ORACLE

MySQL HeatWave

Real-time Analytics for MySQL Database Service

August 2021, Version 3.0

Purpose statement

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

Disclaimer

Benchmark queries are derived from the TPC-H benchmark, but results are not comparable to published TPC-H benchmark results since they do not comply with the TPC-H specification.

Table of contents

Executive Summary

MySQL is the world's most popular open source database because of its reliability, high-performance, and ease of use. It powers the world's most trafficked web sites including Facebook, Twitter, LinkedIn and Booking.com.

The companies that will thrive in the evolving digital landscape, will be those that make data and analytics a core part of their strategy. According to McKinsey[1], 92% of company leaders surveyed believed that their business model would not remain viable at the current rate of digitization. This fear of disruption is the leading driver behind the investment in modern data and analytics platforms.

Oracle MySQL Database Service is a fully managed database service that lets developers quickly develop and deploy secure cloud native applications using the world's most popular open source database. MySQL Database Service is the only MySQL service with a massively-scalable integrated real-time query acceleration engine: HeatWave. This service, only available in Oracle Cloud Infrastructure, overcomes the limitations of traditional data warehouse and analytics environments that use periodic long-running ETL batch jobs to refresh the data. MySQL HeatWave provides a unified MySQL database platform for both analytics and mixed workloads.

"By 2022, public cloud services will be essential for 90% of data and analytics innovation."

Gartner
The IT Roadmap for Data and Analytics

Challenges of Existing Data & Analytics Solutions

MySQL is optimized for OLTP, but it is not designed for analytic processing (OLAP). As a result, organizations which need to efficiently run analytics need to move their data to another database.

This approach of moving data to another database introduces complexity and additional cost to customers in multiple ways:

1. **Applications need to define complex logic** for extracting relevant data from MySQL.
2. **The extracted data needs to be transported to another database** across networks securely, consuming network bandwidth and incurring latency.
3. **Data in the other database needs to be manually kept in sync** with the MySQL database and as a result the data on which analytics is performed can become stale.
4. **Additional cost and overhead of managing multiple databases** for running OLTP and analytics applications.

Performance: Real Time Analytics

HeatWave is designed to enable customers to run analytics on data which is stored in MySQL databases, without the need for ETL. This service is built on an innovative, in-memory analytics engine which is architected for scalability and performance and is optimized for Oracle Cloud Infrastructure (OCI). This results in a very performant solution for SQL analytics at a fraction of the cost compared to other industry solutions.

Compared to other MySQL solutions, with a 4TB TPC-H Benchmark workload, HeatWave is much faster and cheaper:

HeatWave advantages on 4TB TPC-H			
Other MySQL solution	**Speedup**	**Price Performance**	**Cost**
Amazon Aurora	1400x	2800x	1/2
Amazon RDS for MySQL	5400x	8100x	2/3

Figure 1. Summary of HeatWave performance and cost advantage over other MySQL solutions on 4TB TPC-H workload

Customers with HeatWave will experience a 5400x speedup over Amazon RDS for MySQL and pay 1/3 less.

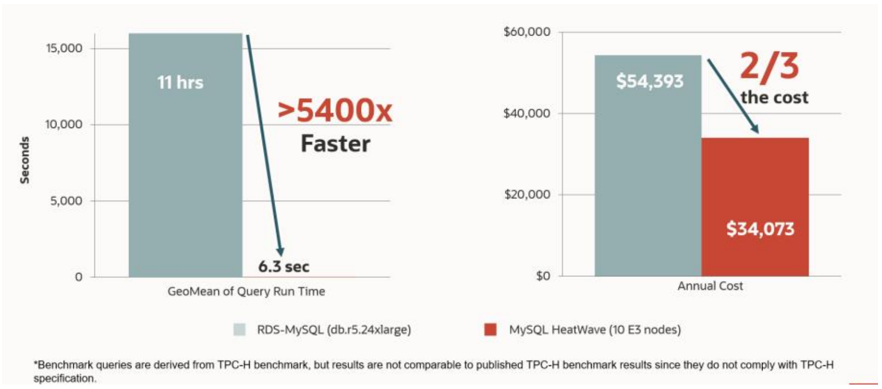


Figure 2. HeatWave accelerates Amazon RDS for MySQL queries by 5400x

Compared to Amazon Aurora, MySQL HeatWave offers dramatic improvement in performance for complex and analytic queries. MySQL HeatWave is more than 1400x faster than Amazon Aurora and costs half of Aurora. Furthermore, with HeatWave there is no need to create indexes on the base table which takes days with Amazon Aurora. As a result, the data is available to query much sooner with HeatWave than with Aurora.

"We successfully migrated our 6TB database and in-house digital marketing and media management applications from AWS Aurora to MySQL HeatWave on OCI that reduced our costs by 60 percent and improved performance for complex queries by more than 1000x and overall workloads improved 85 percent."

Amit Palshikar
Co-Founder, CTO
Red3i

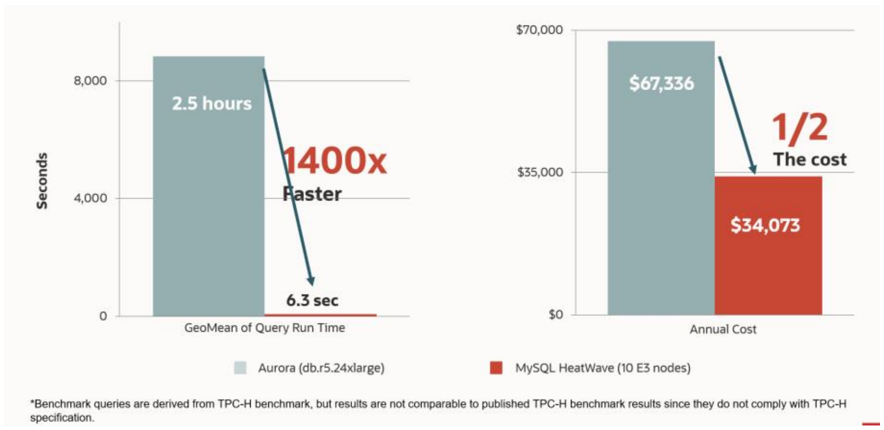


Figure 3. HeatWave is 1400x faster than Amazon Aurora

“We found MySQL HeatWave improved performance by 10 times and significantly dropped our costs after migrating from AWS Aurora. We also did not have to modify our application for a great experience.”

