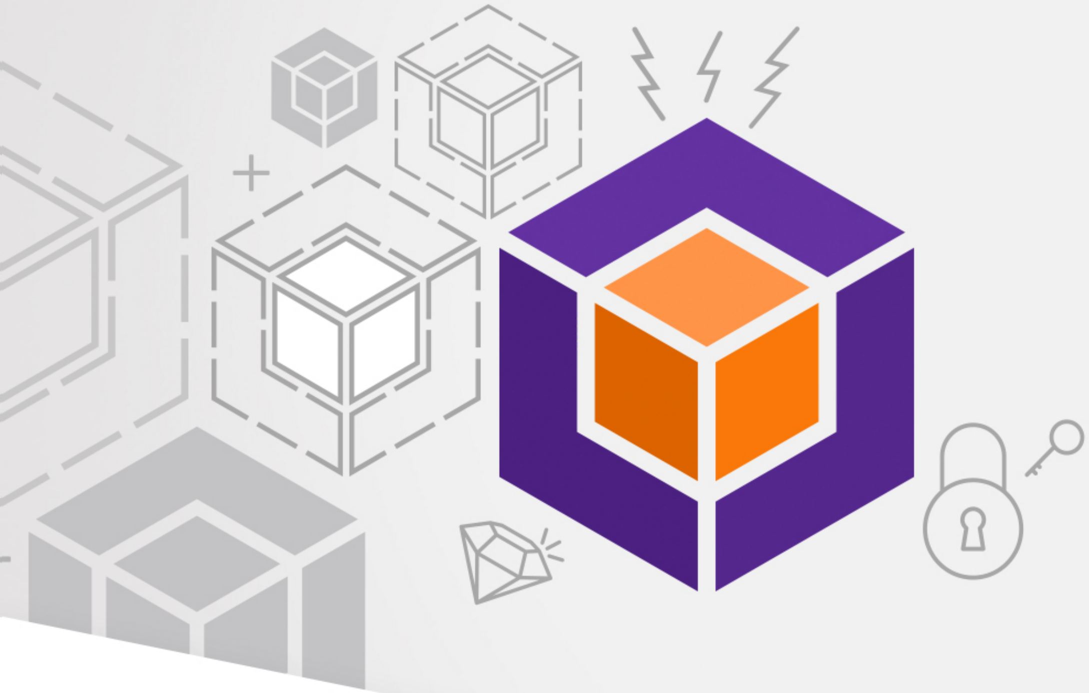


Tuning YottaDB Performance

Agenda

- Performance – Philosophy
- Application
- Database
- Platform
- Performance – Tuning
- Questions & Discussion

Performance – Philosophy

What is Performance?

- Maximizing logical database operations/second
  - Without compromising integrity of data or persistence required
- Repeatable workload & repeatable computing platform
  - Stable conversion between database operations and application metrics (throughput, response time, etc.)
  - Statistical repeatability is essential, even if actual repeatability is hard

Balance

- Ultimately, something always limits throughput
  - Hardware is never infinitely fast
  - Application logic always has critical sections
- Balanced system – making one component (CPU, memory, storage) faster or adding more of it has only a limited effect on throughput because some other component will limit throughput
  - Balance = cost effectiveness

What Limits Performance … 1

- Application design and coding
  - Can usually be detected outside the application
  - Can sometimes be ameliorated outside the application … but only sometimes
  - Application issues can make the fastest database and the fastest computer look slow

