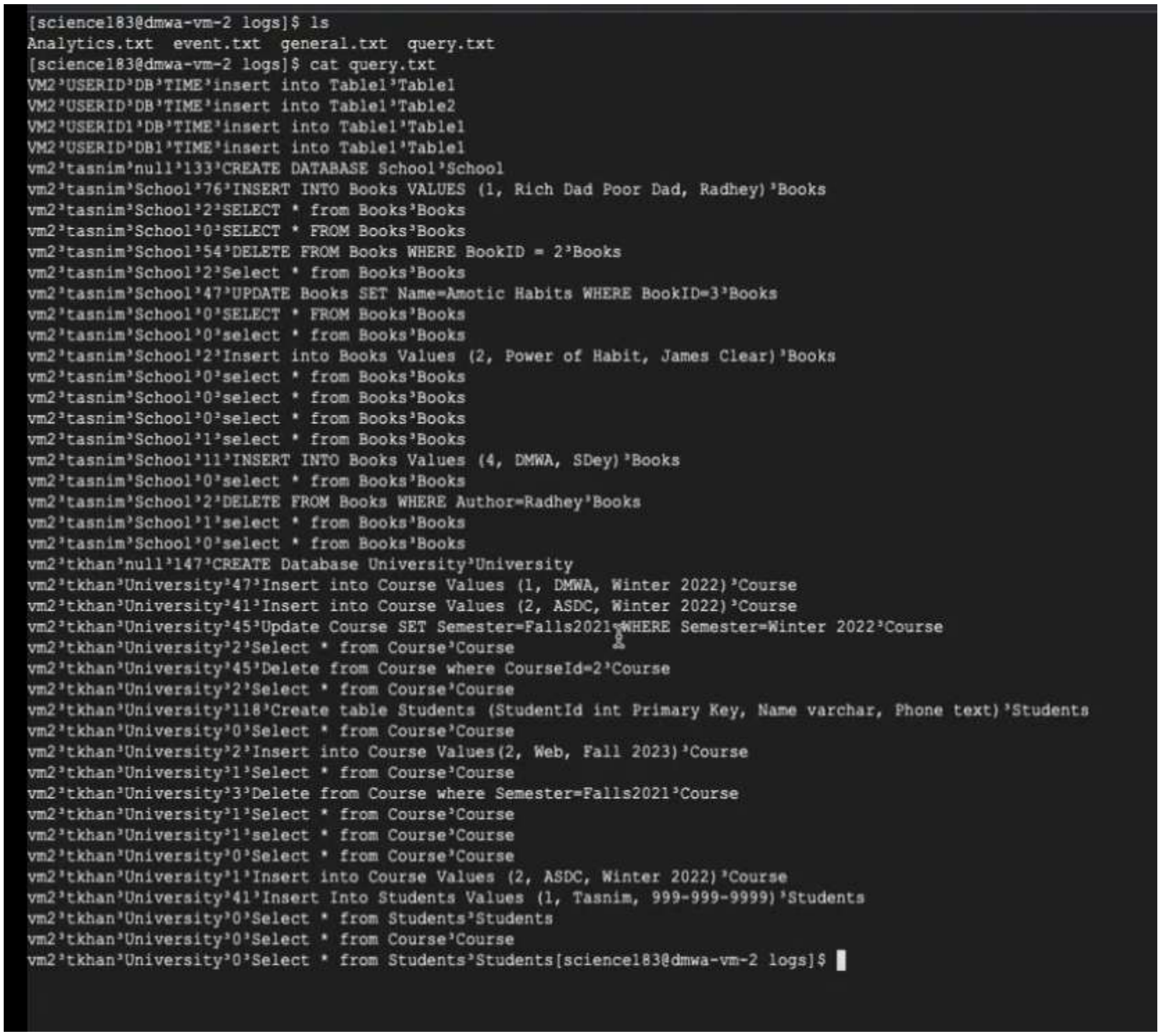
Test Cases

For Testing purpose, we executed some queries and the logs were as following:



```

385: vm1contains  3database and contains  5table
554: vm1contains  3database and contains  6table
18: vm1contains  3database and contains  7table
644: vm1contains  3database and contains  8table
588: vm1contains  3database and contains  9table
5[illegible]6: vm1contains  3database and contains  9table
647: vm1contains  3database and contains  9table
```