Kanami Suzuki
Developer
Fan Communications

The performance improvement of MySQL HeatWave over Aurora increases with the size of data.

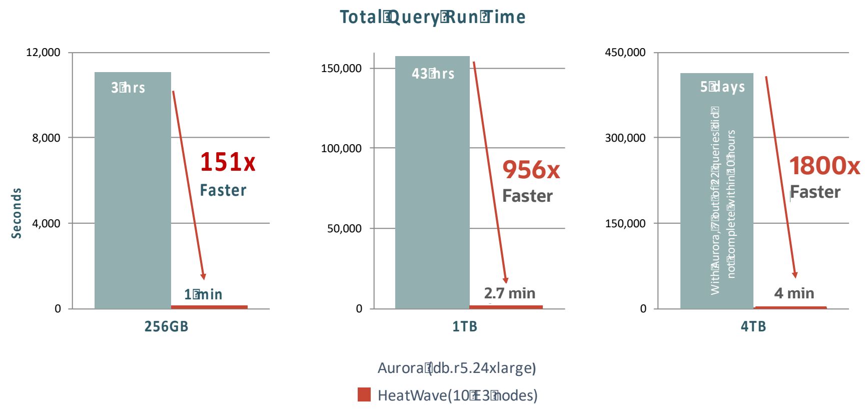


Figure 4. The performance advantage of HeatWave increases with data size vs. Amazon Aurora

MySQL HeatWave is faster than all the competing analytics services at a fraction of the cost.

HeatWave advantages on 10TB TPC-H			
Other specialized analytics solution	**Speedup**	**Price Performance**	**Cost**
Snowflake on AWS	6.8x	35x	1/5
Amazon Redshift with AQUA	6.8x	13x	1/2

Figure 5. Summary of HeatWave performance and cost advantages over other analytics specific solutions on 10TB TPC-H workload

HeatWave advantages on 30TB TPC-H			
Other specialized analytics solution	**Speedup**	**Price Performance**	**Cost**
Snowflake on AWS	6.8x	45x	1/6
Amazon Redshift	2.8x	17x	1/6
Google BigQuery	9x	36x	1/4
Azure Synapse	3x	15x	1/5

Figure 6. Summary of HeatWave performance and cost advantages over other analytics specific solutions on 30TB TPC-H workload

Amazon Redshift, which is designed for analytics, is offered in multiple shapes. With their latest instance shape (ra3.4xlarge) and latest advanced query accelerator (AQUA) turned on, HeatWave is 6.8x faster at half the cost.

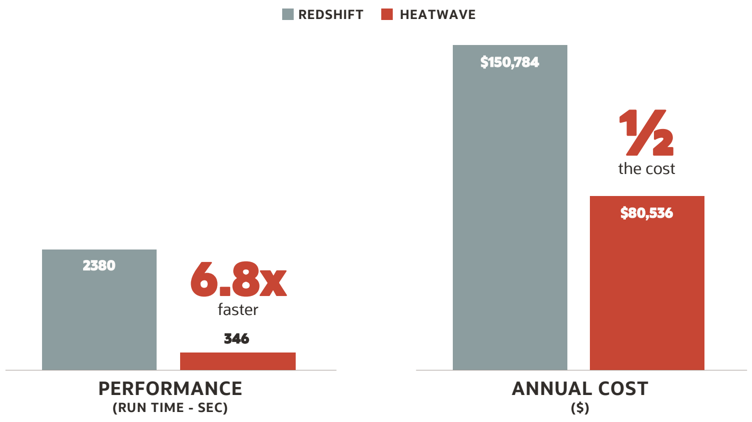


Figure 7. HeatWave performance and cost advantages Amazon RedShift AQUA on 10TB and 30TB TPC-H workload

Also, unlike Amazon Redshift, HeatWave is capable of both OLTP and OLAP, without the need for ETL.

Customers who use HeatWave will benefit from significantly better performance, eliminating the need for ETL, support for real-time analytics, reduced monthly cost and a single database for OLTP and OLAP.

Mixed workload

Most real-world applications have a mix of OLTP and complex OLAP queries. For such workloads, MySQL HeatWave is much faster and costs a fraction of Amazon Aurora.

Using industry standard CH-benCHmark on a 100GB dataset:

For OLAP queries: HeatWave is 18x faster, provides 110x better throughput and is 2.4x cheaper than Aurora for OLAP queries resulting in 42x better price performance.

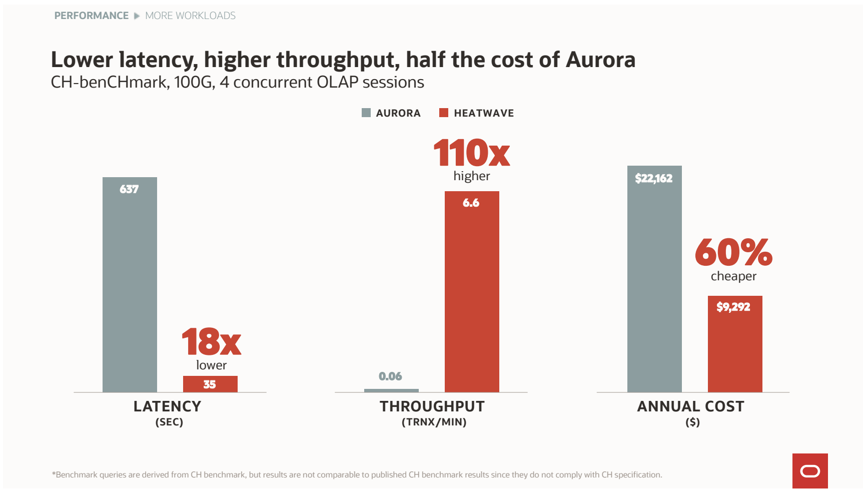


Figure 7. HeatWave advantages over Amazon Aurora for OLAP queries on 100GB mixed workload

"MySQL HeatWave dramatically reduced our AWS Aurora and Redshift cost by more than 50 percent, we are no longer moving data around so now we have blazing fast, real-real-time insights with no effort. More importantly, scalability has made our expansion plan possible, allowing us to onboard more data and new clients without impact to costs. It's a dream come true."