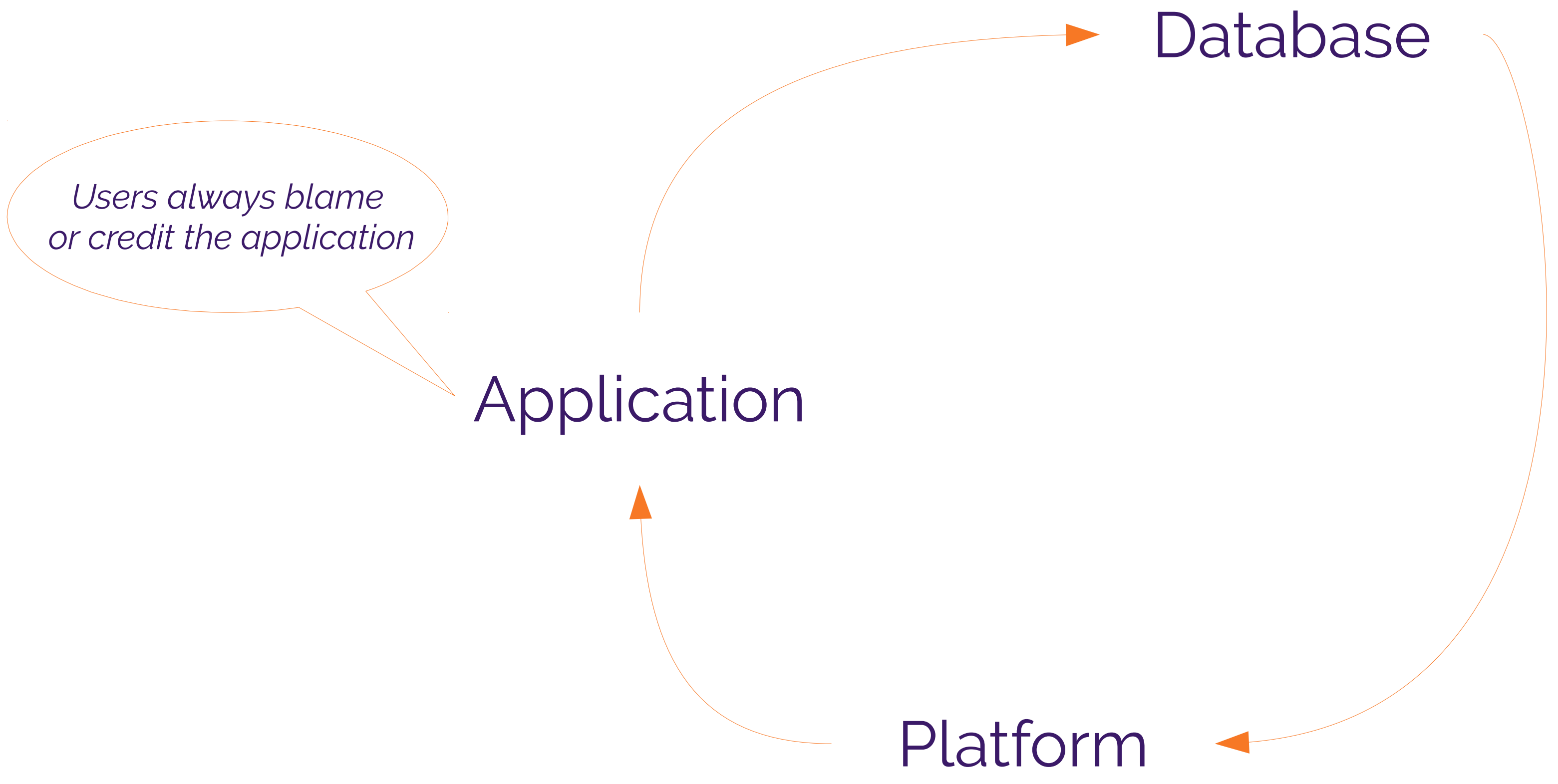
What Limits Performance … 2

- Application design and coding
- Database configuration
  - YottaDB has complete set of tools

What Limits Performance … 3

- Application design and coding
- Database configuration
- Computing Platform
  - Issues are often, but not always, visible or obvious
  - Requires expertise beyond the application and database

Performance is a journey, not a destination



Performance Tuning is like Cooking

- More art than science
- What is optimal for one application or even one configuration or one workload of an application may not be optimal for another
- But there are underlying principles and some methodology to the process

Application

Application Design & Coding Issues

- Single-threaded calculations
- Inefficient algorithms
- Repeated calculations

Tools to Identify Application Issues … 1

- YottaDB
  - M code profiling
  - $view("gblstat",*region*)
    - TP restarts
    - Lock fails
  - MUPIP INTRPT & MUPIP Journal Extract
  - %YGBLSTAT

Tools to Identify Application Issues … 2

- YottaDB
- External to YottaDB (typically available using package manager for your Linux distribtion), e.g.
  - gdb – https://www.gnu.org/software/gdb/
  - Oprofile – http://oprofile.sourceforge.net/news/

