Amazon Web Services (AWS) customers rely on [Amazon Relational Database Service (Amazon RDS)](https://aws.amazon.com/rds/) to store their data for all kinds of workloads. For high availability (HA), you can use the Multi-AZ feature of Amazon RDS to provide additional resiliency by maintaining two copies of data across different Availability Zones (AZs). This feature is described in the blog post [Amazon RDS Under the Hood: Multi-AZ](https://aws.amazon.com/blogs/database/amazon-rds-under-the-hood-multi-az/).

Failures are rare, but as a best practice, applications should design around potential failures. The RDS Multi-AZ configuration is the recommended approach for production environments due to its ability to support low *RTO* (recovery time objective) and *RPO* (recovery point objective) requirements. RTO is the targeted amount of time for a recovery to complete in the event of failure. RPO is the targeted amount of time during which data is at risk for loss in the event of a failure.

With millions of active customers using AWS monthly, there are some customer workloads that do not require the level of HA provided by RDS Multi-AZ and the additional costs associated with it. These workloads might have more relaxed RTO and RPO requirements, and Single-AZ configurations might be sufficient to meet those needs. However, before embarking on a Single-AZ only solution, you should understand what recovery expectations and scenarios are available with a Single-AZ RDS instance.

This post describes Amazon RDS Single-AZ RTO and RPO expectations for [MySQL](https://aws.amazon.com/rds/mysql/), [MariaDB](https://aws.amazon.com/rds/mariadb/" \t "_blank), [PostgreSQL](https://aws.amazon.com/rds/postgresql/), [Oracle](https://aws.amazon.com/rds/oracle/), and [Microsoft SQL Server](https://aws.amazon.com/rds/sqlserver/) databases. [Amazon Aurora](https://aws.amazon.com/rds/aurora) uses a different technology and storage subsystem designed for the cloud. Its single instance recovery process and scenarios are described in the [Aurora FAQ](https://aws.amazon.com/rds/aurora/faqs/).

**Design concepts**

Each RDS instance runs on an [Amazon EC2](https://aws.amazon.com/ec2/) instance backed by an [Amazon EBS](https://aws.amazon.com/ebs/) volume for storage. RDS takes daily snapshots of the database, which are stored durably in [Amazon S3](https://aws.amazon.com/s3/) behind the scenes. It also regularly copies transaction logs to S3—up to 5-minute intervals—providing point-in time-recovery when needed. Point-in-time recovery is not automatic in RDS; you must trigger it, either manually or via a script as part of an event. This recovery is created in a new RDS instance.

RTO for recovery with an RDS Single-AZ instance failure can vary from minutes to hours. The duration depends on the size of the database and the failure and recovery approach required, as described later in this post.

RPO for recovery with an RDS Single-AZ instance failure is typically 5 minutes (the interval required for copying transaction logs to Amazon S3), but it can vary. You can confirm this by calling RDS:describe-db-instances:LatestRestorableTime*.*This service returns the latest time to which a database can be restored with point-in-time restore.

When we design and plan for failure with an RDS Single-AZ instance, we look at the following scenarios:

* Recoverable instance failure – The individual EC2 node suffered a hardware failure but could be recovered automatically by RDS.
* Non-recoverable instance failure – The individual EC2 node suffered a hardware failure but could *not* be recovered automatically by RDS.
* EBS volume failure – The EBS volume suffered a data loss failure.
* Availability Zone disruption – Failure at the Availability Zone level that would affect the RDS instance.

We discuss recovery expectations for these scenarios in the following sections.

**Recoverable instance failures**

An Amazon RDS instance failure occurs when the underlying EC2 instance suffers a failure. When this occurs, an event notification specific to the issue at hand is sent out to alert the customer (see [Using Amazon RDS Event Notification](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_Events.html) for details). However, the RDS instance status remains *available*.

RDS automatically tries to launch a new instance in the same Availability Zone, attach the EBS volume, and recover. In this scenario, RTO is typically under 30 minutes. In this case, RPO is zero because the EBS volume was recovered.

The EBS volume is in a single Availability Zone, and this recovery occurs in the same Availability Zone as the original instance.  The data and table changes in flight might not have been fully committed and completed before the failure occurred.  Because the RDS DNS name does not change for Single-AZ instances, the connection endpoints remain the same.

