Amazon RDS

Overview

Features

DB Engines

Pricing

Resources

FAQs

Customers

Partners

Amazon Relational Database Service (Amazon RDS) is a managed service that makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity, while managing time-consuming database administration tasks, freeing you up to focus on your applications and business.

Amazon RDS gives you access to the capabilities of a familiar MySQL, MariaDB, Oracle, SQL Server, or PostgreSQL database. This means that the code, applications, and tools you already use today with your existing databases should work seamlessly with Amazon RDS. Amazon RDS can automatically back up your database and keep your database software up to date with the latest version. You benefit from the flexibility of being able to easily scale the compute resources or storage capacity associated with your relational database instance. In addition, Amazon RDS makes it easy to use replication to enhance database availability, improve data durability, or scale beyond the capacity constraints of a single database instance for read-heavy database workloads. As with all Amazon Web Services, there are no up-front investments required, and you pay only for the resources you use.

**Q: Which relational database engines does Amazon RDS support?**

Amazon RDS supports Amazon Aurora, MySQL, MariaDB, Oracle, SQL Server, and PostgreSQL database engines.

**Q: What does Amazon RDS manage on my behalf?**

Amazon RDS manages the work involved in setting up a relational database: from provisioning the infrastructure capacity you request to installing the database software. Once your database is up and running, Amazon RDS automates common administrative tasks such as performing guidance on which solution is best for you.

**Q: How do I get started with Amazon RDS?**

To sign up for Amazon RDS, you must have an Amazon Web Services account. Create an account if you do not already have one. After you are signed up, please refer to the Amazon RDS documentation, which includes our Getting Started Guide.

Amazon RDS is part of the AWS Free Tier so that new AWS customers can get started with a managed database service in the cloud for free.

**Q: Are there hybrid or on-premises deployment options for Amazon RDS?**

Yes, you can run RDS on premises using Amazon RDS on Outposts and Amazon RDS on VMware. Please see the Amazon RDS on Outposts and Amazon RDS on VMware FAQs for additional information.

## Database Instances

**Q: What is a database instance (DB instance)?**

You can think of a DB instance as a database environment in the cloud with the compute and storage resources you specify. You can create and delete DB instances, define/refine infrastructure attributes of your DB instance(s), and control access and security via the AWS Management Console, Amazon RDS APIs, and AWS Command Line Interface. You can run one or

Once your DB instance is available, you can retrieve its endpoint via the DB instance description in the AWS Management Console, DescribeDBInstances API or describe-db-instances command. Using this endpoint you can construct the connection string required to connect directly with your DB instance using your favorite database tool or programming language. In order to allow network requests to your running DB instance, you will need to authorize access. For a detailed explanation of how to construct your connection string and get started, please refer to our Getting Started Guide.

**Q: How many DB instances can I run with Amazon RDS?**

By default, customers are allowed to have up to a total of 40 Amazon RDS DB instances. Of those 40, up to 10 can be Oracle or SQL Server DB instances under the "License Included" model. All 40 can be used for Amazon Aurora, MySQL, MariaDB, PostgreSQL and Oracle under the "BYOL" model. Note that RDS for SQL Server has a limit of up to 100 databases on a single DB instance to learn more see the Amazon RDS SQL Server User Guide.

If your application requires more DB instances, you can request additional DB instances via this request form.

**Q: How many databases or schemas can I run within a DB instance?**

- RDS for Amazon Aurora: No limit imposed by software
- RDS for MySQL: No limit imposed by software
- RDS for MariaDB: No limit imposed by software

securely.

**Q: What is a maintenance window? Will my DB instance be available during maintenance events?**

The Amazon RDS maintenance window is your opportunity to control when DB instance modifications, database engine version upgrades, and software patching occurs, in the event they are requested or required. If a maintenance event is scheduled for a given week, it will be initiated during the maintenance window you identify.

Maintenance events that require Amazon RDS to take your DB instance offline are scale compute operations (which generally take only a few minutes from start-to-finish), database engine version upgrades, and required software patching. Required software patching is automatically scheduled only for patches that are security and durability related. Such patching occurs infrequently (typically once every few months) and should seldom require more than a fraction of your maintenance window.

If you do not specify a preferred weekly maintenance window when creating your DB instance, a 30 minute default value is assigned. If you wish to modify when maintenance is performed on your behalf, you can do so by modifying your DB instance in the AWS Management Console, the ModifyDBInstance API or the modify-db-instance command. Each of your DB instances can have different preferred maintenance windows, if you so choose.

Running your DB instance as a Multi-AZ deployment can further reduce the impact of a maintenance event. Please refer to the Amazon RDS User Guide for more information on maintenance operations.

If you're using RDS for SQL Server, you can use the client side SQL Server traces to identify slow queries. For information on accessing server side trace file data, please refer to Amazon RDS User Guide.

## Database Engine Versions

**Q: Which relational database engine versions does Amazon RDS support?**

For the list of supported database engine versions, please refer to the documentation for each engine:

- Amazon RDS for MySQL
- Amazon RDS for MariaDB
- Amazon RDS for PostgreSQL
- Amazon RDS for Oracle
- Amazon RDS for SQL Server
- Amazon Aurora

**Q: How does Amazon RDS distinguish between “major” and “minor” DB engine versions?**

Refer to the FAQs page for each Amazon RDS database engine for specifics on version numbering.

Amazon RDS strives to keep your database instance up to date by providing you newer versions of the supported database engines. After a new version of a database engine is released by the vendor or development organization, it is thoroughly tested by our database engineering team before it is made available in Amazon RDS.

We recommend that you keep your database instance upgraded to the most current minor version as it will contain the latest security and functionality fixes. Unlike major version upgrades, minor version upgrades only include database changes that are backward-compatible with previous minor versions (of the same major version) of the database engine.

If a new minor version does not contain fixes that would benefit RDS customers, we may choose not to make it available in RDS. Soon after a new minor version is available in RDS, we will set it to be the preferred minor version for new DB instances.

