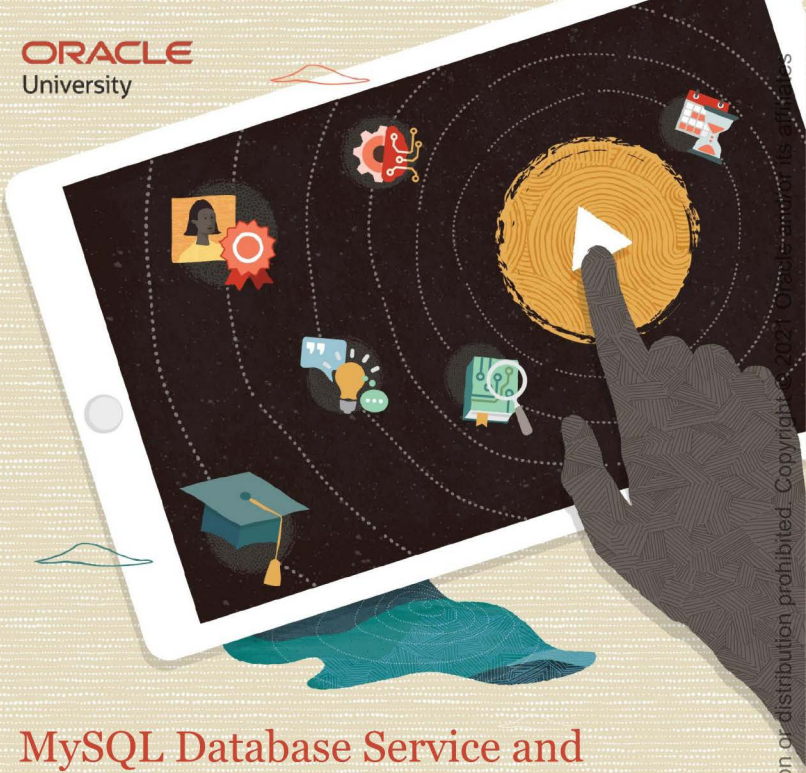
ORACLE
University



MySQL Database Service and Heatwave Tech Overview

Student Guide

S1102379GC10

Learn more from Oracle University at ***education.oracle.com***

Disclaimer

This document contains proprietary information and is protected by copyright and other intellectual property laws. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Restricted Rights Notice

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

Trademark Notice

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

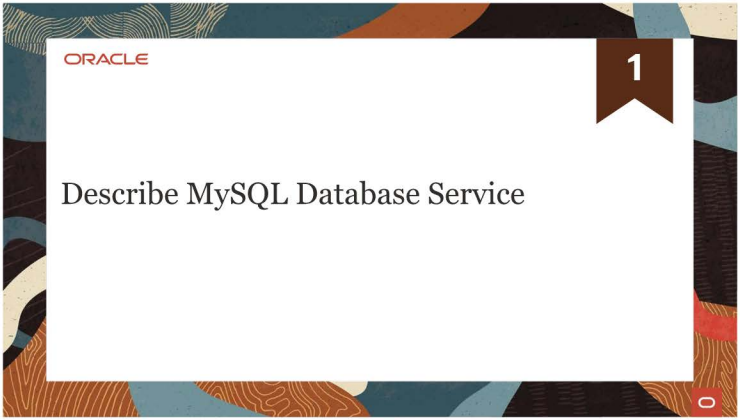
Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

Third-Party Content, Products, and Services Disclaimer

This documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

1010222021

Contents



Let's now talk about MySQL Database Service

Objectives

After completing this lesson, you should be able to learn about:

- MySQL Database background
- MySQL Database Service ease of use
- MySQL Database Service Security
- MYSQL Database Service Enterprise Ready

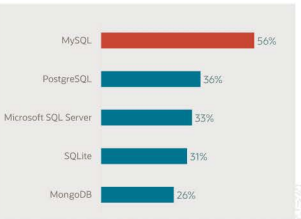
2

We'll start by setting the context and reviewing key trends around cloud and open source. Then we'll briefly talk about MySQL and MySQL customers in general before focusing on MySQL Database Service.

MySQL Is the #1 Open-Source Database

MySQL is the most popular database for developers.

Rank Jan 2021	Rank Dec 2020	Rank Jan 2020	DBMS	Database Model	S Jan 2021
1.	1.	1.	Oracle	Relational, Multi-model	1322.93
2.	2.	2.	MySQL	Relational, Multi-model	1252.06
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	1031.23
4.	4.	4.	PostgreSQL	Relational, Multi-model	552.23
5.	5.	5.	MongoDB	Document, Multi-model	457.22



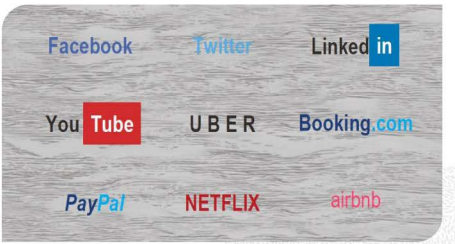
Stackoverflow survey 2020

MySQL is the #1 open-source database and the second-most popular DB overall, after the Oracle Database.

MySQL also received the DBMS of the year award from DB Engines.com.

According to a stackoverflow survey, MySQL has been for a long time and remains the #1 choice for developers, primarily because of its ease of use, reliability, and performance.

Innovative Enterprises Across Many Industries Run MySQL



MySQL is used by the world's most innovative companies. This includes Twitter, Facebook, Netflix, and Uber.

Many of these companies did not exist 20 years ago. Today they are disrupting multiple industries and are operating at a scale that is hard to imagine.

MySQL on Third-Party Clouds

Reality:

- MySQL Community or MySQL Fork
- Locks in data, cannot go anywhere else
- High cost
- No MySQL Enterprise Edition Advanced Features and Tools
- Not supported by MySQL Experts

Today MySQL can be found in Amazon RDS and Aurora, Google Cloud SQL, and Microsoft AZURE database for MYSQL.

They all offer a cloud-managed version of MySQL community edition with all of its limitations.

These MySQL Cloud services are expensive, and it is not easy to move data away from their cloud.

And most important of all, they are not supported by the Oracle MySQL experts and does not include the **MySQL Enterprise Edition Advanced Features and Tools.**

Introducing MySQL Database Service

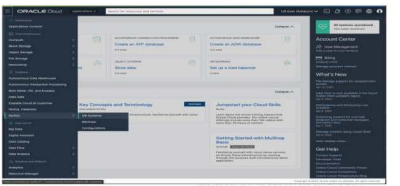
MySQL Database Service in Oracle Cloud Infrastructure is the only MySQL database service built on MySQL Enterprise Edition and 100 percent built, managed, and supported by the MySQL team.

Let's focus on the three major categories that makes MySQL Database Service better than the other MySQL cloud offerings:

1. Ease of use
2. Security, and
3. Enterprise readiness

MySQL Database Service: Ease of Use

- **Fully Managed Service**
  - Automate time-consuming tasks
  - Configuration, security patching, backup, and monitoring
- **Instant Provisioning**
  - Connect to production-ready, pre-configured MySQL databases
  - Provision fast, reliable, and secure cloud storage
  - Set up fast, predictable networking
- **Latest Features**
  - Fast-paced delivery of new features for modern applications
  - X Dev API, MySQL Shell, Document Store



8

DBAs tend to be overloaded with mundane database administration tasks.

