Mayank Ahuja • Following

Follow for Your Daily Dose of AI, Software Development & System Design Tips | Exploring AI SaaS - Tink...

View my blog

3h •

Give me 5 minutes, and I'll explain isolation levels in databases.

📌 Isolation levels are a fundamental concept in database concurrency control.

Some background -

▪ ACID is a set of properties that guarantee reliable database transactions.

▪ Though many databases claim to be 'ACID compliant' they often implement weaker isolation levels by default.

▪ Different applications have different needs for isolation.

▪ Some may prioritize performance and tolerate some inconsistencies, while others may require strict data integrity.

▪ Concurrency, the simultaneous execution of multiple transactions, poses a significant challenge for data systems.

▪ If not managed carefully, it can lead to anomalies such as

👉 dirty reads

👉 non-repeatable reads

👉 phantom reads

(I have written a post earlier on these anomalies, I will put it in the comments)

📌 Isolation levels define how much a transaction is protected from the effects of concurrent transactions.

Let's talk about different Isolation Levels -

[1.] Read Uncommitted (the lowest isolation level)

- Transactions are not isolated from each other.
- All transactions can see the effects of other concurrent transactions, even if those transactions are not yet committed.
- Data read in this isolation level may be rolled back later, leading to dirty reads.

[2.] Read Committed

- A transaction running in this isolation level does not see 'dirty' data from other transactions.
- It sees only committed data.
- Read Committed typically uses shared locks on the data being read.
- This isolation level avoids dirty reads but can still result in non-repeatable reads. (it doesn't guarantee that the data you read will remain the same throughout your transaction)

[3.] Repeatable Read

- The core principle of Repeatable Read is to provide each transaction with a consistent snapshot of the database at the beginning of the transaction.
- This snapshot is like a frozen picture of the database.
- Any changes made by other transactions after this point are not visible to the current transaction until it commits.

Two ways to implement it

1. Locking Reads - places shared locks on all rows read by a transaction
2. Multi-Version Concurrency Control (MVCC) - maintains multiple versions of each row, with each version associated with a timestamp or transaction ID

- Eliminates Non-Repeatable Reads

[4.] Serializable

- This level provides the strictest transaction isolation.
- It simulates serial execution of transactions, as if transactions had been executed one after the other, serially, rather than concurrently.

▪ Involves stricter locking mechanisms, such as range locks or predicate locks.

📌 Isolation levels offer a trade-off between data consistency and performance.

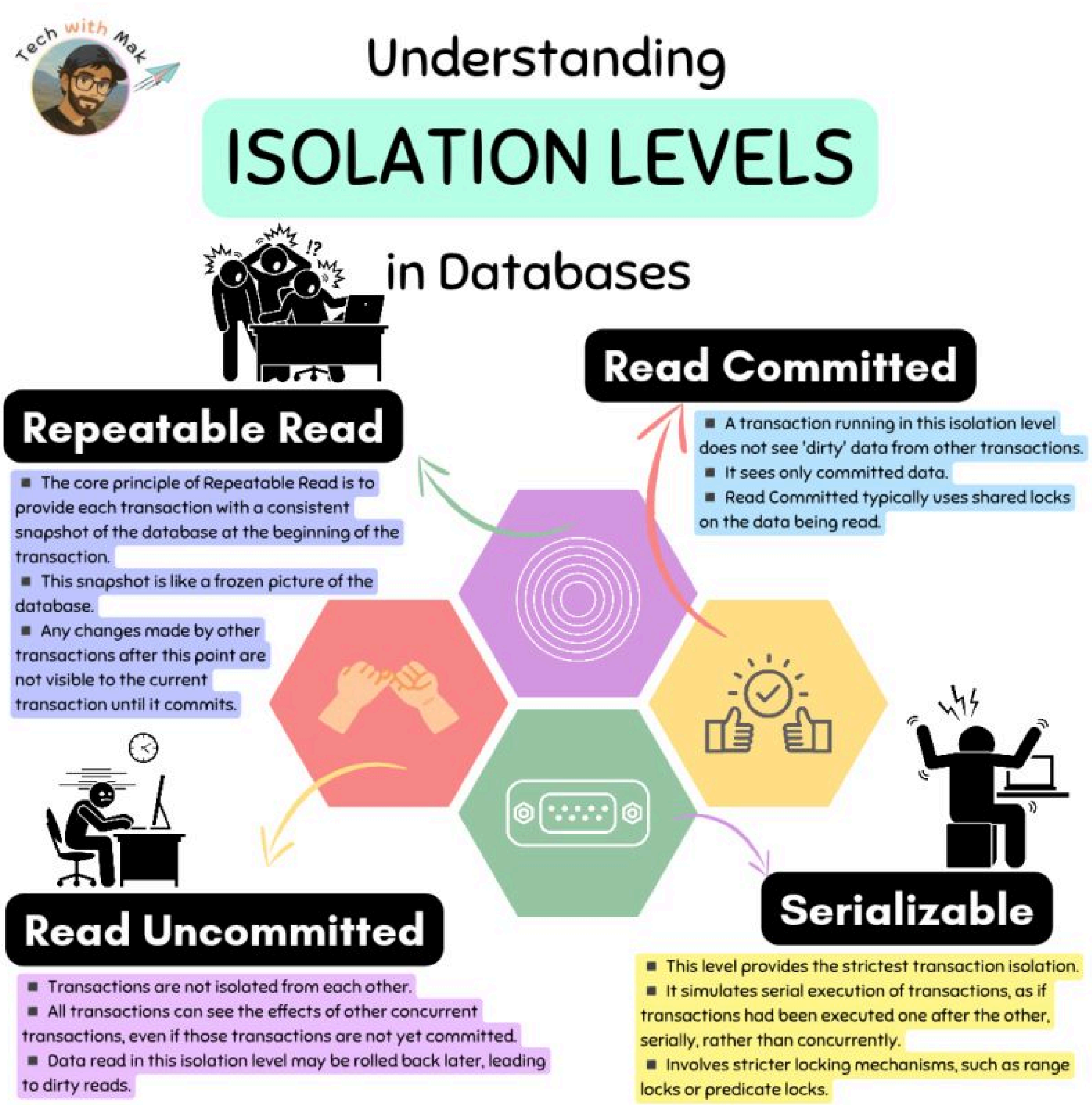
Higher isolation levels provide stronger guarantees of data integrity but may incur performance overhead due to locking or transaction serialization.

Choose wisely.

——

📒 Newsletter - https://lnkd.in/dJByxEYY

#databases



Understanding

ISOLATION LEVELS

in Databases

Read Committed

- A transaction running in this isolation level does not see 'dirty' data from other transactions.
- It sees only committed data.
- Read Committed typically uses shared locks on the data being read.

Repeatable Read

- The core principle of Repeatable Read is to provide each transaction with a consistent snapshot of the database at the beginning of the transaction.
- This snapshot is like a frozen picture of the database.
- Any changes made by other transactions after this point are not visible to the current transaction until it commits.

Serializable

- This level provides the strictest transaction isolation.
- It simulates serial execution of transactions, as if transactions had been executed one after the other, serially, rather than concurrently.
- Involves stricter locking mechanisms, such as range locks or predicate locks.

Read Uncommitted

- Transactions are not isolated from each other.
- All transactions can see the effects of other concurrent transactions, even if those transactions are not yet committed.
- Data read in this isolation level may be rolled back later, leading to dirty reads.

121

40 comments 9 reposts