To manually upgrade a database instance to a supported engine version, use the **Modify DB Instance** command on the AWS Management Console or the ModifyDBInstance API and set the **DB Engine Version** parameter to the desired version. By default, the upgrade will be applied or during your next maintenance window. You can also choose to upgrade immediately by selecting the **Apply Immediately** option in the console API.

If we determine that a new engine minor version contains significant bug fixes compared to a previously released minor version, we will schedule automatic upgrades for DB instances which have the **Auto Minor Version Upgrade** setting to “Yes”. These upgrades will be scheduled to occur during customer-specified maintenance windows.

Yes. You can do so by creating a DB snapshot of your existing DB instance, restoring from the DB snapshot to create a new DB instance, and then initiating a version upgrade for the new DB instance. You can then experiment safely on the upgraded copy of your DB instance before deciding whether or not to upgrade your original DB instance.

For more information about restoring a DB snapshot, refer to the Amazon RDS User Guide.

**Q: Does Amazon RDS provide guidelines for deprecating database engine versions that are currently supported?**

- We intend to support major version releases (e.g., MySQL 5.6, PostgreSQL 9.6) for at least 3 years after they are initially supported by Amazon RDS.
- We intend to support minor versions (e.g., MySQL 5.6.37, PostgreSQL 9.6.1) for at least 1 year after they are initially supported by Amazon RDS.

Periodically, we will deprecate major or minor engine versions. For major versions, this is typically when the version has moved to extended support or is no longer receiving software fixes or security updates. For minor versions, this is when a minor version has significant bugs or security issues that have been resolved in a later minor version.

While we strive to meet these guidelines, in some cases we may deprecate specific major or minor versions sooner, such as when there are security issues. In the unlikely event that such cases occur, Amazon RDS will automatically upgrade your database engine to address the issue. Specific circumstances may dictate different timelines depending on the issue being addressed.

**Q: What happens when an RDS DB engine version is deprecated?**

Amazon RDS

Overview

Features

DB Engines

Pricing

Resources

FAQs

Customers

Partners

## Billing

### Q: How will I be charged and billed for my use of Amazon RDS?

You pay only for what you use, and there are no minimum or setup fees. You are billed based on:

- DB instance hours – Based on the class (e.g. db.t2.micro, db.m4.large) of the DB instance consumed. Partial DB instance hours consumed are billed as full hours.
- Storage (per GB per month) – Storage capacity you have provisioned to your DB instance. If you scale your provisioned storage capacity within the month, your bill will be pro-rated.
- I/O requests per month – Total number of storage I/O requests you have *(for Amazon RDS Magnetic Storage and Amazon Aurora only)*
- Provisioned IOPS per month – Provisioned IOPS rate, regardless of IOPS consumed *(for Amazon RDS Provisioned IOPS (SSD) Storage only)*
- Backup Storage – Backup storage is the storage associated with your automated database backups and any customer-initiated database snapshots. Increasing your backup retention period or taking additional database snapshots increases the backup storage consumed by your database.
- Data transfer – Internet data transfer in and out of your DB instance.

For Amazon RDS pricing information, please visit the pricing section on the Amazon RDS product page.

### Q: When does billing of my Amazon RDS DB instances begin and end?

The storage provisioned to your DB instance for your primary data is located within a single Availability Zone. When your database is backed up, the backup data (including transactions logs) is geo-redundantly replicated across multiple Availability Zones to provide even greater levels of data durability. The price for backup storage beyond your free allocation reflects this extra replication that occurs to maximize the durability of your critical backups.

**Q: How will I be billed for Multi-AZ DB instance deployments?**

If you specify that your DB instance should be a Multi-AZ deployment, you will be billed according to the Multi-AZ pricing posted on the Amazon RDS pricing page. Multi-AZ billing is based on:

- Multi-AZ DB instance hours – Based on the class (e.g. db.t2.micro, db.m4.large) of the DB instance consumed. As with standard deployments in a single Availability Zone, partial DB instance hours consumed are billed as full hours. If you convert your DB instance deployment between standard and Multi-AZ within a given hour, you will be charged both applicable rates for that hour.
- Provisioned storage (for Multi-AZ DB instance) – If you convert your deployment between standard and Multi-AZ within a given hour, you will be charged the higher of the applicable storage rates for that hour.
- I/O requests per month – Total number of storage I/O requests you have. Multi-AZ deployments consume a larger volume of I/O requests than standard DB instance deployments, depending on your database write/read ratio. Write I/O usage associated with

## Free Tier

### Q: What does the AWS Free Tier for Amazon RDS offer?

The AWS Free Tier for Amazon RDS offer provides free use of Single-AZ Micro DB instances running MySQL, MariaDB, PostgreSQL, Oracle ("Bring-Your-Own-License (BYOL)" licensing model) and SQL Server Express Edition. The free usage tier is capped at 750 instance hours per month. Customers also receive 20 GB of General Purpose (SSD) database storage and 20 GB of backup storage for free per month.

### Q: For what time period will the AWS Free Tier for Amazon RDS be available to me?

New AWS accounts receive 12 months of AWS Free Tier access. Please see the AWS Free Tier FAQs for more information.

### Q: Can I run more than one DB instance under the AWS Free Usage Tier for Amazon RDS?

Yes. You can run more than one Single-AZ Micro DB instance simultaneously and be eligible for usage counted under the AWS Free Tier for Amazon RDS. However, any use exceeding 750 instance hours, across all Amazon RDS Single-AZ Micro DB instances, across all eligible database engines and regions, will be billed at standard Amazon RDS prices.

For example: if you run two Single-AZ Micro DB instances for 400 hours each in a single month, you will accumulate 800 instance hours of usage, of which 750 hours will be free. You will be billed for the remaining 50 hours at the standard Amazon RDS price.

**Q: What is a reserved instance (RI)?**

Amazon RDS reserved instances give you the option to reserve a DB instance for a one or three year term and in turn receive a significant discount compared to the on-demand instance pricing for the DB instance. There are three RI payment options -- No Upfront, Partial Upfront, All Upfront -- which enable you to balance the amount you pay upfront with your effective hourly price.

