Pseudocode:

TransactionManager

Initialization:

```
function TransactionManager():
    set inTransaction to false

Set Current Database Name:
function setCurrentDBName(currentDBName):
    set this.currentDBName to currentDBName
    set originalDatabasePath to "DataStore/" + currentDBName
```

Get Instance:

```
function getInstance():
    if instance is null:
        lock TransactionManager class:
            if instance is null:
                set instance to new TransactionManager()
    return instance
```

Start Transaction:

```
function startTransaction():
    if inTransaction:
        print "Transaction already in progress"
        throw CustomException("Transaction already in progress")

    if currentDBName is null:
        print "No database set currently"
        throw CustomException("No database set currently")

    set tempDatabasePath to "DataStore/" + currentDBName + "_temp"
    call copyDirectory with originalDatabasePath and tempDatabasePath

    set inTransaction to true
```

Commit Transaction:

```
function commitTransaction():
    if not inTransaction:
        throw CustomException("No transaction in progress")
```

```
    call deleteDirectory with originalDatabasePath
    move tempDatabasePath to originalDatabasePath with
StandardCopyOption.ATOMIC_MOVE
    set inTransaction to false
```

Rollback Transaction:

```
function rollbackTransaction():
    if not inTransaction:
        throw CustomException("No transaction in progress")

    call deleteDirectory with tempDatabasePath
    set inTransaction to false
```

Copy Directory:

```
function copyDirectory(source, target):
    for each src in walk source:
        set dest to target.resolve(source.relativize(src))
        try:
            copy src to dest with StandardCopyOption.REPLACE_EXISTING
        catch IOException as e:
            print stack trace of e
```

Delete Directory:

```
function deleteDirectory(path):
    for each file in walk path in reverse order:
        delete file
```

Get Database Path:

```
function getDatabasePath():
    return inTransaction ? tempDatabasePath : originalDatabasePath
```

Test Cases:

1. Start a transaction

```
SQL> use test;
SQL> start transaction;
SQL>
```

Figure 1: start a transaction in the console

```
[Successful] [2024-07-27 17:15:13.342] User 'testuser' logged in successfully.
[Successful] [2024-07-27 17:15:20.362] Using database 'test'.
[Successful] [2024-07-27 17:15:25.979] start transaction; executed in 10 ms. State= Database: test| Tables:
```

Figure 2: transaction successfully started

2. Start a transaction before using a database

```
SQL> start transaction;
No database set currently
SQL>
```

Figure 3: Starting a transaction without using a database in the console

```
[Successful] [2024-07-27 17:09:51.855] User 'testuser' logged in successfully.
[Failed] [2024-07-27 17:11:59.172] Error processing query 'start transaction; ms: No database set currently
```

Figure 4: Logs show that the transaction cannot start because no database is selected

2. Start a transaction while another transaction is active

```
SQL> use test;
SQL> start transaction;
SQL> start transaction;
Transaction already in progress
```

Figure 5: start a transaction in the console while another transaction is running

```
[Successful] [2024-07-27 17:15:20.362] Using database 'test'.
[Successful] [2024-07-27 17:15:25.979] start transaction; executed in 10 ms. State= Database: test| Tables:
[Failed] [2024-07-27 17:17:14.493] Error processing query 'start transaction; ms: Transaction already in progress
```

Figure 6: Logs show that the new transaction cannot start because another transaction is in progress

3. Commit without starting a transaction

```
Enter your SQL queries (type 'EXIT' to end):
SQL> use test;
SQL> commit;
SQL>
```

Figure 7: using commit without starting a transaction in the console

```
[Successful] [2024-07-27 17:09:51.855] User 'testuser' logged in successfully.
[Successful] [2024-07-27 17:19:50.198] Using database 'test'.
[Failed] [2024-07-27 17:19:52.951] Error processing query 'commit; ms: No transaction in progress
```

Figure 8: Logs show that commit cannot be done because no transaction is in progress

4. Rollback without starting a transaction

```
Enter your SQL queries (type 'EXIT' to end):
SQL> use test;
SQL> rollback;
SQL>
```

Figure 9: using commit without starting a transaction in the console

```
[Successful] [2024-07-27 17:09:51.855] User 'testuser' logged in successfully.
[Successful] [2024-07-27 17:21:11.102] Using database 'test'.
[Failed] [2024-07-27 17:21:14.163] Error processing query 'rollback; ms: No transaction in progress
```

Figure 10: Logs show that commit cannot be done because no transaction is in progress

