

MongoDB vs. Redis Comparison

[Get started free with MongoDB](#)

MongoDB and Redis are modern NoSQL databases. Even though they both fall under the same umbrella term—NoSQL—they have conceptually different storage models. MongoDB stores data on disk whereas Redis is an in-memory store.

In this article, we'll explore the differences between the two databases and their use cases.

MongoDB vs. Redis key differences

The following table should serve as a quick comparison guide between MongoDB and Redis.

MongoDB

Storage Model
On-disk storage by default
In-memory storage engine is available with MongoDB Enterprise Edition.
BSON (Binary JSON) documents

MongoDB

Data Model	<p>Document size is up to 16MB.</p> <p>Supports multiple data types: String, Boolean, Number (Integer, Float, Long, Decimal128...), Array, Object, Date, Raw Binary, GeoJSON.</p>
Query Language	<p>MongoDB Query API</p> <p>Query documents by single or multiple keys, ranges, or text search.</p> <p>Perform graph traversals, geospatial queries. Create materialized views on demand.</p> <p>Analyze data with advanced aggregation pipelines.</p>
Indexes	<p>Rich and easy to create indexes</p> <p>Secondary indexes enable developers to build applications that can query and analyze the data in multiple ways.</p> <p>You can create compound, TTL, text, geospatial, hashed, wildcard and compound wildcard indexes.</p> <p>MongoDB Atlas's Performance Advisor suggests new indexes to improve query performance. MongoDB Atlas also allows you to build indexes in a rolling fashion to minimize the impact on the working system.</p>
	<p>Built-in Sharding</p>

Scaling-Out

Sharding allows you to scale out across multiple nodes and geographic regions.

Language-agnostic feature supported by all official and community supported drivers.

Partition data by range, hash, or zone. Perform cross-shard operations.

Live resharding allows you to redistribute your data across shards without downtime. Introduced in MongoDB 7.0, shard key advisor commands will generate metrics that will help you refine your shard keys.

Multi-cloud support, consistent backups with point in time recovery by MongoDB Atlas.

High Availability

Replica sets

Create up to 50 copies of your data, provisioned across separate nodes, data centers, and geographic regions.

Automatic failover with replica set elections.

Transactional Data Integrity

Multi-document ACID transactions

Multi-statement with easy-to-use syntax similar to relational databases.

Redis

Storage Model	In-memory storage with on-disk persistence
Data Model	<p>Key-value store</p> <p>Keys are binary safe strings with length up to 512MB.</p> <p>Values: String and multiple data structures, such as List, Set, Bitmap, Hash.</p> <p>The stored dataset is limited by the size of the available memory.</p>
Query Language	<p>Key-value queries only</p> <p>Limited query functionality. By design—only primary key access.</p> <p>Query functionality can be extended with third-party Redis Modules.</p>
Indexes	<p>Hard to build secondary indexes</p> <p>Secondary indexes should be manually built and maintained.</p>
Scaling-Out	<p>Redis Cluster</p> <p>No multi-shard database operations.</p> <p>No consistent cross-shard backups and no strong consistency.</p> <p>Hash sharding only. Manual maintenance of the shards.</p> <p>Limited driver support with community maintained libraries.</p>

Redis

High Availability

Redis Sentinel

Monitor the state of your Redis Cluster with a set of independent processes.

Manual failover required if you need to promote a replica in another data center to master.

Transactional Data Integrity

MULTI command

Muti-record transaction support with the MULTI command.

No rollbacks support—you need to implement rollback in the application code.

What is MongoDB?

MongoDB is the leading **document database** chosen by architects and developers for a wide variety of **use cases**. The key characteristics of MongoDB, built into it from day one, are:

1. Distributed architecture that allows systems to scale out across multiple nodes.
2. Document data model that maps naturally to the objects in your code.
3. Focus on great developer experience with official support for over a **dozen programming languages**.

MongoDB stores data on the disk in **BSON** (Binary JSON) records called documents. The documents have a flexible schema, meaning that the structure of the stored data can change over time. You can add new fields or remove existing ones by modifying the document.

Similar or related documents are grouped in collections. Documents in a single collection can have a different set of fields. Flexible schema doesn't mean chaotic data, though—you can specify validation rules on your collections based on your application requirements.

MongoDB Atlas is MongoDB's multi-cloud database platform. It allows you to distribute and move your data across all major cloud providers—Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP).

What is Redis?

Redis, shortened from Remote Dictionary Server, is an in-memory key-value store. Most of its advantages and disadvantages stem from this very definition.

What is an in-memory store?

In-memory means that data is stored on the host's RAM (Random Access Memory) and not on disk. Reading and writing from the RAM is much faster than performing disk operations. This allows in-memory stores, such as Redis, to perform millions of requests per second.

