What's New in MongoDB 3.0

August 2015

mongoDB

Table of Contents

Introduction

MongoDB 3.0 is the most significant release to date of the world's fastest growing database. The release dramatically expands the set of mission-critical applications that are best suited for MongoDB by introducing a new and highly flexible storage architecture, incorporating the WiredTiger technology acquired by MongoDB in 2014. Performance and scalability enhancements included in this release now place MongoDB at the forefront of the database market, eliminating the need for niche alternatives, and positioning MongoDB as the standard DBMS for modern applications.

With MongoDB 3.0 and Ops Manager, organizations can leverage the essential capabilities they have come to expect from a database – mature management tools, a powerful query language, comprehensive indexes, and a strong consistency model – while capitalizing on the innovations of the NoSQL movement including a flexible document data model, horizontal scalability, global always-on architecture, and unprecedented performance. The union of these capabilities dramatically improves time to value for organizations, and further extends MongoDB in new classes of applications, such as Internet of Things (IoT), time series data analysis, messaging, and fraud detection.

MongoDB 3.0 combines 7x to 10x higher performance, 80% lower storage requirements, and 95% less operational overhead. These enhancements, coupled with a new pluggable storage engine architecture, enable organizations to use MongoDB 3.0 in building applications never before possible, and run at efficiency levels never before attainable.

Key innovations delivered in MongoDB 3.0 include:

Pluggable storage engines: addressing diverse use-cases with a single technology

- The new MongoDB storage engine API makes it fast and easy for MongoDB and the ecosystem to build new pluggable storage engines that allow the database to be extended with new capabilities, and to be configured for specific hardware architectures. Now users can leverage the same MongoDB Query Language, data model, scaling, security and operational tooling across different applications each powered by different pluggable MongoDB storage engines.

- MongoDB 3.0 ships with two supported storage engines: MMAPv1 (Memory Mapped Version 1) engine

– an improved version of the engine used in prior MongoDB releases; and the new WiredTiger storage engine bringing higher concurrency and compression. Both engines can coexist within the same MongoDB replica set, making it simple to evaluate and migrate between them. An experimental in-memory storage engine is also shipped with MongoDB 3.0.

7x to 10x higher write performance with 80% greater storage efficiency

- The MongoDB WiredTiger storage engine implements document level concurrency control, delivering blazing performance gains and vertical scalability to many workloads.
- Native compression reduces the amount of storage required by up to 80%, unlocking new infrastructure efficiency and cost savings.
- Replica sets can now grow up to 50 members, reducing the effects of network latency by geographically distributing data closer to users.
- Operations teams can bring new replica set members into service faster with parallelized replay of the oplog across multiple collections.

95% reduction in operational overhead

- MongoDB Ops Manager reduces tasks such as deployment, scaling, upgrades and backups to just a few clicks or an API call. Continuous, point-in-time backups and real-time alerting on over 100 system metrics help ensure always-on availability. Ops Manager is available as part of MongoDB Enterprise Advanced.
- Greater control over MongoDB's logging granularity coupled with the addition of severity messages to each log message makes it possible to yield deeper visibility into the database for diagnostics and debugging, without overwhelming DBAs or systems with extraneous log data.

End-to-end auditing for security compliance

- The auditing framework introduced in MongoDB 2.6 has been extended to include the logging of read and write (DML) operations to the database. With these extensions administrators can now construct and filter audit trails for any operation against MongoDB, without having to rely on third party tools.
- Role-based auditing has been added to enhance auditing selectivity. Now it is possible to log and report on activities by role.
- Auditing is available as part of MongoDB Enterprise Advanced.

Enhanced query language and tools

- Data can be loaded and extracted at higher speed and with greater efficiency using MongoDB's revised multi-threaded `mongoimport`, `mongoexport`, `mongodump`, `mongorestore` and `mongooplog` tools.
- DBA productivity is increased with the new MongoDB `explain()` method. It is now possible to calculate and review query plans without first running the query. The query plan can be applied to a broader set of query types, and error handling is improved.
- With the addition of big polygon selections, MongoDB's geospatial support is extended to include multi-hemisphere queries that can span areas extending across more than 50% of the earth's surface.
- Developers can construct richer time-series analytics queries with less code using the aggregation framework's new `$dateToString` operator.

You can learn more about each of the new MongoDB 3.0 features in the rest of this guide.

Pluggable Storage Engines: Extending MongoDB to New Applications

Simplify Building and Operating Modern Apps: Storage Engine API

With users building increasingly complex data-driven apps, there is no longer a "one size fits all" database storage technology capable of powering every type of application built by the business. Modern applications need to support a variety of workloads with different access patterns and price/performance profiles – from low latency, in-memory read and write applications, to real time analytics to highly compressed "active" archives.

Through the use of pluggable storage engines exposed by the new storage engine API, MongoDB can be extended with new capabilities, and configured for optimal use of specific hardware architectures. This approach significantly reduces developer and operational complexity compared to running multiple databases. Now users can leverage the same MongoDB query language, data model, scaling, security and operational tooling across different applications, each powered by different pluggable MongoDB storage engines.

Multiple storage engines can co-exist within a single MongoDB replica set, making it easy to evaluate and migrate engines. Running multiple storage engines within a replica set can also simplify the process of managing the data lifecycle. For example as different storage engines for MongoDB are developed, it would be possible to create a mixed replica set configured in such a way that:

1. Operational data requiring low latency and high throughput performance is managed by replica set members using the WiredTiger or in-memory storage engine (currently experimental).

