https://mydbops.wordpress.com/2018/06/22/back-to-basics-isolation-levels-in-mysql/

In this blog, we will see the very basic thing “I” of “ACID” and an important property of Transaction ie., “ISOLATION”

The isolation defines the way in which the MySQL server (InnoDB) separates each transaction from other concurrent running transaction in the server and also ensures that the transactions are processed in a reliable way. If transactions are not isolated then one transaction could modify the data that another transaction is reading hence creating data inconsistency. Isolation levels determine how isolated the transactions are from each other.

MySQL supports all four the isolation levels that SQL-Standard defines.The four isolation levels are

- READ UNCOMMITTED
- READ COMMITTED
- REPEATABLE READ
- SERIALIZABLE

The Isolation level’s can be set globally or session based on our requirements.

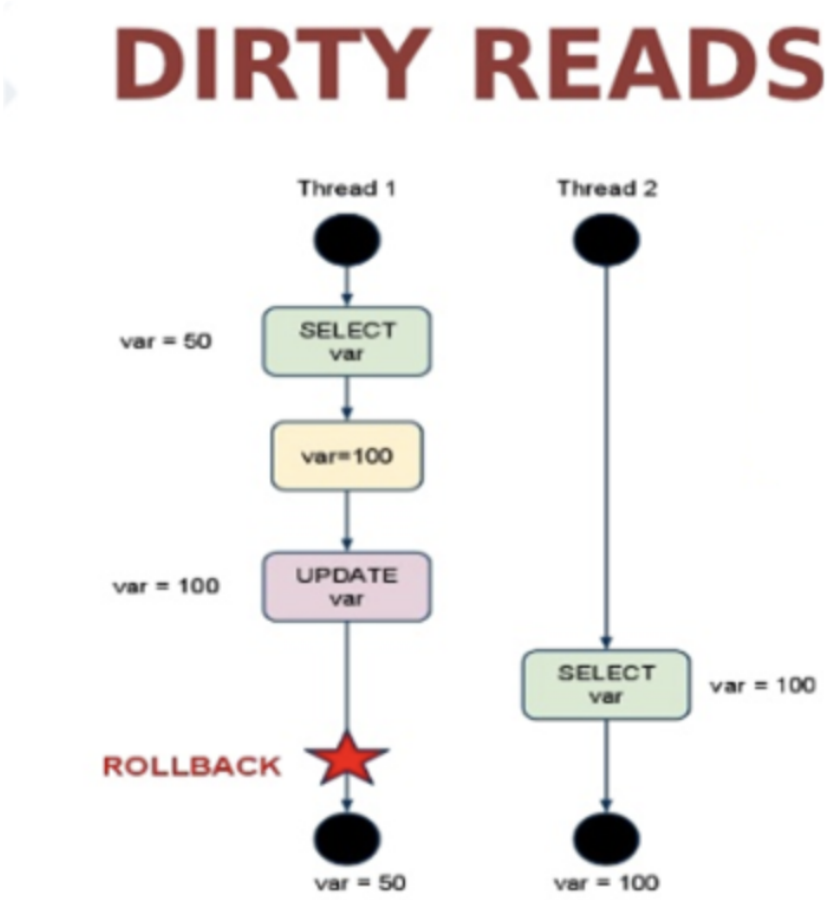
Choosing the best isolation level based, have a great impact on the database, Each level of isolation comes with a trade-off, let’s discuss on each of them,

READ UNCOMMITTED:

In READ-UNCOMMITTED isolation level, there isn’t much isolation present between the transactions at all, ie ., No locks. A transaction can

see changes to data made by other transactions that are not committed yet. This is the lowest level in isolation and highly performant since there is no overhead of maintaining locks, With this isolation level, there is always for getting a "**Dirty-Read**"

That means transactions could be reading data that may not even exist eventually because the other transaction that was updating the data rolled-back the changes and didn't commit. lest see the below image for better understanding