**Non-recoverable instance failures or EBS volume failures**

If RDS instance recovery attempts are unsuccessful, or the underlying EBS volume suffers a data loss failure, the instance state is set to *failed*. This situation requires a point-in-time recovery. You can automate the recovery via an AWS Lambda script, or you can do it manually.

The RTO timing requires starting up a new Amazon RDS instance and then applying all changes since the last backup. The RPO is typically 5 minutes, but you can find it by calling RDS:describe-db-instances:LatestRestorableTime. This time can vary from 10 minutes to hours, depending on the number of logs that need to be applied. It can only be determined by testing because it depends on the size of the database, the number of changes made since the last backup, and the workload levels on the database. The RDS backups and transaction logs are stored in Amazon S3, so this recovery can occur in any supported Availability Zone in the Region.

When a new RDS instance is created, a new DNS name is also created. You can point your application to the new DNS name. You also have the option to [rename the newly restored DB instance](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_RenameInstance.html) using the old DB instance’s endpoint name. You can do this when it is difficult or impossible to change the configuration of your application’s ODBC/JDBC settings. It requires you to delete the old failed instance first. You would therefore lose the ability for AWS Support to troubleshoot the root cause of the issue, so do it only if necessary.

**Availability Zone disruptions**

Availability Zone failures are unlikely, and are usually only temporary. If the Availability Zone failure is more permanent, the instance is set to a *failed*state. The recovery would work as described previously, and a new instance could be created in a different Availability Zone using point-in-time recovery. Because this Single-AZ instance was factored into the overall architecture, you need to perform this step yourself, either manually or by scripting, and the strategy for this recovery scenario would be part of your larger disaster recovery (DR) plans.

If the Availability Zone failure is temporary, the database will be down, but it remains in the *available* state. You are responsible for application-level monitoring or other third-party tools to detect this scenario. In this case, you could wait for the Availability Zone to recover, or you could choose to recover the instance to another Availability Zone with a point-in-time recovery.

The RTO would be the time it takes to start up a new RDS instance and then apply all the changes since the last backup. The RPO might be longer, up to the time the Availability Zone failure occurred.

**Considerations**

I/O performance is impacted during backup and snapshot operations on an RDS Single-AZ instance. For MariaDB, MySQL, Oracle, and PostgreSQL, all I/O to the RDS instance is briefly suspended during backups and snapshots. This is an important consideration when scheduling backups or performing snapshots.

For RDS Single-AZ instances, patching the OS affects availability and performance because there is no failover alternative. Patching occurs during maintenance windows, and the system is not available during the patch.

To maximize availability, Multi-AZ instances patch the OS on the standby first, and then on the master instance. In both cases, the system is not available during RDBMS updates.

A Multi-AZ RDS deployment is also required to apply to the [Amazon RDS Service Level Agreement](https://aws.amazon.com/rds/sla/).

**Conclusion**

AWS recommends that you use an RDS Multi-AZ deployment as part of its *Reliability* pillar in the [Well-Architected Framework](https://d1.awsstatic.com/whitepapers/architecture/AWS_Well-Architected_Framework.pdf) as a best practice. However, you can use an RDS Single-AZ instance when reduced availability requirements and lower cost are considerations. With this understanding of RTO and RPO expectations for Single-AZ RDS instances, you can make an informed decision based on availability, risk, and costs

#### **Amazon RDS SLA Exclusions**

The Service Commitment does not apply to any unavailability, suspension or termination of Amazon RDS, or any other Amazon RDS performance issues: (i) caused by factors outside of our reasonable control, including any force majeure event or Internet access or related problems beyond the demarcation point of Amazon RDS; (ii) that result from any voluntary actions or inactions from you or any third party; (iii) that result from instances belonging to the Micro DB instance class or other instance classes which have similar CPU and memory resource limitations; (iv) that result from you not following the [basic operational guidelines](http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_BestPractices.html) described in the Amazon RDS User Guide (e.g., overloading a database instance to the point it is inoperable, creating excessively large number of tables that significantly increase the recovery time etc.); (v) caused by underlying database engine software that lead to repeated database crashes or an inoperable database instance; (vi) that result in long recovery time due to insufficient IO capacity for your database workload; (vii) that result from your equipment, software or other technology and/or third party equipment, software or other technology (other than third party equipment within our direct control); (viii) that result from any maintenance as provided for pursuant to the AWS Agreement; or (ix) arising from our suspension and termination of your right to use Amazon RDS in accordance with the Agreement (collectively, the "Amazon RDS SLA Exclusions").

