MySQL HeatWave Lakehouse —Technical overview

Query hundreds of terabytes of data in object storage with unparalleled price-performance

Table of contents

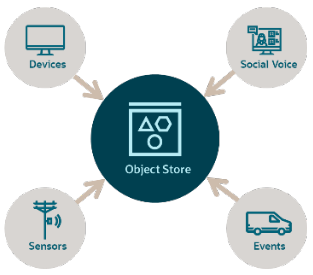
Purpose statement

This document provides an overview of features and enhancements included in Oracle MySQL HeatWave Lakehouse. It is intended solely to help you assess the benefits of MySQL HeatWave Lakehouse and to plan your IT projects.

Disclaimer

Executive Summary

MySQL HeatWave is a fully managed data processing service with a scale out architecture designed to process data residing in the object store and inside the MySQL database. This scale out architecture enables MySQL HeatWave to efficiently process data in the object store with the industry's best performance and price-performance. MySQL HeatWave is designed to enable analytics, machine learning, generative AI, vector query processing, and running JavaScript—across structured, semi-structured, and unstructured files in the object store. MySQL HeatWave provides the capability to create a vector store from documents in object storage, enabling enterprises to take advantage of generative AI with their proprietary content. MySQL HeatWave is available on OCI, AWS, and Azure.



MySQL HeatWave Lakehouse offers capabilities to combine data in object storage with transactional data in MySQL Database in a single query. Data in the object store remains in the object store and is not copied into the database. Changes made to files in object storage are propagated to MySQL HeatWave such that users always get the most up-to-date results. Users can train their machine learning models, perform inferences, and explain these inferences without having to ingest data into a database or move it to an external machine-learning service. MySQL HeatWave Lakehouse scales out to 512 nodes on a single cluster.

With MySQL HeatWave Lakehouse, customers can query hundreds of terabytes of data in the object store in a variety of file formats such as CSV, Parquet, Avro, JSON, exports from databases (e.g., Aurora, Redshift, MySQL, Oracle), as well as unstructured documents in file formats like PDF, DOC(X), PPT(X), TXT, and HTML. Querying data in the object store is as fast as querying the data in the database—an industry first. Customers can also run generative AI on unstructured content in the object store to summarize the content, query it using natural language, or augment it using retrieval augmented generation (RAG).

As demonstrated by a TPC-H benchmark with 500 TB of data, the query performance of MySQL HeatWave Lakehouse is 18X faster than Snowflake, 15X faster than Amazon Redshift, 18X faster than Databricks, 35X faster than Google BigQuery. The load performance of MySQL HeatWave Lakehouse is 2X faster than Snowflake, 9X faster than Amazon Redshift, 6X faster than Databricks, and 8X faster than Google BigQuery.

Challenges Facing Lakehouse Solutions

The exponential growth of data has created several challenges that any viable lakehouse solution must address:

- **Complexity**. Multiple services, for transactional data, data warehousing, data lake analytics, and machine learning add complexity to configure, secure, manage, integrate, and maintain these services. The introduction of vector stores and generative AI services exacerbates these challenges.

- **High costs**. Multiple services mean additional costs; for provisioning, configuration, maintenance, plus the actual services costs. Additionally, for GenAI and vector store workloads, customers need to pay for external large language models (LLMs).

- **Poor query performance and scalability.** Database and query optimizations for in-database data are not applicable to object storage data. When working with massive amounts of data, the infrastructure is unable to scale and ingest data at high speed.

- **Combining transactional data with object storage data**. Most enterprises have relatively small, but high value transactional data that requires a stable, proven data processing engine. A lakehouse solution must include an integrated and robust transactional data store.

- **Support for structured, semi-structured, and unstructured data in the data lake.** Data lakes support structured and semi-structured data, but also unstructured data in formats like DOC(X), PPT(X), TXT, HTML. A lakehouse solution must support ingesting unstructured document formats into a vector store and provide semantic querying using vector search capabilities.

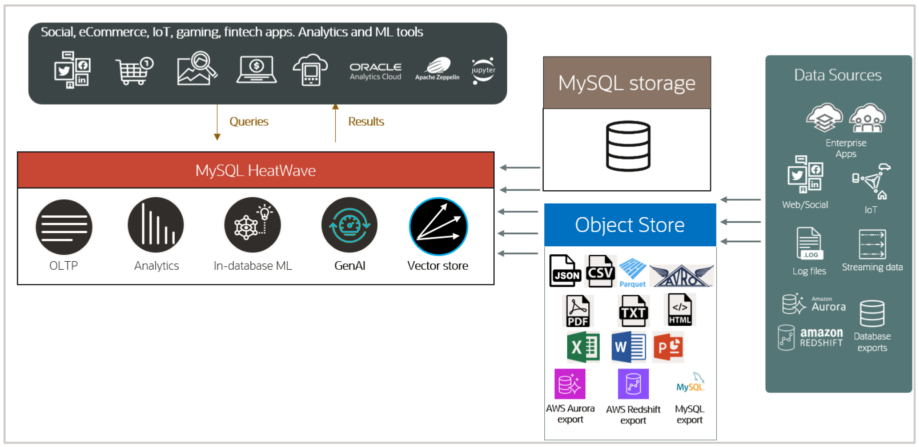
- **Proprietary query syntax for heterogenous data in object storage.** Managing different database systems for different kinds of data processing—OLTP, OLAP, data lake, machine learning— is a usability hindrance and requires extra data orchestration efforts across the systems, adding complexity and data quality challenges.

- **Machine learning on object store data.** A lakehouse solution must be able to query and combine data gathered from telemetry, sensors, IoT devices, web applications, and other sources with transactional data to build machine learning models and derive inferences, without requiring data movement or using disparate services.