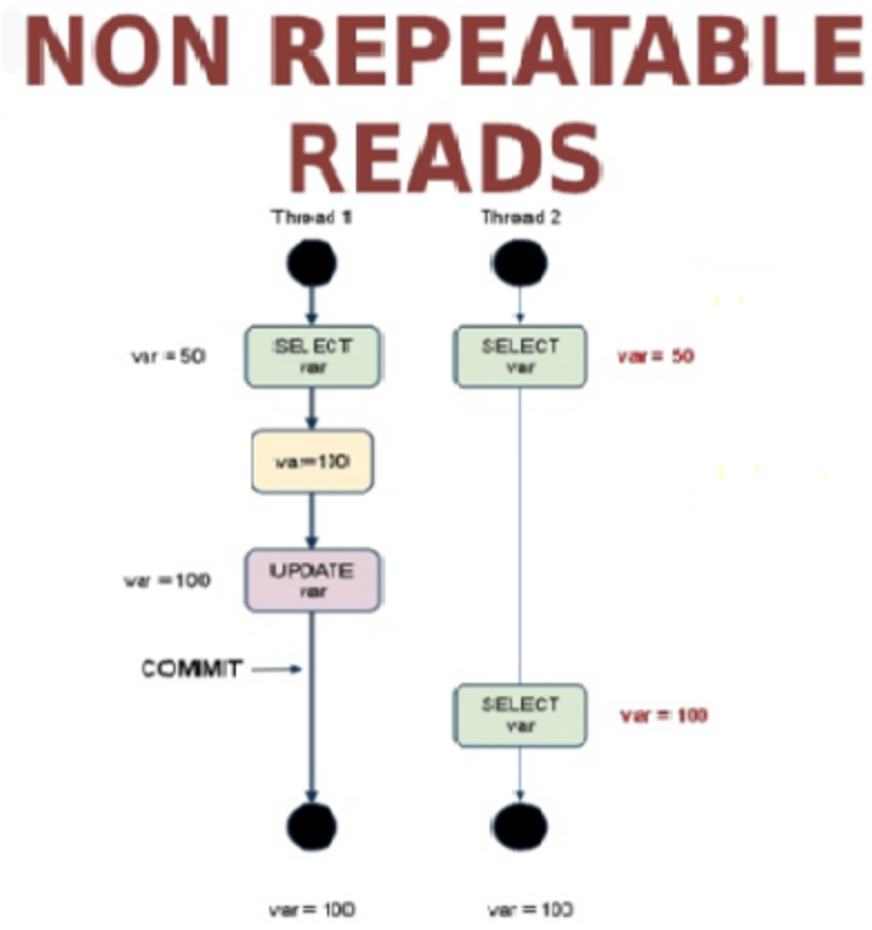


Suppose a transaction T1 modifies a row if a transaction T2 reads the row and sees the modification even though T1 has not committed it, that is a dirty read, the problem here is if T1 rolls back, T2 doesn't know that and will be in a state of "totally perplexed"

READ COMMITTED:

IN READ-COMMITTED isolation level, the phenomenon of dirty read is avoided, because any uncommitted changes are not visible to any other transaction until the change is committed. This is the default isolation level with most of popular RDBMS software, but not with MySQL.

Within this isolation level, each SELECT uses its own snapshot of the committed data that was committed before the execution of the SELECT. Now because each SELECT has its own snapshot, here is the trade-off now, so the same SELECT, when running multiple times during the same transaction, could return different result sets. This phenomenon is called **non-repeatable read**.



A non-repeatable occurs when a transaction performs the same transaction twice but gets a different result set each time. Suppose T2 reads some of the rows and T1 then change a row and commit the change, now T2 reads the same row set and gets a different result ie.., the initial read is non-repeatable.

Read-committed is the recommended isolation level for Galera (PXC, MariaDB Cluster) and InnoDB clusters.

