MySQL HeatWave Release Notes

Abstract

This document contains release notes for the changes in each release of MySQL HeatWave.

For additional MySQL HeatWave documentation, see MySQL HeatWave User Guide.

Updates to these notes occur as new product features are added, so that everybody can follow the development process. If a recent version is listed here that you cannot find on the download page (https://dev.mysql.com/downloads/), the version has not yet been released.

The documentation included in source and binary distributions may not be fully up to date with respect to release note entries because integration of the documentation occurs at release build time. For the most up-to-date release notes, please refer to the online documentation instead.

For legal information, see the Legal Notices.

For help with using MySQL, please visit the MySQL Forums, where you can discuss your issues with other MySQL users.

Document generated on: 2026-04-21 (revision: 31090)

Table of Contents

Preface and Legal Notices 2
Changes in MySQL HeatWave 4
Changes in MySQL HeatWave 9.7.0 (2026-04-21) 4
Changes in MySQL HeatWave 9.6.1 (2026-02-17) 5
Changes in MySQL HeatWave 9.6.0 (2026-01-20) 5
Changes in MySQL HeatWave 9.5.2 (2025-12-23) 7
Changes in MySQL HeatWave 9.5.1 (2025-11-18) 7
Changes in MySQL HeatWave 9.5.0 (2025-10-21) 8
Changes in MySQL HeatWave 9.4.2 (2025-09-23) 10
Changes in MySQL HeatWave 9.4.1 (2025-08-25) 10
Changes in MySQL HeatWave 9.4.0 (2025-07-22) 12
Changes in MySQL HeatWave 9.3.2 (2025-06-24) 14
Changes in MySQL HeatWave 9.3.1 (2025-05-27) 15
Changes in MySQL HeatWave 9.3.0 (2025-04-15) 16
Changes in MySQL HeatWave 9.2.2 (2025-03-18) 18
Changes in MySQL HeatWave 9.2.1 (2025-02-20) 19
Changes in MySQL HeatWave 9.2.0 (2025-01-24) 20
Changes in MySQL HeatWave 9.1.2 (2024-12-17) 21
Changes in MySQL HeatWave 9.1.1 (2024-11-19) 22
Changes in MySQL HeatWave 9.1.0 (2024-10-15) 22
Changes in MySQL HeatWave 9.0.1-u1 (2024-09-02) 23
Changes in MySQL HeatWave 9.0.1 (2024-07-23) 25
Changes in MySQL HeatWave 9.0.0 (2024-07-02) 25
Changes in MySQL HeatWave 8.4.0 (2024-04-30) 28
Changes in MySQL HeatWave 8.3.0-u2 (2024-02-26) 28
Changes in MySQL HeatWave 8.3.0 (2024-01-17) 29
Changes in MySQL HeatWave 8.2.0-u2 (2023-12-19) 30
Changes in MySQL HeatWave 8.2.0-u1 (2023-12-05) 30

Changes in MySQL HeatWave 8.2.0 (2023-10-25) 31
Changes in MySQL HeatWave 8.1.0-u4 (2023-09-26) 32
Changes in MySQL HeatWave 8.1.0-u3 (2023-09-12) 33
Changes in MySQL HeatWave 8.1.0-u2 (2023-08-08) 33
Changes in MySQL HeatWave 8.1.0-u1 (2023-07-25) 34
Changes in MySQL HeatWave 8.1.0 (2023-07-18) 34
Changes in MySQL HeatWave 8.0.33-u3 (2023-06-09) 35
Changes in MySQL HeatWave 8.0.33 (2023-04-27) 35
Changes in MySQL HeatWave 8.0.32-u2 (2023-02-21) 36
Changes in MySQL HeatWave 8.0.32-u1 (2023-02-21) 36
Changes in MySQL HeatWave 8.0.32 (2023-01-17) 36
Changes in MySQL HeatWave 8.0.31 (2022-10-11) 37
Changes in MySQL HeatWave 8.0.30-u1 (2022-09-06) 39
Changes in MySQL HeatWave 8.0.30 (2022-07-26) 39
Changes in MySQL HeatWave 8.0.28-u3 (2022-04-19) 41
Changes in MySQL HeatWave 8.0.28-u2 (2022-03-29) 42
Changes in MySQL HeatWave 8.0.28-u1 (2022-02-15) 42
Changes in MySQL HeatWave 8.0.27-u3 (2021-12-15) 42
Changes in MySQL HeatWave 8.0.27-u2 (2021-12-07) 43
Changes in MySQL HeatWave 8.0.26-u2 (2021-09-21) 43
Changes in MySQL HeatWave 8.0.26-u1 (2021-08-10) 44
Changes in MySQL HeatWave 8.0.26 (2021-07-23) 45
Changes in MySQL HeatWave 8.0.25 (2021-05-11) 46
Changes in MySQL HeatWave 8.0.24 (2021-04-20) 47
Changes in MySQL HeatWave 8.0.23-u2 (2021-03-15) 47
Changes in MySQL HeatWave 8.0.23-u1 (2021-02-09) 47
Index 49

Preface and Legal Notices

This document contains release notes for the changes in each release of MySQL HeatWave.

Legal Notices

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support for Accessibility

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Changes in MySQL HeatWave

Changes in MySQL HeatWave 9.7.0 (2026-04-21)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave GenAI

- The `NL_SQL` routine now supports Gemini and OpenAI models. (Bug #38824151)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now enables flexible data ingestion from AVRO files with enhanced support for column re-mapping with the `match_columns_by` parameter. You can load columns into Lakehouse tables by choosing from three column mapping modes: by order, name (case-sensitive), or name (case-insensitive). MySQL HeatWave Lakehouse now also supports selective loading of AVRO files with improved schema inference capabilities. The `allow_missing_columns` parameter lets you disable or enable the ability to include all unique columns by name found during inference. When you enable `allow_missing_columns`, MySQL HeatWave Lakehouse loads all unique columns even if they are not present in every file, and loads data with missing or extra columns without errors. These enhancements provide more control over data loading and ensure seamless integration with both auto schema inference and Guided Load processes, making it easier to load AVRO files with varying column structures.

  For more information, see Column Matching Options and Allow Missing Columns Option. (WL #17195, WL #17196)

MySQL HeatWave

- MySQL HeatWave now supports bulk loading of tables with foreign keys, ensuring that all foreign key constraints are satisfied for each record. This update enables you to successfully load data into tables with foreign key constraints, while also providing error reporting for violated constraints. Although this

added functionality may impact performance due to the extra checks required, it provides a necessary trade-off for supporting tables with foreign keys, making it easier to manage complex data relationships within MySQL HeatWave.

For more information, see Bulk Ingest Data. (WL #17159)

- MySQL HeatWave now supports the tracking of causes for loaded tables becoming stale. With this enhancement, you can easily identify and diagnose issues, and in some cases take corrective action.

  For more information, see The rpd_tables Table. (WL #17002)

Changes in MySQL HeatWave 9.6.1 (2026-02-17)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports enhanced functionality with the introduction of metadata columns and KEY partitioning. This update enables the storage of source information about file names in a special reserved column, `_metadata_filename`, and allows for more efficient data management through KEY partitioning. With this feature, you can create tables with partitioning definitions by using `PARTITION BY KEY (partition_col)`, allowing you to improve data organization and querying capabilities. Additionally, generated column support is now complete for JSON and Delta tables. These enhancements expand the capabilities of MySQL HeatWave Lakehouse, providing you with more flexibility and control over your data.

  For more information, see Create Tables with Metadata Columns and Create Tables with Key Partitioning. (WL #17183, WL #17206)

MySQL HeatWave

- MySQL HeatWave now supports binary data type columns. With this enhancement, you can leverage the benefits of MySQL HeatWave for a wider range of data types, improving overall performance and functionality.

  For more information, see Supported Data Types for MySQL HeatWave. (WL #17204)

Changes in MySQL HeatWave 9.6.0 (2026-01-20)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave AutoML
- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse

- MySQL HeatWave

MySQL HeatWave AutoML

- The `NL2ML` routine includes a new `skip_generate` parameter, which lets you to skip content generation and just return citations, enabling integration with external LLMs or MCP servers for response generation or relevant data reference.

  For more information, see NL2ML and Use NL2ML with In-Database LLMs. (Bug #38397642)

MySQL HeatWave GenAI

- OpenAI and Gemini models are now available in MySQL HeatWave GenAI with OCI Generative AI Service LLMs.

  To learn how to view the list of available models, see View Available Models. (Bug #38752998)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports the loading of complex and nested Parquet types, including lists, maps, and other nested structures, which are flattened into a JSON representation. This enhancement enables you to load any complex type into a JSON column, while still maintaining existing conversions for simple lists to vectors and strings. With this update, Parquet columns of type list, map, or nested/group nodes are inferred as JSON, providing a more comprehensive and flexible data loading experience.

  For more information, see Parquet File Format. (WL #17152)

- MySQL HeatWave Lakehouse now supports load validation of external tables with the introduction of the `VALIDATE ONLY` clause for the `ALTER TABLE ... SECONDARY_LOAD` statement. This enhancement enables you to validate whether a load of a table may succeed without actually loading the data into the secondary engine, providing an additional layer of control and flexibility in managing external tables. The `VALIDATE ONLY` clause can be used to validate a specified number of rows or all rows, and also supports Guided Load options, making it easier to ensure data integrity and optimize loading processes.

  For more information, see Validate Load. (WL #17124)

MySQL HeatWave

- MySQL HeatWave now supports bulk loading into non-empty tables, lifting the previous restriction of only being able to bulk load into empty tables. This update enables users to efficiently merge new data from CSV files with existing data in the table, leveraging a temporary duplicate table and parallel readers to compare and insert records in the appropriate order. With this enhancement, MySQL HeatWave can handle bulk loads of any size, making it a more versatile and powerful tool for managing large datasets.

  For more information, see Bulk Ingest Data. (WL #16441)

- MySQL HeatWave now supports auto materialized view substitution, a powerful internal optimization that enhances query performance. This feature enables the MySQL HeatWave optimizer to automatically substitute a subtree of a query with an existing materialized view, leading to more efficient query execution. With this update, you can expect improved performance without any changes to the external interface, as the way to create MySQL HeatWave materialized views remains the same. This enhancement is designed to work seamlessly in the background, enabling you to benefit from faster query results without requiring any additional configuration or setup.

  For more information, see Automated Substitution of Materialized Views and rapid_auto_create_materialized_view Variable. (WL #17104)

- MySQL HeatWave now supports Grouping Sets, a powerful extension that enables you to specify multiple groupings of data in a single query. This feature provides a concise way to simplify complex queries and improve performance by reducing the number of passes over the data. With Grouping Sets, you can define one or more grouping sets, each containing a comma-separated list of group by columns or expressions, allowing for more flexible and efficient data analysis. This update enhances the capabilities of MySQL HeatWave, making it an even more robust and efficient solution for data processing and analysis.

  For more information, see GROUPING SETS Modifier. (WL #16494)

Changes in MySQL HeatWave 9.5.2 (2025-12-23)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse

MySQL HeatWave GenAI

- MySQL HeatWave GenAI includes a new schema retrieval routine that enables you to find the most relevant tables based on natural-language input. This innovative feature provides a compact and easy-to-read DDL output, focusing on essential elements such as column names, simplified data types, comments, and basic foreign keys, making it ideal for large language model (LLM) prompts and related workflows.

  For more information, see ML_RETRIEVE_SCHEMA_METADATA. (WL #17201)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports the inference and loading of unique columns that are matched by name. With the introduction of the `allow_missing_columns` parameter, you can choose to disable or enable the ability to include all unique columns by name found during inference. When you enable `allow_missing_columns`, MySQL HeatWave Lakehouse loads all unique columns even if they are not present in every file, and loads data with missing or extra columns without errors. This feature enhances the overall data loading experience, providing a more robust and adaptable solution for handling diverse data sets.

  For more information, see Allow Missing Columns Option. (WL #17165)

Changes in MySQL HeatWave 9.5.1 (2025-11-18)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports a new `datetime_format` parameter in JSON Engine Attribute parameters that lets you specify formats for both `TIMESTAMP` and `DATETIME` types. This update aligns the configuration experience with SQL Syntax and enhances overall usability.

For more information, see Dialect Parameters for CSV Files. (WL #17170)

- MySQL HeatWave Lakehouse now supports specifying stored generated columns when using the `CREATE TABLE` statement for Lakehouse tables. This feature was introduced in 9.5.0 for DB System tables, and this release extends the ability to specify stored generated columns for Lakehouse tables as well. Using stored generated columns lets you create tables with computed values based on existing columns or constants. This enhancement enables you to perform complex data transformations on data at the time of loading or refreshing data from Object Storage and storing it as part of the table.

  For more information, see Create Tables with Stored Generated Columns. (WL #16995)

Changes in MySQL HeatWave 9.5.0 (2025-10-21)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave AutoML
- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports enhanced log anomaly detection capabilities with the introduction of semantic feature embeddings. This update enriches the feature space by extracting semantic features from logs using embedding models and combining them with existing statistical TF-IDF features, ultimately improving the accuracy and effectiveness of log anomaly detection.

