HeatWave on AWS Service Guide

F82097-20

HeatWave on AWS Service Guide ,

F82097-20

Contents

6 Importing Data

7 Inbound Replication

8 Manage Data in HeatWave with Workspaces

Preface and Legal Notices

This is the *HeatWave on AWS Service Guide*.

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.

This documentation is NOT distributed under a GPL license. Use of this documentation is subject to the following terms:

You may create a printed copy of this documentation solely for your own personal use. Conversion to other formats is allowed as long as the actual content is not altered or edited in any way. You shall not publish or distribute this documentation in any form or on any media, except if you distribute the documentation in a manner similar to how Oracle disseminates it (that is, electronically for download on a Web site with the software) or on a CD-ROM or similar medium, provided however that the documentation is disseminated together with the software on the same medium. Any other use, such as any dissemination of printed copies or use of this documentation, in whole or in part, in another publication, requires the prior written consent from an authorized representative of Oracle. Oracle and/or its affiliates reserve any and all rights to this documentation not expressly granted above.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at https://www.oracle.com/corporate/accessibility/.

Access to Oracle Support for Accessibility

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit https://www.oracle.com/corporate/accessibility/learning-support.html#support-tab.

1 Overview

HeatWave on AWS is a fully managed service, developed and supported by Oracle. Oracle automates tasks such as database and operating system patching. You are responsible for managing your data, schema designs, and access.

- HeatWave
- MySQL Database
- Integration with Oracle Cloud Infrastructure (OCI)
- Region Availability
- Identity and Access Management
- Security

1.1 HeatWave

HeatWave is a massively parallel, high performance, in-memory query accelerator that accelerates MySQL performance by orders of magnitude for analytics and mixed workloads.

With HeatWave, you can provision DB Systems and HeatWave Clusters. HeatWave consists of a MySQL DB System and one or more HeatWave nodes. The MySQL DB System node is responsible for cluster management, query scheduling, and returning query results. HeatWave nodes store data in memory and process analytics queries.

When a HeatWave Cluster is enabled, queries that meet certain prerequisites are automatically offloaded from the MySQL DB System to the HeatWave Cluster for accelerated processing. Queries are issued from Query Editor in the HeatWave Console or from a MySQL client or application that interacts with the HeatWave Cluster by connecting to the MySQL DB System.

> **Tip:**
>
> This *HeatWave on AWS Service Guide* covers what you need to know to set up and use DB Systems and HeatWave Clusters on AWS, mainly using the HeatWave Console. For more detailed information about using HeatWave and related utilities, refer to the HeatWave User Guide . Some of that information does not apply to HeatWave on AWS, so be sure to check the *HeatWave on AWS Service Guide* first for the most relevant information.

1.2 MySQL Database

The MySQL Database on HeatWave on AWS is built on the MySQL Enterprise Edition Server, which allows developers to quickly create and deploy secure cloud native applications using the world's most popular open source database. It is the only MySQL database that supports HeatWave. For MySQL Server documentation, refer to the MySQL Reference Manual .

HeatWave on AWS supports MySQL Enterprise Edition 8.4. You can choose from the latest MySQL 8.4 releases when creating a DB System. See Creating a DB System.

1.3 Integration with Oracle Cloud Infrastructure (OCI)

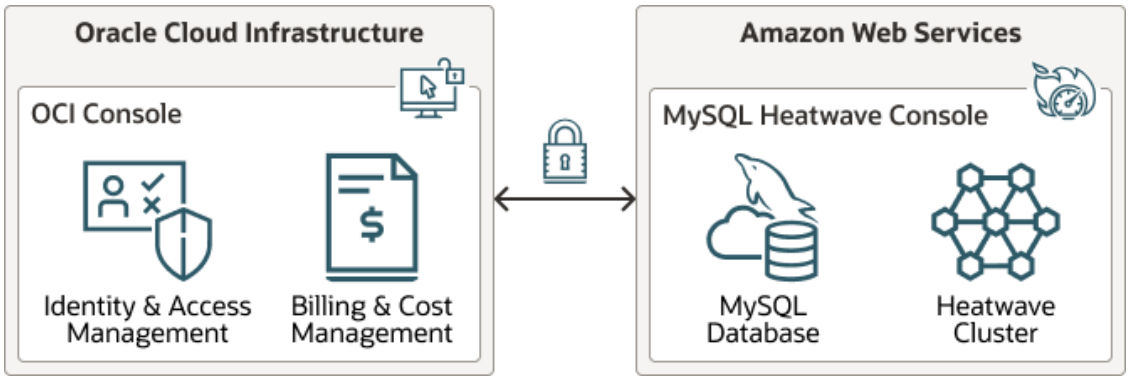
HeatWave on AWS is integrated with Oracle Cloud Infrastructure (OCI) for identity and access management, and for billing and cost management.

When you sign up for HeatWave on AWS, you are directed to OCI where you must sign up for an Oracle Cloud Account if you do not have one. After signing up, you are directed to the OCI Console to complete the sign-up process. When signing in to the HeatWave Console, you are seamlessly directed to OCI for authentication and then back to the HeatWave Console.

Identity and access management and billing for HeatWave on AWS is managed in OCI. For more information, see Billing.

The following diagram illustrates HeatWave on AWS integration with Oracle Cloud Infrastructure (OCI).

Figure 1-1 HeatWave on AWS and OCI Integration



1.4 Region Availability

HeatWave on AWS is currently available in five regions. Each AWS Region maps to an OCI region, and requires a subscription to that OCI region.

Table 1-1 OCI and AWS Regions

OCI Region name	OCI Region identifier	AWS Region name	AWS Region code
Germany Central (Frankfurt)	eu-frankfurt-1	Europe (Frankfurt)	eu-central-1
India West (Mumbai)	ap-mumbai-1	Asia Pacific (Mumbai)	ap-south-1
Japan East (Tokyo)	ap-tokyo-1	Asia Pacific (Tokyo)	ap-northeast-1
UK South (London)	uk-london-1	Europe (London)	eu-west-2
US East (Ashburn)	us-ashburn-1	US East (N. Virginia)	us-east-1

See Subscribed Region Limits.

1.5 Identity and Access Management

HeatWave on AWS integrates with Oracle Cloud Infrastructure (OCI) for identity and access management.

HeatWave on AWS uses predefined OCI Identity and Access Management (IAM) groups to control access to the HeatWave Console. For example, predefined group membership determines who can create DB Systems and HeatWave Clusters and who can use them. An Administrator in your organization is responsible for adding users to the appropriate groups. For more information, see User and Group Management.

HeatWave on AWS also supports federation with third-party Identity Providers (IdPs). See Federating with Identity Providers.

If you are a regular user (not an Administrator) who needs access to HeatWave on AWS, contact your Administrator to set up an account. An OCI IAM user account is required to access the HeatWave Console, and that user must be added to one of the predefined OCI IAM groups. For more information, see User and Group Management.

A separate MySQL user account is required to access HeatWave on AWS from a MySQL client or application. This can be the MySQL Administrator user specified when creating the DB System (see Creating a DB System) or a MySQL user account created on the DB System using `CREATE USER` . If you are not the MySQL Administrator and you do not have a MySQL user account, have your MySQL Administrator create one for you.

1.6 Security

Oracle considers cloud security its highest priority. The following security features help keep your data safe and secure.

- Database access control and account management

  MySQL provides security features to control access and manage your account. See Access Control and Account Management.

- Connection Control

  HeatWave on AWS supports a plugin library that enables Administrators to introduce an increasing delay in server response to connection attempts after a configurable number of consecutive failed attempts. This capability provides a deterrent that slows down brute force attacks against MySQL user accounts. The default connection control settings, which cannot be modified, are as follows:

  - `connection_control_failed_connections_threshold`
  - `connection_control_max_connection_delay`
  - `connection_control_min_connection_delay`

  See The Connection-Control Plugins.

- Encryption at rest

  Your data is always encrypted at rest. HeatWave on AWS uses Amazon EBS encryption. Boot volumes are encrypted on MySQL DB System and HeatWave nodes, and the database volume is encrypted on the MySQL DB System. For information about EBS encryption, see Amazon EBS Encryption, in the *Amazon EC2 User Guide for Linux Instances*.

- Encryption in transit

Your data is always encrypted while in transit. HeatWave on AWS supports TLSv1.2 and requires that all MySQL client and application connections are encrypted. For added security, download a signed certificate bundle to enable host name identity verification for connecting clients and applications. For more information, see Enabling Host Name Identity Verification.

- MySQL Enterprise Data Masking and De-Identification

  Use data masking and de-identification to protect your sensitive data. HeatWave on AWS supports various MySQL data masking functions that mask data to remove identifying characteristics and generate random data with specific characteristics. The following data masking functions are supported:

  - `gen_range()`
  - `gen_rnd_email()`
  - `gen_rnd_ssn()`
  - `gen_rnd_us_phone()`
  - `mask_inner()`
  - `mask_outer()`
  - `mask_pan()`
  - `mask_pan_relaxed()`
  - `mask_ssn()`

  For more information, see MySQL Enterprise Data Masking and De-Identification.

- Password Validation

  HeatWave on AWS enforces strong passwords with the `validate_password` component, serves to improve security by requiring account passwords and enabling strength testing of potential passwords. The default value of the variables of the validate_password component are as follows, and you cannot change the default values:

  - `validate_password.check_user_name`
  - `validate_password.length`
  - `validate_password.mixed_case_count`
  - `validate_password.number_count`
  - `validate_password.policy`
  - `validate_password.special_char_count`

  Make sure your applications comply with the password requirements. For more information, see The Password Validation Component.

- MySQL Enterprise Firewall

  MySQL Enterprise Firewall enables database Administrators to permit or deny SQL statement execution based on matching against lists of accepted statement patterns. This helps harden MySQL against attacks such as SQL injection or attempts to exploit applications by using them outside of their legitimate query workload characteristics. For more information, see MySQL Enterprise Firewall.

2
Getting Started

This chapter describes how to access HeatWave on AWS, how to sign up for the HeatWave on AWS service, how to sign in, and provides an introduction to the HeatWave Console.

- Accessing HeatWave on AWS
- Signing Up
- Signing In
- Using the Console

2.1 Accessing HeatWave on AWS

Access HeatWave on AWS with the **HeatWave Console**. It is a browser-based interface available at https://cloud.mysql.com/.

The access requires that you have signed up for HeatWave on AWS. See Signing Up for details.

Once you have signed up, sign in to HeatWave Console. See Signing In for details. See Using the Console on how to use HeatWave Console.

After signing in and creating a DB System, you can also access HeatWave on AWS from a MySQL client or application. A MySQL user account is required; see Connecting from a Client.

2.2 Signing Up

This chapter describes how to sign up for the HeatWave on AWS Service.

- Sign-up Overview
- Sign-up Procedure

2.2.1 Sign-up Overview

The sign-up process for HeatWave on AWS consists of the following major steps

- Create an Oracle Cloud Free Tier Account, if you do not already have one.
- Enable the HeatWave on AWS service in your home region.

See Sign-up Procedure for detailed instructions.

Free Trial

Once signed up, you will receive free cloud credits you can use for up to 30 days to try out HeatWave on AWS. Once the free trial ends, you will no longer be able to update your resources or create new ones, but the resources you already created may still be viewed or deleted within a grace period, after which they will all be deleted. Upgrade to a paid account before the end of the grace period to maintain the data in your free-trial resources and to continue using HeatWave on AWS.

> **Note:**
>
> During free trial, you can enable the HeatWave on AWS service only in your home region. If you have chosen a home region where the HeatWave on AWS service is not available, you cannot use the HeatWave on AWS service during the free trial period. Check Region Availability. Upgrade to a paid account to enable the HeatWave on AWS service in any supported region.

See the Service Limits that apply during a free trial.

2.2.2 Sign-up Procedure

To sign up for HeatWave on AWS:

1. Create an Oracle Cloud Account (or proceed to Step 2 below if you already have one):
   a. Navigate to https://cloud.mysql.com and click **Sign Up**.
   b. Enter an email address and click **Continue**.
   c. Enter the **Account Information** and click **Verify my email**.
   d. In the verification email, click **Verify email** to verify the email address and continue the account setup process.
   e. Complete the **Account Information**, including **Password**, **Customer type**, **Cloud Account Name**, and **Home Region**. Click **Continue** after you are done.

      > **Tip:**
      >
      > An Oracle Cloud Account has a single region limit by default. For the **Home Region**, choose the OCI region that maps to your preferred AWS region (for which HeatWave on AWS is supported) to avoid the need for a region increase. See: Region Availability and Subscribed Region Limits for details.

   f. Complete the **Address Information** and click **Continue**.
   g. Click **Add payment verification method** and complete **Payment Verification**.
   h. Select **Agreement**, and click **Start my free trial**. Your Oracle Cloud account is being set up, which might take a while. You will then be directed to the Oracle Cloud Sign In Page.
2. Sign in to the Oracle Cloud Console:

> **Note:**
>
> To sign up for HeatWave on AWS, the registered user must either be a member of the `Administrators` group in the OCI `Default` identity domain (which is the case if you have gone through Step 1 above) or have the following permissions:
>
> - `manage heatwave-registration`
> - `manage policies`
> - `manage domains`
> - `manage groups`
> - `read limits`
> - `update tenancies`
>
> See Manage Identity Domain and Getting Started with Policies for more details on identity domains and policies on OCI.

a. On the Oracle Cloud Sign In page, enter your Oracle Cloud account name and click **Next**. Continue your sign in on the next page by providing your account credentials. After a successful sign in, you are directed to the **Oracle Cloud Console**.

b. Go to the HeatWave on AWS **Administration** page by doing one of the following:

- If you have gone through Steps 1 to sign up for an account, you should see on the **Get Started** tab the **Get started with HeatWave on AWS** banner. Click the **Go to service** button on it.
- If you already had an Oracle Cloud account and did not go through Step 1 before signing in to the **Oracle Cloud Console**, open the navigation menu on the top left and select **Databases**. Under , click **Administration** to go to the HeatWave on AWS **Administration** page.

3. Enable HeatWave on AWS service:

On the **HeatWave on AWS Region Management** pane, if the OCI region that maps to your preferred AWS region shows **Subscribed**, do the following to enable the HeatWave on AWS service:

a. Click **Enable** next to the preferred AWS region.

b. In the **Enable HeatWave on AWS on <AWS Region>** dialog box, click **Enable**. The HeatWave on AWS Administration page shows the current status of the sign-up process. Wait for the process to finish.

If the OCI region that maps to your preferred AWS region does NOT show **Subscribed**, follow these steps to subscribe to the desired OCI region (skip to step b if yours is already a paid account, and skip to step c if you already have sufficient region limits to subscribe to a new OCI region):

a. Request an account upgrade

i. On the HeatWave on AWS **Administration** page, click the **Request Upgrade** button.

ii. On the **Upgrade** page, complete the payment information by clicking the **Add Payment Method** button and following the steps.

iii. Unless you want to **Request a Sales Call** (click the corresponding button in that case), under **Pay as You Go**, select the account type (**Corporate** or **Individual**), agree to the **terms and conditions**, and click the **Upgrade your account** button.

iv. From the **Upgrade Confirmation** dialog box, click **Continue**.
The account upgrade might take some time.

b. Request a region limit increase:

i. From the OCI Console, Open the **Help** menu (?), and under **Targeted help**, click **Request service limit increase**. The Support Options page opens.

ii. Follow the steps in Requesting a Limit Increase to the Subscribed Region Count.

c. Subscribe to the preferred region and enable the HeatWave on AWS service:

i. From the OCI Console, open the navigation menu, and select **Databases**. Under HeatWave on AWS, click **Administration**.

ii. On the **HeatWave on AWS Region Management** pane, click **Subscribe** next to the OCI region that maps to the preferred AWS region.

iii. In the **Subscribe to <OCI Region>** dialog box, read the privacy notice, and then click **Subscribe**.

iv. Click **Enable** next to the preferred AWS region.

v. In the **Enable HeatWave on AWS on <AWS Region>** dialog box, click **Enable**.

> **Caution:**
>
> Do not change the Oracle Cloud Account name after enabling the HeatWave on AWS service, as it can cause a loss of access to the HeatWave on AWS service.

4. Start using HeatWave on AWS:

   On the HeatWave on AWS **Administration** page, click **Open HeatWave on AWS Console** on the **Welcome to HeatWave on AWS** banner. This will direct you to the HeatWave Console— for more information on it see Using the Console.

2.3 Signing In

This procedure assumes that you are an Oracle Cloud Account customer registered to use HeatWave on AWS. If you are not registered, see Signing Up.

Signing in to HeatWave on AWS requires:

- An Oracle Cloud Account name. This is the Oracle Cloud Account you registered with, chosen during account signup, or that was provided to you by an Account Administrator. In either case, you can find your Oracle Cloud Account name in your Oracle Cloud Account welcome email.
- Your user name and password.

If you forgot your Oracle Cloud Account name, click **Get help** in the sign-in dialog and enter the email address associated with the Cloud Account. Oracle will send you an email with a summary of your account information.

To sign in to your account:

1. Navigate your browser to https://cloud.mysql.com.

You are directed to the HeatWave on AWS welcome page.

2. Enter your Oracle Cloud Account name.
3. Click **Continue**.

   You are directed to the **Oracle Cloud Account Sign In** dialog.
4. Enter your user name and password and click **Sign In**.

   Once your user name and password are authenticated, you are directed to the HeatWave Console.

2.4 Using the Console

The HeatWave Console supports browser platforms supported by Oracle Jet. See Platforms supported by Oracle JET, in the *JavaScript Extension Toolkit* documentation.

Firewalls, proxy servers, or other devices that control access to the internet can affect the ability to connect to the HeatWave Console and OCI Console.

To allow network access to both the HeatWave Console and OCI Console, add the following URLs to the allowlist for firewalls and proxy servers:

```
*.mysql.com
*.oracle.com
*.oraclecloud.com
*.oracleinfinity.io
oracle.112.2o7.net
consent.trustarc.com
```

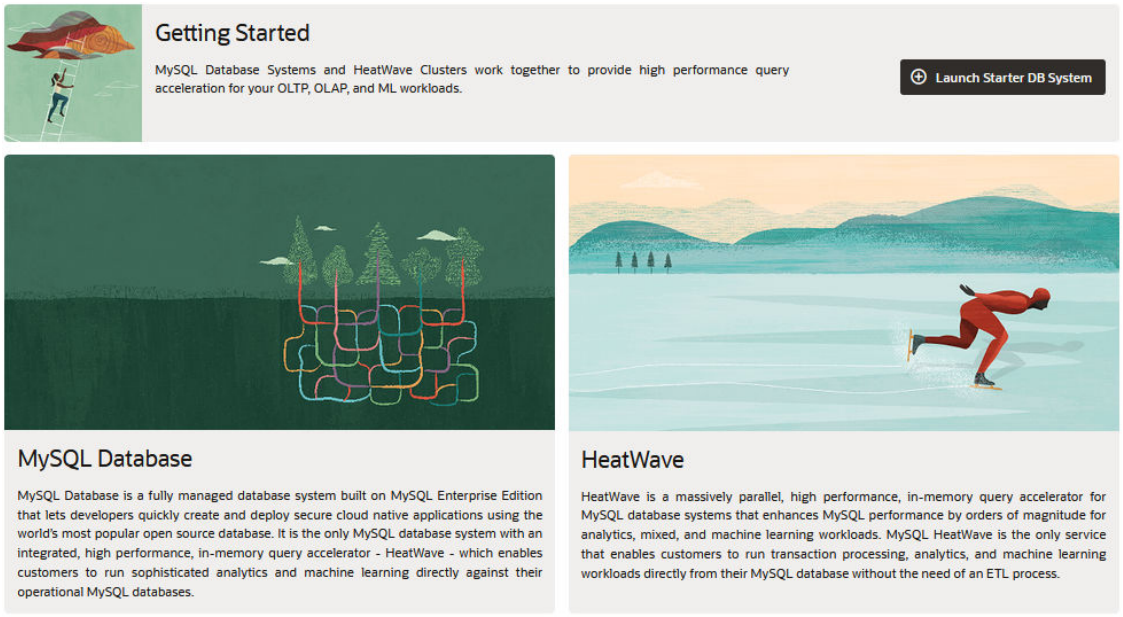
The last URL is for OCI Console cookie preferences.

- Console Overview

2.4.1 Console Overview

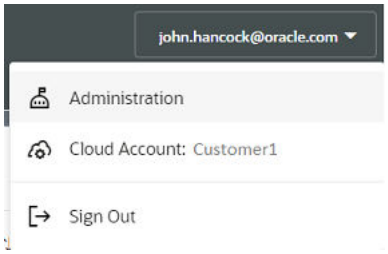
When you sign in to the HeatWave Console, you are directed to the **Introduction** page.

Figure 2-1 HeatWave Console Introduction Page



The profile menu on the HeatWave Console shows the account you are logged in to.

Figure 2-2 Profle Menu on the HeatWave Console



The navigation tabs allow you to navigate the pages of the HeatWave Console. The HeatWave Console pages include:

- **Introduction**

  This is the landing page where users are directed after signing in. It describes MySQL Database and HeatWave. The **Getting Started** region of the page provides a **Launch Starter DB System** button for Launching a Starter DB System.

- **HeatWave MySQL**

  Displays the **HeatWave MySQL** page, where you can create and manage MySQL DB Systems and backups.

- **HeatWave Clusters**

  Displays the **HeatWave Clusters** page, where you can create and manage HeatWave Clusters.

- **Workspaces**

Displays the **Workspaces** page, where you can connect to a DB System, manage HeatWave Cluster data, and run DB System and HeatWave queries using the Query Editor.

- **Performance**

  Displays the **Performance** page, where you can monitor HeatWave and MySQL performance metrics.

Signing Out

To sign out of the HeatWave Console, open the profile menu and click **Sign Out**.

3
DB Systems

This chapter covers how to create and manage MySQL DB Systems.

3.1 About DB Systems

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

DB System Components

A DB System consists of the following components:

- An Amazon EC2 instance type (with resources defined by the associated shape; see Supported Shapes for more information).
- Oracle Linux Operating System

> **Note:**
>
> It is not possible to access the operating system. Only the MySQL instance is exposed through the DB System endpoint.

- MySQL Server Enterprise Edition. For more information, see MySQL Server.
- Virtual Network Interface Card (VNIC) which attaches the DB System to a subnet of the Virtual Private Cloud (VPC). For connections to the DB System, a public endpoint is exposed as a fully-qualified domain name (FQDN). Specific IP addresses in CIDR format can be specified to limit access to the public endpoint.
- Network-attached block storage. HeatWave on AWS uses Amazon EBS block storage. For more information, see Amazon Elastic Block Store (Amazon EBS) in the *Amazon EC2 User Guide for Linux Instances*.

DB System Storage

Storage size should be determined based on your data size and performance requirements.

For information about increasing the storage available to the DB System, see Increasing DB System Storage.

3.1.1 Supported Shapes

HeatWave on AWS supports the following MySQL and HeatWave node shapes.

Table 3-1 MySQL Shapes

Shape Name	ECPUs	vCPUs	Memory (GiB)
MySQL.2.16GB	1	2	16
MySQL.4.32GB	2	4	32
MySQL.8.64GB	4	8	64
MySQL.32.256GB	16	32	256

Table 3-2 HeatWave Node Shapes

Shape Name	Memory (GiB)
HeatWave.16GB	16
HeatWave.256GB	256

- If you intend to use HeatWave AutoML functionality, the `HeatWave.256GB` node shape is recommended when creating a HeatWave Cluster. The `HeatWave.16GB` node shape may not have enough memory for training on large data sets. If you see error messages about this (such as `ML003024`), use the larger shape instead.

3.1.2 MySQL Server

This section describes various aspects of MySQL Server including versioning, upgrades, error logging, unsupported features, and limitations.

3.1.2.1 Server Versioning

The MySQL Server included in the HeatWave on AWS uses a versioning system consisting of three numbers, an update version, and the `-cloud` suffix. For example: `8.0.30-u1-cloud`.

- *First number*: The major release number.
- *Second number*: The minor release number. Taken together, the major and minor numbers constitute the release series number.
- *Third number*: The version number within the release series. This is incremented for each new release.
- *uN* : The MySQL Server-specific update number. Fixes and feature development for the cloud version of MySQL Server are delivered according to a different schedule than the on-premise version.
- *cloud*: suffix indicating this version of MySQL Server was built for the cloud, only.

To retrieve the MySQL Server version number, connect to your DB System using a MySQL client, and run `SELECT @@version;`. The following example shows the command and typical output:

```
mysql> SELECT @@version;
+-----------------+
| @@version       |
+-----------------+
| 8.0.30-u1-cloud |
+-----------------+
```

3.1.2.2 Server Upgrades

HeatWave on AWS supports the most recent release version of MySQL.

- Minor versions, such as 8.4.*nn*, must be applied manually. See Upgrade MySQL Version for information.
- Upgrade versions, such as 8.4.*nn*-u*n*, are applied automatically and according to the maintenance window defined on the DB System.

For more information about MySQL Server versioning, see Server Versioning.

It is not possible to roll back (downgrade) an upgrade. It is recommended to perform a full backup before upgrading your DB System.

3.1.2.3 Server Error Logging

HeatWave on AWS logs MySQL Server diagnostic information such as errors, warnings, and status updates to the Performance Schema `error_log` table. You can use a command-line client such as MySQL Client or MySQL Shell to view this data. For information about connecting to the DB System from a client, see Connecting from a Client. If connecting from MySQL Shell, you must switch to SQL mode by typing `\sql` at the MySQL Shell prompt.

To view logs in the `error_log` table, run the following statement:

```
mysql> SELECT * FROM performance_schema.error_log;
```

You get a response similar to the following, which displays the event timestamp, the thread ID, the event priority, the event error code (if present), subsystem in which the event occurred, and text describing the event:

```
*************************** 1. row ***************************
    LOGGED: 2022-04-07 19:17:03.981201
 THREAD_ID: 0
      PRIO: Warning
ERROR_CODE: MY-011068
 SUBSYSTEM: Server
      DATA: The syntax 'skip_slave_start' is deprecated and will be removed
            in a future release. Please use skip_replica_start instead.
*************************** 2. row ***************************
    LOGGED: 2022-04-07 19:17:03.98388
 THREAD_ID: 0
      PRIO: Note
ERROR_CODE: MY-010096
 SUBSYSTEM: Server
      DATA: Ignoring --secure-file-priv value as server is running with
            --initialize(-insecure).
*************************** 3. row ***************************
    LOGGED: 2022-04-07 19:17:03.983921
 THREAD_ID: 0
      PRIO: Note
ERROR_CODE: MY-010949
 SUBSYSTEM: Server
      DATA: Basedir set to /usr/.
....
```

To filter the logs to show only errors, run this statement:

```
mysql> SELECT * FROM performance_schema.error_log
          WHERE PRIO='error';
```

To filter the logs to only show errors from the `RAPID`, HeatWave, subsystem, run this statement:

```
mysql> SELECT * FROM performance_schema.error_log
          WHERE SUBSYSTEM IN ('RAPID');
```

You can select from the following subsystems:

- `Server`
- `InnoDB`
- `RAPID`

To filter on the `RAPID` (HeatWave) subsystem over a 2 hour time interval, run the following command:

```
mysql> SELECT * FROM performance_schema.error_log
          WHERE SUBSYSTEM IN ('RAPID');
```

To retrieve a count of how many instances of a specific error occurred in a single day, run the following command:

```
mysql> SELECT HOUR(LOGGED), count(*)
          FROM performance_schema.error_log
          WHERE ERROR_CODE = 'MY-010914'
          AND LOGGED > DATE_SUB(NOW(),INTERVAL 1 DAY)
          GROUP BY HOUR(LOGGED);
```

To retrieve a count of how many instances of a specific error occurred in a week, run the following command:

```
mysql> SELECT DAY(LOGGED), count(*)
          FROM performance_schema.error_log
          WHERE ERROR_CODE = 'MY-010914'
          AND LOGGED > DATE_SUB(NOW(),INTERVAL 1 WEEK)
          GROUP BY DAY(LOGGED);
```

3.1.2.4 Unsupported MySQL Server Features

The following features of MySQL Server are currently unsupported in HeatWave on AWS:

- Authentication plugins other than Native Pluggable Authentication (`mysql_native_password`), SHA-256 Pluggable Authentication (`sha256_password`), and Caching SHA-2 Pluggable Authentication (`caching_sha2_password`).
- Modification of system tables
- Binary log access
- Error Logging to MySQL Server error log. Error logging is available through the Performance Schema. See Server Error Logging.
- Group Replication plugin
- InnoDB Tablespace Encryption
- Setting global variables
- Persisted system variables
- Replication filters
- Semisynchronous replication
- Transportable tablespaces

3.1.2.5 MySQL Storage Engines

HeatWave on AWS supports only the InnoDB storage engine.

If you intend to migrate to HeatWave on AWS, and are not currently using the InnoDB storage engine, your data must be converted to use the `InnoDB` storage engine before the data is imported.

You can manually convert tables to InnoDB using the following `ALTER TABLE` statement:

```
mysql> ALTER TABLE table_name ENGINE=InnoDB;
```

The recommended method for migrating data is to use the MySQL Shell client. MySQL Shell dump utilities provide a `ocimds` option that checks for various incompatibilities, including tables defined with unsupported storage engines. Should incompatibilities exist, the MySQL Shell `compatibility` option can be used to alter MySQL Shell dump files to fix the incompatibilities, including changing the storage engine to InnoDB. For more information, see Importing Data.

3.1.2.6 Plugins and Components

The following MySQL Server plugins and components are loaded by default. You do not need to install any of these plugins.

- MySQL Enterprise Thread Pool

  Implements a thread pool that increases server performance by efficiently managing statement execution threads for large numbers of client connections. For more information, refer to MySQL Enterprise Thread Pool, in the *MySQL Reference Manual*.

- Connection Control Plugins

  Introduces an increasing delay in server response to connection attempts after a number of consecutive failed attempts. This capability provides a deterrent that slows down brute force attacks against MySQL user accounts. For more information, refer to The Connection-Control Plugins, in the *MySQL Reference Manual*.

- Password Validation Component

  Improves security by requiring account passwords and enabling strength testing of potential passwords. For more information, refer to The Password Validation Component, in the *MySQL Reference Manual*.

- MySQL Enterprise Data Masking and De-Identification

  Helps protect sensitive data from unauthorized uses by hiding and replacing real values with substitutes. For more information, refer to MySQL Enterprise Data Masking and De-Identification, in the *MySQL Reference Manual*.

- MySQL Enterprise Firewall

  Protects your data by monitoring, alerting, and blocking unauthorized database activity. For more information, refer to MySQL Enterprise Firewall, in the *MySQL Reference Manual*.

- MySQL Enterprise Encryption

  Provides built-in, server-side asymmetric encryption, key generation, digital signatures, and other cryptographic features to help protect confidential data using public and private keys. Only the MySQL Enterprise Encryption *component* functions are supported. The MySQL Enterprise Encryption *plugin* functions are deprecated and not supported by HeatWave on AWS. For more information, refer to MySQL Enterprise Encryption, in the *MySQL Reference Manual*.

- MySQL Enterprise Audit

  Enables the MySQL Server to produce a log file containing an audit record of server activities. The log contents can include information such as when clients connected and disconnected, what actions they performed while connected, and which databases and tables they accessed. See MySQL Enterprise Audit for details.

3.1.2.7 MySQL Enterprise Audit

HeatWave on AWS supports the MySQL Enterprise Audit plugin for MySQL Server 8.1.0 and later. The audit plugin enables the MySQL Server to produce a log file containing an audit record of server activities. The log contents can include information such as when clients connected and disconnected, what actions they performed while connected, and which databases and tables they accessed.

You can add statistics for the time and size of the queries to detect outliers. By default, the audit plugin is disabled. You have to enable the plugin and define audit plugin filters to log auditable events for all or specific users.

To use the database audit plugin, you need to:

- Connect to the DB System
- Use the **Query Editor** on the **Workspaces** page or your command-line client to perform the following operations.

Enable Audit Plugin

The audit plugin is disabled in the default MySQL configurations on HeatWave on AWS. Enable it by setting the MySQL server system variable `audit_log_disable` to `OFF` in a custom configuration for your MySQL server and then use it to create your new DB system or update an existing one (see Creating a MySQL Configuration for details on setting preferred values for User-Configurable System Variables).