"It has been a given since Big Data has been around that Big Data / Lakehouse queries are substantially slower than transactional queries. HeatWave ends that once and forever, demonstrating that Lakehouse performance can be identical to transaction query performance—unheard of and even unthinkable."

Holger Mueller
Vice President & Principal Analyst
Constellation Research

Introducing MySQL HeatWave Lakehouse



MySQL HeatWave Lakehouse

MySQL HeatWave Lakehouse is designed to address the aforementioned challenges through its built-in MySQL HeatWave in-memory query accelerator, which combines transactions and analytics across data warehouses and data lakes, machine learning, and generative AI into a single service. It delivers real-time, secure analytics without the complexity, latency, and extra cost of ETL duplication. MySQL HeatWave Lakehouse provides industry-leading performance and price-performance with the following important highlights:

- **Simplicity**. A unified service that delivers data warehousing, data lake analytics, transaction processing, machine learning, and generative AI capabilities with built-in ML-based automation.
- **Highly performant query engine.** MySQL HeatWave automatically compresses relevant columns— ensuring customers get the most out of their provisioned MySQL HeatWave cluster, delivering the best performance and price-performance even for massive amounts of data.
- **A scale-out architecture** that can ingest, manage, and execute queries at record speeds on hundreds of terabytes of data, and deliver industry leading query performance with the ability to scale from one MySQL HeatWave node to 512 nodes on a single cluster.
- **MySQL HeatWave Autopilot** uses ML-based optimizations to automate common data management tasks, including automatic schema inference for semi-structured data and auto data loading.
- **Built-in machine learning** that makes it fast and easy to build machine learning models on data in object storage (or on data in the MySQL database) for predictions and explanations. The same set of APIs is used to train, predict, and explain a model, irrespective of the source of data—database or object storage; and independently from the underlying format of the object storage data—CSV, Avro, Parquet, JSON, or other database exports.

"HeatWave Lakehouse scales out very well for loading data from object storage and for running queries on object store. The load time and the query times are nearly constant as the size of the data grows and the HeatWave cluster size grows correspondingly. This scale out characteristic of HeatWave Lakehouse for data management is key to efficiently processing very large amounts of data."

Henry Tullis
Leader
Cloud Infrastructure & Engineering
Deloitte Consulting

- **Unstructured documents support**. Using the same MySQL HeatWave Lakehouse commands, you can load unstructured documents into MySQL HeatWave Vector Store, automatically parsing them into segments and creating the vector embeddings. These are available immediately for querying using both exact and similarity searches, via either SQL or natural language.

- **Standard SQL syntax, no proprietary syntax**. Whether querying data in MySQL Database or data loaded into MySQL HeatWave from object storage, the same MySQL syntax is used. This ensures compatibility with downstream applications that can query the data lake data without any new connectors or syntax.

- **Combine transactional data and object storage data in a single query**. Users can, in a single query, query data in a MySQL transactional store as well as object storage data loaded into MySQL HeatWave and apply join, aggregate, and filter predicates.

Scale-out Architecture

MySQL HeatWave Lakehouse is powered by a massively parallel, high-performance, in-memory query processing engine optimized to manage massive amounts of data across a cluster of hundreds of compute nodes. To design a scale-out lakehouse system, we not only require query processing to scale out, but also require efficient and fast transformation and loading of data into the MySQL HeatWave cluster memory. The other challenge is scaling the data ingestion along with an efficient transformation of multiple file formats into hybrid columnar in-memory data representation. MySQL HeatWave Lakehouse uses a massively parallel and scalable data transformation engine that fully utilizes all the compute nodes and the CPU cores in the cluster for a truly scale-out lakehouse architecture.

MySQL HeatWave Lakehouse is meticulously optimized to efficiently scale out with increasing nodes and data sizes in the following ways:

- Scaling the distribution of data scans and transformation tasks across the cluster can be challenging when performing data-driven partitioning. MySQL HeatWave Lakehouse is optimized for avoiding any synchronization overheads across compute nodes via a novel technique called super-chunking that divides the source data into smaller units of work.
- Dynamic task load balancing across the cluster avoids stragglers by ensuring that no CPU core in the cluster is left idle, distributing tasks across the nodes adaptively while observing the CPU utilization in each.
- A novel adaptive data flow mechanism on each node in the cluster independently moderates its own rate of object store requests to match the maximum rate available at any given time. The presence of this novel technique avoids excessive read requests from just one node, which may otherwise result in poor performance and scalability degradation.
- When loading unstructured documents (e.g., file formats like DOC(X), PPT(X), HTML, TXT, and PDF) from the object store into MySQL HeatWave Vector Store, the parsing of documents, creation of vector embeddings, and then

"Data is growing exponentially and so is the amount of data we store in our data lake. The ability to use standard MySQL syntax to query data across our database and object storage to get real-time insights is very important for Natura. This opens up new opportunities to explore and could represent new competitive advantages if we can analyze all this data faster than our competition."

Fabricio Rucci
Solution Architect Analyst
Natura&Co

insertion into the vector store is carried out using MySQL HeatWave's highly parallelized architecture that distributes these tasks across the nodes and cores of a MySQL HeatWave cluster. Once unstructured documents have been loaded into MySQL HeatWave Vector Store, performing semantic searches leverages the parallelism in MySQL HeatWave to deliver the fast performance that scales across hundreds of nodes in a cluster.

Incremental Updates

Once a table has been loaded with data from the object store, the underlying files can change — new files can be added, files can be deleted, and existing files may be updated with new content. MySQL HeatWave Lakehouse can process these changes by monitoring the object store locations, processing only the incremental changes, and updating its copy of the MySQL HeatWave data accordingly. Incremental updates are designed to be low-latency and fast in MySQL HeatWave Lakehouse, and can process massive amounts of changes at scale.