They're responsible for many databases, their performance, security, availability and more.

It is really difficult for them to focus on innovation and on addressing the demands of lines of business.

MySQL Database Service automates all those time-consuming tasks so they can improve productivity and focus on higher value tasks.

Developers can quickly get all the latest features directly from the MySQL team to deliver new modern apps.

They don't get that in other clouds that rely on outdated or forked versions of MySQL.

Developers can use the MySQL Document Store to mix and match SQL and NoSQL content in the same database as well as the same application.

MySQL Database Service: Security First

- Built on Gen 2 Cloud Infrastructure - Security
- Built with security-first design principles
- User data stored on OCI Block Volumes resistant to failure
- Data is encrypted for privacy
- Gen2 provides maximum isolation and protection
  - Oracle cannot see customer data.
  - Users cannot access our cloud control computer.

Due to data location requirements or government concerns, some applications or data may need to stay on premises.

MySQL Database Service is 100 percent compatible with on-premises MySQL. This makes it easier to migrate applications to the cloud without vendor lock-in.

MySQL provides organizations the flexibility of a hybrid deployment model.

Now let's talk about **Gen 2 Cloud Infrastructure.**

Oracle Cloud is secure by design and architected very differently from the gen 1 clouds of our competitors. Gen2 provides maximum isolation and protection.

That means:

- Oracle cannot see customer data
- Users cannot access our cloud control computer

Gen2 architecture allows us to offer superior performance on our compute objects!

Finally, Oracle Cloud is open. Customers can run Oracle software, third-party options, open source … whatever you choose without modifications, trade-offs, or lock-ins.

MySQL Database Service: Security and Regulatory Compliance

- **Reduce Risk of Data Breaches**
  - Protect your data with encryption, masking, firewall, and more
- **Regulatory Compliance (GDPR, PCI, HIPPA)**
  - Advanced Security with MySQL Enterprise Edition
- **Latest Security Updates**
  - Latest MySQL security fixes from the MySQL team to limit exposure to security vulnerabilities

10

Data security has become a top priority for all organizations.

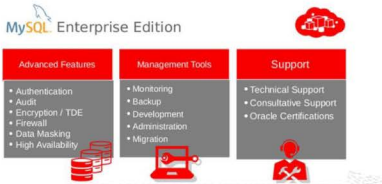
MySQL Database Service can help you protect your data against external attacks as well as internal malicious users with a range of advanced security features.

Those advanced security features can also help you meet industry and regulatory compliance requirements, including GDPR, PCI, and HIPPA.

When a security vulnerability is discovered, you'll get the fix directly from the MySQL team, from the team that actually develops MySQL.

MySQL Database Service: Enterprise Ready

- **Built on MySQL Enterprise Edition**
  - Highest reliability and security
  - 24x7 support from the MySQL Team
- **Integration with Oracle Technologies**
  - Oracle Data Integrator, Audit Vault, Container Engine for Kubernetes…
- **100% Compatible with On-Premises MySQL**
  - Easy path to cloud
  - Hybrid cloud deployments
  - No cloud fork lock-in



11

MySQL Database Service is the only public cloud service built on MySQL Enterprise Edition, which includes 24/7 support from the team that actually builds MySQL, at no additional cost.

All of the other cloud vendors are using the community edition of MySQL. So, they lack the Enterprise edition features and tools.

The following features are available and enabled by default in MySQL Database Service:

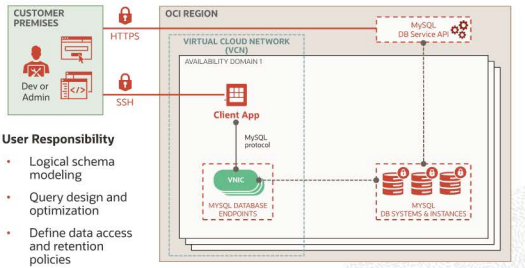
- MySQL Enterprise scalability, also known as the Thread Pool plugin
- Data at-rest encryption
- Native backup
- OCI built-in native monitoring

You can also install MySQL Enterprise Monitor to monitor MySQL Database Service remotely.

MySQL works well with your existing Oracle investment, like Oracle Data Integrator, Oracle Analytics Cloud, Oracle Goldengate, and more.

MySQL Database Service Customers can easily use Docker and Kubernetes for Dev-ops operations.

Managed MySQL: Focus on Your Business



User Responsibility

- Logical schema modeling
- Query design and optimization
- Define data access and retention policies

Oracle Responsibility

- Backup and recovery
- Database and OS patching
- Monitoring and log handling
- Security with advanced options available in MySQL Enterprise Edition

Requisites for Starting

- Tenancy to sign-in
- Compartment to store resources
- Group with granted policies

12

MySQL Database Service is “fully managed” database service.

A MySQL Database Service user is responsible for:

- Logical schema modeling
- Query design and optimization
- Define data access and retention policies

The MySQL team is responsible for providing automation for:

- OS installation
- Database and OS patching (including security patches)
- Backup and recovery (the system backs up the data for you, but in an emergency, you can restore to a new instance with a click)
- Monitoring and log handling
- Security with advanced options available in MySQL Enterprise Edition
- And, of course, maintaining the data center for you

The diagram illustrates:

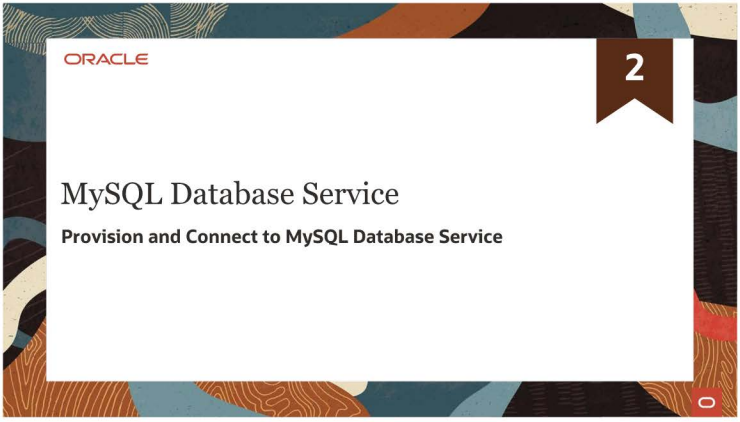
A user has a tenancy and manages policies to access the services and creates cloud resources there such as Compute instances. The MySQL team has internal tenancies that are not visible to the user. This is where the actual databases are placed. The user accesses the MySQL Database Service with a web console, CLI, or SDKs to interact with an internal Control Plane, which will deal with the MySQL instances life cycle operations, such as provisioning a new MySQL service instance or backing up databases.

To facilitate the management, MySQL database instances are grouped in DB Systems. When a user creates a DB System, an endpoint is exposed to his tenancy. Then the user can use the MySQL protocol to connect to this endpoint.