Figure 27 Logs Generated in a txt format



```
[science183@dmwa-vm-2 logs]$ ls
Analytics.txt  event.txt  general.txt  query.txt
[science183@dmwa-vm-2 logs]$ cat query.txt
VM2³USERID³DB³TIME³insert into Table1³Table1
VM2³USERID³DB³TIME³insert into Table1³Table2
VM2³USERID1³DB³TIME³insert into Table1³Table1
VM2³USERID³DB1³TIME³insert into Table1³Table1
vm2³tasnim³null³133³CREATE DATABASE School³School
vm2³tasnim³School³76³INSERT INTO Books VALUES (1, Rich Dad Poor Dad, Radhey)³Books
vm2³tasnim³School³2³SELECT * from Books³Books
vm2³tasnim³School³0³SELECT * FROM Books³Books
vm2³tasnim³School³54³DELETE FROM Books WHERE BookID = 2³Books
vm2³tasnim³School³2³Select * from Books³Books
vm2³tasnim³School³47³UPDATE Books SET Name=Amotic Habits WHERE BookID=3³Books
vm2³tasnim³School³0³SELECT * FROM Books³Books
vm2³tasnim³School³0³select * from Books³Books
vm2³tasnim³School³2³Insert into Books Values (2, Power of Habit, James Clear)³Books
vm2³tasnim³School³0³select * from Books³Books
vm2³tasnim³School³0³select * from Books³Books
vm2³tasnim³School³0³select * from Books³Books
vm2³tasnim³School³1³select * from Books³Books
vm2³tasnim³School³11³INSERT INTO Books Values (4, DMWA, SDey)³Books
vm2³tasnim³School³0³select * from Books³Books
vm2³tasnim³School³2³DELETE FROM Books WHERE Author=Radhey³Books
vm2³tasnim³School³1³select * from Books³Books
vm2³tasnim³School³0³select * from Books³Books
vm2³tkhan³null³147³CREATE Database University³University
vm2³tkhan³University³47³Insert into Course Values (1, DMWA, Winter 2022)³Course
vm2³tkhan³University³41³Insert into Course Values (2, ASDC, Winter 2022)³Course
vm2³tkhan³University³45³Update Course SET Semester=Falls2021³WHERE Semester=Winter 2022³Course
vm2³tkhan³University³2³Select * from Course³Course
vm2³tkhan³University³45³Delete from Course where CourseId=2³Course
vm2³tkhan³University³2³Select * from Course³Course
vm2³tkhan³University³118³Create table Students (StudentId int Primary Key, Name varchar, Phone text)³Students
vm2³tkhan³University³0³Select * from Course³Course
vm2³tkhan³University³2³Insert into Course Values(2, Web, Fall 2023)³Course
vm2³tkhan³University³1³Select * from Course³Course
vm2³tkhan³University³3³Delete from Course where Semester=Falls2021³Course
vm2³tkhan³University³1³Select * from Course³Course
vm2³tkhan³University³1³select * from Course³Course
vm2³tkhan³University³0³Select * from Course³Course
vm2³tkhan³University³1³Insert into Course Values (2, ASDC, Winter 2022)³Course
vm2³tkhan³University³41³Insert Into Students Values (1, Tasnim, 999-999-9999)³Students
vm2³tkhan³University³0³Select * from Students³Students
vm2³tkhan³University³0³Select * from Course³Course
vm2³tkhan³University³0³Select * from Students³Students[science183@dmwa-vm-2 logs]$ █
```

Figure 28 Log In VM2

Module 5: Data Modelling – Reverse Engineering

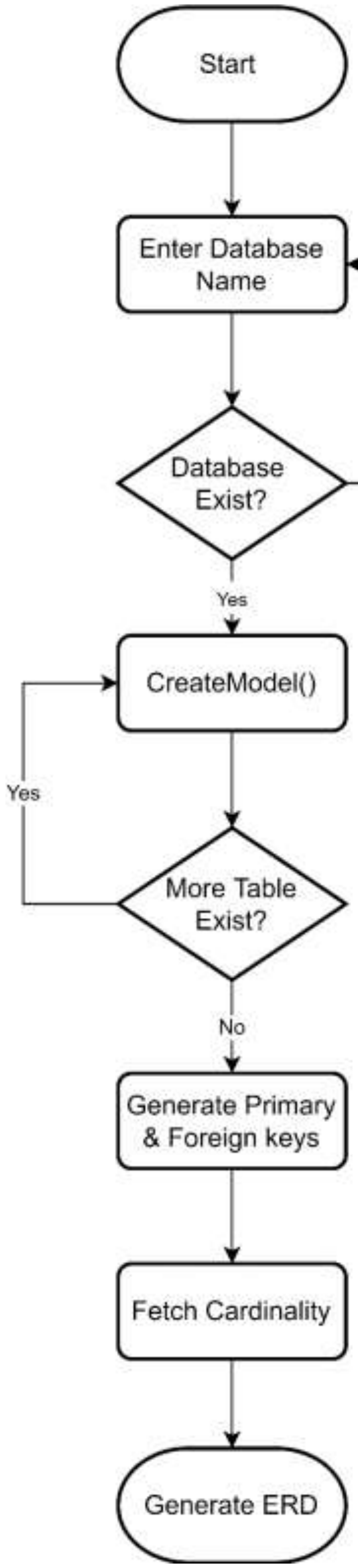
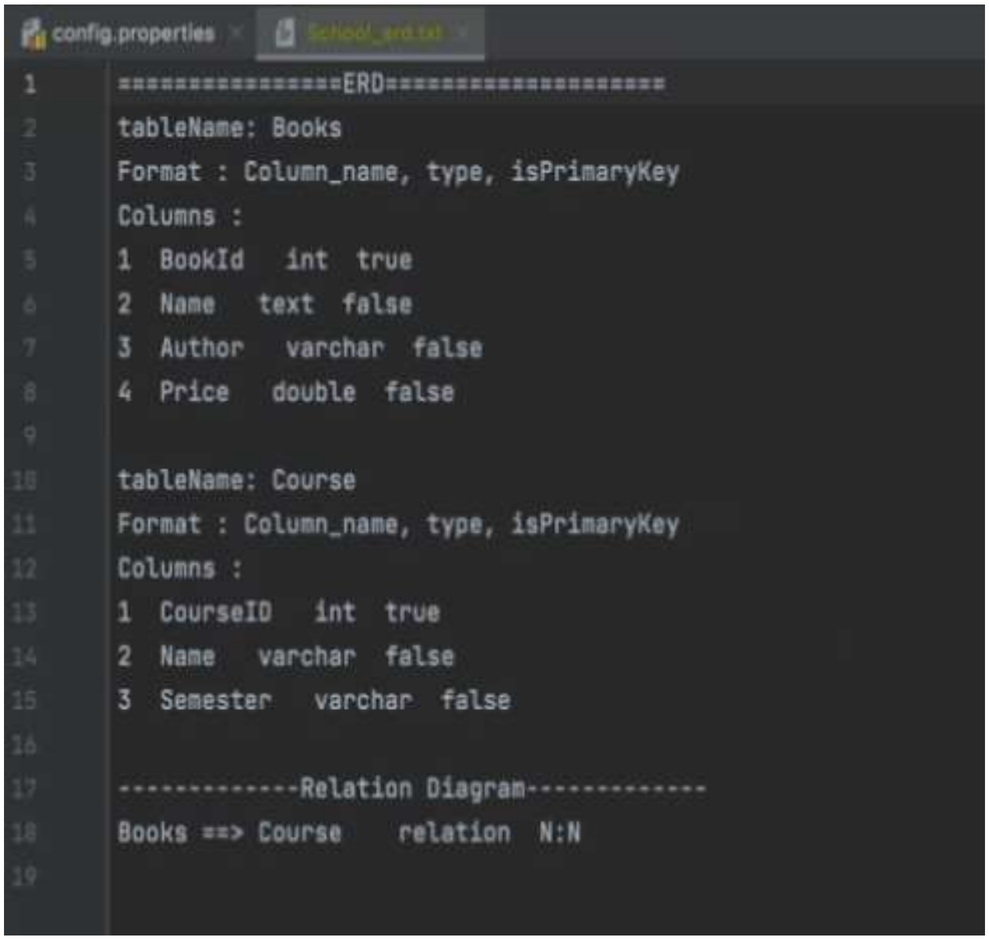


