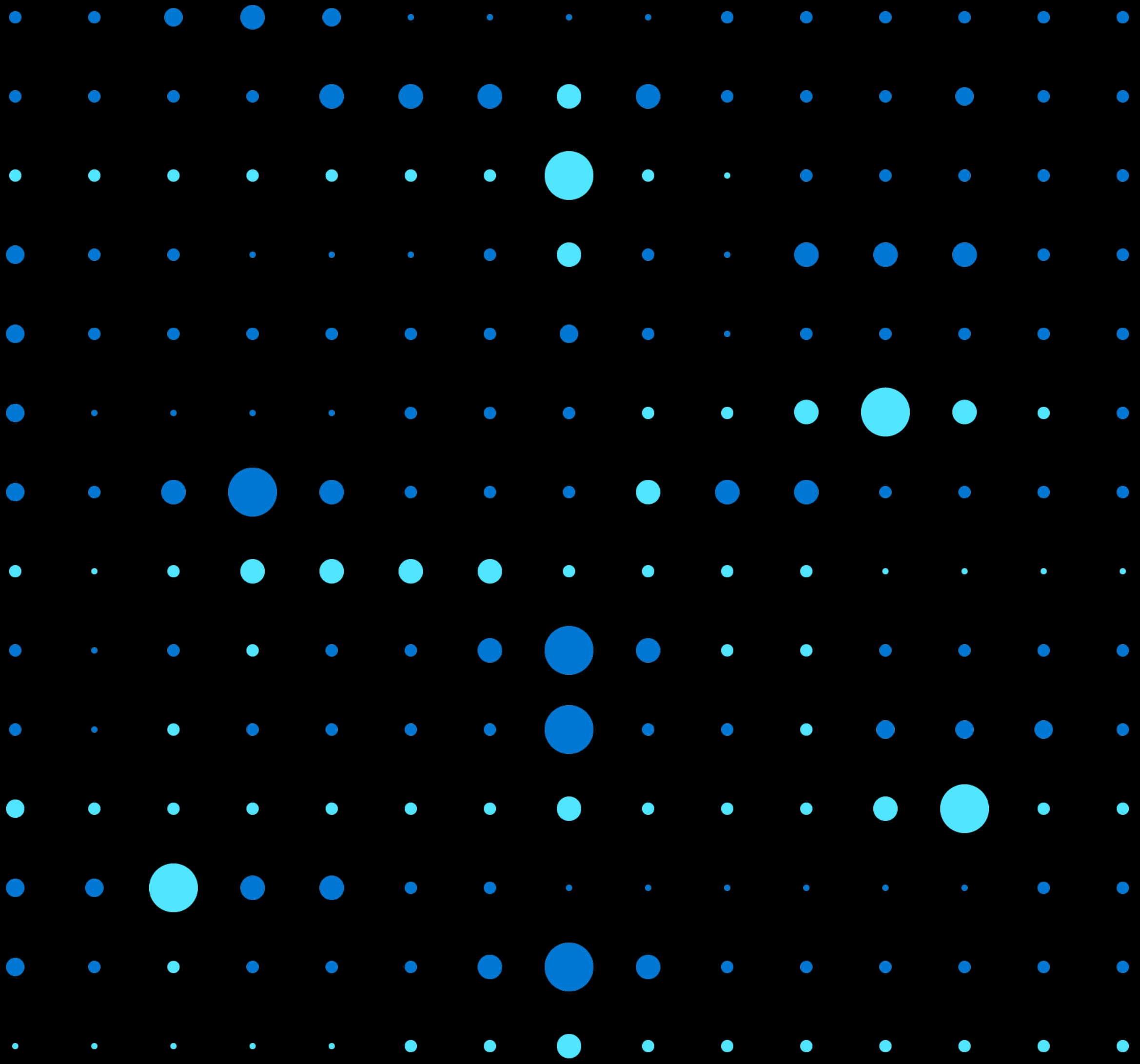
E-book Series

Microsoft Azure

Developer's Guide to Getting Started with Microsoft Azure Database for MySQL



Azure Database for MySQL makes it easy to build new applications

Azure Database for MySQL makes it easy to build new applications or migrate existing ones to the cloud using the programming languages and frameworks of your choice. It's a fully managed service, so you can provision it in minutes and independently scale compute and storage in seconds, benefiting from built-in high availability and a service-level agreement of 99.99 percent. You can pay as you go under a simple yet flexible pricing model, so you won't need to pay extra for automatic patching and backups, monitoring, or essential security features. You also won't have to devote precious resources to handling such things yourself. In addition, like all open-source database services from Microsoft, Azure Database for MySQL is built on a world-class infrastructure with unmatched global reach, plugging you into the worldwide network of Microsoft Azure datacenters.

