Continuent Clustering

Comparison with Amazon RDS (MySQL and Aurora)

continuent

Overview

Enterprises require high availability for their business-critical applications. Even the smallest unplanned outage or even a planned maintenance operation can cause lost sales, productivity, and erode customer confidence. Additionally, updating and retrieving data needs to be robust to keep up with user demand.

Replicas and Failover

Both Continuent Clustering and RDS (when enabled) maintain multiple copies of your database. However, when using MySQL on RDS, there is a maximum of ONE candidate for failover, while Continuent can provide 2 or more (Aurora does provide additional failover candidates, each priced the same as the primary instance). More failover candidates translates to higher availability – your cluster is online even if you lose 2 (or more) of your database instances.

The failover process for RDS, both MySQL and Aurora, happens automatically and takes between 1 – 2 minutes. It also updates the DNS record for the database to point to the failover replica. As a result, the application will be offline for 1-2 minutes during a failover. In addition, there must be a process in place the bring the application back online. Your customers may experience errors or blank pages during this time.

Continuent Clustering handles failover transparent the application. Your applications stay connected, and failover is handled often within seconds in the background, with NO CHANGES required to your application, and without error pages to the end-user. After failover, there will still be additional candidates for failover available.

Performance and Scalability

RDS/MySQL provides “read replicas,” which, although not automatic failover candidates, are replicas of the primary instance and can be used for reading data, offloading some traffic from the master. Note that a read replica can be manually promoted to a master. A read replica will have a different IP address, thus to take advantage of using it for reads, your application must be designed to send reads to the read replica and writes to the master.

Note that the “failover replica” for MySQL/RDS (mentioned above) uses synchronous replication. This means that EACH write to primary database will block until the write has been committed and acknowledged on the failover replica. This will introduce high latency to your applications, and it could be significant for systems with a lot of writes.

Aurora improves on MySQL/RDS by offering replicas that share the same storage as the primary instance so that effectively there is no overhead for adding replicas. Again, the cost for this is equal to the cost of the primary instance.

To summarize, RDS for MySQL and Aurora offer read replicas to offload reads but will require changes to your application to use them. Also, enabling “Multi-AZ” in RDS for MySQL will add additional latency to your application.

Clustering Style

In a Continuent Cluster, a slave is not only a failover candidate, but can be used for reads as well. That 3-node cluster mentioned above already has 3 nodes available for reading, and once again, using the power of the Connector, reads can be automatically directed to slaves without modifying our application! If your application is already read/write aware, we can leverage your existing logic. If not, the Connector offers read/write algorithms for you to use.

Also note that by adding more slaves in a Continuent Cluster, you are scaling the number of nodes available for reads without making changes your application.

Maintenance

With maintenance tasks, you are in control with Continuent Clustering. Plan your maintenance when you want, and perform many maintenance tasks, like OS patches and MySQL upgrades, with no downtime. For instance, you can upgrade from MySQL 5.6 to MySQL 5.7 with NO downtime.

RDS for both MySQL and Aurora requires a maintenance window, and during that window, your instances may be restarted. This of course translates to application downtime.

Benefits of a True Cluster

Beyond providing high availability and performance scaling in a local cluster, Continuent clusters can be extended across regions, sites, clouds, and local infrastructure. For instance, you can have a primary cluster in AWS, but have a DR site in Google Cloud, and a second DR site on local infrastructure. Or have true multi-master, where two or more sites are active, replicating between each other and serving all of your customers locally.

The replication engine can easily be extended to replicate into data warehouse target such as Vertica, Hadoop, Redshift, Kafka and more, all supported by Continuent.

About Continuent

Continuent Ltd. is a leading provider of database clustering and replication, enabling enterprises to run business-critical applications on cost-effective open source software. Our customers represent many innovative and successful organizations throughout the world, handling billions of transactions daily across a wide range of industries.

For more information on our products and services, please visit www.continuent.com, email us at sales@continuent.com or call us at (800) 270-9035, and follow us on Twitter @Continuent.