Scaling Your Amazon RDS Instance Vertically and Horizontally

by Marie Yap | on 10 OCT 2016 | in [Amazon RDS\*](https://aws.amazon.com/blogs/database/category/database/amazon-rds/), [Database\*](https://aws.amazon.com/blogs/database/category/database/), [RDS](https://aws.amazon.com/blogs/database/category/rds/), [RDS For MySQL\*](https://aws.amazon.com/blogs/database/category/database/amazon-rds/rds-for-mysql/), [RDS MySQL](https://aws.amazon.com/blogs/database/category/rds-mysql/) | [Permalink](https://aws.amazon.com/blogs/database/scaling-your-amazon-rds-instance-vertically-and-horizontally/) | [Comments](https://commenting.awsblogs.com/embed.html?disqus_shortname=aws-database-blog&disqus_identifier=123&disqus_title=Scaling+Your+Amazon+RDS+Instance+Vertically+and+Horizontally&disqus_url=https://aws.amazon.com/blogs/database/scaling-your-amazon-rds-instance-vertically-and-horizontally/) | [Share](https://aws.amazon.com/blogs/database/scaling-your-amazon-rds-instance-vertically-and-horizontally/)

*Marie Yap is a solutions architect for Amazon Web Services.*

As a managed service, [Amazon RDS](https://aws.amazon.com/rds/) takes care of the scaling of your relational database so your database can keep up with the increasing demands of your application or applications.

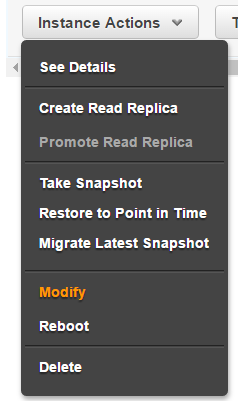
In this blog post, we are going to take a look into how we can vertically and horizontally scale your RDS instance. You can scale vertically to address the growing demands of an application that uses a roughly equal number of reads and writes. Or you can scale horizontally for read-heavy applications.

Vertical Scaling  
To handle a higher load in your database, you can vertically scale up your master database with a simple push of a button. There are currently over 18 instance sizes that you can choose from when resizing your RDS MySQL, PostgreSQL, MariaDB, Oracle, or Microsoft SQL Server instance. For Amazon Aurora, you have 5 memory-optimized instance sizes to choose from. The wide selection of instance types allows you to choose the best resource and cost for your database server.

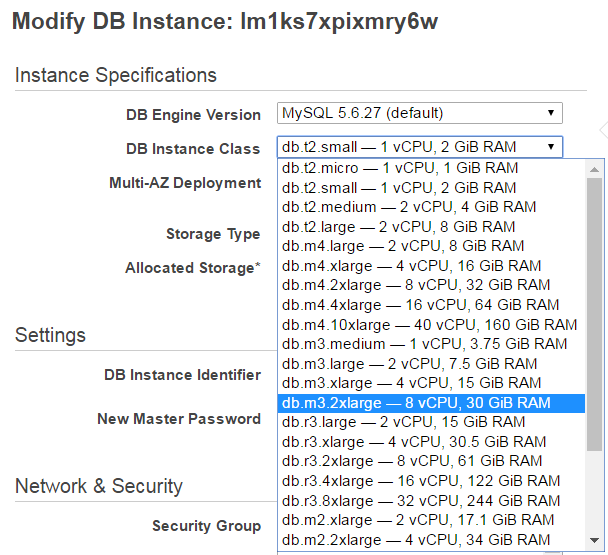
Following are some things to consider when scaling up an RDS instance:

* Before you scale, make sure you have the correct licensing in place for commercial engines (SQL Server, Oracle) especially if you Bring Your Own License (BYOL). One important thing to call out is that for commercial engines, you are restricted by the license, which is usually tied to the CPU sockets or cores.
* Determine when you want to apply the change. You have an option to apply the change immediately or during the maintenance window specified for the instance.
* Storage and instance type are decoupled. When you scale your database instance up or down, your storage size remains the same and is not affected by the change. You can separately modify your DB instance to increase the allocated storage space or improve the performance by changing the storage type (such as to General Purpose SSD to Provisioned IOPS SSD).
* There is minimal downtime when you are scaling up on a Multi-AZ environment because the standby database gets upgraded first, then a failover will occur to the newly sized database. A Single-AZ instance will be unavailable during the scale operation.

