Use Case Guidance:

Selecting the best applications for MongoDB, and when to evaluate alternative options

---

July 2021

mongoDB.

Table of Contents

Introduction

Data and software are today at the heart of every business, but for many organizations, realizing the full potential of the digital economy remains a significant challenge. Since the inception of MongoDB, we believe the biggest challenge developers face comes from working with data:

- Demands for higher productivity and faster time to market – where release cycles are compressed to days and weeks – are being held back by rigid relational data models that impose complex interdependencies between developers, DBAs, and operations teams.
- An inability to capture, work with, and extract insights from the massive increases in new and rapidly changing data types – structured, semi-structured, and polymorphic – generated by today's applications & systems.
- Monolithic and fragile legacy application data architectures that inhibit the wholesale shift to distributed systems and cloud computing. These new computing models provide the business with the resilience and scale demanded by modern applications serving global audiences, while also meeting a whole new set of regulatory demands for data privacy.

For these reasons, non-tabular (sometimes called NoSQL or non-relational) databases have seen rapid adoption over the past decade. But many of these NoSQL databases are simply bandaids, offering a niche set of functionality.

The problem is that typical NoSQL databases do one or two things well – they might offer more flexibility in the data model than traditional databases, or scale-out easily. But to do this they discard the most valuable features of relational databases. They often sacrifice data integrity and the ability to work with data in the ways needed to build rich and valuable applications – whether these are new digital touchpoints with an organizations' customers, or modernized core backend business processes.

MongoDB was launched in 2009 as a completely new approach to help developers work with data, and quickly established itself as the database most wanted by developers. This is because MongoDB takes the best of both relational and NoSQL databases. It is built around an intuitive data model and powerful, idiomatic query API to serve almost any class of application. It offers distributed transactions at global scale, next generation privacy and security controls, a flexible schema, and unique geo-aware data distribution.

Over the past decade, MongoDB has expanded beyond its core database technology into the industry's first **application data platform**. MongoDB's application data platform provides developers a unified interface to serve transactional plus search, real-time and data lake applications needs. This enables developers to move fast and simplify how they build with data for almost any class of application.

Development teams often get started with MongoDB in tactical projects, using the experiences they gain as a proving ground for a new class of data platform. As they start to realize the value MongoDB offers, many ask where else they can apply the technology to other projects in the

business. This white paper guides you to applicable use cases for MongoDB, and those workloads where you should evaluate alternative solutions.

When Should I Use MongoDB?

Through its design, MongoDB provides a technology foundation that enables customers to meet a broad set of application requirements. Three core architectural principles underpin MongoDB:

1. The **document data model**, giving developers the easiest and most intuitive way to work with data.
2. The **MongoDB Query API**, giving developers the fastest way to innovate in building transactional, operational and analytical applications.
3. Delivered as a **multi-cloud, global platform**, giving developers the freedom to run their applications anywhere with the flexibility to move data across private and public clouds as requirements evolve – without having to change a single line of code.

All of this comes together as the **MongoDB Application Data Platform**, providing a unified developer experience for modern applications that span cloud to edge.

MongoDB Database

The MongoDB database is general purpose, providing the most useful features of both relational and NoSQL systems:

- Wherever you are thinking about using a relational database, you should consider MongoDB.
- Wherever you are thinking about using a NoSQL database, you should consider MongoDB.

Whether you plan to run your applications as a serverless, cloud-native solution; in your own facilities; or a hybrid deployment model in between, MongoDB provides complete infrastructure agility.

The best way to run the MongoDB database is in Atlas, our fully managed and fully automated global cloud service available in ~80 regions across AWS, Azure, and GCP. Alternatively you can manage MongoDB yourself on your own infrastructure with MongoDB Enterprise Advanced providing a complete range of tooling and integrations to build your own private cloud.

MongoDB Atlas Search

It's incredibly important for modern applications to deliver fast, relevant search functionality: it powers discoverability and personalization of content, which in turn drives user engagement and retention.

Atlas Search exposes a fully-managed Apache Lucene index directly on top of the MongoDB database, enabling developers to create rich, relevance-based search experiences without

having to move their data into a separate search engine. Atlas Search offers advanced full-text search features including auto-complete, typo tolerance, function scoring, and synonyms – all fully integrated into the MongoDB API so developers don't need to context switch to another query language.

MongoDB Atlas Data Lake

MongoDB Atlas Data Lake extends the power and productivity of MongoDB to data beyond the database. Cloud object storage, in particular AWS S3, has emerged as the new standard for durable, long-term, cost-effective data storage. Gaining valuable insights quickly from data stored in cloud storage, and combining it with live, operational data is a strategic goal for most organizations today but is increasingly difficult to achieve at modern scale. With Atlas Data Lake you can query, combine, and analyze data across AWS S3 and MongoDB Atlas Databases without complex integrations, working with data in its native format using the MongoDB Query API.

The cost benefits of cloud object storage data is highlighted in data tiering scenarios. Atlas Online Archive (which is powered by Atlas Data Lake), let's developers automatically archive historical data from the MongoDB database to a fully-managed, queryable Atlas Data Lake. This enables savings on transactional database storage costs **without** compromising on data accessibility. Using a single, federated query endpoint you can combine your live and historical data without data movement, and you can work with complex data assets stored in multiple formats immediately, without the need for data transformation.

By leveraging the serverless, scalable Atlas Data Lake query service you control costs and remove the operational burden of moving data to unlock value from all of your data faster.

MongoDB Realm

MongoDB Realm empowers mobile developers to quickly build highly performant, cross-platform apps that sync in real-time and work even when offline.

- **Realm Mobile Database:** Allows developers to store data locally on iOS and Android devices using a rich data model that's intuitive to them. Combined with the MongoDB Realm sync-to-Atlas, Realm makes it simple to build reactive, reliable apps that work even when users are offline.
- **Realm App Services:** Allow developers to validate and build key features quickly. Application development services like Realm Sync for mobile and Realm's GraphQL service, can be used with Realm Functions, Triggers, and Data Access Rules – all simplifying the code required to build secure and performant apps.