Pablo Lemos
Co-Founder, CTO
Tetris.co

For OLTP queries, MySQL HeatWave has the same performance as Aurora and costs 2.4x less resulting in 2.4x better price performance.

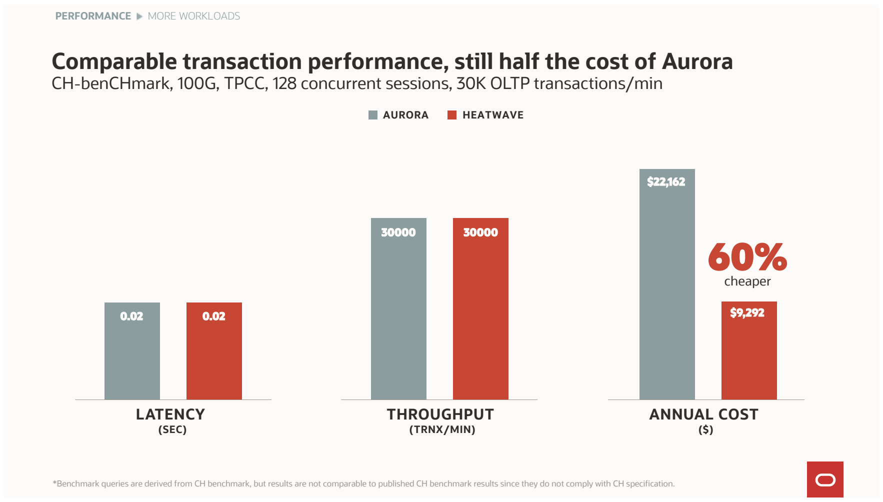


Figure 8. HeatWave advantages over Amazon Aurora for OLTP queries on 100GB mixed workload

“We recently migrated our production workload from another cloud solution to MySQL HeatWave. Doing so reduced our cost by 3x and it also significantly accelerated many of our queries which were taking a long time before.”

Chien Hoang
Director of Engineering
Tamara.co

Deployment Scenarios

HeatWave is a fully managed service and is available exclusively in the Oracle Cloud Infrastructure. The MySQL database has been enhanced to natively integrate this service, as a result, customers who store data in MySQL can seamlessly run analytics by enabling this service.

A HeatWave instance is a cluster composed of a MySQL Database Service (MDS) instance and multiple HeatWave nodes. When HeatWave is enabled, HeatWave server is installed on the MDS node. It is responsible for cluster management, loading data into the memory of the HeatWave nodes, query scheduling and query execution. MySQL applications written in Java, PHP, Ruby, etc. work seamlessly with HeatWave using standard MySQL ODBC/JDBC connectors. HeatWave is an in-memory accelerator and supports all MySQL syntax. Hence, all existing tools and applications built using standard SQL will work without requiring any modification to queries (Figure 9).

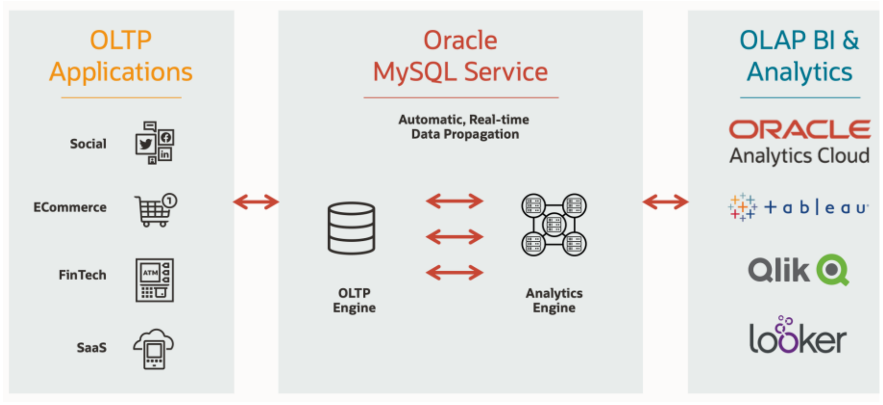


Figure 9. Existing tools and applications work seamlessly with HeatWave

Data which is needed for analytic processing is stored in memory of the HeatWave nodes, in a hybrid columnar compressed format. The number of nodes needed to run a workload depends on the amount of data present for analytic processing, the compression factor which is achieved on the dataset, and the query characteristics. The number of nodes needed can be automatically derived by using the Auto Provisioning advisor which is available with HeatWave.

HeatWave currently supports up to 64 nodes in one cluster and with a processing capacity of approximately 32 TB of analytics data. 32TB is the approximate amount of data which can be populated in the memory of the HeatWave nodes at a given moment. There is no limit to the amount of data which can be stored in the MySQL database and customers can choose which tables or columns from MySQL database schema to load into the memory of HeatWave nodes. If the tables are no longer needed by queries, user can remove the tables from the memory to make room for other data.

HeatWave provides a great solution for customers who need to run both transactional and analytical workloads. While transactional queries are run in the MySQL node, data updated in MySQL InnoDB is transparently propagated to the HeatWave cluster for accelerated analytical processing. This enables customers to run both OLTP and real-time analytics workloads simultaneously within a single database platform (Figure 10).

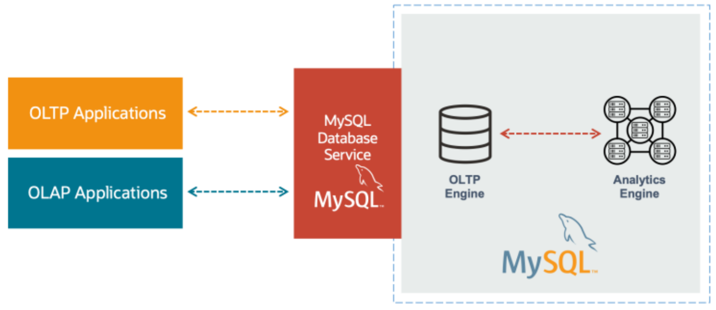


Figure 10. Single MySQL database for OLTP & OLAP applications, no ETL required.

