DALHOUSIE UNIVERSITY

CSCI 5408

Data Management, Warehousing and Analytics

Assignment 2

Problem - 2

Name : Sagarkumar Pankajbhai Vaghasia

CSID : vaghasia

Banner ID : #B00878629

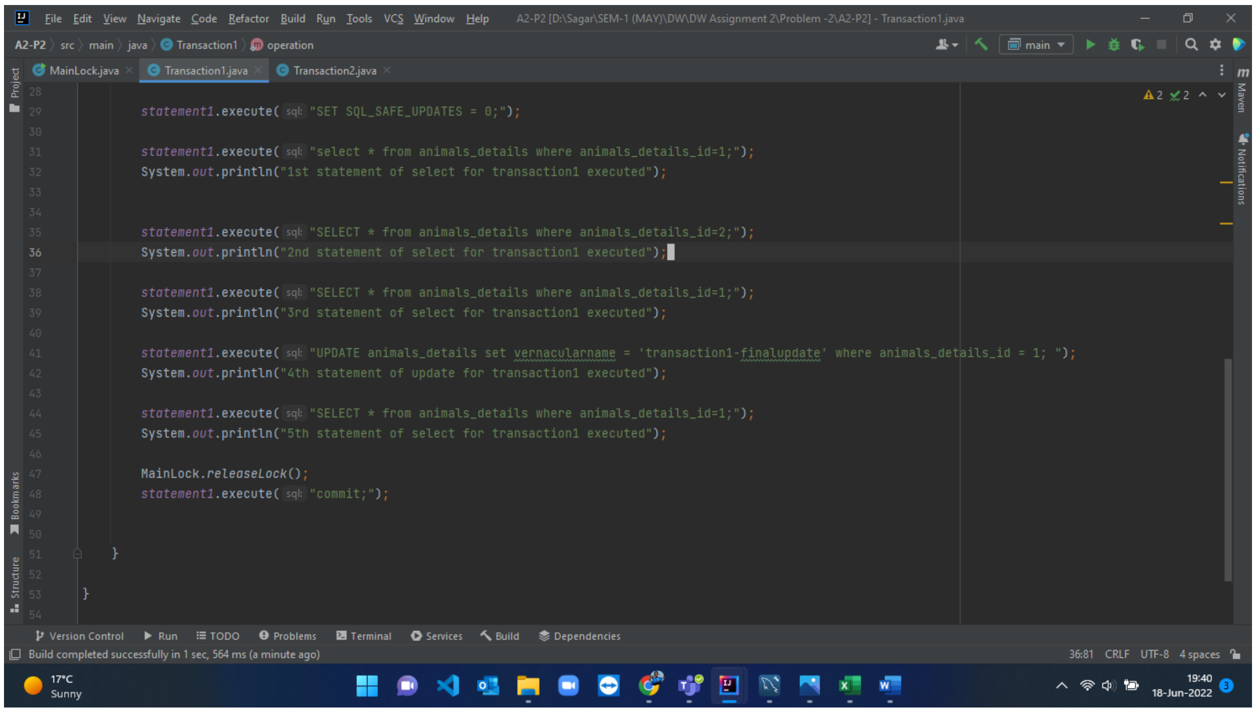
GitLab Link :
https://git.cs.dal.ca/vaghasia/csci5408_s22_sagarkumar_vaghasia_b00878629

Problem #2

I wrote 2 transaction blocks. Each transaction contains 5 statements. Both the transactions are performing the operations on the same field.

The SQL queries for transactions are attached below:

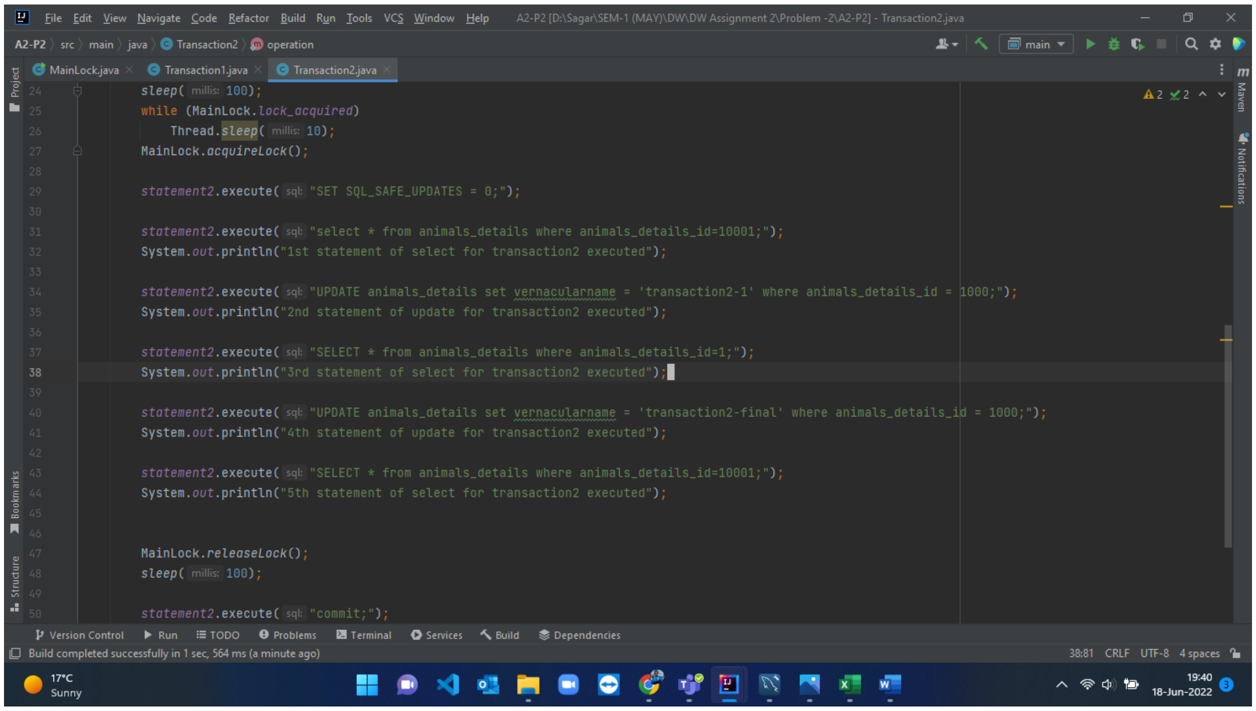
Transaction -1 queries:


A2-P2 [D:\Sagar\SEM-1 (MAY)\DW\DW Assignment 2\Problem -2\A2-P2] - Transaction1.java
A2-P2 › src › main › java › Transaction1 › operation
MainLock.java
Transaction1.java
Transaction2.java

```
        statement1.execute( sql: "SET SQL_SAFE_UPDATES = 0;");

        statement1.execute( sql: "select * from animals_details where animals_details_id=1;");
        System.out.println("1st statement of select for transaction1 executed");


        statement1.execute( sql: "SELECT * from animals_details where animals_details_id=2;");
        System.out.println("2nd statement of select for transaction1 executed");

        statement1.execute( sql: "SELECT * from animals_details where animals_details_id=1;");
        System.out.println("3rd statement of select for transaction1 executed");

        statement1.execute( sql: "UPDATE animals_details set vernacularname = 'transaction1-finalupdate' where animals_details_id = 1; ");
        System.out.println("4th statement of update for transaction1 executed");

        statement1.execute( sql: "SELECT * from animals_details where animals_details_id=1;");
        System.out.println("5th statement of select for transaction1 executed");

        MainLock.releaseLock();
        statement1.execute( sql: "commit;");


    }

}
```

Build completed successfully in 1 sec, 564 ms (a minute ago)

Transaction - 2 queries:


A2-P2 [D:\Sagar\SEM-1 (MAY)\DW\DW Assignment 2\Problem -2\A2-P2] - Transaction2.java
A2-P2 › src › main › java › Transaction2 › operation
MainLock.java
Transaction1.java
Transaction2.java

```
        sleep( millis: 100);
        while (MainLock.lock_acquired)
            Thread.sleep( millis: 10);
        MainLock.acquireLock();

        statement2.execute( sql: "SET SQL_SAFE_UPDATES = 0;");

        statement2.execute( sql: "select * from animals_details where animals_details_id=10001;");
        System.out.println("1st statement of select for transaction2 executed");

        statement2.execute( sql: "UPDATE animals_details set vernacularname = 'transaction2-1' where animals_details_id = 1000;");
        System.out.println("2nd statement of update for transaction2 executed");

        statement2.execute( sql: "SELECT * from animals_details where animals_details_id=1;");
        System.out.println("3rd statement of select for transaction2 executed");

        statement2.execute( sql: "UPDATE animals_details set vernacularname = 'transaction2-final' where animals_details_id = 1000;");
        System.out.println("4th statement of update for transaction2 executed");

        statement2.execute( sql: "SELECT * from animals_details where animals_details_id=10001;");
        System.out.println("5th statement of select for transaction2 executed");


        MainLock.releaseLock();
        sleep( millis: 100);

        statement2.execute( sql: "commit;");
```

Build completed successfully in 1 sec, 564 ms (a minute ago)

I have created a lock manager by using two methods. 1) acquireLock() and 2) releaseLock(). acquireLock() method is used to acquire locks for the transaction and releaseLock() method is used to release all the locks acquired by the transaction.