Granting Audit Administration Privileges

By default, the administrator user you defined while creating the DB system has the audit administration privileges. You can grant the privileges to more users (for example, to the user `User001`) by running the following command:

```
GRANT AUDIT_ADMIN ON *.* TO <User001>;
```

Defining Audit Plugin Filters

Define your audit plugin filters to log auditable events for all or specific users. See Writing Audit Log Filter Definitions for details. Here are some sample filters and the commands to define them:

- To audit all events, run the following command:

  ```
  SELECT audit_log_filter_set_filter('log_all', '{ "filter": { "log":
  true } }');
  ```

- To audit connection events only, run the following command:

  ```
  SELECT audit_log_filter_set_filter('log_conn_events', '{"filter":
  {"class": { "name": "connection" }}}');
  ```

- To view audit filters, run the following command:

  ```
  SELECT * FROM mysql_audit.audit_log_filter;
  ```

Assign the filters you created above to all or specific users for them to access the audit data. For example:

- To assign the default audit filter to log all events from any account, use the wildcard character `%`:

```
SELECT audit_log_filter_set_user('%', 'log_all');
```

- To assign the default audit filter to log all connect events from any account, use the wildcard character `%`:

```
SELECT audit_log_filter_set_user('%', 'log_conn_events');
```

- To assign the default audit filter to log all events from a specific user such as `user_dba`, run the following command:

```
SELECT audit_log_filter_set_user('user_dba@%', 'log_all');
```

- To view the assigned rules, run the following command:

```
SELECT * FROM mysql_audit.audit_log_user;
```

- To unassign the rules from the user `user_dba`, run the following command:

```
SELECT audit_log_filter_remove_user('user_dba@%');
```

Accessing and Analyzing Audit Data

Use the audit data to monitor the DB system. Use these statements to access the audit data through the functions `audit_log_read()` and `audit_log_read_bookmark()` :

- To view any new logs since you last checked, run the following command (this ensures you are always updated with the latest audit log entries):

```
SELECT audit_log_read(audit_log_read_bookmark());
```

- To extract audit log contents starting from a particular timestamp, provide additional parameters within the `audit_log_read()` function:

```
SELECT audit_log_read('{ "start": { "timestamp": "2024-01-24 12:30:00" },
"max_array_length": 500 }');
```

- To view the audit data in an easier to read format, use the `JSON_PRETTY()` and `CONVERT()` functions; for example:

```
SELECT JSON_PRETTY(CONVERT(audit_log_read( '{ "start": { "timestamp":
"2024-01-24 12:30:00" }, "max_array_length": 500 }') USING UTF8MB4));
```

- To transform data into a tabular format, use the MySQL JSON functions. For example, you can transform a subset of the JSON name-value pairs into a structured table format, making it easier to interact with and analyze data:

```
SELECT @@server_uuid as server_uuid, ts, class, event,
login_ip,login_user,connection_id,
 status,connection_type,_client_name,_client_version,
 command,sql_command,command_status
FROM
```

```
JSON_TABLE
(
  AUDIT_LOG_READ( '{ "start": {\"timestamp\": \"2024-01-16 15:33:37\"},
\"max_array_length\": 10 }' ),
  '$[*]'
  COLUMNS
  (
    ts TIMESTAMP PATH '$.timestamp',
    class VARCHAR(20) PATH '$.class',
    event VARCHAR(80) PATH '$.event',
    login_ip VARCHAR(200) PATH '$.login.ip',
    login_user VARCHAR(200) PATH '$.login.user',
    connection_id VARCHAR(80) PATH '$.connection_id',
    status INT PATH '$.connection_data.status',
    connection_type VARCHAR(40) PATH '$.connection_data.connection_type',
    _client_name VARCHAR(80) PATH
'$.connection_data.connection_attributes._client_name',
    _client_version VARCHAR(80) PATH
'$.connection_data.connection_attributes._client_version',
    command VARCHAR(40) PATH '$.general_data.command',
    sql_command VARCHAR(40) PATH '$.general_data.sql_command',
    command_status VARCHAR(40) PATH '$.general_data.status'
   )) as audit_log;
```

- To further refine the data extraction, use `WHERE` clauses (for example, `WHERE connection_type <> 'SSL'`) in the SQL statements.

Related Topics

- MySQL Enterprise Audit

3.1.2.8 HeatWave on AWS Service Restrictions

The following items are currently not permitted in HeatWave on AWS:

- Custom TLS certificates.
- Changing the shape associated with a DB System. To change a shape, you must restore a backup of the DB System to a new DB System, and change the shape as part of that process. See Restoring a Backup to a New DB System for more information.

3.1.2.9 Default MySQL Privileges

This section lists the MySQL privileges granted to the MySQL Administrator user on the DB System and those explicitly revoked on the `mysql` and `sys` schemas. Revoked privileges cannot be granted to any MySQL user. For more information, see MySQL Privileges.

- *Global Privileges Granted*
  - `ALTER`
  - `ALTER ROUTINE`
  - `CREATE`
  - `CREATE ROLE`
  - `CREATE ROUTINE`
  - `CREATE TEMPORARY TABLES`

  - CREATE USER
  - CREATE VIEW
  - DELETE
  - DROP
  - DROP ROLE
  - EVENT
  - EXECUTE
  - INDEX
  - INSERT
  - LOCK TABLES
  - PROCESS
  - REFERENCES
  - REPLICATION CLIENT
  - REPLICATION SLAVE
  - SELECT
  - SHOW DATABASES
  - SHOW VIEW
  - TRIGGER
  - UPDATE
- *Global Dynamic Privileges Granted*
  - APPLICATION_PASSWORD_ADMIN
  - BACKUP_ADMIN
  - CONNECTION_ADMIN
  - FIREWALL_ADMIN
  - FIREWALL_USER
  - FLUSH_OPTIMIZER_COSTS
  - FLUSH_STATUS
  - FLUSH_TABLES
  - FLUSH_USER_RESOURCES
  - REPLICATION_APPLIER
  - ROLE_ADMIN
  - XA_RECOVER_ADMIN
- *Privileges Revoked from mysql Schema*
  - ALTER
  - ALTER ROUTINE
  - CREATE

  - CREATE ROUTINE
  - CREATE TEMPORARY TABLES
  - CREATE VIEW
  - DELETE
  - DROP
  - EVENT
  - EXECUTE
  - INDEX
  - INSERT
  - LOCK TABLES
  - REFERENCES
  - TRIGGER
  - UPDATE
- *Privileges Revoked from sys Schema*
  - ALTER
  - ALTER ROUTINE
  - CREATE
  - CREATE ROUTINE
  - CREATE TEMPORARY TABLES
  - CREATE VIEW
  - DROP
  - EVENT
  - INDEX
  - LOCK TABLES
  - REFERENCES
  - TRIGGER

3.1.2.10 Reserved User Names

The following user names are reserved and cannot be used for the MySQL Administrator user name:

- `administrator`
- `ociadmin`
- `ocimonitor`
- `ocirpl`
- `mysql.sys`
- `mysql.session`
- `mysql.infoschema`

3.2 Launching a Starter DB System

A Starter DB System is a preconfigured HeatWave MySQL DB System with preloaded sample data that you can easily launch for testing out HeatWave on AWS. The following are included with your Starter DB System launch:

- A DB System running the latest MySQL version with 2 vCPUs, 16GiB memory, and 50 GiB storage.
- Preloaded `airportDB` and `TPCH` sample databases.
- An attached one-node HeatWave Cluster with 256 GiB memory.

Follow these steps to launch the Starter DB System:

- In the HeatWave Console, select the **Introduction** tab.
- Click the **Launch Starter DB System** button in the **Getting Started** region. The **Launch Starter DB System** dialog appears.
- Specify the **Administrator credentials**.
  - **Username**: Name of the user that will be granted a specific set of MySQL Server administrator privileges. See Default MySQL Privileges. The administrator username can have up to 32 characters. Certain names are reserved. See Reserved User Names.
  - **Password**: The administrator password, which must be between 8 and 32 characters, and include at least one number, one uppercase letter, one lowercase letter, and one character from `,.-+*;:_!#%&/()=?><`.
  - **Confirm Password**: Reenter the administrator password.
- Click the **Launch** button. A DB system named **Starter DB System** (you can change the name later) is being launched.

You are redirected to the **HeatWave MySQL** tab, on which the **DB Systems** tab allows you to monitor the state of the launch operation, which may take some time to complete. The state will change from **Creating** to **Active** when the operation has completed successfully.

Click the row of the starter DB system to open the **MySQL DB System Details** pane below the list of DB Systems or click on the **Starter DB System** to open the **MySQL DB System Details** page, to review the **DB System Information**.

3.3 Creating a DB System

Check that the correct AWS region is shown at the top of the HeatWave Console. If it is not, see: Manage Regions

Creating a DB System requires administrator credentials, a hardware configuration, an AWS Availability Zone, a MySQL configuration and version, a maintenance window, network settings, backup policy, and a HeatWave Cluster configuration.

- Administrator credentials.
  HeatWave on AWS grants a specific set of MySQL Server privileges to an administrator. See Default MySQL Privileges. The administrator user name can have up to 32 characters. Certain names are reserved. See Reserved User Names.

The administrator password must be between 8 and 32 characters, and include at least one number, one uppercase letter, one lowercase letter, and one character from ,.-+*;:_! #%&/()=?><

- Hardware configuration.
  The shape determines the resources that HeatWave on AWS allocates to the DB System. See Supported Shapes.

  > **Note:**
  >
  > HeatWave on AWS does not support changes to the MySQL shape. To use a different shape requires a new DB System with the new shape. Restore the data from a backup of the old DB System to the new DB System.

  MySQL uses the data storage for all data, logs, and temporary files. Binaries are not stored in this block storage. The maximum storage allocation is 65536 GiB.

  > **Note:**
  >
  > Ensure that the data storage capacity is sufficient for any data import.

- AWS Availability Zone.
  An Availability Zone ID, AZ ID, identifies the physical Availability Zone. See Availability Zone IDs for your AWS resources, in the *AWS RAM User Guide*.

  > **Note:**
  >
  > To maximize performance, place the DB System in the same Availability Zone as the applications that will use the DB System.

- MySQL configuration.
  Use a MySQL configuration to determine the values of global system variables. See Configuration.

  > **Note:**
  >
  > Not all MySQL configurations support HeatWave Clusters. Make sure to select a MySQL configuration that does support HeatWave Clusters to attach a HeatWave Cluster to the DB System. The default configuration will support HeatWave Clusters.

- Maintenance Window.
  The maintenance window is a two-hour period specified in Coordinated Universal Time, UTC. When an update is available, HeatWave on AWS initiates it during this window. The time required to apply patches and updates may extend beyond the maintenance window and require a DB System restart. For more information, see Maintenance.

  Modify the Maintenance Window later by editing the DB System. See Editing a DB System.

- Network settings: allowed client addresses.

Specify the public-facing client IPv4 addresses that are permitted to connect to the DB System endpoint. These must all be public IP addresses that are capable of accessing the Internet, and not private IP addresses. For example:

- The public IP address for the system you are using to set up HeatWave on AWS.
- The public IP address for a system that an administrator will use to manage HeatWave on AWS.
- The public IP addresses of application servers where client applications that will use HeatWave on AWS are running.
- A public IP address range assigned to your organization.

The IP addresses are specified in Classless Inter-Domain Routing, CIDR, format; for example: `1.2.3.4/24`. Multiple addresses in CIDR format can be specified in a semicolon-separated list; for example: `1.2.3.4/24; 1.2.3.4/32`.

CIDR notation permits specifying a range of IP addresses in a single address. A CIDR address looks like a regular IP address but is suffixed with a forward slash followed by a number. The number is a compact representation of the subnet mask; for example, `/24` is equivalent to `255.255.255.0`, where 24 is the number of bits in the binary representation of `255.255.255.0` (`11111111.11111111.11111111.00000000`). `1.2.3.4/24` is equivalent to `1.2.3.4/255.255.255.0`, which permits this IP address range: `1.2.3.0` to `1.2.3.255`.

The following table shows common subnet masks, the equivalent CIDR notation, and the number of IP addresses in the range.

Table 3-3 CIDR Notation Examples

Subnet Mask	CIDR Notation	Number of IP Addresses
255.0.0.0	/8	16777216
255.255.0.0	/16	65536
255.255.255.0	/24	256
255.255.255.128	/25	128
255.255.255.192	/26	64
255.255.255.224	/27	32
255.255.255.240	/28	16
255.255.255.248	/29	8
255.255.255.252	/30	4
255.255.255.254	/31	2
255.255.255.255	/32	1

Specify a combination of CIDR addresses to permit a specific range of IP address. For example, to permit five IP addresses in the range of `192.0.2.0` to `192.0.2.4`, specify these CIDR addresses: `192.0.2.0/30; 192.0.2.4/32`. To permit a single IP address such as `192.0.2.10`, specify `192.0.2.10/32`. More information about CIDR notation including IP range to CIDR calculators can be found by searching online.

- Backup policy.
  Automatic backups are optional. The retention period is from 1 to 35 days for automatic backups, with a default of 7 days. An automatic backup will start during the hour following the **Backup start time**, if this is set. The default time to start an automatic backup is a time between 11:00 and 07:00 UTC, and will be the same time every day.
- HeatWave Cluster configuration.
  The HeatWave Cluster shape determines the resources allocated to each HeatWave node. See Supported Shapes.

The cluster size is the number of HeatWave nodes, and can be between 1 and 128.

Because the new DB System does not yet contain any data, HeatWave Autopilot cannot estimate the required cluster size, see: Estimating Cluster Size with HeatWave Autopilot.

> **Tip:**
>
> To make an estimate after loading the data to the DB System, delete the HeatWave Cluster and create a new one following the instructions in Creating a HeatWave Cluster.

HeatWave on AWS does not support changes to the HeatWave Cluster shape. To use a different shape requires removing the existing HeatWave Cluster and creating a new one.

> **Note:**
>
> For HeatWave AutoML functionality, the `HeatWave.256GB` node shape is recommended when creating a HeatWave Cluster. The `HeatWave.16GB` node shape may not have enough memory for training on large data sets. If error messages appear, such as `ML003024`, use the larger shape.

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **DB Systems** page, click **Create MySQL DB System** to open the **Create MySQL DB System and HeatWave Cluster** dialog.
3. **Basic information**: Provide a name and description for the DB System:
   - **Display Name**: Specify a display name for the DB System or use the generated default name.
   - **Description**: Specify a user friendly description for the DB System.
4. **Administrator credentials**: Specify the administrator credentials.
   - **Username**: Specify a user name for the administrator.
   - **Password**: Specify a password for the administrator.
   - **Confirm Password**: Re-enter the administrator password.
5. **Hardware configuration**: Select a suitable hardware configuration.
   - **Shape**: Select a shape for the DB System.
   - **Data Storage Size (GiB)**: Specify the amount of block storage, in GiB, to allocate to the DB System.
6. **Availability zone**: This determines the physical location of the DB System:
   - **Automatic**: AWS selects the physical AWS Availability Zone.
   - **Manual**: Select a physical AWS Availability Zone.
7. **MySQL Configuration**: Select a MySQL configuration.
   - Click **Change** to change the default configuration.
   - Select a configuration from the list.
   - Click **Apply MySQL Configuration**.

8. **Database Version**: Select the MySQL Server version to deploy. The latest MySQL Server version is selected by default.
9. **Maintenance Window**: Select the start time for the maintenance window.
   - Select **Automatic** for HeatWave on AWS to choose the **Start day** and **Start time**.
   - Select **Manual** to specify the maintenance window **Start day** and **Start time**.
10. **Networking**: Configure network settings:
    - **Allowed Client Addresses**: Specify the public-facing client IPv4 addresses that are permitted to connect to the DB System endpoint.
    - **Port**: The port on which the MySQL server listens. The default is port 3306. Specify a port number between 1024 and 65535.
    - **X Protocol Port**: The X Protocol port on which the MySQL server listens, supported by clients such as MySQL Shell. The default port is 33060. Specify a port number between 1024 and 65535.
11. **Backup policy**: Select the backup retention period, and the start time for automatic backups.
    - Select **Enable Automatic Backups** for HeatWave on AWS to take backups.
    - Set a retention period between 1 and 35 days.
    - Select **Select Backup Window** to set a preferred time to start a backup.
    - Set the **Backup start time**.
12. **IAM roles**:
    - **Data Import role ARN**: If you want to import data with Console, specify the data import role ARN. See Creating an IAM Role to Access an Amazon S3 Bucket.
13. Click **Next**.
14. **Provision HeatWave Cluster**: Select whether to provision a HeatWave Cluster. This is only available if the chosen MySQL configuration supports HeatWave.
    - **Basic information**: Provide a name and description for the HeatWave Cluster:
      - **Display Name**: Specify a display name for the HeatWave Cluster or use the generated default name.
      - **Description**: Specify a user-friendly description of the HeatWave Cluster.
    - **HeatWave Cluster configuration**: Specify the shape and cluster size.
      - **Shape**: Select the shape to use for the HeatWave Cluster.
      - **Cluster Size**: Specify the size of the cluster between 1 and 128 HeatWave nodes.
15. Click **Create**.
    HeatWave Console returns to the **DB Systems** page to monitor the state of the operation, which may take some time to complete. The state will change from **Creating** to **Active** when the operation has completed successfully.

    Click the name of the DB System to open the **DB System Details** pane and review the **DB System Information**.

3.4 Managing a DB System

This section describes how to manage MySQL DB Systems using the Console.

- Stopping, Starting, or Restarting a DB System

- Editing a DB System
- Update Networking
- Update MySQL Configuration
- Upgrade MySQL Version
- Increasing DB System Storage
- Deleting a DB System

3.4.1 Stopping, Starting, or Restarting a DB System

A DB System starts automatically after it is created, and the **Start** button is disabled. A running DB System shows the **Active** state. For information on DB System states, see MySQL DB System Details.

Restarting a DB System with the **Restart** button shuts down the DB System, and then restarts it immediately.

Stopping a DB System with the **Stop** button stops billing for it. However, billing continues for storage. Billing for the DB System resumes when the DB System starts again.

Start, stop, or restart operations on a DB System also affect an associated HeatWave Cluster. When a DB System restarts, the HeatWave Cluster also restarts, and it reloads data from the DB System. See: Data Load Progress and Status Monitoring.

To start, stop, or restart a DB System:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **DB Systems** tab, in the list of DB Systems, find the DB System you want to start, stop, or restart, and do one of the following:
   - Click on the row of the DB System to highlight it, and then click the **Start**, **Stop**, or **Restart** button.
   - Click the name of the DB System to open the **DB System Details** page, and then click the **Start**, **Stop**, or **Restart** button.
3. When you select **Stop** or **Restart**, the **Stop/Restart MySQL DB System** dialog is displayed for you to choose the shutdown type:
   - **Slow** flushes dirty pages and purges undo log pages for older transactions. The shutdown itself can take longer, but the subsequent startup is faster.
   - **Fast** flushes dirty pages before shutting down the DB System. Some flush operations must be performed during next startup, potentially increasing the duration of the startup process.
   - **Immediate** does not flush dirty pages and does not purge any undo log pages. Stops MySQL immediately. Page flushes and log purging will take place during the next startup, increasing the duration of the startup process.

   Select a shutdown type and click the **Stop** or **Restart** button, depending on the action you are taking.

3.4.2 Editing a DB System

Use the Console to edit a DB System, and update the details about the DB System, the maintenance window, the backup policy, or the data import role ARN.

This task requires the following:

- (For data import feature only) Permissions to assume the data import role.

Do the following to edit a DB System:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **DB Systems** tab, in the list of DB Systems, find the DB System to edit, and do one of the following:
   - Click on the row of the DB System to highlight it, and choose **Edit DB System** from the **Actions** menu.
   - Click the name of the DB System to open the **DB System Details** page. Click the **Edit DB System** button.
3. **Basic information**: Edit the name and description of the DB System:
   - **Display Name**: Edit the display name for the DB System.
   - **Description**: Edit the user friendly description for the DB System.
4. **Maintenance Window**: Edit the start time for the maintenance window. The maintenance window is a two-hour period specified in Coordinated Universal Time, UTC. When an update is available, HeatWave on AWS initiates it during this window. The time required to apply patches and updates may extend beyond the maintenance window and require a DB System restart. See Maintenance.
   - Select **Automatic** for HeatWave on AWS to choose the **Start day** and **Start time**.
   - Select **Manual** to specify the maintenance window **Start day** and **Start time**.
5. **Backup policy**: Edit the backup retention period, and the start time for automatic backups.
   - Select **Enable Automatic Backups** for HeatWave on AWS to take backups. Automatic backups are optional. The retention period is from 1 to 35 days for automatic backups, with a default of 7 days. An automatic backup will start during the hour following the **Backup start time**, if this is set. The default time to start an automatic backup is a time between 11:00 and 07:00 UTC, and will be the same time every day.
   - Set a retention period between 1 and 35 days.
   - Select **Select Backup Window** to set a preferred time to start a backup.
   - Set the **Backup start time**.
6. **IAM roles**:
   - **Data Import role ARN**: If you want to import data with Console, specify the data import role ARN. See Creating an IAM Role to Access an Amazon S3 Bucket.
7. Click **Save**.

3.4.3 Update Networking

For the allowed client addresses, specify public-facing client IPv4 addresses that are permitted to connect to the DB System endpoint. Use CIDR format, for example: `1.2.3.4/24`. Specify multiple addresses in CIDR format with a semicolon-separated list, for example: `1.2.3.4/24; 1.2.3.4/32`. For information about the CIDR format, see Creating a DB System.

To update the network for a DB System:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **DB Systems** tab, in the list of DB Systems, find the DB System to edit, and do one of the following:

- Click on the row of the DB System to highlight it, and choose **Update Networking** from the **Actions** menu.
- Click the name of the DB System to open the **DB System Details** page. Click the **Update Networking** button.

3. Edit the **Allowed Client Addresses**.
4. Click **Save** to save the changes.

3.4.4 Update MySQL Configuration

Update the MySQL configuration for a DB System. This will change the user configurable system variables for the DB System to the values in the new configuration. See User-Configurable System Variables.

> **Note:**
>
> You cannot change the System Initialization Variables by updating the DB System's configuration. If you try to update your DB System with a MySQL configuration containing different system initialization variable values from those your DB System's, the update will fail.

During the update, the DB System will be in the **UPDATING** state until the update is finished. If the new configuration changes any system variables that are not dynamic, the MySQL database process, `mysqld`, will restart. If the DB System has an attached HeatWave Cluster, it will restart and reload data from the DB System.

If an update fails, the DB System will be rolled back to its original configuration. But if the rollback fails, the DB System will end up in the **UPDATING_ERROR** state.

To update the MySQL configuration for a DB System, make sure it is running, and follow these steps:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **DB Systems** tab, in the list of DB Systems, find the DB System to update, and do one of the following:
   - Click on the row of the DB System to highlight it, and choose **Update MySQL Configuration** from the **Actions** menu.
   - Click the name of the DB System to open the **DB System Details** page, and from the **Actions** menu choose **Update MySQL Configuration**.
3. Click **Change**. The **Select MySQL Configuration for shape …** dialog box appears.
4. Select the desired MySQL configuration from a list. Select a filter for the list:
   - **Any**: Show any configuration that can be applied to your DB System
   - **Default**: Show the default configuration for the MySQL shape of your DB System.
   - **Custom**: Show all custom configurations you have created by Creating a MySQL Configuration or Copying a MySQL Configuration that are applicable to your DB System.

   Refine the list by typing into the search bar part of the names of the configurations you are interested in.

> **Note:**
>
> Only configurations that support the DB System's shape are shown. Also, if a DB System has a HeatWave Cluster attached, only configurations that support HeatWave are shown; to change the configuration to one that does not support HeatWave, remove the DB System's HeatWave Cluster.

Click **Apply MySQL Configuration** to confirm your choice.

5. Click **Update** to actually apply the configuration to your DB System.

   If any problems occur updating the configuration, the DB System will roll back to its previous configuration.

Related Topics

- System Initialization Variables
- User-Configurable System Variables
- User-Configurable Shape-Dependent System Variables
- Creating a MySQL Configuration
- Copying a MySQL Configuration
- View MySQL Configuration Details

3.4.5 Upgrade MySQL Version

> **Note:**
>
> It is not possible to roll back a DB System upgrade. It is recommended to perform a full backup of the DB System before upgrading. See Creating a Backup.

See Server Versioning. The available options will depend upon the current version. These include:

- **Do not change**

  This option is always available, and shows the current version.

- **Update**

  This option is only available if the current version is still supported. Updates happen automatically during the Maintenance Window (see Editing a DB System). This option provides a manual update.

- ***A more recent MySQL version***

  This option is only available if a more recent version is available. It shows the version number, for instance, **8.4.*x***.

  More than one version might be available.

To upgrade the MySQL version:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.

2. On the **DB Systems** tab, in the list of DB Systems, find the DB System to edit, and do one of the following:
   - Click on the row of the DB System to highlight it, and choose **Upgrade MySQL Version** from the **Actions** menu.
   - Click the name of the DB System to open the **DB System Details** page. Click the **Upgrade MySQL Version** button.
3. Choose one of the following options (see detailed descriptions of the options above):
   - **Do not change**: Choose this option to ignore any update or upgrade option.
   - **Update**: Choose this option to update the current version.
   - ***A more recent MySQL version*** : Choose this option to upgrade to a more recent version.

3.4.6 Increasing DB System Storage

You can increase the DB System storage by updating the data storage size of a DB System, by backup and restore, or by export and import.

Increasing Storage by Updating the Data Storage Size of a DB System

Use the Console to increase the data storage size of a DB System that is in the `ACTIVE`, `STARTING_ERROR`, or `RESTARTING_ERROR` state. If the DB System is operating normally, resizing the storage will not restart the DB System, and you can continue to query it while the storage is being increased. Otherwise, resizing the storage may restart the DB System in order to return it to a normal operating state.

It is recommended that you create a backup of the DB System before updating the data storage size. See Creating a Backup.

> **Caution:**
>
> You can only increase the data storage size once every 6 hours. It is recommended that you increase your storage by an amount large enough to support your storage requirement in the next 6 hours.

Follow these steps to update the data sotrage size of a DB system:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **DB Systems** tab, in the list of DB Systems, find the DB System for which you want to update the data storage size, and do either of the following:
   - Click on the row of the DB System to highlight it, click **Actions**, and then click **Update Data Storage Size**.
   - Click the name of the DB system to open the **DB System Details** page. Click **Actions**, and then click **Update Data Storage Size**.
3. In the **Update Data Storage Size** dialog, enter the required storage size in GiB. You can only increase the storage size, and the maximum storage size is 65536.
4. Click **Update**.

Increasing Storage by Backup and Restore

1. Create a backup of the DB System. See Creating a Backup.

2. Create a new DB System from the backup, and define a larger storage in the new DB System. See Restoring a Backup to a New DB System.

Increasing Storage by Export and Import

1. Export the data from your current DB System using MySQL Shell. See Exporting Data Using MySQL Shell.
2. Use the HeatWave Console to create a new DB System with a larger storage size. See Creating a DB System.
3. Import the data into the new DB System using MySQL Shell. See Importing Data.

3.4.7 Deleting a DB System

Deleting a DB System permanently deletes it, along with all of the data in the database. Ensure all the data is appropriately backed up before deleting a DB System (see Backups).

To delete a DB System:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **DB Systems** tab, in the list of DB Systems, find the DB System you want to delete, and do one of the following:
   - Click on the row of the DB System to highlight it, and choose the **Delete** action from the **Actions** menu.
   - Click the name of the DB System to open the **DB System Details** page. Click the **Delete** button.

   The **Delete MySQL DB System** dialog is displayed.
3. Click **Delete MySQL DB System** to go ahead with the deletion.

3.5 Viewing DB System Details

Do the following to view DB System details:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **DB Systems** tab, in the list of DB System, find the DB System you want to view details for, and do one of the following:
   - Click the row of the DB System to highlight it. The DB System details appear below the list of DB Systems.
   - Click the name of the DB System to open the **MySQL DB System Details** page for the selected DB System.

For descriptions of DB System details, see MySQL DB System Details.

- MySQL DB System Details

3.5.1 MySQL DB System Details

The **MySQL DB System Details** page is divided into the following sections:

Table 3-4 MySQL DB System Details Page

Name	Description
Summary	DB System summary details. See Table 3-5.
DB System Information	DB System general information, configuration, endpoint, and HeatWave Cluster details. See Table 3-6.
Backups	DB System backup details. See Table 3-7.
Events	DB System event details. See Table 3-8.

Table 3-5 MySQL DB System Summary Details

Field	Description
State	The state of the DB System. • **CREATING**: Resources are being reserved for the DB System, the system is booting, and the initial database is being created. Creating can take several minutes. The system is not ready to use yet. • **ACTIVE**: The DB System was successfully created and is ready to be used. • **STARTING**: The DB System is being started. • **STOPPING**: The DB System is being stopped. • **RESTARTING**: The DB System is being restarted. • **INACTIVE**: The DB System is powered off. • **UPDATING**: The DB System is in the process of being updated. • **UPGRADING**: The DB System is in the process of being upgraded. • **DELETING**: The DB System is being deleted. • **DELETED**: The DB System has been deleted and is no longer available. • **FAILED**: An error condition prevented the creation or continued operation of the DB System. • **STARTING_ERROR**: An attempt to start the DB System failed. • **STOPPING_ERROR**: An attempt to stop the DB System failed. • **RESTARTING_ERROR**: An attempt to restart the DB System failed. • **UPDATING_ERROR**: An attempt to update the DB System failed. • **UPGRADING_ERROR**: An attempt to upgrade the DB System failed. • **DELETING_ERROR**: An attempt to delete the DB System failed.
Host Name	The host name of the DB System. The host name is a fully qualified domain name (FQDN).
Resource ID	The unique resource identifier assigned to the DB System when it is created.

Table 3-5 (Cont.) MySQL DB System Summary Details

Field	Description
Shape	The resource template for the MySQL instance. For information about shapes, see Supported Shapes.

Table 3-6 DB System Information Details

Field	Description
General Information	• **Description**: The user-defined description of the DB System. • **Created**: The date and time the DB System was created. • **Last Updated**: The date and time the DB System was last updated.
DB System Configuration	• **Storage Size**: The amount of storage available to the DB System. • **MySQL Version**: The MySQL Server version used by the DB System. • **Maintenance Window**: When updates are available, they are initiated during the maintenance window, which is a two-hour period starting at this time. The time required to apply patches and updates may extend beyond the maintenance window and require DB System restarts. For more information, see Maintenance.
Endpoint	• **Availability Zone**: The physical Availability Zone where the DB System resides. • **MySQL Port**: The port on which the MySQL Server listens. • **MySQL Port X**: The port on which the MySQL Server listens for clients using MySQL's X Protocol. • **Allowed Client Addresses**: The public-facing client IPv4 addresses that are permitted to connect to the DB System endpoint, specified in CIDR format. For information about how IP addresses are specified in CIDR format, see Creating a DB System.

Table 3-6 (Cont.) DB System Information Details

Field	Description
HeatWave Cluster	• **State**: The state of the HeatWave Cluster. – **CREATING**: Resources are being reserved for the HeatWave Cluster, and the cluster is being created. Creating can take several minutes. The system is not ready to use yet. – **ACTIVE**: The HeatWave Cluster was successfully created and is ready to be used. – **STARTING**: The HeatWave Cluster is being started. – **STOPPING**: The HeatWave Cluster is being stopped. – **RESTARTING**: The HeatWave Cluster is being restarted. – **INACTIVE**: The HeatWave Cluster is powered off. – **UPDATING**: The HeatWave Cluster is in the process of being updated. – **UPGRADING**: The HeatWave Cluster is in the process of being upgraded. – **DELETING**: The HeatWave Cluster is being deleted. – **DELETED**: The HeatWave Cluster has been deleted and is no longer available. – **FAILED**: An error condition prevented the creation or continued operation of the HeatWave Cluster. – **STARTING_ERROR**: An attempt to start the HeatWave Cluster failed. – **STOPPING_ERROR**: An attempt to stop the HeatWave Cluster failed. – **RESTARTING_ERROR**: An attempt to restart the HeatWave Cluster failed. – **UPDATING_ERROR**: An attempt to update the HeatWave Cluster failed. – **UPGRADING_ERROR**: An attempt to upgrade the HeatWave Cluster failed. – **DELETING_ERROR**: An attempt to delete the HeatWave Cluster failed. • **Name**: The name of the HeatWave Cluster. • **Cluster Size**: The number of HeatWave Cluster nodes.

