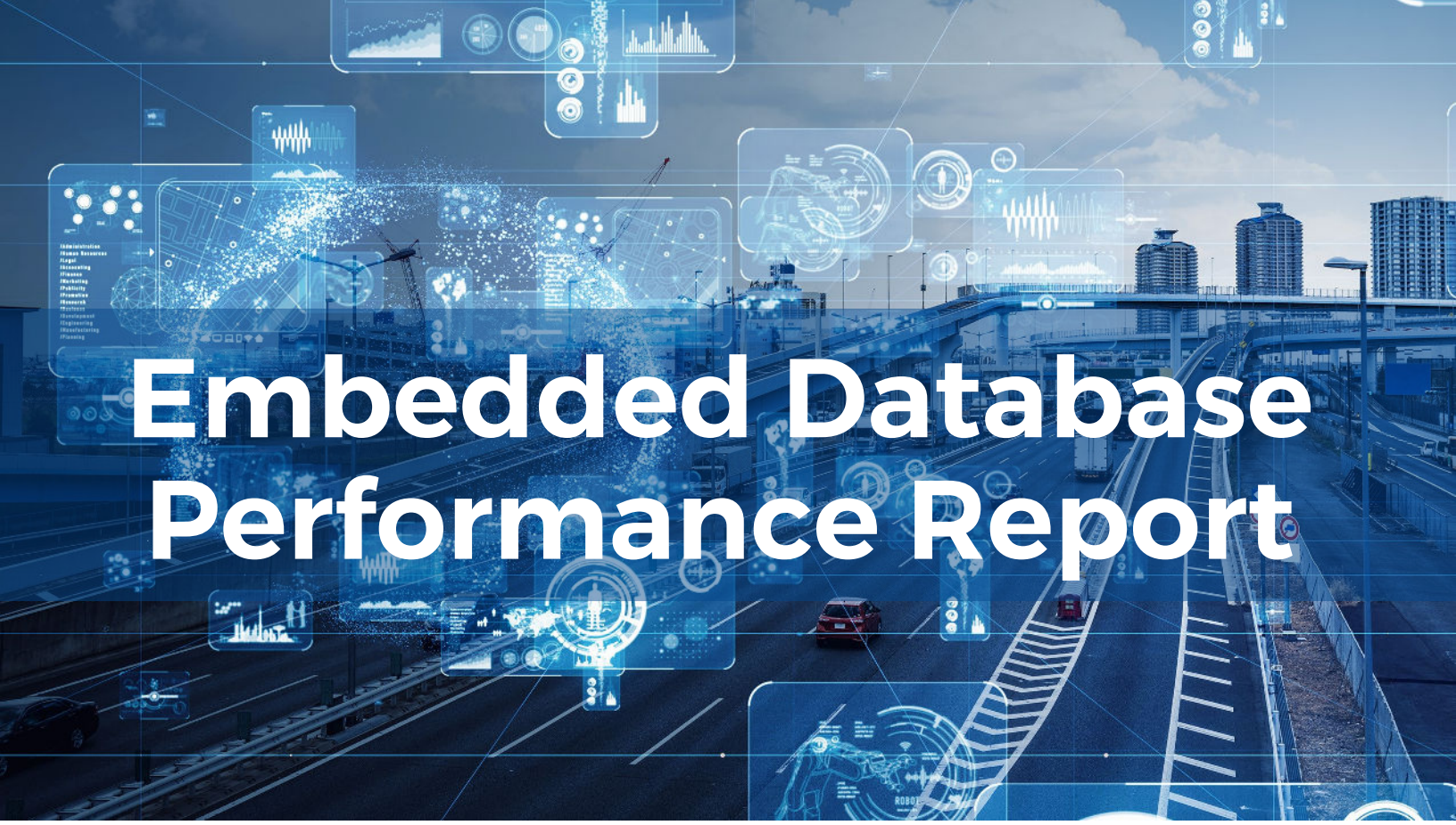


Embedded Database Performance Report

Actian Zen more than 25x faster than InfluxDB for time-series data

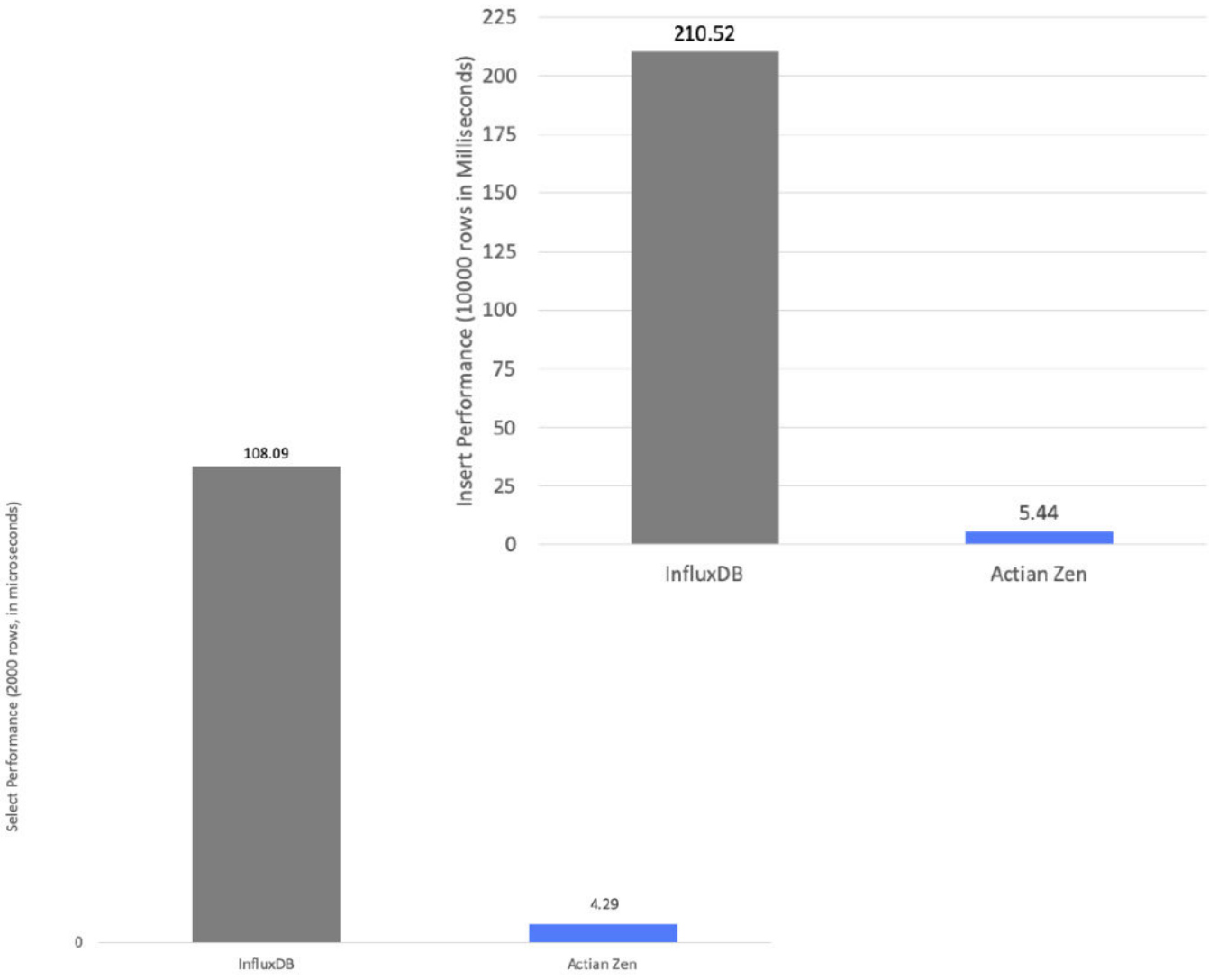
MCG Global Services Benchmark Results

May 2019

ACTIAN™

Key insights

- This benchmark did a head-to-head comparison of Actian Zen and InfluxDB for IoT time series data, both installed on a Raspberry Pi running Linux ARM/Raspbian using their native APIs (NoSQL).
- Actian Zen outperformed InfluxDB for data management by nearly **40x on inserts** of 10,000 data points and **25x on queries** of 2,000 data points



Checkout the Actian Zen performance advantage today!