  For more information, see Anomaly Detection for Logs. (WL #17097)

- MySQL HeatWave AutoML now supports advanced recommendation capabilities with the introduction of Deep Recommendation models, a deep learning-based approach to building embedding models for personalized recommendations. This new feature leverages PyTorch to construct a deep learning pipeline, enabling faster similarity searches between you and products through the generation of user and item embeddings. With this update, MySQL HeatWave AutoML lays the groundwork for delivering highly accurate and scalable personalized recommendations, enhancing the overall user experience. (WL #16656)

MySQL HeatWave GenAI

- MySQL HeatWave GenAI now supports hybrid search, combining the strengths of semantic and keyword-based search. With hybrid search, you can effectively retrieve data that may not be covered by semantic search, such as queries or datasets with specific product names, stock keeping units (SKUs), or brand names, leading to higher quality search results.

  For more information, see Retrieve Context and Generate Content Using Hybrid Search. (WL #16950)

- MySQL HeatWave GenAI now supports automatic creation of Vector Indexes using advanced index structures like Hierarchical Navigable Small World (HNSW) for frequently queried vector columns in vector store and embedding tables, enabling accelerated similarity search queries, and rapid retrieval of relevant results while balancing accuracy and speed.

For more information, see Automatic Vector Index Creation. (WL #16429, WL #16881)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now automatically detects temporal data formats, eliminating the need to manually specify common date and time patterns. This streamlines data onboarding and simplifies integration of temporal data from diverse sources, making your Lakehouse workflows more efficient.

  For more information, see About Lakehouse Auto Parallel Load Schema Inference. (WL #17151)

- MySQL HeatWave Lakehouse now supports enhanced CSV parsing capabilities, allowing you to ignore quote and escape characters by setting their values to empty strings. This update enables the successful loading of CSV files containing JSON columns or special characters, previously resulting in parsing errors. With this improvement, you can efficiently handle complex CSV data, ensuring seamless integration with MySQL HeatWave Lakehouse tables.

  For more information, see Lakehouse External Table SQL Syntax and Lakehouse External Table JSON Syntax. (WL #17074)

- MySQL HeatWave Lakehouse now enables flexible data ingestion from Parquet files with enhanced support for column re-mapping. You can load columns into Lakehouse tables by choosing from three column mapping modes: by order, name (case-sensitive), or name (case-insensitive). This new functionality gives you greater control over the data loading process.

  For more information, see Lakehouse External Table JSON Syntax. (WL #16965)

- MySQL HeatWave Lakehouse now supports reading Delta Lake tables, introducing a new dialect format called "delta". This enhancement enables you to load Delta Lake tables as Lakehouse tables. With this update, MySQL HeatWave Lakehouse provides an efficient way to work with Delta Lake tables, making it easier to manage and analyze data in a Lakehouse environment.

  For more information, see Delta Lake Tables . (WL #16685)

MySQL HeatWave

- MySQL HeatWave now supports bulk loading of tables with generated columns, enhancing its data import capabilities. This update allows for seamless handling of both stored and virtual generated columns, enabling you to efficiently load data from CSV files while leveraging the benefits of generated columns in their database design. With this enhancement, MySQL HeatWave provides improved flexibility and performance for managing complex data sets, making it an even more powerful tool for data analytics and management.

  For more information, see Bulk Ingest Data. (WL #17016)

- MySQL HeatWave now supports materialized views, introducing a new level of efficiency in data management. With this update, you can create materialized views using the `CREATE MATERIALIZED VIEW` statement, allowing for faster query performance and improved data analysis. The materialized view is populated lazily during the execution of a query that references it directly, ensuring that data is always up-to-date and optimized for performance. This new feature enhances the overall functionality of MySQL HeatWave, providing you with more powerful tools to manage and analyze your data. (WL #16886)

- MySQL HeatWave now supports using stored functions inside MySQL HeatWave queries, with a few limitations. This enhancement enables you to execute complex queries with improved performance.

  For more information, see Other Limitations. (WL #16724)

Changes in MySQL HeatWave 9.4.2 (2025-09-23)

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave GenAI
- MySQL HeatWave

MySQL HeatWave GenAI

- Document ingestion using Auto Parallel Load now supports enhanced document parsing for PDF files using OCI Generative AI Service Vision Language Models (VLMs) that are now available in MySQL HeatWave. This enhancement improves the document parsing quality by extracting complex structures like tables, and enhances Retrieval Augmented Generation (RAG) and vector search queries.

  For more information, see Ingest Files Using Auto Parallel Load. (WL #17078)

MySQL HeatWave

- MySQL HeatWave now supports automatic and periodic refreshing of Advanced Cardinality Estimation (ACE) models for loaded tables. With this enhancement, the system periodically attempts to rebuild stale ACE models in the background, reducing the need for manual intervention and improving overall performance. This enhancement allows for more efficient query optimization by ensuring that ACE models are up-to-date, even when tables are modified.

  For more information, see System Variables. (WL #17068)

Changes in MySQL HeatWave 9.4.1 (2025-08-25)

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave AutoML
- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports enhanced data preparation capabilities with `NL2ML`, enabling you to efficiently prepare your data for training and testing. This update introduces additional data preparation information to `NL2ML`'s responses that enable you to join dataset tables, exclude unnecessary columns, and split your data into training and testing tables through the use of a new `TRAIN_TEST_SPLIT` procedure. With these new capabilities, you can streamline your data preparation workflow, making it easier to get started with machine learning tasks.

  For more information, see Prepare Training and Testing Datasets. (WL #17057)

- MySQL HeatWave AutoML now supports enhanced machine learning capabilities with improved performance and robustness in the `ML_PREDICT_TABLE` and `ML_EXPLAIN_TABLE` routines. The new flow loads the entire input table into the ML driver, enabling more ML operations to run in parallel, resulting in better overall performance. Additionally, the update ensures that large input tables are handled efficiently by returning results in blocks, preventing failures due to the `max_allowed_packet`

size limitations. Furthermore, to maintain consistency in log anomaly detection tasks, the input table now requires a primary key, enhancing the reliability of the results.

For more information, see Generate Predictions for a Table and Generate Prediction Explanations for a Table. (WL #16905)

- MySQL HeatWave AutoML and MySQL HeatWave GenAI now support enhanced concurrency, enabling the processing of multiple queries simultaneously across different sessions, which significantly improves overall performance and efficiency. This update allows for a more seamless and responsive experience, making it easier for you to leverage the full potential of MySQL HeatWave AutoML and MySQL HeatWave GenAI for your machine learning and generative AI workloads.

  For more information, see System Variables. (WL #16847)

MySQL HeatWave GenAI

- MySQL HeatWave GenAI now lets you generate SQL queries from natural-language statements using the new `NL_SQL` routine, making it easier for you to interact with databases. This feature collects information on the schemas, tables, and columns that you have access to, and then uses a Large Language Model (LLM) to generate an SQL query for the question pertaining to your data. It also lets you run the generated query and view the result set.

  For more information, see Generate SQL Queries From Natural-Language Statements. (WL #17005, WL #16315)

- MySQL HeatWave GenAI now supports using Oracle Cloud Infrastructure (OCI) Generative AI Service embedding models during vector store ingest using Auto Parallel Load, enabling you to create vector stores using advanced models and improving the accuracy and relevance of results.

  For more information, see Ingest Files Using Auto Parallel Load. (WL #17060)

- MySQL HeatWave GenAI now supports Vision Language Models (VLMs) provided by OCI Generative AI Service, enhancing its capabilities and expanding the range of available AI-powered features. This update provides you with more advanced and efficient processing of vision-language tasks within MySQL HeatWave GenAI.

  For more information, see OCI Generative AI Service LLMs, ML_GENERATE, and ML_GENERATE_TABLE. (WL #17042)

- MySQL HeatWave GenAI now supports advanced document summarization, enabling you to effectively summarize unstructured documents stored in Object Storage.

  For more information, see Summarize Unstructured Files. (WL #16545)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports event-based automatic incremental load on OCI, allowing for seamless and automated data refresh of tables loaded from object storage. With this update, customers can configure a Lakehouse table to enable automated incremental loading of object changes into corresponding Lakehouse tables. This feature provides an efficient and automated way to keep data up-to-date, eliminating the need for manual refreshes. This feature is available for all three HeatWave node shapes: Standard, Small, and Free.

  For more information, see Refresh Data Using Event-Based Incremental Load. (WL #16513)

- MySQL HeatWave Lakehouse now supports an optional clause for the `ALTER TABLE SECONDARY_LOAD` statement, enabling you to control whether MySQL HeatWave Guided Load is

used or not. This enhancement provides flexibility and choice, enabling you to opt-out of Guided Load when loading tables, if desired. With this update, you can specify the `GUIDED {ON | OFF}` clause to determine whether Guided Load should be performed, making it easier to manage your data loading processes.

For more information, see Load Tables . (WL #17062)

MySQL HeatWave

- MySQL HeatWave now supports improved query performance through the introduction of a statistics cache from the `InnoDB` execution engine. This enhancement leverages statistics feedback to optimize query plans, particularly for queries with multiple joins and high MySQL costs. By injecting actual cardinalities from previous query executions into subsequent optimizations, MySQL HeatWave can produce more accurate and efficient plans, leading to significant performance improvements. With this update, MySQL HeatWave delivers enhanced query performance and more efficient use of system resources.

  This feature can be controlled through session and global variables `rapid_enable_my_sc` and `rapid_ap_stats_cache_max_entries`.

  For more information, see System Variables. (WL #16701)

Changes in MySQL HeatWave 9.4.0 (2025-07-22)

Note

MySQL HeatWave User Guide has been restructured and improved to provide better visibility of high-level tasks.

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave AutoML
- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports the recording of hyperparameters of trained models, enhancing the transparency and reproducibility of machine learning processes. This update allows for more detailed insights into model training, enabling better optimization and management of machine learning workflows. With this feature, you can access and utilize hyperparameter information to refine your models, leading to improved performance and accuracy.

  For more information, see Model Metadata. (WL #16946)

- MySQL HeatWave now supports a new feature called *NL2ML*, which provides an intuitive natural-language interface to machine learning (ML). With `NL2ML`, you can ask questions about MySQL HeatWave AutoML and receive step- by-step guidelines on how to use ML for specific business problems. This innovative feature allows you to generate AutoML queries that can be copied and

executed, making it easier to get started with ML. Additionally, *NL2ML* offers a chat-bot interface for follow-up questions and leverages your schemas, tables, and columns to create personalized AutoML queries.

For more information, see Learn About MySQL HeatWave AutoML with Oracle Cloud Infrastructure Generative AI. (WL #16785)

- MySQL HeatWave AutoML now supports enhanced memory management capabilities, enabling more efficient resource utilization. MySQL HeatWave can now check memory usage and release resources based on severity levels, ensuring optimal performance and reliability. This enables better management of memory between MySQL HeatWave GenAI and MySQL HeatWave AutoML. (WL #16759)

- The `rpd_nodes` performance schema table includes new columns that enable you to monitor memory usage by MySQL HeatWave AutoML on MySQL HeatWave nodes.

  For more information, see The rpd_nodes Table. (WL #16761)

MySQL HeatWave GenAI

- MySQL HeatWave GenAI now supports using Dedicated Clusters from OCI Generative AI Service, providing you with more flexibility and capabilities to leverage the power of generative AI. With this addition, MySQL HeatWave GenAI continues to expand its feature set, offering a more comprehensive and robust solution for you to explore and innovate.

  

  Important

  Support for `mistral-7b-instruct-v1` and `llama3-8b-instruct-v1` in-MySQL HeatWave LLMs has now been deprecated.

  MySQL 9.4.1 onwards, if you use any of the deprecated in-MySQL HeatWave LLMs: `llama2-7b-v1`, `mistral-7b-instruct-v1`, or `llama3-8b-instruct-v1`, then MySQL HeatWave GenAI will automatically replace it with `llama3.2-3b-instruct-v1`.

  For more information, see Supported Models and Languages. (WL #16836)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports creation and modification of Lakehouse tables using SQL syntax for setting table options, in addition to the existing way of using JSON strings for setting the `ENGINE_ATTRIBUTE`. This provides another way for you to define and load data from Object Storage for Lakehouse tables.

  For more information, see Lakehouse External Table SQL Syntax. (WL #16837)

- The `CREATE EXTERNAL TABLE` syntax in MySQL HeatWave Lakehouse has been enhanced to provide a more streamlined way of creating tables. The primary and secondary engines for MySQL HeatWave Lakehouse tables are now determined by session variables, eliminating the need to specify engines explicitly.

  For more information, see Lakehouse External Table Syntax. (WL #16831)

MySQL HeatWave

- MySQL HeatWave now supports the use of the `URI` keyword in addition to the `URL` keyword for loading data with Bulk Load and exporting query results.

For more information, see Bulk Ingest Data and Export Query Results to Object Storage. (WL #16988)

- Bulk Load now supports loading tables without primary keys. (WL #16805)
- The syntax for creating MySQL HeatWave temporary tables now uses the `ENGINE=RAPID` option instead of `SECONDARY_ENGINE=RAPID` making `RAPID` the primary engine for these tables. This enhancement also lets you specify the primary engine for MySQL HeatWave temporary tables using the `default_tmp_storage_engine` variable.