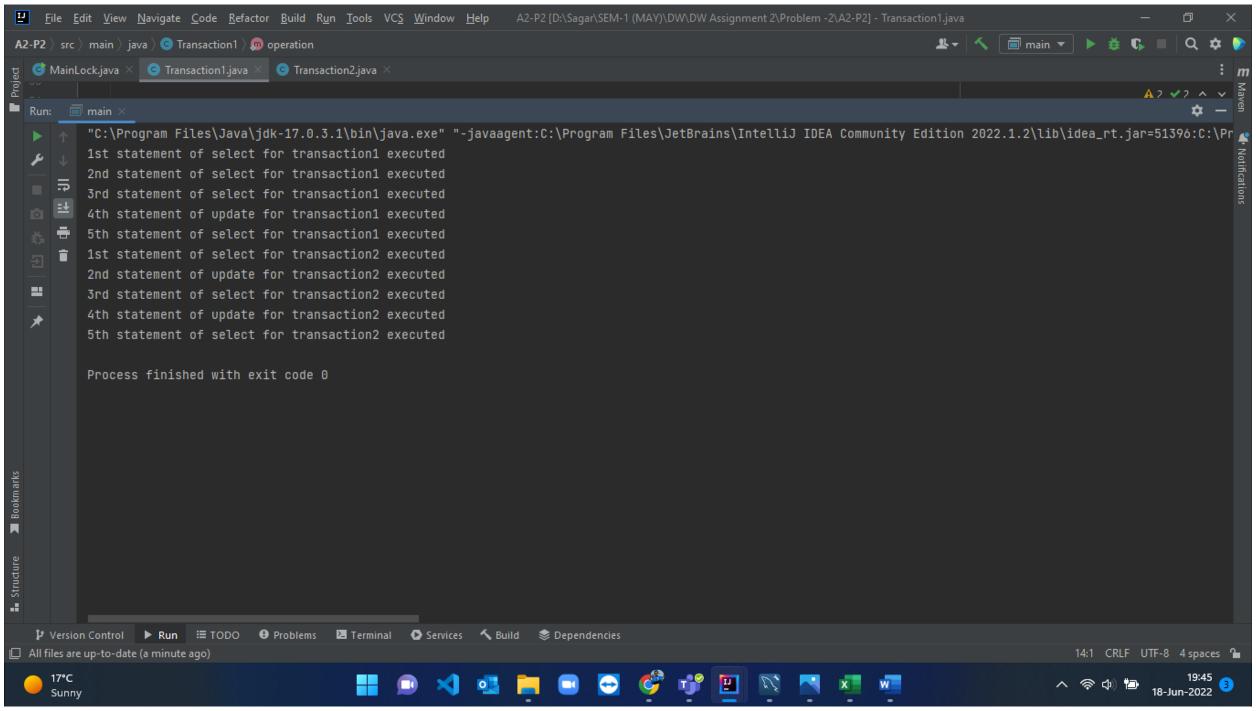
In Main class, I created both the methods for acquiring and releasing locks.
I have created an operation() method in both the transaction classes. In this method, the actual execution of the transaction block takes place. In other words, in the operation() method, all queries related to transactions are executed and the methods for acquiring locks and releasing locks are called.

I have declared one boolean variable lock_acquired whose value initially I set as false.
I set the auto commit to false for both the transactions and then I start both the transactions from Main class. Now both the threads are started and are running parallel.