Visit https://www.actian.com/zen-embedded-database

ACTIAN™

Embedded Time Series Database Performance Field Test

Product Profile and Evaluation:
Actian Zen and InfluxDB

McKnight Consulting Group
April 2019

Sponsored by

Executive Overview

Embedded databases are built into software, transparent to the application's end user and require little or no ongoing maintenance. Embedded databases are growing in ubiquity with the rise of mobile applications and internet of things (IoT) giving innumerable devices robust capabilities via their own local database management system (DBMS). Developers can create sophisticated applications right on the remote device. For these uses, the embedded architecture is preferred over client-server approaches which rely on database servers accessed by client applications via interfaces. Today, to fully harness data to gain a competitive advantage, embedded databases need a high level of performance to provide real-time processing at scale.

Web, mobile, and IoT applications have generated a new set of technology requirements. Embedded database architecture needs to be far more agile than ever before and requires an approach to real-time data management that can accommodate unprecedented levels of scale, speed, and data flexibility.

Within the world of IoT, the time series application is of particular importance. Most of the data generated today by the billions of IoT devices springing up around the globe are time-oriented. The devices take the same measurements repeated again and again over time. This creates a stream of data to ingest and analyze and often it's how the data fluctuates or changes over time that offers the most insight or value—rather than individual data points themselves.

It's not just the sheer number of devices generating time-series data streams, increasingly with high-resolution cameras, survey measurement instruments, and other high-bandwidth devices, the data streams from any one device can product terabytes of data per day. Furthermore, one level upstream from these devices, there is often a requirement to normalize, combine, and analyze multiple types of time series data sets against each other to avoid flooding cloud and backend data center storage systems with data that may or may not undergo later retrieval and post-processing let alone generate actionable insights.

Some of the more traditional home-grown methods of handling these time series data streams such as allocation of temporary memory or file storage or even proprietary, industry-specific vendor solutions are incapable of handling the data rates and variety of data types often found in these new edge environments. Many Operational Technology practitioners working with IoT are beginning to turn their attention to the commercial off-the-shelf marketplace, looking at various types of relational databases.

However, not all relational databases are unable to meet these new requirements, and developers are therefore turning to NoSQL database technology. NoSQL use cases abound where the need for flexible schema or schema-less data would trip up conventional, relational databases.

To quantify embedded time series database performance, we conducted this field test study, which focuses on the performance of mobile application-ready, NoSQL, embedded database solutions Actian Zen and InfluxDB. The intent of the field test's design was to represent a set of basic database transactions that an organization developing edge applications might encounter.

The test methodology was based on and largely followed the Performance comparison of (relational) embedded databases for .NET by Christophe Diericx. However, our field test harness was developed and adapted to a time series use case; (e.g., we eliminated the update and delete tests.) We conducted the field test on Actian Zen and InfluxDB installed on the same Raspberry Pi device. In our experience, performance is a very important aspect of an embedded database selection, but it is only one aspect and many factors should be considered.

Overall, the field test results were insightful in revealing the query execution performance of Actian Zen and InfluxDB revealing some of the differentiators in the two products.

Actian Zen was faster across the board including the area where it tends to really matter in embedded databases—write speed. This is the essential performance metric for IoT data. Actian Zen Edge outperformed InfluxDB by nearly 40x on inserts of 10,000 data points and 25x on queries of 2,000 data points.

Actian Zen is a mature platform for embedded database applications with over 30 years of engineering and development behind it. Features that contributed to its extremely fast performance include, but are not limited to, the Btrieve API and Turbo Write Accelerator.

Embedded IoT Time Series Database Selection

Organizations that utilize application-laden smart devices rely on embedded database platforms to process edge data at high speed and bring it in with consistency to harmonize an ecosystem of activity. Volumes for data that can be utilized at the edge is rapidly expanding—placing significant performance demands on embedded architectures. Thus, a key differentiator is the depth by which a database maintains performance to scale with simple queries representative of real-world use cases of embedded databases—SQL and NoSQL alike.

While performance is very important, it is not the only consideration. Developers choosing NoSQL must consider data access, scalability, and availability.

While the subject of this field test is IoT time series application within an embedded device, Actian Zen is a multi-purpose, embedded database. Actian Zen is part of the overall Zen family of Zen Core, Zen Edge, Zen Enterprise, Zen Reporting Engine and Zen Cloud. When combined, this suite of products enables not only embedded applications, but client-server (with zero ETL) and cloud deployments as well. For this test, we used the Beta release of Zen v14 for Linux ARM/Raspian, which includes new timeseries functionality—including automatic timestamps.