  For more information, see Create MySQL HeatWave Temporary Tables. (WL #16898)
- MySQL HeatWave now supports enhanced offload debuggability by introducing significant enhancements, including unified error handling and improved integration with MySQL Shell. This enhancement gives you access to more informative error messages and workarounds, making it easier to diagnose and resolve offload errors. (WL #16409)

Changes in MySQL HeatWave 9.3.2 (2025-06-24)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave AutoML
- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports drift detection for anomaly detection, enabling you to identify changes in data distribution. With this update, you can review the model metadata and enable the `additional_details` option with the `ML_PREDICT_TABLE` and `ML_PREDICT_ROW` routines to get valuable insights into data drift, enhancing the overall anomaly detection experience.

  For more information, see Analyze Data Drift. (WL #16368)

MySQL HeatWave GenAI

- Content generation using MySQL HeatWave GenAI now supports speculative decoding, which enables faster response token generation and speeds up text generation if the target Large Language Model (LLM) supports speculative decoding.

  For more information, see ML_GENERATE, ML_GENERATE_TABLE, and @chat_options Parameters. (WL #16809)
- Document ingestion using Auto Parallel Load has been enhanced to support customised segmentation of text during vector store creation. This enhancement lets you have greater control over how text is segmented, enabling more precise and tailored text analysis capabilities.

  For more information, see Ingest Files Using Auto Parallel Load. (WL #16683)
- MySQL HeatWave GenAI now supports automated discovery and listing of all available LLMs and embedding models in MySQL HeatWave. This enhancement lets you stay up-to-date on model

availability, and enables you to make informed usage decisions especially in the case of OCI Generative AI Service models as the list of models available in MySQL HeatWave might change before a new version of MySQL is available.

For more information, see Supported Models and Languages. (WL #16664)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports the direct loading of compressed files from Object Storage into tables, enhancing data ingestion capabilities and simplifying data pipelines. This update lets you load `gzip`, `zip`, and `bzip2` compressed files that contain CSV and JSON data into MySQL HeatWave. This enhancement streamlines the process of working with compressed data, making it more efficient and cost effective for you to manage and analyze your data within MySQL HeatWave Lakehouse.

  For more information, see Lakehouse External Table Syntax. (WL #16650)

- MySQL HeatWave Lakehouse now supports loading vector columns from CSV and Parquet files, enabling you to bring your own custom vector embeddings into MySQL HeatWave and leverage MySQL HeatWave GenAI features. With this enhancement, you can optimize and manage semantic search capabilities for your datasets and integrate with the large ecosystem of GenAI capabilities.

  For more information, see Supported File Formats and Data Types. (WL #16563)

Changes in MySQL HeatWave 9.3.1 (2025-05-27)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave GenAI

- MySQL HeatWave GenAI is now available on all MySQL HeatWave Cluster shapes, including the `HeatWave.Free` shape.

  Also, the following in-database LLMs are now available in MySQL HeatWave GenAI: `llama3.2-3b-instruct-v1`, `llama3.2-1b-instruct-v1`, `llama3.1-8b-instruct-v1`, and `mistral-7b-instruct-v3`. With the introduction of new LLMs, MySQL HeatWave GenAI now uses `llama3.2-3b-instruct-v1` as the default LLM for content generation instead of `mistral-7b-instruct-v1`.

  For more information, see Supported Models and Languages. (Bug #37551135, WL #16509)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports seamless loading of highly-encoded Parquet files, including those with sparse data, highly repetitive values, and consecutive values. (WL #16812)

- MySQL HeatWave Lakehouse now supports unified loading of files using different URI formats. With this enhancement, you can load files from Object Storage using any of the three URI formats: OCIFS URI, PAR URI, and native URI. These URIs support patterns, prefixes, or names, which makes it easier to

work with large datasets. Additionally, error and warning messages have been improved to include the relevant URI information for better troubleshooting.

For more information, see Access Object Storage with Uniform Resource Identifiers (URI). (WL #16822)

- MySQL HeatWave now supports exporting query results to Object Storage as Newline Delimited-JSON (ND-JSON) files. Using ND-JSON files lets you export query results as line-separated JSON logs, which enables easy and efficient integrations of your data with other web applications. This enhancement also makes it easier to load the exported results into a table with a single column of `JSON` data type, or use the `HEATWAVE_LOAD` routine to load the exported results into MySQL HeatWave Cluster for further processing.

  For more information, see Export Query Results to ND-JSON Files. (WL #16778)

- MySQL HeatWave now supports `MEDIUMTEXT`, `LONGTEXT`, and `JSON` data types with a column width of up to 4MB for Lakehouse tables. With this enhancement, you can leverage the expanded capabilities of MySQL HeatWave Lakehouse to load and manage larger datasets, boosting overall performance and productivity.

  For more information, see Lakehouse Limitations for all File Formats. (WL #16472)

MySQL HeatWave

- MySQL HeatWave now supports automated partition management, extending the Auto Parallel Load feature of automatically loading and unloading only the necessary partitions in partitioned tables based on usage. This enhancement accelerates workloads on partitioned tables while offering memory savings, allowing for more efficient data management and improved performance.

  For more information, see Automatic Loading and Unloading of DB System Tables and Partitions. (WL #15994)

- MySQL HeatWave now supports creating asynchronous tasks for long-running queries and commands, allowing them to run in the background as separate tasks. This enhancement enables you to run complex queries without blocking other operations, thus enhancing the overall performance and usability of MySQL HeatWave.

  For more information, see Run Tasks Asynchronously. (WL #16539)

Changes in MySQL HeatWave 9.3.0 (2025-04-15)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

Important

As of MySQL 9.3.0, to help you generate better quality embeddings, MySQL HeatWave GenAI uses `multilingual-e5-small` as the default embedding model for encoding documents in all supported languages including English. This means that `minilm` is no longer used as the default embedding model for encoding English documents.

This default value change impacts the following processes:

- New vector store tables are created using `multilingual-e5-small` by default.

- Retrieval augmented generation (RAG) searches tables created using the default embedding model, `multilingual-e5-small`, unless you explicitly specify the embedding model to use.

- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave GenAI

- MySQL HeatWave GenAI now supports partial failures with better error reporting in batch queries that are run using the `ML_GENERATE_TABLE`, `ML_RAG_TABLE`, and `ML_EMBED_TABLE` routines. This enhancement allows queries on certain rows to fail in case of errors without discarding successful work on other rows. (WL #16749)

MySQL HeatWave Lakehouse

- MySQL HeatWave now supports automatic recovery of loaded Lakehouse tables from the MySQL HeatWave Storage Layer after a planned or unplanned restart of the primary DB system. When a DB system restarts, data for Lakehouse tables is loaded from the MySQL HeatWave Storage Layer in Object Storage. After a successful load, the data is current as of the last refresh of the table. If this operation fails, data for these Lakehouse tables is then loaded from the Object Storage bucket.

  This feature extends the existing functionality of automatic recovery from Storage Layer for Standalone DB systems to High Availability DB systems, thus reducing the recovery time, increasing the availability, and improving the overall system reliability.

  For more information, see About MySQL HeatWave. (WL #16758)

MySQL HeatWave

- MySQL HeatWave now supports the creation of temporary tables, which are stored in a hybrid columnar format within the MySQL HeatWave Cluster. If you use temporary tables to store intermediate results during data transformation, aggregation or consolidation for data analysis or reporting purposes, you can now accelerate the processes with MySQL HeatWave.

  For more information, see Create MySQL HeatWave Temporary Tables. (WL #16541)

- Bulk Load, which is used for ingesting data into the DB System, now supports tables that include the `VECTOR` data type. This enables you to easily and quickly import existing or pre-generated embeddings to MySQL HeatWave for your MySQL HeatWave GenAI workloads.

  For more information, see Bulk Ingest Data Type Support (WL #16510)

- MySQL HeatWave now supports the use of the `ANALYZE TABLE` statement to update statistics for tables loaded into the MySQL HeatWave Cluster. This enhancement improves query performance by refreshing critical statistics used during query processing. In mixed or Lakehouse workloads, where changing data characteristics can degrade performance, this feature helps maintain optimal query execution by keeping statistics up-to-date.

  For more information, see Analyze Tables. (WL #16641)

- MySQL HeatWave now supports acceleration of queries with general quantified comparison predicates.

For more information, see Optimizer Notes and Optimizing ANY and ALL Subqueries. (WL #13052)

Changes in MySQL HeatWave 9.2.2 (2025-03-18)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

Important

MySQL 9.3.0 onwards, to help you generate better quality embeddings, MySQL HeatWave GenAI will use `multilingual-e5-small` as the default embedding model for encoding documents in all supported languages including English. This means that `minilm` will not be used as the default embedding model for encoding English documents.

This default value change will impact the following processes:

- New vector store tables will be created using `multilingual-e5-small` by default.
- Retrieval augmented generation (RAG) will search tables created using the default embedding model, `multilingual-e5-small`, unless you explicitly specify the embedding model to use.

- MySQL HeatWave AutoML
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports log anomaly detection, a new task that enables you to identify unusual patterns in log data. This feature is designed to help you detect and investigate potential issues more efficiently.

  For more information, see Anomaly Detection for Logs. (WL #16579)

MySQL HeatWave Lakehouse

- A new table state, `RECOVERYFAILED_RPDGSTABSTATE`, has been introduced for Lakehouse tables. During a MySQL HeatWave Cluster restart, if a Lakehouse table cannot be recovered, its state is set to `RECOVERYFAILED_RPDGSTABSTATE` to indicate that it could not be recovered. The cluster remains usable even if recovery of Lakehouse tables fail. Tables in the `RECOVERYFAILED_RPDGSTABSTATE` state cannot be used for queries or change propagation. You must reload tables in this state by running `SECONDARY_UNLOAD` followed by `SECONDARY_LOAD`, which lets you identify errors and take corrective action.

  For more information, see The rpd_tables Table. (WL #16626)

- Lakehouse now supports detailed warnings summary for load and inference operations. The summary is included in the warning messages, providing an accurate list of warning codes and their respective

counts. This enhancement helps you better understand and manage warnings encountered during Lakehouse operations.

For more information, see MySQL HeatWave Lakehouse Error Messages. (WL #16637)

MySQL HeatWave

- MySQL HeatWave now supports columns wider than 64 KB, with a new maximum limit of 4 MB.

  For more information, see Column Limits. (WL #16289)

Changes in MySQL HeatWave 9.2.1 (2025-02-20)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave GenAI
- MySQL HeatWave

MySQL HeatWave GenAI

- Context search in MySQL HeatWave GenAI now lets you use your own embedding tables. This enhancement enables more flexible query enrichment for Large Language Models (LLMs). This update extends the search capabilities of the `ML_RAG`, `ML_RAG_TABLE`, and `HEATWAVE_CHAT` routines to include your own embedding tables.

  For more information, see Use Your Own Embeddings and Run MySQL HeatWave Chat. (WL #16537)

MySQL HeatWave

- MySQL HeatWave now supports automatic recovery of loaded InnoDB tables from Storage Layer after a planned or unplanned restart of the primary DB system. This feature extends the existing functionality of automatic recovery from Storage Layer for Standalone DB systems to HA DB systems, thus reducing the recovery time, increasing the availability, and improving the overall system reliability. However, Lakehouse tables in HA DB systems still need to be recovered from Object Storage.

  For more information, see About MySQL HeatWave. (WL #16627)

- MySQL HeatWave now supports automated zone map construction on non-primary key columns, improving RAPID query processing performance. This feature uses a workload-driven approach to select the most beneficial zone map columns based on query history, continuously adapting to the workload.

  For more information, see About MySQL HeatWave. (WL #15993)

- The `POOL_TYPE` column in MySQL HeatWave Performance Schema tables no longer includes the `RAPID_LOAD_POOL_` prefix. The possible values are now `SNAPSHOT` and `TRANSACTIONAL`.

  For more information, see The rpd_tables Table. (WL #16630)

- For tracking the usage of various components, MySQL HeatWave Option tracking now supports a single, fast counter, `usedCounter`, instead of a boolean value, `used`. The counter indicates the number of times a component is used.

  For more information, see Option Tracker. (WL #16721, WL #16732)

Changes in MySQL HeatWave 9.2.0 (2025-01-24)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave GenAI

- The MySQL HeatWave GenAI Javascript API now supports batch processing, making it possible to perform multiple operations in a single call. This update introduces batch processing functionality to the `LLM` class and API convenience methods (see Convenience Methods), allowing for more efficient embedding, text-based content generation, and Retrieval Augmented Generation (RAG).

  For more information, see JavaScript GenAI API. (WL #16514)

MySQL HeatWave Lakehouse

- Lakehouse Autopilot can now automatically detect the dialect for CSV files, which lets you load the CSV files without knowing the exact values of the `dialect` delimiter parameters in advance. This feature automates the detection of record delimiters, field delimiters, and composite record delimiters with field delimiters as prefixes.

