Cloud SQL (https://cloud.google.com/sql/)
Documentation (https://cloud.google.com/sql/docs/)
MySQL (https://cloud.google.com/sql/docs/mysql/) Guides

Upgrading a First Generation instance to Second Generation

MySQL | PostgreSQL | SQL Server

This page describes how to upgrade your First Generation MySQL instance to Second Generation.

Note: Second Generation is replacing First Generation; support for First Generation instances ends January 30, 2020. To upgrade a First Generation instance to Second Generation, see Upgrading a First Generation Instance to Second Generation (https://cloud.google.com/sql/docs/mysql/upgrade-2nd-gen).

Introduction

Cloud SQL Second Generation offers significant advantages (https://cloud.google.com/sql/docs/mysql/1st-2nd-gen-differences) over First Generation. For eligible instances, Cloud SQL supports an in-place upgrade from a First Generation to a Second Generation instance. If your First Generation instance is eligible, it has an **Upgrade** button next to its name in the Console Instance listing page (https://console.cloud.google.com/sql), or on its **Instance details** page.

The upgrade process is designed to minimize the impact to your instance and applications. For example, the upgrade allows you to control when cutover occurs and preserves your instance's IPv4 address, so you do not need to make immediate changes to your application. However, because the supported features for Cloud SQL Second Generation instances are different than for First Generation, some changes to your instance or application might be required.

Before you begin

Before proceeding with upgrading an instance, you should familiarize yourself with the overall process, and ensure that you understand how your instance and applications will be affected

by the upgrade.

Preconfiguration requirements

The steps you must take before upgrading to Second Generation depend on your current instance configuration and how your clients connect.

If your First Generation instance...	You must...	More information
Is configured with MySQL 5.5	Review the list of differences between MySQL 5.5 and 5.6, and take any steps necessary. After the upgrade, your instance will have MySQL 5.6.	See Changes Affecting Upgrades to MyS (https://dev.mysql.com/doc/refman/5. series.html) .
Has tables that use the MEMORY storage engine	Remove all tables that use the MEMORY storage engine before starting the upgrade process. If the data in these tables is required by your application, you must move it to a new table that uses the InnoDB storage engine.	You can find these tables by using the fo `SELECT table_schema, table_name FROM information_schema.tabl WHERE engine = 'MEMORY' AND table_schema NOT IN ('mysql','information_schema` Cloud SQL highlights these tables during check.
Has tables in the performance_schema database that do not use the PERFORMANCE_SCHEMA storage engine	Before starting the upgrade process, remove all tables from the performance_schema database that do not use the PERFORMANCE_SCHEMA storage engine. If you do not remove those tables, the upgrade process renames the performance_schema database to performance_schema_cloudsqlmigbackup, and updates the references in the **`mysql.proc`** and **`mysql.event`** tables.	You can find these tables with the follow `SELECT table_name FROM information_schema.tabl WHERE table_schema='performa AND engine != 'performance_s` Cloud SQL highlights these tables during check.
Has data in the **`slow_query_log`** or **`general_log`** tables.	Truncate these tables.	The data in these tables will not be brou Generation instance. In addition, if these data, they could cause longer downtime switchover. Cloud SQL recommends setting the **`log`** sends log output to the Google Cloud Co

(https://cloud.google.com/sql/docs/m

If your application...	Understand that...	M
Connects using IPv6, and your instance does *not* have an IPv4 address assigned	You should choose between incurring a few extra minutes of downtime for your applications and performing extra application updates.	If ac af ac If al yc in up up **M** m yc ne
Contains code that is not GTID safe (https://cloud.google.com/sql/docs/mysql/features#2nd-gen-unsupported)	You should update your applications to eliminate the unsupported statements.	Yc bu Fc
Uses Apps Script and connects using the legacy JDBC string (`jdbc:google:rdbms`)	You must update your applications to use the supported connection string.	Se se (h sc .
Uses Java on App Engine and connects using the legacy JDBC string (`jdbc:google:rdbms`)	You must update your applications to use the supported connection string.	If th (h sc . C Al dr dif th sp ex
Uses Python on App Engine and includes a legacy connection method	You must update your applications to use a newer connection method.	Se (h en

		. L ar • •
Uses the anonymous user in the MySQL user table	The anonymous user ('@'localhost') is a default user in First Generation instances. The anonymous user isn't safe and may cause a connection issue for the Cloud SQL Proxy (https://cloud.google.com/sql/docs/mysql/sql-proxy) with Second Generation instances. You should remove the anonymous user from the MySQL user table.	Du ('@ th th (h My '@ co de To Cr (h m : • • •

Preconfiguration recommendations

The following steps are not required, but taking them can help simplify or shorten the upgrade process.

