Active-Passive Geo-Distributed Redis (Replica-Of)

In Redis Enterprise, active-passive geo-distribution provides applications read-only access to replicas of the data set from different geographical locations. The Redis Enterprise implementation of active-passive replication is called Replica Of.

In Replica Of, an administrator designates a database as a replica (destination) of one or more databases (sources). After the initial data load from source to destination is completed, all write commands are synchronized from the sources to the destination. Replica Of lets you distribute the read load of your application across multiple databases or synchronize the database, either within Redis Enterprise or external to Redis Enterprise, to another database.

You can create Active-Passive databases on Redis Enterprise Software or Redis Cloud.

Active-Active Geo-Distribution (CRDB) provides these benefits and also provides write access to all of the database replicas.

⚠️ **Warning** - Configuring a database as a replica of the database that it replicates creates a cyclical replication and is not supported.

The Replica Of is defined in the context of the destination database by specifying the source databases.

A destination database can have a maximum of thirty-two (32) source databases.

If only one source is defined, then the command execution order in the source is kept in the destination. However, when multiple sources are defined, commands that are replicated from the source databases are executed in the order in which they reach the destination database. As a result, commands that were executed in a certain order when compared across source databases might be executed in a different order on the destination database.

💡 **Note:** The Replica Of feature should not be confused with the in-memory Database replication feature, which is used for creating a master / slave configuration that enables ensuring database high-availability.

For a quick overview of Replica Of capabilities watch this quick video.
Replication process

When a database is defined as a replica of another database, all its existing data is deleted and replaced by data that is loaded from the source database.

Once the initial data load is completed, an ongoing synchronization process takes place to keep the destination always synchronized with its source. During the ongoing synchronization process, there is a certain delay between the time when a command was executed on the source and when it is executed on the destination. This delay is referred to as the Lag.

When there is a synchronization error, the process might stop or it might continue running on the assumption that the error automatically resolves. The result depends on the error type. See more details below.

In addition, the user can manually stop the synchronization process.

When the process is in the stopped state - whether stopped by the user or by the system - the user can restart the process. Restarting the process causes the synchronization process to flush the DB and restart the process from the beginning.

Replica Of status

The replication process can have the following statuses:

- **Syncing** - indicates that the synchronization process has started from scratch. Progress is indicated in percentages (%).
- **Synced** - indicates that the initial synchronization process was completed and the destination is synchronizing changes on an ongoing basis. The Lag delay in synchronization with the source is indicated as a time duration.
- **Sync stopped** - indicates that the synchronization process is currently not running and the user needs to restart it in order for it to continue running. This status happens if the user stops the process, or if certain errors arose that prevent synchronization from continuing without manual intervention. See more details below.

The statuses above are shown for the source database. In addition, a timestamp is shown on the source indicating when the last command from the source was executed on the destination.

The system also displays the destination database status as an aggregate of the statuses of all the sources.

**Note:** If you encounter issues with the Replica Of process, refer to the troubleshooting section.

Synchronization errors

Certain errors that occur during the synchronization process require user intervention for their resolution. When such errors occur, the synchronization process is automatically stopped.

For other errors, the synchronization process continues running on the assumption that the error automatically resolves.

Examples of errors that require user intervention for their resolution and that stop the synchronization process include:

- Error authenticating with the source database.
- Cross slot violation error while executing a command on a sharded destination database.
- Out-of-memory error on a source or on the destination database.

Example of an error that does not cause the synchronization process to stop:

- Connection error with the source database. A connection error might occur occasionally, for example as result of temporary network issues that get resolved. Depending on the connection error and its duration the process might be able to start syncing from the last point it reached (partial sync) or require a complete resynchronization from scratch across all sources.
Encryption

Replica Of supports the ability to encrypt uni-directional replication communications between source and destination clusters utilizing TLS 1.2 based encryption.

Data compression for Replica Of

When the Replica Of is defined across different Redis Enterprise Software clusters, it may be beneficial to compress the data that flows through the network (depending on where the clusters physically reside and the available network).

Compressing the data reduces the traffic and can help:

- Resolve throughput issues
- Reduce network traffic costs

Compressing the data does have trade-offs, which is why it should not always be turned on by default. For example:

- It uses CPU and disk resources to compress the data before sending it to the network and decompress it on the other side.
- It takes time to compress and decompress the data which can increase latency.
- Replication is disk-based and done gradually, shard by shard in the case of a multi-shard database. This may have an impact on replication times depending on the speed of the disks and load on the database.
- If traffic is too fast and the compression takes too much time it can cause the replication process to fail and be restarted.

It is advised that you test compression out in a lower environment before enabling it in production.

In the Redis Enterprise Software management UI, when designating a Replica Of source from a different Redis Enterprise Software cluster, there is also an option to enable compression. When enabled, gzip compression with level -6 is utilized.

Database clustering (sharding) implications

If a source database is sharded, that entire database is treated as a single source for the destination database.

If the destination database is sharded, when the commands replicated from the source are executed on the destination database, the destination database’s hashing function is executed to determine to which shard/s the command refers.

The source and destination can have different shard counts and functions for placement of keys.

Synchronization in Active-Passive Replication

In Active-Passive databases, one cluster hosts the source database that receives read-write operations and the other clusters host destination databases that receive synchronization updates from the source database.

When there is a significant difference between the source and destination databases, the destination database flushes all of the data from its memory and starts synchronizing the data again. This process is called a full sync.

For example, if the database updates for the destination databases that are stored by the destination database in a synchronization backlog exceed their allocated memory, the source database starts a full sync.

**Warning** - When you failover to the destination database for write operations, make sure that you disable Replica Of before you direct clients to the destination database. This avoids a full sync that can overwrite your data.

Updated: November 9, 2020