Benchmarking

HarperDB v. InfluxDB

May 2020

HarperDB Inc. | 1001 17th St. Suite 680. Denver, CO 80202 | hello@harperdb.io

Overview

We built HarperDB out of frustration with the existing landscape of database products. As a result, with the release of HarperDB 2.0, we wanted to see how we stacked up against the most popular time series database in the world, InfluxDB. Turns out, we're faster across the board, but for concurrent reads it's just silly.

We asked an external technology team, Mycos Technologies, to create a simple benchmark for HarperDB compared against a modern, analytical, highly-available database, InfluxDB. We believe HarperDB is easier to use and provides more flexibility than InfluxDB. In this benchmark, we intend to prove that HarperDB holds up performance-wise as well.

Highlights

HarperDB is up to **6.8 times faster** than InfluxDB on writes.

InfluxDB struggled significantly with concurrent read benchmarking.

HarperDB **uses half the system resources** of InfluxDB at scale.

HarperDB **scales out-of-the-box** for concurrent operations, InfluxDB crashed.

HarperDB Inc. | 1001 17th St. Suite 680. Denver, CO 80202 | hello@harperdb.io

Methodology

This benchmark evaluates data writes, data reads, and a combination of reads and writes on HarperDB and InfluxDB. Tests were executed on both databases using their default configurations. Benchmarks were executed based on recommended benchmarking best practices for both databases. The benchmark tests were executed using Apache JMeter for 10 minutes, with a 10-second ramp-up, and with a simulated user count of 1, 10, and 50. Database requests were executed via HTTP API endpoints for both HarperDB and InfluxDB.

The following tests were executed on both databases:

- / Write data with 1 user
- / Read data with 1 user
- / Read/write data with 1 user
- / Write data with 10 users
- / Read data with 10 users
- / Read/write data with 10 users
- / Write data with 50 users
- / Read data with 50 users
- / Read/write data with 50 users

Each test was run two times and using the following order of operations:

1. Create schema/table for HarperDB or database/measurement for InfluxDB
2. Loop test action for 10 minutes. Test actions:
    a. Write a single, randomly generated record to the database
    b. Read a single, randomly selected record from the database
    c. Write a single, randomly generated record to the database, then read a single randomly generated record form the database
3. Clean environment

HarperDB Inc. | 1001 17th St. Suite 680. Denver, CO 80202 | hello@harperdb.io

Data Format

In all tests, object data was generated with the following 7 attributes:

```
{
  "id": "${__counter()}",
  "timestamp": "${__time()}",
  "photosensor": "${__Random(1,999)}.${__Random(1,100)}",
  "sensor_uuid": "probe-${__UUID}",
  "radiation_level": "${__Random(1,999)}",
  "ambient_temperature": "${__Random(1,100)}.${__Random(1,99)}",
  "humidity": "${__Random(1,999)}.${__Random(1,999)}"
}
```

Figure 1: Click to Enlarge

Read tests requested a single record from a query request filtering on a random id value

```
where id = _Random(0,<number of written records>)
```

The read and write test sequentially writes data and then reads data per looped iteration.

Test Results

This benchmark shows that HarperDB is faster than InfluxDB in every benchmark test. The only case where the two databases performed comparably was large scale concurrent data writes, which is what time series databases are designed for. In the concurrent data read cases InfluxDB is crippled. Moving from 1 to 10 concurrent users querying InfluxDB resulted in a 5x reduction in samples returned within the test period, consequently moving from 10 to 50 users crashed InfluxDB on the server. Write and read data tests on InfluxDB for 10 and 50 users resulted in error response rates of 3.4% and 8.0% respectively.