**Q: How are reserved instances different from on-demand DB instances?**

Functionally, reserved instances and on-demand DB instances are exactly the same. The only difference is how your DB instance(s) are billed: With Reserved Instances, you purchase a one or three year reservation and in return receive a lower effective hourly usage rate (compared with on-demand DB instances) for the duration of the term. Unless you purchase reserved instances in a Region, all DB instances will be billed at on-demand hourly rates.

**Q: How do I purchase and create reserved instances?**

You can purchase a reserved instance in the "Reserved Instance" section of the AWS Management Console for Amazon RDS. Alternatively, you can use the Amazon RDS API or AWS Command Line Interface to list the reservations available for purchase then purchase a DB instance reservation.

Once you have made a reserved purchase, using a reserved DB instance is no different than an On-Demand DB instance. Launch a DB instance using the same instance class, engine and region for which you made the reservation. As long as your reservation purchase is active, Amazon RDS will apply the reduced hourly rate for which you are eligible to the new DB instance.

and would like to reserve. If the reservation purchase is successful, Amazon RDS will automatically apply your new hourly usage charge to your existing DB instance.

**Q: If I sign up for a reserved instance, when does the term begin? What happens to my DB instance when the term ends?**

Pricing changes associated with a reserved instance are activated once your request is received while the payment authorization is processed. You can follow the status of your reservation on the AWS Account Activity page or by using the DescribeReservedDBInstances API or describe-reserved-db-instances command. If the one-time payment cannot be successfully authorized by the next billing period, the discounted price will not take effect.

When your reservation term expires, your reserved instance will revert to the appropriate On-Demand hourly usage rate for your DB instance class and Region.

**Q: How do I control which DB instances are billed at the reserved instance rate?**

The Amazon RDS operations for creating, modifying, and deleting DB instances do not distinguish between On-Demand and reserved instances. When computing your bill, our system will automatically apply your Reservation(s) such that all eligible DB instances are charged at the lower hourly reserved DB instance rate.

**Q: If I scale my DB instance class up or down, what happens to my reservation?**

Each reservation is associated with the following set of attributes: DB engine, DB instance class, Multi-AZ deployment option, license model and Region.

**Q: Can I move a reserved instance from one Region or Availability Zone to another?**

Each reserved instance is associated with a specific Region, which is fixed for the lifetime of the reservation and cannot be changed. Each reservation can, however, be used in any of the available AZs within the associated Region.

**Q: Are reserved instances available for Multi-AZ deployments?**

Yes. When you purchase a reserved instance, you can select the Multi-AZ option in the DB instance configuration available for purchase. In addition, if you are using a DB engine and license model that supports reserved instance size-flexibility, a Multi-AZ reserved instance will cover usage for two Single-AZ DB instances.

**Q: Are reserved instances available for read replicas?**

A DB instance reservation can be applied to a read replica, provided the DB instance class and Region are the same. When computing your bill, our system will automatically apply your Reservation(s), such that all eligible DB instances are charged at the lower hourly reserved instance rate.

**Q: Can I cancel a reservation?**

No, you cannot cancel your reserved DB instance and the one-time payment (if applicable) is not refundable. You will continue to pay for every hour during your Reserved DB instance term regardless of your usage.

application's compute, memory and storage needs. For information the about the DB instance classes available, please refer to the Amazon RDS User Guide.

**Q: How do I scale the compute resources and/or storage capacity associated with my Amazon RDS Database Instance?**

You can scale the compute resources and storage capacity allocated to your DB instance with the AWS Management Console (selecting the desired DB instance and clicking the **Modify** button), the RDS API, or the AWS Command Line Interface. Memory and CPU resources are modified by changing your DB Instance class, and storage available is changed when you modify your storage allocation. Please note that when you modify your DB Instance class or allocated storage, your requested changes will be applied during your specified maintenance window. Alternately, you can use the "apply-immediately" flag to apply your scaling requests immediately. Bear in mind that any other pending system changes will be applied as well.

Some older RDS for SQL Server instances may not be eligible for scaled storage. See the RDS for SQL Server FAQ for more information.

**Q: What is the hardware configuration for Amazon RDS storage?**

Amazon RDS uses EBS volumes for database and log storage. Depending on the size of storage requested, Amazon RDS automatically stripes across multiple EBS volumes to enhance IOPS performance. For MySQL and Oracle, for an existing DB instance, you may observe some I/O capacity improvement if you scale up your storage. You can scale the storage capacity allocated to your DB Instance using the AWS Management Console, the ModifyDBInstance API, or the modify-db-instance command.

thereby spreading your data across multiple DB instances.

**Q: What is Amazon RDS General Purpose (SSD) storage?**

Amazon RDS General Purpose (SSD) Storage is suitable for a broad range of database workloads that have moderate I/O requirements. With the baseline of 3 IOPS/GB and ability to burst up to 3,000 IOPS, this storage option provides predictable performance to meet the needs of most applications.

**Q: What is Amazon RDS Provisioned IOPS (SSD) storage?**

Amazon RDS Provisioned IOPS (SSD) Storage is an SSD-backed storage option designed to deliver fast, predictable, and consistent I/O performance. With Amazon RDS Provisioned IOPS (SSD) Storage, you specify an IOPS rate when creating a DB instance, and Amazon RDS provisions that IOPS rate for the lifetime of the DB instance. Amazon RDS Provisioned IOPS (SSD) Storage is optimized for I/O-intensive, transactional (OLTP) database workloads. For more details, please see the Amazon RDS User Guide.

**Q: What is Amazon RDS Magnetic storage?**

Amazon RDS magnetic storage is useful for small database workloads where data is accessed less frequently. Magnetic storage is not recommended for production database instances.

**Q: How do I choose among the Amazon RDS storage types?**

Choose the storage type most suited for your workload.