REPEATABLE READ:

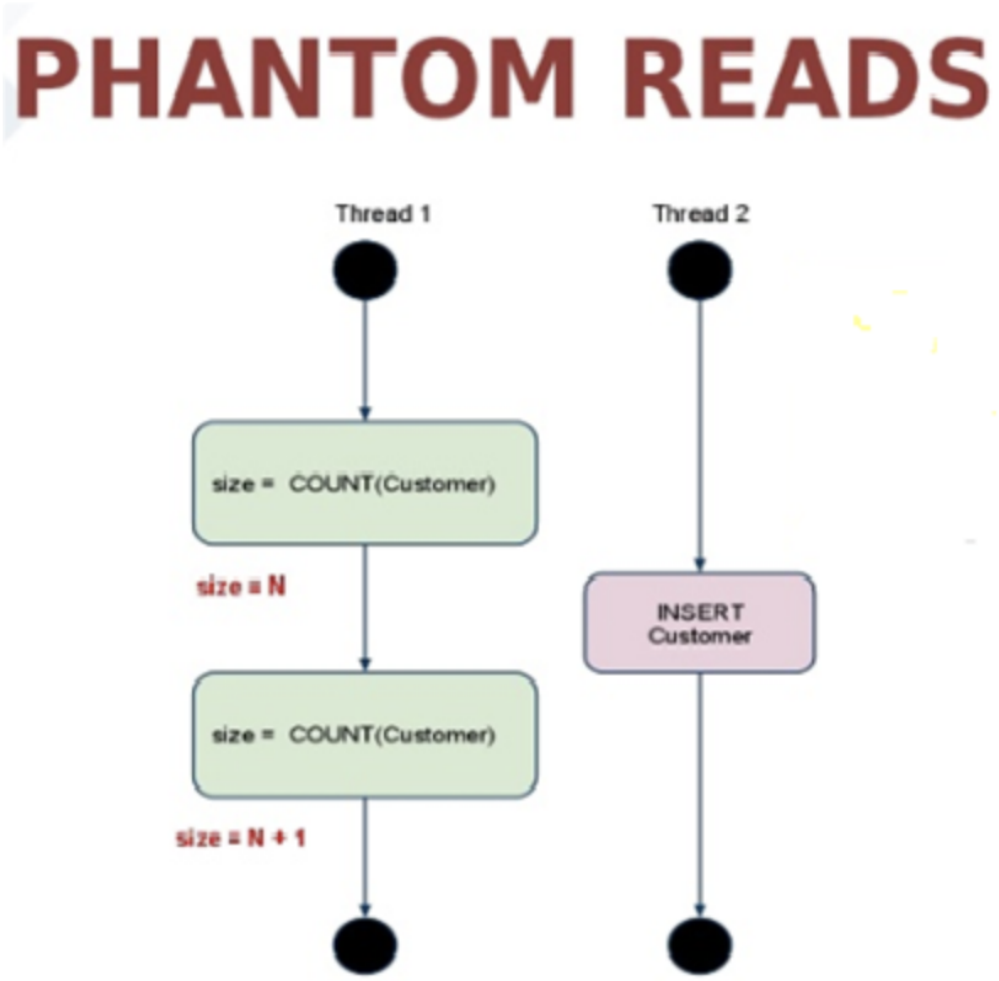
In REPEATABLE-READ isolation level, the phenomenon of non-repeatable read is avoided. It is the default isolation in MySQL.This isolation level returns the same result set throughout the transaction execution for the same SELECT run any number of times during the progression of a transaction.

This is how it works, a snapshot of the SELECT is taken the first time the SELECT is run during the transaction and the same snapshot is used throughout the transaction when the same SELECT is executed. A transaction running in this isolation level does not take into account any changes to data made by other transactions, regardless of whether the changes have been committed or not. This ensures that reads are always consistent(repeatable). Maintaining a snapshot can cause extra overhead and impact some performance

Although this isolation level solves the problem of non-repeatable read, another possible problem that occurs is **phantom reads**.

A Phantom is a row that appears where it is not visible before. **InnoDB** and **XtraDB** solve the phantom read problem with multi-version concurrency control.

REPEATABLE READ is MySQL's default transaction isolation level.



SERIALIZABLE

SERIALIZABLE completely isolates the effect of one transaction from others. It is similar to REPEATABLE READ with the additional restriction that row selected by one transaction cannot be changed by another until the first transaction finishes. The phenomenon of phantom reads is avoided. This isolation level is the strongest possible isolation level. AWS Aurora do not support this isolation level.