Summary

In this lesson, you should have learned about:

- MySQL Database background
- MySQL Database Service ease of use
- MySQL Database Service security
- MYSQL Database Service Enterprise ready



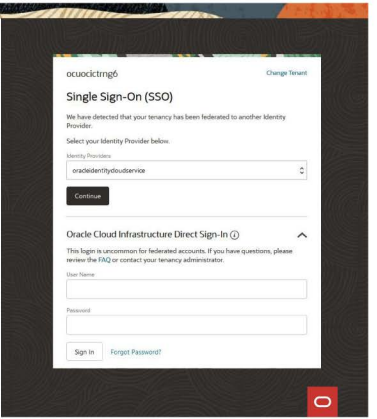
Objectives

After completing this lesson, you should be able to:

- Get started
- Create MDS System
- Connect to MDS System
- Manage MDS System

Access the MySQL Database Service from the Oracle Cloud Console

- **Create and manage MySQL databases** using the Console.
- **Only users granted with the necessary policies** can create and manage MySQL DB Systems.
- **Getting started details:**
  - Go to https://docs.oracle.com/en-us/iaas/mysql-database/doc/getting-started.html



3

Oracle Cloud Infrastructure is a set of complementary cloud services that makes it easy for you to build and run a wide range of applications and services in a highly available hosted environment. MySQL Database Service relies on several of those services.

You can create and access Oracle Cloud Infrastructure MySQL Database Service using the Console (our browser-based interface) or the REST API. In this presentation, we will focus on the Console method.

In order to create a MySQL Database Service, you need an Oracle Cloud account and an OCI cloud tenancy.

From the OCI tenancy, you should be able to sign in to the OCI console with a valid user account.

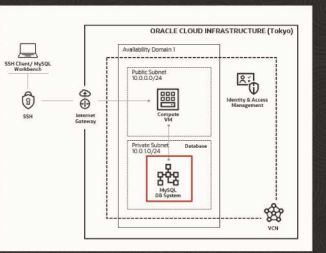
You need to have a compartment to store the database system resources.

Your user account must belong to a group that has been granted mandatory policies for MySQL Database Service.

Finally, you need to set up the Virtual Cloud Network (VCN) for your database system.

MDS System Architecture

- A Compute instance
- Oracle Linux Operating System
- MySQL Enterprise Edition – always upgraded to latest release
- Virtual Network Interface
- Network-attached block storage



A Database System is a logical container for the MySQL instance. It provides an interface enabling management of tasks such as provisioning, backup and restore, monitoring, and so on. It also provides a read/write endpoint, enabling you to connect to the MySQL instance using the standard protocols.

A MySQL Database Service DB System consists of the following components:

- A Compute instance
- An Oracle Linux Operating System
- The latest version of MySQL Server Enterprise Edition
- A Virtual Network Interface Card (VNIC) that attaches the DB System to a subnet of the Virtual Cloud Network (VCN)
- Network-attached higher performance block storage

The diagram on the right in the slide illustrates a simple MySQL Database Service System framework:

Let's start with the inside of the Oracle Cloud Infrastructure Region… This one is in Tokyo.

Our system is one data center; we call it an availability domain. We are using one Virtual cloud network, VCN for short.

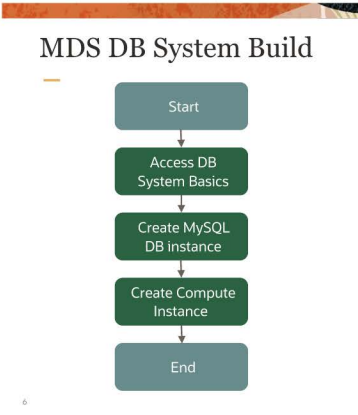
Like a traditional data center network, VCNs give us complete control over the network environment.

This VCN is divided into two subnets: a public and a private subnet.

The Oracle Linux computer is on the public subnet and the MySQL Database Service instance is on the private subnet.

The user accesses the MySQL Database Service with SSH and the Linux computer.

Let's get started…



After successfully logging in to your Oracle Cloud Infrastructure tenancy with the OCI console, you can start provisioning your MDS DB system.

You will need to perform the following three major tasks:

1. Access your DB System basics
2. Create your MySQL DB instance
3. Create your Compute Instance

Compartment: You or the OCI administrator should have created a compartment to store your resources. You can store everything in the root of the tenancy, but compartments are recommended good practice.

You can find the compartment under the OCI Identity / Compartment menu.

Policies: You or the OCI administrator should have created the MySQL Policies. These policies isolate who can interact with the database service. You, or your group, should have been granted the policies, so you can create and manage the MySQL Database Service system.

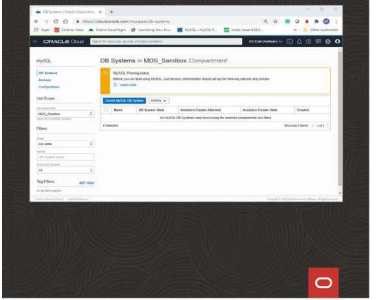
You can find the policies under the OCI Identity / Policies menu and typically on the root compartment.

VCN: You or the OCI administrator should have created the Virtual Cloud Network for the system. The VCN is the software-defined version of a traditional physical network. We talked about it earlier, and we will revisit it later on.

You can find the VCN under the OCI Networking / Virtual Cloud Network menu.

Create MySQL DB System

1. Console menu > MySQL > DB System
2. Create MySQL DB System
3. Provide basic information for the DB System
4. Set up your required DB System
5. Create administrator credentials
6. Configure networking
7. Configure placement
8. Configure hardware
9. Configure backups
10. Create



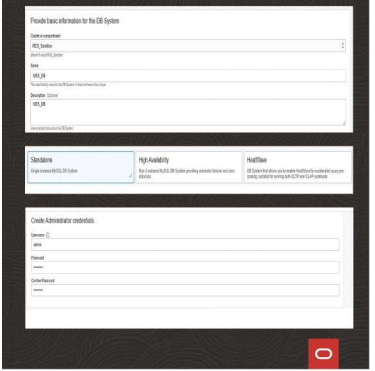
Step 1. Use the console menu to go to MySQL > DB System.

Then click the Create DB system button.

Create MySQL DB System

1. Console menu > MySQL > DB System
2. Create MySQL DB System
3. Provide basic information for the DB System
4. Set up your required DB System
5. Create administrator credentials

   "mysql.sys" is reserved username
6. Configure networking
7. Configure placement
8. Configure hardware
9. Configure backups
10. Create



On the Create MySQL DB System page, **in the Provide basic information for the DB System section:**

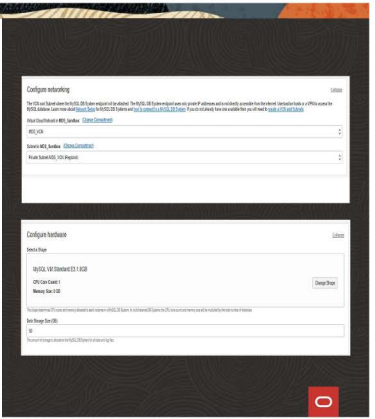
- You will enter some required information for the DB system.
- From the Compartment drop-down list, select a compartment. If you want to launch the MySQL DB System in a compartment other than the current compartment, select the required compartment from the list. If you do not select a different compartment, the current compartment is used.
- In the Name field, enter a user-friendly display name for the DB system. The name does not need to be unique. An Oracle Cloud Identifier (OCID) uniquely identifies the DB system.
- In the Description field, enter a user-friendly description of the DB System and its purpose.
- In the Set up your required DB System section, you will see several options. Select the **Standalone** option to specify the single-instance DB System that we want to create.