All of the components of MongoDB's application data platform are designed to accelerate developer productivity and eliminate the tax on innovation that comes with sprawling data infrastructures. We believe it provides the best approach with a unified interface that can support any type of application workload, without limiting you to a single deployment environment.

Key Strategic Initiatives Supported by MongoDB

Finding new ways to compete in the digital economy involves much more than simply inserting new technology into your application stack. Fully harnessing the opportunities presented by digital requires bringing together **people, processes,** and **platform technologies** to support your organization's strategic initiatives. Underpinned by MongoDB's application data platform, we have developed a suite of solution stacks that provide:

- **Advisory consulting** to understand your strategic objectives and build a roadmap to deliver them.
- **Program governance** for delivery controls throughout a project.
- **Application lifecycle expertise** supported by design patterns and reference architectures, implementation best practices, and technical training.

From projects to modernize legacy apps, to moving to the cloud, exposing enterprise data as a service, or enabling business agility, MongoDB solutions can help you address your organization's most transformational strategic initiatives.

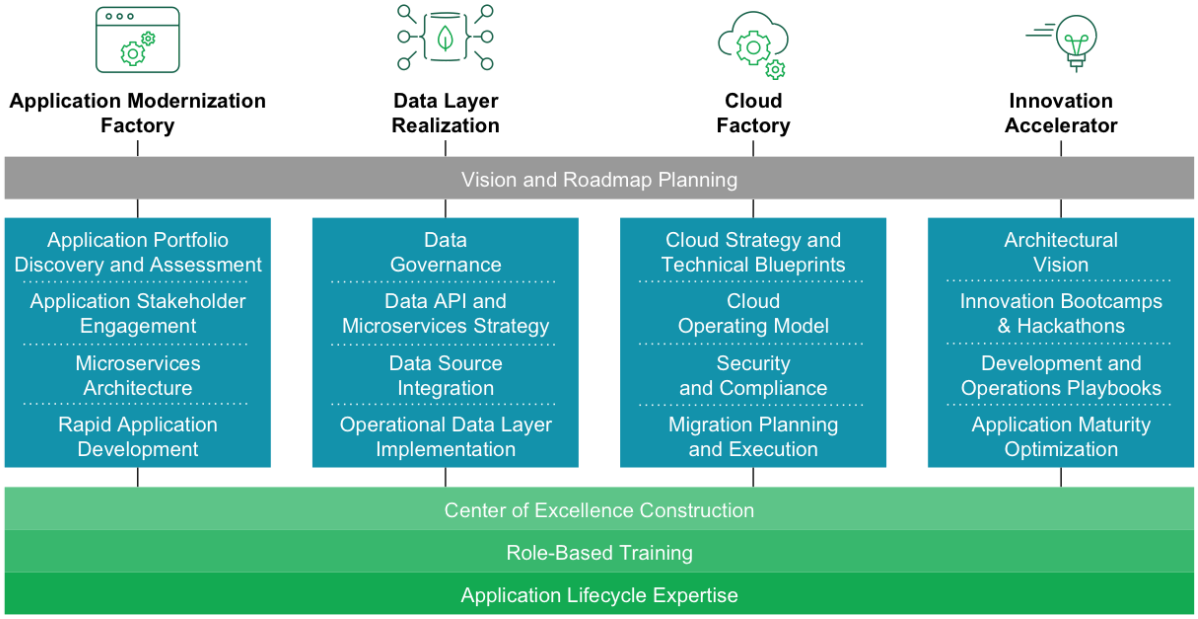

Figure 1: *MongoDB Professional Services solutions stacks*

Legacy Modernization

Legacy Modernization enables you to apply the latest innovations in development methodologies, architectural patterns, and technologies to refresh your portfolio of legacy applications.

We work with you to build an Application Modernization Factory (AMF), cultivated from best practices developed with some of the world's largest organizations. We partner with your teams to accelerate the assessment, prioritization, and redesign of legacy apps, quantifying the

economic value of change and providing a roadmap for delivery. We work with you through the modernization efforts of redevelopment, consolidation, and optimization, harnessing patterns and technologies such as agile and DevOps, microservices, cloud computing, and MongoDB best practices.

Review our Legacy Modernization overview to learn more, and download the Relational Database Migration Guide for best practices in schema design, application development, and data migration when moving from relational databases to MongoDB.

Cloud Data Strategy

Companies have long realized the agility and cost benefits of running on cloud infrastructure instead of maintaining their own data centers. However, a successful Cloud Data Strategy is much more than just using someone else's computer. To derive the agility and cost benefits that the cloud promises, you need a comprehensive approach that relies on using the right technologies and operational processes.

MongoDB Atlas is a cloud-native application data platform that can meet you wherever you are on your cloud journey. Harnessing Atlas, the MongoDB Cloud Factory helps your organization take a cloud-first stance on MongoDB application development:

- We collaborate with your teams to develop a Cloud Operating Model, serving as a foundation for both development of new applications in the cloud and migrating existing workloads.
- We analyze your application portfolio to rapidly and iteratively identify applications most suitable for running in the cloud and provide technical expertise and best practices throughout the application development lifecycle.

Review our Cloud Data Strategy overview to learn more

Data as a Service (DaaS)

DaaS is an investment in consolidating and organizing your enterprise data in one place, then making it available to serve new and existing digital initiatives. Data as a Service becomes a system of innovation, exposing data as a cross-enterprise asset. It unlocks data from legacy systems to drive new applications, without the need to disrupt existing backends. Typical use cases include customer single view, analytics and AI, and mainframe offloading.

The path to Data as a Service is to implement an Operational Data Layer (ODL). This data layer sits in front of legacy systems, enabling you to meet challenges that the existing platforms can't handle – without a full "rip and replace" of those existing systems.