On-premises customers who cannot move their MySQL deployment to a Cloud due to compliance or regulatory requirements, can still leverage HeatWave by using the hybrid deployment model as shown in Figure 11. In such a hybrid deployment, customers can leverage MySQL replication to replicate on-premises MySQL data to HeatWave without the need for ETL.

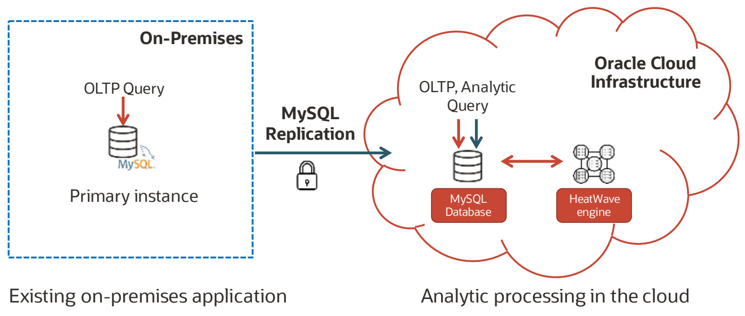


Figure 11. Hybrid deployment for enabling analytics on data stored on premise

HeatWave Architecture

There are four key architectural choices which lead to a compelling performance and cost advantage with HeatWave:

1. Innovative in-memory hybrid columnar analytics engine designed for scalability and performance which implements state of the art algorithms.

2. Optimized for Oracle Cloud Infrastructure to provide the best price performance database service based on commodity hardware.

3. HeatWave scale out data management layer enables HeatWave representation of data to be stored in OCI object storage, allowing fast data reload for operations like error recovery, maintenance and system restart to increase service uptime.

4. MySQL Autopilot provides machine learning automation that improves performance, scalability, and ease of use of HeatWave. It automates the database lifecycle operations including provisioning, data loading, query processing, and error handling.

Heatwave engine uses a columnar in-memory representation that facilitates vectorized processing, leading to very good query performance (Figure 12). The data is encoded and compressed prior to being loaded in memory. This compressed and optimized in memory representation is used for both numeric and string data. This results in significant performance speed up and reduced memory footprint which translates to reduced cost for customer.

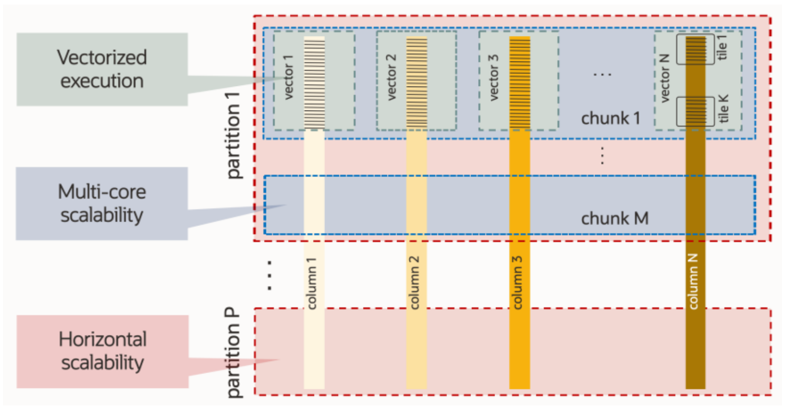


Figure 12. Vectorized in-memory columnar representation for analytic processing

One of the key design points of the HeatWave engine is to massively partition data across a cluster of HeatWave nodes, which can be operated upon in parallel in each node. This enables high cache hits for analytic operations and provides good inter-node scalability. Each HeatWave node within a cluster and each core within a node can process partitioned data in parallel, including parallel scans, joins, group-by, aggregation and top-k processing.

HeatWave has implemented state of the art algorithms for distributed in-memory analytic processing. Joins within a partition are processed fast by using vectorized build and probe join kernels. The highly-optimized network communication between analytics nodes is achieved by using asynchronous batch I/Os. The algorithms are designed to overlap compute time with communication of data across nodes which helps achieve good scalability (Figure 13).

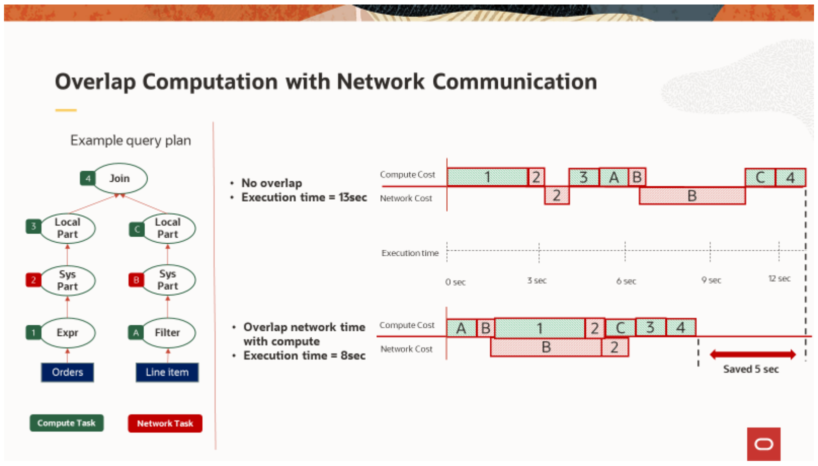


Figure 13. HeatWave implements state of art algorithms for distributed in-memory analytic processing.

Native MySQL Analytics

Integration of HeatWave with MySQL Database Service provides a single data management platform for all OLTP and analytics needs of an enterprise. HeatWave is designed as a MySQL pluggable storage engine, which completely shields all the low-level implementation details at the storage level from the end users. As a result, users can manage both HeatWave and MySQL Database Service with the same management tools including OCI console, REST API, and command line interface.