In Step 5, you will create administrator credentials by specifying the DB System administrator's credentials. You will enter the username of the administrator user. The administrator user is granted a specific set of privileges that differ from the root user. Do not use "mysql.sys" or other reserved username. Please check the OCI document for more details. Enter a valid administrator user's password.

Confirm the password by re-entering the administrator user's password.

Create MySQL DB System

1. Console menu > MySQL > DB System
2. Create MySQL DB System
3. Provide basic information for the DB System
4. Set up your required DB System
5. Create administrator credentials
6. Configure networking
7. Configure placement
8. Configure hardware
9. Configure backups
10. Create



11

To Configure Networking:

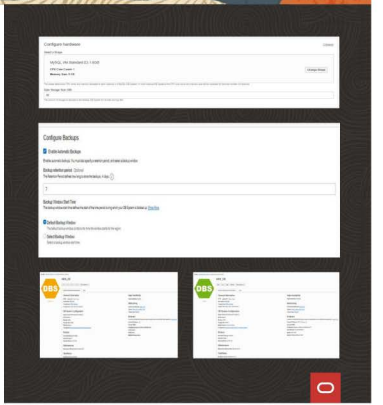
- Select the Virtual Cloud Network: which is the VCN in which to launch the DB System
- Select the required subnet: which is the private subnet of the selected VCN … Remember for security reasons, the MDS DB must use a private subnet.

In Configure placement, keep "Availability Domain" checked. This is the data center location of the DB System. Do not check "Choose a Fault Domain" for this DB System. Oracle will choose the best placement for you.

Create MySQL DB System

1. Console menu > MySQL > DB System
2. Create MySQL DB System
3. Provide basic information for the DB System
4. Set up your required DB System
5. Create administrator credentials
6. Configure networking
7. Configure placement
8. Configure hardware
9. Configure backups
10. **Create**

12



In **Configure Hardware,** you can select the shape to use for your DB system. The shape determines the resources that will be allocated to the system. Let's keep the default shape for now.

Data Storage Size is the amount of block storage, in GB, to allocate to the DB System. This block storage stores all data, logs, and temp files. Let's keep the Data Storage default value.

In **Configure Backups:**

Select to enable scheduled backups. If you do not specify scheduled backups, you must manage your backup strategy manually. It is strongly recommended to enable scheduled backups.

In **Retention Period:** Define how long to retain the backups, in days. Keep the default of seven days.

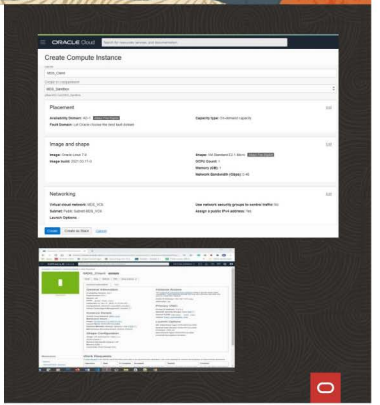
A backup window is defined as a time during which the backup is initiated. Select default Backup Window.

Click Create button.

The state will be shown as Creating during the creation, and it will be a yellow DBS sign. After a few minutes, the green active DBS sign will show up … This means the DB system is ready for use.

Create Compute Instance

1. Go to Console, menu, Compute, Instances
2. Choose compartment
3. Choose the operating system
4. Configure placement
5. Choose same VCN as MySQL DB System
6. Select Assign public IP address
7. Add SSH key
8. Click “Create Instance”



You will need a client machine to connect to your brand-new MySQL database.

To launch a Linux Compute instance, go to the Console, menu Compute, Instances, and click Create Instance.

Enter the instance name and make sure the correct compartment is selected.

Keep the Oracle Linux default for the operating system image.

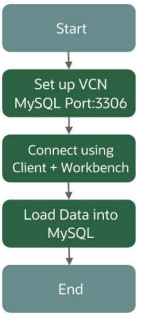
For Configure placement and hardware, keep the default availability domain and instance shape.

For VCN, make sure you select the same VCN as MySQL DB System and the "Assign a public IP address" should be set to Yes.

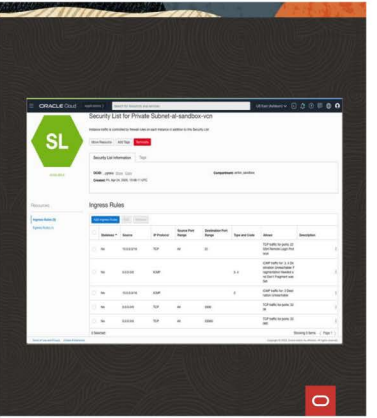
Generate an SSH key or add your existing SSH Key to the Add SSH Key section.

Click “Create” button, and the new client machine will be ready to use after a few minutes. The state Running, green box, indicates that the Virtual Machine is ready to use.

MDS Connect and Load



14



To connect to the new OCI MySQL database, you need to perform the following steps:

- Step 1: Set up your VCN MySQL Port: 3306 access.
- Step 2: Connect to MDS using MySQL Client.
- Step 3: Or you can connect using MySQL Workbench.
- Step 4: Load data into the MySQL database.

Let's do Step 1: To configure a network to enable communication between VPN or Compute and DB System, you must configure your VCN's subnets with security rules.

These rules permit traffic from specific IP addresses and ports between resources.

Use the console menu to go to Networking> Virtual cloud Networks.

Then click the VCN that you are using for the MySQL Database Service Instance. This will open the VCN's Details page

Select Security Lists from the Resources section.

Click **Add Ingress Rules**. The **Add Ingress Rules** dialog is displayed.

Add the following information to the Ingress Rule:

- **Stateless:** Do not select.
- **Source Type:** CIDR
- **Source CIDR:** The CIDR of the public subnet. You can narrow the range down to more specific IP addresses if it is required.
- **IP Protocol:** TCP
- **Source Port Range:** Leave blank
- **Destination Port Range:** The port the DB System will listen on. Default is 3306 for MySQL Classic and 33060 for MySQL X Protocol.

SSH Compute and MySQL Shell

- Compute instance
- Connect SSH with Endpoint (Public IP Address)
- Install MySQL Shell
- Connect with mysql

$ ssh opc@[your compute IP] -I ~/.ssh_id_oci_demo

$ sudo yum install https://dev.mysql.com/get/mysql80-community-release-el7-3.noarch.rpm

16

From a terminal window on your local system, connect to the Compute instance with the SSH command. You will need a MySQL client tool to connect to your new MySQL DB System from your client machine. So, install the latest Install MySQL release package and install MySQL Shell on the on-client instance.

