WHITE PAPER

Why RedisTimeSeries is the Right Choice for Your Time Series Data

Roshan Kumar, Redis

CONTENTS

Time series data is broadly defined as a series of data stored in time order. Examples of time series data can range from stock prices over a period of many years to CPU performance metrics from the past few hours. Redis is popularly used in solutions that store and retrieve time series data. As described in our previous white paper, the combination of Sorted Sets and Hashes can help you sort and aggregate time series data. In this white paper, we will explore how RedisTimeSeries fills the gaps that Sorted Sets and Hashes leave behind.

Characteristics of time series data

Time series data is widely used across many industry verticals. It has carved out its own category of databases, because relational, document-oriented and streaming databases do not fulfill the needs of this particular type of data. Due to its distinct characteristics (listed below), time series data is typically inefficient with other databases:

1. **High-speed data ingest:** Whether it is an IoT use case or market analysis data, you have a steady stream of data that arrives at high speeds and often in bursts. For most solutions, the data arrives 24/7, 365 days a year.
2. **Immutable data:** Once inserted in the database, a data point does not undergo any changes until it is expired or deleted. The data is typically log data with a timestamp and a few data points.
3. **Unstructured labels:** Time series data is generally produced continuously over a long period of time by many sources. For example, in an IoT use case, every sensor is a source of time series data. In such situations, each data point in the series stores the source information and other sensor measurements as labels. Data labels from every source may not conform to the same structure or order.
4. **Diminishing value over time:** Only an aggregated summary of data with an appropriate time range would be relevant in the future. For example, in a year from now, most users will not require every data point stored at the range of milliseconds. Only aggregated and generalized data by a minute, hour or day would make sense in that case.
5. **Queries are aggregated by time intervals:** Charts based on time series data enable you to zoom in and out. They do so by aggregating their data by time intervals. Typically, time series data queries are aggregations. This is in contrast to retrieving individual records from the database.

Problems with using traditional databases for time series use cases

When we surveyed our customers about their time series data use cases, eight out of 10 customers responded saying they stored time series data in a relational database. This has many drawbacks, because relational databases:

- Are designed and optimized for transactional use cases.
- Carry the overhead of locking and synchronization that are not required for the immutable time series data. This results in slower-than-required performance for both ingest and queries. Enterprises then end up investing in additional compute resources to scale out.
- Enforce a rigid structure for labels and cannot accommodate unstructured data.
- Require scheduled jobs for cleaning up old data.
- Are used for multiple use cases. Overuse of running time series queries may affect other workloads.

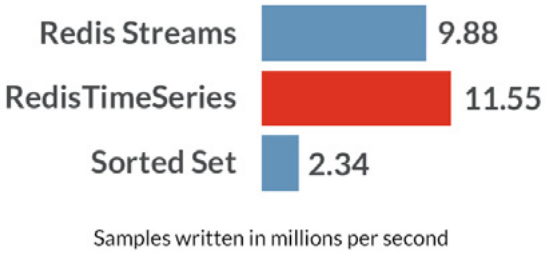
By introducing Redis in front of relational databases, our customers achieved scaling out and managed to deliver sub-millisecond latency for their queries. However, this did not solve all of their needs. They still found limitations to be addressed:

1. They aggregated the results at the application layer.
2. Data ingest did not scale.
3. The data did not expire.
4. They had to manage label indexing in their application layer, which was an overhead.

