DEFINE RESEARCH JOURNAL
Online Open Access publishing platform for Multidisciplinary Research

Research Article **ISSN 3107 – 7927**

The Relevance of MariaDB in Modern Data Science Workflows

Shellymol[1] , Asha Mary Chacko[2] ,Dr.Peter Varghese[3]

1-Research Scholar,ShriVenkatrshwara University, Gajraula, UP

Email: Shellymol1986@gmail.com

2- Research Scholar, IES University Bhopal

Email:ashamarychacko1994@gmail.com

3-, Principal, De Paul Arts and Science College- Edathotty,

Email:peterooroth@gmail.com

Abstract

Data science, as a process that owes its existence to the need for knowledge and insight extraction from data, has always been dependent on efficient data management systems. This paper will explore the growing importance of MariaDB in the context of modern data science processes. MariaDB, which started its life as a community-supported fork of the MySQL relational database management system, has grown from being a simple relational database management system to a flexible platform that is capable of supporting a wide variety of data types and processing requirements. With its pluggable storage engine architecture, powerful SQL engine, and specific functionality for analytical, semi-structured, and geospatial data, MariaDB has emerged as a very attractive option for different levels of the data science process. This paper will explore the architectural advantages of MariaDB, its ability to integrate with other popular data science tools, its specific functionalities such as ColumnStore for OLAP and MindsDB for in-database machine learning, and its relative position in the data systems space compared to other popular alternatives.

Key Words

Database Management , Data Science, AI, Data Analysis

1. Introduction

The field of data science is essentially built on the premise of effective acquisition, processing, analysis, and interpretation of large and diverse datasets. As organizations begin to increasingly rely on data for strategic decision-making and the development of intelligent applications, the supporting data management infrastructure assumes paramount importance. Data scientists need databases that are not only adept at storing and retrieving data efficiently but also enable complex analytical tasks, seamless integration with various programming platforms, and scalability to handle the ever-increasing volumes of data. The requirements of databases in the data science community are complex and varied, including high-throughput transaction processing for data ingestion, as well as complex analytical queries for model development and deployment.

MariaDB, an open-source relational database management system, has recently come into prominence in this area. Developed by the same team of individuals who created MySQL after its acquisition by Oracle Corporation, MariaDB was conceptualized to retain the open-source ethos while incorporating large improvements in performance, scalability, and functionality.1 Its inherent compatibility with MySQL makes it easy

to migrate to and takes advantage of an existing infrastructure of tools and applications, causing minimal disruption to organizations undergoing a transition in database infrastructure.1

This paper argues that the dynamic capabilities and comprehensive ecosystem of MariaDB make it extremely relevant in contemporary data science applications. The flexibility of its architecture, especially with its pluggable storage engine architecture, makes it capable of handling the entire range of data science needs, starting from structured data management to analytics and even machine learning. Through an examination of the inherent capabilities of MariaDB, its relevance to the data science life cycle, and its specialized capabilities, this analysis will attempt to clarify the increasing significance of MariaDB as a building block for data-driven endeavors.

2. MariaDB Fundamentals and Architectural Strengths

MariaDB's foundational design and continuous development have established it as a robust and adaptable database system, offering several inherent strengths that align well with the demands of data science.

Core Definition, Open-Source Philosophy, and MySQL Compatibility

MariaDB is defined as an open-source relational database management system (RDBMS) that functions as a drop-in replacement for MySQL.1 This lineage is crucial, as it ensures high compatibility with MySQL, allowing for easy migration without significant application changes.1 The project's commitment to an open-source model, primarily under the GPL license, fosters a vibrant community and ensures that a full-featured package is available, unlike some proprietary alternatives that gate advanced functionalities behind enterprise editions.2 This open nature contributes to its cost-effectiveness and flexibility, reducing operational expenses and offering a powerful alternative to proprietary database solutions.1

Key Features: Performance, Scalability, Security, and Cost-Effectiveness

The architecture of MariaDB focuses on a number of key features that are essential for data-intensive applications.

One of the major benefits is its performance, which is highly optimized to handle a large number of connections efficiently.1 It shows better performance than MySQL, especially when querying views, by optimizing the mechanism to query only the required tables, as opposed to MySQL, which may end up querying all the connected tables unnecessarily.2 Additionally, the MyRocks storage engine and RocksDB in MariaDB are optimized for performance with flash storage, which is becoming more common in contemporary data systems.2 The use of a segmented key cache also increases efficiency by breaking down the locks into 64 segments, enabling some processes to run in parallel.3

Another important aspect of MariaDB is its scalability. MariaDB is designed to be reliable and easy to use, whether it is for small-scale or enterprise-level processing tasks.2 Its multi-threading capabilities, such as the thread pooling capability, enable it to process more tasks and support 200,000 connections simultaneously, a feature that is normally available in the enterprise version of other databases.2 This makes it possible for applications to scale out and support sudden increases in traffic or rapid

business growth.2 In large-scale data management, MariaDB is more scalable and faster in query processing than MySQL.4

Another important aspect of MariaDB is its security features, which include the use of various mechanisms to protect critical data. It has advanced encryption capabilities for tables, logs, and communication, which ensure data integrity and confidentiality.1 The use of data encryption and secure communication channels is an essential part of its security features, which protect data from any kind of unauthorized access.5 Another important feature is its ability to audit user activities, which is extremely useful for security and compliance purposes, especially in industries where data security is a major concern.6

Cost-effectiveness is a natural byproduct of the open-source model that MariaDB uses. The fact that MariaDB is free means that there are no licensing costs, which can be a major factor in total cost of ownership, particularly when scaling out applications, since there are no CPU costs to factor into the equation.1

Understanding MariaDB's Pluggable Storage Engine Architecture

One of the main strengths of MariaDB's flexibility is its pluggable storage engine architecture, which enables users to choose engines tailored for particular use cases. This is especially important in data science, where there are many different data handling needs.

