# Google Cloud SQL database disaster recovery

In Google Cloud, database disaster recovery (DR) is about providing continuity of processing, specifically when a region fails or becomes unavailable. Cloud SQL (https://cloud.google.com/sql/docs/features?authuser=0)) is a regional service (when Cloud SQL is configured for HA (https://cloud.google.com/sql/docs/mysql/high-availability?authuser=0)). Therefore, if the Google Cloud region that hosts a Cloud SQL database becomes unavailable, then the Cloud SQL database also becomes unavailable.

To continue processing, you must make the database available in a secondary region as soon as possible. The DR plan for both Cloud SQL for MySQL and Cloud SQL for PostgreSQL requires you to configure a cross-region read replica in Cloud SQL. A failover based on export/import is also possible, but that approach takes longer, especially for large databases.

The following business scenarios are examples that warrant a cross-region failover configuration:

- The service level agreement of the business application is greater than the regional Cloud SQL Service Level Agreement (https://cloud.google.com/sql/sla?authuser=0) (99.95% availability). By failing over to another region, you can mitigate an outage.
- All tiers of the business application are already multi-regional and can continue processing when a region outage occurs. The cross-region failover configuration ensures the continued availability of a database.
- The required recovery time objective (RTO) and recovery point objective (RPO) are in minutes rather than in hours. Failing over to another region is faster than recreating a database.

In general, there are two variants for the DR process:

- A database fails over to a secondary region. After the database is ready and used by an application, it becomes the new primary database and remains the primary database.
- A database fails over to a secondary region but falls back to the primary region after the primary region is recovered from its failure.

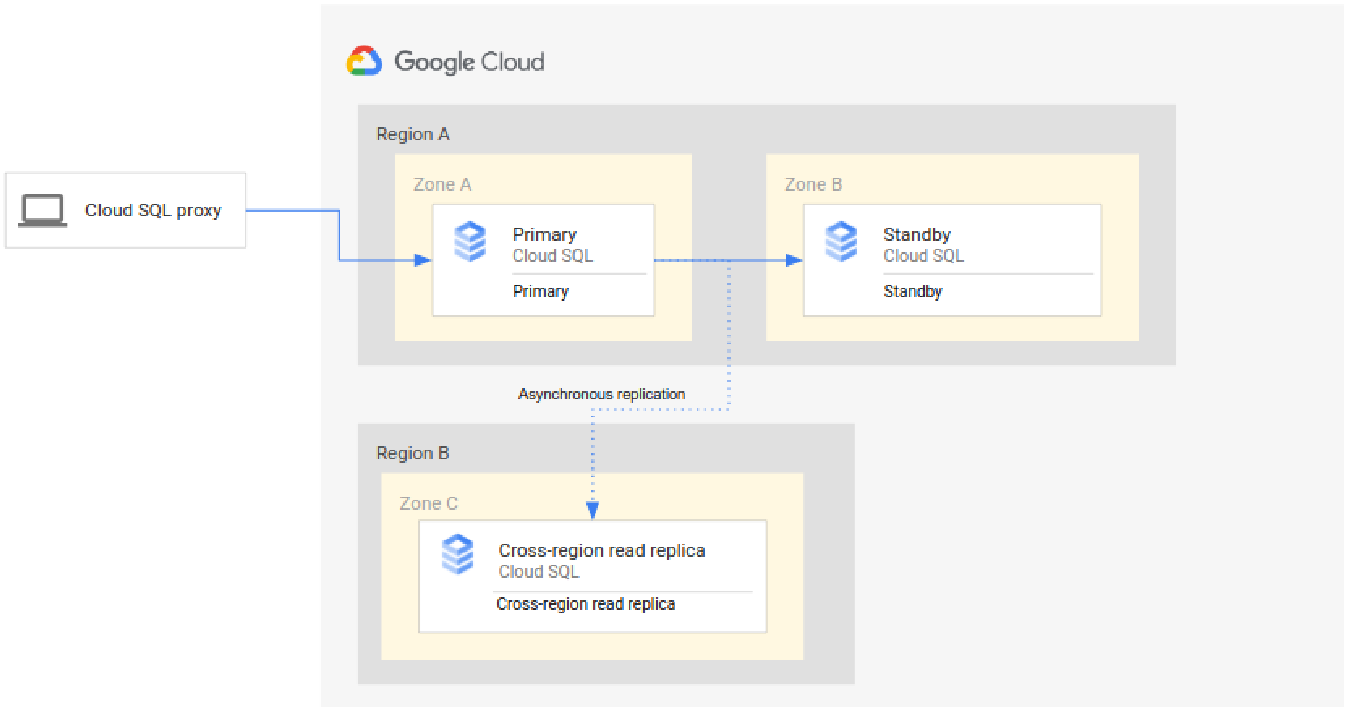
This Google Cloud SQL database disaster recovery overview describes the second variant—when a failed database is recovered and falls back to the primary region. This DR process variant is especially relevant for databases that must run in the primary region because of network latency, or because some resources are available only in the primary region. With this variant, the database runs in the secondary region only for the duration of outage in the primary region.

