**Encrypt unencrypted RDS Instnace**

**Objectives:**

**1.** Signin to AWS Management Console

**2.** Create RDS MySQL instance (***uncheck - enabling encrypt option***)

**3.** Create a snapshot

**4.** Make a copy of the snapshot and encrypt it

**5.** Restore DB Instance from the encrypted snapshot

**6.** Change the name of the original DB Instance

**7.** Change the name of the Restored DB Instance to the original DB Instance name

**8.** Delete the original RDS Instance and snapshot

**9.** Delete AWS Resources

**Promoting read replica as Primary DB**

**Encrypting ReadReplicas**

You can't have an encrypted read replica of an unencrypted DB instance or an unencrypted read replica of an encrypted DB instance. Encrypted read replicas must be encrypted with the same KMS key as the source DB instance when both are in the same AWS Region.

**RDS:**

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the AWS Cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.

Amazon RDS DB instances

A *DB instance* is an isolated database environment running in the cloud. It is the basic building block of Amazon RDS. A DB instance can contain multiple user-created databases, and can be accessed using the same client tools and applications you might use to access a standalone database instance. DB instances are simple to create and modify with the AWS command line tools, Amazon RDS API operations, or the AWS Management Console.

DB instance classes

The DB instance class determines the computation and memory capacity of an Amazon RDS DB instance. The DB instance class that you need depends on your processing power and memory requirements.

## DB instance class types

Amazon RDS supports DB instance classes for the following use cases:

* [General-purpose](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.DBInstanceClass.html#Concepts.DBInstanceClass.Types.general-purpose)
* [Memory-optimized](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.DBInstanceClass.html#Concepts.DBInstanceClass.Types.memory)
* [Burstable-performance](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.DBInstanceClass.html#Concepts.DBInstanceClass.Types.burstable)

Amazon RDS DB instance storage

* DB instances for Amazon RDS for MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server use Amazon Elastic Block Store (Amazon EBS) volumes for database and log storage.

## Amazon RDS storage types

Amazon RDS provides three storage types: General Purpose SSD (also known as gp2 and gp3), Provisioned IOPS SSD (also known as io1), and magnetic (also known as standard).

The following list briefly describes the three storage types:

* **General Purpose SSD** – General Purpose SSD volumes offer cost-effective storage that is ideal for a broad range of workloads running on medium-sized DB instances. General Purpose storage is best suited for development and testing environments.
* **Provisioned IOPS SSD** – Provisioned IOPS storage is designed to meet the needs of I/O-intensive workloads, particularly database workloads, that require low I/O latency and consistent I/O throughput. Provisioned IOPS storage is best suited for production environments.

.

* **Magnetic** – Amazon RDS also supports magnetic storage for backward compatibility. We recommend that you use General Purpose SSD or Provisioned IOPS SSD for any new storage needs. The maximum amount of storage allowed for DB instances on magnetic storage is less than that of the other storage types.

# Regions, Availability Zones, and Local Zones

[**PDF**](https://docs.aws.amazon.com/pdfs/AmazonRDS/latest/UserGuide/rds-ug.pdf#Concepts.RegionsAndAvailabilityZones)[**RSS**](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/rdsupdates.rss)

Amazon cloud computing resources are hosted in multiple locations world-wide. These locations are composed of AWS Regions, Availability Zones, and Local Zones. Each AWS Region is a separate geographic area. Each AWS Region has multiple, isolated locations known as Availability Zones.

By using Local Zones, you can place resources, such as compute and storage, in multiple locations closer to your users. Amazon RDS enables you to place resources, such as DB instances, and data in multiple locations. Resources aren't replicated across AWS Regions unless you do so specifically.

Amazon operates state-of-the-art, highly-available data centers. Although rare, failures can occur that affect the availability of DB instances that are in the same location. If you host all your DB instances in one location that is affected by such a failure, none of your DB instances will be available.


   AWS Region
  

# Creating an Amazon RDS DB instance

The basic building block of Amazon RDS is the DB instance, where you create your databases. You choose the engine-specific characteristics of the DB instance when you create it. You also choose the storage capacity, CPU, memory, and so on, of the AWS instance on which the database server runs.

In **Templates**, choose the template that matches your use case. If you choose **Production**, the following are preselected in a later step:

* **Multi-AZ** failover option
* **Provisioned IOPS SSD (io1)** storage option
* **Enable deletion protection** option

# Creating Amazon RDS resources with AWS CloudFormation

Amazon RDS is integrated with AWS CloudFormation, a service that helps you to model and set up your AWS resources so that you can spend less time creating and managing your resources and infrastructure. You create a template that describes all the AWS resources that you want (such as DB instances and DB parameter groups), and AWS CloudFormation provisions and configures those resources for you.