The automated backup feature of Amazon RDS enables point-in-time recovery of your DB instance. When automated backups are turned on for your DB Instance, Amazon RDS automatically performs a full daily snapshot of your data (during your preferred backup window) and captures transaction logs (as updates to your DB Instance are made). When you initiate a point-in-time recovery, transaction logs are applied to the most appropriate daily backup in order to restore your DB instance to the specific time you requested. Amazon RDS retains backups of a DB Instance for a limited, user-specified period of time called the retention period, which by default is 7 days but can be set to up to 35 days. You can initiate a point-in-time restore and specify any second during your retention period, up to the Latest Restorable Time. You can use the DescribeDBInstances API to return the latest restorable time for you DB instance, which is typically within the last five minutes. Alternatively, you can find the Latest Restorable Time for a DB instance by selecting it in the AWS Management Console and looking in the “Description” tab in the lower panel of the Console.

DB Snapshots are user-initiated and enable you to back up your DB instance in a known state as frequently as you wish, and then restore to that specific state at any time. DB Snapshots can be created with the AWS Management Console, CreateDBSnapshot API, or create-db-snapshot command and are kept until you explicitly delete them.

The snapshots which Amazon RDS performs for enabling automated backups are available to you for copying (using the AWS console or the copy-db-snapshot command) or for the snapshot restore functionality. You can identify them using the "automated" Snapshot Type. In addition, you can identify the time at which the snapshot has been taken by viewing the "Snapshot Created Time" field. Alternatively, the identifier of the "automated" snapshots also contains the time (in UTC) at which the snapshot has been taken.

automated backups, please refer to the Amazon RDS User Guide.

**Q: What is a backup window and why do I need it? Is my database available during the backup window?**

The preferred backup window is the user-defined period of time during which your DB Instance is backed up. Amazon RDS uses these periodic data backups in conjunction with your transaction logs to enable you to restore your DB Instance to any second during your retention period, up to the LatestRestorableTime (typically up to the last few minutes). During the backup window, storage I/O may be briefly suspended while the backup process initializes (typically under a few seconds) and you may experience a brief period of elevated latency. There is no I/O suspension for Multi-AZ DB deployments, since the backup is taken from the standby.

**Q: Where are my automated backups and DB snapshots stored and how do I manage their retention?**

Amazon RDS DB snapshots and automated backups are stored in S3.

You can use the AWS Management Console, the ModifyDBInstance API, or the modify-db-instance command to manage the period of time your automated backups are retained by modifying the RetentionPeriod parameter. If you desire to turn off automated backups altogether, you can do so by setting the retention period to 0 (not recommended). You can manage your user-created DB Snapshots via the "Snapshots" section of the Amazon RDS Console. Alternatively, you can see a list of the user-created DB Snapshots for a given DB Instance using the DescribeDBSnapshots API or describe-db-snapshots command and delete snapshots with the DeleteDBSnapshot API or delete-db-snapshot command.

Automated backups are deleted when the DB instance is deleted. Only manually created DB Snapshots are retained after the DB Instance is deleted.

## Security

### Q: What is Amazon Virtual Private Cloud (VPC) and how does it work with Amazon RDS?

Amazon VPC lets you create a virtual networking environment in a private, isolated section of the AWS cloud, where you can exercise complete control over aspects such as private IP address ranges, subnets, routing tables and network gateways. With Amazon VPC, you can define a virtual network topology and customize the network configuration to closely resemble a traditional IP network that you might operate in your own datacenter.

One way that you can take advantage of VPC is when you want to run a public-facing web application while still maintaining non-publicly accessible backend servers in a private subnet. You can create a public-facing subnet for your webservers that has access to the Internet, and place your backend RDS DB Instances in a private-facing subnet with no Internet access. For more information about Amazon VPC, refer to the Amazon Virtual Private Cloud User Guide.

### Q: How is using Amazon RDS inside a VPC different from using it on the EC2-Classic platform (non-VPC)?

If your AWS account was created before 2013-12-04, you may be able to run Amazon RDS in an Amazon Elastic Compute Cloud (EC2)-Classic environment. The basic functionality of Amazon RDS is the same regardless of whether EC2-Classic or EC2-VPC is used. Amazon RDS manages

Amazon RDS to create a new standby in another Availability Zone should the need arise. You need to do this even for Single-AZ deployments, just in case you want to convert them to Multi-AZ deployments at some point.

**Q: How do I create an Amazon RDS DB Instance in VPC?**

For a procedure that walks you through this process, refer to Creating a DB Instance in a VPC in the Amazon RDS User Guide.

**Q: How do I control network access to my DB Instance(s)?**

Visit the Security Groups section of the Amazon RDS User Guide to learn about the different ways to control access to your DB Instances.

**Q: How do I connect to an RDS DB Instance in VPC?**

DB Instances deployed within a VPC can be accessed by EC2 Instances deployed in the same VPC. If these EC2 Instances are deployed in a public subnet with associated Elastic IPs, you can access the EC2 Instances via the internet.

DB Instances deployed within a VPC can be accessed from the Internet or from EC2 Instances outside the VPC via VPN or bastion hosts that you can launch in your public subnet, or using Amazon RDS's Publicly Accessible option:

- To use a bastion host, you will need to set up a public subnet with an EC2 instance that acts as a SSH Bastion. This public subnet must have an internet gateway and routing rules that your DB instance into a VPC. See the Amazon RDS User Guide for more details. You can also take a snapshot of your DB Instance outside VPC and restore it to VPC by specifying the DB Subnet Group you want to use. Alternatively, you can perform a “Restore to Point in Time” operation as well.

**Q: Can I move my existing DB instances from inside VPC to outside VPC?**

Migration of DB Instances from inside to outside VPC is not supported. For security reasons, a DB Snapshot of a DB Instance inside VPC cannot be restored to outside VPC. The same is true with “Restore to Point in Time” functionality.

**Q: What precautions should I take to ensure that my DB Instances in VPC are accessible by my application?**

You are responsible for modifying routing tables and networking ACLs in your VPC to ensure that your DB instance is reachable from your client instances in the VPC.

For Multi-AZ deployments, after failover, your client EC2 instance and RDS DB Instance may be in different Availability Zones. You should configure your networking ACLs to ensure that cross-AZ communication is possible.