2. Replica set members configured with an HDFS storage engine expose the operational data to analytical processes running in a Hadoop cluster, which is executing interactive or batch operations rather than real time queries.

MongoDB replication automatically migrates data between primary and secondary replica set members, independent of their underlying storage format. This eliminates complex ETL tools that have traditionally been used to manage data movement.

MongoDB 3.0 ships with two supported storage engines:

- The default MMAPv1 engine, an improved version of the engine used in prior MongoDB releases, now enhanced with collection level concurrency control.
- The new WiredTiger storage engine. For many applications, WiredTiger's more granular concurrency control and native compression will provide significant benefits in the areas of lower storage costs, greater hardware utilization, higher throughput, and more predictable performance.

Both storage engines can co-exist in a single replica set, managed by MongoDB Ops Manager or MongoDB Cloud Manager, discussed later in this guide. MongoDB 3.0 also ships with an experimental In-Memory storage engine. Other engines under development by MongoDB and the community include the RocksDB Key-Value engine, HDFS storage engine and a FusionIO engine that bypasses the filesystem. These and other engines may be supported in the future, based on customer demand.

MongoDB WiredTiger: A New Storage Engine for High Scale Apps

WiredTiger is a new storage engine for MongoDB, developed by the architects of Berkeley DB, the most widely deployed embedded data management software in the world. WiredTiger scales on modern, multi-CPU architectures. Using a variety of programming techniques such as hazard pointers, lock-free algorithms, fast latching and message passing, WiredTiger performs more work per CPU core than alternative engines. To minimize on-disk overhead and I/O, WiredTiger uses compact file formats, and optionally, compression.

For many applications, WiredTiger will provide significant benefits in the areas of lower storage costs, greater hardware utilization, and more predictable performance, especially by reducing query latency in 95th and 99th percentile.

Upgrades to the WiredTiger storage engine are non-disruptive for existing replica set deployments; applications will be 100% compatible, and upgrades can be performed with zero downtime through a rolling upgrade of the MongoDB replica set. This approach makes it very simple to migrate and test existing applications. Review the documentation for a checklist and full instructions on the upgrade process.

The WiredTiger storage engine ships as part of MongoDB alongside the default MMAPv1 storage engine, and can be configured when starting the server using the following option:

```
mongod --storageEngine wiredtiger
```

	MongoDB WiredTiger	MongoDB MMAPv1
Write Performance	Excellent Document-Level Concurrency Control	Good Collection-Level Concurrency Control
Read Performance	Excellent	Excellent
Compression Support	Yes	No
MongoDB Query Language Support	Yes	Yes
Secondary Index Support	Yes	Yes
Replica Sets	Yes	Yes
Automatic Sharding	Yes	Yes
Ops Manager & Cloud Manager Support	Yes All features including deployment, upgrade backup, restore, and monitoring	Yes All features including deployment, upgrade backup, restore, and monitoring
Security Controls	Yes	Yes
Platform Availability	Linux, Windows, Mac OS X	Linux, Windows, Mac OS X, Solaris (x86)

* GridFS supports larger file sizes

Table 1: Comparing the MongoDB WiredTiger and MMAPv1 storage engines

Whether configured with the WiredTiger or MMAPv1 storage engines, developers and administrators interact with MongoDB in exactly the same way. The Performance Best Practices whitepaper and the Operations Guide detail specific optimizations that can be applied for each storage engine. The MongoDB storage documentation highlights differences in journaling and record allocation strategies.

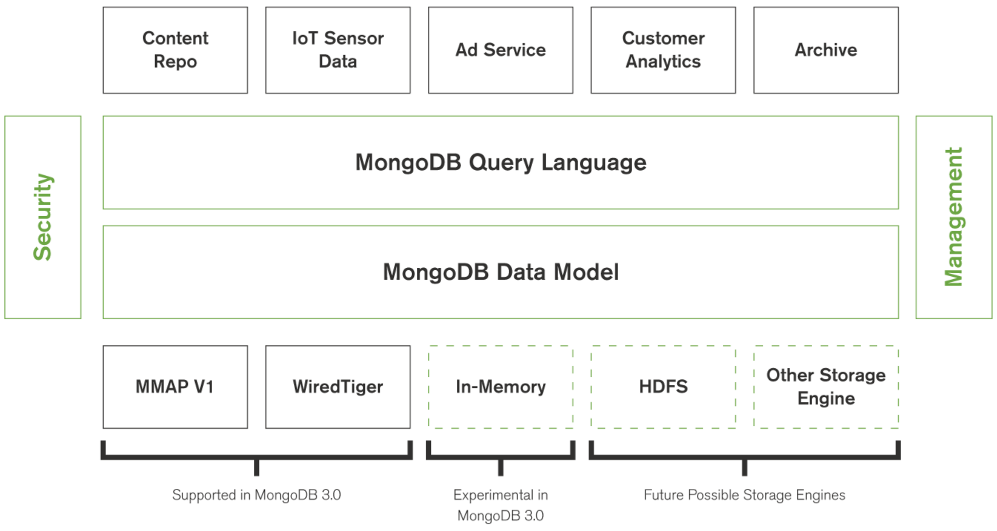


Figure 1: Mix and match storage engines within a single MongoDB replica set

Higher Performance & Efficiency

Between 7x and 10x Greater Write Performance

MongoDB 3.0 provides more granular document-level concurrency control, delivering between 7x and 10x greater throughput for most write-intensive applications, while maintaining predictable low latency.