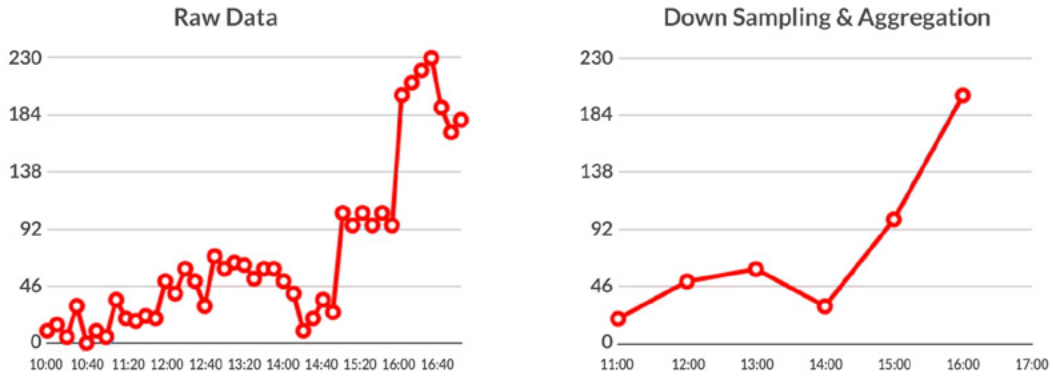
RedisTimeSeries advantages

RedisTimeSeries is purpose-built to collect, manage and deliver time series data at scale. It meets all of the performance and flexibility requirements of a time series database. It is written in C and runs inside Redis using the latter's Modules API, thereby sharing its memory and blazingly fast performance. Below, we have outlined some of RedisTimeSeries' critical advantages:

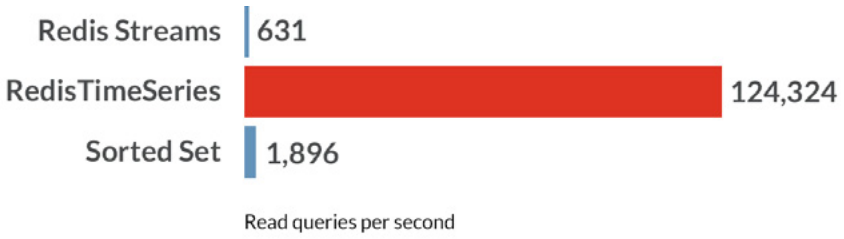
- **Fast data ingest:** Slow data ingestion is a problem not only with relational databases, but also with disk-based time series databases. The problem is worse when a search engine is used to store and index time series data. As an in-memory database, RedisTimeSeries can ingest over 500,000 records per second on a standard node. Our benchmarks show that on Redis Enterprise, with a cluster of 16 shards, you can ingest over 11.5 million records per second.



- **Resource efficiency:** With RedisTimeSeries, you could add rules to compact data by downsampling. For example, if you have collected more than one billion data points in a day, you could aggregate the data by every minute in order to downsample it, thereby reducing the dataset size to 24 * 60 = 1,440 data points. You could also set data retention policies and expire the data by time when you don't need them anymore.



- **Easy, fast queries:** RedisTimeSeries allows you to aggregate data by average, minimum, maximum, sum, count, range, first and last. You could run over 100,000 aggregation queries per second with sub-millisecond latency. You could also perform reverse lookups on the labels in a specific time range.



- **Ready integrations:** RedisTimeSeries has connectors that offer out-of-the-box integration with external services such as Prometheus, Grafana, Telegraf, Redis Streams and relational databases.
- **Client libraries in popular languages:** It's easy to integrate your apps with RedisTimeSeries. Redis has created custom libraries for RedisTimeSeries in Java and Go. You could run RedisTimeSeries commands using other popular Redis clients in Python, Node.js, C, .NET and so on.

Benefits of RedisTimeSeries

RedisTimeSeries combines all the benefits of Redis and a purpose-built time series database. It can help your business in many ways, including by saving on resources, supporting more end users and taking your apps faster to the market with easy integration. Here are a few other benefits you can expect by adopting RedisTimeSeries:

- **Instant user experience:** With RedisTimeSeries, you can deliver powerful, real-time user experience on time-based charts and graphical applications. RedisTimeSeries also eliminates the latency between data ingest and query, thereby serving real-time use cases such as fraud and anomaly detection. Graphics powered by RedisTimeSeries have minimal overhead on data processing because all of the aggregation and time-bucketing is performed in the backend.
- **Cost savings at scale:** Just like Redis, RedisTimeSeries is extremely lightweight. A small cluster in RedisTimeSeries, if not a single node, enables you to perform data ingest and run queries from thousands of sources, all simultaneously. It therefore supports many more users than you could with other database platforms.
- **Drive innovation by doing more with the data:** Using RedisTimeSeries will ensure that you are no longer limited by how many metrics you can present on your dashboard. With RedisTimeSeries, you will be able to run more queries at a faster speed, and your dashboard will subsequently get richer with more data points.

If you are seeking the ideal solution for your time series use cases, RedisTimeSeries is the right choice for you. It outperforms other databases with its advanced capabilities and integration with other applications, and it never compromises on high performance.

700 E El Camino Real, Suite 250
Mountain View, CA 94040
(415) 930-9666
redis.com