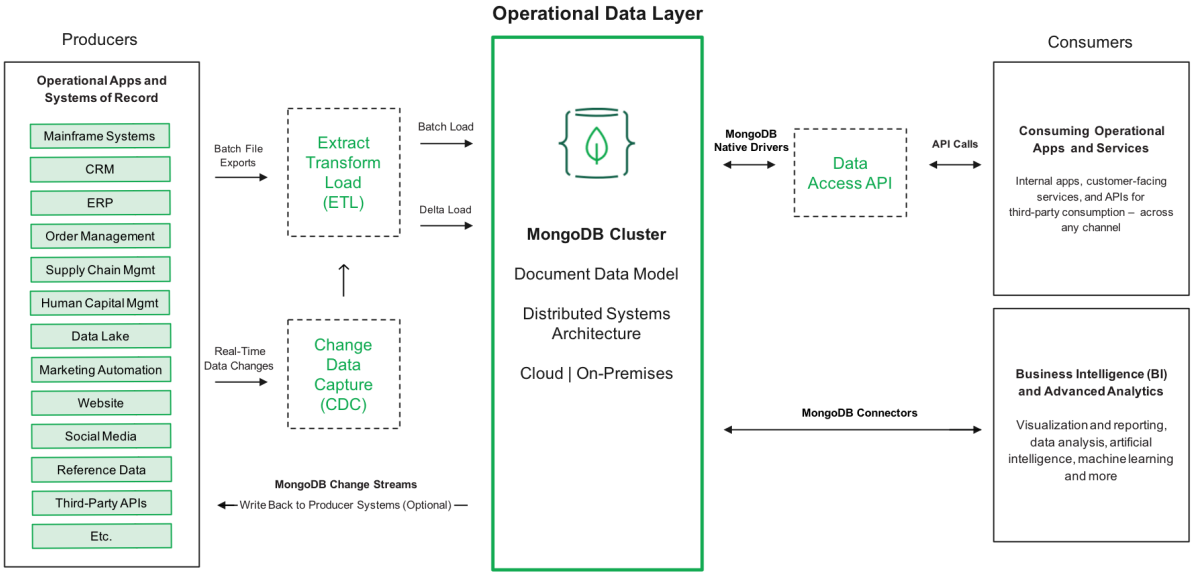

Figure 2: *Reference architecture for an Operational Data Layer*

MongoDB has developed a tried and tested approach to constructing an Operational Data Layer. The Data Layer Realization methodology helps you unlock the value of data stored in silos and legacy systems, driving rapid, iterative integration of data sources for new and consuming applications. Data Layer Realization offers the expert skills of MongoDB's consulting engineers, but also helps develop your own in-house capabilities, building deep technical expertise and best practices.

Review our Data as a Service overview to learn more, and download the ODL Reference Architecture for best practices in implementing an Operational Data Layer.

Business Agility with the MongoDB Innovation Accelerator

Organizations that are equipped to respond quickly to new opportunities, threats, and changing regulatory conditions will solidify their competitive positions and gain market share. You need to build new applications and functionality to address emerging and changing use cases better and faster than ever before.

MongoDB's Innovation Accelerator helps achieve widespread transformation of your development organization and processes to achieve faster time to market across multiple business applications and use cases. Our Professional Services team works with you to re-equip your development teams, supporting agile application development and continuous delivery of new functionality. The Innovation Accelerator:

- Establishes a roadmap for achieving business agility.
- Ensures rapid prototyping of solutions.
- Builds deep technical expertise and best practices across your development teams.
- Supports agile and iterative development, testing, and deployment.

Review our Business Agility overview to learn more.

Use Cases for MongoDB

Through the solutions discussed above, MongoDB serves a broad range of transactional and analytical applications. In each of the following use cases, we provide a definition of the application, along with required capabilities and how MongoDB can help. Review the MongoDB Architecture Guide for more detail on specific MongoDB features and capabilities applicable across all of these use cases.

Single View

A Single View, sometimes called Customer 360 or a data hub, aggregates data from multiple source systems into a central repository to create a single view of a business entity. By creating a single, real-time view, organizations enhance business visibility and enable new classes of analytics to better serve their customers and improve oversight of key resources.

While the most common use case is building a single view of your customers, the same approach can be applied to create a single view of supply chains, financial asset classes, products, and more.

Required Capabilities	Why MongoDB?
Data model • Ingest data of any structure from source systems • Dynamically adapt as a source system's schema changes	**Data model** • Document data model to store richly structured data from any source • Flexible and dynamic schema that eliminates migrations whenever source systems are updated • MongoDB Connector for Apache Kafka to synchronize data changes in real time to and from source systems and the single view
Query model • Multiple access patterns and query types • Simple lookups through to search and sophisticated analytics for business insight and personalization	**Query model** • Expressive query API, secondary indexing, aggregation pipeline, Lucene-based full-text search, and on-demand materialized views • Federated queries across your Atlas database and Atlas Data Lake to access customer data wherever it is stored • Realm GraphQL service to build and expose APIs to apps consuming the single view • Data visualization: MongoDB Charts and BI Connector

	• AI: Spark Connector, idiomatic R and Python drivers
Grow and protect • Scale as new data sources are on-boarded • Robust security and data sovereignty controls	**Grow and protect** • Distributed systems architecture with native sharding for scale out • Comprehensive access controls and auditing, client-side encryption protecting the most sensitive PII • Global clusters to control data sovereignty for regulatory compliance
Design and implement • Governance processes to build and maintain the single view	**Design and implement** • 10-Step methodology to creating a single view

Table 1: *Required capabilities for single view use cases*

Customer example: As the UK's fastest growing electronics retailer, AO.com was challenged to maintain business expansion without losing touch with its customers. It turned to MongoDB, to build a single customer view, delivering the project in just 3 months. With its single view of the customer, the business has been able to reduce call handling times by 40%, cut fraud processing from hours to seconds, and enabled its legal and marketing teams to better support new GDPR requirements.

Beyond technology, organizations need to implement the business processes needed to deliver and maintain a single view. Built on experiences gained over many years working with organizations of all sizes and industries, we have developed a 10-Step Single View Methodology to share best practices.

