

CSCI 5408

Data Management and Warehousing

Group-04
Project: Tiny DB

SPRINT 3 REPORT

Group Members:

Ramya Kommalapati (B00982851)

Parth Madhvani (B00981268)

Shifa Mirza(B00981843)

GitLab:

https://git.cs.dal.ca/mirza/csci_5408_s24_4

Table of Contents:

Contents

| | |
|--|---|
| Pseudo code:..... | 3 |
| Test cases and evidence of testing:..... | 5 |

Pseudo code:

Class TransactionManager

```
Private boolean inTransaction = false;  
Private File tempLogFile;  
Private Regex regex = new Regex();  
Private QueryExecuter queryExecuter = new QueryExecuter();  
Private String database = "";
```

Method beginTransaction(Database db2)

```
Set inTransaction to true;  
Initialize tempLogFile as a new File("temp_operation.log");  
Set database to db2.getName();
```

Method commitTransaction

```
If inTransaction is true  
Try  
    Open tempLogFile for reading using BufferedReader;  
    While there are lines to read  
        Read a line into log;  
        Call applyLog(log);  
        Call logToFile(log, new File("operation.log"));  
  
    Catch IOException e  
        Print stack trace;  
Finally  
    Set inTransaction to false;  
    Call deleteTempFiles();
```

Method rollbackTransaction

```
If inTransaction is true  
Try  
    //nothing  
Finally  
    Set inTransaction to false;  
    Call deleteTempFiles();
```

Method logOperation(String operation)

```
If inTransaction is true  
Try  
    Open tempLogFile for writing using BufferedWriter (append mode);
```

```
    Write operation to tempLogFile;
    Write a new line to tempLogFile;
Catch IOException e
    Print stack trace;
```

```
Else
    Try
        Call applyLog(operation);
        Call logToFile(operation, new File("operation.log"));
    Catch Exception e
        Print error message and stack trace;
```

Private Method applyLog(String operation) throws IOException

```
    Match operation with regex for insert;
    Match operation with regex for update;
    Match operation with regex for delete;

    If insertMatcher matches
        Call queryExecuter.insertData(operation, database);
    Else If updateMatcher matches
        Call queryExecuter.updateQuery(operation, regex, database);
    Else If deleteMatcher matches
        Call queryExecuter.deleteQuery(operation, regex, database);
    Else
        Print unknown operation error message;
```

Private Method logToFile(String operation, File file)

```
    Try
        Open file for writing using BufferedWriter (append mode);
        Write operation to file;
        Write a new line to file;
    Catch IOException e
        Print stack trace;
```

Private Method deleteTempFiles()

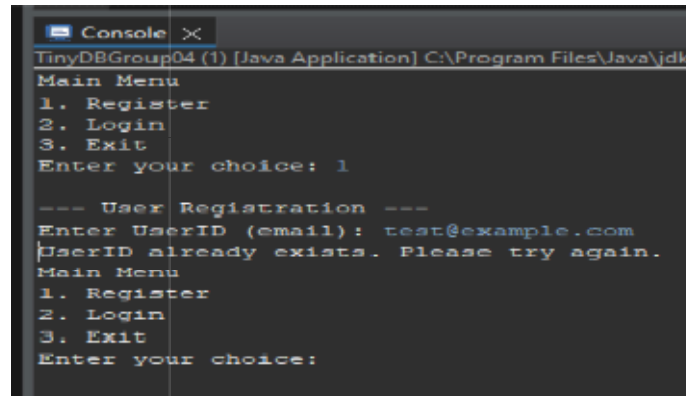
```
    If tempLogFile is not null and tempLogFile exists
        Delete tempLogFile;
```

Public Method isInTransaction()

```
    Return inTransaction;
```

Test cases and evidence of testing:

Description:Input:Existing UserID, Output: UserID already exists. Please try again

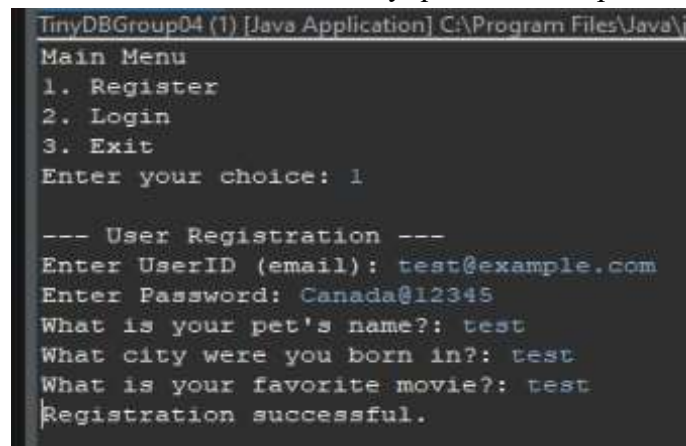


```
Console x
TinyDBGGroup04 (1) [Java Application] C:\Program Files\Java\jdk-
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 1

--- User Registration ---
Enter UserID (email): test@example.com
UserID already exists. Please try again.
Main Menu
1. Register
2. Login
3. Exit
Enter your choice:
```