There are two tutorials associated with this DR recovery article:

- Cloud SQL for MySQL disaster recovery: A complete failover and fallback process (https://cloud.google.com/solutions/cloud-sql-mysql-disaster-recovery-complete-failover-fallback?authuser=0)
- Cloud SQL for PostgreSQL disaster recovery: A complete failover and fallback process (https://cloud.google.com/architecture/cloud-sql-postgres-disaster-recovery-complete-failover-fallback?authuser=0)

## DR architecture

The following diagram shows the minimal architecture that supports database DR for an HA Cloud SQL instance:



The architecture works as follows:

- Two instances of Cloud SQL (a primary instance and a standby instance) are located in two separate zones within a single region (the primary region). The instances are synchronized by using regional persistent disks.

- One instance of Cloud SQL (the cross-region read replica) is located in a second region (the secondary region). For DR, the cross-region read replica is set up to synchronize (by using asynchronous replication) with the primary instance using a read replica setup.

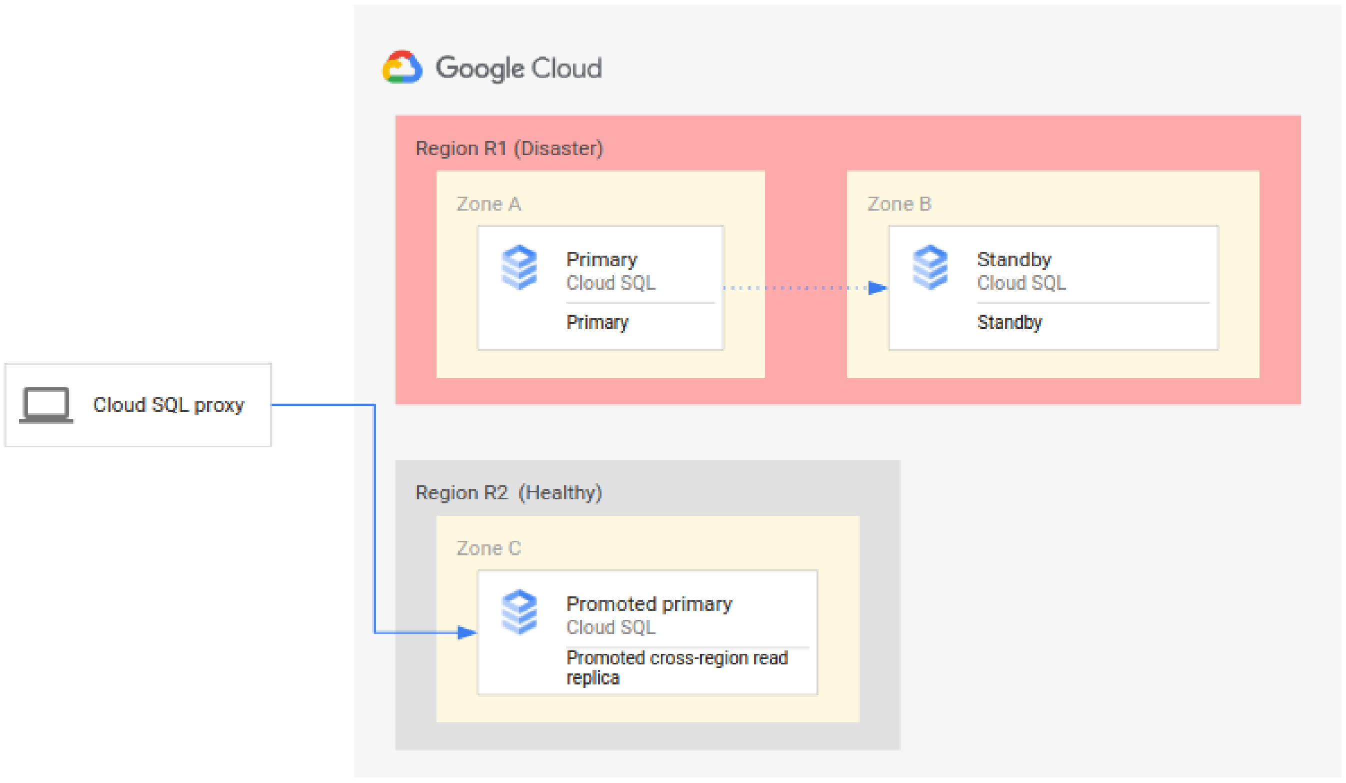
The primary and standby instances share the same regional disk, and so their states are identical.