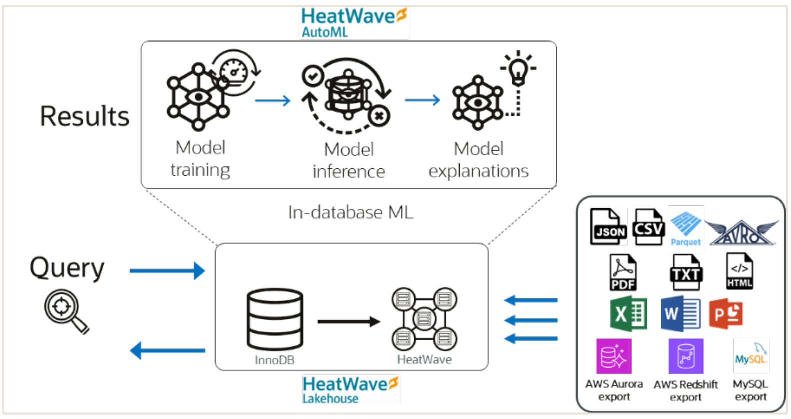
"For HeatWave Lakehouse to deliver record performance for both loading data and querying data is an unprecedented innovation in cloud data services."

Ron Westfall
Senior Analyst and Research Director
Futurum Research

GenAI and Machine Learning with MySQL MySQL HeatWave Lakehouse

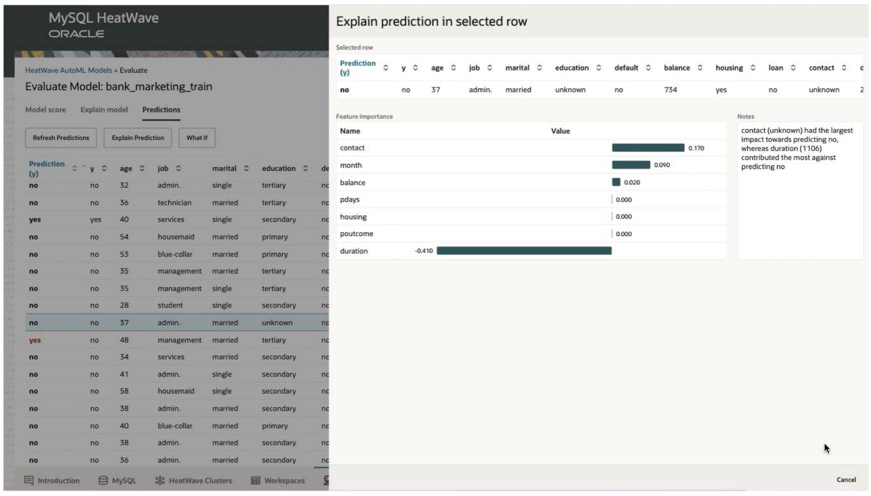
The integrated capabilities of MySQL HeatWave Lakehouse, MySQL HeatWave AutoML, and MySQL HeatWave GenAI enable several use-cases. For example, manufacturing log data written to object store can be loaded into Lakehouse where machine learning inference can be run to detect anomalies. The output can then be sent to MySQL HeatWave GenAI as a RAG prompt by referencing other documents already loaded in MySQL HeatWave Vector Store to summarize the findings, and to then present the response in natural language.

Customers can train, predict, and explain their machine learning models on data loaded from object storage—in both OCI and AWS. MySQL HeatWave AutoML uses a common set of APIs to train, predict, and explain a model, irrespective of whether the data is in the lakehouse or in the database. Users are provided with a unified API to perform machine learning. Once the data is loaded into MySQL HeatWave from object storage, users can create a model, train the model, and use the trained model to make predictions. The interactive console simplifies the process of creating models, explaining them, deriving inferences, and performing scenario analysis. The console enables non-technical users to perform machine learning with ease.



In the screenshot below, a model has been trained using a bank's marketing dataset to predict whether the bank's sales calls successfully led to term deposit subscriptions. Users can also run explanations on these predictions.

Both plain-text explanations and an array of attributions are displayed to assist in determining which attributes were the most impactful while making the prediction. Users can see the reasoning behind the model predictions and take decisions accordingly.



Evaluate Model: bank_marketing_train

Prediction (y)	y	age	job	marital	education	de
no	no	32	admin.	single	tertiary	nc
no	no	36	technician	married	tertiary	nc
yes	yes	40	services	single	secondary	nc
no	no	54	housemaid	married	primary	nc
no	no	53	blue-collar	married	primary	nc
no	no	35	management	married	tertiary	nc
no	no	35	management	single	tertiary	nc
no	no	28	student	single	secondary	nc
no	no	37	admin.	married	unknown	nc
yes	no	48	management	married	tertiary	nc
no	no	34	services	married	secondary	nc
no	no	41	admin.	single	secondary	nc
no	no	58	housemaid	single	secondary	nc
no	no	38	admin.	married	secondary	nc
no	no	40	blue-collar	married	primary	nc
no	no	38	admin.	married	secondary	nc
no	no	36	admin.	married	secondary	nc

Explain prediction in selected row

Selected row

Prediction (y)	y	age	job	marital	education	default	balance	housing	loan	contact	d
no	no	37	admin.	married	unknown	no	734	yes	no	unknown	2

Feature Importance

Name	Value
contact	0.170
month	0.090
balance	0.020
pdays	0.000
housing	0.000
poutcome	0.000
duration	-0.410

Notes

contact (unknown) had the largest impact towards predicting no, whereas duration (1106) contributed the most against predicting no

Explaining predictions for a selected result on object store data

"Simply put: HeatWave Lakehouse enables you to stay ahead of the competition by taking swift action on meaningful business insights."