Users of conventional relational databases will find Actian Zen to be familiar. It has complete CRUD (create, read, update, and delete) functionality, as well as other features you might expect—such as indexes and ANSI-compliant SQL. Additionally, Actian Zen exclusively offers the high performance Btrieve 2 API (which is tested in this field test.) The Btrieve 2 API supports NoSQL and native development support for Java and C/C++ based devices and SWIG for Python, Perl, and PHP—in addition to its SQL support. InfluxDB is exclusively NoSQL, and only offers software development kits for mobile devices, such as iOS, Android, and .NET.

In a client-server configuration, Actian Zen comes with the capability to automatically synchronize in real time between Zen Core or Edge on a remote device and Zen Enterprise on a server—without ETL. This capability is critical for today's needs and uses, because the potential number of IoT devices could easily number in the thousands, and all that information may need to funnel into a core database on a server – but on a gateway device within the remote operational technology environment rather than within the confines of a data center. Actian has real-time synchronization capability of Actian Zen Edge or Core to Zen Edge or Enterprise via the Btrieve API without additional architecture, which can allow you to achieve scale with simplicity.

InfluxDB is for time series applications. InfluxDB is a component to the InfluxData Platform, which is built upon a set of open source projects — Telegraf, InfluxDB, Chronograf, and Kapacitor, which are collectively called the TICK Stack. Kapacitor is used to process both stream and batch data from InfluxDB, while Telegraf and Chronograf are reporting and admin user interfaces.

In an InfluxDB database, “rows” are referred to as points, and points are contained by a measurement—which one might liken to a table in a conventional DBMS. Points and measurements are organized by timestamps—stored as 64-bit integers. Points can also be tagged. Tag “fields” are indexed by the database for faster selecting, sorting, and grouping, but value “fields” are not. In terms of basic CRUD operations, InfluxDB does not update data points. Instead new data for an existing timestamp is inserted as a new point that supersedes the previous point. InfluxDB cannot delete individual points of data manually. Instead, it can be configured to expire data after a defined length of time, and then automatically deleting any expired data from the database. InfluxDB also offers a SQL-like query language for interacting with data. InfluxDB has several official client libraries for languages like Python, Java, and C#. However, InfluxDB is not ported for direct deployment on mobile devices running Android or iOS.

Both InfluxDB and Actian Zen were designed to “set it and forget it,” with little-to-no ongoing database administration. Actian Zen has features like auto-reconnect networking, automated defragmentation, multi-user support, and concurrent write capabilities.

Actian Zen is natively NoSQL and is flexible enough to be a time-series database as well as a document-based or key-value store. Actian Zen was initially designed as Btrieve (and later PSQL) and has been in production with many multi-national organizations with over 30 years of engineering and enhancement.

This report focuses on the performance of two embedded IoT time series database options. It is important to get into the right embedded database early in the development cycle when the stakes are less critical.

Field Test Setup

The field test was executed using the following setup, environment, standards, and configurations.

Data Preparation

An aim of the field test is to simulate a typical real-world scenario and use case for embedded time series databases. In our field test, we chose a simple "schema" for an application that stores timestamps and values in the embedded database. The model is represented by the following JSON:

```
{
   "data_points":[
      {
         "timestamp":1554229400630093,
         "value":33
      }, ...
   ]
}
```

The data used in the field test was generated randomly in real time by a Python application during the field test execution.

Configuration

Our field test included two different embedded time series databases—Actian Zen and InfluxDB—installed on the same Raspberry Pi 3 device. InfluxDB is the latest generally available release of the product. Actian Zen is the Beta release of version 14.

Time Series Databases

Embedded RDBMS	**Actian Zen**	**InfluxDB**
Version	14.00.999 Beta	1.7.6

Raspberry Pi

Hardware	**Raspberry Pi 3+**
Processor	1x Broadcom BCM2837B0 SoC 1.4 GHz 64-bit quad-core ARM Cortex-A53 (512 KB shared L2 cache)
RAM	1 GB
OS	Raspbian GNU/Linux 9.4 (Stretch)

The Raspberry Pi is shown below.

