Unlocking Operational Intelligence from the Data Lake

November 2017

Table of Contents

Introduction

The one thing no business lacks today is data – from streams of sensor readings, to social sentiment, to machine logs, mobile apps, and more. Analysts estimate data volumes growing at 40% per annum, with 90% of it unstructured. Uncovering new insights by collecting and analyzing this data carries the promise of competitive advantage and efficiency savings. However, the traditional Enterprise Data Warehouse (EDW) is straining under the load, overwhelmed by the sheer volume and variety of data pouring into the business, and then being able to store it in a cost-efficient way. As a result a number of organizations have turned to Hadoop as a centralized repository for this new data, creating what some call a "data lake".

In our data-driven world, milliseconds matter. In fact, research from IBM observed that 60% of data loses its value within milliseconds of generation. For example, what is the value in identifying a fraudulent transaction minutes after the trade was processed? For all the benefits a Hadoop-based data lake can bring to the business, it is not designed for real time access. Furthermore, Gartner analysts predict that 70% of Hadoop deployments will not meet cost savings and revenue generation objectives due to skills and integration challenges.

Being able to generate and serve analytics from the Hadoop data lake to online applications and users in real time can help address these challenges, demanding the integration of a highly scalable, highly flexible operational database layer. Industry leaders are using MongoDB as that database layer, uniting analytical and operational workloads to accelerate returns on their Hadoop investment by bringing greater context and intelligence to online applications.

The companies that win in the future will not be those that have the largest data lakes. Rather it will be those who are the fastest in acting on the insights and intelligence that data itself creates. Operational data platforms are essential to executing on the data lake vision.

The Hadoop Data Lake

With its ability to store data of any structure without a predefined schema and scale-out on commodity hardware, Hadoop provides levels of performance, efficiency and low Total Cost of Ownership (TCO) unmatched by the EDW.

The Hadoop Distributed File System (HDFS) is designed for large scale batch processing. Providing a write-once, read-many, append-only storage model for unindexed data stored in files of up 128MB, HDFS is best suited to long running, sequential scans across TBs and PBs of data. This makes Hadoop useful for mining large swathes of multi-structured data to create analytics that companies can use to better inform their business. Example outputs can include:

- Customer segmentation models for marketing campaigns and eCommerce recommendations.
- Predictive analytics for fleet maintenance and optimization.
- Risk modeling for security and fraud detection.

These types of models are typically built from Hadoop queries executed across the data lake with latencies in the range of minutes and hours. However, the Hadoop data lake is not designed to provide real-time access to this data from the operational applications that need to consume it.

Operationalizing the Hadoop Data Lake with MongoDB

Users need to make analytic outputs from Hadoop available to their online, operational apps. These applications have specific access demands that cannot be met by HDFS, including:

- Millisecond latency query responsiveness.
- Random access to indexed subsets of data.
- Supporting expressive ad-hoc queries and aggregations against the data, making operational applications smarter and contextual.
- Updating fast-changing data in real time as users interact with online applications, without having to rewrite, and then re-process, the entire data set.

Bringing together operational and analytical processing across high volumes of variably structured data in a single data platform requires capabilities unique to MongoDB:

- **Workload isolation.** MongoDB replica sets can be provisioned with dedicated analytic nodes. This allows users to simultaneously run real-time analytics and reporting queries against live data, without impacting nodes servicing the operational application, and avoiding lengthy ETL cycles.
- **Dynamic schema, coupled with data governance.** MongoDB's document data model makes it easy for users to store and combine data of any structure, without giving up sophisticated validation rules, data access and rich indexing functionality. If new attributes need to be added – for example enriching user profiles with geolocation data – the schema can be modified without application downtime, and without having to update all existing records.
- **Expressive queries.** The MongoDB query language enables developers to build applications that can query and analyze the data in multiple ways – by single keys, ranges, search with faceted navigation, graph transversals, and geospatial queries through to complex aggregations and MapReduce jobs, returning responses in milliseconds. Complex queries are executed natively in the database without having to use additional analytics frameworks or tools, and avoiding the latency that comes from moving data between operational and analytical engines.
- **Rich secondary indexes.** Providing fast filtering and access to data by any attribute, MongoDB supports compound, unique, array, partial, TTL, geospatial, sparse, and text indexes to optimize for multiple query patterns, data types and application requirements. Indexes are essential when operating across slices of the data, for example updating the churn analysis of a subset of high net worth customers, without having to scan all customer data.
- **Streaming data pipelines.** MongoDB change streams enable developers and data engineers to build reactive, real-time apps that can view, filter, and act on data changes as they occur in the database. Change streams enable seamless data movement across distributed database and analytics systems, making it simple to stream data changes and trigger analytics actions wherever they are needed, using a fully reactive programming style.