  For more information, see Lakehouse External Table Syntax. (WL #16546)

MySQL HeatWave

- Bulk load, which is used for ingesting data into MySQL Server, now supports the following:

  - Secondary indexes, which enable faster data loading and improved performance. This feature allows for the creation of secondary indexes during the bulk load process, reducing the need for additional `ALTER TABLE` statements.
  - Faster and more efficient bulk loading of CSV files. This enhancement allows for parallelized CSV parsing, reducing the need for redundant reads and improving overall performance.

  For more information, see Bulk Ingest Data. (WL #16266, WL #15986)

- MySQL HeatWave now supports JSON tables, enabling you to extract data from JSON documents and return it as relational tables.

  For more information, see JSON Functions. (WL #16450)

- MySQL HeatWave now supports offloading the `INSERT INTO SELECT` queries to the secondary engine, even when the table being inserted into has triggers defined on it. This enables faster query execution and improved performance for workloads that involve inserting data into tables with triggers.

  For more information, see INSERT ... SELECT Statements. (WL #16235)

- MySQL HeatWave now reloads data from Storage Layer after a planned or unplanned restart of Standalone DB system, thus reducing the recovery time and improving overall system reliability.

For more information, see About MySQL HeatWave. (WL #14953)

Changes in MySQL HeatWave 9.1.2 (2024-12-17)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave AutoML
- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports more efficient memory usage, reducing the footprint of its processes during training and explanations for large datasets. (WL #16606)

MySQL HeatWave GenAI

- MySQL HeatWave GenAI supports enhanced unstructured document ingestion with Optical Character Recognition (OCR) now enabled by default. (WL #16584)
- The `ML_RAG` and `ML_RAG_TABLE` routines now supports advanced retrieval options, enabling more precise and context-aware results. Three new parameters have been added to the routines:
  - `max_distance` for filtering segments based on distance from the input query.
  - `percentage_distance` for adaptive distance filtering of segments from the input query.
  - `segment_overlap` for retrieving segments adjacent to the segment nearest to the input query for providing continuous context.

  These new parameters are integrated into the existing `@options` parameter, and can be enabled through a new JSON object parameter, `@retrieval_options`. (WL #16543)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports selective load, which lets you update the file paths comprising a Lakehouse table by manipulating Lakehouse metadata. The changes are incrementally applied to the table data. This feature enables you to add or remove Object Storage bucket data directories from a Lakehouse table without reloading the entire table. Changes are executed transactionally, ensuring data consistency and query isolation.

  For more information, see Add or Remove Files Using Selective Load. (WL #16531)
- MySQL HeatWave Lakehouse now supports enhanced error and warning messages, providing more data points for easier debugging. This update adds row numbers and character offsets to relevant warning messages in non-strict mode during secondary load, as well as updates byte offset text messages to point to column positions. JSON datatype error and warning messages have also been improved with error and warning offset information within the column.

  For more information, see MySQL HeatWave Lakehouse Error Messages. (WL #16530)

- MySQL HeatWave Lakehouse now supports efficient loading of Parquet files with large row groups (larger than 500MB and upto 10GB) using large shapes. This helps in overcoming previous memory budget constraints and significantly reducing load times. This enhancement also introduces row group sharing, caching, and synchronization mechanisms to minimize network requests, decoding time, and memory usage during transformation.

  For more information, see Lakehouse Limitations for the Parquet File Format. (WL #16512)

Changes in MySQL HeatWave 9.1.1 (2024-11-19)

Note

These release notes were created with the assistance of MySQL HeatWave GenAI.

- MySQL HeatWave AutoML
- MySQL HeatWave Lakehouse

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports the Orbit forecasting model, which uses Local Global Trend (LGT). This addition expands the scope of predictive analytics and enables more advanced time-series forecasting for businesses.

  For more information, see Generate Forecasts. (WL #16231)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports exporting query results directly to an object store in OCI or AWS. This lets you store, transform, and persist data in CSV and Parquet formats. The new syntaxes introduced make it easier to export data, especially in command-line environments, and provide more control over the output format. This enhancement enables efficient data storage, retrieval, and transformation, improving data management and analysis workflows.

  For more information, see Export Query Results to Object Storage. (WL #16214)

Changes in MySQL HeatWave 9.1.0 (2024-10-15)

- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave GenAI

- MySQL HeatWave GenAI now lets you extract and encode text from images stored in unstructured documents using Optical Character Recognition (OCR). The extracted text is converted into vector embeddings and stored in a vector store the same way regular text in unstructured documents is encoded and stored in a vector store.

  For more information, see About Optical Character Recognition. (WL #16474)

- MySQL JavaScript Stored Programs now include a new GenAI API that you can use to call different MySQL HeatWave GenAI routines using JavaScript functions.

For more information, see JavaScript GenAI API. (WL #16487)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports zone maps, which helps you improve range queries in OLAP and mixed workloads.

  MySQL HeatWave uses statistics based on minimum and maximum values to create zone maps for every primary key column. It then uses the zone maps for range and point queries to only scan data chunks that are relevant for the query, and accelerates these queries by an order of magnitude. (WL #16414)

MySQL HeatWave

- Auto Parallel Load now uses Autopilot to collect statistics about frequently used InnoDB and MySQL HeatWave tables. Auto Parallel Load then automatically loads these tables to MySQL HeatWave. If a manually loaded table might cause a memory conflict, Auto Unload unloads automatically loaded tables to free up memory. (WL #15491)

- MySQL HeatWave now supports InnoDB partitions, and `INFORMATION_SCHEMA.PARTITIONS` includes a `SECONDARY_LOAD` column that indicates whether a partition has been loaded.

  When an unload operation successfully unloads the last partition, it unloads the whole table.

  MySQL HeatWave now persists partition information, and during a restart, only the previously loaded partitions are loaded.

  For more information, see Load Partitions (WL #15517, WL #16268)

Changes in MySQL HeatWave 9.0.1-u1 (2024-09-02)

- MySQL HeatWave AutoML
- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports semi-supervised learning for anomaly detection. This type of machine learning algorithm uses a specific set of labeled data along with unlabeled data to detect anomalies. (WL #16373)

- MySQL HeatWave AutoML now supports topic modeling. This is an unsupervised machine learning technique that is capable of scanning a set of documents, detecting word and phrase patterns within them, and automatically clustering word groups and similar expressions that best characterize the documents. (WL #16275)

MySQL HeatWave GenAI

- MySQL HeatWave GenAI now supports multiple languages, which lets you use the MySQL HeatWave GenAI APIs in non-English languages. This includes ingestion of documents written in languages other than English, vector and similarity search, and querying these documents by giving prompts in the same language.

For more information, see Languages. (WL #16475)

- MySQL HeatWave GenAI now supports large language model (LLM) inference batch processing, which lets you run the MySQL HeatWave GenAI routines on multiple queries, in parallel, across different nodes in the MySQL HeatWave Cluster. Thus, it improves the LLM inference performance and throughput while keeping the inference latency of every request unchanged.

  For more information, see ML_GENERATE_TABLE, ML_RAG_TABLE, and ML_EMBED_TABLE. (WL #16399)

- The MySQL HeatWave GenAI `ML_RAG` routine now includes new filtering options, `exclude_vector_store` and `exclude_document_name`, which let you exclude specific vector store tables or documents from context retrieval. (WL #16401)

- You can now use incremental load to refresh vector store tables. For more information, see Load Data Incrementally into the Vector Store Table. (WL #15817)

- Auto Parallel Load for unstructured data provides a new `format` option setting, `auto_unstructured`, which lets you ingest multiple files with different unstructured data formats in a single load.

  Additionally, for ingesting Microsoft PowerPoint and Microsoft Word files into the vector store, Auto Parallel Load introduces the following new format aliases: `pptx` and `docx`.

  For more information, see Ingest Files Using Auto Parallel Load. (WL #16433)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports `timestamp_format` as a `dialect` parameter to customize the format for columns of the `DATETIME` and `TIMESTAMP` data types. You can also set the formats for specific columns by using the `ENGINE_ATTRIBUTE` option. Formats set at the column level override formats set with the `dialect` parameter. For more information, see Lakehouse External Table Syntax. (WL #16425)

- Some MySQL HeatWave Lakehouse error messages include the URL to the external object. The URL can contain up to 1024 characters, and exceed the error message limit of 512 characters. The truncated error message will now include a MySQL command to access the full error message.

  It is now possible to filter out MySQL HeatWave Lakehouse warning messages. The console **Total Warnings** will include both the displayed and the filtered warnings count. See: MySQL HeatWave Lakehouse Error Messages.

  MySQL HeatWave Lakehouse now supports `max_error_count`. (WL #16141)

MySQL HeatWave

- • `GROUP_CONCAT()` function now supports the `CUBE`, `ROLLUP`, and `WITH ROLLUP` options.

  • `COUNT(DISTINCT)` function now supports the `CUBE`, `ROLLUP`, and `WITH ROLLUP` options. (WL #16365)

- MySQL HeatWave now supports the following JSON functions:

  - `JSON_INSERT()`
  - `JSON_REMOVE()`
  - `JSON_REPLACE()`

- `JSON_SET()`
- `JSON_ARRAY_APPEND()`
- `JSON_ARRAY_INSERT()`
- `JSON_MERGE_PATCH`
- `JSON_MERGE_PRESERVE`
- `JSON_MERGE()`
- `JSON_SCHEMA_VALID()`
- `JSON_SCHEMA_VALIDATION_REPORT()`

(WL #16345)

Changes in MySQL HeatWave 9.0.1 (2024-07-23)

This release does not include any new features.

Changes in MySQL HeatWave 9.0.0 (2024-07-02)

- MySQL HeatWave AutoML
- MySQL HeatWave Autopilot
- MySQL HeatWave GenAI
- MySQL HeatWave Lakehouse
- MySQL HeatWave
- Functionality Added or Changed

MySQL HeatWave AutoML

- MySQL HeatWave AutoML has increased the maximum model size from 900MB. The MySQL HeatWave cluster shape you set for a DB system now defines the total memory available to train a model and for all loaded models. For imported models, we recommend individual models have a size of 4GB or less. (WL #15931)

MySQL HeatWave Autopilot

- MySQL HeatWave Autopilot includes improvements to cardinality estimation for queries that run in MySQL HeatWave and MySQL HeatWave Lakehouse. This improves the query plan, and provides improved performance (WL #15995)
- MySQL HeatWave Autopilot now includes Auto Indexing Advisor that offers indexing suggestions to improve the performance and latency of OLTP workloads running on the MySQL HeatWave primary engine. (WL #15663)

MySQL HeatWave GenAI

- MySQL HeatWave GenAI introduces similarity search capabilities that support the `DISTANCE` function that measures the distance between two vector values. It can include an optional `distance_metric`

that can have a value of `DOT`, `COSINE` or `EUCLIDIAN`. This is available in the primary and secondary engines. (WL #16081)

- MySQL HeatWave GenAI introduces the new native `VECTOR` data type and in-database vector store support. This provides a fast, end-to-end, fully integrated pipeline which automates the vector store creation:
  - Reading unstructured data in PDF, HTML, TXT, PPT or DOCX format from object store.
  - Parsing the text in the documents.
  - Partitioning the text into smaller paragraphs and segments.
  - Encoding the paragraphs.
  - Storing the encoded paragraphs in a standard MySQL table in MySQL HeatWave.

  (WL #16106)
- MySQL HeatWave GenAI introduces a set of intuitive chat functionalities that enable enterprises to quickly build their own AI powered chatbot. It also introduces an out-of-the-box chatbot – MySQL HeatWave Chat, which integrates the chat functionalities that enable users to interact with MySQL HeatWave using natural language in addition to SQL. (WL #16248)
- MySQL HeatWave GenAI introduces in-database LLM with support for the following LLMs, see Perform AI-Powered Search and Content Generation:
  - Mistral-7B-Instruct.
  - Llama-2-8B-Instruct.
  - LLMs from OCI GenAI.

  (WL #15844, WL #16145)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports unstructured documents, and loads them to the GenAI vector store. Supported formats include PDF, HTML, TXT, PPT and DOCX. (WL #16100)
- MySQL HeatWave Lakehouse introduces Lakehouse Incremental Load that uses change propagation to refresh data views following changes to a loaded table. (WL #15817)

MySQL HeatWave

- MySQL HeatWave includes improvements to query optimization that lead to better performance for low latency queries in MySQL HeatWave and MySQL HeatWave Lakehouse. (WL #16222)
- MySQL HeatWave includes improvements to change propagation performance. (WL #16148)
- MySQL HeatWave improves the debugging of offloaded queries. (WL #15988)
- MySQL HeatWave now supports the use of cursors from stored procedures. (WL #16142)
- MySQL HeatWave now has support for the `JSON` data type with the following functions:
  - Cast Functions and Operators.
  - `COALESCE()` and `IN()`, see: Comparison Functions and Operators.

- Control Flow Functions and Operators.
- Mathematical Functions.
- String Functions and Operators.