Since HeatWave is an in-memory processing engine, data is persisted in MySQL InnoDB storage engine. This allows users to manage analytics data the same way they manage transactional data in MySQL.

Users and applications interact with HeatWave through the MySQL database node in the cluster. Users connects to HeatWave through standard tools and standard-based ODBC/JDBC connectors. HeatWave supports the same ANSI SQL standard and ACID properties as MySQL and supports diverse data types. This enables existing applications to take advantage of HeatWave without any changes to their application, allowing easy and quick integration.

Once users submit a query to the MySQL database, the MySQL query optimizer transparently decides if the query should be offloaded to HeatWave cluster for accelerated execution. This is based on whether all operators and functions referenced in the query are supported by HeatWave and if the estimated time to process the query with HeatWave engine is less than with MySQL. If both conditions are met, the query is pushed to HeatWave nodes for processing. Once processed, the results are sent back to the MySQL database node and returned to users (Figure 14).

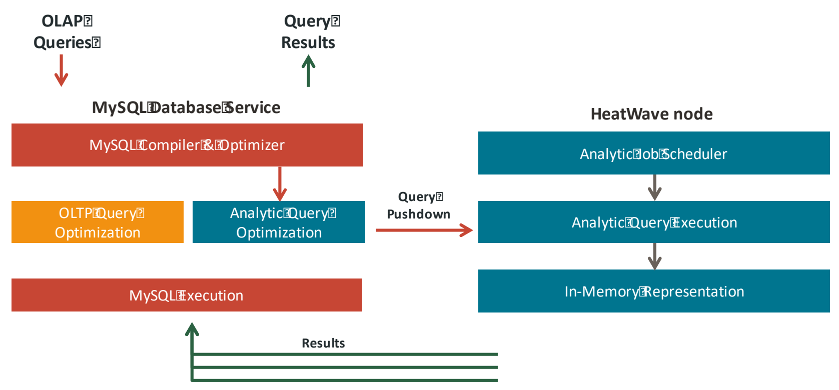


Figure 14. MySQL Integration for Query Processing

Any updates to the tables in InnoDB that are also loaded in HeatWave, are automatically propagated to the memory of the HeatWave nodes in real time. This allows subsequent queries to always have access to the latest data, as shown in Figure 15. This is done behind the scene by a light-weight change propagation algorithm that can keep up with MySQL data update rates.

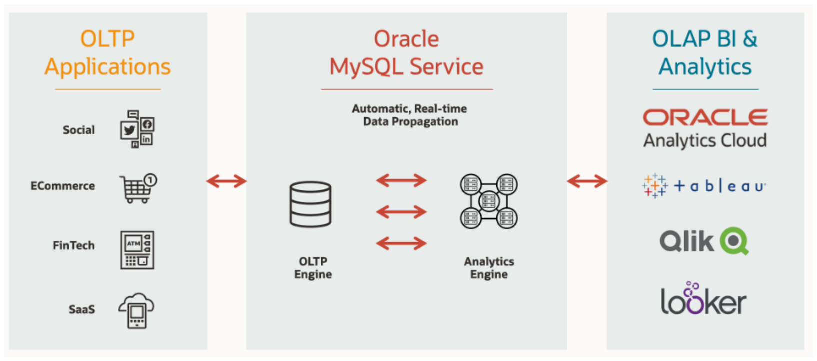


Figure 15. HeatWave integration with MySQL Database Service

Scale-out Data Management

MySQL InnoDB database stores data in row-based format, while HeatWave stores data in memory in a hybrid columnar format. Loading data from MySQL to HeatWave involves transforming data into HeatWave columnar format, which can take considerable amount of time depending on data size. Since data is stored in memory in HeatWave, operations like error recovery, maintenance and system restart can take a long time as data needs to be re-transformed and reloaded after the cluster is ready.

To improve service uptime, HeatWave introduced a new storage layer that is built on OCI Object Storage. This new architecture enables storing HeatWave formatted data in a persistent storage, allowing reload of data in constant time regardless of data size.

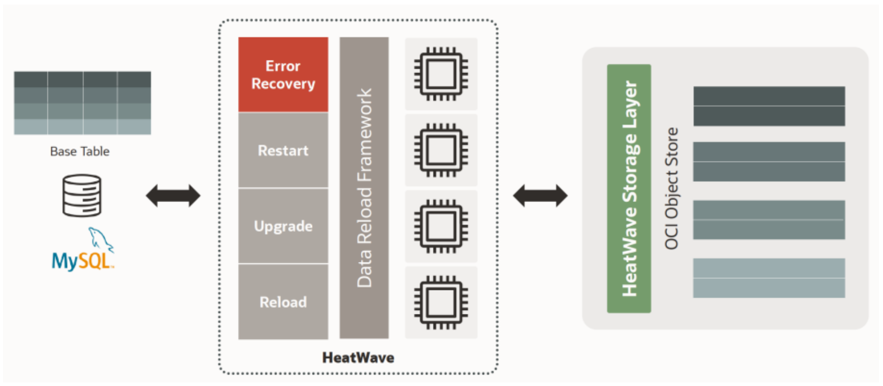


Figure 16. HeatWave scale-out storage layer

Figure 16 shows the new HeatWave architecture with the scale-out storage layer. In the HeatWave storage layer, persisted data is organized in the same way as that of in-memory data. Each HeatWave node can restore data independently and in parallel, allowing a very fast and near constant time data reload.

MySQL Autopilot

MySQL Autopilot automates many of the most important and often challenging aspects of achieving high query performance at scale - including provisioning, data loading, query execution and failure handling. It uses advanced techniques to sample data, collect statistics on data and queries, and build machine learning models to model memory usage, network load and execution time. These machine learning models are then used by MySQL Autopilot to execute its core capabilities. MySQL Autopilot makes the HeatWave query optimizer increasingly intelligent as more queries are executed, resulting in continually improving system performance over time.

Autopilot focuses on four aspects of the service lifecycle: system setup, data load, query execution and failure handling (Figure 17).