The **in-memory storage** model comes with its disadvantages. RAM is a lot more expensive than disk storage and therefore not cost-effective for large data sets. Simply put, your data set is limited by the memory allocated for the Redis process.

What is a key-value store?

Key-value stores treat the data as a single collection. Each record in the collection is a pair—a key and a value associated with it. Keys must be unique as they are used to retrieve values.

The data types and allowed structures for keys and values are implementation-specific. For keys, Redis allows you to use binary safe strings up to 512 MB. For values, you can choose from a wider range of data structures. Among the supported types are strings, lists, sets, maps, and streams.

A disadvantage of using a key-value store, such as Redis, is that there's no query language. Additionally, Redis doesn't natively support secondary indexes. This limits your data access flexibility. If you need an access path to the data **different** from the primary key, you have to build and maintain your own indexes, storing them in the already limited memory.

Redis use cases

Based on its configuration, Redis can be put to different uses—a cache, a session management system, or a message broker are among the most common ones. Due to the in-memory storage model shortcomings, it's rare to use Redis as a system of record.

Redis is often used in combination with an on-disk database such as MongoDB. While the on-disk database is the primary storage solution, Redis may be used as a caching layer, or for performing real-time analytics.

If you're already using MongoDB, you can achieve similar results without adding Redis to your system. MongoDB's storage engine, [WiredTiger](#), has an [internal cache](#) that may be enough to accommodate your application's working set. In addition, [MongoDB Enterprise Advanced](#) offers an [in-memory storage engine](#).

MongoDB vs. Redis: data storage

The most obvious difference between MongoDB and Redis is their conceptually different data storage model. Redis is an in-memory data store, while MongoDB's default storage engine, WiredTiger, stores data on disk.

Does Redis persist data?

Despite being an in-memory store, Redis persists data on disk to ensure data durability. If the Redis process crashes and restarts, it can restore the data from the disk.

Redis offers different persistence options that can be adjusted to meet your application requirements. More frequent writes to the disk mean higher durability of the data, but also degraded performance.

Does MongoDB support in-memory storage?

MongoDB supports an [in-memory storage engine](#) as part of [MongoDB Enterprise](#). The in-memory storage engine combines the predictable latency benefits of the in-memory storage model with the rich query capabilities of MongoDB.

The in-memory storage engine doesn't write data to persistent storage. To ensure data durability, you can deploy replica sets that use a combination of the in-memory storage engine and the default persistent storage engine. In case of a crash and restart, the nodes using the in-memory storage engine can sync from the nodes using persistent storage.

Does MongoDB Atlas support in-memory storage?

The in-memory storage engine is not available for MongoDB Atlas. However, you can use [NVMe SSD drives](#) that offer more predictable latency than regular hard disk storage.

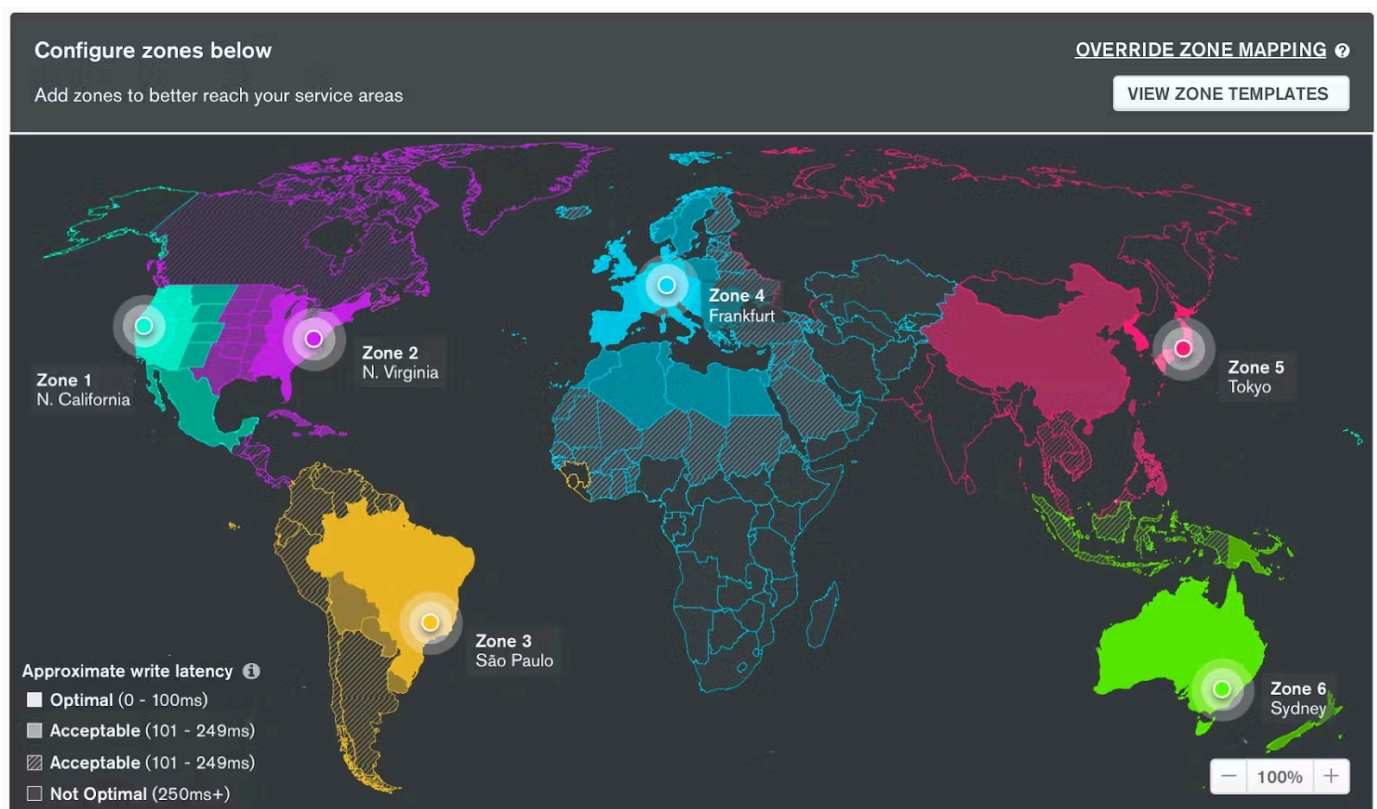
Another option is to increase the available RAM on your deployed cluster. WiredTiger, MongoDB's storage engine, will use part of the memory for cache. For more information about in-memory use in MongoDB Atlas, check out the dedicated section in the [in-memory database](#) article.

