# Giới thiệu

# Storage Engine

## WiredTiger

<https://docs.mongodb.com/manual/core/wiredtiger/>

### Snapshots

(Checkpoints)

A snapshot presents a consistent view of the in-memory data

WiredTiger sử dụng MultiVersion Concurrency Control (MVCC)

với mỗi “operation” sẽ tương tác với một “snapshot” của data tức nhất quán với mỗi “operation”

và độc lập với các operations khác

Khi ghi dữ liệu WiredTiger ghi toàn bộ data trong “snapshot” tạo ra một “checkpoint” nằm trong files đã ghi xuống

Khi tạo một “checkpoint” mới “checkpoint” cũ vẫn giữ nguyên

chỉ khi “checkpoint” mới và “meta data” đã được ghi xuống files và có thể truy cập thì “checkpoint” cũ sẽ được giải phóng

do vậy nếu trong quá trình ghi "checkpoint" mới có xảy ra lỗi thì có thể khôi phục từ checkpoint cũ

Checkpoint có thể được sử dụng làm “recovery points”

Từ bản 3.6 MongoDB tự động tạo checkpoint mỗi 60s

### Journal

WiredTiger sử dụng write-ahead log (WAL) và chứa log “all data modifications” giữa 2 “checkpoint”

WiredTiger sử dụng “checkpoint” và “journal” để đảm bảo tính an toàn dữ liệu bằng cách

"replay" lại tất cả chỉnh sửa từ lần cuối "checkpoint"

Journal Process

<https://docs.mongodb.com/v4.4/core/journaling/#std-label-journal-process>

Nếu tắt thì khi thoát, crash mongod bạn sẽ mất toàn bộ chỉnh sửa từ lần cuối checkpoint

## In-Memory

(Enterprise)

# Backup - Restore

<https://docs.mongodb.com/manual/core/backups/>

## MongoDB Tools

<https://docs.mongodb.com/manual/tutorial/backup-and-restore-tools/>

mongodb version: 5.0, 4.4, 4.2, 4.0

mongodump

<https://docs.mongodb.com/database-tools/mongodump/#mongodb-binary-bin.mongodump>

- dump từ primary

mongodump --uri="mongodb://vm-01.local:27017,vm-02.local:27017/?replicaSet=rs0"

mongodump --uri="mongodb://vm-01.local:27017,vm-02.local:27017/?replicaSet=rs0" -u admin -p admin

- dump từ secondary

mongodump --uri="mongodb://vm-01.local:27017/?replicaSet=rs0&readPreference=secondary"

mongodump --uri="mongodb://vm-01.local:27017/?replicaSet=rs0&readPreference=secondary" -u admin -p admin

- mongodump có tương tác với instance dẫn tới ảnh hưởng performance khi chạy (traffic, memory)

<https://docs.mongodb.com/manual/tutorial/backup-and-restore-tools/#performance-considerations>

- dump từ secondary không ảnh hưởng primary

- mongodump output dữ liệu tại máy chạy lệnh

- mặc định dump toàn bộ các database (trừ local) và collection

- cấu trúc indexes của collection được dump theo data (để mongorestore và mongod tự build lại)

<https://docs.mongodb.com/database-tools/mongodump/#data-exclusion>

mongorestore

<https://docs.mongodb.com/database-tools/mongorestore/#mongodb-binary-bin.mongorestore>

mongorestore mongodb://vm-01.local:27017,vm-02.local:27017,vm-03.local:27017 dump/

mongorestore mongodb://vm-01.local:27017,vm-02.local:27017,vm-03.local:27017 dump/ -u admin -p admin

- indexes data khi mongodump được build lại (sau khi dữ liệu được load hết)