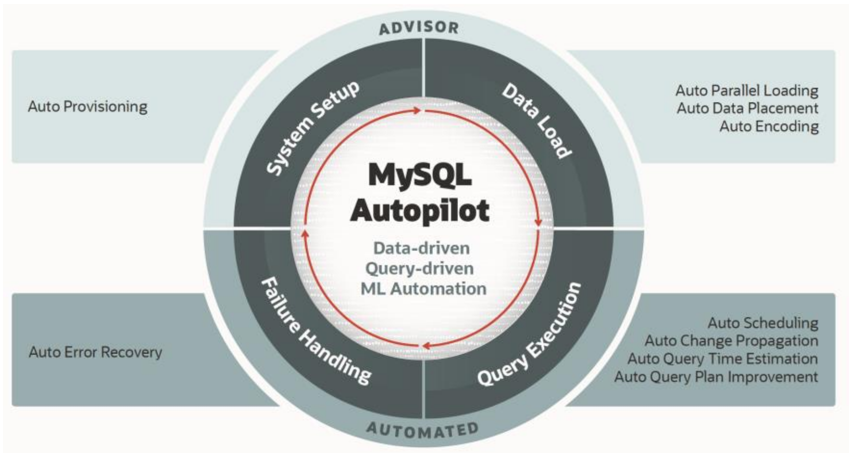


Figure 17. MySQL Autopilot automates different aspects of the service to improve performance, scalability and usability of the system

System Setup

1. ***Auto provisioning*** predicts the number of HeatWave nodes required for running a workload by adaptive sampling of table data on which analytics is required. This means that customers no longer need to manually estimate the optimal size of their cluster.

Data Load

2. ***Auto parallel loading*** optimizes the load time and memory usage by predicting the optimal degree of parallelism for each table being loaded into HeatWave.
3. ***Auto encoding*** determines the optimal representation of columns being loaded into HeatWave taking queries into consideration. This optimal representation provides the best query performance and minimizes the size of the cluster to minimize the cost.
4. ***Auto data placement*** predicts the column on which tables should be partitioned in-memory to achieve the best performance for queries. It also predicts the expected gain in query performance with the new column recommendation.

Query Execution

5. ***Auto query plan improvement*** learns various statistics from the execution of queries and improves the execution plan of future queries. This improves the performance of the system as more queries are run.
6. ***Auto query time estimation*** estimates the execution time of a query prior to executing the query, allowing quick tryout and test on different queries
7. ***Auto change propagation*** intelligently determines the optimal time when changes in MySQL Database should be propagated to the HeatWave storage layer. This ensures that changes are being propagated at the right optimal cadence.
8. ***Auto scheduling*** determines which queries in the queue are short running and prioritizes them over long running queries in an intelligent way to reduce overall wait time.

Failure Handling

9. ***Auto error recovery:*** Provisions new nodes and reloads necessary data from the HeatWave storage layer if one or more HeatWave nodes is unresponsive due to software or hardware failure

Auto Provisioning

Auto Provisioning provides recommendation on how many HeatWave nodes are needed to run a workload. When the service is started, database tables on which analytics queries are run need to be loaded to HeatWave cluster memory. The size of the cluster needed depends on tables and columns required to load, and the compression achieved in memory for this data. Figure 18 compares the traditional (i.e., manual) approach to estimating the cluster size with Auto Provisioning. In traditional provisioning, users need to guess a cluster size. Underestimation results in data load or query execution failure due to space limitations. Overestimation results in additional costs for unneeded resources. As a result, users iterate until they determine the right cluster size, and this size estimate becomes inaccurate when tables are updated.

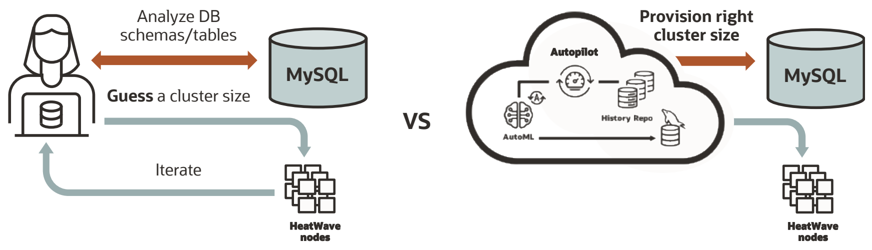


Figure 18. Comparison of a manual provisioning vs Auto provisioning

The right side of figure 18 shows how auto provisioning, a ML-based cluster size estimation advisor, solves this problem. By leveraging well trained and accurate ML models, the user consults auto provisioning advisor to obtain the right cluster size for their dataset. As a result, users do not need to guess the cluster size. Later, if the customer data grows or additional tables are added, the users can again take advantage of the auto provisioning advisor.

Auto Parallel Load

Loading data into HeatWave involves several manual steps. The time required to perform these steps depends on the number of schemas, tables, and columns, and statistics. Auto parallel load automates these steps by predicting the degree of parallelism per table via machine-learning models to achieve optimal load speed and memory usage.

Auto Encoding

HeatWave supports two string column encoding types: variable-length and dictionary. The type of encoding affects the query performance as well as the supported query operations. It also affects the amount of memory required for HeatWave nodes. By default, HeatWave applies variable-length encoding to string columns when data is loaded, which may not be the optimal encoding choice for query performance and cluster memory usage for certain workloads.