Audience

This paper is targeted at developers who are thinking of building new cloud apps on MySQL or moving existing MySQL apps to the cloud. It provides an overview of Azure Database for MySQL and the value that it provides as a fully managed, open-source database service. It also covers the key concepts you'll need to understand to best put Azure Database for MySQL to use, together with a list of quickstarts, tutorials, and other resources to help you get started using it.

Introduction: MySQL databases in the cloud

In the early days of computing, developers shared the code they wrote to learn from one another and evolve their software. This spirit of collaboration remains alive today with open-source software, which gives users rights to access, modify, and distribute source code—relying on community contributions from across the globe to achieve more than any one organization could on its own.

To fully appreciate the value of open-source software, it's worth examining a few of the benefits it provides. These include open standards that are accessible to everyone, along with full visibility into the code base. Everyone benefits from community efforts to improve and support the software, a lack of licensing fees, and lower development costs. Openly available source code speeds the delivery of new offerings, and freedom from vendor lock-in facilitates increased choice across platforms. Best practices for maximizing performance and availability are shared freely, as are those for deployment and operations, and reliability is maximized through community vetting. In addition, the flexibility that open-source software provides for developers can help attract more and better talent.

It's for these reasons that open-source databases such as MySQL have grown to be so popular. MySQL Community Edition is supported by a large and vibrant community of open-source developers. As the database component of the popular LAMP (Linux, Apache, MySQL, PHP) web application stack, MySQL remains one of the most popular database management systems among developers today.

The pervasiveness of MySQL is fueling explosive growth in MySQL database *services*, which let you reap the benefits of using MySQL while avoiding the many pitfalls of on-premises solutions or hosted, self-managed virtual machines. For example, with cloud services, you won't need to deal with deployment efforts, continual patching, painstaking performance optimization, complex high-availability mechanisms, intricate security and compliance concerns, and continual worries about scalability—all of which can lead to unpredictable delays and costs.

To make MySQL work in the enterprise, however, you'll need a lot more than just a hosted MySQL database service. Rarely does any database exist in isolation, which is why you'll also want to make sure your chosen platform offers everything else you'll need to effectively put your chosen database service to use. The specific services you'll need will depend on your app, such as whether you'll be ingesting massive data volumes, implementing real-time streaming analytics, building an AI-powered chatbot, or simply deploying an instance of Drupal or WordPress. And don't forget about development tools, software frameworks, and integration; you won't want to complicate things unnecessarily in those areas either. After all, even if a cloud platform provides all the services you need, it won't do you much good if you need to learn an entirely new skill set before you can effectively put those services to use.

In the rest of this paper, we'll provide an overview of Microsoft Azure Database for MySQL, a fully managed MySQL database service, and the value that it brings to you as a developer. We'll also cover the key concepts you'll need to understand to best put Azure Database for MySQL to use, together with a list of quickstarts, tutorials, and other resources to help you get started using it.

Azure Database for MySQL

Azure Database for MySQL combines the community edition of the MySQL database engine with all the benefits of the cloud, freeing you from the complexity of infrastructure and database management so you can focus on building exceptional apps. As a fully managed service, Azure Database for MySQL lets you easily build new applications or migrate existing ones to the cloud using the programming languages and frameworks of your choice.