- version server mongodump phải match version mongorestore VD: 4.4.x và 4.4.x //nhưng test dữ liệu thông thường thì vẫn OK 😀

trình tự khôi phục

B1: (nếu cần) restart các node die

B2: kết nối tới 1 node bất kì

B3: (nếu cần) khởi tạo lại replica set

rs.initiate( {

\_id : "rs0",

members: [

{ \_id: 0, host: "vm-01.local:27017" },

{ \_id: 1, host: "vm-02.local:27017" },

{ \_id: 2, host: "vm-03.local:27017" }

]

})

B4: khôi phục

mongorestore mongodb://vm-01.local:27017,vm-02.local:27017,vm-03.local:27017 dump/

## Copy Files

trình tự backup

copy files với những công cụ như cp, scp, ...

để đảm bảo dữ liệu đã được ghi xuống đĩa và trong quá trình backup không có bất kì update nào được thực thi

chúng ta cần lock db và flush toàn bộ snapshot từ RAM xuống files

<https://docs.mongodb.com/manual/reference/method/db.fsyncLock/>

db.fsyncLock()

sau khi lock việc copy files có thể diễn ra an toàn

khi copy xong chúng là unlock instance

<https://docs.mongodb.com/manual/reference/method/db.fsyncUnlock/>

db.fsyncUnlock()

trình tự khôi phục

<https://docs.mongodb.com/v4.4/tutorial/restore-replica-set-from-backup/>

khởi bật một instance với mode standalone

kết nối tới instance đó và drop database local ?

use local

db.drop()

restart instance đó với mode replica set

khởi tạo primary replica set trên chính instance này

add lần lượt lại các members (các member có db rỗng)

## Filesystem Snapshot

<https://docs.mongodb.com/manual/tutorial/backup-with-filesystem-snapshots/>

<https://docs.mongodb.com/manual/tutorial/backup-with-filesystem-snapshots/#std-label-lvm-backup-and-restore>

create snapshot

<https://docs.mongodb.com/manual/tutorial/backup-with-filesystem-snapshots/#create-a-snapshot>

lvcreate --size 100M --snapshot --name mdb-snap01 /dev/vg0/mongodb

archive a snapshot

<https://docs.mongodb.com/manual/tutorial/backup-with-filesystem-snapshots/#archive-a-snapshot>

umount /dev/vg0/mdb-snap01

dd if=/dev/vg0/mdb-snap01 | gzip > mdb-snap01.gz

restore a snapshot

<https://docs.mongodb.com/manual/tutorial/backup-with-filesystem-snapshots/#restore-a-snapshot>

lvcreate --size 1G --name mdb-new vg0

gzip -d -c mdb-snap01.gz | dd of=/dev/vg0/mdb-new

mount /dev/vg0/mdb-new /srv/mongodb

<https://docs.mongodb.com/manual/tutorial/backup-with-filesystem-snapshots/#restore-directly-from-a-snapshot>

<https://docs.mongodb.com/manual/tutorial/backup-with-filesystem-snapshots/#remote-backup-storage>

...

<https://docs.mongodb.com/manual/tutorial/restore-replica-set-from-backup/>

# Performance

## Database Profiler

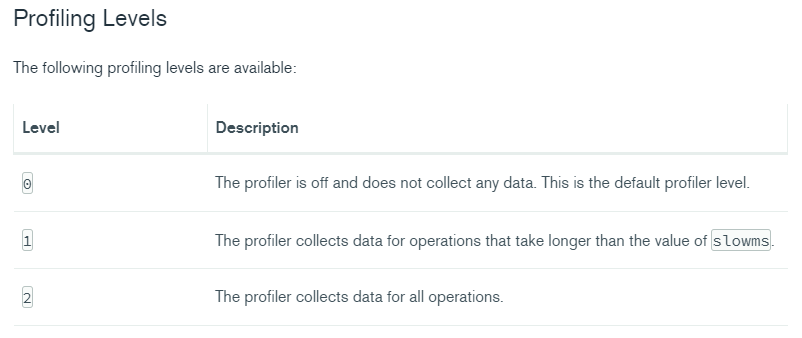
<https://docs.mongodb.com/manual/tutorial/manage-the-database-profiler/>

mongodb cho phép lưu lại những "database command" đã được thực thi trong một mongod instance