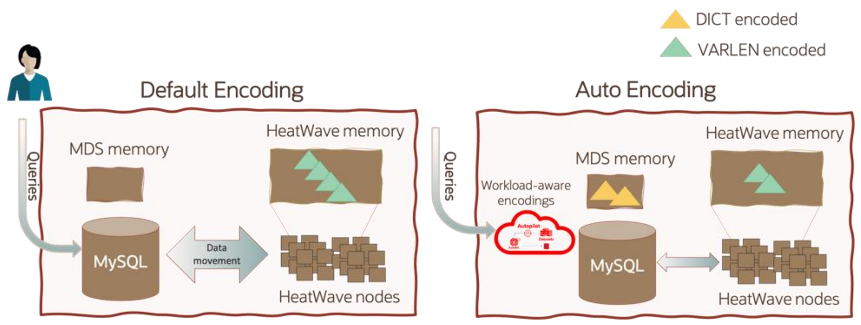


Figure 19. Comparison of a default encoding vs auto encoding

Auto encoding provides recommendations for string columns that reduce memory usage and help improve query performance. Figure 19 shows the difference between default encoding and auto encoding. In the default case, the variable-length encoding ensures best query offload capability. However, this can impact ideal query performance due to data movement between HeatWave nodes and MySQL nodes. Auto encoding uses machine learning to analyze column data, HeatWave query history, and available MySQL node memory to identify which string columns can be coded in dictionary encoding. When the suggestion is applied, the overall query performance is improved due to reduced data movement in system, and HeatWave memory usage is reduced due to efficient (i.e., smaller) dictionary codes and the corresponding dictionaries that maintain the mapping between the strings and the codes reside in the memory of the MDS node.

Auto Data Placement

Data placement keys are used to partition table data when loading tables into HeatWave. Partitioning table data by JOIN and GROUP BY key columns can improve query performance by avoiding costs associated with redistributing data among HeatWave nodes at query execution time.

Determining the best data placement key is a tedious task requiring understanding query access patterns and system behavior. Moreover, picking wrong partitioning keys can lead to sub-optimal performance due to increased data distribution costs during query execution time.

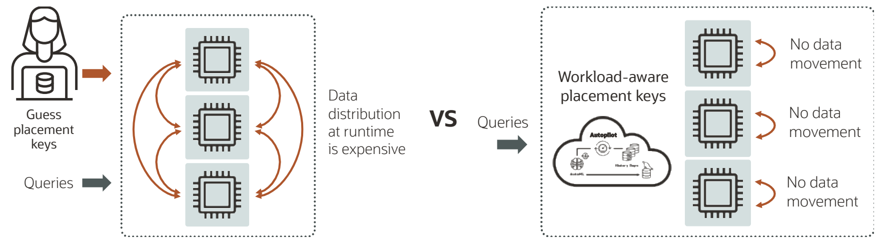
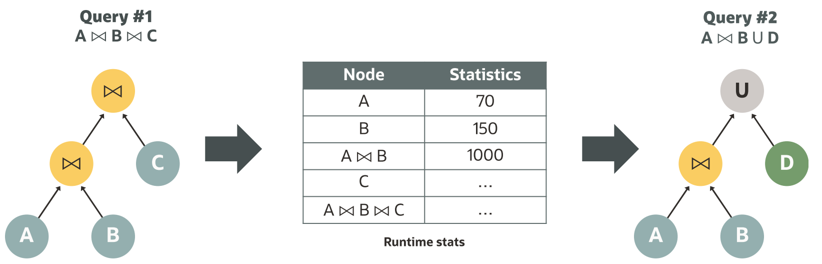


Figure 20. Comparison of manual data placement vs auto data placement

Figure 20 depicts a comparison between default query execution and execution with auto data placement. Based on machine learning models, auto data placement recommends appropriate data placement keys by analyzing table statistics and HeatWave query history and provides an estimation of query performance improvement. Once the suggestion is applied, query performance is improved by minimizing the data movement between nodes during execution.

Auto Query Plan Improvement

Auto query plan improvement enhances query performance by improving query plan statistics based on previous query executions. By maintaining more accurate query statistics, HeatWave creates better query plan and makes better decisions on the underlying physical operators; consequently improves the overall query performance.



Node	Statistics
A	70
B	150
A ⋈ B	1000
C	…
A ⋈ B ⋈ C	…

Figure 21. Query 2 benefits from statistics of a previous similar query (query 1) with auto query plan improvement

Figure 21 shows how auto query plan improvement works without user intervention. After a query (Q1) executes on HeatWave, auto query plan improvement collects and stores the cardinalities of all operations in the query execution plan (e.g., scan, join, group by). When a similar (or identical) query arrives (Q2), the system checks whether it can take advantage of the previously collected statistics information for Q2. If the system determines a similarity between the two query plans, a better query plan is generated based on statistics information from Q1. In doing so, it improves query performance and cluster memory usage significantly.

Auto Query Time Estimation

Users are often interested in accurate query time estimates before running the query. Such functionality allows users to estimate their application performance better and to understand the resource needed. Auto query time estimation not only provides user-visible estimations for query run times, but it also uses the same building blocks internally to improve query performance by optimizing query (sub-)plans.

Instead of using static, analytical models, auto query time estimation integrates a data-driven query time estimation module, which improves as queries run. To do so, HeatWave leverages load- and run-time statistics and dynamically tunes query cost models during execution. As a result, auto query time estimation improves with time as more queries are executed on the system.

Auto Change Propagation

Data updated in MySQL is propagated and persisted to HeatWave data layer as change logs. During data reload, HeatWave first restores data from the base data, then applies the data from the change logs. Over time, the persisted change log volume increases, which can result in an increased reload time as all the change logs need to be applied to the base data. So, the change logs are consolidated from time-to-time to alleviate increased reload latency as shown in Figure 22. However, determining when to consolidate is not an easy task, which depends on several factors such as transaction rate, system load, failure probability.

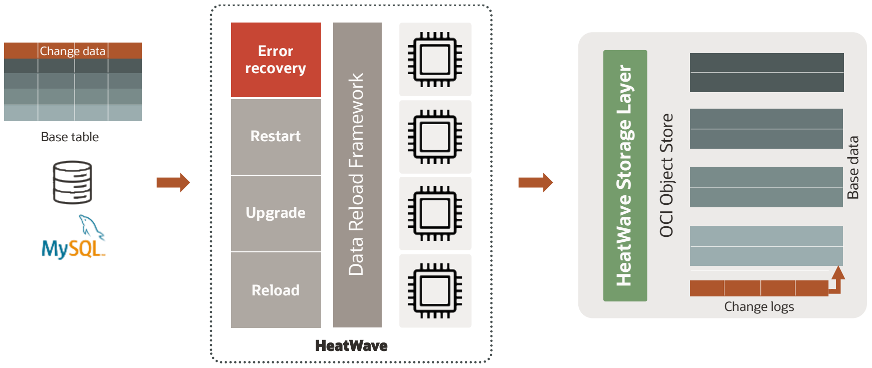


Figure 22. Auto change propagation

To minimize consolidation time during reloading from the storage layer, auto change propagation uses data driven mechanism to determine the best change propagation interval and choice. Auto change propagation analyzes rate of changes, incoming DMLs, object storage resources, and previously seen change activity. As a result, the changes are propagated at the best time interval, which results in optimized consolidation time for critical system operations.