	MongoDB	Relational Database	Column-Oriented Datastore (i.e. HBase)
Isolate analytics from operational workloads	Yes	Yes	No
Filter Spark & Hadoop queries with the database's secondary indexes	Yes	Expensive add-on	No
Fully dynamic schema	Yes	No	No
Data validation rules	Yes	Yes	No
Expressive queries	Yes	Yes	No
Rich secondary indexes	Yes	Yes	No
Native BI Connectivity	Yes	Yes	3rd party connectors
Robust security controls, including encryption at rest	Yes	Expensive add-on	Partial. Distribution dependent
Scale-out on commodity hardware	Yes	No	Yes
Geographic distribution	Yes	Expensive add-on	No
Advanced management tooling	Yes	Yes	No
High skills availability	Yes	Yes	No

Table 1: How MongoDB stacks up for operational intelligence

- **BI & analytics integration.** The MongoDB Connector for BI enables industry leading analytical and visualization tools such as Tableau to efficiently access data stored in MongoDB using standard SQL. The MongoDB Connector for Apache Spark exposes all of Spark's libraries for analysis with machine learning, graph, streaming, and SQL APIs.
- **Robust security controls.** Extensive access controls, auditing for forensic analysis and encryption of data both in-flight and at-rest enables MongoDB to protect valuable information and meet the demands of big data workloads in regulated industries.
- **Scale-out on commodity hardware.** MongoDB can be scaled within and across geographically distributed data centers and cloud regions, providing extreme levels of availability and scalability. As your data lake grows, MongoDB scales easily alongside it with no downtime and no application changes.
- **Advanced management and cloud platform.** To reduce data lake TCO and risk of application downtime, MongoDB Ops Manager provides powerful tooling to automate database deployment, scaling, monitoring and alerting, and disaster recovery. Further simplifying operations, MongoDB Atlas delivers MongoDB as a service, providing the features of the database, without the operational heavy lifting required for any new application. MongoDB Atlas is available on-demand through a pay-as-you-go model and billed on an hourly basis.
- **High skills availability.** With availability of Hadoop skills cited by Gartner analysts as a top challenge, it is essential you choose an operational data platform with a large available talent pool. This enables you to find staff who can rapidly build differentiated big data applications. Across multiple measures, including DB Engines Rankings, The 451 Group NoSQL Skills Index and the Forrester Wave for Big Data NoSQL, MongoDB is the leading non-relational database.

In addition, the ability to apply the same distributed processing frameworks such as Apache Spark, MapReduce and Hive to data stored in both HDFS and MongoDB allows developers to converge analytics of both real time, rapidly changing data sets with the models created by batch Hadoop jobs. Through sophisticated connectors, Spark and Hadoop can pass queries as filters and take advantage of MongoDB's rich secondary indexes to extract and process only the range of data it needs – for example, retrieving all customers located in a specific geography. This is very different from typical NoSQL data stores that do not support a rich query language or secondary indexes. In these cases, Spark and Hadoop jobs are limited to extracting all data based on a simple primary key, even if only a subset of that data is required for the query. This means more data movement between the data lake and the database, more processing overhead, more hardware, and longer time-to-insight for the user.

As demonstrated in Table 1, operational intelligence requires a fully-featured data platform serving as a System of Record for online applications. These requirements exceed the capabilities of simple key-value or column-oriented datastores that are typically used for short lived, transient data, or legacy relational databases structured around rigid row and column table formats and scale-up architectures.