(WL #16048)

- MySQL HeatWave now supports the following JSON functions:
  - `JSON_ARRAYAGG()`
  - `JSON_CONTAINS()`
  - `JSON_CONTAINS_PATH()`
  - `JSON_KEYS()`
  - `JSON_OBJECTAGG()`
  - `JSON_OVERLAPS()`
  - `JSON_PRETTY()`
  - `JSON_QUOTE()`
  - `JSON_SEARCH()`
  - `JSON_STORAGE_FREE()`
  - `JSON_STORAGE_SIZE()`
  - `JSON_TYPE()`
  - `JSON_VALID()`
  - `JSON_VALUE()`
  - `MEMBER OF()`

  (WL #16046, WL #16047)

- MySQL HeatWave now includes support for `GROUP_CONCAT()` with `DISTINCT` optimization and the `ORDER BY` clause. (WL #15906)
- MySQL HeatWave now supports dynamic query offload which uses machine learning based on query properties to determine the best engine, InnoDB or MySQL HeatWave analytic engine, to process the query. (WL #15829)
- MySQL HeatWave now supports concurrent query execution that reduces query latencies, and increases resource utilization. (WL #14826)

Functionality Added or Changed

- MySQL InnoDB on MySQL HeatWave includes a bulk load extension to the `LOAD DATA` statement. This now uses multiple threads to load a series of CSV files to improve performance, and allow the LOAD operation to scale with multiple CPUs. (WL #15611)

- MySQL InnoDB on MySQL HeatWave includes a bulk load extension to the `LOAD DATA` statement. This now supports large objects with `VARCHAR` and `VARBINARY`. It also supports the following data types:
  - `BIT`.
  - `ENUM`.
  - `JSON`.
  - `SET`.
  - `TIMESTAMP`.
  - `YEAR`.
  - `TINYBLOB`, `BLOB`, `MEDIUMBLOB`, and `LONGBLOB`.
  - `TINYTEXT`, `TEXT`, `MEDIUMTEXT`, and `LONGTEXT`.
  - `GEOMETRY`, `GEOMETRYCOLLECTION`, `POINT`, `MULTIPOINT`, `LINESTRING`, `MULTILINESTRING`, `POLYGON`, and `MULTIPOLYGON`,

  See: Bulk Ingest Data. (WL #16264, WL #16265)

Changes in MySQL HeatWave 8.4.0 (2024-04-30)

- MySQL HeatWave Lakehouse
- Functionality Added or Changed

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports primary key and unique key constraint validation with the `check_constraints` parameter. When you set this parameter to `true`, a validation for primary key and unique key constraints occurs only during the initial load of a table. (WL #16094)
- MySQL HeatWave Lakehouse now supports the Newline Delimited JSON file format, see Supported File Formats and Data Types. (WL #16079)
- MySQL HeatWave Lakehouse now has improved support for missing files. Auto Parallel Load now has a `validation` mode to validate files before loading them. (WL #16078)

Functionality Added or Changed

- MySQL HeatWave now supports named time zones such as `MET` or `Europe/Amsterdam`. (WL #16098)
- MySQL HeatWave Auto Parallel Load and Auto Unload now support a table inclusion list, as well as a table exclusion list. (WL #16070)
- MySQL InnoDB on MySQL HeatWave includes a bulk load extension to the `LOAD DATA` statement. This can now load MySQL Shell dump files and compressed files. There is also a progress monitor. See: Bulk Ingest Data. (WL #15814, WL #15607, WL #15608, WL #15609)

Changes in MySQL HeatWave 8.3.0-u2 (2024-02-26)

- MySQL HeatWave AutoML

- MySQL HeatWave Lakehouse
- Functionality Added or Changed

MySQL HeatWave AutoML

- MySQL HeatWave AutoML adds prediction intervals to the forecasting task. (WL #16072)
- MySQL HeatWave AutoML adds the PCA and GLOF algorithms to the anomaly detection task. (WL #16104)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports the JSON data type. (WL #16037)
- MySQL HeatWave Lakehouse now supports Guided Load to perform pre-load validation checks and schema inference. (WL #16004)
- MySQL HeatWave Lakehouse now supports the following:
  - Point-in-time-recovery.
  - High Availability.
  - Read Replication.
  - Outbound Replication.

  (WL #15926)
- MySQL HeatWave Lakehouse now supports Auto Data Compression. This dynamically determines which compression algorithm to use for each column based on its data characteristics. This provides the optimum balance between memory usage and query performance. See: About MySQL HeatWave. (WL #16061)

Functionality Added or Changed

- MySQL HeatWave primary and secondary engines now support the HyperLogLog, `HLL()` aggregate function. This is similar to `COUNT(DISTINCT)`, but faster and with user defined precision. This is only available in MySQL HeatWave, see: Aggregate Functions. (WL #15992)
- MySQL HeatWave can now reload all tables, and unload all tables. (WL #16023)
- MySQL HeatWave secondary engine now supports the `TABLESAMPLE` clause to use with `SELECT` statements. (WL #15902)

Changes in MySQL HeatWave 8.3.0 (2024-01-17)

- MySQL HeatWave Lakehouse
- Functionality Added or Changed

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse extends Auto Parallel Load support for external tables that have a pre-defined schema. It is now able to reconcile the inferred schema with the pre-defined table definition when there is a column type mismatch between the two. Schema inference adjusts this schema to

prevent possible errors during data loading. See: Use Lakehouse Auto Parallel Load with external_tables Option. (WL #16040)

- MySQL HeatWave Lakehouse now has improved error reporting. See: MySQL HeatWave Lakehouse Error Messages. (WL #15951)

Functionality Added or Changed

- `GREATEST()` and `LEAST()` comparison and string functions now support the `YEAR` data type. (WL #15910)

Changes in MySQL HeatWave 8.2.0-u2 (2023-12-19)

- MySQL HeatWave AutoML
- MySQL HeatWave Lakehouse

MySQL HeatWave AutoML

- The MySQL HeatWave AutoML recommendation task now supports content based recommendation models. See Generate Recommendations. (WL #15726)
- MySQL HeatWave AutoML introduces data drift detection for classification and regression models. (WL #15866)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now supports all queries that MySQL HeatWave supports. See: MySQL HeatWave Lakehouse Limitations. (WL #15730)

Changes in MySQL HeatWave 8.2.0-u1 (2023-12-05)

- MySQL HeatWave Lakehouse
- Functionality Added or Changed

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now uses Auto Parallel Load to load Avro files. (WL #15778)
- MySQL HeatWave Lakehouse schema inference now includes support for unicode column headers. (WL #16011)

Functionality Added or Changed

- MySQL HeatWave secondary engine now supports the `QUALIFY` clause to use with `SELECT` statements. (WL #15864)
- Auto Data Compression dynamically determines which compression algorithm to use for each column based on its data characteristics. This provides the optimum balance between memory usage and query performance. See: About MySQL HeatWave. (WL #15248)
- MySQL HeatWave now supports the following aggregate functions used as window functions:
  - `STD()`
  - `STDDEV()`

- `STDDEV_POP()`
- `STDDEV_SAMP()`
- `VAR_POP()`
- `VAR_SAMP()`
- `VARIANCE()`

(WL #15367)

- MySQL HeatWave secondary engine now supports the HyperLogLog, `HLL()` aggregate function. This is similar to `COUNT(DISTINCT)`, but faster and with user defined precision. This is only available in MySQL HeatWave, see: Aggregate Functions. (WL #15914)
- The `GROUP BY` clause permits a `CUBE` modifier. This is only available in MySQL HeatWave, see: GROUP BY Modifiers . (WL #15843)
- MySQL HeatWave now supports the following JSON functions:
  - `->`
  - `->>`
  - `JSON_ARRAY()`
  - `JSON_DEPTH()`
  - `JSON_EXTRACT()`
  - `JSON_LENGTH()`
  - `JSON_OBJECT()`
  - `JSON_UNQUOTE()`

  See: JSON Functions. (WL #15355)

Changes in MySQL HeatWave 8.2.0 (2023-10-25)

Functionality Added or Changed

- MySQL HeatWave Guided Load reduces the number of steps to manually load data, see Load Data Manually. (WL #15811)
- MySQL HeatWave now supports the following string functions:
  - `BIN()`
  - `BIT_LENGTH()`
  - `ELT()`
  - `EXPORT_SET()`
  - `FIELD()`

  - `MAKE_SET()`

  These functions do not support `ENUM` type columns, see Data Type Limitations. (WL #15628)

- MySQL HeatWave now supports full-table aggregation and group-by support for the following temporal data types:

  - `DATE`
  - `TIME`
  - `DATETIME`
  - `TIMESTAMP`
  - `YEAR`

  MySQL HeatWave now supports the following functions with temporal data types:

  - `SUM()`
  - `AVG()`
  - `STD()`
  - `STDDEV()`
  - `STDDEV_POP()`
  - `STDDEV_SAMP()`
  - `VAR_POP()`
  - `VAR_SAMP()`
  - `VARIANCE()`

  See: Aggregate Functions. (WL #15637)

- MySQL HeatWave now enables the internal conversion of values from one data type to another, including all combinations of these different encodings:

  - Numeric values `INTEGER`, `FLOAT`, and `DECIMAL`.
  - String values, excluding dictionary-encoded strings.
  - Temporal values `DATE`, `TIME`, `DATETIME`, and `YEAR` data types.

  This now permits MySQL HeatWave to offload queries that contain functions with unexpected data types by adding an internal cast from the unexpected data type to an expected data type. (WL #15465)

Changes in MySQL HeatWave 8.1.0-u4 (2023-09-26)

- MySQL HeatWave AutoML
- MySQL HeatWave Lakehouse
- Functionality Added or Changed

MySQL HeatWave AutoML

- The MySQL HeatWave AutoML recommendation task now supports implicit feedback. See Generate Recommendations. (WL #15802)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse now uses schema inference to create an external table, see Load Structured Data Using Lakehouse Auto Parallel Load. (WL #15851)
- MySQL HeatWave Lakehouse now includes support for MySQL HeatWave AutoML. See: Use MySQL HeatWave AutoML with Lakehouse. (WL #15539)

Functionality Added or Changed

- MySQL InnoDB on MySQL HeatWave now has support for a bulk load extension to the `LOAD DATA` statement. This is only available on MySQL HeatWave on AWS. See: Bulk Ingest Data. (WL #14717, WL #14772, WL #15131, WL #15132, WL #15133, WL #15612)

Changes in MySQL HeatWave 8.1.0-u3 (2023-09-12)

- MySQL HeatWave AutoML
- MySQL HeatWave Lakehouse
- Functionality Added or Changed

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports text data types, see: Supported Data Types for MySQL HeatWave AutoML. (WL #15743)

MySQL HeatWave Lakehouse

- MySQL HeatWave Lakehouse can now use a regular expression to define a set of files. (WL #15840)
- MySQL HeatWave Lakehouse now uses Auto Parallel Load to load Parquet files. (WL #15656)
- MySQL HeatWave Lakehouse now supports the Avro file format, see Supported File Formats and Data Types. (WL #15494)

Functionality Added or Changed

- Adaptive Query Execution dynamically adjusts the query execution plan based on runtime statistics to improve query execution time. See: About MySQL HeatWave. (WL #15368)

Changes in MySQL HeatWave 8.1.0-u2 (2023-08-08)

- MySQL HeatWave AutoML
- Functionality Added or Changed

MySQL HeatWave AutoML

- MySQL HeatWave AutoML progress tracking can now track the progress with finer granularity, and show the overall progress as well as the progress of subcomponents, see: Track Progress for MySQL HeatWave AutoML Routines. (WL #15531)

Functionality Added or Changed

- MySQL HeatWave improves time zone support. When the time zone is set to `SYSTEM` or set to an offset from UTC, queries run in MySQL HeatWave. `DATE`, `DATETIME`, `TIMESTAMP`, and `TIME` are processed based on the time-zone value. (Bug #35710134)

Changes in MySQL HeatWave 8.1.0-u1 (2023-07-25)

Functionality Added or Changed

- MySQL HeatWave now supports the following functions:
  - `MID`
  - `REGEXP_INSTR`
  - `SOUNDEX`
  - `SPACE`
  - `WEIGHT_STRING`

  (WL #15627)

Changes in MySQL HeatWave 8.1.0 (2023-07-18)

- MySQL HeatWave AutoML
- Functionality Added or Changed

MySQL HeatWave AutoML

- MySQL HeatWave AutoML has enhanced recommendation systems that can now recommend new users who are similar to a specific user and new items that are similar to other items. See Generate Recommendations. (WL #15674)
- Improvements to MySQL HeatWave AutoML models, the model catalog, and `ML_MODEL_IMPORT` now fully support ONNX and MySQL HeatWave AutoML model import. (WL #15383, WL #15559)

Functionality Added or Changed

- Memory-aware query optimization now automatically adjusts MySQL HeatWave to optimize query execution time or memory usage and reduce the likelihood of queries running out of memory without user intervention. (WL #15529)
- It is now possible to track the memory used by change propagation and the number of transactions committed to MySQL InnoDB that are waiting to be propagated to MySQL HeatWave. (WL #15326, WL #15688)
- MySQL HeatWave now accelerates queries with `INTERSECT` and `EXCEPT` clauses. With this support, MySQL HeatWave can now offload and accelerate all 97 TPC-DS queries. (WL #15362)
- MySQL HeatWave now supports the following temporal functions:
  - `GET_FORMAT()`
  - `MAKETIME()`

- `PERIOD_ADD()`
- `PERIOD_DIFF()`
- `SEC_TO_TIME()`
- `SUBTIME()`
- `SYSDATE()`
- `TIMEDIFF()`

(WL #14958)

Changes in MySQL HeatWave 8.0.33-u3 (2023-06-09)

MySQL HeatWave Lakehouse

- MySQL HeatWave expands to include MySQL HeatWave Lakehouse, letting organizations process and query hundreds of terabytes of data residing in Object Storage—in a variety of file formats, such as CSV and Parquet.