If availability is impacted by factors other than those explicitly used in our Monthly Uptime Percentage calculation, then we may issue a Service Credit considering such factors at our discretion.

#### **Definitions**

* "Monthly Uptime Percentage" for a given Multi-AZ instance is calculated by subtracting from 100% the percentage of 1 minute intervals during the monthly billing cycle in which the Multi-AZ instance was "Unavailable". If you have been running that Multi-AZ instance for only part of the month, your Multi-AZ instance is assumed to be 100% available for the portion of the month that it was not running. Monthly Uptime Percentage measurements exclude downtime resulting directly or indirectly from any Amazon RDS SLA Exclusion.
* "Multi-AZ instance" means an Amazon RDS for MySQL, MariaDB, Oracle, PostgreSQL, or SQL Server database instance with the Multi-AZ parameter set to true.
* A "Service Credit" is a dollar credit, calculated as set forth above, that we may credit back to an eligible account.
* "Unavailable" means that all connection requests to the running Multi-AZ instance fail during a 1 minute interval.

Somewhat this depends on your definition of RPO. Under most definitions of RPO it is the interval between backups and is measured in hours. It only applies to situations in which your live data is completely lost and you need to recover using a copy not maintained in real-time. Because the database also backs up the log file, RPO can be brought down to minutes (vs other kinds of data volumes). But the preferred mechanism is to use multiple synchronously maintained copies so that your RPO under all but the most extreme circumstances is 0.  
  
If an RDS database instance's volumes were to be lost (logical or physical corruption), requiring recreating it from backup, then the RPO for Single-AZ, Multi-AZ, and even Aurora is typically around 5 minutes. That is the target interval for RDS to perform log backups to S3, so on a database volume loss you could have 5 minutes of log data that is also lost. There is no way to change the log backup interval, though that might be an interesting feature to add (hint: they would almost certainly have to charge for this as it would take a significant increase in resources behind the scenes to accomplish this at scale).  
  
With Single-AZ the only live copy of your data is the EBS volume that holds the data for the instance. While EBS uses mirroring of data under the covers to provide durability and availability, there are several scenarios where you would have no choice other than to recover from backups. In this case you might want to apply the 5 minute log backup interval as your RPO.  
  
With Multi-AZ the odds of data loss go way down because you have a separate synchronous copy of the volume being maintained in a separate data center (AZ). If your primary instance fails, you failover to the secondary instance with **no** data loss. There are far fewer scenarios where recreating the database from backup would be required, but there are still a few. Since volume-level replication is used, a corruption on the primary's volume may be replicated to the secondary's volume. And as rare as this scenario is, it would necessitate recovery from backups. I believe most customers think of Multi-AZ as having an RTO of 1-2 minutes and an RPO of 0, since they lose no data on any common failure. Again putting this into more traditional terms, even if a natural disaster were to destroy the data center housing the primary, the secondary would take over with no data loss. So assuming an RPO of 0 makes sense.  
  
With Aurora the odds of data loss take another significant drop as it maintains 6 copies spread over 3 AZs, and it does that at a granularity of 10GB. So if something does become corrupt then it is a 10GB chunk of which there are 5 other copies plus backup information on S3, making it easy to transparently recover that one copy of the 10GB segment. There are almost no scenarios in which you would need to recreate the entire instance from backup. So truly an RPO of 0.  
  
Bottom line is that I think for availability purposes RDS offers an RPO of 0 minutes. The next step would be to decide if you have a separate RPO for disaster recovery purposes, and what your disaster recovery plan looks like. Maybe it is just backups, or cross-region snapshot copies, or cross-region read replicas. None of these can achieve an RPO of 1 minute BTW, but DR strategies rarely require that.

<https://blogs.cornell.edu/cloudification/category/devops/>