Database

Database Issues

- Contention
  - Pathological
  - Consequential – resulting from other factors
- Excessive IO
- Memory usage

Tools to Identify Database Issues … 1

- YottaDB
  - $view("gblstat",*region*)
    - TP restarts
    - Lock fails
    - Database global buffer effectiveness
    - Critical section acquisition
  - Database file header

Tools to Identify Database Issues … 2

- YottaDB
- External to YottaDB
  - vmstat, iostat, sar…

Platform

Platform Issues

- Missed opportunities
  - Hardware & operating systems
  - Filesystems & storage
  - Memory usage
  - OS tuning

Tools to Identify Platform Issues

- External to YottaDB
  - *With a few exceptions, outside our expertise*

Performance – Tuning

Access Methods … 1

- BG
  - Traditional
  - Required for encrypted databases and backward recovery

Access Methods … 2

- BG
- MM
  - Potentially faster

Access Methods … 3

- BG
- MM
- Choosing
  - MM (on `/dev/shm`) for temporary / scratch globals
  - BG for encrypted globals
  - Operational: MM if forward recovery is acceptable, BG otherwise

Database Fileheader Statistics

- $view("gblstat",*region*)
  - M function accessible with standard M code
    - e.g., gvstat (a personal tool, not yet supported software)
- Also accessible with DSE

Metrics … 1a

- Identify contention with TP restarts
  - Pathological
    - $TC0 \cong TC1 \cong TC2$
    - Address with application design / changes
    - Potentially ameliorate with database configuration

Metrics … 1b

- Identify contention with TP restarts
  - Pathological
  - Consequential
    - Address with both application design / changes as well as database configuration changes

Metrics … 1c

- Identify contention with TP restarts
  - Pathological
  - Consequential
  - Random
    - Address with database configuration changes

Metrics … 2

- Identify contention with TP restarts
- Global buffer effectiveness
  - No way to measure perfectly; proxies are
    - Database blocks per global buffer
    - Database operations per filesystem read
  - Balance empirically
  - Improve with database configuration

Metrics … 3

- Identify contention with TP restarts
- Global buffer effectiveness
- Lock acquisition efficiency
  - No way to measure perfectly; proxy is failures per successful acquisition
  - Address with both application and database configuration changes

Metrics … 4

- Identify contention with TP restarts
- Global buffer effectiveness
- Lock acquisition efficiency
- Critical section acquisition efficiency
  - No way to measure perfectly; proxies are acquisition statistics
  - Address with database configuration

Partitioning

- Fewer regions – easier to configure & manage, more efficient TP commit
- More regions – easier to reorg, opportunity to design application for fewer collisions
- Try to keep an entire global variable in one region unless there is a benefit to mapping at subscript level
- Assign globals to regions for operational reasons

Block Size

- Smaller – more efficient CPU usage, less random TP collision
- Bigger – potentially more efficient IO
- Choosing
  - Default choice is file system block size (4KiB)
  - Smaller to reduce random TP collisions
  - Bigger to ensure most global nodes fit in one block

Global Buffers vs. Filesystem Cache

- Database IO from global buffers is more efficient
- Global buffers – specific to each region
- Filesystem cache – common to all regions
- Strategy
  - Ensure adequate global buffers for working set and to minimize TP restarts
  - Balance empirically

Journal Buffers

- Always 512 bytes, not database block size
- Ensure enough for journal records of one transaction including before image records
- Probably not much value in more journal buffers than minimum – but probably not much performance lost from too many
- Size generously, but don’t go overboard

Other Journaling

- Journal sync_io
  - Probably a good idea, but benchmark before using
- Epoch Taper
  - Probably a good idea, but benchmark before using

Other … 1

- Shared memory for routines
  - No reason not to on current releases
- Hugepages
  - No reason not to for shared memory
  - Transparent hugepages – balance benefit vs. impact

Other … 2

- Swap space – avoid configuring unless required
- Storage
  - PCIeNVMe preferable to SATA
  - Directly plugged in storage preferable to SAN
- Filesystems – ext4 vs. xfs vs. f2fs (where supported)
- Compare Linux distributions, especially Ubuntu vs. Red Hat Enterprise Linux

Questions & Discussion

YottaDB

Thank You!

yottadb.com