  The Lakehouse feature of MySQL HeatWave enables query processing on Object Storage-resident data. The source data is read from Object Storage, transformed to the MySQL HeatWave format, stored in the MySQL HeatWave persistence storage layer in OCI Object Storage, and then loaded to MySQL HeatWave Cluster memory.

  - Provides in-memory query processing on Object Storage-resident data.
  - Data is not loaded into the MySQL InnoDB storage layer.
  - Supports structured and relational data in CSV and Parquet formats.
  - With this feature, users can now analyze data in both MySQL InnoDB and an object store using familiar SQL syntax in the same query.

  See Load Data from Object Storage into MySQL HeatWave Cluster. (WL #15575)

Changes in MySQL HeatWave 8.0.33 (2023-04-27)

- MySQL HeatWave AutoML
- Functionality Added or Changed

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports a recommendation task, see Generate Recommendations. (WL #15416)

Functionality Added or Changed

- The new MySQL HeatWave Auto Unload utility automates the process of unloading tables from MySQL HeatWave. (WL #15447)
- The following columns were added to the `performance_schema.rpd_query_stats` table:
  - `CONNECTION_ID`: The ID of the connection.

- `STATEMENT_ID`: The global query ID.

(WL #15526)

Changes in MySQL HeatWave 8.0.32-u2 (2023-02-21)

MySQL HeatWave AutoML

- MySQL HeatWave AutoML now supports unsupervised anomaly detection, which is the data mining task that finds unusual patterns in data. (WL #15518)

Changes in MySQL HeatWave 8.0.32-u1 (2023-02-21)

- MySQL HeatWave AutoML
- Functionality Added or Changed

MySQL HeatWave AutoML

- MySQL HeatWave AutoML adds support for multivariate endogenous forecasting models, and exogenous forecasting models. (WL #15511)

Functionality Added or Changed

- The `rpd_tables` table now supports recovery time measurement with `RECOVERY_SOURCE`, `RECOVERY_START_TIMESTAMP` and `RECOVERY_END_TIMESTAMP`. The `performance_schema.global_status` now includes the `rapid_recovery_time` variable. (WL #15441)
- Arithmetic operators are now supported with variable-length encoded string columns. Mathematical functions are now supported with variable-length columns. (WL #15405)

Changes in MySQL HeatWave 8.0.32 (2023-01-17)

- Deprecation and Removal Notes
- MySQL HeatWave AutoML
- Functionality Added or Changed

Deprecation and Removal Notes

- MySQL 8.0.32 deprecates the `heatwave_load_report` and `heatwave_advisor_report` tables, and replaces them with the `heatwave_autopilot_report` table in the `sys` schema. A future release will remove them. (Bug #34727481)

MySQL HeatWave AutoML

- MySQL 8.0.32 introduces a number of changes for MySQL HeatWave AutoML routines:
  - The routines: `ML_PREDICT_ROW`, `ML_PREDICT_TABLE`, `ML_EXPLAIN_ROW`, and `ML_EXPLAIN_TABLE` now include the `ml_results` column, which uses `JSON` format to return the results.
  - The routines: `ML_PREDICT_ROW`, `ML_PREDICT_TABLE`, `ML_EXPLAIN_ROW`, and `ML_EXPLAIN_TABLE` now allow additional columns that are not required for prediction or explanation.

  - The routines: `ML_PREDICT_ROW`, and `ML_PREDICT_TABLE` now allow an *`options`* parameter in `JSON` format.
  - The `ML_TRAIN` routine also runs the `ML_EXPLAIN` routine with the default Permutation Importance model.

  (WL #15420)

- MySQL 8.0.32 introduces progress tracking for `ML_TRAIN`. Use a second MySQL Client window to track the progress of `ML_TRAIN` with calls to the performance schema. It also introduces the `rapid_ml_operation_count` status variable. (WL #15384)

Functionality Added or Changed

- MySQL HeatWave now supports the following encryption and compression functions:
  - `COMPRESS()`
  - `MD5()`
  - `RANDOM_BYTES()`
  - `SHA()`
  - `SHA1()`
  - `SHA2()`
  - `UNCOMPRESS()`
  - `UNCOMPRESSED_LENGTH()`

  (WL #15283)

- MySQL HeatWave now supports the following mathematical functions:
  - `CRC32()`
  - `LOG2()`
  - `LOG10()`
  - `RAND()`

  (WL #15256)

Changes in MySQL HeatWave 8.0.31 (2022-10-11)

- MySQL HeatWave AutoML
- Functionality Added or Changed

MySQL HeatWave AutoML

- MySQL HeatWave AutoML queries are now monitored and recorded in the Performance Schema tables `rpd_query_stats` and `rpd_exec_stats`. Where a single MySQL HeatWave AutoML query contains a number of sub-queries, there is one record in `rpd_query_stats` and multiple records in `rpd_exec_stats`. (WL #15243)

- New functions have been added to MySQL HeatWave AutoML to help you manage models:

  - When you run the `ML_TRAIN` routine on a training dataset, you can now specify a model handle to use for the model instead of the generated one.

  - A new column `notes` has been added to the `MODEL_CATALOG` table, which you can use to record notes about the models in your model catalog.

  - The new column `model_metadata` in the `MODEL_CATALOG` table records metadata for models, such as the training score, training time, and information about the training dataset. If an error occurs during training or you cancel the training operation, MySQL HeatWave AutoML records the error status in this column.

  (WL #15243)

- MySQL HeatWave AutoML now supports the upload of pre-trained models in ONNX (Open Neural Network Exchange) format to the model catalog. You can load them using the stored procedure `ML_MODEL_IMPORT` that provides the conversion required to store the model in a MySQL table. (WL #15243)

- A new stored procedure `ML_EXPLAIN` lets you train a variety of model explainers and prediction explainers for MySQL HeatWave AutoML, in addition to the default Permutation Importance model and prediction explainers:

  - The Partial Dependence model explainer shows how changing the values of one or more columns will change the value that the model predicts.

  - The SHAP model explainer produces global feature importance values based on Shapley values.

  - The Fast SHAP model explainer is a subsampling version of the SHAP model explainer which usually has a faster runtime.

  - The Permutation Importance prediction explainer explains the prediction for a single row or table.

  - The SHAP prediction explainer uses feature importance values to explain the prediction for a single row or table.

  When you use the `ML_EXPLAIN_TABLE` and `ML_EXPLAIN_ROW` stored procedures to generate explanations for a prediction, you can now use the SHAP prediction explainer as an alternative to the default Permutation Importance prediction explainer. SHAP produces feature importance values (explanations) based on Shapley values. (WL #15243)

- MySQL HeatWave AutoML now supports timeseries forecasting using the existing stored procedures `ML_TRAIN`, `ML_PREDICT_TABLE`, and `ML_SCORE`. You can create a forecast for a single column (a univariate endogenous variable) with a numeric data type. The forecasting task is specified as a JSON object when you call the `ML_TRAIN` stored procedure. (WL #15243)

Functionality Added or Changed

- MySQL HeatWave uses dictionary encoding to compress string columns (CHAR, VARCHAR, TEXT). These dictionaries are built for each string column with the `RAPID_COLUMN=ENCODING=SORTED` keyword. MySQL HeatWave now supports 8.5 billion dictionary entries (up from 4 billion), which means MySQL HeatWave can now encode string columns with number of distinct value (NDV) up to 8.5 billion. (WL #14742)

- MySQL HeatWave now uses statistics based on minimum and maximum values to create zone maps for every primary key column. MySQL HeatWave then uses the zone maps for range and point queries

to only scan data chunks that are relevant for the query, and accelerates these queries by an order of magnitude. This is particularly useful for improving range queries in OLAP and mixed workloads. (WL #14713)

- A new MySQL optimizer is introduced for MySQL HeatWave to provide a holistic cost model across MySQL and MySQL HeatWave, create better query plans based on statistics used in MySQL Autopilot, reduce compilation time, eliminate the need of query hints for join order, and improve join query performance. With the new optimizer, MySQL HeatWave can now run all 22 TPC-H queries without straight join hints. Before 8.0.31, a straight join hint is needed for 10 out of 22 TPC-H to reach peak performance. (WL #14449)

- DDL statements such as `ALTER TABLE`, `RENAME TABLE`, and `TRUNCATE TABLE` are now permitted on a table that has RAPID defined as the secondary engine. If a DDL operation is successfully carried out on a table that is loaded to a MySQL HeatWave Cluster at the time, MySQL HeatWave automatically reloads the table from MySQL InnoDB. Note that if the DDL operation makes the table's structure incompatible with MySQL HeatWave, the table is unloaded from the MySQL HeatWave Cluster. (WL #15129)

Changes in MySQL HeatWave 8.0.30-u1 (2022-09-06)

Functionality Added or Changed

- MySQL HeatWave now supports the `ABS()`, `POWER()`, and `SIGN()` mathematical functions. (WL #15246)

- MySQL HeatWave now supports the `GROUP_CONCAT()` aggregation function with variable-length (`VARLEN`) string columns. (WL #14767)

- Change propagation memory management was enhanced to improve MySQL HeatWave shutdown time. (WL #15306)

- Query performance was optimized for compressed data. Decompression of data that is queried but does not participate in filter evaluation is delayed until after a filter is applied. With this optimization, decompression is avoided entirely for data that is filtered out. (WL #15169)

Changes in MySQL HeatWave 8.0.30 (2022-07-26)

- Advisor
- MySQL HeatWave AutoML
- Functionality Added or Changed

Advisor

- MySQL HeatWave Advisor Auto Encoding, which recommends string column encodings, now provides encoding recommendations that optimize query performance. Recommendations are based on performance models that use query execution data. Previously, string column encoding recommendations were optimized for cluster memory usage only. A performance improvement estimate is provided with string column encoding recommendations. (Bug #34145862)

MySQL HeatWave AutoML

- You can now train MySQL HeatWave AutoML models on tables containing `DATE`, `TIME`, `DATETIME`, `TIMESTAMP`, and `YEAR` data types. (Bug #33895503)

- MySQL HeatWave AutoML now generates a model explanation when you train a machine learning model. Model explanations help identify the features that are most important to a model. For more information, see The Model Catalog.

  The following columns were added to the `MODEL_CATALOG` table:

  - `column_names`: The feature columns used to train the model.
  - `last_accessed`: The last time the model was accessed. MySQL HeatWave AutoML routines update this value to the current timestamp when accessing the model.
  - `model_explanation`: The model explanation generated during training.
  - `model_type`: The type of model (algorithm) selected by `ML_TRAIN` to build the model.
  - `task`: The task type specified in the `ML_TRAIN` query (`classification` or `regression`).

  `ML_PREDICT_*` and `ML_EXPLAIN_*` routine performance was improved, resulting in faster prediction and explanation processing. (WL #15088, WL #15014)

- The following MySQL HeatWave AutoML enhancements were implemented:

  - `ML_TRAIN` options for advanced users. These options permit users to customize various aspects of the ML training pipeline including algorithm selection, feature selection, and hyperparameter optimization.
    - The `model_list` option permits specifying the type of model to be trained.
    - The `exclude_model_list` option specifies models types to exclude from consideration during model selection.
    - The `optimization_metric` option specifies the scoring metric to optimize for when training a machine learning model.
    - The `exclude_column_list` option specifies feature columns to exclude from consideration when training a machine learning model.

    For more information, see ML_TRAIN.

  - Support was added for Support Vector Machine `SVC` and `LinearSVC` classification and regression models. For a complete list of supported model types, see Model Types.
  - The `ML_TRAIN` routine now reports a message if a trained model does not meet expected quality criteria.
  - `ML_EXPLAIN_ROW` and `ML_EXPLAIN_TABLE` routines now provide information to help interpret explanations. The routines also report a warning when a model quality issue is detected, enabling users to revisit their data in order to improve model quality.