Test Use Cases

The test involves simple uses cases of the most basic database CRUD operations: selecting, updating, and deleting rows based on indexed and non-indexed columns. However, update and delete operations are atypical and rare events in time series database use cases. Those tests were not timed. Also, the time series databases we used are already indexed by timestamp, so we omitted specific "non-index" tests. This left the insert and select tests. Therefore, we opted for tests that would demonstrate raw performance that could be found in most embedded timeseries database implementations.

Both platforms support a robust set of NoSQL capabilities. For both Actian Zen and InfluxDB we used their native APIs (as opposed to a third-party API) to execute the database transactions in order to test its functionality and performance, rather than SQL (Actian Zen) or InfluxQL.

Use Case 1: Insert Performance

Time series databases and their applications will undoubtedly need excellent insert performance. This may be the single most important metric for many use cases. For example, consider an IoT device is a sensor taking rapid-fire readings at sub-millisecond intervals. In the case of real-time or rapid sensor readings, insert performance is critical.

Test 1	**Insert 10,000 data points**

NOTE: At the beginning of the test, the database was empty. The Insert test provided the test data for the select test.

Use Case 2: Select Performance

Certainly, we must consider both platforms' ability to retrieve data. Our test cases involve selecting bulk data points, rather than single data points via their timestamp.

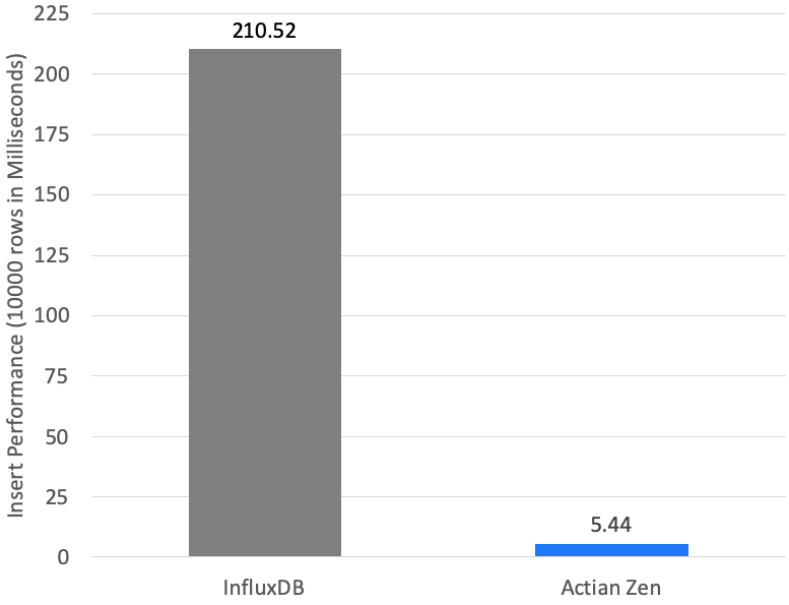
Test 2	**Select 2,000 data points**

Field Test Results

The following figures display the average time elapsed for each database transaction for both Actian Zen and InfluxDB. Each test was executed 5 times and the median value was used.

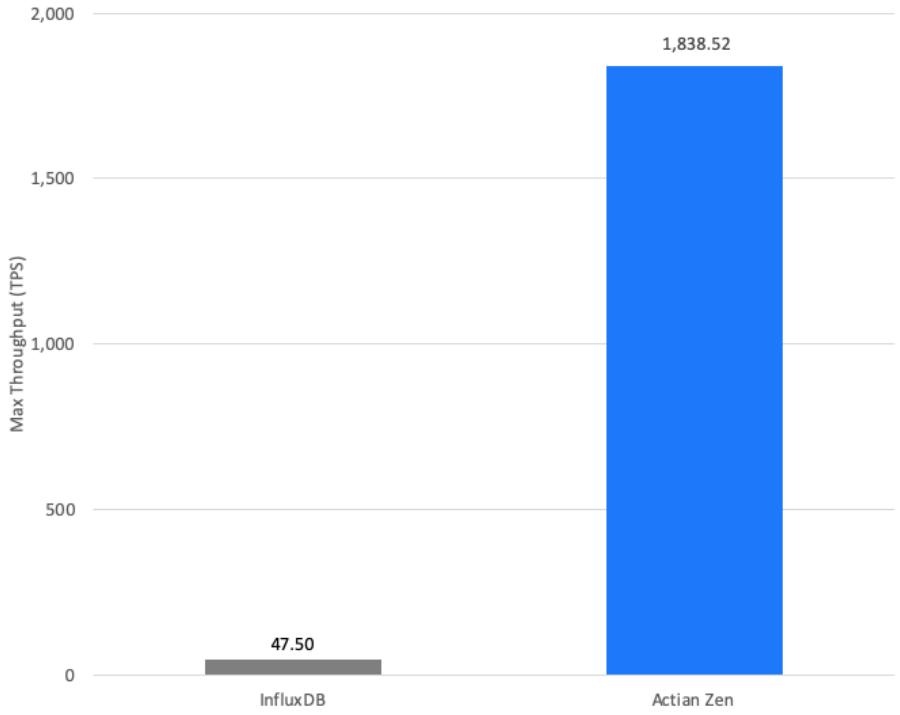
Test 1: Insert 10,000 data points in rapid succession