Figure 29 Reverse Engineering (ERD Generation)

Description

When the user types in a database name, the module's logic checks to see whether one exists. If the check succeeds, the createModel() function is called.
The code for the requested database goes to the desired database's file directory and retrieves a list of all the tables in the database.
After obtaining a list of tables, code refers to the tables' metadata and data files. We execute a loop for each table and implement the logic of Primary Keys, which aids in the generation of a HashMap of the data.
Finally, we will receive a HashMap with a key equal to the Column name and a value equal to the tables where the key is a Primary Key or Foreign Key. The code then proceeds to produce the final text file, which contains Table names, Columns, Primary Keys, etc. after we obtain the Primary Keys.

Test Cases



```
================ERD====================
tableName: Books
Format : Column_name, type, isPrimaryKey
Columns :
1  BookId   int  true
2  Name   text  false
3  Author   varchar  false
4  Price   double  false

tableName: Course
Format : Column_name, type, isPrimaryKey
Columns :
1  CourseID   int  true
2  Name   varchar  false
3  Semester   varchar  false

-------------Relation Diagram------------
Books ==> Course    relation  N:N
```

Figure 30 ERD saved in the text format

Module 6: Export Structure and Value

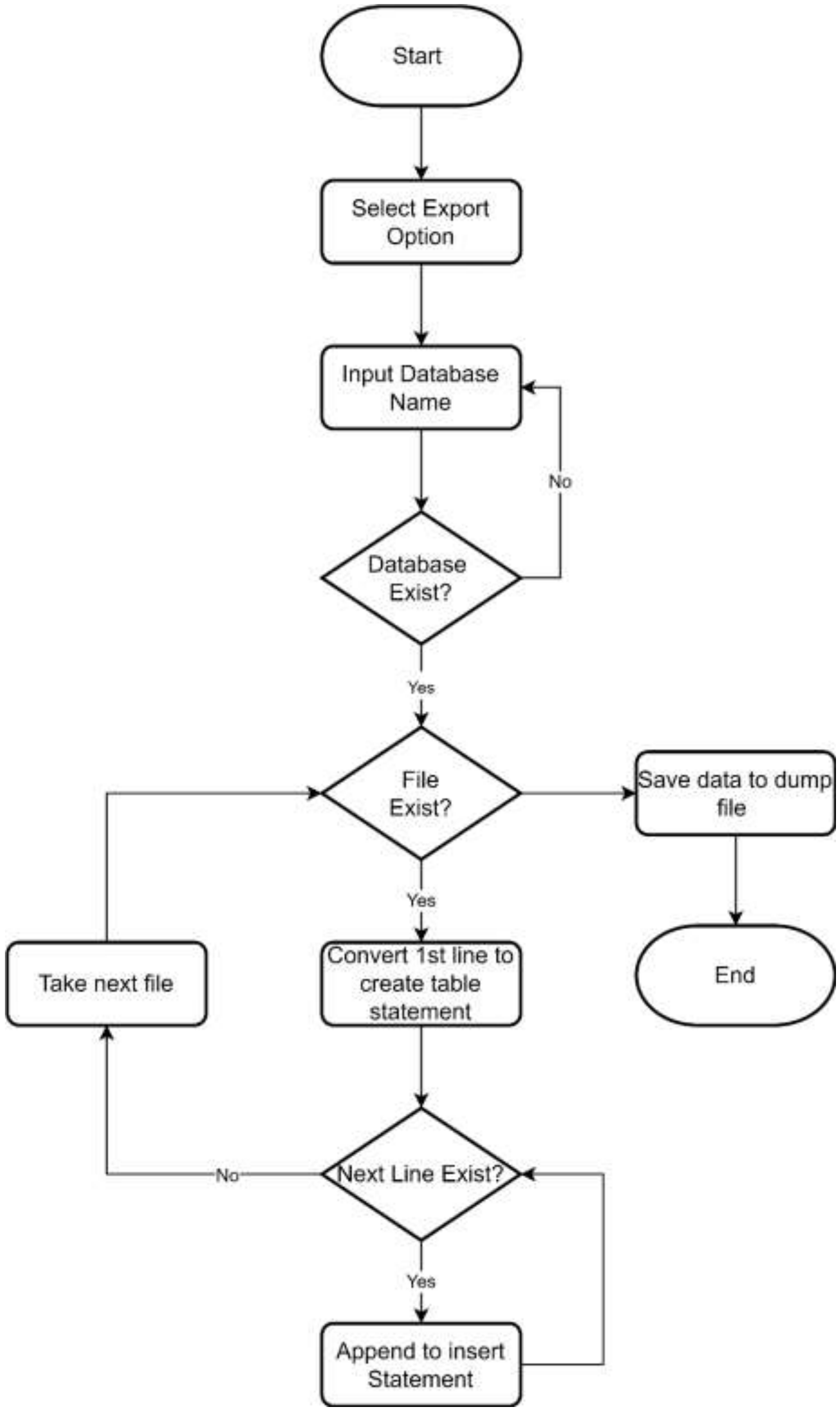


Figure 31 Export MySQL Dump file

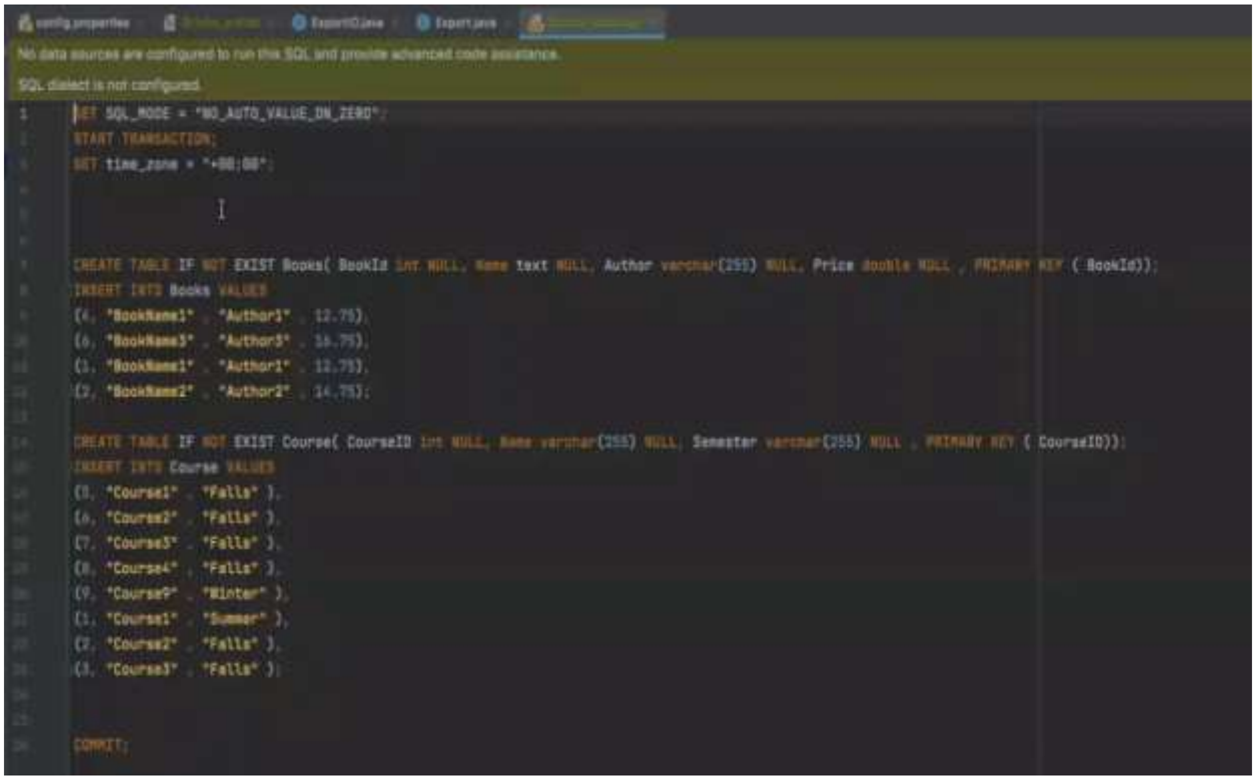
Description

Our D2 DB is also providing the export feature. Using the export feature, the user could take the whole backup of his database. This database backup generates the output as a dump file in the standard MySQL format. Using this dump file, the user could easily migrate his/her data to any other MySQL database service provider. This Dump file contains all the