Because this setup uses asynchronous replication, it's possible that the cross-region read replica lags behind the primary instance. When a failover occurs, the cross-region read replica might support an RPO of zero.

## Overview of the basic DR process

The basic DR process starts when the primary region becomes unavailable and the primary instance fails over to resume processing in a secondary region. The DR process prescribes the operational steps that must be performed, either manually or automatically (#comparison), to mitigate the regional failure and establish a running primary instance in a secondary region.

The following diagram shows the DR process:



The preceding DR process consists of the following steps:

1. The primary region (R1), which is running the primary instance, becomes unavailable.
2. The operations team recognizes and formally acknowledges the disaster and decides whether a failover is required.
3. If a failover is required, the cross-region read replica in the secondary region (R2) is made the new primary instance.
4. Client connections are reconfigured to resume processing on the new primary instance and access the primary instance in R2.

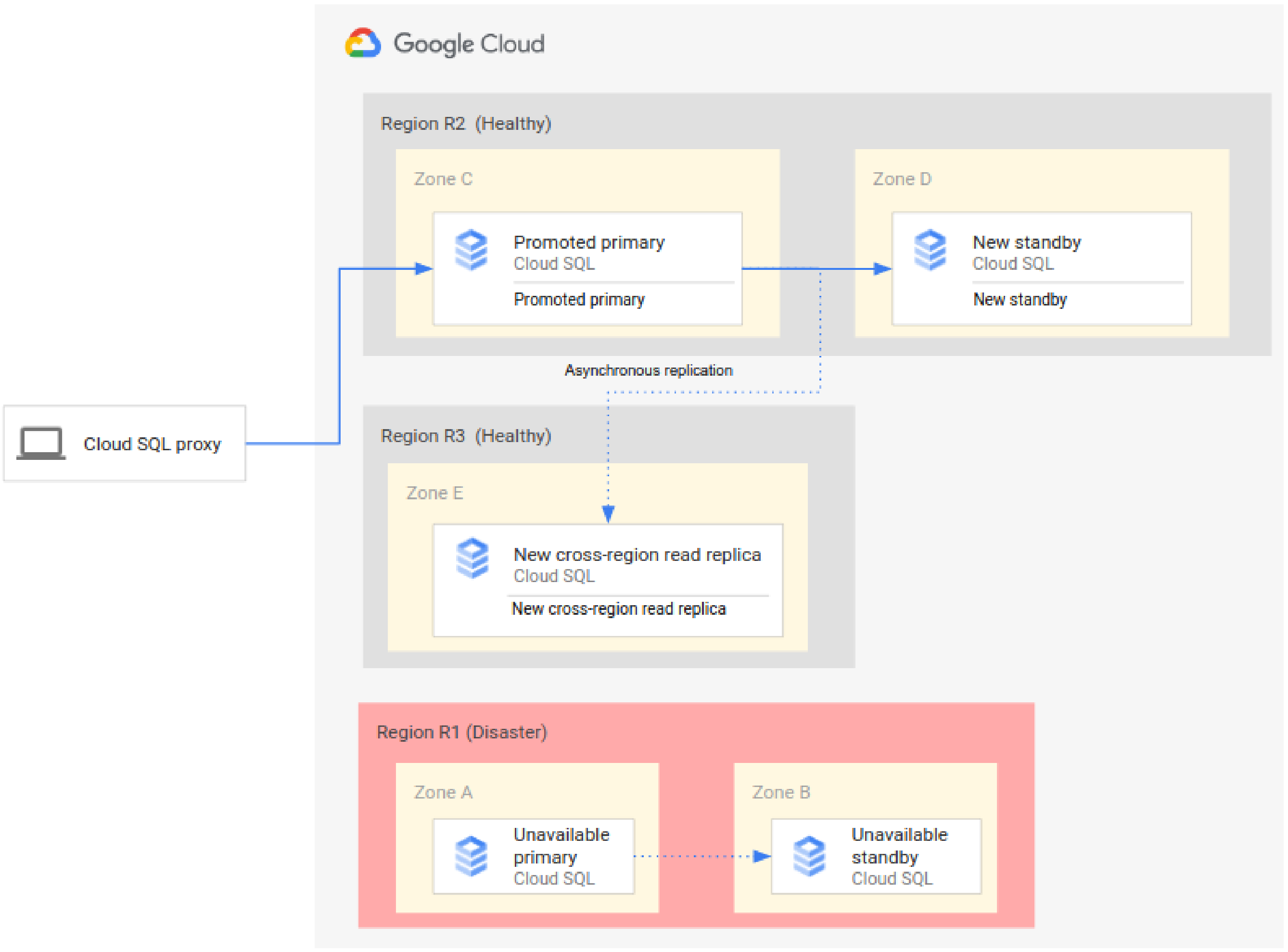
This basic process establishes a working primary database again. However, it doesn't establish a complete DR architecture, where the new primary instance itself has a standby instance and a cross-region read replica.