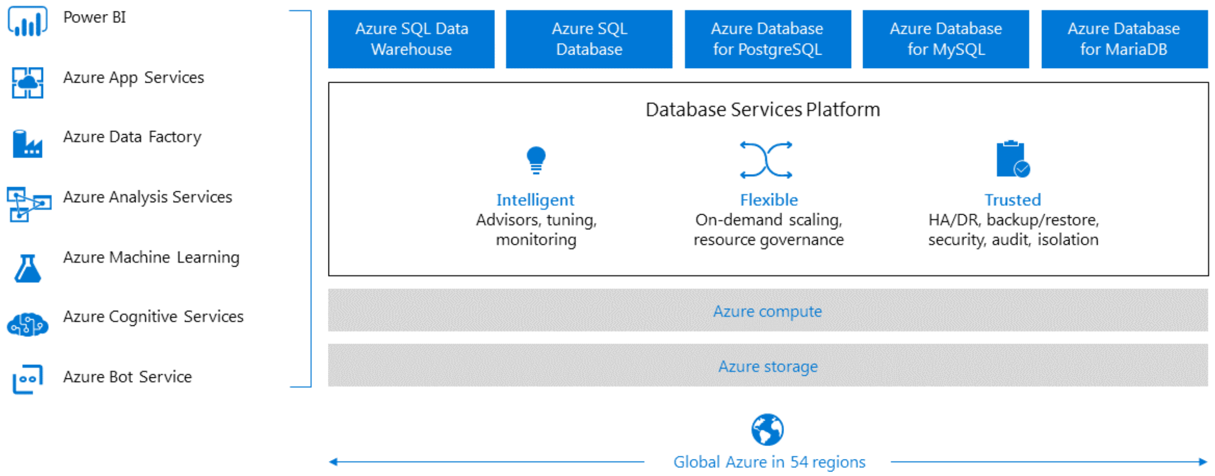
With Azure Database for MySQL, you can provision new databases in minutes and independently scale compute and storage, knowing that you'll get built-in high availability and a service-level agreement of 99.99 percent. In fact, Azure Database for MySQL provides everything you need under a simple, flexible pricing model, so you won't need to pay extra for things like automatic database patching and backups, monitoring capabilities, or essential security features.

These characteristics make Azure Database for MySQL ideal for hosting enterprise-scale production workloads. You can get started immediately and build a broad range of solutions, knowing that you won't be limited by infrastructure issues, scalability, availability concerns, geographic presence, or excessive upfront costs. The possibilities are endless, which is one reason why open-source database services are becoming so popular.

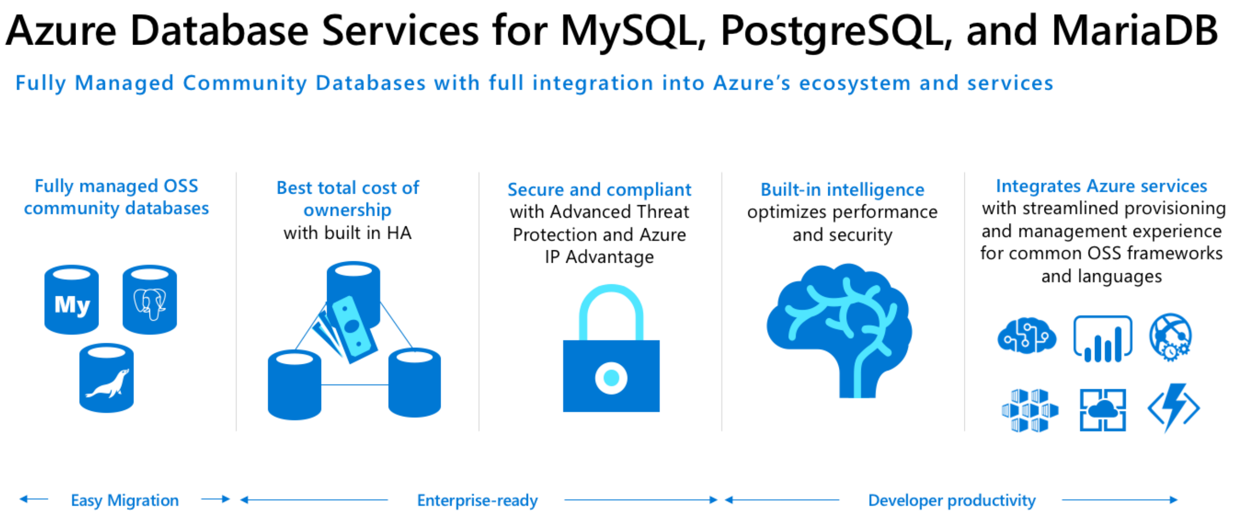
That said, we're seeing some patterns in the types of solutions that customers are building:

- Scalable e-commerce solutions that engage customers through customized offerings and process transactions quickly and securely
- Engaging and scalable web and mobile apps that are capable of reaching across platforms to connect with users on iOS, Android, Windows, or Mac computers
- Rich digital-marketing experiences, including digital campaigns that can automatically scale based on customer demand
- Financial management apps that securely store critical data, aggregating that information to deliver high-value analytics and insights with uncompromised performance
- Gaming experiences that can scale elastically, ensuring the ability to accommodate unexpected bursts of traffic and deliver consistent, low-latency, multiplayer experiences

Azure Database for MySQL is built on the same platform as Azure SQL Database and Azure SQL Data Warehouse, as are Azure Database for PostgreSQL and Azure Database for MariaDB, our other open-source database services. This means that you'll benefit from the same on-demand scalability no matter which service you choose. It also means that, where it makes sense, we'll be able to infuse the same intelligent features that we built for Azure SQL Database into Azure Database for MySQL that much faster. Advanced Threat Protection, one such feature that we've already started rolling out to Azure Database for MySQL, is a good example.