necessary information for both the structure as well as the Data. Importing this file to any other MySQL database will clone the structure and data of the Database that was stored in our Server. To Develop this feature, our team has implemented the logic of decrypting the data stored in our personalized encrypted format and then line by line appending it to the String in the format of Standard MySQL. After appending all the data in the String with the MySQL format, the String is stored to the file whose name is kept as <dbname>_dump.sql, where <dbname> would be the name of the database for which the dump file is generated. The main beauty of this service is, it fetches the all the datas and metadatas that are available on both the servers and then start generating the dump file, in such a way that there is no primary key duplication.

Pseudo Code

```
Export(<database_name>){
        FinalQuery = “SET SQL_MODE = \"NO_AUTO_VALUE_ON_ZERO\";\n" +
                  "START TRANSACTION;\n" +
                  "SET time_zone = \"+00:00\";\n\n\n\n”
        TableList = get TableList;
        Loop(till tablelist end){
                CreateQuery = “CREATE TABLE <table name>(”
                Split[] = parse first line which is the metadata of the table
                CreateQuery.append(Split[] in proper format)
                InsertQuery = “INSERT INTO <table name> VALUES”
                Loop(till line ends in file){
                        SplitData[] = parse line which data of tables
                        InsertQuery.append(SplitData[] in proper format)
                }
                FinalQuery.append(CreateQuery)
                FinalQuery.append(InsertQuery)
        }
        Storedata(<database_name>_dump.sql)
}
```

Test Cases

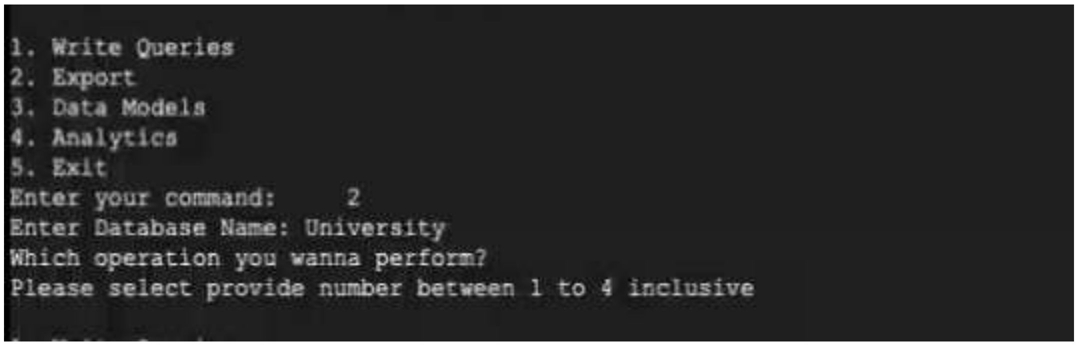


No data sources are configured to run this SQL and provide advanced code assistance.