To change the instance type, choose Modify from the Instance Actions menu on the RDS console.



Then choose the new DB instance class.



Finally, determine if you want to apply the change immediately or not. To apply the change immediately, select the Apply Immediately check box at the bottom of the Modify page. If you don’t apply the change immediately, then the change will be scheduled to occur during the preferred maintenance window that you defined.

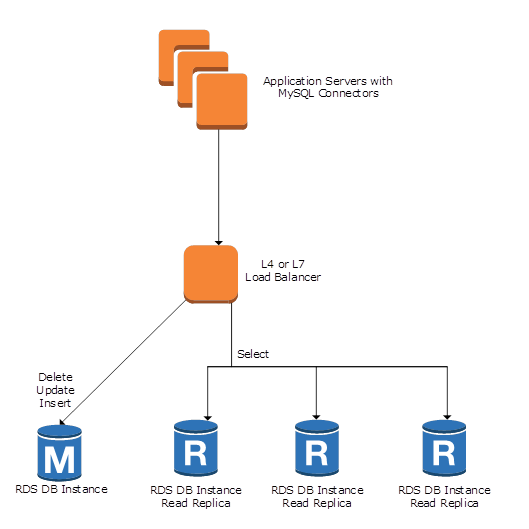
Horizontal Scaling  
In addition to scaling your master database vertically, you can also improve the performance of a read-heavy database by using read replicas to horizontally scale your database. RDS MySQL, PostgreSQL, and MariaDB can have up to 5 read replicas, and Amazon Aurora can have up to 15 read replicas.

Read replicas allow you to create read-only copies that are synchronized with your master database. You can also place your read replica in a different AWS Region closer to your users for better performance. Also, you can use read replicas to increase the availability of your database by promoting a read replica to a master for faster recovery in the event of a disaster. However, read replicas are not a replacement for the high availability and automatic failover capabilities that Multi-AZ provides.

Currently, RDS read replicas support transparent load balancing of queries or connections. Each replica has a unique Domain Name Service (DNS) endpoint so that an application can implement load balancing by connecting to the replica endpoint. Let’s look at the options on how we can make applications aware of RDS read replicas.

If your application is using the native MySQL driver, there are [MySQL Connectors](https://dev.mysql.com/doc/index-connectors.html) that allow you to do read/write splitting and read-only endpoint load balancing without a major change to your application. For example, if you have a PHP application, you can use the MySQL native driver’s PHP Mysqlnd replication and load-balancing plugin.

In addition to using a MySQL Connector, you can add a load balancer between your application and database servers. You make this addition so that you have a single database endpoint presented to the application. This approach allows for a more dynamic environment where you can transparently add or remove read replicas behind the load balancer without constantly updating the database connection string of the application. You can also perform a custom health check by using scripts.



As shown in the diagram, you can use a transport or a layer 4 load balancer together with the MySQL Connector. Currently, the Elastic Load Balancing (ELB) load balancer does not support the routing of traffic to RDS instances. Therefore, you might want to consider other options such as HAProxy, which is a open-source software-based load balancer that many people use. In this solution, you can configure HAProxy to listen on one port for read queries and another port for write queries.

Another option is to use a layer 7 SQL-aware load balancer, which allows you to forward queries to your databases using complex rules. This type of load balancer has a more sophisticated capability of understanding how to properly perform the read/write splits on multi-statements than a MySQL Connector does. This solution handles the scaling issues in a distributed database environment, so you don’t have to handle scaling on the application layer, resulting in little or no change to the application itself. To accomplish this, there are several open-source solutions (such as MaxScale, ProxySQL, and MySQL Proxy) and also commercial solutions, some of which can be found in the [AWS Marketplace](https://aws.amazon.com/marketplace).

Conclusion  
In summary, you can scale your RDS configuration up or out to meet the growing needs of your applications. RDS takes care of the heavy lifting in scaling your database so you can focus more on your application or applications.