Steve McDowell
Principal Analyst & Founding Partner
NAND Research

MySQL HeatWave Autopilot Reduces Complexity for Customers

MySQL HeatWave Autopilot provides machine learning-powered automation for MySQL HeatWave. Several existing Autopilot features have been enhanced to support MySQL HeatWave Lakehouse and new capabilities have been introduced. **Auto-provisioning** predicts the number of required MySQL HeatWave compute nodes for running a workload and has been enhanced to support and consume files directly from the object store. **Auto query plan improvement** learns various run-time statistics from the execution of queries to further improve the execution plan of unique queries in the future. **Auto parallel loading** analyzes data to predict the load time into MySQL HeatWave and loads data efficiently from the object store with a high degree of parallelism.

MySQL HeatWave Autopilot capabilities for MySQL HeatWave Lakehouse include:

- **Auto-schema inference** samples a small fraction of data in object storage and infers the number of columns, the data types, and the precision of these columns. This is particularly advantageous when working with CSV files that do not contain any metadata. As a result, you don't need to manually define and update the schema mapping of files, saving time and effort.
- **Adaptive data sampling** intelligently samples files to derive information needed for automation and the nature of the data in question. Using these novel techniques, MySQL HeatWave Autopilot can scan and

propose schema predictions on a set of data files totalling 500 TBs in under one minute.

- **Adaptive data flow** learns and coordinates network bandwidth utilization of the object store across a large cluster of nodes, dynamically adapting to the performance of the underlying object store, resulting in optimal performance and availability.
- **Auto query plan improvement**: MySQL HeatWave Autopilot learns query and data statistics from previously executed queries, which improves the optimizer statistics, and, therefore, subsequent query execution plans.

When it comes to data lakes, common file formats may not be structured, and often it is not trivial to define strict data models for such data sources. Specifically, CSV is a good example of a semi-structured file format where the column types are not pre-defined in the file. Without prior knowledge or insight from the data, users often choose conservative data types and sizes that would be wasteful or lead to sub-optimal query performance (e.g., using `varchar` for all types). With Autopilot, this process is now fully automated and data-driven, eliminating user guesswork.

All these intelligent optimizations by MySQL HeatWave Autopilot are interactive, even for large data sizes (as large as 500TB), and use an efficient adaptive sampling algorithm on a relevant subset of the underlying data to make suggestions.

“HeatWave, now with Lakehouse, may be the most significant open-source cloud database innovation in the last decade.”

Marc Staimer
Senior Analyst
Wikibon

Deployment and Use Case Scenarios

To best understand the capabilities and usability of our managed service, we will walk through a deployment scenario that is uniquely possible with MySQL HeatWave Lakehouse. The deployment goal here is to have the following tables managed and be query-ready in MySQL HeatWave Lakehouse:

- *Table inside database:* **Sales** is a traditional MySQL transactional table managed by the InnoDB engine and loaded into the MySQL HeatWave cluster. This table is frequently updated by many cloud applications. Any change done to this table through InnoDB is propagated in real-time and is readily available in the MySQL HeatWave cluster for queries.
- *Object store files:* **Sensor** is a CSV file generated by an application. **SensorInventory** contains the data exported from an Amazon Aurora database as a Parquet file, which has been uploaded to the object store.

Let us assume that all OLTP tables are already managed by MySQL HeatWave, with the Lakehouse feature enabled. You will provide MySQL HeatWave Lakehouse access to the objects in the object storage. This can be done with two access control methods: OCI Resource Principal mechanism or PAR.

To start using this external data as external tables, users need to:

- Define the schema of the external tables. To do this, run MySQL HeatWave Autopilot on data in the Object Store.

```
mysql> CALL
sys.heatwave_load(<db_names>,<info_about_file_in_ObjectStore>);
```

Autopilot runs and provides the DDL for the Sensor table.

- The table is created by running the DDLs returned by Autopilot:

```
mysql> CREATE TABLE Sensor
            (`id` INT NOT NULL,
             `date` DATE NOT NULL,
             `temperature` INT NOT NULL)
             ENGINE=lakehouse
             SECONDARY_ENGINE=RAPID
             ENGINE_ATTRIBUTE='{"file": [{"bucket": "bucket name",
            "region": "region name"…}],
                    "dialect": {"format": "parquet"…}}';
```

- Load the data from the Object Store into MySQL HeatWave.

```
mysql> ALTER TABLE Sensor SECONDARY_LOAD;
```

Just as the **Sensor** table was loaded into MySQL HeatWave, the **SensorInventory** Amazon Aurora table exported to and copied to object storage can also be loaded into MySQL HeatWave. With the two new external tables now loaded, users and developers can use the familiar MySQL syntax to construct queries:

```
mysql> SELECT count(*) FROM Sensor, Sales
            WHERE Sensor.degrees> 30 AND Sensor.id= Sales.id;
```

- Such queries are not only limited to InnoDB and external tables but also work across different external tables in different file formats, e.g., a join between the **Sensor** and **SensorInventory.**

In all the above scenarios, customers do not need any lengthy ETL processes between disparate systems, nor do they require the cloud application to be aware of the different data sources.

Unstructured Documents and Vector Store

Using the same commands available in MySQL HeatWave Lakehouse, users can upload enterprise documents to object storage and securely ingest them into MySQL HeatWave Vector Store. Once ingested, users can perform similarity (semantic) search on them. With MySQL HeatWave Vector Store, customers can now converse with unstructured documents in natural language.

When working with unstructured data loaded from object storage, the documents are parsed into segments and then converted into embeddings; each embedding is represented numerically as a point in an n-dimensional space, and these embeddings are stored in a vector store. Using MySQL HeatWave's support for vector data type and functions, queries can be written to perform similarity searches. Alternatively, users can ask questions using natural language.