SQL dialect is not configured.

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";



CREATE TABLE IF NOT EXIST Books( BookId int NULL, Name text NULL, Author varchar(255) NULL, Price double NULL , PRIMARY KEY ( BookId));
INSERT INTO Books VALUES
(4, "BookName1" , "Author1" , 12.75),
(6, "BookName3" , "Author3" , 16.75),
(1, "BookName1" , "Author1" , 12.75),
(2, "BookName2" , "Author2" , 14.75);

CREATE TABLE IF NOT EXIST Course( CourseID int NULL, Name varchar(255) NULL, Semester varchar(255) NULL , PRIMARY KEY ( CourseID));
INSERT INTO Course VALUES
(5, "Course1" , "Falls" ),
(6, "Course2" , "Falls" ),
(7, "Course3" , "Falls" ),
(8, "Course4" , "Falls" ),
(9, "Course9" , "Winter" ),
(1, "Course1" , "Summer" ),
(2, "Course2" , "Falls" ),
(3, "Course3" , "Falls" );


COMMIT;
```

Figure 32 Dump File



```
1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     2
Enter Database Name: University
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 33 Export

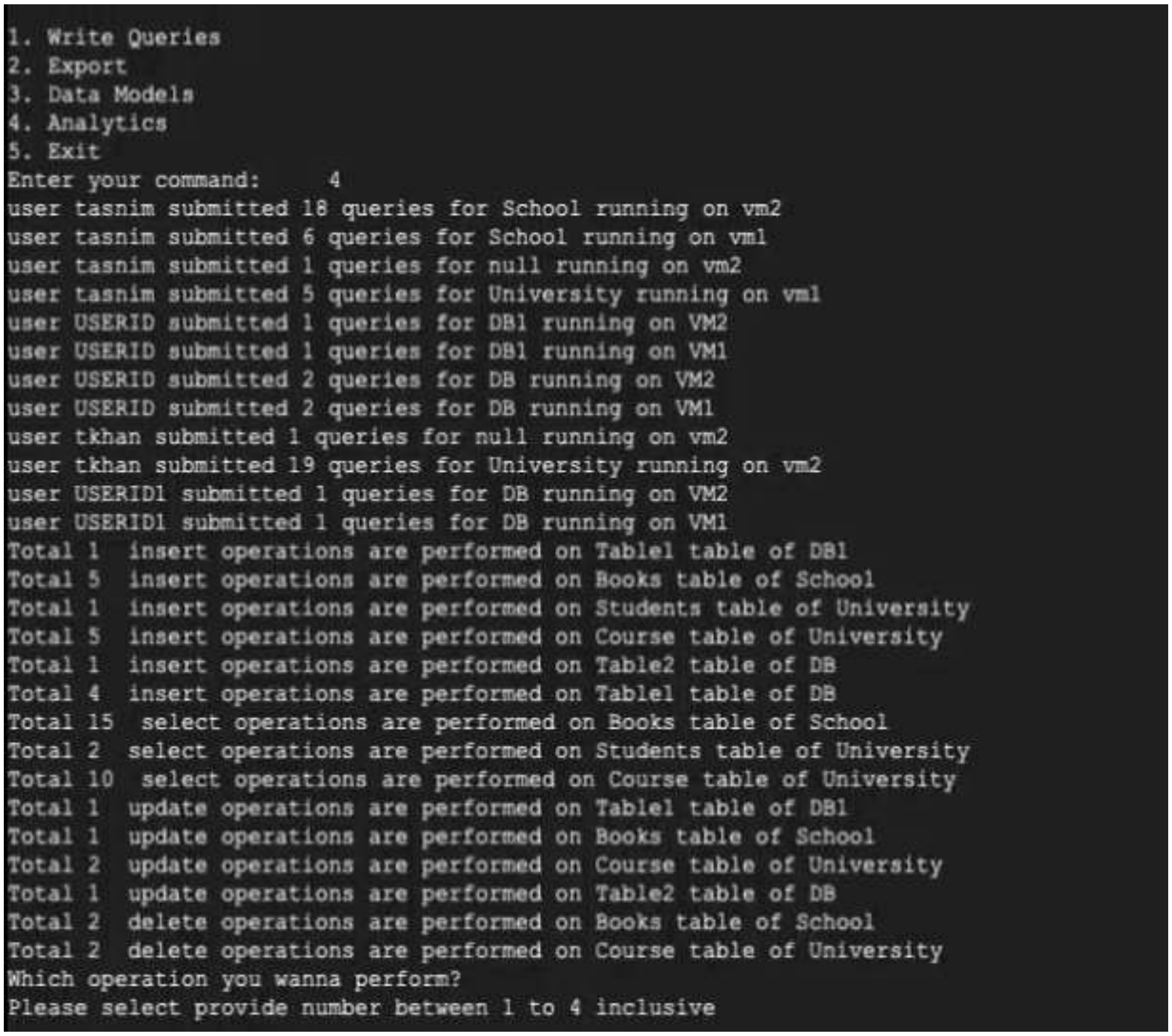
Module 7: Analytics

Description

Count queries will simply determine the number of queries performed in the provided database by the user logged in. The backend code will read the query logs file to count the total number of queries sent by the logged-in user in that database when the user types "COUNT QUERIES DbName," where DbName is the name of the database.