Figure 1 Menu to Login or Register into the System

Description: Input:UserId, Password and Security questions, Output: Registration successful

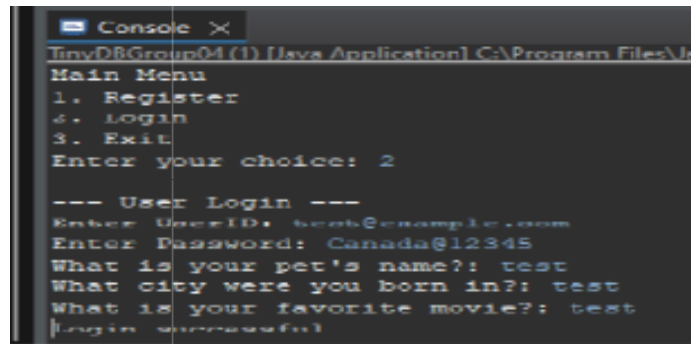


```
TinyDBGGroup04 (1) [Java Application] C:\Program Files\Java\j
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 1

--- User Registration ---
Enter UserID (email): test@example.com
Enter Password: Canada@12345
What is your pet's name?: test
What city were you born in?: test
What is your favorite movie?: test
Registration successful.
```

Figure 2 Successful User Registration

Description: Input: Login with registered UserId and Password, Output: Login Successful

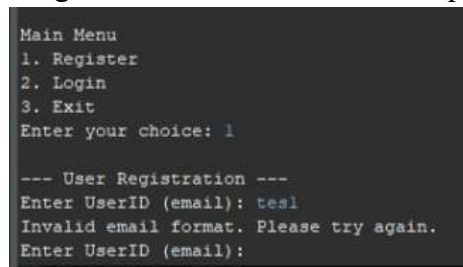


```
Console X
InvD8Group04 (1) [Java Application] C:\Program Files\Java\
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 2

--- User Login ---
Enter UserID: test@example.com
Enter Password: Canada@12345
What is your pet's name?: test
What city were you born in?: test
What is your favorite movie?: test
Login successful
```

Figure 3 Successful Login

Description: Input: Login with incorrect UserID, Output: UserID incorrect

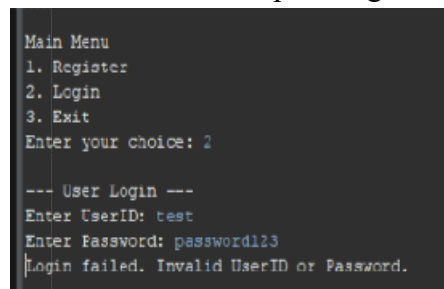


```
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 1

--- User Registration ---
Enter UserID (email): test
Invalid email format. Please try again.
Enter UserID (email):
```

Figure 4 Registering with invalid UserID

Description: Input: Invalid UserID, Output: Login Failed. Invalid UserID



```
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 2

--- User Login ---
Enter UserID: test
Enter Password: password123
Login failed. Invalid UserID or Password.
```

Figure 5 Login with Invalid UserID

Description: Input:Invalid Password, Output: Login Failed. Invalid Password

```
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 2

--- User Login ---
Enter UserID: test@example.com
Enter Password: password123
Login failed. Invalid UserID or Password.

Main Menu
1. Register
2. Login
3. Exit
Enter your choice:
```

Figure 6 Login with Invalid Password

Description: Input:Incorrect security answer, Output: Login Failed. Incorrectanswer to security question

```
TinyDBGGroup04 (1) [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe (Dur
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 2

--- User Login ---
Enter UserID: test@example.com
Enter Password: Canada@12345
What is your pet's name?: test1
Login failed. Incorrect answers to security questions.

Main Menu
1. Register
2. Login
3. Exit
Enter your choice:
```

Figure 7 Login Failed due to Incorrect security Question

Description: Input: Create a valid database, Output: Query executed successfully

```
Enter your query: create database test
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice:
```

Figure 8 Create database successfully

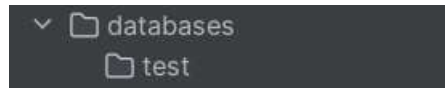


Figure 9 Folder created for the database

Description: Input:Existing database creation, Output: Failed to execute query.

```
Enter your choice: 1
Enter your query: create database test
Failed to execute query.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice:
```