Figure 1 presents a design pattern for integrating MongoDB with a Hadoop data lake:

- Data streams are ingested to a pub/sub message queue, which routes all raw data into HDFS. Processed events that drive real-time actions, such as personalizing an offer to a user browsing a product page, or alarms for vehicle telemetry, are routed to MongoDB for immediate consumption by operational applications.
- Distributed processing frameworks such as Spark or MapReduce jobs materialize batch views from the raw data stored in the Hadoop data lake.
- MongoDB exposes these models to the operational processes, serving queries and updates against them with real-time responsiveness.
- The distributed processing frameworks can re-compute analytics models, against data stored in either HDFS or MongoDB, continuously flowing updates from the operational database to analytics views.

Customer Case Studies: Operationalizing the Hadoop Data Lake with MongoDB

The following examples demonstrate how leading companies are using the design pattern discussed above to operationalize their Hadoop data lakes.

Prescient

Traveler Safety Platform Analyzing Petabytes of Data with MongoDB, Hadoop, Apache NiFi & SAP HANA

Leading risk management company Prescient delivers real-time threat intelligence to corporate security stakeholders and individuals. It delivers concise, actionable recommendations to help travelers avoid danger and, if necessary, react smartly to it.

Prescient Traveler ingests massive volumes of structured and unstructured data – social media, breaking news, RSS feeds, real-time weather and geological alerts, public safety bulletins, economic stability indicators, as well as regional crime, health and natural disaster statistics – and uses advanced analytic systems to evaluate, visualize and disseminate relevant safety information. Text sources are analyzed by sophisticated software that determines sentiment, then identifies facts and events worth reporting to subscribers based on a variety of criteria.

The platform uses dozens of custom Apache NiFi processors for source metadata management and initial parsing of data feeds. From there, data is selectively routed to SAP HANA and Hadoop for complex analyses according to defined "escalation criteria." Following these text and geospatial analyses, Prescient's threat-vulnerability correlation process is completed when user profiles and locations persisted in MongoDB are queried to determine if threats relate to a specific person or population, based on their physical location and personal attributes.

You can learn more from the customer profile.