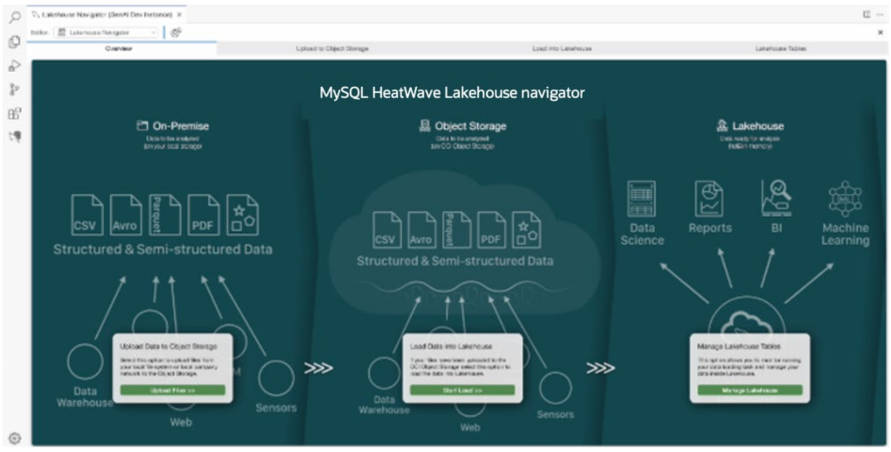
MySQL HeatWave supports a new native data type called VECTOR, designed specifically to store and manage embeddings. VECTOR data is stored in the OLTP-optimized row-major format in InnoDB, ensuring efficient transactional processing. In MySQL HeatWave, VECTOR data is stored in an in-memory hybrid columnar format, optimized for high-performance analytical queries. The implementation ensures that VECTOR data is treated as a first-class citizen within the database, allowing a seamless integration and support for all standard database operations.

MySQL HeatWave automates the entire process of data loading into a vector store, including the choice of the embedding model, creation of the embeddings, and then using MySQL HeatWave's highly parallel architecture to insert the embeddings.

MySQL HeatWave supports running LLMs in-database. Not only does this minimize data movement and costs, but since the in-database run on CPUs, they're available in every OCI region and OCI Dedicated Region.

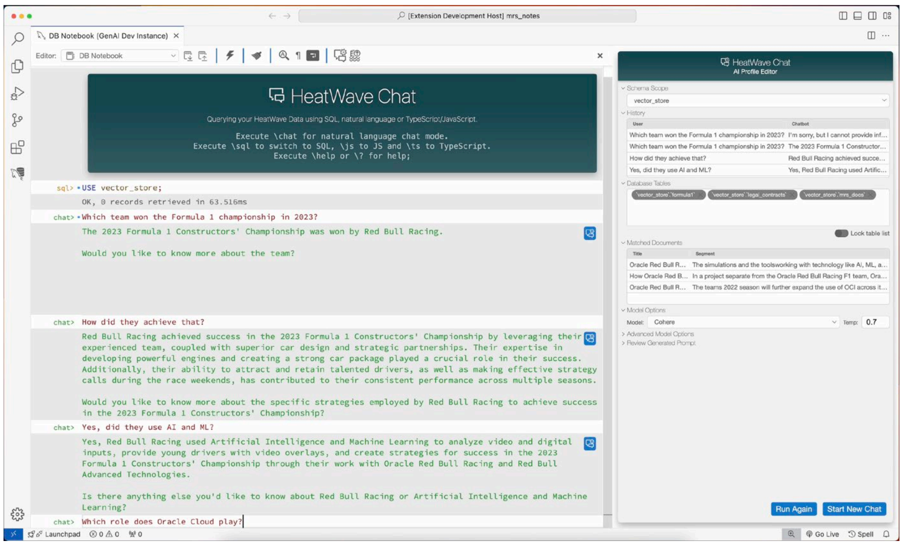
MySQL HeatWave Navigator

An intuitive GUI is available with the MySQL Shell for VS Code plug-in that enables users to browse documents in their object store buckets, upload documents from local storage to object storage, and ingest them into MySQL HeatWave Vector Store.



MySQL HeatWave Navigator (available with MySQL Shell for VS Code)

When using the interactive, natural language capabilities of MySQL HeatWave Chat, the MySQL HeatWave Navigator also provides a link to the documents in the vector store that matched the results' output. Users can select the schema to use with the chat editor and browse the documents already loaded.


[Extension Development Host] mrs_notes

DB Notebook (GenAI Dev Instance)

Editor: DB Notebook

HeatWave Chat

Querying your HeatWave Data using SQL, natural language or TypeScript/JavaScript.

```
Execute \chat for natural language chat mode.
Execute \sql to switch to SQL, \js to JS and \ts to TypeScript.
Execute \help or \? for help;

sql> •USE vector_store;
     OK, 0 records retrieved in 63.516ms
chat> •Which team won the Formula 1 championship in 2023?
     The 2023 Formula 1 Constructors' Championship was won by Red Bull Racing.

     Would you like to know more about the team?

chat> How did they achieve that?
     Red Bull Racing achieved success in the 2023 Formula 1 Constructors' Championship by leveraging their
     experienced team, coupled with superior car design and strategic partnerships. Their expertise in
     developing powerful engines and creating a strong car package played a crucial role in their success.
     Additionally, their ability to attract and retain talented drivers, as well as making effective strategy
     calls during the race weekends, has contributed to their consistent performance across multiple seasons.

     Would you like to know more about the specific strategies employed by Red Bull Racing to achieve success
     in the 2023 Formula 1 Constructors' Championship?
chat> Yes, did they use AI and ML?
     Yes, Red Bull Racing used Artificial Intelligence and Machine Learning to analyze video and digital
     inputs, provide young drivers with video overlays, and create strategies for success in the 2023
     Formula 1 Constructors' Championship through their work with Oracle Red Bull Racing and Red Bull
     Advanced Technologies.

     Is there anything else you'd like to know about Red Bull Racing or Artificial Intelligence and Machine
     Learning?
chat> Which role does Oracle Cloud play?
```

HeatWave Chat
AI Profile Editor

Schema Scope: vector_store

History

User	Chatbot
Which team won the Formula 1 championship in 2023?	I'm sorry, but I cannot provide inf...
Which team won the Formula 1 championship in 2023?	The 2023 Formula 1 Constructor...
How did they achieve that?	Red Bull Racing achieved succe...
Yes, did they use AI and ML?	Yes, Red Bull Racing used Artific...

