1.MySQL installation and removing

installing Mysql 8 community edition involve below steps

1.Add MySQL Repository package.
2. Install MySQL Community Server from the repository.
3. Enable MySQL server to auto-start.
4. Start MySQL server.
5 .Verify if MySQL is running.
Follow the steps in the provided link to add the repository and install MySQL.
click here

installing MySQL percona edition

Percona Server for MySQL indeed offers additional features and enhancements beyond what is available in the MySQL Community Edition. These additions are designed to improve performance, scalability, and manageability of MySQL databases. While Percona Server for MySQL aims to provide enterprise-level functionality, it remains an open-source solution, distinguishing itself from MySQL Enterprise Edition, which is proprietary software provided by Oracle.

Feature	Percona Server for MySQL	MySQL Community Edition
Performance Improvements	Often includes performance optimizations not found in MySQL Community Edition	May lack some of the performance optimizations present in Percona Server
Feature Differences	May include additional features or enhancements beyond those found in MySQL Community Edition	Standard features and functionalities of MySQL
Storage Engines	Supports multiple storage engines, may include enhancements specific to certain engines	Supports multiple storage engines, including InnoDB, MyISAM, etc.
Monitoring and Management Tools	May come bundled with additional monitoring and management tools for performance analysis, query profiling, etc.	Lacks additional monitoring and management tools provided by Percona
Support and Maintenance	Offers commercial support options including consulting, training, and technical support subscriptions	Relies primarily on community support, though commercial support is available through Oracle
Licensing	Distributed under the terms of the GNU General Public License (GPL), may offer additional commercial licensing options	Distributed under the terms of the GNU GPL, primarily open-source with commercial support available
Community and Ecosystem	Has its own community and ecosystem, including forums, blogs, and events focused on Percona products and technologies	Benefits from a large and active community of developers, contributors, and users

installing percona MySQL edition

the first step is to add percona repo by using the below command

steps below are based on this link from official percona site

also you can use the following link for more deep steps

```
$ sudo yum install https://repo.percona.com/yum/percona-release-latest.noarch.rpm
```

```
Complete!
[root@mysql-STG ~]# sudo yum install https://repo.percona.com/yum/percona-relea
se-latest.noarch.rpm
Updating Subscription Management repositories.
Last metadata expiration check: 0:00:13 ago on Thu 21 Mar 2024 11:00:32 AM +03.
percona-release-latest.noarch.rpm               19 kB/s |  20 kB     00:01
Dependencies resolved.
================================================================================
 Package                Architecture  Version        Repository           Size
================================================================================
Installing:
 percona-release        noarch        1.0-27         @commandline         20 k

Transaction Summary
================================================================================
Install  1 Package

Total size: 20 k
Installed size: 32 k
Is this ok [y/N]: y
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : percona-release-1.0-27.noarch                          1/1
```

We initiate the setup process by executing the following command:

```
sudo percona-release setup ps-80
```

```
  https://www.percona.com/doc/percona-repo-config/percona-release.html


  Verifying        : percona-release-1.0-27.noarch                      1/1
Installed products updated.

Installed:
  percona-release-1.0-27.noarch

Complete!
[root@mysql-STG ~]# sudo percona-release setup ps-80
* Disabling all Percona Repositories
On Red Hat 8 systems it is needed to disable the following DNF module(s): mysql  to install Percona-Server
Do you want to disable it? [y/N] y
Disabling dnf module...
Updating Subscription Management repositories.
Percona Release release/noarch YUM repository                  1.6 kB/s | 1.8 kB     00:01
Dependencies resolved.
================================================================================
 Package                Architecture          Version                Repository                Size
================================================================================
Disabling modules:
 mysql

Transaction Summary
================================================================================

Complete!
DNF mysql module was disabled
* Enabling the Percona Server 8.0 repository
* Enabling the Percona Tools repository
<*> All done!
[root@mysql-STG ~]#
```

final step is to starting the instalation

```
sudo yum install percona-server-server
```

```
[dba@mysqlPerconaSTG ~]$ sudo yum install percona-server-server
Percona Server 8.0 release/x86_64 YUM repository                 1.9 MB/s | 6.5 MB     00:03
Percona Tools release/x86_64 YUM repository                      1.1 MB/s | 5.1 MB     00:04
Last metadata expiration check: 0:00:03 ago on Sun 24 Mar 2024 11:02:28 PM +03.
Dependencies resolved.
================================================================================
 Package                 Architecture   Version                  Repository   Size
================================================================================
Installing:
Installing:
Installing:
Installing:
```

```
Installed:
  compat-openssl10-1:1.0.2o-4.el8_6.x86_64
  percona-icu-data-files-8.0.36-28.1.el8.x86_64
  percona-server-server-8.0.36-28.1.el8.x86_64
  percona-server-shared-compat-8.0.36-28.1.el8.x86_64
  perl-Data-Dumper-2.167-399.el8.x86_64
  perl-Digest-MD5-2.55-396.el8.x86_64
  perl-Errno-1.28-422.el8.x86_64
  perl-File-Path-2.15-2.el8.noarch
  perl-Getopt-Long-1:2.50-4.el8.noarch
  perl-IO-1.38-422.el8.x86_64
  perl-IO-Socket-SSL-2.066-4.module+el8.9.0+1517+e71a7a62.noarch
  perl-Mozilla-CA-20160104-7.module+el8.9.0+1521+0101edce.noarch
  perl-PathTools-3.74-1.el8.x86_64
  perl-Pod-Perldoc-3.28-396.el8.noarch
  perl-Pod-Usage-4:1.69-395.el8.noarch
  perl-Socket-4:2.027-3.el8.x86_64
  perl-Term-ANSIColor-4.06-396.el8.noarch
  perl-Text-ParseWords-3.30-395.el8.noarch
  perl-Time-Local-1:1.280-1.el8.noarch
  perl-Unicode-Normalize-1.25-396.el8.x86_64
  perl-interpreter-4:5.26.3-422.el8.x86_64
  perl-libs-4:5.26.3-422.el8.x86_64
  perl-parent-1:0.237-1.el8.noarch
  perl-threads-1:2.21-2.el8.x86_64
  make-1:4.2.1-11.el8.x86_64
  percona-server-client-8.0.36-28.1.el8.x86_64
  percona-server-shared-8.0.36-28.1.el8.x86_64
  perl-Carp-1.42-396.el8.noarch
  perl-Digest-1.17-395.el8.noarch
  perl-Encode-4:2.97-3.el8.x86_64
  perl-Exporter-5.72-396.el8.noarch
  perl-File-Temp-0.230.600-1.el8.noarch
  perl-HTTP-Tiny-0.074-2.el8.noarch
  perl-IO-Socket-IP-0.39-5.el8.noarch
  perl-MIME-Base64-3.15-396.el8.x86_64
  perl-Net-SSLeay-1.88-2.module+el8.9.0+1517+e71a7a62.x86_64
  perl-Pod-Escapes-1:1.07-395.el8.noarch
  perl-Pod-Simple-1:3.35-395.el8.noarch
  perl-Scalar-List-Utils-3:1.49-2.el8.x86_64
  perl-Storable-1:3.11-3.el8.x86_64
  perl-Term-Cap-1.17-395.el8.noarch
  perl-Text-Tabs+Wrap-2013.0523-395.el8.noarch
  perl-URI-1.73-3.el8.noarch
  perl-constant-1.33-396.el8.noarch
  perl-libnet-3.11-3.el8.noarch
  perl-macros-4:5.26.3-422.el8.x86_64
  perl-podlators-4.11-1.el8.noarch
  perl-threads-shared-1.58-2.el8.x86_64

Complete!
[dba@mysqlPerconaSTG ~]$
```

note: there some dependency that are required such as perl

need to check you repo if it have this dependcay

post installation check

after the installation there are some additional steps which include enable daemon for mysql and starting up daemon for MySQL

1.start and enable Mysql daemon

use the below to enable mysql daemon to start auto during OS boot

```
sudo systemctl enable mysqld.service
```

```
[dba@mysqlPerconaSTG ~]$ sudo systemctl enable mysqld.service
[sudo] password for dba:
[dba@mysqlPerconaSTG ~]$
```

next start mysql daemon

```
sudo systemctl start mysqld.service
```

now final step is to check the status of mysql services make sure that they are running

```
systemctl status mysqld
```

```
[dba@mysqlPerconaSTG ~]$ systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Sun 2024-03-24 23:09:53 +03; 41s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 41853 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 41933 (mysqld)
   Status: "Server is operational"
    Tasks: 39 (limit: 24155)
   Memory: 469.6M
   CGroup: /system.slice/mysqld.service
           └─41933 /usr/sbin/mysqld

Mar 24 23:09:40 mysqlPerconaSTG systemd[1]: Starting MySQL Server...
Mar 24 23:09:53 mysqlPerconaSTG systemd[1]: Started MySQL Server.
[dba@mysqlPerconaSTG ~]$
```

2.VERIFICATION

last is to check and confirm that MySQL is processes are running

We'll first utilize the `pidof` utility to check if the MySQL server process is running:

`pidof mysqld`

```
[dba@mysqlPerconaSTG ~]$ pidof mysqld
41933
[dba@mysqlPerconaSTG ~]$
```

Next, we'll utilize the `netstat` command to confirm that MySQL is listening for connections on port 3306

`netstat -ntlp | grep 3306`

```
[dba@mysqlPerconaSTG ~]$ netstat -ntlp | grep 3306
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
tcp6       0      0 :::33060                :::*                    LISTEN      -
tcp6       0      0 :::3306                 :::*                    LISTEN      -
[dba@mysqlPerconaSTG ~]$
```

Finally, we'll employ the `lsof` command to inspect the files currently open by the MySQL server process:

`sudo lsof -u mysql`

```
[dba@mysqlPerconaSTG ~]$ sudo lsof -u mysql
[sudo] password for dba:
COMMAND   PID  USER   FD      TYPE             DEVICE SIZE/OFF      NODE NAME
mysqld  41933 mysql  cwd       DIR              253,0     4096 101183285 /var/lib/mysql
mysqld  41933 mysql  rtd       DIR              253,0      224       128 /
mysqld  41933 mysql  txt       REG              253,0 66373040     53299 /usr/sbin/mysqld
mysqld  41933 mysql  mem       REG              253,0    59464  67164345 /usr/lib64/mysql/plugin/component_validate_password.so
mysqld  41933 mysql  DEL       REG               0,17           102581 /[aio]
mysqld  41933 mysql  DEL       REG               0,17           102580 /[aio]
mysqld  41933 mysql  DEL       REG               0,17           102579 /[aio]
mysqld  41933 mysql  DEL       REG               0,17           102578 /[aio]
mysqld  41933 mysql  DEL       REG               0,17           102577 /[aio]
mysqld  41933 mysql  DEL       REG               0,17           102576 /[aio]
mysqld  41933 mysql  DEL       REG               0,17           102575 /[aio]
mysqld  41933 mysql  DEL       REG               0,17           102574 /[aio]
mysqld  41933 mysql  mem       REG              253,0    54352      1721 /usr/lib64/libnss_files-2.28.so
mysqld  41933 mysql  mem       REG              253,0    46408      4058 /usr/lib64/libnss_sss.so.2
mysqld  41933 mysql  mem       REG              253,0    99656      2029 /usr/lib64/libz.so.1.2.11
mysqld  41933 mysql  mem       REG              253,0  2089936      1709 /usr/lib64/libc-2.28.so
mysqld  41933 mysql  mem       REG              253,0    99664       144 /usr/lib64/libgcc_s-8-20210514.so.1
mysqld  41933 mysql  mem       REG              253,0  1598848      1713 /usr/lib64/libm-2.28.so
mysqld  41933 mysql  mem       REG              253,0  1661448      2085 /usr/lib64/libstdc++.so.6.0.25
mysqld  41933 mysql  mem       REG              253,0    19128      1711 /usr/lib64/libdl-2.28.so
```

configure root user

after the installation you to setup the root user so you will able to connect to mysql
the steps invovle Restart the server with the --skip-grant-tables option to allow access without a password. This option is insecure. This option also disables remote connections.

```
$ sudo systemctl stop mysqld
$ sudo systemctl set-environment MYSQLD_OPTS="--skip-grant-tables"
$ sudo systemctl start mysqld
$ mysql
```

```
[dba@mysqlPerconaSTG ~]$ sudo systemctl stop mysqld
[sudo] password for dba:
[dba@mysqlPerconaSTG ~]$ sudo systemctl set-environment MYSQLD_OPTS="--skip-grant-tables"
[dba@mysqlPerconaSTG ~]$ sudo systemctl start mysqld
[dba@mysqlPerconaSTG ~]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
```

next we need to setup the password for root user to be able to connect normaly from local host

also we need to reload grant tables tso that we can run alter command

```
mysql> FLUSH PRIVILEGES;
mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'Awersdfzxc.1';
```

```
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.02 sec)

mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'Awersdfzxc.1';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

now stop and run command to start MySQL in normal mode without option --skip-grant-tables

```
$ sudo systemctl stop mysqld
$ sudo systemctl unset-environment MYSQLD_OPTS
$ sudo systemctl start mysqld
```

```
[dba@mysqlPerconaSTG ~]$ sudo systemctl stop mysqld
[sudo] password for dba:
[dba@mysqlPerconaSTG ~]$ sudo systemctl unset-environment MYSQLD_OPTS
[dba@mysqlPerconaSTG ~]$ sudo systemctl start mysqld
[dba@mysqlPerconaSTG ~]$
```

now try to login to MySQL engin using the password you setup for root and it should connect normaly

`mysql -u root -p`

```
[dba@mysqlPerconaSTG ~]$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

removing MySQL server

the steps are done as follow

1. remove MySQL related repo packages
2. remove all MySQL packages
3. remove MySQL user from OS
4. remove any left-over MySQL files

5. verfiyy MySQL is not persent at all on OS

STEP 1. SEARCH FOR REPOSITORY PACKAGES

using `rpm` we will search for repo for MySQL , and using `grep` to filter for our tagert repo using unqaie pattern

`rpm -qa | grep -i mysql`

```
[root@Mysqlcom-RHEL-STG ~]# rpm -qa | grep -i mysql
mysql80-community-release-el7-3.noarch
[root@Mysqlcom-RHEL-STG ~]#
```

STEP 2. REMOVE REPO

we have located the full name of MySQL repo now we will use `yum` to remove the repo from repo list

`sudo yum -y remove mysql80-community-release-el7-3.noarch`

```
[root@Mysqlcom-RHEL-STG ~]# rpm -qa | grep -i mysql
mysql80-community-release-el7-3.noarch
[root@Mysqlcom-RHEL-STG ~]# sudo yum -y remove mysql80-community-release-el7-3.noarch
Dependencies resolved.
================================================================================
 Package                    Architecture        Version             Repository        Size
================================================================================
Removing:
 mysql80-community-release  noarch              el7-3               @@commandline     31 k

Transaction Summary
================================================================================
Remove  1 Package

Freed space: 31 k
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Erasing          : mysql80-community-release-el7-3.noarch                 1/1
  Verifying        : mysql80-community-release-el7-3.noarch                 1/1

Removed:
  mysql80-community-release-el7-3.noarch

Complete!
[root@Mysqlcom-RHEL-STG ~]#
```

3. SEARCH FOR INSTALLED MYSQL PACKAGES

next steps is to locate all MySQL packages installed by using the below , this important as MySQL has a lot of packages installed other that MySQL community server .

using the below grep command will help locate all these packages.

`rpm -qa | grep -i mysql`

```
[root@Mysqlcom-RHEL-STG dba]# rpm -qa | grep -i mysql
mysql-community-server-8.0.36-1.el8.x86_64
mysql-community-client-8.0.36-1.el8.x86_64
mysql-community-client-plugins-8.0.36-1.el8.x86_64
mysql-community-libs-8.0.36-1.el8.x86_64
mysql-community-icu-data-files-8.0.36-1.el8.x86_64
mysql80-community-release-el8-9.noarch
mysql-community-common-8.0.36-1.el8.x86_64
[root@Mysqlcom-RHEL-STG dba]#
```

STEP 4. REMOVE MYSQL PACKAGES

now we will start by removing the packages using `yum remove` and then specifying the list of mysql we have found

```
 yum remove mysql-community-server-8.0.36-1.el8.x86_64 mysql-community-client-8.0.36-1.el8.x86_64 mysql-community-client-plugins-8.0.36-1.el8.x86_64 mysql-community-libs-8.0.36-1.el8.x86_64 mysql-community-icu-data-files-8.0.36-1.el8.x86_64 mysql80-community-release-el8-9.noarch mysql-community-common-8.0.36-1.el8.x86_64
```

```
   CGroup: /system.slice/mysqld.service
           └─47289 /usr/sbin/mysqld

Apr 01 20:59:29 Mysqlcom-RHEL-STG systemd[1]: Starting MySQL Server...
Apr 01 20:59:43 Mysqlcom-RHEL-STG systemd[1]: Started MySQL Server.
[root@Mysqlcom-RHEL-STG dba]# rpm -qa | grep -i mysql
mysql-community-server-8.0.36-1.el8.x86_64
mysql-community-client-8.0.36-1.el8.x86_64
mysql-community-client-plugins-8.0.36-1.el8.x86_64
mysql-community-libs-8.0.36-1.el8.x86_64
mysql-community-icu-data-files-8.0.36-1.el8.x86_64
mysql80-community-release-el8-9.noarch
mysql-community-common-8.0.36-1.el8.x86_64
[root@Mysqlcom-RHEL-STG dba]# yum remove mysql-community-server-8.0.36-1.el8.x86_64 mysql-community-client-8.0.36-1.el8.x86_64 mysql-community-client-plugins-8.0.36-1.el8.x86_64 mysql-community-libs-8.0.36-1.el8.x86_64 mysql-community-icu-data-files-8.0.36-1.el8.x86_64 mysql80-community-release-el8-9.noarch mysql-community-common-8.0.36-1.el8.x86_64
Dependencies resolved.
================================================================================
 Package                    Architecture Version                 Repository      Size
================================================================================
Removing:
 mysql-community-client     x86_64       8.0.36-1.el8            @mysql80-community  79 M
 mysql-community-client-plugins
                            x86_64       8.0.36-1.el8            @mysql80-community  20 M
 mysql-community-common     x86_64       8.0.36-1.el8            @mysql80-community  10 M
 mysql-community-icu-data-files
                            x86_64       8.0.36-1.el8            @mysql80-community 3.9 M
 mysql-community-libs       x86_64       8.0.36-1.el8            @mysql80-community 7.4 M
 mysql-community-server     x86_64       8.0.36-1.el8            @mysql80-community 295 M
 mysql80-community-release  noarch       el8-9                   @@commandline      15 k
Removing unused dependencies:
```

```
Removed:
  mysql-community-client-8.0.36-1.el8.x86_64                  mysql-community-client-plugins-8.0.36-1.el8.x86_64
  mysql-community-common-8.0.36-1.el8.x86_64                  mysql-community-icu-data-files-8.0.36-1.el8.x86_64
  mysql-community-libs-8.0.36-1.el8.x86_64                    mysql-community-server-8.0.36-1.el8.x86_64
  mysql80-community-release-el8-9.noarch                      perl-Carp-1.42-396.el8.noarch
  perl-Data-Dumper-2.167-399.el8.x86_64                       perl-Digest-1.17-395.el8.noarch
  perl-Digest-MD5-2.55-396.el8.x86_64                         perl-Encode-4:2.97-3.el8.x86_64
  perl-Errno-1.28-422.el8.x86_64                              perl-Exporter-5.72-396.el8.noarch
  perl-File-Path-2.15-2.el8.noarch                            perl-File-Temp-0.230.600-1.el8.noarch
  perl-Getopt-Long-1:2.50-4.el8.noarch                        perl-HTTP-Tiny-0.074-2.el8.noarch
  perl-IO-1.38-422.el8.x86_64                                 perl-IO-Socket-IP-0.39-5.el8.noarch
  perl-IO-Socket-SSL-2.066-4.module+el8.9.0+1517+e71a7a62.noarch perl-MIME-Base64-3.15-396.el8.x86_64
  perl-Mozilla-CA-20160104-7.module+el8.9.0+1521+0101edce.noarch perl-Net-SSLeay-1.88-2.module+el8.9.0+1517+e71a7a62.x86_64
  perl-PathTools-3.74-1.el8.x86_64                            perl-Pod-Escapes-1:1.07-395.el8.noarch
  perl-Pod-Perldoc-3.28-396.el8.noarch                        perl-Pod-Simple-1:3.35-395.el8.noarch
  perl-Pod-Usage-4:1.69-395.el8.noarch                        perl-Scalar-List-Utils-3:1.49-2.el8.x86_64
  perl-Socket-4:2.027-3.el8.x86_64                            perl-Storable-1:3.11-3.el8.x86_64
  perl-Term-ANSIColor-4.06-396.el8.noarch                     perl-Term-Cap-1.17-395.el8.noarch
  perl-Text-ParseWords-3.30-395.el8.noarch                    perl-Text-Tabs+Wrap-2013.0523-395.el8.noarch
  perl-Time-Local-1:1.280-1.el8.noarch                        perl-URI-1.73-3.el8.noarch
  perl-Unicode-Normalize-1.25-396.el8.x86_64                  perl-constant-1.33-396.el8.noarch
  perl-interpreter-4:5.26.3-422.el8.x86_64                    perl-libnet-3.11-3.el8.noarch
  perl-libs-4:5.26.3-422.el8.x86_64                           perl-macros-4:5.26.3-422.el8.x86_64
  perl-parent-1:0.237-1.el8.noarch                            perl-podlators-4.11-1.el8.noarch
  perl-threads-1:2.21-2.el8.x86_64                            perl-threads-shared-1.58-2.el8.x86_64

Complete!
[root@Mysqlcom-RHEL-STG dba]#
```

STEP 5. REMOVE MYSQL USER

To remove the MySQL user that was added during the MySQL installation process, you can utilize the `userdel` command.

First, to verify if the MySQL user still exists within the operating system, the `grep` command can be used to search for the MySQL user in the `passwd` file. Once confirmed, you can proceed to delete the user. The `userdel` command facilitates this process. You have the option to run `userdel` followed by the username to initiate an interactive user deletion wizard.

Alternatively, for a more direct approach, you can use `userdel -r [username]` to not only delete the user but also remove their home directory and any assigned mailbox, if applicable. The `-r` option ensures that both the user's home directory and their mailbox (if they have one) are deleted alongside the user account.

```
grep mysql /etc/passwd
sudo userdel
sudo userdel -r mysql
```

```
[root@Mysqlcom-RHEL-STG dba]# grep mysql /etc/passwd
mysql:x:27:27:MySQL Server:/var/lib/mysql:/bin/false
[root@Mysqlcom-RHEL-STG dba]#
```

```
[root@Mysqlcom-RHEL-STG dba]# sudo userdel -r mysql
userdel: mysql mail spool (/var/spool/mail/mysql) not found
[root@Mysqlcom-RHEL-STG dba]#
```

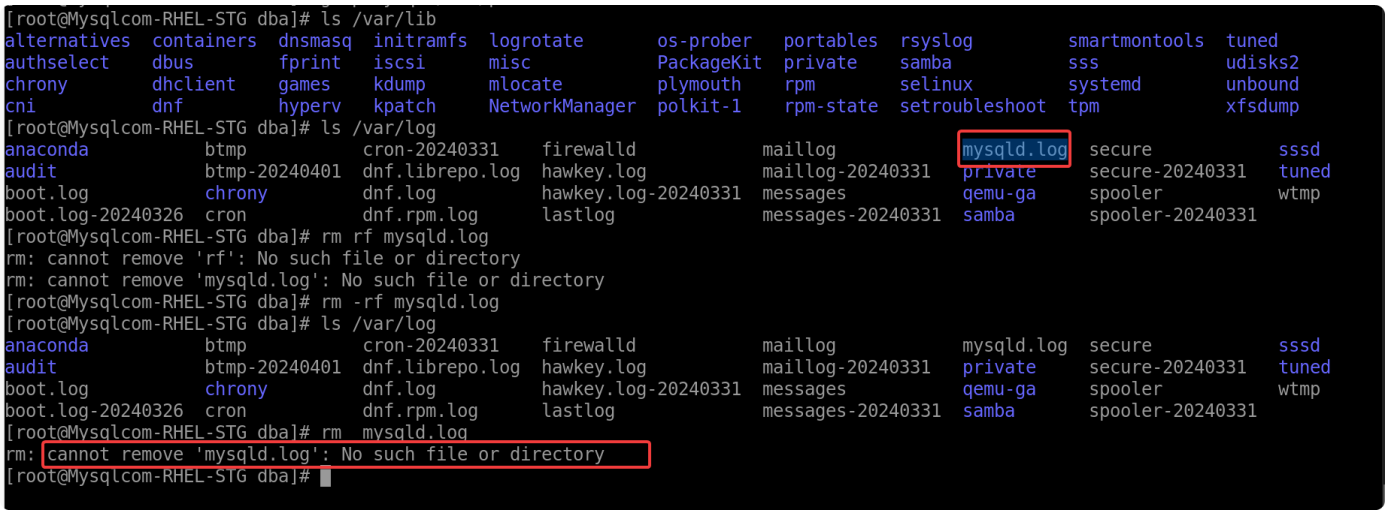
you can ignore the error regarding mail spool since MySQL don't have mailbox

```
[root@Mysqlcom-RHEL-STG dba]# grep mysql /etc/passwd
[root@Mysqlcom-RHEL-STG dba]#
```

STEP 6. REMOVE OTHER LEFTOVER FILES

now we have remove MySQL packages and deleted the user , next steps is to remove all left over directory created by MySQL installation

normally there will be data directory located in `/var/lib`
and log file for MySQL located in `/var/log`
search for MySQL file using `ls` and if there is still some MySQL leftovers proceed with deleting these files
`ls /var/lib ls /var/log`

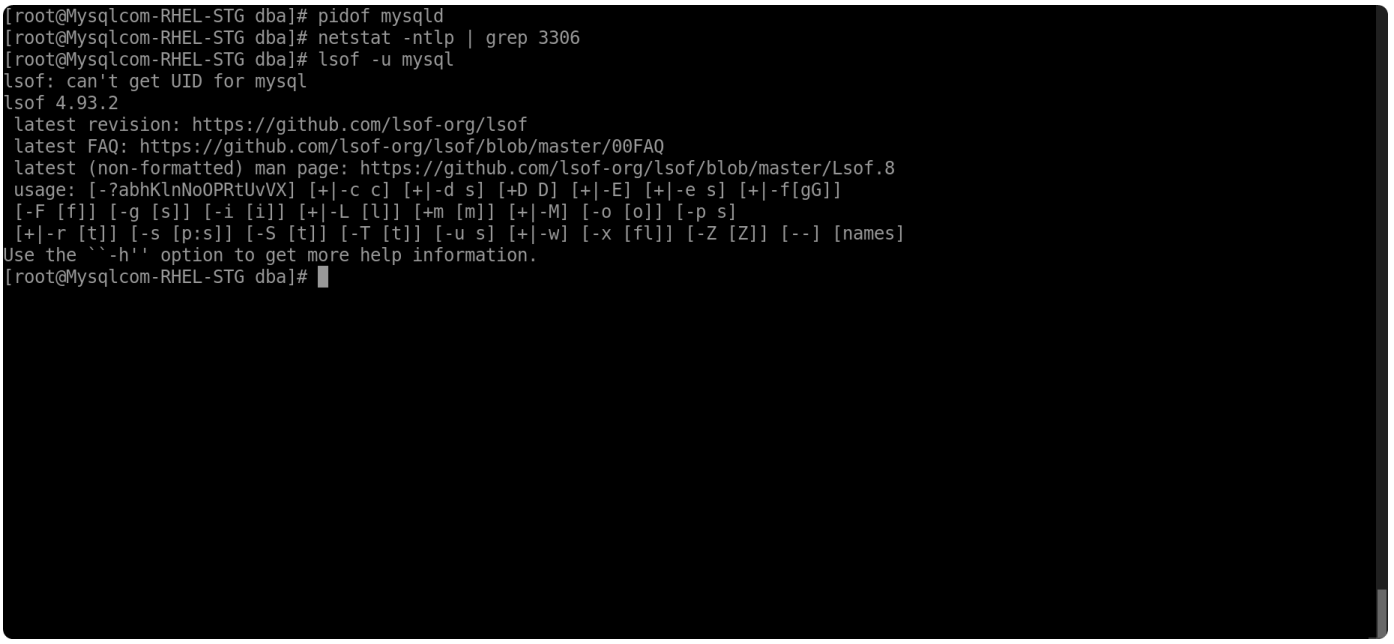


VERIFICATION

last steps is to confirm that we have successfully remove MySQL from the OS

while run the below command , nothing should popup for the below command

`pidof mysqld netstat -ntlp | grep 3306 sudo lsof -u mysql`



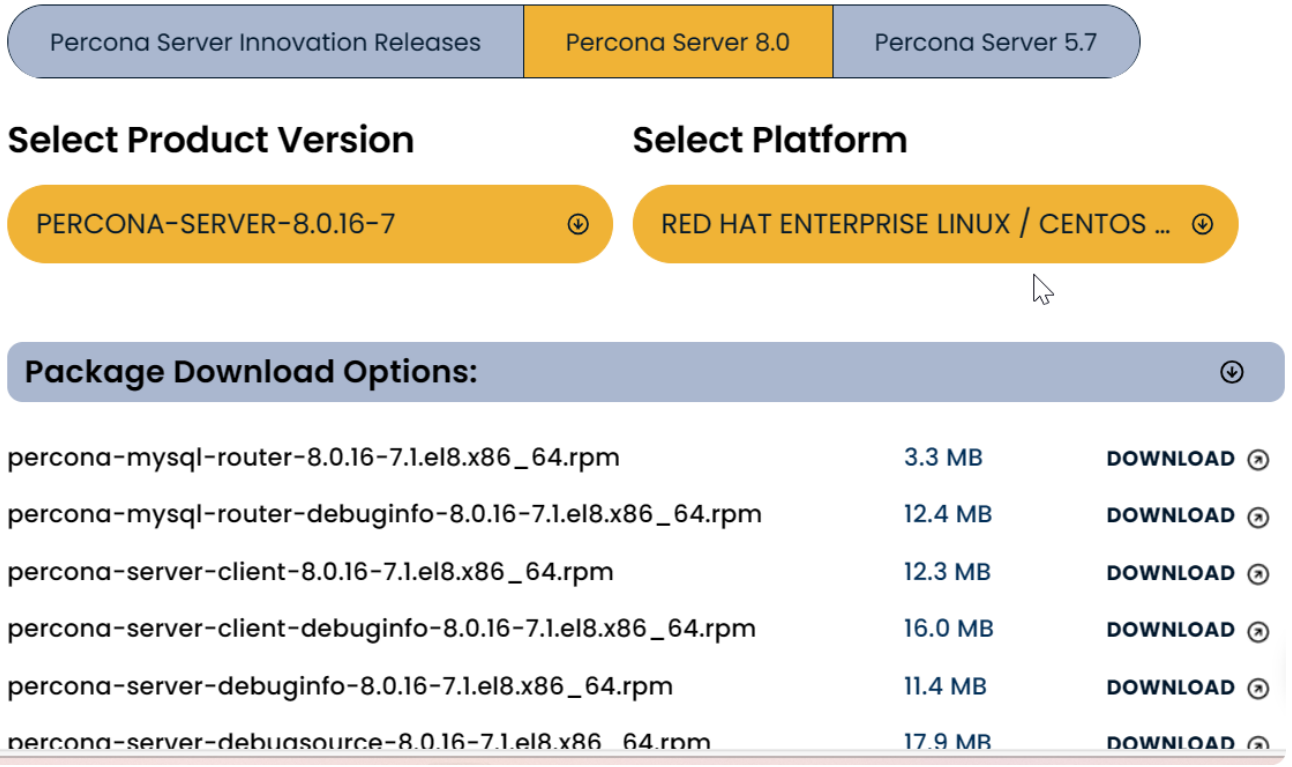
Installing Specific Version of MySQL

link for reference steps

When setting up a MySQL environment, particularly for testing purposes, it's important for database administrators (DBAs) to often install specific versions of MySQL to ensure compatibility with applications. In your scenario, you're looking to install MySQL version 8.0.26-16.1. Here's a streamlined guide on how to do this, with an emphasis on downloading the correct version from Percona, as Percona offers optimized versions of MySQL

1. Visit Percona's Downloads Section: Start by navigating to Percona's official downloads page. You can reach it by following this link: Percona Downloads.
2. Select the MySQL Version: Once on the downloads page, you'll need to find the MySQL server version you're interested in, which is MySQL 8.0.26-16.1 in this case. Keep in mind that navigating the Percona website might require you to select not just the version but also the specific distribution

and OS you are using, such as Ubuntu, Debian, CentOS, etc.



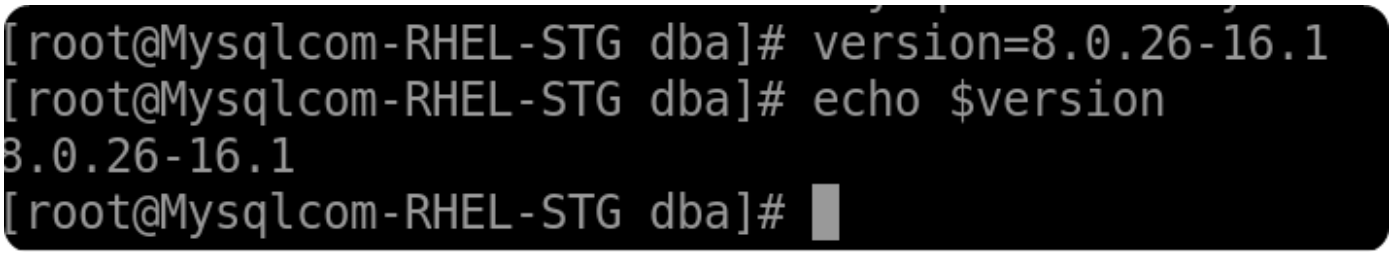
3. Download the Repository Package: Percona packages its releases in repository packages. You'll need to download the repository package suitable for your operating system. This might involve selecting your OS version and then downloading a .deb package for Debian-based systems like Ubuntu, or a .rpm package for Red Hat-based systems like CentOS.
   you can use `wget` to download the repo directly on OS
   `wget https://downloads.percona.com/downloads/Percona-Server-8.0/Percona-Server-8.0.16-7/binary/redhat/8/x 86_64/Percona-Server-8.0.16-7-r613e312-el8-x86_64-bundle.tar`

```
2024-04-04 23:16:11 (357 KB/s) - ‘x’ saved [128273]

[dba@mysqlPerconaSTG ~]$ wget https://downloads.percona.com/downloads/Percona-Server-8.0/Percona-Server-8.0.16-7/binary/redhat/8/x
86_64/Percona-Server-8.0.16-7-r613e312-el8-x86_64-bundle.tar
--2024-04-04 23:16:36--  https://downloads.percona.com/downloads/Percona-Server-8.0/Percona-Server-8.0.16-7/binary/redhat/8/x86_64
/Percona-Server-8.0.16-7-r613e312-el8-x86_64-bundle.tar
Resolving downloads.percona.com (downloads.percona.com)... 147.135.54.159, 2604:2dc0:200:69f::2
Connecting to downloads.percona.com (downloads.percona.com)|147.135.54.159|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 717352960 (684M) [application/octet-stream]
Saving to: ‘Percona-Server-8.0.16-7-r613e312-el8-x86_64-bundle.tar’

-Server-8.0.16-7-r613e312-el8-x8   0%[                                                    ]   3.29M   581KB/s    eta 19m 32s
```

one done extract the package and using tar
`tar -xvf yourfile.tar`

4. Install the Repository Package: Once the repository package is downloaded, you will need to install it. This step varies depending on your OS. For Debian-based systems, you'd use dpkg to install a .deb file, and for Red Hat-based systems, you'd use rpm to install a .rpm file.
   to speed up the process setup a variable with the version you required

`version=8.0.26-16.1 echo $version`




```
sudo yum install percona-mysql-router-$version.x86_64
sudo yum install mysql-community-client-$version.x86_64
```

Performing MySQL Secure Installation

Securing a fresh MySQL installation is crucial, and this can be accomplished by running the `mysql_secure_installation` script. This utility is designed to enhance the security of MySQL in several key ways:

- It is executed exclusively by the root user to ensure administrative privileges.
- The script significantly boosts MySQL's security by prompting you to set a password for the root account, thereby safeguarding it.
- It prevents the root user from logging in remotely, adding an extra layer of protection.
- The script also removes anonymous users, eliminating potential unauthorized access points.
- It deletes the default test database, which is created during the MySQL installation process, to prevent unintended access.
- Lastly, the script immediately reloads privilege settings, applying these security enhancements without delay to the MySQL environment.
  reference details steps

access the root password

To initiate the `mysql_secure_installation` process, it's necessary to obtain the temporary root password generated during the MySQL installation. This password can be found in the `/var/log/mysqld.log` file. By employing the `grep` command, you can efficiently filter the contents of this file to exclusively display the root password, using the following command:

`sudo grep 'password' /var/log/mysqld.log`

This command searches the specified log file for any lines containing the word "password," effectively isolating and revealing the temporary root password required for proceeding with the secure installation script.

RUN MYSQL_SECURE_INSTALLATION SCRIPT

Next, proceed by executing the `mysql_secure_installation` command and respond to the prompts as previously outlined. The script will request the root password, which you've retrieved from the mysqld.log file.

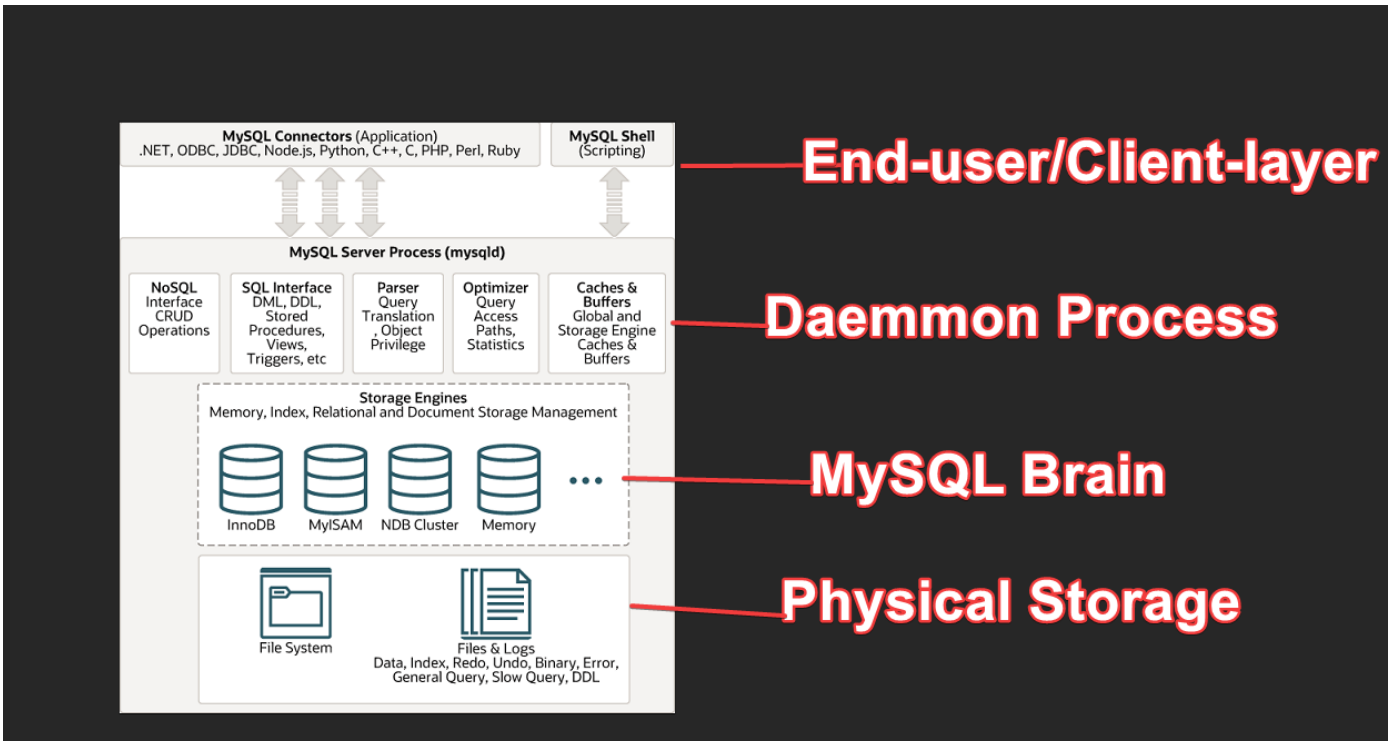
```
[root@Mysqlcom-RHEL-STG dba]# mysql_secure_installation

Securing the MySQL server deployment.

Enter password for user root:

The existing password for the user account root has expired. Please set a new password.

New password:

Re-enter new password:
The 'validate_password' component is installed on the server.
The subsequent steps will run with the existing configuration
of the component.
Using existing password for root.

Estimated strength of the password: 100
Change the password for root ? ((Press y|Y for Yes, any other key for No) : n

 ... skipping.
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : y
Success.

Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : y
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.


Remove test database and access to it? (Press y|Y for Yes, any other key for No) : y
 - Dropping test database...
Success.

 - Removing privileges on test database...
Success.

Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : y
Success.

All done!
[root@Mysqlcom-RHEL-STG dba]#
```

2-Exploring MySQL Server

- MySQL Architecture
- MySQL installed File Locations
  - DATA DIRECTORY
  - LOGS FILES
  - GLOVBAL CONFIGURATION
- MYSQL Executable programmes
- MySQL services under systemd
- MySQL Shell Commands
  - help command
  - quit
  - status
  - system
- MySQL Socket File
  - what happen if we delete both file
- MYSQL GLOBAL Variables
  - how to see the values of the global variables
  - retrieve and set these global values
  - set the value for global variable
  - session variables what are they and how retrieve and edit session variables
  - how retrieve and edit session variables
  - edit session variables
  - GETTING SYSTEM VARIABLE HELP
- MySQL show commands
- MySQL System Databases
- MySQL Local vs Remote Connections
  - localhost-connections
  - specific-host-connection:
  - any-host-connection
    - retrieve the created user and there connection setting
- MySQL Shell

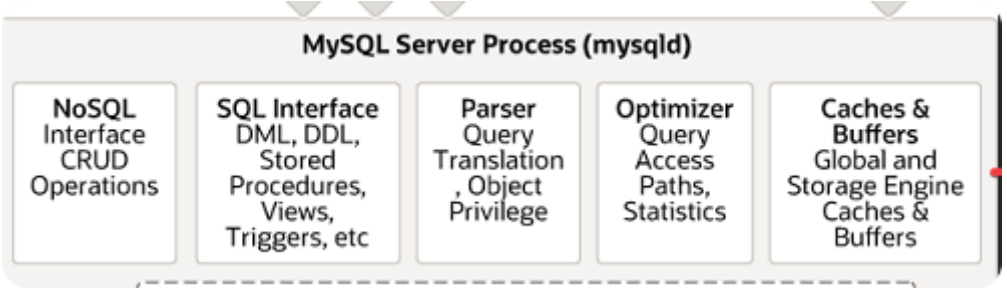
- [syntax]
    - [example]

MySQL Architecture



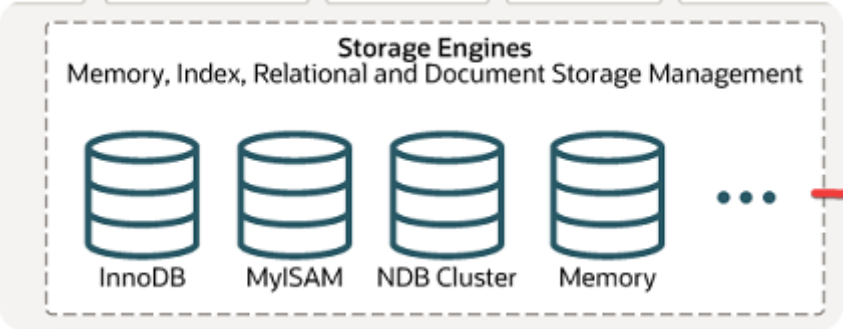
MySQL Architecture is divided into layers , its quit simple but for now we will focus on outer layer 'End-User/Client Layer' and 'physical storage'

- **End-User/Client Layer:** This is where applications or tools that need to work with MySQL, like scripts, come in. They use something called a MySQL connector—a type of software that lets them talk to the MySQL server. This way, they can log in, send their queries, and manage the database. It works with many programming languages, including C, C++, and PHP.
- **Physical Storage:** Whenever data in the database is altered, whether through a programming language or a tool, these changes are permanently stored in the database's data files.
- **Daemon Process**: The heart of MySQL's operations is the main daemon process, known as "mysqld." It interfaces with the client layer, directing tasks to various specialized processes based on whether they involve stored procedures, functions, data definitions, DML, and so on. These specialized processes are referred to as worker processes.



hey analyze queries for syntax errors, check if the queries are already present in a memory segment called the **cache**, and determine the most efficient method to fetch the results swiftly. Once these

steps are completed, the query is handed off to the **storage engine** for execution.
-**MySQL brain** (storage engine): this have deferent storage engines



The storage engine handles how data is stored, including details like where to put certain information, which indexes to use, and where files are kept. It also keeps track of changes for recovering data if needed. This engine works closely with the physical storage of the database. **InnoDB** is the default storage engine used.

MySQL installed File Locations

DATA DIRECTORY

- This is also referred to as the **datadir**
- by default the location is `/var/lib/mysql`
- The directory is owned by the `mysql` OS user, serving as its home directory, this user us automatically created when we install MySQL. if you attempt to delete the user using `userdell -rf` the whole data directory will be delete as well
- Whenever new databases are created, they are stored in separate directories within this data directory.
  ca

```
[root@Mysqlcom-RHEL-STG dba]# ll /var/lib/mysql
total 91592
-rw-r-----. 1 mysql mysql       56 Apr  4 23:28  auto.cnf
-rw-r-----. 1 mysql mysql      826 Apr  4 23:46  binlog.000001
-rw-r-----. 1 mysql mysql       16 Apr  4 23:28  binlog.index
-rw-------. 1 mysql mysql     1676 Apr  4 23:28  ca-key.pem
-rw-r--r--. 1 mysql mysql     1112 Apr  4 23:28  ca.pem
-rw-r--r--. 1 mysql mysql     1112 Apr  4 23:28  client-cert.pem
-rw-------. 1 mysql mysql     1680 Apr  4 23:28  client-key.pem
-rw-r-----. 1 mysql mysql   196608 Apr  4 23:48 '#ib_16384_0.dblwr'
-rw-r-----. 1 mysql mysql  8585216 Apr  4 23:28 '#ib_16384_1.dblwr'
-rw-r-----. 1 mysql mysql     5824 Apr  4 23:28  ib_buffer_pool
-rw-r-----. 1 mysql mysql 12582912 Apr  4 23:46  ibdata1
-rw-r-----. 1 mysql mysql 12582912 Apr  4 23:28  ibtmp1
drwxr-x---. 2 mysql mysql     4096 Apr  4 23:28 '#innodb_redo'
drwxr-x---. 2 mysql mysql      187 Apr  4 23:28 '#innodb_temp'
drwxr-x---. 2 mysql mysql      143 Apr  4 23:28  mysql
-rw-r-----. 1 mysql mysql 26214400 Apr  4 23:46  mysql.ibd
srwxrwxrwx. 1 mysql mysql        0 Apr  4 23:28  mysql.sock
-rw-------. 1 mysql mysql        6 Apr  4 23:28  mysql.sock.lock
drwxr-x---. 2 mysql mysql     8192 Apr  4 23:28  performance_schema
-rw-------. 1 mysql mysql     1676 Apr  4 23:28  private_key.pem
-rw-r--r--. 1 mysql mysql      452 Apr  4 23:28  public_key.pem
-rw-r--r--. 1 mysql mysql     1112 Apr  4 23:28  server-cert.pem
-rw-------. 1 mysql mysql     1676 Apr  4 23:28  server-key.pem
drwxr-x---. 2 mysql mysql       28 Apr  4 23:28  sys
-rw-r-----. 1 mysql mysql 16777216 Apr  4 23:48  undo_001
-rw-r-----. 1 mysql mysql 16777216 Apr  4 23:31  undo_002
[root@Mysqlcom-RHEL-STG dba]#
```

LOGS FILES

- default location is `/var/log/mysql.log`

- This important file records all errors, warnings, and info, etc.
- also contain initial root password for newly installed MySQL

```
[root@Mysqlcom-RHEL-STG dba]# ll /var/log/
total 1756
drwxr-xr-x. 2 root   root     4096 Mar 24 22:22 anaconda
drwx------. 2 root   root       23 Mar 24 22:24 audit
-rw-------. 1 root   root        0 Mar 26 03:44 boot.log
-rw-------. 1 root   root    17889 Mar 26 03:44 boot.log-20240326
-rw-rw----. 1 root   utmp      768 Apr  1 11:39 btmp
-rw-rw----. 1 root   utmp      768 Mar 26 23:15 btmp-20240401
drwxr-x---. 2 chrony chrony      6 Nov 14  2022 chrony
-rw-------. 1 root   root    36829 Apr  5 00:01 cron
-rw-------. 1 root   root    47824 Mar 31 03:01 cron-20240331
-rw-r--r--. 1 root   root   303406 Apr  4 23:28 dnf.librepo.log
-rw-r--r--. 1 root   root   914429 Apr  4 23:28 dnf.log
-rw-r--r--. 1 root   root    47276 Apr  4 23:28 dnf.rpm.log
-rw-r-----. 1 root   root      744 Apr  1 20:54 firewalld
-rw-r--r--. 1 root   root     6240 Apr  4 23:28 hawkey.log
-rw-r--r--. 1 root   root     2880 Mar 31 00:34 hawkey.log-20240331
-rw-rw-r--. 1 root   utmp   292292 Apr  4 23:28 lastlog
-rw-------. 1 root   root        0 Mar 31 03:21 maillog
-rw-------. 1 root   root        0 Mar 24 22:13 maillog-20240331
-rw-------. 1 root   root    63833 Apr  5 00:18 messages
-rw-------. 1 root   root   252671 Mar 31 02:50 messages-20240331
-rw-r-----. 1 mysql  mysql    3064 Apr  4 23:28 mysqld.log
drwx------. 2 root   root        6 Mar 24 22:13 private
drwxr-xr-x. 2 root   root        6 Jan 10 21:45 qemu-ga
drwx------. 3 root   root       17 Feb 20 21:01 samba
-rw-------. 1 root   root     5399 Apr  4 23:40 secure
-rw-------. 1 root   root     4201 Mar 29 20:42 secure-20240331
```

GLOVBAL CONFIGURATION

-default location is `/etc/my.cnf`

contains all the configuration settings that will be loaded when MySQL server start .

it acutely discrip how MySQl server should behave , how MySQL server should run.

you can see from the below image , the file is owned by root , only root or user with sudo can edit this file .

```
[root@Mysqlcom-RHEL-STG dba]# ll /etc/*.cnf
-rw-r--r--. 1 root root 1243 Dec 12 23:12 /etc/my.cnf
[root@Mysqlcom-RHEL-STG dba]#
```

MYSQL Executable programmes

In this section, we'll explore the various MySQL executable programs that come with the installation of the MySQL server. Each of these programs is executable and their names begin with "mysql". As long as a user has the necessary permissions, they can run these programs.

Executable Name	Description
`mysqld`	The main MySQL server daemon.
`mysql`	The command-line tool to interact with MySQL.
`mysqladmin`	A client for performing administrative operations.
`mysqldump`	Utility for backing up MySQL databases.
`mysql_secure_installation`	Script to secure MySQL installations.
`mysqlshow`	Displays database, table, and column information.
`mysqlimport`	A command-line interface to load data into MySQL.
`mysqlcheck`	Checks, repairs, optimizes, and analyzes tables.

these programmes are located under `/usr/bin/mysql*`

```
[root@Mysqlcom-RHEL-STG dba]# ll /usr/bin/mysql*
-rwxr-xr-x. 1 root root 7717368 Dec 12 23:22 /usr/bin/mysql
-rwxr-xr-x. 1 root root 7403392 Dec 12 23:22 /usr/bin/mysqladmin
-rwxr-xr-x. 1 root root 7878672 Dec 12 23:22 /usr/bin/mysqlbinlog
-rwxr-xr-x. 1 root root 7416896 Dec 12 23:22 /usr/bin/mysqlcheck
-rwxr-xr-x. 1 root root 6520296 Dec 12 23:22 /usr/bin/mysql_config_editor
-rwxr-xr-x. 1 root root    4368 Dec 12 22:49 /usr/bin/mysqld_pre_systemd
-rwxr-xr-x. 1 root root 7485016 Dec 12 23:22 /usr/bin/mysqldump
-rwxr-xr-x. 1 root root    7669 Dec 12 22:49 /usr/bin/mysqldumpslow
-rwxr-xr-x. 1 root root 7399896 Dec 12 23:22 /usr/bin/mysqlimport
-rwxr-xr-x. 1 root root 7492144 Dec 12 23:22 /usr/bin/mysql_migrate_keyring
-rwxr-xr-x. 1 root root 8025920 Dec 12 23:22 /usr/bin/mysqlpump
-rwxr-xr-x. 1 root root 7387400 Dec 12 23:22 /usr/bin/mysql_secure_installation
-rwxr-xr-x. 1 root root 7394104 Dec 12 23:22 /usr/bin/mysqlshow
-rwxr-xr-x. 1 root root 7413664 Dec 12 23:22 /usr/bin/mysqlslap
-rwxr-xr-x. 1 root root 6555632 Dec 12 23:22 /usr/bin/mysql_ssl_rsa_setup
-rwxr-xr-x. 1 root root 6431424 Dec 12 23:22 /usr/bin/mysql_tzinfo_to_sql
-rwxr-xr-x. 1 root root 7510056 Dec 12 23:22 /usr/bin/mysql_upgrade
[root@Mysqlcom-RHEL-STG dba]#
```

MySQL services under systemd

MySQL services are controlled by systemd

systemd provide utility called `systmctl` that we can use to start ,stop , enable , disable MySQL

services

```
[root@Mysqlcom-RHEL-STG dba]# syst mctl status mysqld.service
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2024-04-04 23:28:56 +03; 1h 24min ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 52527 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 52600 (mysqld)
   Status: "Server is operational"
    Tasks: 38 (limit: 24155)
   Memory: 506.5M
   CGroup: /system.slice/mysqld.service
           └─52600 /usr/sbin/mysqld

Apr 04 23:28:43 Mysqlcom-RHEL-STG systemd[1]: Starting MySQL Server...
Apr 04 23:28:56 Mysqlcom-RHEL-STG systemd[1]: Started MySQL Server.
```

MySQL Shell Commands

there are some basic shell commands that are available to use that can be very useful for our day to day DBA activity's
the command run under MySQL shell , so we need to login to MySQL before we run these commands

help command

-syntax is `\h` or `\?`

- prints help about MySQL Shell and all available shell command
- Display help for any of the shell commands

```
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> \h

For information about MySQL products and services, visit:
   http://www.mysql.com/
For developer information, including the MySQL Reference Manual, visit:
   http://dev.mysql.com/
To buy MySQL Enterprise support, training, or other products, visit:
   https://shop.mysql.com/

List of all MySQL commands:
Note that all text commands must be first on line and end with ';'
?         (\?) Synonym for `help'.
clear     (\c) Clear the current input statement.
connect   (\r) Reconnect to the server. Optional arguments are db and host.
delimiter (\d) Set statement delimiter.
edit      (\e) Edit command with $EDITOR.
ego       (\G) Send command to mysql server, display result vertically.
exit      (\q) Exit mysql. Same as quit.
go        (\g) Send command to mysql server.
help      (\h) Display this help.
nopager   (\n) Disable pager, print to stdout.
notee     (\t) Don't write into outfile.
pager     (\P) Set PAGER [to_pager]. Print the query results via PAGER.
print     (\p) Print current command.
```

quit

-quits or exits from MySQL Shell syntax `\q`

```
mysql> \q
Bye
[root@Mysqlcom-RHEL-STG dba]#
```

status

- shortcut is `\s`
- display for how long MySQL server has been up , what is my connection id , version of MySQL
- display if the current user logged in locally or from remote location.