As an Azure data service, Azure Database for MySQL is backed by Azure compute and Azure Storage, and is built to work with the extensive range of other Azure services—including Azure Power BI Embedded, Azure App Service, and the Azure AI platform. And like all Azure services, Azure Database for MySQL is built on a world-class infrastructure with unmatched global reach; Azure has more global regions than any other cloud provider, enabling you to bring your apps closer to your users around the world, preserve data residency, and satisfy compliance demands. Finally, when you choose Azure Database for MySQL, you'll get comprehensive compliance and Azure IP Advantage, a program for Azure customers that provides protection against intellectual property (IP) risks


Azure Database Services for MySQL, PostgreSQL, and MariaDB

Fully Managed Community Databases with full integration into Azure's ecosystem and services

Fully managed OSS community databases
My
Best total cost of ownership with built in HA
Secure and compliant with Advanced Threat Protection and Azure IP Advantage
Built-in intelligence optimizes performance and security
Integrates Azure services with streamlined provisioning and management experience for common OSS frameworks and languages
Easy Migration
Enterprise-ready
Developer productivity

Key concepts

Following are some of the key concepts you'll need to understand to get the most out of Azure Database for MySQL. If you'd rather just jump in and learn via a hands-on approach, feel free to skip this chapter for now and jump to Getting Started with Azure Database for MySQL.

Databases and servers

Azure Database for MySQL exposes access and features at the server level. An Azure Database for MySQL *server,* which is created within an Azure subscription, is the parent resource for *one or more individual databases*. It collocates the computing resources for those databases in a specified Azure region, provides a namespace for the databases, and serves as a connection endpoint for server and database access.

An Azure Database for MySQL server also defines the scope for management policies that apply to those databases—including sign-in, firewall, users, roles, and configurations. You can create, manage, and delete Azure Database for MySQL servers and databases by using the Azure portal or the Azure CLI. The online documentation for Azure Database for MySQL provides more information on the concepts of databases and servers.

Server pricing tiers and resources

Pricing for an Azure Database for MySQL server is determined by the configuration of resources that you specify for that server, including the pricing tier, number of vCores, and amount of storage. Create a single database under the server, and that database has exclusive access to those resources; create multiple databases under the server, and those resources are shared.

You can create an Azure Database for MySQL server in one of three pricing tiers, which are differentiated by the amount of compute (vCores) that can be provisioned, the amount of memory per vCore, and the technology used for data storage. In general:

- **Basic** is good for workloads that require only light compute and I/O—such as test servers and small-scale, infrequently used applications.
- **General purpose** is good for workloads that require balanced compute and memory, with scalable I/O—including web apps, mobile apps, and many other enterprise apps. This is our most popular option, and it works well for most scenarios.
- **Memory optimized** is good for workloads that require in-memory performance for faster transaction processing and greater concurrency—including real-time data processing and high-performance transactional or analytical apps.

After you create a server, you can adjust the number of vCores, the hardware generation, and the pricing tier up or down, within seconds. Similarly, you can increase the amount of storage or adjust your backup retention period up or down without application downtime.

Note: Changing to and from the basic tier is not supported, so if you're planning to scale, we recommend you start with the general-purpose or memory-optimized tiers. Also, you can't change the backup storage type after a server is created.

Finally, it's worth pointing out that each option within each pricing tier has an associated set of limits, such as the number of concurrent connections.

High availability

Azure Database for MySQL provides guaranteed high availability to help you avoid application downtime—including a financially backed service-level agreement of 99.99 percent for all services that are in general availability. This high-availability model is based on built-in failover mechanisms that are triggered when a node-level interruption occurs, such as in the event of a hardware failure.