Below are the average times (in microseconds) it took to insert a timestamp and randomly-generated value pair into Actian Zen and InfluxDB.

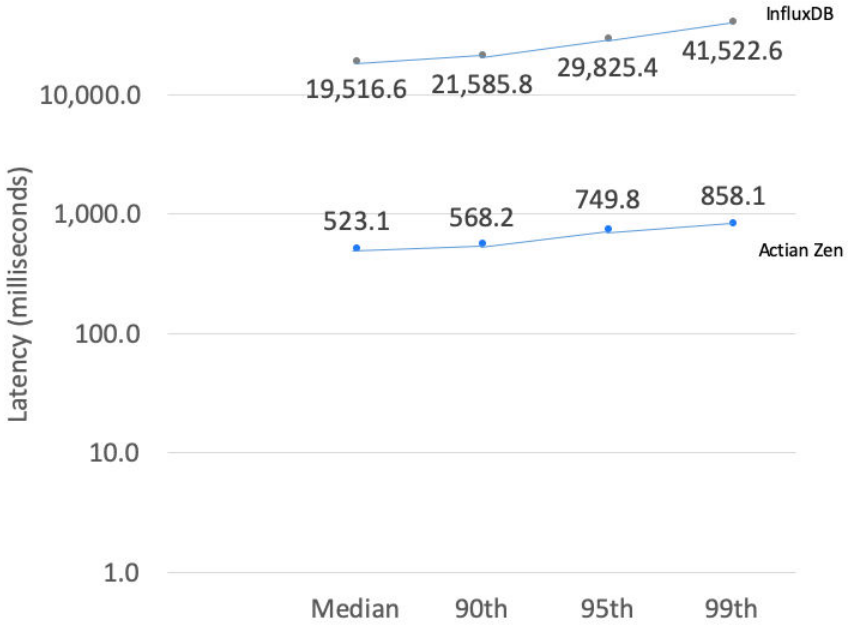


This test revealed the first major performance differentiator. Actian Zen's average time to insert a single data point (taking the average of all 10,000 inserts) was almost 40 times faster than InfluxDB inserts.

The next chart is the maximum throughput of insert transactions in terms of transaction per second that were achieved during our testing.