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> \s
--------------
mysql  Ver 8.0.36 for Linux on x86_64 (MySQL Community Server - GPL)

Connection id:          30
Current database:
Current user:           root@localhost
SSL:                    Not in use
Current pager:          stdout
Using outfile:          ''
Using delimiter:        ;
Server version:         8.0.36 MySQL Community Server - GPL
Protocol version:       10
Connection:             Localhost via UNIX socket
Server characterset:    utf8mb4
Db     characterset:    utf8mb4
Client characterset:    utf8mb4
Conn.  characterset:    utf8mb4
UNIX socket:            /var/lib/mysql/mysql.sock
Binary data as:         Hexadecimal
Uptime:                 1 hour 39 min 38 sec

Threads: 2  Questions: 22  Slow queries: 0  Opens: 151  Flush tables: 3  Open tables: 67  Queries per second avg: 0.003
--------------

mysql>
```

system

- syntax is `\!`
- allow you to run OS command from inside MySQL shell

```
mysql> clear
mysql> \! pwd
/home/dba
mysql>
```

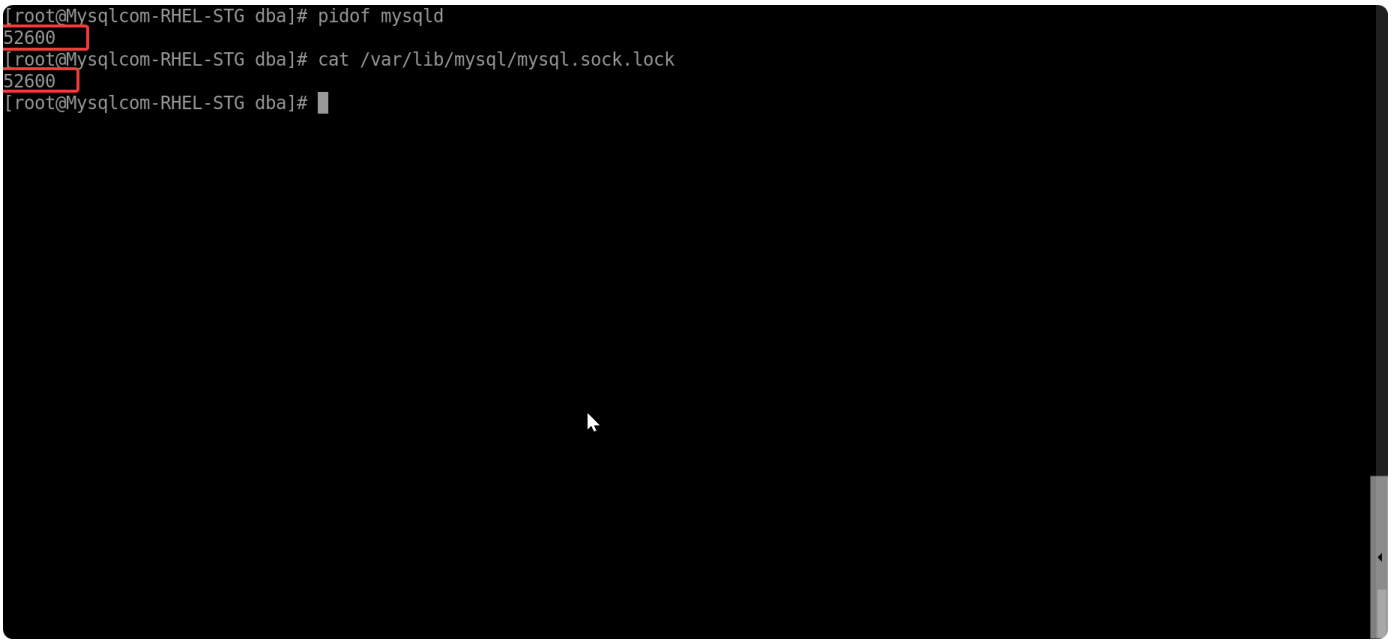
```
mysql> \! systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2024-04-04 23:28:56 +03; 1h 40min ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 52527 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 52600 (mysqld)
   Status: "Server is operational"
    Tasks: 38 (limit: 24155)
   Memory: 505.4M
   CGroup: /system.slice/mysqld.service
           └─52600 /usr/sbin/mysqld

Apr 04 23:28:43 Mysqlcom-RHEL-STG systemd[1]: Starting MySQL Server...
Apr 04 23:28:56 Mysqlcom-RHEL-STG systemd[1]: Started MySQL Server.
mysql>
```

MySQL Socket File

- special file that Manges connection to the MySQL server, if user is on the database host and want to connect to MySQL locally , then this file is needed without this file users cannot connect
- its owned by MySQL OS user and default location is `/var/lib/mysql`
- keep in in mind the following terms for reference **UNIX socket**=local connection , **TCP/IP**= Remote connection
- this file is empty by default , but MySQL server creates another file `mysql.sock.lock` and add pid for MySQLd services

```
[root@Mysqlcom-RHEL-STG dba]# ll /var/lib/mysql
total 91592
-rw-r-----. 1 mysql mysql       56 Apr  4 23:28  auto.cnf
-rw-r-----. 1 mysql mysql      826 Apr  4 23:46  binlog.000001
-rw-r-----. 1 mysql mysql       16 Apr  4 23:28  binlog.index
-rw-------. 1 mysql mysql     1676 Apr  4 23:28  ca-key.pem
-rw-r--r--. 1 mysql mysql     1112 Apr  4 23:28  ca.pem
-rw-r--r--. 1 mysql mysql     1112 Apr  4 23:28  client-cert.pem
-rw-------. 1 mysql mysql     1680 Apr  4 23:28  client-key.pem
-rw-r-----. 1 mysql mysql   196608 Apr  4 23:48 '#ib_16384_0.dblwr'
-rw-r-----. 1 mysql mysql  8585216 Apr  4 23:28 '#ib_16384_1.dblwr'
-rw-r-----. 1 mysql mysql     5824 Apr  4 23:28  ib_buffer_pool
-rw-r-----. 1 mysql mysql 12582912 Apr  4 23:46  ibdata1
-rw-r-----. 1 mysql mysql 12582912 Apr  4 23:28  ibtmp1
drwxr-x---. 2 mysql mysql     4096 Apr  4 23:28 '#innodb_redo'
drwxr-x---. 2 mysql mysql      187 Apr  4 23:28 '#innodb_temp'
drwxr-x---. 2 mysql mysql      143 Apr  4 23:28  mysql
-rw-r-----. 1 mysql mysql 26214400 Apr  4 23:46  mysql.ibd
srwxrwxrwx. 1 mysql mysql        0 Apr  4 23:28  mysql.sock
-rw-------. 1 mysql mysql        6 Apr  4 23:28  mysql.sock.lock
drwxr-x---. 2 mysql mysql     8192 Apr  4 23:28  performance_schema
-rw-------. 1 mysql mysql     1676 Apr  4 23:28  private_key.pem
-rw-r--r--. 1 mysql mysql      452 Apr  4 23:28  public_key.pem
-rw-r--r--. 1 mysql mysql     1112 Apr  4 23:28  server-cert.pem
-rw-------. 1 mysql mysql     1676 Apr  4 23:28  server-key.pem
drwxr-x---. 2 mysql mysql       28 Apr  4 23:28  sys
-rw-r-----. 1 mysql mysql 16777216 Apr  4 23:48  undo_001
-rw-r-----. 1 mysql mysql 16777216 Apr  4 23:31  undo_002
[root@Mysqlcom-RHEL-STG dba]#
```

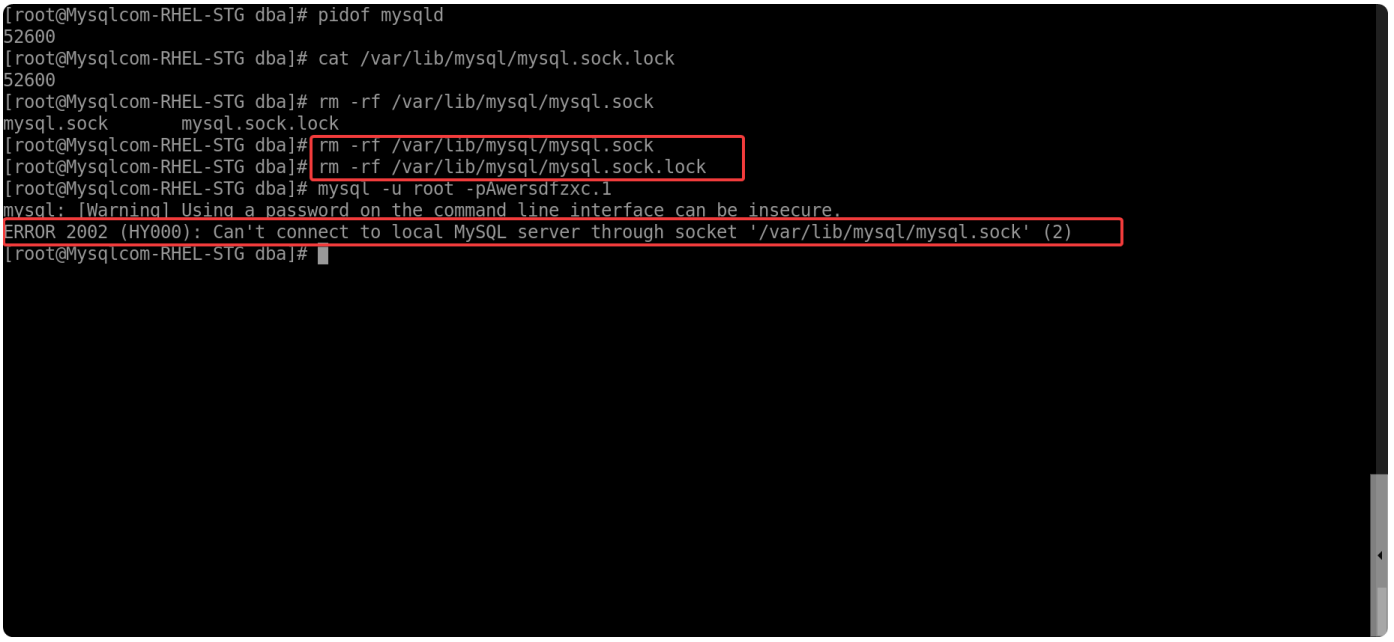
if we check pid of the mysqld services using `pidof mysqld` and then check the content of the file mysql.sock.lock we will find them matching the same pid



what happen if we delete both file

let see if we attempt to delete both mysql.sock and mysql.sock.lock , will we be able to connect to mysql

you can use `rm -rf [filepath]` to delete both



we are unable to connect because of the absents of the two files,.

in order to have the files back we need to to restart the mysqld so MySQL can recreate these files
`systemctl restart mysqld`

```
[root@Mysqlcom-RHEL-STG dba]# systemctl restart  mysqld.service
(failed reverse-i-search)`': ^C
[root@Mysqlcom-RHEL-STG dba]# mysql -u root -pAwersdfzxc.1
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.36 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

MYSQL GLOBAL Variables

- MySQL server maintains many system variables that are used to configure how MySQL should operate
- these system variables has two scops **1.GLOBAL scope** & **2.SESSION scope**
- global variable effect the overall operation of mysql server
- each value in global variable can be changed in option file or on command line .
- each global variable has default value
- all these global variable are identified by using `@@` sing

how to see the values of the global variables

there two ways we can see the values of the global variables

1. using `show global [variable_name ]`
2. or using `select @@[variable_name]`

example of system variable :

- `max_connection`
- `server_id`
- `sql_mode`

retrieve and set these global values

to show all the values in global variable we can use the below command , which will retwrive all the values
`show global variables;`

```
ERROR:
No query specified

mysql> show global variables \g;
```

```
| slave_checkpoint_period                    | 300                    |
| slave_compressed_protocol                  | OFF                    |
| slave_exec_mode                            | STRICT                 |
| slave_load_tmpdir                          | /tmp                   |
| slave_max_allowed_packet                   | 1073741824             |
| slave_net_timeout                          | 60                     |
```

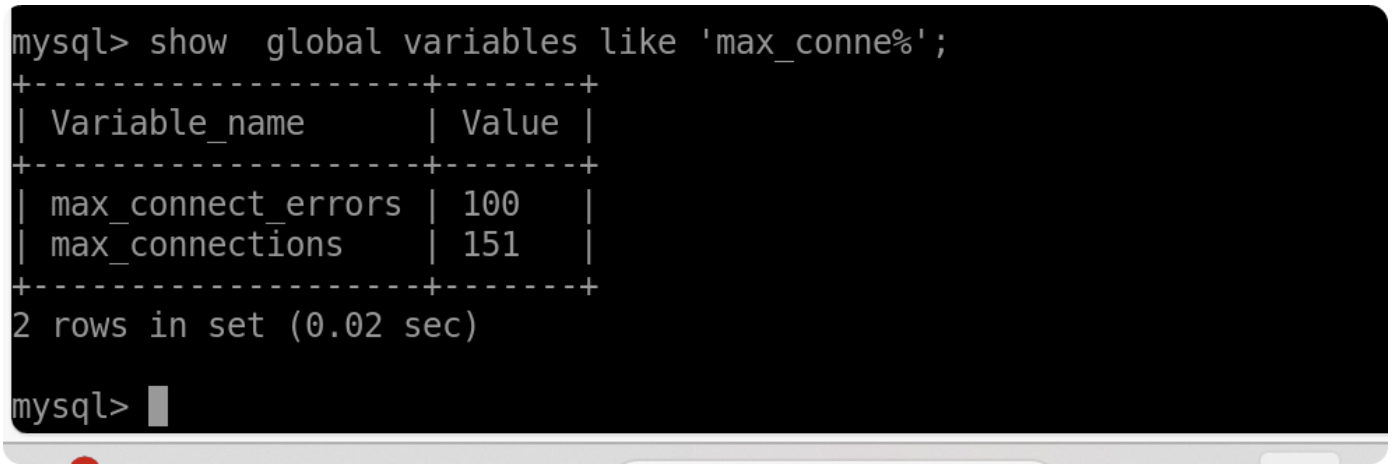
these are all by default system variables that come along with MySQL server that are bundled in mysql8 version
deferent version might have deferent list
to find specific global variables , we can filter the list using `like` and then putting the variable we are looking for

`show global variables like 'max_connections';`

```
mysql> show global variables like 'max_connections';
+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| max_connections | 151   |
+-----------------+-------+
1 row in set (0.02 sec)

mysql>
```

if you don't know the exact name of the variable you can type the name follow by `%`

`show global variables like 'max_conne%';`

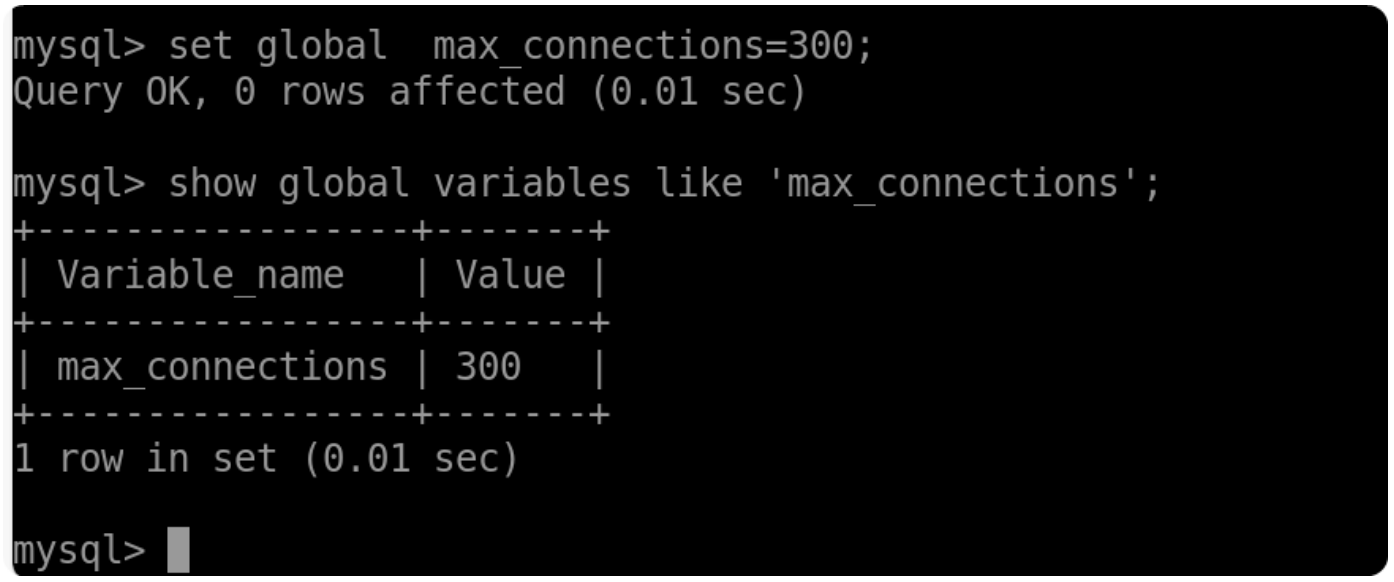


set the value for global variable

to set the value for global variable we will use the `set global [variable_name]=[new_value];`

for this insist we will change the value for max_connections to 300

`set global max_connections=300;`



note: that the changed value will return to default if we restart the MySQL , to make the global variable value permeate then we need to make the changes in the file itself .

```
mysql> \! systemctl restart mysqld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to restart 'mysqld.service'.
Authenticating as: dba
Password:
==== AUTHENTICATION COMPLETE ====
mysql> show global variables like 'max_connections';
ERROR 2013 (HY000): Lost connection to MySQL server during query
No connection. Trying to reconnect...
Connection id:    8
Current database: *** NONE ***

+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| max_connections | 151   |
+-----------------+-------+
1 row in set (0.01 sec)

mysql>
```

session variables what are they and how retrieve and edit session variables

- session variables affect only in the current session that you login in
- default value for session variables can only be changed on command line
- once you disconnect the session , new values for session variables will be rollback to the default values
- session variables value can be retrieve by using `@@variabales`
- or using `show session variables like 'variable_name ';`
- or using `select @@[variable_name];`
- example of session variables is `sql_mode`

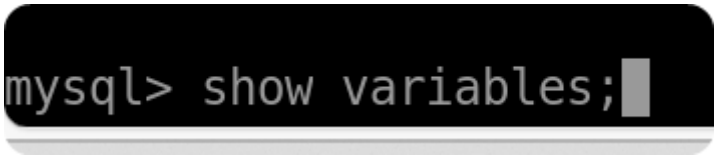
how retrieve and edit session variables

to view all session variables we can use the below command

`show variables ;`

or you can use the below

`show session variables;`



```
| report_port                                | 3306
|
| report_user                                |
|
| require_row_format                         | OFF
|
| require_secure_transport                   | OFF
|
| resultset_metadata                         | FULL
|
| rpl_read_size                              | 8192
```

to retrieve certain variable we can use `like` follow by the variable we are looking for

`show session variables like 'sql_mode';`

```
mysql> show session variables like 'sql_mode';
+---------------+-----------------------------------------------------------------------------------------------------------------------------------------------+
| Variable_name | Value                                                                                                                                         |
+---------------+-----------------------------------------------------------------------------------------------------------------------------------------------+
| sql_mode      | ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION |
+---------------+-----------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)

mysql>
```

or using the `select @@[variable_name];`

`select @@sql_mode;`

```
mysql> select @@sql_mode
    -> ;
+-----------------------------------------------------------------------------------------------------------------------------------------------+
| @@sql_mode                                                                                                                                    |
+-----------------------------------------------------------------------------------------------------------------------------------------------+
| ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION |
+-----------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

edit session variables

the syntax is similar to what we did in global variables , we will use the syntax `set sesssion [variable_name]=[new value ];`

for this example we will edit the value for `sql_mode`

first retrieve the value for `sql_mode` copy the one of output then we will set it up by new value

```
mysql> select @@sql_mode
    -> ;
+--------------------------------------------------------------------------------------------------------------------------+
| @@sql_mode                                                                                                               |
+--------------------------------------------------------------------------------------------------------------------------+
| ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION |
+--------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql>
mysql> set session  sql_mode = NO_ENGINE_SUBSTITUTION ;
Query OK, 0 rows affected (0.00 sec)

mysql> select @@sql_mode;
+------------------------+
| @@sql_mode             |
+------------------------+
| NO_ENGINE_SUBSTITUTION |
+------------------------+
1 row in set (0.00 sec)

mysql>
```

if we disconnect from session and login again the value will rollback

```
| NO_ENGINE_SUBSTITUTION |
+------------------------+
1 row in set (0.00 sec)

mysql> Bye
[dba@mysqlPerconaSTG ~]$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select @@sql_mode;
+--------------------------------------------------------------------------------------------------------------------------+
| @@sql_mode                                                                                                               |
+--------------------------------------------------------------------------------------------------------------------------+
| ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION |
+--------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

GETTING SYSTEM VARIABLE HELP

for us is very difficult to remember all the MySQL global variables , what are they , what type of values we can apply.
for that is best if we relay on MySQL pages to get to know these variables
link for MySQL global variables page.

MySQL show commands

MySQL show command is special query to view the information schema of any record that are stored in their database.
this is read only SQL statement , so you can run as many command as you want without effect or changing anything on database .
they have many uses such as

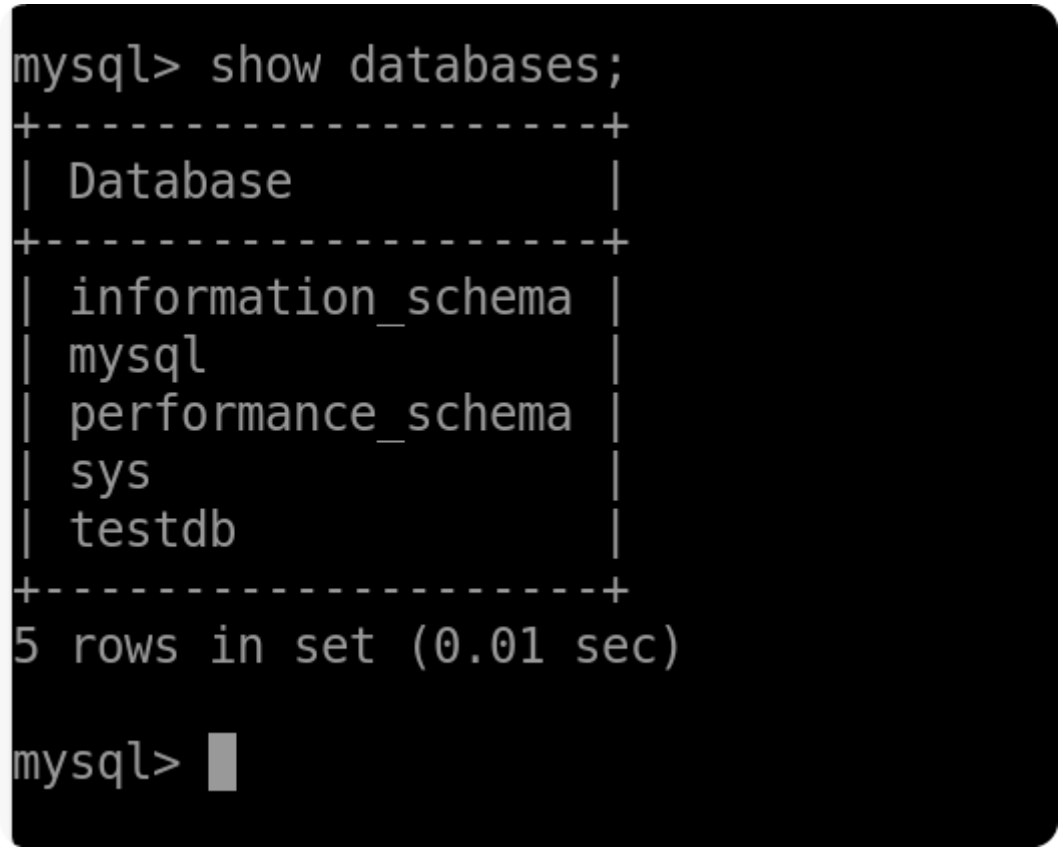
- display all connected session

- the connections to the database
- list all the database
- list all the tables in certain database
- list all the database jobs that are scheduled to run
- it can show how certain table , a user , a database , a trigger event was created .

example of it are :

- `show databases;`
- `show tables like '%view%';`
- `show binary logs;`
- `show binlog events;`
- `show engines;`
- `show create table | user | database;`
- `show errors;`
- `show warninges;`
- `show events;`
- `show triggers;`
- `show processlist;`

the command `show databases;` will list all the databases in MySQL server



the command `show processlist;` will view all connections connected to mysql server .

```
mysql> show processlist;
+----+-----------------+-----------+------+---------+------+-----------------------+------------------+---------+-----------+----
-----------+
| Id | User            | Host      | db   | Command | Time | State                 | Info             | Time_ms | Rows_sent | Row
s_examined |
+----+-----------------+-----------+------+---------+------+-----------------------+------------------+---------+-----------+----
-----------+
|  5 | event_scheduler | localhost | NULL | Daemon  | 2035 | Waiting on empty queue | NULL             | 2034332 |         0 |
         0 |
| 10 | root            | localhost | NULL | Query   |    0 | init                  | show processlist |       0 |         0 |
         0 |
+----+-----------------+-----------+------+---------+------+-----------------------+------------------+---------+-----------+----
-----------+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

the output showed two rows which means two connection are they one of them is the user that i connecting to , the other one is event scheduler.

the command `show events;` will display any jobs that are scheduled.

```
mysql> show events;
ERROR 1046 (3D000): No database selected
mysql>
```

the command `show tables ;` will display all the tables in the database;

to use it you must change to a database you want to view

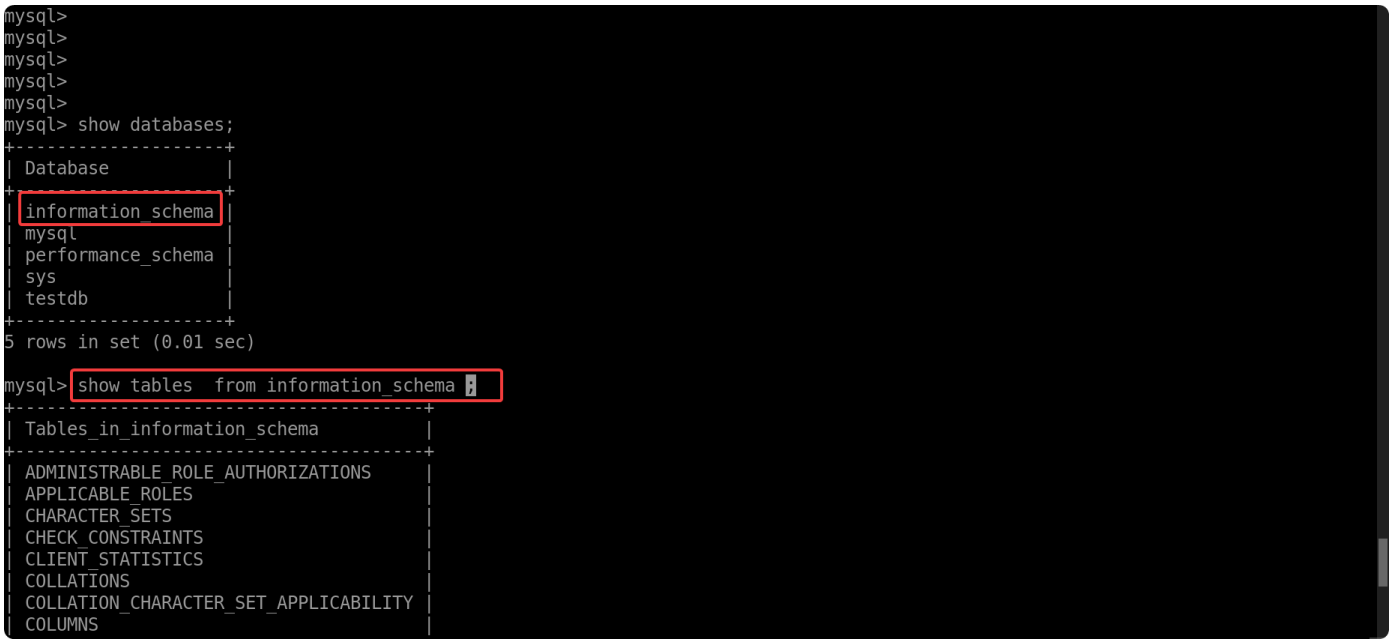
`use [database_name];`

```
| time_zone_transition_type                  |
| user                                       |
+--------------------------------------------+
38 rows in set (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| testdb             |
+--------------------+
5 rows in set (0.00 sec)

mysql> use mysql;
Database changed
mysql> show tables ;
+---------------------------------------------+
| Tables_in_mysql                             |
+---------------------------------------------+
| columns_priv                                |
| component                                   |
| db                                          |
| default_roles                               |
| engine_cost                                 |
| func                                        |
```

alternately you can view all tables in databases without the need to t change to the databases using `use`

syntax

`show tables from [database name];`

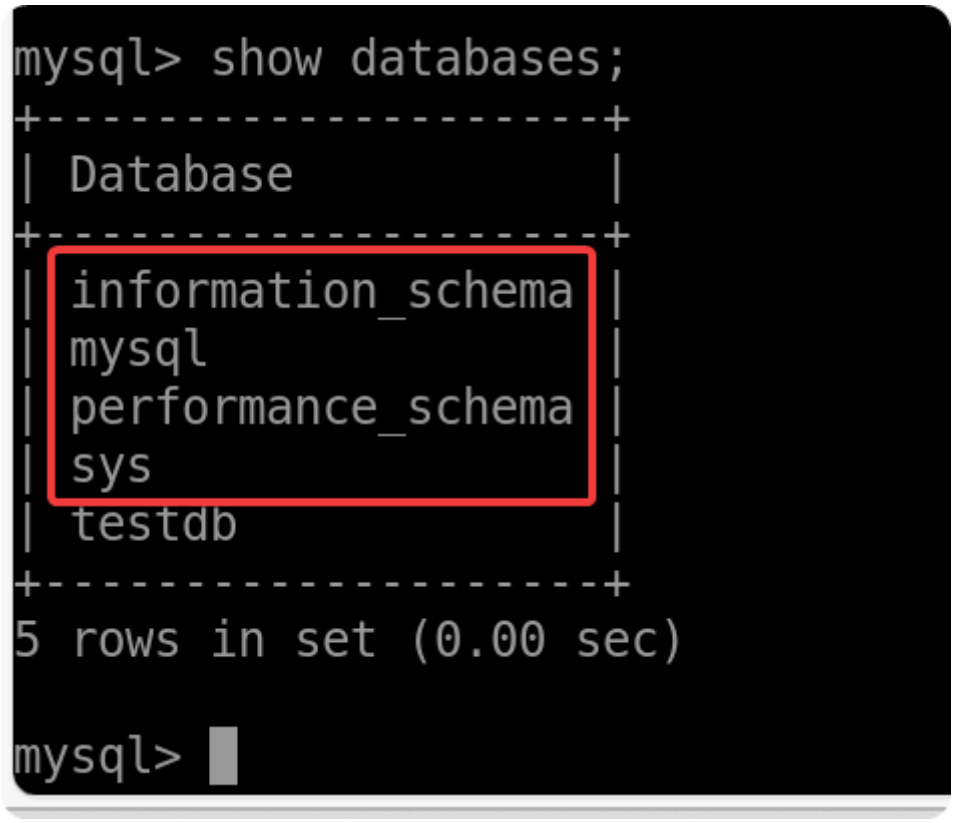


MySQL System Databases

MySQL server comes with default system databases

- **information_schema**: A built-in database in every MySQL instance, also known as the system catalogue or data dictionary. It provides access to metadata, which is data about other data. The "tables" in this database are actually read-only views, meaning you cannot perform insert, update, or delete operations on them.
- **mysql**: This database contains tables essential for the MySQL server's operation. It includes information about user accounts, the event scheduler's registry, installed plugins, replication system tables, and time zone data.
- **performance_schema**: Focuses on the internal execution of the server, offering insights into how the server is performing. It is designed to collect performance data, including event waits, database locks, and memory allocation details.
- **sys**: A more user-friendly alternative to the performance_schema, providing a collection of views, functions, and stored procedures. These tools assist MySQL administrators in gaining insights into database usage, including the number of connections established, memory usage by users, and the frequency of table scans.
- **Test**: This is a default database that can be safely removed as part of securing a MySQL installation, typically through the `mysql_secure_installation` script.

when you login in MySQL you can see these databases by using command `show databases;`



let first check the performance_schema ad list all there tables.

```
| VIEWS                                |
| VIEW_ROUTINE_USAGE                   |
| VIEW_TABLE_USAGE                     |
+--------------------------------------+
88 rows in set (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| testdb             |
+--------------------+
5 rows in set (0.00 sec)

mysql> use information_schema;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+---------------------------------------+
| Tables_in_information_schema          |
+---------------------------------------+
| ADMINISTRABLE_ROLE_AUTHORIZATIONS     |
| APPLICABLE_ROLES                      |
+---------------------------------------+
```

```
+---------------------------------------+
| Tables_in_information_schema          |
+---------------------------------------+
| ADMINISTRABLE_ROLE_AUTHORIZATIONS     |
| APPLICABLE_ROLES                      |
| CHARACTER_SETS                        |
| CHECK_CONSTRAINTS                     |
| CLIENT_STATISTICS                     |
| COLLATIONS                            |
| COLLATION_CHARACTER_SET_APPLICABILITY |
| COLUMNS                               |
| COLUMNS_EXTENSIONS                    |
| COLUMN_PRIVILEGES                     |
| COLUMN_STATISTICS                     |
| COMPRESSION_DICTIONARY                |
| COMPRESSION_DICTIONARY_TABLES         |
| ENABLED_ROLES                         |
| ENGINES                               |
| EVENTS                                |
| FILES                                 |
| GLOBAL_TEMPORARY_TABLES               |
| INDEX_STATISTICS                      |
| INNODB_BUFFER_PAGE                    |
| INNODB_BUFFER_PAGE_LRU                |
| INNODB_BUFFER_POOL_STATS              |
| INNODB_CACHED_INDEXES                 |
| INNODB_CMP                            |
| INNODB_CMPMEM                         |
| INNODB_CMPMEM_RESET                   |
```

these are acutely not tables , these are acutely views , so you cannot make in modification on them , you are free to explore these views .

before we dig deep its best if we get some information about the table using `describe` command , whcih will show details about the table and values that are stored and column names .
for instate we will check table called engines
`describe engines;`

```
mysql> describe  engines
    -> ;
+--------------+------------+------+-----+---------+-------+
| Field        | Type       | Null | Key | Default | Extra |
+--------------+------------+------+-----+---------+-------+
| ENGINE       | varchar(64) | NO   |     |         |       |
| SUPPORT      | varchar(8)  | NO   |     |         |       |
| COMMENT      | varchar(80) | NO   |     |         |       |
| TRANSACTIONS | varchar(3)  | YES  |     |         |       |
| XA           | varchar(3)  | YES  |     |         |       |
| SAVEPOINTS   | varchar(3)  | YES  |     |         |       |
+--------------+------------+------+-----+---------+-------+
6 rows in set (0.01 sec)

mysql>
```

reset can be done on all system databases;

MySQL Local vs Remote Connections

each user that connect to MySQL server ae connected as

1. localhost-connections
2. specific-host-connections
3. any-host-connection

localhost-connections

- localhost : means this user can only log in from the database server itself .
- `root@localhost` means root can only login from database server itself and not from anywhere else .

specific-host-connection:

in some cases you want to create a specific application user that can only connect to MySQL server from a specific web server where the application resides .
it could be hostname , or it could be ip address
-host or Ip Address such as webserver01 or 192.168.10.10 it could be hostname , or it could be ip address both are allowed

- `app_user@webserver01` clearly states that user `app_user` can only connect from hostname called webserver01

any-host-connection

this will remove all restriction and allow the user to connect from anywhere
-`%` means any source are allowed

-`dba@%` means the user dba can connect to MySQL server from anywhere .

retrieve the created user and there connection setting

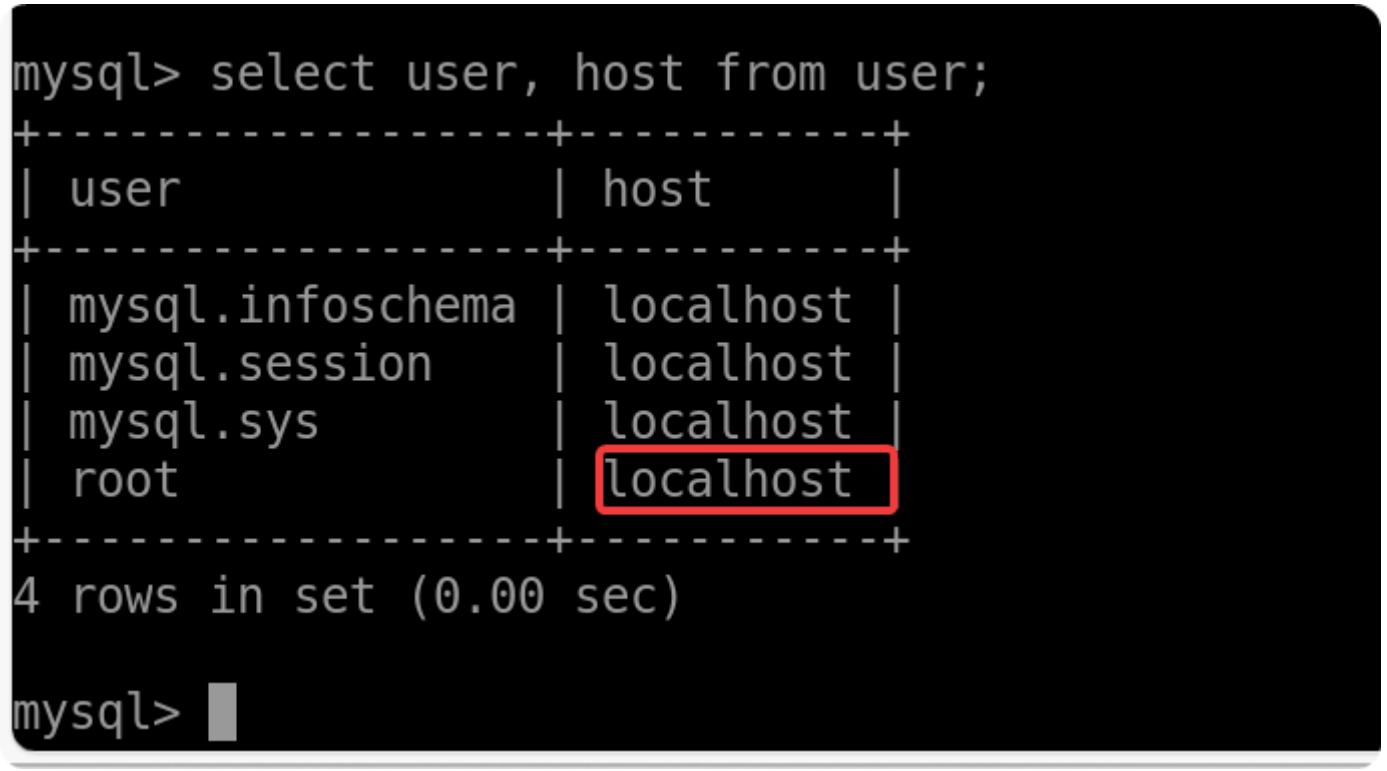
each user that created to connect to MySQL server is stored in user table indie `mysql` system database

```
mysql> use mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+---------------------------------------------------------+
| Tables_in_mysql                                         |
+---------------------------------------------------------+
| columns_priv                                            |
| component                                               |
| db                                                      |
| default_roles                                           |
| engine_cost                                             |
| func                                                    |
| general_log                                             |
| global_grants                                           |
| gtid_executed                                           |
| help_category                                           |
| help_keyword                                            |
| help_relation                                           |
| innodb_index_stats                                      |
| innodb_table_stats                                      |
| ndb_binlog_index                                        |
| password_history                                        |
| plugin                                                  |
| procs_priv                                              |
| proxies_priv                                            |
| replication_asynchronous_connection_failover            |
| replication_asynchronous_connection_failover_managed    |
| replication_group_configuration_version                 |
| replication_group_member_actions                        |
| role_edges                                              |
| server_cost                                             |
| servers                                                 |
| slave_master_info                                       |
| slave_relay_log_info                                    |
| slave_worker_info                                       |
| slow_log                                                |
| tables_priv                                             |
| time_zone                                               |
| time_zone_leap_second                                   |
| time_zone_name                                          |
| time_zone_transition                                    |
| time_zone_transition_type                               |
| user                                                    |
+---------------------------------------------------------+
38 rows in set (0.00 sec)

mysql>
```

we can use `describe user;` to help us understand the structure of the table

```
| slave_worker_info                         |
| slow_log                                  |
| tables_priv                               |
| time_zone                                 |
| time_zone_leap_second                     |
| time_zone_name                            |
| time_zone_transition                      |
| time_zone_transition_type                 |
| user                                      |
+-------------------------------------------+
38 rows in set (0.00 sec)

mysql> describe user;
+------------------------+---------------------------------+------+-----+---------------------+-------+
| Field                  | Type                            | Null | Key | Default             | Extra |
+------------------------+---------------------------------+------+-----+---------------------+-------+
| Host                   | char(255)                       | NO   | PRI |                     |       |
| User                   | char(32)                        | NO   | PRI |                     |       |
| Select_priv            | enum('N','Y')                   | NO   |     | N                   |       |
| Insert_priv            | enum('N','Y')                   | NO   |     | N                   |       |
| Update_priv            | enum('N','Y')                   | NO   |     | N                   |       |
| Delete_priv            | enum('N','Y')                   | NO   |     | N                   |       |
| Create_priv            | enum('N','Y')                   | NO   |     | N                   |       |
| Drop_priv              | enum('N','Y')                   | NO   |     | N                   |       |
| Reload_priv            | enum('N','Y')                   | NO   |     | N                   |       |
| Shutdown_priv          | enum('N','Y')                   | NO   |     | N                   |       |
| Process_priv           | enum('N','Y')                   | NO   |     | N                   |       |
| File_priv              | enum('N','Y')                   | NO   |     | N                   |       |
| Grant_priv             | enum('N','Y')                   | NO   |     | N                   |       |
```

let's view all user that are created in database
`select user, host from user;`



you can see that user `root` is set to only allowed to connect from localhost.

MySQL Shell

its separate program that you install on either Linux or windows
the syntax of MySQL shell to start MySQL shell is `mysqlsh`

- advanced command-line client and code editor for MySQL
- offers scripting capabilities for Python and JavaScript

- mysql shell connect to MySQL server through the X protocol (mysqlx.sock mysqlx.sock.lock).

syntax

```
mysqlsh> \connect-mysqlx|--mysql user@server:port
```

example

```
mysqlsh>\connect-mysql bob@centos7:3306
```

3- Basic MySQL Database Administration

Storing MySQL Authentication Credentials

mysql_config_editor:

- we can create encrypted file that contain root credential , so root doesn't have to fill password each time it connect to MySQL
- **mysql_config_edito**r is one of MySQL executable programmes that configure, it configure the authentication information in a hidden file of the current user's directory.

- the hidden file name is `.mylogin.cnf`
- so when we invoke a client program like MySQL to connect to the server client actually uses this `.mylogin.cnf` to read the crentials .
- syntax: `mysql_config_editor set-login=client --host=user=--password`
- and by default client reads the [client] and [mysql] group
-

using mysql_config_editor

[reference docs]

To configure a MySQL account to store credentials in an encrypted format, allowing users to log in without entering a password, use the following syntax:
`mysql_config_editor set --user=root --login-path=client --password`.

To explore what can be done and to understand the available options, use the `help` command with:
`mysql_config_editor set --help`.

```
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Sat Apr  6 22:25:23 2024 from 10.10.10.4
[dba@mysqlPerconaSTG ~]$ mysql_config_editor set --help
mysql_config_editor  Ver 8.0.36-28 for Linux on x86_64 (Percona Server (GPL), Release 28, Revision 47601f19)
Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2012, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

MySQL Configuration Utility.

Description: Write a login path to the login file.
Usage: mysql_config_editor [program options] [set [command options]]
  -?, --help          Display this help and exit.
  -h, --host=name     Host name to be entered into the login file.
  -G, --login-path=name
                      Name of the login path to use in the login file. (Default
                      : client)
  -p, --password      Prompt for password to be entered into the login file.
  -u, --user=name     User name to be entered into the login file.
  -S, --socket=name   Socket path to be entered into login file.
  -P, --port=name     Port number to be entered into login file.
  -w, --warn          Warn and ask for confirmation if set command attempts to
                      overwrite an existing login path (enabled by default).
                      (Defaults to on; use --skip-warn to disable.)
```

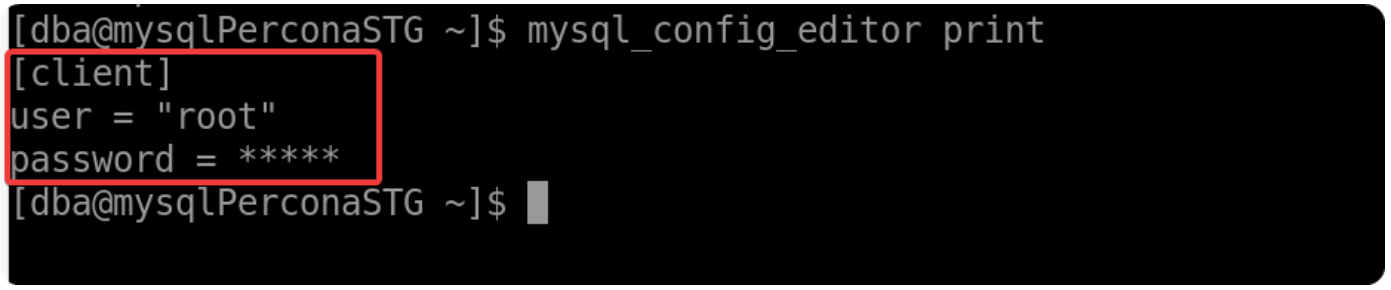
```
owners.

MySQL Configuration Utility.

Description: Write a login path to the login file.
Usage: mysql_config_editor [program options] [set [command options]]
  -?, --help          Display this help and exit.
  -h, --host=name     Host name to be entered into the login file.
  -G, --login-path=name
                      Name of the login path to use in the login file. (Default
                      : client)
  -p, --password      Prompt for password to be entered into the login file.
  -u, --user=name     User name to be entered into the login file.
  -S, --socket=name   Socket path to be entered into login file.
  -P, --port=name     Port number to be entered into login file.
  -w, --warn          Warn and ask for confirmation if set command attempts to
                      overwrite an existing login path (enabled by default).
                      (Defaults to on; use --skip-warn to disable.)
                                          I
Variables (--variable-name=value)
and boolean options {FALSE|TRUE}  Value (after reading options)
--------------------------------- ----------------------------------------
host                              (No default value)
login-path                        client
user                              (No default value)
socket                            (No default value)
port                              (No default value)
warn                              TRUE
[dba@mysqlPerconaSTG ~]$
```

there are option such as specify which host can user login without needing to entering the password . will start by configuring file for root user the command as below

`mysql_config_editor set --user=root --login-path=client --password`

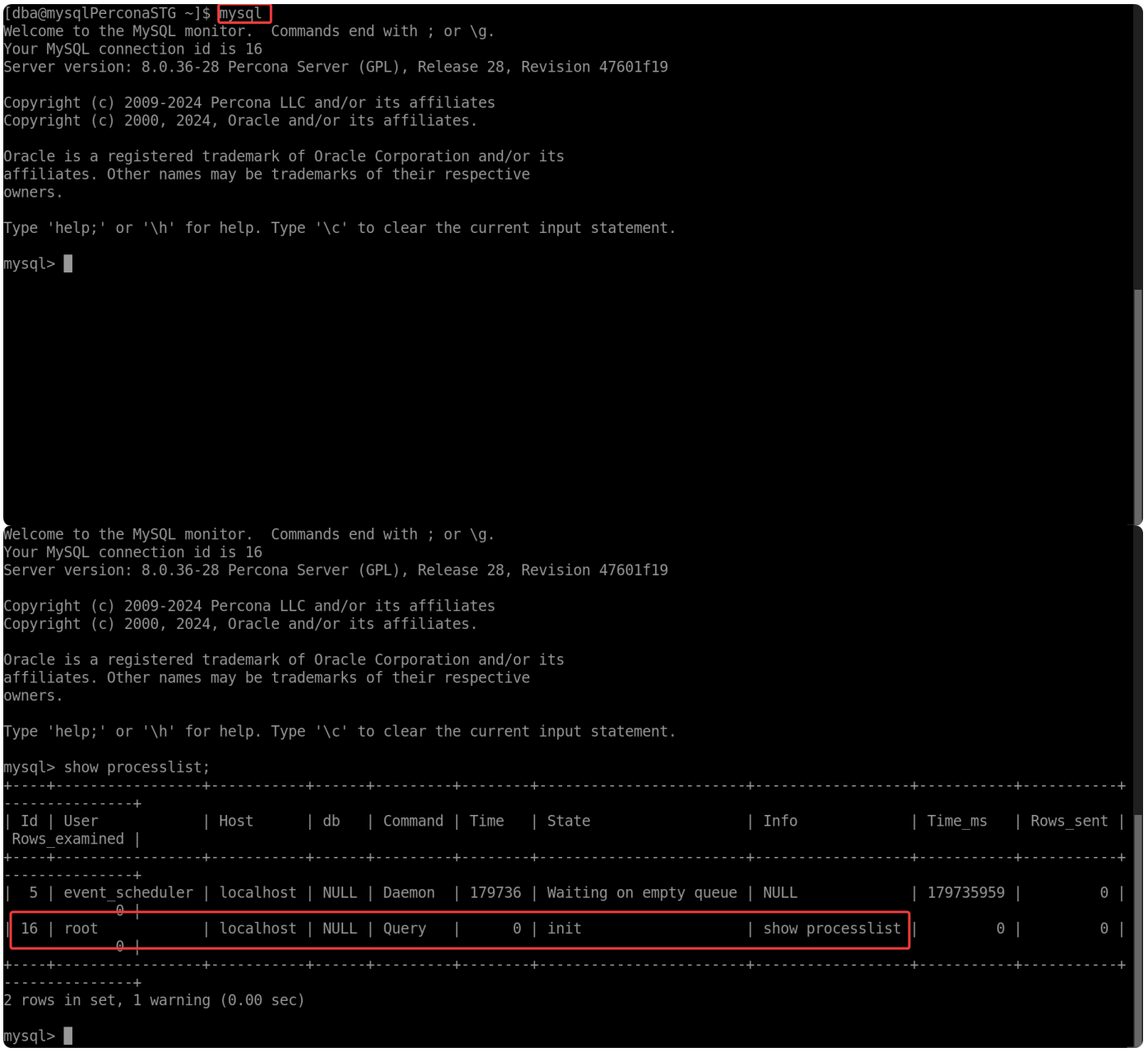
the command will ask you to put the root password for MySQL

to confirm that mysql_config_editor has created the password file we can use the below command `mysql_config_editor print`



now to connect we don't need to use the following `mysql -uroot -p`

we will simply just type `mysql` and it should connect directory .



MySQL admin program

The MySQL admin client is a utility for performing various administrative tasks on a MySQL server. With it, you can execute several operations, such as:

- Shutting down the MySQL server.
- Creating or dropping a database with `create [database name]` or `drop [database name]`.
- Checking the current status of the server.
- Pinging the server to verify if it's running.
- Starting or stopping replication.

The syntax to use MySQL admin is: `mysqladmin [options] [command]`.
After configuring the root user's credentials with `mysql_config_editor`, it's no longer necessary to use `-uroot -p` to authenticate. Instead, you can directly use `mysqladmin [command]`.
example:

- `mysqladmin status`
- `mysqladmin ping`
- `mysqladmin create database`
- `mysqladmin drop database`

using MySQL admin program

Before using `mysqladmin` to create a database, it's a good idea to first explore all available options with the `-help` command to familiarize yourself with `mysqladmin` functionalities.

`mysqladmin --help`

```
                      Also read groups with concat(group, suffix)
--login-path=#        Read this path from the login file.

Where command is a one or more of: (Commands may be shortened)
  create databasename   Create a new database
  debug                 Instruct server to write debug information to log
  drop databasename     Delete a database and all its tables
  extended-status       Gives an extended status message from the server
  flush-hosts           Flush all cached hosts
  flush-logs            Flush all logs
  flush-status          Clear status variables
  flush-tables          Flush all tables
  flush-threads         Flush the thread cache
  flush-privileges      Reload grant tables (same as reload)
  kill id,id,...        Kill mysql threads
  password [new-password] Change old password to new-password in current format
  ping                  Check if mysqld is alive
  processlist           Show list of active threads in server
  reload                Reload grant tables
  refresh               Flush all tables and close and open logfiles
  shutdown              Take server down
  status                Gives a short status message from the server
  start-replica         Start replication
  start-slave           Deprecated: use start-replica instead
  stop-replica          Stop replication
  stop-slave            Deprecated: use stop-replica instead
  variables             Prints variables available
  version               Get version info from server
[dba@mysqlPerconaSTG ~]$
```

for example we can get the current version of MySQL server

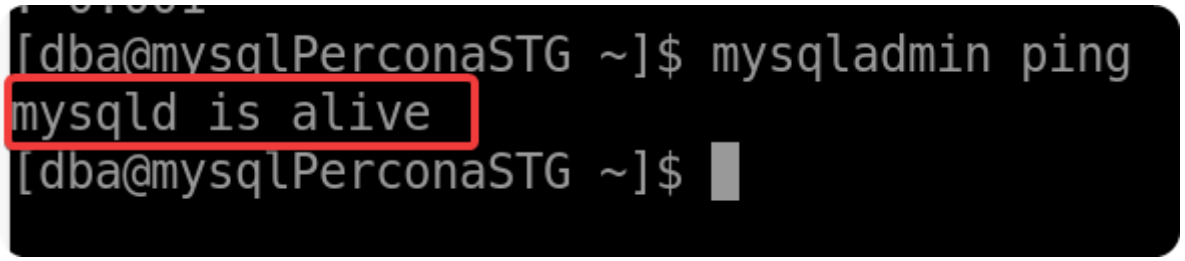
`mysqladmin version`

```
[dba@mysqlPerconaSTG ~]$ mysqladmin version
mysqladmin  Ver 8.0.36-28 for Linux on x86_64 (Percona Server (GPL), Release 28, Revision 47601f19)
Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Server version          8.0.36-28
Protocol version        10
Connection              Localhost via UNIX socket
UNIX socket             /var/lib/mysql/mysql.sock
Uptime:                 2 days 2 hours 7 min 53 sec

Threads: 2  Questions: 238  Slow queries: 0  Opens: 242  Flush tables: 3  Open tables: 161  Queries per second avg: 0.001
[dba@mysqlPerconaSTG ~]$
```

we can use `mysqladmin status` to check if MySQL server is up or not

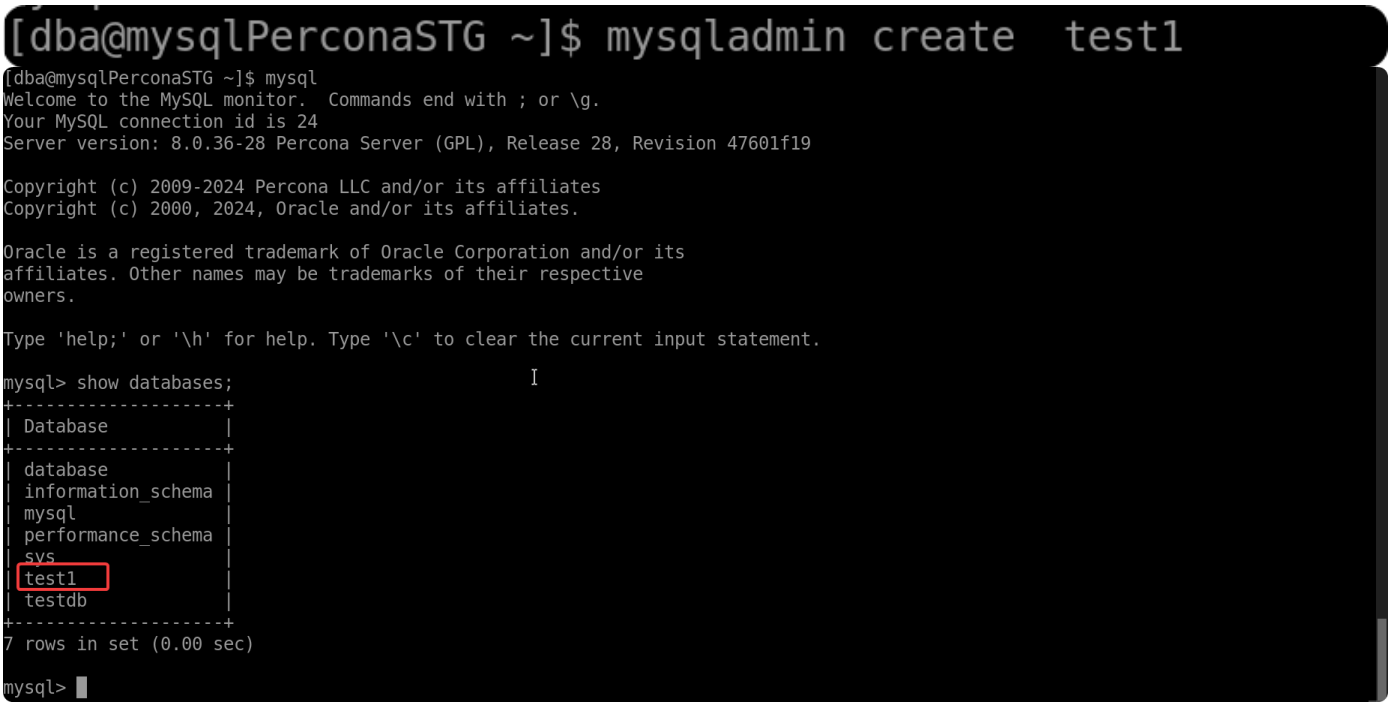
```
[dba@mysqlPerconaSTG ~]$ mysqladmin status
Uptime: 180566  Threads: 2  Questions: 240  Slow queries: 0  Opens: 242  Flush tables: 3  Open tables: 161  Queries per second avg
: 0.001
[dba@mysqlPerconaSTG ~]$
```

Wanna play ping-pong with your MySQL server to see if it's awake? Just serve a `mysqladmin ping` and see if it hits back. It's the quickest way to check if your MySQL server is up and ready to volley!



now we will use `mysqladmin` for our main task which is to create database

`mysqladmin create test1`



MYSQL Execute SQL Files

In this section, we'll cover how to execute SQL files on a MySQL server, a common task for MySQL DBAs. Typically, you might be given an SQL file that contains statements for creating databases, users, tables, etc. The key requirement is that the `.sql` file should contain only SQL statements executable in the MySQL shell. Here are four main methods for executing an `.sql` file:

1. **Source**: Inside the MySQL shell, use the `source` command or `\.` shortcut.
   Syntax: `mysql> \. file.sql` or `mysql> source file.sql`
2. **mysql Command**: Execute the file by using the mysql client program, specifying the database and credentials.

Syntax: `mysql --host=hostname --user=username --password=your_password database_name < file.sql`

3. **Shell Script**: Create an executable shell script that runs the SQL file.
   Example: `mysql --host=hostname database_name < $1`
4. **Pipe Method**: Use the `cat` command to display the contents of your SQL file, and then pipe it (`|`) directly into MySQL.
   Syntax:

```
cat file.sql | mysql
```

executing SQL Files in MYSQL

[reference link]

we will use employee.sql file and used `wget` to download it directly on the OS

`wget https://github.com/datacharmer/test_db/archive/refs/heads/master.zip`

```
[dba@mysqlPerconaSTG ~]$ wget https://github.com/datacharmer/test_db/archive/refs/heads/master.zip
--2024-04-09 01:09:00--  https://github.com/datacharmer/test_db/archive/refs/heads/master.zip
Resolving github.com (github.com)... 20.233.83.145
Connecting to github.com (github.com)|20.233.83.145|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://codeload.github.com/datacharmer/test_db/zip/refs/heads/master [following]
--2024-04-09 01:09:00--  https://codeload.github.com/datacharmer/test_db/zip/refs/heads/master
Resolving codeload.github.com (codeload.github.com)... 20.233.83.150
Connecting to codeload.github.com (codeload.github.com)|20.233.83.150|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/zip]
Saving to: 'master.zip'

master.zip                          [                    <=>                                ]  34.99M  4.95MB/s    in 9.2s

2024-04-09 01:09:10 (3.82 MB/s) - 'master.zip' saved [36688498]

[dba@mysqlPerconaSTG ~]$
```

```
[dba@mysqlPerconaSTG ~]$ ls
master.zip
```

next we will unzip the file

`unzip master.zip`

```
[dba@mysqlPerconaSTG ~]$ unzip master.zip
Archive:  master.zip
3c7fa05e04b4c339d91a43b7029a210212d48e6c
   creating: test_db-master/
  inflating: test_db-master/Changelog
  inflating: test_db-master/README.md
  inflating: test_db-master/employees.sql
  inflating: test_db-master/employees_partitioned.sql
  inflating: test_db-master/employees_partitioned_5.1.sql
   creating: test_db-master/images/
  inflating: test_db-master/images/employees.gif
  inflating: test_db-master/images/employees.jpg
  inflating: test_db-master/images/employees.png
  inflating: test_db-master/load_departments.dump
  inflating: test_db-master/load_dept_emp.dump
  inflating: test_db-master/load_dept_manager.dump
  inflating: test_db-master/load_employees.dump
  inflating: test_db-master/load_salaries1.dump
  inflating: test_db-master/load_salaries2.dump
  inflating: test_db-master/load_salaries3.dump
  inflating: test_db-master/load_titles.dump
  inflating: test_db-master/objects.sql
   creating: test_db-master/sakila/
  inflating: test_db-master/sakila/README.md
  inflating: test_db-master/sakila/sakila-mv-data.sql
  inflating: test_db-master/sakila/sakila-mv-schema.sql
```

```
[dba@mysqlPerconaSTG ~]$ ls
master.zip
test_db-master
[dba@mysqlPerconaSTG ~]$
```

inside test_db-master you should find a file called called `employees.sql` that we will use

```
[dba@mysqlPerconaSTG ~]$ cd  test_db-master/
[dba@mysqlPerconaSTG test_db-master]$ ls
Changelog                   load_departments.dump   load_salaries2.dump  sakila                 test_versions.sh
employees_partitioned_5.1.sql  load_dept_emp.dump      load_salaries3.dump  show_elapsed.sql
employees_partitioned.sql   load_dept_manager.dump  load_titles.dump     sql_test.sh
employees.sql               load_employees.dump     objects.sql          test_employees_md5.sql
images                      load_salaries1.dump     README.md            test_employees_sha.sql
```

if use `cat` to read content of the file you will find SQL statement for creating table and inserting value the file will also create database call employees

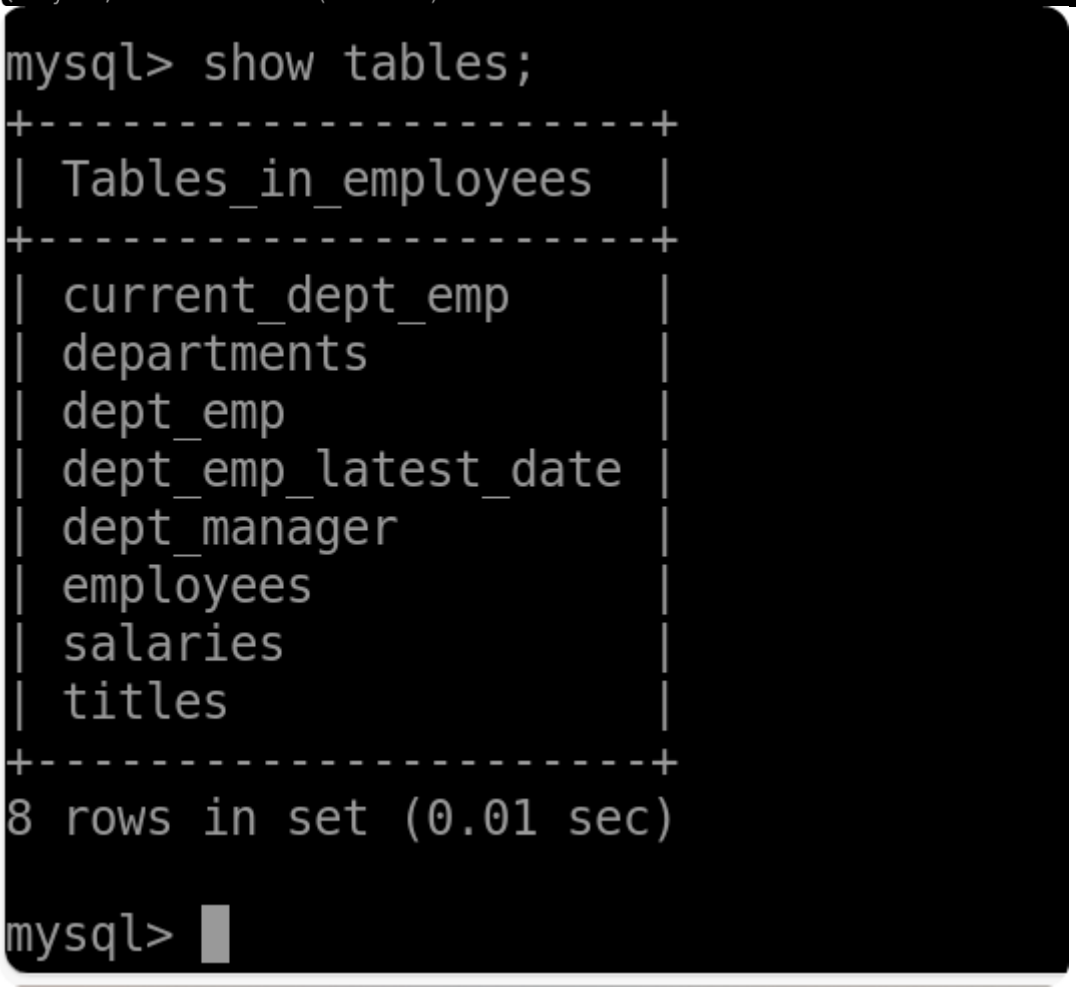
```
[dba@mysqlPerconaSTG test_db-master]$ cat employees.sql
-- Sample employee database
-- See changelog table for details
-- Copyright (C) 2007,2008, MySQL AB
--
-- Original data created by Fusheng Wang and Carlo Zaniolo
-- http://www.cs.aau.dk/TimeCenter/software.htm
-- http://www.cs.aau.dk/TimeCenter/Data/employeeTemporalDataSet.zip
--
-- Current schema by Giuseppe Maxia
-- Data conversion from XML to relational by Patrick Crews
--
-- This work is licensed under the
-- Creative Commons Attribution-Share Alike 3.0 Unported License.
-- To view a copy of this license, visit
-- http://creativecommons.org/licenses/by-sa/3.0/ or send a letter to
-- Creative Commons, 171 Second Street, Suite 300, San Francisco,
-- California, 94105, USA.
--
--  DISCLAIMER
--  To the best of our knowledge, this data is fabricated, and
```

```
                  salaries,
                  employees,
                  departments;

/*!50503 set default_storage_engine = InnoDB */;
/*!50503 select CONCAT('storage engine: ', @@default_storage_engine) as INFO */;

CREATE TABLE employees (
    emp_no      INT             NOT NULL,
    birth_date  DATE            NOT NULL,
    first_name  VARCHAR(14)     NOT NULL,
    last_name   VARCHAR(16)     NOT NULL,
    gender      ENUM ('M','F')  NOT NULL,
    hire_date   DATE            NOT NULL,
    PRIMARY KEY (emp_no)
);

CREATE TABLE departments (
    dept_no     CHAR(4)         NOT NULL,
    dept_name   VARCHAR(40)     NOT NULL,
    PRIMARY KEY (dept_no),
    UNIQUE  KEY (dept_name)
);

CREATE TABLE dept_manager (
   emp_no       INT             NOT NULL,
   dept_no      CHAR(4)         NOT NULL,
   from_date    DATE            NOT NULL,
   to_date      DATE            NOT NULL,
```

execute using source

To execute the file, first log into MySQL. Switch to the employees database by running `use employees`. Next, execute the employees.sql file with the command: `source employees.sql`.