InnoDB: This storage engine is very popular and supports ACID transactions, which is essential for data integrity and consistency in transactional data import tasks in data science applications.2

Aria and XtraDB: These storage engines improve performance and transaction handling, which enables faster data operations.1 Aria, for example, is optimized for crash recovery and provides faster data operations compared to MyISAM, while also ensuring data integrity during bulk inserts.7

ColumnStore: This engine is a key component for analytical queries (OLAP). It offers high-performance analytics and data warehousing functionality, which enables interactive and ad-hoc queries on very large data sets, even hundreds of billions of rows, without manual indexing. The engine's columnar storage layout is optimized for heavy reads and big aggregations, which makes it much faster for complex analytical queries compared to traditional row-storage engines.1

MyRocks and TokuDB: These engines are optimized for performance with flash storage and big data volumes, respectively.2

Connect: This engine enhances the functionality of MariaDB by enabling easy access to unstructured data from within the database. It has the ability to import data from other DBMSs, data files, and even REST APIs, making data ingestion easier for various sources.2

Spider: This engine enables sharding, which involves distributing data across multiple servers to improve scalability and handle large datasets.2

This flexibility in database architecture enables MariaDB to be customized for specific data science operations, ranging from transactional data acquisition to large-scale data analysis in a single database solution.

MariaDB in Cloud Environments (e.g., SkySQL)

The relevance of MariaDB is further heightened by its extensive presence and capabilities within the cloud environment. SkySQL, the cloud version of MariaDB, comes with high availability, scalability, and security that are inherent to cloud infrastructure.2 MariaDB'sSkySQL is built on robust cloud infrastructure such as Google Cloud and Google Kubernetes Engine (GKE). It harnesses the power of cloud-native capabilities to optimize deployment and management.13 It uses a mix of high-performance block storage such as Google Persistent Disk for transactions and low-cost object storage such as Cloud Storage for analytics, giving users flexibility and cost-effectiveness.13 This cloud-native capability enables MariaDB to make real-time updates to active services, ensuring that users are always working with the latest bug fixes, security patches, and enhancements.13 The ease of deployment and management, along with elastic scalability, ensures that the service can scale seamlessly with the growth of an organization, from small-scale operations to large-scale enterprises.13

3. Integrating MariaDB into the Data Science Workflow

A general data science process involves a number of iterative steps, ranging from problem formulation to model development and evaluation.14 MariaDB's functionalities and compatibility features make it an important component throughout these steps.

Data Collection and Preparation: Efficient Storage and Retrieval

The first step in any data science process is to identify, collect, and prepare relevant and quality data.14 MariaDB, being a relational database management system, is very useful for efficient data storage and retrieval, which is a basic requirement for this step.1 Data can be obtained from different sources, such as SQL servers, which MariaDB supports easily.14 Its high performance and scalability enable it to process a large amount of raw data for effective ingestion and organization, which is a solid starting point for further analysis.1 The capability to support a large number of concurrent connections ensures that multiple data sources or data collection tasks can be processed by MariaDB at the same time without compromising performance.

Data Cleaning and Preprocessing: Leveraging SQL Capabilities

Raw data is usually messy, incomplete, or inconsistent, and this requires intense cleaning and preprocessing.14 MariaDB's powerful SQL environment is a huge asset during this process. Data scientists can utilize conventional SQL queries to clean raw data, handle missing data, correct errors, and remove inconsistencies.1 SQL queries can be utilized to transform, normalize, or reformat data to ensure it is in its most optimal form. For example, complicated UPDATE queries, JOIN statements, and CASE statements can be utilized to clean and normalize data directly in the database, eliminating the need for multiple processing steps for certain tasks.

Exploratory Data Analysis (EDA) and Feature Engineering

However, exploratory data analysis (EDA) is an important step in understanding the data before predictive modeling is done.14 MariaDB's SQL environment makes this easy. Data scientists can run complex queries to obtain descriptive statistics (such as mean, median, standard deviation, and quartiles), aggregate data, and detect outliers.14 The optimized views of the database, which query only the necessary

tables, can speed up EDA by allowing faster access to virtual tables that result from complex joins.3

Feature engineering, the process of creating new variables from existing ones to improve model performance, is also aided by MariaDB's SQL environment. Complex SQL queries, such as window queries and common table expressions (CTEs), can be used to create new features in the database, such as moving averages, differences, and transformations.9 This can speed up data transfer and take advantage of the optimized query execution of the database.

Supporting Model Building, Deployment, and Monitoring

The capabilities of MariaDB's integration extend to the support of the latter stages of the data science process, such as model building, deployment, and monitoring.

Programming Language Connectors are critical for data scientists who spend most of their time in a programming environment such as Python or R. MariaDB supports official connectors for these programming languages, allowing for smooth interaction with the database.