Figure 10 Create database failed because database already present

Description: Input:Invalid create database query, Output: Failed to execute query.

```
Enter your query: create test2
Unsupported query type
Failed to execute query.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: |
```

Figure 11 Create database failed because of invalid query

Description: Input:Create table query , Output: Query executed successfully

```
Enter your query: create table student (id int,name varchar)
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
```

Figure 12 Create table successful

| 1 | id ### name |
|---|-------------|
| 2 | |

Figure 13 Table created

Description: Input:Insert query, Output: Query executed successfully

```
Enter your query: insert into student (1,'parth')
Data inserted successfully into table: student
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
```

Figure 14 Data inserted successfully

| 1 | id ### name |
|---|-------------|
| 2 | 1 ### parth |

Figure 15 data.txt file after insertion

Description: Input:Delete query, Output: Query executed successfully

```
Enter your query: delete from student where id='1'  
Query executed successfully.  
User Menu  
1. Write Queries  
2. Export Data and Structure  
3. ERD  
4. Exit
```

Figure 16 Delete from table successful

| | |
|---|-------------|
| 1 | id ### name |
| 2 | |

Figure 17 Valued deleted successfully

Description: Input:Invalid DeleteQuery, Output: Failed to execute query

```
Enter your query: delete from student where address='2'  
Failed to execute query.  
User Menu  
1. Write Queries  
2. Export Data and Structure  
3. ERD  
4. Exit
```

Figure 18 Delete unsuccessful because of invalid field name

```
Enter your query: delete from student2 where id='1'  
Table does not exist  
Failed to execute query.  
User Menu  
1. Write Queries  
2. Export Data and Structure  
3. ERD  
4. Exit
```

Figure 19 Delete unsuccessful because of invalid table name

Description : Drop query and the due to valid table name its successful

```
Enter your query: drop table student
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
```

Figure 20 Drop table successful

Description : Drop query and the due to invalid table name and unsuccessful

```
Enter your choice: 1
Enter your query: drop table student2
Table does not exist
Failed to execute query.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
```

Figure 21 Drop table unsuccessful because of invalid table name

Description : Update query successful due to valid table and field names

```
Enter your query: update student set name='ALEX' where id='1'
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
```

Figure 22 Update table successful

| | |
|---|-------------|
| 1 | id ### name |
| 2 | 1 ### ALEX |

Figure 23 Data updated successfully in data.txt file

Description : Drop query and the due to invalid field name and its unsuccessful

```
Enter your query: update student set address='halifax' where id='1'
address is not a valid field
Failed to execute query.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
```

Figure 24 Update invalid because of invalid field name

Description: The project folder structure before the creation of log directory

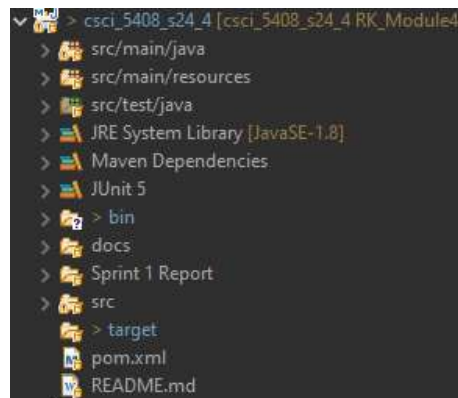


Figure 25 : Before the creation of logs directory

Description : The application is started so the general logs is created because the start and exit of program comes under general log and since general log doesn't exists before it was created

```
Log directory created successfully: logs/general/
Main Menu
1. Register
2. Login
3. Exit
Enter your choice:
```

Figure 26 : general log created successfully

Description: As the application started, and user registration was chosen the event directory was created as it does not exist previously

```
Log directory created successfully: logs/general/
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 1
Log directory created successfully: logs/event/

--- User Registration ---
Enter UserID (email):
```

Figure 27 : Event log is created successfully

Description: Once the user login is successful, he can write and execute the DB queries so option to write queries was chosen as event logs exists before the log was created

```
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 2

--- User Login ---
Enter UserID: test1@example.com
Enter Password: 123456789
What is your pet's name?: test
What city were you born in?: test
What is your favorite movie?: test
Login successful.

User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Log directory created successfully: logs/query/
```

Figure 28 : Query log was created successfully

Description: The folder structure of the project after the logs directory was created.