"database command" này bao gồm CRUD, administration command

những thông tin này lưu trong collections system.profile (capped collection) nằm trong database mà operations này tương tác

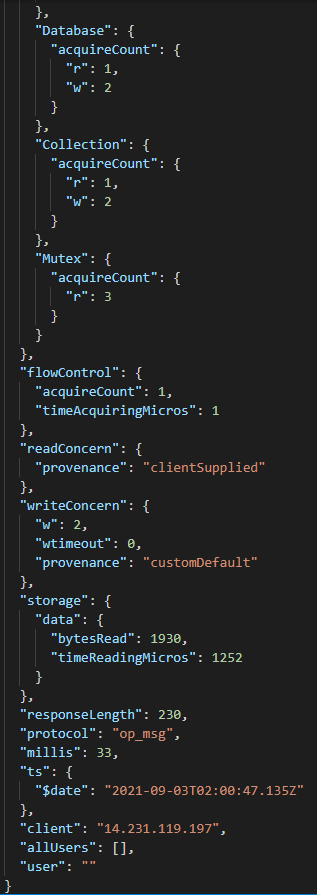
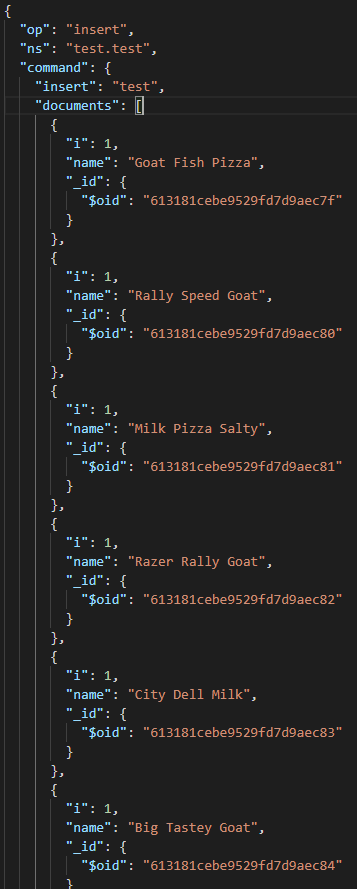
những level mà profiler collect dữ liệu



để bật tính năng profiler

db.setProfilingLevel(2) // 2 là level trong bảng ở trên

một vài ví dụ của một details operation trong collections system.profile



## Config

Connection Pool

<https://docs.mongodb.com/manual/reference/configuration-options/#mongodb-setting-net.maxIncomingConnections>

phía client driver mặc định sử dụng connections pool

config số lượng connection đồng thời tối đa cho một mongod hoặc mongos

config này hữu ích cho mongos khi client tạo nhiều kết nối và cho phép connection "timeout" thay vì bị đóng

do vậy nên set giá trị cao hơn số lượng tối đa clients tạo kết nối or cao hơn size của connections pool

config này cũng giúp mongos không bị "connections spikes" tức số lượng connection cao đột biến

(tức nếu vượt quá config các connections sau đó có thể bị drop ?)

những spike này sẽ gây ảnh hưởng tới các operation và "memory allocation" trên các shards

net.maxIncomingConnections

Type: integer

Default: 65536

Cache

<https://docs.mongodb.com/manual/reference/configuration-options/#mongodb-setting-storage.wiredTiger.engineConfig.cacheSizeGB>

internal cache nằm trên RAM chứa các indexes và collections để giảm disk IO

giá trị nằm từ 0.25 GB tới 10000 GB

storage.wiredTiger.engineConfig.cacheSizeGB

Type: float

Default: 50% of (RAM - 1 GB) or 256 MB

# Reference