The updated MongoDB MMAPv1 storage engine implements collection level concurrency control while the new MongoDB WiredTiger storage engine further improves performance for many workloads by implementing concurrency control at the document level.

As demonstrated by recent benchmarks, implementing concurrency control at the document level improves performance significantly when compared to the previous MongoDB 2.6 release. In each test, predictable low latency is maintained as the workload is scaled. Update latency is significantly improved in MongoDB 3.0: it has been reduced by almost 90% at both the 95th and 99th percentiles.

Migrating to the WiredTiger storage engine will deliver the most noticeable performance gains on highly write-intensive applications. Examples include

- IoT applications: sensor data ingestion and analysis;
- Customer data management and social apps: updating all user interactions and engagement from multiple activity streams;
- Mobile applications: SMS, CDRs (Call Detail Records) and gaming.

Higher concurrency also drives infrastructure simplification. Applications can fully utilize available server resources, simplifying the architecture needed to meet performance SLAs. With the more coarse grained database-level locking of previous MongoDB generations, users often had to implement sharding in order to scale workloads stalled by a single write lock to the database, even when sufficient memory, I/O bandwidth and disk capacity was still available in the host system. Greater system utilization enabled by fine-grained concurrency reduces this overhead, eliminating unnecessary cost and management load.

Compression: Up to 80% Reduction in Storage Costs

Despite data storage costs declining 30% to 40% per annum, overall storage expenses continue to escalate as data volumes double every 12 to 18 months. To make matters worse, improvements to storage bandwidth and latency are not keeping pace with data growth, making disk I/O a common bottleneck to scaling overall database performance.

MongoDB now supports native compression with the WiredTiger engine, reducing physical storage footprint by as much as 80% compared to MongoDB configured with MMAPv1. In addition to reduced storage space, compression enables much higher storage I/O scalability as fewer bits are read from disk.

Administrators have the flexibility to configure specific compression algorithms for collections, indexes and the journal, choosing between:

- Snappy (the default library for documents and the journal), providing a good balance between high compression ratio – typically around 70%, depending on document data types – and low CPU overhead.
- zlib, providing higher document and journal compression ratios for storage-intensive applications, at the expense of extra CPU overhead.
- Prefix compression for indexes reducing the in-memory footprint of index storage by around 50% (workload dependent), freeing up more of the working set for frequently accessed documents.

Administrators can modify the default compression settings for all collections and indexes. Compression is also configurable on a per-collection and per-index basis during collection and index creation.

By introducing compression, operations teams get higher performance per node and reduced storage costs.

	New York City	San Francisco	London	Sydney
New York City	--	84.42	70.64	231.46
San Francisco	85.31	--	171.91	158.26
London	69.39	168.20	--	318.77
Sydney	231.14	158.26	315.53	--

Table 2: Network latencies (milliseconds) between cities[1]

Creating Multi-Temperature Storage

Combining compression with MongoDB's location-aware sharding, administrators can build highly efficient tiered storage models to support the data lifecycle. Administrators can balance query latency with storage density and cost by assigning data sets to specific storage devices. For example, consider an application where recent data needs to be accessed quickly, while for older data, latency is less of a priority than storage costs:

- Recent, frequently accessed data can be assigned to high performance SSDs with Snappy compression enabled.
- Older, less frequently accessed data is tagged to higher capacity, lower-throughput hard disk drives where it is compressed with zlib to attain maximum storage density and lower cost-per-bit.

MongoDB will automatically migrate data between storage tiers based on user-defined policies without administrators having to build tools or ETL processes to manage data movement.

You can learn more about using location-aware sharding for this deployment model by reading the Tiered Storage Models in MongoDB post.

Improving Customer Experience: Large Replica Sets

Delivering a low latency experience to customers wherever they are located is a key design consideration for distributed systems. Using MongoDB's native replica sets, copies (replicas) of the database can be deployed to sites physically closer to users, thereby reducing the effects of network latency.

Reads can be issued with the `nearest` read preference, ensuring the query is served from the replica closest to the user, based on ping distance. This use case is especially relevant for reference data sets used in global financial reporting applications, or retail applications where sales, catalog or inventory data can be pushed out to local stores.

Previously MongoDB supported a maximum of 12 members per replica set, which limited deployments beyond more than three or four remote offices (depending on the number of replica set members deployed per location).

MongoDB 3.0 now supports up to 50 members per replica set, enabling wider data distribution across a greater number of sites, and greater resilience through additional node redundancy at each location. As with earlier releases, seven members are eligible to vote in replica set elections.

In addition to broader data distribution, replica sets can also be configured with a greater number of members performing specialized tasks:

- More hidden replica set members can be deployed to run applications such as analytics and reporting that require isolation from regular operational workloads.
- More delayed replica set members can be deployed to provide "historical" snapshots of data at different intervals in time for use in recovery from certain errors, such as "fat-finger" mistakes dropping databases or collections.

Building on the replication enhancements introduced with MongoDB 3.0, operations teams can bring new replica set members into service faster with improved parallel oplog replay across multiple collections.