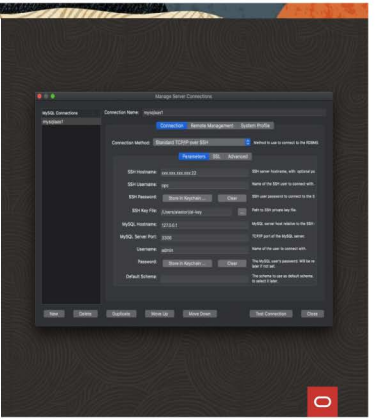
From your compute instance, connect to MySQL using the MySQL Shell client tool.

The endpoint (IP Address) can be found on the MySQL DB System Details page, under the "Endpoints" resource. Take a look at the example in the slide.

From there, you can run SQL commands or use other languages such as JavaScript and Python

SSH and Workbench

- Use MySQL Workbench from the local machine to connect to MySQL.
- Compute Public IP as jump box.
- In MySQL Workbench, configure a connection using "Standard TCP/IP over SSH."
- Compute instance credentials for SSH.
- https://lefred.be/content/using-mysql-workbench-with-mysql-database-service/



17

At this point, you can also use MySQL Workbench from your local machine to connect to the MySQL endpoint using your new Compute instance as a jump box. In MySQL Workbench, configure a connection using the method "Standard TCP/IP over SSH" and use the credentials of the Compute instance for SSH.

You need to enter the following information to make it work:

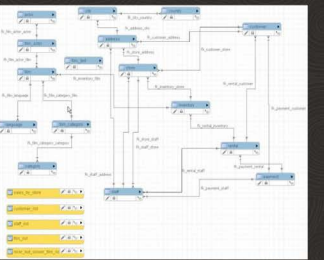
- **SSH Hostname:** The public IP address of the Client instance
- **SSH Username:** opc
- **SSH Keyfile:** Path to your SSH private key
- **MySQL Hostname:** The IP address of the MySQL Endpoint
- **MySQL Server Port:** The port the MySQL Endpoint is listening on
- **Username/Password:** Credentials you defined when creating the DB System

Click **Test Connection** to confirm the connection details are valid.

Loading Data into MySQL

- Download and unpack the Sakila sample dataset.
- Import the `Sakila` schema and data using MySQL Shell.
- Use `Sakila`.

https://docs.cloud.oracle.com/en-us/iaas/mysql-database



The Sakila Schema

18

To test-drive your Oracle Infrastructure MySQL Database Service system, you can download and import `Sakila`.

The `Sakila` sample database can be used to test different features of MySQL such as Views, Stored Procedures, and Triggers.

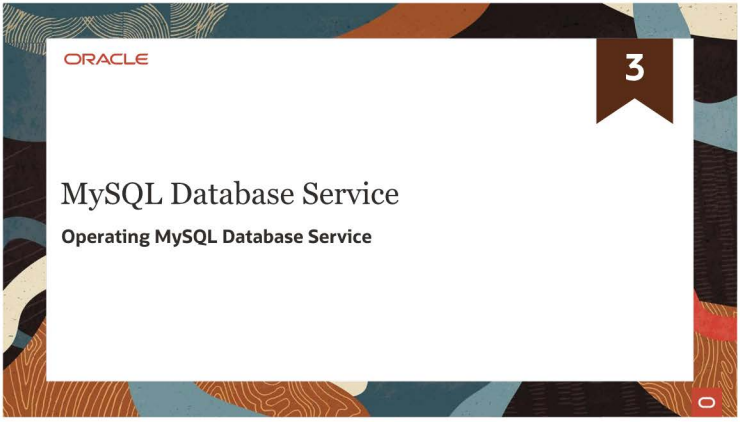
To learn more about OCI MDS system, check out the Oracle Cloud Infrastructure MySQL Database Service Documentation site.

Summary

In this lesson, you should have learned how to:

- Get started
- Create MDS System
- Connect to MDS System
- Manage MDS System

19



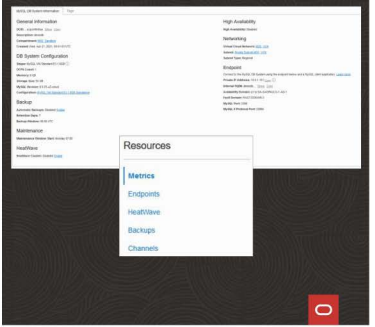
Objectives

After completing this lesson, you should be able to:

- Manage MySQL DB System
- Maintain MySQL DB System
- Monitor MySQL DB System
- Back up MySQL DB System
- Manage performance of MySQL DB System

Manage MySQL Data Service

- Details
  - CLI - DB System OCID
  - Information
  - Resource
- Status
  - Start
  - Stop
  - Reboot
- Edit
- Delete



The MySQL DB System Details page lets you view and manage your Database System.

You can retrieve DB System details using the OCI Command Language Interface by providing the DB System OCID to the OCI get command.

In this module, we will focus on using the OCI console to work with the DB System details page. This page is divided into two major sections: the Information section and the Resource section.

The DB System Information section shows the database system creation date, the number of OCPUs allocated to the system, the private IP address of the DB System, the maintenance window start time, and the automatic backup window start time if backup is enabled.

The resource section contains information about the fully qualified domain name of the DB System and the system status.

It shows the HeatWave details if HeatWave clusters have been installed and shows MySQL database replication channel information. We will discuss the Metrics and Backups details later on.

Manage MySQL Data Service

- Details
  - Information
  - Resource
- Status
  - Start
  - Stop – OCPU billing stops
  - Restart – OCPU billing resumes
- Edit
- Delete

You can see the status of the MySQL DB System is from the detail page. The status is a color-coded icon with associated text.

"CREATING" and a yellow icon means resources are being reserved for the DB system, the system is booting, and the initial database is being created. Provisioning can take several minutes. The system is not ready to use yet.

"ACTIVE" and a green icon means the DB system was successfully created and can now be used.

"UPDATING" and yellow icon means the DB system is in the process of starting, stopping, or restarting. This status is also used if a replication channel associated with the DB System is updating.

"INACTIVE" and a gray icon means the DB system is powered off by the stop or reboot action in the Console or API.

"DELETING" and a yellow icon means the DB system is being deleted by the terminate action in the Console or API.

"DELETED" and a gray icon means the DB system has been deleted and is no longer available.

"FAILED" and a red icon means an error condition prevented the creation or continued operation of the DB system.

At the top of the detail page, there are buttons to start, stop, or restart the DB system.

Start: Starts a stopped DB System. After the DB System is started, the Stop action is enabled and the Start option is disabled.

Stop: Stops a running DB System. After the DB System is powered off, the Start action is enabled.

Restart: Shuts down a DB System and restarts it.

Remember stopping a DB System stops billing for all OCPUs associated with it.

- Billing continues for storage.
- Billing for OCPUs resumes if you restart the DB System.