When a user "Ridham" logs in to the application and types "COUNT QUERIES DbName" into the console window, the application counts the total number of queries performed by the logged-in user Ridham on the database DbName and outputs something like "user 'Ridham' has executed x queries," where x is the total number of submitted queries.

Test Cases



```
1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:     4
user tasnim submitted 18 queries for School running on vm2
user tasnim submitted 6 queries for School running on vm1
user tasnim submitted 1 queries for null running on vm2
user tasnim submitted 5 queries for University running on vm1
user USERID submitted 1 queries for DB1 running on VM2
user USERID submitted 1 queries for DB1 running on VM1
user USERID submitted 2 queries for DB running on VM2
user USERID submitted 2 queries for DB running on VM1
user tkhan submitted 1 queries for null running on vm2
user tkhan submitted 19 queries for University running on vm2
user USERID1 submitted 1 queries for DB running on VM2
user USERID1 submitted 1 queries for DB running on VM1
Total 1  insert operations are performed on Table1 table of DB1
Total 5  insert operations are performed on Books table of School
Total 1  insert operations are performed on Students table of University
Total 5  insert operations are performed on Course table of University
Total 1  insert operations are performed on Table2 table of DB
Total 4  insert operations are performed on Table1 table of DB
Total 15  select operations are performed on Books table of School
Total 2  select operations are performed on Students table of University
Total 10  select operations are performed on Course table of University
Total 1  update operations are performed on Table1 table of DB1
Total 1  update operations are performed on Books table of School
Total 2  update operations are performed on Course table of University
Total 1  update operations are performed on Table2 table of DB
Total 2  delete operations are performed on Books table of School
Total 2  delete operations are performed on Course table of University
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive
```

Figure 34 Result of Analytics

Module 8: User Interface and Security

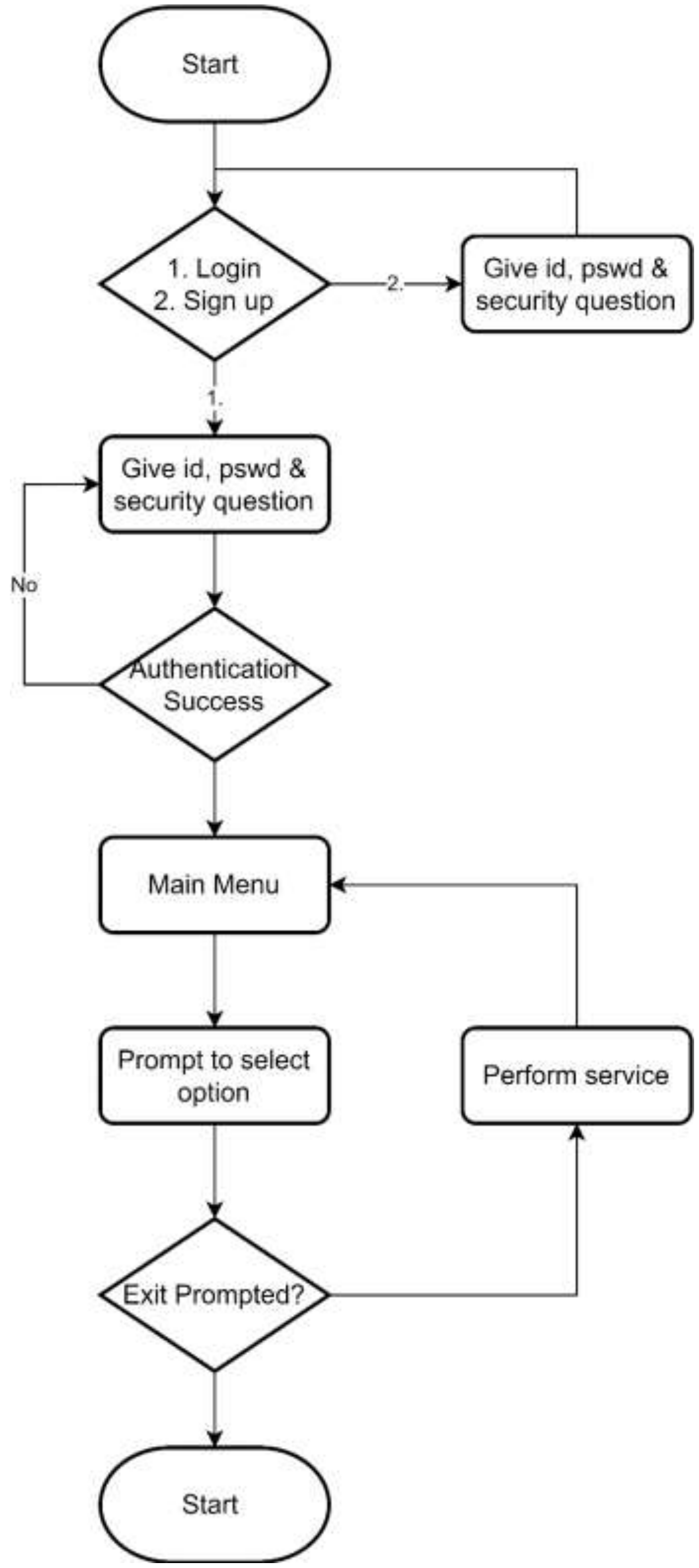
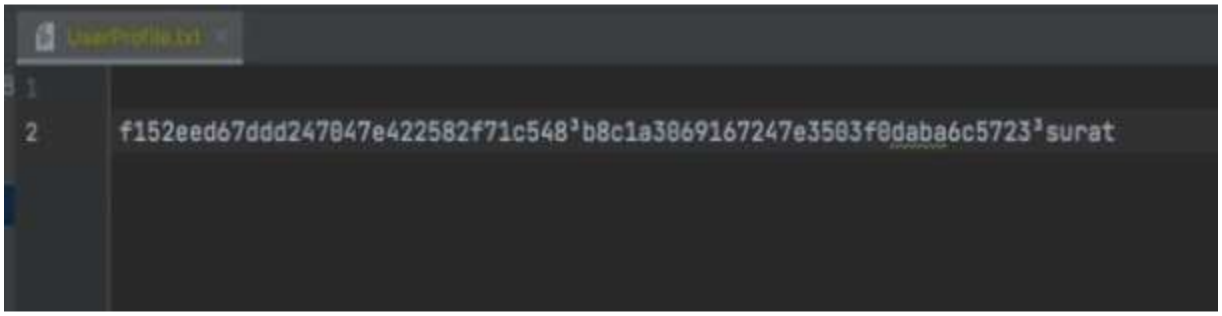