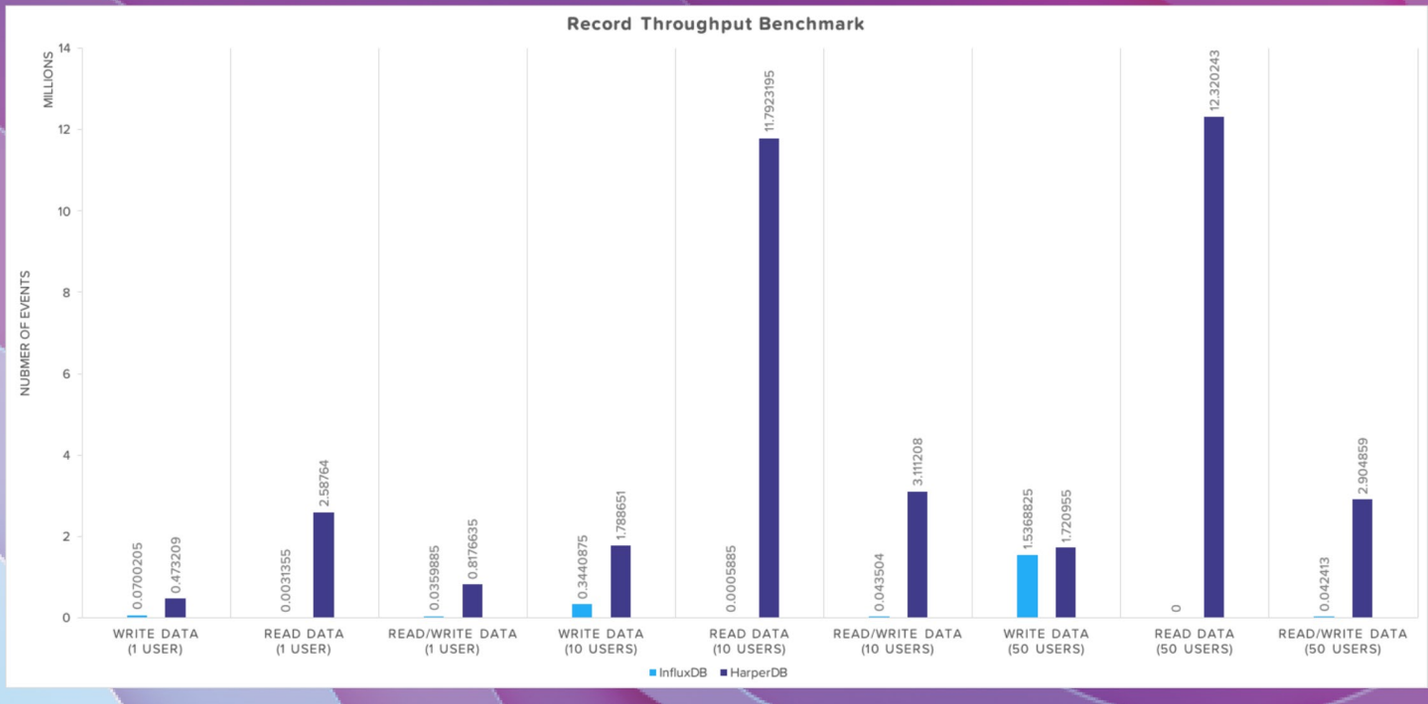

Figure 2: Record Throughput Results per Database, Click to Enlarge

HarperDB Inc. | 1001 17th St. Suite 680. Denver, CO 80202 | hello@harperdb.io

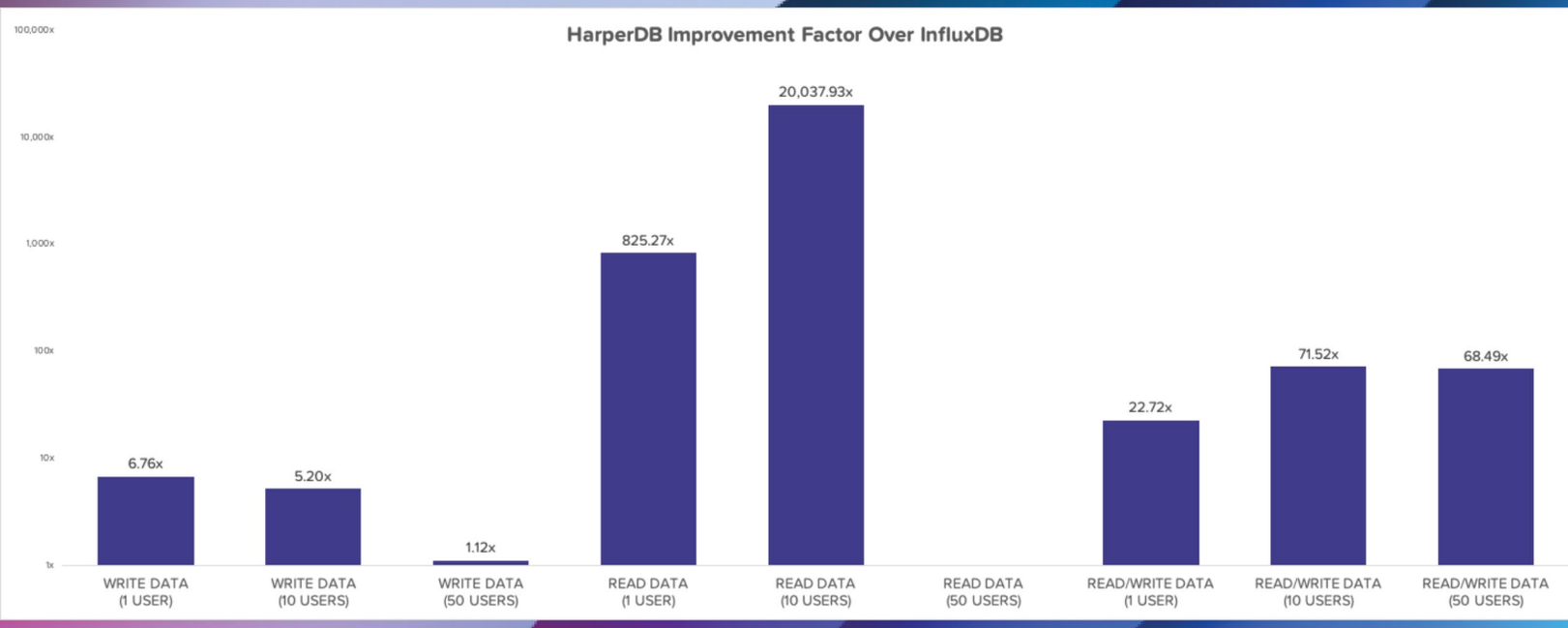

Figure 3: HarperDB Performance Multiplication Factor Compared to InfluxDB, Click to Enlarge

Conclusion

Time series databases are designed for high throughput ingestion, InfluxDB proved to be effective at that with relative performance improving as concurrent writers increased. Where they fall flat is concurrent data reads, that is evident in the test results with read performance.

Our goal in developing HarperDB was to build a product that would empower developers through ease of use while remaining highly scalable across all use cases, while not settling just read or just write performance. The benchmark results have demonstrably shown that HarperDB is effective for both reads and writes. HarperDB is built to be simple to install and use, without requiring costly product experts for tuning. We feel that this test conclusively demonstrates that at scale with numerous clients, HarperDB is more performant and can be scaled out cost-effectively.

Questions, comments, or feedback? Email us at benchmarks@harperdb.io

Curious if HarperDB is more performant than other tools you're using? Use code HARPERRUNSFAST when signing up for HarperDB Cloud and receive $300 in credits towards our paid tiers

Evaluation Machine Specs

CPU	CPU Intel® Core™ i7-8700 6 Core 3.20GHz
Memory	32 GiB DDR3
Disks	2x 250 GB SSD formatted with ext4