**Project Report**

Gauri Pasalkar – 1001836923

Implement a program that simulates the behavior of the two-phase locking (2PL) protocol for concurrency control.

The implementation of this will make use of  **“List”** Data Structure and Object oriented programming from Python for storing and processing the transactions and the locks applied on the data items.

**Programming Language**: Python

**Data Structures**:

* List
* Transaction Table: We will make use of class named transactionTable. It contains details such as transaction id, transaction time stamp, transaction state, LockedItems, BlockedOperations. We will add and update the transactions and also we will change the transaction states to active, waiting, commit or abort based on the current transaction. It will consist of following elements:

|  |  |
| --- | --- |
| **Elements** | **Datatype** |
|  |  |
| transaction\_id | int |
| trans\_timestamp | timestamp |
| trans\_state | string |
| trans\_locked\_items | list |
| trans\_blocked\_operations | list |
|  |  |

* Lock Table: We will make use of class named lockTable to add and update the transactions and also we will change the lock states to read lock or write lock based on the current transaction. It will consist of following elements:

|  |  |
| --- | --- |
| **Elements** | **Datatype** |
|  |  |
| lock\_id | int |
| lock\_item | string |
| lock\_state | string |
| tid\_on\_hold | list |
| tid\_on\_wait | list |
|  |  |

**Execution Steps:**

1. Open the “.ipynb” file on jupyter note book
2. Upload the input files to jupyter working directory
3. Run all the cells at once
4. User need to specify the Input file name to the output prompt (Use the **.txt** extension)

**Pseudo Code for each function:**

1. Begin

Input: Transaction item

Insert item into transactionTable

Set trans\_state := active

1. Read

Input: Transaction item

For iteration through lockTable

If the Transaction item is locked

If lock\_state of the item is WriteLocked

Wound-Wait protocol will be implemented from transaction item to determine if it is blocked or aborted

Else if lock\_state of the item is ReadLocked

Insert into lockTable.

Add item into trans\_locked\_items

End if

Else if transaction item is not locked

Set lock\_state of the item to ReadLocked

End else

End For

1. Write

Input: Transaction item

For iteration through lockTable

If the Transaction item is locked

If lock\_state of the item is ReadLocked

Set lock\_state of the item to WriteLocked

Else if lock\_state of the item is WriteLocked

Wound-Wait protocol will be implemented from transaction item to determine if it is blocked or aborted

End if

Else transaction item is not locked

Set lock\_state of the item to WriteLocked

End else

End For

1. End

Input: Transaction item

For iteration through transactionTable

If trans\_state is NOT equal to abort

trans\_state := commit

end if

End For