  (WL #15089)

Functionality Added or Changed

- The amount of heap memory allocated on the MySQL node for each table loaded into MySQL HeatWave was reduced, increasing the maximum number of tables that can be loaded. For `MySQL.HeatWave.VM.E3.Standard` shapes, the maximum was raised from 100k tables to 400k tables. For `MySQL.HeatWave.BM.E3.Standard` shapes, the maximum number was raised from 400k

tables to 1600k tables. The actual number of tables that can be loaded is dependent on the table's data. (Bug #33951708)

- The `performance_schema.rpd_column_id` table was modified to remove redundant data. The `NAME`, `SCHEMA_NAME`, `TABLE_NAME` columns were removed, and a `TABLE_ID` column was added. (Bug #33899183)
- Support was added for the `FROM_DAYS()` temporal function, and `GREATEST()` and `LEAST()` comparison and string functions which now support `DATE`, `DATETIME`, `TIME`, and `TIMESTAMP` columns. (WL #14956)
- Support was added for built-in server-side data masking and de-identification to help protect sensitive data from unauthorized uses by hiding and replacing real values with substitutes. Data masking and de-identification operations are performed on the server, and queries involving data masking and de-identification functions are accelerated by MySQL HeatWave. The following data masking and de-identification functions are supported:
  - `gen_range()`
  - `gen_rnd_email()`
  - `gen_rnd_pan()`
  - `gen_rnd_ssn()`
  - `gen_rnd_us_phone()`
  - `mask_inner()`
  - `mask_outer()`
  - `mask_pan()`
  - `mask_pan_relaxed()`
  - `mask_ssn()`

  See Data Masking and De-Identification Functions. (WL #15143)
- Optimizations were implemented to improve performance for `JOIN` and `GROUP BY` queries with execution plans involving multiple consecutive rounds of data partitioning. (WL #15143)
- *`expr`* `IN (`*`value`*`,...)` comparisons, where the expression is a single value and compared values are constants of the same data type and encoding, have been optimized. For example, the following `IN()` comparison has been optimized:

```
SELECT * FROM Customers WHERE Country IN ('Germany', 'France', 'Spain');
```

  (WL #14952)

Changes in MySQL HeatWave 8.0.28-u3 (2022-04-19)

Functionality Added or Changed

- Tables that have become stale due to a change propagation failure resulting from an out-of-code error are now automatically reloaded. A check for stale tables is performed periodically when the MySQL HeatWave Cluster is idle. Previously, identifying and reloading stale tables was a manual process. See About Change Propagation. (WL #14914)

Changes in MySQL HeatWave 8.0.28-u2 (2022-03-29)

- MySQL HeatWave AutoML
- Functionality Added or Changed

MySQL HeatWave AutoML

- MySQL HeatWave customers now have access to MySQL HeatWave AutoML, which is a fully managed, highly scalable, cost-efficient, machine learning solution for data stored in MySQL. MySQL HeatWave AutoML provides a simple SQL interface for training and using predictive machine learning models, which can be used by novice and experienced ML practitioners alike. With MySQL HeatWave AutoML, you can train a model with a single call to an SQL routine. Similarly, you can generate predictions with a single `CALL` or `SELECT` statement which can be easily integrated with your applications.

  With MySQL HeatWave AutoML, data and models never leave the MySQL Database Service, saving you time and effort while keeping your data and models secure. MySQL HeatWave AutoML is optimized for MySQL HeatWave shapes and scaling, and all MySQL HeatWave AutoML processing is performed on the MySQL HeatWave Cluster. ML computation is distributed among MySQL HeatWave nodes, taking advantage of MySQL HeatWave's scalability and massively parallel processing capabilities. For more information about MySQL HeatWave's machine learning capabilities, see Train and Use Machine Learning Models. (WL #14661, WL #14836, WL #15014)

Functionality Added or Changed

- MySQL HeatWave now compresses data as it is loaded, which permits MySQL HeatWave nodes to store more data. More data per node reduces costs by minimizing the size of the MySQL HeatWave Cluster required to store your data. Data compression is enabled by default but can be disabled at runtime using the `rapid_compression` session variable. For more information, see Enable or Disable Data Compression. (WL #14868)

Changes in MySQL HeatWave 8.0.28-u1 (2022-02-15)

Functionality Added or Changed

- MySQL HeatWave now supports up to 1017 columns for base relations (tables as loaded into MySQL HeatWave), and up to 1800 columns for intermediate relations (intermediate tables used during query processing). The maximum column width for base relations and intermediate relations was increased to 65532 bytes. See Column Limits. (WL #14918)

Changes in MySQL HeatWave 8.0.27-u3 (2021-12-15)

Functionality Added or Changed

- Support was added for the `CONVERT_TZ()` and `LAST_DAY()` functions, which are used to manipulate temporal values. (WL #14768)

- The MySQL HeatWave Cluster recovery process was optimized to avoid applying a large volume of changelogs during recovery. Snapshots are now taken when the volume of changelogs and the time required to apply those changes exceed specific thresholds, and recovery from Object Storage is performed using those snapshots. (WL #14615)

- MySQL HeatWave now supports automatic data reload when the MySQL HeatWave Cluster is restarted. Previously, when a MySQL HeatWave Cluster was stopped by a stop or restart action, data had to be

reloaded manually after the cluster was restarted. Now, when starting or restarting a MySQL HeatWave Cluster, data that was previously loaded is reloaded automatically. Data changes that occur on the DB System while the MySQL HeatWave Cluster is offline are included in the reloaded data.

Automatic data reload does not occur if the MySQL HeatWave Cluster was stopped as a result of a stop or restart action performed on the DB System to which the MySQL HeatWave Cluster is attached. In this case, data loaded in the MySQL HeatWave Cluster must be reloaded manually after the MySQL HeatWave Cluster is restarted. (WL #14729)

Changes in MySQL HeatWave 8.0.27-u2 (2021-12-07)

Functionality Added or Changed

- MySQL HeatWave now supports querying views. See Query Views. (WL #13568)
- Bloom filter join optimizations were introduced. For MySQL HeatWave queries that join large and small relations, bloom filters reduce the amount of data processed by early filtering and the amount of memory used during query processing. (WL #14752)
- The `rpd_query_stats` table, which stores MySQL HeatWave query history (compilation and execution statistics), now stores data for the last 1000 executed queries. Previously, data was stored for the last 200 queries.

  The following columns were added to the `performance_schema.rpd_tables` table:

  - `SIZE_BYTES`: The amount of data loaded per table, in bytes.
  - `QUERY_COUNT`: The number of queries that referenced the table.
  - `LAST_QUERIED`: The timestamp of the last query that referenced the table.
  - `LOAD_END_TIMESTAMP`: The load completion timestamp for the table.

  The following column was added to the `performance_schema.rpd_columns` table:

  - `DICT_SIZE_BTYES`: The dictionary size per column, in bytes.

  The following column was added to the `performance_schema.rpd_nodes` table:

  - `BASEREL_MEMORY_USAGE`: The base relation memory footprint per node.

  The `rapid_query_stats` and `rpd_exec_stats` tables are now synchronized. If a query record is removed from the `rapid_query_stats` table, it is also removed from the `rpd_exec_stats` table. (WL #14759)

Changes in MySQL HeatWave 8.0.26-u2 (2021-09-21)

Functionality Added or Changed

- The following function support was added:
  - `YEARWEEK(`*`date`*`)`, `YEARWEEK(`*`date,mode`*`)`
  - The mode argument for the two-argument form of the `WEEK()` function: `WEEK(`*`date[,mode]`*`)`
  - `MAKEDATE()`

- “Zero” handling for dates such as '2001-11-00' was implemented for `WEEK()`, `YEARWEEK()`, and `MAKEDATE()` functions.
- `CAST()` of `FLOAT` and `DOUBLE` values to `DECIMAL`

(Bug #33163625, Bug #33138534, WL #14714)

- The new `hw_data_scanned` global status variable tracks the total cumulative megabytes scanned by successfully executed MySQL HeatWave queries.

  The number of megabytes scanned by an individual MySQL HeatWave query can be obtained by querying the `performance_schema.rpd_query_stats` table.

  An estimated number of megabytes scanned by an individual query can be obtained by running the query with `EXPLAIN` and querying the `performance_schema.rpd_query_stats` table.

  For more information, see Scanned Data Monitoring. (WL #14738)

Changes in MySQL HeatWave 8.0.26-u1 (2021-08-10)

- MySQL HeatWave Network Layer
- MySQL HeatWave Data Management Layer
- Functionality Added or Changed

MySQL HeatWave Network Layer

- MySQL HeatWave network layer optimizations have improved scalability and network performance. (WL #14513)

MySQL HeatWave Data Management Layer

- Data loaded into MySQL HeatWave, including propagated changes, are now persisted to OCI Object Storage for recovery in case of a MySQL HeatWave node or cluster failure. Previously, data was recovered from the MySQL DB System. Loading data from OCI Object Storage is faster because data does not need to be converted to the MySQL HeatWave storage format, as is required when loading data from the MySQL DB System. If data recovery from OCI Object Storage fails, MySQL HeatWave falls back to recovering data from the MySQL DB System. Data removed from MySQL HeatWave when a table is unloaded is removed from OCI Object Storage in a background operation. For related information, see MySQL HeatWave Cluster Failure and Recovery. (WL #14478, WL #14046, WL #14541)

Functionality Added or Changed

- MySQL HeatWave now supports `COUNT(NULL)`, except in cases where it is used as an input argument for non-aggregate operators. (Bug #33005146)
- Full support was added for the `DISTINCT` modifier. Previously, multiple instances of (`DISTINCT` *`value`*) expressions in a query were only permitted if the same *`value`* was specified. (Bug #32865043, Bug #33007714, WL #14574)
- MySQL HeatWave now supports the `WITH ROLLUP` modifier in `GROUP BY` clauses. (WL #14533)
- MySQL HeatWave now supports window functions. For optimal performance, window functions in MySQL HeatWave utilize a massively parallel, partitioning-based algorithm. For more information, see Window Functions. (WL #14674)

Changes in MySQL HeatWave 8.0.26 (2021-07-23)

- Advisor
- Auto Parallel Load
- Auto Scheduling
- Functionality Added or Changed

Advisor

- The new MySQL HeatWave Advisor provides string column encoding and data placement key recommendations based on machine learning models, data analysis, and MySQL HeatWave query history. Implementing MySQL HeatWave Advisor recommendations can improve query performance and reduce the amount of memory required on MySQL HeatWave nodes.

  The MySQL HeatWave Advisor also provides a Query Insights feature, which provides runtimes for successfully executed queries, and runtime estimates for `EXPLAIN` queries, queries canceled using `Ctrl+C`, and queries that fail due to out of memory errors. Runtime data is useful for query optimization, troubleshooting, and estimating the cost of running a particular query or workload.

  The MySQL HeatWave Advisor is implemented as a stored procedure named `heatwave_advisor`, which resides in the MySQL `sys` schema. Running Advisor involves issuing a `CALL` statement for the stored procedure with optional arguments.

  ```
  CALL sys.heatwave_advisor (options);
  ```

  For more information about the MySQL HeatWave Advisor, see Optimize Workloads for OLAP. (WL #14328, WL #14510, WL #14431, WL #14328, WL #14651)

Auto Parallel Load

- The new MySQL HeatWave Auto Parallel Load utility automates the process of preparing and loading tables into MySQL HeatWave and loads data using an optimized number of parallel load threads.

  The MySQL HeatWave Auto Parallel Load utility is implemented as a stored procedure named `heatwave_load`, which resides in the MySQL `sys` schema. Running Auto Parallel Load involves issuing a `CALL` statement for the stored procedure, which takes a list of schemas and options as arguments.

  ```
  CALL sys.heatwave_load (db_list,[options]);
  ```

  For more information about the MySQL HeatWave Auto Parallel Load utility, see Load Data Using Auto Parallel Load. (WL #14149)

Auto Scheduling

- The MySQL HeatWave query scheduling algorithm was improved. The revised algorithm prioritizes queries based on estimated cost and wait time in the queue, which enables dynamic, workload-aware query prioritization. Previously, queries were prioritized using a static cost-based prioritization model. (WL #14608)

Functionality Added or Changed

- `DATE_ADD()` and `DATE_SUB()` functions now support precision `INTERVAL` values (`DECIMAL`, `DOUBLE`, and `FLOAT`). (Bug #32725985, Bug #32438123)

- Support was added for multiple instances of `COUNT(DISTINT)` in a query. (Bug #32422984)
- Query compilation and processing was improved to permit combining aggregate operators into a single task in the physical query plan, which avoids fully materializing intermediate result sets. This enhancement reduces memory allocation and deallocation operations, memory usage, and execution time for affected queries. (WL #14614)
- The cost model that estimates MySQL HeatWave query runtimes can now use statistics from previously executed queries, which improves the accuracy of query runtime estimates. (WL #14546)
- MySQL HeatWave now supports `CREATE TABLE ... SELECT` statements where the `SELECT` query is offloaded to MySQL HeatWave and the table is created on the MySQL Database Service instance. This feature improves `CREATE TABLE ... SELECT` performance in cases where the `SELECT` portion of the statement is a long running, complex query. For more information, see CREATE TABLE ... SELECT Statements. (WL #14516)
- Support was added for `REGEXP_REPLACE()` and `REGEXP_SUBSTR()` regular expression functions, and error messaging was improved for `REGEXP()` function syntax mismatches, expression errors, and input argument errors. (WL #14641)

Changes in MySQL HeatWave 8.0.25 (2021-05-11)

Functionality Added or Changed

- Support was added for `CAST()` of `ENUM` column values to `CHAR` or `VARCHAR` where the `ENUM` value is cast to a `FLOAT` value, as in the following example:

  ```
  SELECT CAST(CAST(enum_col AS FLOAT) AS CHAR(3)) FROM tbl_name;
  ```

  (Bug #32618454)

- Support was added for `SELECT DISTINCT` queries that order the result set by a column that is not defined in the `SELECT` list. For example, the following query can now be offloaded to MySQL HeatWave for execution:

  ```
  SELECT DISTINCT a FROM t1 ORDER BY c DESC;
  ```

  (Bug #32583856)

- Query plan statistics are now collected and stored in a statistics cache when a query is executed in MySQL HeatWave. When a new query shares query execution plan nodes with previously executed queries, the actual statistics collected from previously executed queries are used instead of estimated statistics, which improves query execution plans, cost estimations, execution times, and memory efficiency.