Python: The MariaDB Connector/Python enables Python programmers to connect to the database, run SQL commands, and fetch results.16 The process of installation is easy using pip, and the usual workflow includes importing the mariadb module, connecting to the database using host, port, user, and password information, creating a cursor object, running SQL statements (SELECT, INSERT), fetching results, and committing transactions.16

R:The RMariaDB package offers a database interface and a MariaDB driver for R, striving to fully support the R DBI specification.20 Data analysts can install this package using install.packages("RMariaDB"), and then load it using library(RMariaDB). They can then connect to the database using dbConnect(). Other functions such as dbListTables(), dbWriteTable(), dbReadTable(), and dbSendQuery() help data analysts manipulate and query data from MariaDB databases directly from their R environment.21

ETL and Data Sync Tools further extend the capabilities of MariaDB by making data pipelines easier to manage. Tools such as Skyvia enable cloud data integration in a universal fashion, allowing one-way or two-way synchronization of MariaDB data with other sources.22 This includes creating complex data pipelines and replicating MariaDB data to data warehouses for further analysis with BI tools or Python scripts.22 Panoply also enables code-free data pipelines and ETL, automatically structuring MariaDB data into queryable tables and connecting to popular BI tools such as Shiny (R).20 These tools make it easy to automate the complex process of extracting, transforming, and loading data, ensuring that analytical applications are always working with the latest cleaned data.20

API Sharing enables real-time data access for deployed models or applications. Tools such as Skyvia Connect enable the creation of a web API layer for MariaDB data, exposing it through SQL API and OData REST API endpoints.22 This allows for real-time integration with a variety of tools and data-oriented applications, which is essential for the operationalization of machine learning models that need real-time data inputs or outputs.22

For Model Monitoring and Maintenance, MariaDB can be used as a repository for storing model predictions, performance, and input data drifts. Monyog, which comes with MariaDB TX, is a tool that provides comprehensive monitoring features, which can be used by database administrators and data scientists to monitor the status of the server and query performance, which can be essential for data pipelines feeding into or from models.6

4. Advanced MariaDB Features for Data Science Applications

MariaDB's relevance in data science is significantly bolstered by its specialized features designed to handle complex data types and analytical workloads.

MariaDB ColumnStore for Analytical Workloads (OLAP)

MariaDB ColumnStore is a highly efficient storage engine designed and optimized for high-performance analytics and data warehousing, complementing MariaDB's transactional strengths.1 Unlike conventional row-storage engines, ColumnStore is designed to store data in a columnar fashion, which is extremely efficient for analytical queries that typically involve aggregating data from numerous rows but only a few columns.10

ColumnStore's columnar storage architecture enables it to efficiently manage enormous amounts of data, even petabytes of compressed data stored on shared storage.10 It is particularly adept at

handling complex ad-hoc queries and aggregations, which it can accomplish in seconds, compared to minutes, hours, or even days for a traditional relational database management system.10 This is especially important for interactive data analysis and real-time visualization of large datasets, enabling users to derive valuable insights without having to manually index them extensively.9 ColumnStore's architecture is designed to avoid locking, ensuring that reads are not blocked, even by

ALTER TABLE statements, which is common in OLAP databases that are optimized for heavy read activity.10

Moreover, MariaDB Enterprise Platform, with the use of ColumnStore, provides a cloud-native storage solution. This solution enables flexible storage choices, including the ability to use Amazon S3-compatible object storage, which can result in substantial cost reductions and provide virtually unlimited scalability for analytics data.9

MariaDB also provides support for Hybrid Transactional/Analytical Processing (HTAP) via its "smart transactions" feature. This enables the best-of-breed capabilities of row-based storage (optimized for fast transactions) and column-based storage (optimized for fast analytics) in a unified platform.9 This allows developers to embed real-time analytics and historical data directly into applications, enabling more insightful customer experiences without the need for data warehouses or batch processing.9 For example, SaaS companies can provide self-service analytics to their customers, who can then analyze their own data directly.9

One of the most interesting use cases of the capabilities of MariaDBColumnStore is in the Institute for Health Metrics and Evaluation (IHME). With data points increasing exponentially from 2 billion in 2010 to close to 100 billion in 2015, IHME required a solution to handle multi-billion row tables for its Global Burden of Disease (GBD)

analysis. MariaDB AX, with the ColumnStore engine, offered a solution that cut disk I/O by orders of magnitude for read-heavy analytics workloads. It also improved scalability and optimized resources by compressing data to 40% of its original size, thus requiring less hardware. This enabled IHME to analyze data using standard SQL for both transactional and analytical workloads in a single enterprise-class solution, resulting in faster analysis and effortless handling of growing amounts of data.25 Similarly,

DBS Bank used MariaDB TX and its MaxScale database proxy's change data capture (CDC) modules to automatically feed data from their transactional MariaDB database to Hadoop clusters in real-time. This was essential for them to immediately derive insights from their customers' data, thus highlighting the use of MariaDB in real-time analytics.6

JSON Functions for Semi-Structured Data

The emergence of semi-structured data, especially in JSON form, requires database systems capable of efficiently storing and querying this type of data. Although MariaDB is mainly an RDBMS, it has incorporated strong JSON functions to support this type of data. MariaDB supports JSON data as LONGTEXT, which provides flexibility in schema modification, although it is different from MySQL's native JSON data type.3

MariaDB provides a suite of functions to manipulate JSON data, including:

- JSON_EXTRACT(): Reads a JSON document and returns the value at a specified path. This is crucial for extracting specific fields from nested JSON objects.[26]
- JSON_UNQUOTE(): Used to unquote the plain text value returned by JSON_EXTRACT(), making it suitable for direct comparison or display.[26]
- JSON_VALID(): Validates if a string contains valid JSON, which is important for data quality checks during ingestion.[26]
- JSON_SET(): Allows updating a single JSON key within a document, returning an updated JSON document that can be written back to the database.[26]

For efficient querying of JSON data, it is recommended to create generated columns on top of JSON_UNQUOTE(attributes->'$.path') and index the generated columns.26 This enables the database to pre-parse and store the values, which results in a substantial speedup over parsing JSON in queries.26 It is also recommended to avoid using wildcard paths and to keep JSON documents small to avoid full row scans.26 These features make it possible for data scientists to store and analyze semi-structured data, together with relational data, in MariaDB.