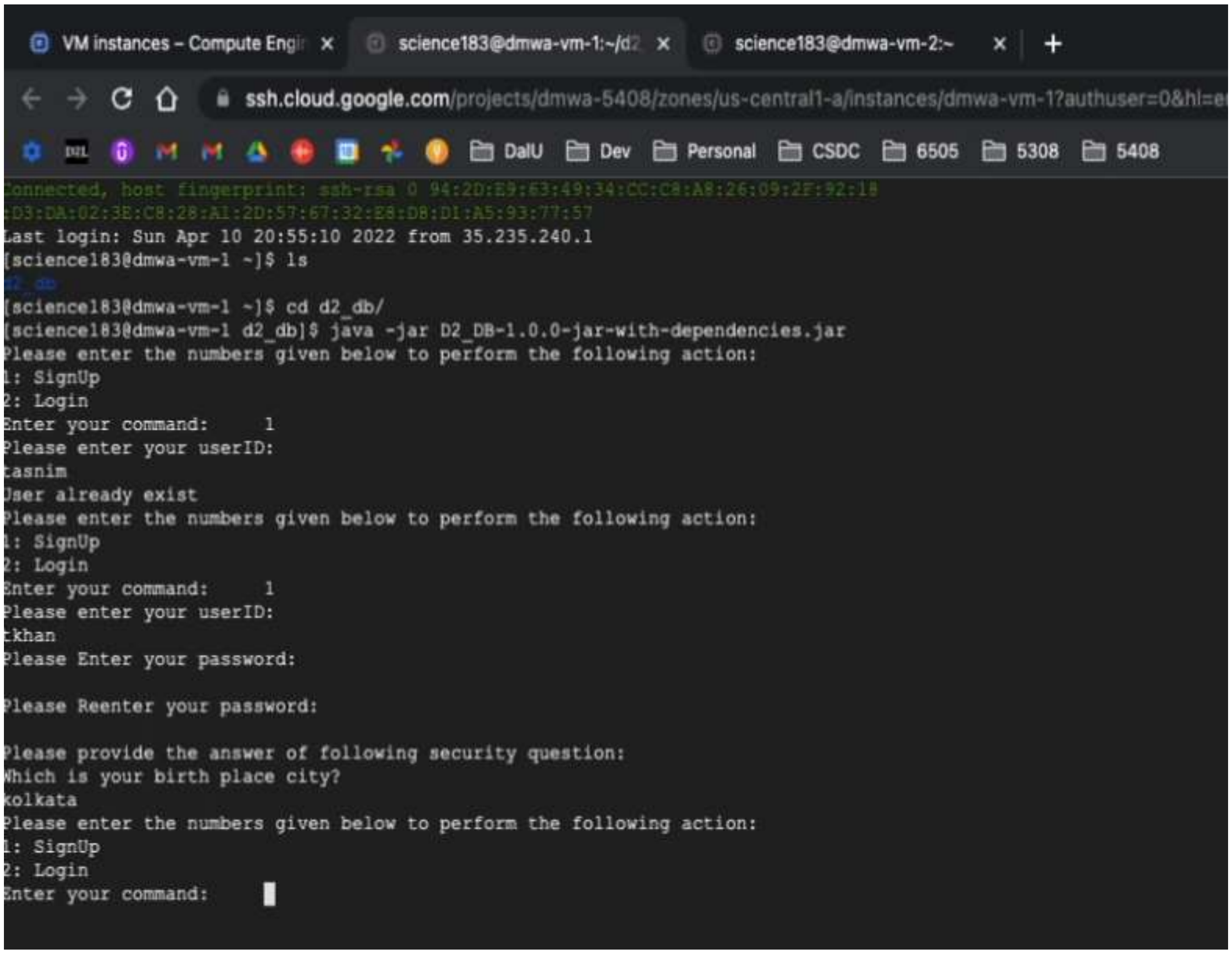
Figure 35 User Interface Flowchart

Test Cases



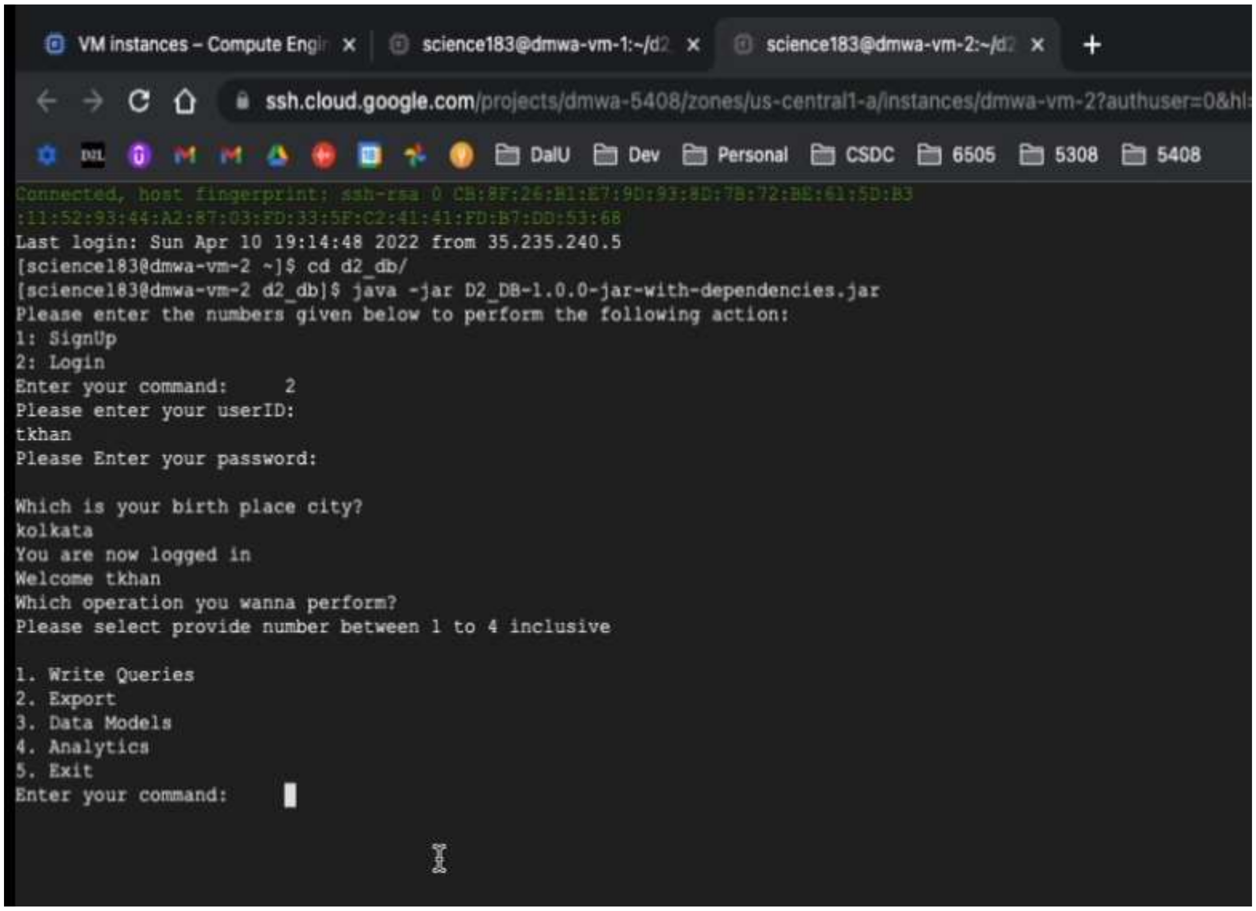
```
f152eed67ddd247847e422582f71c548³b8c1a3869167247e3583f8daba6c5723³surat
```

Figure 36 User Data Stored in MySQL Format



```
Connected, host fingerprint: ssh-rsa 0 94:2D:E9:63:49:34:CC:C8:A8:26:09:2F:92:18
:D3:DA:02:3E:C8:28:A1:2D:57:67:32:E8:D8:D1:A5:93:77:57
Last login: Sun Apr 10 20:55:10 2022 from 35.235.240.1
[science183@dmwa-vm-1 ~]$ ls
d2_db
[science183@dmwa-vm-1 ~]$ cd d2_db/
[science183@dmwa-vm-1 d2_db]$ java -jar D2_DB-1.0.0-jar-with-dependencies.jar
Please enter the numbers given below to perform the following action:
1: SignUp
2: Login
Enter your command:     1
Please enter your userID:
tasnim
User already exist
Please enter the numbers given below to perform the following action:
1: SignUp
2: Login
Enter your command:     1
Please enter your userID:
tkhan
Please Enter your password:

Please Reenter your password:

Please provide the answer of following security question:
Which is your birth place city?
kolkata
Please enter the numbers given below to perform the following action:
1: SignUp
2: Login
Enter your command:     
```

Figure 37 Sign Up (If user already exist)


VM instances – Compute Engi
science183@dmwa-vm-1:~/d2
science183@dmwa-vm-2:~/d2
ssh.cloud.google.com/projects/dmwa-5408/zones/us-central1-a/instances/dmwa-vm-2?authuser=0&hl
DalU
Dev
Personal
CSDC
6505
5308
5408

```
Connected, host fingerprint: ssh-rsa 0 CB:8F:26:B1:E7:9D:93:8D:7B:72:BE:61:5D:B3
:11:52:93:44:A2:87:03:FD:33:5F:C2:41:41:FD:B7:DD:53:68
Last login: Sun Apr 10 19:14:48 2022 from 35.235.240.5