If your First Generation instance...	Consider...	More information
Supports production applications	Creating a clone of the instance and completing the upgrade process on the clone, including testing your applications against the upgraded clone.	Testing the upgrade process on the clone enables you to determine the required changes (if any) for your instance and applications before you attempt the upgrade in your production environment.

		For more information about cloning instances, see Cloning Instances (https://cloud.google.com/sql/docs/mysql/clone-instance).
Does not have automatic backups or binary logging enabled	Enabling both of these settings.	You can also do this step during the upgrade process. This update requires an instance restart.

Second Generation configuration and management

Second Generation instances have different management requirements than First Generation instances. You should review the list of differences between First Generation and Second Generation (https://cloud.google.com/sql/docs/mysql/1st-2nd-gen-differences#differences) and ensure that you understand how those differences will affect your Cloud SQL administration practices.

Second Generation instances provide some additional configuration options, for example storage autogrow (https://cloud.google.com/sql/docs/mysql/instance-settings#automatic-storage-increase-2ndgen), configurable maintenance windows (https://cloud.google.com/sql/docs/mysql/instance-settings#maintenance-window-2ndgen), and on-demand backups (https://cloud.google.com/sql/docs/mysql/backup-recovery/backing-up#on-demand). See Settings for Second Generation instances (https://cloud.google.com/sql/docs/mysql/instance-settings#settings-2ndgen) for more information.

Storage requirements and pricing

Second Generation instances reserve more storage for system use. When you upgrade your instance to Second Generation, the amount of storage it uses increases by approximately .75 GB.

Storage pricing for Second Generation is based on your storage capacity, rather than the amount you are using. For more information, see Storage and networking pricing (https://cloud.google.com/sql/docs/mysql/pricing/#2nd-gen-storage-networking-prices).

Machine types

Second Generation machine types are different from First Generation instance tiers, and have different pricing. Your First Generation instance will be upgraded to a Second Generation instance using the mapping shown in the table below. The upgraded instance provides at least as many resources for Second Generation as are provided for First Generation. You can change the machine type of your Second Generation instance if needed after the upgrade.

The First Generation prices shown are the "Always On" Package price.

Note: Second Generation pricing varies based on storage, backup, and network usage. The prices shown for Second Generation are example calculations. You can use the Google Cloud Platform Pricing Calculator (https://cloud.google.com/products/calculator/) to get pricing estimates for your Cloud SQL Second Generation instances.

First Generation Tier	Second Generation Machine Type
D0 (0.125 CPU, 128 MB RAM) Max connections: 250 $10.80/month	**db-f1-micro** (1 shared CPU, 0.60 GB RAM) Max connections: 250 $10.35/month ★ Note: The Cloud SQL SLA (https://cloud.google.com/sql/sla) does not apply to this machine type.
D1 (0.25 CPU, 512 MB RAM) Max connections: 250 $43.80/month	**db-g1-small** (1 shared CPU, 1.70 GB RAM) Max connections: 1,000 $28.25/month ★ Note: The Cloud SQL SLA (https://cloud.google.com/sql/sla) does not apply to this machine type.
D2 (0.5 CPU, 1 GB RAM) Max connections: 250 $87.90/month	**db-n1-standard-1** (1 CPU, 3.75 GB RAM) Max connections: 4,000 $104.00/month
D4 (1 CPU, 2 GB RAM) Max connections: 500 $132.00/month	**db-n1-standard-1** (1 CPU, 3.75 GB RAM) Max connections: 4,000 $104.00/month
D8 (2 CPU, 4 GB RAM) Max connections: 1,000 $263.40/month	**db-n1-standard-2** (2 CPU, 7.50 GB RAM) Max connections: 4,000 $210.00/month

D16 (4 CPU, 8 GB RAM) Max connections: 2,000 $527.10/month	**db-n1-standard-4** (4 CPU, 15 GB RAM) Max connections: 4,000 $448.00/month
D32 (8 CPU, 16 GB RAM) Max connections: 4,000 $1,053.90/month	**db-n1-standard-8** (8 CPU, 30 GB RAM) Max connections: 4,000 $996.00/month

For more information about Second Generation pricing, including sample configuration pricing, see the pricing page (https://cloud.google.com/sql/docs/mysql/pricing#2nd-gen-pricing).