Table 3-7 MySQL DB System Backup Details

Field	Description
Name	The name of the backup

Table 3-7 (Cont.) MySQL DB System Backup Details

Field	Description
State	The state of the backup • **CREATING**: Resources are being reserved for the DB System backup and the backup is being created. Creating can take several minutes. The backup is not ready to use yet. • **ACTIVE**: The backup was successfully created. • **UPDATING**: The backup is in the process of being updated. • **INACTIVE**: The backup is not available. • **DELETING**: The backup is being deleted. • **DELETED**: The backup has been deleted and is no longer available. • **FAILED**: An error condition prevented the creation or continued availability of the backup.
MySQL DB System	The name of the MySQL DB System used to create the backup.
Size (GiB)	The size of the backup - note that a backup is the size of the whole storage allocation for the DB System from which it was created, even if the volume does not contain that much data.
Created	The date and time the backup was created.

See Backups for details on MySQL DB System backups.

Table 3-8 MySQL DB System Event Details

Field	Description
Type	The event and its status.
Severity	The severity level of the event, which can be one of • `INFO`: General information is provided. No special concerns • `WARNING`: Warning on an event that might impact the DB System's operation if no actions are taken. • `CRITICAL`: A critical event. An urgent action is required to avoid adverse impact on your DB System.
Message	The message body of the event report.
Created	The date and time the event report was created.

See Events for details on event reports.

4
HeatWave Clusters

This chapter describes how to create and manage HeatWave Clusters.

A HeatWave Cluster consists of one or more HeatWave nodes. The MySQL DB System includes a HeatWave plugin that is responsible for cluster management, query scheduling, and returning query results to the MySQL DB System. HeatWave nodes store data in memory and process queries.

When a HeatWave Cluster is enabled and data is loaded, queries that meet certain prerequisites are automatically offloaded from the MySQL DB System to the HeatWave Cluster for accelerated processing.

4.1 Creating a HeatWave Cluster

A HeatWave Cluster must be associated with an active DB System, and a DB System can have only one HeatWave Cluster. Before you create a HeatWave Cluster, ensure that you have created a DB System and that the DB System does not already have a HeatWave Cluster. With HeatWave on AWS, you create a DB System and HeatWave nodes at the same time, but you can delete the HeatWave Cluster afterwards if you want to re-create it. See Creating a DB System for instructions.

When you create a HeatWave Cluster, you are presented with an option to estimate the required HeatWave Cluster size based on data that is loaded on your DB System (see Estimating Cluster Size with HeatWave Autopilot). If you have not loaded data into your DB System, and you want to estimate the optimal HeatWave Cluster size, load data into the DB System before you create a HeatWave Cluster. See Importing Data.

> **Note:**
>
> Changing the number of nodes in an existing HeatWave Cluster is not supported. If you require a larger or smaller cluster, you must delete the existing cluster and create a new one with the desired number of nodes.

To create a HeatWave Cluster, do the following:

1. In the HeatWave Console, select the **HeatWave Clusters** tab.
2. Click **Create HeatWave Cluster**. The **Create HeatWave Cluster** dialog is displayed.
3. On the **Create HeatWave Cluster** dialog, provide the following information:
   - **Basic Information**
     - **Display Name**: Specify a user-friendly display name for the HeatWave Cluster.

  - **Description**: Specify a user-friendly description of the HeatWave Cluster.
- **DB System Name**: Select a DB System from the drop down menu.
- **HeatWave Cluster Configuration**
  - **Shape**: Select a HeatWave node shape. For information about supported shapes, see Supported Shapes.

    > **Note:**
    >
    > If you intend to use HeatWave AutoML functionality, the `HeatWave.256GB` node shape is recommended when creating a HeatWave Cluster. The `HeatWave.16GB` node shape may not have enough memory for training on large data sets. If you see error messages about this (such as `ML003024`), use the larger shape instead.

  - **Cluster Size**: The number of HeatWave nodes to create. Enter a number between 1 and 128. Optionally, click **Estimate Cluster Size** to estimate the required cluster size using HeatWave Autopilot. For instructions, see Estimating Cluster Size with HeatWave Autopilot.

4. Click **Create** to create the HeatWave Cluster.

You are returned to the **HeatWave Clusters** page where you can monitor the state of the operation, which may take some time to complete. The state changes from **Creating** to **Active** when the operation has completed successfully.

- Estimating Cluster Size with HeatWave Autopilot

4.1.1 Estimating Cluster Size with HeatWave Autopilot

This topic describes how to estimate the optimal HeatWave Cluster size for your data.

A cluster size estimate is generated using HeatWave Autopilot machine learning techniques. HeatWave Autopilot analyzes the data on your MySQL DB System and recommends a cluster size. If you have not loaded data into your DB System, and you want to estimate the optimal HeatWave Cluster size, load data into the DB System before you create a HeatWave Cluster. See Importing Data.

Prerequisites:

- The data you intend to load into the HeatWave Cluster must be available on the DB System.
- Optionally, log into your DB System and run `ANALYZE TABLE` on tables you intend to load into the HeatWave Cluster. Estimates should generally be valid without running `ANALYZE TABLE` , but running `ANALYZE TABLE` ensures that estimates are as accurate as possible.

To estimate a cluster size:

1. Click **Estimate Cluster Size**.

   The **Estimate Cluster Size with Autopilot** dialog is displayed.

2. Select the schemas and tables you want to include in the estimate. Schemas are displayed in the **Schemas** pane. Tables belonging to the selected schema appear in the **Tables from selected schemas** pane.

   When schemas and tables are selected, the **Summary** details are adjusted automatically.

The **Schemas** pane provides the following information:

- **Name**: The schema name.
- **HeatWave Cluster Memory Usage (GiB)**: The estimated amount of HeatWave Cluster memory used by the schema.
- **Tables Selected**: The number of tables selected expressed as a fraction of the total number of tables.
- **Warnings**: The number of table warnings.

The **Tables from selected schemas** pane provides the following information:

- **Name**: The table name.
- **Warnings**: The number of table warnings. For a description of table warnings, see Cluster Size Estimate Table Warnings.
- **Memory Size Estimate (GiB)**: The estimated amount of HeatWave Cluster memory required for the table.
- **Rows Estimate**: The estimated number of table rows.

3. Review the **Summary** details, which include memory required by the schemas and tables selected, memory provided per node, HeatWave Cluster nodes required, and memory provided by the cluster.
4. To apply the cluster size estimate, click **Apply Cluster Size Estimate**.

   You are returned to the **Create HeatWave Cluster** dialog where the estimate is applied to the **Cluster Size** field.

Cluster Size Estimate Table Warnings

This topic describes table warnings that may appear in the **Tables from selected schemas** pane, in the **Estimate Cluster Size with MySQL Autopilot** dialog.

Table 4-1 Cluster Size Table Warnings

Table Status Issue	Description
`TOO MANY COLUMNS TO LOAD`	The table has too many columns. The column limit is 1017.
`ALL COLUMNS MARKED AS NOT SECONDARY`	There are no columns to load. All table columns are defined as `NOT SECONDARY`. Columns defined as `NOT SECONDARY` are excluded from the estimate. For more information, see Excluding Table Columns, in the *HeatWave User Guide*.
`CONTAINS VARLEN COLUMN WITH >65532 BYTES`	A VARLEN column exceeds the 65532 byte limit. For more information on `VARLEN`, see Variable-length Encoding in the *HeatWave User Guide*
`ESTIMATION COULD NOT BE CALCULATED`	The estimate could not be calculated. For example, a table estimate may not be available if statistics for `VARLEN` columns are unavailable.
`UNABLE TO LOAD TABLE WITHOUT PRIMARY KEY`	A table must be defined with a primary key before it can be loaded into HeatWave.

4.2 Managing a HeatWave Cluster

4.2.1 Starting, Stopping, or Restarting a HeatWave Cluster

A HeatWave Cluster starts automatically after it is created, and the **Start** button is disabled. A running HeatWave Cluster shows the **Active** state. For information on HeatWave Cluster states, see HeatWave Cluster Details.

Restarting a HeatWave Cluster with the **Restart** button shuts down the HeatWave Cluster and then restarts it immediately. Stopping a HeatWave Cluster with the **Stop** button stops it without restarting.

When a HeatWave Cluster is stopped through a stop or restart action, the data loaded in HeatWave Cluster memory is unloaded. When a HeatWave Cluster restarts, data is reloaded from AWS S3, and includes any changes that occur on the DB System while the HeatWave Cluster is offline.

Start, stop, or restart actions on a HeatWave Cluster have no effect on the DB System with which the HeatWave Cluster is associated.

To start, stop, or restart a HeatWave Cluster:

1. In the HeatWave Console, select the **HeatWave Clusters** tab.
2. In the list of HeatWave Clusters, find the HeatWave Cluster you want to start, stop, or restart, and do one of the following:
   - Click on the row of the HeatWave Cluster to highlight it, and then click the **Start**, **Stop**, or **Restart** button.
   - Click the name of the HeatWave Cluster to open the **HeatWave Cluster Details** page, and then click the **Start**, **Stop**, or **Restart** button.

4.2.2 Editing a HeatWave Cluster

Use the Console to edit the display name and description of a HeatWave cluster. You can also change the shape of the HeatWave Cluster nodes and resize the number of nodes in the HeatWave Cluster.

1. In the HeatWave Console, click the **HeatWave Clusters** tab.
2. In the list of HeatWave Clusters, find the HeatWave Cluster you want to edit, and do one of the following:
   - Click the row of the HeatWave Cluster to highlight it, and click **Edit HeatWave Cluster** from the **Actions** menu.
   - Click the name of the HeatWave Cluster to open the **Details** page and click **Edit HeatWave Cluster** from the **Actions** menu.
3. On the **Edit HeatWave Cluster** panel, provide the following information:

- **Basic information**
  - **Display Name**: Specify a name of the HeatWave Cluster.
  - **Description**: Specify a description of the HeatWave Cluster.
- **HeatWave Cluster configuration**

> **Note:**
>
> The HeatWave Cluster will be offline while the configuration is being changed.

> **Note:**
>
> You must reload the data into the HeatWave Cluster after changing the configuration.

  - **Shape**: Select a HeatWave node shape. For information about supported shapes, see Supported Shapes.

> **Note:**
>
> If you intend to use HeatWave AutoML functionality, the `HeatWave.256GB` node shape is recommended when creating a HeatWave Cluster. The `HeatWave.16GB` node shape may not have enough memory for training on large data sets. If you see error messages about this (such as `ML003024`), use the larger shape instead.

  - **Cluster Size**: The number of HeatWave nodes for the cluster. Enter a number between 1 and 128. Optionally, click **Estimate Cluster Size** to estimate the required cluster size using HeatWave Autopilot. For instructions, see Estimating Cluster Size with HeatWave Autopilot.

4. Click **Save**.

The details and configurations of the selected HeatWave Cluster is updated.

4.2.3 Deleting a HeatWave Cluster

Deleting a HeatWave Cluster removes the HeatWave Cluster nodes permanently. The DB System with which the HeatWave Cluster is associated is unaffected.

To delete a HeatWave Cluster:

1. In the HeatWave Console, select the **HeatWave Clusters** tab.
2. In the list of HeatWave Clusters, find the HeatWave Cluster you want to delete, and do one of the following:
   - Click on the row of the HeatWave Cluster to highlight it, and choose the **Delete** action from the **Actions** menu.
   - Click the name of the HeatWave Cluster to open the **HeatWave Cluster Details** page. Click the **Delete** button.

The **Delete HeatWave Cluster** dialog is displayed.

3. Click **Delete HeatWave Cluster** to go ahead with the deletion.

4.3 Viewing HeatWave Cluster Details

To view HeatWave Cluster Details:

1. In the HeatWave Console, select the **HeatWave Clusters** tab.
2. In the list of HeatWave Clusters, find the HeatWave Cluster you want to view details for, and do one of the following:
   - Click on the row of the HeatWave Cluster to highlight it. The HeatWave Cluster details appear below the list of HeatWave Clusters. Click on the arrow icon to display or hide the details.
   - Click the name of the HeatWave Cluster to open the **HeatWave Cluster Details** page for the selected HeatWave Cluster.

For descriptions of HeatWave Cluster details, see HeatWave Cluster Details.

- HeatWave Cluster Details

4.3.1 HeatWave Cluster Details

The **HeatWave Cluster Details** page is divided into the following sections:

Table 4-2 HeatWave Cluster Details Page

Name	Description
Summary	HeatWave Cluster summary details. See Table 4-3.
General Information	General HeatWave Cluster details. See Table 4-4.
MySQL DB System	MySQL DB System details. See Table 4-5.
HeatWave Nodes	HeatWave node details. See Table 4-6.
HeatWave Cluster Events	HeatWave Cluster event details. See Table 4-7.

Table 4-3 HeatWave Cluster Summary Details

Field	Description
State	The state of the HeatWave Cluster. • **CREATING**: Resources are being reserved for the HeatWave Cluster, and the cluster is being created. Creating can take several minutes. The system is not ready to use yet. • **ACTIVE**: The HeatWave Cluster was successfully created and is ready to be used. • **STARTING**: The HeatWave Cluster is being started. • **STOPPING**: The HeatWave Cluster is being stopped. • **RESTARTING**: The HeatWave Cluster is being restarted. • **INACTIVE**: The HeatWave Cluster is powered off. • **UPDATING**: The HeatWave Cluster is in the process of being updated. • **UPGRADING**: The HeatWave Cluster is in the process of being upgraded. • **DELETING**: The HeatWave Cluster is being deleted. • **DELETED**: The HeatWave Cluster has been deleted and is no longer available. • **FAILED**: An error condition prevented the creation or continued operation of the HeatWave Cluster. • **STARTING_ERROR**: An attempt to start the HeatWave Cluster failed. • **STOPPING_ERROR**: An attempt to stop the HeatWave Cluster failed. • **RESTARTING_ERROR**: An attempt to restart the HeatWave Cluster failed. • **UPDATING_ERROR**: An attempt to update the HeatWave Cluster failed. • **UPGRADING_ERROR**: An attempt to upgrade the HeatWave Cluster failed. • **DELETING_ERROR**: An attempt to delete the HeatWave Cluster failed.
Cluster Size	The number of HeatWave Cluster nodes.
Resource ID	The unique resource identifier assigned to the HeatWave Cluster when it is created.
Shape	The shape used for HeatWave Cluster nodes. See Supported Shapes.

Table 4-4 General Information

Field	Description
Description	A user-specified description of the HeatWave Cluster.
Created	The date and time the HeatWave Cluster was created.

Table 4-4 (Cont.) General Information

Field	Description
Last Updated	The date and time the HeatWave Cluster was last updated.

Table 4-5 MySQL DB System Details

Field	Description
Name	The name of the DB System that the HeatWave Cluster is associated with
State	The state of the DB System that the HeatWave Cluster is associated with. • **CREATING**: Resources are being reserved for the DB System, the system is booting, and the initial database is being created. Creating can take several minutes. The system is not ready to use yet. • **ACTIVE**: The DB System was successfully created and is ready to be used. • **STARTING**: The DB System is being started. • **STOPPING**: The DB System is being stopped. • **RESTARTING**: The DB System is being restarted. • **INACTIVE**: The DB System is powered off. • **UPDATING**: The DB System is in the process of being updated. • **UPGRADING**: The DB System is in the process of being upgraded. • **DELETING**: The DB System is being deleted. • **DELETED**: The DB System has been deleted and is no longer available. • **FAILED**: An error condition prevented the creation or continued operation of the DB System. • **STARTING_ERROR**: An attempt to start the DB System failed. • **STOPPING_ERROR**: An attempt to stop the DB System failed. • **RESTARTING_ERROR**: An attempt to restart the DB System failed. • **UPDATING_ERROR**: An attempt to update the DB System failed. • **UPGRADING_ERROR**: An attempt to upgrade the DB System failed. • **DELETING_ERROR**: An attempt to delete the DB System failed.
Description	The user-defined description of the DB System

Table 4-6 HeatWave node Details

Field	Description
Node ID	The HeatWave node ID
State	The state of the HeatWave node. • **CREATING**: Resources are being reserved for the HeatWave node, and the cluster is being created. Creating can take several minutes. The system is not ready to use yet. • **ACTIVE**: The HeatWave node was successfully created and is ready to be used. • **STARTING**: The HeatWave node is being started. • **STOPPING**: The HeatWave node is being stopped. • **RESTARTING**: The HeatWave node is being restarted. • **INACTIVE**: The HeatWave node is powered off. • **UPDATING**: The HeatWave node is in the process of being updated. • **UPGRADING**: The HeatWave node is in the process of being upgraded. • **DELETING**: The HeatWave node is being deleted. • **DELETED**: The HeatWave node has been deleted and is no longer available. • **FAILED**: An error condition prevented the creation or continued operation of the HeatWave node. • **STARTING_ERROR**: An attempt to start the HeatWave node failed. • **STOPPING_ERROR**: An attempt to stop the HeatWave node failed. • **RESTARTING_ERROR**: An attempt to restart the HeatWave node failed. • **UPDATING_ERROR**: An attempt to update the HeatWave node failed. • **UPGRADING_ERROR**: An attempt to upgrade the HeatWave node failed. • **DELETING_ERROR**: An attempt to delete the HeatWave node failed.

Table 4-7 HeatWave Cluster Event Details

Field	Description
Type	The event and its status.

Table 4-7 (Cont.) HeatWave Cluster Event Details

Field	Description
Severity	The severity level of the event, which can be one of • `INFO`: General information is provided. No special concerns • `WARNING`: Warning on an event that might impact the DB System's operation if no actions are taken. • `CRITICAL`: A critical event. An urgent action is required to avoid adverse impact on your DB System.
Message	The message body of the event report.
Created	The date and time the event report was created.

See Events for details on event reports.

4.4 HeatWave Cluster Failure and Recovery

HeatWave triggers the recovery process whenever the HeatWave cluster is in an unhealthy state.

The HeatWave cluster status is being monitored regularly and if the cluster does not get a response from the cluster node, the cluster becomes unhealthy due to a node failure and error recovery is triggered.

During the recovery process, HeatWave automatically attempts to bring the node online, and reload data that was previously loaded. HeatWave reloads data from the HeatWave Storage layer, which is created when you enable the HeatWave cluster for the first time. To facilitate recovery, data is persisted to AWS S3 when data is loaded into HeatWave and when data changes is propagated from the DB system to HeatWave. Loading data from AWS S3 is faster because the data does not need to be converted to the HeatWave storage format, as is required when loading data from the DB system.

When you unload a table, the data is removed from HeatWave, and in a background operation, it is removed from AWS S3 too.

5
Connecting to a DB System

Clients and applications interact with a HeatWave Cluster by connecting to the MySQL DB System. This chapter describes how to connect to the MySQL DB System from the HeatWave Console and from MySQL clients including MySQL Shell, MySQL Command-Line-Client, and MySQL Workbench. For connecting from applications, refer to the MySQL Connectors documentation. See MySQL Connectors.

5.1 Connecting from the Console

Connecting to a DB System from the HeatWave Console requires a Oracle Cloud Account user name and password for signing into the HeatWave Console, and a MySQL user account for connecting to the DB System.

Use the MySQL Administrator user specified when creating the DB System, see Creating a DB System, or use a MySQL user account created on the DB System using `CREATE USER` . Otherwise, ask the MySQL Administrator to create an account.

Use the **Workspaces** page to connect to a DB System, and then manage HeatWave Cluster data and use the **Query Editor** to run DB System and HeatWave queries. For more information, see Manage Data in HeatWave with Workspaces, and Running Queries.

To connect to a DB System from the HeatWave Console:

1. Sign-in to the HeatWave Console. For instructions, see Signing In.
2. Select the **Workspaces** tab in the HeatWave Console, and then click **Connect to MySQL DB System**.
3. Select a DB System from the drop-down menu.
4. Enter a MySQL user name and password for the DB System.
5. Click **Connect**.

To disconnect, click **Disconnect**.

5.2 Connecting from a Client

Connecting to a DB System from a MySQL client requires a MySQL user account on the MySQL DB System. You can use the MySQL Administrator user that you specified when creating the DB System (see Creating a DB System) or a MySQL user account created on the DB System using `CREATE USER` . If you are not the MySQL Administrator and you do not have a MySQL user account, have your MySQL Administrator create one for you.

You cannot connect from a MySQL client to a DB System using the Oracle Cloud Account user name and password used to access the HeatWave Console.

For MySQL client connections to the DB System, a public endpoint is exposed as a fully qualified domain name (the *host name* of the DB System). The host name is found on the **MySQL DB System Details** page. See Viewing DB System Details.

The HeatWave on AWS Administrator may have restricted access to your DB System to certain public-facing IPv4 client IP addresses or address ranges. Allowed client addresses are specified in CIDR format and are found on the **MySQL DB System Details** page. See Viewing DB System Details. To edit allowed client addresses, see Editing a DB System. Specifying IP addresses in CIDR format is discussed in Creating a DB System.

If you are connecting from a MySQL client that resides in a private subnet, you have the option of connecting to a DB System through a public Network Address Translation (NAT) gateway, which permits clients and applications in a private subnet to access services outside of the private subnet while preventing external services from initiating inbound connections. When establishing a NAT gateway, ensure that the elastic IP address of the NAT gateway is added as an *Allowed Client Address*, as described above. For example, if your NAT gateway elastic IP address is 1.2.3.4, edit your DB System to add `1.2.3.4/32` (the address in CIDR notation) to your DB System's **Allowed Client Addresses**. See Editing a DB System. For more information about NAT gateways, refer to NAT Gateways, in the *Amazon VPC User Guide*.

HeatWave on AWS supports TLSv1.2 and requires that all connections are encrypted. For added security, you can download a signed certificate bundle and enable host name identity verification. For more information, see Enabling Host Name Identity Verification.

To reduce network costs and avoid potential latency issues and bandwidth fluctuations, it is recommended that connecting clients reside in the same *Region* as the HeatWave on AWS instance. Latency and bandwidth fluctuations experienced by connections from outside the HeatWave on AWS *Region* are outside of the control HeatWave on AWS service managers. Connecting from the same *Availability Zone* is also recommended to avoid potential latency issues.

- Connecting with MySQL Shell
- Connecting with MySQL Command-Line Client
- Connecting with MySQL Workbench

5.2.1 Connecting with MySQL Shell

This topic describes how to connect to a MySQL DB System using MySQL Shell.

Prerequisites:

- A machine or compute instance with internet connectivity for connecting to the MySQL DB System.
- A MySQL Shell command-line utility installed on the machine or compute instance. For installation instructions, refer to Installing MySQL Shell.
- A MySQL user account on the MySQL DB System to connect with. You can use the MySQL Administrator user that you specified when creating the DB System or a MySQL user account created on the DB System using `CREATE USER` .
- A public-facing IP address for your machine or compute instance that is permitted to connect to the DB System. **Allowed Client Addresses** information is available on the **MySQL DB System Details** page. See Viewing DB System Details.

- The host name of the MySQL DB System as defined on the **MySQL DB System Details** page. See Viewing DB System Details.

To connect to a DB System:

1. Start MySQL Shell and connect to the MySQL DB System using the following command:

```
$> mysqlsh Username@HostNameOfMySQLDBSystem

Please provide the password for 'Username@HostNameOfMySQLDBSystem':
```

This command starts a global session. MySQL Shell attempts to connect to port 33060 by default and, if that port is not available, falls back to port 3306.

The connection is made and message similar to the following is displayed:

```
                                MySQL Shell 8.0.30

Copyright (c) 2016, 2022, Oracle and/or its affiliates.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates.
Other names may be trademarks of their respective owners.

Type '\help' or '\?' for help; '\quit' to exit.
Creating a session to 'Username@HostNameOfMySQLDBSystem'
Fetching schema names for autocompletion... Press ^C to stop.
Your MySQL connection id is 38 (X protocol)
Server version: 8.0.30-u1-cloud MySQL Enterprise - Cloud
No default schema selected; type \use <schema> to set one.
MySQL  HostNameOfMySQLDBSystem.dbsystem JS >
```

5.2.2 Connecting with MySQL Command-Line Client

This topic describes how to connect to a MySQL DB System using a MySQL Command-Line Client.

Prerequisites:

- A machine or compute instance with internet connectivity for connecting to the MySQL DB System.
- A MySQL Command-Line Client installed on the machine or compute instance.
- A MySQL user account on the MySQL DB System to connect with. You can use the MySQL Administrator user that you specified when creating the DB System or a MySQL user account created on the DB System using `CREATE USER` .
- A public-facing IP address for your machine or compute instance that is permitted to connect to the DB System. **Allowed Client Addresses** information is available on the **MySQL DB System Details** page. See Viewing DB System Details.
- The host name of the MySQL DB System as defined on the **MySQL DB System Details** page. See Viewing DB System Details.

To connect to a DB System:

1. Start MySQL client and connect to the MySQL DB System using the following command:

```
$> mysql --host HostNameOfMySQLDBSystem -u Username -p
```

The connection is made and a message similar to the following is displayed:

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 40
Server version: 8.0.30-u1-cloud MySQL Enterprise - Cloud

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql>
```

5.2.3 Connecting with MySQL Workbench

This topic describes how to connect to a MySQL DB System with MySQL Workbench.

Prerequisites:

- A machine or compute instance with internet connectivity for connecting to the MySQL DB System.
- A MySQL Workbench client installed on your machine or compute instance. For instructions, refer to the MySQL Workbench Reference Manual.
- A MySQL user account on the MySQL DB System to connect with. You can use the MySQL Administrator user that you specified when creating the DB System or a MySQL user account created on the DB System using `CREATE USER` .
- A public-facing IP address for your machine or compute instance that is permitted to connect to the DB System. **Allowed Client Addresses** information is available on the **MySQL DB System Details** page. See Viewing DB System Details.
- The host name of the MySQL DB System as defined on the **MySQL DB System Details** page. See Viewing DB System Details.

To connect to a DB System from MySQL Workbench:

1. In MySQL Workbench, select **Database** > **Connect to Database** to open the **Connect to Database** dialog.
2. Enter the connection details.
   - **Connection Method**: The connection method. Select **Standard (TCP/IP)**.
   - **Hostname**: The host name of the MySQL DB System
   - **Port**: The port address that the MySQL DB System is listening on. Use the default `3306` port.
   - **Username**: The user name of the MySQL user that will connect to the DB System

- **Password**: The password of the MySQL user
- **Default Schema**: Optionally, specify a default schema. Leave blank to select it later.

3. Click **OK** to establish the connection.

5.3 MySQL Connectors

You can use MySQL Connectors to connect your application to a DB System. The following list provides links to the MySQL Connectors documentation:

The requirements outlined for connecting from a MySQL client also apply when connecting from an application. See Connecting from a Client.

- Connector/J
- Connector/Python
- Connector/C++
- Connector/NET
- Connector/Node.js

5.4 Enabling Host Name Identity Verification

HeatWave on AWS supports TLSv1.2 and requires that all MySQL client and application connections are encrypted. For added security, you can download a signed certificate bundle and enable host name identity verification for your connecting clients and applications.

When a DB System is provisioned, a TLS certificate is installed on the MySQL Server. The certificate, which defines the DB System host name as the `Common Name`, is signed by a regional Certificate Authority (CA). When a client connects to the DB System with host name identity verification enabled and a CA certificate matching the one used by the server, the server and client place their trust in the same CA certificate and the client verifies that the host to which it connected is the one intended.

To obtain a CA certificate file, you must download a HeatWave on AWS certificate bundle. Store the certificate bundle in a secure location that is accessible to the client or application. The bundle is a regional CA certificate file in PEM format. You can download the regional certificate bundle for the supported regions:

- AWS US East (N. Virginia) Region (`us-east-1`): https://cloud.mysql.com/us-east-1/aws-us-east-1-cabundle.pem
- Germany Central (Frankfurt) Region (`eu-central-1`): https://cloud.mysql.com/eu-central-1/aws-eu-central-1-cabundle.pem
- India West (Mumbai) Region (`ap-south-1`): https://cloud.mysql.com/ap-south-1/aws-ap-south-1-cabundle.pem
- Japan East (Tokyo) Region (`ap-northeast-1`): https://cloud.mysql.com/ap-northeast-1/aws-ap-northeast-1-cabundle.pem
- UK South (London) Region (`eu-west-2`): https://cloud.mysql.com/eu-west-2/aws-eu-west-2-cabundle.pem

> **Note:**
>
> HeatWave on AWS is currently supported in the regions mentioned in Region Availability. Certificate bundles for other regions will be made available as support for HeatWave on AWS is extended to other regions.

Alternatively, you can download the regional certificate bundle from the Console using cURL as shown below:

```
$> curl -o aws-us-east-1-cabundle.pem \
https://cloud.mysql.com/us-east-1/aws-us-east-1-cabundle.pem


$> curl -o aws-ap-south-1-cabundle.pem \
https://cloud.mysql.com/ap-south-1/aws-ap-south-1-cabundle.pem


$> curl -o aws-eu-central-1-cabundle.pem \
https://cloud.mysql.com/eu-central-1/aws-eu-central-1-cabundle.pem


$> curl -o aws-ap-northeast-1-cabundle.pem \
https://cloud.mysql.com/ap-northeast-1/aws-ap-northeast-1-cabundle.pem


$> curl -o aws-eu-west-2-cabundle.pem \
https://cloud.mysql.com/eu-west-2/aws-eu-west-2-cabundle.pem
```

> **Note:**
>
> It is recommend that you update your HeatWave on AWS certificate bundle quarterly to ensure that you always have the latest version. Issues connecting with `--ssl-mode` may indicate that your certificate bundle is outdated.

To establish an encrypted connection, launch the MySQL client with the `--ssl-ca` and `--ssl-mode` options; for example:

MySQL Shell:

```
$> mysqlsh --host=HostNameOfMySQLDBSystem \
           --user=user1 \
           --password \
           --port=3306 \
           --ssl-mode=VERIFY_IDENTITY \
           --ssl-ca=aws-us-east-1-cabundle.pem
```

MySQL Command-Line Client:

```
$> mysql --host=HostNameOfMySQLDBSystem \
         --user=user1 \
         --password \
         --protocol=TCP \
         --port=3306 \
         --ssl-mode=VERIFY_IDENTITY \
         --ssl-ca=aws-us-east-1-cabundle.pem
```

where:

- `--host` specifies the host name of the DB System. The host name is found on the **MySQL DB System Details** page. See Viewing DB System Details.
- `--user` specifies the user name of the MySQL account to use for connecting to the server.
- `-p` specifies the password of the MySQL account used for connecting to the server. The password value is optional. If not given as in the examples above, `mysql` prompts for one.
- `--protocol` specifies transport protocol to use for connecting to the server. This option is not applicable to MySQL Shell.
- `--ssl-mode` is the security state of the connection to server. The `VERIFY_IDENTITY` mode ensures that an encrypted connection is established, that the TLS certificate is verified against the configured CA certificate, and that the host name identity is verified by checking the host name the client uses for connecting to the server against the identity in the certificate that the server sends to the client.
- `--ssl-ca` specifies the fully qualified path to the CA certificate file.

For more information about `ssl-*` connection options, see Command Options for Connecting to the Server.

5.5 PrivateLink

A PrivateLink enables you to connect directly to a DB System in HeatWave on AWS using private IP addresses. A PrivateLink exists between the HeatWave on AWS Virtual Private Cloud (VPC) and your VPC. You can use the PrivateLink to query or manage the DB System.

Do the following to use a PrivateLink:

1. Create a PrivateLink in HeatWave on AWS and associate it with a DB System. See Creating a PrivateLink.
2. Create an endpoint in your AWS account. See Creating an Endpoint.
3. You can now access the DB System over a PrivateLink. See Connecting to a DB System With a PrivateLink.

- Creating a PrivateLink
- Updating Authorized Principals
- Configuring IAM Policies for Endpoints
- Creating an Endpoint
- Connecting to a DB System With a PrivateLink

- Managing a PrivateLink
- PrivateLink Limitations

5.5.1 Creating a PrivateLink

Use the HeatWave Console to create a PrivateLink to connect to a DB System in HeatWave on AWS using private IP addresses.

This task requires the following:

- A DB System in the `Active` state.
- ARNs of authorized principals.

Do the following to create a PrivateLink:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **PrivateLink** tab, click **Create PrivateLink.**
3. Enter the following:
   - **Basic information**:
     - **Display name**: Specify a display name for the PrivateLink or use the generated default name.
     - **Description**: (Optional) Specify a description for the PrivateLink.
   - **Configure PrivateLink**:
     - **PrivateLink type**: Select the type of PrivateLink:
       * **Query**: Provide connectivity from a customer application to a HeatWave on AWS DB System using private IP addresses.
     - **ARNs of Authorized Principals**: Authorize principal ARNs to create connections to the PrivateLink. You can specify more than one ARN delimited by space. You can specify either of the following:
       * (Recommended) Entire AWS accounts in the following format:

         ```
         arn:aws:iam::<ACCOUNT_ID>:root
         ```

       * Specific principals in the following format:

         ```
         arn:aws:iam::<ACCOUNT_ID>:user/<user_id>
         ```

         ```
         arn:aws:iam::<ACCOUNT_ID>:role/<role_id>
         ```

         See Amazon Resource Names (ARNs).

         For enhanced security, authorize a specific set of principals. In this case, the authorization to create a PrivateLink is checked twice: first inside the AWS account requesting the new endpoint, and then in HeatWave on AWS to ensure that the entity requesting the endpoint is in the set of authorized principals. Once you have updated the authorized principals list, configure IAM policies in your AWS account to grant principals the permissions to create and delete VPC endpoints. See Configuring IAM Policies for Endpoints.

- **Target DB System**: Select the DB System with which you want to associate the PrivateLink.

4. Click **Create**.

You can see the details of the PrivateLink including a new **Hostname** and **Service name**. Note the **Service name** because you will need it to create an endpoint.

Related Topics

- Creating an Endpoint

5.5.2 Updating Authorized Principals

Use the HeatWave Console to update the authorized principals of a PrivateLink.

This task requires the following:

- A PrivateLink in the `Active` state.

Do the following to update the authorized principals of a PrivateLink:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **PrivateLink** tab, in the list of PrivateLinks, find the PrivateLink for which you want to update the authorized principals, and do one of the following:
   - Click the row of the PrivateLink to highlight it, and click **Update Authorized Principals**.
   - Click the name of the PrivateLink to open the **PrivateLink Details** page, and click **Update Authorized Principals**.
3. In the **Configure PrivateLink** section, enter the following:
   - **ARNs of Authorized Principals**: Authorize principal ARNs to create connections to the PrivateLink. You can specify more than one ARN delimited by space. You can specify either of the following:
     – (Recommended) Entire AWS accounts in the following format:

       ```
       arn:aws:iam::<ACCOUNT_ID>:root
       ```

     – Specific principals in the following format:

       ```
       arn:aws:iam::<ACCOUNT_ID>:user/<user_id>
       ```

       ```
       arn:aws:iam::<ACCOUNT_ID>:role/<role_id>
       ```

       For enhanced security, authorize a specific set of principals. In this case, the authorization to create a PrivateLink is checked twice: first inside the AWS account requesting the new endpoint, and then in HeatWave on AWS to ensure that the entity requesting the endpoint is in the set of authorized principals.
4. Click **Save**.

After you have updated the authorized principals in HeatWave on AWS, configure IAM policies in your AWS account to grant specific principals the permissions to create and delete VPC endpoints. See Configuring IAM Policies for Endpoints.

5.5.3 Configuring IAM Policies for Endpoints

After you have updated the authorized principals in HeatWave on AWS, configure IAM policies in your AWS account to grant principals the permissions to create and delete VPC endpoints. See Principal.

If you are already using AWS-managed IAM policies for permission management, check that your current AWS-managed IAM policies grant principals the permission to create and delete VPC endpoints. If not, add the appropriate managed policies to the principals you wish to authorize for VPC endpoint management. The following policies enable you to create and delete endpoints:

- `NetworkAdministrator`
- `AmazonVPCFullAccess` and `AmazonRoute53FullAccess`

These AWS managed policies grant broader permissions than those strictly required for PrivateLink. For enhanced security, administer your AWS account to grant least privileges to your IAM principals. For AWS guidance on how to grant least privilege, see Grant least privilege .

If you are using least-privilege permissions in your AWS account, add policies enabling specific principals to create and delete VPC endpoints for a given PrivateLink. See Control the service names that can be specified for VPC endpoint services.

To limit access to PrivateLink in policies for least privilege, you can either specify the service name of the PrivateLink or the account ID as shown below. See Viewing PrivateLink Details.