5. Full transaction process with commit

```
Enter your SQL queries (type 'EXIT' to end):
SQL> use test;
SQL> start transaction;
SQL> create table test-table (id INT,name varchar);
Enter Field Name: exit
SQL>  insert into test-table values (1,Arta);
SQL> update test-table set name = Prashanth where id = 1;
SQL>  create table test-table2 (id INT);
Enter Field Name: exit
SQL> drop table test-table2;
Deleted successfully
SQL>  insert into test-table values (2,jay);
SQL> delete from test-table where id = 2;
SQL>
```

Figure 11: running a transaction in the console with Create, Insert, update, delete, and drop commands

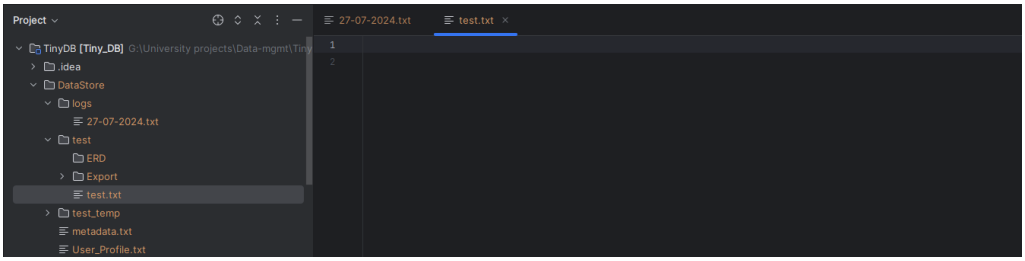

Figure 12: files are not changed before committing the transaction

```
Enter your SQL queries (type 'EXIT' to end):
SQL> use test;
SQL> start transaction;
SQL> create table test-table (id INT,name varchar);
Enter Field Name: exit
SQL>  insert into test-table values (1,Arta);
SQL> update test-table set name = Prashanth where id = 1;
SQL>  create table test-table2 (id INT);
Enter Field Name: exit
SQL> drop table test-table2;
Deleted successfully
SQL>  insert into test-table values (2,jay);
SQL> delete from test-table where id = 2;
SQL> commit;
```

Figure 13: the transaction is committed

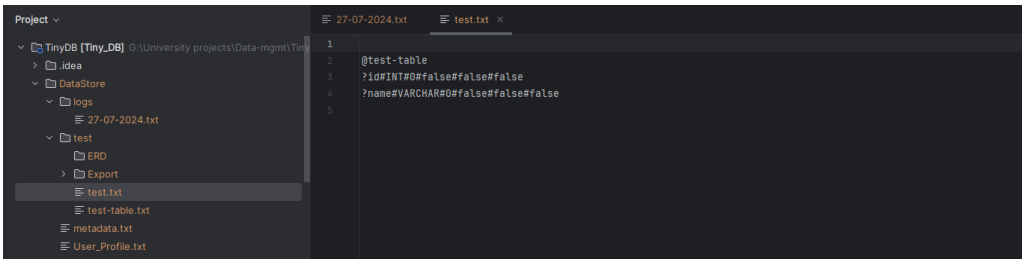

Figure 14: changes appeared on the database file

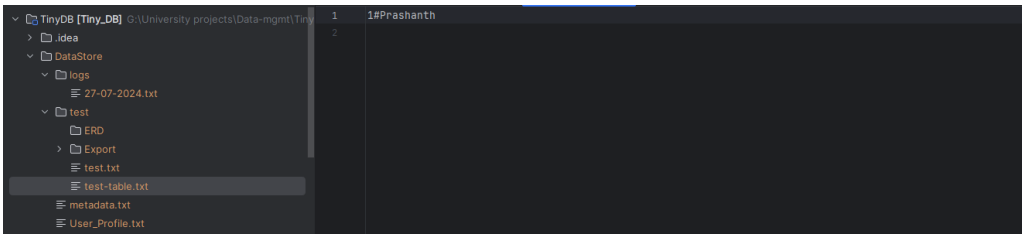

Figure 15: A new table is created with the latest changes

6. Full transaction process with rollback

```
Enter your SQL queries (type 'EXIT' to end):
SQL> use test;
SQL> start transaction;
SQL> create table test-table2 (id int);
Enter Field Name: exit
SQL> drop table test-table;
Deleted successfully
SQL> insert into test-table2 values (1);
SQL> update test-table2 set id = 2 where id = 1;
SQL>
```

Figure 16: running a transaction in the console with Create, Insert, update, delete, and drop commands