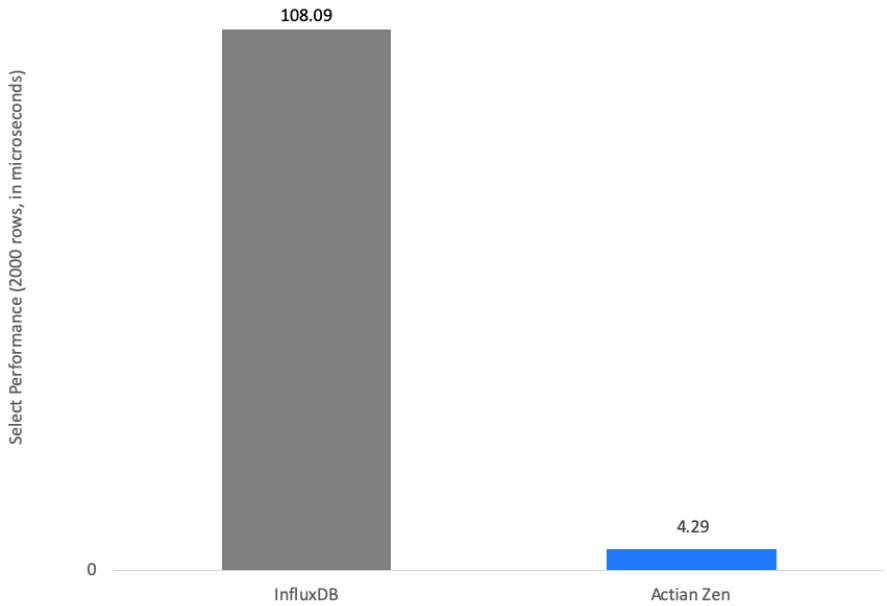


Below are the latencies of individual inserts quantified at the median, 90th, 95th, and 99th percentiles. Note the logarithmic scale of the y-axis.



Test 2: Select 2,000 data points

Below are the average times per document (in microseconds) it took to bulk select the first 2,000 data points from the databases on Actian Zen and InfluxDB.



Both platforms responded very quickly. InfluxDB's fetch rate per document (taking the average of all 2,000 documents) was 25 times slower than that of Actian Zen's.

Conclusion

Actian Zen outperformed InfluxDB in both of the fundamental database operations most critical to time-series use cases. These tested operations underlie nearly all operations that occur on an embedded time series database for an IoT or similar implementation, so it is unlikely more complex operations would have a different result.

Actian Zen is a mature platform for embedded database applications with over 30 years of engineering and development behind it. The Btrieve 2 API had clear performance advantages. Also, Zen's Turbo Write Accelerator could also shed light into its performance advantages. Since it costs much less to continue writing than to stop and restart, contiguous writes are significantly faster than non-contiguous writes. The Turbo Write Accelerator (TWA) pre-allocates open slots within the physical file so that multiple pages can be written as a single coalesced page—improving I/O performance and reducing the overhead of interaction with the operating system.

The result of the application of the methodology to the architecture shows a performance advantage to Actian Zen. This is especially true in the important write operations insert, update and delete.

Overall, Actian Zen is an excellent choice for IoT or mobile companies needing high performance and a scalable embedded database for time series applications.

About Actian

Actian, the hybrid data management, analytics and integration company, delivers data as a competitive advantage to thousands of customers worldwide. Through the deployment of innovative hybrid data technologies and solutions Actian ensures that business critical systems can transact and integrate at their very best – on premise, in the cloud or both. For more information about Actian Zen and the entire Actian portfolio of embedded data management, visit www.actian.com.

About McKnight Consulting Group

McKnight Consulting Group (MCG) (http://www.mcknightcg.com) services span strategy, implementation, and training for turning information into the asset it needs to be for your organization. MCG strategizes, designs and deploys in the disciplines of Master Data Management, Big Data, Data Warehousing, Analytic Databases and Business Intelligence.

In Closing

Performance is important but is only one criterion for an embedded database selection. This is only one point-in-time check into specific performance. There are numerous other factors to consider in selection across factors of Administration, Integration, Workload Management, User Interface, Scalability, Vendor, Reliability, and numerous other criteria. It is also our experience that performance changes over time and is competitively different for different workloads. Also a performance leader can hit up against the point of diminishing returns and viable contenders can quickly close the gap.

MCG runs all of its performance tests to strict ethical standards. The results of the report are the objective results of the application of queries to the simulations described in the report. The report clearly defines the selected criteria and process used to establish the field test. The report also clearly states the data set sizes, the platforms, the queries, etc. used. The reader is left to determine for themselves how to qualify the information for their individual needs. The report does not make any claim regarding third-party certification and presents the objective results received from the application of the process to the criteria as described in the report. The report strictly measures performance and does not purport to evaluate other factors that potential customers may find relevant when making a purchase decision.

This is a sponsored report. Actian chose the competitors, the test and the Actian configuration. MCG chose the most compatible configurations for the other tested platforms and ran the queries. Choosing compatible configurations is subject to judgment. We have attempted to describe our decisions in this paper.

In this writeup, all the information necessary is included to replicate this test. You are encouraged to compile your own representative queries, data sets, data sizes and compatible configurations and test for yourself.