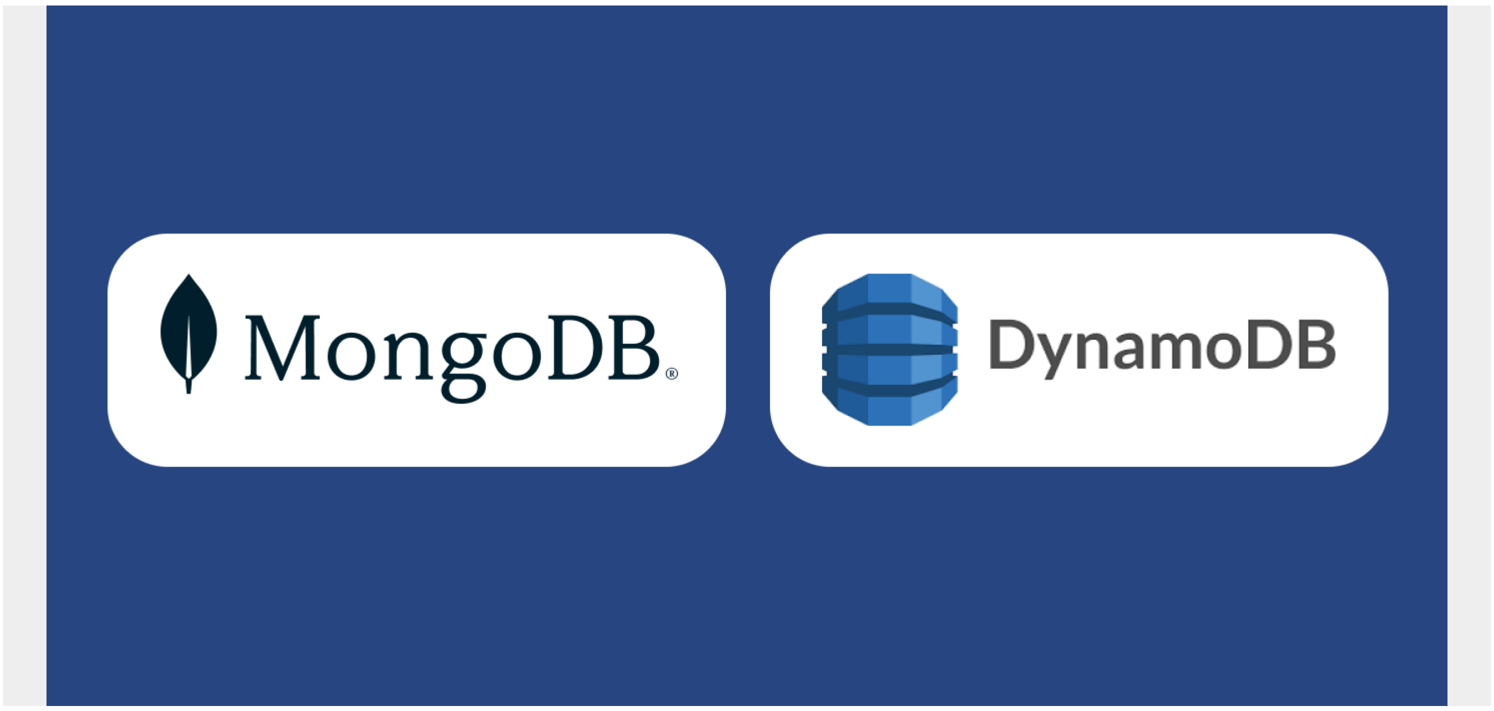
DYNAMODB VS MONGODB: COMPARING NOSQL DATABASES



With the need for more flexible backend solutions, NoSQL databases have gained enormous popularity. They power applications that require a more flexible data structure than traditional structured databases can provide.

Comparing the leading NoSQL Databases

Two of the most popular market leaders for NoSQL databases are MongoDB and DynamoDB, which provide robust feature-rich NoSQL database platforms.

In this article, we will look at the difference between MongoDB and DynamoDB to help you choose the right database for your project.

What is MongoDB?

MongoDB is a general purpose, document-based distributed NoSQL database. It is open source, so you don't have to pay licensing fees and can scale it across multiple services for low-cost flexibility and extensibility. You are not locked into a single vendor, giving you autonomy in how you source and manage your database infrastructure. With an active user community, continuous development, testing and troubleshooting support, and faster innovation, it provides you with ready talent that can help you integrate and customize according to your application. MongoDB offers:

- A free and production-ready community version
- An enterprise edition with direct support from MongoDB Inc.