Simplified Operations

The Best Way to Run MongoDB: Ops Manager

MongoDB Ops Manager is the best way to run MongoDB within your own data center or public cloud, making it fast and easy for operations teams to deploy, monitor, backup and scale MongoDB. Ops Manager was created by the engineers who develop the database and reduces the overhead of operating large scale MongoDB deployments by as much as 95% for many activities. Ops Manager is available with MongoDB Enterprise Advanced.

Ops Manager incorporates best practices to help keep managed databases healthy and optimized. It ensures operational continuity by converting complex manual tasks into reliable, automated procedures executed with the click of a button or an API call. Ops Manager takes care of the low-level details for common tasks without taking the database offline.

- **Deploy.** Any topology, at any scale
- **Manage.** Deploy new or manage, monitor and backup existing clusters[2]
- **Upgrade.** In minutes, with no downtime
- **Scale.** Add capacity, without taking the application offline
- **Point-in-time, Scheduled Backups.** Restore to any point in time, because disasters aren't predictable
- **Performance Alerts.** Monitor 100+ system metrics and get custom alerts before the system degrades

MongoDB consulting services assists you in every stage of planning and implementing your Ops Manager deployment, including the production of a MongoDB operations playbook.

For those operations teams who do not want to maintain their own management and backup infrastructure in-house, the benefits of Ops Manager are also available through MongoDB Cloud Manager, a cloud service hosted by MongoDB. Organizations that subscribe to MongoDB Enterprise Advanced can choose between Ops Manager and Cloud Manager for their deployments.

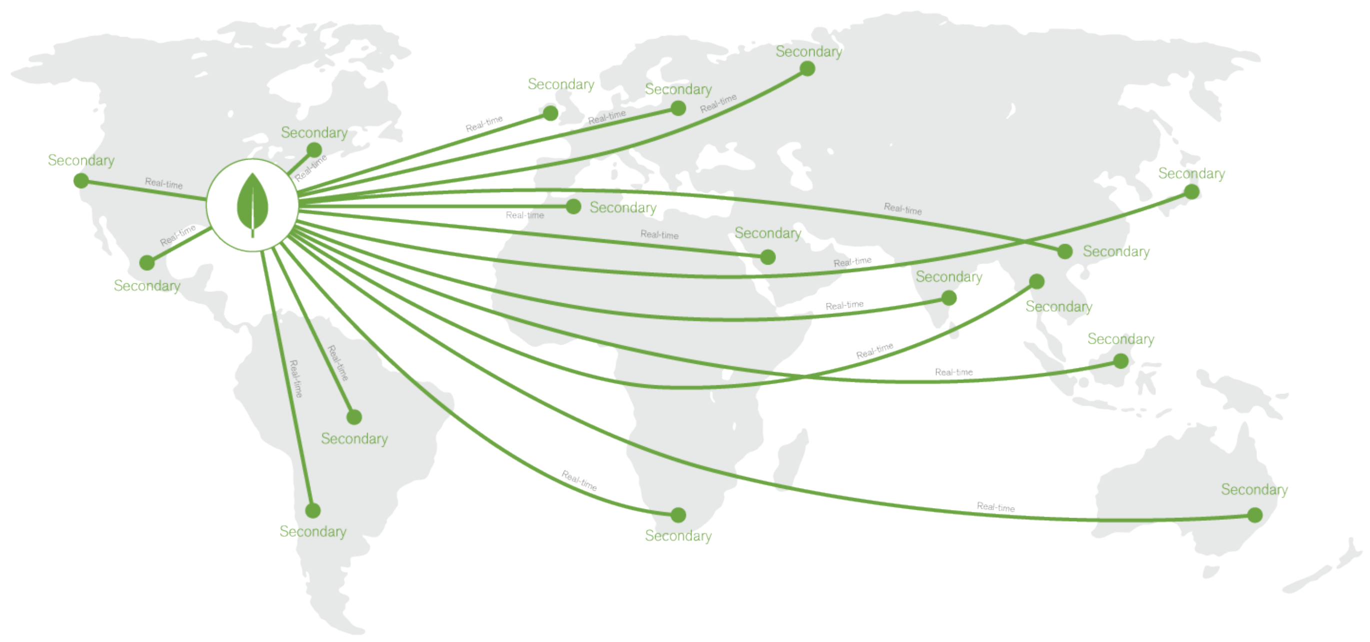
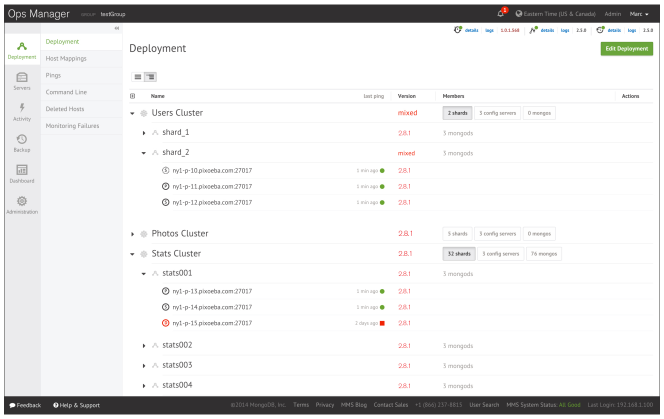


Figure 2: Up to 50 members per MongoDB replica set delivers higher fault tolerance and reduced read latency for highly distributed consumers

1. https://wondernetwork.com/pings/
2. Support for adopting existing MongoDB deployments scheduled for Q1 Calendar Year 2015



