Transaction Isolation Levels

Piyush Jain

Friday, June/17/2022

YFTT
YUGABYTEDB
FRIDAY
TECH TALKS

yugabyteDB

Isolation levels in PostgreSQL

1. **Read Committed** (default PostgreSQL isolation that never throws serialization errors).
2. **Repeatable Read** (aka Snapshot Isolation, writes conflict with writes and results in serialization errors).
3. **Serializable** (successfully executed transactions behave as if they executed in some serial order. If serial order can't be guaranteed for a set of active transactions, serialization errors are thrown to abort some of them).

In Repeatable Read and Read Committed, **readers don't block writers and vice-versa**. This is achievable due to MVCC (Multi Version Concurrency Control).

Isolation levels in PostgreSQL

1. **Read Committed** (default PostgreSQL isolation that never throws serialization errors).
2. **Repeatable Read** (aka Snapshot Isolation, writes conflict with writes and results in serialization errors).
3. **Serializable** (successfully executed transactions behave as if they executed in some serial order. If serial order can't be guaranteed for a set of active transactions, serialization errors are thrown to abort some of them).

In Repeatable Read and Read Committed, **readers don't block writers and vice-versa**. This is achievable due to MVCC (Multi Version Concurrency Control).

Isolation levels in YugabyteDB's YSQL

1. **Read Committed** (**New!**)
2. **Repeatable Read** (a beneficial difference from PostgreSQL is that column level locks allow concurrent updates to a row)
3. **Serializable**

YugabyteDB also uses MVCC. So **readers don't block writers and vice-versa** in Repeatable Read and Read Committed.

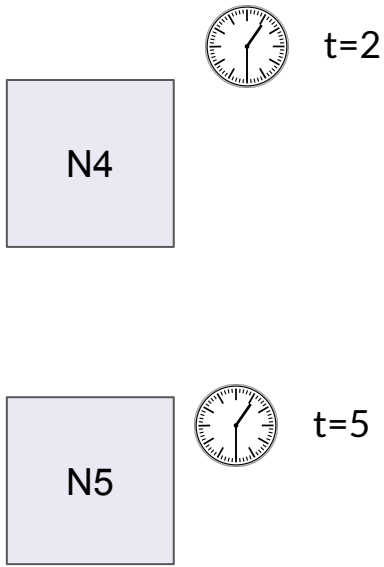
Read Committed (beta feature)

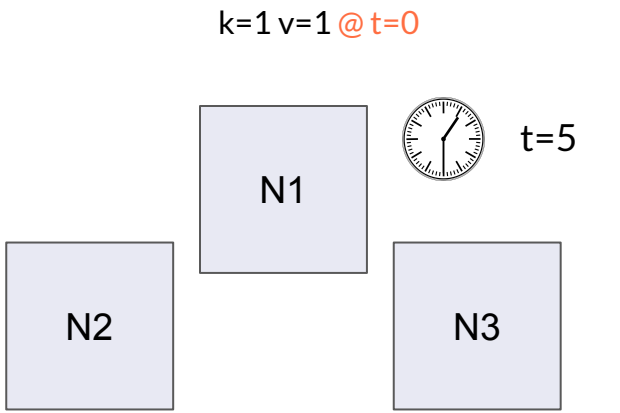
1. New snapshot per statement in the transaction
2. Pessimistic locking is built in
3. Helps avoid below errors by retrying the statement -
    a. Conflicts
    b. Read restart errors

Usage: set tserver gflag `yb_enable_read_committed_isolation=true`

By default, Read Committed in YSQL to maps to Snapshot Isolation (i.e., Repeatable read). If the flag is set to true, it maps to the new Read Committed implementation.

