**Read committed isolation**

Many read threads , many write threads , everything is excueting same time , so we have to ensure the correctness. It introduces race condition.

A screen shot of a blackboard

Description automatically generatedA screen shot of a black board

Description automatically generatedA screen shot of a black screen

Description automatically generated

Dirty reads – reading over uncommitted data

Dirty writes – writing over uncommitted write

Row level lockings

We can use locks , but It is very slow . If one lock is writing , then writes are slow, so our reads will get slow too. So , what can we do is , we can store old value somewhere until it is committed , once done update with new value , but it is creating storage overhead.

Read committed" is one of the isolation levels in database management systems (DBMS). Isolation levels define how transactions interact with each other and the level of visibility of changes made by concurrent transactions. In the context of "read committed" isolation:

Read Committed Isolation Level: In this isolation level, a transaction can only read data that has been committed by other transactions. This means that when a transaction reads a data item, it sees the most recent committed value of that item in the database at the time the read operation is performed. However, it doesn't necessarily guarantee consistency throughout the entire transaction, as the data could be modified by other transactions before the current transaction completes.

Characteristics:

Read committed isolation level provides a higher level of concurrency than stricter isolation levels like "serializable."

It prevents dirty reads, meaning that transactions cannot read uncommitted data from other transactions.

However, it does allow non-repeatable reads and phantom reads. Non-repeatable reads occur when a transaction reads the same data multiple times during its execution and gets different results because the data has been updated by other transactions in between. Phantom reads occur when a transaction reads a set of rows that satisfy a certain condition, but another transaction inserts or deletes rows that also meet that condition before the first transaction completes.

Usage:

Read committed isolation level is commonly used in situations where data consistency is important but strict isolation is not necessary.

It strikes a balance between data consistency and concurrency, making it suitable for most transactional applications where a moderate level of isolation is sufficient.

Implementation:

Read committed isolation level is typically implemented using locking mechanisms or multiversion concurrency control (MVCC) techniques, depending on the specific database system.

Lock-based implementations involve acquiring locks on data items during read and write operations to ensure consistency and prevent conflicts.

MVCC implementations create snapshots of data at the beginning of each transaction and allow transactions to read from these snapshots, ensuring consistency while allowing concurrent access to data.

In summary, "read committed" isolation level ensures that transactions read only committed data, providing a balance between data consistency and concurrency in database systems. However, it does allow phenomena like non-repeatable reads and phantom reads, which should be considered based on the requirements of the application.