```
[dba@mysqlPerconaSTG test_db-master]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 26
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use employees
Database changed
mysql> source employees.sql
Query OK, 0 rows affected (0.01 sec)

Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
```



execute using mysql Command

For our second method, we'll leverage the MySQL client program. This approach involves specifying the host and directing the contents of the SQL file into the MySQL server for a given database. Here's how you do it:

Use the command `mysql --host=localhost employees < employees.sql`. This instructs the MySQL client to connect to the MySQL server on `localhost`, target the `employees` database, and

execute the SQL statements contained within the `employees.sql` file. This method is efficient for applying a series of SQL commands stored in a file directly to your database.

`mysql --host=localhost employees < employees.sql`

```
[dba@mysqlPerconaSTG test_db-master]$ mysql --host=localhost employees < employees.sql
INFO
CREATING DATABASE STRUCTURE
INFO
storage engine: InnoDB
INFO
LOADING departments
INFO
LOADING employees
```

```
mysql> show tables from employees ;
+----------------------+
| Tables_in_employees  |
+----------------------+
| current_dept_emp     |
| departments          |
| dept_emp             |
| dept_emp_latest_date |
| dept_manager         |
| employees            |
| salaries             |
| titles               |
+----------------------+
8 rows in set (0.01 sec)

mysql>
```

execute using SHELL SCRIPT

For our third method, we'll craft a shell script, a handy approach for those who frequently execute `.sql` files, as it streamlines the process. Here's the step-by-step guide:

1. **Create a Shell Script**: Begin by making a new `.sh` file named `employees.sh`. This file will contain all the necessary commands to execute your `.sql` file.
2. **Script Content**: Open your script file and insert the following command:

   ```
   mysql --host=localhost employees < $1
   ```

   This line tells the script to run the `mysql` command, connect to the local MySQL server, select the `employees` database, and execute the SQL commands from the file specified as the first argument to the script.
3. **Make it Executable**: To allow your script to run, you need to modify its permissions to make it executable. Use the command:

   ```
   chmod u+x employees.sh
   ```

4. **Executing the Script**: Finally, to run your script along with the `.sql` file, use:

   ```
   bash employees.sh employees.sql
   ```

full command

```
vi employees.sh
msyql --host=localhost employees < $1
chmod u+x employees.sh
bash employees.sh employees.sql
```

```
[dba@mysqlPerconaSTG test_db-master]$ vi employees.sh
```

```
mysql --host=localhost employees < $1
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"employees.sh" [New] 1L, 38C written                                    1,37          All
```

```
[dba@mysqlPerconaSTG test_db-master]$ chmod u+x employees.sh
[dba@mysqlPerconaSTG test_db-master]$ ll
total 168344
-rw-rw-r-- 1 dba dba      964 Aug 26  2023 Changelog
-rw-rw-r-- 1 dba dba     7948 Aug 26  2023 employees_partitioned_5.1.sql
-rw-rw-r-- 1 dba dba     6276 Aug 26  2023 employees_partitioned.sql
-rwxrw-r-- 1 dba dba       38 Apr  9 01:34 employees.sh
-rw-rw-r-- 1 dba dba     4193 Aug 26  2023 employees.sql
drwxrwxr-x 2 dba dba       69 Aug 26  2023 images
-rw-rw-r-- 1 dba dba      250 Aug 26  2023 load_departments.dump
-rw-rw-r-- 1 dba dba 14159880 Aug 26  2023 load_dept_emp.dump
-rw-rw-r-- 1 dba dba     1090 Aug 26  2023 load_dept_manager.dump
-rw-rw-r-- 1 dba dba 17722832 Aug 26  2023 load_employees.dump
-rw-rw-r-- 1 dba dba 39806034 Aug 26  2023 load_salaries1.dump
-rw-rw-r-- 1 dba dba 39805981 Aug 26  2023 load_salaries2.dump
-rw-rw-r-- 1 dba dba 39080916 Aug 26  2023 load_salaries3.dump
-rw-rw-r-- 1 dba dba 21708736 Aug 26  2023 load_titles.dump
-rw-rw-r-- 1 dba dba     4568 Aug 26  2023 objects.sql
-rw-rw-r-- 1 dba dba     4325 Aug 26  2023 README.md
drwxrwxr-x 2 dba dba       77 Aug 26  2023 sakila
-rw-rw-r-- 1 dba dba      272 Aug 26  2023 show_elapsed.sql
-rwxr-xr-x 1 dba dba     1800 Aug 26  2023 sql_test.sh
-rw-rw-r-- 1 dba dba     4711 Aug 26  2023 test_employees_md5.sql
-rw-rw-r-- 1 dba dba     4715 Aug 26  2023 test_employees_sha.sql
-rwxr-xr-x 1 dba dba     2013 Aug 26  2023 test_versions.sh
[dba@mysqlPerconaSTG test_db-master]$
```

```
[dba@mysqlPerconaSTG test_db-master]$ bash employees.sh employees.sql
INFO
CREATING DATABASE STRUCTURE
INFO
storage engine: InnoDB
INFO
LOADING departments
INFO
LOADING employees
```

execute using pipe method

The fourth method is the simplest of all. Simply use `cat` to display the contents of your SQL file, and then pipe it (`|`) into MySQL for direct execution. This straightforward approach requires just a single line:

```
cat file.sql | mysql
```

This command concatenates the file's content and directs it straight into MySQL, allowing the SQL statements within the file to be executed seamlessly.

`cat employees.sql | mysql`

```
[dba@mysqlPerconaSTG test_db-master]$ cat  employees.sql | mysql
INFO
CREATING DATABASE STRUCTURE
INFO
storage engine: InnoDB
INFO
LOADING departments
INFO
LOADING employees
```

Executing SQL Commands From Terminal

This section will cover executing SQL commands directly from the MySQL client program without logging into the MySQL shell, utilizing the `-e` option.

example : `mysql -e ''select @@hostname , @@version ''`

```
[dba@mysqlPerconaSTG test_db-master]$ mysql -e 'select @@version'
+-----------+
| @@version |
+-----------+
| 8.0.36-28 |
+-----------+
[dba@mysqlPerconaSTG test_db-master]$
```

Importing data with mysqlimport

`mysqlimport` is a utility for importing data from text files into a MySQL table. The syntax allows specifying multiple text files as input, facilitating the loading of data directly into the database.
Here's how the syntax looks:

```
mysqlimport [options] database file1.txt [file2.tx]...
```

note : the txt file name must be the same as table name that we want to import data to

using mysqlimport

reference link

i have this file called staff,txt , i will use `cat` to view the content of the file it has four rows

```
[dba@mysqlPerconaSTG home]$ cd  share/
[dba@mysqlPerconaSTG share]$ ls
staff.txt
[dba@mysqlPerconaSTG share]$ cat staff.txt
John    Doe     Manager Y
Johny   Doe     DBA     Y
Tom     Doe     Analyst Y
Tommy   Doe     Tester  Y
[dba@mysqlPerconaSTG share]$
```

lets see if the staff table is there and if there are any existing rows

```
mysql> show tables;
+----------------------+
| Tables_in_employees  |
+----------------------+
| current_dept_emp     |
| departments          |
| dept_emp             |
| dept_emp_latest_date |
| dept_manager         |
| employees            |
| salaries             |
| staff                |
| titles               |
+----------------------+
9 rows in set (0.01 sec)

mysql> select * from staff ;
Empty set (0.01 sec)

mysql>
```

now lets' import the data in using mysqlimport

```
mysqlimport employees staff.txt
```

```
[dba@mysqlPerconaSTG share]$ mysqlimport employees staff.txt
mysqlimport: Error: 1290, The MySQL server is running with the --secure-file-priv option so it cannot execute this statement when
 using table: staff
[dba@mysqlPerconaSTG share]$
```

The `mysqlimport` command encountered an issue due to the `--secure-file-priv` variable, which restricts where you can load files from. To find out the directory this variable points to, log into MySQL and query the variable using the `LIKE` operator:

```
SHOW VARIABLES LIKE 'secure_file_priv';
```

```
mysql> show variables like 'secure_file_priv';
+------------------+-----------------------+
| Variable_name    | Value                 |
+------------------+-----------------------+
| secure_file_priv | /var/lib/mysql-files/ |
+------------------+-----------------------+
1 row in set (0.01 sec)

mysql>
```

The `secure_file_priv` variable specifies the directory from which files can be loaded. To comply, you'll need to move your `staff.txt` file into the designated path defined by `secure_file_priv`. You can do this by using the `mv` command in your operating system's shell. Alternatively, you can execute the `mv` command directly from within MySQL by prefixing it with `\!`. This allows you to run system commands from the MySQL prompt.

```
mysql> show variables like 'secure_file_priv';
+------------------+-----------------------+
| Variable_name    | Value                 |
+------------------+-----------------------+
| secure_file_priv | /var/lib/mysql-files/ |
+------------------+-----------------------+
1 row in set (0.01 sec)

mysql> \! mv staff.txt /var/lib/mysql-files/
mv: cannot stat '/var/lib/mysql-files/staff.txt': Permission denied
mysql> \! mv sudo staff.txt /var/lib/mysql-files/
mv: cannot stat 'sudo': No such file or directory
mv: cannot stat '/var/lib/mysql-files/staff.txt': Permission denied
mysql> \!sudo  mv  staff.txt /var/lib/mysql-files/
mv: cannot stat '/var/lib/mysql-files/staff.txt': Permission denied
mysql> \! sudo  mv  staff.txt /var/lib/mysql-files/
[sudo] password for dba:
mysql>
```

```
mysql> \! sudo  ls /var/lib/mysql-files/
staff.txt
mysql>
```

no rerun the command by putting the new path of staff.txt file

```
mysqlimport employees /var/lib/mysql-files/staff.txt
```

```
[dba@mysqlPerconaSTG /]$ mysqlimport employees /var/lib/mysql-files/staff.txt
mysqlimport: Error: 1366, Incorrect integer value: 'John' for column 'id' at row 1, when using table: staff
[dba@mysqlPerconaSTG /]$
```

The issue stems from the structure of the `staff` table, which doesn't permit null values for the `id` column, and it exclusively accepts integers. Running `DESC staff;` in MySQL would reveal that the `id` column is set to disallow nulls and is configured to accept only integer values, aligning with its role as a primary key that auto-increments. This setup ensures data integrity by preventing null or non-integer values from being inserted into the `id` column, which could otherwise lead to database inconsistencies.

```
mysql> desc staff;
+----------+--------------+------+-----+---------+----------------+
| Field    | Type         | Null | Key | Default | Extra          |
+----------+--------------+------+-----+---------+----------------+
| id       | int          | NO   | PRI | NULL    | auto_increment |
| fname    | varchar(255) | NO   |     | NULL    |                |
| lname    | varchar(255) | NO   |     | NULL    |                |
| title    | varchar(255) | NO   |     | NULL    |                |
| isActive | char(1)      | NO   |     | y       |                |
+----------+--------------+------+-----+---------+----------------+
5 rows in set (0.01 sec)
[dba@mysqlPerconaSTG /]$ mysqlimport employees /var/lib/mysql-files/staff.txt
mysqlimport: Error: 1366, Incorrect integer value: 'John' for column 'id' at row 1, when using table: staff
[dba@mysqlPerconaSTG /]$
```

The `staff.txt` file attempts to insert 'john' into the `id` column, which only accepts integer values. To resolve this, you need to skip the first field during the import process. Check the `mysqlimport` documentation or use `mysqlimport --help` to see if there's an option to specify or exclude specific column names during import.

```
--login-path=#          Read this path from the login file.
  --bind-address=name IP address to bind to.
  --character-sets-dir=name
                      Directory for character set files.
  --default-character-set=name
                      Set the default character set.
  -c, --columns=name  Use only these columns to import the data to. Give the
                      column names in a comma separated list. This is same as
                      giving columns to LOAD DATA INFILE.
  -C, --compress      Use compression in server/client protocol.
  -#, --debug[=#]     This is a non-debug version. Catch this and exit.
  --debug-check       This is a non-debug version. Catch this and exit.
  --debug-info        This is a non-debug version. Catch this and exit.
  --default-auth=name Default authentication client-side plugin to use.
  -d, --delete        First delete all rows from table.
  --enable-cleartext-plugin
                      Enable/disable the clear text authentication plugin.
  --fields-terminated-by=name
                      Fields in the input file are terminated by the given
                      string.
  --fields-enclosed-by=name
                      Fields in the import file are enclosed by the given
                      character.
  --fields-optionally-enclosed-by=name
                      Fields in the input file are optionally enclosed by the
                      given character.
  --fields-escaped-by=name
                      Fields in the input file are escaped by the given
                      character.
```

`-c, --columns=name` allow us to specify column we want to import data to.

`mysqlimport --columns=fname,lname,title,isActive employees /var/lib/mysql-files/staff.txt`

```
[dba@mysqlPerconaSTG /]$ mysqlimport  --columns=fname,lname,title,isActive employees /var/lib/mysql-files/staff.txt
employees.staff: Records: 4  Deleted: 0  Skipped: 0  Warnings: 0
[dba@mysqlPerconaSTG /]$
```

```
mysql> select * from staff;
+----+-------+-------+---------+----------+
| id | fname | lname | title   | isActive |
+----+-------+-------+---------+----------+
|  1 | John  | Doe   | Manager | Y        |
|  2 | Johny | Doe   | DBA     | Y        |
|  3 | Tom   | Doe   | Analyst | Y        |
|  4 | Tommy | Doe   | Tester  | Y        |
+----+-------+-------+---------+----------+
4 rows in set (0.00 sec)

mysql>
```

Maintaining Integrity with `mysqlcheck`

After importing data into a table and executing `.sql` files against a database, how can we ensure the integrity of the data? That's where the `mysqlcheck` utility becomes crucial.

- `mysqlcheck` serves as a table maintenance tool, offering functionalities to check, repair, optimize, and analyze tables.
- It examines tables for errors and attempts to fix any issues encountered, requiring the name of the table as input.
- **Important note:** Tables undergoing a `mysqlcheck` operation are locked, meaning no other database operations can be performed on them during this time. It's advisable to run `mysqlcheck` during maintenance windows to avoid disrupting database access.
  syntax:
  `mysqlcheck [options] db_name table_name`

how to use `mysqlcheck`

reference link

as mentioned before to use `mysqlcheck` as usual check the `--help` to see other options

`mysqlcheck --help`

to use `mysqlcheck` is straight forward

`mysqlcheck employees staff`

```
[dba@mysqlPerconaSTG /]$ mysqlcheck employees staff
employees.staff                                    OK
[dba@mysqlPerconaSTG /]$
```

Displaying useful Information with `mysqlshow`

As a DBA, you might often be asked for information like the number of tables in a specific database, the columns within a particular table, or the data types of those columns. In MySQL, there's a convenient utility called `mysqlshow` that can help you with these requests. This tool is designed to display details about databases, tables, and columns. You can use it to:

- Show information about databases, tables, and their columns.
- Accept both database and table names as inputs to fetch specific details.

The syntax to use `mysqlshow` is as follows:

To display information about a table in a database:

```
mysqlshow [options] db_name table_name
```

To get more detailed information about specific columns within a table:

```
mysqlshow [options] db_name table_name [column_name]
```

using `mysqlshow`

will will use `mysqlshow` how many tables in employees database and print out staff table information

first as always we will view the help option to see the deferent options

```
mysqlshow --help
```

first we will display all table in employees database

```
mysqlshow employees
```

```
[dba@mysqlPerconaSTG ~]$ mysqlshow employees
Database: employees
+----------------------+
|        Tables        |
+----------------------+
| current_dept_emp     |
| departments          |
| dept_emp             |
| dept_emp_latest_date |
| dept_manager         |
| employees            |
| salaries             |
| staff                |
| titles               |
+----------------------+
[dba@mysqlPerconaSTG ~]$
```

next we will display about column in staff table

```
mysqlshow employees staff
```

```
[dba@mysqlPerconaSTG ~]$ mysqlshow employees staff
Database: employees  Table: staff
+----------+-------------+--------------------+------+-----+---------+----------------+---------------------------------+--------
-+
| Field    | Type        | Collation          | Null | Key | Default | Extra          | Privileges                      | Comment
 |
+----------+-------------+--------------------+------+-----+---------+----------------+---------------------------------+--------
-+
| id       | int         |                    | NO   | PRI |         | auto_increment | select,insert,update,references |
 |
| fname    | varchar(255) | utf8mb4_0900_ai_ci | NO   |     |         |                | select,insert,update,references |
 |
| lname    | varchar(255) | utf8mb4_0900_ai_ci | NO   |     |         |                | select,insert,update,references |
 |
| title    | varchar(255) | utf8mb4_0900_ai_ci | NO   |     |         |                | select,insert,update,references |
 |
| isActive | char(1)      | utf8mb4_0900_ai_ci | NO   |     | y       |                | select,insert,update,references |
 |
+----------+-------------+--------------------+------+-----+---------+----------------+---------------------------------+--------
-+
[dba@mysqlPerconaSTG ~]$
```

finally will will display the column [id] information

```
mysqlshow employees staff id
```

```
[dba@mysqlPerconaSTG ~]$ mysqlshow employees staff id
Database: employees  Table: staff  Wildcard: id
+-------+------+-----------+------+-----+---------+----------------+---------------------------------+---------+
| Field | Type | Collation | Null | Key | Default | Extra          | Privileges                      | Comment |
+-------+------+-----------+------+-----+---------+----------------+---------------------------------+---------+
| id    | int  |           | NO   | PRI |         | auto_increment | select,insert,update,references |         |
+-------+------+-----------+------+-----+---------+----------------+---------------------------------+---------+
[dba@mysqlPerconaSTG ~]$
```

Time Zone Tables

In many applications that require handling time zones, it becomes necessary for the backend database, such as a MySQL server, to manage different time zone data efficiently.

MySQL includes a utility specifically for this purpose, named `mysql_tzinfo_to_sql`. This tool is designed to import time zone data from the zoneinfo database, which is a collection of files describing various time zones, typically found in `/usr/share/zoneinfo` on Linux systems, into the MySQL system database.

Time Zone Tables in MySQL:
The MySQL system database includes several tables that store time zone information:

1. `Time_zone`
2. `Time_zone_name`
3. `Time_zone_transition`
4. `Time_zone_transition_type`
5. `Time_zone_leap_second`

To load the time zone data into MySQL, use the following syntax:

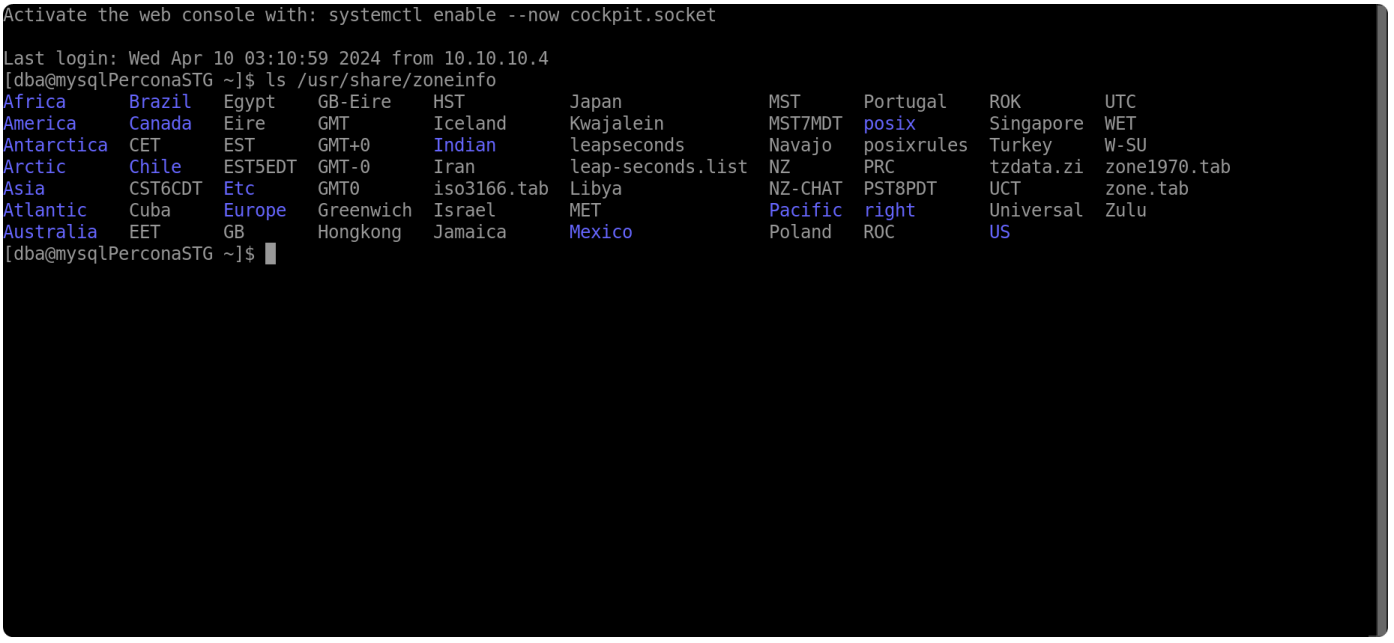
```
mysql_tzinfo_to_sql /usr/share/zoneinfo | mysql [options] db_name
```

This command reads the zoneinfo database and pipes its SQL representation directly into the MySQL database specified, allowing MySQL to handle time zone information more accurately for applications that rely on it.

using mysql_tzinfo_to_sql

Before we begin, it's crucial to verify the existence of the zoneinfo database directory on the operating system. Check if the path `/usr/share/zoneinfo` is present by using the command:

```
ls /usr/share/zoneinfo
```



With the confirmation that the path `/usr/share/zoneinfo` exists and contains the time zone data files,

you're all set to proceed with loading this data into your MySQL server. To do this, you'll use the `mysql_tzinfo_to_sql` utility. This command reads the time zone information from `/usr/share/zoneinfo` and pipes it into the MySQL system database. Here's how you run the utility:

```
mysql_tzinfo_to_sql /usr/share/zoneinfo | mysql -u root -p mysql
```

Remember to replace `-u root -p` with the appropriate username and password options for your MySQL server if necessary. This command will load the time zone data into the MySQL system database, ensuring your MySQL server can handle time zone conversions accurately.

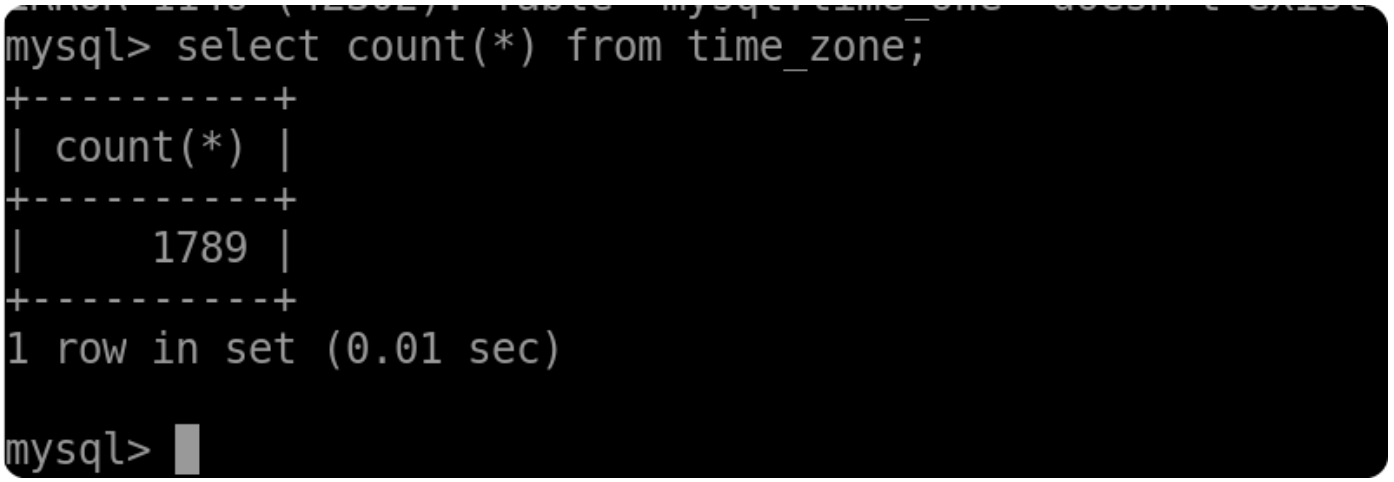
```
[dba@mysqlPerconaSTG ~]$ mysql_tzinfo_to_sql /usr/share/zoneinfo | mysql mysql
Warning: Unable to load '/usr/share/zoneinfo/iso3166.tab' as time zone. Skipping it.
Warning: Unable to load '/usr/share/zoneinfo/leap-seconds.list' as time zone. Skipping it.
Warning: Unable to load '/usr/share/zoneinfo/leapseconds' as time zone. Skipping it.
Warning: Unable to load '/usr/share/zoneinfo/tzdata.zi' as time zone. Skipping it.
Warning: Unable to load '/usr/share/zoneinfo/zone.tab' as time zone. Skipping it.
Warning: Unable to load '/usr/share/zoneinfo/zone1970.tab' as time zone. Skipping it.
[dba@mysqlPerconaSTG ~]$
```

the command will skip some files `.tab` and `.zi` these are unnecessary in this case
the command didn't output any error
let's go login to MySQL and change to MySQL database and see if the data is loaded on the table we discussed before .

```
mysql> show tables like 'time%';
+----------------------------+
| Tables_in_mysql (time%)    |
+----------------------------+
| time_zone                  |
| time_zone_leap_second      |
| time_zone_name             |
| time_zone_transition       |
| time_zone_transition_type  |
+----------------------------+
5 rows in set (0.00 sec)

mysql>
```

let's see if one of table got data populated

```
select count(*) from time_zone;
```



Listing Binary Logs Events with mysqlbinlog

The `mysqlbinlog` utility is designed for reading binary log files in MySQL. Here's a closer look:

- The binary log file is where any database changes are logged by the server.
- These changes are recorded as events within the binary log, and the log itself is written in a binary format hence their written in encrypted form we cannot display them .
- To convert and view these logs in a readable, plain text format, the `mysqlbinlog` utility comes into play.

Syntax examples include:

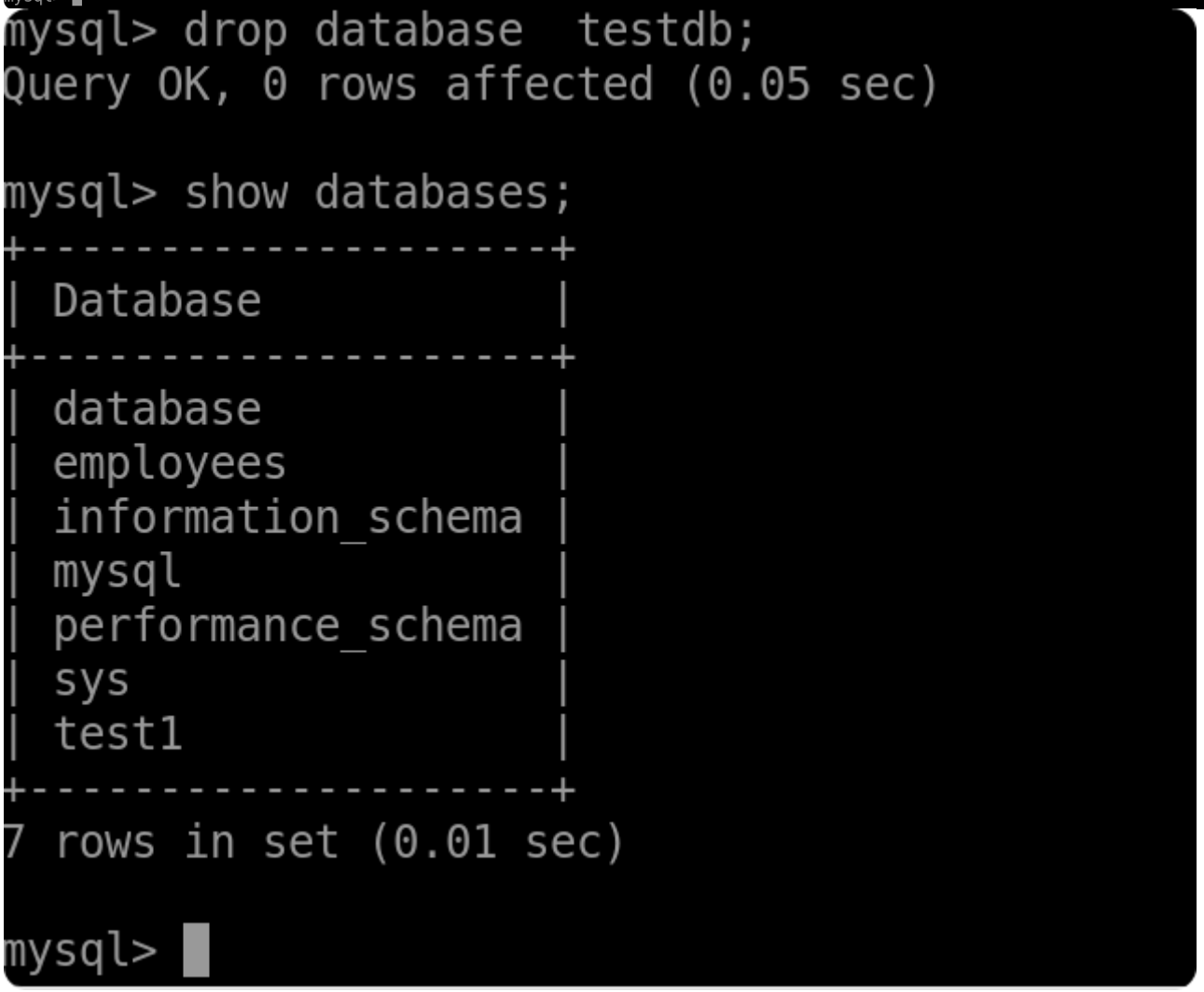
- To list binary logs: `SHOW BINARY LOGS;`
- To show events from a specific binary log file: `SHOW BINLOG EVENTS IN 'binary_log_file';`
- And to read a binary log file with `mysqlbinlog`:

```
mysqlbinlog [options] binary_log_file
```

using `mysqlbinlog` and reading the bin-log

in this senario we will drop a database in check if this cause event in binary log file

```
+----------+
| count(*) |
+----------+
|     1789 |
+----------+
1 row in set (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| database           |
| employees          |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test1              |
| testdb             |
+--------------------+
8 rows in set (0.01 sec)

mysql> drop testdb;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the r
ight syntax to use near 'testdb' at line 1
mysql> drop database  testdb;
Query OK, 0 rows affected (0.05 sec)

mysql>
```



let's first display how many binary log files we have in the system

`SHOW BINARY LOGS;`

```
mysql> SHOW BINARY LOGS;
+----------+-----------+-----------+
| Log_name | File_size | Encrypted |
+----------+-----------+-----------+
| 1.000001 |       196 | No        |
| 1.000002 |       387 | No        |
| 1.000003 |       180 | No        |
| 1.000004 |       180 | No        |
| 1.000005 |      5292 | No        |
| 1.000006 |  66383539 | No        |
| 1.000007 |  66383539 | No        |
| 1.000008 |  66383540 | No        |
| 1.000009 |  69419740 | No        |
+----------+-----------+-----------+
9 rows in set (0.01 sec)

mysql>
```

Given that there are 9 binary logs, with the assumption that the most recent activities, such as a DROP statement, are in the last log, we'll use `mysqlbinlog` to inspect this. Specifically, we'll target the binary log file named `1.000009`. The command will look something like this:

```
sudo mysqlbinlog /mysqldata/mysql/1.000009 > events.log
```

Here, `> events.log` directs the output into a file named `events.log`, which allows for easy reading of the contents.

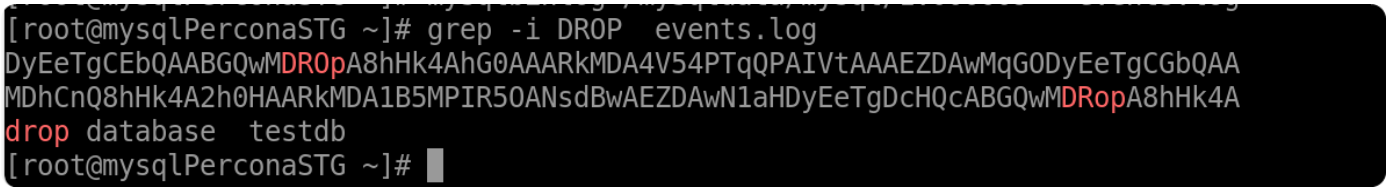
It's important to note that in this instance, the MySQL data directory is located at `/mysqldata/mysql/`, diverging from the default location, which is typically `/var/lib/mysql/`. This customization means the binary log files are stored in the specified, non-standard directory.

```
mysql> select @@datadir
    -> ;
+------------------+
| @@datadir        |
+------------------+
| /mysqldata/mysql/ |
+------------------+
1 row in set (0.00 sec)

mysql> exit
Bye
[dba@mysqlPerconaSTG mysql]$ cd /mysqldata/mysql/
[dba@mysqlPerconaSTG mysql]$ ls
 1.000001   1.000006   auto.cnf        ca.pem           '#ib_16384_0.dblwr'  '#innodb_redo'        private_key.pem   test1
 1.000002   1.000007   binlog.000001   client-cert.pem  '#ib_16384_1.dblwr'  '#innodb_temp'        public_key.pem    undo_001
 1.000003   1.000008   binlog.000002   client-key.pem    ib_buffer_pool       mysql                 server-cert.pem   undo_002
 1.000004   1.000009   binlog.index    database          ibdata1             mysql.ibd             server-key.pem
 1.000005   1.index    ca-key.pem      employees         ibtmp1              performance_schema    sys
[dba@mysqlPerconaSTG mysql]$
```

```
[root@mysqlPerconaSTG ~]# mysqlbinlog /mysqldata/mysql/1.000009 > events
[root@mysqlPerconaSTG ~]#
```

To explore the contents of the `events.log` file, you can use the `cat` command. However, if you're specifically looking for actions such as a DROP statement within the file, employing the `grep` command with the `-i` option for case-insensitive search is more efficient. Here's how you can filter the output to find occurrences of "DROP":

```
grep -i DROP events.log
```



4- MySQL Storage Engines

Storage Engines

MySQL supports a variety of table types to cater to different needs:

- Some tables participate in transactions, while others do not.
- Certain tables are created temporarily.
- Some tables exist in memory for quick data retrieval.
- Others might be stored as CSV files within the MySQL server.

To manage these diverse table types, MySQL uses different components known as storage engines. These engines are responsible for handling SQL operations for the tables.

The storage engines include:

1. Federated
2. InnoDB
3. MyISAM
4. Archive
5. Blackhole
6. CSV
7. MEMORY
8. PERFORMANCE_SCHEMA

MySQL employs a Pluggable Storage Engine architecture, which allows for the dynamic loading and unloading of engines as plugins on a running MySQL server.

- To see which storage engines are installed and which one is set as the default, use the command: `SHOW ENGINES;`.
- Storage engines are installed as plugins and reside in a shared library location. This location can be found using the system variable `plugin_dir`, which points to where all the shared library `.so` files are located.
- It's possible to install and uninstall additional storage engines:
  - To install a storage engine, the syntax is `INSTALL PLUGIN engine SONAME 'engine.so';`. The `engine.so` file must be located in the `plugin_dir` directory.
  - To uninstall a storage engine, the syntax is `UNINSTALL PLUGIN engine;`.

This flexibility allows MySQL to adapt to various data storage and access requirements by selecting the most suitable storage engine for each table.

Exploring Storage Engines

lets first check the storage engine we have in MySQL server by running command `show engines`

```
mysql> show engines ;
+--------------------+---------+----------------------------------------------------------------------------+--------------+------
+------------+
| Engine             | Support | Comment                                                                    | Transactions | XA
| Savepoints |
+--------------------+---------+----------------------------------------------------------------------------+--------------+------
+------------+
| ndbcluster         | NO      | Clustered, fault-tolerant tables                                           | NULL         | NULL
| NULL       |
| FEDERATED          | NO      | Federated MySQL storage engine                                             | NULL         | NULL
| NULL       |
| PERFORMANCE_SCHEMA | YES     | Performance Schema                                                         | NO           | NO
| NO         |
| InnoDB             | DEFAULT | Percona-XtraDB, Supports transactions, row-level locking, and foreign keys | YES          | YES
| YES        |
| MEMORY             | YES     | Hash based, stored in memory, useful for temporary tables                  | NO           | NO
| NO         |
| MyISAM             | YES     | MyISAM storage engine                                                      | NO           | NO
| NO         |
| ndbinfo            | NO      | MySQL Cluster system information storage engine                            | NULL         | NULL
| NULL       |
| MRG_MYISAM         | YES     | Collection of identical MyISAM tables                                      | NO           | NO
| NO         |
| BLACKHOLE          | YES     | /dev/null storage engine (anything you write to it disappears)             | NO           | NO
| NO         |
| CSV                | YES     | CSV storage engine                                                         | NO           | NO
| NO         |
| ARCHIVE            | YES     | Archive storage engine                                                     | NO           | NO
```

to view the output in vertical way we use `\G`

`show engines\G`

```
11 rows in set (0.00 sec)

mysql> show engines\G
*************************** 1. row ***************************
      Engine: ndbcluster
     Support: NO
     Comment: Clustered, fault-tolerant tables
Transactions: NULL
          XA: NULL
  Savepoints: NULL
*************************** 2. row ***************************
      Engine: FEDERATED
     Support: NO
     Comment: Federated MySQL storage engine
Transactions: NULL
          XA: NULL
  Savepoints: NULL
*************************** 3. row ***************************
      Engine: PERFORMANCE_SCHEMA
     Support: YES
     Comment: Performance Schema
Transactions: NO
          XA: NO
  Savepoints: NO
*************************** 4. row ***************************
      Engine: InnoDB
     Support: DEFAULT
     Comment: Percona-XtraDB, Supports transactions, row-level locking, and foreign keys
Transactions: YES
```

in the result you will see Support either yes our no , this refers to if the engines is enabled our disabled by default

```
| CSV                | YES     | CSV storage engine                                                                                  | NO           | NO
| NO         |
| ARCHIVE            | YES     | Archive storage engine                                                                              | NO           | NO
| NO         |
+--------------------+---------+-----------------------------------------------------------------------------------------------------+--------------+------
+------------+
11 rows in set (0.00 sec)

mysql> show engines\G
*************************** 1. row ***************************
      Engine: ndbcluster
     Support: NO
     Comment: Clustered, fault-tolerant tables
Transactions: NULL
          XA: NULL
  Savepoints: NULL
*************************** 2. row ***************************
      Engine: FEDERATED
     Support: NO
     Comment: Federated MySQL storage engine
Transactions: NULL
          XA: NULL
  Savepoints: NULL
*************************** 3. row ***************************
      Engine: PERFORMANCE_SCHEMA
     Support: YES
     Comment: Performance Schema
Transactions: NO
          XA: NO
```

some engines doesn't support some feature such as PERFORMANCE_SCHEMA doesnt support transactions , XA and savepoints

```
     Comment: Clustered, fault-tolerant tables
Transactions: NULL
          XA: NULL
  Savepoints: NULL
*************************** 2. row ***************************
      Engine: FEDERATED
     Support: NO
     Comment: Federated MySQL storage engine
Transactions: NULL
          XA: NULL
  Savepoints: NULL
*************************** 3. row ***************************
      Engine: PERFORMANCE_SCHEMA
     Support: YES
     Comment: Performance Schema
Transactions: NO
          XA: NO
  Savepoints: NO
*************************** 4. row ***************************
      Engine: InnoDB
     Support: DEFAULT
     Comment: Percona-XtraDB, Supports transactions, row-level locking, and foreign keys
Transactions: YES
          XA: YES
  Savepoints: YES
*************************** 5. row ***************************
      Engine: MEMORY
     Support: YES
     Comment: Hash based, stored in memory, useful for temporary tables
```

some engines will have support followed by DEFAULT , meaning this is the default storage engine such as InnoDB

```
    Support: NO
    Comment: Federated MySQL storage engine
Transactions: NULL
         XA: NULL
  Savepoints: NULL
*************************** 3. row ***************************
     Engine: PERFORMANCE_SCHEMA
    Support: YES
    Comment: Performance Schema
Transactions: NO
         XA: NO
  Savepoints: NO
*************************** 4. row ***************************
     Engine: InnoDB
    Support: DEFAULT
    Comment: Percona-XtraDB, Supports transactions, row-level locking, and foreign keys
Transactions: YES
         XA: YES
  Savepoints: YES
*************************** 5. row ***************************
     Engine: MEMORY
    Support: YES
    Comment: Hash based, stored in memory, useful for temporary tables
Transactions: NO
         XA: NO
  Savepoints: NO
*************************** 6. row ***************************
     Engine: MyISAM
    Support: YES
```

all of the storage engine have comment that show details about the storage engine

```
    Support: NO
    Comment: Federated MySQL storage engine
Transactions: NULL
         XA: NULL
  Savepoints: NULL
*************************** 3. row ***************************
     Engine: PERFORMANCE_SCHEMA
    Support: YES
    Comment: Performance Schema
Transactions: NO
         XA: NO
  Savepoints: NO
*************************** 4. row ***************************
     Engine: InnoDB
    Support: DEFAULT
    Comment: Percona-XtraDB, Supports transactions, row-level locking, and foreign keys
Transactions: YES
         XA: YES
  Savepoints: YES
*************************** 5. row ***************************
     Engine: MEMORY
    Support: YES
    Comment: Hash based, stored in memory, useful for temporary tables
Transactions: NO
         XA: NO
  Savepoints: NO
*************************** 6. row ***************************
     Engine: MyISAM
    Support: YES
```

last we will show have to view the shared library for plugin directory using the variable `plugin_dir`

`show variables like '%plugin%'`

```
mysql> show variables like '%plugin%'
    -> ;
+-------------------------------------------------+-------------------------+
| Variable_name                                   | Value                   |
+-------------------------------------------------+-------------------------+
| default_authentication_plugin                   | caching_sha2_password   |
| plugin_dir                                      | /usr/lib64/mysql/plugin/ |
| replication_optimize_for_static_plugin_config | OFF                     |
+-------------------------------------------------+-------------------------+
3 rows in set (0.01 sec)

mysql>
```

```
+-------------------------------------------+------------------------+
3 rows in set (0.01 sec)

mysql> Bye
[dba@mysqlPerconaSTG /]$ ls /usr/lib64/mysql/plugin/
adt_null.so                        component_mysqlbackup.so                    keyring_udf.so
audit_log_filter.so                component_query_attributes.so               keyring_vault.so
audit_log.so                       component_reference_cache.so                libfnv1a_udf.so
authentication_fido_client.so      component_test_audit_api_message.so         libfnv_udf.so
authentication_fido.so             component_test_component_deinit.so          libmemcached.so
authentication_kerberos_client.so  component_test_host_application_signal.so  libmurmur_udf.so
authentication_ldap_sasl_client.so component_test_mysql_system_variable_set.so libpluginmecab.so
authentication_ldap_sasl.so        component_test_mysql_thd_store_service.so   locking_service.so
authentication_ldap_simple.so      component_test_server_telemetry_traces.so   mypluglib.so
authentication_oci_client.so       component_test_table_access.so              mysql_clone.so
auth_pam_compat.so                 component_test_udf_services.so              mysql_no_login.so
auth_pam.so                        component_validate_password.so              procfs.so
auth_socket.so                     connection_control.so                       rewrite_example.so
binlog_utils_udf.so                data_masking.ini                            rewriter.so
component_audit_api_message_emit.so data_masking.so                            semisync_master.so
component_encryption_udf.so        ddl_rewriter.so                             semisync_replica.so
component_keyring_file.so          debug                                       semisync_slave.so
component_keyring_kmip.so          dialog.so                                   semisync_source.so
component_keyring_kms.so           group_replication.so                        test_services_host_application_signal.so
component_log_filter_dragnet.so    ha_example.so                               test_udf_wrappers.so
component_log_sink_json.so         ha_mock.so                                  validate_password.so
component_log_sink_syseventlog.so  innodb_engine.so                            version_token.so
component_masking_functions.so     keyring_file.so
[dba@mysqlPerconaSTG /]$
```

FEDERATED Storage Engine

The FEDERATED storage engine in MySQL is disabled by default. When you create a table using the FEDERATED Storage Engine, it essentially serves as a pointer to a table located on another MySQL instance, which could be on a different server. This functionality is akin to Microsoft's Linked Server or Oracle's Database Link, where:

- Both the local and remote tables must have identical names and definitions.
- The local table, which uses the FEDERATED engine, acts similarly to a view, referring to the remote table.
- The target (remote) table can utilize a different storage engine, but the local (requester) table must be specifically created with the FEDERATED engine.

Syntax for creating a FEDERATED table:

```
CREATE TABLE bank(
  id INT,
  salary INT
) ENGINE=FEDERATED
CONNECTION = 'mysql://db_user@targetserver:3306/database-name/bank';
```

Given that the FEDERATED engine is not enabled by default, its usage is relatively infrequent.

MEMORY Storage Engine

Previously known as **HEAP**, this storage engine is now recognized for its in-memory capabilities, making it exceptionally fast and particularly suited for scenarios where data needs to be rapidly accessed and stored directly in memory. However, its main limitation is the lack of data persistence; since the data isn't saved to permanent storage devices like SSDs or HDDs, any server crash or restart

of MySQL services will result in data loss. Consequently, it's advisable to employ this storage engine for specific purposes, such as temporary tables or caching read-only data.

Use Cases:

- **Static Tables**: Ideal for lookup tables containing static data that the application can easily regenerate in case of data loss.
- **Temporary Tables**: Beneficial for temporary data storage during session or process-specific operations.

Drawbacks:

- Lacks transaction support.
- Does not offer referential integrity support, meaning foreign key relationships cannot be defined.
- Incompatible with certain data types, including TEXT and BLOB columns, restricting its use with these data types.

using MEMORY Storage Engine

reference link
For this exercise, we'll create a table named 'continents', ensuring it utilizes the in-memory storage engine to facilitate fast data retrieval and manipulation. This approach is suitable for scenarios where persistence through server restarts is not required.

First, we'll define the 'continents' table and specify that it should use the MEMORY storage engine: we simply append `ENGINE=MEMORY`; to the table definition.

```
CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL)
ENGINE=MEMORY;
```

```
mysql> use test1
Database changed
mysql> CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL) ENGINE=MEMORY;
Query OK, 0 rows affected (0.08 sec)

mysql> show tables;
+----------------+
| Tables_in_test1 |
+----------------+
| continents      |
+----------------+
1 row in set (0.01 sec)

mysql>
```

Next, let's populate the table with data:

```
INSERT INTO continents(cid, cname)
VALUES
  (1, 'Asia'),
  (2, 'Africa'),
  (3, 'Europe'),
```

```
  (4, 'North America'),
  (5, 'South America'),
  (6, 'Australia'),
  (7, 'Antarctica');
```

To review the data we've just inserted, we'll use:

```
SELECT * FROM continents;
```

```
1 row in set (0.01 sec)

mysql> INSERT INTO continents(cid, cname)
    -> VALUES
    ->   (1, 'Asia'),
    ->   (2, 'Africa'),
    ->   (3, 'Europe'),
    ->   (4, 'North America'),
    ->   (5, 'South America'),
    ->   (6, 'Australia'),
    ->   (7, 'Antarctica');
Query OK, 7 rows affected (0.01 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM continents;
+-----+--------------+
| cid | cname        |
+-----+--------------+
|   1 | Asia         |
|   2 | Africa       |
|   3 | Europe       |
|   4 | North America |
|   5 | South America |
|   6 | Australia    |
|   7 | Antarctica   |
+-----+--------------+
7 rows in set (0.00 sec)

mysql>
```

Now, we'll fetch table information from the `INFORMATION_SCHEMA` to confirm the 'continents' table is indeed using the MEMORY engine:

```
SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM INFORMATION_SCHEMA.TABLES WHERE ENGINE='MEMORY';
```

```
mysql> SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM INFORMATION_SCHEMA.TABLES WHERE ENGINE='MEMORY';
+------------+------------+--------+------------+---------------------+
| TABLE_NAME | TABLE_TYPE | ENGINE | TABLE_ROWS | CREATE_TIME         |
+------------+------------+--------+------------+---------------------+
| continents | BASE TABLE | MEMORY |          7 | 2024-04-10 04:57:10 |
+------------+------------+--------+------------+---------------------+
1 row in set (0.02 sec)

mysql>
```

After confirming the table setup, we'll restart the MySQL server services to simulate a scenario that tests the non-persistence of the MEMORY storage engine:

```
sudo systemctl stop mysqld && sudo systemctl start mysqld
```

```
mysql> \! sudo systemctl stop mysqld && sudo systemctl start mysqld
[sudo] password for dba:
mysql>
```

Finally, we'll verify whether the data persists after the MySQL server restart, which, due to the nature of the MEMORY engine, it should not:

```
SELECT * FROM continents;
```

```
mysql> SELECT * FROM continents;
ERROR 2013 (HY000): Lost connection to MySQL server during query
No connection. Trying to reconnect...
Connection id:    8
Current database: test1

Empty set (0.06 sec)

mysql>
```

This process demonstrates the MEMORY engine's characteristics, especially its volatility and the consequent disappearance of data upon server restart.

BLACKHOLE Storage Engine

The Blackhole storage engine in MySQL functions exactly as its name suggests: like a black hole, where anything that enters disappears forever. Here's what it entails:

- You can insert as much data as you like into a table using the Blackhole storage engine.
- However, trying to retrieve data from such a table will always result in an empty response because the data effectively vanishes upon insertion.
- It does not support transactions.
- A practical application is in master-slave replication setups where you might have multiple slave nodes but only need one to actually store data. The Blackhole engine can serve as a placeholder on other nodes to absorb data without storing it.
- Another scenario is for performance testing, particularly when you want to eliminate storage as a potential bottleneck. Directing data to a Blackhole engine table can help isolate other areas of concern.

To use this engine, specify it during table creation like so:

```
CREATE TABLE tablename ([table column definition]) ENGINE=BLACKHOLE;
```

using BLACKHOLE Storage Engine

Let's proceed to create the 'continents' table using the BLACKHOLE Storage Engine, insert data into it, and then attempt to retrieve the data to see what happens:

First, we create the table with the BLACKHOLE engine:

```
CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL)
ENGINE=BLACKHOLE;
```

```
mysql> drop table continents   ;
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL) ENGINE=BLACKHOLE;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

Next, we'll insert data into the table:

```
INSERT INTO continents(cid, cname)
VALUES
  (1, 'Asia'),
  (2, 'Africa'),
  (3, 'Europe'),
  (4, 'North America'),
  (5, 'South America'),
  (6, 'Australia'),
  (7, 'Antarctica');
```

```
mysql> INSERT INTO continents(cid, cname)
    -> VALUES
    ->   (1, 'Asia'),
    ->   (2, 'Africa'),
    ->   (3, 'Europe'),
    ->   (4, 'North America'),
    ->   (5, 'South America'),
    ->   (6, 'Australia'),
    ->   (7, 'Antarctica');
Query OK, 7 rows affected (0.01 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql>
```

To confirm the storage engine used by our table, we'll query the performance schema:

```
SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM
INFORMATION_SCHEMA.TABLES WHERE ENGINE='BLACKHOLE';
```

```
mysql> SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM INFORMATION_SCHEMA.TABLES WHERE ENGINE='BLACKHOLE';
+------------+------------+-----------+------------+---------------------+
| TABLE_NAME | TABLE_TYPE | ENGINE    | TABLE_ROWS | CREATE_TIME         |
+------------+------------+-----------+------------+---------------------+
| continents | BASE TABLE | BLACKHOLE |          0 | 2024-04-10 05:17:47 |
+------------+------------+-----------+------------+---------------------+
1 row in set (0.02 sec)

mysql>
```

Finally, attempting to retrieve the data from the table will yield an empty result, as expected with the BLACKHOLE engine:

```
SELECT * FROM continents;
```

This exercise demonstrates the unique behaviour of the BLACKHOLE Storage Engine: despite accepting data insertions, it retains nothing, leading to empty query results.

CSV Storage Engine

MySQL Server is capable of storing tables in text files formatted as comma-separated values (CSV). This functionality leverages the CSV Storage Engine to create `.csv` files within the MySQL `$data_dir` directory as plain text. These CSV files can be directly read and written by spreadsheet applications like Excel. However, it's important to note that the CSV Storage Engine does not support transactions, and tables stored in this format are not indexed, meaning you cannot create indexes on fields within a CSV-based table.

Use Case:
This engine is particularly useful when there's a need to share data with other applications that also utilize the CSV format, enabling a seamless data interchange.

To create a table using the CSV Storage Engine, the syntax is as follows:

```
CREATE TABLE table_name ([table column definition]) ENGINE=CSV;
```

using CSV Storage Engine

we will create same 'continents' in CSV storage engine , and then we should found the CSV file under data_diretcory

First, we create the table with the CSV engine:

```
CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL)
ENGINE=CSV;
```

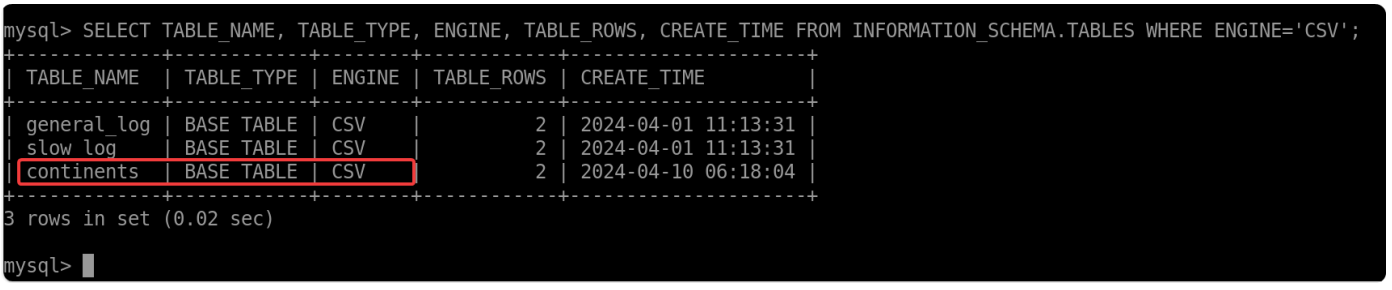
```
mysql> CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL) ENGINE=CSV;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

Next, we'll insert data into the table:

```
INSERT INTO continents(cid, cname)
VALUES
  (1, 'Asia'),
  (2, 'Africa'),
  (3, 'Europe'),
  (4, 'North America'),
  (5, 'South America'),
  (6, 'Australia'),
  (7, 'Antarctica');
```

To confirm the storage engine used by our table, we'll query the performance schema:

```
SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM
INFORMATION_SCHEMA.TABLES WHERE ENGINE='CSV';
```



now we will look for CSV file under the data directory of the MySQL

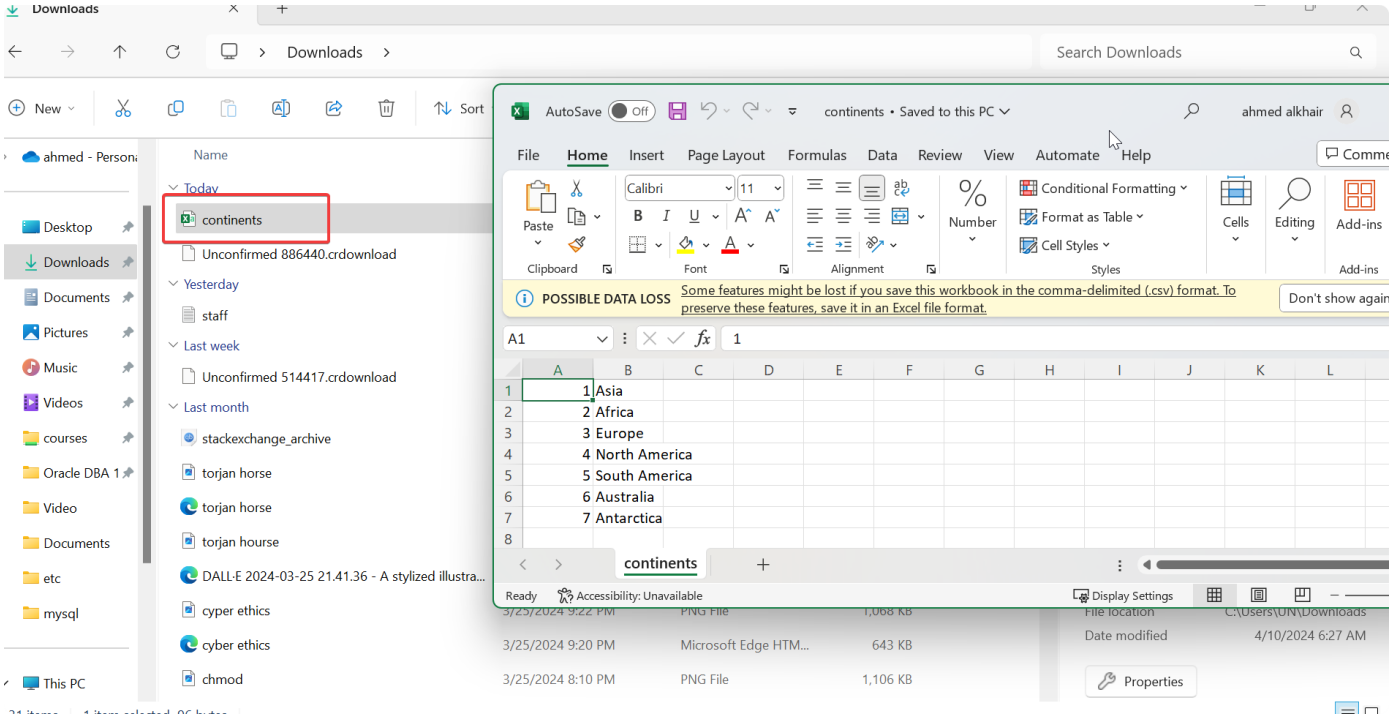
you can get the the data directory using the query `select @@datadir;`

```
mysql> select @@datadir;
+-------------------+
| @@datadir         |
+-------------------+
| /mysqldata/mysql/ |
+-------------------+
1 row in set (0.00 sec)

mysql>
```

the table is created under the database test1 , you should find directory called test1 , under it you should find csv file of the table

```
[dba@mysqlPerconaSTG /]$ cd  mysqldata/mysql/
[dba@mysqlPerconaSTG mysql]$ ls
 1.000001   1.000006   1.index         ca-key.pem        employees            ibtmp1          performance_schema   sys
 1.000002   1.000007   auto.cnf        ca.pem           '#ib_16384_0.dblwr'  '#innodb_redo'   private_key.pem      test1
 1.000003   1.000008   binlog.000001   client-cert.pem  '#ib_16384_1.dblwr'  '#innodb_temp'   public_key.pem       undo_001
 1.000004   1.000009   binlog.000002   client-key.pem    ib_buffer_pool       mysql           server-cert.pem      undo_002
 1.000005   1.000010   binlog.index    database          ibdata1              mysql.ibd       server-key.pem
[dba@mysqlPerconaSTG mysql]$
```

```
[dba@mysqlPerconaSTG mysql]$ sudo ls test1/
continents_417.sdi  continents.CSM  continents.CSV
[dba@mysqlPerconaSTG mysql]$
```



MyISAM Storage Engine

MyISAM, which stands for "My Indexed Sequential Access Method," is essentially a storage engine in MySQL. The name "My" comes from the country code of Malaysia, where the co-founder of MySQL originates from, combined with ISAM, an indexing algorithm developed by IBM. This algorithm is designed for efficient retrieval of data from large datasets.

Up until MySQL version 5.5, around 2009-2010, MyISAM was the default storage engine. It was later replaced by the InnoDB storage engine, which offers advantages such as transactional support, following the ACID model, and better speed for operations involving a mix of reading and writing.

MyISAM is particularly noted for its speed, making it suitable for data warehousing scenarios where there's a high volume of reads compared to writes. However, it's worth noting that InnoDB has closed the speed gap, even surpassing MyISAM in many cases.

Given its lack of support for transactional properties and the growing preference for InnoDB, MySQL plans to phase out MyISAM in future releases.

Use Case:
MyISAM is best suited for data warehousing environments characterized by a high number of read operations.

To create a table with MyISAM as the storage engine, use the syntax:

```
CREATE TABLE tablename ([table column definition]) ENGINE=MyISAM;
```

MyISAM Storage Engine

we will create same 'continents' in MyISAM storage engine , and then we will insert data

then we will start transaction , committed , and then attempt to Rollback which will show the drawback of MyISAM where doesnt support transaction
First, we create the table with the MyISAM engine:

```
CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL)
ENGINE=MyISAM;
```

```
mysql> CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL) ENGINE=MyISAM;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

Next, we'll insert data into the table:

```
INSERT INTO continents(cid, cname)
VALUES
  (1, 'Asia'),
  (2, 'Africa'),
  (3, 'Europe'),
  (4, 'North America'),
```

```
  (5, 'South America'),
  (6, 'Australia'),
  (7, 'Antarctica');
```

```
mysql> INSERT INTO continents(cid, cname)
    -> VALUES
    ->   (1, 'Asia'),
    ->   (2, 'Africa'),
    ->   (3, 'Europe'),
    ->   (4, 'North America'),
    ->   (5, 'South America'),
    ->   (6, 'Australia'),
    ->   (7, 'Antarctica');
Query OK, 7 rows affected (0.00 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql>
```

To confirm the storage engine used by our table, we'll query the performance schema:

```
SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM
INFORMATION_SCHEMA.TABLES WHERE ENGINE='MyISAM';
```

```
mysql> SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM INFORMATION_SCHEMA.TABLES WHERE ENGINE='MyISAM';
+------------+------------+--------+------------+---------------------+
| TABLE_NAME | TABLE_TYPE | ENGINE | TABLE_ROWS | CREATE_TIME         |
+------------+------------+--------+------------+---------------------+
| continents | BASE TABLE | MyISAM |          7 | 2024-04-10 06:42:55 |
+------------+------------+--------+------------+---------------------+
1 row in set (0.01 sec)

mysql>
```

now we will start transaction by type `START TRANSACTION;`

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

then we will update values of the table continents with the following

```
update continents set cname='ant' where cid=7;
```

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> update continents set cname='ant' where cid=7;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

now le's do rollback by running the command `ROLLBACK;`

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> update continents set cname='ant' where cid=7;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> ROLLBACK;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql>
```

we can see the output stated 0 row affected and 1 warning , if we show al content of the table we can see that table hasn't rollback to original value .

