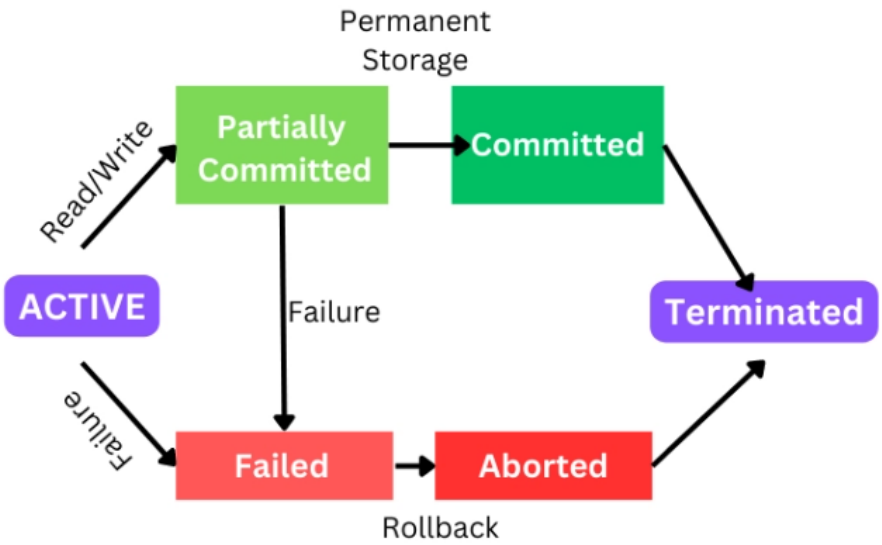
Transaction states in dbms

- transactions are carried out to alter the data in a database.
- The states of transaction in DBMS control the processes determining whether a transaction will proceed or be terminated.
- these states of transaction in DBMS outline the transaction's current situation and provide details on how it will be handled going forward.
- Transaction's life cycle



- Six states of transaction in DBMS
  1. Active state
     - → First stage of a transaction in DBMS
     - → A transaction is said to be active if its instructions are being executed
     - → One can perform operations such as INSERT or DELETE in the database, main memory buffer (portion of main memory that temporarily store changes made by the transaction begore they are written to the actual database files) now contains all changes made by the transaction.
  2. Partially committed state
     - → All the instructions within the transaction have been successfully executed.
     - → But why it is not in the Committed state? because the changes made by the transaction are still residing in the main memory buffer. This is a temporary storage area that the DBMS uses to hold modifications made by transactions before they are written to the actual database files on the disk.
     - → And according to ACID (Atomicity, Consistency, Isolation, Durability) properties, a transaction is only considered complete and "Committed" when its changes are permanently stored in the database.
     - → Why it stores in a buffer before writing them to actual database file disk? Less time consuming because writing data to a disk is relatively slow process compared to writing it in a buffer.

→ And kapag may error kang naencounter, ung buffer lang natin ung maaapektuhan hindi ung pinaka database file disk

3. Committed state
   → This can be considered as the completion of a transaction.
   → All the changes stored in the buffer are transferred to the databases.
   → In other words, all the instructions of a respective transaction are successfully completed, and changes are made in the database. So if may need ka baguhin sa transaction, the only option is to execute another transaction
4. Failed state
   → If the transaction is active or partially committed and it experiences a failure that prevents it from continuing to execute. Then, the transaction is said to be in a failed state.
   → Reasons for failure? This could be an error within the transaction itself. For example ung transaction natin nag aattempt to insert data na hindi tugma sa data types or formats. Or it can error in applications code
      1. Transaction failure – transaction failure could be logical error where transaction cannot complete due to some code error or syntax error where the dbms itself terminates an active transaction because the database is not able to execute it.
      2. System crash – it can occur due to power failure or other hardware or software failure. It could also be an error within the system for example, hardware failures such as memory corruption or processor malfunctions. software failures, power failures, network failures etc.
5. Aborted state
   → This means that it was not able to complete its transactions due to a failure.
   → The database is rolled back to the state it was in before the transaction started.
   → The DBMS recovery system performs one of the two operations when a system reaches to aborted state:
      1. Kill the transaction – the system will delete the transaction so it cannot affect other operations
      2. Re-start the transaction – the system sends the operation to an active state
6. terminated state
   → Final state
   → It indicates that transaction has either been committed or aborted and is now complete

Operations involved in Transaction management in dbms

〉 The operations involved in transaction management allows data manipulation and ensure transactional integrity (ineensure nya na ung characteristic ng data natin is reliable, consistent, etc.)
〉 Eto ung mga task na pineperform sa data within the database
〉 Here are the main operations involved in transaction management:

1. Read
   → This operation selects relevant information and enables us to access and view the data's current state.

- → this operation reads the information from the database and stores the relevant data in a buffer in the main memory.
- → A read operation retrieves data from the database.
- → In some cases, depending on the isolation level, a read operation may also acquire a shared lock on the data to prevent it from being modified by other transactions before the read operation is completed.

2. Write
   - → used for writing the information back into the database from the buffer.
   - → This operation updates, inserts, or deletes records.
   - → typically acquires an exclusive lock on the data being modified, ensuring that no other transaction can simultaneously modify the same data.
3. Commit
   - → The commit statement saves the changes made after the transaction.
   - → Once a transaction has completed all of its read and write operations, and it has ensured that all of its operations have executed successfully without any errors, it issues a commit operation.
   - → This operation signals that the transaction is ready to be finalized, and all its changes should be made permanent in the database.
   - → Any locks held by the transaction are released, allowing other transactions to access the modified data.
4. Rollback
   - → The rollback operation undoes all the changes made by a transaction, reverting the database to a state prior to the start of the transaction.
   - → used to undo the changes made by a transaction in case of an error or an explicit request to abort the transaction. It reverts the database to its state before the transaction began.
5. Savepoint
   - → A savepoint operation allows a transaction to set a point within itself. This point can be used to return to if a rollback is necessary.

How does the buffer help in ensuring data integrity during a transaction?

Buffer temporary storage area for changes made by the transaction before they are written to the actual database files on the disk.

References

https://www.codingninjas.com/studio/library/transaction-management-in-dbms?fbclid=IwAR3OKsg5zoiAzwaElnfWIBlx5j4SXnQ-gWuEXOPLta-mKiJtHlo33Zpsk3Q

https://www.codingninjas.com/studio/library/states-of-transaction-in-dbms