## Overview of the complete DR process

A complete DR process ensures that the single instance, the new primary, is enabled for HA and has a cross-region read replica. A complete DR process also provides a fallback to the original deployment in the original primary region.

### Failing over to a secondary region

A complete DR process extends the basic DR process by adding steps to establish a complete DR architecture after failover. The following diagram shows a complete database DR architecture after the failover:



The complete database DR process consists of the following steps:

1. The primary region (R1), which is running the primary database, becomes unavailable.
2. The operations team recognizes and formally acknowledges the disaster and decides whether a failover is required.
3. If a failover is required, the cross-region read replica in the secondary region (R2) is made the new primary instance.
4. Client connections are reconfigured to access and process on the new primary instance (R2).
5. A new standby instance is created and started in R2 and added to the primary instance. The standby instance is in a different zone from the primary instance. The primary instance is now highly available because a standby instance was created for it.
6. In a third region (R3), a new cross-region read replica is created and attached to the primary instance. At this point, a complete disaster recovery architecture is recreated and

operational.

If the original primary region (R1) becomes available before step 6 is implemented, the cross-region read replica can be placed in region R1, rather than region R3, right away. In this case, the fallback to the original primary region (R1) is less complex and requires fewer steps.

## Avoiding a split-brain state

A failure of the primary region (R1) doesn't mean that the original primary instance and its standby instance are automatically shut down, removed, or otherwise made inaccessible when R1 becomes available again. If R1 becomes available, clients might read and write data (even by accident) on the original primary instance. In this case, a split-brain situation (https://wikipedia.org/wiki/Split-brain_(computing)) can develop, where some clients access stale data in the old primary database, and other clients access fresh data in the new primary database, leading to problems in your business.

To avoid a split-brain situation, you must ensure that clients can no longer access the original primary instance after R1 becomes available. Ideally, you should make the original primary inaccessible before clients start using the new primary instance, then delete the original primary right after you make it inaccessible.

## Establishing an initial backup after failover

When you promote the cross-region read replica to be the new primary in a failover, the transactions in the new primary might not be fully synchronized with transactions from the original primary. Therefore, those transactions are unavailable in the new instance.

As a best practice, we recommend that you immediately back up the new primary instance at the start of the failover and before clients access the database. This backup represents a consistent, known state at the point of the failover. Such backups can be important for regulatory purposes or for recovering to a known state if clients encounter issues when accessing the new primary.

## Falling back to an original primary region

As outlined earlier, this tutorial provides the steps to fall back to the original region (R1). There are two different versions of the fallback process.

- If you created the new cross-region read replica in a tertiary region (R3), you must create another (second) cross-region read replica in the primary region (R1).
- If you created the new cross-region read replica in the primary region (R1), you don't need to create another additional cross-region read replica in R1.

After the cross-region read replica in R1 exists, the Cloud SQL instance can fall back to R1. Because this fallback is manually triggered and not based on an outage, you can choose an appropriate day and time for this maintenance activity.

Thus, to achieve a complete DR that has a primary, standby, and cross-region read replica, you need two failovers. The first failover is triggered by the outage (a true failover), and the second failover re-establishes the starting deployment (a fallback or switchover).

## What's next

- Try the Cloud SQL for MySQL disaster recovery (https://cloud.google.com/solutions/cloud-sql-mysql-disaster-recovery-complete-failover-fallback?authuser=0) tutorial
- Try the Cloud SQL for PostgreSQL disaster recovery (https://cloud.google.com/architecture/cloud-sql-postgres-disaster-recovery-complete-failover-fallback?authuser=0) tutorial