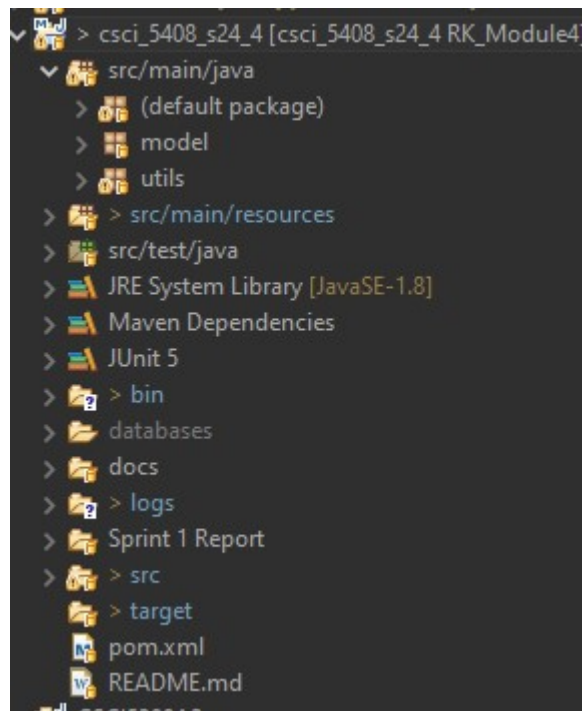


Figure 29 : Project folder structure after logs directory created

Description: As there are different log categories defined in the enum so different log categories folders were created in the log directory

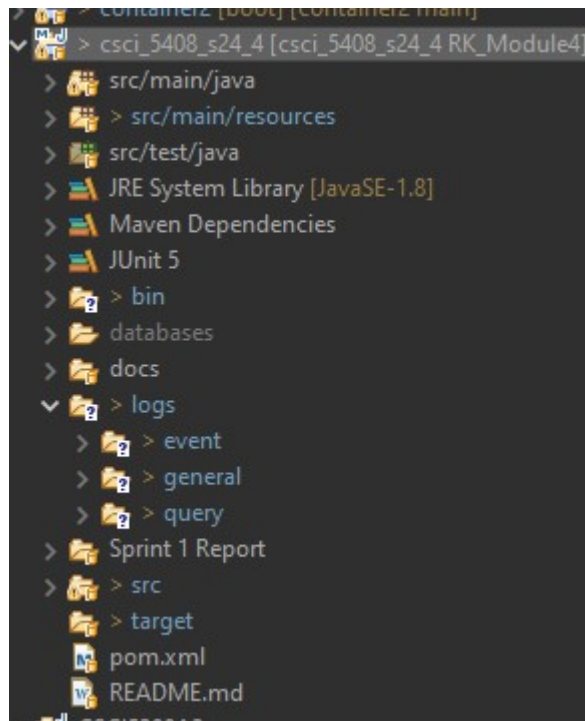


Figure 30 : different categories of logs created

Description: The naming convention of the log text files that stores logs. These are synchronized and create new log text file for every day.

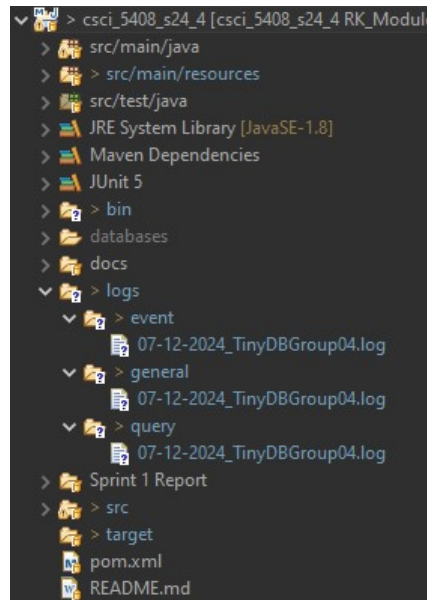


Figure 31 : The file name format for each category of log

Description: As the application was started and series of activates are done like registering of user, writing queries etc .so the logs are recorded as below in general log file

```
1 [07-12-2024 17:36:13] ### [GENERAL] Application started.
2 [07-12-2024 17:36:13] ### [GENERAL] Data directory verified/created successfully.
3 [07-12-2024 17:41:12] ### [GENERAL] User menu accessed by: test1@example.com
4 [07-12-2024 17:42:48] ### [GENERAL] Unsupported query type: show databases;
5 [07-12-2024 17:43:01] ### [GENERAL] No databases present in directory: C:\Users\hp\Desktop\CSCI 5408 Data mgmt\project\cs
6 [07-12-2024 17:44:11] ### [GENERAL] Database created successfully: test;
7 [07-12-2024 17:44:18] ### [GENERAL] Invalid choice in user menu by: test1@example.com
8 [07-12-2024 17:44:31] ### [GENERAL] Database found: test;
9 [07-12-2024 17:45:16] ### [GENERAL] Invalid CREATE TABLE query: create table test_table(id int);
10 [07-12-2024 17:45:33] ### [GENERAL] Invalid choice in user menu by: test1@example.com
11 [07-12-2024 17:45:59] ### [GENERAL] Invalid CREATE TABLE query: create table test(id int);
12
```