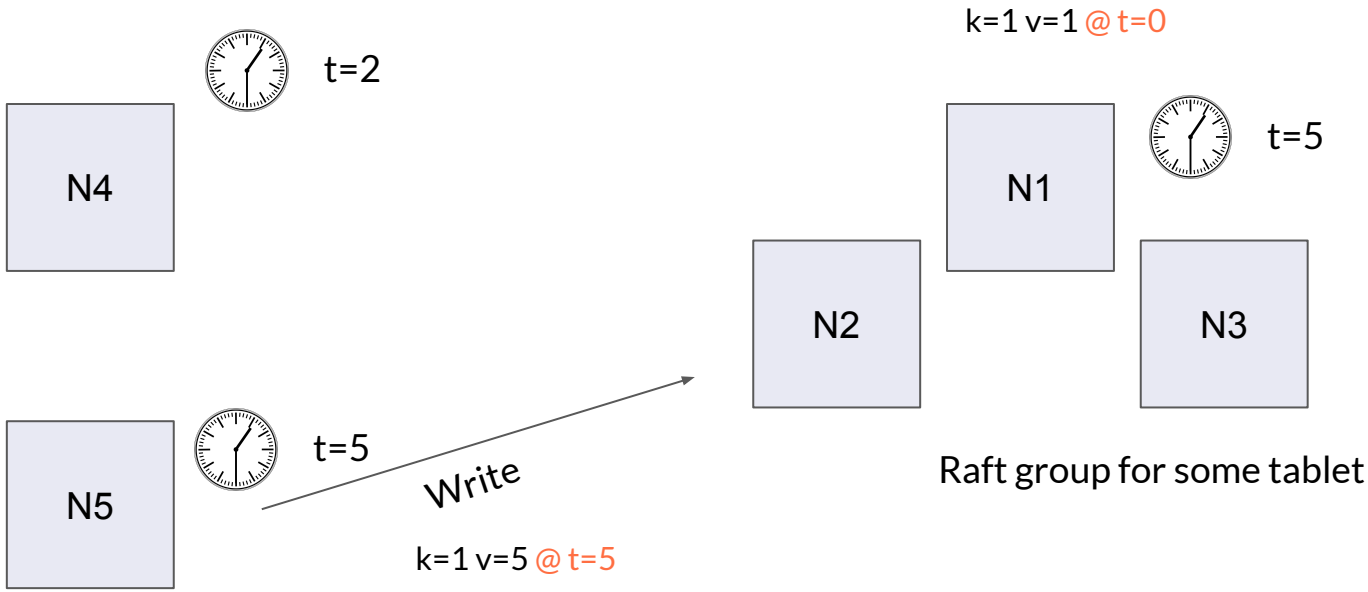
A 2 min. detour to declutter read restarts…