Database Tables: `vector_store`.`formula1` `vector_store`.`legal_contracts` `vector_store`.`mrs_docs`

Lock table list

Matched Documents

Title	Segment
Oracle Red Bull R...	The simulations and the toolsworking with technology like AI, ML, a...
How Oracle Red B...	In a project separate from the Oracle Red Bull Racing F1 team, Ora...
Oracle Red Bull R...	The teams 2022 season will further expand the use of OCI across it...

Model Options

Model: Cohere Temp: 0.7

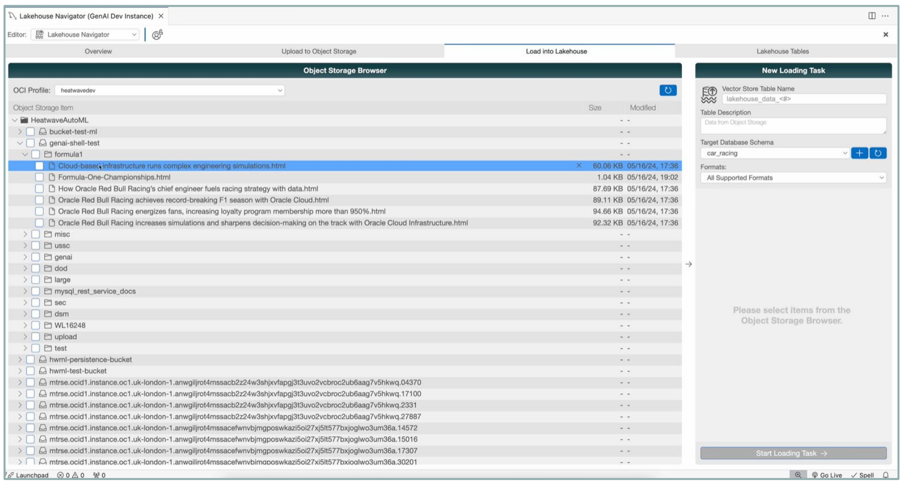
> Advanced Model Options
> Review Generated Prompt

Run Again Start New Chat

Launchpad ⓧ 0 ⚠ 0 0

MySQL HeatWave Chat interface with AI profile editor

When working with MySQL HeatWave Generative AI, in all the above scenarios, customers do not need any lengthy ETL processes between disparate systems, nor do they require the cloud application to be aware of the different data sources.



OCI Profile: heatwavedev

Object Storage Item	Size	Modified
HeatwaveAutoML		- -
bucket-test-ml		- -
genai-shell-test		- -
formula1		- -
Cloud-based infrastructure runs complex engineering simulations.html	60.06 KB	05/16/24, 17:36
Formula-One-Championships.html	1.04 KB	05/16/24, 19:02
How Oracle Red Bull Racing's chief engineer fuels racing strategy with data.html	87.69 KB	05/16/24, 17:36
Oracle Red Bull Racing achieves record-breaking F1 season with Oracle Cloud.html	89.11 KB	05/16/24, 17:36
Oracle Red Bull Racing energizes fans, increasing loyalty program membership more than 950%.html	94.66 KB	05/16/24, 17:36
Oracle Red Bull Racing increases simulations and sharpens decision-making on the track with Oracle Cloud Infrastructure.html	92.32 KB	05/16/24, 17:36
misc		- -
ussc		- -
genai		- -
dod		- -
large		- -
mysql_rest_service_docs		- -
sec		- -
dsm		- -
WL16248		- -
upload		- -
test		- -
hwml-persistence-bucket		- -
hwml-test-bucket		- -
mtrse.ocid1.instance.oc1.uk-london-1.anwgiljrot4mssacb2z24w3shjxvfapgj3t3uvo2vcbroc2ub6aag7v5hkwq.04370		- -
mtrse.ocid1.instance.oc1.uk-london-1.anwgiljrot4mssacb2z24w3shjxvfapgj3t3uvo2vcbroc2ub6aag7v5hkwq.17100		- -
mtrse.ocid1.instance.oc1.uk-london-1.anwgiljrot4mssacb2z24w3shjxvfapgj3t3uvo2vcbroc2ub6aag7v5hkwq.2331		- -
mtrse.ocid1.instance.oc1.uk-london-1.anwgiljrot4mssacb2z24w3shjxvfapgj3t3uvo2vcbroc2ub6aag7v5hkwq.27887		- -
mtrse.ocid1.instance.oc1.uk-london-1.anwgiljrot4mssacefwnvbjmgposwkazi5oi27xj5lt577bxjoglwo3um36a.14572		- -
mtrse.ocid1.instance.oc1.uk-london-1.anwgiljrot4mssacefwnvbjmgposwkazi5oi27xj5lt577bxjoglwo3um36a.15016		- -
mtrse.ocid1.instance.oc1.uk-london-1.anwgiljrot4mssacefwnvbjmgposwkazi5oi27xj5lt577bxjoglwo3um36a.17307		- -
mtrse.ocid1.instance.oc1.uk-london-1.anwgiljrot4mssacefwnvbjmgposwkazi5oi27xj5lt577bxjoglwo3um36a.30201		- -

Vector Store Table Name: lakehouse_data_<#>
Table Description: Data from Object Storage
Target Database Schema: car_racing
Formats: All Supported Formats

Please select items from the Object Storage Browser.