The entire failover process typically takes tens of seconds. Here's how it works: At all times, changes to an Azure Database for MySQL server occur in the context of a transaction, with changes recorded synchronously in Azure storage when the transaction is committed. If a node-level interruption occurs, the database server automatically creates a new node and attaches data storage to the new node. Any active connections are dropped, any in-flight transactions aren't committed, and an internal gateway transparently redirects any new connections to the new instance.

Because of how the failover process works, it's important that you build your database applications to detect and retry dropped connections and failed transactions. When your application retries, its connection is transparently redirected to the newly created instance. And because the redirect is handled transparently by the internal Azure gateway, your client application's connection string remains the same.

The same failover model makes it easy to scale compute resources up or down. When you do so, a new compute instance with the specified resources is created, and existing data storage is detached from the original instance and attached to the new instance. Client applications are disconnected, open, uncommitted transactions are canceled, and, after the client application retries the connection or makes a new connection, the gateway directs the connection to the newly sized instance.

The online documentation for Azure Database for MySQL provides more information on high availability.

Business continuity

Azure Database for MySQL supports locally redundant backups in the basic tier, and both locally redundant and geographically redundant backups in the general-purpose and memory-optimized tiers. You can take advantage of these features to provide business continuity, such as recovering a server after a user or application error, or in the unlikely event of an Azure regional datacenter outage. The online documentation provides a good overview of recovery scenarios and how they can affect your Estimated Recovery Time and Recovery Point Objective.

To understand how you can use backup and restore as part of a comprehensive business continuity strategy, it's worth taking a closer look at how these features work: Azure Database for MySQL automatically creates full, differential, and transaction-log backups, which you can use to restore a server to any point in time within your configured backup retention period. Generally, full backups occur weekly, differential backups occur twice a day, and transaction-log backups occur every five minutes. The default backup retention period is 7 days and can be configured for up to 35 days. All backups are secured using AES 256-bit encryption.

The basic tier offers only locally redundant backup storage, whereas the general-purpose and memory-optimized tiers provide the option to augment locally redundant backups with geographically redundant backups. When you choose geo-redundant backup storage, your backups aren't only stored within the region in which your server is hosted, but are also replicated to a paired datacenter in another region within the same geography, allowing you to restore your server in any Azure region in the event of a disaster. There’s a delay between when backups are taken and when they’re replicated to another region, so, if a disaster occurs, you may face some data loss.

When you provision a server, you need to decide whether to augment locally redundant backups with geo-redundant backups because you can’t change this option afterward. Whichever option you choose, Azure Database for MySQL provides up to 100 percent of your provisioned server storage as backup storage at no additional cost.

Performing a restore creates a new server from the original server's backups. Recovery time depends on several factors, including the size of the database or databases, transaction log size, network bandwidth, and the total number of databases being recovered in the same region at the same time. The estimated recovery time is usually less than 12 hours.

Point-in-time restore is available with either backup redundancy option, and it creates a new server in the same region as your original server. Geo-restore, which is available only if you configured your server for geo-redundant storage, allows you to restore your server to a different region.

The online documentation for Azure Database for MySQL provides additional detail on how backup and restore work.

Data access and security

With Azure Database for MySQL, you have several capabilities to help ensure safe and secure access.

Firewall rules

By default, all access to an Azure Database for MySQL server is blocked by its firewall. You can use firewall rules to specify which computers can access a server, based on the originating IP address for each request. These firewall rules apply to all databases on the logical server and are configured as ranges of IP addresses. You can also configure the firewall to allow access from all Azure services and all Azure subnets. Firewall rules can be configured by using the Azure portal or the Azure CLI.

The online documentation for Azure Database for MySQL provides more information about firewall rules, including connecting from the internet, connecting from Azure, programmatically managing firewall rules, and troubleshooting the database server firewalls.

Virtual network rules

Virtual network rules, a firewall security feature, control whether your Azure Database for MySQL server accepts communications sent from particular subnets on virtual networks. To create a virtual network rule, you must first have a virtual network and a virtual network service endpoint to reference. If you configure your database firewall to allow access from all Azure services and all Azure subnets, you can use virtual network rules to achieve much more granular control.

The online documentation for Azure Database for MySQL provides more information about virtual network rules, including their benefits, characteristics, limitations, and use with ExpressRoute.