At the top of the detail page, there is also a button to edit the DB system.

You can edit the DB system name, description, shape, configuration, and Maintenance Window Start Time.

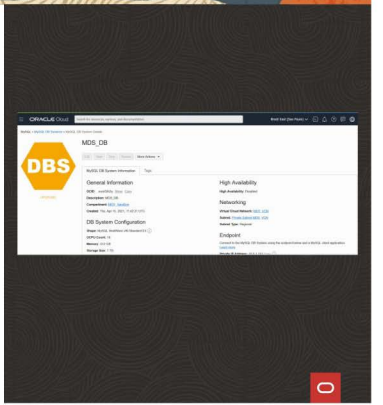
Finally, you can delete the DB system from the detail page by clicking the "More Actions" button and selecting the Delete option.

A dialog box will open, and it will ask you to confirm the deletion. Enter the word, all caps, "DELETE" and click the "Delete MySQL DB System" button.

When the delete process is done, **MDS_DB** will be set to Delete status.

Maintain MySQL Data Service

- Automatic process
- Patches underlying operating system
- Patches MySQL server
- Patches underlying hardware
  - DB System's status changes to **UPDATING**.



The maintenance of MySQL DB Systems is an automatic process.

It patches off the underlying operating system, the MySQL server itself, and any underlying hardware.

These tasks are performed during the maintenance window defined on the DB System detail page.

If you did not define a maintenance window, one was defined for you automatically.

When maintenance is performed, your DB System's status changes to **UPDATING**, and the DB System becomes unavailable for a short time while the maintenance completes.

The MySQL DB Systems maintenance is performed infrequently, and only when absolutely necessary. This is usually for security or reliability issues.

Monitor MySQL Data Service

- Metrics
  - Connection
  - Activity
  - Latency
  - Utilization
- Alarms
- Notifications

You can monitor the health, capacity, and performance of your Oracle Cloud Infrastructure MySQL Database Service resources by using metrics, alarms, and notifications.

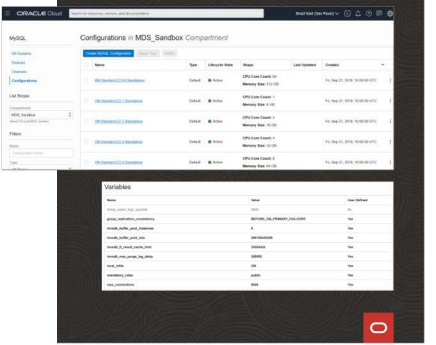
The MySQL Database Service metrics enable you to measure useful quantitative data about your MySQL Databases, such as current connection information, statement activity and latency, host CPU, memory and disk IO utilization, and so on. You can use metrics data to diagnose and troubleshoot problems with MySQL Databases.

To view the view metric charts for a DB System, go to its detail page under Resources and click Metrics. The Metrics page displays a set of charts for the current MySQL Database System.

OCI Monitoring service allows you to create alarms for keeping an eye on your MDS system behaviors that are interesting and significant for you. The alarms allow you to focus time and energy on other important tasks instead of regularly monitoring the resource page metrics and charts. Once you set the alarms, it uses the OCI Notifications service to notify you about the selected events. Notifications service uses topics and subscriptions to send out notifications. Messages are published to topics and then sent out to Email, PagerDuty, Slack, HTTPS URLs, or Oracle Functions.

MySQL Data Service Performance

- MySQL Configurations for DB Systems
  - my.ini and my.cnf
  - MySQL
  - Create, copy, edit
  - MySQL Server System Variables
- Supported MySQL Shapes
  - OCPU
  - Memory



Besides performance monitoring, Oracle Cloud Infrastructure provides other tools to help manage the performance of your MySQL DB System. They are known as

- MySQL Configurations for DB Systems
- Supported MySQL Shapes

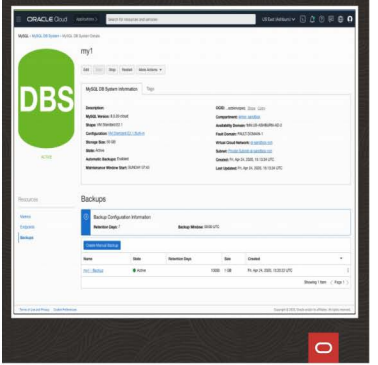
MySQL Configurations for DB System are collections of MySQL variables that define the operation of the MySQL DB System. They are analogous to the **my.ini** or "**my.cnf**" files used in local installations of MySQL Server. Their default configurations are already built in OCI. You can create your customized configurations. The MySQL Configurations for DB System are linked to supported OCI shapes and are their associated configuration details.. An OCI shape is a template that determines the number of OCPUs, amount of memory, and other resources that are allocated to the DB System instance.

From the navigation menu, Databases, and MySQL link, you can create a custom configuration by providing the compartment, name, shape, and configuration variable. The configuration variable is a MySQL Server System Variable. You can learn more about these variables from the MySQL manual.

You can also edit and copy a custom MySQL Configuration. It is not possible to edit an OCI Default MySQL configuration.

MySQL Data Service Backup

- Backup
  - FULL - all DB data
  - INCREMENTAL- added DB data
  - Auto / Manual
  - Policies
    - _mysql-backups
    - _ mysql-instances
    - _ mysql-work-requests



By backing up your MySQL Database Service System, you can protect the database against loss if a failure occurs. By restoring from a backup, you can restore that database to its state at the time of the backup. Oracle Cloud infrastructure provides MySQL Database Service the following backup types:

- FULL: a backup of all data contained in the DB System
- INCREMENTAL: a backup of only the data that has been added or changed since the last FULL backup

Backups are run in either of the following ways:

- Manual - A backup initiated by an action in the console or request made through the API. Manual backups can be retained for a minimum of 1 day and a maximum of 365 days. Currently, there is a limit of 100 manual backups per tenancy. If you no longer want to manually back up your MySQL Database Service system, simply update the MySQL Database Service system and enable Automatic Backups.
- Automatic - Scheduled backups that run, without any required interaction, at a time of the user's choosing. Automatic backups are retained for between 1 and 35 days. The default retention value is 7 days. Once defined, it is not possible to edit the retention period of an automatic backup.