```
mysql> select * from continents ;
+-----+---------------+
| cid | cname         |
+-----+---------------+
|   1 | Asia          |
|   2 | Africa        |
|   3 | Europe        |
|   4 | North America |
|   5 | South America |
|   6 | Australia     |
|   7 | ant           |
+-----+---------------+
7 rows in set (0.00 sec)

mysql>
```

this show you the limitation of MyISAM compare to innoDB when it come to transaction

, Storage Engine

The ARCHIVE Storage Engine is tailored for specific use cases, where it excels at storing a massive volume of unindexed data in a compact format. This engine is an ideal choice for large tables that don't use indexes. Typically, a table is made up of table data and index data. The ARCHIVE engine, however, is designed to compress only those tables that lack index data.

When you use the ARCHIVE engine, it compresses the table into `.ARZ` files, which are named after the table itself. These `.ARZ` files are binary and not directly readable, serving as the storage format for archived data. The compression is achieved using the `gzip` utility, with each table row being compressed individually.

Limitations include:

- Inability to perform delete or update operations on the data.
- Lack of support for partitioning.

To create a table that utilizes the ARCHIVE engine, the syntax is as follows:

```
CREATE TABLE tablename ([table column definition]) ENGINE=ARCHIVE;
```

using ARCHIVE Storage Engine

we will create same 'continents' in ARCHIVE storage engine , and then we will insert data , then look for `.ARZ` files in the data_dir

First, we create the table with the ARCHIVE engine:

```
CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL)
ENGINE=ARCHIVE;
```

```
mysql> drop table continents ;
Query OK, 0 rows affected (0.03 sec)

mysql> CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL) ENGINE=ARCHIVE;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

Next, we'll insert data into the table:

```
INSERT INTO continents(cid, cname)
VALUES
  (1, 'Asia'),
  (2, 'Africa'),
  (3, 'Europe'),
  (4, 'North America'),
  (5, 'South America'),
  (6, 'Australia'),
  (7, 'Antarctica');
```

```
mysql> INSERT INTO continents(cid, cname)
    -> VALUES
    ->   (1, 'Asia'),
    ->   (2, 'Africa'),
    ->   (3, 'Europe'),
    ->   (4, 'North America'),
    ->   (5, 'South America'),
    ->   (6, 'Australia'),
    ->   (7, 'Antarctica');
Query OK, 7 rows affected (0.00 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql>
```

To confirm the storage engine used by our table, we'll query the performance schema:

```
SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM
INFORMATION_SCHEMA.TABLES WHERE ENGINE='ARCHIVE';
```

```
mysql> SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM INFORMATION_SCHEMA.TABLES WHERE ENGINE='ARCHIVE';
+------------+------------+---------+------------+---------------------+
| TABLE_NAME | TABLE_TYPE | ENGINE  | TABLE_ROWS | CREATE_TIME         |
+------------+------------+---------+------------+---------------------+
| continents | BASE TABLE | ARCHIVE |          7 | 2024-04-10 07:04:52 |
+------------+------------+---------+------------+---------------------+
1 row in set (0.02 sec)

mysql>
```

To locate the `.ARZ` files generated by the ARCHIVE Storage Engine, you'll first need to identify the data directory of your MySQL server. If you're unsure where this is, you can find out by running the query `SELECT @@datadir;`. Once you know the data directory, navigate to the specific database directory where your table is located. For instance, if your table was created in the `test1` database, there should be a `test1` directory within the data directory. Inside this `test1` directory, you'll find the `.ARZ` files corresponding to your ARCHIVE engine tables.

```
mysql> SELECT @@datadir;
+--------------------+
| @@datadir          |
+--------------------+
| /mysqldata/mysql/  |
+--------------------+
1 row in set (0.00 sec)
```

```
[root@mysqlPerconaSTG ~]# ls /mysqldata/mysql/test1/
continents_419.sdi  continents.ARZ
[root@mysqlPerconaSTG ~]#
```

InnoDB Storage Engine

The InnoDB Storage Engine stands out as a fully ACID-compliant storage solution in MySQL, ensuring reliable transaction processing with support for operations like commit and rollback. It offers consistency and robust crash recovery mechanisms, alongside various isolation levels to tailor transactional behaviour. Optimized to work closely with the underlying hardware, InnoDB aims to deliver top-notch performance, making it the default choice for MySQL databases. Known for its efficiency, especially in OLTP (Online Transaction Processing) scenarios, InnoDB is widely utilized across different sectors, including finance and aviation. It features row-level locking and advanced indexing capabilities to enhance concurrency and speed. Additionally, InnoDB maintains a buffer pool for caching table and index data, significantly speeding up data retrieval.

When creating tables in MySQL, the syntax to specify InnoDB explicitly is:

```
CREATE TABLE tablename ([table column definition]) ENGINE=InnoDB;
```

However, it's important to note that InnoDB is the default storage engine, so even if you don't specify an engine when creating a table, MySQL will automatically use InnoDB.

using InnoDB Storage Engine

We'll proceed to create the 'continents' table using the InnoDB storage engine. Then, we'll populate it with data and explore various operations including committing transactions, rolling back transactions, and creating an index.

To begin, we create the table. Remember, specifying the storage engine is optional when creating a table since MySQL defaults to using InnoDB:

```
CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL)
ENGINE=InnoDB;
```

```
mysql> CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL) ENGINE=InnoDB;
Query OK, 0 rows affected (0.03 sec)

mysql>
```

Next, we'll insert data into the table:

```
INSERT INTO continents(cid, cname)
VALUES
  (1, 'Asia'),
  (2, 'Africa'),
  (3, 'Europe'),
  (4, 'North America'),
  (5, 'South America'),
```

```
  (6, 'Australia'),
  (7, 'Antarctica');
```

```
mysql> INSERT INTO continents(cid, cname)
    -> VALUES
    ->   (1, 'Asia'),
    ->   (2, 'Africa'),
    ->   (3, 'Europe'),
    ->   (4, 'North America'),
    ->   (5, 'South America'),
    ->   (6, 'Australia'),
    ->   (7, 'Antarctica');
Query OK, 7 rows affected (0.00 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql>
```

To confirm the storage engine used by our table, we'll query the performance schema:

```
SELECT TABLE_NAME, TABLE_TYPE, ENGINE, TABLE_ROWS, CREATE_TIME FROM
INFORMATION_SCHEMA.TABLES WHERE ENGINE='InnoDB';
```

```
| time_zone_transition_type                          | BASE TABLE | InnoDB |      10533 | 2024-04-01 11:13:31 |
| time_zone_leap_second                              | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:31 |
| procs_priv                                         | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:31 |
| component                                          | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:31 |
| slave_relay_log_info                               | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:31 |
| slave_master_info                                  | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:31 |
| slave_worker_info                                  | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:31 |
| gtid_executed                                      | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:31 |
| replication_asynchronous_connection_failover       | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:31 |
| replication_asynchronous_connection_failover_managed | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:31 |
| replication_group_member_actions                   | BASE TABLE | InnoDB |          2 | 2024-04-01 11:13:31 |
| replication_group_configuration_version            | BASE TABLE | InnoDB |          1 | 2024-04-01 11:13:31 |
| server_cost                                        | BASE TABLE | InnoDB |          6 | 2024-04-01 11:13:31 |
| engine_cost                                        | BASE TABLE | InnoDB |          2 | 2024-04-01 11:13:31 |
| proxies_priv                                       | BASE TABLE | InnoDB |          1 | 2024-04-01 11:13:32 |
| ndb_binlog_index                                   | BASE TABLE | InnoDB |          0 | 2024-04-01 11:13:32 |
| sys_config                                         | BASE TABLE | InnoDB |          6 | 2024-04-01 11:13:33 |
| employees                                          | BASE TABLE | InnoDB |     299290 | 2024-04-09 01:40:18 |
| departments                                        | BASE TABLE | InnoDB |          9 | 2024-04-09 01:40:18 |
| dept_manager                                       | BASE TABLE | InnoDB |         24 | 2024-04-09 01:40:18 |
| dept_emp                                           | BASE TABLE | InnoDB |     331143 | 2024-04-09 01:40:18 |
| titles                                             | BASE TABLE | InnoDB |     442308 | 2024-04-09 01:40:18 |
| salaries                                           | BASE TABLE | InnoDB |    2838426 | 2024-04-09 01:40:18 |
| staff                                              | BASE TABLE | InnoDB |          4 | 2024-04-09 02:00:35 |
| continents                                         | BASE TABLE | InnoDB |         14 | 2024-04-10 07:27:12 |
+----------------------------------------------------+------------+--------+------------+---------------------+
45 rows in set (0.11 sec)

mysql>
```

now we will try to rollback a transaction will start transaction by type `START TRANSACTION;`

then we will update values of the table continents with the following

```
update continents set cname='ant' where cid=7;
```

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> update continents set cname='ant' where cid=7;
Query OK, 2 rows affected (0.00 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> select * from continents ;
+-----+--------------+
| cid | cname        |
+-----+--------------+
|   1 | Asia         |
|   2 | Africa       |
|   3 | Europe       |
|   4 | North America |
|   5 | South America |
|   6 | Australia    |
|   7 | ant          |
|   1 | Asia         |
|   2 | Africa       |
|   3 | Europe       |
|   4 | North America |
|   5 | South America |
|   6 | Australia    |
|   7 | ant          |
+-----+--------------+
14 rows in set (0.00 sec)

mysql>
```

nowlet's do rollback by running the command `ROLLBACK;`

```
mysql> rollback;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from continents ;
+-----+---------------+
| cid | cname         |
+-----+---------------+
|   1 | Asia          |
|   2 | Africa        |
|   3 | Europe        |
|   4 | North America |
|   5 | South America |
|   6 | Australia     |
|   7 | Antarctica    |
|   1 | Asia          |
|   2 | Africa        |
|   3 | Europe        |
|   4 | North America |
|   5 | South America |
|   6 | Australia     |
|   7 | Antarctica    |
+-----+---------------+
14 rows in set (0.01 sec)

mysql>
```

we can see that we are able to rollback to original data
let's create an index which one of inndb features

```
create index idx_name on continents (cname );
```

```
mysql> create index idx_name on continents (cname );
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
```

let's check another feature of inndb called row-level-locking
we will start by runinng `START TRANSACTION;` and then updating row using the same command
then we will open another mysql session we will try to delete from the same row

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> update continents set cname='ant' where cid=7;
Query OK, 2 rows affected (0.00 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> select * from continents ;
+-----+---------------+
| cid | cname         |
+-----+---------------+
|   1 | Asia          |
|   2 | Africa        |
|   3 | Europe        |
|   4 | North America |
|   5 | South America |
|   6 | Australia     |
|   7 | ant           |
|   1 | Asia          |
|   2 | Africa        |
|   3 | Europe        |
|   4 | North America |
|   5 | South America |
|   6 | Australia     |
|   7 | ant           |
+-----+---------------+
14 rows in set (0.00 sec)

mysql>
```

now i will open another session of MySQL and i will attempt to delete the same row .

```
Last login: Wed Apr 10 07:08:56 2024
[root@mysqlPerconaSTG ~]# su  dba
[dba@mysqlPerconaSTG root]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use test1
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> delete from continents where cid = 7 ;
```

it will never go through while the transaction is not committed or rollback , this typical example of row level locking .

once we commit the transaction by running `commit;` the delete operation will go through

```
mysql> commit;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

```
Your MySQL connection id is 15
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use test1
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> delete from continents where cid = 7 ;
ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transaction
mysql> delete from continents where cid = 7 ;
Query OK, 2 rows affected (0.01 sec)

mysql>
```

Checking Storage Engine Status

To examine the status of various storage engines, including their current state and operational details, the `SHOW ENGINE` command is a common tool. We've previously touched on its utility and how it's used to gather information about the different engines MySQL supports.

Syntax for general usage:

```
SHOW ENGINE [engine_name] STATUS;
```

Replace `[engine_name]` with the specific storage engine you're interested in, such as InnoDB, ARCHIVE, MyISAM, or any other available engine.

Given that InnoDB is often the focal point of many operations, to specifically check its status, you can use:

```
SHOW ENGINE INNODB STATUS\G;
```

This command provides a detailed snapshot of InnoDB's internal workings, including transaction and locking information, which is invaluable for in-depth analysis and troubleshooting.

```
---------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------
--------------------------+
1 row in set (0.00 sec)

mysql> show engine innoDb status\G
*************************** 1. row ***************************
  Type: InnoDB
  Name:
Status:
=====================================
2024-04-10 07:46:16 140087546312448 INNODB MONITOR OUTPUT
=====================================
Per second averages calculated from the last 31 seconds
-----------------
BACKGROUND THREAD
-----------------
srv_master_thread loops: 34 srv_active, 0 srv_shutdown, 9852 srv_idle
srv_master_thread log flush and writes: 0
----------
SEMAPHORES
----------
OS WAIT ARRAY INFO: reservation count 44
OS WAIT ARRAY INFO: signal count 44
RW-shared spins 0, rounds 0, OS waits 0
RW-excl spins 0, rounds 0, OS waits 0
RW-sx spins 0, rounds 0, OS waits 0
Spin rounds per wait: 0.00 RW-shared, 0.00 RW-excl, 0.00 RW-sx
------------
=====================================
Per second averages calculated from the last 31 seconds
-----------------
BACKGROUND THREAD
-----------------
srv_master_thread loops: 34 srv_active, 0 srv_shutdown, 9852 srv_idle
srv_master_thread log flush and writes: 0
----------
SEMAPHORES
----------
OS WAIT ARRAY INFO: reservation count 44
OS WAIT ARRAY INFO: signal count 44
RW-shared spins 0, rounds 0, OS waits 0
RW-excl spins 0, rounds 0, OS waits 0
RW-sx spins 0, rounds 0, OS waits 0
Spin rounds per wait: 0.00 RW-shared, 0.00 RW-excl, 0.00 RW-sx
------------
TRANSACTIONS
------------
Trx id counter 5274
Purge done for trx's n:o < 5274 undo n:o < 0 state: running but idle
History list length 0
LIST OF TRANSACTIONS FOR EACH SESSION:
---TRANSACTION 421562675551608, not started
0 lock struct(s), heap size 1128, 0 row lock(s)
---TRANSACTION 421562675550760, not started
0 lock struct(s), heap size 1128, 0 row lock(s)
---TRANSACTION 421562675549912, not started
0 lock struct(s), heap size 1128, 0 row lock(s)
--------
FILE I/O
--------
I/O thread 0 state: waiting for completed aio requests ((null))
I/O thread 1 state: waiting for completed aio requests (insert buffer thread)
I/O thread 2 state: waiting for completed aio requests (read thread)
I/O thread 3 state: waiting for completed aio requests (read thread)
I/O thread 4 state: waiting for completed aio requests (read thread)
I/O thread 5 state: waiting for completed aio requests (read thread)
I/O thread 6 state: waiting for completed aio requests (write thread)
I/O thread 7 state: waiting for completed aio requests (write thread)
I/O thread 8 state: waiting for completed aio requests (write thread)
Pending normal aio reads: [0, 0, 0, 0] , aio writes: [0, 0, 0, 0] ,
 ibuf aio reads:
Pending flushes (fsync) log: 0; buffer pool: 0
2066 OS file reads, 15210 OS file writes, 5210 OS fsyncs
0.00 reads/s, 0 avg bytes/read, 0.00 writes/s, 0.00 fsyncs/s
-------------------------------------
INSERT BUFFER AND ADAPTIVE HASH INDEX
```

```
BUFFER POOL AND MEMORY
----------------------
Total large memory allocated 0
Dictionary memory allocated 583141
Buffer pool size   8191
Buffer pool size, bytes 134201344
Free buffers       5763
Database pages     2421
Old database pages 873
Modified db pages  0
Pending reads      0
Pending writes: LRU 0, flush list 0, single page 0
Pages made young 155, not young 4
0.00 youngs/s, 0.00 non-youngs/s
Pages read 2035, created 387, written 1348
0.00 reads/s, 0.00 creates/s, 0.00 writes/s
No buffer pool page gets since the last printout
Pages read ahead 0.00/s, evicted without access 0.00/s, Random read ahead 0.00/s
LRU len: 2421, unzip_LRU len: 0
I/O sum[0]:cur[0], unzip sum[0]:cur[0]
```

```
ROW OPERATIONS
--------------
0 queries inside InnoDB, 0 queries in queue
0 read views open inside InnoDB
0 RW transactions active inside InnoDB
Process ID=12954, Main thread ID=140087119566592 , state=sleeping
Number of rows inserted 14, updated 4, deleted 2, read 84
0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 0.00 reads/s
Number of system rows inserted 135410, updated 398, deleted 41, read 7451
0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 0.00 reads/s
----------------------------
END OF INNODB MONITOR OUTPUT
============================

1 row in set (0.01 sec)
```

The command `SHOW ENGINE INNODB STATUS\G;` offers a concise overview of how the InnoDB storage engine is performing, covering key areas such as transactions, file I/O, buffer pool and memory usage, as well as row operations. This summary is particularly useful for quickly assessing the health and efficiency of the storage engine.

As a DBA, you'll likely find yourself frequently consulting the ROW OPERATIONS and transactions sections of this report. These sections provide valuable insights into the database's operational dynamics, helping you to manage and optimize performance effectively.

ROW OPERATIONS

In the row operations section, we typically examine whether there are any active read and write transactions within InnoDB. This information is crucial for identifying potential blocking or locking issues

that could impact database performance.

```
-------------
ROW OPERATIONS
-------------
0 queries inside InnoDB, 0 queries in queue
0 read views open inside InnoDB
0 RW transactions active inside InnoDB
Process ID=12954, Main thread ID=140087119566592 , state=sleeping
Number of rows inserted 14, updated 4, deleted 2, read 84
0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 0.00 reads/s
Number of system rows inserted 135410, updated 398, deleted 41, read 7451
0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 0.00 reads/s
----------------------------
END OF INNODB MONITOR OUTPUT
============================

1 row in set (0.01 sec)
```

TRANSACTIONS section

the TRANSACTIONS section will show if there any transaction that facing locking or being blocked

```
------------
TRANSACTIONS
------------
Trx id counter 5274
Purge done for trx's n:o < 5274 undo n:o < 0 state: running but idle
History list length 0
LIST OF TRANSACTIONS FOR EACH SESSION:
---TRANSACTION 421562675551608, not started
0 lock struct(s), heap size 1128, 0 row lock(s)
---TRANSACTION 421562675550760, not started
0 lock struct(s), heap size 1128, 0 row lock(s)
---TRANSACTION 421562675549912, not started
0 lock struct(s), heap size 1128, 0 row lock(s)
--------
```

Switching Storage Engine

As a DBA, you might encounter situations where legacy applications, running on older versions of MySQL, use tables created with the MyISAM storage engine. When upgrading to a newer MySQL version, you may inherit these tables. Given the advantages of the more modern InnoDB storage engine, it's advisable to migrate tables from MyISAM to InnoDB before or after an upgrade. This process ensures you benefit from InnoDB's features, such as transaction support and better crash recovery. In this demonstration, we'll show how you can migrate or alter tables from one storage engine to another.

To begin, we create the table uisng MyISAM storage engine

```
CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL)
ENGINE=MyISAM;
```

```
mysql> CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL) ENGINE=MyISAM;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

Next, we'll insert data into the table:

```
INSERT INTO continents(cid, cname)
VALUES
  (1, 'Asia'),
  (2, 'Africa'),
  (3, 'Europe'),
  (4, 'North America'),
  (5, 'South America'),
  (6, 'Australia'),
  (7, 'Antarctica');
```

now we will try to convert the table storage engine to innoDB
`alter table continents engine=innoDB;`

```
mysql> alter table continents engine=innoDB;
Query OK, 7 rows affected (0.05 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql>
```

let's confirm status of table by using command `show create table continents ;`

```
mysql> show create table continents ;
+------------+-----------------------------------------------------------------------------------
---------------------------------+
| Table      | Create Table
                                 |
+------------+-----------------------------------------------------------------------------------
---------------------------------+
| continents | CREATE TABLE `continents` (
  `cid` int NOT NULL,
  `cname` varchar(25) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+------------+-----------------------------------------------------------------------------------
---------------------------------+
1 row in set (0.00 sec)

mysql>
```

Installing New Storage Engine

first of all we need to located the shared library path by running command `show variables like 'plugin%';`

```
mysql> show variables like 'plugin%';
+---------------+-----------------------------+
| Variable_name | Value                       |
+---------------+-----------------------------+
| plugin_dir    | /usr/lib64/mysql/plugin/ |
+---------------+-----------------------------+
1 row in set (0.01 sec)

mysql>
```

```
mysql> ^C^C
mysql> Bye
[dba@mysqlPerconaSTG mysql]$ sudo ls /usr/lib64/mysql/plugin/
[sudo] password for dba:
adt_null.so                        component_mysqlbackup.so                  keyring_udf.so
audit_log_filter.so                component_query_attributes.so             keyring_vault.so
audit_log.so                       component_reference_cache.so              libfnv1a_udf.so
authentication_fido_client.so      component_test_audit_api_message.so       libfnv_udf.so
authentication_fido.so             component_test_component_deinit.so        libmemcached.so
authentication_kerberos_client.so  component_test_host_application_signal.so  libmurmur_udf.so
authentication_ldap_sasl_client.so component_test_mysql_system_variable_set.so libpluginmecab.so
authentication_ldap_sasl.so        component_test_mysql_thd_store_service.so locking_service.so
authentication_ldap_simple.so      component_test_server_telemetry_traces.so mypluglib.so
authentication_oci_client.so       component_test_table_access.so            mysql_clone.so
auth_pam_compat.so                 component_test_udf_services.so            mysql_no_login.so
auth_pam.so                        component_validate_password.so            procfs.so
auth_socket.so                     connection_control.so                     rewrite_example.so
binlog_utils_udf.so                data_masking.ini                          rewriter.so
component_audit_api_message_emit.so data_masking.so                          semisync_master.so
component_encryption_udf.so        ddl_rewriter.so                           semisync_replica.so
component_keyring_file.so          debug                                     semisync_slave.so
component_keyring_kmip.so          dialog.so                                 semisync_source.so
component_keyring_kms.so           group_replication.so                      test_services_host_application_signal.so
component_log_filter_dragnet.so    ha_example.so                             test_udf_wrappers.so
component_log_sink_json.so         ha_mock.so                                validate_password.so
component_log_sink_syseventlog.so  innodb_engine.so                          version_token.so
component_masking_functions.so     keyring_file.so
[dba@mysqlPerconaSTG mysql]$
```

we will try to install `ha_example.so`

```
mysql> ^C^C
mysql> Bye
[dba@mysqlPerconaSTG mysql]$ sudo ls /usr/lib64/mysql/plugin/
[sudo] password for dba:
adt_null.so                        component_mysqlbackup.so                  keyring_udf.so
audit_log_filter.so                component_query_attributes.so             keyring_vault.so
audit_log.so                       component_reference_cache.so              libfnv1a_udf.so
authentication_fido_client.so      component_test_audit_api_message.so       libfnv_udf.so
authentication_fido.so             component_test_component_deinit.so        libmemcached.so
authentication_kerberos_client.so  component_test_host_application_signal.so  libmurmur_udf.so
authentication_ldap_sasl_client.so component_test_mysql_system_variable_set.so libpluginmecab.so
authentication_ldap_sasl.so        component_test_mysql_thd_store_service.so locking_service.so
authentication_ldap_simple.so      component_test_server_telemetry_traces.so mypluglib.so
authentication_oci_client.so       component_test_table_access.so            mysql_clone.so
auth_pam_compat.so                 component_test_udf_services.so            mysql_no_login.so
auth_pam.so                        component_validate_password.so            procfs.so
auth_socket.so                     connection_control.so                     rewrite_example.so
binlog_utils_udf.so                data_masking.ini                          rewriter.so
component_audit_api_message_emit.so data_masking.so                          semisync_master.so
component_encryption_udf.so        ddl_rewriter.so                           semisync_replica.so
component_keyring_file.so          debug                                     semisync_slave.so
component_keyring_kmip.so          dialog.so                                 semisync_source.so
component_keyring_kms.so           group_replication.so                      test_services_host_application_signal.so
component_log_filter_dragnet.so    ha_example.so                             test_udf_wrappers.so
component_log_sink_json.so         ha_mock.so                                validate_password.so
component_log_sink_syseventlog.so  innodb_engine.so                          version_token.so
component_masking_functions.so     keyring_file.so
[dba@mysqlPerconaSTG mysql]$
```

```
install plugin example SONAME 'ha_example.so';
```

```
mysql> install plugin example SONAME 'ha_example.so';
Query OK, 0 rows affected (0.04 sec)

mysql>
```

now to confirm if it installed we will use the below command

`show engines\G;`

```
mysql> show engines\G;
*************************** 1. row ***************************
     Engine: EXAMPLE
    Support: YES
    Comment: Example storage engine
Transactions: NO
         XA: NO
 Savepoints: NO
*************************** 2. row ***************************
     Engine: ndbcluster
    Support: NO
    Comment: Clustered, fault-tolerant tables
Transactions: NULL
         XA: NULL
 Savepoints: NULL
*************************** 3. row ***************************
     Engine: FEDERATED
    Support: NO
    Comment: Federated MySQL storage engine
```

similar way we can uninstall the storage engine you don't need to five the shared library name , simply give the storage name

`uninstall plugin example;`

```
mysql> uninstall plugin example;
Query OK, 0 rows affected (0.03 sec)

mysql> show engines\G;
```

```
Transactions: NO
         XA: NO
  Savepoints: NO
12 rows in set (0.00 sec)

ERROR:
No query specified

mysql> uninstall plugin example;
Query OK, 0 rows affected (0.03 sec)

mysql> show engines\G;
*************************** 1. row ***************************
      Engine: ndbcluster
     Support: NO
     Comment: Clustered, fault-tolerant tables
Transactions: NULL
         XA: NULL
  Savepoints: NULL
*************************** 2. row ***************************
      Engine: FEDERATED
     Support: NO
     Comment: Federated MySQL storage engine
Transactions: NULL
         XA: NULL
  Savepoints: NULL
*************************** 3. row ***************************
      Engine: PERFORMANCE_SCHEMA
     Support: YES
```

Disabling Storage Engine

As a DBA, if you have completed migrating all tables from older storage engines like MyISAM to the more robust InnoDB, and you now wish to prevent users from creating tables using the MyISAM engine, there is a straightforward method to achieve this. You can disable specific storage engines to prevent their use using the `disabled_storage_engines` variable. This setting allows you to specify which engines should be prohibited, effectively steering all table creation towards preferred technologies such as InnoDB. This proactive approach ensures consistency and leverages the advanced features of modern storage engines.

if you check the status of `disabled_storage_engines` variable you will find the value to b empty , meaning no storaged engin was disabled

```
+---------------------------+-------+
| Variable_name             | Value |
+---------------------------+-------+
| disabled_storage_engines  |       |
+---------------------------+-------+
```

copy the variable name and we will edit `my.cnf` with a list of storage engine that we want to disable

as motioned before `my.cnf` is located under the `/etc` directory .

`sudo vi /etc/my.cnf`

```
[dba@mysqlPerconaSTG mysql]$ sudo vi /etc/my.cnf
```

at the end of the file put the variable follow by `=` then put the list of storage engine you want to disable .

```
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines = "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
-- INSERT --
```

next restart MySQL server services

`sudo systemct restart mysqld`

after that if we run the `show variables like 'disabled%';` we will find value added with disabled engine we listed

```
mysql> show variables like 'disabled%';
+--------------------------+-------------------------------------+
| Variable_name            | Value                               |
+--------------------------+-------------------------------------+
| disabled_storage_engines | MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE |
+--------------------------+-------------------------------------+
1 row in set (0.01 sec)

mysql>
```

if if the user attemtp to created table with MyISAM will not go through

```
CREATE TABLE continents (cid INT NOT NULL, cname VARCHAR(25) NOT NULL)
ENGINE=MyISAM;
```

```
Database changed
mysql> CREATE TABLE continent1 (cid INT NOT NULL, cname VARCHAR(25) NOT NULL) ENGINE=MyISAM;
ERROR 3161 (HY000): Storage engine MyISAM is disabled (Table creation is disallowed).
mysql>
```

5- MySQL User Administration

DBA Account

Up until now, we've predominantly used the root account on localhost for database management. However, as we expand access to remote users, it’s essential to understand how to establish non-root user accounts for remote access. This setup will allow users to connect via various clients like the standard MySQL Client, the newer MySQL Shell, or graphical interfaces such as MySQL Workbench. Here are the steps and concepts we'll cover:

- **Creating Your First Database Account**:
  - We will create a database account that can be used to log in remotely, using different MySQL clients.
- **Understanding 'WITH GRANT OPTION'**:
  - We'll explain the significance of the 'WITH GRANT OPTION' and why it's crucial for DBAs to have this ability, which allows them to grant permissions to other users.

- **Distinguishing Between MySQL Roles and Users**:
  - A discussion on the differences between roles and users in MySQL will be provided, clarifying their distinct uses and management.
- **Granting Permissions to Roles and Users**:
  - We'll demonstrate how to grant permissions effectively to roles, and subsequently to users, to streamline user administration and enhance security.
- **Locking and Unlocking MySQL Accounts**:
  - Instructions on how to secure MySQL accounts by locking and unlocking them as needed for security management.
- **Creating Expired Accounts**:
  - We will show how to set up accounts with expiration dates to control access and enforce security policies.

These steps will ensure comprehensive access management and enhanced security for MySQL database operations remotely.

MySQL Permissions

Permissions in MySQL are privileges granted to users that allow them to perform specific actions within the database system. Here is a detailed list of common permissions available in MySQL:

- **General Permissions**:
  - `ALL`: Grants all available permissions to a user. This is a comprehensive privilege that covers all specified actions within the scope it is applied to (e.g., a database, table, or procedure).
- **Specific Permissions**:
  - `ALTER`: Allows the user to modify the structure of a database or table (e.g., changing the schema).
  - `CREATE`: Enables the creation of new databases, tables, or other objects.
  - `DROP`: Provides the ability to delete databases, tables, indexes, etc.
  - `EXECUTE`: Permits the user to run stored procedures.
  - `INSERT`, `DELETE`, `UPDATE`, `RENAME`: These are row-level permissions that allow the user to perform respective actions on data within tables.
  - `SELECT`, `SHOW`: These permissions enable read-only access, allowing users to view but not modify data.
- **Wildcard Permission**:
  - `.*`: This notation is used to apply a permission to all objects within a database. For example, specifying `GRANT SELECT ON database_name.*` allows a user to perform `SELECT` operations

on all tables within `database_name`.

- **Replication-Related Permissions**:
  - Includes privileges like `REPLICATION CLIENT` and `REPLICATION SLAVE`, which control the ability to manage and monitor replication processes.
- **Using `GRANT`**:
  - The `GRANT` keyword is used to assign any of the above permissions to a user. The syntax for granting permissions typically follows the pattern: `GRANT PERMISSION_TYPE ON object TO 'username'@'host';`

WITH GRANT OPTION

The `WITH GRANT OPTION` clause is pivotal when creating MySQL user accounts, particularly for users who will have administrative roles, such as DBAs. Here's how it works and why it's important:

- **Purpose of `WITH GRANT OPTION`**:
  - This clause allows a user not only to possess certain privileges but also to grant those privileges to other users.
- **When to Use**:
  - If you intend for a user to have the ability to propagate their permissions to others, include the `WITH GRANT OPTION` clause when granting them privileges. This is especially crucial for users who manage permissions, such as DBAs.
- **Recommendation for DBAs**:
  - Always use the `WITH GRANT OPTION` when creating DBA accounts to ensure they have the necessary authority to manage user permissions effectively.
- **Syntax Example**:
  - When granting a DBA user all privileges on all databases and the ability to grant those privileges to others, you would use:

```
GRANT ALL PRIVILEGES ON *.* TO 'dba_username'@'host' WITH GRANT OPTION;
```

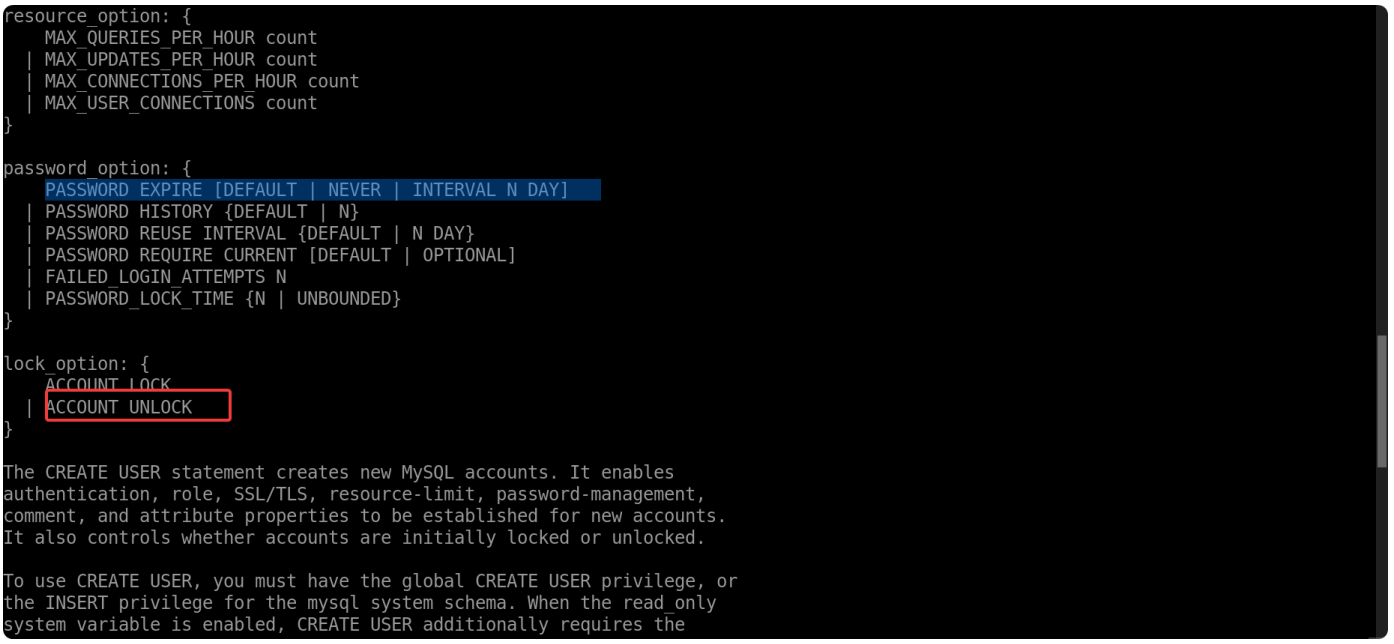
creating DBA account .

reference link
we will create our first non-root DBA user , always before you start any command see if there is help section available
`mysql>help create user`

```
mysql> help create user;
Name: 'CREATE USER'
Description:
Syntax:
CREATE USER [IF NOT EXISTS]
    user [auth_option] [, user [auth_option]] ...
    DEFAULT ROLE role [, role ] ...
    [REQUIRE {NONE | tls_option [[AND] tls_option] ...}]
    [WITH resource_option [resource_option] ...]
    [password_option | lock_option] ...
    [COMMENT 'comment_string' | ATTRIBUTE 'json_object']

user:
    (see )

auth_option: {
    IDENTIFIED BY 'auth_string' [AND 2fa_auth_option]
  | IDENTIFIED BY RANDOM PASSWORD [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin BY 'auth_string' [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin BY RANDOM PASSWORD [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin AS 'auth_string' [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin [initial_auth_option]
}
```

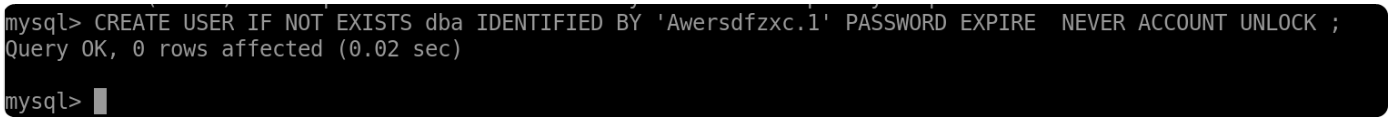
```
Description:
Syntax:
CREATE USER [IF NOT EXISTS]
    user [auth_option] [, user [auth_option]] ...
    DEFAULT ROLE role [, role ] ...
    [REQUIRE {NONE | tls_option [[AND] tls_option] ...}]
    [WITH resource_option [resource_option] ...]
    [password_option | lock_option] ...
    [COMMENT 'comment_string' | ATTRIBUTE 'json_object']

user:
    (see )

auth_option: {
    IDENTIFIED BY 'auth_string' [AND 2fa_auth_option]
  | IDENTIFIED BY RANDOM PASSWORD [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin BY 'auth_string' [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin BY RANDOM PASSWORD [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin AS 'auth_string' [AND 2fa_auth_option]
  | IDENTIFIED WITH auth_plugin [initial_auth_option]
}

2fa_auth_option: {
    IDENTIFIED BY 'auth_string' [AND 3fa_auth_option]
  | IDENTIFIED BY RANDOM PASSWORD [AND 3fa_auth_option]
  | IDENTIFIED WITH auth_plugin [AND 3fa_auth_option]
  | IDENTIFIED WITH auth_plugin BY 'auth_string' [AND 3fa_auth_option]
  | IDENTIFIED WITH auth_plugin BY RANDOM PASSWORD [AND 3fa_auth_option]
```

```
password_option: {
    PASSWORD EXPIRE [DEFAULT | NEVER | INTERVAL N DAY]
  | PASSWORD HISTORY {DEFAULT | N}
  | PASSWORD REUSE INTERVAL {DEFAULT | N DAY}
  | PASSWORD REQUIRE CURRENT [DEFAULT | OPTIONAL]
  | FAILED_LOGIN_ATTEMPTS N
  | PASSWORD_LOCK_TIME {N | UNBOUNDED}
}

lock_option: {
    ACCOUNT LOCK
  | ACCOUNT UNLOCK
}

The CREATE USER statement creates new MySQL accounts. It enables
authentication, role, SSL/TLS, resource-limit, password-management,
comment, and attribute properties to be established for new accounts.
It also controls whether accounts are initially locked or unlocked.

To use CREATE USER, you must have the global CREATE USER privilege, or
the INSERT privilege for the mysql system schema. When the read_only
system variable is enabled, CREATE USER additionally requires the
```



this a very help section we can copy the syntax for creating user as showing in above photo and edit as you which , this context i have setup the below command using the help section

`CREATE USER IF NOT EXISTS dba IDENTIFIED BY 'Awersdfzxc.1' PASSWORD EXPIRE NEVER ACCOUNT UNLOCK ;`



syntax explanation

- **User Creation Basics**:
  - `CREATE USER [IF NOT EXISTS] [user-name]` allows you to create a new user. Using `IF NOT EXISTS` is optional but recommended if you want to prevent duplicate usernames.
- **Specifying the Host**:
  - `username` without an `@` and host specifies that the user can connect from any host.
  - `user@'localhost'` restricts the user to connect only from the local machine.
  - `user@'ip'` allows the user to connect from a specified IP address.
  - `user@'%'` permits the user to connect from any IP address or host.
- **Setting a Password**:
  - `IDENTIFIED BY 'auth_string'` sets the user's password by replacing `'auth_string'` with the desired password.
- **Password Expiry Options**:
  - `PASSWORD EXPIRE [DEFAULT | NEVER | INTERVAL N DAY]` controls how the user's password expiration is handled:
    - `DEFAULT`: Applies the default password policy of the server.

    - `NEVER`: The password never expires.
    - `INTERVAL N DAY`: The password expires after N days.
- **Account Status**:
    - `ACCOUNT UNLOCK`: This option ensures the user account is active and unlocked, allowing immediate login access.

granting permission

we have create the user `dba` now we need to grant permission
snice this is DBA account i will grant it all permission using `GRANT ALL PRIVILEGES`
also we will add `WITH GRANT OPTION;` clause so DBA account can give permission to other users
SYNTAX
`mysql>GRANT [ALL PRIVILEGES | delete , create etc ] ON [*.* | table name | database name ] TO [username] WITH GRANT OPTION [is optional];`

`mysql>GRANT ALL PRIVILEGES ON *.* TO DBA WITH GRANT OPTION;`

```
mysql> GRANT ALL PRIVILEGES ON *.* TO dba WITH GRANT OPTION;
Query OK, 0 rows affected (0.02 sec)

mysql>
```

Syntax Explanation in Points:

- `GRANT ALL PRIVILEGES`: Assigns all available permissions to the user.
- `ON *.*`: These permissions apply across all databases and tables.
- `TO 'DBA'`: The privileges are granted to the user 'DBA'.
- `WITH GRANT OPTION;`: Allows 'DBA' to grant their permissions to other users, a necessary feature for administrative roles.

FLUSH PRIVILEGES

once you have granted privileges you must run `FLUSH PRIVILEGES;` so that grant tables can reactivate right away
`mysql>FLUSH PRIVILEGES;`

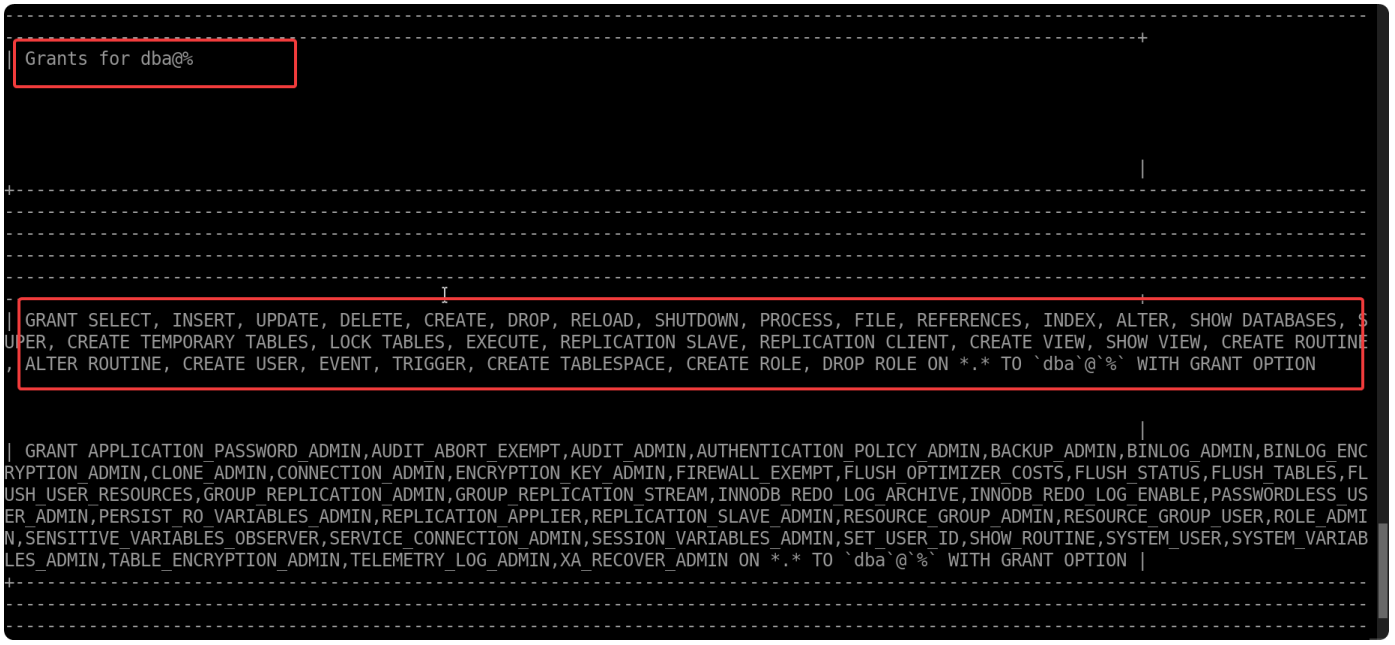
```
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

view the user privileges

we can use the below command that will show permission for certain user

`show grants for [username];`

`show grants for dba;`



Connecting to MySQL

[reference link]

since we have created our first MySQL account now its time to connect to MySQL remotely
we will show three methods to connect

connecting using MySQL client

To connect to a remote MySQL server using a MySQL client installed on another server, you will use the following syntax:

```
mysql --host=[ip or hostname] --user=[username] --password=[password]
```

Alternatively, if you prefer to enter the password interactively to ensure security, you can omit the password value, which will then prompt you to enter the password:

```
mysql --host=[ip or hostname] --user=[username] --password
```

For example, to connect as the user 'dba' to a MySQL server at the IP address `10.10.10.112`, you would use:

```
mysql --host=10.10.10.112 --user=dba --password
```

```
[root@Mysqlcom-RHEL-STG ~]# mysql --host=10.10.10.112 --user=dba --password
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

to confirm we are connected to remote MySQL server we will use `select user();`

```
mysql> select user();
+--------------------+
| user()             |
+--------------------+
| dba@10.10.10.138   |
+--------------------+
1 row in set (0.00 sec)

mysql>
```

also if we run `\s` will show status

```
+--------------------+
1 row in set (0.00 sec)

mysql> \s
--------------
mysql  Ver 8.0.36 for Linux on x86_64 (MySQL Community Server - GPL)

Connection id:          10
Current database:
Current user:           dba@10.10.10.138
SSL:                    Cipher in use is TLS_AES_256_GCM_SHA384
Current pager:          stdout
Using outfile:          ''
Using delimiter:        ;
Server version:         8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19
Protocol version:       10
Connection:             10.10.10.112 via TCP/IP
Server characterset:    utf8mb4
Db     characterset:    utf8mb4
Client characterset:    utf8mb4
Conn.  characterset:    utf8mb4
TCP port:               3306
Binary data as:         Hexadecimal
Uptime:                 16 hours 7 min 42 sec

Threads: 3  Questions: 24  Slow queries: 0  Opens: 177  Flush tables: 3  Open tables: 96  Queries per second avg: 0.000
--------------

mysql>
```

connecting using connecting using MySQL client

we will connect remotely using MySQL shell using SQL mode

below is the syntax

```
mysqlsh --sql --uri=username@ip or host:3306/[database-name ]
```

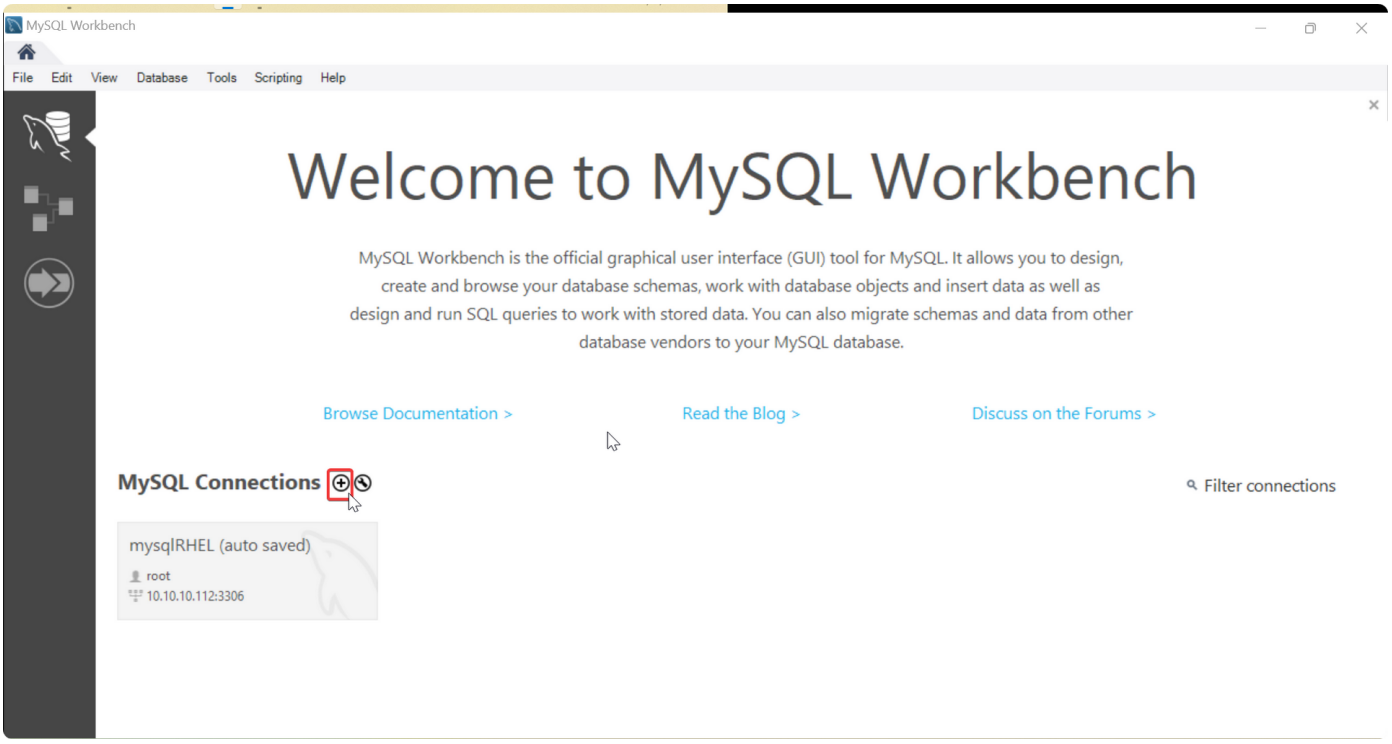
```
mysqlsh --sql --uri=dba@10.10.10.112:3306/mysql
```

```
[root@Mysqlcom-RHEL-STG ~]# mysqlsh --sql --uri=dba@10.10.10.112:3306/mysql
Please provide the password for 'dba@10.10.10.112:3306': ************
Save password for 'dba@10.10.10.112:3306'? [Y]es/[N]o/Ne[v]er (default No): y
MySQL Shell 8.0.36

Copyright (c) 2016, 2023, Oracle and/or its affiliates.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.

Type '\help' or '\?' for help; '\quit' to exit.
Creating a session to 'dba@10.10.10.112:3306/mysql'
Fetching global names, object names from `mysql` for auto-completion... Press ^C to stop.
Your MySQL connection id is 12
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19
Default schema set to `mysql`.
 MySQL  10.10.10.112:3306 ssl  mysql  SQL > show user ();
ERROR: 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the
right syntax to use near 'user ()' at line 1
 MySQL  10.10.10.112:3306 ssl  mysql  SQL > \s
MySQL Shell version 8.0.36

Connection Id:                12
Current schema:               mysql
Current user:                 dba@10.10.10.138
SSL:                          Cipher in use: TLS_AES_256_GCM_SHA384 TLSv1.3
Using delimiter:              ;
Server version:               8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19
Protocol version:             Classic 10
Client library:               8.0.36
```

connecting using connecting using MySQL workbench

for that we need to download MySQL workbench , you use the following link to download it

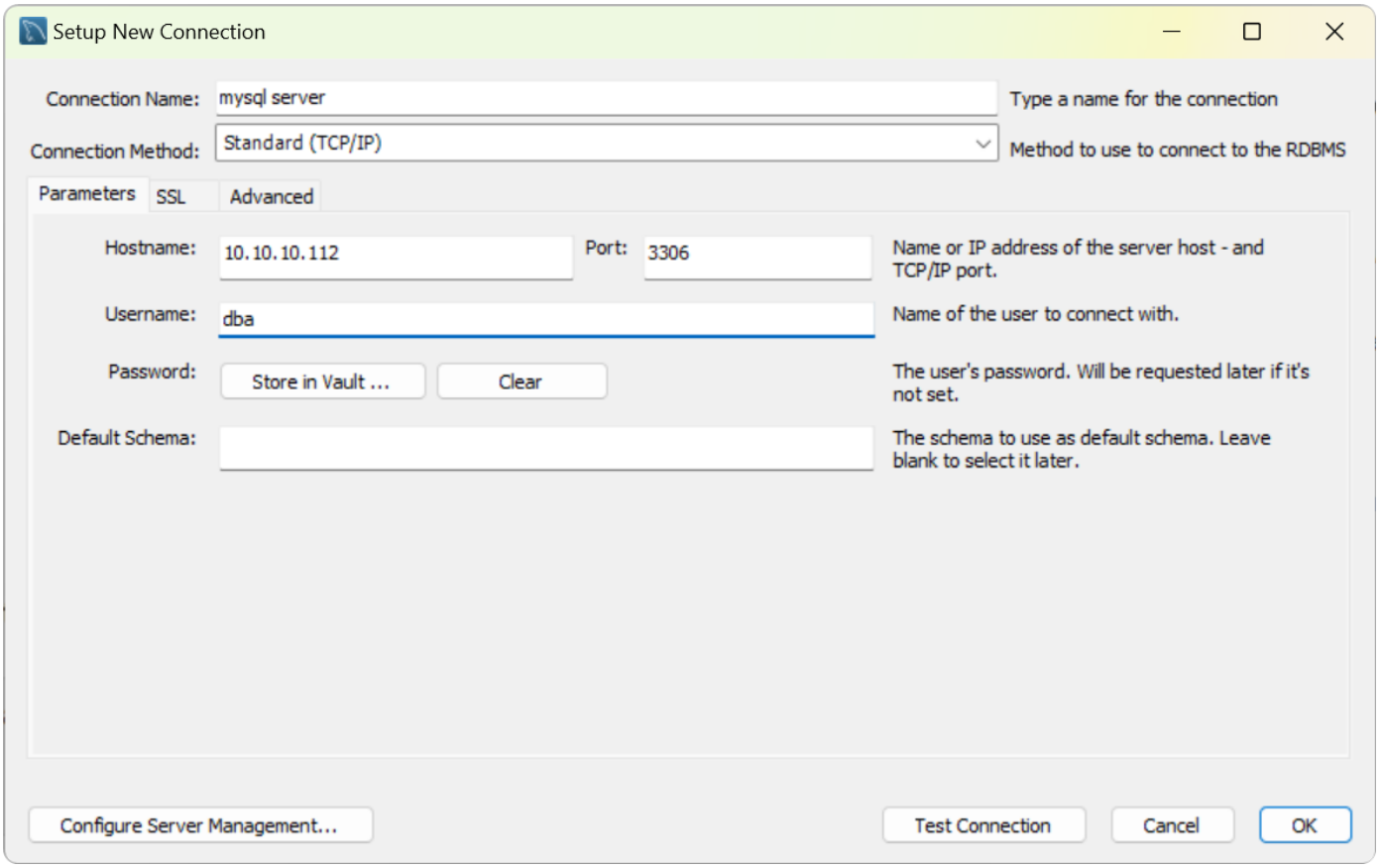
one you have installed MySQL workbench start up and then + button to start filling the connection string



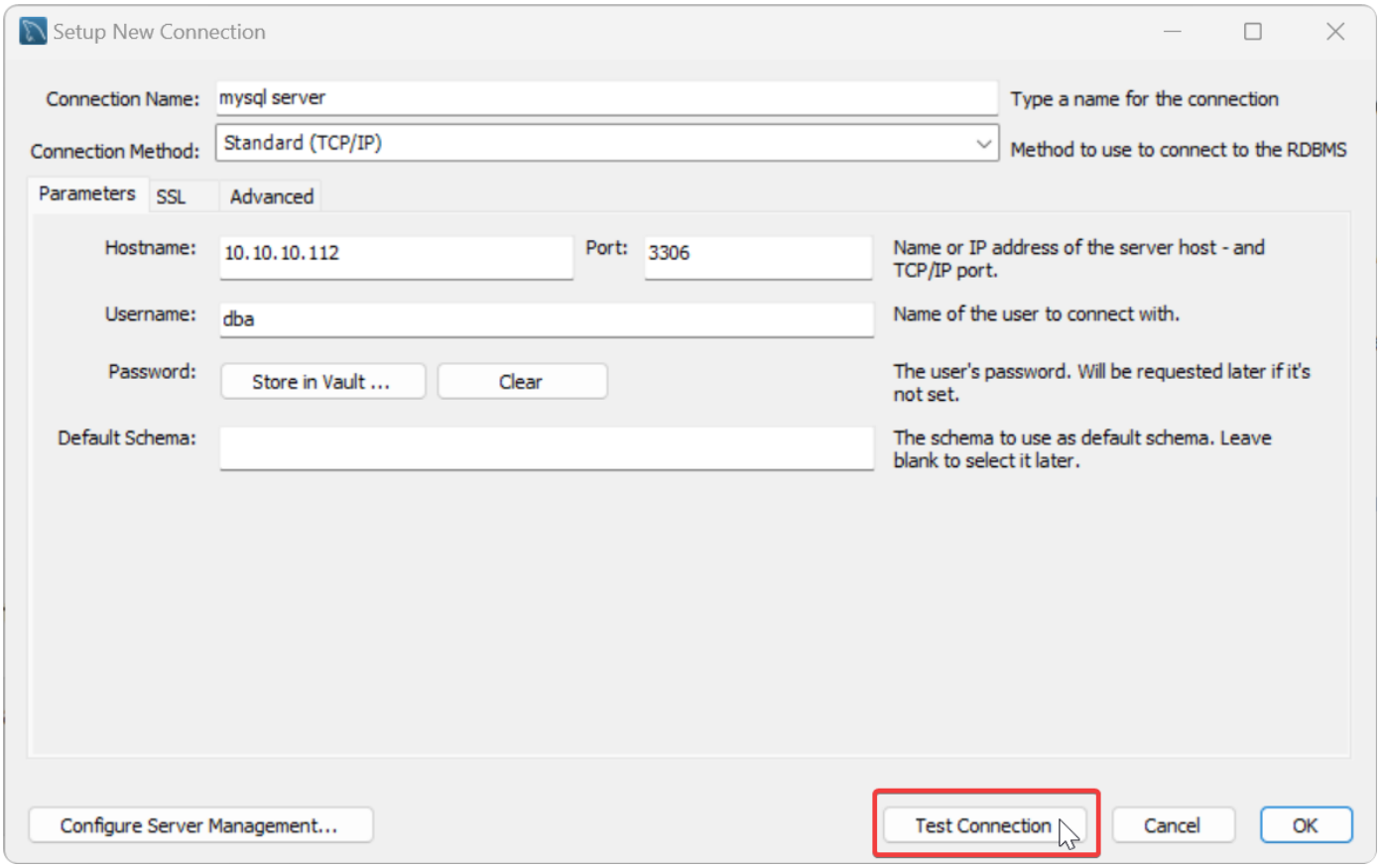
in the connection name give any friendly name for your connection

in host section provide hostname or Ip of the remote MySQL server

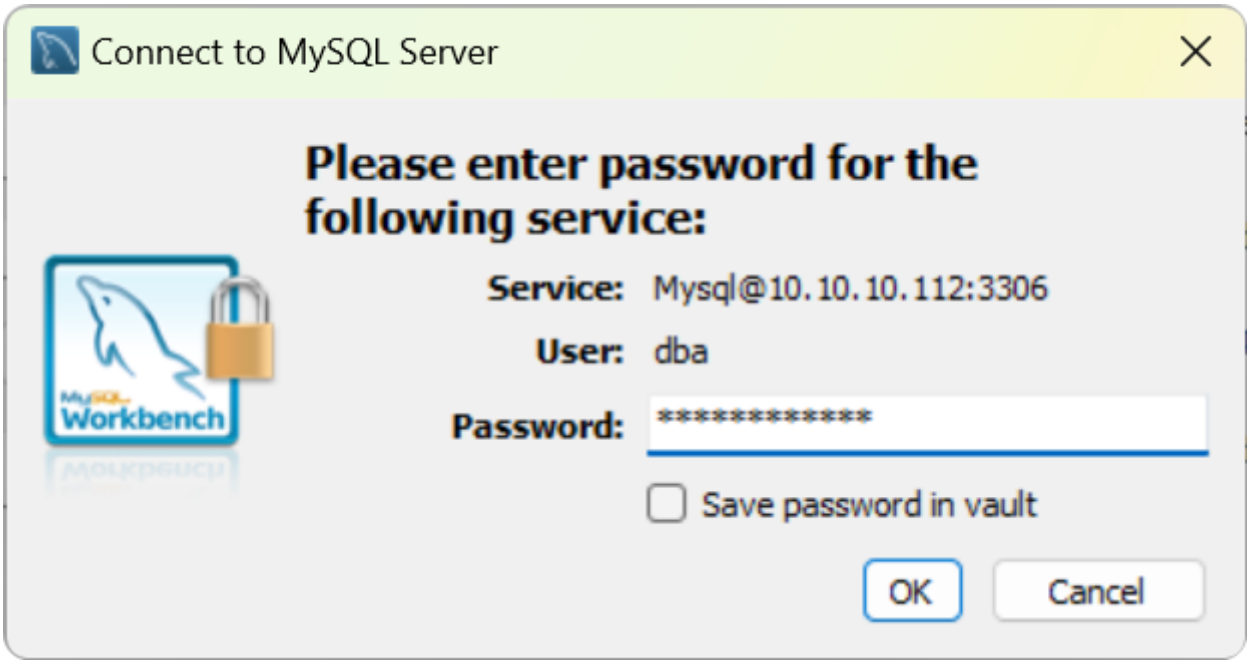
in the username section put the MYSQL account that have permission in our case we will fill dba

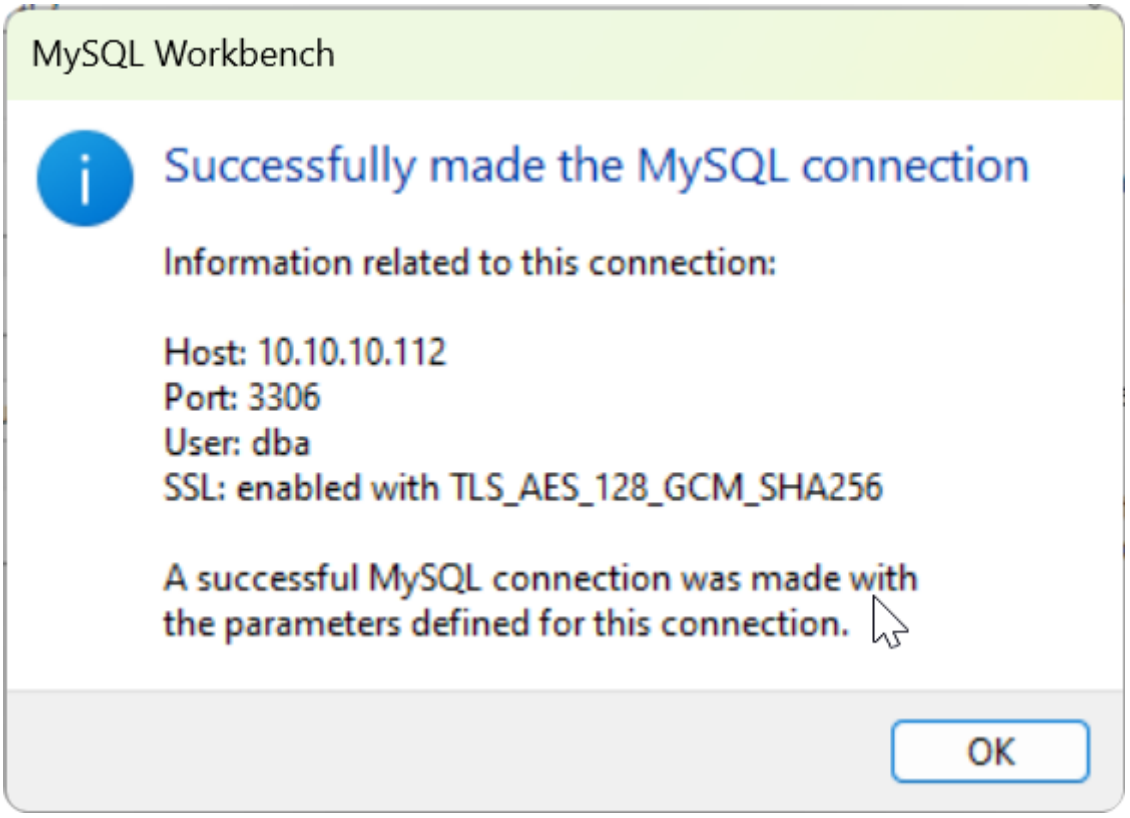


once done click Test connection to see if the connection is enabled

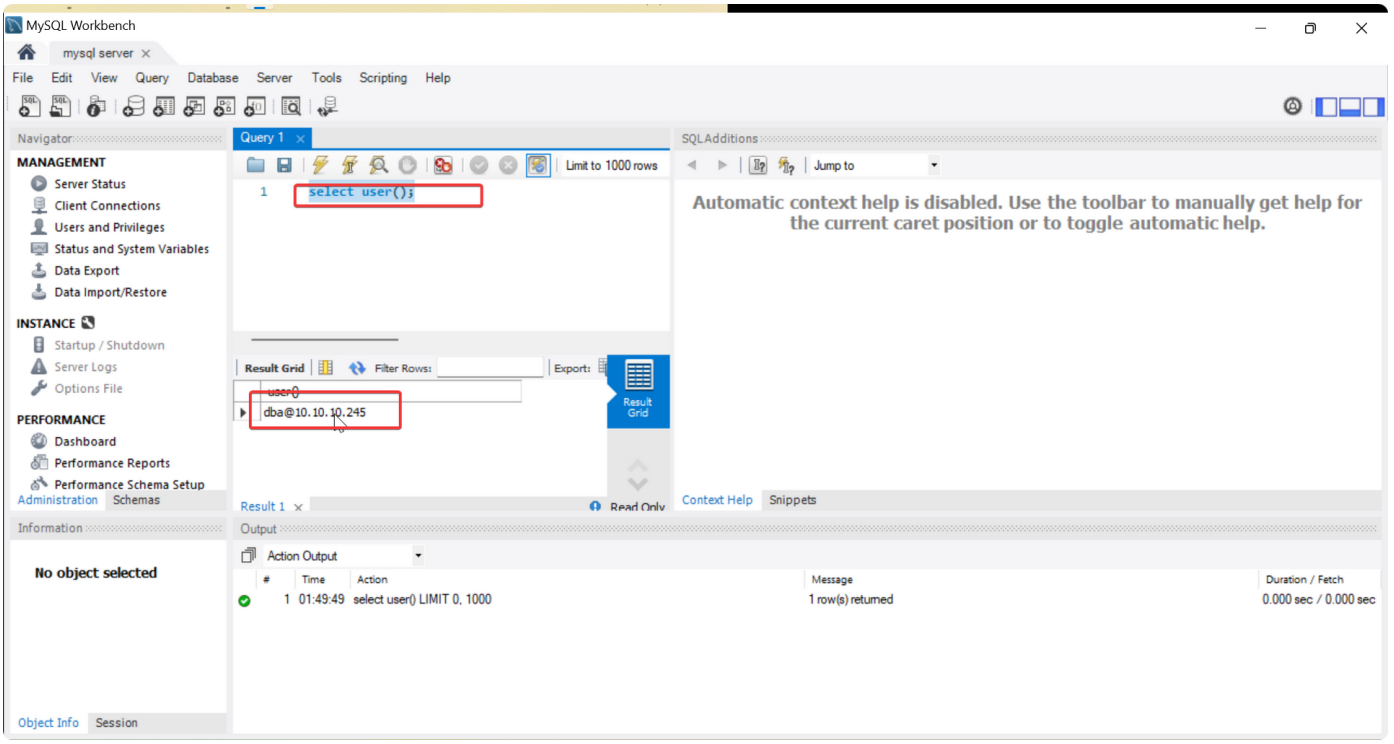


you will prompt to fill password for your account





the connection seceded now you can click OK button and you will MySQL to mysql workbench



mysql_native_password & caching_sha2_password auth plugins

MySQL supports two primary authentication plugins which are essential for configuring user security: **mysql_native_password** and **caching_sha2_password**. Understanding the differences and configurations of these plugins is crucial for effective database management.

mysql_native_password

- **Functionality**: Implements the native pluggable authentication, known as 'NPM', which is based on the password hashing methods used before the introduction of pluggable authentication.
- **Compatibility**: This plugin is non-pluggable, meaning it cannot be loaded or unloaded on the fly while MySQL server is running. There is no separate library file because this plugin is built into the MySQL server itself.
- **Usage**: In MySQL Server versions 5.x and older, `mysql_native_password` was the default authentication method.