`ON_DEMAND` activation policy

Second Generation instances do not support the `ON_DEMAND` activation policy. You can stop your instance manually (https://cloud.google.com/sql/docs/mysql/start-stop-restart-instance), but it does not respond to any incoming connection requests until you start it again. Learn more (https://cloud.google.com/sql/docs/mysql/instance-settings#activation-policy-2ndgen).

MySQL system variables and options

Some MySQL system variables and options have different default values for Second Generation instances. If your instance has MySQL variables set to a value that is not supported for Second Generation, the value will be changed during the upgrade process. These variables are not configurable.

The following table lists the affected variables, and the value they will have after the upgrade:

System Variable	Second Generation Value
default_time_zone (https://dev.mysql.com/doc/refman/5.7/en/time-zone-support.html)	Format changed to offset from UTC.
expire_logs_days (https://dev.mysql.com/doc/refman/5.7/en/replication-options-binary-log.html#sysvar_expire_logs_days)	0
innodb_log_file_size (https://dev.mysql.com/doc/refman/5.7/en/innodb-parameters.html#sysvar_innodb_log_file_size)	512MB

System Variable	Second Generation Value
max_connections (https://dev.mysql.com/doc/refman/5.7/en/server-system-variables.html#sysvar_max_connections)	See machine type table (#machine-types).
sync_binlog (https://dev.mysql.com/doc/refman/5.7/en/replication-options-binary-log.html#sysvar_sync_binlog)	enabled

Downtime

Your application will experience periods of downtime during the upgrade, as described below:

1. If your First Generation instance does not have binary logging enabled, it will be restarted to enable binary logging, which is required to perform the upgrade.
2. Your First Generation instance will be restarted to install a SSL/TLS certificate to support secure replication between the instance and its Second Generation replica during the upgrade.
3. When you initiate the switchover process, and the Second Generation instance starts acting as the primary instance, there is a period of time when the First Generation instance has stopped responding to requests and the Second Generation instance is not yet online.

   The amount of time required for switchover is strongly affected by the amount of replication lag (https://cloud.google.com/sql/docs/mysql/high-availability#replication-lag) between the instance and its replica. For this reason, you should reduce or stop write operations before switching over.

Considerations for applications connecting from App Engine standard environment

Service account

For First Generation instances, you control access to your instance from an App Engine application by authorizing the app's project. This authorization is on a per-instance basis.

For Second Generation instances, access from an App Engine application is authorized by using a service account, which authorizes access to all Cloud SQL instances in a specific project.

When you upgrade an instance with an authorized App Engine project from First Generation to Second Generation, Cloud SQL creates a special service account that provides the same access as the authorized App Engine project did before the upgrade. Because this service account authorizes access only to a specific instance, rather than the entire project, this service account is not visible in the IAM service account page, and you cannot update or delete it.

Connection latency

In general, connection latency is decreased for Second Generation instances as compared to First Generation instances. However, for applications connecting from App Engine standard environment, latency is increased by approximately .3 ms.

Predefined MySQL user account changes

The host name for the predefined MySQL user account used to connect from App Engine changes from `root@localhost` to `root@cloudsqlproxy~%`.

If you have other MySQL users that use `localhost` as their host, they change in a similar fashion. The host name is automatically updated in the MySQL `grant` and `user` tables during the migration process, and generally, you won't need to update your App Engine app to account for this change.

However, if your app or other tools you use modify user tables and grants directly in MySQL, you should update them to use `cloudsqlproxy~%` as the host after the upgrade is complete. Do not modify user tables or grants during the upgrade.

This change does not affect the performance of connecting using this account.

`rdbms` connection library

If your applications are using the deprecated `rdbms` connection library, you must update them to use a supported connection method (https://cloud.google.com/sql/docs/mysql/connect-app-engine). The `rdbms` library will not work with an upgraded Second Generation instance.

Performing the upgrade

After you have completed the preparation steps (#preparation), you can proceed with the upgrade.

1. Go to the Cloud SQL Instances page in the Google Cloud Console.

   GO TO THE CLOUD SQL INSTANCES PAGE (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/SQL/INSTANCE
2. Find the instance you want to upgrade, and click the **Upgrade** button next to its name.
3. In the **Upgrade to Second Generation** panel, click **Check configuration**.
4. If your instance needs updates before the upgrade starts, click **Execute changes**, and follow any instructions given.
5. Click **Continue** to start the upgrade process.