MongoDB Atlas is the cloud version of the database. Unlike other offerings, MongoDB Atlas is cloud-agnostic and can be deployed on any major cloud provider with guaranteed availability and scalability while conforming to all the compliance and security standards. It also provides a free MongoDB cluster that can be used for application development.

(Explore our multi-part MongoDB Tutorial Guide.)

MongoDB core components

Here are the core components of MongoDB and their usage:

MongoDB	RDBMS Counterpart	Description
Collection	Table	A set of MongoDB documents.
Document	Row	Collection of data stored in BSON format
Field	Field/Column	A single element in a MongoDB document containing values as field and value pairs

What is DynamoDB?

DynamoDB is a proprietary NoSQL database by Amazon that supports key-value and document data offered via the Amazon Web Services. This AWS cloud-only offering provides a scalable, highly available, and secure managed database platform for any application.

DynamoDB is fully managed by AWS, so you don't need to worry about provisioning, configuring, or maintaining servers to run it. You can automatically scale up or down, depending on need. As for security and availability, the distributed AWS infrastructure protects you against any local issues or hardware failures. Backups are automatic and restoring is easy.

AWS is also well-known for its data-at-rest (DAR) and data-in-transit (DIT) encryption that meet many security framework requirements.

DynamoDB offers limited storage, ever free database instances to AWS users using their AWS Free Tier for testing and development purposes.

(Learn more in our DynamoDB Guide.)