Deployment

Name	last ping	Version	Members	Actions
Users Cluster		mixed	2 shards / 3 config servers / 0 mongos	
shard_1		2.8.1	3 mongods	
shard_2		mixed	3 mongods	
ny1-p-10.pixoeba.com:27017	1 min ago ●	2.8.1		
ny1-p-11.pixoeba.com:27017	1 min ago ●	2.8.1		
ny1-p-12.pixoeba.com:27017	1 min ago ●	2.8.1		
Photos Cluster		2.8.1	5 shards / 3 config servers / 0 mongos	
Stats Cluster		2.8.1	32 shards / 3 config servers / 76 mongos	
stats001		2.8.1	3 mongods	
ny1-p-13.pixoeba.com:27017	1 min ago ●	2.8.1		
ny1-p-14.pixoeba.com:27017	1 min ago ●	2.8.1		
ny1-p-15.pixoeba.com:27017	2 days ago 🟥	2.8.1		
stats002		2.8.1	3 mongods	
stats003		2.8.1	3 mongods	
stats004		2.8.1	3 mongods	

Figure 3: Ops Manager: simple, intuitive and powerful. Deploy and upgrade entire clusters with a single click.

Deployments and Upgrades

Ops Manager coordinates critical operational tasks across the nodes in a MongoDB system. It communicates with the infrastructure through agents installed on each server. The servers can reside in the public cloud or a private data center. Ops Manager reliably orchestrates the tasks that administrators have traditionally performed manually – deploying a new cluster, upgrades, point in time recovery, and many other operational tasks.

Ops Manager is designed to reliably make changes to the system. Here is how it works.

- Ops Manager agents are installed on servers (where MongoDB will be deployed), either through provisioning tools such as Chef or Puppet, or by an administrator.
- The administrator creates a new design goal for the system, either as a modification to an existing deployment (e.g., upgrade, oplog resize, new shard), or as a new system.
- The agents periodically check in with Ops Manager and receive the new design instructions.
- Agents create and follow a plan for implementing the design. Using a sophisticated rules engine, agents continuously adjust their individual plans as conditions change. When faced with a failure scenarios – such as server failures and network partitions – agents will revise their plans to reach a safe state.
- Minutes later, the system is deployed, safely and reliably.

In addition to initial deployment, Ops Manager makes it possible to dynamically resize capacity by adding shards and replica set members. Other maintenance tasks such as upgrading MongoDB or resizing the oplog can be reduced from dozens or hundreds of manual steps to the click of a button, all with zero downtime.

Administrators can use the Ops Manager interface directly, or invoke the Ops Manager RESTful API from existing enterprise tools, including popular monitoring and orchestration frameworks.

Monitoring

Featuring charts, custom dashboards, and automated alerting, Ops Manager tracks 100+ key database and systems health metrics including operations counters, memory and CPU utilization, replication status, open connections, queues and any node status.

The metrics are securely reported to Ops Manager where they are processed, aggregated, alerted and visualized in a browser, letting administrators easily determine the health of MongoDB in real-time. Views can be based on explicit permissions, so project team visibility can be restricted to their own applications, while systems administrators can monitor all the MongoDB deployments in the organization.

Historic performance can be reviewed in order to create operational baselines and to support capacity planning. Integration with existing monitoring tools is also straightforward via the Ops Manager RESTful API, making the deep insights from Ops Manager part of a consolidated view across your operations.

Ops Manager allows administrators to set custom alerts when key metrics are out of range. Alerts can be configured for a range of parameters affecting individual hosts, replica sets, agents and backup. Alerts can be sent via SMS and email or integrated into existing incident management systems such as PagerDuty and HipChat to proactively warn of potential issues, before they escalate to costly outages.

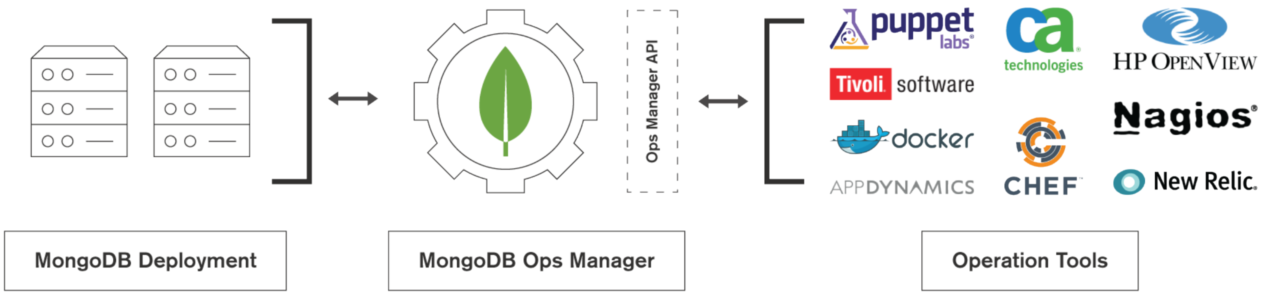


Figure 4: With its RESTful API Ops Manager integrates with leading enterprise operations orchestration and management tools.