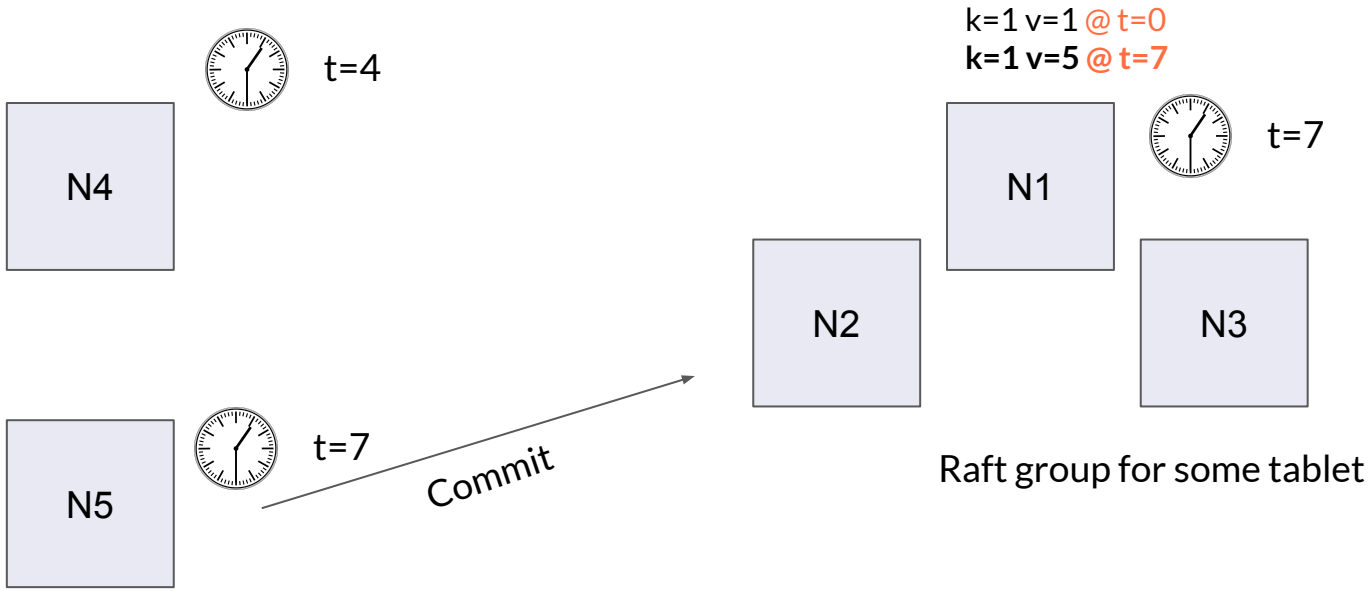
Raft group for some tablet

Assume max clock skew = 8 units



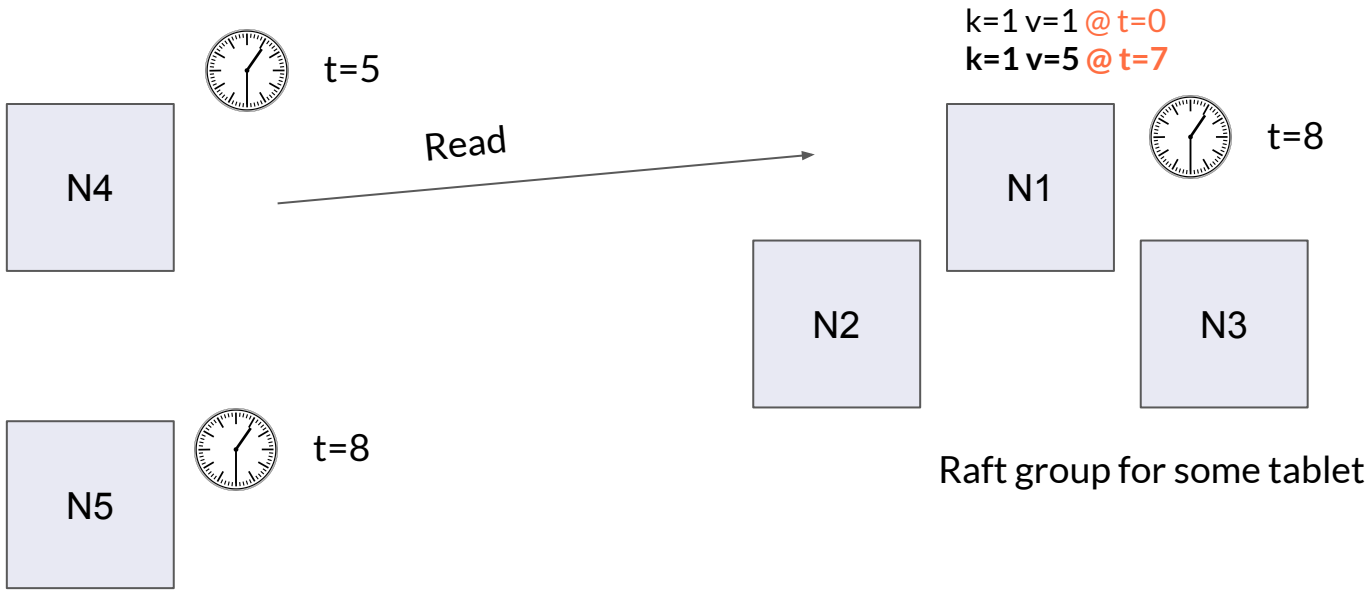
Assume max clock skew = 8 units

2 times units later .....



Assume max clock skew = 8 units

1 time unit later .....



Assume max clock skew = 8 units

The Temenos High Water Benchmark in (big) numbers

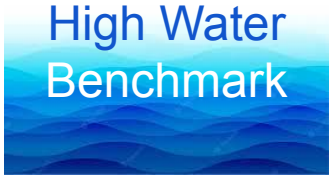
temenos

3000 Global Banking Customers

41/50 Of The Top Global Banks

1.2 Bn Global Bank Customers

Investing **20%** Revenue in R&D



102K Business Transactions Per Second

100M Customers

200M Accounts

4.1x More Efficient For A Smaller CO_2 Footprint

+40% Better Performance

yugabyteDB

350K Database Reads Per Second

80K Database Writes Per Second

Inserts **3** ms
Selects **1** ms
Deletes **1** ms

39/3 DB Nodes / AWS AZ

Roadmap

Roadmap

1. Read Committed (ETA: GA in 2.14.*)

   Known limitations:

   a. Different semantics for volatile functions and procedures
   b. Limitation on ysql_output_buffer_size for read committed semantics
   c. Rely on statement timeout to avoid deadlocks in read committed isolation

2. Improved Pessimistic locking (ETA: Beta in 2.15.*)
   a. Move from exponential back-off polling to signalling mechanism
   b. Detect distributed deadlocks
   c. Extend semantics to repeatable read and serializable isolation levels

3. Consider making Read Committed as the default isolation level in future releases

GH issue #5683 is tracking all of this and more

Thank You

Join us on Slack: yugabyte.com/slack (#yftt channel)

Star us on Github: github.com/yugabyte/yugabyte-db

YFTT YUGABYTEDB FRIDAY TECH TALKS

yugabyteDB