Test Cases

Case 1 : If thread for transaction 1 enters to the system first then it will acquire all the locks and when transaction 2 will call the method for the acquiring lock but as the locks are not released by transaction 1 so transaction 2 must wait until transaction 1 releases the locks. After completing the transaction, it will release the locks.4

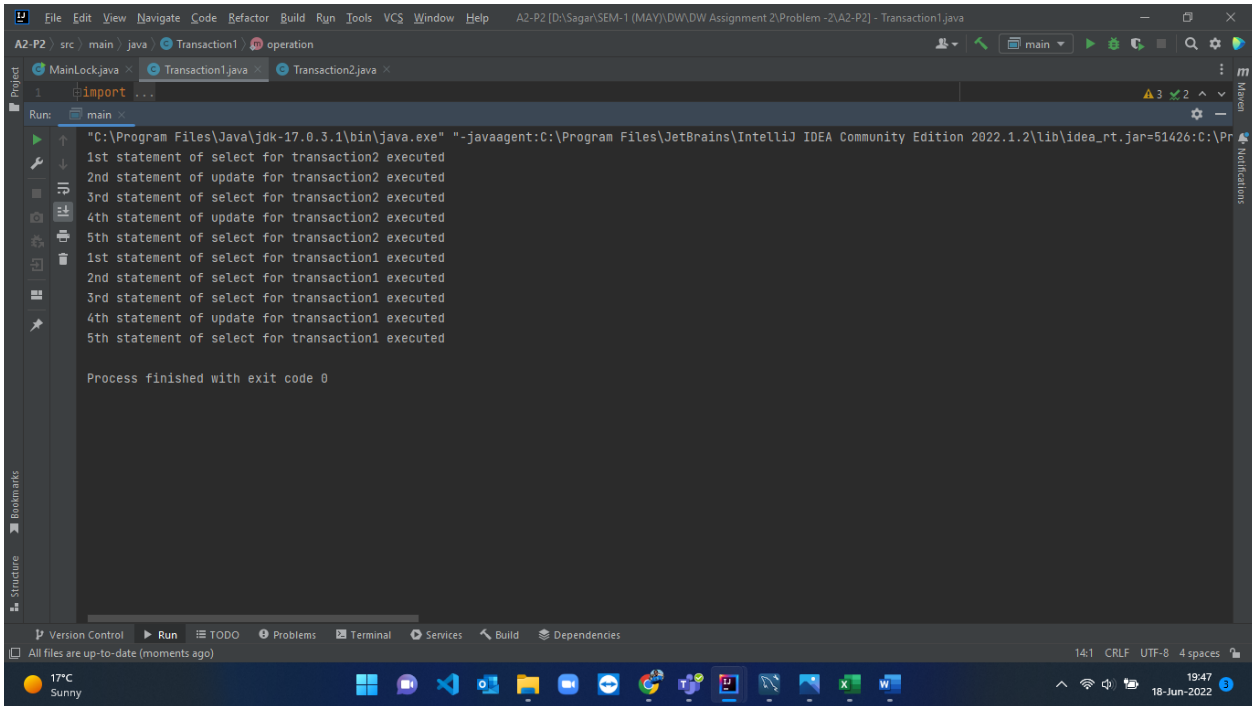
The screenshot of the output window for case 1 is attached below:


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
A2-P2 [D:\Sagar\SEM-1 (MAY)\DW\DW Assignment 2\Problem -2\A2-P2] - Transaction1.java
A2-P2 src main java Transaction1 operation
MainLock.java
Transaction1.java
Transaction2.java
Run: main
Version Control Run TODO Problems Terminal Services Build Dependencies
All files are up-to-date (a minute ago)
14:1 CRLF UTF-8 4 spaces
17°C Sunny
19:45 18-Jun-2022

```
"C:\Program Files\Java\jdk-17.0.3.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1.2\lib\idea_rt.jar=51396:C:\Pr
1st statement of select for transaction1 executed
2nd statement of select for transaction1 executed
3rd statement of select for transaction1 executed
4th statement of update for transaction1 executed
5th statement of select for transaction1 executed
1st statement of select for transaction2 executed
2nd statement of update for transaction2 executed
3rd statement of select for transaction2 executed
4th statement of update for transaction2 executed
5th statement of select for transaction2 executed

Process finished with exit code 0
```

Case 2 : If thread for transaction 2 enters the system first then it will acquire all the locks and when transaction 1 will call the method for the acquiring lock but as the locks are not released by transaction 2 so transaction 1 must wait until transaction 2 releases the locks. After completing the transaction, it will release the locks.

The screenshot of the output window for case 2 is attached below:



```
"C:\Program Files\Java\jdk-17.0.3.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1.2\lib\idea_rt.jar=51426:C:\Pr
1st statement of select for transaction2 executed
2nd statement of update for transaction2 executed
3rd statement of select for transaction2 executed
4th statement of update for transaction2 executed
5th statement of select for transaction2 executed
1st statement of select for transaction1 executed
2nd statement of select for transaction1 executed
3rd statement of select for transaction1 executed
4th statement of update for transaction1 executed
5th statement of select for transaction1 executed

Process finished with exit code 0
```