```
"Condition": {
              "StringEquals": {
                 "ec2:VpceServiceName": "<privatelink-service-name>"
          }
    }


"Condition": {
             "StringEquals": {
                 "ec2:VpceServiceOwner": "612981981079"
          }
    }
```

5.5.4 Creating an Endpoint

Use the AWS Management Console to create an endpoint in the same region and availability zone as the DB System.

This task requires the following:

- Access to AWS Management Console.
- The service name of the PrivateLink. See Viewing PrivateLink Details.
- Properly configured policies to create an endpoint. See Configuring IAM Policies for Endpoints.
- A running virtual private cloud with subnets in the same availability zone as the DB System. See Creating a VPC, and Viewing DB System Details.

Do the following to create an endpoint:

1. Open the AWS Management Console and sign in with your credentials.
2. Switch to the same region as the DB System.
3. In the AWS Management Console home page, click **Services**, click **Networking & Content Delivery**, and then click **VPC**.
4. In the navigation pane of the Console, under **Virtual private cloud**, click **Endpoints**, and then click **Create endpoint**.
5. Enter the following:
   a. **Endpoint settings**:
      i. **Name tag**: (Optional) Specify a name for the endpoint.
      ii. **Service category**: Select **Other endpoint services**.
   b. **Service settings**:
      i. **Service name**: Specify the service name of the PrivateLink. See Viewing PrivateLink Details. Ensure that you create the endpoint in the same region as the PrivateLink.
      ii. Click **Verify service**.
      If the permissions are configured correctly in the PrivateLink, the service name is verified correctly. If service verification is unsuccessful, ensure that the authorized principals field in the PrivateLink is correct, and that your IAM permissions are configured accordingly. See Updating Authorized Principals.
   c. **VPC**:
      i. **VPC**: Select the VPC in which to create the endpoint.
      ii. Click **Additional settings**.
      iii. **Enable DNS name**: It is recommended to check this box. Checking this box configures the VPC to resolve the hostname of the PrivateLink to the private IP address of the endpoint. If you leave this box unchecked, you cannot connect to the PrivateLink using its hostname automatically, and you need to configure the DNS of the VPC manually.
   d. **Subnet**: Select the subnet in which you wish to create the endpoint.
   e. **Security groups**: Select the appropriate security groups to associate with the endpoint. The security groups must allow inbound traffic from your applications.
6. Click **Create endpoint**.

5.5.5 Connecting to a DB System With a PrivateLink

Connect to a DB System in HeatWave on AWS with a PrivateLink using standard MySQL clients and connectors.

This task requires the following:

- A DB System in HeatWave on AWS.
- A correctly configured PrivateLink. See Creating a PrivateLink.
- An EC2 compute instance in the same VPC as the private endpoint.
- HeatWave Certificate Authority certificate. See Enabling Host Name Identity Verification.

Do the following to connect to a DB System with a PrivateLink:

1. Login to the EC2 instance. See Connect to your Linux instance.
2. Use a MySQL client to access the DB System.

```
mysql -h <hostname> -u <username> -p <password> --ssl-mode VERIFY_IDENTITY
--ssl-ca
      <ca-file>
```

   - `<hostname>`: Specify the hostname of the PrivateLink.
   - `<username>`: Specify the username of the administrator.
   - `<password>`: Specify the password of the administrator.
   - `<ca-file>`: Specify the name of the HeatWave Certificate Authority certificate.

You are now connected to a DB System in HeatWave on AWS over a PrivateLink.

Related Topics

- Creating an Endpoint

5.5.6 Managing a PrivateLink

Use the HeatWave Console to edit the name or description of a PrivateLink, update its authorized principals, delete the PrivateLink, or view its details.

- Editing a PrivateLink
- Deleting a PrivateLink
- Viewing PrivateLink Details

5.5.6.1 Editing a PrivateLink

Use the HeatWave Console to edit the name or description of a PrivateLink.

This task requires the following:

- A PrivateLink in the `Active` state.

Do the following to edit a PrivateLink:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **PrivateLink** tab, in the list of PrivateLinks, find the PrivateLink to edit, and do one of the following:
   - Click the row of the PrivateLink to highlight it, and click **Edit PrivateLink**.
   - Click the name of the PrivateLink to open the **PrivateLink Details** page, and click **Edit PrivateLink**.
3. Enter the following:
   - **Basic information**:
     - **Display name**: Edit the display name of the **PrivateLink**.
     - **Description**: (Optional) Edit the description for the PrivateLink.
4. Click **Save**.

5.5.6.2 Deleting a PrivateLink

Use the HeatWave Console to delete a PrivateLink . If you delete a DB System with a PrivateLink, the PrivateLink is deleted automatically.

This task requires the following:

- A PrivateLink in the `Active` state.

Do the following to delete a PrivateLink.

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **PrivateLink** tab, in the list of PrivateLinks, find the PrivateLink to edit, and do one of the following:
   - Click the row of the PrivateLink to highlight it, and click **Delete**.
   - Click the name of the PrivateLink to open the **PrivateLink Details** page, and click **Delete**.
3. In the **Delete PrivateLink** dialog, click **Delete PrivateLink**.

Also, you can delete any resources you created in the AWS account, such as the VPC endpoint.

5.5.6.3 Viewing PrivateLink Details

Use the HeatWave Console to view the details of a PrivateLink.

This task requires the following:

- A PrivateLink in the `Active` state.

Do the following to view the details of a PrivateLink:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. On the **PrivateLink** tab, in the list of PrivateLinks, find the PrivateLink whose details you want to view, and do one of the following:
   - Click the row of the PrivateLink to highlight it. The details of the PrivateLink appear below the list of PrivateLinks.
   - Click the name of the PrivateLink to open the **PrivateLink Details** page.

For descriptions of PrivateLink details, see PrivateLink Details.

- PrivateLink Details

5.5.6.3.1 PrivateLink Details

The **PrivateLink Details** page is divided into the following sections:

Table 5-1 Summary

Field	Description
State	The state of the PrivateLink: • **CREATING**: The PrivateLink is being created and is not ready yet to be used • **ACTIVE**: The PrivateLink is successfully created and is ready to be used. • **UPDATING**: The PrivateLink is in the process of being updated. For example, adding a new principal, or changing metadata. • **UPDATING_ERROR**: The PrivateLink encountered an error during `UPDATE`. It continues to function with the parameters before the update was initiated. • **DELETING**: The PrivateLink is being deleted. • **DELETING_ERROR**: The PrivateLink encountered an error during `DELETE`. It may continue to function with its current set of parameters. • **DELETED**: The PrivateLink has been deleted and is no longer available. • **FAILED**: An error condition prevented creating or updating the PrivateLink. • **NEEDS_ATTENTION**: The PrivateLink needs a user action to restore connectivity.
Resource ID	The unique resource identifier assigned to the PrivateLink.
Link type	The type of PrivateLink.

Table 5-2 General Information

Field	Description
Description	The user-defined description of the PrivateLink.
Created	The date and time the PrivateLink was created.
Last Updated	The date and time the PrivateLink was last updated.
Port	The port on which the PrivateLink listens for traffic using the MySQL protocol. The default value is 3306.
Port X	The port on which the PrivateLink listens for clients using the MySQL X Protocol, such as MySQL Shell. The default value is 33060.

Table 5-3 MySQL DB System

Field	Description
Name	The name of the associated DB System.
State	The state of the DB System. See MySQL DB System Details.
Description	The user-defined description of the DB System.

Table 5-4 Link Details

Field	Description
Hostname	The host name of the PrivateLink. The host name is a fully qualified domain name (FQDN).
Service name	The service name that you need while creating an endpoint. See Creating an Endpoint.
Authorized principals	The principal ARNs that are authorized to create VPC endpoints for this PrivateLink.

5.5.7 PrivateLink Limitations

- You can create one PrivateLink per DB System.
- You cannot change the value of the configuration variable, `max_connect_errors`. Changing this configuration variable results in a loss of connectivity on the PrivateLink.
- When you connect to a DB System over PrivateLink, you cannot specify public or private IP addresses in the `host_name` field of the MySQL account names. You must use wildcard (%) in the `host_name` field. See Specifying Account Names.

6
Importing Data

You can import data to a DB System in HeatWave on AWS. If your data is present in a MySQL instance running on-premises, in other cloud vendors as managed or unmanaged services, or another HeatWave on AWS instance, you have to export the data first. See Exporting Data.

You can import data using either of the following:

- **Data import feature**: Use the data import feature in the Console to import MySQL Shell dump and text files from an Amazon S3 bucket. See Data Import Feature.
- **Bulk ingest feature**: Connect to the DB System and use the bulk ingest feature to import text files such as CSV and TSV from an Amazon S3 bucket. This method is the fastest and most efficient in terms of computing and memory consumption. This method is ideal for importing large tables. See Bulk Ingest Feature.
- **Dump loading utility**: Use the dump loading utility of MySQL Shell to import a MySQL Shell dump manually. See Importing Data Using the Dump Loading Utility.

After importing data to a DB System in HeatWave on AWS, load data into HeatWave. See Loading or Unloading Data into HeatWave Cluster.

- Exporting Data
- Importing Data

6.1 Exporting Data

Use the dump utility of MySQL Shell to export a logical dump of the data from a MySQL instance. Connect to the MySQL instance containing the data you want to dump using MySQL Shell. You can dump all schemas in the instance, a selected schema, or selected tables and views.

Use either of the following dump utility:

- `util.dumpInstance(outputUrl[, options])`: MySQL instance export utility that exports all compatible schemas to an Amazon S3 bucket, Object Storage bucket, or to local files. By default, this utility exports users, events, routines, and triggers. See Dump Utilities.
- `util.dumpSchemas(schemas, outputUrl[, options])`: MySQL schema export utility that exports selected schemas to an Amazon S3 bucket, Object Storage bucket, or to local files.
- `util.dumpTables(schema, tables, outputUrl[, options])`: MySQL table export utility that exports selected tables of a schema to an Amazon S3 bucket, Object Storage bucket, or to local files.

The source and destination MySQL instances can be in different MySQL versions.

The MySQL Shell dump utility performs compatibility checks and transformations to ensure the data can be later successfully imported and the potential data import issues can be identified and fixed as early as possible. See MySQL Server Compatibility.

- About MySQL Shell

- MySQL Server Compatibility
  HeatWave on AWS has several security-related restrictions that are not present in an on-premise instance of MySQL. To make it easier to load existing databases into the Service, the dump commands in MySQL Shell can detect potential issues and, in some cases, automatically adjust your schema definition to be compliant.
- Exporting Data Using MySQL Shell

6.1.1 About MySQL Shell

MySQL Shell dump and load utilities are built for use with MySQL DB Systems. To get the best functionality, always use the most recent version of MySQL Shell that is available to you.

The MySQL Server version used by HeatWave on AWS is fully supported by MySQL Shell. The minimum supported source MySQL Server versions supported by MySQL Shell are:

- MySQL 8.0.11
- MySQL 5.7.9

MySQL Shell provides the following utilities:

- `dumpInstance()`: MySQL instance export utility that dumps all compatible schemas to an Amazon S3 bucket, OCI Object Storage, or local files. By default, this utility dumps users, events, routines, and triggers. See MySQL Shell Instance and Schema Dump Utilities.
- `dumpSchemas()`: A schema export utility that dumps selected schemas to an Amazon S3 bucket, OCI Object Storage, or local files. See MySQL Shell Instance and Schema Dump Utilities.
- `dumpTables()`: A table export utility that dumps selected tables of a schema to an Amazon S3 bucket, OCI Object Storage, or local files.
- `loadDump()`: An import utility that imports schemas to a DB System. See MySQL Shell Dump Loading Utility. To import a schema to a DB System, MySQL Shell must be installed on a machine with access to the DB System. See Connecting with MySQL Shell .

MySQL Shell dump files are exported as DDL files specifying the schema structure and tab-separated value (`.tsv`) files containing the data. The `.tsv` files are compressed using `zstd`, by default, but `gzip` is also available as an option. You can also choose no compression.

To improve performance, large tables are chunked by default. The default chunk size is 32MB. Chunking can be disabled, but this is not recommended for large databases. During import, the chunks can be imported by parallel threads, which can greatly improve import performance.

For more information about MySQL Shell, refer to the MySQL Shell User Guide.

6.1.2 MySQL Server Compatibility

HeatWave on AWS has several security-related restrictions that are not present in an on-premise instance of MySQL. To make it easier to load existing databases into the Service, the dump commands in MySQL Shell can detect potential issues and, in some cases, automatically adjust your schema definition to be compliant.

The `ocimds` option, when set to true, performs compatibility checks on the schemas for these issues and aborts the dump if any are found, while producing a detailed list of those issues and suggests additional steps to correct those issues. `loadDump` command only allows import of dumps created with the `ocimds` option enabled.

Some issues found by the `ocimds` option may require you to manually edit your schema before it can be loaded into the HeatWave on AWS. The MySQL Shell `compatibility` option can be used to automatically modify the dumped schemas, resolving some of these compatibility issues. You can pass one or more modifiers to the MySQL Shell `compatibility` option in a comma-separated list.

The MySQL Shell `compatibility` option applies the specified requirements for compatibility with HeatWave on AWS for all tables in the dump output, altering the dump files as necessary. From MySQL Shell 8.0.23, this option is available for all MySQL Shell utilities, and before that release, it is only available for the instance dump utility and schema dump utility.

MySQL Shell `compatibility` option modifiers include:

`force_innodb`
HeatWave on AWS supports the InnoDB storage engine, only. This option modifies the `ENGINE=` clause of `CREATE TABLE` statements to specify `INNODB` for tables that do not already use the `InnoDB` storage engine.

`skip_invalid_accounts`
Skips user accounts created with external authentication plugins that are not supported in HeatWave on AWS. From MySQL Shell 8.0.26, this option also removes user accounts that do not have passwords set, except where an account with no password is identified as a role, in which case it is dumped using the `CREATE ROLE` statement.

`strip_definers`
Removes the `DEFINER=account` clause from views, routines, events, and triggers. HeatWave on AWS requires special privileges to create these objects with a definer other than the user loading the schema. By removing the `DEFINER` clause, these objects will be created with that default definer. Views and routines have their `SQL SECURITY` clause changed from `DEFINER` to `INVOKER`. This ensures that the access permissions of the account querying or calling these objects are applied, instead of the user that created them. If your database security model requires views and routines have more privileges than their invoker, you must manually modify the schema before loading it. For more information, see The DEFINER Attribute and The SQL SECURITY Characteristic.

`strip_restricted_grants`
Certain privileges are restricted in HeatWave on AWS, such as `RELOAD` , `FILE` , `SUPER` , `BINLOG_ADMIN` , and `SET_USER_ID` . It is not possible to create users granting these privileges. This option removes those privileges from dumped `GRANT` statements.

`strip_tablespaces`
HeatWave on AWS has some restrictions on tablespaces. This modifier removes the `TABLESPACE` clause from `GRANT` statements, so all tables are created in their default tablespaces.

Additionally, `DATA DIRECTORY`, `INDEX DIRECTORY`, and `ENCRYPTION` options in `CREATE TABLE` statements are always commented out in DDL scripts if the MySQL Shell `ocimds` option is enabled.

> **Note:**
>
> If you intend to export data from an older version of MySQL, you should run the MySQL Shell Upgrade Checker Utility to generate a report of all potential issues with your migration. For more information, see MySQL Shell User Guide - Upgrade Checker Utility .

6.1.3 Exporting Data Using MySQL Shell

Use MySQL Shell to export data from a source MySQL Server to an Amazon S3 buket.

This task requires the following:

- The MySQL Shell command-line utility. The commands in this task use the JS (JavaScript) execution mode of MySQL Shell. For installation instructions, refer to Installing MySQL Shell.

  > **Note:**
  >
  > Exports created by MySQL Shell 8.0.27, or higher, cannot be imported by earlier versions of MySQL Shell. Using the latest version of MySQL Shell is always recommended.

- You have run the `dumpSchema` command with the `dryRun` and `ocimds` parameters set to true:

  ```
  util.dumpSchemas("<Schema>", {s3bucketName: "<BucketName>", dryRun: true,
  ocimds: true})
  ```

  This performs a test run of the export checking for compatibility issues. The compatibility issues and remediation steps are listed in the output. See MySQL Shell Instance and Schema Dump Utilities.

To export a schema from a source MySQL Server:

1. Start a MySQL Shell session and connect to the source MySQL Server:

   ```
   mysqlsh <user>@<HostNameOfSourceMySQLServer>
   ```

   For information about different MySQL Shell session and connection options, refer to the MySQL Shell User Guide.

2. Export schema to an Amazon S3 bucket:

   ```
     util.dumpInstance("<Schema>",{s3bucketName: "<BucketName>", threads:
   <ThreadSize>,
           compatibility: ["strip_restricted_grants", "strip_definers",
   "ignore_missing_pks"]})
   ```

   - `<Schema>`: The name of the schema being exported. For example, `tpch`.
   - `<BucketName>`: The Amazon S3 bucket to which the dump is to be written.
   - `<ThreadSize>`: The number of processing threads to use for this task. The default is 4. For best performance, it is recommended to set this parameter to twice the number of CPUs used by the target DB System.
   - `compatibility`: Specify compatibility modifiers that modify the exported data for compatibility with HeatWave on AWS. See MySQL Server Compatibility.

   See Options for S3-compatible Services.

6.2 Importing Data

Use either of the following features to import data from an Amazon S3 bucket:

- **Data import**: Use the data import feature in the Console to import MySQL Shell dump and text files from Amazon S3 bucket. See Data Import Feature.
- **Bulk ingest**: Connect to the DB System and use the bulk ingest feature to import text files such as CSV and TSV from Amazon S3 bucket. This method is the fastest and most efficient in terms of computing and memory consumption. This method is suitable for importing large tables. See Bulk Ingest Feature, and Connecting to a DB System.
- **Dump loading utility**: Use the dump loading utility of MySQL Shell to import a MySQL Shell dump manually. See Importing Data Using the Dump Loading Utility.

6.2.1 Data Import Feature

Use the data import feature in HeatWave Console to import data in a variety of formats such as MySQL dump files and text files from an Amazon S3 bucket to a DB System in HeatWave on AWS in the same region.

6.2.1.1 Importing Sample Database

Oracle provides on AWS sample databases for you to test out HeatWave on AWS. Use the **Import sample data** feature in HeatWave Console to import the sample database into a DB System in any region by just a few clicks.
Currently, the following sample databases are available from Oracle.

- airportdb. See Running AirportDB Queries and Additional AirportDB Queries.
- TPCH. You have to select a size for your TPCH sample database—you are suggested to choose a value that matches the expected workload on your DB system. Please also pay attention to the **Disclaimer**. See Running tpch Queries and Sample TPCH queries.

This task requires the following:

- Access to HeatWave Console.
- An established connection from the HeatWave Console to the DB System into which you want to import data.

Follow these steps to import the sample database:

1. In the HeatWave Console, click the **Workspaces** tab, and then click the **Data Imports** tab.
2. Click **Import Data**.

3. In the **Import data into DB System** pane, under **Source**, select **Import sample data**. A list of sample databases appear, together with information for each of them like the name of the destination schema and the size.
4. Select a sample database. You might have to make additional selections, depending on the sample database you choose.
5. Click the **Import** button. The data import operation begins.

The **Data Imports** tab is shown, on which you can view details on your import operation.

6.2.1.2 Importing Data Using the Data Import Feature

Use the data import feature in the HeatWave Console to import data from an Amazon S3 bucket to a DB System in the same region.

This task requires the following:

- Data in an Amazon S3 bucket that you want to import. If your data is present in a MySQL instance running on-premises, in other cloud vendors as managed or unmanaged services, or another HeatWave on AWS instance, you have to export the data into an Amazon S3 bucket first.
- An IAM policy to access the Amazon S3 bucket that contains the data you want to import. See Creating an IAM Policy to Access an Amazon S3 Bucket.
- An IAM role ARN that grants the DB System access to the Amazon S3 bucket that contains the data you want to import. See Creating an IAM Role to Access an Amazon S3 Bucket.
- An established connection from the HeatWave Console to the DB System into which you want to import data.

Follow these steps to import the data:

1. In the HeatWave Console, configure the DB System into which you want to import data with the IAM role ARN. See Editing a DB System.
2. Click the **Workspaces** tab, and then click the **Data Imports** tab.
3. Click the **Import Data** button.
4. In the **Import data into DB System** pane, enter the following details:
   - **Basic information**:
     - **Display name**: Specify a name for the data import operation. By default, a name is generated for you in the format of `dataImportYYYYMMDDHHMMSS`.
     - **Description**: (Optional) Specify a description for the data import operation.
   - **Source**:
     - Select **Bring your own data** (unless you want to **Import sample data** instead).
     - **S3 URI**: Specify the Amazon S3 URI. The Amazon S3 bucket and DB System must be in the same region.

> **Note:**
>
> You are responsible for managing the Amazon S3 bucket. Once you finish the data import, it is recommended that you review the Amazon S3 bucket's access permissions and remove access by the DB System if necessary.

- **Authentication method**: Select either one of the following authentication methods by clicking its radio button:
  - **IAM role**: Use the displayed data import role ARN, which was specified while creating or editing the DB System. See Creating a DB System and Editing a DB System.
  - **User access key**: Enter the **Access key ID** and **Secret access key**. See Managing Access Keys.
- **File type**: Select either of the following file types:
  - **MySQL dump files**: Specify the **File parsing settings**:
    - **Character set**: Enter the character set of the import data. The default character set is `utf8mb4`. See Character Sets and Collations in MySQL.
    - **Update GTID set**: For data that contain GTIDs, to enable GTID-based replication, apply the `gtid_executed` GTID set from the source MySQL instance, as recorded in the dump metadata, to the `gtid_purged` GTID set on the target MySQL instance using one of the following options:
      - **OFF**: (Default) Do not update the GTID set. This is also the option for data that do not contain GTIDs.
      - **APPEND**: Append the `gtid_executed` GTID set from the source MySQL instance to the `gtid_purged` GTID set on the target MySQL instance.
      - **REPLACE**: Replace the `gtid_purged` GTID set on the target MySQL instance with the `gtid_executed` GTID set from the source MySQL instance.

      See the description for `gtid_purged` for more information.
  - **Text files**:
    - **Names of data files in S3 bucket**: Specify the names of data files in the Amazon S3 bucket. Apart from text files, you can specify text files in compressed formats such as gzip (`.gz`) and zstd (`.zst`). You can specify ranges of files using wildcard pattern matching. To match any single character, use `?`, and to match any sequence of characters, use `*`. For example, `data_c?`, and `/backup/replica/2021/*.tsv`.
    - **File parsing settings**:
      - **Character set**: Enter the character set to import data to the target MySQL instance. It must correspond to the character set given in the dump metadata that was used when the MySQL dump was created by MySQL Shell instance dump utility, schema dump utility, or table dump utility. The character set must be permitted by the `character_set_client` system variable and supported by the MySQL instance. The default character set is `utf8mb4`. See Character Sets and Collations in MySQL.

* **Dialect**: Select the dialect of the imported data file. You can select **CSV (Unix)**, **CSV**, or **TSV**. The default dialect is **CSV (Unix)**. The dialect selects the default values of the following parsing settings: **Fields terminated by**, **Fields enclosed by**, **Fields escaped by** and **Fields optionally enclosed**. To view the default settings per dialect, see Dialect settings.
* **Skip rows**: Specify the number of rows that is skipped from the beginning of the imported data file or, in the case of multiple import files, at the beginning of every file included in the file list. By default, no rows are skipped.
* **Lines terminated by**: Specify one or more characters (or an empty string) with which each of the lines are terminated in the imported data file. For example, `\r\n`. The default is as per the specified dialect.
* **Fields terminated by**: Specify one or more characters (or an empty string) with which each of the fields are terminated in the imported data file. For example, `\t`. The default is as per the specified dialect.
* **Fields enclosed by**: Specify a single character (or an empty string) with which the utility encloses each of the fields in the imported data file. For example, `"`. The default is as per the specified dialect.
* **Fields escaped by**: Specify the character that is to begin escape sequences in the imported data file. For example, `\`. The default is as per the specified dialect.
* **Fields optionally enclosed**: Select the option to enclose a field only if it has a string data type such as `CHAR`, `BINARY`, `TEXT`, or `ENUM`, and deselect the option to enclose all of the fields in the imported data file.

* **Destination**: Select the **Schema** and the **Table** in the DB System to which you want to import data.

– Click **Import**. The data import operation begins.

The **Data Imports** tab is shown, on which you can view details on your import operation..

6.2.1.3 Viewing Data Import Details

To view data import details:

1. In the HeatWave Console, select the **Workspaces** tab.
2. Select the **Data Imports** tab. In the list of data imports, select the import for which you want to view the details. The **Data Import Details** pane shows its information.
3. If your selected import is still in progress and you want to cancel it, click the **Cancel** button on top of the import list.

- Data Import Details

6.2.1.3.1 Data Import Details

Use the Console to view the **Data Import Details**. For more information on the details being displayed, see Importing Data Using the Data Import Feature.

Table 6-1 Data Import Details Pane

Name	Description
Summary	Data import summary. See Table 6-2.
Source	Data import source. See Table 6-3.
Destination	Data import destination. See Table 6-4.
Messages	Data import messages. See Table 6-5.

Table 6-2 Data Import Summary

Field	Description
State	The state of the data import operation: • **In Progress**: The data import is in progress • **Succeeded**: The data import has succeeded • **Failed**: The data import has failed
Progress (%)	The progress of the data import in percentage.
Resource ID	The unique resource identifier assigned to the data import.
Description	The optional user-defined description for the data import.
Last updated	The date and time when the **Data Import Details** were last updated.
Created	The date and time when the data import operation started.

Table 6-3 Data Import Source

Field	Description
S3 URI	The Amazon S3 URI of the S3 bucket containing the data being imported.
Character set	The character set for the import data.
Update GTID set (for MySQL dump file imports only)	For data that contain GTIDs, how the `gtid_executed` GTID set from the source MySQL instance is applied to the `gtid_purged` GTID set on the target MySQL instance. • **OFF**: (Default) Do not update the GTID set. This is also the option for data that do not contain GTIDs. • **APPEND**: Append the `gtid_executed` GTID set from the source MySQL instance to the `gtid_purged` GTID set on the target MySQL instance. • **REPLACE**: Replace the `gtid_purged` GTID set on the target MySQL instance with the `gtid_executed` GTID set from the source MySQL instance.
Files (for text file imports only)	The names of imported data files.
Skip rows (for text file imports only)	The number of rows that is skipped from the beginning of the imported data file or, in the case of multiple import files, at the beginning of every file included in the file list.

Table 6-3 (Cont.) Data Import Source

Field	Description
Lines terminated by (for text file imports only)	The character(s) with which each of the lines are terminated in the imported data file.
Fields terminated by (for text file imports only)	The character(s) with which each of the fields are terminated in the imported data file.
Fields enclosed by (for text file imports only)	The character with which the utility encloses each of the fields in the imported data file.
Fields escaped by (for text file imports only)	The character that is to begin escape sequences in the imported data file.
Fields optionally enclosed (for text file imports only)	Whether a field is enclosed only if it has a string data type.

Table 6-4 Data Import Destination Details

Field	Description
Schema/Table	The tables being imported, listed under the schemas they are under.
Progress	The progress on the table import.
Size	The size of the table.

Table 6-5 Data Import Messages

Field	Description
Errors	Codes for any errors in the data import operation.
Warnings	Warnings on the import operation that require user attention.
Info	Information on the data import.

6.2.2 Bulk Ingest Feature

Use the bulk ingest feature to import text files in bulk from an Amazon S3 bucket to a DB System in HeatWave on AWS in the same region. Bulk import is ideal for importing large tables.

This task requires the following:

- Access to AWS Management Console.
- Access to HeatWave Console.
- A client that is connected to the DB system. See Connecting to a DB System.

Do the following to bulk ingest data to a DB System:

1. Create an IAM policy to access the Amazon S3 bucket. See Creating an IAM Policy to Access an Amazon S3 Bucket.
2. Create an IAM role that grants the DB System access to read data from the desired Amazon S3 bucket. See Creating an IAM Role to Access an Amazon S3 Bucket.

3. (Optional) If a MySQL user (and not a DB System administrator) is importing data using the bulk ingest feature, grant the `LOAD_FROM_S3` privilege. See Granting Privileges to Bulk Ingest Data From Amazon S3.
4. Create a new DB System or edit an existing DB System, and enter the data import role ARN details. See Creating a DB System and Editing a DB System.
5. Bulk ingest data from an Amazon S3 bucket. See Importing Data Using the Bulk Ingest Feature.

- Granting Privileges to Bulk Ingest Data From Amazon S3
  By default, a DB System administrator has the `LOAD_FROM_S3` privilege. However, if a user is importing data using the bulk ingest feature, the `LOAD_FROM_S3` privilege is required.
- Importing Data Using the Bulk Ingest Feature
- Bulk Ingest Limitations

6.2.2.1 Granting Privileges to Bulk Ingest Data From Amazon S3

By default, a DB System administrator has the `LOAD_FROM_S3` privilege. However, if a user is importing data using the bulk ingest feature, the `LOAD_FROM_S3` privilege is required.

This task requires the following:

- A running DB System. See Creating a DB System.

Do the following to grant the `LOAD_FROM_S3` privilege to a user:

1. Connect to the DB System. See Connecting to a DB System.
2. Grant the `LOAD_FROM_S3` privilege to the user:

```
GRANT LOAD_FROM_S3 ON <Schema.Table> TO <User>
```

6.2.2.2 Importing Data Using the Bulk Ingest Feature

Connect to a DB System and bulk ingest data from an Amazon S3 bucket to a DB System in the same region.

This task requires the following:

- A client that is connected to the DB System. See Creating a DB System.
- Properly configured data import role ARN in the DB System. See Creating an IAM Role to Access an Amazon S3 Bucket, Creating a DB System, and Editing a DB System.
- Data, which you want to bulk ingest, in an Amazon S3 bucket. See Exporting Data.

Do the following to bulk ingest data from an Amazon S3 bucket:

1. Connect to a DB System. See Connecting to a DB System.
2. Create a new database or use an existing database to bulk ingest data. For example, the following query creates a new database, `airportdb`, and selects the database for further actions:

```
CREATE DATABASE airportdb;
USE airportdb;
```

3. Use an existing empty table or create a new table. For example, the following query creates a new table, `booking`:

```
CREATE TABLE IF NOT EXISTS `booking`
   (`booking_id` int NOT NULL AUTO_INCREMENT,
    `flight_id` int NOT NULL,
    `seat` char(4) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci
DEFAULT NULL,
    `passenger_id` int NOT NULL,
    `price` decimal(10,2) NOT NULL,
     PRIMARY KEY (`booking_id`)) ENGINE=InnoDB AUTO_INCREMENT=55099799
DEFAULT CHARSET=utf8mb4
        COLLATE=utf8mb4_unicode_ci COMMENT='Verifying bulk ingest';
```

4. Bulk ingest data from an Amazon S3 bucket into the table you created using the `LOAD DATA` query with the `BULK` algorithm:

```
LOAD DATA FROM S3 '<Amazon-S3-URL>' INTO TABLE <Table-name>
        COLUMNS TERMINATED BY '\t' LINES TERMINATED BY '\n' ALGORITHM=BULK;
```

   - `<Amazon-S3-URL>`: Specify the Amazon S3 bucket URL present in the same region as the DB System in the following format:

     ```
     s3-<Region>://<Bucketname>/<FilenameOrPrefix>
     ```

     - `<Region>`: The AWS region that contains the Amazon S3 bucket to load.
     - `<Bucketname>`: The name of the Amazon S3 bucket that contains the data to load.
     - `<FilenameOrPrefix>`: The filename or prefix of one or more text files to load.
   - `<Table-name>`: Specify an already existing empty table or create a new table.
   - Specify the column and line terminators used by the in the data files you are loading. The statement above assumes the column terminator is `\t` and the line terminator is `\n`.

   It is recommended to split the data into multiple files to improve the performance of bulk ingest. For example, the following query loads data from an Amazon S3 bucket that is split across 25 files. The files are named, `booking.tsv1`, `booking.tsv2`, ..., `booking.tsv25`:

   ```
   LOAD DATA FROM S3 's3-us-east-1://mysql-heatwave-data-us-east-1/airportdb/
   booking.tsv.'
           COUNT 25 IN PRIMARY KEY ORDER INTO TABLE booking
           COLUMNS TERMINATED BY '\t' LINES TERMINATED BY '\n' ALGORITHM=BULK;
   ```

   See Bulk Ingest Data to MySQL Server.

6.2.2.3 Bulk Ingest Limitations

The following are some limitations to the Bulk Ingest Feature of HeatWave on AWS when using the `LOAD DATA FROM S3` statement with `ALGORITHM=BULK`:

- – The target table must be empty. The state of the table should be as though it has been freshly created. If the table has instantly added or dropped column, run the `TRUNCATE TABLE` statement before running `LOAD DATA FROM S3` with `ALGORITHM=BULK`.

- The target table must not be partitioned.
- The target table must not contain secondary indexes.
- The target table must be in a file-per-table tablespace—it must not be in a shared tablespace.
- The target table must have the default row format `ROW_FORMAT=DYNAMIC`. If needed, use `ALTER TABLE` to change the table's row format after running `LOAD DATA FROM S3` with `ALGORITHM=BULK`.
- The target table must contain a primary key.
- Prefix indexing for the primary key is not supported.
- The target table must not contain virtual or stored generated columns.
- The target table must not contain foreign keys.
- The target table must not contain `CHECK` constraints.
- The target table must not contain triggers.
- The target table must not be replicated to other nodes.
- The target table must not use a secondary engine. If needed, set the secondary engine after running `LOAD DATA FROM S3` with `ALGORITHM=BULK`.
- The `INFILE` and `URL` clauses are not supported for `LOAD DATA FROM S3` with `ALGORITHM=BULK`.
- `LOAD DATA FROM S3` with `ALGORITHM=BULK` locks the target table exclusively and does not allow other operations on the table when the query is running.
- Automatic rounding or truncation of the input data is not supported. `LOAD DATA FROM S3` with `ALGORITHM=BULK` will fail if the input data requires rounding or truncation in order to be loaded.
- The target table cannot be a temporary table.
- `LOAD DATA FROM S3` with `ALGORITHM=BULK` is atomic but not transactional. It commits any transaction that is already running. On failure, the `LOAD DATA FROM S3` statement is completely rolled back.
- `LOAD DATA FROM S3` with `ALGORITHM=BULK` cannot be executed when the target table is explicitly locked by a `LOCK TABLES` statement.

6.2.3 Dump Loading Utility

The dump loading utlity is an import utility that imports schemas to a DB System. See MySQL Shell Dump Loading Utility. To import a schema to a DB System, MySQL Shell must be installed on a machine with access to the DB System.

- Importing Data Using the Dump Loading Utility

6.2.3.1 Importing Data Using the Dump Loading Utility

Use the dump loading utility of MySQL Shell to load a dump from an Amazon S3 bucket to a DB System in HeatWave on AWS.

This task requires the following:

- MySQL Shell 8.0.27, or higher. Exports created by MySQL Shell 8.0.27, or higher, cannot be imported by earlier versions of MySQL Shell.

- Data exported from a source MySQL Server following the instructions described in Exporting Data Using MySQL Shell.
- Enough storage space on your DB System for the data you intend to import. To check DB System storage space, see Viewing DB System Details.
- You have run the `loadDump()` command in `dryrun` mode to check that there are no issues when the dump files are loaded from an Amazon S3 bucket into the connected DB System:

  ```
  util.loadDump("<Schema>", {s3bucketName: "<BucketName>", dryRun: true,
  ocimds: true})
  ```

  This performs a test run of the import checking for compatibility issues. The compatibility issues and remediation steps are listed in the output. See MySQL Shell Instance and Schema Dump Utilities.

Do the following to import data using the dump loading utility:

1. Start a MySQL Shell session and connect to the DB System:

   ```
   mysqlsh <Username>@<HostNameOfMySQLDBSystem>
   ```

   See Connecting with MySQL Shell .

2. Import data using the dump loading utility:

   ```
   util.loadDump("<Prefix>",{s3BucketName: "<BucketName>", threads:
   <ThreadSize>})
   ```

   - `<Prefix>`: (Optional) Adds a prefix to the files uploaded to the bucket. If you specify this option, the files are uploaded to the defined bucket with the prefix in the following format: `<BucketPrefix>/filename`, similarly to a file path. For example, if `<BucketPrefix>` is set to `test`, every file uploaded to the defined bucket, `<BucketName>`, is done so as `test/<filename>`. If you download the file, the prefix is treated as a folder in the download. For local exports, this parameter is the path to the local directory you want to export to.
   - `<BucketName>`: The an Amazon S3 bucket from which the dump is to be loaded.
   - `<ThreadSize>`: The number of processing threads to use for this task. The default is 4. For best performance, it is recommended to set this parameter to twice the number of CPUs used by the target DB System.

7

Inbound Replication

This chapter covers how to create and manage **Channels** for implementing inbound replication.

7.1 About Inbound Replication

Inbound replication enables asynchronous replication from a MySQL-based source (an on-premises or other cloud MySQL database instance, or other HeatWave on AWS DB System) to a HeatWave on AWS DB System. It uses a replication channel configured in HeatWave on AWS to copy transactions from the source to a HeatWave on AWS DB System (replica). The channel connects the source to the replica, and copies data from the source to the replica.

Asynchronous replication means that the replica does not need to be running and connected to the source all the time. It can pick up new updates whenever it is online and reconnects.

A replica DB System can connect to only one MySQL source. The source server does not control the DB System replica and does not need to have permission to write to it. The replica uses the configured replication channel to connect to the source, providing a set of replication user credentials. Over the connection formed by the channel, the replica retrieves committed transactions from the source. Then the replica writes those transactions to its own copy of the databases.

> **Note:**
>
> Inbound replication is not a managed functionality. You are responsible for configuring and maintaining the channel, and for ensuring that the traffic between source and replica is properly configured.

Impact of the DB System Operations on the Replication Channel

The DB System operations affect the replication channel:

- If you stop the DB System, it stops any enabled Channels and changes its state to `Needs Attention`.
- If you delete the DB System, it deletes any Channels attached to the DB System.

- If you restart, upgrade, or update a DB System, it temporarily suspends the Channel while the operation is ongoing. When the operation is complete, replication is resumed. The Channel state changes to `Resuming` while the channel is being resumed.

7.2 Source Configuration

To use inbound replication in HeatWave on AWS, the source and its network connection to the replica DB System must have the following configurations.

- The network must be configured to permit replication traffic between the DB System replica and the source server.
- The minimum supported version of MySQL for a replication source is 5.7.9.
- The source cannot be running a higher version of MySQL than the DB System. The replica must be running the same version as, or a higher version of MySQL than, the source.
- If you intend to encrypt the communication between source and replica, the source must be configured to use SSL. See Server-side Configuration for Encrypted Connections.
- The source and the DB System must run with the same `lower_case_table_names` system variable value. See System Initialization Variables
- The source must have binary logging enabled, with the system variable `log_bin=ON`.
- The source must use row-based binary logging, with the system variable `binlog_format=ROW`.
- *For GTID-based replication only*: The source must use GTIDs, with the system variables `gtid_mode=ON` and `enforce_gtid_consistency=ON`.
- The replication user must be present on the source server with the required privileges. See Creating a Replication User On a Source Server.

7.3 Creating a Replication User On a Source Server

The replication user for inbound replication must be present on the source server with the required privileges. The replica uses this user account when it connects to the source server.

This task requires:

- MySQL Shell or MySQL client.

Do the following to add a replication user to the source server:

1. Open MySQL Shell and connect to the MySQL source server.
2. Run the following SQL command to create a replication user, and to permit only encrypted connections for all accounts named by the statement. In this example, the username for the replication user is `rpluser001`:

```
CREATE USER rpluser001 IDENTIFIED BY 'password' REQUIRE SSL;
```

> **Note:**
>
> Inbound replication in HeatWave on AWS requires MySQL source password to contain 8 to 32 characters including at least one number, one uppercase letter, one lowercase letter, and one character from `,.-+*;:_!#%&/()=?><`

3. Run the following command to grant the `REPLICATION SLAVE` privilege to the new replication user, `rpluser001` in this example:

```
GRANT REPLICATION SLAVE on *.* to rpluser001;
```

The replication user is created and granted the required privileges. Note the user name and password so that you can specify them when you create the replication channel..

7.4 Creating a Channel

> **Note:**
>
> In MySQL replication, a **replica** is a MySQL server that receives changes from another MySQL server (source) and applies the changes. In HeatWave on AWS, a Channel references the replica as the **target** DB System.

Use the HeatWave Console to create a channel. A channel connects the source (an on-premises or other cloud MySQL database instance, or other HeatWave on AWS DB System) to a target HeatWave on AWS DB System (replica), and copies data changes from the source to the target.

This task requires the following:

- A MySQL source server that meets the requirements in Source Configuration.
- A replication user on the MySQL source server. See Creating a Replication User On a Source Server.
- An active DB System as the target of the Channel. If the source server contains data, import existing data into the DB System:
  - Export the data from the MySQL source server (see Exporting Data Using MySQL Shell for instructions) and:
    - *If using GTID-based replication:* Capture the value of the source server's system variable `gtid_executed`, from which replication should begin. The value is also available as metadata in the `gtidExecuted` field in the `@.json` dump file created by MySQL Shell's dump utilities.
    - *If not using GTID-based replication:* Capture the source server's binary log coordinates (the log file name and the log position), from which replication should begin. See Obtaining the Replication Source Binary Log Coordinates on how to do that.
  - Import the data into the target DB System (see Importing Data Using the Data Import Feature for instructions), and then:
    - *If using GTID-based replication:* apply the source's `gtid_executed` GTID set to the target's `gtid_purged` GTID set (see Update GTID set on how to do that).
    - *If not using GTID-based replication:* supply the binary log coordinates to the Replication Positioning settings when you create or edit a replication channel on the HeatWave Console.

Do the following to create a replication channel:

1. In the HeatWave Console, select the **HeatWave MySQL** tab, and then click **Channels** to open the **Channels** page.

2. On the **Channels** page, click **Create Channel** to open the **Create Channel** dialog.
3. In the **Create Channel** dialog, enter the following information:
   - **Basic information**:
     - **Display name**: Specify a display name for the channel. If you do not specify a name, one is generated for you in the format, `channelyyyyMMddHHmmssSSS`.
     - **Description**: (Optional) Specify a description of the channel and its purpose.
     - **Enabled automatically upon creation**: Specify whether the channel should start automatically after it is successfully created. By default, it is enabled. If you disable this option, the Channel will be in Inactive state and will not start replication. You have to enable the channel manually after the channel is created to start replication.
   - **Source connection**: Configure the MySQL source from where you want to replicate the data:
     - **Hostname**: Specify the hostname of the MySQL source. You can either specify an IP address or a fully qualified domain name. For example, 112.123.121.12, or `server.yourdomainname.com`.
     - **Port**: (Optional) Specify the port number (between 1024 and 65535) the MySQL source listens on. The default value is 3306.
     - **Allowed Outbound addresses**: Specify the IP addresses (in the CIDR format) of the MySQL source from which the DB System replicates the data. If you want to specify more than one IP address ranges, use a semicolon. For example, 10.2.3.1/32;10.3.0.0/16.
     - **Username**: Specify the replication username for the account that you created on the MySQL source server. The channel uses these credentials to connect to the source. See Creating a Replication User On a Source Server.
     - **Password**: Specify the password for the replication user account.

       > **Note:**
       >
       > Inbound replication in HeatWave on AWS requires MySQL source password to contain 8 to 32 characters including at least one number, one uppercase letter, one lowercase letter, and one character from `,.-+*;:_!#%&/()=?><`
       > You have to change the password on the MySQL source if it does not meet these requirements.

     - **SSL mode**: Select the required SSL mode. The selected mode is used to populate the SSL-specific values of the connection to the MySQL Source. Select either of the following SSL modes:
       - **Disabled**: Establishes an unencrypted connection between the source and target.
       - **Required**: (Default) If the server supports encrypted connection, establishes an encrypted connection. The connection attempt fails if an encrypted connection cannot be established.
       - **Verify certificate authority**: Like the **Required** mode, establishes an encrypted connection if the server supports encrypted connections and additionally verifies the Certificate Authority certificate configured on the

source against the Certificate Authority's X509 certificate (PEM). You have to upload your Certificate Authority's X509 certificate.

  - **Verify identity**: Like the **Verify certificate authority** mode, establishes an encrypted connection, verifies the Certificate Authority's certificate, and additionally verifies the source hostname, which you define in the source SSL certificate, against the hostname that you define in the **Hostname** field. You have to upload your Certificate Authority's X509 certificate.
  - **Certificate authority's X509 certificate (PEM)**: Enables you to upload the source Certificate Authority's X509 certificate in PEM format. It is displayed only when you select the **Verify certificate authority** or **Verify identity** mode. The certificate is used to verify the Certificate Authority's certificate on the source.

- **Replication positioning**: Configure the Source GTID settings:
  - **Use GTID auto-positioning (recommended)**: Select this option when the system variable, `gtid_mode`, is set to `ON` on the source. It means the source server can provide the replica with GTID information for auto-positioning. For more information, see GTIDs.
  - **Do not use GTID auto-positioning**: Select this option when the system variable, `gtid_mode`, is set to `OFF`,`OFF_PERMISSIVE`, or `ON_PERMISSIVE` on the source. It means the source server cannot provide the replica with GTID information for auto-positioning. Select one of the following options to convert a transaction that is anonymous to one that has GTIDs:
    - **Manually specify a UUID**: There are two options:
      - Define your own UUID by typing it into the **UUID** field.
      - Accept the generated UUID shown in the **UUID** field, or generate a new one by clicking the regenerate button.

      See GTID Format and Storage for more information on UUIDs.
    - **Same UUID as target DB system**: Select this option to use the same UUID as the target DB System.

    Also specify the binary log coordinates (see Obtaining the Replication Source Binary Log Coordinates on how to obtain them) when not using GTID auto-positioning:
    - **Binary log file name**: Specify the name of the binary log file, which contains events that describe database changes on the source. See The Binary Log.
    - **Binary log offset**: Specify the binary log offset within the binary log file from which the replica should start reading the source's binary log.
- **Target DB system:** Configure the DB System to which you want to replicate the data.
  - **Channel name**: (Optional) Specify the channel name. The target DB System uses this replication channel to communicate with the MySQL source. If you do not specify a name, `replication_channel` is used.
  - **Applier username**: (Optional) Specify the username of the replication applier on the target DB System. If you do not specify a username, the credentials of the DB System administrator is used.
  - **Replication delay**: Set the amount of time, in seconds, that the channel waits before applying a transaction received from the source.
  - **Tables without primary key**: Specify how a replication channel handles the creation and alteration of tables with no primary keys on the source.

        - **Raise an error** : Raise an error when replicating a `CREATE TABLE` or `ALTER TABLE` transaction with no primary keys.
        - **Allow**: (Default) Allow replicating a `CREATE TABLE` or `ALTER TABLE` transaction with no primary keys.
        - **Generate primary key**: Allow replicating a `CREATE TABLE` or `ALTER TABLE` transaction with no primary keys, and automatically generate a new invisible primary key column while creating a table without primary keys on the target.
    - **Select DB system**: Select the DB System to use as the replication target.
    - **Channel filter**: (Optional) Click **Channel filter options** to configure the type and value of replication filters for the channel. See Channel Filter Rules for Inbound Replication for details.
        - **Common filter templates**: You can use already existing filter templates that populate the **Type** and **Value**. Select a filter template from the available list that matches the source.
        - **Type**: Select the filter type. You can select a variety of filter types such as ignore a database or table and rewrite a database.
        - **Value**: Enter a value for the filter type.
        - Click **Add new filter** to add another filter.
- Click **Create**.

The state of the channel changes to **Creating**, and when the create is complete, the status changes to one of the following:

- **Inactive** if **Enabled automatically upon creation** is disabled.
- **Active** if the channel is enabled successfully.
- **Needs Attention** if the channel encounters any error while enabling it. The replication cannot be started.

See the next section on the channel filter rules for configuring replication filters.

- Channel Filter Rules for Inbound Replication

7.4.1 Channel Filter Rules for Inbound Replication

Each filter rule can contain only one filter type and value. However, you can add multiple filter rules and the result is the logical addition of all the rules. For example, if you create a filter rule with type, `REPLICATE_DO_DB`, and filter value, `mysql1`, and another filter rule, with type, `REPLICATE_DO_DB`, and filter value, `mysql2`, then both `mysql1` and `mysql2` databases are replicated.

Table 7-1 Channel Filter Rules

Filter Type	Filter Value	Details
`REPLICATE_DO_DB`	`database`	Replicates the specified database.
`REPLICATE_IGNORE_DB`	`database`	Restricts the replication of the specified database.
`REPLICATE_DO_TABLE`	`database.table`	Replicates the specified table in the database.

Table 7-1 (Cont.) Channel Filter Rules

Filter Type	Filter Value	Details
REPLICATE_IGNORE_TABLE	database.table	Restricts the replication of the specified table in the database.
REPLICATE_WILD_DO_TABLE	database.table	Replicates the table that matches the specified wildcard pattern.
REPLICATE_WILD_IGNORE_TABLE	database.table	Restricts the replication of any table that matches the specified wildcard pattern.
REPLICATE_REWRITE_DB	from_database->to_database	Translates from_database on the source to to_database.

Wildcard Patterns: REPLICATE_WILD_DO_TABLE and REPLICATE_WILD_IGNORE_TABLE support wildcards. Wildcard patterns can contain the % and _ wildcard characters. % represents 0 or more characters and _ represents any single character. For example:

- A filter value of db%.table% in filter type REPLICATE_WILD_DO_TABLE replicates only those tables where the database name starts with db and the table name starts with table.
- A filter value of db%.% in filter type REPLICATE_WILD_DO_TABLE replicates all tables where the database name starts with db%.
- A filter value of db_.table_ in filter type REPLICATE_WILD_IGNORE_TABLE ignores those tables where the database name contains three characters and starts with db and the table name contains six character and starts with table.

If you want to use the two wildcard characters literally in the database or table names in your rule, escape them with backslashes (\). For example, my_db.top_90\%.

7.5 Managing Replication Channels

This section describes how to manage Channels using the HeatWave Console.

- Disabling or Enabling a Channel
- Editing a Channel
- Resuming a Channel
- Resetting a Channel
- Deleting a Channel

7.5.1 Disabling or Enabling a Channel

Disabling a channel pauses the replication and **enabling** a channel resumes the replication.

> **Note:**
>
> If the source server contains existing data, it is necessary to copy over this data to the target server before enabling the channel and starting replication for the first time. For more information, see **Choosing a Method for Data Snapshots** and Importing Data.

To disable or enable a Channel:

1. In the HeatWave Console, select the **HeatWave MySQL** tab, and then click **Channels** to open the **Channels** page.
2. In the list of channels, find the channel you want to disable or enable.
3. To disable an active channel, do one of the following:
   - Select the row of the channel to highlight it, and click the **Disable** button.
   - Click the name of the channel to open the **Channel Details** page. Click the **Disable** button.

   The **Disable MySQL Channel** dialog is displayed. Click **Disable** to confirm.

   The state of the channel changes to **Updating** and then to **Inactive**.
4. To enable an inactive channel, do one of the following:
   - Select the row of the channel to highlight it, and click the **Enable** button.
   - Click the name of the channel to open the **Channel Details** page. Click the **Enable** button.

   The **Enable MySQL Channel** dialog is displayed. Click **Enable** to confirm.

   The state of the channel changes to **Updating** and then to **Active**. The final state becomes **Needs Attention** if it encounters any error and replication cannot start.

Related Topics

- Importing Data

7.5.2 Editing a Channel

Editing a channel allows you to change the configuration of the channel.

> **Note:**
>
> You cannot change the target DB System of a channel. To change the target DB System, you must delete the Channel and create a new Channel. See Deleting a Channel and Creating a Channel.

To edit a Channel:

1. In the HeatWave Console, select the **HeatWave MySQL** tab, and then click **Channels** to open the **Channels** page.
2. On the **Channels** page, in the list of channels, find the channel you want to edit, and do one of the following:
   - Select the row of the channel to highlight it, and choose the **Edit Channel** action from the **Actions** menu.
   - Click the name of the channel to open the **Channel Details** page. Click the **Edit Channel** button from the **Actions** menu.

   The **Edit Channel** dialog is displayed.
3. In the **Edit Channel** dialog, update the required changes. You cannot change the target DB System.
4. Click **Save** to update the changes.

7.5.3 Resuming a Channel

The channel state becomes **Needs Attention** when an error occurs in the replication channel. After fixing the error, you can resume the channel to restart the replication.

To resume a Channel:

1. In the HeatWave Console, select the **HeatWave MySQL** tab, and then click **Channels** to open the **Channels** page.
2. On the **Channels** page, in the list of channels, find the channel you want to resume, and do one of the following:
   - Select on the row of the channel to highlight it, and choose the **Resume** action from the **Actions** menu.
   - Click the name of the channel to open the **Channel Details** page. Click the **Resume** button from the **Actions** menu.

   The **Resume MySQL Channel** dialog is displayed.
3. Click **Resume** to restart the replication.

The state of the channel becomes **Resuming**. When the channel restarts successfully, the state changes to **Active**. If the error persists or other error occurs, and the replication cannot start, the state changes to **Needs Attention**.

7.5.4 Resetting a Channel

Resetting a channel removes all replication data on the channel, except the channel configuration. It is equivalent to RESET REPLICA ALL FOR CHANNEL. When there is an unrecoverable issue, the reset operation clears the records associated with replication so that the channel can have a clean start.

The target DB System drops its position in the source binary log, clears the replication metadata repositories, deletes the relay log files, and starts a new relay log file.

You can only reset an **Inactive** channel. To reset a Channel:

1. In the HeatWave Console, select the **HeatWave MySQL** tab, and then click **Channels** to open the **Channels** page.
2. On the **Channels** page, in the list of channels, find the channel you want to reset, and do one of the following:
   - Click on the row of the channel to highlight it, and choose the **Reset** action from the **Actions** menu.
   - Click the name of the channel to open the **Channel Details** page. Click the **Reset** button from the **Actions** menu.

   The **Reset MySQL Channel** dialog is displayed.
3. Click **Reset** to reset the channel.

The state of the channel becomes **Resetting**, and when the reset is complete, the state changes to **Inactive**. Then, you can enable the channel to restart the replication.

7.5.5 Deleting a Channel

Deleting a Channel stops the replication and removes the channel configuration.

To delete a Channel:

1. In the HeatWave Console, select the **HeatWave MySQL** tab, and then click **Channels** to open the **Channels** page.
2. On the **Channels** page, in the list of channels, find the channel you want to delete, and do one of the following:
   - Click on the row of the channel to highlight it, and choose the **Delete** action from the **Actions** menu.
   - Click the name of the channel to open the **Channel Details** page. Click the **Delete** button from the **Actions** menu.

   The **Delete MySQL Channel** dialog is displayed.
3. Click **Delete MySQL Channel** to go ahead with the deletion.

The state of the channel becomes **Deleting**, and when the delete is complete, the state changes to **Deleted**. You can click the name of the **Channel** to view the details. However, you cannot perform any other operation on the deleted channel.

7.6 Viewing Channel Details

Do the following to view Channel details:

1. In the HeatWave Console, select the **HeatWave MySQL** tab, and then click **Channels** to open the **Channels** page.
2. On the **Channels** tab, in the list of channels, find the channel you want to view details for, and do one of the following:
   - Click the row of the Channel to highlight it. The Channel details appear below the list of channels.
   - Click the name of the Channel to open the **Channel Details** page for the selected Channel.

For descriptions of DB System details, see Channel Details.

- Channel Details

7.6.1 Channel Details

The **Channel Details** page is divided into the following sections:

Table 7-2 Channel Details Page

Name	Description
Summary	Channel summary details. See Table 7-2.
Channel Source and Target Information	Channel source and target details. See Table 7-3.
Channel Event Details	Channel event details. See Table 7-5.

Table 7-3 Channel Summary Details

Field	Description
State	The state of the Channel. • **Creating**: The Channel is being created. Provisioning can take several minutes. The channel is not yet replicating data from the source to the target. • **Active**: The Channel is running and replicating successfully. • **Inactive**: The Channel is not replicating because the user has disabled it. • **Needs Attention**: This status is displayed for the following reasons: – The Channel is not replicating due to an error. – The target DB System is `Inactive`, that is, the Channel is created or updated while the DB System is in `Inactive` state. Pending actions are applied when the DB System is started. • **Updating**: The Channel is in the process of being updated. • **Resuming**: The Channel is in the process of being resumed. • **Resetting**: The Channel is in the process of being reset. • **Deleting**: The Channel is being deleted. • **Deleted**: The Channel has been deleted and is no longer available. • **Resuming Error**: An attempt to resume the Channel failed. • **Resetting Error**: An attempt to reset the Channel failed. • **Updating Error**: An attempt to update the Channel failed.
Resource ID	The unique resource identifier assigned to the Channel when it is created.
Description	The user-defined description of the Channel.
Enabled	The status of the channel. **Yes** for enabled, **No** for disabled.
Last updated	The date and time the Channel was last updated.
Created	The date and time the Channel was created.

Table 7-4 Channel Source and Target Details

Field	Description
Source	• **Hostname**: The IP address or hostname of the source MySQL server. • **Port**: The port number the source MySQL server listens for incoming connections. • **Username**: The username of the replication user. • **Allowed outbound addresses**: The IP addresses (in the CIDR format) of the MySQL source to enable egress connections from the target DB System. • **SSL mode**: The security state of the connection to the source. Possible values are: – **Disabled**: No encryption required between source and target. – **Required**: (default) Establishes an encrypted connection if the server supports encrypted connections. The connection attempt fails if an encrypted connection cannot be established. – **Verify certificate authority**: Like Required, but additionally verify the CA certificate configured on the source against the Certificate Authority (CA) certificate (X509 PEM file). – **Verify identity**: Like Verify Certificate Authority, but additionally verify the source's hostname, defined in the source's SSL certificate, against the hostname defined in the Hostname field. • **SSL CA certificate**: The contents of the uploaded CA X509 certificate if the SSL mode is **Verify certificate authority** or **Verify identity**. • **Use GTID auto-positioning**: Whether the source server can provide the replica with GTID information for auto-positioning. Possible values are: – **Yes**: Replication is GTID-based. – **No**, for which these fields appear: * **Source of the UUID**: The source of the UUID. Possible values are: * **Manual input**: The UUID was user-defined or was generated by the Console. A **UUID** field shows the UUID value. * **Same as target DB system**: Use target DB System's UUID.

Table 7-4 (Cont.) Channel Source and Target Details

Field	Description
Target	• **DB System name**: The display name of the target DB System and a link to it. • **DB System state**: The state of the target DB System. • **Channel name**: The replication channel name. This is the replication channel used by the target MySQL DB System for the communication with the MySQL source. • **Applier username**: The username of the replication applier. • **Replication delay**: The amount of time, in seconds, that the channel waits before applying a transaction received from the source. • **Table without primary key**: How the creation and alteration of tables with no primary keys on the source are handled. Possible values are: – **Raise an error** – **Allow** – **Generate primary key**: Allows replicating a `CREATE TABLE` or `ALTER TABLE` transaction with no primary keys, and automatically generates a new invisible primary key column while creating a table without primary keys on the target.
Channel filters	• **Type**: The filter type. • **Value**: The value for the filter type. See Channel Filter Rules for Inbound Replication for details on the filter types and their values .

Table 7-5 Channel Event Details

Field	Description
Type	The event and its status.
Severity	The severity level of the event, which can be one of • `INFO`: General information is provided. No special concerns • `WARNING`: Warning on an event that might impact the DB System's operation if no actions are taken. • `CRITICAL`: A critical event. An urgent action is required to avoid adverse impact on your DB System.
Message	The message body of the event report.
Created	The date and time the event report was created.

See Events for details on event reports.

7.7 Limitations

Inbound replication in HeatWave on AWS does not support some of the configurations that are possible for MySQL replication.

- Only row-based replication is supported. This binary log format is the default. Statement-based replication and mixed replication are not supported.
- Only asynchronous replication is supported. Semi-synchronous replication is not supported.
- Only replication from a single source is supported. Multi-source replication is not supported.
- Changes to the `mysql` schema are not replicated and cause replication to stop.
- Using a `root@localhost` or `administrator` user in the DEFINER clause of CREATE PROCEDURE and CREATE FUNCTION statements is not supported. Instead, create a dedicated user with the privileges required to run the statements, and then edit the statements with the dedicated user in the DEFINER clause.

8
Manage Data in HeatWave with Workspaces

The **Workspaces** page in the HeatWave Console enables you to perform the following:

- **Loading data into or unloading data from a HeatWave Cluster**: Load operations are performed using *Auto Parallel Load*, which optimizes operation time and memory usage by predicting and using an optimal degree of parallelism. Data is loaded into HeatWave from the associated DB System. If you have not loaded data into the DB System, see Importing Data. Unload operations are performed using *Auto Unload*.
- Loading or Unloading Data into HeatWave Cluster
- Creating Lakehouse Data Mapping
- Maximum Number of Tables Loadable into a HeatWave Cluster

8.1 Loading or Unloading Data into HeatWave Cluster

Use HeatWave Console to load data into or unload data from a HeatWave Cluster.

This task requires the following MySQL privileges:

- The `PROCESS` privilege.
- The `EXECUTE` privilege on the `sys` schema.
- The `SELECT` privilege on the Performance Schema.

The time required to load data from the DB System into the HeatWave Cluster depends on the data size.

> **Note:**
>
> HeatWave Cluster data can also be managed using SQL or the Auto Parallel Load interface from a MySQL client. See Loading Data, in the *HeatWave User Guide*.

To load data, do the following:

1. Connect to a DB System that has a HeatWave Cluster. See Connecting from the Console.
2. Click the **Manage Data in HeatWave** tab. You can view the following information about schemas and tables:
   - **Name**: The name of the schema or table.
   - Source: The source of the schema or table. For example, InnoDB.
   - **Memory Size Estimate (GiB)**: An estimate of the memory required on the HeatWave Cluster for the schema or table.
   - **Loaded**: The number of tables loaded into HeatWave. You can click the **Refresh** icon to refresh the load status.
   - **Rows Estimate**: The estimated number of rows in the table.

- **String Column Encoding**: The number of columns to be encoded as `VARLEN` (variable-length) columns, expressed as a fraction of the total number of table columns. For information about HeatWave string column encoding, see Encoding String Columns, in the *HeatWave User Guide*.
- **Predicted Load Time (s)**: The predicted load time in seconds.

3. Select the schemas and tables that you want to load by clicking the check box present in front of the schema or table name. If you select a schema, all the tables in the schema are selected. Expanding a schema using the drop-down control displays the table in the schema. The **Load Status** column provides the load status as a percentage value.
4. After you have selected the tables you want to load or unload, click **Load** or **Unload**. If loading data, the **MySQL Auto Parallel Load tables into HeatWave** dialog appears, providing a summary of the load operation to be executed. The following information is shown:
   - **DB System**: The DB System name
   - **Estimated load size**: The estimated size of the data to be loaded
   - **Estimated load time**: The estimated time required to load the data
   - **Selected schemas and tables**: The selected schemas and tables to be loaded
     - **Name**: Name of the schema or table
     - **Memory Size Estimate (GiB)**: The memory size estimate for the schema or table on the HeatWave Cluster
   - **Estimated total memory footprint**: A pie chart showing the estimated percentage of HeatWave Cluster memory required for each table
   - **Estimated load times (seconds)**: A pie chart showing estimated load times for each table

   If unloading data, the **MySQL Auto Parallel Unload tables from HeatWave** dialog appears, providing a summary of the unload operation to be executed. The following information is shown:
   - **DB System**: The name of the DB System
   - **Estimated unload size**: The estimated size of the data to be unloaded
   - **Selected schemas and tables**: The selected schema and tables to be unloaded
     - **Name**: Name of the schema or table
     - **Memory Size Estimate (GiB)**: The memory size estimate for the schema or table on the HeatWave Cluster
   - **Estimated memory foot print (GiB)**: The estimated HeatWave Cluster memory footprint showing occupied memory, current free memory, and additional available free memory after the unload operation
5. Click **Confirm Load** or **Confirm Unload** to start the load or unload operation.

The **Cluster Memory Snapshot** shows the amount of HeatWave Cluster memory used.

8.2 Creating Lakehouse Data Mapping

For HeatWave Lakehouse to process data in Amazon S3, it needs to first map the data in files on S3 to a HeatWave Lakehouse table. Once the S3 data is provided, the schema for the Lakehouse table is inferred automatically by HeatWave Autopilot. After the data mapping is

created, it allows the S3 data to be loaded into the Lakehouse table for querying by Lakehouse.

This task requires the following:

- A running DB system with an active HeatWave Cluster. See Creating a HeatWave Cluster.
- Properly configured Lakehouse IAM role ARN. See Creating an IAM Role to Access an Amazon S3 Bucket.
- S3 URIs for one or more files of similar data.

Do the following to create a Lakehouse data mapping:

1. Connect to a DB System that has a HeatWave Cluster with Lakehouse enabled. See Connecting from the Console.
2. Click the **Manage Data in HeatWave** tab, and then click **Create Lakehouse Mapping**.
3. **Source**:

> **Note:**
>
> You can map more than one Amazon S3 bucket file to the destination schema and table, but all the files should be of the same format (i.e., they are all either **CSV**, **Parquet**, or **Avro** files). Use the **Add new file** button to create multiple entries for data sources,

   - **File path type**: Select either of the following:
     - **Prefix**: Enables you to map all files present under an Amazon S3 bucket prefix to the destination schema and table.
     - **Name**: Enables you to map a specific file present in an Amazon S3 bucket to the destination schema and table.
     - **Pattern**: Enables you to map a set of files present in an Amazon S3 bucket defined by a regular expression to the destination schema and table.
   - **File path**: Depending on the **File path type** you select, specify one of the following
     - The Amazon S3 bucket prefix path, `s3://<bucket-name>/<prefix>`
     - The Amazon S3 bucket path with the file name, `s3://<bucket-name>/<prefix>/<filename>`
     - The Amazon S3 bucket path containing a regular expression, `s3://<bucket-name>/<regex>` (the pattern of the regular expression follows the modified ECMAScript regular expression grammar)
   - **Strict mode**: Select **Default**, **Enabled**, or **Disabled**, which controls how Lakehouse handles invalid or missing values for data-changing statements such as INSERT or UPDATE when loading data into HeatWave. This parameter overrides the **Strict mode** setting that you specify in the **File parsing settings** for **CSV** files. See Strict SQL Mode. The following shows what happens if parsing errors, formatting errors, or empty columns are found in a file during data loading, for each possible value of **Strict mode**:
     - **Default**: The situation is handled according to the MySQL server's SQL mode setting.
     - **Enabled**: An error is displayed, interrupting the operation.

- **Disabled**: A warning is displayed, without interrupting the operation. Empty columns are automatically filled with default column values or NULL.

4. **File parsing settings**:

- **Format**: Select the format of the file you want to map. You can select **CSV**, **Parquet**, or **Avro.** The default format is **CSV**. All the files that you want to map should have the same format.

> **Note:**
>
> See HeatWave Lakehouse Limitations for the limitations for mapping and loading files of each of the supported formats.

If you select the **CSV** format, you can set the following:

- **Skip rows**: Specify the number of rows that is skipped from the beginning of the data file (applicable to every file included in the file list). By default, no rows are skipped, and the maximum rows to skip is 20.
- **Field delimiter**: Specify one or more characters to enclose fields. The maximum field delimiter length is 64 characters. The default delimiter is `,`.
- **Record delimiter**: Specify one or more characters to delimit records. The maximum record delimiter length is 64 characters. The default delimiter is `\n`.
- **Escape character**: Specify one or more characters to escape special characters. The escape character is `\`.
- **Quotation marks**: Specify one or more characters to enclose fields. The default character to enclose fields is `"`.
- **Encoding**: Specify the character set to map data. The default character set is `utf8mb4`. See Character Sets and Collations in MySQL.
- **Date format**: See Date_format. The default date format is `auto`.
- **Time format**: See String and Numeric Literals in Date and Time Context. The default time format is `auto`.
- **Strict mode**: Specify whether the mapping takes place during data mapping in strict mode (**Enabled**) or non-strict mode (**Disabled**). See Strict SQL Mode
  - **Default**: The situation is handled according to the MySQL server's SQL mode setting.
  - **Enabled**: If parsing errors, formatting errors, or empty columns are found in the file, an error is displayed, interrupting the operation.
  - **Disabled**: If parsing errors, formatting errors, or empty columns are found in the file, a warning is displayed, without interrupting the operation. Empty columns are automatically filled with default column values or NULL.
- **Has header**: Specify whether the file has a header.
  - **Default**: Default value is **Disabled**
  - **Enabled**: Treat the first row of the file as the header
  - **Disabled**: Treat the file as if it does not have a header.
- **Trim spaces**: Select whether to remove leading and trailing spaces.
  - **Enabled**: Remove leading and trailing spaces.

    - **Disabled**: Do not remove leading and trailing spaces.

5. **Destination**: Specify the destination of the Lakehouse table into which the data is to be loaded.
    - **Schema**: Name of the destination schema.
    - **Table**: Name of the destination table.
6. Click **Next**.
7. Under **Autopilot schema inference**, review the **Definitions**, **Errors**, and **Warnings**. Do one of the following:
    - Click **Create** to create the destination Lakehouse table (which remains empty after its creation, until data is loaded).
    - Click the **Back** button to go back and change your mapping settings.
    - Click the **Cancel** button to cancel the mapping.
    - Click the **Copy** button to copy the DDL statement for data mapping into your clipboard.
    - Click the **Refresh** button to refresh the step with the latest HeatWave schema and table information, to make sure the mapping remains valid after it was first generated.

> **Note:**
>
> Mapping data onto an existing Lakehouse table is not supported. Mapping will fail if the specified **Destination** points to a Lakehouse table that already exists. If **Destination** points to an existing InnoDB table on the target MySQL server, the mapping operation will change the destination table on the server into a Lakehouse table if the table is empty, or the operation will fail if the table is not empty.

Once the Lakehouse table is successfully created, you can load the mapped data into it. See instructions in Loading or Unloading Data into HeatWave Cluster.

8.3 Maximum Number of Tables Loadable into a HeatWave Cluster

The maximum number of tables that can be loaded into a HeatWave Cluster depends on the shape of the DB system and the total number of columns of the tables to be loaded. If the tables have more columns, fewer tables can be loaded. See Table 8-1 for the approximate maximum number of columns and tables for each shape. As an illustration, the table shows the maximum number of tables with an average of 100 columns per table.

> **Tip:**
>
> You can use the HeatWave Autopilot to verify that there is sufficient memory to load the tables into your HeatWave Cluster.

Table 8-1 Limits on the Number of Columns and Tables Loadable into a HeatWave Cluster

DB System Shape	Maximum number of columns (approx.)	Maximum number of tables with an average of 100 columns per table (approx.)
MySQL.2.16GB	81,900	819
MySQL.4.32GB	81,900	819
MySQL.8.64GB	81,900	819
MySQL.16.128GB	81,900	819
MySQL.32.256GB	3,538,900	35,389

Note:

The estimates assume numerical data for the tables. The maximum number of columns and tables that can be loaded is reduced with string columns, because the data dictionary encoding for them incurs more memory.

9 Running Queries

The Workspaces tab in the HeatWave Console provides a Query Editor for running MySQL and HeatWave queries.

> **Note:**
>
> Queries can also be run from a MySQL client. See Connecting from a Client.

To run a query using the Query Editor:

1. Connect to a DB System. For instructions, see Connecting from the Console.
2. Enter a query into the **Query Editor**.
3. Click **Run Query**.

   The query results are displayed in tabular format by default. You can select the **JSON** tab to view results in `JSON` format.

   For multi-statement queries, only the results of the last query are displayed.

   The **Job Information** tab provides the **Job ID**, the number of rows returned, and the query statement that was processed.

Query Editor Limitations

The Query Editor has the following limitations:

- It is not intended for loading data into a DB System. More efficient methods of loading data into a DB System are available. For more information, see Importing Data.
- Only results for the last executed query are displayed.
- Multi-set results from calling stored procedures are not displayed. Only the last result is displayed.
- Results from queries with SQL code comments might not be displayed accurately.
- Long query results might be truncated.
- Running HeatWave Queries

9.1 Running HeatWave Queries

When a HeatWave Cluster is enabled and the data you want to query is loaded in HeatWave, queries that qualify are automatically offloaded from the MySQL DB System to the HeatWave Cluster for accelerated processing. No special action is required. Simply run the query from the Query Editor, as described above, or from a MySQL client that is connected to the DB System.

Before running a query, you can use `EXPLAIN` to determine if the query will be offloaded to HeatWave; for example:

```
mysql> EXPLAIN SELECT O_ORDERPRIORITY, COUNT(*)
          AS ORDER_COUNT FROM tpch.orders
          WHERE O_ORDERDATE >= DATE '1994-03-01'
          GROUP BY O_ORDERPRIORITY
          ORDER BY O_ORDERPRIORITY;
```

If the query can be offloaded to HeatWave, the `Extra` column of `EXPLAIN` output shows `"Using secondary engine RAPID"`. If that information does not appear, the query cannot be offloaded.

For more information about running HeatWave queries, see Running Queries, in the HeatWave User Guide. You can also refer to the *Quickstarts* in the HeatWave User Guide, which show how to import data into a DB System, load data into HeatWave, and run queries:

- tpch Analytics Quickstart
- AirportDB Analytics Quickstart

10
HeatWave AutoML

Use the HeatWave Console to create and evaluate HeatWave AutoML models. For more information about HeatWave AutoML, see HeatWave AutoML.

- HeatWave AutoML Requirements
- Create a HeatWave AutoML model
- Evaluate a HeatWave AutoML model

10.1 HeatWave AutoML Requirements

HeatWave AutoML has the following requirements:

- An operational DB System with sufficient resources and a HeatWave Cluster. See the following:
  - Supported Shapes.
  - Creating a DB System
  - Creating a HeatWave Cluster
- The MySQL account for the user who will train a model does not have a period character (".") in the username. For example, the username `'joesmith'@'domain'` can train a model, but the username `'joe.smith'@'domain'` cannot. For more information about this requirement, see Limitations.
- The MySQL account that will use HeatWave AutoML has been granted the following privileges:
  - `SELECT` and `ALTER` privileges on the schema that contains the machine learning datasets; for example:

    ```
    mysql> GRANT SELECT, ALTER ON schema_name.* TO 'user_name'@'%';
    ```

  - `SELECT` and `EXECUTE` on the MySQL `sys` schema where HeatWave AutoML routines reside; for example:

    ```
    mysql> GRANT SELECT, EXECUTE ON sys.* TO 'user_name'@'%';
    ```

10.2 Create a HeatWave AutoML model

Use the HeatWave Console to create a new HeatWave AutoML model.

Create a HeatWave AutoML model includes an advanced option to choose one or more algorithms. HeatWave AutoML uses the selected algorithms as a pool of algorithms, and will choose the most applicable algorithm from this pool to evaluate the model. By default, all algorithms are selected. To use a single, specific algorithm to evaluate the model, deselect all the other algorithms.

1. Select the **HeatWave AutoML** tab in the HeatWave Console, and then click **Connect to MySQL DB System**.
2. Select a DB System from the drop-down menu.
3. Enter a MySQL user name and password for the DB System.
4. Click **Connect**.
5. Click **Create Model**
6. **Name**: Specify a name for the model or use the generated default name.
7. **Description**: Specify a user friendly description for the model.
8. **Training table**: Select a training dataset.
9. **Columns to include in the model**: Select the **Target column**, and then choose which columns to **Include in model**.
10. **Machine learning task**: Choose a task.
11. **Advanced**:
    - **Optimization metric**: Choose an optimization metric.
    - **Algorithms to evaluate**: Choose which algorithms HeatWave AutoML should consider to evaluate the model.
12. Click **Create**.

    The HeatWave Console returns to the **HeatWave AutoML** page, and shows the new model at the top of the page.

10.3 Evaluate a HeatWave AutoML model

Evaluate a HeatWave AutoML model includes scores for the model, predictions and explanations. What if analysis compares the baseline results to alternative values.

During testing, the model can include a column of known values. After evaluation, if a prediction does not match the known value, HeatWave Console marks the prediction in red.

1. Select the **HeatWave AutoML** tab in the HeatWave Console, and then click **Connect to MySQL DB System**.
2. Select a DB System from the drop-down menu.
3. Enter a MySQL user name and password for the DB System.
4. Click **Connect**.
5. Select a model from the list. The lower pane shows details for the model: **Training table**, **Target column**, **Description**, **Selected algorithm**, **Machine learning task**, and **Training score**.
6. Click **Evaluate**.
7. **Select table**: The prompt has a reminder of the name of the training table. Select a test table compatible with the training table, and click **Next**.

   The **Evaluate Model: *model name*** dialog opens.
8. **Model score**:
    - Select one of the **Scoring metrics**, and click **Calculate score**. The **Results** shows the score.
9. **Explain model**:

- **Feature Importance**: Shows the names of each feature and their relative importance to the model.

10. **Predictions**:
    - Click **Generate Predictions** to create a table of predictions for each row in the test table.
    - Select a prediction row from the table, and click **Explain Prediction**.

      The dialog shows the **Selected row**, and **Feature Importance**. **Notes** explains which feature had the largest impact on the prediction, and might also include the feature that had the largest impact against the prediction.
    - Click **Back** to return to the predictions table.
    - Select a prediction row from the table, and click **What If**.
    - **Values for comparison from included features**: Click the i to show information for that feature column.

      This includes **Minimum**, **Mean** and **Maximum**, and a bar chart of values.
    - Click **Back** to return to the predictions table.
    - Select a prediction row from the table, and click **What If**.
    - **Values for comparison from included features**: Adjust the values, and click **Create**.

      The dialog shows the **Comparison data**, and **Feature Importance**. **Notes** has explanations for the Baseline and the Comparison. Both explain which feature had the largest impact on that prediction, and might also include the feature that had the largest impact against that prediction.
    - Click **Back** to return to the predictions table.

11 System Variables

11.1 System Variables

The following global system variables are specific to HeatWave on AWS and are not available in the on-premise version of MySQL Server.

- `telemetry_log_disable`

Property	Description
Introduced	8.0.32
System Variable	`telemetry_log_disable`
Scope	Global
Dynamic	Yes
SET_VAR Hint Applies	No
Type	Boolean
Default Value	`OFF`

Whether to disable the telemetry log.

12 Events

Statuses of actions and operations on HeatWave on AWS resource are reported as **Events** on various pages of the HeatWave Console. To check for any failed actions or operations, go to the **Events** tabs, where the failures are reported and explained by the error messages.

Operations and Resources Monitored

Table 12-1 shows the operations and resources on HeatWave on AWS for the events, the types of events that are reported, and the locations on HeatWave Console where the **Events** tab (which shows the event reports) can be found.

Table 12-1 Events Reported on the HeatWave Console

Resources and Operations	Types of Actions or Operations that Generate Event Reports	Location of Events Tab
DB System	Create, delete, resize, start, stop, restart, update, upgrade, configuration update	MySQL DB System Details
HeatWave Cluster	Create, delete, start, stop, restart	HeatWave Cluster Details
Inbound Replication Channel	Create, delete, update, resume, reset channel	Channel Details
Backups	Create, update, delete	Backup Details
MySQL Configurations	Create, update, delete	MySQL Configuration Details

Information in an Event Report

The following information is available in each event:

- **Type**: The event and its status (**In Progress**, **Succeeded**, **Failed**). For example, `Backup Delete Succeeded`, `Channel Create In Progress`.
- **Severity**: The severity level of the event, which can be one of
  - `INFO`: General information is provided. No special concerns
  - `WARNING`: Warning on an event that might impact the DB System's operation if no actions are taken.
  - `CRITICAL`: A critical event. An urgent action is required to avoid adverse impact on your DB System.
- **Message**: The message body of the event report. It usually contains the information in **Type** plus the ID of the resource involved. For example, `DB System e7fe1776-0066-4508-a904-ccb87b62ed46 stop in progress`.
- **Created**: The date and time the event report was created.

The following additional information is available for each event in the **Event Details** window, which opens when you click on the event in the **Events** tab:

- **Event ID**: A unique ID of an event.

- The type of resource involved and its resource ID. For example, **MySQL Configuration** followed by **Default MySQL Configuration for MySQL.32.256GB (Supports HeatWave)** .

13
Performance Monitoring

The **Performance** tab in the HeatWave Console provides HeatWave *Cluster* and *Workload* performance data. An active HeatWave Cluster is required for cluster and workload performance monitoring.

- HeatWave Cluster Performance
- Workload Performance
- Autopilot Shape Advisor

13.1 HeatWave Cluster Performance

To view HeatWave Cluster performance data:

1. In the HeatWave Console, select the **Performance** tab.
2. Select a DB System and HeatWave Cluster from the drop-down.

   The **Cluster** tab shows cluster performance data for HeatWave and for the DB System.

   Click the **Refresh** icon to refresh the performance data.
3. To see performance data for another DB System and HeatWave Cluster, select them from the drop-down, and click the **Refresh** icon.

For descriptions of HeatWave Cluster performance data, see HeatWave Cluster Performance Data.

- HeatWave Cluster Performance Data

13.1.1 HeatWave Cluster Performance Data

- **HeatWave**
  - **Cluster Memory Utilization**

    The percentage of total HeatWave Cluster memory that is currently utilized.
  - **Node Memory Utilization**

    The percentage of memory utilized on each cluster node. Select from the drop-down menu to view data in order of node **ID** or **Memory Utilization**.
  - **Data Dictionary**

    The total data dictionary size. Dictionaries for dictionary-encoded string columns are stored on the MySQL DB System node. For more information, see String Column Encoding Reference, in the *HeatWave User Guide*.
  - **Dataset**

    The size of the dataset loaded on the HeatWave Cluster.
- **MySQL**
  - **CPU Utilization**

The percentage of CPU utilization on the MySQL DB System.

- **Memory Utilization**

  The percentage of memory utilization on the MySQL DB System,

- **Buffer Pool**

  The buffer pool size on the MySQL DB System.

- **Disk Operations**

  The number of disk operations on the MySQL DB System (write, read, and miscellaneous).

- **Connections**

  The number of idle and active client connections on the MySQL DB System.

13.2 Workload Performance

To view HeatWave workload performance data:

1. On the HeatWave Console, select the **Performance** tab.
2. Select a DB System and HeatWave Cluster from the drop-down.

   The **Workload** tab shows workload performance data for HeatWave and for the MySQL node.

   Click the **Refresh** icon to refresh the performance data.
3. To see workload performance data for another DB System and HeatWave Cluster, select them from the drop-down, and click the **Refresh** icon.

For HeatWave workload performance data descriptions, see HeatWave Workload Performance Data.

- HeatWave Workload Performance Data

13.2.1 HeatWave Workload Performance Data

Information is shown for the last 1000 query executions.

- **Executions**

  The number of query executions on a time scale.

- **Recent Queries**

  Displays recently executed queries with the **Query Text**, the query **Start Time**, and the **Duration (ms)** of the query.

  You are provided with options to **Search by Query Text** for specific queries and to **Aggregate Executions** to view aggregated **Start Time** and **Duration (ms)** data for the same query.

13.3 Autopilot Shape Advisor

Use the Auto Shape Prediction feature in HeatWave Autopilot for HeatWave on AWS to analyze the workload and assess the suitability of the current MySQL shape.

> **Note:**
>
> The Auto Shape Prediction feature is available prior to MySQL 8.0.32, but is enabled by default only for MySQL 8.0.32 and later.

The Auto Shape Prediction feature begins to collect MySQL statistics that reflect the current workload. It collects statistics at varying intervals, and Auto Shape Prediction creates a prediction every five minutes while it is active. If there is insufficient or no activities in a five-minute interval, or if buffer pool usage is growing, Auto Shape Prediction cannot make a prediction for that interval.

The Auto Shape Prediction looks at buffer pool usage, workload activity, access patterns, and CPU statistics to estimate the required buffer pool (for MySQL version 8.0.32 and later) and CPU sizes (for MySQL 8.2.0-u2 and later). The system administrator should then make sure the DB system shape can accommodate the recommended buffer pool size and number of CPU cores. If the current buffer pool or CPU size is smaller than the predicted requirements, although MySQL can usually manage the workload while keeping the system stable, the performance of the DB System might suffer due to elevated disk I/O or insufficient CPU cores for processing the workload.

> **Tip:**
>
> **For DB systems using MySQL versions earlier than 8.2.0-u2**: If Auto Shape Prediction suggests a possible downsize, consider the DB System CPU usage and memory usage before downsizing. For heavy CPU utilization, downsizing to a shape with fewer CPUs is not recommended.

When Auto Shape Prediction is running, it keeps a rotating history of the collected feature data for seven days. While there is a monitoring overhead from periodic statistics collection and prediction events, for most workloads and shape combinations the overhead is negligible. If necessary, it is possible to disable Auto Shape Prediction.

When Auto Shape Prediction is disabled it clears the internal statistics tables, but keeps the past predictions.

An upgrade of the DB System drops and re-installs the `mysql_autopilot` schema, which removes the past predictions.

For MySQL 8.0.32 and later, there are two ways to access the Autopilot Shape Advisor: with the HeatWave Console or with a MySQL client. Prior to MySQL 8.0.32, use a MySQL client.

> **Note:**
>
> The SQL output by a MySQL client can only provide some hints. The HeatWave Console can provide detailed and graphical information, and can recommend an improved shape by its specific shape name. See: Autopilot Shape Advisor with HeatWave Console .

- Autopilot Shape Advisor with HeatWave Console
- Autopilot Shape Advisor with a MySQL Client

13.3.1 Autopilot Shape Advisor with HeatWave Console

To view the Autopilot Shape Advisor:

1. In the HeatWave Console, select the **Performance** tab.
2. Select a DB System and HeatWave Cluster from the drop-down list.

   The **Autopilot Shape Advisor** tab shows the Auto Shape Prediction data for the DB System.

   Click the **Refresh** icon beside the selected DB system to refresh the performance data.
3. To see performance data for another DB System and HeatWave Cluster, select them from the drop-down list.

For descriptions of Auto Shape Prediction data, see:

- Auto Shape Prediction Data

13.3.1.1 Auto Shape Prediction Data

The **Autopilot Shape Advisor** tab shows data for the last seven days and contains the following information:

- **Summary**

  This summarizes the basic information of your DB system and HeatWave cluster including their names, their states, and the shape and size of the cluster.
- **Recommendation**

  This section contains the **MySQL Shape Prediction** and the recommended **Action**. You need to select a time period in the drop-down list on the right, upon which the prediction will be based. The choices in the drop-down list include:

  – **Past 12 hours**

  – **Past day**

  – **Past 3 days**

  – **Past week**

  The time period can also be chosen by using the horizontal slider controls at the bottom of the page.

  MySQL Shape Prediction

  – **Current**

    The current shape size, with **Shape Name**, **Memory Size**, **CPU Count**, and **Buffer Pool Size**.

  – **Recommended**

    The recommended shape size, with **Shape Name**, **Memory Size**, **CPU Count**, and **Buffer Pool Size**.

  Action

  The prediction is based on the selected time period, and there are five possible cases:

  – **Upsize**: Move the DB System to a larger MySQL shape.

  – **Downsize**: Move the DB System to a smaller MySQL shape.

- **No Change**: No need to change the MySQL shape.
- **Not Enough Data**: There is not enough workload activity, or Auto Shape Prediction cannot make a prediction.
- **Prediction Data Not Available**: Auto Shape Prediction is disabled, or it is enabled but has not produced the first prediction, or the HeatWave Console data has not been refreshed within the last 5 minutes.

After the recommendation, **Steps to take** provides the instructions to follow for adopting the recommended shape.

- **Shape Prediction Insights**

  This sections contains the following information:

  - **Average Statistics**
    - **Buffer Pool**

      A graph that shows the **Average Buffer Pool Usage** and **Current Buffer Pool Size**.
    - **Buffer Pool Hit Rate**

      A percentage value calculated for the selected time period.
    - **CPU Utilization**

      Average CPU utilization calculated for the selected time period.
  - **Recent Statistics**

    Use the left and right slider controls to adjust the time period for these time graphs:
    - **Buffer Pool**

      A detailed graph that shows the **Buffer Pool Size** and **Recommended Buffer Pool Size** for the selected time period.
    - **CPU Count**

      A detailed graph that shows the CPU count predictions for downsize cases for the selected time period.
    - **Buffer Pool Hit Rate**

      A detailed graph that shows the **Buffer Pool Hit Rate** for the selected time period.
    - **CPU Utilization**

      A detailed graph that shows the history of **CPU Utilization** for the selected time period.

    Workload markers on the time series graphs indicate recent activities:
    - Green markers: The workload is running and stable. This allows good predictions.
    - Orange markers: The workload is running, but is not stable. This makes predictions difficult.
    - No markers: No recent activities, and no predictions.

13.3.2 Autopilot Shape Advisor with a MySQL Client

Auto Shape Prediction records predictions in the `shape_predictions` table in the `mysql_autopilot` schema on the MySQL server. This is a system schema that is always

present even if Auto Shape Prediction is not active. The schema also contains tables for the statistics used to calculate the predictions.

> **Note:**
>
> The SQL output can only provide some hints. The HeatWave Console provides much more information and can recommend an improved shape. See: Autopilot Shape Advisor with HeatWave Console .

This example shows output from the `shape_predictions` table with predictions made over the course of 45 minutes:

```
mysql> SELECT * FROM mysql_autopilot.shape_predictions LIMIT 10;
+----------------------------+------------+-------------+---------------
+-----------------+---------------------+--------------------------
+-----------------------------------+
| prediction_time            | bp_size_gb | bp_hit_rate | cpu_core_count |
cpu_utilization | predicted_bp_size_gb | predicted_cpu_core_count |
prediction_outcome                |
+----------------------------+------------+-------------+---------------
+-----------------+---------------------+--------------------------
+-----------------------------------+
| 2023-12-11 14:34:43.566972 |         43 |        NULL |              8
|       0.0188317 |                NULL |                         NULL | EMPTY
FEATURE DATA                |
| 2023-12-11 14:39:43.630568 |         43 |        NULL |              8
|        0.809462 |                NULL |                         NULL | NOT
ENOUGH FEATURE DATA SNAPSHOTS |
| 2023-12-11 14:44:43.682870 |         43 |           1 |              8
|        0.917938 |                NULL |                         NULL | FEATURE
DATA IS NOT STABLE        |
| 2023-12-11 14:49:43.691465 |         43 |           1 |              8
|        0.517613 |                NULL |                         NULL | FEATURE
DATA IS NOT STABLE        |
| 2023-12-11 14:54:43.707952 |         43 |       0.875 |              8
|       0.0378884 |                NULL |                         NULL | FEATURE
DATA IS NOT STABLE        |
| 2023-12-11 14:59:43.724345 |         43 |        NULL |              8
|      0.00403875 |                NULL |                         NULL | LOW
ACTIVITY                    |
| 2023-12-11 15:04:43.734345 |         43 |           1 |              8
|        0.120723 |                12.3 |                         4    |
DOWNSIZE                          |
| 2023-12-11 15:09:43.744345 |         43 |           1 |              8
|        0.110723 |                12.5 |                         4    |
DOWNSIZE                          |
| 2023-12-11 15:14:43.754345 |         43 |           1 |              8
|        0.120723 |                12.4 |                         4    |
DOWNSIZE                          |
| 2023-12-11 15:19:43.764345 |         43 |           1 |              8
|        0.130723 |                12.2 |                         4    |
DOWNSIZE                          |
+----------------------------+------------+-------------+---------------
```