To start MySQL server using mysql_native_password, use the following syntax:

```
mysql --default-auth=mysql_native_password
```

caching_sha2_password

- **Default Setting**: From MySQL server 8.0 onwards, `caching_sha2_password` is the default password authentication plugin.
- **Recommendation**: MySQL recommends using this plugin as it provides a more secure password encryption using SHA-256.

- **Functionality**: The server assigns this plugin to an account and uses it to encrypt the password. These encrypted values are then stored in the `authentication_string` in the user table of the MySQL system database.
- **Integration**: This plugin is built into the server and needs to be explicitly loaded; it cannot be disabled by unloading.

which one to use ? mysql_native_password or caching_sha2_password

in some cases you might face issue where user will complain that he is unable to connect to MySQL server
the user also add that he is getting the following error 'error authentication plugin caching_sha2_password cannot be loaded '

the error means that the application or the connector is still using the legacy old authentication method meaning they are still using mysql_native_password
quit possible that the connector our the application has not been upgraded to use caching_sha2_password

workaround
you as a dba will have to go back to the user and alter the user to use mysql_native_password

syntax :
`ALTER USER [user-name] IDENTIFIED WITH 'mysql_native_password' by 'password for the user';`

create user that uses either mysql_native_password or caching_sha2_password

another option you have is that you can specify which authentication method the user will use while creating the user

syntax:
`CREATE USER [USERNAME] WITH 'mysql_native_password or caching_sha2_password' by password ['password for the user '];`

MySQL Roles

- ROLES is named collection of privileges
- to create role your user must have `GLOBAL CREATE ROLE` or `CREATE USER` privilege
- whenever you create a role this will be recorded as an event in database and this will be written in binary log
- when role is created is by default locked , and used the default auth plugin

- when creating a role there will not be any password given to the role . so the authentication string is empty
- Roles can be considered as Users in the MySQL database user table
  syntax

```
CREATE ROLE IF NOT EXISTS 'USERNAME' , 'USERNAME2' , 'USERNAME3';
```

CREATING MySQL Roles

REFREANCE LINK

as usual we will check help section for creating role

```
HELP CREATE ROLE
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> help create role;
Name: 'CREATE ROLE'
Description:
Syntax:
CREATE ROLE [IF NOT EXISTS] role [, role ] ...

CREATE ROLE creates one or more roles, which are named collections of
privileges. To use this statement, you must have the global CREATE ROLE
or CREATE USER privilege. When the read_only system variable is
enabled, CREATE ROLE additionally requires the CONNECTION_ADMIN
privilege (or the deprecated SUPER privilege).

A role when created is locked, has no password, and is assigned the
default authentication plugin. (These role attributes can be changed
later with the ALTER USER statement, by users who have the global
CREATE USER privilege.)

CREATE ROLE either succeeds for all named roles or rolls back and has
no effect if any error occurs. By default, an error occurs if you try
to create a role that already exists. If the IF NOT EXISTS clause is
given, the statement produces a warning for each named role that
already exists, rather than an error.

The statement is written to the binary log if it succeeds, but not if
it fails; in that case, rollback occurs and no changes are made. A
statement written to the binary log includes all named roles. If the IF
```

now we will create the following roles 'reader' , writer , admin

```
CREATE ROLE IF NOT EXISTS READER, WRITER, ADMIN;
```

```
mysql> CREATE ROLE IF NOT EXISTS READER, WRITER, ADMIN;
Query OK, 0 rows affected (0.05 sec)

mysql>
```

then we will granted it permission as per the user name suggest

```
GRANT SELECT ON employees.employees TO READER;
```

```
GRANT INSERT, UPDATE, DELETE ON employees.employees TO WRITER;
```

```
GRANT ALL ON employees.* TO ADMIN;
```

```
mysql> GRANT SELECT ON employees.employees  TO READER;
Query OK, 0 rows affected (0.02 sec)

mysql> GRANT INSERT, UPDATE, DELETE ON employees.employees TO WRITER;
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL ON employees.* TO ADMIN
    -> ;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

next we we will confirm the status of the role and there privilege using the below query

```
SHOW GRANTS FOR READER;
SHOW GRANTS FOR WRITER;
SHOW GRANTS FOR ADMIN;
```

```
mysql> SHOW GRANTS FOR READER;
+-----------------------------------------------------+
| Grants for READER@%                                 |
+-----------------------------------------------------+
| GRANT USAGE ON *.* TO `READER`@`%`                  |
| GRANT SELECT ON `employees`.`employees` TO `READER`@`%` |
+-----------------------------------------------------+
2 rows in set (0.00 sec)

mysql> SHOW GRANTS FOR WRITER;
+-------------------------------------------------------------------------+
| Grants for WRITER@%                                                     |
+-------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `WRITER`@`%`                                      |
| GRANT INSERT, UPDATE, DELETE ON `employees`.`employees` TO `WRITER`@`%` |
+-------------------------------------------------------------------------+
2 rows in set (0.01 sec)

mysql> SHOW GRANTS FOR ADMIN;
+-----------------------------------------------+
| Grants for ADMIN@%                            |
+-----------------------------------------------+
| GRANT USAGE ON *.* TO `ADMIN`@`%`             |
| GRANT ALL PRIVILEGES ON `employees`.* TO `ADMIN`@`%` |
+-----------------------------------------------+
2 rows in set (0.00 sec)

mysql>
```

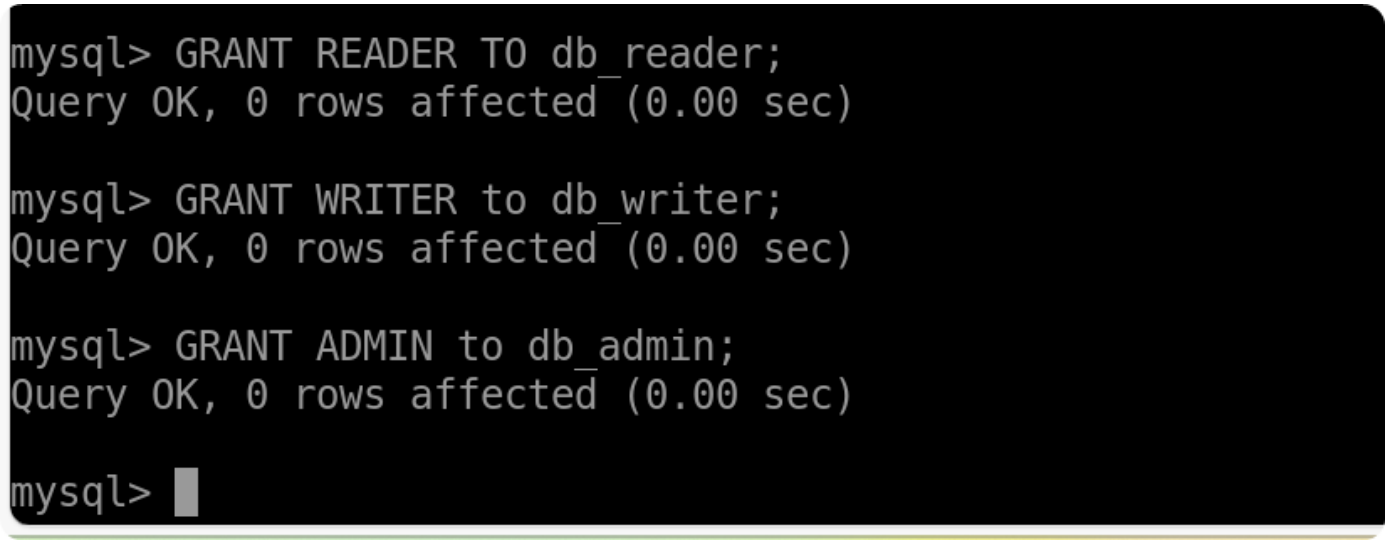
next we will create 3 user and assignee them the roles as privilege

```
CREATE USER IF NOT EXISTS db_reader IDENTIFIED BY 'P@ssw0rd';
CREATE USER IF NOT EXISTS db_writer IDENTIFIED BY 'P@ssw0rd';
CREATE USER IF NOT EXISTS db_admin IDENTIFIED BY 'P@ssw0rd';
```

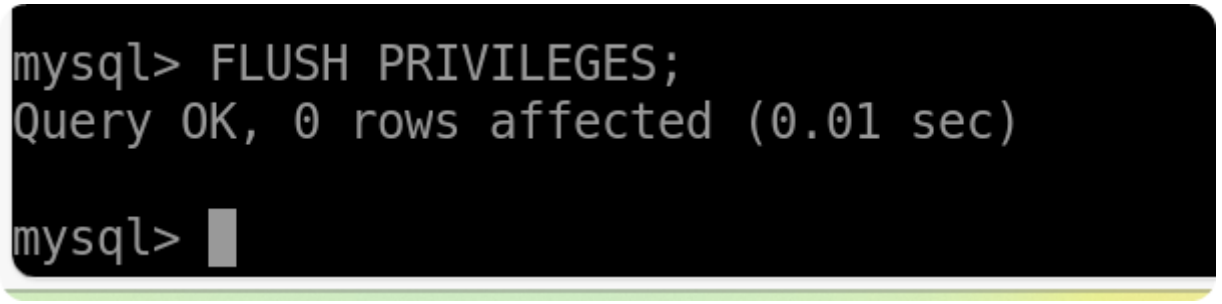
```
mysql> CREATE USER IF NOT EXISTS db_reader IDENTIFIED BY 'P@ssw0rd';
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE USER IF NOT EXISTS db_writer IDENTIFIED BY 'P@ssw0rd';
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE USER IF NOT EXISTS db_admin IDENTIFIED BY 'P@ssw0rd';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

now we will start GRANT ROLES TO USERS we have created before

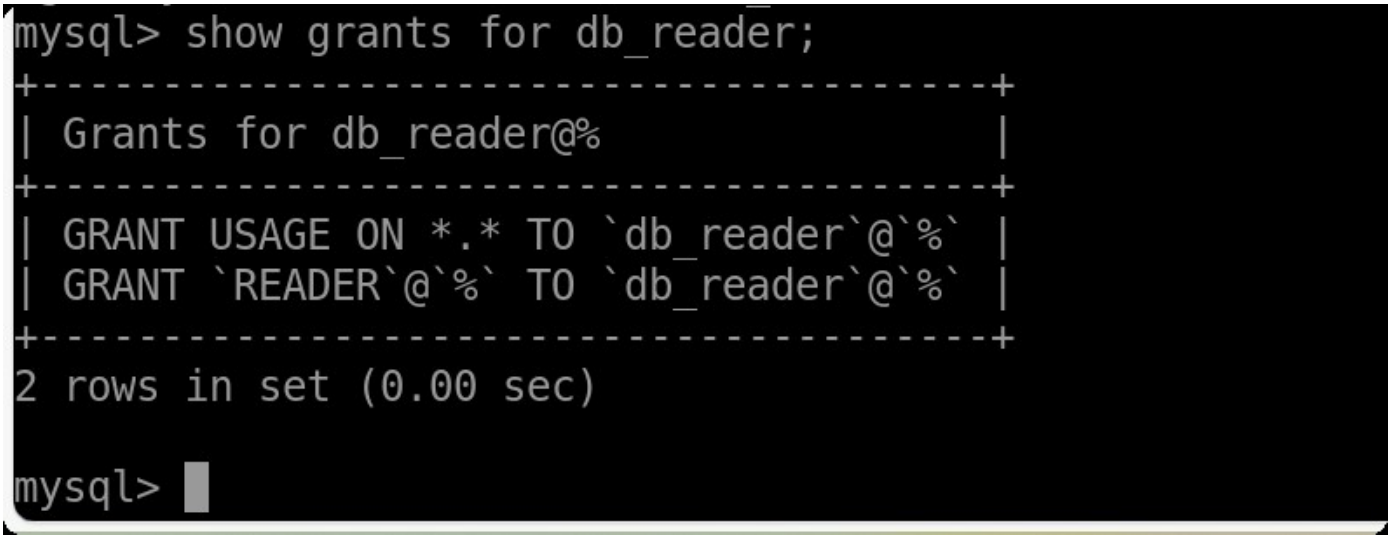
```
GRANT READER TO db_reader;
GRANT WRITER to db_writer;
GRANT ADMIN to db_admin;
```



finally we have to run `FLUSH PRIVILEGES;` to make grants table recreated with the new privileges we have assinged



now let's confirm the privileges of our user using `SHOW GRANTS FOR [username];`



6- MySQL Server Configuration

MySQL Default Configuration File

- **option Files**
  - also called as **Configuration Files**
  - most of MySQL programs can read the **start-up options** from the option files also know as configuration files
  - meaning programs such as `mysqld` , `mysqladmin` , `mysqlimport` when they start they takes some options from option files , those options can be defined on command line or they can be put and saved in a configuration or option file
  - to retrieve which default options a MySQL program we use the following syntax `program --verbose --help` `mysql --verbose --help`

```
[dba@mysqlPerconaSTG ~]$ mysql --verbose --help
mysql  Ver 8.0.36-28 for Linux on x86_64 (Percona Server (GPL), Release 28, Revision 47601f19)
Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Usage: mysql [OPTIONS] [database]
  -?, --help          Display this help and exit.
  -I, --help          Synonym for -?
  --auto-rehash       Enable automatic rehashing. One doesn't need to use
                      'rehash' to get table and field completion, but startup
                      and reconnecting may take a longer time. Disable with
                      --disable-auto-rehash.
                      (Defaults to on; use --skip-auto-rehash to disable.)
  -A, --no-auto-rehash
                      No automatic rehashing. One has to use 'rehash' to get
```

```
--defaults-extra-file=# Read this file after the global files are read.
--defaults-group-suffix=#
                      Also read groups with concat(group, suffix)
--login-path=#        Read this path from the login file.

Variables (--variable-name=value)
and boolean options {FALSE|TRUE}         Value (after reading options)
---------------------------------------- ------------------------------
auto-rehash                              TRUE
auto-vertical-output                     FALSE
bind-address                             (No default value)
binary-as-hex                            FALSE
character-sets-dir                       (No default value)
column-type-info                         FALSE
comments                                 FALSE
compress                                 FALSE
database                                 (No default value)
default-character-set                    auto
delimiter                                ;
enable-cleartext-plugin                  FALSE
vertical                                 FALSE
force                                    FALSE
histignore                               (No default value)
named-commands                           FALSE
ignore-spaces                            FALSE
init-command                             (No default value)
local-infile                             FALSE
no-beep                                  FALSE
host                                     (No default value)
```

```
  --zstd-compression-level=#
                      Use this compression level in the client/server protocol,
                      in case --compression-algorithms=zstd. Valid range is
                      between 1 and 22, inclusive. Default is 3.
  --load-data-local-dir=name
                      Directory path safe for LOAD DATA LOCAL INFILE to read
                      from.
  --fido-register-factor=name
                      Specifies authentication factor, for which registration
                      needs to be done.
  --authentication-oci-client-config-profile=name
                      Specifies the configuration profile whose configuration
                      options are to be read from the OCI configuration file.
                      Default is DEFAULT.
  --oci-config-file=name
                      Specifies the location of the OCI configuration file.
                      Default for Linux is ~/.oci/config and %HOME/.oci/config
                      on Windows.

Default options are read from the following files in the given order:
/etc/my.cnf /etc/mysql/my.cnf /usr/etc/my.cnf ~/.my.cnf
The following groups are read: mysql client
The following options may be given as the first argument:
--print-defaults      Print the program argument list and exit.
--no-defaults         Don't read default options from any option file,
                      except for login file.
--defaults-file=#     Only read default options from the given file #.
--defaults-extra-file=# Read this file after the global files are read.
--defaults-group-suffix=#
```

  - any program starts with `-no-default` option read no option file other than `.mylogin.cnf`

- **Option Files Format**
  - option files are plain-text files - expect `.mysql.cnf` , encrypted by `mysql_config_editor`
- **Option Files Processing Order**
  - Global Options : `/etc/my.cnf` `etc/mysql/my.cnf`
  - Server Only Options : `$MYSQL_HOME/my.cnf`
  - User-Specific Options : `¬/.my.cnf`

- Client-only Options :`¬/.mylogin.cnf`

Locate Default Option File

reference link

we will start by locating `mysqld` options file using the below command

`mysqld --verbose --help`

on the beginning of the page we should fine details of the program and the option files

to make the view better we can pipe line with `less` and it will show you the output in pages that you can between them using the space bar

`mysqld --verbose --help | less`

```
utility-user-schema-access                              (No default value)
validate-config                                         FALSE
validate-user-plugins                                   TRUE
verbose                                                 TRUE
version-suffix
wait-timeout                                            28800
windowing-use-high-precision                            TRUE
xa-detach-on-prepare                                    TRUE

To see what values a running MySQL server is using, type
'mysqladmin variables' instead of 'mysqld --verbose --help'.
[dba@mysqlPerconaSTG ~]$ mysqld --verbose --help | less
mysqld  Ver 8.0.36-28 for Linux on x86_64 (Percona Server (GPL), Release 28, Revision 47601f19)
BuildID[sha1]=ac51aa1c6d32c6256b61231b6ae205a3d133c39c
Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Starts the MySQL database server.

Usage: mysqld [OPTIONS]

Default options are read from the following files in the given order:
/etc/my.cnf /etc/mysql/my.cnf /usr/etc/my.cnf ~/.my.cnf
The following groups are read: mysql_cluster mysqld server mysqld-8.0
The following options may be given as the first argument:
```

to exist out we simply jus press "q' button .

in the above image you can see that `mysqld` read option from given file in order

one of the files not present in the sequence , it will go to the other option file

let's check if the file are there in the OS

```
mysqld  Ver 8.0.36-28 for Linux on x86_64 (Percona Server (GPL), Release 28, Revision 47601f19)
[dba@mysqlPerconaSTG ~]$ ls /etc/
adjtime                 environment     libnl                     pam.d                 shells
aliases                 ethertypes      libreport                 passwd                skel
alternatives            exports         libssh                    passwd-               smartmontools
anacrontab              favicon.png     libuser.conf              pinforc               sos
at.deny                 filesystems     locale.conf               pkcs11                ssh
audit                   firewalld       localtime                 pki                   ssl
authselect              fonts           login.defs                plymouth              sssd
bash_completion.d       fprintd.conf    logrotate.conf            pm                    subgid
bashrc                  fstab           logrotate.d               polkit-1              subgid-
bindresvport.blacklist  fuse.conf       lsm                       popt.d                subuid
binfmt.d                gcrypt          lvm                       printcap              subuid-
centos-release          gnupg           machine-id                profile               sudo.conf
chkconfig.d             GREP_COLORS     magic                     profile.d             sudoers
chrony.conf             groff           mailcap                   protocols             sudoers.d
chrony.keys             group           makedumpfile.conf.sample  qemu-ga               sudo-ldap.conf
cifs-utils              group-          man_db.conf               qemu-kvm              sysconfig
cni                     grub2.cfg       mcelog                    rc0.d                 sysctl.conf
cockpit                 grub.d          microcode_ctl             rc1.d                 sysctl.d
containers              gshadow         mime.types                rc2.d                 systemd
cron.d                  gshadow-        mke2fs.conf               rc3.d                 system-release
cron.daily              gss             modprobe.d                rc4.d                 system-release-cpe
cron.deny               host.conf       modules-load.d            rc5.d                 tcsd.conf
cron.hourly             hostname        motd                      rc6.d                 terminfo
cron.monthly            hosts           motd.d                    rc.d                  tmpfiles.d
crontab                 idmapd.conf     mtab                      rc.local              trusted-key.key
cron.weekly             init.d          multipath                 redhat-release        tuned
crypto-policies         inittab         my.cnf                    resolv.conf           udev
```

```
[dba@mysqlPerconaSTG ~]$ ls /etc/my.cnf
my.cnf    my.cnf.d/
[dba@mysqlPerconaSTG ~]$ ls /etc/my.cnf
```

```
[dba@mysqlPerconaSTG ~]$ ls /usr/
bin  games  include  lib  lib64  libexec  local  sbin  share  src  tmp
[dba@mysqlPerconaSTG ~]$
```

we are able to find only `/etc/my.cnf` remaining are not there , for `~/.my.cnf` as motioned before is User-Specific Options
so `mysqld` start-up using options file `/etc/my.cnf`

MySQL Option/Configuration File Syntax

- any comment in option file start with `#` sign
- there are option groups in the sometime is called **stanza**
- when editing option file we provide the option name follow by `=` then value `option = value`
- space is allowed either side between `option` and ``=`and`value`
- value can be without quote , single-quote , double-quote , recommended that any value with some pound sign or any special to be in double-quote
- any option that may be given at command-line can be given in the option file as well
- for example for starting mysqld we are saying `mysqld --server-id` in the command line , in the option file it will be `server-id = value`
- variable cannot have `-` the have `_` and they will show up in system variables as an option
- **option IS NOT variable **
- **option groups**

- there could be many Group in the option file , some of common are `mysqld,mysqladmin,client,mysql,server`
- **note** client option group is read by all client programs expect mysqld

Re-Write Default Option File

first thing I will check variables for instance server-id

`show variables like '%server%';`

```
Your MySQL connection id is 10
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show variables like '%server%';
+--------------------------------+--------------------------------------+
| Variable_name                  | Value                                |
+--------------------------------+--------------------------------------+
| character_set_server           | utf8mb4                              |
| collation_server               | utf8mb4_0900_ai_ci                   |
| immediate_server_version       | 999999                               |
| innodb_dedicated_server        | OFF                                  |
| innodb_ft_server_stopword_table |                                      |
| original_server_version        | 999999                               |
| server_id                      | 1                                    |
| server_id_bits                 | 32                                   |
| server_uuid                    | b8a78760-efff-11ee-809e-6aa8d1f44170 |
+--------------------------------+--------------------------------------+
9 rows in set (0.04 sec)

mysql>
```

you can see the variable name is `server_id` bit if you came back to command line and type `mysqld --verbose --help | less` you will find option is written as `--server-id` so this is considered as option not variable

```
                      secondary storage engine. Only statements that have a
                      cost estimate higher than this value will be attempted
                      executed in a secondary storage engine.
  --secure-file-priv=name
                      Limit LOAD DATA, SELECT ... OUTFILE, and LOAD_FILE() to
                      files within specified directory
  --secure-log-path=name
                      Limit location of general log, slow log and buffered
                      error logwithin specified directory
  --select-into-buffer-size[=#]
                      Buffer size for SELECT INTO OUTFILE/DUMPFILE.
  --select-into-disk-sync
                      Synchronize flushed buffer with disk for SELECT INTO
                      OUTFILE/DUMPFILE.
  --select-into-disk-sync-delay[=#]
                      The delay in milliseconds after each buffer sync for
                      SELECT INTO OUTFILE/DUMPFILE. Requires
                      select_into_sync_disk = ON.
  --server-id=#       Uniquely identifies the server instance in the community
                      of replication partners
  --server-id-bits=#  Set number of significant bits in server-id
  --session-track-gtids=name
                      Controls the amount of global transaction ids to be
                      included in the response packet sent by the
                      server.(Default: OFF).
  --session-track-schema
                      Track changes to the 'default schema'.
                      (Defaults to on; use --skip-session-track-schema to disable.)
:
```

now we will add option for `server-id` in `my.cnf`

`sudo vi /etc/my.cnf` and then add at end of file `server-id = 3`

```
[dba@mysqlPerconaSTG ~]$ sudo vi /etc/my.cnf
```

```
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
-- INSERT --
```

now restart mysql using `systemct restart mysqld`

```
Your MySQL connection id is 8
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show variables like '%server%';
+---------------------------------+--------------------------------------+
| Variable_name                   | Value                                |
+---------------------------------+--------------------------------------+
| character_set_server            | utf8mb4                              |
| collation_server                | utf8mb4_0900_ai_ci                   |
| immediate_server_version        | 999999                               |
| innodb_dedicated_server         | OFF                                  |
| innodb_ft_server_stopword_table |                                      |
| original_server_version         | 999999                               |
| server_id                       | 3                                    |
| server_id_bits                  | 32                                   |
| server_uuid                     | b8a78760-efff-11ee-809e-6aa8d1f44170 |
+---------------------------------+--------------------------------------+
9 rows in set (0.02 sec)

mysql>
```

Variable or Option in Option File?

- In the option file, you can specify variables that appear when using the command `show variables like '%server%';` in MySQL.
- MySQL recommends that any option appearing when running `mysqld --verbose --help | less` can also be used without the need to log in to MySQL and retrieve system variables, which could be time-consuming.
- Essentially, you can use both the variable name and the option name interchangeably in the option file.

Changing Default Option Files Location

reference link

in this section we will change the default option file location using the following steps

1. create directory `/etc/mysql`
2. copy `/etc/my.cnf` to `/etc/mysql/my.cnf`
3. rename the existing `/etc/my.cnf` to `/etc/my.cnf.old`
4. restart mysqld and verify
   remember that mysqld program read the following diretcory `/etc/my.cnf /etc/mysql/my.cnf /usr/etc/my.cnf ~/.my.cnf` you can verfiy this uisng the command `mysqld --verbose --help | less`

```
mysqld  Ver 8.0.36-28 for Linux on x86_64 (Percona Server (GPL), Release 28, Revision 47601f19)
BuildID[sha1]=ac51aa1c6d32c6256b61231b6ae205a3d133c39c
Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Starts the MySQL database server.

Usage: mysqld [OPTIONS]

Default options are read from the following files in the given order:
/etc/my.cnf /etc/mysql/my.cnf /usr/etc/my.cnf ~/.my.cnf
The following groups are read: mysql_cluster mysqld server mysqld-8.0
The following options may be given as the first argument:
--print-defaults        Print the program argument list and exit.
--no-defaults           Don't read default options from any option file,
                        except for login file.
--defaults-file=#       Only read default options from the given file #.
--defaults-extra-file=# Read this file after the global files are read.
--defaults-group-suffix=#
                        Also read groups with concat(group, suffix)
--login-path=#          Read this path from the login file.

  --abort-slave-event-count=#
                    Option used by mysql-test for debugging and testing of
:
```

1. create directory `/etc/mysql`

```
sudo mkdir /etc/mysql
```

```
[dba@mysqlPerconaSTG ~]$ sudo mkdir /etc/mysql
[sudo] password for dba:
[dba@mysqlPerconaSTG ~]$ ls /etc/ | grep mysql
mysql
[dba@mysqlPerconaSTG ~]$
```

2. copy `/etc/my.cnf` to `/etc/mysql/my.cnf`

```
sudo cp /etc/my.cnf /etc/mysql/my.cnf
```

```
[dba@mysqlPerconaSTG ~]$ sudo cp /etc/my.cnf /etc/mysql/my.cnf
[dba@mysqlPerconaSTG ~]$ ls /etc/mysql/
my.cnf
[dba@mysqlPerconaSTG ~]$
```

3. rename the existing `/etc/my.cnf` to `/etc/my.cnf.old`

```
sudo mv /etc/my.cnf  /etc/my.cnf.old
```

```
[dba@mysqlPerconaSTG ~]$ sudo mv /etc/my.cnf  /etc/my.cnf.old
[dba@mysqlPerconaSTG ~]$ ls /etc/ ^C
[dba@mysqlPerconaSTG ~]$ ls /etc/ | grep my.cnf.ld
[dba@mysqlPerconaSTG ~]$ ls /etc/ | grep my.cnf.old
my.cnf.old
[dba@mysqlPerconaSTG ~]$
```

4. restart mysqld and verify

```
sudo systemct   restart mysqld
sudo systemct   status  mysqld
```

```
[dba@mysqlPerconaSTG ~]$ sudo systemctl restart mysqld
[dba@mysqlPerconaSTG ~]$ sudo systemctl status  mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2024-04-12 01:10:55 +03; 7s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 44048 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 44083 (mysqld)
   Status: "Server is operational"
    Tasks: 39 (limit: 24168)
   Memory: 402.0M
   CGroup: /system.slice/mysqld.service
           └─44083 /usr/sbin/mysqld

Apr 12 01:10:51 mysqlPerconaSTG systemd[1]: mysqld.service: Succeeded.
Apr 12 01:10:51 mysqlPerconaSTG systemd[1]: Stopped MySQL Server.
Apr 12 01:10:51 mysqlPerconaSTG systemd[1]: Starting MySQL Server...
Apr 12 01:10:55 mysqlPerconaSTG systemd[1]: Started MySQL Server.
[dba@mysqlPerconaSTG ~]$
```

STRACE & LSOF With MySQL

This section addresses essential aspects of troubleshooting MySQL server startup issues. When diagnosing why `mysqld` failed to start, it's crucial to examine:

- The option file that was read during startup. To verify this, you can stop the MySQL service and then initiate its startup with the `strace` utility.
- The files accessed by MySQL during startup. This can be determined by utilizing the `lsof` command.

START MYSQLD WITH STRACE

When initiating `strace` to observe how `mysqld` starts and the files it attempts to access, the process might fail due to MySQL being managed by `systemd`. However, despite this failure, the output from `strace` can still provide valuable insights, including the location of the log file and the option file being accessed.

```
sudo systemctl stop mysqld
sudo strace mysqld
```

```
[dba@mysqlPerconaSTG ~]$
[dba@mysqlPerconaSTG ~]$ sudo systemctl stop mysqld
```

`sudo strace mysqld`

```
brk(0x7595000)                          = 0x7595000
brk(NULL)                               = 0x7595000
unlink("/mysqldata/mysql/mysqld_tmp_file_case_insensitive_test.LOWER-TEST") = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/mysqldata/mysql/mysqld_tmp_file_case_insensitive_test.lower-test", O_RDWR|O_CREAT, 0666) = 3
lstat("/mysqldata", {st_mode=S_IFDIR|0755, st_size=19, ...}) = 0
lstat("/mysqldata/mysql", {st_mode=S_IFDIR|0755, st_size=4096, ...}) = 0
close(3)                                = 0
stat("/mysqldata/mysql/mysqld_tmp_file_case_insensitive_test.LOWER-TEST", 0x7ffef717a210) = -1 ENOENT (No such file or directory)
lstat("/mysqldata", {st_mode=S_IFDIR|0755, st_size=19, ...}) = 0
lstat("/mysqldata/mysql", {st_mode=S_IFDIR|0755, st_size=4096, ...}) = 0
unlink("/mysqldata/mysql/mysqld_tmp_file_case_insensitive_test.lower-test") = 0
geteuid()                               = 0
write(2, "2024-04-11T22:23:28.388518Z 0 [W"..., 2162024-04-11T22:23:28.388518Z 0 [Warning] [MY-010097] [Server] Insecure configura
tion for --secure-log-path: Current value does not restrict location of generated files. Consider setting it to a valid, non-empty
 path.
) = 216
write(2, "2024-04-11T22:23:28.388873Z 0 [S"..., 1222024-04-11T22:23:28.388873Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (m
ysqld 8.0.36-28) starting as process 44149
) = 122
write(2, "2024-04-11T22:23:28.400297Z 0 [E"..., 1562024-04-11T22:23:28.400297Z 0 [ERROR] [MY-010123] [Server] Fatal error: Please
read "Security" section of the manual to find out how to run mysqld as root!
) = 156
futex(0x484fbb8, FUTEX_WAKE_PRIVATE, 2147483647) = 0
munmap(0x7f771976c000, 421888)          = 0
write(2, "2024-04-11T22:23:28.401146Z 0 [E"..., 682024-04-11T22:23:28.401146Z 0 [ERROR] [MY-010119] [Server] Aborting
) = 68
write(2, "2024-04-11T22:23:28.401718Z 0 [S"..., 1692024-04-11T22:23:28.401718Z 0 [System] [MY-010910] [Server] /usr/sbin/mysqld: S
hutdown complete (mysqld 8.0.36-28)  Percona Server (GPL), Release 28, Revision 47601f19.
) = 169
```

scroll up and look for `stat` and look for `stat` regarding `my.cnf`

LSOF With MySQL

now we will use the utility `LSOF` to see which files mysqld has opened

`sudo lsof -u mysql`

```
[dba@mysqlPerconaSTG ~]$ sudo lsof -u mysql
COMMAND   PID  USER   FD      TYPE             DEVICE SIZE/OFF      NODE NAME
mysqld  44187 mysql  cwd       DIR              8,17     4096       131 /mysqldata/mysql
mysqld  44187 mysql  rtd       DIR              253,0     4096       128 /
mysqld  44187 mysql  txt       REG              253,0 66373040     53299 /usr/sbin/mysqld
mysqld  44187 mysql  mem       REG              253,0    59464  67164345 /usr/lib64/mysql/plugin/component_validate_password.so
mysqld  44187 mysql  DEL       REG               0,17            167098 /[aio]
mysqld  44187 mysql  DEL       REG               0,17            167097 /[aio]
mysqld  44187 mysql  DEL       REG               0,17            167096 /[aio]
mysqld  44187 mysql  DEL       REG               0,17            167095 /[aio]
mysqld  44187 mysql  DEL       REG               0,17            167094 /[aio]
mysqld  44187 mysql  DEL       REG               0,17            167093 /[aio]
mysqld  44187 mysql  DEL       REG               0,17            167092 /[aio]
mysqld  44187 mysql  DEL       REG               0,17            167091 /[aio]
mysqld  44187 mysql  mem       REG              253,0    99656      2029 /usr/lib64/libz.so.1.2.11
mysqld  44187 mysql  mem       REG              253,0  2089936      1709 /usr/lib64/libc-2.28.so
mysqld  44187 mysql  mem       REG              253,0    99664       144 /usr/lib64/libgcc_s-8-20210514.so.1
mysqld  44187 mysql  mem       REG              253,0  1598848      1713 /usr/lib64/libm-2.28.so
mysqld  44187 mysql  mem       REG              253,0  1661448      2085 /usr/lib64/libstdc++.so.6.0.25
mysqld  44187 mysql  mem       REG              253,0    19128      1711 /usr/lib64/libdl-2.28.so
mysqld  44187 mysql  mem       REG              253,0    51392      2800 /usr/lib64/libnuma.so.1.0.0
mysqld  44187 mysql  mem       REG              253,0    12264      2389 /usr/lib64/libaio.so.1.0.1
mysqld  44187 mysql  mem       REG              253,0  3087888      3532 /usr/lib64/libcrypto.so.1.1.1k
mysqld  44187 mysql  mem       REG              253,0   615728      3534 /usr/lib64/libssl.so.1.1.1k
mysqld  44187 mysql  mem       REG              253,0    42744      1727 /usr/lib64/librt-2.28.so
mysqld  44187 mysql  mem       REG              253,0   149976      1723 /usr/lib64/libpthread-2.28.so
```

Option File Inclusions

When configuring MySQL, you have two options to include additional option files:

- Use `!include = file` to specify a single additional configuration file.
- Use `!includedir = directory` to include all configuration files within a specified directory.

Syntax:

- To include a single file: `!include = file_path`
- To include files from a directory: `!includedir = directory_path`

Example:

- Include a single file: `!include /home/bob/bob-options.cnf`
- Include all `.cnf` files from a directory: `!includedir /home/bob`

Note:

- All included files must have a `.cnf` extension to be recognized as configuration files.

using Option File Inclusions

- we will start by creating directory called `/etc/percona`
  `sudo mkdir /etc/percona`

```
[dba@mysqlPerconaSTG ~]$ sudo mkdir /etc/percona
[sudo] password for dba:
[dba@mysqlPerconaSTG ~]$ ls  /etc/  | grep percona
percona
percona-toolkit
[dba@mysqlPerconaSTG ~]$
```

- copy the `/etc/mysql/my.cnf` to `/etc/percona/my.cnf`

  `sudo cp /etc/mysql/my.cnf /etc/percona/my.cnf`

```
[dba@mysqlPerconaSTG ~]$ sudo cp /etc/mysql/my.cnf  /etc/percona/my.cnf
[dba@mysqlPerconaSTG ~]$ ls /etc/percona/
my.cnf
[dba@mysqlPerconaSTG ~]$
```

- edit `/etc/mysql/my.cnf` and add `!includedir`

  `sudo vi /etc/mysql/my.cnf`

```
[dba@mysqlPerconaSTG ~]$ sudo vi /etc/mysql/my.cnf
```

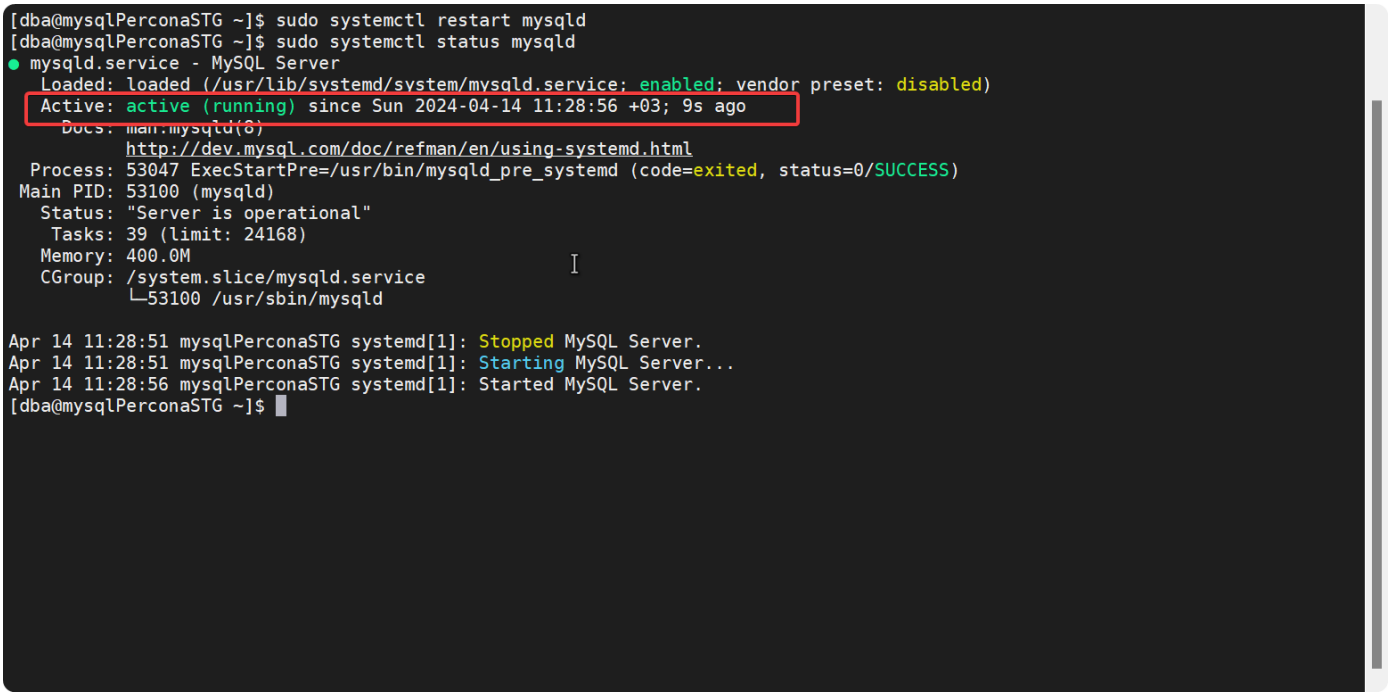
and remove every content of the file

```
# Percona Server template configuration
#
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
!includedir /etc/percona/

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/etc/mysql/my.cnf" 8L, 205C
```

and only add the `!includedir /etc/percona`

- restart mysqld and verify

  `sudo systemctl restart mysqld sudo systemct status mysqld`



MySQL Data Directory

When MySQL is installed:

- It defaults its data directory path to `/var/lib/mysql`.
- A system user named **mysql** is created, which uses the data directory as its home.
- Ownership of the data directory is assigned to the **mysql** user.
- It is advisable to place the data directory on its own dedicated file system.
- The location of the data directory can be customized by setting the `datadir` variable in the `my.cnf` file.
- if we change the location of data directory then we need to make appropriate adjustments in `my.cnf` file with `datadir` as an option

Move Data_DIR MySQL Data Directory

To relocate the MySQL data directory to a separate filesystem, follow these steps:

1. Shut down the MySQL server to prevent any data corruption or access issues.
2. Create the new directory on the separate filesystem and change its ownership to the MySQL user.
3. Update the `my.cnf` configuration file to set the new data directory location using the `datadir` variable.
4. Restart the MySQL server to apply the changes and start using the new data directory.

note : for runinng the below command you need root user or sudo privilages

1- Shut down the MySQL server.

```
systemctl        stop mysqld

systemctl status mysqld
```

```
[root@Mysqlcom-RHEL-STG /]# systemctl stop mysqld.service
[root@Mysqlcom-RHEL-STG /]# systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: inactive (dead) since Sun 2024-04-14 15:38:46 +03; 9s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 52810 ExecStart=/usr/sbin/mysqld $MYSQLD_OPTS (code=exited, status=0/SUCCESS)
  Process: 52781 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 52810 (code=exited, status=0/SUCCESS)
   Status: "Server shutdown complete"

Apr 05 01:26:00 Mysqlcom-RHEL-STG systemd[1]: Starting MySQL Server...
Apr 05 01:26:05 Mysqlcom-RHEL-STG systemd[1]: Started MySQL Server.
Apr 14 15:38:44 Mysqlcom-RHEL-STG systemd[1]: Stopping MySQL Server...
Apr 14 15:38:46 Mysqlcom-RHEL-STG systemd[1]: mysqld.service: Succeeded.
Apr 14 15:38:46 Mysqlcom-RHEL-STG systemd[1]: Stopped MySQL Server.
[root@Mysqlcom-RHEL-STG /]#
```

2- Create the new directory on the separate filesystem and change its ownership to the MySQL user.

```
mkdir mysqldata
chown -R mysql:mysql mysqldata/
```

```
[root@Mysqlcom-RHEL-STG /]# mkdir mysqldata
[root@Mysqlcom-RHEL-STG /]# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  mysqldata  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
[root@Mysqlcom-RHEL-STG /]#
```

```
[root@Mysqlcom-RHEL-STG /]# chown -R mysql:mysql mysqldata/
[root@Mysqlcom-RHEL-STG /]# ll
total 24
lrwxrwxrwx.   1 root  root     7 Oct 11  2021 bin -> usr/bin
dr-xr-xr-x.   5 root  root  4096 Apr  1 20:55 boot
drwxr-xr-x.  20 root  root  3080 Apr 14 15:37 dev
drwxr-xr-x. 106 root  root  8192 Apr 11 00:40 etc
drwxr-xr-x.   3 root  root    17 Mar 24 22:21 home
lrwxrwxrwx.   1 root  root     7 Oct 11  2021 lib -> usr/lib
lrwxrwxrwx.   1 root  root     9 Oct 11  2021 lib64 -> usr/lib64
drwxr-xr-x.   2 root  root     6 Oct 11  2021 media
drwxr-xr-x.   2 root  root     6 Oct 11  2021 mnt
drwxr-xr-x.   2 mysql mysql    6 Apr 14 15:38 mysqldata
drwxr-xr-x.   2 root  root     6 Oct 11  2021 opt
dr-xr-xr-x. 137 root  root     0 Mar 24 23:56 proc
dr-xr-x---.   3 root  root  4096 Apr 14 15:37 root
drwxr-xr-x.  38 root  root  1040 Apr  4 23:28 run
lrwxrwxrwx.   1 root  root     8 Oct 11  2021 sbin -> usr/sbin
drwxr-xr-x.   2 root  root     6 Oct 11  2021 srv
dr-xr-xr-x.  13 root  root     0 Mar 24 23:56 sys
drwxrwxrwt.   3 root  root    85 Apr 14 15:43 tmp
drwxr-xr-x.  12 root  root   144 Mar 24 22:12 usr
drwxr-xr-x.  21 root  root  4096 Mar 24 22:24 var
[root@Mysqlcom-RHEL-STG /]#
```

-3 move data file from default path to the new path

default path is located in `/var/lib/mysql`

use `mv` to move data files to the new directory

```
cd /var/lib/mysql
```

```
[root@Mysqlcom-RHEL-STG /]# cd /var/lib/mysql
[root@Mysqlcom-RHEL-STG mysql]# ls
 auto.cnf         ca-key.pem         employees            ibdata1         mysql.ibd           server-cert.pem   undo_002
 binlog.000001    ca.pem            '#ib_16384_0.dblwr'  '#innodb_redo'   performance_schema  server-key.pem
 binlog.000002    client-cert.pem   '#ib_16384_1.dblwr'  '#innodb_temp'   private_key.pem     sys
 binlog.index     client-key.pem     ib_buffer_pool       mysql           public_key.pem      undo_001
[root@Mysqlcom-RHEL-STG mysql]#
```

usng `*` you will basicly telling to move everting on currnt diretcory to the new diretcory

```
mv * /mysqldata/
```

```
[root@Mysqlcom-RHEL-STG mysql]# mv * /mysqldata/
[root@Mysqlcom-RHEL-STG mysql]# ls
[root@Mysqlcom-RHEL-STG mysql]# ls  /mysqldata/
 auto.cnf         ca-key.pem         employees            ibdata1         mysql.ibd           server-cert.pem   undo_002
 binlog.000001    ca.pem            '#ib_16384_0.dblwr'  '#innodb_redo'   performance_schema  server-key.pem
 binlog.000002    client-cert.pem   '#ib_16384_1.dblwr'  '#innodb_temp'   private_key.pem     sys
 binlog.index     client-key.pem     ib_buffer_pool       mysql           public_key.pem      undo_001
[root@Mysqlcom-RHEL-STG mysql]#
```

4- Update the `my.cnf` configuration file to set the new data directory location using the `datadir` variable.

we will update `data_dir` and add the new directory instat of the default one

`vi /etc/my.cnf`

```
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
"/etc/my.cnf" 31L, 1243C
```

```
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

5- Restart the MySQL server to apply the changes and start using the new data directory.

`systemct restart mysqld`

`systemctl status mysqld`

```
[root@Mysqlcom-RHEL-STG ~]# systemctl restart mysqld
[root@Mysqlcom-RHEL-STG ~]# systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Mon 2024-04-15 04:28:55 +03; 33s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 19728 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 19776 (mysqld)
   Status: "Server is operational"
    Tasks: 38 (limit: 24168)
   Memory: 395.6M
   CGroup: /system.slice/mysqld.service
           └─19776 /usr/sbin/mysqld

Apr 15 04:28:49 Mysqlcom-RHEL-STG systemd[1]: Starting MySQL Server...
Apr 15 04:28:55 Mysqlcom-RHEL-STG systemd[1]: Started MySQL Server.
[root@Mysqlcom-RHEL-STG ~]#
```

```
mysql> show variables "datadir";
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL
server version for the right syntax to use near '"datadir"' at line 1
mysql> show global variables "datadir";
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL
server version for the right syntax to use near '"datadir"' at line 1
mysql> show  variables like  "datadir";
+---------------+-------------+
| Variable_name | Value       |
+---------------+-------------+
| datadir       | /mysqldata/ |
+---------------+-------------+
1 row in set (0.02 sec)

mysql>
```

MySQL Binary Logs

- MySQL Binary Logs log any chnages that has happen on databases in special encrypted file called binaray logs
- and the changes in the file are recoreded as database evenets
- binary log are encrypted in special format we cannot read it , in order to read it in text format we will use mysql utility called `mysqlbinlog`
- binary log contain information on how long each DML statement took to complete executing

- binary log play a critical role in the Replication part such as Master Slave setup , which relay heavily on binary logs
- binary log provide **point-in-time** recovery , bring back the database to a data from point of backup
- binary log play critical role in backup , in insist where backup is restored , the events in binary log file after backup was made are re-executed
- by default binary log is enabled in MySQL Server
- default size for binary log is 1GB - the value controlled by variable called `max_binlog_size`
- Retention determine for how long you want to keep binary logs , this is controlled by system variable called `binlog_expire_logs_seconds`

Enable Binary Logging
you have option customize binary log

- to enable binary log we will use system variable called `log_bin` and specify value to be `ON`
- system variable `log_bin_basename` control the naming convention of binary log file , you can change the name of the file as follow `mysqld-bin` , `binlog` , `prod-bin`
- `log_bin_index` is a system variable also give option to give binary log any name but the name must end with `.index` , `binlog.index` , `mysqld-bin.index` , `prod-bin.index`

Disable Binary Logging
to disable bin log for database used for UAT or Test simply add in `my.cnf` file one of the below variables without the need to specify any value
1. `skip-log-bin`
2. `disable-log-bin`

Purging Binary Log Files

we will run some basic command that are important we managing binlog ,
`show binray logs;` will show the list of binary logs files and the files are numbered in a sequential form

```
mysql> show binary logs;
+---------------+-----------+-----------+
| Log_name      | File_size | Encrypted |
+---------------+-----------+-----------+
| binlog.000001 |       851 | No        |
| binlog.000002 |       157 | No        |
+---------------+-----------+-----------+
2 rows in set (0.00 sec)

mysql>
```

`show binlog events in 'binlog.number';` allow us to view the events in one of binlog files

```
mysql> show binlog events in 'binlog.000002';
+---------------+-----+----------------+-----------+-------------+-----------------------------------+
| Log_name      | Pos | Event_type     | Server_id | End_log_pos | Info                              |
+---------------+-----+----------------+-----------+-------------+-----------------------------------+
| binlog.000002 |   4 | Format_desc    |         1 |         126 | Server ver: 8.0.36, Binlog ver: 4 |
| binlog.000002 | 126 | Previous_gtids |         1 |         157 |                                   |
+---------------+-----+----------------+-----------+-------------+-----------------------------------+
2 rows in set (0.01 sec)

mysql> show binlog events in 'binlog.000001';
+---------------+-----+----------------+-----------+-------------+-----------------------------------
---------------------------------------------------------------------------------------------------
----------------+
| Log_name      | Pos | Event_type     | Server_id | End_log_pos | Info

                |
+---------------+-----+----------------+-----------+-------------+-----------------------------------
---------------------------------------------------------------------------------------------------
----------------+
| binlog.000001 |   4 | Format_desc    |         1 |         126 | Server ver: 8.0.36, Binlog ver: 4

                |
| binlog.000001 | 126 | Previous_gtids |         1 |         157 |

                |
| binlog.000001 | 157 | Anonymous_Gtid |         1 |         236 | SET @@SESSION.GTID_NEXT= 'ANONYMOUS'

                |
| binlog.000001 | 236 | Query          |         1 |         479 | ALTER USER 'root'@'localhost' IDENTIFIE
D WITH 'caching_sha2_password' AS '$A$005$kD-&mI\Z
                                                  [\ZY# ;\'n5NJz1RpmJTK9UMOcsSCpwjKFdtpbQ9pgsb3Gbz/RkC' /*
 xid=2 */ |
| binlog.000001 | 479 | Anonymous_Gtid |         1 |         556 | SET @@SESSION.GTID_NEXT= 'ANONYMOUS'

                |
| binlog.000001 | 556 | Query          |         1 |         661 | DROP DATABASE IF EXISTS test

                |
```

the last command is concern when the you filesystem is filled and database transaction halted because lack of space and you want to quickly delete binlog files

`purge binary logs to 'which number';`

this will allow you to delete the binlog file to end file you specify

```
mysql> show binary logs;
+----------+-----------+-----------+
| Log_name | File_size | Encrypted |
+----------+-----------+-----------+
| 1.000001 |       196 | No        |
| 1.000002 |       387 | No        |
| 1.000003 |       180 | No        |
| 1.000004 |       180 | No        |
| 1.000005 |      5292 | No        |
| 1.000006 |  66383539 | No        |
| 1.000007 |  66383539 | No        |
| 1.000008 |  66383540 | No        |
| 1.000009 |  69420440 | No        |
| 1.000010 |   3047678 | No        |
| 1.000011 |      3602 | No        |
| 1.000012 |       180 | No        |
| 1.000013 |       180 | No        |
| 1.000014 |       180 | No        |
| 1.000015 |       180 | No        |
| 1.000016 |       157 | No        |
+----------+-----------+-----------+
16 rows in set (0.01 sec)

mysql> purge binary logs to '1.000008';
Query OK, 0 rows affected (0.04 sec)

mysql> show binary logs;
+----------+-----------+-----------+
| Log_name | File_size | Encrypted |
+----------+-----------+-----------+
| 1.000008 |  66383540 | No        |
| 1.000009 |  69420440 | No        |
| 1.000010 |   3047678 | No        |
| 1.000011 |      3602 | No        |
| 1.000012 |       180 | No        |
| 1.000013 |       180 | No        |
| 1.000014 |       180 | No        |
| 1.000015 |       180 | No        |
| 1.000016 |       157 | No        |
+----------+-----------+-----------+
```

enable and disable binary log

we will demonstrate two action typically done by DBA

1. Disable binary logging
2. Enable Binary logging and moving it to new location to make binary log file stored in separate filesystem

Disable binary logging

use `cat` or `vi` to edit the `my.cnf` file

`vi /etc/my.cnf`

at the end of the file we will add the below variable

`disable-log-bin` or `skip-log-bin`

```
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugi
n
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

disable-log-bin
~
~
~
~
-- INSERT --
```

now we will restart MySQL server and verify if binlog is disabled

```
systemctl restart mysqld
systemct; status mysqld
```

```
[root@Mysqlcom-RHEL-STG /]# systemctl restart  mysqld
[root@Mysqlcom-RHEL-STG /]# systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Mon 2024-04-15 05:39:59 +03; 15s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 20037 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 20081 (mysqld)
   Status: "Server is operational"
    Tasks: 38 (limit: 24168)
   Memory: 393.7M
   CGroup: /system.slice/mysqld.service
           └─20081 /usr/sbin/mysqld