Figure 32 : The general log file

Description: As the application was started and series of activates are done like registering of user, writing queries etc .so the log are recorded as below in query log file

```
07-12-2024_TinyDBGGroup04.log X
1 [07-12-2024 17:41:14] ### [QUERY] User test1@example.comwriting query to execute
2 [07-12-2024 17:42:48] ### [QUERY] Failed to execute query: show databases;
3 [07-12-2024 17:42:54] ### [QUERY] User test1@example.comwriting query to execute
4 [07-12-2024 17:43:01] ### [QUERY] Failed to execute query: use test;
5 [07-12-2024 17:43:59] ### [QUERY] User test1@example.comwriting query to execute
6 [07-12-2024 17:44:11] ### [QUERY] Query executed successfully: create database test;
7 [07-12-2024 17:44:27] ### [QUERY] User test1@example.comwriting query to execute
8 [07-12-2024 17:44:31] ### [QUERY] Query executed successfully: use test;
9 [07-12-2024 17:44:36] ### [QUERY] User test1@example.comwriting query to execute
10 [07-12-2024 17:45:16] ### [QUERY] Failed to execute query: create table test_table(id int);
11 [07-12-2024 17:45:37] ### [QUERY] User test1@example.comwriting query to execute
12 [07-12-2024 17:45:59] ### [QUERY] Failed to execute query: create table test(id int);
13
```

Figure 33 : The query log file after executing some queries in the application

Description: As the application was started and series of activates are done like registering of user ,writing queries etc .so the log are recorded as below in event log file

```
64 [07-12-2024 17:40:38] ### [EVENT] User login started.
65 [07-12-2024 17:41:05] ### [EVENT] User deserialized from string: ramyak@example.com
66 [07-12-2024 17:41:05] ### [EVENT] User created: ramyak@example.com
67 [07-12-2024 17:41:05] ### [EVENT] User deserialized from string: test@example.com
68 [07-12-2024 17:41:05] ### [EVENT] User created: test@example.com
69 [07-12-2024 17:41:05] ### [EVENT] User deserialized from string: p@g.com
70 [07-12-2024 17:41:05] ### [EVENT] User created: p@g.com
71 [07-12-2024 17:41:05] ### [EVENT] User deserialized from string: test1@example.com
72 [07-12-2024 17:41:05] ### [EVENT] User created: test1@example.com
73 [07-12-2024 17:41:05] ### [EVENT] User deserialized from string: test5@example.com
74 [07-12-2024 17:41:05] ### [EVENT] User created: test5@example.com
75 [07-12-2024 17:41:05] ### [EVENT] Users loaded successfully from file: src/main/resources/users.txt
76 [07-12-2024 17:41:12] ### [EVENT] Login successful: test1@example.com
77 [07-12-2024 17:43:01] ### [EVENT] Executing USE DATABASE query: use test;
78 [07-12-2024 17:44:11] ### [EVENT] Executing CREATE DATABASE query: create database test;
79 [07-12-2024 17:44:31] ### [EVENT] Executing USE DATABASE query: use test;
80 [07-12-2024 17:45:16] ### [EVENT] Executing CREATE TABLE query: create table test_table(id int);
81 [07-12-2024 17:45:59] ### [EVENT] Executing CREATE TABLE query: create table test(id int);
82
```

Figure 34 : Event logs file after the events has happened like user registration and login etc

Description: Creating table with reference keyword.

```
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: create table course (courseId int,name varchar,studentId varchar references student.id)
Query executed successfully.
```

Figure 35 Create table with references keyword

Description: Inserting table into just created table.

```
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: insert into course (1,Advanced topics in software development,1)
Data inserted successfully into table: course
Query executed successfully.
```

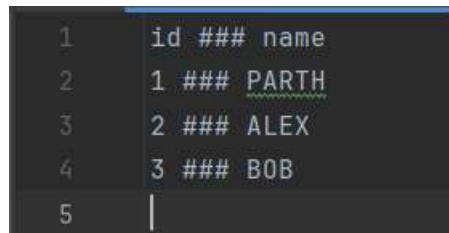
Figure 36 Inserting data into course table

Description: Metadata of the referenced table , which is student table.