Geospatial (GIS) Functions for Location-Based Analytics

Location-based data is increasingly vital across various domains, from urban planning to logistics and environmental monitoring. MariaDB's geospatial (GIS) functions and spatial extensions enable efficient storage and analysis of geographical information.

MariaDB supports standard geospatial data types, including:

- POINT: Represents a single location defined by X and Y coordinates, ideal for addresses or specific points of interest.[27]

- LINESTRING: A sequence of points forming a continuous line, suitable for paths, roads, or rivers.[27]
- POLYGON: Represents an enclosed area, perfect for outlining parks, city boundaries, or property lots, and can include holes for more detailed representations.[27]
- Multi-Point, Multi-LineString, and Multi-Polygon: Used for storing complex geometries composed of multiple points, lines, or polygons, respectively, accommodating more intricate geographical features.[27]

The database's GIS functionality allows the execution of complex geographical queries and the implementation of spatial indexes to optimize querying of location-based data, leading to vastly improved response times and efficiency.[27] Key spatial functions include:

- ST_Distance: Measures the proximity between points, critical for applications like location-based services.[27]
- ST_Intersects: Identifies overlapping areas, aiding in geographical analytics across industries.[27]
- ST_Contains: Determines if a point falls within a polygon, enabling real-time data filtering for applications such as environmental monitoring or delivery services.[27]
- ST_Union, ST_Collect, ST_ConvexHull: Functions for combining, gathering, or simplifying geometries, useful for creating larger shapes or simplified representations of area coverage.[27]

These capabilities allow data scientists to perform spatial counts, find closest records, and identify spatial overlaps, providing rich insights from geographical datasets.[28] The ability to integrate and analyze geospatial data directly within MariaDB enhances its utility for a wide array of data science applications that rely on location intelligence.

Temporal Data Tables for Historical Analysis

Data evolution over time is important for many analytical purposes, such as trend analysis, auditing, and reporting. MariaDB 10.5 brought support for temporal tables, which enable users to query data as it was at a certain point in time.3 This is extremely useful for analytics, recovery, reporting, and investigation of data, as it gives a direct way to access the historical versions of the data without having to manually archive or snapshot the data.3 For data scientists, temporal tables make it easier to analyze changes in data, re-execute analysis on previous states, or compare current data with previous standards directly in the database environment. This does away with the need for a separate data warehouse for accessing historical data only.9

Machine Learning Integration with MindsDB

One of the major improvements in the relevance of MariaDB to data science is its ability to work seamlessly with MindsDB, an open-source machine learning engine. This enables data scientists to perform machine learning tasks directly in MariaDB using SQL queries.29

This is made possible by the Connect Storage Engine in MariaDB.29 MindsDB communicates with MariaDB Server using a normal user account, creating a special

mindsdb database in MariaDB. This mindsdb database holds tables such as commands and predictors, which are used as an interface to communicate with MindsDB.29

Data scientists can use a single SQL INSERT command into the predictors table to trigger the training of a machine learning model. This SQL command is a signal to MindsDB, indicating the name of the model, the column to predict, and the SQL query to fetch the training data from MariaDB.29 For instance,

INSERT INTO predictors (name, predict, select_data_query) VALUES ('bikes_model', 'count', 'SELECT * FROM test.bike_data'); tells MindsDB to build a model named 'bikes_model' that predicts the 'count' column based on data from the test.bike_data table in MariaDB.29 MindsDB, as an AutoML engine, takes care of all the complexities involved in building a model. This revolutionary combination makes it extremely easy to build and deploy machine learning models and enables data scientists to carry out predictive analysis directly where their data is, using a SQL interface.

5. Performance Optimization for Data Science Workloads

Optimizing MariaDB's performance is crucial for handling the large and often complex datasets typical in data science. Several strategies can be employed to enhance data loading, ingestion, and query execution.

Data Loading and Ingestion

Bulk loading large amounts of data into MariaDB is an essential initial process for most data science tasks.

LOAD DATA INFILE is a very efficient syntax for bulk inserts from text files into tables. This statement can be as much as 20 times faster than using INSERT statements, especially for data sets above 10,000 records.7

Bulk import statements using INSERT or REPLACE statements can also cut down on transactional overheads compared to other processes.7

Batch size optimization is also necessary, and case studies recommend loading data in batches of 100,000 rows for faster processing, as larger batches can cause memory issues and slower processing.7

Disablingindexes temporarily on the table where data is being loaded and turning the autocommit mode to 0 can further boost data loading speeds (up to 5 times faster, according to reports) by allowing for bulk commits later in the process.7 Indexes can be rebuilt after data loading is complete.

For real-time data ingestion and analytics, MariaDB also supports streaming data adapters such as MaxScale CDC and Apache Kafka. These adapters enable continuous data ingestion from transactional systems, ensuring that analytical data is always up-to-date without manual intervention.30

Query Performance

Optimizing query performance is essential for responsive data analysis and model training.