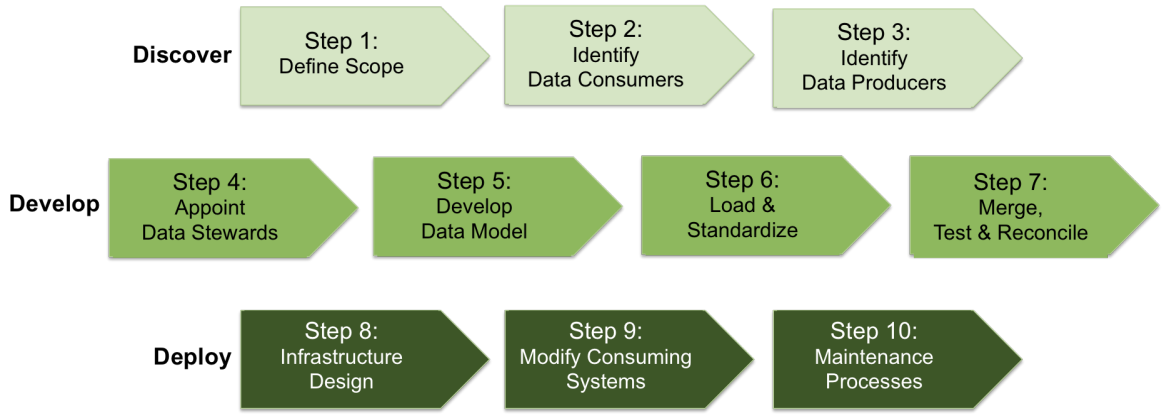

Figure 3: *10-Step methodology to building a single view with MongoDB*

The MongoDB 10-step methodology provides a step-by-step guide to each stage of the single view project – from discovery to data loading and reconciliation, through to deployment, along

with the governance and tools essential to successfully delivering a single view app with MongoDB.

Customer Data Management and Personalization

The single view use case is an example of customer data management. In this context, customer data also applies to data held on employees, citizens, patients, students, partners, through to complete corporate entities. This data is used by both backend Line of Business software such as CRM, HR, billing, and order processing, as well as front-end apps such as eCommerce user profiles and customer self-service portals.

The same set of required capabilities defined earlier for a single view applies equally to customer data management, especially in personalizing a user's experience as they interact with your apps and digital properties. To stand out from competitors, you need to offer contextual and engaging experiences that are personalized for each user – reflecting their interests and providing recommendations to them, in real time and on any device.

Customer example: Expedia is a virtual concierge – it knows if you want family-friendly holidays, business travel, or a once-in-a-lifetime break. Using MongoDB, it pushes special travel offers to users in real time by analyzing their searches and comparisons across its site.

Managing any type of personally identifiable information makes privacy a key requirement for your customer data management apps. The European Union's GDPR (General Data Protection Regulation) and the California Consumer Privacy Act (CCPA) has ushered in a new wave of global privacy regulations that govern how organizations like yours collect, store, process, retain, and share the personal data of citizens. You can learn more about how MongoDB can support your compliance initiatives by downloading our Global Data Privacy guide.

Industry-first innovations like MongoDB's Client-Side Field Level Encryption also make it much easier to meet modern privacy regulations. When a user invokes their right to be forgotten, simply destroy their encryption key and their associated PII is rendered unreadable and irrecoverable across your databases, backups, and logs.

Internet of Things (IoT) and Time-Series Data

Today, the IoT is enabling companies to blend the physical with the digital worlds. The business value of connecting all of these "things" is realized through the creation of new revenue models, improved productivity, and the ability to generate new insights that drive operational efficiencies. The IoT already connects billions of devices worldwide, and that number is growing every day.

IoT is one example of an application with time-series data at its core. Other time-series use cases include financial trading systems, clickstreams, and asset tracking. What's common about time series applications is that they place very specific demands on data ingestion, storage, and access that are difficult to meet with general-purpose databases. It is for this

reason that MongoDB has introduced a whole suite of native time series optimizations for its application data platform.

Time series collections, clustered indexing, and window functions make it easier, faster, and lower cost to build and run time series applications, and to enrich your enterprise data with time series measurements. MongoDB automatically optimizes your schema for high storage efficiency, low latency queries, and real time analytics against temporal data.

Running your time series applications on our application data platform eliminates the time and the complexity of having to stitch together multiple technologies yourself. You can manage the entire time series data lifecycle in MongoDB – from ingest, storage, querying, real-time analysis, and visualization through to online archiving or automatic expiration as data ages – all through a single API within a single platform.

Time series collections can sit right alongside regular collections in your MongoDB database, making it really easy to combine time series data with your enterprise data within a single versatile, flexible database – using a single query API to power almost any class of workload.

Required Capabilities	Why MongoDB?
Data model • Efficiently ingest and store vast streams of high velocity time series data • Accommodate complex and quickly changing time-series data generated by sources such as heterogeneous sensors, connected devices, log files, and financial stock tickers	**Data model** • Native time series collections automatically store your data in an optimized and compressed format for low storage consumption and reduced I/O, boosting performance and scale • Flexible data model makes it easy to ingest new data supporting expanded app functionality without expensive schema migrations
Analytics • Concurrent real-time analysis against live, operational data, without the latency of moving it out to separate systems • Discover valuable insights with analytics and machine learning.	**Analytics** • Clustered indexes automatically created against time series collections allowing fast, low latency access • Window functions to run analytical queries like moving averages and cumulative sums that uncover hidden patterns • Workload isolation to separate data ingest from analytics processes running on the same physical database cluster • Federated queries across your Atlas database and Atlas Data Lake to analyze time-series data wherever it stored • Visualization: MongoDB Charts and BI Connector

	• AI: Spark Connector, idiomatic R and Python drivers
Scale, Resilience, and Data Tiering • Ingest high volumes of sensor and event data from geographically distributed assets at low latency • Always-on • Cost-effectively store and access all of your time-series data, irrespective of its age	**Scale, Resilience, and Data Tiering** • Distributed systems architecture with native sharding that brings database nodes close to your data sources • Replica sets and global, multi-cloud clusters to distribute data across multiple regions, withstanding data center failures • Atlas Online Archive to automatically tier aged time-series data out of your database into fully managed cloud object storage for lower cost, while retaining the ability to query and analyze it

Table 2: *Required capabilities for IoT and time-series use cases*

IoT customer example: Bosch has built its Internet of Things suite on MongoDB, bringing the power of big data to a new range of industrial and consumer IoT applications including manufacturing, automotive, retail, energy and many others. Learn more from the case study.