Configuring SSL

Enforcing Secure Sockets Layer (SSL) connections between your client applications and an Azure Database for MySQL server encrypts all traffic between the two, which can help protect against “man-in-the-middle” attacks. Enforcement of SSL connections is enabled by default when a new Azure Database for MySQL server is provisioned. It’s worth noting, however, that some application frameworks don’t enable SSL by default during installation, so you’ll want to check this if your app is failing to connect to your database server.

The online documentation for Azure Database for MySQL provides more information on configuring SSL.

Advanced Threat Protection

Advanced Threat Protection for Azure Database for MySQL can help detect anomalous activities that may indicate unusual or potentially harmful attempts to access or exploit your databases. It's part of the Azure Advanced Threat Protection offering, a unified package of advanced security capabilities that's available at an additional cost.

With Advanced Threat Protection for Azure Database for MySQL, you'll receive an alert when suspicious database activities, potential vulnerabilities, and anomalous database access and query patterns are detected. It works with Azure Security Center, which provides details about suspicious activity and recommended actions to investigate and mitigate the threat.

The online documentation for Azure Database for MySQL provides more information about Advanced Threat Protection for Azure Database for MySQL, including how-to articles that cover how to set it up and begin using it.

Monitoring and tuning

Azure Database for MySQL provides several metrics that give you insight into server behavior, which you can use for both workload optimization and troubleshooting. All metrics have a one-minute frequency, and each metric provides 30 days of history. You can also enable logging on your server.

The online documentation for Azure Database for MySQL provides more information about the available metrics and logging features.

Replication

Azure Database for MySQL supports two types of replication:

- **Data-in replication,** which you can use to synchronize data from an existing MySQL server into the Azure Database for MySQL service. The existing MySQL server can be running on-premises, running in a virtual machine, or hosted by another cloud provider. You set up data-in replication by using stored procedures, which can be run in the MySQL Shell, MySQL Workbench, or any other tool that can be used to run queries against MySQL.
- **Read replicas,** which you can use to replicate data from an Azure Database for MySQL server (master) to up to five read-only servers (replicas).

Read replicas can help improve the performance and scale of read-intensive workloads, which can be isolated to the replicas while write workloads are directed to the master. A common scenario is to have business intelligence and analytical workloads use the read replica as the data source for reporting.

Replicas are updated asynchronously with the native replication technology in the MySQL engine. From a provisioning perspective, read replicas are considered new servers, and you manage them similarly to how you manage regular Azure Database for MySQL servers. For each replica, you're billed for the provisioned compute and storage.

Both data-in replication and read replicas in Azure Database for MySQL use the replication technology that's based on the binary log (binlog) file position native to MySQL. The online documentation for Azure Database for MySQL provides more information on data-in replication and read replicas.

Getting started with Azure Database for MySQL

The MySQL database engine can be used with many programming languages, the most prevalent being Java and PHP. It's the database component of the popular LAMP (Linux, Apache, MySQL, PHP) web application stack, which is used for leading open-source content management systems like WordPress, Drupal, and Joomla. MySQL is also used in many industries, including gaming.

The following resources can help you get started with Azure Database for MySQL—from creating your first database server to using the features that we covered in the Key concepts section.

Supported versions

Azure Database for MySQL is based on MySQL Community Edition, using the InnoDB storage engine. As of March 2019, Azure Database for MySQL supports the following versions:

- MySQL Version 5.6
- MySQL Version 5.7

Supported versions will change over time, so it's best to check the online documentation for MySQL versions that are currently supported.

Updates and upgrades

The Azure Database for MySQL service automatically manages patching for bug-fix version updates. Currently, minor version upgrades aren't supported. For example, upgrading from MySQL 5.6 to MySQL 5.7 isn't supported. If you want to upgrade to the next minor version, you can dump and restore a MySQL database to a server that was created with the new engine version.

Drivers and tools compatibility

Azure Database for MySQL is compatible with a wide variety of programming languages and drivers, including PHP, .NET, Node.js, Go, Python, and Java. It's best to check the online documentation for the latest information on supported drivers and version compatibility.

Your existing tools should continue to work with Azure Database for MySQL, as long as database manipulation operates within the confines of user permissions. Again, the online documentation is the best source of information about which management tools have been tested and found to be compatible with Azure Database for MySQL.

Regardless of which tools you choose, remember that you'll need to think about connection resiliency and how to handle transient errors.

5-minute quickstarts

Our 5-minute quickstarts can help you get started with Azure Database for MySQL in the time it takes to walk down the hall for a cup of coffee. To use them, you'll need an Azure subscription. If you don't have an Azure subscription, you can create a free Azure account before you begin.