- **Buffer Sizes:** Tuning database configuration settings, especially the innodb_buffer_pool_size, can dramatically affect insert and query performance. Recommendations suggest setting this parameter to 70-80% of available memory

for large systems.[7]

- **Column Data Types:** Using appropriate data types for columns helps avoid unnecessary storage overhead. For instance, using INT instead of BIGINT when possible can reduce the data footprint by 50%, which in turn can lead to faster query processing due to less data being read from disk.[7]
- **Indexing Strategy:** While indexes are crucial for query speed, their management during data loading is critical. Limiting indexes on tables during data import and adding them post-load can prevent significant slowdowns.[7] For analytical queries, MariaDB'sColumnStore engine can perform ad-hoc queries on massive datasets without requiring manual indexing for every potential query.[9]
- **Normalization vs. Denormalization:** Data normalization reduces redundancy but can result in more complex queries involving multiple joins, potentially impacting performance metrics.[7] In analytical contexts, particularly with columnar stores like ColumnStore, a degree of denormalization can be beneficial for optimizing query performance, as it allows for more parallelization and reduces the overhead of join operations.[31]
- **Temporary Tables:** Utilizing temporary tables for staging data during complex loads or transformations has shown to improve load times by 30% in various benchmarks.[7]
- **Execution Plans:** Periodically reviewing and adjusting execution plans based on real-time analysis is crucial for continually optimizing database configuration and maximizing efficiency, especially for companies relying on timely data integration for analytics.[7]

Storage Engine Selection

The choice of storage engine directly impacts MariaDB's performance characteristics for different workloads.

- **InnoDB:** Ideal for transactional workloads (OLTP) where data integrity and ACID compliance are paramount. It ensures consistency, though it may incur some overhead during insertions due to its safety mechanisms.[7]
- **MyISAM:** Generally faster for read-heavy workloads where ACID compliance is not a strict requirement. However, it may experience table locking during large data imports, potentially causing write delays.[7]
- **Aria:** Designed with crash recovery in mind, it offers faster operations compared to MyISAM while ensuring data safety during bulk inserts, making it a compelling choice for specific applications.[7]
- **ColumnStore:** As discussed, this engine is specifically optimized for analytical workloads (OLAP), providing high performance for intensive reads and complex aggregations on large datasets.[10]

Selecting the appropriate storage engine based on the primary workload (transactional vs. analytical) and data characteristics is a fundamental optimization strategy in MariaDB for data science applications.

6. Comparative Analysis with Other Data Systems in Data Science

MariaDB operates within a diverse ecosystem of data management systems, each with its own strengths and weaknesses. Understanding its positioning relative to other prominent databases used in data science is essential for informed decision-making.

MariaDB vs. MySQL

MariaDB has its roots in a MySQL fork, resulting in many similarities but also some key differences that impact its relevance in data science.

- **Open-Source and Licensing**: MariaDB is completely GPL licensed and open-source, which encourages a more vibrant community. MySQL, although having a community edition licensed under GPLv2, is owned and distributed by Oracle, and some functionalities, such as thread pooling, are available only in its Enterprise version.2 The open-source dedication of MariaDB might result in more flexibility and cost-effectiveness.1
- **Performance**: MariaDB is generally faster than MySQL, especially when querying views because it optimizes by selecting only relevant tables, while MySQL might select unnecessarily.3 MariaDB also provides better performance with flash storage using engines such as MyRocks and RocksDB.2
- **Features**: MariaDB provides additional storage engines and plugins such as Aria, Connect, Spider for sharding, and TokuDB for big data.2 Thread pooling is a standard feature in MariaDB, which is available only in the enterprise version of MySQL.3 MariaDB 10.5 also provides temporal data tables for historical analysis, which is not available in MySQL.3
  - **Data Types**: MySQL provides a native JSON data type that enables parsing and fast access to JSON data. MariaDB's JSON data type stores data in LONGTEXTformat, requiring functions such as JSON_EXTRACT() for parsing, which is slower without generated columns and indexes.3

Overall, MariaDB's continuous innovation, community-driven development, and inclusion of advanced features as standard make it a compelling alternative to MySQL for many data science use cases, especially where open-source flexibility and cost-effectiveness are priorities.

MariaDB vs. PostgreSQL

Both MariaDB and PostgreSQL are full-featured, open-source relational database management systems that are used for storing data in a tabular form and supporting complex SQL queries.17 However, they have some unique features in the context of data science.

- **Database Model and Features**: MariaDB is a MySQL variant that emphasizes high performance, scalability, and flexibility through its pluggable storage engines.5 PostgreSQL is an object-relational database management system that provides a wide range of features, such as materialized views and partial indexing for faster read performance, and advanced data modeling capabilities.5 PostgreSQL is generally chosen for applications requiring strong support for complex ACID transactions and handling complex workloads where data integrity and consistency are of utmost importance.5

- **Query Performance and Scalability**: Benchmarking experiments have shown that PostgreSQL is more efficient than MariaDB in performing complex queries, as it has consistently demonstrated outstanding query response times and high query processing rates.5 Although both databases support vertical and horizontal scaling, PostgreSQL has a scalable design that can efficiently process higher workloads with little extra resource overhead and support seamless horizontal scaling without sacrificing performance.5
- Data Types and Extensibility: MariaDB has a wide range of data types, including spatial and temporal data types, which makes it flexible in terms of matching data types.5 PostgreSQL is more rigid in terms of data types but highly extensible, which enables developers to create new data types and extensions.5
- Security: Both are highly concerned with encrypting data and using secure communication channels. PostgreSQL is famous for its holistic approach to user access control and data security.5

The power of MariaDB is in its flexibility in terms of storage engines, which can be customized depending on the need for transactional, columnar, or high-throughput databases, making it a very flexible solution where the need for customizable performance is paramount.5 PostgreSQL, with its powerful query processing and sophisticated data modeling, is often the choice for large-scale analytics.5

MariaDB vs. NoSQL Databases (e.g., MongoDB)

The contrast between MariaDB (RDBMS) and NoSQL databases such as MongoDB sheds light on the differences between the two in terms of data models and applications.