Lakehouse Navigator displaying object storage content

Performance and Price-Performance

A MySQL HeatWave whitepaper would likely be incomplete without published benchmark results… The benchmark is designed to answer common questions customers face when evaluating a new service. We share performance numbers for both structured and unstructured content.

Structured and semi-structured content

MySQL HeatWave Lakehouse supports structured and semi-structured content in object storage in several popular formats like CSV, Avro, Parquet, and JSON.

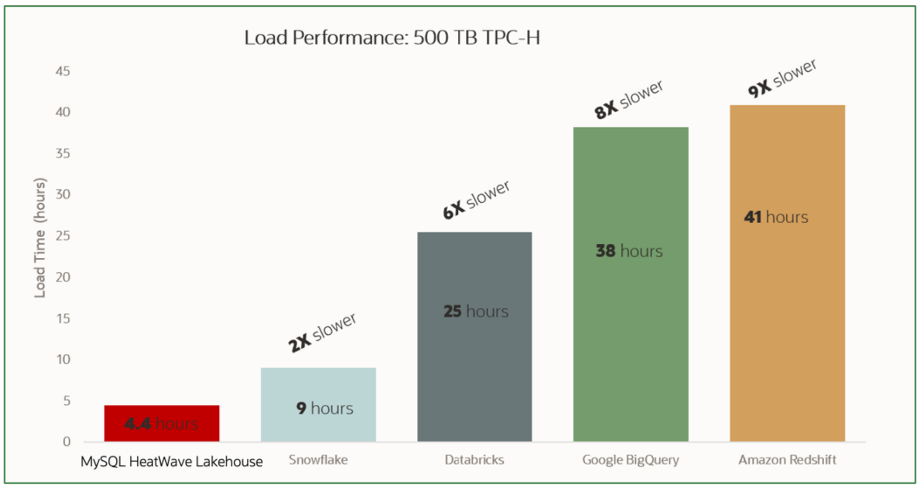
Load Performance

Load performance on the 100 TB TPC-DS data:

TPC-DS 100TB	MYSQL HEATWAVE	SNOWFLAKE 3X LARGE	REDSHIFT 10 RA3.16XLARGE	BIGQUERY 3200 SLOTS	DATABRICKS 2XLARGE
Hourly cost ($)	56.43	128	86.06	74.56	103.39
Load time (hrs)	1.21	3.3	7.74	3.63	67.46
MySQL HeatWave load advantage		**2.7x**	**6.4x**	**3x**	**6.1x**
Total time (seconds)	3,719	5,379	5,108	11,694	13,704
Price-perf($)	58	191	122	242	6.394
MySQL HeatWave price-perf advantage		**3.3x**	**2.1x**	**4.1x**	**6.8x**

Load and query performance comparison of MySQL HeatWave with other vendors

Load performance with 500 TB TPC-H data:



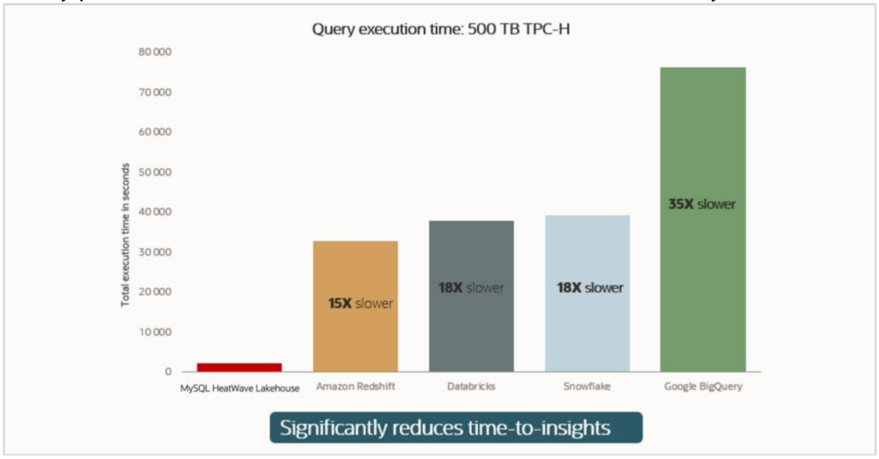
**Benchmark queries are derived from the TPC-H benchmarks, but results are not comparable to published TPC-H benchmark results since these do not comply with the TPC-H specifications.*

Configuration: MySQL HeatWave Lakehouse: 512 nodes; Snowflake: 4X-Large Cluster; Databricks: 3X-Large Cluster; Amazon Redshift: 20-ra3.16xlarge; Google BigQuery: 6400 slots

Such record speed is possible because of the scale-out architecture of our processes that partition and balance tasks and utilize all the available CPU cores to get external files query-ready, guaranteeing that all the 512 nodes in the cluster are used in-tandem, ensuring massive scalability.

Query Performance

Query performance on a 500 TB TPC-H dataset loaded into MySQL HeatWave:

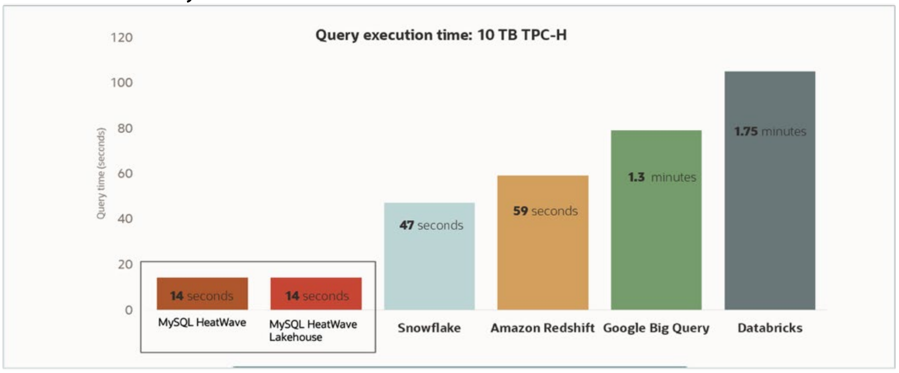


**Benchmark queries are derived from the TPC-H benchmarks, but results are not comparable to published TPC-H benchmark results since these do not comply with the TPC-H specifications.*