When you use AWS CloudFormation, you can reuse your template to set up your RDS resources consistently and repeatedly. Describe your resources once, and then provision the same resources over and over in multiple AWS accounts and Regions.

# Connecting to an Amazon RDS DB instance

After Amazon RDS provisions your DB instance, use any standard client application or utility for your DB engine to connect to the DB instance. In the connection string, specify the DNS address from the DB instance endpoint as the host parameter. Also, specify the port number from the DB instance endpoint as the port parameter.

The endpoint is unique for each DB instance, and the values of the port and user can vary. The following list shows the most common port for each DB engine:

* MariaDB – 3306
* Microsoft SQL Server – 1433
* MySQL – 3306
* Oracle – 1521
* PostgreSQL – 5432

# Working with parameter groups

Database parameters specify how the database is configured. For example, database parameters can specify the amount of resources, such as memory, to allocate to a database.

You manage your database configuration by associating your DB instances and Multi-AZ DB clusters with parameter groups. Amazon RDS defines parameter groups with default settings. You can also define your own parameter groups with customized settings.

A DB parameter group acts as a container for engine configuration values that are applied to one or more DB instances. DB cluster parameter groups apply to Multi-AZ DB clusters only. In a Multi-AZ DB cluster, the settings in the DB cluster parameter group apply to all of the DB instances in the cluster. The default DB parameter group for the DB engine and DB engine version is used for each DB instance in the DB cluster.

# Creating an Amazon ElastiCache cluster using Amazon RDS DB instance settings

ElastiCache is a fully managed, in-memory caching service that provides microsecond read and write latencies that support flexible, real-time use cases. ElastiCache can help you accelerate application and database performance. You can use ElastiCache as a primary data store for use cases that don't require data durability, such as gaming leaderboards, streaming, and data analytics. ElastiCache helps remove the complexity associated with deploying and managing a distributed computing environment. You can use the Amazon RDS console for creating ElastiCache clusters.

Amazon ElastiCache works with both the Redis and Memcached engines.

# Stopping an Amazon RDS DB instance temporarily

Suppose that you use a DB instance intermittently, for temporary testing, or for a daily development activity. If so, you can stop your Amazon RDS DB instance temporarily to save money. While your DB instance is stopped, you are charged for provisioned storage (including Provisioned IOPS). You're also charged for backup storage, including manual snapshots and automated backups within your specified retention window. However, you're not charged for DB instance hours.

# Automatically connecting an EC2 instance and a DB instance

You can use the Amazon RDS console to simplify setting up a connection between an Amazon Elastic Compute Cloud (Amazon EC2) instance and a DB instance. Often, your DB instance is in a private subnet and your EC2 instance is in a public subnet within a VPC. You can use a SQL client on your EC2 instance to connect to your DB instance . The EC2 instance can also run web servers or applications that access your private DB instance .

Blue/Green Deploymentss

A blue/green deployment copies a production database environment in a separate, synchronized staging environment. By using Amazon RDS Blue/Green Deployments, you can make changes to the database in the staging environment without affecting the production environment. For example, you can upgrade the major or minor DB engine version, change database parameters, or make schema changes in the staging environment. When you are ready, you can promote the staging environment to be the new production database environment.

Cross-Region automated backups

By using backup replication in Amazon RDS, you can configure your RDS DB instance to replicate snapshots and transaction logs to a destination Region. When backup replication is configured for a DB instance, RDS starts a cross-Region copy of all snapshots and transaction logs when they're ready.

###### To enable backup replication for an existing DB instance

1. Sign in to the AWS Management Console and open the Amazon RDS console at <https://console.aws.amazon.com/rds/>.
2. In the navigation pane, choose **Automated backups**.
3. On the **Current Region** tab, choose the DB instance for which you want to enable backup replication.
4. For **Actions**, choose **Manage cross-Region replication**.
5. Under **Backup replication**, choose **Enable replication to another AWS Region**.
6. Choose the **Destination Region**.
7. Choose the **Replicated backup retention period**.
8. If you've enabled encryption on the source DB instance, choose the **AWS KMS key** for encrypting the backups.
9. Choose **Save**.

In the source Region, replicated backups are listed on the **Current Region** tab of the **Automated backups** page. In the destination Region, replicated backups are listed on the **Replicated backups** tab of the **Automated backups** page.

Cross-Region read replicas

By using cross-Region read replicas in Amazon RDS, you can create a MariaDB, MySQL, Oracle, PostgreSQL, or SQL Server read replica in a different Region from the source DB instance.


                Cross-Region read replica configuration
            