| | |
|---|--------------|
| 1 | id int |
| 2 | name varchar |

Figure 37 Student table metadata

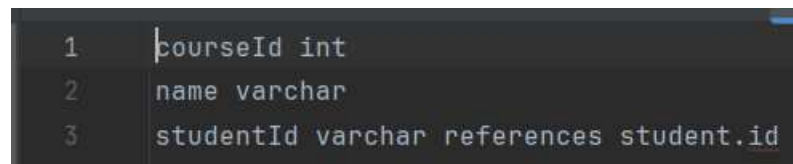
Description: Data of the referenced table(student). Here the course table references to the student table.



| 1 | id ### name |
|---|-------------|
| 2 | 1 ### PARTH |
| 3 | 2 ### ALEX |
| 4 | 3 ### BOB |
| 5 | |

Figure 38 Student table data

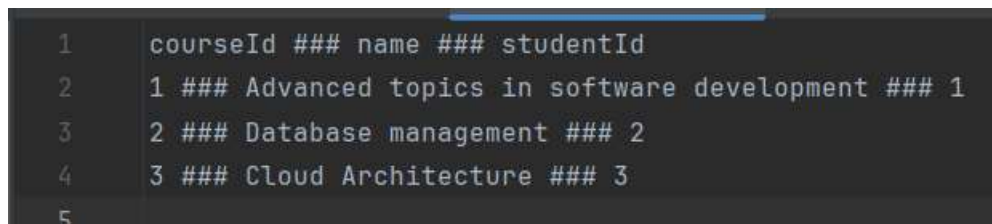
Description: Metadata of the course table(student). The course table references to the id column of the student table.



| | |
|---|---|
| 1 | courseId int |
| 2 | name varchar |
| 3 | studentId varchar references student.id |

Figure 39 Course table metadata

Description: Data of the course table. This table references to the id column of the student table.



| 1 | courseId ### name ### studentId |
|---|---|
| 2 | 1 ### Advanced topics in software development ### 1 |
| 3 | 2 ### Database management ### 2 |
| 4 | 3 ### Cloud Architecture ### 3 |
| 5 | |

Figure 40 Course table data

Description: Files created for both the tables.

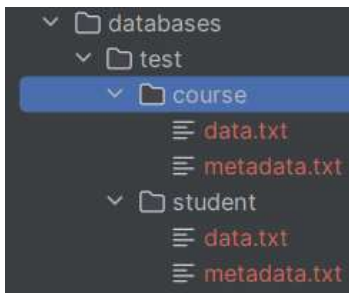


Figure 41 Both table folders created

Description: Asking user which database to export the erd..

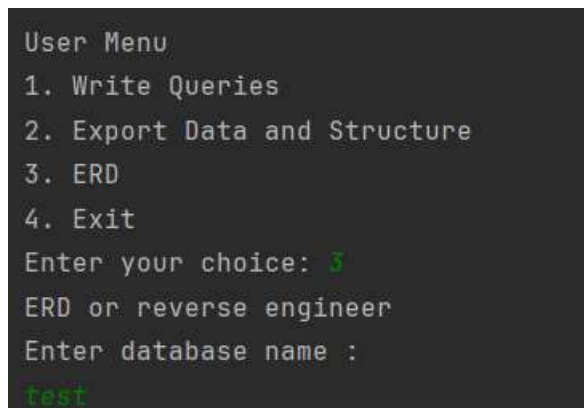


Figure 42 Asking user which database to export erd

Description: erd.txt file created.

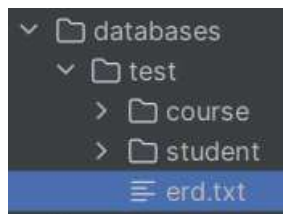


Figure 43 ERD file created

Description: Content of the erd file. The erd file contains the information like name and fields of the table, the cardinality of the table, and which field is referencing to which table.

```
1 Table : course
2 Cardinality : student 1-to-1 course
3     courseId int
4     name varchar
5     studentId varchar references student.id
6 *****
7 Table : student
8     id int
9     name varchar
10
11 *****
12
```

Figure 44 ERD file

Description: Export ERD with reference to other tables.

```
1 Table : course
2 Cardinality : student 1-to-1 course
3     id int
4     name varchar
5     studentid int references student.id (Relation: course registred by student )
6
7 *****
8 Table : professor
9 Cardinality : student 1-to-1 professor
10    id int
11    name varchar
12    studentid int references student.id (Relation: professor teaches student )
13
14 *****
15 Table : student
16    id int pk
17    name varchar
18
19 *****
```

Figure 45

Description: Trying to reference a table which does not exist in the database. Here activity table is trying to refer fafda table, which does not exist in the database.

```
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: create table activity (activityId int,name varchar,studentId int references fafda.id)
Table does not exist
Referenced table does not exist
```