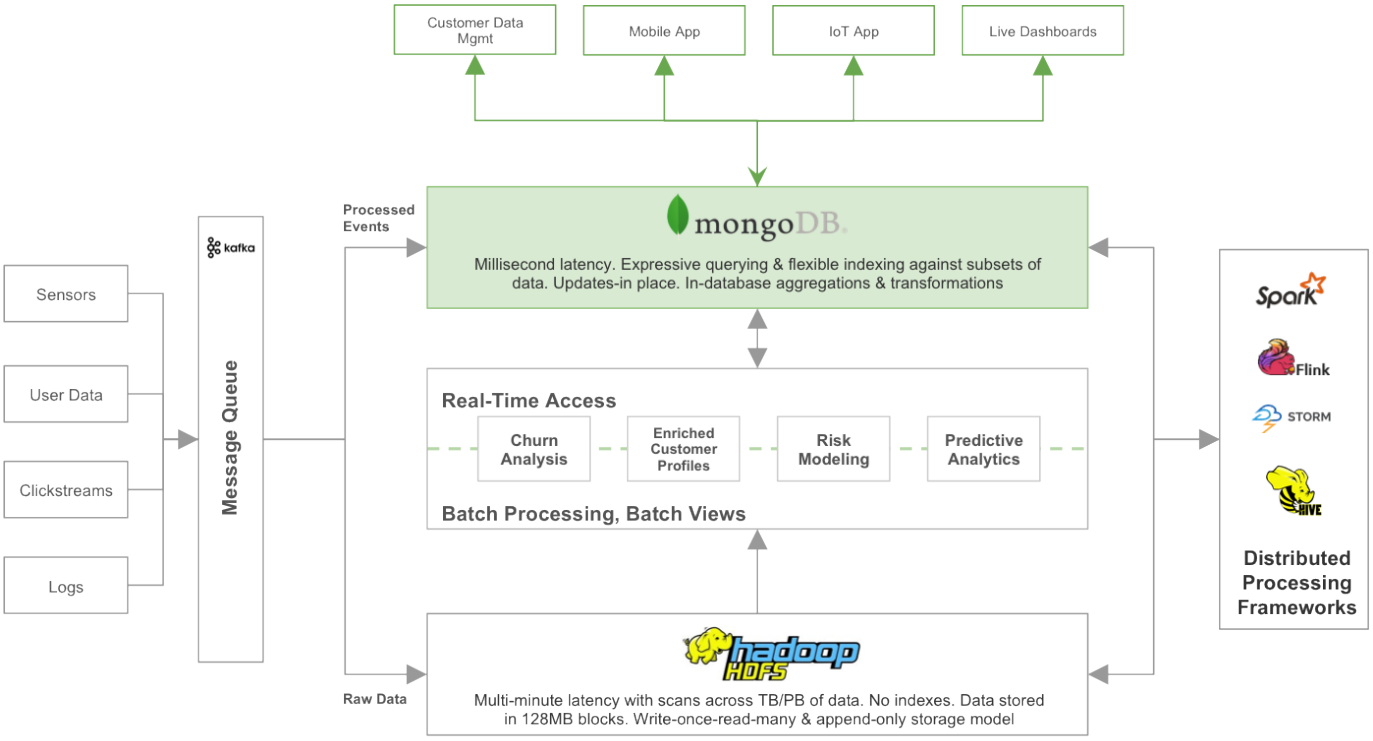

Figure 1: Design pattern for operationalizing the data lake

UK's Leading Price Comparison Site: comparethemarket.com

Out-Innovating Competitors with MongoDB, Hadoop, Microservices, Docker, and the Cloud The UK's leading price comparison provider, and one of the country's best known household brands has standardized on MongoDB Enterprise Advanced as the default operational database across its microservices architecture. The company's online comparison systems need to collect customer details efficiently and then securely submit them to a number of different providers. Once the insurers' systems respond, comparethemarket.com can aggregate and display prices for consumers. At the same time, MongoDB generates real-time analytics to personalize the customer experience across the company's web and mobile properties.

With the previous generation of systems, all application state was stored in the database, and then imported every 24 hours from backups into the company's data warehouse. But that approach presented several critical issues:

- No real-time insight as the analytics processes were working against aged data.
- Application changes broke the ETL pipeline.
- The management overhead increased as more applications were added and data volumes grew.

As comparethemarket.com transitioned to microservices, the data warehousing and analytics stacks were also modernized. While each microservice uses its own MongoDB database, the company needs to maintain synchronization between services, so every application event is written to a Kafka queue. Event processing runs against the queue to identify relevant events that can then trigger specific actions – for example customizing customer questions, firing off emails, presenting new offers and more. Relevant events are written to MongoDB, enabling the user experience to be personalized in real time as customers interact with the service.

All events are also written into Hadoop where they can be aggregated and processed with historical activity, in conjunction with additional customer data from the insurance providers. This enables the company to build

enriched data views such as user profiles or policy offers. The models are then imported into the operational MongoDB databases to further enhance user experience, and maximize cross and upsell opportunities.

As a result of its modernized architecture, comparethemarket.com has established a leading position in the highly competitive price comparison market, while achieving 2x faster time to market after migrating from its former relational database to MongoDB, and enabled continuous delivery to push new features live every day.

Leading Global Airline

Revenue Optimization with MongoDB, Spark, and Hadoop

Through a series of mergers and acquisitions, the airline's customer data was scattered across 100 different systems. As a result, the company had no way to gain a single, 360 degree view of the business in order to analyze customer behavior, identify gaps in product portfolios, or present a consistent and personalized passenger experience across airline brands.

With its data lake built on Hadoop, the airline initially evaluated Apache HBase to serve operational applications, but found the column-oriented data model to be restrictive. The need to pre-define column families meant that any functional change in the online applications would break HBase's single view schema. The lack of secondary indexes prevented the database from efficiently handling the array of queries needed for customer care applications and real-time analytics.

After further technology evaluation, the company has been able to bring together customer profiles into a single view stored in MongoDB Enterprise Advanced, distributed across multiple data centers to service the online web, mobile and call center applications. All customer interactions, ticket sales and account data are processed and stored in MongoDB, and then written to the company's Hadoop cluster where Spark machine learning jobs are run to build customer classifications, optimize ticket pricing and identify churn risks. These are then retrieved by MongoDB to serve the online applications. Spark processes are also run against the live operational data in MongoDB to update customer classifications and personalize offers in real time, as the customer is live on the web or speaking with the call center.

With MongoDB, Hadoop, and Spark powering its modern data architecture, the airline is meeting its goals of delivering personalized experiences to the millions of passengers it carries every year, while optimizing ticket prices and enhancing service offerings that reduce competitive threat.