MongoDB vs. Redis: scalability

MongoDB has a horizontal scale-out architecture built-in from day one. Horizontal scaling is achieved with **sharding** which allows you to distribute data across multiple nodes. Data is partitioned across nodes based on the configured shard key. If you decide to refine your shard key, Live Resharding allows you to redistribute your data without downtime. Introduced in MongoDB 7.0, shard key advisor commands generate metrics that will help you refine your shard keys. Redis Cluster is Redis's solution for scaling out by sharding data across multiple nodes.

Sharding strategies

Redis Cluster supports only hashed sharding. MongoDB offers different shard key strategies –ranged sharding, hashed sharding, and zoned sharding. You can choose the strategy that best suits the needs of your system. With **Global Clusters** in MongoDB Atlas you can set up sharded cluster zones to support geographically distributed applications.



Consistency and backups

Redis Cluster doesn't guarantee strong consistency, meaning that the cluster may lose data in case of a failure. MongoDB offers one of the strongest data safety and consistency guarantees among databases **tested by Jepsen**.

Redis Cluster doesn't offer cross-shard backups. Each shard must be backed up individually. MongoDB Atlas allows you to perform **consistent backups** with point-in time recovery.

Language support

Redis Cluster is supported by a limited number of community-maintained libraries. Currently, there are actively maintained libraries for Node.js, Ruby, Python, PHP, Java, C#, and Go. Sharding in MongoDB is a language-agnostic feature which is supported by all official and community libraries, covering dozens of languages.

Summary

MongoDB is a general-purpose persistent database used as a primary storage solution by businesses in various **industries**. Redis is an in-memory data store that can be configured to work in different ways depending on your system's needs.

You can achieve in-memory data store performance with your MongoDB database by:

- configuring your cluster's RAM to accommodate the working set.
- using NVMe SSD drives.
- using the in-memory storage engine.

Ready to get started?

Launch a new cluster or migrate to MongoDB Atlas with zero downtime.

Try free

FAQs

When to use Redis vs MongoDB

Having conceptually opposing storage strategies—in-memory vs on-disk—Redis and MongoDB can serve different purposes in a system. Redis is best suited for querying real-time data when durability and consistency aren't relevant. MongoDB is a general-purpose database solution that can also be effectively used as an in-memory store.



Can MongoDB replace Redis?



MongoDB can be optimized for intensive data access and uses cache mechanisms to achieve in-memory database performance. With that said, MongoDB can support real-time data access for most modern applications.



Can you use Redis with MongoDB?

Redis can be used as a cache, a message broker, or a session manager together with a persistent storage database such as MongoDB.



 English 

About

Careers

Legal Notices

Security Information

Investor Relations

Privacy Notices

Trust Center

Support

[Contact Us](#)

[Customer Portal](#)


[Atlas Status](#)


[Customer Support](#)


[Manage Cookies](#)

Social

 [GitHub](#)


 [Stack Overflow](#)

 [LinkedIn](#)

 [YouTube](#)

 [X](#)

 [Twitch](#)

 [Facebook](#)

© 2024 MongoDB, Inc.