**Q: Can I change the DB Subnet Group of my DB Instance?**

An existing DB Subnet Group can be updated to add more subnets, either for existing Availability Zones or for new Availability Zones added since the creation of the DB Instance. Removing subnets from an existing DB Subnet Group can cause unavailability for instances if

For MySQL, the default privileges for the master user include: create, drop, references, event, alter, delete, index, insert, select, update, create temporary tables, lock tables, trigger, create view, show view, alter routine, create routine, execute, trigger, create user, process, show databases, grant option.

For Oracle, the master user is granted the "dba" role. The master user inherits most of the privileges associated with the role. Please refer to the Amazon RDS User Guide for the list of restricted privileges and the corresponding alternatives to perform administrative tasks that may require these privileges.

For SQL Server, a user that creates a database is granted the "db_owner" role. Please refer to the Amazon RDS User Guide for the list of restricted privileges and the corresponding alternatives to perform administrative tasks that may require these privileges.

**Q: Is there anything different about user management with Amazon RDS?**

No, everything works the way you are familiar with when using a relational database you manage yourself.

**Q: Can programs running on servers in my own data center access Amazon RDS databases?**

Yes. You have to intentionally turn on the ability to access your database over the internet by configuring Security Groups. You can authorize access for only the specific IPs, IP ranges, or subnets corresponding to servers in your own data center.

**Q: Can I encrypt connections between my application and my DB Instance using SSL/TLS?**

**Q: Can I encrypt data at rest on my Amazon RDS databases?**

Amazon RDS supports encryption at rest for all database engines, using keys you manage using AWS Key Management Service (KMS). On a database instance running with Amazon RDS encryption, data stored at rest in the underlying storage is encrypted, as are its automated backups, read replicas, and snapshots. Encryption and decryption are handled transparently. For more information about the use of KMS with Amazon RDS, see the Amazon RDS User's Guide.

You can also add encryption to a previously unencrypted DB instance or DB cluster by creating a DB snapshot and then creating a copy of that snapshot and specifying a KMS encryption key. You can then restore an encrypted DB instance or DB cluster from the encrypted snapshot.

Amazon RDS for Oracle and SQL Server support those engines' Transparent Data Encryption (TDE) technologies. For more information, see the Amazon RDS User's Guide for Oracle and SQL Server.

**Q: How do I control the actions that my systems and users can take on specific RDS resources?**

You can control the actions that your AWS IAM users and groups can take on RDS resources. You do this by referencing the RDS resources in the AWS IAM policies that you apply to your users and groups. RDS resources that can be referenced in an AWS IAM policy includes DB instances, DB snapshots, read replicas, DB security groups, DB option groups, DB parameter groups, event subscriptions and DB subnet groups. In addition, you can tag these resources to add additional metadata to your resources. By using tagging, you can categorize your resources (e.g. "Development" DB instances, "Production" DB instances, and "Test" DB instances), and write AWS already have an executed BAA, no action is necessary to begin using these services in the account(s) covered by your BAA. If you do not have an executed BAA with AWS, or have any other questions about HIPAA-compliant applications on AWS, please contact your account manager.

## Database Configuration

**Q: How do I choose the right configuration parameters for my DB Instance(s)?**

By default, Amazon RDS chooses the optimal configuration parameters for your DB Instance taking into account the instance class and storage capacity. However, if you want to change them, you can do so using the AWS Management Console, the Amazon RDS APIs, or the AWS Command Line Interface. Please note that changing configuration parameters from recommended values can have unintended effects, ranging from degraded performance to system crashes, and should only be attempted by advanced users who wish to assume these risks.

**Q: What are DB Parameter groups? How are they helpful?**

A database parameter group (DB Parameter Group) acts as a “container” for engine configuration values that can be applied to one or more DB Instances. If you create a DB Instance without specifying a DB Parameter Group, a default DB Parameter Group is used. This default group contains engine defaults and Amazon RDS system defaults optimized for the DB Instance you are running. However, if you want your DB Instance to run with your custom-specified engine configuration values, you can simply create a new DB Parameter Group, modify

## Multi-AZ Deployments

**Q: What does it mean to run a DB instance as a Multi-AZ deployment?**

When you create or modify your DB instance to run as a Multi-AZ deployment, Amazon RDS automatically provisions and maintains a synchronous “standby” replica in a different Availability Zone. Updates to your DB Instance are synchronously replicated across Availability Zones to the standby in order to keep both in sync and protect your latest database updates against DB instance failure. During certain types of planned maintenance, or in the unlikely event of DB instance failure or Availability Zone failure, Amazon RDS will automatically fail over to the standby so that you can resume database writes and reads as soon as the standby is promoted. Since the name record for your DB instance remains the same, your application can resume database operation without the need for manual administrative intervention. With Multi-AZ deployments, replication is transparent: you do not interact directly with the standby, and it cannot be used to serve read traffic. More information about Multi-AZ deployments is in the Amazon RDS User Guide.

**Q: What is an Availability Zone?**

Availability Zones are distinct locations within a Region that are engineered to be isolated from failures in other Availability Zones. Each Availability Zone runs on its own physically distinct, independent infrastructure, and is engineered to be highly reliable. Common points of failures like generators and cooling equipment are not shared across Availability Zones. Additionally, they are physically separate, such that even extremely uncommon disasters such as fires, tornados or flooding would only affect a single Availability Zone. Availability Zones within the same Region benefit from low-latency network connectivity.

of a DB instance component failure or loss of availability in one Availability Zone. For example, if a storage volume on your primary fails, Amazon RDS automatically initiates a failover to the standby, where all of your database updates are intact. This provides additional data durability relative to standard deployments in a single AZ, where a user-initiated restore operation would be required and updates that occurred after the latest restorable time (typically within the last five minutes) would not be available.

You also benefit from enhanced database availability when running your DB instance as a Multi-AZ deployment. If an Availability Zone failure or DB instance failure occurs, your availability impact is limited to the time automatic failover takes to complete. The availability benefits of Multi-AZ also extend to planned maintenance. For example, with automated backups, I/O activity is no longer suspended on your primary during your preferred backup window, since backups are taken from the standby. In the case of patching or DB instance class scaling, these operations occur first on the standby, prior to automatic fail over. As a result, your availability impact is limited to the time required for automatic failover to complete.