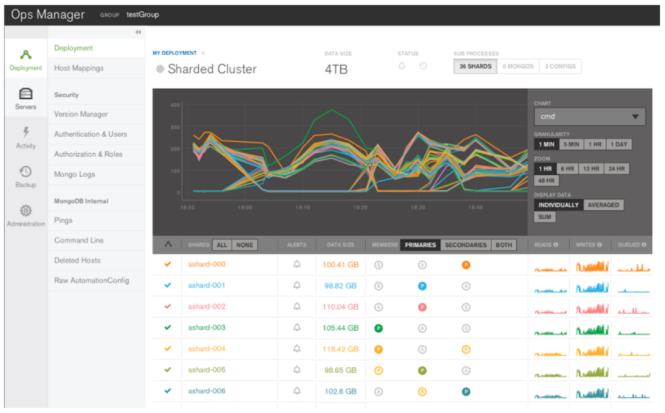
Disaster Recovery: Backups & Point-in-Time Recovery

A backup and recovery strategy is necessary to protect your mission-critical data against catastrophic failure, such as a fire or flood in a datacenter, or human error such as bad code. It is part of multi-faceted strategy which includes replication to withstand failures of critical infrastructure. With a backup and recovery strategy in place, administrators can restore business operations without data loss, and the organization can meet regulatory and compliance requirements. Taking regular backups offers other advantages, as well. The backups can be used to seed new environments for development, staging, or QA without impacting production systems.

Ops Manager continuously maintains backups, therefore if MongoDB experiences a failure, the most recent backup is only moments behind, minimizing exposure to data loss. Ops Manager is the only MongoDB solution that offers point-in-time backups of replica sets and cluster-wide snapshots of sharded clusters. You can restore to precisely the moment you need, quickly and safely. In addition, the on-going performance impact to the production system is minimal – similar to that of adding an additional member to a replica set.

Customers can deploy Ops Manager to control backups in their local data center, or use Cloud Manager which offers a fully managed backup solution with a pay-as-you-go model. Using Cloud Manager, dedicated MongoDB engineers monitor customer backups on a 24x365 basis, alerting operations teams if problems arise.

You can learn more about MongoDB Ops Manager by reviewing the documentation.



SHARDS	ALERTS	DATA SIZE
ashard-000		100.41 GB
ashard-001		98.82 GB
ashard-002		110.04 GB
ashard-003		105.44 GB
ashard-004		118.42 GB
ashard-005		98.65 GB
ashard-006		102.6 GB

Figure 5: Ops Manager provides real time & historic visibility into the MongoDB deployment.

Faster Issue Resolution: Enhanced Logging

Log analysis is a critical part of identifying issues and determining root cause. Now in MongoDB 3.0 developers, QA and operations staff have much greater control over the granularity of log messages and specific functional areas of the server to more precisely investigate issues.

Users can configure which specific components of the database should be exposed for higher definition logging, coupled with the addition of severity levels for each log message. For example:

- If developers are diagnosing specific query issues, verbose logging can be enabled just for query and indexing operations.

- If the operations team are debugging network performance issues, they could activate deeper logging levels for replication, networking and sharding components of the mongod process.

Logging controls can be adjusted at run-time when specific diagnostics are needed, and can also be defined in the permanent configuration.

With the selective configuration of logging verbosity by server component, QA staff can expose more granular details of specific MongoDB internals without overwhelming either the systems or IT staff with extraneous log data. This coupled with more parsable logging output helps staff more quickly identify and resolve issues, whether in development, QA or production.

End-to-End Auditing for Compliance

Single View of Database Activity: Auditing Enhancements

Auditing is an essential element of regulatory compliance initiatives, especially those applications managing user data in healthcare, finance, retail and government systems. Auditing captures the activities of users, administrative staff and applications in accessing and processing data, providing a log for security analysis. Coupled with authorization, authentication and encryption, auditing functionality enables MongoDB to be used for a variety of projects subject to regulatory compliance including:

- PCI DSS for managing cardholder information.
- HIPAA standards for managing healthcare information.
- NIST 800-53 catalogs security controls for all U.S. federal information systems except those related to national security.
- STIG for secure installation and maintenance of computer systems, defined for the US Department of Defense.
- European Union Data Protection Directive.
- Asia Pacific Economic Cooperation (APEC) data protection standardization.

The auditing framework introduced in MongoDB 2.6 now extends beyond capturing administrative actions (i.e. schema operations, authentication and authorization activities) to include the logging of read and write (DML) operations to the database. Administrators can construct and filter audit trails for any operation against MongoDB, whether DML, DCL or DDL without having to rely on third party tools. For example, it is possible to log and audit the identities of users who retrieved specific documents, and any changes made to the database during their session.

Administrators can configure MongoDB to log all actions or apply filters to capture only specific events, users or roles. The audit log can be written to multiple destinations in a variety of formats including to the console and syslog (in JSON format), and to a file (JSON or BSON), which can then be loaded to MongoDB and analyzed to identify relevant events. Each MongoDB server writes events to its attached storage. The DBA can then use their own tools to merge these events into a single audit log, enabling a cluster-wide view of operations that affected multiple nodes.

A further enhancement in MongoDB 3.0 is support for role-based auditing. Now it is possible to log and report activities by specific role, such as userAdmin or dbAdmin – coupled with any inherited roles each user has – rather than having to extract activity for each individual administrator.

Auditing adds performance overhead to a MongoDB system. The amount is dependent on several factors including which events are logged, the log format and where the audit log is maintained, such as on an external storage device. Users should consider the specific needs of their application for auditing and their performance goals in order to determine their optimal configuration.