- **Database Model**: MariaDB is a relational database that follows a relational database model and has a predefined schema for data ingestion, storing data in the form of rows and columns.8 MongoDB, on the other hand, is a document-oriented NoSQL database that stores data in schema-less documents using key-value pairs.8
- **Data Format and Schema Evolution**: MongoDB's schema-less model facilitates easy schema evolution, where new fields can be added to documents without affecting existing data pipelines.32 MariaDB, being relational, makes schema evolution difficult, as adding a new column requires schema modification that may affect existing applications.32
- **Scalability and Performance**: MongoDB performs well in horizontal scaling across multiple servers, ensuring high availability and performance for handling large amounts of unstructured data.8 It employs sharding for optimal document organization.32 Although MariaDB is scalable, it is not preferred for handling "BigData" processing tasks involving MapReduce engines, which are supported by MongoDB.32 MariaDB has introduced NoSQL-like capabilities in version 10, including the Connect engine for handling unstructured data and dynamic columns.2
- **ACID Compliance**: MariaDB, being an RDBMS, emphasizes ACID compliance (Atomicity, Consistency, Isolation, Durability), ensuring data integrity and consistency for transactional applications.8 NoSQL databases such as MongoDB

emphasize flexibility and scalability over ACID properties, although they provide their own consistency models.

MariaDB is preferred for applications requiring data integrity and consistency for structured data, especially for OLTP transactions.8 MongoDB is preferred for applications requiring flexibility and scalability for unstructured or dynamically changing data models.8

MariaDBColumnStore vs. Cloud Data Warehouses (e.g., Snowflake, BigQuery, Redshift)

For big analytical workloads, MariaDBColumnStore faces competition from cloud data warehouses such as Snowflake, Google BigQuery, and Amazon Redshift.

- **Architecture and Deployment**: MariaDBColumnStore provides flexibility to deploy solutions on-prem, in the cloud, or in a hybrid model, giving more control over infrastructure.12 Cloud data warehouses are fully managed cloud-native services that decouple compute and storage, enabling independent scaling.12 They are developed on top of the leading cloud platforms (AWS, Azure, GCP).12
- **Scalability and Performance**: Cloud data warehouses are engineered to handle petabyte-scale data and extremely large analytical workloads, employing columnar storage, auto-partitioning, and query optimization.23 They are optimized for highly parallelized queries and can dynamically scale compute capacity according to workload demand.33 MariaDBColumnStore also provides high-performance analytics on very large datasets and supports petabytes of compressed data.9

For large-scale analytical workloads, MariaDBColumnStore competes with specialized cloud data warehouses like Snowflake, Google BigQuery, and Amazon Redshift.

- **Cost Model**: The cost model of cloud data warehouses is pay-per-gigabyte scanned or flat rate, which may be cost-effective for large and infrequent queries but potentially very expensive for highly concurrent or inefficient queries.23 MariaDBColumnStore, as an open-source solution, provides a cost-effective alternative by avoiding expensive licenses for modern, on-demand analytics at scale.11
- **Features**: Cloud data warehouses offer many sophisticated features such as secure data sharing, zero-copy cloning, and time-travel functionality.12 MariaDBColumnStore, although very capable, is more labor-intensive in terms of infrastructure and optimization compared to the fully managed nature of cloud data warehouses.12

MariaDBColumnStore is an attractive open-source alternative for organizations that require powerful analytics capabilities without the vendor lock-in and potentially higher operational costs of fully managed cloud data warehouses. It enables organizations to apply their existing MariaDB knowledge for both OLTP and OLAP applications, providing a single platform for different data processing requirements.10

7. Case Studies and Real-World Applications

The applicability of MariaDB in data science can be better explained through its usage in real-world applications, which showcase the capability of MariaDB to handle challenging data management and analysis tasks.

DBS Bank, the largest bank in Southeast Asia, initiated a major project to move mission-critical applications from proprietary Oracle databases to open-source platforms, with MariaDB TX being chosen as a key component. The move was necessitated by the need for open-source technology, cost savings, flexibility, scalability, and alignment with tight financial regulations.6 The bank, which processes close to three million financial transactions every month, began cautiously by evaluating MariaDB with non-critical applications before adopting it widely. In two years, DBS Bank had more than 30 applications running on MariaDB, including complex corporate banking applications.6The benefits realized were substantial:

- **Cost Savings**: DBS Bank was able to realize cost savings of between 30% and 70% depending on the application and workload, mainly because of the avoidance of CPU charges that were applicable to proprietary databases.6
- **Flexibility and Scalability**: MariaDB gave DBS Bank the flexibility to select the most suitable database topologies and also facilitated the running of applications with microservices in containers such as Docker, which helped with resiliency and independent scaling.6
- **Real-time Data Streaming**: One of the most important features for DBS Bank was the ability to automatically stream data from their MariaDB TX transactional database to Hadoop clusters in real-time using the change data capture (CDC) modules of MariaDBMaxScale. This was essential for them to gain immediate insights into their customer data, highlighting the importance of MariaDB in real-time analytics.6