Another implied benefit of running your DB instance as a Multi-AZ deployment is that DB instance failover is automatic and requires no administration. In an Amazon RDS context, this means you are not required to monitor DB instance events and initiate manual DB instance recovery (via the RestoreDBInstanceToPointInTime or RestoreDBInstanceFromSnapshot APIs) in the event of an Availability Zone failure or DB instance failure.

**Q: Are there any performance implications of running my DB instance as a Multi-AZ deployment?**

you are using the Amazon RDS APIs, you would call the CreateDBInstance API and set the "Multi-AZ" parameter to the value "true." To convert an existing standard (single AZ) DB instance to Multi-AZ, modify the DB instance in the AWS Management Console or use the ModifyDBInstance API and set the Multi-AZ parameter to true.

**Q: What happens when I convert my RDS instance from Single-AZ to Multi-AZ?**

For the RDS for MySQL, MariaDB, PostgreSQL and Oracle database engines, when you elect to convert your RDS instance from Single-AZ to Multi-AZ, the following happens:

- A snapshot of your primary instance is taken
- A new standby instance is created in a different Availability Zone, from the snapshot
- Synchronous replication is configured between primary and standby instances

As such, there should be no downtime incurred when an instance is converted from Single-AZ to Multi-AZ. However, you may see increased latency while the data on the standby is caught up to match to the primary.

**Q: What events would cause Amazon RDS to initiate a failover to the standby replica?**

Amazon RDS detects and automatically recovers from the most common failure scenarios for Multi-AZ deployments so that you can resume database operations as quickly as possible without administrative intervention. Amazon RDS automatically performs a failover in the event of any of the following:

- Loss of availability in primary Availability Zone

Event Notifications to be notified when specific DB events occur.

**Q: What happens during Multi-AZ failover and how long does it take?**

Failover is automatically handled by Amazon RDS so that you can resume database operations as quickly as possible without administrative intervention. When failing over, Amazon RDS simply flips the canonical name record (CNAME) for your DB instance to point at the standby, which is in turn promoted to become the new primary. We encourage you to follow best practices and implement database connection retry at the application layer.

Failovers, as defined by the interval between the detection of the failure on the primary and the resumption of transactions on the standby, typically complete within one to two minutes. Failover time can also be affected by whether large uncommitted transactions must be recovered; the use of adequately large instance types is recommended with Multi-AZ for best results. AWS also recommends the use of Provisioned IOPS with Multi-AZ instances, for fast, predictable, and consistent throughput performance.

**Q: Can I initiate a “forced failover” for my Multi-AZ DB instance deployment?**

Amazon RDS will automatically fail over without user intervention under a variety of failure conditions. In addition, Amazon RDS provides an option to initiate a failover when rebooting your instance. You can access this feature via the AWS Management Console or when using the RebootDBInstance API call.

**Q: How do I control/configure Multi-AZ synchronous replication?**

**Q: After failover, my primary is now located in a different Availability Zone than my other AWS resources (e.g. EC2 instances). Should I be concerned about latency?**

Availability Zones are engineered to provide low latency network connectivity to other Availability Zones in the same Region. In addition, you may want to consider architecting your application and other AWS resources with redundancy across multiple Availability Zones so your application will be resilient in the event of an Availability Zone failure. Multi-AZ deployments address this need for the database tier without administration on your part.

**Q: How do DB Snapshots and automated backups work with my Multi-AZ deployment?**

You interact with automated backup and DB Snapshot functionality in the same way whether you are running a standard deployment in a Single-AZ or Multi-AZ deployment. If you are running a Multi-AZ deployment, automated backups and DB Snapshots are simply taken from the standby to avoid I/O suspension on the primary. Please note that you may experience increased I/O latency (typically lasting a few minutes) during backups for both Single-AZ and Multi-AZ deployments.

Initiating a restore operation (point-in-time restore or restore from DB Snapshot) also works the same with Multi-AZ deployments as standard, Single-AZ deployments. New DB instance deployments can be created with either the RestoreDBInstanceFromSnapshot or RestoreDBInstanceToPointInTime APIs. These new DB instance deployments can be either standard or Multi-AZ, regardless of whether the source backup was initiated on a standard or Multi-AZ deployment.

**Q: When would I want to consider using an Amazon RDS read replica?**

There are a variety of scenarios where deploying one or more read replicas for a given source DB instance may make sense. Common reasons for deploying a read replica include:

- Scaling beyond the compute or I/O capacity of a single DB instance for read-heavy database workloads. This excess read traffic can be directed to one or more read replicas.
- Serving read traffic while the source DB instance is unavailable. If your source DB Instance cannot take I/O requests (e.g. due to I/O suspension for backups or scheduled maintenance), you can direct read traffic to your read replica(s). For this use case, keep in mind that the data on the read replica may be "stale" since the source DB Instance is unavailable.
- Business reporting or data warehousing scenarios; you may want business reporting queries to run against a read replica, rather than your primary, production DB Instance.
- You may use a read replica for disaster recovery of the source DB instance, either in the same AWS Region or in another Region.

**Q: Do I need to enable automatic backups on my DB instance before I can create read replicas?**

Yes. Enable automatic backups on your source DB Instance before adding read replicas, by setting the backup retention period to a value other than 0. Backups must remain enabled for read replicas to work.

**Q: Which versions of database engines support Amazon RDS read replicas?**

### Q: How do I deploy a read replica for a given DB instance?

You can create a read replica in minutes using the standard CreateDBInstanceReadReplica API or a few clicks on the AWS Management Console. When creating a read replica, you can identify it as a read replica by specifying a SourceDBInstanceIdentifier. The SourceDBInstanceIdentifier is the DB Instance Identifier of the “source” DB Instance from which you wish to replicate. As with a standard DB Instance, you can also specify the Availability Zone, DB instance class, and preferred maintenance window. The engine version (e.g., PostgreSQL 9.3.5) and storage allocation of a read replica is inherited from the source DB instance. When you initiate the creation of a read replica, Amazon RDS takes a snapshot of your source DB instance and begins replication. As a result, you will experience a brief I/O suspension on your source DB instance as the snapshot occurs. The I/O suspension typically lasts on the order of one minute, and is avoided if the source DB instance is a Multi-AZ deployment (in the case of Multi-AZ deployments, snapshots are taken from the standby). Amazon RDS is also currently working on an optimization (to be released shortly) such that if you create multiple Read Replicas within a 30 minute window, all of them will use the same source snapshot to minimize I/O impact (“catch-up” replication for each Read Replica will begin after creation).