Learn more from the MongoDB auditing documentation. Auditing is available with MongoDB Enterprise Advanced.

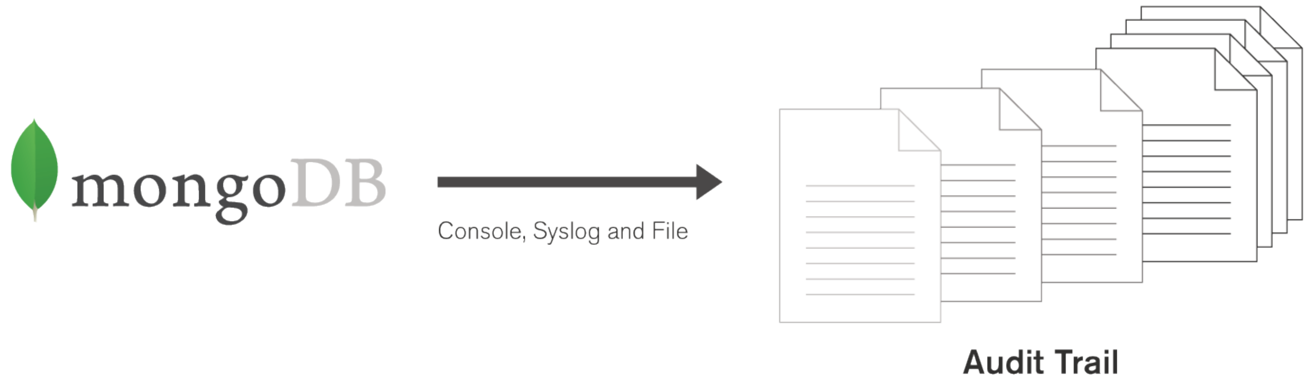


Figure 6: MongoDB logs all operations against the database, providing a complete audit trail for regulatory compliance.

Enhanced Query Language and Tools

Parallelized Data Loading & Export: Multi-Threaded MongoDB Tools

Key MongoDB tools (`mongoimport`, `mongoexport`, `mongodump`, `mongorestore`, `mongostat`, `mongotop` and `mongooplog`) have been re-written as multi-threaded processes in Go, allowing faster operations and smaller binaries.

`mongodump` and `mongorestore` now execute parallelized backup and recovery for small MongoDB instances. Dumps created by earlier releases can be restored to instances running MongoDB 3.0. **Note:** As the only backup solutions that offer cluster-wide snapshots of sharded clusters, Ops Manager and Cloud Manager are recommended for larger MongoDB deployments.

`mongoimport` can parallelize loads across multiple collections with multi-threaded bulk inserts allowing for significantly faster imports of CSV, TSV and JSON data exported from other databases or applications. Ensuring data quality, `mongoimport` now also supports input validation of field names during the import process.

You can read more about each of these tools in the MongoDB Package Components documentation.

Improved DBA Productivity: Enhanced Query Engine Introspection

The MongoDB `explain()` method is an invaluable tool for DBAs in optimizing performance. Using `explain()` output, DBAs can review query plans, ensuring common queries are serviced by well-defined indexes, as well as eliminating any unnecessary indexes that can increase write latency and add overhead during query planning and optimization.

In the latest MongoDB 3.0 release `explain()` has been significantly enhanced:

- The query plan can now be calculated and returned without first having to run the query. This enables DBAs to review which plan will be used to execute the query, without having to wait for the query to run to completion.
- DBAs can run `explain()` to generate detailed statistics on all query plans considered by the optimizer. Execution statistics are available for every evaluated plan, down to the granularity of execution stage. Now, for example, it is possible for the DBA to distinguish the amount of time a query plan spent sorting the result set from the amount of time spent reading index keys.
- The `explain()` method exposes query introspection to a wider range of operations, including find, count, update, remove, group, and aggregate, enabling DBAs to optimize for a wider range of query types.