Time-series customer example: Man AHL's Arctic application uses MongoDB to store high frequency financial services market data, handling 250M ticks per second. The hedge fund manager's quantitative researchers ("quants") use MongoDB to research, construct, and deploy new trading models in order to understand how markets behave. With MongoDB, Man AHL realized a 40x cost saving when compared to an existing proprietary database. In addition to cost savings, they were able to increase processing performance by 25x over the previous solution.

Product Catalogs and Content Management

With online and mobile sales volumes growing at around 50% every year – growth that has been accelerated by the Coronavirus pandemic – it is vital for companies to scale their eCommerce product catalogs to meet explosive demand and a tougher competitive market.

At the same time, the increasing ubiquity of high speed internet connectivity and smart mobile devices is changing the content management landscape. Sites have to create engaging, relevant, and immersive experiences enlivened with rich media assets and user generated content, all of which have to be delivered with low latency to any device, anywhere on the planet.

While most catalog use cases are centered on ecommerce product catalogs, other examples include asset catalogs used for internal inventory management and trade catalogs used in financial services.

Content management use cases include websites, online publications, research and training materials, document management, and open data repositories. Many content management systems are custom-built, but there are also packaged platforms such as Adobe Experience Manager and Sitecore, both of which use MongoDB for data management.

Required Capabilities	Why MongoDB?
Data model • Handle massive variability in content and catalog attributes, metadata, media assets, and UGC • Quickly update the data model as new products and content are offered	**Data model** • Document data model to store rich, multi-structured data • Flexible schema that instantly adapts with no schema migrations
User experience • Rich queries and search as users browse the product catalog and CMS • Never slow down under the peak loads generated by promotions, seasonal spikes, or new publications • Never go down, always available	**User experience** • Expressive MongoDB Query API and Atlas Search based on Lucene powers discoverability and personalization of content • Native sharding for horizontal and elastic scale in response to customer demand • Global Clusters to colocate data close to users for low latency, and replication for platform resilience
Real time analytics • Serve up personalized recommendations • Monitor sales performance and content consumption in real time	**Real time analytics** • Enabled by the aggregation pipeline • AI: Spark Connector, idiomatic R and Python drivers • Dashboards and reporting: MongoDB Charts and BI Connector

Table 3: *Required capabilities for product catalogs and content management*

Product catalog customer example: As a top 10 global retail brand with 1.4 billion active listings across 190 markets around the world, eBay cannot afford systems downtime. This is why the company relies on MongoDB as one of its core enterprise data platform standards, powering multiple, customer-facing applications that run ebay.com. The company's product catalog is distributed and scaled on a 50-node MongoDB replica set, spread across multiple data centers.

Content management customer example: Around $4 trillion is invested globally every year in medical and scientific research. Elsevier publishes 17% of the content and discoveries generated from that research. MongoDB is at the core of the Elsevier cloud-based platform, enabling the company to apply software and analytics that turns content into actionable knowledge and new insights for its customers.

Payment Processing

With payment processing moving to web and mobile channels, organizations are seeking to modernize existing backend transactional systems to deliver the availability and scale needed to reliably serve more customers. They have to do this without giving up the strong data integrity guarantees they have come to expect from traditional relational databases.

Unlike most modern, distributed databases, MongoDB supports multi-document ACID transactions. Through snapshot isolation, transactions provide a consistent view of data, and enforce all-or-nothing execution to maintain data integrity. Transactions in MongoDB feel just like transactions developers are familiar with from relational databases, making them simple to add to any application that needs them. Unlike relational databases, MongoDB's transactions can operate against highly scaled-out sharded clusters. Learn more by reading our Multi-Document ACID Transactions whitepaper.

Required Capabilities	Why MongoDB?
Data integrity • Handle payment and order processing with data correctness guarantees	**Data integrity** • Multi-document ACID transactions with snapshot isolation and all-or-nothing execution
Data model • Support variability in payment data and financial instruments • High precision number, temporal, and geospatial data types for lossless processing, sorting and comparisons	**Data model** • Flexible document data model to store rich, multi-structured data • Bank to Bank interchange standards (SWIFT, Maestro) now define JSON implementations • Advanced BSON data types including decimal, datetime, and GeoJSON data
Analytics • Fraud detection, risk profile and liquidity monitoring • Monitor sales performance in real time • Cost-effectively store and access all of your payments data, irrespective of its age, to maintain regulatory compliance	**Analytics** • Aggregation pipeline with Window Functions to analyze payments over time. • Workload Isolation, enabling translytics on the payment platform • Atlas Online Archive to tier aged payments data out to low cost object storage, maintaining regulatory compliance for data retention • Federated queries across your Atlas database and Atlas Data Lake to analyze payments data wherever it stored • AI: Spark Connector, R & Python drivers • Dashboards: MongoDB Charts and BI Connector, with data exposed to 3rd parties via merchant portals

Resilience & Latency • Maintain availability in the face of outages and planned maintenance	**Resilience & Latency** • MongoDB replica sets and Global Clusters to distribute data across multiple regions
Data Integration • Integrate eCommerce apps with payment providers • Push payment and fraud alerts to consuming systems	**Data Integration** • MongoDB Atlas triggers and functions to call 3rd party payment provides • MongoDB Connector for Apache Kafka, change streams and Atlas triggers for event-driven pipelines

Table 4: *Required capabilities for payment processing*

Payment processing use cases include eCommerce backends, financial trading systems, billing engines, and mobile payment gateways. Key examples of use cases built on MongoDB include the following.

Cisco migrated its eCommerce platform, handling over $40bn of revenue per annum, from a legacy relational database to MongoDB. As a result, customer experience is improved by reducing latency 5x and improving availability by 100x. The migration has driven developer productivity gains, eliminating 25+ business critical backlog features. Developers can build new applications faster, while the company's eCommerce platform can tap into the business agility enabled by cloud computing.