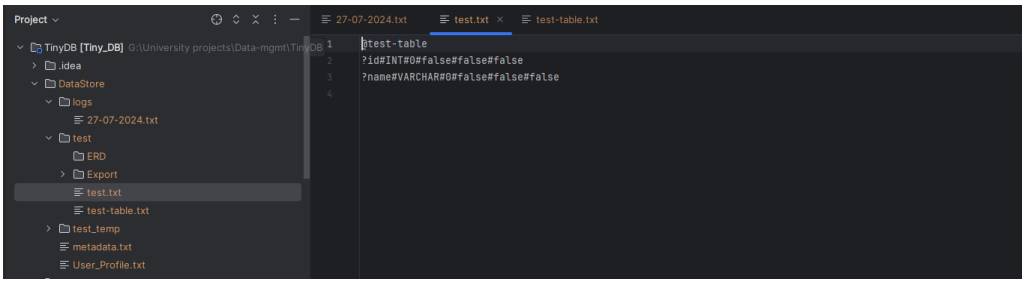

Figure 17: files are not changed before rolling back the transaction

```
Enter your SQL queries (type 'EXIT' to end):
SQL> use test;
SQL> start transaction;
SQL> create table test-table2 (id int);
Enter Field Name: exit
SQL> drop table test-table;
Deleted successfully
SQL> insert into test-table2 values (1);
SQL> update test-table2 set id = 2 where id = 1;
SQL> rollback;
```

Figure 18: the transaction is rolled back

Figure 19: no changes are made in database files and no new tables are created

7. Stop the program while a transaction is running

```
Enter your SQL queries (type 'EXIT' to end):
SQL> use test;
SQL> start transaction;
SQL> create table test-table2 (name varchar);
Enter Field Name: exit
SQL>
```

Figure 20: started a transaction and created a table without committing or rolling back

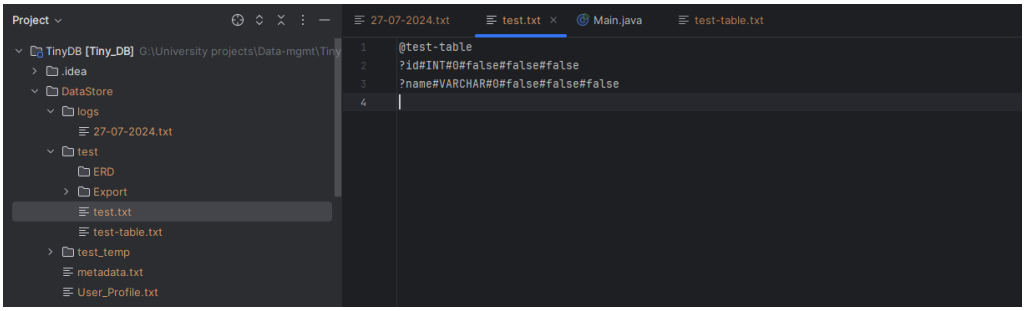

Figure 21: state of the database files before stopping the program

After stopping and running the program again:

```
Enter your SQL queries (type 'EXIT' to end):
SQL> use test;
SQL> commit;
SQL> rollback;
SQL> start transaction;
SQL>
```

Figure 22: trying to commit, rollback and start a new transaction in the console

```
[Successful] [2024-07-27 17:46:50.815] User 'testuser' logged in successfully.
[Successful] [2024-07-27 17:47:01.792] Using database 'test'.
[Failed] [2024-07-27 17:47:05.078] Error processing query 'commit; ms: No transaction in progress
[Failed] [2024-07-27 17:47:08.469] Error processing query 'rollback; ms: No transaction in progress
[Successful] [2024-07-27 17:47:15.24] start transaction; executed in 4 ms. State= Database: test| Tables: (test-table, 1 records),
```

Figure 23: logs show that there is no transaction in progress and the new transaction has successfully started

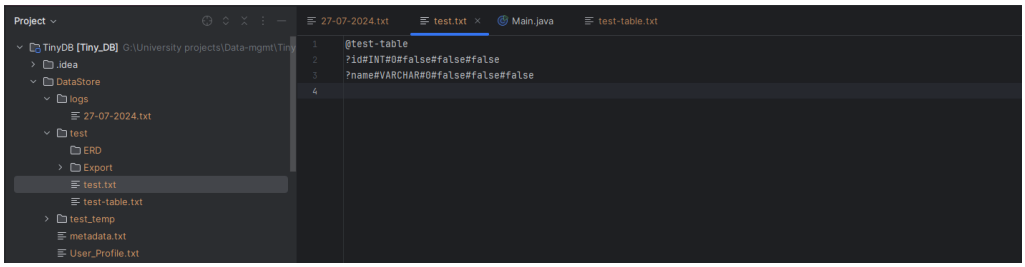

Figure 23: state of the database files after stopping and running the program again