Using Logs to Redo and Undo Operations in MySQL

In MySQL, the concepts of REDO and UNDO logs are used to maintain data consistency and durability. These logs are part of the InnoDB storage engine's crash recovery and transaction management mechanism.

1. REDO Logs

- Purpose: REDO logs store the changes that have been made to the database so that they can be reapplied in case of a crash before the changes were permanently written to the data files. - Process: 1. A transaction modifies data in memory (buffer pool). 2. The change is recorded in the REDO log. 3. If the system crashes before flushing changes to the disk, MySQL uses the REDO log during recovery to reapply the modifications. - Example: If you INSERT a row and MySQL crashes, the REDO log ensures that the row is still inserted upon recovery.

2. UNDO Logs

- Purpose: UNDO logs store the information needed to roll back a transaction. - Process: 1. Before modifying data, the original value is saved in the UNDO log. 2. If a transaction is rolled back, MySQL restores the old values from the UNDO log. 3. This ensures that partially completed transactions do not affect the database state. - Example: If you UPDATE a row and then ROLLBACK, the UNDO log restores the original row values.

3. Combined Role in Crash Recovery

- REDO logs: Ensure durability by reapplying committed changes after a crash. - UNDO logs: Ensure atomicity by rolling back incomplete transactions. - Together, they help maintain ACID properties in MySQL.