mongoDB.

Modernization Toolkit Datasheet

Modernize from Legacy Databases to MongoDB

Legacy modernization is a strategic initiative that enables you to apply the latest innovations in development methodologies, architectural patterns, and technologies to refresh your portfolio of existing applications.

The best approach for migrating legacy systems to the cloud is to modernize from relational database management systems (RDBMS) to MongoDB instead of lifting and shifting. The Modernization Toolkit is a compilation of a Modernization Scorecard, recommended partner tooling, and CDC/delta load mechanisms.

Benefits of Modernizing to the Cloud

Cloud based services provide auto-elasticity, allowing customers to easily scale up for increased capacity and down when demand decreases. Cloud-native systems also guarantee high availability and higher levels of resilience for mission critical applications. Putting customer-accessed data in the cloud means you can distribute data globally to keep it close to your user base, minimizing latency. Fully managed cloud services also provide automatic software upgrades, advanced security configurations, and audit compliance.

Moving to the cloud bypasses the extensive costs of hosting your own servers on-prem, not only from the purely financial perspective of buying and running equipment, but also from the perspective of day-to-day operational costs. These “invisible” operational expenses include:

- Provisioning of servers
- Installation
- Configuration and ongoing monitoring to account for high availability, security, testing, monitoring and alerts, backups, and more
- Ongoing upgrades, and maintenance

Cloud infrastructure incurs small and regular operational expenses instead of large upfront capital expenses, removing lengthy procurement processes and reducing lead times to test and deploy new products on pay per use models.

The MongoDB Modernization Scorecard

MongoDB's Modernization Scorecard helps you to evaluate the suitability of MongoDB for both new and existing applications. The scorecard scores MongoDB and other databases against multiple criteria in each of the following categories of application requirements:


Data modeling


Query requirements


Performance & scalability


Availability & disaster recovery


Operational management


Deployment model & TCO


Download the Modernization Scorecard here

Recommended Partner Tooling

We recommend the partner tools listed in our Modernization Toolkit. This partner technology supports the one-time movement of data from RDBMS into MongoDB at scale, as well as continuous data movement, including change data capture, reconciliation, and validation procedures. We anticipate that both our customers and our system integrator partners, who help our customers with digital transformation and migrations to the cloud, will use these tools extensively.

CDC / Delta Load Mechanisms

Migration typically starts with an initial load, copying data from the source system into the target MongoDB cluster. This is often followed by an ongoing delta load for some period of time: as long as data is changing on the source system, those deltas are also loaded into the target MongoDB. There are two ways of handling the ongoing delta load.

- **Option 1:** Change Data Capture (CDC) tools can track changes on the source RDBMS database, convert them into event streams, and sync them to the target MongoDB database on an ongoing basis.
- **Option 2:** Leverage existing systems/applications with additional implementation in the service layer to invoke the new data access layer in parallel, which injects the data into MongoDB.

Delta loading and CDC are particularly important because they enable different approaches to modernizing. They can be used briefly after the initial load, only to catch up to changes before a full cutover. Alternatively, you might choose to run both systems in parallel for a period: for example, for a more gradual phased transition, a separation of reads and writes between systems, or a split of user base across the systems. In this case, CDC is absolutely necessary to keep the systems in sync. This pattern can also be used for a transition from monolith to microservices, where a monolithic source system needs to stay up and running for a period of time while it's gradually decomposed into new microservices running on MongoDB; ultimately the monolithic source system can be retired.

Contact partners@mongodb.com for additional information.

mongoDB.