Step 1: Create a server

The first thing you'll need to do is create an Azure Database for MySQL server. You have two options:

- Create an Azure Database for MySQL server by using the Azure portal shows you how to use the Azure portal to create an Azure Database for MySQL server within an Azure resource group. You'll also configure a server-level firewall rule, get connection information, connect to the MySQL database by using the mysql command-line tool and the MySQL

Workbench tool, and clean up resources when you're done.

- Create an Azure Database for MySQL server using Azure CLI shows you how to use Azure Cloud Shell to create an Azure resource group, and then an Azure Database for MySQL server. You'll also configure a server-level firewall rule, configure SSL settings, get connection information, connect to the MySQL database by using the mysql command-line tool and the MySQL Workbench tool, and clean up resources when you're done.

If you plan to continue to experiment with Azure Database for MySQL (such as by trying one or more of the following Connect and query quickstarts), you may want to skip the last part, which covers cleaning up resources, in the previous quickstarts.

Step 2: Connect and query

After you create a server, you can connect to and query its database in many different ways. The quickstarts in the following table are identified by language or tool. They assume you've already completed at least one of the previous quickstarts and that you're already familiar with whichever development environment you choose.

Language or tool	Scenario
PHP	Connect to an Azure Database for MySQL database using a PHP application, and then use SQL statements to query, insert, update, and delete data.
Java	Connect to an Azure Database for MySQL database by using a Java application with a JDBC driver, and then use SQL statements to query, insert, update, and delete data.
.NET	Connect to an Azure Database for MySQL database by using a C# application, and then use SQL statements to query, insert, update, and delete data.
Python	Connect to an Azure Database for MySQL database by using Python, and then use SQL statements to query, insert, update, and delete data.
Node.js	Connect to an Azure Database for MySQL database by using Node.js, and then use SQL statements to query, insert, update, and delete data.
Ruby	Connect to an Azure Database for MySQL database by using a Ruby application and the mysql2 gem, and then use SQL statements to query, insert, update, and delete data.
C++	Connect to an Azure Database for MySQL database by using a C++ application, and then use SQL statements to query, insert, update, and delete data.
Go	Connect to an Azure Database for MySQL database by using Go, and then use SQL statements to query, insert, update, and delete data.
MySQL Workbench	Connect to an Azure Database for MySQL database by using the MySQL Workbench application.

Step-by-step tutorials

After trying a quickstart or two, when you're ready to dive deeper, you can follow the step-by-step tutorials to design a database and then create a web app.

To design a database, choose one of the following tutorials:

- Design an Azure Database for MySQL database using the Azure portal walks you through using the Azure portal to create a database. You'll also configure a server firewall; use the mysql command-line tool to create the database; load sample data; and then query, update, and restore that data.
- Design an Azure Database for MySQL using Azure CLI walks you through using the Azure CLI and other utilities to create a database. You'll also configure a server firewall; use the mysql command-line tool to create the database; load sample data; and then query, update, and restore that data. You can use Azure Cloud Shell in the browser or install the Azure CLI on your own computer to run the code blocks in this tutorial.

After you're designed a database, you'll be ready to create a web app. Build a PHP and MySQL app in Azure teaches you how to create a PHP app in Azure and connect it to a MySQL database. When you're finished, you'll have a Laravel app running on Azure App Service on Windows. To deploy to App Service on Linux, see Build a PHP and MySQL app in Azure App Service on Linux.

If you're already familiar with WordPress, you can set up a WordPress site on Azure App Service that uses an Azure Database for MySQL database. *Note: This isn't a step-by-step tutorial. It uses the existing Azure Marketplace template to quickly get your WordPress application up and running.*

If you're interested in hosting your WordPress application in a container, you may want to watch the video about how to deploy a web app using a container and integrate it with Azure Database for MySQL, which is described in more detail in the following section.

How-to guides

After completing a few tutorials, you'll probably be ready to begin experimenting with Azure Database for MySQL on your own. Our online how-to guides can walk you through many of the things you'll likely want to do. There are too many to list in this document, but you can find the first one here, with the rest listed immediately below it in the navigation pane.

