a Beta release of Compute Engine 16 KB physical block size persistent disk. This feature is not covered by any SLA o
:ation policy and might be subject to backward-incompatible changes.

This document describes how to use persistent disk with 16 KB physical block size (https://wikipedia.org/wiki/Block_(data_storage)) for performance improvements to a MySQL database.

Write-heavy MySQL workloads usually benefit from disabling the `InnoDB doublewrite` buffer (https://dev.mysql.com/doc/refman/8.0/en/innodb-doublewrite-buffer.html). MySQL's `InnoDB` performs doublewrite during the dirty page flushing process so it can recover possible torn pages.

However, if there is an end-to-end 16 KB atomic write path to ensure that a 16 KB data page won't be partially committed to the disk or torn write, then there is no need to perform doublewrite. When doublewrite is disabled, the database's dirty page flushing capability is essentially doubled, reducing the frequency in which the database falls into a sync flush state, which leads to a more stable, and possibly increased performance.

- If you want to use the command-line examples in this guide:
    1. Install or update to the latest version of the gcloud command-line tool (/compute/docs/gcloud-compute).
    2. Set a default region and zone (/compute/docs/gcloud-compute/#set_default_zone_and_region_in_your_local_client).
- If you want to use the API examples in this guide, set up API access (/compute/docs/api/prereqs).
- Read about persistent disks (/compute/docs/disks/#pdspecs).

You can build an end-to-end 16 KB atomic write path from the database to the block device leveraging a 16 KB persistent disk, so you can safely disable the doublewrite feature in MySQL/`InnoDB` and achieve a more stable and better performance for a high-write load.

Create and attach a persistent disk through the Google Cloud Console (https://console.cloud.google.com/), the `gcloud` tool (/compute/docs/gcloud-compute), or the API (/compute/docs/reference/rest/v1/disks/insert).

1. Create a 16 KB block size persistent disk (/compute/docs/disks/add-persistent-disk) and attach it to your VM. The 16 KB persistent disk provides 16 KB write atomicity at the physical block level.

   Although optional, it is recommended that you configure your MySQL instance to store data files only to the 16 KB persistent disk. Store log files, especially redo log and binlogs, to a 4 KB persistent disk that is attached to the same VM. This ensures log file writes continue to be high performance because small log writes on 16 KB persistent disk might trigger lots of read-modify-writes, which are slower.

2. Format the 16 KB disk (/compute/docs/disks/add-persistent-disk#formatting) using the ext4 file system with the BigAlloc option and set the cluster size to 16 KB. Here is an example `mkfs` command with BigAlloc option specified:

   Using BigAlloc with 16 KB as a cluster size, ensures that the file system allocates files that align with the 16 KB boundary on the disk.

3. When you create the VM instance, choose an OS image from the Google Container-Optimized OS (/container-optimized-os/docs/) image families in the `cos`-cloud project.

   Use the `gcloud` command to see a list of all available `cos` images:

   Select version 67 or higher. For better results, consider choosing an image from the `cos-stable` family.

   Choosing a `cos` image ensures that writes are not improperly split across 16 KB boundary by layers in between the file system and the physical block device layer. The `cos` image qualification process has built-in tests to ensure this result.

4. Ensure `max_segments` and `max_sectors_kb` are configured properly in the OS:

These two variables are already configured in all Compute Engine VMs, if you don't have a script changing these two variables after VM creation then you don't need to do anything here.

You can query these two constants in the OS under this path:

5. Configure `noop` or `none` as the I/O scheduler for the 16 KB persistent disk.

6. Disable I/O requests merging in kernel for the 16 KB persistent disk.

7. Configure `InnoDB` to use `O_DIRECT`. Set (or add) `O_DIRECT` to the `innodb_flush_method` database config.

Now you can safely turn off the `innodb_doublewrite` option (https://dev.mysql.com/doc/refman/8.0/en/innodb-parameters.html#sysvar_innodb_doublewrite).

This method is not the only approach you can take to ensure end-to-end 16 KB atomic writes using a 16 KB block device. For example, if you configure your database to use the block device directly as a raw device without using a filesystem, then you can skip the steps above that describe configuration of the file system.

- Learn more about adding persistent disks (/compute/docs/disks/add-persistent-disk).
- Read more about MySQL performance benchmarking tools (https://dev.mysql.com/downloads/benchmarks.html).