Apr 15 05:39:54 Mysqlcom-RHEL-STG systemd[1]: mysqld.service: Succeeded.
Apr 15 05:39:54 Mysqlcom-RHEL-STG systemd[1]: Stopped MySQL Server.
Apr 15 05:39:54 Mysqlcom-RHEL-STG systemd[1]: Starting MySQL Server...
Apr 15 05:39:59 Mysqlcom-RHEL-STG systemd[1]: Started MySQL Server.
[root@Mysqlcom-RHEL-STG /]#
```

`mysql> show binary logs;`

```
mysql> show binary logs;
ERROR 1381 (HY000): You are not using binary logging
mysql>
```

since we are not using binlog anymore we can remove binlog leftover safely using `rm -f`

```
[root@Mysqlcom-RHEL-STG mysqldata]# rm -f binlog.index
[root@Mysqlcom-RHEL-STG mysqldata]# rm -f binlog.000002
[root@Mysqlcom-RHEL-STG mysqldata]# ls
auto.cnf          client-key.pem       ibdata1         mysql               public_key.pem    undo_001
ca-key.pem       '#ib_16384_0.dblwr'   ibtmp1          mysql.ibd           server-cert.pem   undo_002
ca.pem           '#ib_16384_1.dblwr'  '#innodb_redo'   performance_schema  server-key.pem
client-cert.pem   ib_buffer_pool      '#innodb_temp'   private_key.pem     sys
[root@Mysqlcom-RHEL-STG mysqldata]#
```

we will do MySQL server service restart to see if MySQL is function probably and binlog files are not generating again

`systemctl restart mysqld`

```
[root@Mysqlcom-RHEL-STG mysqldata]# systemctl restart  mysqld
[root@Mysqlcom-RHEL-STG mysqldata]# ls
auto.cnf          client-key.pem       ibdata1         mysql               public_key.pem    undo_001
ca-key.pem       '#ib_16384_0.dblwr'   ibtmp1          mysql.ibd           server-cert.pem   undo_002
ca.pem           '#ib_16384_1.dblwr'  '#innodb_redo'   performance_schema  server-key.pem
client-cert.pem   ib_buffer_pool      '#innodb_temp'   private_key.pem     sys
[root@Mysqlcom-RHEL-STG mysqldata]#
```

Enable Binary logging and moving it to new location

we will now enable binary log but we want to configure binary log in way that binary log file should be stored in separate filesystem

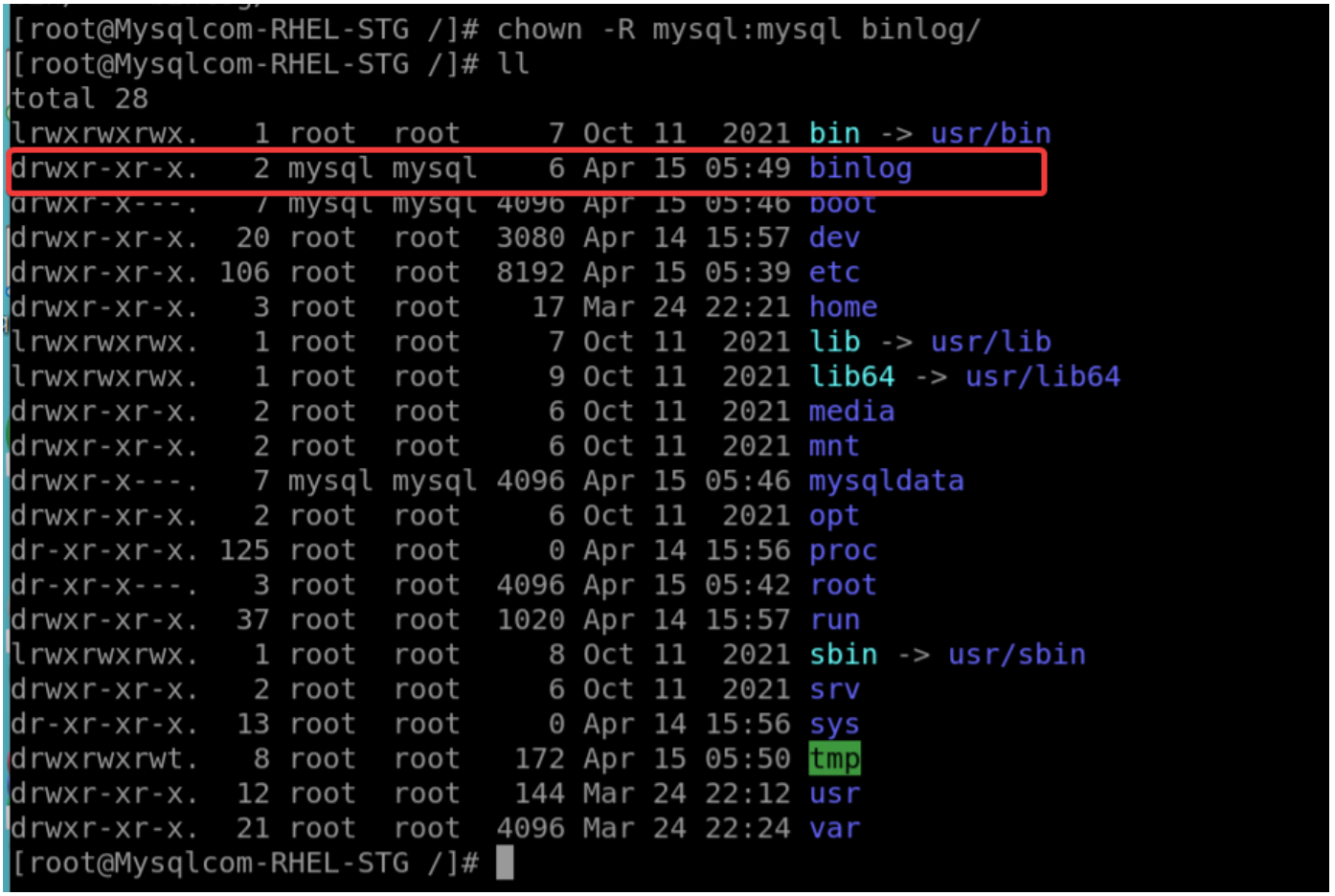
we will create separate filesystem.

`mkdir binlog`

```
[root@Mysqlcom-RHEL-STG /]# mkdir binlog
[root@Mysqlcom-RHEL-STG /]#
```

next we will change the owner to MySQL user

`chown -R mysql:mysql binlog/`



now we will edit `my.cnf` file using either `nano` or `vi` and add option

to get the option we can `mysqld --verbose --help | grep -i bin`

```
binlog-format                                   ROW
binlog-group-commit-sync-delay                  0
binlog-group-commit-sync-no-delay-count         0
binlog-gtid-simple-recovery                     TRUE
binlog-max-flush-queue-time                     0
binlog-order-commits                            TRUE
binlog-rotate-encryption-master-key-at-startup  FALSE
binlog-row-event-max-size                       8192
binlog-row-image                                FULL
binlog-row-metadata                             MINIMAL
binlog-row-value-options
binlog-rows-query-log-events                    FALSE
binlog-stmt-cache-size                          32768
binlog-transaction-compression                  FALSE
binlog-transaction-compression-level-zstd       3
binlog-transaction-dependency-history-size      25000
binlog-transaction-dependency-tracking          COMMIT_ORDER
character-set-filesystem                        binary
innodb-api-enable-binlog                        FALSE
log-bin                                         (No default value)
log-bin-index                                   (No default value)
log-bin-trust-function-creators                 FALSE
log-bin-use-v1-row-events                       FALSE
log-statements-unsafe-for-binlog                TRUE
max-binlog-cache-size                           18446744073709547520
max-binlog-dump-events                          0
max-binlog-size                                 1073741824
max-binlog-stmt-cache-size                      18446744073709547520
mysqlx-bind-address                             *
ndb-log-bin                                     FALSE
ndb-log-binlog-index                            TRUE
ndb-report-thresh-binlog-epoch-slip             10
ndb-report-thresh-binlog-mem-usage              10
ndbinfo                                         OFF
ndbinfo-max-bytes                               0
ndbinfo-max-rows                                10
ndbinfo-show-hidden                             FALSE
relay-log                                       Mysqlcom-RHEL-STG-relay-bin
relay-log-index                                 Mysqlcom-RHEL-STG-relay-bin.index
```

we will use two option `log-bin` & `log-bin-index` copy both option and go to `vi /etc/my.cnf`

remove or comment out `disable-log-bin`

```
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugi
n
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

disable-log-bin

~
~
~
"/etc/my.cnf" 33L, 1256C
```

```
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugi
n
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

# disable-log-bin
~
~
~
~
-- INSERT --
```

add the below line

`log-bin = /binlog/binlog log-bin-index =/binlog/binlog.index`

```
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugi
n
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

# disable-log-bin
log-bin = /binlog/binlog
log-bin-index = /binlog/binlog.index
~
~
```

now we will restart mysql server and check if the files are created in the new directory

```
systemctl restart mysqld


ls /binlog/
```

```
[root@Mysqlcom-RHEL-STG /]# systemctl restart  mysqld
[root@Mysqlcom-RHEL-STG /]# ls /binlog/
binlog.000001  binlog.index
[root@Mysqlcom-RHEL-STG /]#
```

```
mysql> show binary logs;
+---------------+-----------+-----------+
| Log_name      | File_size | Encrypted |
+---------------+-----------+-----------+
| binlog.000001 |       157 | No        |
+---------------+-----------+-----------+
1 row in set (0.00 sec)

mysql>
```

`mysql > show global variables like '%bin%';`

```
| binlog_checksum                          | CRC32                |
| binlog_direct_non_transactional_updates  | OFF                  |
| binlog_encryption                        | OFF                  |
| binlog_error_action                      | ABORT_SERVER         |
| binlog_expire_logs_auto_purge            | ON                   |
| binlog_expire_logs_seconds               | 2592000              |
| binlog_format                            | ROW                  |
| binlog_group_commit_sync_delay           | 0                    |
| binlog_group_commit_sync_no_delay_count  | 0                    |
| binlog_gtid_simple_recovery              | ON                   |
| binlog_max_flush_queue_time              | 0                    |
| binlog_order_commits                     | ON                   |
| binlog_rotate_encryption_master_key_at_startup | OFF            |
| binlog_row_event_max_size                | 8192                 |
| binlog_row_image                         | FULL                 |
| binlog_row_metadata                      | MINIMAL              |
| binlog_row_value_options                 |                      |
| binlog_rows_query_log_events             | OFF                  |
| binlog_stmt_cache_size                   | 32768                |
| binlog_transaction_compression           | OFF                  |
| binlog_transaction_compression_level_zstd | 3                   |
| binlog_transaction_dependency_history_size | 25000              |
| binlog_transaction_dependency_tracking   | COMMIT_ORDER         |
| innodb_api_enable_binlog                 | OFF                  |
| log_bin                                  | ON                   |
| log_bin_basename                         | /binlog/binlog       |
| log_bin_index                            | /binlog/binlog.index |
| log_bin_trust_function_creators          | OFF                  |
| log_bin_use_v1_row_events                | OFF                  |
| log_statements_unsafe_for_binlog         | ON                   |
| max_binlog_cache_size                    | 18446744073709547520 |
| max_binlog_size                          | 1073741824           |
| max_binlog_stmt_cache_size               | 18446744073709547520 |
| mysqlx_bind_address                      | *                    |
| sync_binlog                              | 1                    |
+------------------------------------------+----------------------+
37 rows in set (0.01 sec)

mysql>
```

Binary Logs Retention

we will set a expiry date for our binary log files by setting retention days for binlog

first let's check what is the current retention period for our binlog by runing the below command on MySQL shell

`show variables like '%expire%';`

```
mysql> show variables like '%expire%';
+----------------------------------+---------+
| Variable_name                    | Value   |
+----------------------------------+---------+
| binlog_expire_logs_auto_purge    | ON      |
| binlog_expire_logs_seconds       | 2592000 |
| disconnect_on_expired_password   | ON      |
| expire_logs_days                 | 0       |
+----------------------------------+---------+
```

the value is showing in seconds we will have to convert them to days
which will come to 30 days

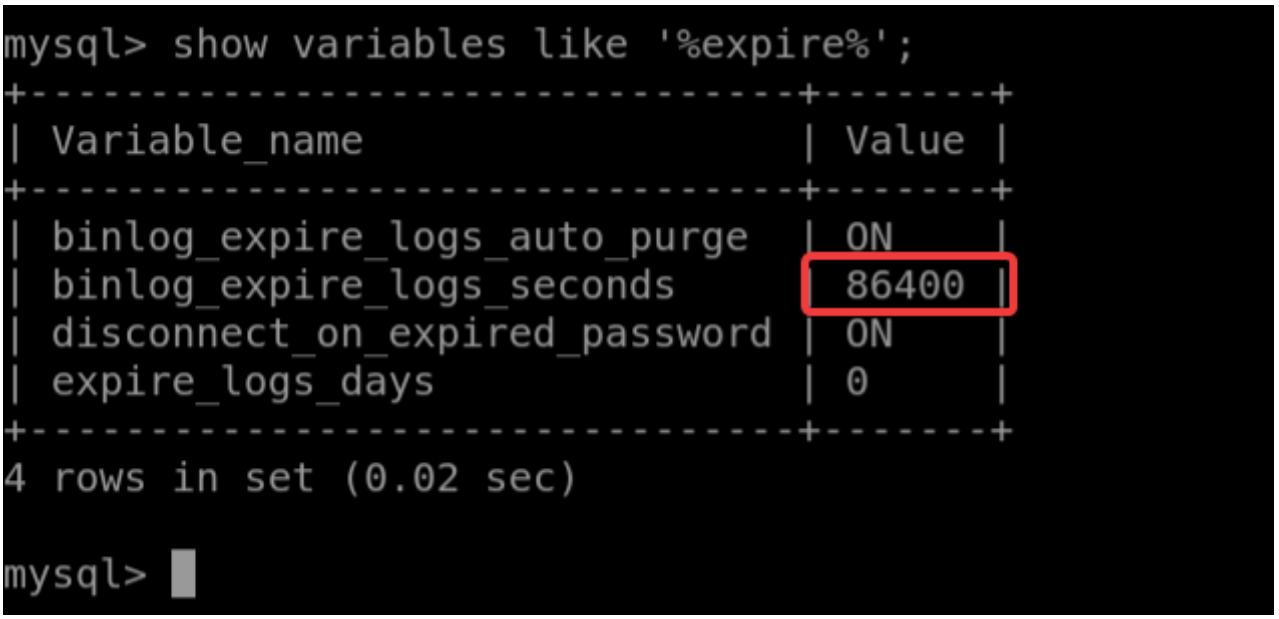
if we want to keep them for 5 days which will come to 86400 second we will edit the option `binlog_expire_logs_seconds` in `my.cnf` file

`vi /etc/my.cnf`

```
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugi
n
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

# disable-log-bin
log-bin = /binlog/binlog
log-bin-index = /binlog/binlog.index

binlog_expire_logs_seconds =  86400
:wq
```

then we will restart MySQL server and check if the retention value is changed

`systemctl restart mysqld`



MySQL Error Log File

- MySQL error log contain the record of MySQL startup and shutdown time
- is also contains diagnostic messages like errors , warning that occurs during startup or shutdown , and while the server is running .
- this file is very important will help to understand why the server is not starting up or not shutting down
- different MySQL components writes log events in the errors log example , system , innodb etc
- `log-error` is the system variable
- default path for error log is `/var/log/mysqld.log`

Error Logging
how to configuring error log falls under three points

1. if we have not given `log-error` variable , we have told where to defiant and where to locate the log file , then MySQL will write the error log to the console
2. if `log-error` is given but we have not given the name of the log file , then MySQL will write the error logs to file called `host_name.err`
3. if `log-error` is given with path and the file name , then MySQL will write the error log on file you specify

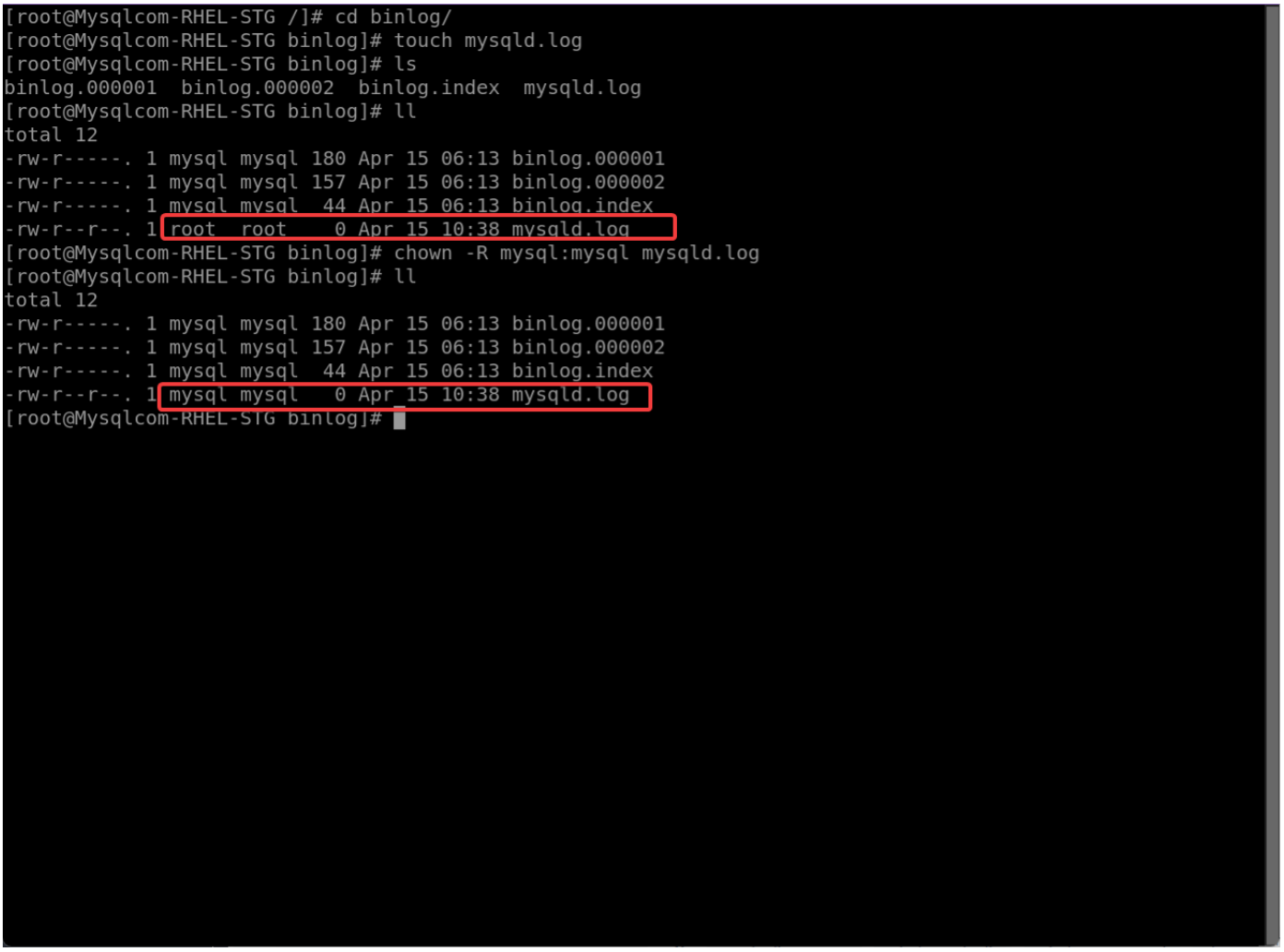
change MySQL Error log location

we will place the error log under directory called `binlog`
we will create empty file using `touch` under `binlog`
`touch mysqld.log`

```
[root@Mysqlcom-RHEL-STG /]# cd binlog/
[root@Mysqlcom-RHEL-STG binlog]# touch mysqld.log
[root@Mysqlcom-RHEL-STG binlog]# ls
binlog.000001  binlog.000002  binlog.index  mysqld.log
[root@Mysqlcom-RHEL-STG binlog]#
```

next ensure that file owner is MySQL system user .

```
chown -R mysql:mysql mysqld.log
```



now open the `my.cnf` file using `nano` or `vi`

`vi /etc/my.cnf`

and add `log-error` option with value of error log path we have created

`log-error = /binlog/mysqld.log`

```
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugi
n
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

# disable-log-bin
log-bin = /binlog/binlog
log-bin-index = /binlog/binlog.index

binlog_expire_logs_seconds =  86400



log-error =  /binlog/mysqld.log
:wq
```

final steps is to restart MySQL services and then check if new error log file have error written on it

`systemctl restart mysqld`

`vi /binlog/mysqld.log`

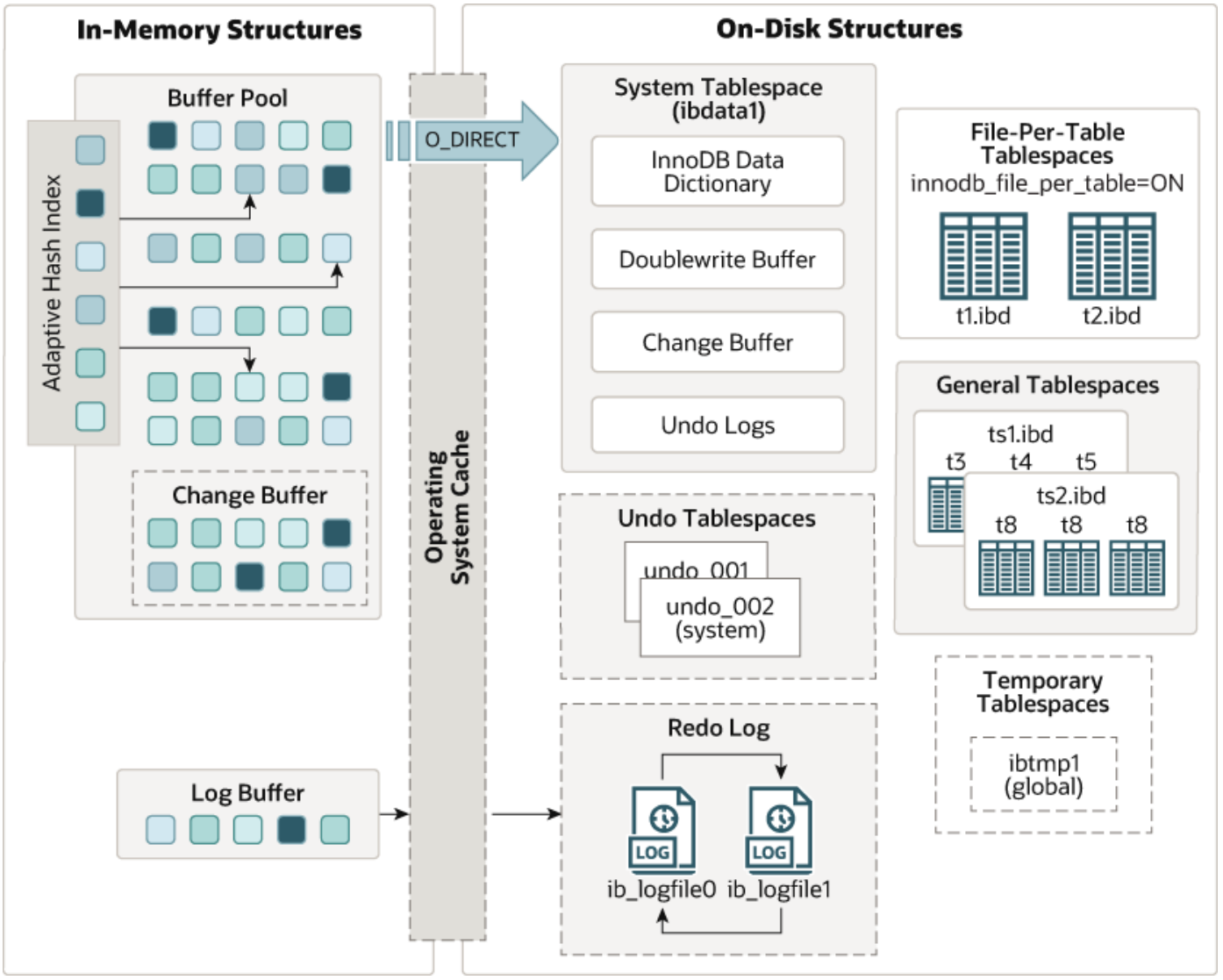
```
[root@Mysqlcom-RHEL-STG /]# cat /binlog/mysqld.log
2024-04-15T09:46:34.990383Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.36) starting a
rocess 21194
2024-04-15T09:46:35.045673Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2024-04-15T09:46:35.912824Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2024-04-15T09:46:36.439632Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
2024-04-15T09:46:36.439746Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TL
Encrypted connections are now supported for this channel.
2024-04-15T09:46:36.479106Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections. Ve
on: '8.0.36'  socket: '/var/lib/mysql/mysql.sock'  port: 3306  MySQL Community Server - GPL.
2024-04-15T09:46:36.479335Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-addres
'::' port: 33060, socket: /var/run/mysqld/mysqlx.sock
[root@Mysqlcom-RHEL-STG /]#
```

7. InnoDB Storage Engine Configuration

InnoDB Storage Engine

- Great general-purpose storage engine that balances high reliability and high performance
- InnoDB Great for general purpose means its not only suitable for OLTP data processing , its also important for **OLAP** (online analytical possessing)
- InnoDB not only give the reliability of the data , it also give you performance because it has a different memory structure , it has in memory storage where it actually caches the most recently accessed data.
- MySQL 8 default engine is InnoDB
- InnoDB support all DML (Data manipulation languages) operation , InnoDB is base on ACID model.
- InnoDB support all type of transaction , commit , rollback, crash-recovery to protect data
- InnoDB provide row level blocking
- InnoDB tables arrange data on disk to optimize query based on PK (Primary Key).
- each InnoDB table has primary key Index also called **clustered index** used to arrange the data physically on Hard Disk
- InnoDB support all referential integrity , it support foreign key constraints .
- **other InnoDB features**
    - compress Data
    - Data Caches both table and index
    - support encrypting of data
    - we can fully have full text search index
    - referential integrity
    - it support replication
    - storage limit for InnoDB is 64 TB
    - support Row Level Locking Granularity.

InnoDB Architecture



- every storage engine is memory hog.
- memory you throw at it , more performance gain
- the faster is the hard disk means better performance for your Database

InnoDB Architecture has two structure

1. **In-Memory Structure**
   - **Buffer Pool** : Area in main Memory where InnoDB caches tables & index as it is accessed
   - **Change Buffer** caches changes to non-cluster index
   - **Adaptive Hash Index** : act like in-memory DB
   - **log Buffer** memory area that holds data to be written to the log files on Disk
2. **In-Disk Structure**
   - System Tablespace (ibdata1)
   - Double write Buffer Files
   - Undo Tablespace
   - Redo Log Files
   - General Tablespace
   - Temporary Tablespaces

InnoDB Buffer Pool

as mentioned before InnoDB Buffer Pool is area in main memory that is given to the storage engine where storage engine utilize this area to caches the table data and index data as it is accessed.

- for that is recommended that MySQL should be installed on dedicated server
- MySQL should be installed side by side by other application which might conflict memory usage with MySQL storage engine
- its always recommended to give 80% of memory to storage engine .

Buffer Pool variables :

- `innodb_buffer_pool_size` this storage value of how mush total memory is given to MySQL
- `innodb_buffer_pool_chunk_size` innodb buffer size allocating works in chunks and you have to determine the chunk size ,by default each chunk is divided of 128 MB
- `innodb_buffer_pool_instance` value will be 1 for small instance
- `innodb_buffer_pool_size` = `innodb_buffer_pool_instance` * `innodb_buffer_pool_chunk_size`

  InnoDB Buffer Pool status

  `show engine innodb status`

InnoDB Buffer Pool demo

to check deferent parameters of innodb including buffer pool we will use the below command `show engine innodb status \G`

```
BUFFER POOL AND MEMORY
----------------------
Total large memory allocated 0
Dictionary memory allocated 479113
Buffer pool size   8191
Buffer pool size, bytes 134201344
Free buffers       7207
Database pages     981
Old database pages 382
Modified db pages  0
Pending reads      0
Pending writes: LRU 0, flush list 0, single page 0
Pages made young 0, not young 0
0.00 youngs/s, 0.00 non-youngs/s
Pages read 838, created 143, written 202
0.00 reads/s, 0.00 creates/s, 0.00 writes/s
No buffer pool page gets since the last printout
Pages read ahead 0.00/s, evicted without access 0.00/s, Random read ahead 0.00/s
LRU len: 981, unzip_LRU len: 0
I/O sum[0]:cur[0], unzip sum[0]:cur[0]
--------------
ROW OPERATIONS
--------------
0 queries inside InnoDB, 0 queries in queue
0 read views open inside InnoDB
0 RW transactions active inside InnoDB
Process ID=53100, Main thread ID=140250845333248 , state=sleeping
Number of rows inserted 0, updated 0, deleted 0, read 0
0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 0.00 reads/s
Number of system rows inserted 8, updated 331, deleted 8, read 4981
0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 0.00 reads/s
----------------------------
END OF INNODB MONITOR OUTPUT
============================

1 row in set (0.01 sec)

ERROR:
No query specified

mysql>
```

the command will show the buffer pool size in allocated
you should check the size in byte

```
BUFFER POOL AND MEMORY
----------------------
Total large memory allocated 0
Dictionary memory allocated 479113
Buffer pool size   8191
Buffer pool size, bytes 134201344
Free buffers       7207
Database pages     981
Old database pages 382
Modified db pages  0
Pending reads      0
Pending writes: LRU 0, flush list 0, single page 0
Pages made young 0, not young 0
0.00 youngs/s, 0.00 non-youngs/s
Pages read 838, created 143, written 202
0.00 reads/s, 0.00 creates/s, 0.00 writes/s
No buffer pool page gets since the last printout
Pages read ahead 0.00/s, evicted without access 0.00/s, Random read ahead 0.00/s
LRU len: 981, unzip_LRU len: 0
I/O sum[0]:cur[0], unzip sum[0]:cur[0]
--------------
ROW OPERATIONS
--------------
0 queries inside InnoDB, 0 queries in queue
0 read views open inside InnoDB
0 RW transactions active inside InnoDB
Process ID=53100, Main thread ID=140250845333248 , state=sleeping
Number of rows inserted 0, updated 0, deleted 0, read 0
0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 0.00 reads/s
Number of system rows inserted 8, updated 331, deleted 8, read 4981
0.00 inserts/s, 0.00 updates/s, 0.00 deletes/s, 0.00 reads/s
----------------------------
END OF INNODB MONITOR OUTPUT
============================

1 row in set (0.01 sec)

ERROR:
No query specified
```

you can check the status also by checking the variable using the below command

```
show variables like 'innodb_buffer_pool%';
```

```
mysql> show variables like 'innodb_buffer_pool%';
+---------------------------------------+----------------+
| Variable_name                         | Value          |
+---------------------------------------+----------------+
| innodb_buffer_pool_chunk_size         | 134217728      |
| innodb_buffer_pool_dump_at_shutdown   | ON             |
| innodb_buffer_pool_dump_now           | OFF            |
| innodb_buffer_pool_dump_pct           | 25             |
| innodb_buffer_pool_filename           | ib_buffer_pool |
| innodb_buffer_pool_in_core_file       | ON             |
| innodb_buffer_pool_instances          | 1              |
| innodb_buffer_pool_load_abort         | OFF            |
| innodb_buffer_pool_load_at_startup    | ON             |
| innodb_buffer_pool_load_now           | OFF            |
| innodb_buffer_pool_size               | 134217728      |
+---------------------------------------+----------------+
11 rows in set (0.04 sec)

mysql>
```

as mentioned below buffer pool is calculated using below

`innodb_buffer_pool_size` = `innodb_buffer_pool_instance` * `innodb_buffer_pool_chunk_size`

to change innodb size we will first grab the option to be added in my.cnf

`mysqld --verbose --help | grep -i innodb`

```
  --innodb-use-native-aio
                      (Defaults to on; use --skip-innodb-use-native-aio to disable.)
  --innodb-validate-tablespace-paths
                      --skip-innodb-validate-tablespace-paths.
                      (Defaults to on; use --skip-innodb-validate-tablespace-paths to disable.)
  --innodb-write-io-threads=#
                      Number of background write I/O threads in InnoDB.
                      [microtime, query_plan, innodb, profiling,
                      Unique prefix is a value assigned by InnoDB cluster to
                      InnoDB ngram full text plugin parser token size in
default-storage-engine                          InnoDB
default-tmp-storage-engine                      InnoDB
innodb-adaptive-flushing                        TRUE
innodb-adaptive-flushing-lwm                    10
innodb-adaptive-hash-index                      TRUE
innodb-adaptive-hash-index-parts                8
innodb-adaptive-max-sleep-delay                 150000
innodb-api-bk-commit-interval                   5
innodb-api-disable-rowlock                      FALSE
innodb-api-enable-binlog                        FALSE
innodb-api-enable-mdl                           FALSE
innodb-api-trx-level                            0
innodb-autoextend-increment                     64
innodb-autoinc-lock-mode                        2
innodb-buffer-pool-chunk-size                   134217728
innodb-buffer-pool-dump-at-shutdown             TRUE
innodb-buffer-pool-dump-now                     FALSE
innodb-buffer-pool-dump-pct                     25
innodb-buffer-pool-filename                     ib_buffer_pool
innodb-buffer-pool-in-core-file                 TRUE
innodb-buffer-pool-instances                    0
innodb-buffer-pool-load-abort                   FALSE
innodb-buffer-pool-load-at-startup              TRUE
innodb-buffer-pool-load-now                     FALSE
innodb-buffer-pool-size                         134217728
innodb-change-buffer-max-size                   25
innodb-change-buffering                         all
innodb-checksum-algorithm                       crc32
innodb-cleaner-lsn-age-factor                   high_checkpoint
innodb-cmp-per-index-enabled                    FALSE
innodb-commit-concurrency                       0
```

we will edit the size in my.cnf file

```
# Percona Server template configuration
#
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
innodb-buffer-pool-size = 256M
-
-- INSERT --
```

you can use `M` as megabyte , or `G` as gigabyte

then restart mysql `systemctl restart mysqld`

```
-
"percona/my.cnf" 39L, 1439C written
[dba@mysqlPerconaSTG etc]$ systemctl restart mysqld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to restart 'mysqld.service'.
Authenticating as: dba
Password:
==== AUTHENTICATION COMPLETE ====
[dba@mysqlPerconaSTG etc]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show variables like 'innodb_buffer_pool%';
+-----------------------------------+----------------+
| Variable_name                     | Value          |
+-----------------------------------+----------------+
| innodb_buffer_pool_chunk_size     | 134217728      |
| innodb_buffer_pool_dump_at_shutdown | ON           |
| innodb_buffer_pool_dump_now       | OFF            |
| innodb_buffer_pool_dump_pct       | 25             |
| innodb_buffer_pool_filename       | ib_buffer_pool |
| innodb_buffer_pool_in_core_file   | ON             |
| innodb_buffer_pool_instances      | 1              |
| innodb_buffer_pool_load_abort     | OFF            |
| innodb_buffer_pool_load_at_startup | ON            |
| innodb_buffer_pool_load_now       | OFF            |
| innodb_buffer_pool_size           | 268435456      |
+-----------------------------------+----------------+
11 rows in set (0.02 sec)

mysql>
```

InnoDB Log Buffer

- allow transaction to run without committing before writing to the log files on disk
- `innodb_log_biffer_size` is system variable for innoDB Log Buffer that define size
- bigger log buffer size can accommodate big transaction to save disk i/o
- default size is 16M

innoDB Log Buffer too small ?

- innoDB_log_wait - Number of times that the log buffer was too small , if you see a lot of log wait means the innoDB Log Buffer size is small
- a wait is required for it to be flushed before continuing .

how to check innoDB Log Buffer and manage it

first we will check innoDB Log Buffer size that is setup on mysql server

by viewing the global variable `innodb_log_biffer_size`

`show global variables like 'innodb_log_buffer_size';`

```
mysql> show global variables like 'innodb_log_buffer_size';
+-------------------------+----------+
| Variable_name           | Value    |
+-------------------------+----------+
| innodb_log_buffer_size  | 16777216 |
+-------------------------+----------+
1 row in set (0.02 sec)

mysql>
```

the size as mentioned is 16M
**to check log wait **

we can use status to check global variable for log wait and it will show how many times log wait happened

`show global status like 'innodb_log_waits'`

```
mysql> show global status like 'innodb_log_waits'
    -> ;
+------------------+-------+
| Variable_name    | Value |
+------------------+-------+
| Innodb_log_waits | 0     |
+------------------+-------+
1 row in set (0.01 sec)

mysql>
```

we can see that log wait has not happened before meaning the default size is ok

InnoDB Flush Method

Whenever data needs to be written to the log buffer, it must be stored in the disk structure and persistently saved to disk files. One method for achieving this is through the "flush method," which transfers data from memory to disk. In Unix or Linux-based systems, this process is often facilitated by the built-in function "**f-sync**." However, the efficiency of this operation can be significantly compromised, particularly when dealing with databases exhibiting poor performance in write-intensive operations like insertions, updates, or deletions.

O_DIRECT OR O_DIRECT_NO_FSYNC

- InnoDB autonomously manages various background tasks, such as flushing dirty pages from the buffer pool and modifying pages not yet written to disk data files.
- The system variable `innodb_flush_method` dictates the chosen flush method.
- By default, **fsync** is the designated flush method, responsible for flushing data, metadata, and log files. However, this default method can lead to double buffering.
- Another flush method, **O_DSYNC**, solely flushes data files but may also incur double buffering.
- **O_DIRECT** exclusively flushes data files, employs fsync without double buffering, and facilitates direct read-write operations to the disk.
- However, **O_DIRECT_NO_FSYNC** shares characteristics with O_DIRECT but bypasses fsync, making it unsuitable for XFS file systems.

which flush method should i use

It is recommended to utilize either **O_DIRECT_NO_FSYNC** or **O_DIRECT** flush methods. Conducting benchmarks for both methods is advisable, and based on the benchmark results, the appropriate flush method can be selected.

Change InnoDB Flush Method

In this section, we'll illustrate the process of altering the InnoDB flush method.
First, let's locate the syntax required for changing the flush method.
`mysqld --verbose --help | grep -i flush`

```
[dba@mysqlPerconaSTG ~]$ mysqld --verbose --help | grep -i flush
  --binlog-max-flush-queue-time=#
                      keep reading transactions before it flush the
  --binlog-skip-flush-commands
                      If set to TRUE, FLUSH <XXX> commands will not be be
  --flush             Flush MyISAM tables to disk between SQL commands
  --flush-time=#      A dedicated thread is created to flush all tables at the
  --innodb-adaptive-flushing
                      Attempt flushing dirty pages to avoid IO bursts at
                      (Defaults to on; use --skip-innodb-adaptive-flushing to disable.)
  --innodb-adaptive-flushing-lwm=#
                      flushing happens.
                      flushing. LEGACY: Original Oracle MySQL formula.
                      innodb_flush_method=O_DIRECT_NO_FSYNC, if supported
                      MySQL handling with single page flushes; BACKOFF: Wait
  --innodb-flush-log-at-timeout[=#]
                      Write and flush logs every (n) second.
  --innodb-flush-log-at-trx-commit[=#]
                      Set to 0 (write and flush once per second), 1 (write and
                      flush at each commit), or 2 (write at commit, flush once
  --innodb-flush-method=name
                      With which method to flush data
  --innodb-flush-neighbors[=#]
                      Set to 0 (don't flush neighbors from buffer pool), 1
                      (flush contiguous neighbors from buffer pool) or 2 (flush
                      neighbors from buffer pool), when flushing a block
  --innodb-flush-sync Allow IO bursts at the checkpoints ignoring io_capacity
                      (Defaults to on; use --skip-innodb-flush-sync to disable.)
  --innodb-flushing-avg-loops=#
                      Number of iterations over which the background flushing
                      which would make InnoDB flush the entire file at once
  --innodb-idle-flush-pct=#
                      Up to what percentage of dirty pages to be flushed when
  --innodb-log-wait-for-flush-spin-hwm=#
                      Maximum value of average log flush time for which
                      spin-delay is used. When flushing takes longer, user
                      threads no longer spin when waiting forflushed redo.
```

The syntax `--innodb-flush-method=name` allows us to specify the desired flush method by replacing `name`. Below, you can see that the current flush method is configured as `fsync`.

```
                      innodb_flush_method=O_DIRECT_NO_FSYNC, if supported
                      MySQL handling with single page flushes; BACKOFF: Wait
  --innodb-flush-log-at-timeout[=#]
                      Write and flush logs every (n) second.
  --innodb-flush-log-at-trx-commit[=#]
                      Set to 0 (write and flush once per second), 1 (write and
                      flush at each commit), or 2 (write at commit, flush once
  --innodb-flush-method=name
                      With which method to flush data
  --innodb-flush-neighbors[=#]
                      Set to 0 (don't flush neighbors from buffer pool), 1
                      (flush contiguous neighbors from buffer pool) or 2 (flush
                      neighbors from buffer pool), when flushing a block
  --innodb-flush-sync Allow IO bursts at the checkpoints ignoring io_capacity
                      (Defaults to on; use --skip-innodb-flush-sync to disable.)
  --innodb-flushing-avg-loops=#
                      Number of iterations over which the background flushing
                      which would make InnoDB flush the entire file at once
  --innodb-idle-flush-pct=#
                      Up to what percentage of dirty pages to be flushed when
  --innodb-log-wait-for-flush-spin-hwm=#
                      Maximum value of average log flush time for which
                      spin-delay is used. When flushing takes longer, user
                      threads no longer spin when waiting forflushed redo.
                      or write/flush of the redo log should be done by each
                      Percentage of dirty pages at which flushing kicks in.
                      Synchronize flushed buffer with disk for SELECT INTO
  --sync-binlog=#     Synchronously flush binary log to disk after every #th
                      write to the file. Use 0 to disable synchronous flushing
  --sync-relay-log=#  Synchronously flush relay log to disk after every #th
                      event. Use 0 to disable synchronous flushing
                      Synchronously flush relay log info to disk after every
                      #th transaction. Use 0 to disable synchronous flushing.
binlog-max-flush-queue-time                     0
binlog-skip-flush-commands                      FALSE
flush                                           FALSE
flush-time                                      0
innodb-adaptive-flushing                        TRUE
innodb-adaptive-flushing-lwm                    10
innodb-flush-log-at-timeout                     1
innodb-flush-log-at-trx-commit                  1
innodb-flush-method                             fsync
innodb-flush-neighbors                          0
innodb-flush-sync                               TRUE
innodb-flushing-avg-loops                       30
innodb-idle-flush-pct                           100
innodb-log-wait-for-flush-spin-hwm              400
[dba@mysqlPerconaSTG ~]$
```

We'll append the system variable `innodb-flush-method` to the `/etc/percona/my.cnf` file and set it to `O_DIRECT`.

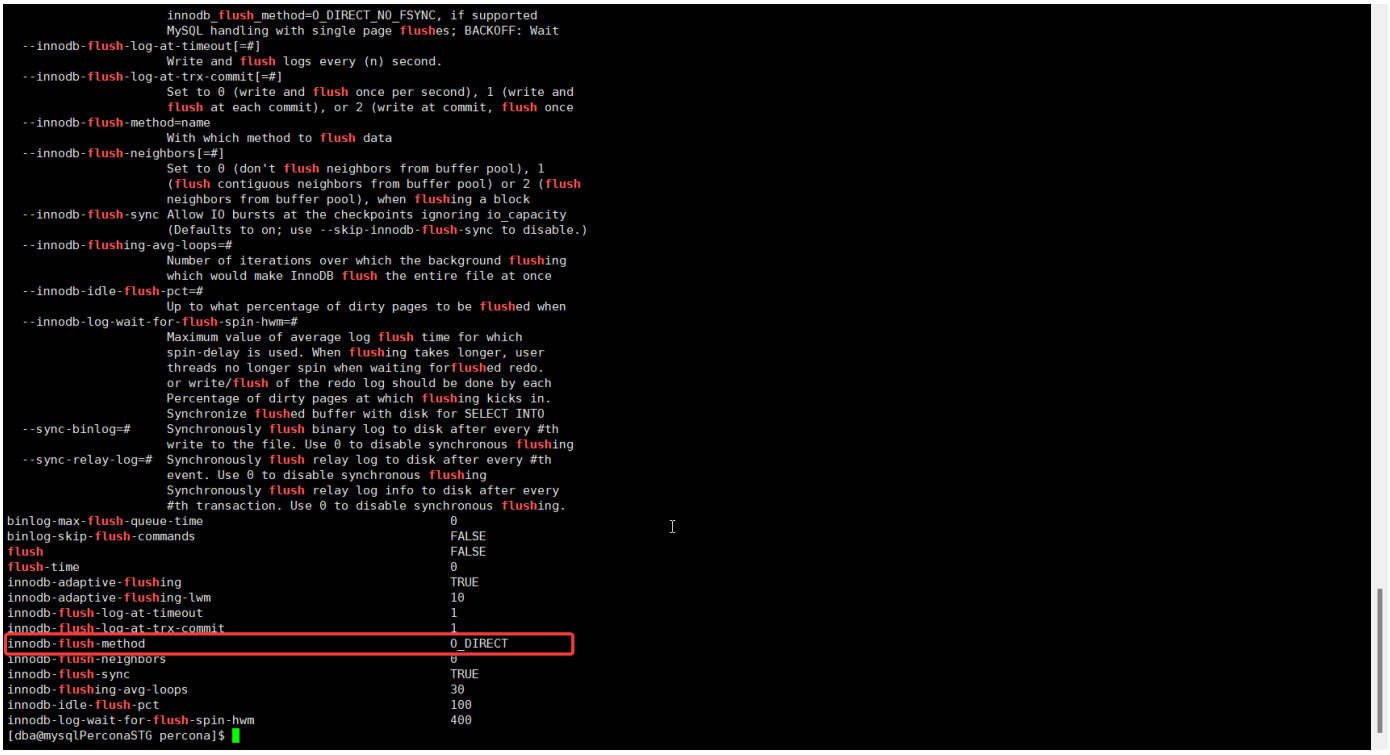
Additionally, we'll include a duplicate line for `O_DIRECT_NO_FSYNC`, which will be commented out.

```
# Percona Server template configuration
#
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
innodb_buffer_pool_size = 256M
innodb-flush-method = O_DIRECT
#innodb-flush-method = O_DIRECT_NO_FSYNC

~
~
~
~
-- INSERT --
```

As usual, we'll restart the MySQL service with the command: `systemctl restart mysqld`.

```
[dba@mysqlPerconaSTG percona]$ systemctl restart mysqld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to restart 'mysqld.service'.
Authenticating as: dba
Password:
==== AUTHENTICATION COMPLETE ====
[dba@mysqlPerconaSTG percona]$ systemctl status  mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2024-05-10 22:03:14 +03; 5s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 129269 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 129304 (mysqld)
   Status: "Server is operational"
    Tasks: 39 (limit: 24168)
   Memory: 402.9M
   CGroup: /system.slice/mysqld.service
           └─129304 /usr/sbin/mysqld

May 10 22:03:08 mysqlPerconaSTG systemd[1]: mysqld.service: Succeeded.
May 10 22:03:08 mysqlPerconaSTG systemd[1]: Stopped MySQL Server.
May 10 22:03:08 mysqlPerconaSTG systemd[1]: Starting MySQL Server...
May 10 22:03:14 mysqlPerconaSTG systemd[1]: Started MySQL Server.
[dba@mysqlPerconaSTG percona]$
```

Run the command `mysqld --verbose --help | grep -i flush`, then scroll down to confirm that the flush method has been updated.



what is Doublewrite Buffer

- InnoDB employs a doublewrite buffer to safeguard against potential data loss during crashes.
- This buffer acts as an intermediate storage area before data is written to the actual data files, ensuring recovery from half-written pages.
- Half-written pages occur due to OS errors, storage issues, or unexpected MySQL process interruptions during page writes.
- The doublewrite buffer stores data twice to facilitate crash recovery without doubling the I/O overhead.
- Prior to MySQL 8, the doublewrite buffer resided in doublewrite files.
- In MySQL 8, the doublewrite buffer storage is consolidated into double write files.
- The system variable `innodb_doublewrite` controls the activation of the double write mechanism.
- Optionally, `innodb_doublewrite_dir` specifies the directory for storing doublewrite files, defaulting to the MySQL data directory.
- Doublewrite files follow a naming convention such as `#ib_16384_0.dblwr` and `ib_16384_1.dblwr`, with "16384 bytes" indicating a 16K InnoDB page size.
- It's recommended to place doublewrite files on the fastest available storage media for optimal performance and reliability.

when should you disbale doublewrite

If data integrity is not a primary concern and performance is paramount, you may choose to disable the doublewrite buffer.

managing doublewrite

In this section, we'll show you how to:

1. Turn off the doublewrite buffer.
2. Activate the doublewrite buffer and specify new file locations.
    1. **Turn off the doublewrite buffer.**

Before disabling the double-write buffer, we'll first check whether it's currently enabled or disabled.

`show global variables like '%doublewrite%';`

```
mysql> show global variables like '%doublewriye%';
Empty set (0.09 sec)

mysql> show global variables like '%doublewrite%';
+---------------------------------+----------------+
| Variable_name                   | Value          |
+---------------------------------+----------------+
| innodb_doublewrite              | ON             |
| innodb_doublewrite_batch_size   | 0              |
| innodb_doublewrite_dir          |                |
| innodb_doublewrite_files        | 2              |
| innodb_doublewrite_pages        | 4              |
| innodb_parallel_doublewrite_path | xb_doublewrite |
+---------------------------------+----------------+
6 rows in set (0.00 sec)

mysql>
```

as you see double-write as mentioned before is enabled by default
also you will notice that there is another variable called `innodb_doublewrite_files` and value is 2 meaning there is two double write files

```
mysql> show global variables like '%doublewrite%';
+---------------------------------+----------------+
| Variable_name                   | Value          |
+---------------------------------+----------------+
| innodb_doublewrite              | ON             |
| innodb_doublewrite_batch_size   | 0              |
| innodb_doublewrite_dir          |                |
| innodb_doublewrite_files        | 2              |
| innodb_doublewrite_pages        | 4              |
| innodb_parallel_doublewrite_path | xb_doublewrite |
+---------------------------------+----------------+
6 rows in set (0.00 sec)

mysql>
```

if you check the data directory you will find the two files it should in fromat `#ib_16384_0.dblwr` and

`ib_16384_1.dblwr` as mentioned before

```
[dba@mysqlPerconaSTG mysqldata]$ cd  mysql/
[dba@mysqlPerconaSTG mysql]$ ll
total 97768
-rw-r----- 1 mysql mysql     3602 Apr 12 00:36  1.000011
-rw-r----- 1 mysql mysql      180 Apr 12 00:54  1.000012
-rw-r----- 1 mysql mysql      180 Apr 12 01:10  1.000013
-rw-r----- 1 mysql mysql      180 Apr 12 01:22  1.000014
-rw-r----- 1 mysql mysql      180 Apr 14 11:28  1.000015
-rw-r----- 1 mysql mysql      180 May  2 11:04  1.000016
-rw-r----- 1 mysql mysql      180 May 10 22:03  1.000017
-rw-r----- 1 mysql mysql      157 May 10 22:03  1.000018
-rw-r----- 1 mysql mysql       88 May 10 22:03  1.index
-rw-r----- 1 mysql mysql       56 Apr  1 11:13  auto.cnf
-rw-r----- 1 mysql mysql      848 Apr  2 10:29  binlog.000001
-rw-r----- 1 mysql mysql      180 Apr  2 10:56  binlog.000002
-rw-r----- 1 mysql mysql       32 Apr  2 10:29  binlog.index
-rw------- 1 mysql mysql     1680 Apr  1 11:13  ca-key.pem
-rw-r--r-- 1 mysql mysql     1120 Apr  1 11:13  ca.pem
-rw-r--r-- 1 mysql mysql     1120 Apr  1 11:13  client-cert.pem
-rw------- 1 mysql mysql     1680 Apr  1 11:13  client-key.pem
drwxr-x--- 2 mysql mysql        6 Apr  9 00:52  database
drwxr-x--- 2 mysql mysql      149 Apr  9 02:00  employees
-rw-r----- 1 mysql mysql   196608 May 10 22:05 '#ib_16384_0.dblwr'
-rw-r----- 1 mysql mysql  8585216 Apr 10 07:33 '#ib_16384_1.dblwr'
-rw-r----- 1 mysql mysql     3494 May 10 22:03  ib_buffer_pool
-rw-r----- 1 mysql mysql 12582912 May 10 22:03  ibdata1
-rw-r----- 1 mysql mysql 12582912 May 10 22:03  ibtmp1
drwxr-x--- 2 mysql mysql     4096 May 10 22:03 '#innodb_redo'
drwxr-x--- 2 mysql mysql      187 May 10 22:03 '#innodb_temp'
drwxr-x--- 2 mysql mysql      143 Apr  1 11:13  mysql
-rw-r----- 1 mysql mysql 32505856 May 10 22:03  mysql.ibd
drwxr-x--- 2 mysql mysql     8192 Apr  1 11:13  performance_schema
-rw------- 1 mysql mysql     1680 Apr  1 11:13  private_key.pem
-rw-r--r-- 1 mysql mysql      452 Apr  1 11:13  public_key.pem
-rw-r--r-- 1 mysql mysql     1120 Apr  1 11:13  server-cert.pem
-rw------- 1 mysql mysql     1676 Apr  1 11:13  server-key.pem
drwxr-x--- 2 mysql mysql       28 Apr  1 11:13  sys
drwxr-x--- 2 mysql mysql       28 Apr 10 08:04  test1
-rw-r----- 1 mysql mysql 16777216 May 10 22:05  undo_001
-rw-r----- 1 mysql mysql 16777216 May 10 22:05  undo_002
[dba@mysqlPerconaSTG mysql]$
```

to disable double write let's discovery the syntax

`mysqld --verbose --help | grep -i doublewrite`

```
[dba@mysqlPerconaSTG mysql]$ mysqld --verbose --help | grep -i doublewrite
  --innodb-doublewrite[=name]
                      Enable InnoDB doublewrite buffer (enabled by default).
                      Disable with --skip-innodb-doublewrite.
  --innodb-doublewrite-batch-size=#
  --innodb-doublewrite-dir=name
                      Use a separate directory for the doublewrite buffer
  --innodb-doublewrite-files=#
  --innodb-doublewrite-pages=#
                      enable or disable encryption of parallel doublewrite
  --innodb-parallel-doublewrite-path=name
                      path to the parallel doublewrite file and has no effect
                      now. Use --innodb-doublewrite-dir instead.
innodb-doublewrite                                ON
innodb-doublewrite-batch-size                     0
innodb-doublewrite-dir                            (No default value)
innodb-doublewrite-files                          0
innodb-doublewrite-pages                          0
innodb-parallel-doublewrite-path                  xb_doublewrite
[dba@mysqlPerconaSTG mysql]$
```

to disable double-write as manual showed we will use `--skip-innodb-doublewrite.` and add it in my.cnf file

```
# Percona Server template configuration
#
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
innodb-buffer-pool-size = 256M
innodb-flush-method = O_DIRECT
#innodb-flush-method = O_DIRECT_NO_FSYNC

skip-innodb-doublewrite
~
~
~
```

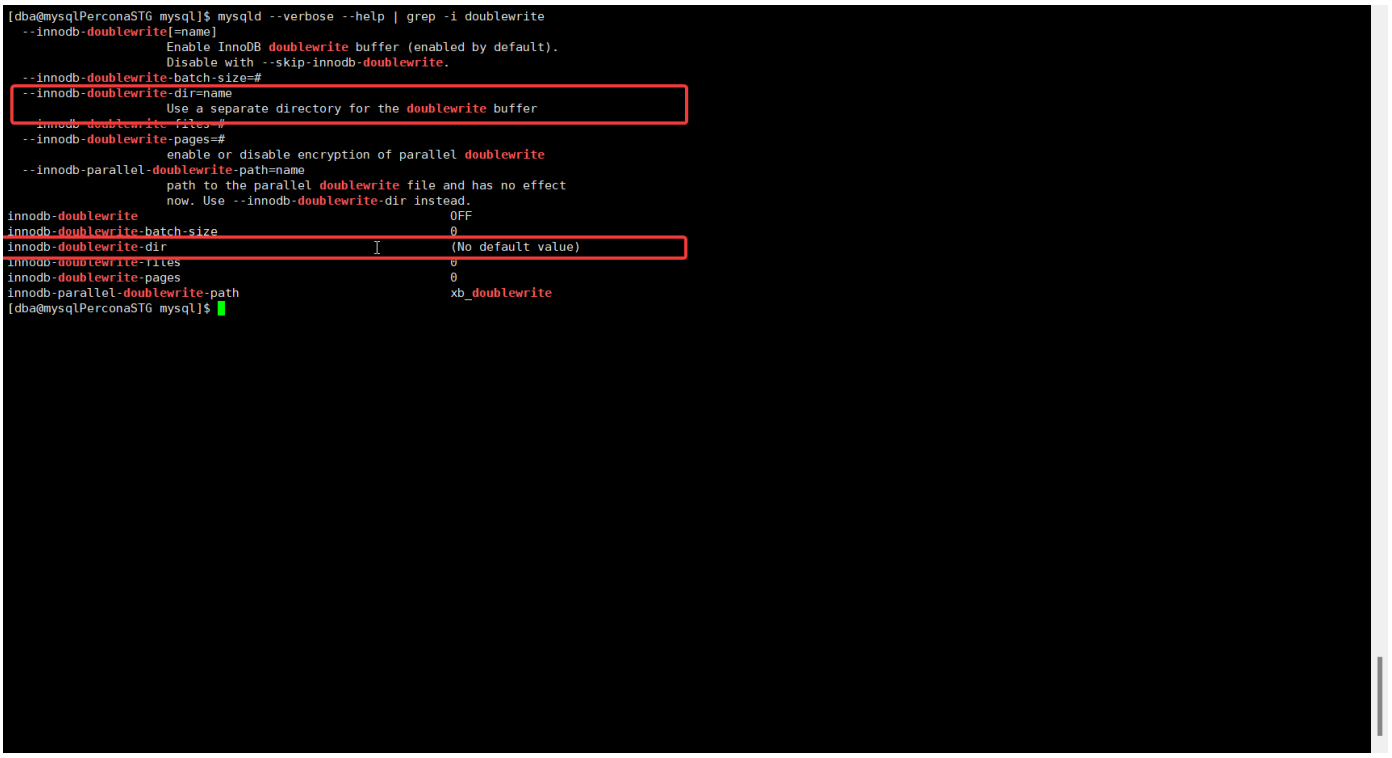
now restart mysqld and verify that double-write is disabled

```
[dba@mysqlPerconaSTG mysql]$ systemctl restart mysqld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to restart 'mysqld.service'.
Authenticating as: dba
Password:
==== AUTHENTICATION COMPLETE ====
[dba@mysqlPerconaSTG mysql]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show global variables like '%doublewrite%';
+---------------------------------+----------------+
| Variable_name                   | Value          |
+---------------------------------+----------------+
| innodb_doublewrite              | OFF            |
| innodb_doublewrite_batch_size   | 0              |
| innodb_doublewrite_dir          |                |
| innodb_doublewrite_files        | 0              |
| innodb_doublewrite_pages        | 0              |
| innodb_parallel_doublewrite_path | xb_doublewrite |
+---------------------------------+----------------+
6 rows in set (0.03 sec)

mysql>
```

2. Activate the doublewrite buffer and specify new file locations.

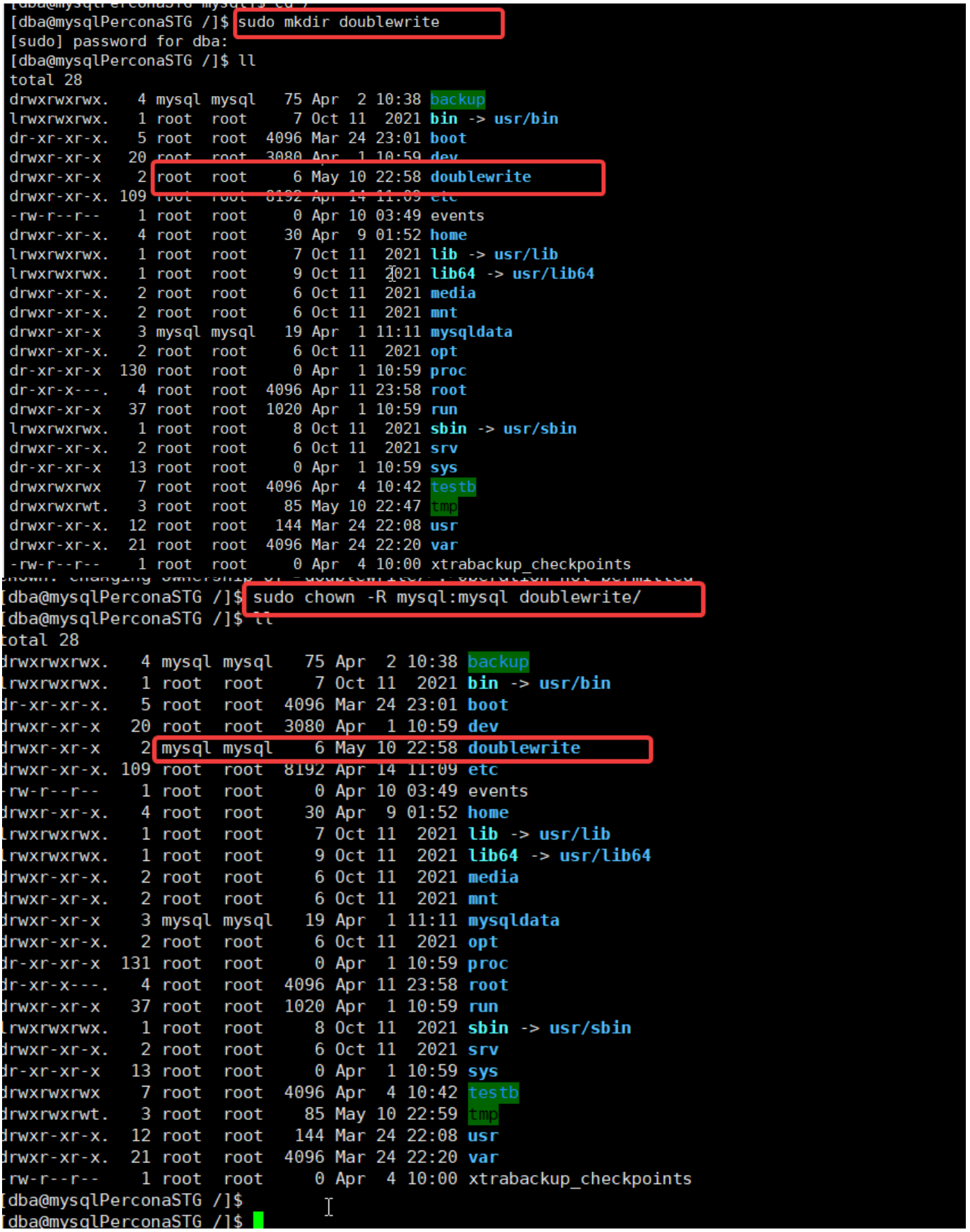
now we will enable doublewrite but we will make file goes to deferent directory other than the default data directory
first search for syntax by using the below command