Stratio

Integrates Apache Spark and MongoDB to Unlock New Customer Insights for One of the World's Largest Banks

The Stratio Apache Spark-certified Big Data (BD) platform is used by an impressive client list including BBVA, Just Eat, Santander, SAP, Sony, and Telefonica. The company has implemented a unified real-time monitoring platform for a multinational banking group operating in 31 countries with 51 million clients all over the world. The bank wanted to ensure a high quality of service and personalized experience across its online channels, and needed to continuously monitor client activity to check service response times and identify potential issues. The application was built on a modern technology foundation including:

- Apache Flume to aggregate log data
- Apache Spark to process log events in real time
- MongoDB to persist log data, processed events and Key Performance Indicators (KPIs).

The aggregated KPIs, stored by MongoDB enable the bank to analyze client and systems behavior in real time in order to improve the customer experience. Collecting raw log data allows the bank to immediately rebuild user sessions if a service fails, with analysis generated by MongoDB and Spark providing complete traceability to quickly identify the root cause of any issue.

The project required a database that provided always-on availability, high performance, and linear scalability. In addition, a fully dynamic schema was needed to support

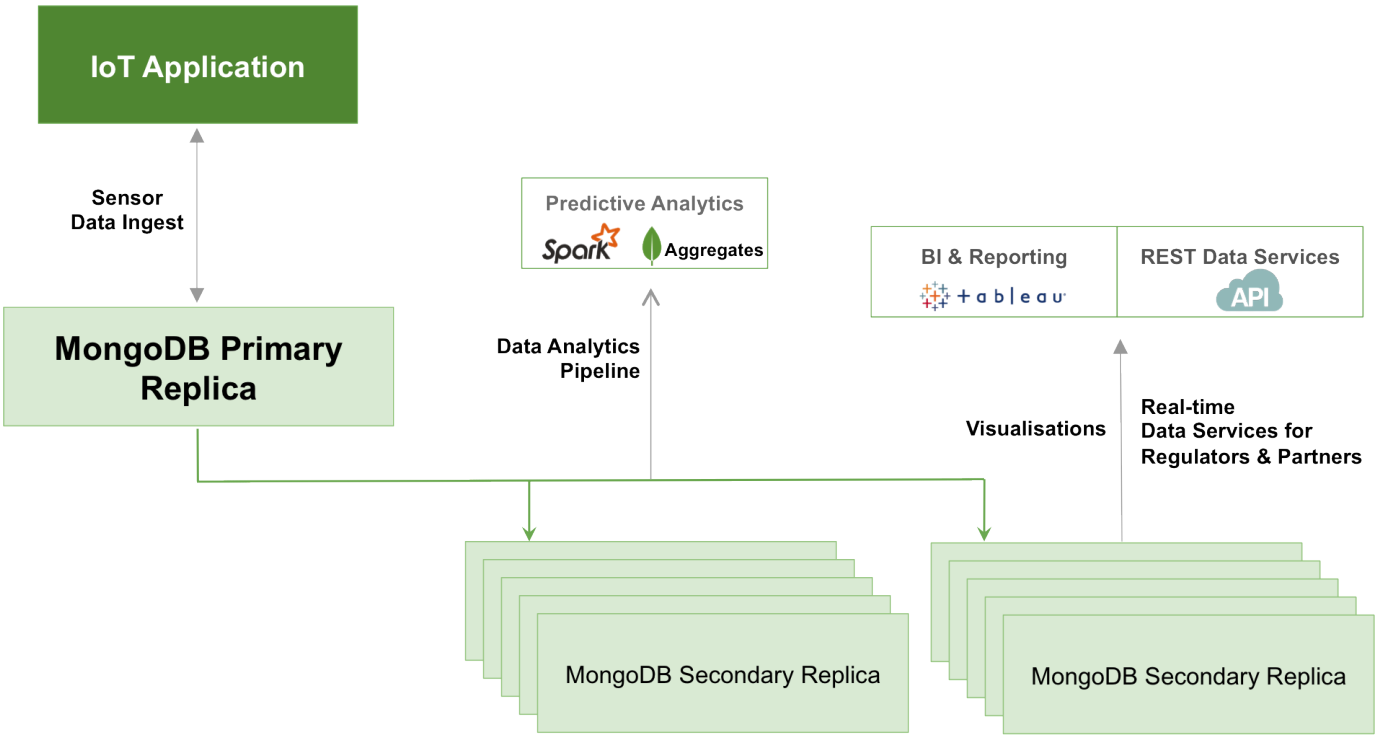

Figure 2: Isolating operational from analytical workloads in a single database cluster