Compliance and Monitoring: The MariaDB features of data-at-rest encryption and user auditing were extremely useful in ensuring that the organization adhered to the strict financial services regulations. The Monyog monitoring tool, which is a part of MariaDB TX, enabled the organization to monitor the status of the server and the performance of queries in a very efficient manner.6

Another example that stands out is that of the **Institute for Health Metrics and Evaluation** (IHME), which is a global health research center. IHME was dealing with the issue of handling multi-billion row tables for its Global Burden of Disease (GBD) project, which was witnessing exponential growth from 2 billion data points in 2010 to close to 100 billion in 2015.25 Their existing MySQL-compliant infrastructure was unable to handle the query and data loading traffic generated by their 16,000 CPU-core high-performance computing cluster.

IHME chose **MariaDB AX**, specifically utilizing its **ColumnStore storage engine**, to overcome these issues.25

Improved Performance: The ColumnStore columnar storage engine in MariaDBColumnStore greatly minimized disk I/O, making it much faster than row-based storage engines for read-heavy analytical workloads on big data. This enabled fast access to the results of aggregate functions, allowing for better data analysis.25

Enhanced Scalability and Resource Utilization: The sophisticated data compression capabilities of MariaDBColumnStore enabled IHME to compress their data to about

40% of its original size, thus minimizing the hardware requirements for their enormous data set deployments.25

Faster Data Analysis: The system enabled IHME to utilize standard SQL to combine both transactional and analytical workloads into a unified enterprise-level system, which is easier to manage and execute.25

These case studies demonstrate the real-world applicability of MariaDB in different data-intensive settings. Apart from these case studies, MariaDB is widely employed in different applications that demand high-quality data management, such as web application development, eCommerce solutions, enterprise applications, cloud computing, and logging applications.1 Its capability to support high-performance applications with high data handling capacity, together with its sophisticated SQL environment and replication and clustering capabilities, makes it a versatile solution for different data-intensive projects.1

8. Conclusion

MariaDB has clearly moved on from its roots as a MySQL fork to become a very relevant and capable database solution in the context of modern data science. Its open-source model, combined with innovation and strong feature sets, makes it a flexible and affordable solution for a broad range of data science-related tasks.

The discussion above highlights the strong architectural points of MariaDB, especially its pluggable storage engine architecture, which enables customized performance for a broad range of data processing tasks. Storage engines such as InnoDB provide support for transactional consistency, and ColumnStore offers high-performance analytics for very large datasets, supporting hybrid transaction/analytical processing (HTAP) workloads. This is very important in data science, where tasks can vary from fast data ingestion to complex analytical queries.

Moreover, MariaDB's strong feature set directly supports modern data science needs. Its JSON support makes it easy to work with semi-structured data, and its strong geospatial support enables complex location-based analytics. The support for temporal tables makes it easy to work with historical data, and the innovative integration with MindsDB introduces in-database machine learning capabilities, which enable data scientists to develop and deploy machine learning models using familiar SQL.

MariaDB's excellent integration with popular data science programming languages such as Python and R, as well as support for ETL and data synchronization tools, ensures smooth integration with existing data science workflows. Performance optimization techniques, ranging from optimized bulk data loading to smart indexing and storage engine choice, further add to its usefulness in handling large-scale data operations.

Although cloud-native data warehouses such as Snowflake or BigQuery provide fully managed services and unparalleled scalability for a particular analytical workload, MariaDBColumnStore provides a highly attractive open-source alternative for organizations that want control over their infrastructure and can thereby cut back on expensive proprietary licensing fees. Again, although NoSQL databases shine in terms of schema flexibility and horizontal scaling for unstructured data, MariaDB's relational core, complemented by its JSON and Connect engine capabilities, provides

a well-rounded solution for handling different types of data. In comparison to PostgreSQL, MariaDB offers tremendous flexibility with its storage engines, which can be tailored to suit different performance needs.

The DBS Bank and IHME case studies of real-world applications of DBS Bank and IHME clearly demonstrate MariaDB's ability to support mission-critical workloads, provide significant cost savings, support real-time analytics, and efficiently process multi-billion row tables.

Conclusion

In summary, the combination of open-source, high-performance, flexible, and specialized capabilities for analytics, semi-structured, geospatial, and machine learning workloads makes MariaDB an important player in the world of data science. MariaDB not only acts as a data storage system but also as an active component of the data science process, enabling data scientists to leverage powerful tools for data management, analysis, and discovery. The continued evolution of MariaDB indicates that it will remain an important and dynamic force in the data science world for years to come.