```
mysqld --verbose --help | grep -i doublewrite
```



the variable we will be using is `--innodb-doublewrite-dir=name`
below you will find `--innodb-doublewrite-dir=name` is not defend and it using the default directory

for testing purpose i will create directory called doublewrite in root directory
also we will put the directory owner to `mysql` user
`--innodb-doublewrite-dir=name`
`sudo chown -R mysql:mysql doublewrite/`



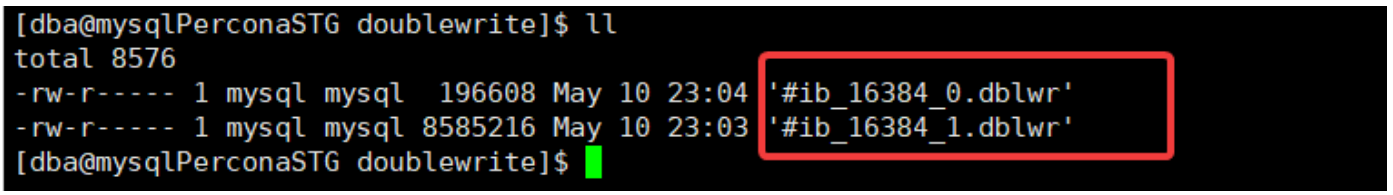
now we will add variable `--innodb-doublewrite-dir=name` and mentioned the new directory for doublewrite we created before in my.cnf file

also we will comment `skip-innodb-doublewrite` to enable doublewrite function

```
# Percona Server template configuration
#
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
innodb-buffer-pool-size = 256M
innodb-flush-method = O_DIRECT
#innodb-flush-method = O_DIRECT_NO_FSYNC

skip-innodb-doublewrite
innodb-doublewrite-dir = /doublewrite
~

~
```

```
[dba@mysqlPerconaSTG /]$ systemctl restart mysqld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to restart 'mysqld.service'.
Authenticating as: dba
Password:
==== AUTHENTICATION COMPLETE ====
[dba@mysqlPerconaSTG /]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show global variables like '%doublewrite%';
+---------------------------------+----------------+
| Variable_name                   | Value          |
+---------------------------------+----------------+
| innodb_doublewrite              | ON             |
| innodb_doublewrite_batch_size   | 0              |
| innodb_doublewrite_dir          | /doublewrite   |
| innodb_doublewrite_files        | 2              |
| innodb_doublewrite_pages        | 4              |
| innodb_parallel_doublewrite_path | xb_doublewrite |
+---------------------------------+----------------+
6 rows in set (0.04 sec)

mysql>
```

if you open the doublewrite directory you will see the two files for doublewrite exist



Flushing Logs at Transaction Commit

The system variable `innodb_flush_log_at_trx_commit` is crucial for balancing performance and I/O-related issues.
When transactions occur in InnoDB, transitioning from memory to permanent disk, it's essential to manage this movement.
You can choose between strict ACID compliance for data integrity or sacrificing some integrity for better performance.
Here are the three possible values for `innodb_flush_log_at_trx_commit`:

1. **1**: This is the default, ensuring full ACID compliance. Logs are written and flushed to disk at every transaction commit.
2. **0**: Logs are written and flushed to disk once per second. Transactions for which logs have not been flushed can be lost in a crash.
3. **2**: Logs are written after each transaction commit and flushed to disk once per second. Transactions for which logs have not been flushed can be lost in a crash.

which value to use :

- 1: Safest option, ensuring full ACID compliance with no data loss.
- 0: Transactions are not immediately flushed to disk on commit, providing better performance but risking the loss of up to one second's worth of transactions.
- 2: Offers better performance, with transactions committed immediately but flushed to disk only once per second, risking the loss of transactions within that second.

managing innodb_flush_log_at_trx_commit

first we will check the value of the system variable `innodb_flush_log_at_trx_commit`

```
show global variables like '%innodb_flush%';
```

```
[dba@mysqlPerconaSTG doublewrite]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show global variables like '%innodb_flush%';
+------------------------------+----------+
| Variable_name                | Value    |
+------------------------------+----------+
| innodb_flush_log_at_timeout  | 1        |
| innodb_flush_log_at_trx_commit | 1      |
| innodb_flush_method          | O_DIRECT |
| innodb_flush_neighbors       | 0        |
| innodb_flush_sync            | ON       |
| innodb_flushing_avg_loops    | 30       |
+------------------------------+----------+
6 rows in set (0.01 sec)

mysql>
```

now we will change the value to 2

first let's grab the syntax using the below command

```
mysqld --verbose --help | grep -i trx
```

```
Bye
[dba@mysqlPerconaSTG doublewrite]$ mysqld --verbose --help | grep -i trx
  --innodb-api-trx-level[=#]
  --innodb-flush-log-at-trx-commit[=#]
innodb-api-trx-level                            0
innodb-flush-log-at-trx-commit                  1
[dba@mysqlPerconaSTG doublewrite]$
```

we will copy `innodb-flush-log-at-trx-commit` add it in my.cnf file and set the value to 2

```
#
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
innodb-buffer-pool-size = 256M
innodb-flush-method = O_DIRECT
#innodb-flush-method = O_DIRECT_NO_FSYNC

#skip-innodb-doublewrite
innodb-doublewrite-dir = /doublewrite


innodb-flush-log-at-trx-commit = 2
-- INSERT --
```

now restart mysqld and verify the current value for `innodb-flush-log-at-trx-commit`

`show global variables like '%innodb_flush%';`

```
mysql> show global variables like '%innodb_flush%';
+--------------------------------+----------+
| Variable_name                  | Value    |
+--------------------------------+----------+
| innodb_flush_log_at_timeout    | 1        |
| innodb_flush_log_at_trx_commit | 2        |
| innodb_flush_method            | O_DIRECT |
| innodb_flush_neighbors         | 0        |
| innodb_flush_sync              | ON       |
| innodb_flushing_avg_loops      | 30       |
+--------------------------------+----------+
6 rows in set (0.02 sec)

mysql>
```

InnoDB Redo Log Files

InnoDB utilizes redo log files to maintain database consistency and durability. Before modifying a page, changes are recorded in the redo log files to ensure they are saved to disk before the corresponding page. In the event of a crash causing memory data loss, InnoDB replays these redo logs during restart to restore the database to its pre-crash state.

Physically, redo log files are represented on disk as `ib_logfile0` and `ib_logfile1`.

The size of the redo log is controlled by the `innodb_log_file_size` system variable, with a default size of 50MB.

By default, there are two redo log files in a group, controlled by the `innodb_log_files_in_group` system variable.

The `innodb_fast_shutdown` system variable determines the shutdown behavior default value is `1`:

- Setting it to `0` ensures a clean shutdown, performing additional flushing operations, resulting in a longer shutdown time but saving time on startup.
- Setting it to `1` enables fast shutdown, shutting down MySQL but reading redo log files on startup.

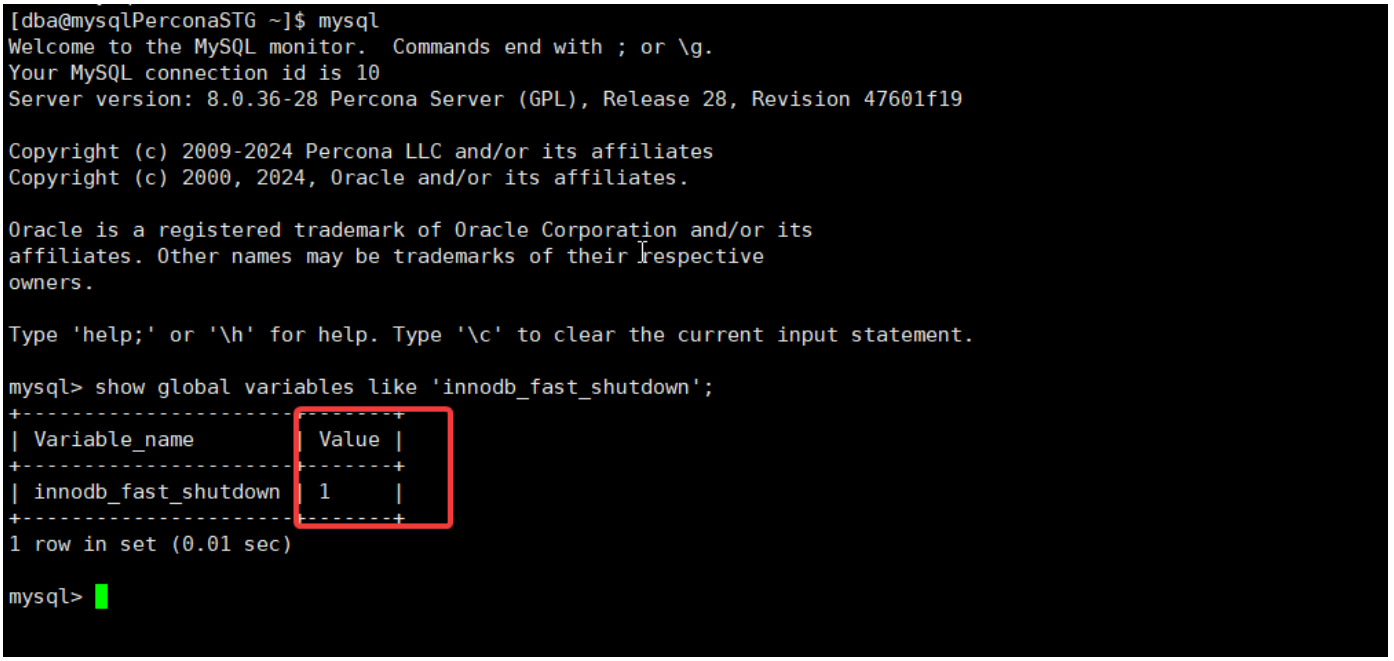
managing redo log files

1. managing innodb_fast_shutdown

reference link

Let's retrieve the value set for the variable `innodb_fast_shutdown`:

```
SHOW GLOBAL VARIABLES LIKE 'innodb_fast_shutdown';
```



The current value is set to `1`, enabling fast shutdown, which is the default behavior. We'll change it to a clean shutdown by setting it to `0`:

```
SET GLOBAL innodb_fast_shutdown = 0;
```

```
mysql> set global  innodb fast shutdown = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> set global variables innodb_fast_shutdown = 0;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to
e 1
mysql> show global variables like 'innodb_fast_shutdown';
+----------------------+-------+
| Variable_name        | Value |
+----------------------+-------+
| innodb_fast_shutdown | 0     |
+----------------------+-------+
1 row in set (0.01 sec)

mysql>
```

Since we've changed the `innodb_fast_shutdown` variable to ensure a clean shutdown, it means we shouldn't require the redo log files anymore. For testing purposes, I'll stop MySQL and then proceed to the data directory to delete the redo log files before starting mysqld.

```
systemctl stop mysqld
```

```
[dba@mysqlPerconaSTG ~]$ systemctl stop  mysqld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to stop 'mysqld.service'.
Authenticating as: dba
Password:
==== AUTHENTICATION COMPLETE ====
[dba@mysqlPerconaSTG ~]$ systemctl status  mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: inactive (dead) since Sat 2024-05-11 16:52:10 +03; 9s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 129674 ExecStart=/usr/sbin/mysqld $MYSQLD_OPTS (code=exited, status=0/SUCCESS)
  Process: 129639 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 129674 (code=exited, status=0/SUCCESS)
   Status: "Server shutdown complete"

May 10 23:25:57 mysqlPerconaSTG systemd[1]: mysqld.service: Succeeded.
May 10 23:25:57 mysqlPerconaSTG systemd[1]: Stopped MySQL Server.
May 10 23:25:57 mysqlPerconaSTG systemd[1]: Starting MySQL Server...
May 10 23:26:03 mysqlPerconaSTG systemd[1]: Started MySQL Server.
May 11 16:52:08 mysqlPerconaSTG systemd[1]: Stopping MySQL Server...
May 11 16:52:10 mysqlPerconaSTG systemd[1]: mysqld.service: Succeeded.
May 11 16:52:10 mysqlPerconaSTG systemd[1]: Stopped MySQL Server.
[dba@mysqlPerconaSTG ~]$
```

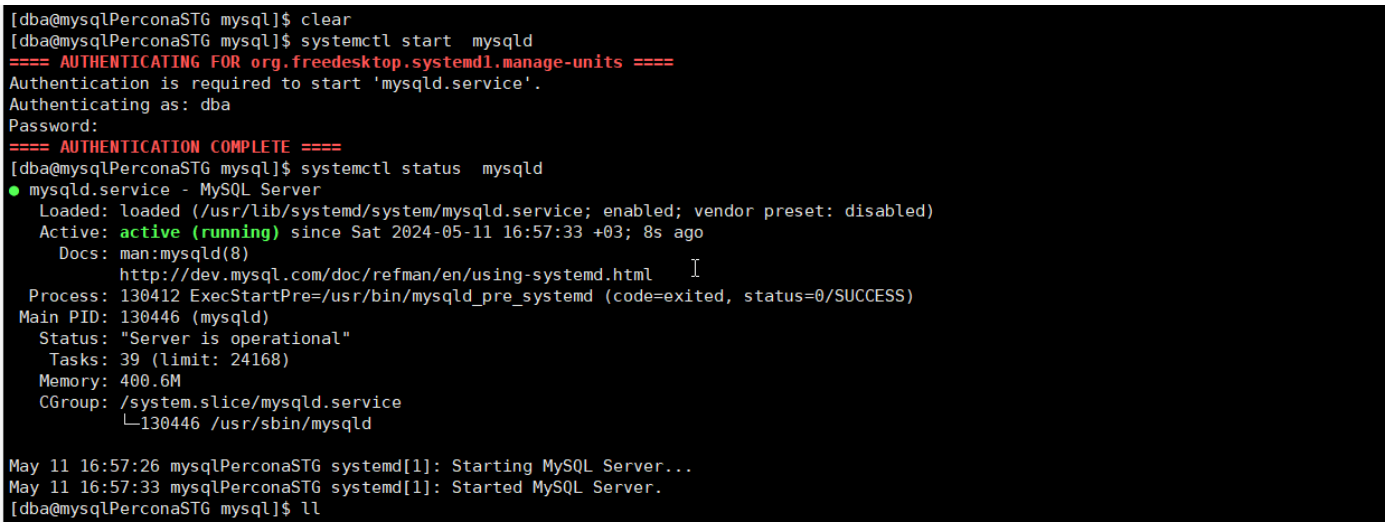
Now, we'll navigate to the data directory and delete the redo log files. As mentioned earlier, the redo log files are named `ib_logfile0` and `ib_logfile1`. We can use the following command to delete both files:

```
rm -f /mysqldata/mysql/ib_logfile*
```

```
[dba@mysqlPerconaSTG mysql]$ rm -f /mysqldata/mysql/ib_logfile*
[dba@mysqlPerconaSTG mysql]$
```

Now, we'll start MySQL again. Ideally, it should start without any issues, and it should recreate the redo log files automatically.

```
systemctl start mysqld
```



change redo log size and location

In this section, we'll demonstrate the following points:

1. Changing the size of the redo log files from 50MB to 100MB.
2. Changing the location of the redo log files.
3. Restarting mysqld and then verifying that the redo log files have been relocated.

We'll start by creating our own file system for the redo log files in the root directory, called `redologs`. While best practice suggests placing redo log files in a separate mount point, for testing purposes, we'll place them in the root directory's mount point.

First, let's create the directory and change the owner to the `mysql` user:

```
sudo mkdir /redologs
sudo chown mysql:mysql /redologs
```

```
[dba@mysqlPerconaSTG /]$ sudo mkdir redologs
[sudo] password for dba:
[dba@mysqlPerconaSTG /]$ sudo chown -R mysql:mysql redologs/
[dba@mysqlPerconaSTG /]$ ll
total 28
drwxrwxrwx.   4 mysql mysql   75 Apr  2 10:38 backup
lrwxrwxrwx.   1 root  root     7 Oct 11  2021 bin -> usr/bin
dr-xr-xr-x.   5 root  root  4096 Mar 24 23:01 boot
drwxr-xr-x   20 root  root  3080 Apr  1 10:59 dev
drwxr-xr-x    2 mysql mysql   56 May 10 23:03 doublewrite
drwxr-xr-x. 109 root  root  8192 Apr 14 11:09 etc
-rw-r--r--    1 root  root     0 Apr 10 03:49 events
drwxr-xr-x.   4 root  root    30 Apr  9 01:52 home
lrwxrwxrwx.   1 root  root     7 Oct 11  2021 lib -> usr/lib
lrwxrwxrwx.   1 root  root     9 Oct 11  2021 lib64 -> usr/lib64
drwxr-xr-x.   2 root  root     6 Oct 11  2021 media
drwxr-xr-x.   2 root  root     6 Oct 11  2021 mnt
drwxr-xr-x    3 mysql mysql   19 Apr  1 11:11 mysqldata
drwxr-xr-x.   2 root  root     6 Oct 11  2021 opt
dr-xr-xr-x  130 root  root     0 Apr  1 10:59 proc
drwxr-xr-x    2 mysql mysql    6 May 11 17:07 redologs
dr-xr-x---.   4 root  root  4096 Apr 11 23:58 root
drwxr-xr-x   37 root  root  1020 Apr  1 10:59 run
lrwxrwxrwx.   1 root  root     8 Oct 11  2021 sbin -> usr/sbin
drwxr-xr-x.   2 root  root     6 Oct 11  2021 srv
dr-xr-xr-x   13 root  root     0 Apr  1 10:59 sys
drwxrwxrwx    7 root  root  4096 Apr  4 10:42 testb
drwxrwxrwt.   3 root  root    85 May 11 17:07 tmp
drwxr-xr-x.  12 root  root   144 Mar 24 22:08 usr
drwxr-xr-x.  21 root  root  4096 Mar 24 22:20 var
-rw-r--r--    1 root  root     0 Apr  4 10:00 xtrabackup_checkpoints
[dba@mysqlPerconaSTG /]$
```

Let's add the following variables to the my.cnf file, which will change the redo log size to 100MB and relocate the redo log to a separate path:

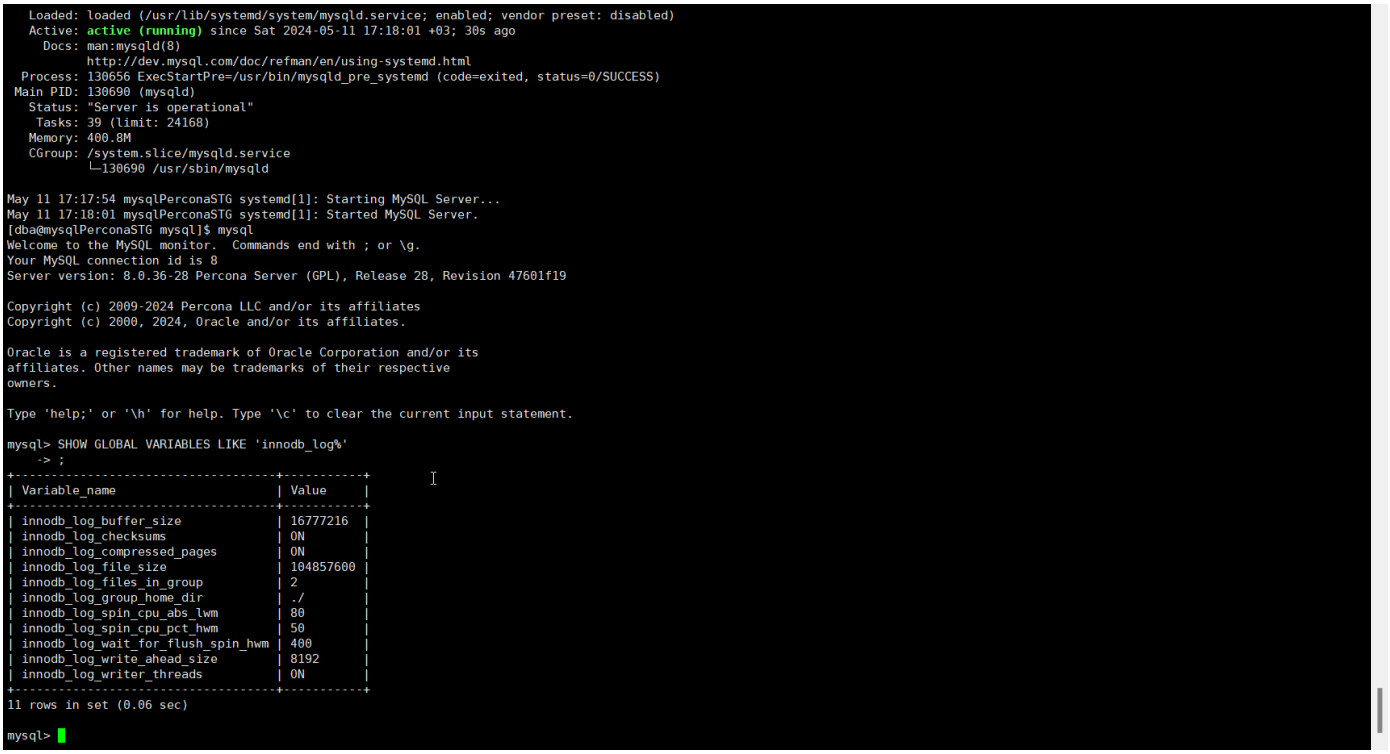
```
innodb-log-file-size = 100M
innodb-log-files-in-group = 2
innodb-log-group-home_dir = /redologs
```

```
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
innodb-buffer-pool-size = 256M
innodb-flush-method = O_DIRECT
#innodb-flush-method = O_DIRECT_NO_FSYNC

#skip-innodb-doublewrite
innodb-doublewrite-dir = /doublewrite

innodb-log-file-size = 100M
innodb-log-files-in-group = 2
innodb-log-group-home_dir = /redologs

innodb-flush-log-at-trx-commit = 2
```

Now, let's restart mysqld and verify that the size has changed and the redo log has been relocated to a different path. We'll also check the directory to see if the redo logs are recreated in the new path.

```
systemctl restart mysqld
```

```
SHOW GLOBAL VARIABLES LIKE 'innodb_log%'
```



System Tablespace

[reference link]

- The system tablespace serves as a storage area for the change buffer.
- It may store table data and indexes if the tables were created within it.
- Ideally, tables and indexes should be created in either the "file-per-table" or general tablespace.
- In older MySQL versions, the InnoDB data dictionary was stored in the system tablespace.
- The system tablespace used to also contain the doublewrite buffer.
- The default name for the system tablespace is `ibdata1`, and by default, it's created in the data directory.
- The system tablespace can consist of one or more data files.
- `innodb_data_file_path` is the system variable that specifies the size, name, and number of data files belonging to the system tablespace. Its syntax is typically as follows: `ibdata1:10M:autoextend`.

managing system tablespace

In this section, we'll demonstrate the following:

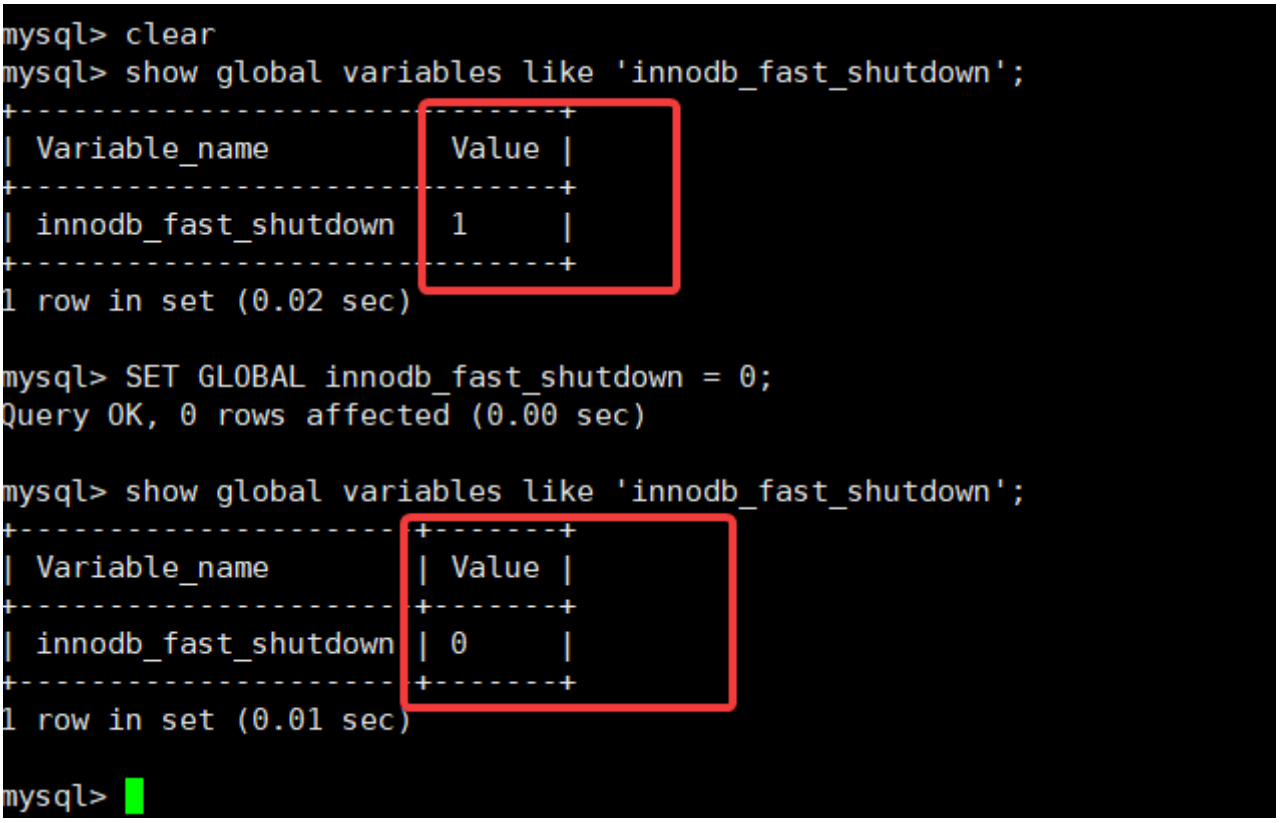
1. Adding another data file for the system tablespace - 10M autoextend.
2. Moving the system tablespace to a new location.
3. Restarting MySQL.

4. Verifying the changes.

   An important note: Before attempting any modifications related to the system tablespace, always ensure you have a backup available to rollback in case of any issues.

first step is to SET CLEAN SHUTDOWN
make sure value is set to `0`

```
show global variables like 'innodb_fast_shutdown';
SET GLOBAL innodb_fast_shutdown = 0;
```



next shutdown MySQL services

```
sudo systemctl stop mysqld

systemctl status mysqld
```

```
[dba@mysqlPerconaSTG mysql]$ sudo systemctl stop mysqld
[sudo] password for dba:
[dba@mysqlPerconaSTG mysql]$ sudo systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: inactive (dead) since Sat 2024-05-11 17:38:33 +03; 10s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 130690 ExecStart=/usr/sbin/mysqld $MYSQLD_OPTS (code=exited, status=0/SUCCESS)
  Process: 130656 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 130690 (code=exited, status=0/SUCCESS)
   Status: "Server shutdown complete"

May 11 17:17:54 mysqlPerconaSTG systemd[1]: Starting MySQL Server...
May 11 17:18:01 mysqlPerconaSTG systemd[1]: Started MySQL Server.
May 11 17:38:32 mysqlPerconaSTG systemd[1]: Stopping MySQL Server...
May 11 17:38:33 mysqlPerconaSTG systemd[1]: mysqld.service: Succeeded.
May 11 17:38:33 mysqlPerconaSTG systemd[1]: Stopped MySQL Server.
[dba@mysqlPerconaSTG mysql]$
```

Let's start by creating a directory for the system tablespace data files and adjusting the file permissions to be owned by the `mysql` user:

```
sudo mkdir /mysqldata/innodb
sudo mv /mysqldata/mysql/ibdata1 /mysqldata/innodb/
sudo chown -R mysql:mysql /mysqldata/innodb
```

```
[dba@mysqlPerconaSTG mysql]$ sudo mkdir /mysqldata/innodb
[dba@mysqlPerconaSTG mysql]$ sudo mv /mysqldata/mysql
mv: missing destination file operand after '/mysqldata/mysql'
Try 'mv --help' for more information.
[dba@mysqlPerconaSTG mysql]$ sudo mv /mysqldata/mysql/ibdata1 /mysqldata/innodb/
[dba@mysqlPerconaSTG mysql]$ ls /mysqldata/innodb
ibdata1
[dba@mysqlPerconaSTG mysql]$ sudo chown -R mysql:mysql /mysqldata/innodb
[dba@mysqlPerconaSTG mysql]$ cd ..
[dba@mysqlPerconaSTG mysqldata]$ ll
total 4
drwxr-xr-x  2 mysql mysql   21 May 11 17:44 innodb
drwxr-xr-x 10 mysql mysql 4096 May 11 17:44 mysql
[dba@mysqlPerconaSTG mysqldata]$ ls innodb/
ibdata1
[dba@mysqlPerconaSTG mysqldata]$ ll innodb/
total 12288
-rw-r----- 1 mysql mysql 12582912 May 11 17:38 ibdata1
[dba@mysqlPerconaSTG mysqldata]$
```

now we will configure my.cnf file to add the new path for system tablespace data file path using the below options

```
sudo vim my.cnf
innodb-data-home-dir  =  /mysqldata/innodb/
innodb-data-file-path = ibdata1:12M;ibdata2:10M:autoextend
```

```
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
innodb-buffer-pool-size = 256M
innodb-flush-method = O_DIRECT
#innodb-flush-method = O_DIRECT_NO_FSYNC

#skip-innodb-doublewrite
innodb-doublewrite-dir = /doublewrite

innodb-log-file-size = 100M
innodb-log-files-in-group = 2
#innodb-log-group-home_dir = /redologs

innodb-flush-log-at-trx-commit = 2

# system tablespace config new path

innodb-data-home-dir  =  /mysqldata/innodb/
innodb-data-file-path = ibdata1:12M;ibdata2:10M:autoextend
```

Let's start the MySQL service and then verify that the files for the system tablespace are in the new path. After that, we'll check the global variables to confirm that our changes have been replicated.

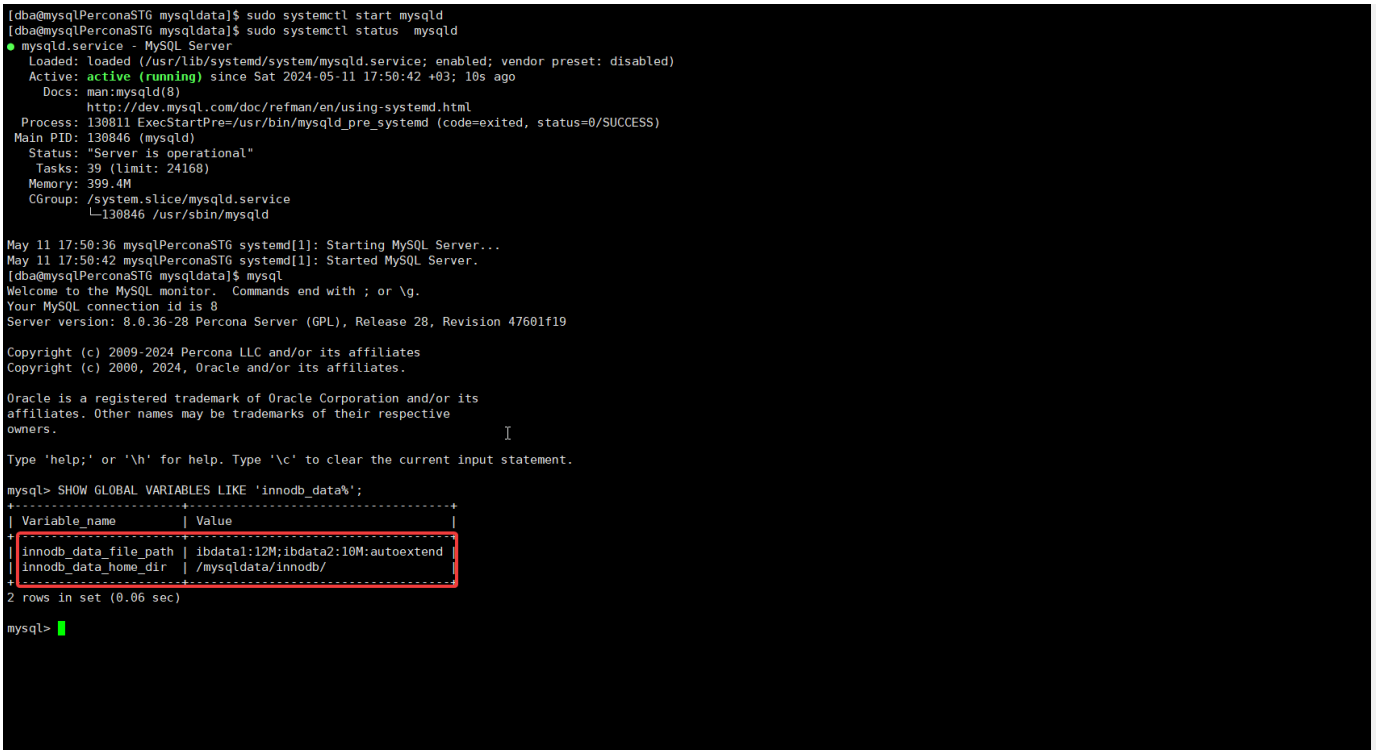
```
systemctl start mysqld
```

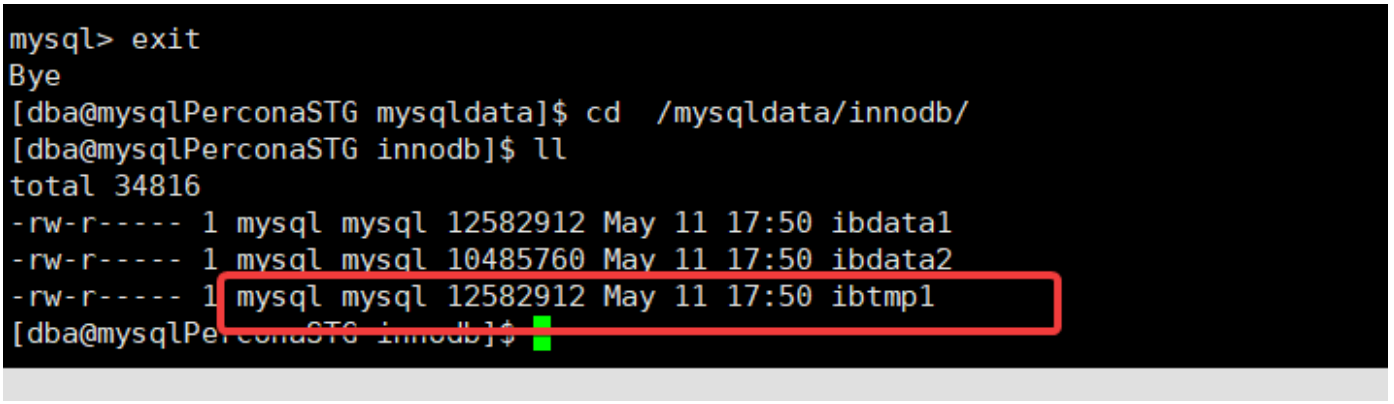
To verify the files for the system tablespace:

```
ls -l /new_system_tablespace_location
```

To check global variables:

```
SHOW GLOBAL VARIABLES LIKE 'innodb_data%';
```





Undo Tablespaces

The undo tablespace comprises two types: **system undo tablespace** and **user-defined undo tablespace**. Both types can contain multiple tablespaces.

Undo tablespaces store undo logs, which are collections of records containing information about how to reverse the latest changes made by a transaction to a clustered index record.

When a MySQL instance initializes, two default undo tablespaces are created by default. At least two undo tablespaces are required. These default undo tablespaces are named `innodb_undo_001` and `innodb_undo_002`.

The system variable `innodb_undo_directory` controls the location of the undo tablespace by default the location is data direct .

managing Undo Tablespaces

[reference link]

The procedure for managing undo tablespaces is similar to what we perform for the system tablespace.

first step is to SET CLEAN SHUTDOWN
make sure value is set to `0`

```
show global variables like 'innodb_fast_shutdown';
SET GLOBAL innodb_fast_shutdown = 0;
```

```
mysql> SET GLOBAL innodb_fast_shutdown = 0;
Query OK, 0 rows affected (0.01 sec)

mysql> show global variables like 'innodb_fast_shutdown';
+----------------------+-------+
| Variable_name        | Value |
+----------------------+-------+
| innodb_fast_shutdown | 0     |
+----------------------+-------+
1 row in set (0.02 sec)

mysql>
```

next shutdown MySQL services

```
sudo systemctl stop mysqld


systemctl status mysqld
```

```
[dba@mysqlPerconaSTG innodb]$ sudo systemctl stop mysqld
[sudo] password for dba:
[dba@mysqlPerconaSTG innodb]$ systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: inactive (dead) since Sat 2024-05-11 18:09:25 +03; 5s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 130846 ExecStart=/usr/sbin/mysqld $MYSQLD_OPTS (code=exited, status=0/SUCCESS)
  Process: 130811 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 130846 (code=exited, status=0/SUCCESS)
   Status: "Server shutdown complete"

May 11 17:50:36 mysqlPerconaSTG systemd[1]: Starting MySQL Server...
May 11 17:50:42 mysqlPerconaSTG systemd[1]: Started MySQL Server.
May 11 18:09:24 mysqlPerconaSTG systemd[1]: Stopping MySQL Server...
May 11 18:09:25 mysqlPerconaSTG systemd[1]: mysqld.service: Succeeded.
May 11 18:09:25 mysqlPerconaSTG systemd[1]: Stopped MySQL Server.
[dba@mysqlPerconaSTG innodb]$
```

For the directory, we will use the `innodb` directory we created for the system tablespace. The only step we will change is to move the undo tablespace.

```
sudo mv /mysqldata/mysql/undo_* /mysqldata/innodb/


sudo chown -R mysql:mysql /mysqldata/innodb
```

```
[dba@mysqlPerconaSTG innodb]$ sudo mv /mysqldata/mysql/undo_* /mysqldata/innodb/
[dba@mysqlPerconaSTG innodb]$ sudo chown -R mysql:mysql /mysqldata/innodb
[dba@mysqlPerconaSTG innodb]$ ll mysqldata/innodb/
ls: cannot access 'mysqldata/innodb/': No such file or directory
[dba@mysqlPerconaSTG innodb]$ ^C
[dba@mysqlPerconaSTG innodb]$ ll /mysqldata/innodb/
total 55300
-rw-r----- 1 mysql mysql     3509 May 11 18:09 ib_buffer_pool
-rw-r----- 1 mysql mysql 12582912 May 11 18:09 ibdata1
-rw-r----- 1 mysql mysql 10485760 May 11 17:50 ibdata2
-rw-r----- 1 mysql mysql 16777216 May 11 17:52 undo_001
-rw-r----- 1 mysql mysql 16777216 May 11 17:52 undo_002
[dba@mysqlPerconaSTG innodb]$
```

we will then open my.cnf and add the option for undo tablespace and specify the new path

```
sudo vim /etc/my.cnf
innodb-undo-directory  =  /mysqldata/innodb
```

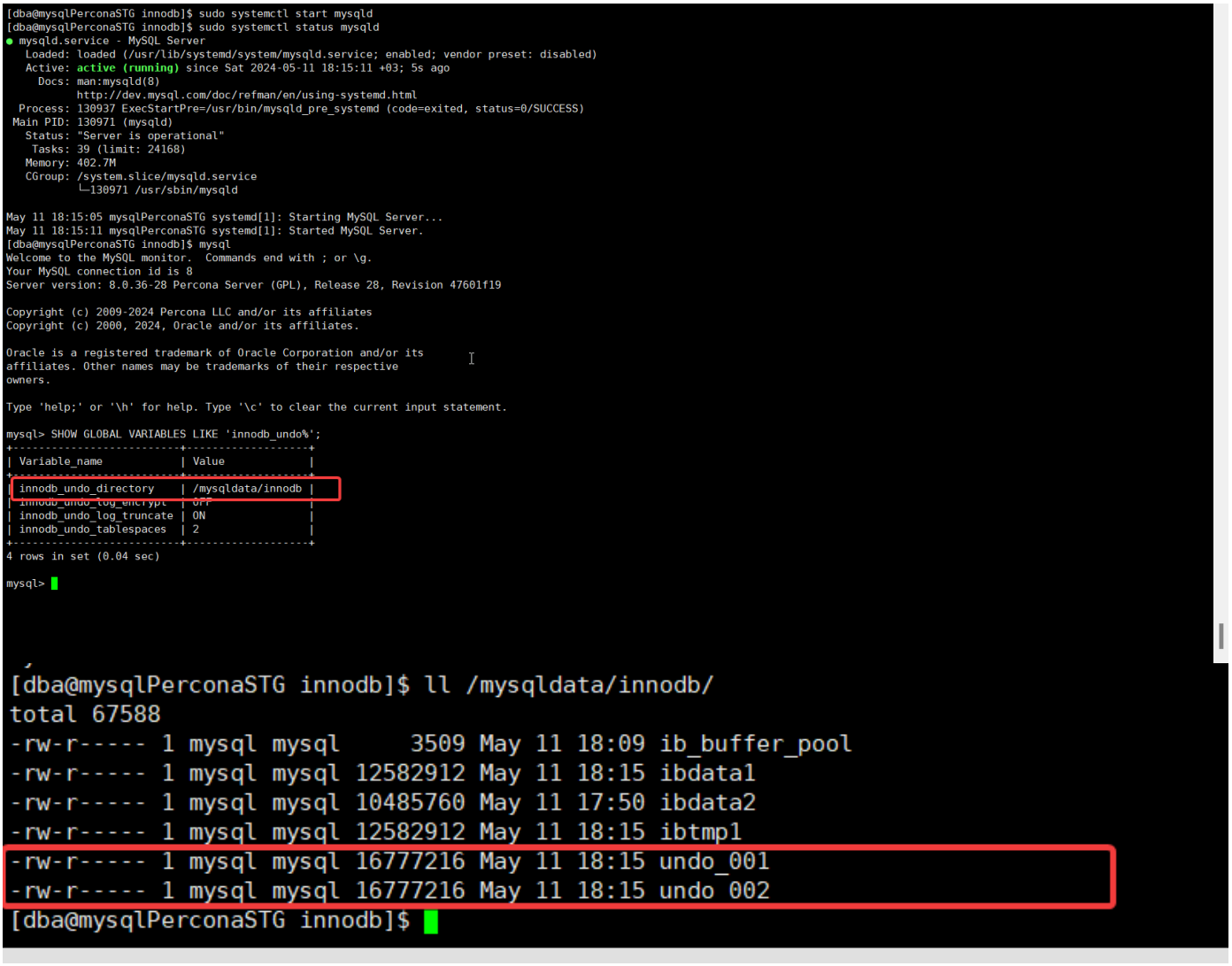
```
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
innodb-buffer-pool-size = 256M
innodb-flush-method = O_DIRECT
#innodb-flush-method = O_DIRECT_NO_FSYNC

#skip-innodb-doublewrite
innodb-doublewrite-dir = /doublewrite

innodb-log-file-size = 100M
innodb-log-files-in-group = 2
#innodb-log-group-home_dir = /redologs

innodb-flush-log-at-trx-commit = 2

# system tablespace config new path

innodb-data-home-dir  =  /mysqldata/innodb/
innodb-data-file-path = ibdata1:12M;ibdata2:10M:autoextend

#undo tablespace configuration changing the default path

innodb-undo-directory  =  /mysqldata/innodb
-- INSERT --
```

Now, let's start MySQL and then verify that the changes have been replicated by checking the system variables inside the MySQL shell.
also we will check the new path to verfiy the files

```
systemctl start mysqld
```

```
ll /mysqldata/innodb/
```

```
SHOW GLOBAL VARIABLES LIKE 'innodb_undo%';
```



Temporary Tablespaces

Temporary Tablespaces consist of two types

- **Session Temporary Tablespaces** user created Temporary tables
- **global Temporary Tablespaces** store rollback segments for changes made to user-created Temporary tables
- `ibtmp1` created single auto-extending data file in data directory
- `ibtmp1` default size is 12MB
- `innodb_temp_data_file_path` is a system variable that control Temporary Tablespaces size
- it is not possible to relocate these file they are tide up with data directory
- Temporary Tablespaces are stored in data directory

```
show global variables like 'innodb_temp_data%';
```

```
mysql> show global variables like 'innodb_temp_data%';
+----------------------------+-----------------------+
| Variable_name              | Value                 |
+----------------------------+-----------------------+
| innodb_temp_data_file_path | ibtmp1:12M:autoextend |
+----------------------------+-----------------------+
1 row in set (0.01 sec)

mysql>
```

File-Per-Table Tablespaces

As the name suggests, every table in MySQL has its own file for its tablespace. In simpler terms, every table is actually a tablespace in itself.

The system variable that controls whether file-per-table is enabled or not is `innodb_file_per_table`. By default, file-per-table tablespaces are enabled.

It is recommended to use file-per-table tablespaces as it simplifies managing tables.

The data file format for file-per-table tablespaces is as follows: `tablename.ibd`, where `tablename` is the name of the table.

To check whether File-Per-Table is enabled, log in to MySQL and use the following query:

```
SHOW GLOBAL VARIABLES LIKE 'innodb_file_per%';
```

```
mysql> show global variables like 'innodb_file_per%';
+-----------------------+-------+
| Variable_name         | Value |
+-----------------------+-------+
| innodb_file_per_table | ON    |
+-----------------------+-------+
1 row in set (0.02 sec)

mysql>
```

Now, let's check the data file for the file-per-table. For example, in the "employees" database, there is a table called "departments".

```
Database changed
mysql> show tables;
+----------------------+
| Tables_in_employees  |
+----------------------+
| current_dept_emp     |
| departments          |
| dept_emp             |
| dept_emp_latest_date |
| dept_manager         |
| employees            |
| salaries             |
| staff                |
| titles               |
+----------------------+
9 rows in set (0.01 sec)

mysql>
```

There should be a data file called "departments.ibd" located in the data directory.

```
[dba@mysqlPerconaSTG mysql]$ ll
total 40452
-rw-r----- 1 mysql mysql     3602 Apr 12 00:36  1.000011
-rw-r----- 1 mysql mysql      180 Apr 12 00:54  1.000012
-rw-r----- 1 mysql mysql      180 Apr 12 01:10  1.000013
-rw-r----- 1 mysql mysql      180 Apr 12 01:22  1.000014
-rw-r----- 1 mysql mysql      180 Apr 14 11:28  1.000015
-rw-r----- 1 mysql mysql      180 May  2 11:04  1.000016
-rw-r----- 1 mysql mysql      180 May 10 22:03  1.000017
-rw-r----- 1 mysql mysql      180 May 10 22:52  1.000018
-rw-r----- 1 mysql mysql      180 May 10 23:03  1.000019
-rw-r----- 1 mysql mysql      180 May 10 23:25  1.000020
-rw-r----- 1 mysql mysql      180 May 11 16:52  1.000021
-rw-r----- 1 mysql mysql      180 May 11 17:14  1.000022
-rw-r----- 1 mysql mysql      180 May 11 17:38  1.000023
-rw-r----- 1 mysql mysql      180 May 11 18:09  1.000024
-rw-r----- 1 mysql mysql      157 May 11 18:15  1.000025
-rw-r----- 1 mysql mysql      165 May 11 18:15  1.index
-rw-r----- 1 mysql mysql       56 Apr  1 11:13  auto.cnf
-rw-r----- 1 mysql mysql      848 Apr  2 10:29  binlog.000001
-rw-r----- 1 mysql mysql      180 Apr  2 10:56  binlog.000002
-rw-r----- 1 mysql mysql       32 Apr  2 10:29  binlog.index
-rw------- 1 mysql mysql     1680 Apr  1 11:13  ca-key.pem
-rw-r--r-- 1 mysql mysql     1120 Apr  1 11:13  ca.pem
-rw-r--r-- 1 mysql mysql     1120 Apr  1 11:13  client-cert.pem
-rw------- 1 mysql mysql     1680 Apr  1 11:13  client-key.pem
drwxr-x--- 2 mysql mysql        6 Apr  9 00:52  database
drwxr-x--- 2 mysql mysql      149 Apr  9 02:00  employees
-rw-r----- 1 mysql mysql   196608 May 10 22:05 '#ib_16384_0.dblwr'
-rw-r----- 1 mysql mysql  8585216 Apr 10 07:33 '#ib_16384_1.dblwr'
-rw-r----- 1 mysql mysql     3539 May 11 17:38  ib_buffer_pool
drwxr-x--- 2 mysql mysql     4096 May 11 18:15 '#innodb_redo'
drwxr-x--- 2 mysql mysql      187 May 11 18:15 '#innodb_temp'
drwxr-x--- 2 mysql mysql      143 Apr  1 11:13  mysql
-rw-r----- 1 mysql mysql 32505856 May 11 18:15  mysql.ibd
drwxr-x--- 2 mysql mysql     8192 Apr  1 11:13  performance_schema
-rw------- 1 mysql mysql     1680 Apr  1 11:13  private_key.pem
-rw-r--r-- 1 mysql mysql      452 Apr  1 11:13  public_key.pem
-rw-r--r-- 1 mysql mysql     1120 Apr  1 11:13  server-cert.pem
-rw------- 1 mysql mysql     1676 Apr  1 11:13  server-key.pem
drwxr-x--- 2 mysql mysql       28 Apr  1 11:13  sys
drwxr-x--- 2 mysql mysql       28 Apr 10 08:04  test1
[dba@mysqlPerconaSTG mysql]$ ls employees/
ls: cannot open directory 'employees/': Permission denied
[dba@mysqlPerconaSTG mysql]$ sudo ls employees/
[sudo] password for dba:
departments.ibd  dept_emp.ibd  dept_manager.ibd  employees.ibd  salaries.ibd  staff.ibd  titles.ibd
[dba@mysqlPerconaSTG mysql]$
```

Dedicated MySQL Server

In the previous lesson, we discussed the memory structure of InnoDB, including components like the buffer pool, log buffer, and redo log. We also demonstrated how to adjust them according to your requirements.

However, there's an option provided by MySQL server called "Dedicated MySQL Server" that allows InnoDB to automatically manage these components and handle the memory structure for you.

When enabled, InnoDB will automatically configure variables such as `innodb_buffer_pool_size`, `innodb_flush_method`, `innodb_log_file_size`, `innodb_log_files_in_group`, and `innodb_redo_log_capacity`.

The `innodb_dedicated_server` system variable accepts values of ON or OFF. By default, this option is OFF.

It's recommended to enable `innodb_dedicated_server` only if the MySQL server instance is running on a dedicated server where it can utilize all available system resources. It's not recommended to enable it if the MySQL instance shares system resources with other applications.

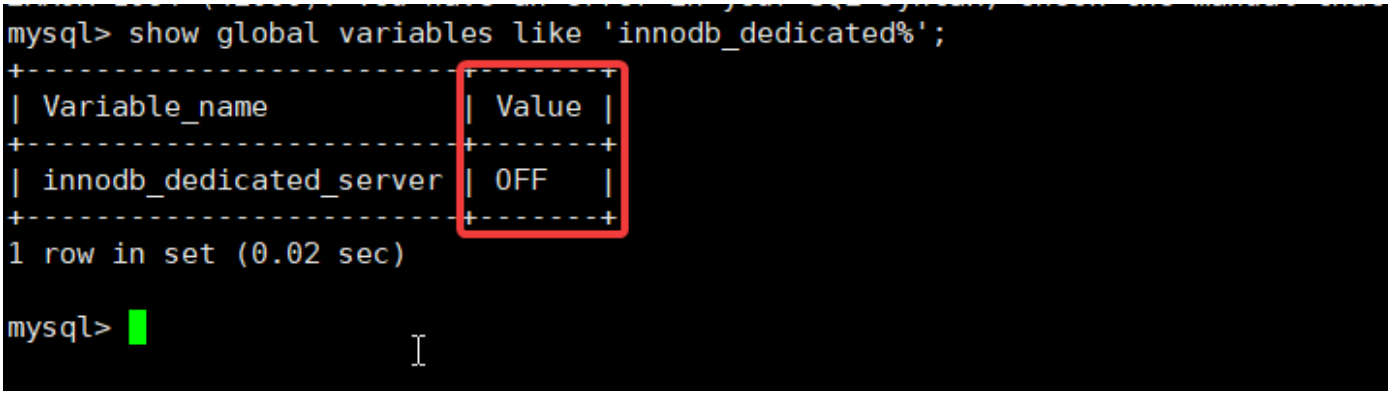
Automatic configuration of buffer pool:

- If memory is less than 1GB, the buffer pool size is set to 128MB.
- If memory is greater than 1GB but less than 4GB, the buffer pool size is set to half of the available memory.
- If memory is greater than 4GB, the buffer pool size is set to 75% of the available memory.

enable innodb_dedicated_server

In this section, we will demonstrate how to enable `innodb_dedicated_server`. Remember that we have modified the flush method and buffer pool size. So, we will check what will happen to our configuration after enabling `innodb_dedicated_server`.
first let's verfy the status of `innodb_dedicated_server` whether its enabled or not
login to MySQL and run the below command

```
show global variables like 'innodb_dedicated%';
```



as mentioned before by default dedicated server is turned OFF
if we check our config file `my.cnf`
and check if any memory structure configuration is added

`sudo vi /etc/my.cnf`

```
# Percona Server template configuration
#
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/8.0/en/server-configuration-defaults.html

[mysqld]
#
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M
#
# Remove the leading "# " to disable binary logging
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
innodb-buffer-pool-size = 256M
innodb-flush-method = O_DIRECT
#innodb-flush-method = O_DIRECT_NO_FSYNC

#skip-innodb-doublewrite
innodb-doublewrite-dir = /doublewrite

innodb-log-file-size = 100M
innodb-log-files-in-group = 2
"/etc/percona/my.cnf" 59L, 1954C
```

in configuration we have modified `innodb_buffer_pool_size`

and flush method

and we have modify the redo log file size

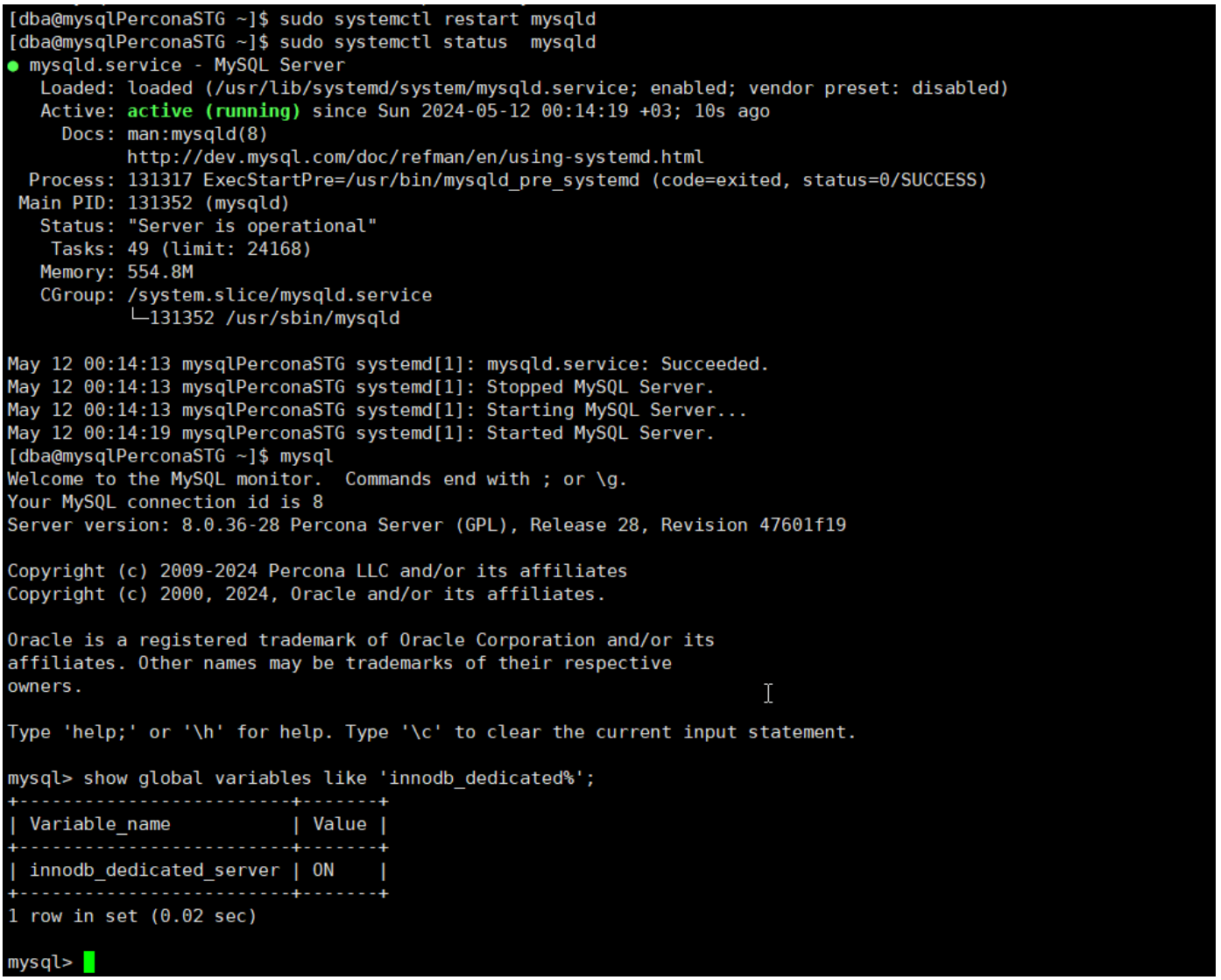
these option need to be disabled so we will comment those options and add option

```
innodb-dedicated-server = ON
```

```
# Binary logging captures changes between backups and is enabled by
# default. It's default setting is log_bin=binlog
# disable_log_bin
#
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
#
# Remove leading # to revert to previous value for default_authentication_plugin,
# this will increase compatibility with older clients. For background, see:
# https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_default_authentication_plugin
# default-authentication-plugin=mysql_native_password


#enable dedicated server

innodb-dedicated-server = ON

datadir=/mysqldata/mysql
socket=/var/lib/mysql/mysql.sock

log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
bind-address=0.0.0.0
log_bin=1
disabled_storage_engines= "MyISAM,CSV,BLACKHOLE,MEMORY,ARCHIVE"
port = 1433
server-id = 3
#innodb-buffer-pool-size = 256M
#innodb-flush-method = O_DIRECT
#innodb-flush-method = O_DIRECT_NO_FSYNC

#skip-innodb-doublewrite
innodb-doublewrite-dir = /doublewrite

#innodb-log-file-size = 100M
#innodb-log-files-in-group = 2
#innodb-log-group-home_dir = /redologs

innodb-flush-log-at-trx-commit = 2

# system tablespace config new path

innodb-data-home-dir  =  /mysqldata/innodb/
-- INSERT --
```

after that restart the server and verify the status

```
systemctl restart mysqld
```



dedicated server is enable let's check what are new value for buffer pool

```
show global variables like 'innodb_buffer_pool%';

show global variables like 'innodb_flush_method%';

show global variables like 'innodb_log%';
```

```
mysql> show global variables like 'innodb_buffer_pool%';
+-------------------------------------+----------------+
| Variable_name                       | Value          |
+-------------------------------------+----------------+
| innodb_buffer_pool_chunk_size       | 134217728      |
| innodb_buffer_pool_dump_at_shutdown | ON             |
| innodb_buffer_pool_dump_now         | OFF            |
| innodb_buffer_pool_dump_pct         | 25             |
| innodb_buffer_pool_filename         | ib_buffer_pool |
| innodb_buffer_pool_in_core_file     | ON             |
| innodb_buffer_pool_instances        | 8              |
| innodb_buffer_pool_load_abort       | OFF            |
| innodb_buffer_pool_load_at_startup  | ON             |
| innodb_buffer_pool_load_now         | OFF            |
| innodb_buffer_pool_size             | 2147483648     |
+-------------------------------------+----------------+
11 rows in set (0.01 sec)

mysql>
```

```
mysql> show global variables like 'innodb_log%';
+-----------------------------------+----------+
| Variable_name                     | Value    |
+-----------------------------------+----------+
| innodb_log_buffer_size            | 16777216 |
| innodb_log_checksums              | ON       |
| innodb_log_compressed_pages       | ON       |
| innodb_log_file_size              | 50331648 |
| innodb_log_files_in_group         | 2        |
| innodb_log_group_home_dir         | ./       |
| innodb_log_spin_cpu_abs_lwm       | 80       |
| innodb_log_spin_cpu_pct_hwm       | 50       |
| innodb_log_wait_for_flush_spin_hwm | 400      |
| innodb_log_write_ahead_size       | 8192     |
| innodb_log_writer_threads         | ON       |
+-----------------------------------+----------+
11 rows in set (0.00 sec)

mysql>
```

```
mysql> show global variables like 'innodb_flush_method%';
+--------------------+------------------+
| Variable_name      | Value            |
+--------------------+------------------+
| innodb_flush_method | O_DIRECT_NO_FSYNC |
+--------------------+------------------+
1 row in set (0.01 sec)
```

8- MySQL Backup & Restore

Physical/Cold Backup

This backup method, also known as a cold backup, involves making a physical copy of MySQL instance files to a backup location.
No backup tool is required; only a simple Unix-based command is used.
It's considered the safest way to preserve your MySQL instance.

In this backup method, you'll back up the following:

- All data directories

- All system-related tablespaces
- All option files

To perform this backup, you need to shut down MySQL using the clean shutdown option.

Perform Physical/Cold Backup

First, we need to perform a clean shutdown, which requires changing the value of the system variable `innodb_fast_shutdown` before using `systemctl` to shut down `mysqld`.

Start by logging in to MySQL and setting the value of `innodb_fast_shutdown` to `0`:

```
SET GLOBAL innodb_fast_shutdown = 0;
```

```
[dba@mysqlPerconaSTG ~]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SET GLOBAL innodb_fast_shutdown = 0;
Query OK, 0 rows affected (0.00 sec)
```

After setting the value of `innodb_fast_shutdown` to `0`, you can stop the MySQL server using the following command:

```
systemctl stop mysqld
```

```
[dba@mysqlPerconaSTG ~]$ systemctl stop mysqld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to stop 'mysqld.service'.
Authenticating as: dba
Password:
==== AUTHENTICATION COMPLETE ====
[dba@mysqlPerconaSTG ~]$
```

Now, we will create a directory called "coldbackup" to hold all the files:

```
sudo mkdir coldbackup
```

```
[dba@mysqlPerconaSTG ~]$ pwd
/home/dba
[dba@mysqlPerconaSTG ~]$ sudo mkdir coldbackup
[sudo] password for dba:
[dba@mysqlPerconaSTG ~]$ ll
total 736524
drwxr-xr-x  2 root  root          6 May 12 00:39 coldbackup
-rw-rw-r--  1 dba   dba    36688498 Apr  9 01:09 master.zip
-rw-rw-r--. 1 dba   dba       17792 Oct 24  2023 mysql80-community-release-el8-9.noarch.rpm
-rw-rw-r--  1 dba   dba   717352960 Aug 15  2019 Percona-Server-8.0.16-7-r613e312-el8-x86_64-bundle.tar
drwxrwxr-x  4 dba   dba        4096 Apr  9 01:34 test_db-master
-rw-rw-r--  1 dba   dba      128273 Apr  4 23:12 x
drwxrwxr-x  2 mysql mysql         6 Apr  2 10:21 xtrabackup_backupfiles
[dba@mysqlPerconaSTG ~]$
```

Next, we will copy the data directory using the `cp` command and place it in the new directory we created before:

```
sudo cp -r /mysqldata/mysql /home/dba/coldbackup/
```

```
[dba@mysqlPerconaSTG ~]$ sudo cp -r /mysqldata/mysql /home/dba/coldbackup/
[dba@mysqlPerconaSTG ~]$ pwd
/home/dba
[dba@mysqlPerconaSTG ~]$ ll coldbackup/
total 4
drwxr-xr-x 10 root root 4096 May 12 00:41 mysql
[dba@mysqlPerconaSTG ~]$
```

Remember to move the system tablespace. The system tablespace path is changed to the directory called `innodb`, so we will take a backup of it as well.