high volumes of rapidly changing semi-structured and unstructured JSON data being ingested from a variety of logs, clickstreams, and social networks. After evaluating the project's requirements, Stratio concluded MongoDB was the best fit. With MongoDB's query projections and secondary indexes, analytic processes run by the Stratio BD platform avoid the need to scan the entire data set, which is not the case with more simple datastores.

Working with some of the world's largest enterprises, Stratio has seen data lakes growing in use, with MongoDB's distributed design and dynamic schema a great fit as it is impossible to predict what type of data structures need to managed at scale.

Learn more by reading the Stratio case study.

Thinking Beyond the Data Lake: Creating an Operational Data Layer

Many organizations are finding that the data lake can be a central landing zone for cross-silo data, but the architecture is focused only on offline analytics. If the data is to be consumed by dashboards and applications with sub-second response times, a data lake is not sufficient.

Coverage of the Gartner 2017 Hype Cycle for Data Management reported "Hadoop distributions are deemed to be obsolete before reaching the Plateau of Productivity because the complexity and questionable usefulness of the entire Hadoop stack is causing many organizations to reconsider its role in their information infrastructure.

To meet these challenges, organizations need an operational data layer to expose cross-silo data in real-time across the whole organization. An operational data layer built on MongoDB presents a more agile, performant and cost effective solution.

Like Hadoop-based data lake, MongoDB offers:

- A flexible "schema-on-read" data model that can ingest, store, and process data of any structure
- A powerful query framework allowing the creation of sophisticated processing pipelines for data analytics and transformations, complemented by native integration with Apache Spark, and multiple SQL interfaces to the underlying data

	MongoDB	Hadoop
Data Storage Granularity	Each record stored in discrete documents ranging from a few KBs up to 16MB of data	Data batched into files stored in blocks of 64MB to 128MB
Read Access Patterns	Scans subsets of the data. Queries against primary and secondary indexes	Scans most or all of the data. Non-indexed queries
Write Access Patters	Frequent updates and inserts	Inserts (append) only
Low Latency Access from Operational Applications	Analytics models and transactional data served directly from the MongoDB operational data layer	Requires integration of operational database layer (MongoDB) on top of the data lake

Table 2: Qualifying requirements against MongoDB operational data layer and Hadoop data lake

- A distributed storage layer that can parallelize processing across multiple replicas, and scale horizontally as data volumes grow

MongoDB builds upon these core capabilities to add more controlled organization and management around the data, making it immediately accessible to both operational and analytical applications that need to consume it:

- Rather than just pack data into large, coarse-grained blocks (64MB - 128MB in size), data is stored in granular documents, and organized into collections that can represent specific business domains, i.e. customer data, product data, social feeds, sensor streams from specific business assets, sales data, etc. In addition to providing the flexibility of schema-on-read, MongoDB schema validation can enforce data governance with schema-on-write controls
- Data can be distributed around the cluster based on specific policy requirements. For example, customer data can be stored in specific data centers for data sovereignty, or pinned to high speed storage devices to meet demanding latency requirements
- Data can be indexed, providing fast access to specific subsets of the data, and then processed by MongoDB's rich query language and aggregation pipeline for real time analytics, graph processing and faceted search

Table 2 identifies the key attributes that can help you determine the appropriate solution.

Operational Data Layer in the Cloud: MongoDB at KPMG

After evaluating Hadoop, KPMG France turned to MongoDB to power its cloud-based data layer. The Loop accounting suite provides an industry-first financial benchmarking service for KPMG customers.