Works cited

1. What is MariaDB? A Beginner's Guide for 2025 - MilesWeb, accessed August 20, 2025, https://www.milesweb.com/blog/hosting/what-is-mariadb/
2. What Is MariaDB and How Does It Work? - Pure Storage, accessed August 20, 2025, https://www.purestorage.com/nl/knowledge/what-is-mariadb.html
3. MariaDB vs MySQL (Updated 2025) - Integrate.io, accessed August 20, 2025, https://www.integrate.io/blog/mariadb-vs-mysql-everything-you-need-to-know/
4. aws.amazon.com, accessed August 20, 2025, https://aws.amazon.com/compare/the-difference-between-mariadb-vs-mysql/#:~:text=MariaDB%20is%20more%20scalable%20and,multiple%20engines%20in%20one%20table.
5. Postgres vs MariaDB: A Comprehensive Guide - RisingWave, accessed August 20, 2025, https://risingwave.com/blog/postgres-vs-mariadb-a-comprehensive-guide/
6. Case Study MariaDB – DBS Bank - DataX Solution, accessed August 20, 2025, https://www.dataxsolution.net/case-study-mariadb/
7. The Truth About LOAD DATA and Its Impact on MariaDB Performance - Myths vs. Facts, accessed August 20, 2025, https://moldstud.com/articles/p-the-truth-about-load-data-and-its-impact-on-mariadb-performance-myths-vs-facts
8. Difference Between MongoDB and MariaDB - GeeksforGeeks, accessed August 20, 2025, https://www.geeksforgeeks.org/mongodb/difference-between-mongodb-and-mariadb/
9. Database Workload Versatility - MariaDB, accessed August 20, 2025, https://mariadb.com/products/enterprise/workload-versatility/
10. The benefits of MariaDBColumnStore - Vettabase, accessed August 20, 2025, https://vettabase.com/the-benefits-of-mariadb-columnstore/
11. Columnar Database for Analytics - MariaDB, accessed August 20, 2025,

https://mariadb.com/database-topics/analytics/

12. Compare MariaDB vs Snowflake - InfluxDB, accessed August 20, 2025, https://www.influxdata.com/comparison/mariadb-vs-snowflake/
13. MariaDB Case Study | Google Cloud, accessed August 20, 2025, https://cloud.google.com/customers/mariadb
14. A Comprehensive Guide to Master the Data Science Workflow | by Anamika Singh - Medium, accessed August 20, 2025, https://medium.com/codex/a-comprehensive-guide-to-master-the-data-science-workflow-739295117d67
15. How to Create and Structure Data Science Workflow - Hevo Data, accessed August 20, 2025, https://hevodata.com/learn/data-science-workflows/
16. How to connect MariaDB with Python - IONOS, accessed August 20, 2025, https://www.ionos.com/digitalguide/websites/web-development/python-mariadb/
17. MariaDB vs PostgreSQL - Difference Between Open-Source Relational Databases - AWS, accessed August 20, 2025,
18.]https://aws.amazon.com /compare/the-difference-between-mariadb-and-postgresql/
19. MariaDB Use Cases for Enterprise, accessed August 20, 2025, https://mariadb.com/products/enterprise/use-cases/
20. How To Setup MariaDB Connector: 3 Easy Integration Steps - Estuary, accessed August 20, 2025, https://estuary.dev/blog/mariadb-connectors/
21. Integrate MariaDB with Shiny for Analytics - Panoply, accessed August 20, 2025, https://panoply.io/connectors/mariadb/shiny/
22. RMariaDB: MariaDB Driver for R, accessed August 20, 2025, https://mariadb.com/docs/connectors/other/rmariadb
23. MariaDB Integration - ETL / ELT, Reverse ETL, Sync - Skyvia, accessed August 20, 2025, https://skyvia.com/connectors/mariadb
24. Overview of BigQuery storage - Google Cloud, accessed August 20, 2025, https://cloud.google.com/bigquery/docs/storage_overview
25. Amazon Redshift | AWS Blog, accessed August 20, 2025, https://aws.amazon.com/blogs/aws/category/amazon-redshift/
26. MariaDB Case Study: IHME, accessed August 20, 2025, https://mariadb.com/wp-content/uploads/2018/10/MariaDb-IHME_Case-Study.pdf
27. Parse JSON in MariaDB: JSON_EXTRACT Guide - Galaxy, accessed August 20, 2025, https://www.getgalaxy.io/learn/glossary/how-to-parse-json-in-mariadb
28. Understanding Spatial Relationships in MariaDB Explained - MoldStud, accessed August 20, 2025, https://moldstud.com/articles/p-understanding-spatial-relationships-in-mariadb-what-why-and-how-explained
29. Open Source Geospatial Database - CrateDB, accessed August 20, 2025,

https://cratedb.com/data-model/geospatial

30. Machine Learning with MindsDB | MariaDB Documentation, accessed August 20, 2025, https://mariadb.com/docs/server/server-usage/storage-engines/machine-learning-with-mindsdb
31. M|18 Real-time Analytics with the New Streaming Data Adapters - SlideShare, accessed August 20, 2025, https://www.slideshare.net/slideshow/m18-realtime-analytics-with-the-new-streaming-data-adapters/89846210
32. BigQuery vs Relational Databases, accessed August 20, 2025, https://www.reddit.com/r/bigquery/comments/pkfzwm/bigquery_vs_relational_databases/
33. MariaDB vs MongoDB: Which One To Choose? - Hevo Data, accessed August 20, 2025, https://hevodata.com/learn/comparison-of-mariadb-vs-mongodb/
34. In-depth Comparison of 7 Leading Data Warehouses and Databases - OWOX BI, accessed August 20, 2025, https://www.owox.com/blog/articles/data-warehouses-comparison
35. Snowflake Competitors: 16 Alternatives Worth Exploring (2025) - Chaos Genius, accessed August 20, 2025, https://www.chaosgenius.io/blog/snowflake-competitors/
36. Redshift vs. BigQuery vs. Snowflake benchmark - Hacker News, accessed August 20, 2025, https://news.ycombinator.com/item?id=15434272