```
+----------------+--------------------+-------------------------
+-----------------------------------+
```

The `shape_predictions` table has these columns:

- `prediction_time`: The timestamp for the prediction. A prediction is attempted every five minutes.
- `bp_size_gb`: The current buffer pool size in GB.
- `bp_hit_rate`: The current buffer pool hit rate.
- `cpu_core_count`: The number of current available CPU cores.
- `cpu_utilization`: The CPU utilization (on the scale of 0 to 1, with 1 meaning full utilization).
- `predicted_bp_size_gb`: The predicted buffer pool size for optimal performance with this workload in GB.
- `predicted_cpu_core_count`: The predicted CPU core count.
- `prediction_outcome`: A recommendation to upsize, downsize, or stay with your current shape, or information on why a prediction cannot be made for that interval.

To use Auto Shape Prediction with a MySQL client, follow these steps:

1. Connect to the DB System for the HeatWave Cluster using the MySQL client (`mysql`) or MySQL Shell in SQL mode. For instructions to connect to HeatWave on AWS using a client, see Connecting from a Client.
2. While a typical workload is running, enable Auto Shape Prediction by issuing the following statement using the client (the feature is enabled by default for MySQL 8.0.32 and later):

   ```
   mysql> CALL mysql_autopilot.shape_prediction(JSON_OBJECT("enable", TRUE));
   ```

3. Wait at least five minutes for the first prediction to be attempted, then start to check the results with a statement like the following on the SQL client:

   ```
   mysql> SELECT * FROM mysql_autopilot.shape_predictions
             ORDER BY prediction_time DESC LIMIT 20;
   ```

   Auto Shape Prediction attempts a new prediction every five minutes. Re-issue the statement every so often while a typical workload is running, until the predictions have stabilized.
4. When the predictions have stabilized, make a note of the maximum values of `predicted_bp_size_gb` and `predicted_cpu_core_count`, and the associated actions suggested in `prediction_outcome`.
5. Leave Auto Shape Prediction running to monitor other workloads, or disable Auto Shape Prediction with this statement:

   ```
   mysql> CALL mysql_autopilot.shape_prediction(JSON_OBJECT("enable", FALSE));
   ```

6. To move to a different DB System shape, follow the steps in Creating a Backup and Restoring a Backup to a New DB System.

14 Backups

This section describes how to create, restore, and manage MySQL DB System backups.

DB System backups are Amazon EBS snapshots which are automatically saved to Amazon Simple Storage Service (Amazon S3). HeatWave on AWS limits the maximum number of backups that you can retain across all DB System instances. If you reach the limit, remove the oldest backups before creating new ones.

It is recommended to create backups on a regular schedule. Backups are incremental, but if the period between backups is too long, a backup operation may take a long time to complete due to the volume of changes.

14.1 Creating a Backup

Backups can be created automatically or manually.

For how to enable automatic backups for a DB system, see Creating a DB System and Editing a DB System.

Use the Console to create a manual backup:

1. In the HeatWave Console, click the **HeatWave MySQL** tab.
2. On the **DB Systems** tab, in the list of DB Systems, find the DB System you want to create a backup for, and do one of the following:
   - Click the row of the DB System to highlight it, and click **Create Backup** from the **Actions** menu.
   - Click the name of the DB System to open the **Details** page and click **Create Backup** from the **Actions** menu.
3. In the **Create MySQL Backup** panel, edit the following fields:
   - **Display name**: Specify a name of the backup. If you do not define a name, one is generated for you in the format `NameOfTheBackup - Backup - Day, Date Time`.
   - **Description**: Specify a description of the backup. If you do not provide a description, one is generated for you in the format `NameOfTheBackup - Manual Backup - Day, Date Time`.
   - **Backup Retention Policy**: Specify how may days you want to retain the user initiated backup. By default, the backups are retained for 365 days.
4. Click **Create** to create the backup.

Under **MySQL**, click the **Backups** tab to view the backup you created. When the backup is created successfully, the backup state changes from **Creating** to **Active**.

14.2 Editing a Backup

Use the Console to edit the display name, description, and backup retention policy of a DB system backup.

1. In the HeatWave Console, click the **HeatWave MySQL** tab.
2. On the **Backups** tab, in the list of DB Systems, find the backup you want to edit, and do one of the following:
   - Click the row of the backup to highlight it, and click **Edit Backup**.
   - Click the name of the backup to open the **Details** page and click **Edit Backup**.
3. In the **Edit MySQL Backup** panel, edit the following:
   - **Display Name**: Specify a name of the backup. If you do not define a name, one is generated for you in the format, `NameOfTheBackup - Backup - Day, Date Time`.
   - **Description**: Specify a description of the backup. If you do not provide a description, one is generated for you in the format, `NameOfTheBackup - Manual Backup - Day, Date Time`.
   - **Backup Retention Policy**: (Only for user initiated backups) Specify how may days you want to retain the user initiated backup. By default, the backups are retained for 365 days.
4. Click **Save**.

The details of the selected backup is updated.

14.3 Viewing Backup Details

To view backup details:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. Select the **Backups** tab. In the list of backups, find the backup you want to view details for and click the name of the backup to open the **MySQL Backup Details** page.

For descriptions of MySQL backup details, see Backup Details.

- Backup Details
  Use the Console to view the **Backup Details** page.

14.3.1 Backup Details

Use the Console to view the **Backup Details** page.

Table 14-1 Backup Details Page

Name	Description
Summary	Backup summary. See Table 14-2.
General information	General backup. See Table 14-3.
MySQL DB System snapshot	MySQL DB System snapshot. See Table 14-4.
Backup Event Details	Backup event details. See Table 14-5.

Table 14-2 Backup Summary

Field	Description
State	The state of the backup. • **CREATING**: Resources are being reserved for the backup, and the backup is being created. Creating a backup can take several minutes. The backup is not ready to use yet. • **ACTIVE**: The backup was successfully created • **UPDATING**: The backup is in the process updating • **INACTIVE**: The backup is unavailable • **DELETING**: The backup is being deleted by a delete action in the Console • **DELETED**: The backup has been deleted and is no longer available • **FAILED**: An error condition prevented the creation or continued availability of the backup
Resource ID	The unique resource identifier assigned to the backup
Storage size	The size of the backup

Table 14-3 General information

Field	Description
Description	A description of the backup
Created	The date and time the backup was created
Last Updated	The date and time the backup was last updated
Retention in days	The number of days the user initiated backup is retained
Expiry time	The date and time when the backup expires
Backup type	The type of the backup: User initiated or scheduled

Table 14-4 MySQL DB System snapshot

Field	Description
Name	The name of the DB System used to create the backup
Description	The user-specified description of the DB System used to create the backup
MySQL version	The MySQL version used to create the backup
Shape	The resource template applied to the DB System used to create the backup
MySQL Configuration	The configuration of the DB System
Availability Zone	The physical Availability Zone of the DB System used to create the backup
Maintenance window	The time when the two-hour maintenance window begins for the DB System used to create the backup
Automatic backups	The status of the automatic backups: Enabled or Disabled

Table 14-5 Backup Event Details

Field	Description
Type	The event and its status.
Severity	The severity level of the event, which can be one of • `INFO`: General information is provided. No special concerns • `WARNING`: Warning on an event that might impact the DB System's operation if no actions are taken. • `CRITICAL`: A critical event. An urgent action is required to avoid adverse impact on your DB System.
Message	The message body of the event report.
Created	The date and time the event report was created.

See Events for details on event reports.

14.4 Restoring a Backup to a New DB System

When you restore a backup, you create a new DB System and restore the backup to it. You can change the shape and the amount of data storage for the new DB System.

A restored DB System uses the same MySQL Administrator user name and password that was in effect when the backup was created.

DB System backups are Amazon EBS snapshots which are automatically saved to Amazon Simple Storage Service (Amazon S3).

> **Tip:**
>
> Amazon EBS snapshots are loaded asynchronously. You can use the new DB System as soon as it is active, while data continues to load in the background. If you access data that has not been loaded yet, the DB System downloads the requested data (which can therefore take longer than usual to access), then resumes asynchronous loading. If you want to download your frequently accessed data immediately, run a full-table scan such as `SELECT *` on the relevant tables.

To restore a backup to a new DB System:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. Select the **Backups** tab. In the list of backups, find the backup you want to restore to a new DB System, and do one of the following:
   - Click on the row of the backup to highlight it, and click **Restore Backup**.
   - Click the name of the backup to open the **MySQL Backup Details** page. Click **Restore Backup**.

   The **Restore Backup to new MySQL DB System** dialog is displayed.
3. Provide basic information for the new DB System:

- **Display Name**: Specify a display name for the new DB System or use the generated default name.
- **Description**: Specify a description for the new DB System or use the generated description.

> **Note:**
>
> A restored DB System uses the same MySQL Administrator user name and password that was in effect when the backup was created.

4. Select the **Hardware shape**. If you are using this process to get a different shape or more storage for the DB System, choose the shape that you want the new DB System to have.
   - **Shape**: Select the shape to use for your DB System. The shape determines the resources allocated to the system. For information about shapes, see Supported Shapes. Selecting a shape for a DB System also selects the configuration associated with that shape. See Configuration.
   - **Data Storage Size (GiB)**: Specify the amount of block storage, in GiB, to allocate to the DB System. This block storage stores all data, logs, and temporary files. Binaries are not stored in this block storage. Enter a number between 50 and 16384.

     > **Note:**
     >
     > Ensure that the specified Data Storage Size is greater than or equal to the size of the backup you are restoring.

   - **Database Version**: Select the MySQL Server version you want to deploy. The latest MySQL Server version is selected by default.
5. Configure the **Availability zone**, which determines the physical location of the DB System:
   - Select **Automatically** to have the physical AWS Availability Zone selected for you.
   - Set **Manually** to select the physical AWS Availability Zone where the MySQL DB System will be created.

   > **Note:**
   >
   > A physical Availability Zone is identified by an Availability Zone ID (AZ ID). For information about AZ IDs and how to view them, see Availability Zone IDs for your AWS resources, in the *AWS RAM User Guide*.

6. Configure **Networking** settings:
   - **Allowed Client Addresses**: Specify public-facing client IPv4 addresses that are permitted to connect to the DB System endpoint. Addresses are specified in CIDR format; for example: `1.2.3.4/24`. Multiple addresses in CIDR format can be specified in a semicolon-separated list; for example: `1.2.3.4/24; 1.2.3.4/32`. For information about specifying IP addresses in CIDR format, see Creating a DB System.
   - **Port**: The port on which the MySQL server listens. The default is port 3306. You can specify a port number between 1024 and 65535.

- **X Protocol Port**: The X Protocol port on which the MySQL server listens, supported by clients such as MySQL Shell. The default port is 33060. You can specify a port number between 1024 and 65535.

7. Click **Next** to proceed to creating a HeatWave Cluster.
8. Provide basic information for the new HeatWave Cluster:
   - **Display Name**: Specify a display name for the new HeatWave Cluster or use the generated default name.
   - **Description**: Specify a description for the new HeatWave Cluster.
9. Under **HeatWave Cluster Configuration**, select a node shape and number of nodes.
   - **Shape**: Select a HeatWave node shape. For information about supported shapes, see Supported Shapes.

     > **Note:**
     >
     > If you intend to use HeatWave AutoML functionality, the `HeatWave.256GB` node shape is recommended when creating a HeatWave Cluster. The `HeatWave.16GB` node shape may not have enough memory for training on large data sets. If you see error messages about this (such as `ML003024`), use the larger shape instead.

   - **Cluster Size**: The number of HeatWave nodes to create. Enter a number between 1 and 128.

     > **Tip:**
     >
     > Because the new DB System does not yet contain any data, you cannot use HeatWave Autopilot to estimate the required cluster size (as described in Estimating Cluster Size with HeatWave Autopilot). If you want to make an estimate, after loading the data to the DB System, you can delete the HeatWave Cluster and create a new one following the instructions in Creating a HeatWave Cluster.

10. Click **Restore** to restore the backup to the new DB System.

    You are returned to the DB Systems page where you can monitor the state of the operation, which may take some time to complete. The state will change from **Creating** to **Active** when the operation has completed successfully.

14.5 Deleting a Backup

Deleting a backup permanently deletes it.

To delete a backup:

1. In the HeatWave Console, select the **HeatWave MySQL** tab.
2. Select the **Backups** tab. In the list of backups, find the backup you want to restore to a new DB System, and do one of the following:
   - Click on the row of the backup to highlight it, and click **Delete**.
   - Click the name of the backup to open the **MySQL Backup Details** page. Click **Delete**.

The **Delete MySQL DB System Backup** dialog is displayed for you to confirm the deletion.

3. Click **Delete MySQL Backup** to go ahead with the deletion.

14.6 Billing and Free Quota for Backup Storage

The backup storage billing depends on the DB system state and the data storage size that you defined while creating a DB system or updating the storage size. You can see a DB System's state and data storage size by Viewing DB System Details.

The way you are billed for backup storage depends on the **DB System State**:

- **Active/Inactive**: You are billed only when the backups' total size exceeds the free quota described below.
- **Failed/Deleted**: There is no free storage available and you are billed for all backup storage.

Free Quota for DB Systems

You get free storage when the DB system is in the **Active** or **Inactive** state. The size of the free storage equals to the data storage size of the DB system. For example: For a DB system with 500 GB of storage size, the free quota for backup storage is 500 GB.

Backups

Let us assume you have two active DB systems with data storage size of 50 GB and 100 GB in the tenancy in region 1. You are entitled to a maximum of 150 GB free backup storage, which is the sum of the data storage size of the two DB systems in region 1.

Now you created 4 manual backups:

- Backup 1 - 30 GB
- Backup 2 - 50 GB
- Backup 3 - 80 GB
- Backup 4 - 85 GB

The total manual backup size is 30 + 50 + 80 + 85 GB = 245 GB. Let us assume the automatic backups have a total size of 50 GB.

You are billed for region 1 as follows:

Billed backup = Total backup size including manual and automatic - free storage, which is (245+50) - 150 = 145 GB.

> **Note:**
>
> If you have tenancies in two regions, each region is billed separately following the billing method as described above.

15
Configuration

The MySQL configuration comprises global system variables and session variables, which define its operation.

15.1 Creating a MySQL Configuration

Use the HeatWave Console to create a new MySQL configuration.

> **Note:**
>
> It is not possible to change the values of any variables once a configuration has been created. However, you can create a custom configuration by copying and modifying an existing configuration. You can then update your DB system's configuration using your custom configuration.

Follow these steps to create your MySQL configuration:

1. In the HeatWave Console, select the **HeatWave MySQL** tab, and then click **Configurations** to open the **Configurations** page.
2. On the **Configurations** page, click **Create MySQL Configuration** to open the **Create MySQL Configuration** dialog.
3. On the **General Information** page of the **Create MySQL Configuration** dialog, provide the following information:
   - **Basic information**:
     - **Display Name**: Specify a display name for the configuration or use the generated default name.
     - **Description**: Specify a simple description for the configuration.
   - **Select MySQL shape**: Select a MySQL shape to create a configuration for.
4. Click **Next**. On the **Configuration Variables** page of the dialog, provide the following information:

- **Support HeatWave**: Select this option to ensure that this MySQL Configuration supports HeatWave.
- **Initialization variables**: Set preferred values for (and only for) initialization variables whose values you want to be different from the default setting.
  - Click **Select a variable name**, and select the variable from the drop-down list.
  - Click **Select a value**, and select the value from the drop-down list, or enter the preferred value. If no value is selected or entered, the value in the base configuration you are copying (shown in the value box) is used.
  - Click **X** to remove a variable you have selected by the steps above, if you change your mind and no longer want to set a value for it.
  - Click the information icon to show the default value of the variable you have selected. A **Learn More** link for more information on the selected variable also appears.
  - Click the setting icon to reset a selected variable to its default value, if it has been set to something else.
- **User variables**: Set preferred values for (and only for) user variables whose values you want to be different from the default settings.
  - Click **Select a variable name**, and select the variable from the drop-down list.
  - Click **Select a value**, and select the value from the drop-down list, or enter the preferred value. If no value is selected or entered, the value in the base configuration you are copying (shown in the value box) is used. For any variables of the string data type, an empty value field means an empty string, except for nullable variables such as `innodb_ft_server_stopword_table` , for which an empty value field is taken as `MYSQL NULL` (i.e., a null value).
  - Click **Add new variable** to set more variables with the steps above, until you specified all the variables you want to.
  - Click **X** to remove a variable you have selected or added by the steps above, if you change your mind and no longer want to set a value for it.
  - Click the information icon to show the default value of the variable you have selected. A **Learn More** link for more information on the selected variable also appears.
  - Click the setting icon to reset a selected variable to its default value, if it has been set to something else.

5. Click **Next**. On the **Review Configuration** page of the dialog, review all the variable values you have selected in Step 4 above.

   Select a filter to show the variables and their values associated with the configuration:

   - **All**: Show all configuration variables (default)
   - **Dynamic**: Show all configuration variables that are dynamic and can be set at runtime.
   - **Shape or feature specific**: Show all configuration variables that have MySQL shape or feature-specific values.
   - **Default**: Show all configuration variables that have not been set by users.
   - **User-defined**: Show all configuration variables that have been set by users. User-defined variables are those that have ever been set in the dialog when the configuration was created. The actual value of the variable does not matter: for example, you could have selected a variable in the configuration dialog and then left its

displayed value (which was the value from your base configuration) unchanged, but the variable would still be taken as "user-defined" and displayed by this filter.

You can also refine the list by typing into the search bar part of the names of the variables you are interested in.

Click **Back** to go back and edit any variable values if needed.

Click **X** to remove a variable that you have added in the Step 4.

6. Click **Create** after you are happy with your configuration.

   HeatWave MySQL checks the values for each variable. If any values fail validation, the message `Could not create MySQL Configuration` appears together with the names of the failed variables. Correct any invalid values, and click **Create** again.

   The HeatWave Console returns to the **Configurations** page and shows the new configuration at the top of the configuration list.

To review the default values for the MySQL global system variables and choose you preferred values, see the related links below. Notice that certain MySQL global system variables define whether a configuration can support HeatWave.

Related Topics

- System Initialization Variables
- User-Configurable System Variables
- User-Configurable Shape-Dependent System Variables
- View MySQL Configuration Details
- Update MySQL Configuration

15.2 Copying a MySQL Configuration

While it is not possible to change the values of any variables once a configuration has been created, you can create a custom configuration by copying and modifying an existing configuration. You can then update your DB system's configuration using your custom configuration.

Follow these steps to copy a MySQL configuration and create a custom configuration out of it:

1. In the HeatWave Console, select the **HeatWave MySQL** tab, and then click **Configurations** to open the **Configurations** page.
2. On the **Configurations** page, select a base configuration you want to copy and then click **Copy MySQL Configuration** to open the **Copy MySQL Configuration** dialog.
3. On the **General Information** page of the **Copy MySQL Configuration** dialog, provide the following information:
   - **Basic information**:
     - **Display Name**: Specify a display name for the configuration or use the generated default name.
     - **Description**: Specify a simple description for the configuration.
   - **Source MySQL Configuration shape**: The MySQL shape of the configuration you are copying is displayed (it cannot be changed).
4. Click **Next**. On the **Configuration Variables** page of the dialog, provide the following information:

- **Support HeatWave**: Select this option to ensure that this MySQL Configuration supports HeatWave.
- **Initialization variables**: Set preferred values for (and only for) any initialization variables whose values you want to be different from the base configuration you are copying:
  - Click **Select a variable name**, and select the variable from the drop-down list.
  - Click **Select a value**, and select the value from the drop-down list, or enter the preferred value. If no value is selected or entered, the value in the base configuration you are copying (shown in the value box) is used.
- **User variables**: Set preferred values for (and only for) any user variables whose values you want to be different from the base configuration you are copying.
  - Click **Select a variable name**, and select the variable from the drop-down list.
  - Click **Select a value**, and select the value from the drop-down list, or enter the preferred value. If no value is selected or entered, the value in the base configuration you are copying (shown in the value box) is used. For any variables of the string data type, an empty value field means an empty string, except for nullable variables such as `innodb_ft_server_stopword_table` , for which an empty value field is taken as `MYSQL NULL` (i.e., a null value).
  - Click **Add new variable** to set more variables with the steps above, until you specified all the variables you want to.
  - Click **X** to remove a variable you have selected or added by the steps above, if you change your mind and no longer want to set a value for it.

> **Note:**
>
> User variables in the base configuration you are copying are pre-selected here and pre-populated with their values in the base configuration. They are not removable by the **X** button, and will remain as user variables in the new custom configuration you are creating.

5. Click **Next**. On the **Review Configuration** page of the dialog, review all the variable values you have selected in Step 4 above.

   Select a filter to show the variables and their values associated with the configuration:

   - **All**: Show all configuration variables (default)
   - **Dynamic**: Show all configuration variables that are dynamic and can be set at runtime.
   - **Shape or feature specific**: Show all configuration variables that have MySQL-shape-specific or feature-specific values.
   - **Default**: Show all configuration variables that have not been set by users.
   - **User-defined**: Show all configuration variables that have been set by users. User-defined variables are those that have ever been set in a configuration dialog when the configuration or its base configuration was created. The actual value of the variable does not matter: for example, you could have selected a variable in the configuration dialog and then left its displayed value (which was the value from your base configuration) unchanged, but the variable would still be taken as "user-defined" and displayed by this filter.

   You can also refine the list by typing into the search bar part of the names of the variables you are interested in.

Click **Back** to go back and edit any variable values if needed.

Click **X** to remove a variable that you have added in the Step 4.

6. Click **Copy** after you are happy with your configuration.

   HeatWave MySQL checks the values for each variable. If any values fail validation, the message `Could not create MySQL Configuration` appears together with the names of the failed variables. Correct any invalid values, and click **Copy** again.

   The HeatWave Console returns to the **Configurations** page and shows the new configuration at the top of the configuration list.

To review the default values for the MySQL global system variables and choose you preferred values, see the related links below. Notice that certain MySQL global system variables define whether a configuration can support HeatWave.

Related Topics

- System Initialization Variables
- User-Configurable System Variables
- User-Configurable Shape-Dependent System Variables
- View MySQL Configuration Details

15.3 MySQL Configuration Details

Follow these steps to view details of your MySQL configurations:

1. In the HeatWave Console, select the **Configurations** tab.
2. In the list of configurations, find the configuration for which you want to view its details, and do one of the following:
   - Click the row of the configuration to highlight it. The configuration details appear below the list of configurations. See MySQL Configuration Details Page for an explanation of the details given.
   - Click the name of the configuration to open the MySQL Configuration Details Page for the selected DB System.

The following actions can be taken for the configuration on either the HeatWave Console **Configurations** tab or on the **MySQL Configuration Details** page:

- Click **Edit Configuration** to open the **Edit MySQL Configuration** dialog box, in which you can edit the **Basic information** for the configuration including the **Display name** and **Description**. Note that no other properties of the configuration can be altered.
- Click **Copy Configuration** to create a new configuration based on the current configuration. After the **Copy MySQL Configuration** dialog opens, follow the instructions given in Copying a MySQL Configuration.
- Click **Delete** to delete the configuration if it is no longer needed by any DB Systems.

The following tables describe the information provided on the **MySQL Configuration Details** page.

Table 15-1 MySQL Configuration Details Page

Name	Description
Summary	Configuration summary details. See Table 15-2.

Table 15-1 (Cont.) MySQL Configuration Details Page

Name	Description
Variables	Variable information. See Table 15-3.
DB Systems	Information on DB Systems that use the configuration. See Table 15-4 .
Events	Event details. See Table 15-5.

Table 15-2 MySQL Configuration Summary Details

Field	Description
State	The state of the configuration: • **ACTIVE**: The configuration was successfully created and is ready to be used. • **DELETED**: The configuration has been deleted and is no longer available.
Resource ID	The resource ID assigned by HeatWave MySQL to the configuration.
Type	The type of configuration: • **Default**: It is a default configuration for a MySQL shape. • **Custom** It is a custom configuration created based on a **Default** configuration or another custom configuration.
Source MySQL Configuration	The source configuration based on which the current configuration was created, if this is a custom configuration.
Supports HeatWave	Whether the configuration supports HeatWave.
Shape	The MySQL shape for which the configuration was created. For information about shapes, see Supported Shapes.
Description	The description that was provided when the configuration was created.
Last updated	The date and time the configuration was last updated. If the configuration has never been updated, it shows the date of time of its creation.
Created	The date and time the configuration was created.

Table 15-3 MySQL Configuration Variables Details

Field	Description
Initialization variables	All the System Initialization Variables in the configuration.

Table 15-3 (Cont.) MySQL Configuration Variables Details

Field	Description
User variables	Select a filter to show the variables and their values associated with the configuration: • **All**: Show all configuration variables (default) • **Dynamic**: Show all configuration variables that are dynamic and can be set at runtime. • **Shape or feature specific**: Show all configuration variables that have MySQL shape or feature-specific values. • **Default**: Show all configuration variables that have not been set by users. • **User-defined**: Show all configuration variables that have been set by users. User-defined variables are those that have ever been set in either the **Create MySQL Configuration** or the **Copy MySQL Configuration** dialog when the configuration was created. The actual value of the variable does not matter: for example, you could have selected a variable in the configuration dialog and then left its displayed value (which was the value from your base configuration) unchanged, but the variable would still be taken as "user-defined" and displayed by this filter. You can also refine the list by typing into the search bar part of the names of the variables you are interested in.

Table 15-4 MySQL Configuration DB Systems Details

Field	Description
Name	Name of a DB System that uses the configuration. Click the name to see the MySQL DB System Details.
State	The state of the DB System that uses the configuration. See Table 3-5 for descriptions of the DB System states.
HeatWave Cluster	The HeatWave Cluster for the DB System that uses the configuration. Click the name to view the HeatWave Cluster Details.
HeatWave State	The state of the HeatWave Cluster of the DB System that uses the configuration. See Table 4-3 for descriptions of the HeatWave Cluster states.
Created	The date and time the DB System that uses the configuration was created.

Table 15-5 MySQL Configuration Event Details

Field	Description
Type	The event and its status.
Severity	The severity level of the event, which can be one of • `INFO`: General information is provided. No special concerns • `WARNING`: Warning on an event that might impact the DB System's operation if no actions are taken. • `CRITICAL`: A critical event. An urgent action is required to avoid adverse impact on your DB System.
Message	The message body of the event report.
Created	The date and time the event report was created.

See Events for details on event reports.

15.4 System Initialization Variables

System initialization variables are global system variables that must be set during DB system initialization. They apply for the life span of the DB system and, once applied, you cannot change them.

Configure the following variables with the HeatWave Console. See Creating a MySQL Configuration and Copying a MySQL Configuration.

> **Note:**
>
> Any change to the configuration of a DB system cannot change the system initialization variables.

Table 15-6 Initialization Variables

Name	Value
lower_case_table_names	• CASE_SENSITIVE: (Default) Table and schema name comparisons are case-sensitive and are stored as you specify them. Selecting CASE_SENSITIVE sets the system variable, lower_case_table_names, to 0. • CASE_INSENSITIVE_LOWERCASE: Table and schema name comparisons are not case-sensitive and stored in lowercase. SelectingCASE_INSENSITIVE_LOWERCASE sets the system variable, lower_case_table_names, to 1.

15.5 User-Configurable System Variables

Configure the following global system variables with the HeatWave Console. See Creating a MySQL Configuration and Copying a MySQL Configuration.

Table 15-7 User-Configurable System Variables

Name	Default Value
audit_log_disable	ON
autocommit	ON
big_tables	OFF
binlog_expire_logs_seconds	3600
binlog-row-metadata	MINIMAL
binlog_row_value_options	PARTIAL_JSON
binlog_transaction_compression	OFF
completion_type	NO_CHAIN
connect_timeout	10
connection_memory_chunk_size	8912
connection_memory_limit	9223372036854775807
cte_max_recursion_depth	1000

Table 15-7 (Cont.) User-Configurable System Variables

Name	Default Value
default_authentication_plugin	CACHING_SHA2_PASSWORD
foreign_key_checks	ON
global_connection_memory_limit	9223372036854775807
global_connection_memory_tracking	OFF
group_replication_consistency	EVENTUAL
information_schema_stats_expiry	86400
innodb_buffer_pool_dump_pct	25
innodb_buffer_pool_instances	Shape dependent, see User-Configurable Shape-Dependent System Variables
innodb_buffer_pool_size	Shape dependent, see User-Configurable Shape-Dependent System Variables
innodb_ddl_buffer_size	1048576
innodb_ddl_threads	4
innodb_ft_enable_stopword	ON
innodb_ft_max_token_size	84
innodb_ft_min_token_size	3
innodb_ft_num_word_optimize	2000
innodb_ft_result_cache_limit	2000000000
innodb_ft_server_stopword_table	NULL
innodb_lock_wait_timeout	50
innodb_log_writer_threads	ON
innodb_max_purge_lag_delay	300000
innodb_max_purge_lag	0
innodb_stats_persistent_sample_pages	20
innodb_stats_transient_sample_pages	8
interactive_timeout	28800
--local-infile	OFF
mandatory_roles	Empty string
max_allowed_packet	67108864
max_binlog_cache_size	4294967296
max_connect_errors	18,446,744,073,709,551,615
max_connections	Shape dependent, see User-Configurable Shape-Dependent System Variables
max_execution_time	0
max_heap_table_size	16777216
max_prepared_stmt_count	Shape dependent, see User-Configurable Shape-Dependent System Variables
mysql_firewall_mode	ON
mysqlx_connect_timeout	30
mysqlx_deflate_default_compression_level	3
mysqlx_deflate_max_client_compression_level	5
mysqlx_interactive_timeout	28800

Table 15-7 (Cont.) User-Configurable System Variables

Name	Default Value
mysqlx_lz4_default_compression_level	2
mysqlx_lz4_max_client_compression_level	8
mysqlx_max_allowed_packet	67108864
mysqlx_read_timeout	28800
mysqlx_wait_timeout	28800
mysqlx_write_timeout	60
mysqlx_zstd_default_compression_level	3
mysqlx_zstd_max_client_compression_level	11
net_read_timeout	30
net_write_timeout	60
parser_max_mem_size	18446744073709551615
regexp_time_limit	32
sort_buffer_size	262144
sql_mode	ERROR_FOR_DIVISION_BY_ZERO, NO_ENGINE_SUBSTITUTION, NO_ZERO_DATE, NO_ZERO_IN_DATE, ONLY_FULL_GROUP_BY, STRICT_TRANS_TABLES
sql_require_primary_key	OFF
sql_warnings	OFF
telemetry_log_disable	OFF
thread_pool_dedicated_listeners	OFF
thread_pool_max_transactions_limit	Shape dependent, see User-Configurable Shape-Dependent System Variables
thread_pool_query_threads_per_group	Shape dependent, see User-Configurable Shape-Dependent System Variables
time_zone	UTC
tmp_table_size	16777216
transaction_isolation	REPEATABLE-READ
wait_timeout	28800

15.6 User-Configurable Shape-Dependent System Variables

Shape-dependent global system variables are linked to, and tuned for, a specific MySQL shape. See Supported Shapes. Configure the following variables with the HeatWave Console. See Creating a MySQL Configuration and Copying a MySQL Configuration.

> **Note:**
>
> `innodb_buffer_pool_instances` , `innodb_buffer_pool_size` , `max_connections` , and `max_prepared_stmt_count` have different default, minimum and maximum values for each shape and for HeatWave support.
>
> `thread_pool_max_transactions_limit` and `thread_pool_query_threads_per_group` have different default values for each shape.

Table 15-8 innodb_buffer_pool_instances

Shape	Supports HeatWave	Default	Minimum	Maximum
MySQL.2.16GB	No	4	1	64
MySQL.2.16GB	Yes	4	1	64
MySQL.4.32GB	No	4	1	64
MySQL.4.32GB	Yes	4	1	64
MySQL.8.64GB	No	4	1	64
MySQL.8.64GB	Yes	4	1	64
MySQL.32.256GB	No	8	1	64
MySQL.32.256GB	Yes	4	1	64

Table 15-9 innodb_buffer_pool_size

Shape	Supports HeatWave	Default	Minimum	Maximum
MySQL.2.16GB	No	10737418240	5242880	10737418240
MySQL.2.16GB	Yes	5368709120	5242880	5368709120
MySQL.4.32GB	No	21474836480	5242880	21474836480
MySQL.4.32GB	Yes	16106127360	5242880	16106127360
MySQL.8.64GB	No	51539607552	5242880	51539607552
MySQL.8.64GB	Yes	46170898432	5242880	46170898432
MySQL.32.256GB	No	206158430208	5242880	206158430208
MySQL.32.256GB	Yes	21474836480	5242880	21474836480

Table 15-10 max_connections

Shape	Supports HeatWave	Default	Minimum	Maximum
MySQL.2.16GB	No	1000	1	1000
MySQL.2.16GB	Yes	1000	1	1000
MySQL.4.32GB	No	2000	1	2000
MySQL.4.32GB	Yes	2000	1	2000
MySQL.8.64GB	No	4000	1	4000
MySQL.8.64GB	Yes	4000	1	4000

Table 15-10 (Cont.) max_connections

Shape	Supports HeatWave	Default	Minimum	Maximum
MySQL.32.256GB	No	8000	1	16000
MySQL.32.256GB	Yes	2000	1	16000

Table 15-11 max_prepared_stmt_count

Shape	Supports HeatWave	Default	Minimum	Maximum
MySQL.2.16GB	No	16382	16382	20000
MySQL.2.16GB	Yes	16382	16382	20000
MySQL.4.32GB	No	16382	16382	40000
MySQL.4.32GB	Yes	16382	16382	40000
MySQL.8.64GB	No	16382	16382	80000
MySQL.8.64GB	Yes	16382	16382	80000
MySQL.32.256GB	No	16382	16382	160000
MySQL.32.256GB	Yes	16382	16382	40000

Table 15-12 thread_pool_max_transactions_limit

Shape	Default	Minimum	Maximum
MySQL.2.16GB	32	0	100000
MySQL.4.32GB	64	0	100000
MySQL.8.64GB	128	0	100000
MySQL.32.256GB	512	0	100000

Table 15-13 thread_pool_query_threads_per_group

Shape	Default	Minimum	Maximum
MySQL.2.16GB	2	1	4096
MySQL.4.32GB	2	1	4096
MySQL.8.64GB	2	1	4096
MySQL.32.256GB	2	1	4096

15.7 Service-Specific System Variables

HeatWave on AWS defines the following global system variables. It is not possible to edit them.

> **Note:**
>
> `generated_random_password_length` , `query_alloc_block_size` , and `query_prealloc_size` are also session variables. See Session Variables.

Table 15-14 Service-Specific Global System Variables

Name	Default Value
generated_random_password_length	20
mysqlx_document_id_unique_prefix	0
mysqlx_enable_hello_notice	ON
mysqlx_idle_worker_thread_timeout	60
mysqlx_min_worker_threads	2
query_alloc_block_size	8192
query_prealloc_size	8192

The `SHOW VARIABLES` statement with the `GLOBAL` modifier displays global system variable values:

```
mysql> SHOW GLOBAL VARIABLES;
```

To obtain the value for a specific variable, use a `LIKE` clause as shown:

```
mysql> SHOW GLOBAL VARIABLES LIKE 'max_join_size';
```

To get a list of variables whose name match a pattern, use the `%` wildcard character in a `LIKE` clause:

```
mysql> SHOW GLOBAL VARIABLES LIKE '%size%';
```

15.8 Shape-Dependent System Variables

Shape-dependent global system variables are linked to, and tuned for, a specific MySQL shape. See Supported Shapes. It is not possible to edit them.

> **Note:**
>
> innodb_parallel_read_threads is also a session variable. See: Session Variables.

Table 15-15 MySQL.2.16GB Global System Variables

Name	Default Value
back_log	1000
innodb_page_cleaners	4
innodb_parallel_read_threads	1
innodb_read_io_threads	1
innodb_redo_log_capacity	2

Table 15-15 (Cont.) MySQL.2.16GB Global System Variables

Name	Default Value
replica_parallel_workers > 0	4
temptable_max_ram	1073741824
thread_pool_size	8

Table 15-16 MySQL.4.32GB Global System Variables

Name	Default Value
back_log	2000
innodb_page_cleaners	4
innodb_parallel_read_threads	2
innodb_read_io_threads	2
innodb_redo_log_capacity	4
replica_parallel_workers > 0	8
temptable_max_ram	1073741824
thread_pool_size	16

Table 15-17 MySQL.8.64GB Global System Variables

Name	Default Value
back_log	4000
innodb_page_cleaners	4
innodb_parallel_read_threads	4
innodb_read_io_threads	4
innodb_redo_log_capacity	8
replica_parallel_workers > 0	16
temptable_max_ram	2147483648
thread_pool_size	32

Table 15-18 MySQL.32.256GB Global System Variables

Name	Default Value
back_log	2000
innodb_page_cleaners	8
innodb_parallel_read_threads	32
innodb_read_io_threads	2
innodb_redo_log_capacity	8
replica_parallel_workers > 0	16
temptable_max_ram	1073741824
thread_pool_size	32

15.9 Session Variables

Session variables remain in effect during the session until the variable changes or the session ends. The change has no effect on other sessions. For new connections, a session variable value is initialized with the corresponding global system variable value. Many of the following session variables are also available as user configurable variables. See: User-Configurable System Variables.

To assign a value to a session variable, precede the variable name with the `SESSION` or `LOCAL` keyword, or with the `@@SESSION.`, `@@LOCAL.`, or `@@` qualifier, or with no keyword or modifier. For example:

```
mysql> SET SESSION sql_mode = 'TRADITIONAL';
mysql> SET LOCAL sql_mode = 'TRADITIONAL';
mysql> SET @@SESSION.sql_mode = 'TRADITIONAL';
mysql> SET @@LOCAL.sql_mode = 'TRADITIONAL';
mysql> SET @@sql_mode = 'TRADITIONAL';
mysql> SET sql_mode = 'TRADITIONAL';
```

Table 15-19 User Settable Session Variables

Name	Default Value
autocommit	ON
big_tables	OFF
block_encryption_mode	aes-128-ecb
character_set_client	utf8mb4
character_set_connection	utf8mb4
character_set_results	utf8mb4
character_set_server	utf8mb4
collation_connection	utf8mb4_0900_ai_ci
collation_database	utf8mb4_0900_ai_ci
collation_server	utf8mb4_0900_ai_ci
completion_type	NO_CHAIN
cte_max_recursion_depth	1000
default-storage-engine	InnoDB
--default_tmp_storage_engine	InnoDB
default_week_format	0
div_precision_increment	4
end_markers_in_json	OFF
eq_range_index_dive_limit	200
foreign_key_checks	ON
generated_random_password_length	20
group_concat_max_len	1024
group_replication_consistency	BEFORE_ON_PRIMARY_FAILOVER
information_schema_stats_expiry	86400
innodb_ddl_buffer_size	1048576

Table 15-19 (Cont.) User Settable Session Variables

Name	Default Value
innodb_ddl_threads	4
innodb_ft_enable_stopword	ON
innodb_ft_user_stopword_table	NULL
innodb_lock_wait_timeout	50
innodb_parallel_read_threads	Shape dependent. See Shape-Dependent System Variables.
internal_tmp_mem_storage_engine	TempTable
join_buffer_size	262144
lc_messages	en_US
lc_time_names	en_US
lock_wait_timeout	86400
long_query_time	10
max_allowed_packet	67108864
max_execution_time	0
max_heap_table_size	16777216
max_join_size	18446744073709551615
max_length_for_sort_data	4096
max_points_in_geometry	65536
max_seeks_for_key	18446744073709551615
max_sort_length	1024
mysqlx_max_allowed_packet	67108864
mysqlx_read_timeout	30
mysqlx_wait_timeout	28800
mysqlx_write_timeout	60
net_buffer_length	16384
net_read_timeout	30
net_retry_count	10
net_write_timeout	60
new	OFF
old_alter_table	OFF
optimizer_prune_level	1
optimizer_search_depth	62

Table 15-19 (Cont.) User Settable Session Variables

Name	Default Value
optimizer_switch	index_merge_sort_union=on, index_merge_intersection=on, engine_condition_pushdown=on, index_condition_pushdown=on, mrr=on, mrr_cost_based=on, block_nested_loop=on, batched_key_access=off, materialization=on, semijoin=on, loosescan=on, firstmatch=on, duplicateweedout=on, subquery_materialization_cost_based=on, use_index_extensions=on, condition_fanout_filter=on, derived_merge=on, use_invisible_indexes=off, skip_scan=on, hash_join=on, subquery_to_derived=off, prefer_ordering_index=on, hypergraph_optimizer=off, derived_condition_pushdown=on
optimizer_trace	enabled=off,one_line=off
optimizer_trace_features	greedy_search=on, range_optimizer=on, dynamic_range=on, repeated_subselect=on
optimizer_trace_limit	1
optimizer_trace_max_mem_size	1048576
optimizer_trace_offset	-1
parser_max_mem_size	18446744073709551615
print_identified_with_as_hex	OFF
pseudo_replica_mode	OFF
pseudo_slave_mode	OFF
query_alloc_block_size	8192
query_prealloc_size	8192
range_alloc_block_size	4096
range_optimizer_max_mem_size	8388608
rbr_exec_mode	STRICT
read_buffer_size	131072
read_rnd_buffer_size	262144
resultset_metadata	FULL
secondary_engine_cost_threshold	100000.000000
session_track_gtids	OFF
session_track_schema	ON
session_track_state_change	OFF
session_track_system_variables	262144
session_track_transaction_info	OFF
show_create_table_skip_secondary_engine	OFF
show_create_table_verbosity	OFF

Table 15-19 (Cont.) User Settable Session Variables

Name	Default Value
sort_buffer_size	262144
sql_auto_is_null	OFF
sql_big_selects	ON
sql_buffer_result	OFF
sql_mode	ONLY_FULL_GROUP_BY, STRICT_TRANS_TABLES, NO_ZERO_IN_DATE, NO_ZERO_DATE, ERROR_FOR_DIVISION_BY_ZERO, NO_ENGINE_SUBSTITUTION
sql_notes	ON
sql_quote_show_create	ON
sql_safe_updates	OFF
sql_select_limit	18446744073709551615
sql_warnings	OFF
thread_pool_high_priority_connection	0
time_zone	UTC
tmp_table_size	16777216
transaction_alloc_block_size	8192
transaction_isolation	REPEATABLE-READ
transaction_prealloc_size	4096
transaction_read_only	OFF
unique_checks	ON
updatable_views_with_limit	YES
use_secondary_engine	ON
wait_timeout	28800
windowing_use_high_precision	ON

16

User and Group Management

HeatWave on AWS uses predefined OCI IAM groups and policies to manage access to the HeatWave Console. Predefined groups and policies are created when the service is provisioned. Defining your own groups and policies for HeatWave on AWS is currently not supported.

An Administrator grants access to the HeatWave Console by adding users to the predefined OCI IAM groups. User management is performed in the OCI Console. The policies associated with each group determine which resources that users can access and the permissions associated with those resources. HeatWave on AWS resources include `DB Systems`, `DB System Backups`, and `HeatWave Clusters`.

> **Note:**
>
> HeatWave on AWS supports federation with third-party Identity Providers (IdPs). For more information, see Federating with Identity Providers, in the Oracle Cloud Infrastructure documentation.

16.1 Groups and Permissions

HeatWave on AWS has three predefined groups. The groups are created in the OCI **Default** identity domain. The predefined groups and associated permissions are described in the following tables.

Table 16-1 OracleMySQLHeatwaveDBUsers Group

Group Description	Resources and Permissions
OracleMySQLHeatwaveDBUsers: Members of this group can use DB Systems, DB System Backup, and HeatWave Clusters resources.	DB Systems • View supported shapes • View supported MySQL versions • View DB Systems • View DB System Details • Update DB Systems • Run queries • View query statuses • Stop queries • Import data • View data imports • View data import details • Cancel data imports DB System Backups • View DB System backups • View DB System backup details • Update DB System backups HeatWave Clusters • View HeatWave Clusters • View HeatWave Cluster details • Estimate HeatWave Cluster size • View supported shapes MySQL Configurations • View configurations • View configuration details • Update configurations • View configuration variable metadata Inbound Replication Channels • View channels • View channel details • Resume channels Service Events • View events • View event details

Table 16-2 OracleMySQLHeatwaveDBAdmin Group

Group Description	Resources and Permissions
OracleMySQLHeatwaveDBAdmin: Members of this group can manage all aspects of DB Systems, DB System Backups, and HeatWave Clusters resources.	**In addition to *OracleMySQLHeatwaveDBUsers* group permissions**, this group has these permissions: DB Systems • Create DB Systems • Delete DB Systems • Start DB Systems • Stop DB Systems • Restart DB Systems DB System Backups • Create DB System backups • Delete DB System backups HeatWave Clusters • Create HeatWave Clusters • Delete HeatWave Clusters • Start HeatWave Clusters • Stop HeatWave Clusters • Restart HeatWave Clusters MySQL Configurations • Create configurations • Delete configurations Inbound Replication Channels • Create channels • Delete channels • Update channels • Reset channels

Table 16-3 OracleMySQLHeatwaveServiceAccountAdmin Group

Group Description	Resources and Permissions
OracleMySQLHeatwaveServiceAccountAdmin: Members of this group can manage all aspects of DB Systems, DB System Backups, and HeatWave Clusters resources.	This group has the same permissions as the *OracleMySQLHeatwaveDBAdmin* group.

> **Note:**
>
> The OCI user account that registered for the HeatWave on AWS service is added to the `OracleMySQLHeatwaveServiceAccountAdmin` group when the service is provisioned.

16.2 Groups and Policies

The predefined groups used to manage HeatWave Console access, described in Groups and Permissions, are created in the OCI **Default** identity domain. For information, see https://docs.oracle.com/en-us/iaas/Content/Identity/domains/overview.htm#the_default_domain, in the *Oracle Cloud Infrastructure Documentation*.

The predefined groups all start with `OracleMySQLHeatWave`. Policies are defined for each predefined group which enable the HeatWave on AWS resource permissions outlined in Groups and Permissions.

> **NOT_SUPPORTED:**
>
> The predefined `OracleMySQLHeatWave` groups and policies are static. You must not modify them, add to them, or remove them. If you do delete one of these groups, HeatWave on AWS will stop working. In this case, submit a My Oracle Support ticket to get the group re-added to your account, specifying " HeatWave on AWS " as the product.

Because the predefined groups are created in OCI, you can define further OCI policies using these groups to provide access to other OCI resources. For example, you can create an OCI policy that allows members of the `OracleMySQLHeatwaveDBUsers` group to access resources in an OCI tenancy. For information about OCI policies, see How Policies Work.

16.3 User Management

User management is performed in the Oracle Cloud Infrastructure (OCI) Console.

HeatWave on AWS uses predefined OCI IAM groups created in the OCI **Default** identity domain to manage access to the HeatWave Console, as described in Groups and Permissions. A HeatWave on AWS Administrator manages the users that belong to those groups.

To access the Oracle Identity Cloud Service from the Oracle Cloud Infrastructure (OCI) Console, open the navigation menu and click **Identity & Security**.

Alternatively, to access the Oracle Identity Cloud Service from the HeatWave Console:

1. Sign into the HeatWave Console as an Administrator. For sign-in instructions, see Signing In.
2. From the profile menu, select **Administration**.
3. Select **Identity Service**.

Refer to Managing Users, in the *Oracle Cloud Infrastructure Documentation*, for the following user management procedures.

- Creating a user
- Editing a user
- Resetting a user's password
- Deleting a user

Refer to Managing Groups, in the *Oracle Cloud Infrastructure Documentation*, for the following group-related user management procedures.

> **Note:**
>
> The predefined groups used by HeatWave on AWS are created in the OCI **Default** identity domain. For more information, see The Default Identity Domain, in the *Oracle Cloud Infrastructure Documentation*.

- Adding users to groups
- Removing users from groups

To add another user with Oracle Cloud Administrator permissions, see Add a User with Oracle Cloud Administrator Permissions, in the *Oracle Cloud Infrastructure Documentation*.

17
Account Management

This chapter discusses region management, service limits, and billing for HeatWave on AWS.

- Manage Regions
- Service Limits
- Billing
- Viewing OCID of the Tenancy
  Use the Oracle Cloud Infrastructure (OCI) Console to view and copy the OCID of the tenancy.
- Manage AWS Access

17.1 Manage Regions

The AWS region is shown at the top of the HeatWave Console. To change regions, click on ***AWS region name*** , and select the preferred region.

To add a new region:

1. Click on ***AWS region name*** , and select **Manage Regions**.
2. Subscribe to an OCI region.
   a. Click **Subscribe** next to the OCI region that maps to the preferred AWS region.

      If a region limit increase is required, do the following to request an increase:

      i. From the **Limit increase needed** dialog box, click **Open Service Options**.

      ii. Follow these steps: Requesting a Limit Increase to the Subscribed Region Count.

      iii. When the region limit increase is complete, click **Subscribe** next to the OCI region that maps to the preferred AWS region.

   b. From the **Subscribe to New Region** dialog box, read the privacy notice, and then click **Subscribe**.
3. Enable the HeatWave on AWS service.
   a. Click **Enable** next to the preferred AWS region.
   b. From the **Enable Service** dialog box, click **Enable**.

17.2 Service Limits

A service limit is the quota or allowance set on a resource. HeatWave on AWS default service limits are shown below. Service limits are per region unless explicitly specified.

To request a service limit increase, submit a My Oracle Support ticket, specifying " HeatWave on AWS " as the product.

Table 17-1 Service Limits

Resource	Default Limit	Free Trial Limit
MySQL Database Block Storage	10 TB	300 GB
MySQL Database Manual Backups	40 (for all DB System instances)	2
Sum of DB System Backup Retention Days*	70	4
MySQL.32.256GB instances	2	1
MySQL.8.64GB instances	1	1
MMySQL.4.32GB instances	1	1
MySQL.2.16GB instances	4	1
HeatWave.16GB nodes	8	4
HeatWave.256GB nodes	8	1
MySQL DB Systems Outbound Data Transfer	None	30 GB per day**

* This refers to the sum of the retention period for all the automatic backups of all the MySQL DB Systems in your tenancy. For example, if there are two DB systems in your tenancy and both have automatic backup set up, with one having a retention period of 5 days and another having a retention period of 7 days, the Sum of DB System Backup Retention Days will be 12.

** During a free trial, when the total outbound data transfer from the MySQL DB Systems exceeds the daily limit, connection to MySQL DB Systems will be blocked for the rest of the day. At midnight in UTC, the daily outbound data transfer is reset and connections are unblocked.

> **Tip:**
>
> Even when connections to MySQL DB Systems are blocked, you can continue to use the HeatWave Console.

17.3 Billing

For information about managing service costs, see Billing, Cost Management, and Payments Overview.

Billing for HeatWave on AWS is managed in the OCI Console.

To access billing information in the Oracle Cloud Infrastructure (OCI) Console, open the navigation menu and select **Billing & Cost Management**.

Alternatively, to access billing information from the HeatWave Console,:

1. Sign into the HeatWave Console as an Account Administrator.
2. From the profile menu, select **Administration**.
3. Select **Billing**.

17.4 Viewing OCID of the Tenancy

Use the Oracle Cloud Infrastructure (OCI) Console to view and copy the OCID of the tenancy.

This task requires the following:

- Access to OCI Console.

Do the following to view and copy the OCID of the tenancy:

1. Open the Oracle Cloud Infrastructure (OCI) Console.
2. Sign in with your **Cloud Account Name** (sometimes referred to as your tenancy name), user name, and password.
3. In the Oracle Cloud Infrastructure Console home page, click the **Navigation menu**, and then click **Governance & Administration**. Under **Account Management**, click **Tenancy Details**.
4. In the **Tenancy details** page, in the **Tenancy information** tab, click **Show** or **Copy** present besides the **OCID** field to view or copy the OCID of the tenancy respectively.

17.5 Manage AWS Access

Use the AWS Management Console to manage AWS access.

- Creating an IAM Policy to Access an Amazon S3 Bucket
- Creating an IAM Role to Access an Amazon S3 Bucket

17.5.1 Creating an IAM Policy to Access an Amazon S3 Bucket

Use the AWS Management Console to create an IAM policy to access an Amazon S3 bucket.

This task requires the following:

- Access to AWS Management Console.
- The name of the Amazon S3 bucket you want to grant access to.

Do the following to create an IAM policy:

1. Open the AWS Management Console and sign in with your credentials.
2. In the AWS Management Console home page, click **Services**, and click **Security, Identity, & Compliance**, and then click **IAM**.
3. In the navigation pane of the Console, under **Access management**, click **Policies**, and then click **Create policy**.

   It opens the **Specify permissions** page.
4. In the **Specify permissions** page, in the **Policy editor** section, click **JSON**, and enter the following Amazon S3 permissions as per the feature you use. For more information on policies, see Generate policies.
   - For bulk ingest:

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::<AmazonS3BucketName>/*"
            ]
        }
    ]
}
```

- For data import:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::<AmazonS3BucketName>/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::<AmazonS3BucketName>"
            ]
        }
    ]
}
```

- If the objects in the Amazon S3 bucket are encrypted with a customer-managed KMS key, add the following permission for the keys used for encrypting the objects:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "<KmsKeyArn>"
            ]
        }
    ]
}
```

See Using IAM policies with AWS KMS.

5. Resolve any warnings or errors generated during permissions validation, and then click **Next**.
6. In the **Review and create** page, in the **Policy details** section, enter the following:
   - **Policy name**: Specify a name to identify this policy.
   - **Description**: (Optional) Specify a description of the policy.
7. Click **Create policy**.

17.5.2 Creating an IAM Role to Access an Amazon S3 Bucket

Use the AWS Management Console to create an IAM role for accessing an Amazon S3 bucket.

This task requires the following:

- Access to AWS Management Console.
- IAM policy that specifies the Amazon S3 and/or KMS permissions required for the feature you want to use. See Creating an IAM Policy to Access an Amazon S3 Bucket.
- The tenancy Oracle Cloud Identifier (OCID).
- If you want to grant access to a specific DB System, the resource ID of the DB System.
- The name of the Amazon S3 bucket you want to grant access to.

Do the following to create an IAM role:

1. Open the AWS Management Console and sign in with your credentials.
2. In the AWS Management Console home page, click **Services**, and click **Security, Identity, & Compliance**, and then click **IAM**.
3. In the navigation pane of the Console, under **Access management**, click **Roles**, and then click **Create role**.
4. In the **Select trusted entity** panel, do the following:
   a. Select **Custom trust policy**.
   b. Specify the following trust policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {
                "AWS": "612981981079"
            },
            "Condition": {
                "StringLike": {
                    "sts:ExternalId": "<IDDetails>"
                }
            }
        }
    ]
}
```

- When you are editing a DB System, specify either of the following in <IDDetails>:
  - To grant access to a specific DB System in the tenancy: <*TenancyOCID*>/<*DBSystemResourceId*>.
  - To grant access to all DB Systems in the tenancy: <*TenancyOCID*>/*.
- When you are creating a DB System, specify the following in <IDDetails>:
  - To grant access to all DB Systems in the tenancy:<*TenancyOCID*>/* Once the DB System is created, update the trust policy to limit access to a specific DB System.

To view the OCID of the tenancy, see Viewing OCID of the Tenancy and to view the resource ID of the DB System, see Viewing DB System Details.
For example:

```
ocid1.tenancy.oc1...axxxaaaat5j...famyhq/5281bb96-99a1-23fe-
a65f-370cd85b979f
```

```
ocid1.tenancy.oc1...axxxaaaat5j...famyhq/*
```

See Using an external ID for third-party access.

5. Resolve any warnings or errors generated during policy validation, and then click **Next**.
6. In the **Add permissions** page, search for the policy you created, and select the check box to attach the policy to your new role. See Creating an IAM Policy to Access an Amazon S3 Bucket.
7. Click **Next**.
8. In the **Name, review, and create** page, in the **Role details** section, enter the following:
   - **Role name**: Enter a name to the identify the role.
   - **Description**: (Optional) Specify a description of the policy.
9. Click **Create role**.
10. Click the role you just created.
11. In the **Summary** section, copy the ARN.

After you create the role, enter the role ARN in an exisiting DB System, or create a new DB System and enter the ARN details. See Editing a DB System and Creating a DB System.

18 Maintenance

This section describes the maintenance of DB Systems and HeatWave.

Essential patching and maintenance is an automatic process that may include patching the underlying operating system, updating the MySQL Server version, and updating underlying hardware. Maintenance is initiated during the maintenance window defined when creating a DB System. The maintenance window start day and time can be viewed on the **DB System Details** page in the HeatWave on AWS Console. The maintenance window is a two hour period during which maintenance is initiated. The time required to apply patches and updates may extend beyond the maintenance window and require DB System restarts. See Viewing DB System Details.

To change a DB System's Maintenance Window start day or start time, see Editing a DB System.

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

> **Note:**
>
> DB Systems maintenance is performed when the DB System is in an `ACTIVE` state or an `INACTIVE` state, and the DB System is returned to the same state after the maintenance completes.

19

Release Notes

Release notes for HeatWave on AWS

2024-06-25, General Availability

- You can now utilize AWS PrivateLink to establish a secure connection to HeatWave on AWS from your VPC. This ensures that sensitive data such as customer details remain protected from the internet, enabling secure and private transfer of critical data in a scalable manner. For more information, please refer to PrivateLink.
- HeatWave on AWS now includes HeatWave Lakehouse, enabling organizations to process and query hundreds of terabytes of data stored in object storage and optionally combine it with transactional data in MySQL databases without copying the data from the object storage into the MySQL database. Customers can also use this data loaded from the object storage to train machine learning models, run inferences, and explain results, without moving the data to a separate ML service. With Lakehouse, HeatWave on AWS offers a single service for analytics across data warehouses and data lakes, machine learning, and transaction processing, eliminating the need for ETL across cloud services. For more information, please refer to Creating Lakehouse Data Mapping.

2024-06-05, General Availability

- You can now try out MySQL HeatWave on AWS for free without having to upgrade to a paid Oracle Cloud account. See Try MySQL HeatWave for free and Sign-up Overview for details.

2024-05-22, General Availability

- MySQL HeatWave on AWS now supports MySQL 8.4.0. See MySQL 8.4.0 Release Notes for more information on the server release.

  MySQL 8.4.0 marks the start of a new series of LTS (Long Term Support) releases. The 8.4 LTS releases will focus on security and bug fixes and are recommended for customers who prefer established behavior, while the next innovation series will include new features.

  We recommend customers who are using any lower versions of MySQL Server to upgrade to MySQL 8.4.0. See Upgrade MySQL Version for instructions.

2024-05-02, General Availability

- You can now explore the capabilities of MySQL HeatWave on AWS by launching a pre-configured DB System with sample data. See Launching a Starter DB System for details.

2024-04-08, General Availability

- You can now explore MySQL HeatWave on AWS features using sample databases. See Importing Sample Database for details.

2024-03-04, General Availability

- Events allow you to monitor the state changes of resources such as DB Systems, HeatWave Clusters, and Backups. An event reports a create, update, or delete operation

for a resource, or its lifecycle state change. Events are listed under the **Events** tab of the corresponding resource. For more details, see Events.

2024-02-20, General Availability

- You can now create a custom configuration by copying an existing MySQL configuration and modifying its user variables. You can then apply the custom configuration to a DB System. For more information, see Copying a MySQL Configuration and Update MySQL Configuration.

2024-02-15, General Availability

- MySQL HeatWave on AWS now supports the MySQL Enterprise Audit plugin. The audit plugin enables the MySQL Server to produce a log file containing an audit record of server activities such as when clients connected and disconnected, what actions they performed while being connected, and which databases and tables they accessed. You can also add statistics for the time and size of the queries performed. See MySQL Enterprise Audit for details.
- You can now create a custom configuration by copying an existing MySQL configuration and modifying its user variables. You can then apply the custom configuration to a DB System. For more information, see Copying a MySQL Configuration and Update MySQL Configuration.

2024-01-24, General Availability

- MySQL HeatWave on AWS now supports MySQL version 8.3.0.

2024-01-17, General Availability

- MySQL HeatWave on AWS supports inbound replication that does not use GTID auto-positioning, and also for tables that do not have primary keys. See Creating a Channel for details.

2023-12-21, General Availability

- The Auto Shape Prediction feature in MySQL Autopilot has been enhanced. In addition to the overall buffer pool usage, workload activity, and access patterns, Auto Shape Prediction now utilizes the CPU statistics to estimate the required buffer pool size and CPU cores. See Autopilot Shape Advisor for details.

2023-11-28, General Availability

- MySQL HeatWave on AWS supports adding filters in inbound replication. Filters allows you to selectively replicate databases and tables from the MySQL source. For more information on inbound replication filters, please see Channel Filter Rules for Inbound Replication.
- MySQL HeatWave on AWS introduces delayed inbound replication. You can choose to specify the time (in seconds) to wait before the target DB System in MySQL HeatWave on AWS replicates the source transactions. For more information on delayed replication, please see Replication delay.

2023-11-16, General Availability

- MySQL HeatWave on AWS supports inbound replication from a MySQL source to a MySQL DB System.
  For more information on the inbound replication feature, see Inbound Replication.

- You can now update a HeatWave Cluster's size and shape. For more information, see Editing a HeatWave Cluster.

2023-10-26, General Availability

- MySQL HeatWave on AWS supports MySQL version 8.2.0.

2023-09-13, General Availability

- MySQL HeatWave on AWS introduces a faster and easier feature to import data from an Amazon S3 bucket into a MySQL DB System. The data import feature enables you to import data in a variety of formats such as MySQL dump and text files (such as CSV and TSV).
  For more information on the data import feature, see Data Import Feature.
- MySQL HeatWave on AWS supports the bulk ingest feature to import text files (such as CSV and TSV) directly from an Amazon S3 bucket into a MySQL DB System. This method is faster and more efficient in terms of computing and storage consumption. To use the bulk ingest feature, first connect to the DB system, and then import data in an existing or new table using the BULK algorithm.
  For more information on the bulk ingest feature, see Bulk Ingest Feature.

2023-08-08, General Availability

- You can now increase the data storage size of an active DB System online. When the DB System is active and healthy, updating the data storage size of the DB System does not restart the DB System, and you can continue to query it while the storage is being increased. You get elasticity without compromising uptime or performance.
  For more information, see Increasing DB System Storage.

2023-07-31, General Availability

- MySQL HeatWave on AWS now supports auto error recovery. Whenever a HeatWave node fails because of a hardware or a software issue, the cluster becomes unhealthy and error recovery is triggered. During the recovery process, HeatWave automatically attempts to bring the node online and reload data that was previously loaded. This reduces manual intervention and improves service uptime.
  For more information, see HeatWave Cluster Failure and Recovery.

2023-07-13, General Availability

- MySQL HeatWave on AWS is now available in the AWS Europe (Frankfurt) and Europe (London) region. See Region Availability.
- Fast data reload is now available when you pause and resume HeatWave on AWS. Besides storing the HeatWave formatted data in-memory, HeatWave also stores the formatted data in HeatWave storage layer on AWS S3. This enables reload of data in constant time regardless of data size. This capability is now available when you pause and resume the HeatWave cluster. You can now pause the HeatWave cluster when you don't need it to save cost, and resume the cluster very fast when you want to use HeatWave for query acceleration.

2023-06-08, General Availability

- MySQL HeatWave on AWS is now available in the AWS Asia Pacific (Mumbai) region. See Region Availability.

2023-05-04, General Availability

- MySQL HeatWave on AWS is now available in the AWS Asia Pacific (Tokyo) region. See Region Availability.

2023-04-27, General Availability

- HeatWave Scale-out Data Management on AWS S3
  MySQL HeatWave on AWS now provides an optimized storage layer built on AWS S3 to store the HeatWave in-memory hybrid columnar representation of the data. This allows data to be reloaded to each HeatWave node independently and in parallel. This significantly improves the service uptime and performance of operations such as error recovery, maintenance, and system restart. See HeatWave Architectural Features.
- MySQL Autopilot: Auto Error Recovery from MySQL failure
  With Auto Error Recovery, now when MySQL fails and restarts, the HeatWave Cluster automatically restarts, identifies the tables which were loaded prior to the failure, and reloads those tables automatically from MySQL. This reduces the intervention by the user and also improves service uptime.
- Auto reload of data in HeatWave Cluster after MySQL upgrade
  HeatWave now automatically reloads data from MySQL InnoDB after a MySQL node restarts due to maintenance upgrades or planned restarts. With auto-reload capability, there are no more manual steps after maintenance or a restart operation. This reduces the operational overhead and improves service availability.
- MySQL Autopilot for OLTP - Auto Shape Prediction
  Auto shape prediction collects the most recent query execution metrics and uses advanced machine learning models to predict the MySQL database instance shape for optimal transactional processing performance. Auto shape prediction continuously monitors OLTP workload to provide a suggestion that will adapt to evolving workload patterns, allowing the MySQL DB System to maintain the best OLTP price performance over time. See Autopilot Shape Advisor.

  Monitor MySQL statistics such as buffer pool usage, workload activity and access patterns, and recommendations for the optimal MySQL shape for the workload with MySQL HeatWave Console.
- MySQL Configuration
  Use MySQL HeatWave Console to view and configure user, system, initialization, and service-specific variables for the MySQL DB System shape with or without HeatWave support. Select a default configuration or create a custom configuration for the DB system. See and Configuration.
- Automatic Backup
  MySQL HeatWave on AWS now support automatic backups. Choose a time for automatic backups during the creation of a MySQL DB System. The retention period can be between 1 and 35 days. The default retention period is 7 days. Scheduled backups are deleted when the DB System is deleted. See Creating a DB System.