Auto Scheduling

Traditional database systems process queries based on their arrival time, which can result in long-running queries starving short-running queries, as shown in Figure 23.

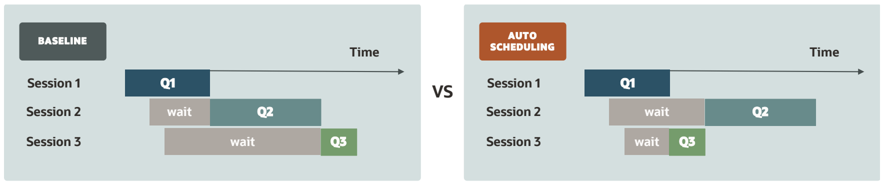


Figure 23. Traditional database system vs HeatWave auto scheduling

On the left, is a sub-optimal case where three queries (Q1, Q2, Q3) from three user sessions arrive one after the other and are scheduled in the FIFO order. After the execution completes, one can identify that waiting time for Q3 could be reduced significantly with minimal impact on Q2 latency.

On the right, it shows how auto scheduling improves user experience for short running queries in a multi-session application. Auto scheduling identifies and prioritizes short-running queries by automatically classifying queries into short or long queries using HeatWave data driven algorithms. Therefore, Q3 is prioritized before Q2 as Q3 is identified as a short-running query.

Auto Scheduling reduces elapsed time for short-running queries significantly when the multi-session applications consist of a mix of short and long running queries. It also ensures long-running queries are not penalized and are not postponed indefinitely.

Auto Error Recovery

HeatWave automatically provisions a new HeatWave node(s) when a hardware or software failure is detected on a node. When the cluster is restored, auto error recovery automatically reloads the data only to the re-provisioned node(s), allowing a very fast recovery.

Integration with Oracle Cloud Services

OCI offers a wide range of services for data analytics, machine learning, and data lake. Native integration with these services makes it easier for existing applications to use HeatWave (Figure 24).

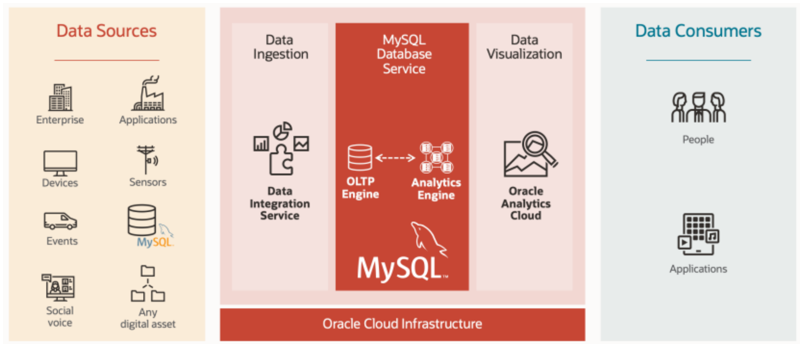


Figure 24. End to end integration with other Oracle Cloud Services from Data Ingestion to Data Visualization

Oracle Analytics Cloud (OAC) provides the industry's most comprehensive cloud analytics in a single unified platform, including self-service visualization and inline data preparation to enterprise reporting, advanced analytics, and self-learning analytics that deliver proactive insights. Integration with OAC provides BI visualization platform for users to analyze their MySQL data.

OCI Data Integration Service provides extract, transform and load (ETL) capabilities to target data warehousing scenarios on the OCI platform. It supports various data sources, starting with relational, cloud and Hadoop. Integration with OCI Data Integration allows users to easily transform and import data from data sources other than MySQL to HeatWave, expanding the scope of data that can take advantage of HeatWave.

1/2 the cost of AWS Redshift and Aurora one-year prepaid

The cost of using HeatWave depends upon the number of HeatWave nodes provisioned. The size of a Heatwave cluster depends on the size of the dataset, and the characteristics of the workload. A single HeatWave node can hold approximately 400GB of data. Customers can expect to see a significant reduction in their costs when they migrate to HeatWave. Compared to Amazon Aurora and Redshift, HeatWave is 1/2 the cost of Amazon reserved instance one-year pre-paid cost (Figure 1 & 5).

Customers are likely to find that the cost benefit with HeatWave is higher because of the following:

- Aurora charges additional for Storage IO cost which can be substantial. There is no such cost with HeatWave
- When using HeatWave, the cost includes both OLTP and OLAP capabilities. With Amazon Redshift the cost is only for OLAP, and additional costs are needed for OLTP.
- Customers need to pay as they move their data around from one database to another with Amazon Aurora or Amazon Redshift.

HeatWave offers pay as you go and Universal Credit Annual Flex pricing. Pay as you go allows customers to pay for only what they use, lowering the cost when they don't need to use the service 24/7. In both pricing models, organizations only need to pay for what they use.

Conclusion

HeatWave is a cloud native service which is exclusively available in Oracle cloud Infrastructure and provides compelling performance and cost for analytic workloads. Organizations using MySQL database for managing their enterprise data can now run analytic queries with HeatWave with significantly better performance, lower cost, not require ETL, and have support for real- time analytics. MySQL Autopilot provides machine learning automation that improves the performance, scalability, and ease of use of HeatWave. It automates the database lifecycle including provisioning, data loading, query processing, and error handling. The service can be deployed in a cloud only or in a hybrid environment, and it simplifies management for both transactional and analytic applications.

[1] How 6 Companies are Using Technology and Data to Transform Themselves. https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/how-six-companies-are-using-technology-and-data-to-transform-themselves. August 2020