### Q: How do I connect to my read replica(s)?

You can connect to a read replica just as you would connect to a standard DB instance, using the DescribeDBInstance API or AWS Management Console to retrieve the endpoint(s) for you read replica(s). If you have multiple read replicas, it is up to your application to determine how read traffic will be distributed amongst them.

**Q: Can I use a read replica to enhance database write availability or protect the data on my source DB instance against failure scenarios?**

If you are looking to use replication to increase database write availability and protect recent database updates against various failure conditions, we recommend you run your DB instance as a Multi-AZ deployment. With Amazon RDS Read Replicas, which employ supported engines' native, asynchronous replication, database writes occur on a read replica after they have already occurred on the source DB instance, and this replication “lag” can vary significantly. In contrast, the replication used by Multi-AZ deployments is synchronous, meaning that all database writes are concurrent on the primary and standby. This protects your latest database updates, since they should be available on the standby in the event failover is required. In addition, with Multi-AZ deployments replication is fully managed. Amazon RDS automatically monitors for DB instance failure conditions or Availability Zone failure and initiates automatic failover to the standby (or to a read replica, in the case of Amazon Aurora) if an outage occurs.

**Q: Can I create a read replica with a Multi-AZ DB instance deployment as its source?**

Yes. Since Multi-AZ DB instances address a different need than read replicas, it makes sense to use the two in conjunction for production deployments and to associate a read replica with a Multi-AZ DB Instance deployment. The “source” Multi AZ-DB instance provides you with enhanced write availability and data durability, and the associated read replica would improve read traffic scalability.

**Q: Can I configure my Amazon RDS read replicas themselves Multi-AZ?**

of additional replication latency introduced as transactions are replicated from the master to the first tier replica and then to the second-tier replica.

*Amazon RDS for PostgreSQL, Amazon RDS for Oracle:* Read Replicas of Read Replicas are not currently supported.

**Q: Can my read replicas only accept database read operations?**

Read replicas are designed to serve read traffic. However, there may be use cases where advanced users wish to complete Data Definition Language (DDL) SQL statements against a read replica. Examples might include adding a database index to a read replica that is used for business reporting, without adding the same index to the corresponding source DB instance.

Amazon RDS for MySQL can be configured to permit DDL SQL statements against a read replica. If you wish to enable operations other than reads for a given read replica, modify the active DB parameter group for the read replica, setting the “read_only” parameter to “0.”

Amazon RDS for PostgreSQL does not currently support the execution of DDL SQL statements against a read replica.

**Q: Can I promote my read replica into a “standalone” DB Instance?**

Yes. Refer to the Amazon RDS User Guide for more details.

**Q: Will my read replica be kept up-to-date with its source DB instance?**

You can use the standard DescribeDBInstances API to return a list of all the DB Instances you have deployed (including Read Replicas), or simply click on the "Instances" tab of the Amazon RDS Console.

Amazon RDS allows you to gain visibility into how far a read replica has fallen behind its source DB instance. The number of seconds that the read replica is behind the master is published as an Amazon CloudWatch metric ("Replica Lag") available via the AWS Management Console or Amazon CloudWatch APIs. For Amazon RDS for MySQL, the source of this information is the same as that displayed by issuing a standard "Show Slave Status" MySQL command against the read replica. For Amazon RDS for PostgreSQL, you can use the pg_stat_replication view on the source DB instance to explore replication metrics.

Amazon RDS monitors the replication status of your Read Replicas and updates the Replication State field in the AWS Management console to "Error" if replication stops for any reason (e.g., attempting DML queries on your replica that conflict with the updates made on the master database instance could result in a replication error). You can review the details of the associated error thrown by the MySQL engine by viewing the Replication Error field and take an appropriate action to recover from it. You can learn more about troubleshooting replication issues in the Troubleshooting a Read Replica Problem section of the User Guide for Amazon RDS for MySQL or PostgreSQL.

If a replication error is fixed, the Replication State changes to Replicating.

**Q: I scaled the compute and/or storage capacity of my source DB instance. Should I scale the resources for associated read replica(s) as well?**

DeleteDBInstance API or AWS Management Console.

If you delete an Amazon RDS for PostgreSQL DB Instance that has read replicas, all Read Replicas will be promoted to standalone DB Instances and will be able to accept both read and write traffic. The newly promoted DB Instances will operate independently of one another. If you desire to delete these DB Instances in addition to the original source DB Instance, you must explicitly do so using the DeleteDBInstance API or AWS Management Console.

**Q: How much do read replicas cost? When does billing begin and end?**

A read replica is billed as a standard DB Instance and at the same rates. Just like a standard DB instance, the rate per “DB Instance hour” for a read replica is determined by the DB instance class of the read replica – please see pricing page for up-to-date pricing. You are not charged for the data transfer incurred in replicating data between your source DB instance and read replica within the same AWS Region.

Billing for a read replica begins as soon as the replica has been successfully created (i.e. when status is listed as “active”). The read replica will continue being billed at standard Amazon RDS DB instance hour rates until you issue a command to delete it.

## Enhanced Monitoring

**Q: What is Enhanced Monitoring for RDS?**

### Q: Which instance types are supported by Enhanced Monitoring?

Enhanced Monitoring supports every instance type except t1.micro and m1.small. The software uses a small amount of CPU, memory and I/O and for general purpose monitoring, we recommend switching on higher granularities for instances that are medium or larger. For non-production DB Instances, the default setting for Enhanced Monitoring is “off”, and you have the choice of leaving it disabled or modifying the granularity when it is on.