   Your data is replicated to a replica, while the original instance remains active. Note: During the time that both your original instance and the replica exist, you are charged for both instances. To minimize extra charges, switch over to the new instance as soon as this step completes.
6. Optional but recommended: Create a clone (https://cloud.google.com/sql/docs/mysql/clone-instance) of your First Generation instance.

   Having a clone enables you to roll back to the current state if you have problems with the upgrade.
7. Optional but recommended: When the replica creation completes, shut down your applications.

   They will be unable to access the instance during the switchover. This is particularly important if your application holds transactions open.
8. When the replica creation completes, click **Switch over**.

   This causes the replica to be promoted, and take over serving data. Your instance stops accepting new connections and incurs 5-10 minutes of downtime. The switchover is displayed as a Failover operation in the Operations pane.
9. After the switchover completes, click **Finish** to complete the upgrade process and proceed to the post-upgrade steps (#confirm).

Confirm and complete the upgrade

The following checklist helps you confirm that everything is working correctly after the upgrade process completes, and ensures that your Second Generation instance is configured correctly.

If you need to rollback the upgrade, see Rolling back the upgrade (#rollback).

- Confirm that your clients and applications are connecting and operating correctly.

  Second Generation instances enable GTID. If your applications still need to be updated to handle this change correctly, you can temporarily disable binary logging, which also disables GTID checking. However, disabling binary logging also disables replication and point-in-time recovery, so you should update your applications to be GTID safe, and re-enable binary logging, as soon as possible.

- If needed, configure your instance for high availability (https://cloud.google.com/sql/docs/mysql/configure-ha).

  If your instance is serving production data, this step is strongly recommended for data durability and availability.

- Recreate any read replicas (https://cloud.google.com/sql/docs/mysql/replication/create-replica) you need.

  You cannot reuse the names for your replicas when you recreate them; you must choose different names.

- If your applications are connecting using IPv6 or an IPv4 address created before the upgrade, update them to use the new Public (primary) address that was assigned during the upgrade.

  An IPv4 address created before the upgrade is displayed as a **Migrated First Generation** IP address. Migrated IP addresses will be deprecated in the future.

- Consider updating your other connection paths.

  Although in most cases you can continue to use the same connection paths you used before the upgrade, there are some new options available to you:

  - Cloud SQL Proxy (https://cloud.google.com/sql/docs/mysql/sql-proxy) (secure connection without having to manage SSL/TLS certificates)
  - Socket libraries (https://cloud.google.com/sql/docs/mysql/connect-external-app#java) (proxy alternative for the Java programming language)

- If your applications are connecting using the First Generation instance connection name:

  `<project_id>:<instance_id>`

update them to use the Second Generation instance connection name:

`<project_id>:<region>:<instance_id>`.

- If your databases have any MyISAM tables (other than system tables), consider converting them to use the InnoDB storage engine.

  You can continue to use existing MyISAM tables. However, for increased data durability, Cloud SQL recommends that you convert them. For more information, see Converting Tables from MyISAM to InnoDB (https://dev.mysql.com/doc/refman/5.6/en/converting-tables-to-innodb.html).

Rolling back the upgrade

If you encounter issues with your upgraded Second Generation instance, you can create a new instance with the same state your First Generation instance had before you started the upgrade process.

Warning: Data updates applied after the upgrade began will be lost, and your new First Generation instance will have a different name and IP address than your original instance.

1. If you created a clone during the upgrade process, update your applications to point to it.
2. Otherwise, complete the following steps:
   a. Create a new First Generation instance (https://cloud.google.com/sql/docs/mysql/create-instance#create-1st-gen), with the same tier as your original instance.
   b. Open the **Backups** page for the upgraded instance, and find the backup taken at the time the upgrade process started.
   c. Restore the backup to the new First Generation instance you just created.

      For more information, see Restoring to a different instance (https://cloud.google.com/sql/docs/mysql/backup-recovery/restoring#restorebackups-another-instance)

      .

What's next

- Learn about differences between First Generation and Second Generation (https://cloud.google.com/sql/docs/mysql/1st-2nd-gen-differences#differences).
- Learn about Second Generation settings (https://cloud.google.com/sql/docs/mysql/instance-settings#settings-2ndgen).
- Learn about options for connecting to an instance (https://cloud.google.com/sql/docs/mysql/external-connection-methods).
- Learn more about setting database flags (https://cloud.google.com/sql/docs/mysql/flags).

Last updated January 17, 2020.