Videos

If you'd rather watch a few videos, here are some options:

- Azure Database for MySQL product videos are a great way to familiarize yourself with the database service.
- Learn about Azure Database for MySQL is the first in a series of videos that show how you can migrate your applications to Azure or build advanced, new applications with Azure Database for MySQL. This video briefly introduces the series and covers the features and benefits of this database service.
- Migrate your MySQL applications to Azure using the Azure Database Migration Service provides a detailed walkthrough and demo on how to migrate your application to Azure Database for MySQL with minimal downtime.
- Deploy a web app using a container and Azure Database for MySQL shows

how you can deploy and scale your web app in the cloud and demonstrates how Azure Database for MySQL works with Azure App Service. The demo shows how to deploy WordPress (which typically uses MySQL as a database) and how to make WordPress work with Azure Database for MySQL.

- Connect applications running in Kubernetes to Azure Database for MySQL shows how to connect container applications running in Kubernetes to Azure Database for MySQL by using the Open Service Broker for Azure, an implementation of the Open Service Broker API for Azure services. The video covers the rationale for using a Service Broker and how it can help you benefit from the built-in high availability and security offered by Azure Database for MySQL.

Replication

Azure Database for MySQL supports two types of replication:

- Data-in replication, which you can use to synchronize data from a MySQL server that's running on-premises, in virtual machines, or in a database service hosted by another cloud provider into the Azure Database for MySQL service. How to configure Azure Database for MySQL data-in replication walks you through creating an Azure Database for MySQL server to be used as a replica, configuring the master server, dumping and restoring the master server, linking the master and replica servers to start data-in replication, and relevant stored procedures.
- Read replicas, which you can use to replicate data from an Azure Database for MySQL server (master) to up to five read-only servers (replicas) within the same Azure region. The how-to guides provide step-by-step instructions on how to create and manage read replicas in Azure Database for MySQL by using the Azure portal or by using the Azure CLI.

Developing with Azure Kubernetes Service

Azure Kubernetes Service provides a managed Kubernetes cluster you can use in Azure. Connecting Azure Kubernetes Service and Azure Database for MySQL covers some options to consider when using Azure Kubernetes Service with Azure Database for MySQL, including accelerated networking and Open Service Broker for Azure.

Advanced Threat Protection

Advanced Threat Protection for Azure Database for MySQL can help detect anomalous activities that may indicate unusual or potentially harmful attempts to access or exploit your databases. Our how-to article on Advanced Threat Protection for Azure Database for MySQL walks you through setting up threat detection by using the Azure portal, exploring anomalous database activities, and exploring threat-detection alerts.

Additional resources

The following are some additional resources to help you get started with Azure Database for MySQL.

Limitations

Limitations in Azure Database for MySQL describes various types of limits in the database service—including maximum connections per pricing tier/number of vCores, storage engine support, data manipulation statement support, functional limitations, and current known issues.

Samples

Azure CLI samples for Azure Database for MySQL provides links to sample Azure CLI scripts for Azure Database for MySQL—including scripts to create a server and firewall rule, scale a server, change server configurations, restore a server, and enable and download server logs.

References

Azure CLI reference for Azure Database for MySQL provides a list of Azure CLI commands for managing Azure Database for MySQL servers, with links to more information on each command.

Azure Database for MySQL REST API reference covers the REST operations you can use with Azure Database for MySQL. You can use these REST operations to create, delete, manage, and list servers, server configurations, databases, server logs, and firewall rules, and to list all available REST operations.

Azure Database for MySQL data-in replication stored procedures covers the stored procedures needed to set up or remove data-in replication between a master and replica.

Azure Database Migration Guide

The Azure Database Migration Guide provides step-by-step guidance on migrating to various Azure data services, in addition to a selection of case studies and partner tools. You can just select your source and target to get started, or, if applicable, you can go directly to the scenario on migrating from MySQL to Azure Database for MySQL.

Other useful links

- Pricing information for Azure Database for MySQL
- Online documentation on Azure Database for MySQL
- Azure pricing calculator for Azure Database for MySQL
- Azure service health dashboard
- Azure product availability by region

Azure Database for MySQL blog

Conclusion

Microsoft Azure provides an end-to-end platform for open-source applications, including a fully managed MySQL database service for app developers. Azure Database for MySQL can be provisioned in minutes and scaled in seconds, with built-in high availability and a simple, flexible pricing model that includes automatic patching and backups, monitoring tools, essential security features, and more. And because it's a part of Azure, when you choose Azure Database for MySQL, you'll have everything you need to build great apps and deploy them across the globe.

➔ Sign up for a free Azure account and get started with Azure Database for MySQL today