Configuration: MySQL HeatWave Lakehouse: 512 nodes; Snowflake: 4X-Large Cluster; Databricks: 3X-Large Cluster; Amazon Redshift: 20-ra3.16xlarge; Google BigQuery: 6400 slots

Identical performance to query from object storage and database

Querying the data in object storage is as fast as querying the databases, as demonstrated by a 10 TB TPC-H benchmark:

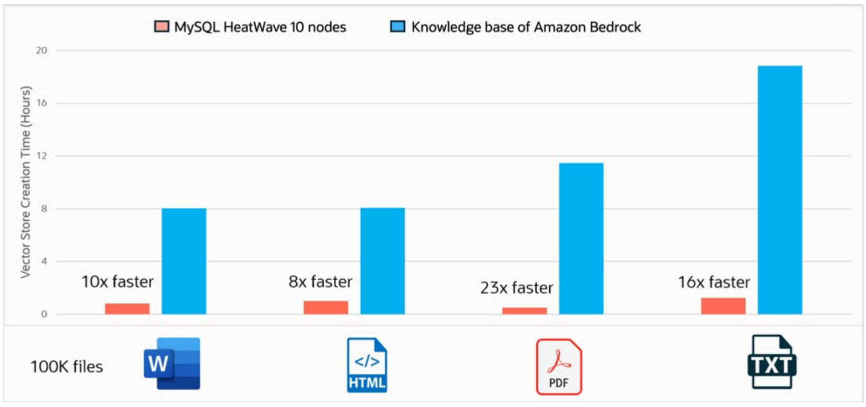


MySQL HeatWave query performance on 10 TB TPC-H benchmark

MySQL HeatWave performance with unstructured content

Load performance for vector store creation

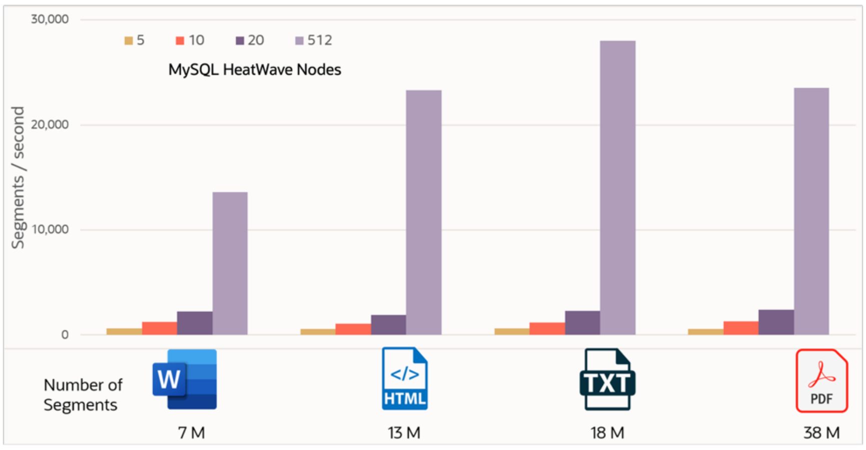
When creating a vector store using 100,000 files with unstructured documents loaded from object storage, MySQL HeatWave is as much as 23X faster than Knowledge Bases for Amazon Bedrock, at ¼ th the cost.



Vector store creation in MySQL HeatWave for 100k files compared to Knowledge base for Amazon Bedrock

MySQL HeatWave scalability with vector store creation

When loading a massive number of documents from object storage into MySQL HeatWave Vector Store, MySQL HeatWave can scale all the way to 512 nodes on a single cluster, processing 223 million segments from 6.8 million HTML documents created in 1.7 hours, averaging more than 35,000 segments per second.



Vector store creation in MySQL HeatWave for different file types scales out to 512 nodes

Query performance on vector store

SIMILARITY SEARCH	MYSQL HEATWAVE	SNOWFLAKE	DATABRICKS	BIGQUERY	AURORA POSTGRES
Hourly cost ($)	1.52	2	9.8	4	4.64
Total time (s)	16	466	238	288	402
MySQL HeatWave perf advantage		**30x**	**15x**	**18x**	**25x**

Similarity search comparison of MySQL HeatWave with other vendors

- Configuration:
  - MySQL HeatWave - 1MySQL + 1 MySQL HeatWave node
  - Snowflake: X-small cluster
  - Databricks: 25 units + 2X-small
  - BigQuery: 100 slots
  - Aurora: db.r5.8xlarge

Conclusion

With the deluge of data created outside of databases (social media files, data from IoT sensors, connected devices, web application telemetry, and other sources) businesses want to rapidly generate new insights and apply machine learning operations to train their data, make predictions, and explain results. With MySQL HeatWave Lakehouse, customers can leverage all the benefits of MySQL HeatWave and the convenience of familiar MySQL commands on data residing in object storage. As demonstrated by several benchmarks, including the 100 TB TPC-DS and 500 TB TPC-H benchmarks, MySQL HeatWave Lakehouse delivers superior query performance, price-performance, and load performance compared to other available offerings. MySQL HeatWave provides a single, fully managed service for transaction processing, analytics across data warehouses and data lakes, machine learning, and generative AI. MySQL HeatWave Lakehouse is available on OCI, AWS, and Microsoft Azure.