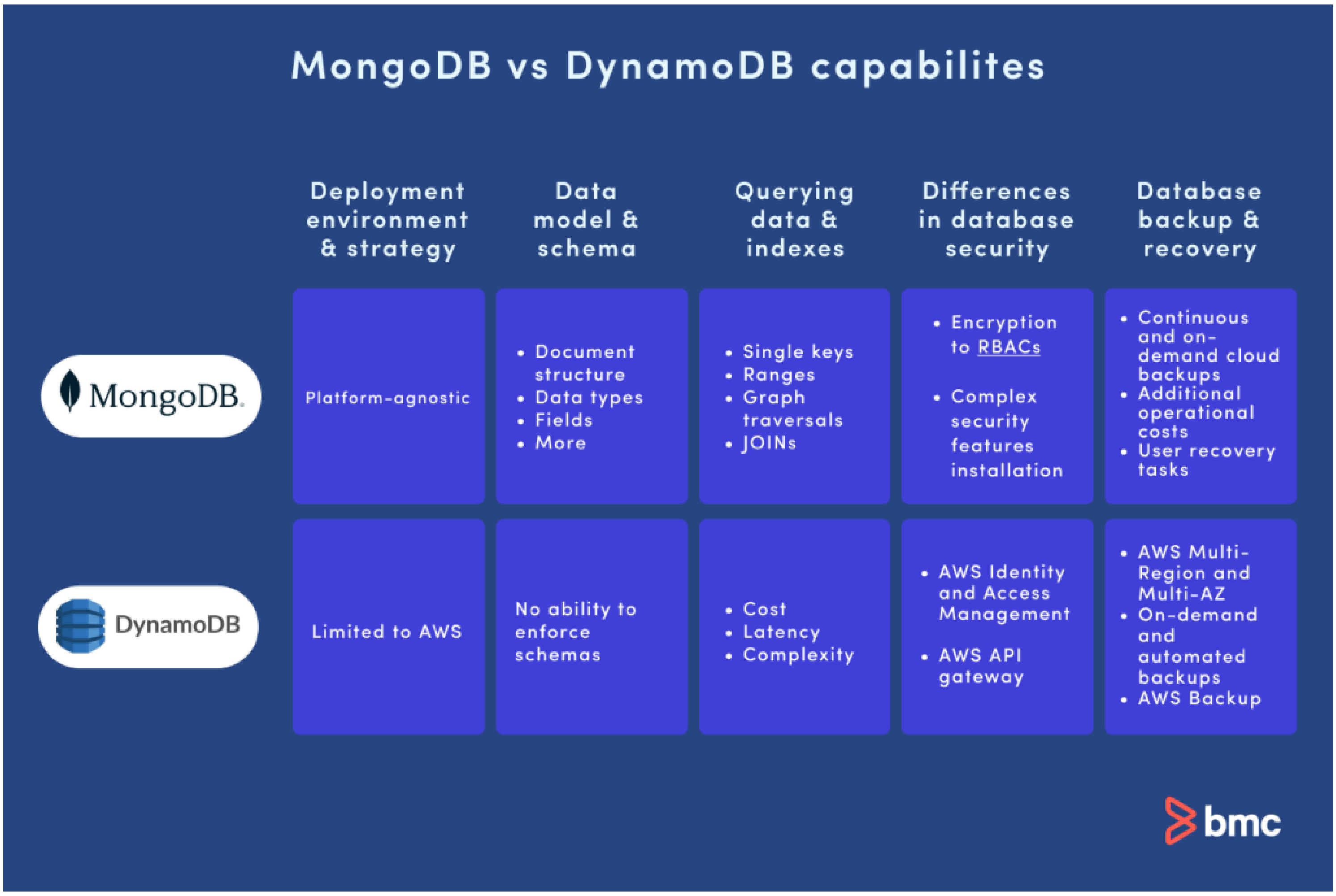
DynamoDB Core Components

These are the core components of DynamoDB:

DynamoDB	RDBMS Counterpart	Description
Table	Table	A grouped collection of DynamoDB Items.
Item	Row	Data records that contain data. Each item consists of one or more attributes.
Attribute	Field/Column	The base element of DynamoDB contains a single data value.

Differences between MongoDB vs DynamoDB

Both MongoDB and DynamoDB offer similar functionality and feature sets. However, there are some key differentiating factors when it comes to selecting between these two.



MongoDB vs DynamoDB capabilites

	Deployment environment & strategy	Data model & schema	Querying data & indexes	Differences in database security	Database backup & recovery
MongoDB	Platform-agnostic	• Document structure • Data types • Fields • More	• Single keys • Ranges • Graph traversals • JOINs	• Encryption to RBACs • Complex security features installation	• Continuous and on-demand cloud backups • Additional operational costs • User recovery tasks
DynamoDB	Limited to AWS	No ability to enforce schemas	• Cost • Latency • Complexity	• AWS Identity and Access Management • AWS API gateway	• AWS Multi-Region and Multi-AZ • On-demand and automated backups • AWS Backup

MongoDB and DynamoDB deployment environment & strategy

The most significant difference between these two databases:

- MongoDB is platform-agnostic.
- DynamoDB is limited to AWS.

With MongoDB, you can configure the database to run virtually anywhere from a local machine, container, or on-premise deployment to any cloud provider.

In contrast, you can only configure and use DynamoDB through AWS. As a native AWS application, DynamoDB has much tighter integration with other tools and services offered by AWS. Even though DynamoDB offers a downloadable version for testing and development, production deployments depend on AWS.

DynamoDB is a fully managed database that allows users to start using the database straightaway, as AWS will manage all the scaling, availability, and updates. It enables provisioning a multi-region, highly available database with just a few clicks, drastically reducing the need for dedicated infrastructure management.

On the other hand, MongoDB requires users to manage all the infrastructure and configurations for a based MongoDB deployment. Though this provides users with the greatest degree of control over the database, it comes with increased complexity.

Fortunately, MongoDB Atlas provides a solution for this complexity issue by offering a fully managed cloud database service that can be deployed across AWS, Google Cloud, and Microsoft Azure. It also provides the easiest option to create a multi-cloud, multi-region NoSQL database solution.

Both these databases can be integrated with CI/CD tools and managed as a part of the DevOps process. Both support Terraform, an Infrastructure as Code tool to provision and manage database infrastructure easily.

Data model & schema in Dynamo vs Mongo

MongoDB uses the BSON format to store its data in documents with support for a greater variety of data types ranging from strings, timestamps to different integer and decimal types. Additionally, it supports document sizes up to 16MB—and you can extend this limit by breaking down data into multiple documents using GridFS.

In contrast, DynamoDB offers a limited number of available data types while single items are limited to 400KB.

MongoDB is a schema-free database. Yet, it allows users to enforce a schema with its built-in schema validation if a need arises. These schemas can be used to validate:

- Document structure
- Data types
- Fields
- Etc.

Conversely, DynamoDB is a schema-less database but with no ability to enforce schemas.

Both databases support ACID transactions.