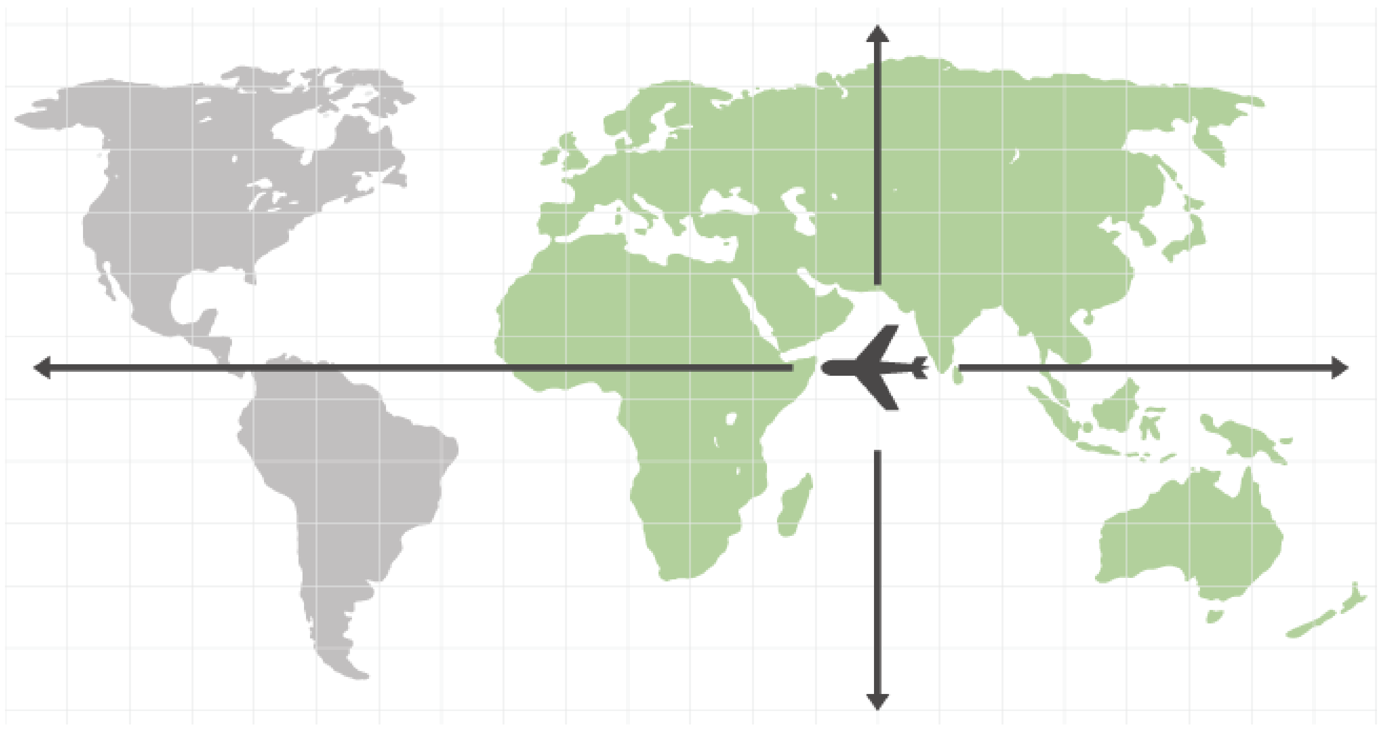


Figure 7: Extending geospatial queries with big polygon support

Review the documentation to learn more about the new `explain()` functionality.

Richer Geospatial Apps: Big Polygon Support for Multi-Hemisphere Queries

MongoDB's geospatial indexes and queries are widely used by developers building modern location-aware applications across industries as diverse as high technology, retail, telecommunications and government. MongoDB 3.0 adds big polygon geospatial support with `$intersects` and `$within` operators, allowing execution of queries over geographic areas extending across multiple hemispheres and areas that exceed 50% of the earth's surface. As an example, an airline can now run queries to identify all its aircraft that have traveled across multiple hemispheres in the past 24 hours.

Learn more from the MongoDB geospatial documentation.

Enhanced Data Type Support: Easier Time-Series Analytics & Reporting

The MongoDB 3.0 aggregation pipeline offers a new `$dateToString` operator that simplifies report generation and grouping data by time interval. The operator formats the ISO Date type as a string with a user-supplied format, allowing developers to construct rich queries with less code.

Documents can be grouped and analyzed by arbitrary dates and times. This grouping is especially useful in the analysis of time series data – for example reporting aggregated sales by hour for each SKU in a specific store.

You can learn more about the `$dateToString` operator from the documentation.

Getting Started with MongoDB 3.0

MongoDB 3.0 delivers significant gains in performance, efficiency and flexibility. It is easy to get started:

- MongoDB 3.0 is available from the MongoDB downloads page
- MongoDB Enterprise Advanced and Ops Manager is available for download at no cost for evaluation and development.

Review the release notes for more detail on upgrading from previous releases to MongoDB 3.0. Ops Manager and Cloud Manager make it easy to upgrade between releases, and to simplify evaluation, users can run both the MMAPv1 and WiredTiger storage engines in the same replica set. The documentation provides a checklist and instructions on upgrading from a standalone MongoDB instance, a replica set and a sharded cluster.

We Can Help

We are the MongoDB experts. Over 2,000 organizations rely on our commercial products, including startups and more than a third of the Fortune 100. We offer software and services to make your life easier:

MongoDB Enterprise Advanced is the best way to run MongoDB in your data center. It's a finely-tuned package of advanced software, support, certifications, and other services designed for the way you do business.

MongoDB Cloud Manager is the easiest way to run MongoDB in the cloud. It makes MongoDB the system you worry about the least and like managing the most.

MongoDB Professional helps you manage your deployment and keep it running smoothly. It includes support from MongoDB engineers, as well as access to MongoDB Cloud Manager.

Development Support helps you get up and running quickly. It gives you a complete package of software and services for the early stages of your project.

MongoDB Consulting packages get you to production faster, help you tune performance in production, help you scale, and free you up to focus on your next release.

MongoDB Training helps you become a MongoDB expert, from design to operating mission-critical systems at scale. Whether you're a developer, DBA, or architect, we can make you better at MongoDB.

Contact us to learn more, or visit www.mongodb.com.

Resources

For more information, please visit mongodb.com or contact us at sales@mongodb.com.

Case Studies (mongodb.com/customers)
Presentations (mongodb.com/presentations)
Free Online Training (university.mongodb.com)
Webinars and Events (mongodb.com/events)
Documentation (docs.mongodb.org)
MongoDB Enterprise Download (mongodb.com/download)

mongoDB

New York • Palo Alto • Washington, D.C. • London • Dublin • Barcelona • Sydney • Tel Aviv
US 866-237-8815 • INTL +1-650-440-4474 • info@mongodb.com