[science183@dmwa-vm-2 ~]$ cd d2_db/
[science183@dmwa-vm-2 d2_db]$ java -jar D2_DB-1.0.0-jar-with-dependencies.jar
Please enter the numbers given below to perform the following action:
1: SignUp
2: Login
Enter your command:     2
Please enter your userID:
tkhan
Please Enter your password:

Which is your birth place city?
kolkata
You are now logged in
Welcome tkhan
Which operation you wanna perform?
Please select provide number between 1 to 4 inclusive

1. Write Queries
2. Export
3. Data Models
4. Analytics
5. Exit
Enter your command:
```

Figure 38 Login

Conclusion:

The D2 DB application has reached the end of its development cycle. User authorisation and authentication are used to safeguard the programme, and it can also develop databases that handle bespoke file formats. Users can interface with the database using D2 DB by writing conventional SQL commands. Aside from these features, users may use the application's built-in functions, which include analytics, data modelling, and data export.

Limitations:

- The data types of the queries are not checked during transaction. Because of this, if any user tries to enter the varchar data in the integer’s location. Then the system won’t throw any error. Instead, it will store the data of wrong data type in the Table.
- Concurrency is the major issue of our system, because only one user could operate the system at a time. Parallel users can’t work on the system.

Future Scopes:

- We need to check the datatypes of the query and validate it.
- We also need to work on the concurrency of the application.

References:

[1] “DalFCS Git,” [Online]. Available: https://git.cs.dal.ca/tasnim/csci-5408-group8-d2-db