All raw accounting data from KPMG's customers' business systems, such as sales data, invoices, bank statements, cash transactions, expenses, payroll and so on, is ingested from Microsoft SQL Server into MongoDB. This data is then accessible to KPMG's Certified Public Accountants (CPAs) to generate the customer's financial Key Performance Indicators (KPIs). The data in the MongoDB data layer allows KPMG customers to benchmark their financial performance against competitors operating in the same industries within a specified geographic region. They can compare salary levels, expenses, margin, marketing costs – in fact almost any financial metric – to help determine their overall market competitiveness. The MongoDB data layer enables KPMG to manage large volumes of structured, semi-structured, and unstructured data, against which users can run both ad-hoc and predefined queries supporting advanced analytics and business intelligence dashboards. KPMG is continuously loading new data to the data layer, while simultaneously supporting thousands of concurrent users.

KPMG also considered building a data lake on Hadoop, but the architecture of MongoDB, coupled with the power of the aggregation pipeline provided a much simpler solution, while delivering the storage and analytics functionality required by the application. Learn more by reading the case study.

MongoDB Data Layer at Leading Travel Technology Provider

Handling over 450m passengers per year, the technology travel provider connects key players in the travel industry: travel agencies, corporations, airlines, airports, hotels, railways, and more. The company built an operational data layer to serve three critical use cases for its airline customers:

- Revenue tracking and auditing of ticket sales – prorating revenue across the airlines, travel agents, and government tax authorities.
- Business analytics and reporting across a range of key performance indicators such as sales by season, carrier, agent, loyalty program, and so on.
- Servicing complex, real-time ad-hoc queries, such as "show me all European passengers stranded in North America as a result of weather-related cancellations", or "give me all passengers with an Air Passenger Duty tax payable to the UK Government".

The company is able to run queries against 50 billion+ documents and 35TB of data stored in MongoDB, while keeping response times below just several seconds for the most complex queries. The company wanted to maintain the ability to execute sophisticated search and analytics processes against the data, with no limitations on the types of queries they ran, or the size of the dataset. Users included business analysts and data scientists, as well as senior managers who wanted to use a GUI to graphically construct queries enabling them to get immediate answer to business questions.

MongoDB was one of multiple options evaluated, including HDFS with a SQL-on-Hadoop execution engine. MongoDB delivered a number of advantages for the use-case:

- Better scalability as data volumes grew
- Greater developer ease-of-use, which gave the company faster time to market
- Storage efficiency with data compression ratios of up to 80%
- Tiered storage architecture with MongoDB zones
- Low operational overhead with MongoDB Ops Manager
- Robust security protection with Kerberos authentication and encryption of data at rest

The travel industry is highly competitive. Passenger volume is increasing, but margins are slim. Technology can give the company competitive advantage, but only if it can launch new services to market quickly. This is the value MongoDB provides. It is allowing the company to build applications at a much higher velocity and lower complexity than traditional relational databases and Hadoop-based data lakes.

Conclusion

Hadoop-based data lakes have enabled some organizations to efficiently capture and analyze unprecedented volumes of data generated from connected devices and users. But without being able to expose that data to operational applications, users are struggling to maximize returns on their Hadoop investments. The longer it takes to surface insight to operational processes, the less valuable that insight is. With its flexible data model, powerful in-database analytics, distributed, scale-out architecture, and low latency performance, MongoDB provides the best solution to operationalize the data lake.

Additional Information to Learn More

- Apache Spark & MongoDB: Turning Analytics into Real-Time Action
- Big Data: Examples and Guidelines for the Enterprise Decision Maker
- MongoDB Architecture Guide

We Can Help

We are the MongoDB experts. Over 4,300 organizations rely on our commercial products, including startups and more than half 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 Atlas is a database as a service for MongoDB, letting you focus on apps instead of ops. With MongoDB Atlas, you only pay for what you use with a convenient hourly billing model. With the click of a button, you can scale up and down when you need to, with no downtime, full security, and high performance.

MongoDB Stitch is a backend as a service (BaaS), giving developers full access to MongoDB, declarative read/write controls, and integration with their choice of services.

MongoDB Cloud Manager is a cloud-based tool that helps you manage MongoDB on your own infrastructure. With automated provisioning, fine-grained monitoring, and continuous backups, you get a full management suite that reduces operational overhead, while maintaining full control over your databases.

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.

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.com)
MongoDB Enterprise Download (mongodb.com/download)
MongoDB Atlas database as a service for MongoDB (mongodb.com/cloud)
MongoDB Stitch backend as a service (mongodb.com/cloud/stitch)

mongoDB

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