Macquarie Bank built its new cloud-based real-time payments platform on MongoDB Atlas. The bank modernized its payments systems by adopting a microservices architecture, along with agile and DevOps processes. *"We've been using MongoDB Atlas for 2 years for Macquarie's NPP solution, and to date it's been the easiest part of the solution, it just works."*

Download the 2021 Payments Analyst Report to learn how nearly 400 CFOs and senior banking executives are approaching the opportunities for payments data monetization.

Mobile Apps

The proliferation of devices following the mobile revolution has transformed how businesses operate with both their customers and with their internal workforces. With 4.5 billion smartphones and tablets in use and the increasing ubiquity of high performance WiFi and cellular networks, organizations that embrace "mobile-first" development are seeing new competitive opportunities.

Many companies are seeking to unlock greater worker productivity by equipping frontline workforces with mobile technology, using it to replace inefficient pen and paper processes or otherwise disconnected systems. Workers are able to more efficiently complete key tasks, and downstream operational teams are empowered to provide a better customer experience with a real-time view of work as it's completed in the field.

Other organizations have started their mobile customer journeys with mobile-optimized websites and refactored web apps such as customer portals and banking – enabling them to deliver services to customers across their preferred channels. Now new waves of mobile apps have emerged, including mobile payments, AI-driven lifestyle and retail experiences, Augmented Reality and gaming, personalized healthcare and fitness tracking, streaming services, and many more - all of which must meet increasingly high customer expectations around performance and functionality.

The MongoDB data platform underpins your mobile apps, from the device through to the backend:

- Realm Mobile Database – Radically simplifies development and improves user experience. Store data locally on your iOS and Android devices using a flexible data model, and easily build apps that run fast and work even when users are offline.
- MongoDB Realm – The best way for your applications and users to access the data and services they need. Realm connects your frontend to MongoDB's data platform in the cloud, providing secure access to data hosted in MongoDB Atlas. You can quickly create rich, secure apps and services without app servers, web hosts, or gateways. With minimal code, easily keep data in sync as it changes across your mobile devices, users, and Atlas.
- MongoDB Atlas – on-demand, elastic, and fully managed global cloud database backend, with baked-in best practices, leaving developers free to concentrate on their apps.

Required Capabilities	Why MongoDB?
Developer velocity • Rapidly build and evolve mobile apps • Create rich app experiences • Feature toggles to incrementally deploy new app functionality	**Developer velocity** • Simplified app architecture reduces complex code • Object and document data structures match objects and JSON syntax used in code • Simple SDKs to access data stored on-device and in the backend • Expressive query language to work on data on the device and in the backend • Flexible data model to support multiple app feature sets
Data portability and security • Sync data between device and backend • Access controls to data • Same mobile database for Android, iOS and IoT	**Data portability and security** • Realm Sync to automatically propagate data between user devices and Atlas • MongoDB change streams and Atlas triggers for reactive event driven pipelines • Realm Data Access Rules to control data visibility

Resilience and scale	Resilience and scale
• Local data available offline • Never go down, always available • Never slow down under the peak loads generated by new app launches	• Realm Mobile Database for local data persistence, enabling offline-first app • Global Clusters to colocate data close to users and replication for resilience • Realm serverless platform: auto-scale • Native sharding for horizontal and elastic scale

Table 5: *Required capabilities for mobile apps*

Consumer app example: 7-Eleven uses MongoDB to expand customer engagement into mobile commerce. Customers can browse a local store's inventory, and then click and collect the products they want to buy, all via their mobile app. As well as handling the transactional needs of the app, MongoDB also layers the analytics capabilities that allow 7-Eleven to further improve customer experience.

Internal app example: Cox Automotive subsidiary RideKleen used the Realm Mobile Database and MongoDB Realm to build a mobile application used by workers to handle all "digital paperwork" when delivering services in the field. A separate web portal allows operational teams to update work requests and immediately push changes to mobile apps as needed - all made possible via data sync to a shared Atlas backend.

Mainframe Offload

Despite its long predicted demise, the mainframe remains a critical IT asset in many large enterprises. But ongoing reliance on the mainframe does not come without challenges. Web, mobile, social, Artificial Intelligence, and Internet of Things applications are driving a deluge of new data. The volume, speed, and diversity of this data is overwhelming mainframe environments. Coupled with pressures to meet new regulatory demands and cut costs, CIOs are challenged in how quickly they can remake the business for digital, while trying to innovate on top of legacy technologies.

Mainframe offloading is the process of replicating commonly accessed mainframe data to an Operational Data Layer (ODL) built on MongoDB, against which operations are redirected from consuming applications. The existing mainframe is left untouched. By offloading mainframe operations to MongoDB, organizations can drive faster innovation, improved customer experience, and reduced cost.

Required Capabilities	Why MongoDB?
Data model • Ingest system of record data from the mainframe, then enrich it with data of any structure from source systems • Dynamically adapt as a source system's schema changes	**Data model** • Document data model to store richly structured data from any source • Flexible and dynamic schema that eliminates schema migrations whenever source systems are updated
Query model • Multiple access patterns and query types over multiple channels • Simple lookups through to sophisticated analytics for business insight	**Query model** • Expressive query API, secondary indexing, aggregation pipeline, and on-demand materialized views • MongoDB Realm provides simple and secure API access from clients and microservices to your data • Data visualization: MongoDB Charts and BI Connector • AI: Spark Connector, idiomatic R and Python drivers
Synchronize • Ensure data is available and consistent across mainframe and operational apps	**Synchronize** • Any data changes can be automatically propagated between the mainframe and ODL as they happen, using the MongoDB Connector for Apache Kafka
Grow and protect • Scale as new data sources are on-boarded • Robust security and data sovereignty controls • Resilient, always-on	**Grow and protect** • Distributed systems architecture with native sharding for scale out • Comprehensive access controls, encryption down to the level of individual fields, and auditing • Global clusters to control data residence

Table 6: *Required capabilities for mainframe offload use cases*

Customer example: Alight Solutions (formerly AON Hewitt) offloaded its Human Capital Services from the mainframe to MongoDB, improving application performance by 250x, reducing costs, and creating a platform for innovation. Review the Mainframe Offload Reference Architecture to learn more about why to offload, common approaches and patterns, enabling technologies, program execution, and governance.