Querying data & indexes in MongoDB vs DynamoDB

MongoDB offers more flexibility in querying data as it enables users to aggregate and query data natively in multiple ways, such as:

- Single keys
- Ranges
- Graph traversals
- JOINs
- Etc.

On the contrary, DynamoDB natively supports only key-value queries yet allows users to carry out complex aggregations using other AWS services such as Amazon Redshift and Elastic MapReduce. However, using these different services increases:

- Cost
- Latency
- Complexity

When it comes to indexing, MongoDB supports creating indexes to any field in a document with full support for secondary indexes. Furthermore, it supports different indexing types such as compound, TTL, hash, wildcard, text, array, etc.… and the indexes are strongly consistent with the underlying data.

Meanwhile, DynamoDB supports two types of secondary indexes:

- **Global Secondary Index (GSI)**, which spans across all data in the base table across all the partitions and can be stored and scaled separately from the underlying data table.
- **Local Secondary Index (LSI)**, which limits the scope to the base table, where the value of the table partition key matches the LSI partition key.

Both these databases support multi-document transactions, but with key differences:

- MongoDB supports read and writes to the same documents and fields in a single database transaction.
- DynamoDB lacks support for multiple operations within a single transaction.

Differences in database security when using MongoDB vs DynamoDB

DynamoDB comes with baked-in security best practices as their service is managed by AWS. The authentication and authorization model is based on AWS Identity and Access Management (IAM) which offers users fine-grained control over users, roles, and policies that are used to interact with DynamoDB.

Besides, DynamoDB is not directly connected to the wider internet since requests are routed through an API gateway where the authorization is managed by AWS.

In a standard MongoDB installation, the users are responsible for most of the security practices from managing access, routing traffic, firewalls, etc. Though this offers greater control over the database, it is a complex and time-consuming task that is not scalable in rapidly changing development lifecycles.

However, MongoDB Atlas provides much better baked-in security best practices from the get-go, such as network solutions and encryption to RBACs for authentication and authorization. Both databases support encryption of data at rest.

Database backup & recovery

DynamoDB offers Multi-Region and Multi-AZ data replication out of the box as a part of its AWS service. This supports both on-demand and automated (continuous) backups with point-in-time recovery. Any other backups can be easily configured via AWS tools like "AWS Backup" and stored with AWS itself.

MongoDB Atlas also supports continuous and on-demand cloud backups, yet it will require more configurations than DynamoDB to get everything properly configured. Furthermore, the users are solely responsible for all the backup and recovery tasks in an on-premises or a manual MongoDB deployment—leading to additional operational costs.

Determining which database is right for your situation

Selecting the right database is not a simple choice. It depends on a multitude of factors:

- User requirements
- Deployment
- Storage requirements

- Functionality
- Other factors

Use cases of DynamoDB vs MongoDB

Even when comparing MongoDB vs DynamoDB, we cannot compare them directly as they are targeted toward different use cases.

For instance, DynamoDB is a managed NoSQL database *service*, while MongoDB is NoSQL database *software*. Thus, the nearest direct comparison would be with MongoDB Atlas, the managed database offered by MongoDB Inc and DynamoDB.

If you are currently using the AWS echo system to deploy and manage applications, then DynamoDB offers the best in:

- Compatibility
- Ease of use
- Integrations

Yet its major downside is vendor locking users without the ability to change the deployment environment easily.

On the other hand, MongoDB Atlas frees users to use any supported cloud provider to:

- Create MongoDB database clusters
- Move to an on-premise MongoDB database with minimal configurations

MongoDB Atlas offers a simple platform to provision and manage MongoDB clusters across multi-cloud environments yet lacks the tight integration of an inbuilt product like DynamoDB.

As a mature platform, MongoDB has an edge in its available feature set for managing the underlying data set with native schema validations, multiple index type support, etc. Additionally, you can configure it to cater to most database needs.

Refer to the official documentation of both MongoDB and DynamoDB as an excellent base to dive deeper into each database.

Which NoSQL database is right for you?

MongoDB and DynamoDB are both solid NoSQL databases to support various user needs. However, careful consideration is required when selecting the best option. In an AWS world, it's a no-brainer to go with the AWS native NoSQL solution if all the required features and functionality are available in DynamoDB.

On the other hand, MongoDB offers robust enterprise-ready NoSQL databases with a rich feature set that supports multi-vendor, multi-cloud deployments without vendor locking users with a proprietary solution.