Figure 46 Referenced table does not exist

Description: Trying to reference a table field which does not exist. Here the activity table is trying to reference to the idd column of student table. But the student table does not have a column with name idd.

```
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: create table activity (activityId int,name varchar,studentId int references student.idd)
Referenced column does not exist
```

Figure 47 Referenced column does not exist in student table

Description: Export Empty Database

Here I'm trying to export the database which does not have any tables

```
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 2
Export Data and Structure
Choose option from below

1: noValueTable
2: tinyDB
3: emptyDb
Choice: 1
SQL dump file created successfully: /Users/shifamirza/Da1/csci_5408_s24_4/./databases/noValueTable/noValueTable_dump.sql
```

Fig 48: Empty Database

```
noValueTable_dump.sql x
1 CREATE DATABASE noValueTable;
2 USE noValueTable;
3
4 CREATE TABLE noValue (
5     id int ,
6     name varchar
7 );
8
9
10
11
LF UTF-8 4 spaces
```

Fig 49: SQL dump for Empty Database

Description: Export table without data

Here's I'm trying to export database which has table but doesn't have any data inside the table

```
Enter your query: insert into student (1,'S1')
Data inserted successfully into table: student
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 2
Export Data and Structure
Choose option from below

1: noValueTable
2: +none
```

Fig50: Empty Table

```
noValueTable_dump.sql x
1 CREATE DATABASE noValueTable;
2 USE noValueTable;
3
4 CREATE TABLE noValue (
5     id int ,
6     name varchar
7 );
8
9
10
11
LF UTF-8 4 spaces
```

Fig51: Empty Database Dump File

Description: Export table with Data

Here I'm trying to export database which as values in it

```
Enter your query: insert into student (1,'S')
Data inserted successfully into table: student
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 2
Export Data and Structure
Choose option from below

1: noValueTable
2: +value
```

Fig 52: Table with values

```
tinyDB_dump.sql x
1 CREATE DATABASE tinyDB;
2 USE tinyDB;
3
4 CREATE TABLE student (
5     id int PRIMARY KEY,
6     name varchar
7 );
8
9 INSERT INTO student VALUES (1, 'S');
10
11
12
```

Fig 53: Table with Values Dump file

Description: Dump already exists case

Here if the dump already exists then it will delete the existing dump and create new dump file

```
Enter your query: insert into student (2, 'M')
Data inserted successfully into table: student
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice:
```

Fig 54: Insert command into table

```
Delete existing dump
SQL dump file created successfully: /Users/shifamirza/Dal/csci\_5408\_s24\_4/./databases/tinyDB/tinyDB\_dump.sql
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice:
```

Fig 55: Deleting existing dump

Description: Invalid choice

If the user selects an invalid choice amongst displayed

```
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 2
Export Data and Structure
Choose option from below

1: noValueTable
2: tinyDB
3: TableWithValues
4: emptyDb
Choice: 5
Invalid choice
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
```

Fig 56: User selects invalid choice

Description: Database with multiple tables

IF a database has multiple tables it will give the dump for both the tables

```
Enter your query: insert into professor (1,'Dr.Ajay')
Data inserted successfully into table: professor
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 2
Export Data and Structure
Choose option from below

1: noValueTable
2: tinyDB
3: TableWithValues
4: twoTableDB
5: emptyDb
Choice: 2
Delete existing dump
SQL dump file created successfully: /Users/shifamirza/Dal/csci_5408_s24_4/.databases/tinyDB/tinyDB_dump.sql
```

Fig 57: Multiple table dump

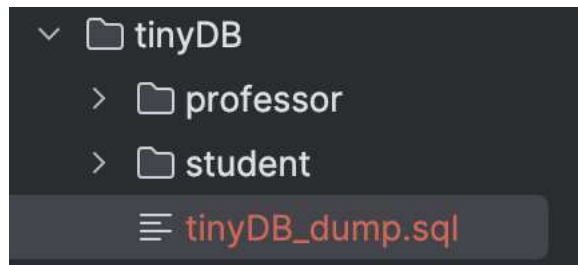


Fig 58: Multiple table dump

```
tinyDB_dump.sql x
1 CREATE DATABASE tinyDB;
2 USE tinyDB;
3
4 CREATE TABLE student (
5     id int PRIMARY KEY,
6     name varchar
7 );
8
9 INSERT INTO student VALUES (1, 'S');
10 INSERT INTO student VALUES (2, 'M');
11
12
13 CREATE TABLE professor (
14     id int PRIMARY KEY,
15     name varchar
16 );
17
18 INSERT INTO professor VALUES (1, 'Dr.Ajay');
19
20
```