Operational Analytics

Every organization is striving to be data-driven. But it isn't only the data itself that is valuable – it is the insight it generates. That insight can help designers better predict new products that customers will love. It can help manufacturing companies model failures in critical components,

before costly recalls. It can help financial institutions detect fraud and retailers better forecast supply and demand, while eCommerce businesses can build recommendation engines that make more informed and relevant suggestions to customers. The list goes on.

How quickly an organization can unlock and act on that insight is becoming a major source of competitive advantage. Collecting data in transactional systems and then relying on nightly batch ETL (Extract Transform Load) processes to update the Enterprise Data Warehouse is no longer sufficient. Speed-to-insight is critical, and so analytics against live operational data to drive real-time action is fast becoming a necessity.

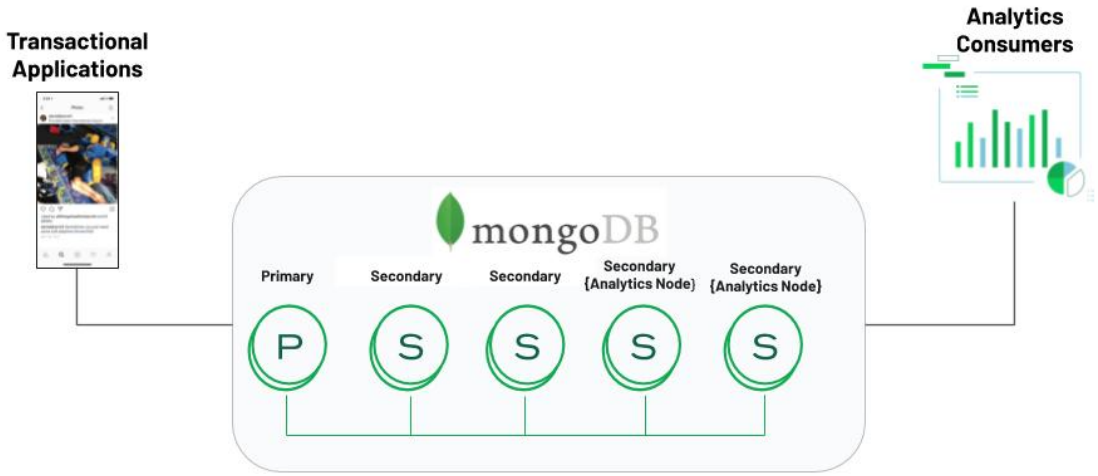

Figure 4: *MongoDB's distributed systems architecture enables the isolation of transactional and real-time operational analytical workloads on a single cluster, providing faster and fresher insights against your data*

Required Capabilities	Why MongoDB?
Data model • Handle data of any structure from operational and transactional systems • Dynamically adapt the schema during machine model training	**Data model** • Document data model to store rich, multi-structured data • Flexible schema with no schema migrations
Analytics Foundation • Multiple access patterns and query types • Simple lookups through to sophisticated data transformations and aggregations • Eliminate fragile and lengthy ETL • Tier and query your data anywhere	**Analytics toolkit** • Expressive query API, secondary indexing, Lucene-based Atlas Search, aggregation pipeline, and on-demand materialized views • Distributed architecture to enable workload isolation • Auto-archival of historical data from your transactional systems to long-term storage • Federated queries across your Atlas database and Atlas Data Lake to analyze

	data in any format and build models wherever data is stored • Persist results of long running queries from your data lake to database to support intelligent transactional applications
Integrated with data analytics ecosystem • Visualize data of any structure • Leverage investments in existing reporting and AI frameworks	**Integrated with data analytics ecosystem** • Create and embed data visualizations with MongoDB Charts • SQL integration for BI tooling (e.g., Tableau, Power BI) viathe BI Connector • Machine Learning / AI: Spark Connector, idiomatic R and Python drivers
Grow and protect • Scale as new data sources are analyzed • Robust security and data sovereignty controls	**Grow and protect** • Native sharding and data tiering for scaling across high data volumes • Comprehensive access controls, encryption, and auditing to protect sensitive data • Global clusters to control residence of personal data

Table 7: *Required capabilities for operational analytics and AI use cases*

Download our whitepaper to learn more about delivering enterprise AI and ML/Ops at scale.

Real-Time Analytics

Modern application experiences require real-time analytics. This is an emerging category of use cases where data from different systems is being used to drive application workflows in real-time. Real-time analytics enables companies to build the next generation of intelligent applications that reach beyond transactional processing and deliver real-time, insight-driven interactions — in milliseconds, not minutes. For example, our customer Longbow Advantage uses real-time manufacturing data to help clients manage shipments. Another customer (Claas) tracks farming equipment to know when a failure occurs and proactively troubleshoots the failure & sets up a maintenance call.

The need for real-time applications is increasing as user's expectations shift to want better experiences and quicker responses from the companies they work with. By combining data from real time events with historic & reference data sets, queries can be optimized for high performance to quickly deliver actionable results for better insight and better customer engagement.

Required Capabilities	Why MongoDB?

Data model ● Handle data of any structure and type from operational and transactional systems ● Dynamically adapt the schema during machine model training ● Flexibility to accommodate new data requirements as the application evolves	**Data model** ● Document data model to store rich, multi-structured data ● Flexible schema allow capturing and aggregating multi-model data into JSON-like documents
Data ingestion ● Ingest data from batch and streaming data sources ● Capture data from multiple sources	**Data ingestion** ● Atlas Data Lake allows for ingesting data from cloud object storage (e.g., AWS S3) ● MongoDB Connector for Apache Kafka allows for ingesting data from streaming sources
Data analysis & processing ● Combine and aggregate multiple data types in a single, refined data set ● Enrich and analyze data in real time	**Data analysis & processing** ● Maintains full transaction integrity and data consistency with ACID compliance as data changes in real time ● Aggregation pipelines allow you to do in-database data preparation, which gets data ready for further analytics
Data consumption ● Insights delivered to apps in real time ● High performance	**Data consumption** ● Low latency queries & responses ● Expressive, native query language (MongoDB Query API) and idiomatic drivers (i.e., Java, Python, R) makes programming insights into apps easier ● Increase query speed by caching common result sets with Materialized Views

Is MongoDB Always the Right Solution?