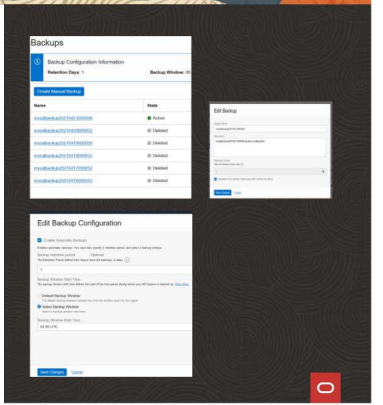
To manually back up a MySQL DB System using the OCI Console or Command Line Interface, your user should have been granted the following policies:

- _mysql-backups
- _ mysql-instances
- _ mysql-work-requests

MySQL Data Service Backup

- Update Backup Details:
  - Edit details
  - Change backup time
- Delete Backup:
  - Remove saved backups
  - Deleted DB backups retained for retention period

10



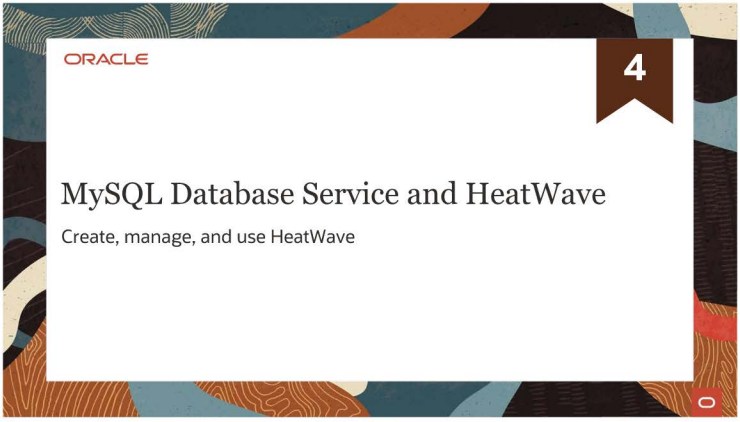
You can update a MySQL Database Service backup's display name, description, and retention period from the Edit Backup page.

Suppose your MySQL Database Service system automatically runs backup between 12 AM and 1 AM and you want to change the starting time to 4 AM, simply go to the MySQL DB System Details page and click the Edit Backup Configuration button. From that page, select Backup Window and set the start time to 4 AM.

Summary

In this lesson, you should have been able to:

- Manage MySQL DB System
- Maintain MySQL DB System
- Monitor MySQL DB System
- Back up MySQL DB System
- Manage performance of MySQL DB System



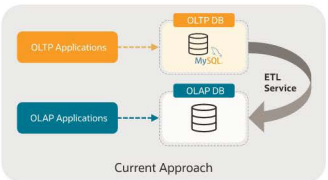
Objectives

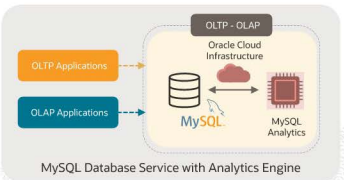
After completing this lesson, you should be able to:

- Understand Heatwave overview
- Discuss the prerequisites
- Add a HeatWave cluster
- Load data into the HeatWave clusters
- Run queries
- Manage HeatWave clusters

Overview - Description

- Exclusively available in Oracle Cloud Infrastructure
- In-memory, query-processing engine designed for fast execution of analytic queries
- Uses machine learning to automate operations, increasing DBA productivity and reducing costs



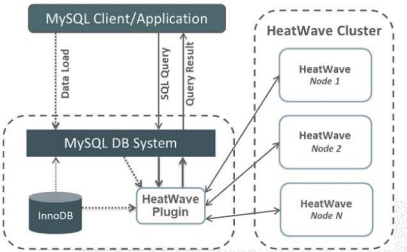


MySQL is the most popular open-source database, and many organizations choose MySQL to store their valuable enterprise data. MySQL is optimized for Online Transaction Processing, OLTP, but it is not designed for online analytic processing OLAP. As a result, organizations that need to efficiently run analytics on data stored in MySQL database move their data to another database to run analytic applications.

MySQL Heatwave is designed to enable customers to run analytics on data that is stored in MySQL database without moving data to another database. It is built on an innovative in-memory analytics engine that is architected for scalability and performance and is optimized for Oracle Cloud Infrastructure (OCI). Various aspects of this service leverage machine learning–driven automation, reducing database administrative costs.

Overview - HeatWave Architecture

- Innovative in-memory columnar analytics engine
- Optimized for Oracle Cloud Infrastructure
- Automation of various capabilities



HeatWave is a distributed, scalable, shared-nothing, in-memory, columnar, query processing engine designed for fast execution of analytic queries. It is enabled when you add a HeatWave cluster to a MySQL DB System.

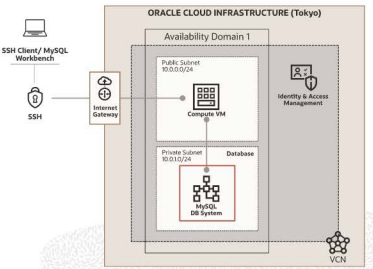
A HeatWave cluster comprises a MySQL DB System node and two or more HeatWave nodes. The MySQL DB System node includes a plugin that is responsible for cluster management, loading data into the HeatWave cluster, query scheduling, and returning query results to the MySQL DB System. The HeatWave nodes store data in memory and process analytics queries. Each HeatWave node contains an instance of the HeatWave.

The number of HeatWave nodes required depends on the size of your data and the amount of compression that is achieved when loading the data into the HeatWave cluster. A HeatWave cluster supports up to 24 nodes.

HeatWave Prerequisites

Before using HeatWave must-haves:

- An operational MySQL DB System created using a BM.Standard.E2.64 or MySQL.HeatWave.VM.Standard.E3 shape
- A running Compute instance attached to a public subnet on the same VCN as the MySQL DB System, installed with MySQL Shell 8.0.22 or higher
- Granted mysql-analytics policies



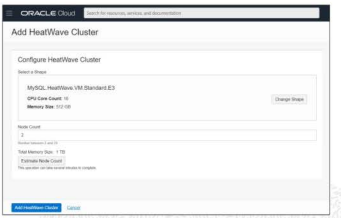
Before using HeatWave, ensure the following are present:

- A working MySQL DB System created using a BM.Standard.E2.64 or MySQL.HeatWave.VM.Standard.E3 shape
- A running Compute instance attached to a public subnet on the same VCN as the MySQL DB System
- MySQL Shell 8.0.22 or higher installed on the Compute instance

In addition to the mandatory policies for the MySQL DB System, you, or your group, must have been granted the **mysql-analytics** policies.

Adding HeatWave Cluster to DB System

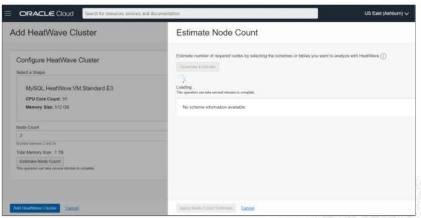
- **Shape Details:**
  Number of OCPUs and RAM
- **Node Count:**
  Number of HeatWave nodes to create



To add a HeatWave cluster to an existing MySQL DB System:

1. Open the navigation menu. Under MySQL, click DB Systems.
2. Add your HeatWave cluster in one of the following ways:
   a) Select DB System and choose Add HeatWave Cluster.
   b) Open DB System and select HeatWave Cluster.
   c) Select Add HeatWave Cluster.
      - Shape Details: Number of OCPUs, RAM, and so on
      - Node Count: Number of HeatWave nodes to create
   d) Click Add HeatWave Cluster to create the HeatWave cluster.