Fig 59: Multiple table dump file

Description: login into the TinyDB and creating a database student to work on transactions

```
Main Menu
1. Register
2. Login
3. Exit
Enter your choice: 2

--- User Login ---
Enter UserID: test1@example.com
Enter Password: 123456789
What is your pet's name?: test
What city were you born in?: test
What is your favorite movie?: test
Login successful.

User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: create database student
Executing query: CREATE DATABASE STUDENT
Query executed successfully.
```

Figure 60: Login into the system and creating the database

Description: Using the created database student

```
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: use student
Executing query: USE STUDENT
Query executed successfully.
```

Figure61:using the created database

Description: creating a table named students in the database student with columns id and name

```
Enter your choice: 1
Enter your query: create table students (id int pk ,name varchar)
Executing query: CREATE TABLE STUDENTS (ID INT PK ,NAME VARCHAR)
Query executed successfully.
```

Figure 12 : creating a students table in the database

Description: Inserting dummy data into the students table without starting a transaction to see the behaviours

```
Enter your choice: 1
Enter your query: insert into students (id,name) values (1,'test')
Executing query: INSERT INTO STUDENTS (ID,NAME) VALUES (1,'TEST')
Data inserted successfully into table: STUDENTS
Query executed successfully.

User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: insert into students (id,name) values (2,'data')
Executing query: INSERT INTO STUDENTS (ID,NAME) VALUES (2,'DATA')
Data inserted successfully into table: STUDENTS
Query executed successfully.

User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: insert into students (id,name) values (3,'deleteTest')
Executing query: INSERT INTO STUDENTS (ID,NAME) VALUES (3,'DELETETEST')
Data inserted successfully into table: STUDENTS
Query executed successfully.
```

Figure 63 :inserting data into students table without transactions

Description: The data in the students table after inserting data without transactions

```
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: select * from students
Executing query: SELECT * FROM STUDENTS
ID | NAME
2 | DATA
3 | DELETETEST
Query executed successfully.
```

Figure 64: students table after inserting the data

Description: Deleting from the table data without transactions and the data after deletion

```
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: delete from students where id='3'
Executing query: DELETE FROM STUDENTS WHERE ID='3'
Data deleted successfully from table: STUDENTS
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: select * from students
Executing query: SELECT * FROM STUDENTS
ID | NAME
2 | DATA
Query executed successfully.
```

Figure 65 :Deleting the data without the transaction

Description: Starting a transaction and inserting some data into the table and then commit the transaction

```
Query executed successfully.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: BEGIN
Executing query: BEGIN
Transaction started.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: insert into students (id,name) values (3,'Ramya');
Executing query: INSERT INTO STUDENTS (ID,NAME) VALUES (3,'RAMYA');
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: COMMIT
Executing query: COMMIT
Data inserted successfully into table: STUDENTS
Transaction committed.
User Menu
```

Figure 67 : starting transaction and committing the transaction

Description: The students tables data after the transaction commit

```
Enter your choice: 1
Enter your query: COMMIT
Executing query: COMMIT
Data inserted successfully into table: STUDENTS
Transaction committed.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: select * from students
Executing query: SELECT * FROM STUDENTS
ID | NAME
1. 'TEST
2 | DATA
3 | RAMYA
Query executed successfully.
```

Figure 68: the students table after transaction commit

Description: starting a transaction and rolling it back with a series of DML statements

```
Enter your choice: 1
Enter your query: BEGIN
Executing query: BEGIN
Transaction started.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: insert into students(id,name) values (4,'K')
Executing query: INSERT INTO STUDENTS(ID,NAME) VALUES (4,'K')
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: delete from students where id='4'
Executing query: DELETE FROM STUDENTS WHERE ID='4'
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: delete from students where id='3'
Executing query: DELETE FROM STUDENTS WHERE ID='3'
```

Figure 69 : transaction rollback

Description: The students table data after doing the transactions rollback

```
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: delete from students where id='3'
Executing query: DELETE FROM STUDENTS WHERE ID='3'
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: ROLLBACK
Executing query: ROLLBACK
Transaction rolled back.
User Menu
1. Write Queries
2. Export Data and Structure
3. ERD
4. Exit
Enter your choice: 1
Enter your query: select * from students
Executing query: SELECT * FROM STUDENTS
ID | NAME
1. 'TEST
2 | DATA
3 | RAMYA
Query executed successfully.
```

Figure 70: students table after rollback

Team meeting Details:

| Date | Time | Agenda | Attendees | Meeting Type | Meeting |
|---------|----------|-----------------------------|---|--------------|---|
| July 15 | 12:30 PM | Transactions implementation | Parth Madhvani, Shifa Mirza, Ramya Kommalapati | Offline | We have discussed about various ways to implement transactions and finally decided to implement the transactions using temp files . |