As illustrated through this Guide, MongoDB serves many transactional and analytics use cases. But of course MongoDB will not be suitable for every workload. There may be specific requirements that are better met with alternative technologies, which we discuss below.

Common Off-the-Shelf Software Built for Relational Databases

While many organizations have successfully migrated from relational databases to MongoDB, you cannot drop-in MongoDB as a replacement for ISV packaged applications that are built around the tabular relational data model and SQL. In these scenarios, it is better to work directly with the ISV to encourage them to support MongoDB as the data persistence layer with their application.

To learn how MongoDB stacks up against the most popular relational databases, check out our resources pages:

- MongoDB and Oracle compared.
- MongoDB and Postgres compared.
- MongoDB and MySQL compared.

Data Warehousing Use-Cases

It is important to understand the characteristics underpinning your analytics use-case to determine if MongoDB is the correct technology fit for your needs. For some, it will be, while for others more traditional data warehouses will be better suited.

First we will cover the analytics workload characteristics that are well aligned to MongoDB, and then discuss those that are better served by alternatives.

When is MongoDB suitable for your analytics environment?

Much of the data that organizations need to analyze is complex and richly structured – requiring many diverse attributes to model the business domain. As the digital economy continues to expand, growing volumes of rich, multi-structured data generated by modern applications are pushing traditional analytical environments beyond their design limits, forcing users to seek alternatives. MongoDB has proven to be that alternative in many projects across a variety of industries – from banking and retail, to manufacturing, utilities, and more.

The **first important criterion** to evaluate for MongoDB suitability is the data model. Consider financial instruments used in banking systems. Each instrument, whether it's a bond, loan, cash, derivative, equity swap, etc., can require hundreds of different attributes to describe and track the asset class. Alternatively product catalogs in retail systems must manage massive diversity in product descriptions, metadata, user-generated reviews, local currencies, and so on.

Traditionally these attributes have been normalized and shred across tens to hundreds of different tables in a relational database. These tables then undergo extensive transformation as they are extracted from transactional systems and loaded into a separate analytics platform to serve long-running and complex queries. To support multiple analytics query patterns, the data is often copied and transformed into multiple different table structures, each designed to service a distinct reporting requirement.

MongoDB's flexible document data model allows users to take **a different approach**. With subdocuments and arrays, users can create domain-driven data structures allowing the complete business entity to be modeled and enriched in a single document, rather than shredding it across separate records and tables. Doing this minimizes the need to JOIN multiple records together when the query is run, or to create multiple copies of the same data – each in its own distinct structure to service specific queries.

Coupling domain-driven data modeling with standard features of the MongoDB database enables you to power demanding analytics workloads at scale. Specific MongoDB features include:

- Powerful secondary indexes to optimize access patterns to the data.
- The aggregation pipeline to enrich, reshape, and filter data as needed for different query types.
- On-demand materialized views to pre-compute and cache query results.
- Users can easily visualize results of analytics queries with MongoDB Charts or with the BI Connector supporting all of the leading SQL-based BI tools.

One retailer was able to reduce the time taken to refresh BI views of their data from six minutes in their existing analytics environment to less than two seconds using the MongoDB features described above. The aggregation pipeline was used to filter out data that wasn't needed for the query results, and materialized views were used to pre-compute and store the result set for low latency access by multiple data consumers.

In addition to visualizations and reporting, you can also feed your MongoDB data into AI frameworks with the MongoDB Spark Connector, along with R and Python drivers.

Experience with existing customers across multiple industries proves that MongoDB is most suitable for complex analytics use-cases when:

1. Users can take advantage of document structures for domain-driven data modeling.
2. Query access patterns can be optimized with secondary indexes, and further accelerated with materialized views.
3. The ratio of hot/warm/cold data is split roughly as 20%/20%/60% of total data size.
4. The hot (i.e., most frequently accessed) data is up to 50TB.

When data sizes exceed the guidelines above, you can couple the MongoDB database with the MongoDB Atlas Data Lake to power analytics use-cases over larger data sets. Aged data can be automatically archived out of Atlas databases to the Atlas Data Lake. This offers the benefit of lowering storage costs for the aged data, which in turn needs to be balanced against higher latency when analytics queries operate against cloud object storage. With Federated Query, you

can submit a single query to your Atlas cluster and Atlas Data Lake S3 store, returning a single query result that blends data from both the database and the data lake.

MongoDB is proven on analytics use-cases matching the criteria listed above with data sizes reaching hundreds of TBs. These capabilities are recognised by industry analyst group Forrester when it studied platforms used to bring together previously separate transactional and analytical workloads – something it calls "Translytics". In its Forrester Wave™: Translytical Data Platforms, Q4 2019, it rated MongoDB as a Strong Performer in an evaluation that included more traditional vendors such as Oracle, IBM, Microsoft, and SAP.

When is it better to use a traditional data warehouse?

If your use-case prevents the use of domain-driven data modeling and queries are mainly SQL-based for BI & reporting output, then a traditional data warehouse can be a more optimal solution.

In this scenario, MongoDB Atlas can complement the data warehouse to store & serve query results that are generated from the data warehouse at low latency and high concurrency to data applications.

Conclusion

MongoDB is the most popular and widely used modern database in the market. Developers working with data touch more than the database, and need the best way to work with data across all their systems. We provide organizations with the industry's first application data platform that allows them to move fast and simplify how they build with data for any application.

MongoDB supports its platform-based approach through an array of enabling technologies, training, and professional services. Get in touch to learn more about MongoDB for modern apps, review use cases, and more

Safe Harbor
The development, release, and timing of any features or functionality described for our products remains at our sole discretion. This information is merely intended to outline our general product direction and it should not be relied on in making a purchasing decision nor is this a commitment, promise or legal obligation to deliver any material, code, or functionality.

mongoDB.

US 866-237-8815 • INTL +1-650-440-4474 • info@mongodb.com