Transaction Isolation Levels

4 Isolation Levels

- READ UNCOMMITTED
  - Provides lowest level of isolation among transactions but best performing
- READ COMMITTED
- REPETABLE READ
- SERIALIZABLE
  - Provides the highest level of isolation among transactions but least performing

READ UNCOMMITTED Isolation level

- Causes 'dirty reads' symptom
  - > Uncommitted changes in one transaction (client #1 below) is visible in other transactions (client #2 below)

Client #1: Start Transaction----INSERT a record A----------ROLLBACK---------->

Client #2: Start Transaction----------(See Record A:Dirty read)----------(Does not see Record A)-->

READ COMMITTED Isolation level

- Committed updates in one transaction (client # 1 below) are visible within another transaction (client #2 below)
- This means identical queries within a transaction (in client #2 below) can return differing results causing “Non-repeatable read” problem

Client #1: Start Transaction----INSERT a record A----------COMMIT---------->

Client #2: Start Transaction----------(Not see Record A)---------(See Record A)------COMMIT------>

REPEATABLE READ Isolation level

- Committed changes in one transaction (client #1 below) is visible in another transaction (client #2 below) only after its own Commit.
  - Within a transaction, all reads are consistent.
- The default isolation level for InnoDB tables.

Client #1: Start Transaction----INSERT a record A----------COMMIT---------->

Client #2: Start Transaction----------(Not see Record A)----------COMMIT--(See Record A) ---->

SERIALIZABLE Isolation level

- Transactions are serialized
- Until the previous transaction ends via either COMMIT or ROLLBACK, the database operations in other transactions are blocked
- Highest isolation level but not practical in real-life environment due to its low performance

Client #1: Start Transaction----INSERT a record A----------COMMIT---------->

Client #2: Start Transaction----------(SELECT Blocked)---------- (SELECT returns)--COMMIT---->

Code with Passion!