Adding HeatWave Cluster - Estimate Node Count

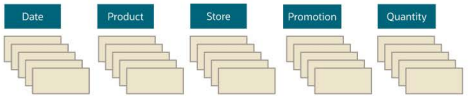


*** Data intended to load into Heatwave cluster must be present on the MySQL DB System.

To generate a node count estimate:

1. Ensure the data intended to load into the HeatWave cluster must be present on the MySQL Database System.
2. Click Estimate Node Count when the Estimate Node Count dialog box is displayed. Click Generate Estimate to create a new estimate.
3. When the operation completes, a table displays information about the schemas that were evaluated.
4. Select the schemas to include in the node count estimate.
5. Review the estimate details in the Summary dialog box.
6. Click Apply Node Count Estimate.

Loading Data Into the HeatWave Clusters



Query the "LOAD_STATUS" data from the HeatWave Performance Schema tables to verify tables are loaded in the HeatWave cluster.

***** Data is distributed among analytics nodes by slicing tables horizontally.**

```
MySQL>SQL> USE performance_schema;
MySQL>SQL> SELECT NAME, LOAD_STATUS FROM rpd_tables,rpd_table_id
           WHERE rpd_tables.ID = rpd_table_id.ID;
+-------------------------------+-------------------+
| NAME                          | LOAD_STATUS       |
+-------------------------------+-------------------+
| tpch.supplier                 | AVAIL_RPDGSTABSTATE |
| tpch.partsupp                 | AVAIL_RPDGSTABSTATE |
| tpch.orders                   | AVAIL_RPDGSTABSTATE |
| tpch.lineitem                 | AVAIL_RPDGSTABSTATE |
| tpch.customer                 | AVAIL_RPDGSTABSTATE |
| tpch.nation                   | AVAIL_RPDGSTABSTATE |
| tpch.region                   | AVAIL_RPDGSTABSTATE |
| tpch.part                     | AVAIL_RPDGSTABSTATE |
+-------------------------------+-------------------+
```

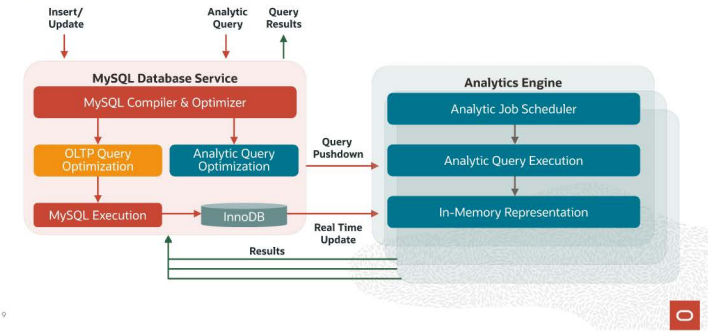
8

Loading data into the HeatWave cluster typically involves identifying the tables you want to load, excluding columns that are not required or supported, encoding string columns, defining **RAPID** as the secondary engine, and executing table load operations.

When loading a table into the HeatWave cluster, data is read from **InnoDB** using batched, multithreaded reads. Data is then converted into columnar format and sent over the network and distributed among the HeatWave nodes. Data is distributed among HeatWave nodes by slicing tables horizontally. Data is partitioned by primary key unless data placement keys are defined.

To verify that the tables are loaded in the HeatWave cluster, query **`LOAD_STATUS`** data from the HeatWave Performance Schema tables. Loaded tables have an **`AVAIL_RPDGSTABSTATE`** load status.

Running Queries in HeatWave



Users and applications interact with HeatWave through the MySQL database node in the cluster. Users connect to HeatWave through standard tools and standard-based ODBC/JDBC connectors.

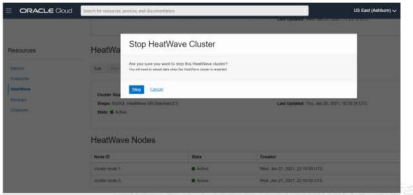
Once users submit a query to the MySQL database, the MySQL query optimizer transparently decides if the query should be offloaded to HeatWave cluster for accelerated execution. This is based on whether all operators and functions referenced in the query are supported by HeatWave and if the estimated 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.

Data of HeatWave is persisted in MySQL InnoDB. Any updates to the tables 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 the figure in the slide. This is done behind the scenes by a lightweight change propagation algorithm that can keep up with MySQL data update rates.

HeatWave Cluster Start and Stop

HeatWave Start and Stop Actions

- **Start:** Starts a stopped HeatWave cluster. Must reload data to the Heatwave cluster.
- **Stop:** Stops a running HeatWave cluster. Billing will stop for the Heatwave cluster.
- **Restart:** Shuts down HeatWave cluster. Must reload data to the Heatwave cluster.



10

Starting and Stopping a HeatWave Cluster

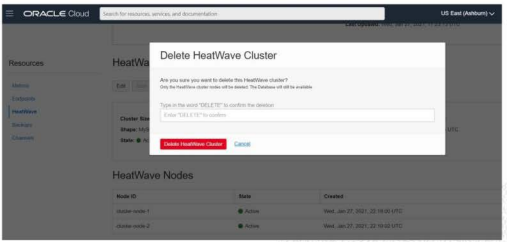
- Open the navigation menu. Under MySQL, click DB Systems.
- Choose your DB System.
- In the Resources list, select HeatWave.

Select one of the following actions:

- **Start:** Starts a stopped HeatWave cluster. After the HeatWave cluster is started, the Stop action is enabled and the Start option is disabled.
- **Stop:** Stops a running HeatWave cluster. After the HeatWave cluster is stopped, the Start action is enabled.
- **Restart:** Shuts down a HeatWave cluster and restarts it. You must reload data to the cluster.

Stopping the HeatWave cluster stops billing for the cluster. Billing resumes if you restart the HeatWave cluster.

HeatWave Cluster Delete



*** Delete the DB system and it will also delete the attached Heatwave Cluster.

11

To delete a HeatWave cluster:

1. Open the navigation menu. Under MySQL, click DB Systems.
2. A list of DB Systems is displayed.
3. In the HeatWave filter, select Attached to filter the DB Systems by those with a HeatWave cluster attached.
4. Open your DB System and select HeatWave from the Resources list.
5. To delete the HeatWave cluster, click Delete and follow the instructions.

Deleting the HeatWave cluster does not delete the DB System or any of the data. However, deleting the DB System also deletes the attached HeatWave cluster.

MySQL + HeatWave

The only MySQL service with a massively scalable, integrated analytics engine:

- Single MySQL database for OLTP and analytics applications
- All existing applications work without any changes
- Enables running analytics on data stored on premises
- Extreme performance: Accelerates MySQL by orders of magnitude, scales to thousands of cores
- Dramatically faster and lower cost compared to other cloud services

12

Organizations using MySQL database for managing their enterprise data can now run analytic queries with HeatWave with significantly better performance, lower cost, not requiring Extract Translate Load (ETL) and support for real-time analytics. The service can be deployed in a cloud only or in a hybrid environment, and it simplifies management for both transactional and analytic applications.

Get started today on a better solution than self-managed, on premises, or in AWS.

Summary

In this lesson, you should have been able to:

- Understand Heatwave overview
- Discuss the prerequisites
- Add a HeatWave cluster
- Load data into the HeatWave clusters
- Run queries
- Manage HeatWave clusters

13