  The statistics cache is an LRU structure. When cache capacity is reached, the least recently used entries are evicted from the cache as new entries are added. The maximum number of entries permitted in the statistics cache is defined by the `rapid_stats_cache_max_entries` setting. The number of entries permitted by default is 65536, which is enough to store statistics for 4000 to 5000 unique queries of medium complexity. (WL #14503)

- Support was added for:

  - `CAST() AS YEAR`. Both variable-length and dictionary-encoded string columns values are supported.
  - The `FORMAT()` function. Variable-length-encoded string columns are not supported.

  (WL #14511)

Changes in MySQL HeatWave 8.0.24 (2021-04-20)

Functionality Added or Changed

- Comparison of different temporal type values is now supported. For example, a query that compares `DATE` values to `TIMESTAMP` values can now be offloaded to MySQL HeatWave. (Bug #32420986)
- Range operators on `VARLEN`-encoded string columns are now supported. For example, the following query, where `L_LINESTATUS` is a `VARLEN`-encoded string column, can now be offloaded to MySQL HeatWave:

  ```
  SELECT COUNT(*) FROM lineitem WHERE L_LINESTATUS  >= 1 and L_LINESTATUS <= 10;
  ```

  (Bug #31721399)
- MySQL HeatWave now supports `INSERT ... SELECT` statements where the `SELECT` query is offloaded to MySQL HeatWave and the result set is inserted into a table on the MySQL Database Service instance. This feature improves `INSERT ... SELECT` performance in cases where the `SELECT` portion of the statement is a long running, complex query. For more information, see INSERT ... SELECT Statements. (WL #14299)
- `VARLEN`-encoded columns are now supported as data placement keys. For information about the data placement feature, see Data Placement Keys. (WL #14491)
- Failure handling was improved for queries involving unsupported internal data types. Such queries now exit with an error indicating that the internal data type of the query is not supported. (WL #14483)

Changes in MySQL HeatWave 8.0.23-u2 (2021-03-15)

Functionality Added or Changed

- Support was added for the following aggregate functions:
  - `STD()`
  - `STDDEV()`
  - `STDDEV_POP()`
  - `STDDEV_SAMP()`
  - `VAR_POP()`
  - `VAR_SAMP()`
  - `VARIANCE()`

  See Aggregate Functions. (WL #14479)
- MySQL HeatWave now uses a priority-based scheduling mechanism based on query cost estimates to schedule queries for execution. Previously, queries were executed in the order of arrival. The scheduling mechanism prioritizes short running queries over long running queries to reduce overall query execution wait times. For more information, see Auto Scheduling. (WL #14423)

Changes in MySQL HeatWave 8.0.23-u1 (2021-02-09)

Functionality Added or Changed

- String column encoding support was added for `TEXT`-type columns. See Encoding String Columns. (WL #14430)
- `UNION` and `UNION ALL` support was extended. The clauses are now supported at any location in a query that is permitted by MySQL. (WL #14455)
- The following temporal functions are now supported:
  - `TO_SECONDS()`
  - `UNIX_TIMESTAMP()`
  - `FROM_UNIXTIME()`
  - `TIME_TO_SEC()`

  See Temporal Functions.

  The following temporal functions are now supported with `VARLEN`-encoded columns:
  - `TO_DAYS()`
  - `DAYOFYEAR()`
  - `QUARTER()`
  - `TO_SECONDS()`

  See Temporal Functions.

  The following string functions are now supported with `VARLEN`-encoded columns:
  - `ORD()`
  - `ASCII()`

  See String Functions and Operators.

  `SET timezone = timezone` with the *`timezone`* value specified as an offset from UTC in the form of *`[H]H:MM`* and prefixed with a `+` or `-` is now supported with the `UNIX_TIMESTAMP()` and `FROM_UNIXTIME()` functions. (WL #14345)
- Offset is now supported with the `LIMIT` clause:

  ```
  SELECT * FROM tbl LIMIT offset,row_count;
  ```

  The PostgreSQL syntax is also supported:

  ```
  SELECT * FROM tbl LIMIT row_count OFFSET offset;
  ```

  (WL #14341)
- New Performance Schema tables provide access to query and execution statistics:
  - `performance_schema.rpd_exec_stats`
  - `performance_schema.rpd_query_stats`

Changes to MySQL HeatWave Performance Schema tables:

- An `NDV` (Number of Distinct Values) column was added to the `performance_schema.rpd_columns` table.
- A `ROWS` column that shows the total number of rows in a table was added to the `performance_schema.rpd_tables` table.
- A `MEMORY_USAGE` column that shows node memory usage was added to the `performance_schema.rpd_columns` table.
- The `performance_schema.rpd_nodes` `DRAM` column was renamed to `MEMORY_TOTAL`. The `MEMORY_TOTAL` column shows the total memory allocated to a MySQL HeatWave node.

See MySQL HeatWave Performance Schema Tables. (WL #14386)

Index

A

ABS(), 39
Adaptive Query Execution, 33
Advisor, 39, 45
Aggregate functions, 28, 30, 47
ALTER TABLE, 37
ASCII(), 47
Auto Data Compression, 28, 30
Auto Encoding, 39
Auto Parallel Load, 22, 23, 28, 29, 30, 32, 33, 45
Auto Unload, 28, 35
AutoML, 5, 8, 10, 12, 14, 18, 21, 22, 23, 25, 28, 30, 32, 33, 33, 34, 35, 36, 36, 36, 37, 39, 42
Autopilot, 22, 25
AVG(), 31
Avro file format, 30, 33

B

BIN(), 31
BIT_LENGTH(), 31
Bloom filter optimization, 43
Bulk load, 25, 28, 32

C

CAST(), 31, 43, 46
Change propagation, 34, 39, 41
Column limits, 42
Comparison functions , 29, 39
Comparisons, 39
COMPRESS(), 36
Compression functions, 36
CONVERT_TZ(), 42
COUNT(DISTINCT), 45
COUNT(NULL), 44
CRC32(), 36

CREATE TABLE ... SELECT, 45
CSV file format, 35
CUBE, 30

D

Data compression, 39, 42
Data placement, 47
Data types, 47
DATE, 31, 33
DATETIME, 31, 33
DATE_ADD(), 45
DATE_SUB(), 45
DAYOFYEAR(), 47
Dictionary encoding, 37
DISTINCT, 44

E

ELT(), 31
Encryption functions, 36
Error reporting, 23, 29
EXCEPT, 34
EXPORT_SET(), 31

F

FIELD(), 31
FORMAT(), 46
FROM_DAYS(), 39
FROM_UNIXTIME(), 47

G

GenAI, 4, 5, 7, 8, 10, 10, 12, 14, 15, 17, 19, 20, 21, 22, 23, 25
gen_range(), 39
gen_rnd_email(), 39
gen_rnd_pan(), 39
gen_rnd_ssn(), 39
gen_rnd_us_phone(), 39
GET_FORMAT(), 34
GREATEST(), 29, 39
GROUP BY, 30, 44
GROUP_CONCAT(), 39
Guided Load, 28, 31

H

Heap segment size, 39
heatwave_advisor_report table, 36
heatwave_autopilot_report table, 36
heatwave_load_report table, 36
High Availability, 28
HLL(), 28, 30
HyperLogLog, 28, 30

I

IN(), 39

INSERT ... SELECT, 47
INTERSECT, 34

J

JavaScript, 22
JSON ->, 30
JSON ->>, 30
JSON column path, 30
JSON data type, 28
JSON file format, 28
JSON functions, 30
JSON inline path, 30
JSON_ARRAY(), 30
JSON_DEPTH(), 30
JSON_EXTRACT(), 30
JSON_LENGTH(), 30
JSON_OBJECT(), 30
JSON_UNQUOTE(), 30

L

Lakehouse, 4, 5, 5, 7, 7, 8, 10, 12, 14, 15, 17, 18, 20, 21, 22,
22, 23, 25, 28, 28, 29, 30, 30, 32, 33, 35
Lakehouse Auto Parallel Load, 28, 29, 30, 32, 33
Lakehouse Incremental Load, 25
LAST_DAY, 42
LEAST(), 29, 39
LIMIT, 47
Loading data, 39
LOG10(), 36
LOG2(), 36

M

Machine learning, 8, 10, 12, 14, 18, 21, 22, 23, 25, 28, 30, 32, 33, 33,
34, 35, 36, 36, 36, 37, 39, 42
MAKEDATE(), 43
MAKETIME(), 34
MAKE_SET(), 31
mask_inner(), 39
mask_outer(), 39
mask_pan(), 39
mask_pan_relaxed(), 39
mask_ssn(), 39
Mathematical functions, 36, 36, 39
MD5(), 36
Memory usage, 34
MID, 34
MySQL, 4, 5, 5, 8, 10, 10, 12, 15, 17, 18, 19, 20, 22, 23, 25
MySQL additional functionality, 25, 28, 28, 30, 32
MySQL HeatWave, 4, 5, 5, 8, 10, 10, 12, 15, 17, 18, 19, 20, 22, 23,
25
MySQL HeatWave AutoML, 5, 8, 10, 12, 14, 18, 21, 22, 23, 25, 28, 30, 32,
33, 33, 34, 35, 36, 36, 36, 37, 39, 42
MySQL HeatWave Autopilot;, 22, 25

MySQL HeatWave GenAI, 4, 5, 7, 8, 10, 10, 12, 14, 15, 17, 19, 20, 21, 22, 23, 25
MySQL HeatWave Lakehouse, 4, 5, 5, 7, 7, 8, 10, 12, 14, 15, 17, 18, 20, 21, 22, 22, 23, 25, 28, 28, 29, 30, 30, 32, 33, 35
MySQL InnoDB, 25, 28, 32

N

Networking, 44

O

Object Storage, 42
OFFSET, 47
Optimizer, 37
ORD(), 47
Outbound Replication, 28

P

Parquet file format, 33, 35
Partitioning, 39
Partitions, 22
Performance Schema, 35, 36, 39, 43, 47
PERIOD_ADD(), 34
PERIOD_DIFF(), 34
Planned shutdown, 42
Point-in-time-recovery, 28
POWER(), 39

Q

QUALIFY, 30
QUARTER(), 47
Queries, 43, 47
Query cost model, 45
Query optimization, 34
Query plans, 37
Query scheduling, 45, 47
Query support, 30

R

RAND(), 36
RANDOM_BYTES(), 36
rapid_stats_cache_max_entries, 46
Read Replication, 28
Recovery, 42, 44
REGEXP_INSTR, 34
REGEXP_REPLACE(), 45
REGEXP_SUBSTR(), 45
Regular expression functions, 45
Reload tables, 28
RENAME TABLE, 37
rpd_cloumns table, 47
rpd_column_id table, 39
rpd_exec_stats table, 47

rpd_nodes table, 47
rpd_query_stats table, 35, 43, 47
rpd_tables table, 36

S

Scalability, 44
Schema inference, 28, 30, 32
SEC_TO_TIME(), 34
SELECT, 28, 30
SELECT DISTINCT, 46
SET timezone, 47
SHA(), 36
SHA1(), 36
SHA2(), 36
SIGN(), 39
SOUNDEX, 34
SPACE, 34
Statistics, 46
STD(), 30, 31, 47
STDDEV(), 30, 31, 47
STDDEV_POP(), 30, 31, 47
STDDEV_SAMP(), 30, 31, 47
String column encoding, 47
String functions, 29, 31, 39, 47
SUBTIME(), 34
SUM(), 31
SYSDATE(), 34

T

TABLESAMPLE, 28
Temporal data types, 31, 39
Temporal functions, 28, 34, 39, 42, 43, 45, 47
Temporal type comparison, 47
TEXT, 47
TIME, 31, 33
TIMEDIFF(), 34
Timeseries forecasting, 37
TIMESTAMP, 31, 33
Timezone, 33
Timezone functions, 28, 47
TIME_TO_SEC(), 47
TO_DAYS(), 47
TO_SECONDS(), 47
TRUNCATE TABLE, 37

U

UNCOMPRESS(), 36
UNCOMPRESSED_LENGTH(), 36
UNION, 47
UNION ALL, 47
UNIX_TIMESTAMP(), 47
Unload tables, 28

V

Variable-length encoding, 47
VARIANCE(), 30, 31, 47
VARLEN encoding, 47
VAR_POP(), 30, 31, 47
VAR_SAMP(), 30, 31, 47
Views, 43

W

WEEK(), 43
WEIGHT_STRING, 34
Window functions, 30, 44
WITH ROLLUP, 44

Y

YEAR, 31
YEARWEEK(), 43

Z

Zone maps, 22, 37