### Q: What information can I view on the RDS dashboard?

You can view all the system metrics and process information for your RDS DB Instances in a graphical format on the console. You can manage which metrics you want to monitor for each instance and customize the dashboard according to your requirements.

### Q: Will all the instances in my RDS account sample metrics at the same granularity?

No. You can set different granularities for each DB Instance in your RDS account. You can also choose the instances on which you want to enable Enhanced Monitoring as well as modify the granularity of any instance whenever you want.

### Q: How far back can I see the historical metrics on the RDS console?

You can see the performance values for all the metrics up to 1 hour back, at a granularity of up to 1 second, based on your settings.

### Q: How can I visualize the metrics generated by RDS Enhanced Monitoring in CloudWatch?

number of time periods. For more details on CloudWatch alarms, please visit the Amazon CloudWatch Developer Guide.

**Q: How do I integrate Enhanced Monitoring with my tool that I currently use?**

RDS Enhanced Monitoring provides a set of metrics formed as JSON payloads which are delivered into your CloudWatch Logs account. The JSON payloads are delivered at the granularity last configured for the RDS instance.

There are two ways you can consume the metrics via a third-party dashboard or application. Monitoring tools can use CloudWatch Logs Subscriptions to set up a near real time feed for the metrics. Alternatively, you can use filters in CloudWatch Logs to bridge metrics across to CloudWatch to and integrate your application with CloudWatch. Please visit Amazon CloudWatch Documentation for more details.

**Q: How can I delete historical data?**

Since Enhanced Monitoring delivers JSON payloads into a log in your CloudWatch Logs account, you can control its retention period just like any other CloudWatch Logs stream. The default retention period configured for Enhanced Monitoring in CloudWatch Logs is 30 days. For details on how to change retention settings, please visit Amazon CloudWatch Developer Guide.

**Q: What impact does Enhanced Monitoring have on my monthly bills?**

Since the metrics are ingested into CloudWatch Logs, your charges will be based on CloudWatch Logs data transfer and storage rates once you exceed CloudWatch Logs free tier. Pricing details

## Amazon RDS Proxy

**Q: What is Amazon RDS Proxy?**

Amazon RDS Proxy is a fully managed, highly available database proxy feature for Amazon RDS. RDS Proxy makes applications more scalable, more resilient to database failures, and more secure.

**Q: Why would I use Amazon RDS Proxy?**

Amazon RDS Proxy is a fully managed, highly available, and easy-to-use database proxy feature of Amazon RDS that enables your applications to: 1) improve scalability by pooling and sharing database connections; 2) improve availability by reducing database failover times by up to 66% and preserving application connections during failovers; and 3) improve security by optionally enforcing AWS IAM authentication to databases, and securely storing credentials in AWS Secrets Manager.

**Q: What use cases does Amazon RDS Proxy address?**

Amazon RDS Proxy addresses a number of use cases related to scalability, availability, and security of your applications, including:

Applications with unpredictable workloads: Applications that support highly variable workloads may attempt to open a burst of new database connections. Amazon RDS Proxy's connection governance allows you to gracefully scale applications dealing with unpredictable workloads by efficiently reusing database connections. First, RDS proxy enables multiple application connections to share a database connection for efficient use of database resources. Second, RDS build applications that can transparently tolerate database failures without needing to write complex failure handling code. RDS Proxy automatically routes traffic to a new database instance while preserving application connections. RDS Proxy also bypasses Domain Name System (DNS) caches to reduce failover times by up to 66% for RDS and Aurora Multi-AZ databases. During database failovers, the application may experience increased latencies and ongoing transactions may have to be retried.

Improved security and centralized credentials management: Amazon RDS Proxy aids you in building more secure applications by giving you a choice to enforce IAM based authentication with relational databases. RDS Proxy also enables you to centrally manage database credentials through AWS Secrets Manager.

**Q: When should I connect to the database directly versus using Amazon RDS Proxy?**

Depending on your workload, Amazon RDS Proxy can add an average of 5 milliseconds of network latency to query or transaction response time. If your application cannot tolerate 5 milliseconds of latency or does not need connection management and other features enabled by RDS Proxy, you may want your application to connect directly to the database endpoint.

**Q: How will serverless applications benefit from Amazon RDS Proxy?**

Amazon RDS Proxy transforms your approach to building modern serverless applications that leverage the power and simplicity of relational databases. First, RDS Proxy enables serverless applications to scale efficiently by pooling and reusing database connections. Second, with RDS Proxy, you no longer need to handle database credentials in your Lambda code. You can use the IAM execution role associated with your Lambda function to authenticate with RDS Proxy and

### Q: Can I access Amazon RDS Proxy using APIs?

- Yes. You can use Amazon RDS Proxy APIs to create a proxy and then define target groups to associate the proxy with specific database instances or clusters. For example:

```
aws rds create-db-proxy
        --db-proxy-name '…'
        --engine-family <mysql|postgresql>
        --auth [{}, {}]
        --role-arn '…'
        --subnet-ids {}
        --require-tls <true|false>
        --tags {}
aws rds register-db-proxy-targets
        --target-group-name '…'
        --db-cluster-identifier  '…'
        --db-instance-identifier '…'
```

## Learn more about RDS partners

Learn more »

Amazon RDS

Overview

Features

DB Engines

Pricing

Resources

FAQs

Customers

Partners

AMAZON DYNAMODB

Fast and flexible NoSQL database service for any scale

WHAT'S NEW WITH AWS

Learn about the latest products, services, and more

AWS INNOVATE AI/ML EDITION | FEBRUARY 19 | FREE ONLINE EVENT

Learn from AWS experts on how to drive innovation with AI & Machine Learning

Sign In to the Console

Learn About AWS

What Is AWS?

Resources for AWS

Getting Started

Training and Certification

Developers on AWS

Developer Center

SDKs & Tools

Amazon is an Equal Opportunity Employer: *Minority / Women / Disability / Veteran / Gender Identity / Sexual Orientation / Age.*

aws

Amazon RDS

Overview

Features

DB Engines

Pricing

Resources

FAQs

Customers

Partners