```
sudo cp -r /mysqldata/innodb /home/dba/coldbackup/
```

```
[dba@mysqlPerconaSTG ~]$ sudo cp -r /mysqldata/innodb /home/dba/coldbackup/
[dba@mysqlPerconaSTG ~]$ ll coldbackup/
total 4
drwxr-xr-x  2 root root   90 May 12 00:43 innodb
drwxr-xr-x 10 root root 4096 May 12 00:41 mysql
[dba@mysqlPerconaSTG ~]$
```

also we will copy the `my.cnf` file too

```
sudo cp -r /etc/percona/my.cnf /home/dba/coldbackup/
```

```
[dba@mysqlPerconaSTG ~]$ sudo cp -r /etc/percona/my.cnf /home/dba/coldbackup/
[dba@mysqlPerconaSTG ~]$ ll coldbackup/
total 8
drwxr-xr-x  2 root root   90 May 12 00:43 innodb
-rw-r--r--  1 root root 2016 May 12 00:44 my.cnf
drwxr-xr-x 10 root root 4096 May 12 00:41 mysql
[dba@mysqlPerconaSTG ~]$
```

Restore From Physical/Cold Backup

For testing purposes, I will start up MySQL and drop some tables. Then, we will try to restore this data using the cold backup we have taken.

we will drop empolyees database and test1 databse

```
[dba@mysqlPerconaSTG ~]$ sudo systemctl start mysqld
[dba@mysqlPerconaSTG ~]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| database           |
| employees          |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test1              |
+--------------------+
7 rows in set (0.01 sec)

mysql>
```

```
mysql> drop database employees;
Query OK, 9 rows affected (0.17 sec)

mysql> drop database test1;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| database           |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql>
```

now we will perform restore , remember always to preform clean shutdown

```
SET GLOBAL innodb_fast_shutdown = 0;
```

`systemctl stop mysqld`

```
mysql> SET GLOBAL innodb_fast_shutdown = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> exit
Bye
[dba@mysqlPerconaSTG ~]$ sudo systemctl stop mysqld
[dba@mysqlPerconaSTG ~]$
```

now we will delete the old data directory

`sudo rm -rf mysqldata/mysql`

```
[dba@mysqlPerconaSTG /]$ sudo rm -rf mysqldata/mysql
[dba@mysqlPerconaSTG /]$ ll mysqldata/
total 0
drwxr-xr-x 2 mysql mysql 90 May 12 00:51 innodb
[dba@mysqlPerconaSTG /]$
```

also we will delete `innodb` directory which hold system tablespace

`sudo rm -rf mysqldata/innodb`

```
[dba@mysqlPerconaSTG /]$ sudo rm -rf mysqldata/innodb
[dba@mysqlPerconaSTG /]$ ll mysqldata/
total 0
[dba@mysqlPerconaSTG /]$
```

now we will copy the cold backup and move it to the data diretcory

```
sudo cp -r /home/dba/coldbackup/mysql /mysqldata/


sudo cp -r /home/dba/coldbackup/innodb /mysqldata/
```

```
[dba@mysqlPerconaSTG /]$ sudo cp -r /home/dba/coldbackup/mysql /mysqldata/
[dba@mysqlPerconaSTG /]$ ll mysqldata/
total 4
drwxr-xr-x 10 root root 4096 May 12 01:03 mysql
[dba@mysqlPerconaSTG /]$ sudo cp -r /home/dba/coldbackup/innodb  /mysqldata/
[dba@mysqlPerconaSTG /]$ ll mysqldata/
total 4
drwxr-xr-x  2 root root   90 May 12 01:03 innodb
drwxr-xr-x 10 root root 4096 May 12 01:03 mysql
[dba@mysqlPerconaSTG /]$
```

modify the ownership of the file to be `mysql` user

```
sudo chown -R mysql:mysql  /mysqldata/mysql


sudo chown -R mysql:mysql  /mysqldata/innodb
```

```
[dba@mysqlPerconaSTG /]$ sudo chown -R mysql:mysql  /mysqldata/mysql
[dba@mysqlPerconaSTG /]$ sudo chown -R mysql:mysql  /mysqldata/innodb
[dba@mysqlPerconaSTG /]$ ll mysqldata/
total 4
drwxr-xr-x  2 mysql mysql   90 May 12 01:03 innodb
drwxr-xr-x 10 mysql mysql 4096 May 12 01:03 mysql
[dba@mysqlPerconaSTG /]$
```

now we will startup mysqld and check data if there

```
systemctl start mysqld
```

```
[dba@mysqlPerconaSTG /]$ systemctl start mysqld
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to start 'mysqld.service'.
Authenticating as: dba
Password:
==== AUTHENTICATION COMPLETE ====
[dba@mysqlPerconaSTG /]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| database           |
| employees          |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test1              |
+--------------------+
7 rows in set (0.38 sec)

mysql>
```

Files needed for Cold Backup

- Data directory (DATA DIR)
- System tablespace
- Any option/configuration file with the extension `*.cnf`

Files not part of Cold Backup

- Redo log files
- Doublewrite buffer files
- Binary log files
- Undo tablespaces
- Temporary files

Logical Backups

- Logical backup copies data only.
- It's best used when you only want to take backups of databases or tables.
- It can backup databases and tables.
- Logical backup works by generating SQL statement files in `.sql` format.
- To take a logical backup, a utility is required. When MySQL is installed, two utilities are provided for this purpose:
  - **mysqldump** old utility
  - **mysqlpump** new utility

MySQLDUMP Backup Program

- Logical backup client utility
- Generates SQL statements for reproducing database objects
- Option to backup entire databases, with ability to exclude specific tables
- Table backups with WHERE clause support for selective row backups
- Simultaneous dumping of one or more databases
  syntax
- `mysqldump [options] db_name [table_name} > backup_name.sql`: This command performs a logical backup of the specified database (`db_name`) and optionally specific tables (`table_name`). The output is redirected to a SQL file named `backup_name.sql`.
- `mysqldump [options] db_name [table_name} -where='condition' > backup_name.sql`: Similar to the previous command, this one also backs up a specific database (`db_name`) and optionally specified tables (`table_name`). However, it includes a WHERE clause (`condition`) to selectively backup rows based on certain criteria. The resulting backup is saved to `backup_name.sql`.
- `mysqldump [options] -databases db1 db2 ..db_name > backup_name.sql`: This command performs logical backups of multiple specified databases (`db1`, `db2`, etc.), along with their respective tables. The output is directed to a SQL file named `backup_name.sql`.

- `mysqldump [options] -all-databases > backup_name.sql`: Here, the command performs a logical backup of all databases present on the MySQL server (`-all-databases` option). The resulting backup is stored in a SQL file named `backup_name.sql`.
  -`mysqldump [options] db_name --ignore-table=db.tbl_name > backup_name.sql` allow you to take backup of a database but ignore some table you specify .

Take Backup with MySQLDUMP

1.taking tabel backup

first we will take logical backup of certain table

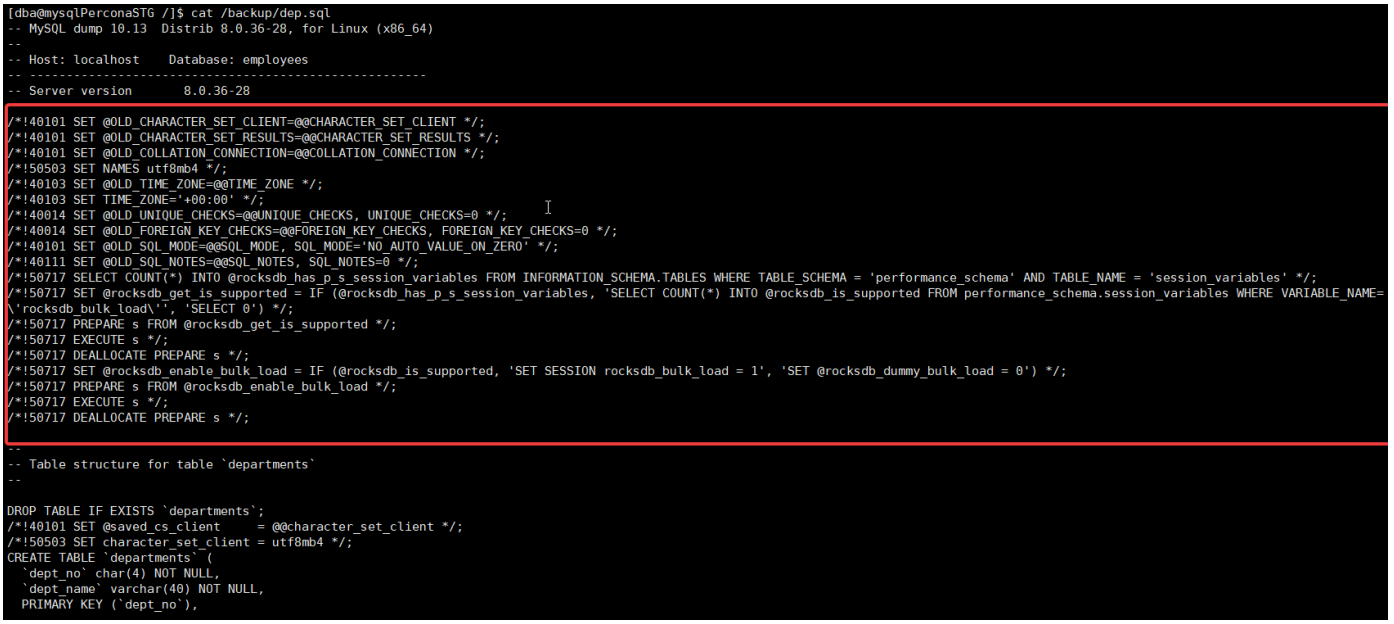
so we will take backup of table called departments inside employees database

```
mysql> show databases ;
+--------------------+
| Database           |
+--------------------+
| database           |
| employees          |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test1              |
+--------------------+
7 rows in set (0.01 sec)

mysql> use employees ;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+----------------------+
| Tables_in_employees  |
+----------------------+
| current_dept_emp     |
| departments          |
| dept_emp             |
| dept_emp_latest_date |
| dept_manager         |
| employees            |
| salaries             |
| staff                |
| titles               |
+----------------------+
9 rows in set (0.00 sec)

mysql>
```

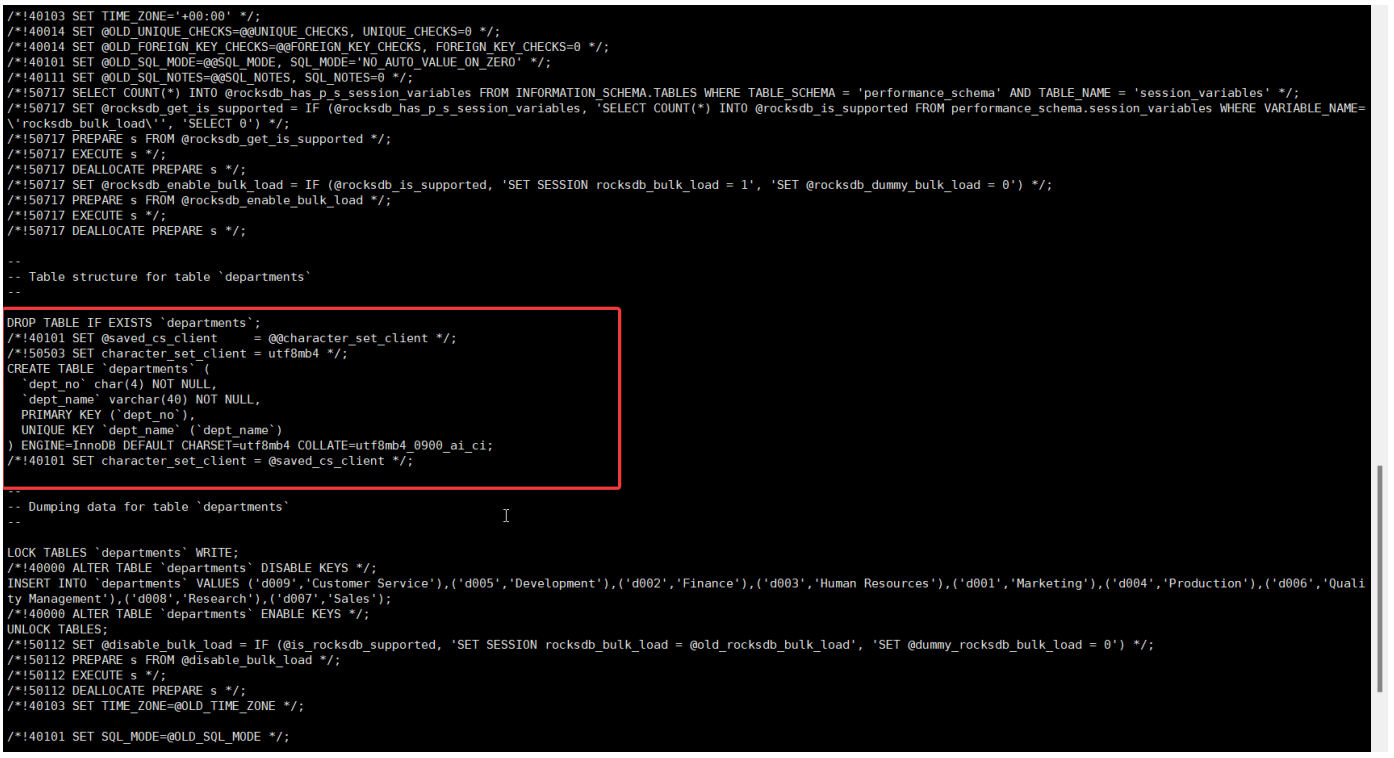
`mysqldump employees departments > backup/dep.sql`

```
[dba@mysqlPerconaSTG /]$ mysqldump  employees departments  > backup/dep.sql
[dba@mysqlPerconaSTG /]$ ll backup/
total 8
drwxr-x---  2 dba  dba     6 Apr  2 10:38 2024_04_02_10-38-03
-rw-rw-r--  1 dba  dba  3219 May 12 09:36 dep.sql
drwxr-x---. 2 root root  124 Mar 30 01:59 instance_dump
-rw-r-----. 1 root root  256 Mar 30 02:04 progress.json
[dba@mysqlPerconaSTG /]$
```

Yes, you can indeed check the contents of the dump file using the `cat` command, which displays the contents of the file directly in the terminal. Alternatively, you can use text editors like `nano` or `vim` to view and edit the contents of the file. So, running `cat /backup/dep.sql` would display the contents of the SQL dump file `dep.sql` located in the `/backup` directory. If you prefer using `nano`, you can run `nano /backup/dep.sql` to open the file in the `nano` text editor for viewing and editing.



In MySQL dump files, you can often find comment lines that provide metadata about the backup. These comments typically include information such as the version of mysqldump used to create the backup and the version of the MySQL server at the time the backup was initiated. Additionally, other internal information may be included in these comments.



In a typical MySQL dump file generated by mysqldump, the structure of the database objects (such as tables) is usually defined using CREATE TABLE statements. These statements first drop the table if it

exists and then recreate it with the specified schema.

After defining the structure of the tables, the dump file typically contains INSERT INTO statements to populate the tables with data. However, it's important to note that the mysqldump command does not inherently lock tables during the backup process. Instead, it typically uses the --lock-tables option to ensure data consistency by obtaining read locks on all tables to be dumped.

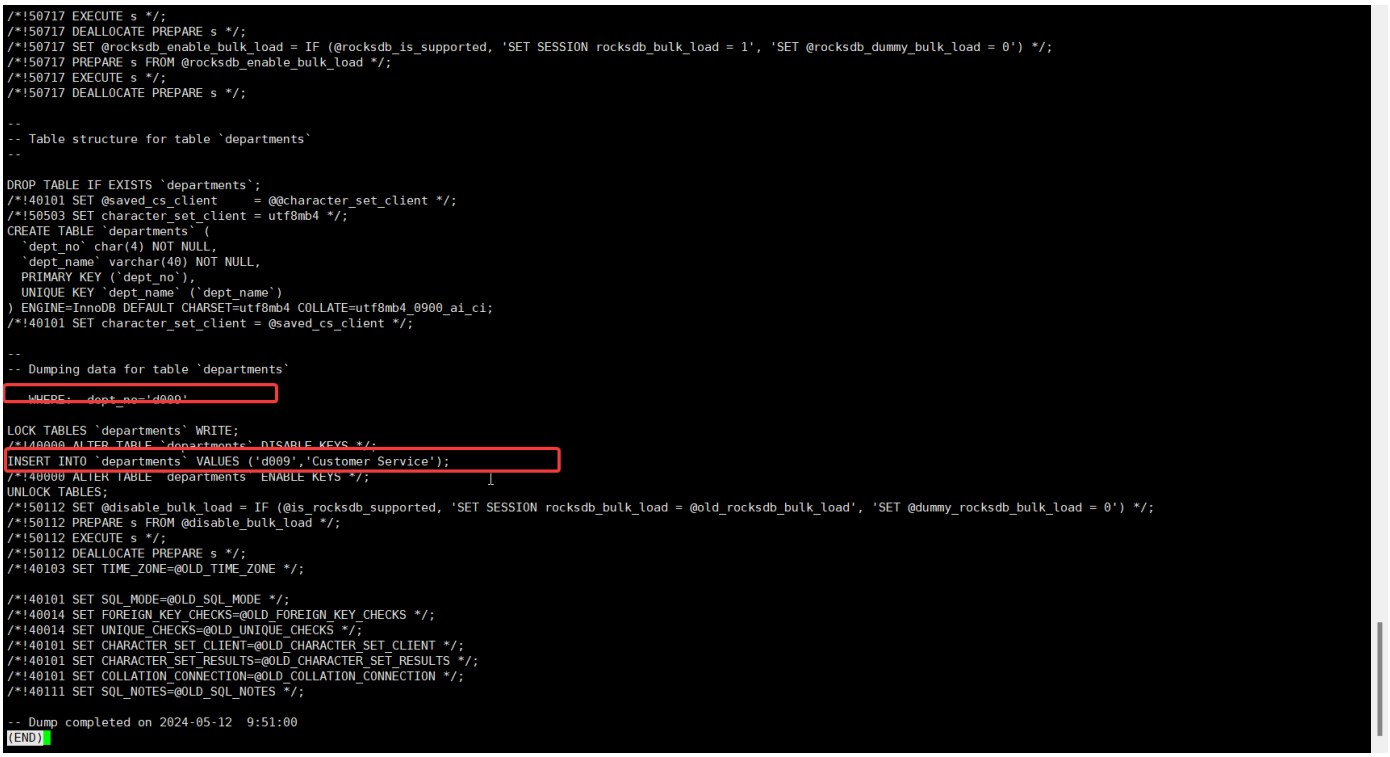
**2.taking backup of table but filter row **

We'll perform a table backup of the "departments" table, selectively including only the row where the department number is 'd009', by applying a filtering WHERE clause

```
mysqldump employees departments --where="dept_no='d009'" > backup/d009.sql
```

```
Database changed
mysql> show tables;
+----------------------+
| Tables_in_employees  |
+----------------------+
| current_dept_emp     |
| departments          |
| dept_emp             |
| dept_emp_latest_date |
| dept_manager         |
| employees            |
| salaries             |
| staff                |
| titles               |
+----------------------+
9 rows in set (0.01 sec)

mysql> select * from departments ;
+---------+--------------------+
| dept_no | dept_name          |
+---------+--------------------+
| d009    | Customer Service   |
| d005    | Development        |
| d002    | Finance            |
| d003    | Human Resources    |
| d001    | Marketing          |
| d004    | Production         |
| d006    | Quality Management |
| d008    | Research           |
| d007    | Sales              |
+---------+--------------------+
9 rows in set (0.00 sec)

mysql> ^C^C^C
```

Upon inspecting the file's content, you'll find that everything remains identical, except for the inclusion of only one row, representing the 'd009' department



**3.taking backup of database but skip table **

We'll back up the "employees" database while excluding the "departments" table using the mysqldump command, and save the result to a file named "nodepartments.sql" in the "backup" directory.

```
mysqldump employees --ignore-table=employees.departments > backup/nodepartments.sql
```

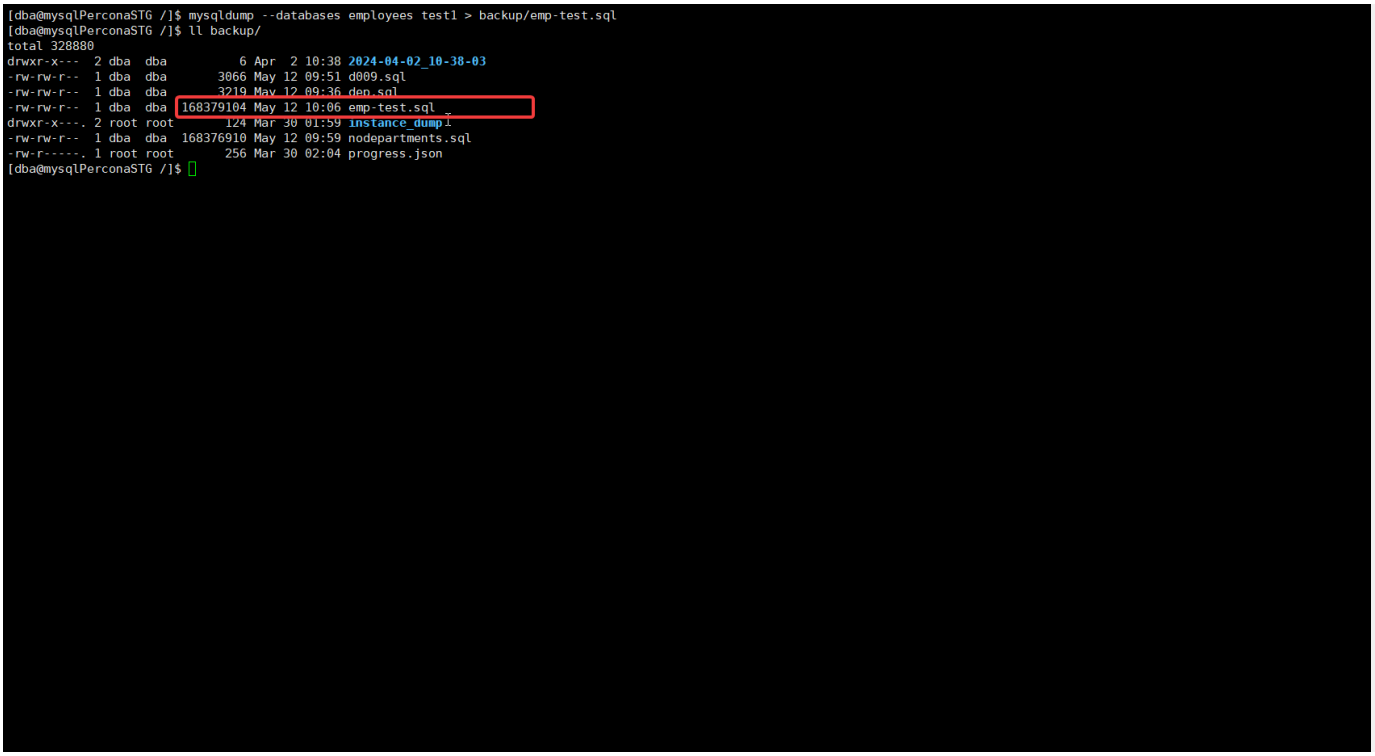
Following the backup, we can employ the `grep` command to search the dump file for `CREATE` statements. By doing so, we should observe the absence of any statement pertaining to the creation of the "departments" table.

```
grep CREATE backup/nodepartments.sql
```

4.taking backup of multiple databases

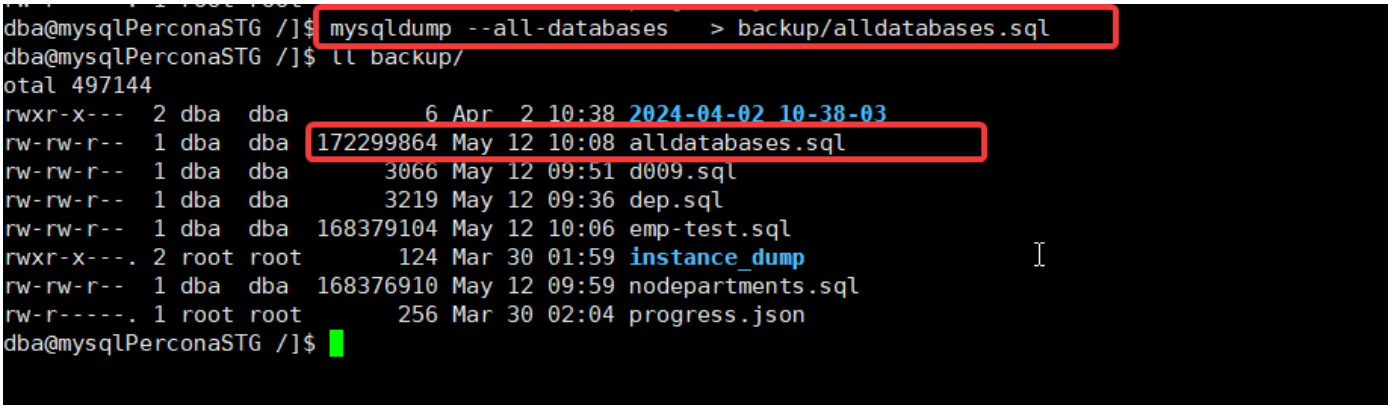
we will take backup of employees and test1 database in single command

```
mysqldump --databases employees test1 > backup/emp-test.sql
```



5.taking backup of all databases

```
mysqldump --all-databases > backup/alldatabases.sql
```



Restoring from MySQLDUMP

we will test restore for table backup we have took for departments table
first we will drop the table and then restore

```
mysql> use employees;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+----------------------+
| Tables_in_employees  |
+----------------------+
| current_dept_emp     |
| departments          |
| dept_emp             |
| dept_emp_latest_date |
| dept_manager         |
| employees            |
| salaries             |
| staff                |
| titles               |
+----------------------+
9 rows in set (0.01 sec)

mysql> drop departments ;
```

to restore we will use the below syntax

`mysql employees < backup/dep.sql`

```
[dba@mysqlPerconaSTG /]$ mysql employees < backup/dep.sql
[dba@mysqlPerconaSTG /]$
```

now we will test database restore

i will drop employees and test1 database and then restore them

`mysql < backup/emp-test.sql`

```
| database           |
| employees          |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test1              |
+--------------------+
7 rows in set (0.01 sec)

mysql> drop database employees ;
Query OK, 9 rows affected (0.23 sec)

mysql> drop database test1 ;
Query OK, 1 row affected (0.03 sec)

mysql> exit
Bye
[dba@mysqlPerconaSTG /]$ mysql < backup/emp-test.sql
[dba@mysqlPerconaSTG /]$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 24
Server version: 8.0.36-28 Percona Server (GPL), Release 28, Revision 47601f19

Copyright (c) 2009-2024 Percona LLC and/or its affiliates
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| database           |
| employees          |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test1              |
+--------------------+
7 rows in set (0.00 sec)

mysql>
```

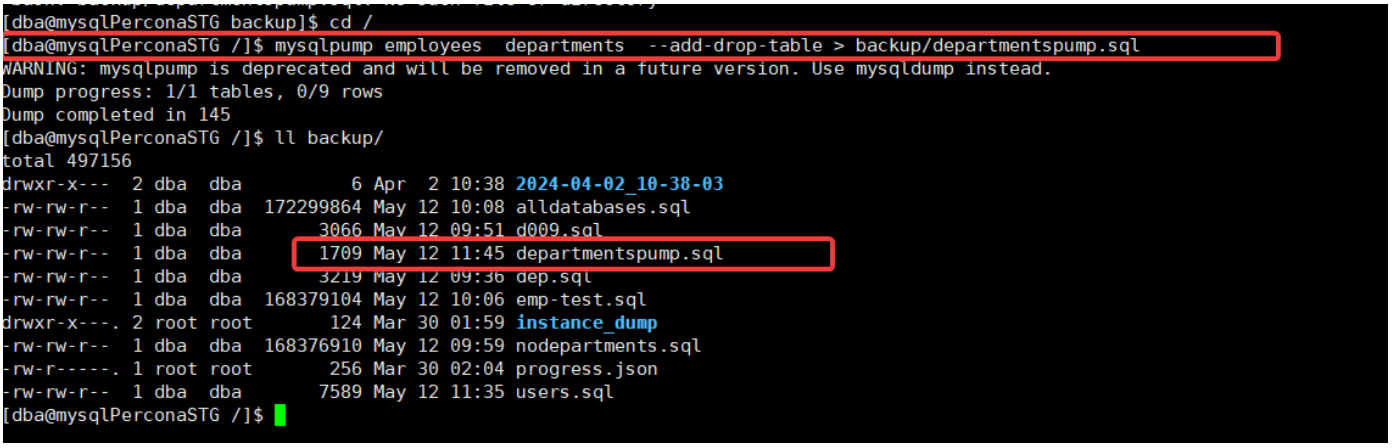
MySQLPUMP Backup Program

- `MySQLpump` offers enhanced functionality and is regarded as a more advanced logical backup tool.
- mysqlpump provides parallel processing of databases to speed up the dump process, utilizing all available CPUs efficiently.
- With mysqlpump, users gain better control over the selection of database objects to dump, including tables, stored procedures, and user accounts.
- `mysqlpump` offers the capability to dump user accounts as account-management statements rather than as insert statements into the MySQL system database.
- `mysqlpump` supports compressed output and provides dump progress information.
- By default, `mysqlpump` dumps all databases.
  Syntax
- `mysqlpump [option] db_name [table_name] --add-drop-table > backup.sql` this option `--add-drop-table` For successful restoration, it's crucial to include the `--add-drop-table` option in the mysqlpump command. Failure to do so may result in restore failures if the tables still exist in the database. Similarly, for database backups, ensure to add the `--add-drop-database` option to mitigate potential restoration issues caused by existing databases.
- `mysqlpump --exclude-databases=% --users > users.sql` the `%` means exclude all databases ,`-users` include the users , this `mysqlpump` syntax will take backup of the users
- `mysqlpump [options] --databases db1 db2 --add-drop-database > backup.sql`
- `mysqlpump [options] all databases > all.sql`

mysqlpump backup and and restore

we will first take table backup with `mysqlpump`

```
mysqlpump employees departments --add-drop-table > backup/departmentspump.sql
```



to restore we will use below command

```
mysql employees < backup/departmentspump.sql
```

```
[dba@mysqlPerconaSTG /]$ mysql employees <  backup/departmentspump.sql
[dba@mysqlPerconaSTG /]$
```

now we will show how to take backup of the database

`mysqlpump --databases employees --add-drop-database > backup/eomployeespump.sql`

```
[dba@mysqlPerconaSTG /]$ mysqlpump --databases employees --add-drop-database > backup/eomployeespump.sql
WARNING: mysqlpump is deprecated and will be removed in a future version. Use mysqldump instead.
Dump progress: 1/1 tables, 0/9 rows
Dump progress: 2/7 tables, 507500/3723820 rows
Dump progress: 5/7 tables, 947377/3723820 rows
Dump progress: 5/7 tables, 1619627/3723820 rows
Dump progress: 6/7 tables, 2346935/3723820 rows
Dump progress: 6/7 tables, 3046435/3723820 rows
Dump progress: 6/7 tables, 3720935/3723820 rows
Dump completed in 6784
[dba@mysqlPerconaSTG /]$
```

to restore the pump we will use the same syntax as we did to restore the table .

`mysql employees < backup/eomployeespump.sql`

```
[dba@mysqlPerconaSTG /]$ mysql employees < backup/eomployeespump.sql
[dba@mysqlPerconaSTG /]$
```

Backing Up MySQL Accounts

It's highly important to maintain regular backups of all MySQL user accounts. It's recommended to schedule backups every two weeks or monthly. These backups prove invaluable in situations where a user account is accidentally dropped, facilitating user recovery.

syntax :

`mysqlpump --exclude-databases=% --users --add-drop-user > users.sql`

```
[dba@mysqlPerconaSTG /]$ mysqlpump --exclude-databases=% --users --add-drop-user > backup/users.sql
WARNING: mysqlpump is deprecated and will be removed in a future version. Use mysqldump instead.
Dump completed in 21
[dba@mysqlPerconaSTG /]$
```

if you check content you will find dump for user account either user or roles

```
alldatabases.sql     departmentspump.sql  emp-test.sql        instance_dump/      progress.json
[dba@mysqlPerconaSTG /]$ cat backup/users.sql
-- Dump created by MySQL pump utility, version: 8.0.36-28, Linux (x86_64)
-- Dump start time: Sun May 12 15:29:21 2024
-- Server version: 8.0.36

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE;
SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
SET @@SESSION.SQL_LOG_BIN= 0;
SET @OLD_TIME_ZONE=@@TIME_ZONE;
SET TIME_ZONE='+00:00';
SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT;
SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS;
SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION;
SET NAMES utf8mb4;
DROP USER 'ADMIN'@'%';
CREATE USER `ADMIN`@`%` IDENTIFIED WITH 'caching_sha2_password' REQUIRE NONE PASSWORD EXPIRE ACCOUNT LOCK PASSWORD HISTORY DEFAULT PASSWORD REUSE INTERVAL DEFAULT PASSWORD REQUIRE CURRENT
 DEFAULT;
GRANT USAGE ON *.* TO `ADMIN`@`%`;
GRANT ALL PRIVILEGES ON `employees`.* TO `ADMIN`@`%`;
DROP USER 'READER'@'%';
CREATE USER `READER`@`%` IDENTIFIED WITH 'caching_sha2_password' REQUIRE NONE PASSWORD EXPIRE ACCOUNT LOCK PASSWORD HISTORY DEFAULT PASSWORD REUSE INTERVAL DEFAULT PASSWORD REQUIRE CURREN
T DEFAULT;
GRANT USAGE ON *.* TO `READER`@`%`;
GRANT SELECT ON `employees`.`employees` TO `READER`@`%`;
DROP USER 'WRITER'@'%';
CREATE USER `WRITER`@`%` IDENTIFIED WITH 'caching_sha2_password' REQUIRE NONE PASSWORD EXPIRE ACCOUNT LOCK PASSWORD HISTORY DEFAULT PASSWORD REUSE INTERVAL DEFAULT PASSWORD REQUIRE CURREN
T DEFAULT;
GRANT USAGE ON *.* TO `WRITER`@`%`;
GRANT INSERT, UPDATE, DELETE ON `employees`.`employees` TO `WRITER`@`%`;
DROP USER 'db_admin'@'%';
CREATE USER `db_admin`@`%` IDENTIFIED WITH 'caching_sha2_password' AS '$A$005$kDc~,1q?/%z#wXhK.v.kHOnthN1IB2xMYvIWAbvv0UTsKQCy8bPOEPD' REQUIRE NONE PASSWORD EXPIRE DEFAULT ACCOUNT UNLOCK
PASSWORD HISTORY DEFAULT PASSWORD REUSE INTERVAL DEFAULT PASSWORD REQUIRE CURRENT DEFAULT;
GRANT USAGE ON *.* TO `db_admin`@`%`;
GRANT `ADMIN`@`%` TO `db_admin`@`%`;
DROP USER 'db_reader'@'%';
CREATE USER `db_reader`@`%` IDENTIFIED WITH 'caching_sha2_password' AS '$A$005$A#y`AO9D.r.#S8\Z%uF5HgtacwNYOzJ5FrG6IwmJCo7rJc8C0Q6hxpHWva0b8' REQUIRE NONE PASSWORD EXPIRE DEFAULT ACCOUNT
UNLOCK PASSWORD HISTORY DEFAULT PASSWORD REUSE INTERVAL DEFAULT PASSWORD REQUIRE CURRENT DEFAULT;
GRANT USAGE ON *.* TO `db_reader`@`%`;
GRANT `READER`@`%` TO `db_reader`@`%`;
DROP USER 'db_writer'@'%';
CREATE USER `db_writer`@`%` IDENTIFIED WITH 'caching_sha2_password' AS '$A$005$?o \'F|dVRywa/0BGJ63gGhk6efdp1g0C2ZRUd4gXpHVBBUzkxSVxgoI48' REQUIRE NONE PASSWORD EXPIRE DEFAULT ACCOUNT UNL
OCK PASSWORD HISTORY DEFAULT PASSWORD REUSE INTERVAL DEFAULT PASSWORD REQUIRE CURRENT DEFAULT;
GRANT USAGE ON *.* TO `db_writer`@`%`;
GRANT `WRITER`@`%` TO `db_writer`@`%`;
DROP USER 'dba'@'%';
```

Restore MySQL Account

To begin the demonstration, let's list the MySQL user accounts using the following command:

```
select user , host from mysql.user;
```

```
mysql> select user , host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| ADMIN            | %         |
| READER           | %         |
| WRITER           | %         |
| db_admin         | %         |
| db_reader        | %         |
| db_writer        | %         |
| dba              | %         |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| root             | localhost |
+------------------+-----------+
11 rows in set (0.01 sec)

mysql>
```

i will drop the user READER

```
drop user READER ;
```

```
mysql> select user , host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| ADMIN            | %         |
| READER           | %         |
| WRITER           | %         |
| db_admin         | %         |
| db_reader        | %         |
| db_writer        | %         |
| dba              | %         |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| root             | localhost |
+------------------+-----------+
11 rows in set (0.01 sec)

mysql> drop user  READER  ;
Query OK, 0 rows affected (0.03 sec)

mysql> select user , host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| ADMIN            | %         |
| WRITER           | %         |
| db_admin         | %         |
| db_reader        | %         |
| db_writer        | %         |
| dba              | %         |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| root             | localhost |
+------------------+-----------+
10 rows in set (0.00 sec)

mysql>
```

best practices if you looking to restore certain user then we can filter the dump file for user using `grep` command and locate SQL statement for that user

`grep READER backup/users.sql`

```
[dba@mysqlPerconaSTG /]$ grep READER  backup/users.sql
DROP USER 'READER'@'%';
CREATE USER `READER`@`%` IDENTIFIED WITH 'caching_sha2_password' REQUIRE NONE PASSWORD EXPIRE ACCOUNT LOCK PASSWORD HISTORY DEFAULT PASSWORD REUSE INTERVAL DEFAULT PASSWORD REQUIRE CURRE
T DEFAULT;
GRANT USAGE ON *.* TO `READER`@`%`;
GRANT SELECT ON `employees`.`employees` TO `READER`@`%`;
GRANT `READER`@`%` TO `db_reader`@`%`;
[dba@mysqlPerconaSTG /]$
```

we can skip the drop statement since we already dropped the user .

now copy the statement and run it on MySQL shell

Compressing MySQL Backups

- MySQLpump offers the capability to compress the output.
- By default, MySQLpump does not compress the output.
- To enable compression, you need to specify the desired compression algorithm.
- The available compression algorithms are LZ4 and ZLIB.
- To compress the output, include the option --compress-output=ZLIB|LZ4.
- To decompress the output, you must have the respective utility installed.
- The utilities Zlib_decompress and lz4_decompress are included in the MySQL distribution.
  Syntax :
  `mysqlpump --database db_name --compress-output > backupcomp.sql`

Taking compress backup

first we will verify which compress algorithm that available
`lz4 --version`

lz4 is not available on the OS
let's confirm the second method if it available on OS

`zlib --version`

```
[dba@mysqlPerconaSTG /]$ lz4  --version
-bash: lz4: command not found
[dba@mysqlPerconaSTG /]$ zlib --version
-bash: zlib: command not found
```

both are not available but if we run `which zlib_decompress`

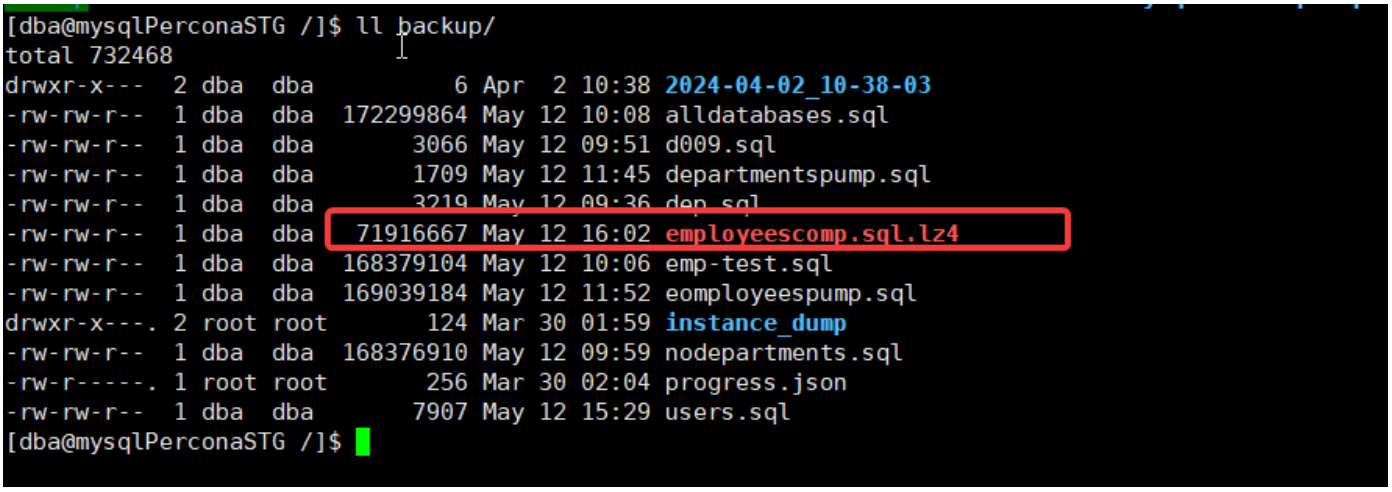
it will show the library is available

```
[dba@mysqlPerconaSTG /]$ which zlib_decompress
/usr/bin/zlib_decompress
```

let's try taking compress backup using lz4 for database employees

`mysqlpump --databases employees --compress-output=lz4 > backup/employeescomp.sql.lz4`

```
[dba@mysqlPerconaSTG /]$ mysqlpump --databases employees --compress-output=lz4 > backup/employeescomp.sql.lz4
WARNING: mysqlpump is deprecated and will be removed in a future version. Use mysqldump instead.
Dump progress: 1/2 tables, 0/331152 rows
Dump progress: 3/7 tables, 587774/3910804 rows
Dump progress: 5/7 tables, 1452127/3910804 rows
Dump progress: 6/7 tables, 2326685/3910804 rows
Dump progress: 6/7 tables, 3260935/3910804 rows
Dump completed in 5075
[dba@mysqlPerconaSTG /]$
```



MySQL Hot Backup

- Also referred to as a physical backup.
- Involves copying database files to a backup device while MySQL remains online.
- Ideal for critical, always-on production applications.
- Particularly suitable for InnoDB tables and transactions.
  Hot Backup Tools

1. `mysqlbackup` oracle
2. `mariabackup` MariaDB
3. `xtrabackup` percona
   1. **`mysqlbackup`**
   Enterprise backup is a superior choice for backups, offering unparalleled speed and efficiency, making it the top choice for your backup strategy. However, it's important to note that MySQL Enterprise Backup, while highly effective, **is not provided for free.**

2. **`mariabackup`**

- Provided by MariaDB as an open-source and free solution.
- Derived from the widely used backup tool XtraBackup.
- Features full support for all major functionalities found in Percona XtraBackup.
  3. **`xtrabackup`**
- Percona provides an open-source hot backup tool for MySQL.
- It ensures databases remain unlocked during backup operations.
- The tool seamlessly performs backups without causing performance disruptions.

XtraBackup Hot Backup Tool

reference link

- A production-grade hot backup tool provided free of cost.

- No licenses are required for usage.
- Completely independent from MySQLbackup or InnoDB hot backup solutions.
- Compatible with both on-premises and cloud environments.
- Enterprise-ready and suitable for automation.
- Supports point-in-time recovery.
- Note that Percona XtraBackup isn't bundled with Percona MySQL or MySQL Community Edition; it needs to be manually installed.

Download & Install XtraBackup

First, you need to verify if Percona XtraBackup is installed or not by using the following command:

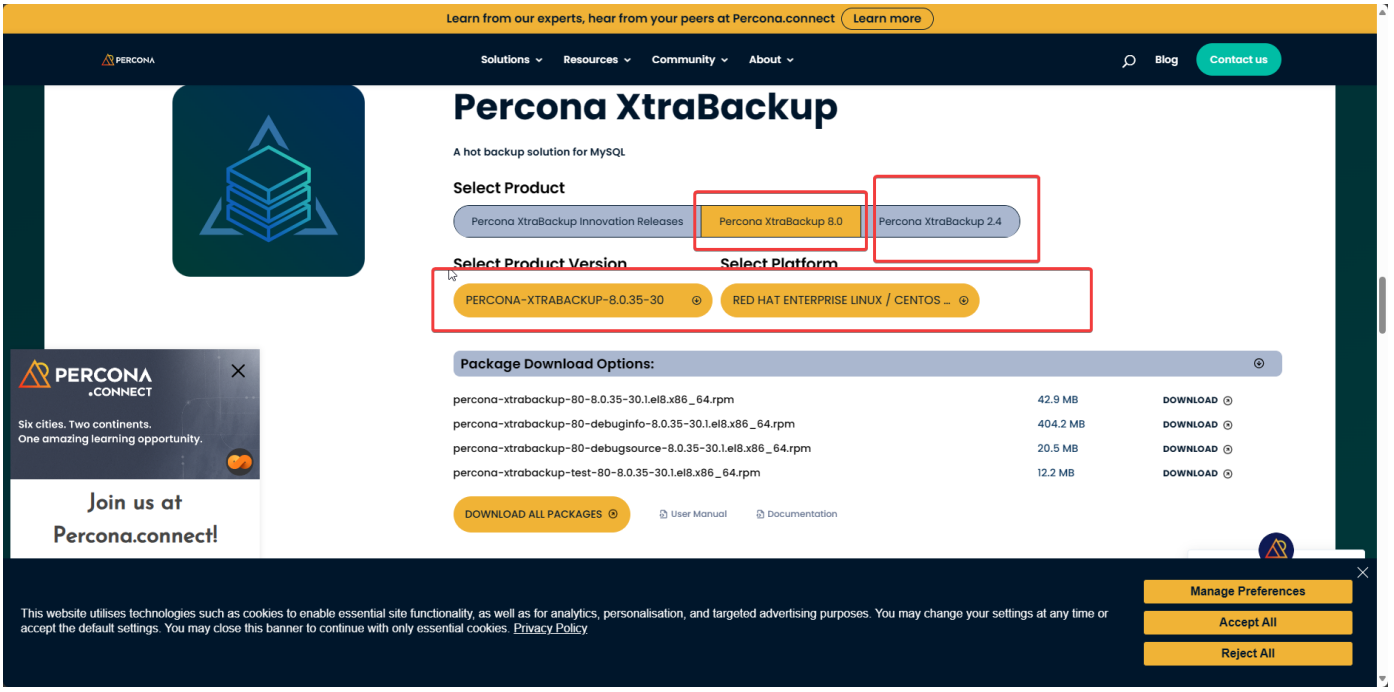
`which xtrabackup`

```
[dba@mysqlPerconaSTG /]$ which xtrabackup
/usr/bin/xtrabackup
[dba@mysqlPerconaSTG /]$
```

```
[dba@Mysqlcom-RHEL-STG ~]$ which xtrabackup
/usr/bin/which: no xtrabackup in (/home/dba/.local/bin:/home/dba/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin)
[dba@Mysqlcom-RHEL-STG ~]$
```

Since we have two installations, one with Community MySQL and the other with Percona MySQL, and you've already installed Percona XtraBackup on the Percona MySQL server, Let's begin by navigating to the Percona website and scrolling down until we locate Percona XtraBackup. Then, we'll select version 8 and choose the platform, which in this case is Red Hat 8.

link



We can use `wget` to download the package directly onto the operating system.

```
wget https://downloads.percona.com/downloads/Percona-XtraBackup-8.0/Percona-XtraBackup-8.0.35-30/binary/redhat/8/x86_64/Percona-XtraBackup-8.0.35-30-r6beb4b49-el8-x86_64-bundle.tar
```

To decompress a tar archive, you can use the tar command with the -xvf options followed by the filename. Here's how you can decompress a tar archive:

`tar -xvf filename.tar`

```
[dba@Mysqlcom-RHEL-STG ~]$ wget https://downloads.percona.com/downloads/Percona-XtraBackup-8.0/Percona-XtraBackup-8.0.35-30/binary/redhat/8/x86_64/Percona-XtraBackup-8.0.35-30-r6beb4b49-el8-x86_64-bundle.tar
--2024-05-12 17:06:53--  https://downloads.percona.com/downloads/Percona-XtraBackup-8.0/Percona-XtraBackup-8.0.35-30/binary/redhat/8/x86_64/Percona-XtraBackup-8.0.35-30-r6beb4b49-el8-x86_64-bundle.tar
Resolving downloads.percona.com (downloads.percona.com)... 147.135.54.159, 2604:2dc0:200:69f::2
Connecting to downloads.percona.com (downloads.percona.com)|147.135.54.159|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 503152640 (480M) [application/octet-stream]
Saving to: 'Percona-XtraBackup-8.0.35-30-r6beb4b49-el8-x86_64-bundle.tar'

Percona-XtraBackup-8.0.35-30-r6beb4b49-el8-x86 100%[===================>] 479.84M  6.14MB/s    in 82s

2024-05-12 17:08:17 (5.88 MB/s) - 'Percona-XtraBackup-8.0.35-30-r6beb4b49-el8-x86_64-bundle.tar' saved [503152640/503152640]

[dba@Mysqlcom-RHEL-STG ~]$
```

`tar -xvf Percona-XtraBackup-8.0.35-30-r6beb4b49-el8-x86_64-bundle.tar`

```
[dba@Mysqlcom-RHEL-STG ~]$ tar -xvf Percona-XtraBackup-8.0.35-30-r6beb4b49-el8-x86_64-bundle.tar
percona-xtrabackup-80-8.0.35-30.1.el8.x86_64.rpm
percona-xtrabackup-80-debuginfo-8.0.35-30.1.el8.x86_64.rpm
percona-xtrabackup-80-debugsource-8.0.35-30.1.el8.x86_64.rpm
percona-xtrabackup-test-80-8.0.35-30.1.el8.x86_64.rpm
[dba@Mysqlcom-RHEL-STG ~]$
```

now we will install xtrabackup using `yum local install`

```
sudo yum localinstall percona-xtrabackup-80-8.0.35-30.1.el8.x86_64.rpm
```

```
================================================================================
 Package                 Architecture    Version                 Repository    Size
================================================================================
Installing:
 percona-xtrabackup-80   x86_64          8.0.35-30.1.el8         @commandline  43 M
Installing dependencies:
 libev                   x86_64          4.24-6.el8              appstream     51 k
 mariadb-connector-c     x86_64          3.1.11-2.el8_3          appstream    199 k
 perl-DBD-MySQL          x86_64          4.046-3.module+el8.9.0+1501+450eec3b appstream 155 k
 perl-DBI                x86_64          1.641-4.module+el8.9.0+1495+f278a004 appstream 739 k
 perl-Math-BigInt        noarch          1:1.9998.11-7.el8       baseos       194 k
 perl-Math-Complex       noarch          1.59-422.el8            baseos       108 k
 zstd                    x86_64          1.4.4-1.el8             appstream    392 k
Enabling module streams:
 perl-DBD-MySQL                          4.046
 perl-DBI                                1.641

Transaction Summary
================================================================================
Install  8 Packages

Total size: 45 M
Total download size: 1.8 M
Installed size: 224 M
Is this ok [y/N]: y
Downloading Packages:
(1/7): libev-4.24-6.el8.x86_64.rpm                  346 kB/s |  51 kB     00:00
(2/7): perl-DBD-MySQL-4.046-3.module+el8.9.0+1501+450eec3b.x86_64.rpm 733 kB/s | 155 kB     00:00
(3/7): mariadb-connector-c-3.1.11-2.el8_3.x86_64.rpm 843 kB/s | 199 kB     00:00
(4/7): zstd-1.4.4-1.el8.x86_64.rpm                  2.3 MB/s | 392 kB     00:00
(5/7): perl-Math-BigInt-1.9998.11-7.el8.noarch.rpm  1.1 MB/s | 194 kB     00:00
(6/7): perl-Math-Complex-1.59-422.el8.noarch.rpm    1.5 MB/s | 108 kB     00:00
(7/7): perl-DBI-1.641-4.module+el8.9.0+1495+f278a004.x86_64.rpm 1.6 MB/s | 739 kB     00:00
--------------------------------------------------------------------------------
Total                                               962 kB/s | 1.8 MB     00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : perl-Math-Complex-1.59-422.el8.noarch                  1/8
  Installing       : perl-Math-BigInt-1:1.9998.11-7.el8.noarch              2/8
  Installing       : perl-DBI-1.641-4.module+el8.9.0+1495+f278a004.x86_64   3/8
  Installing       : zstd-1.4.4-1.el8.x86_64                                4/8
  Installing       : mariadb-connector-c-3.1.11-2.el8_3.x86_64              5/8
  Installing       : perl-DBD-MySQL-4.046-3.module+el8.9.0+1501+450eec3b.x86_64 6/8
  Installing       : libev-4.24-6.el8.x86_64                                7/8
```

now when we run the following command it should show that library for xtrabackup is available

`which xtrabackup`

`xtrabackup --version`

```
[dba@Mysqlcom-RHEL-STG ~]$ which xtrabackup
/usr/bin/xtrabackup
[dba@Mysqlcom-RHEL-STG ~]$
```

```
[dba@Mysqlcom-RHEL-STG ~]$ xtrabackup --version
2024-05-12T17:15:39.530959+03:00 0 [Note] [MY-011825] [Xtrabackup] recognized server arguments: --datadir=/mysqldata --log_bin=/binlog/binlog --log-bin-index=/binlog/binlog.index
xtrabackup version 8.0.35-30 based on MySQL server 8.0.35 Linux (x86_64) (revision id: 6beb4b49)
[dba@Mysqlcom-RHEL-STG ~]$
```

Backup with XtraBackup

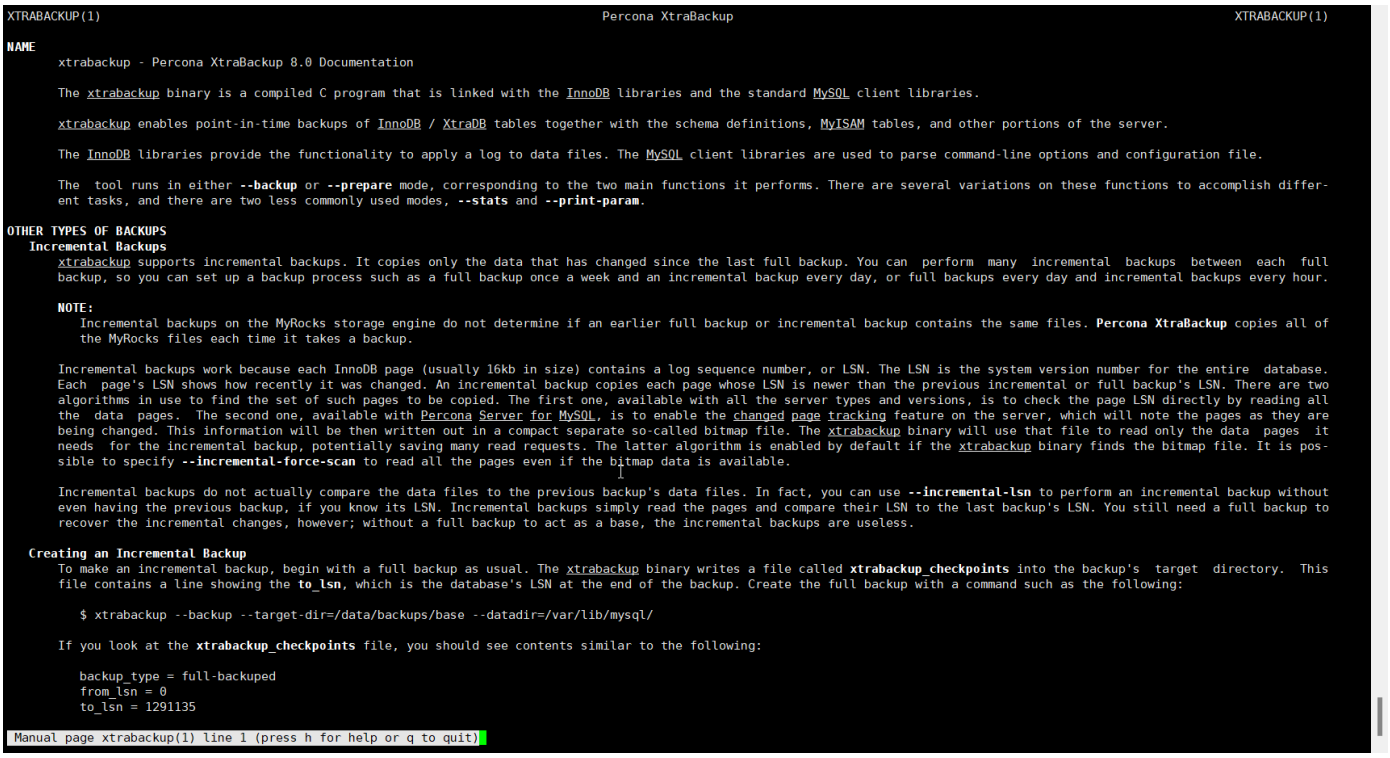
before we use the tool it always recommended to check help section to see the syntax available

```
xtrabackup --help
```

```
xtrabackup version 8.0.35-30 based on MySQL server 8.0.35 Linux (x86_64) (revision id: 6beb4b49)
[dba@Mysqlcom-RHEL-STG ~]$ xtrabackup --help
2024-05-12T17:17:42.613708+03:00 0 [Note] [MY-011825] [Xtrabackup] recognized server arguments: --datadir=/mysqldata --log_bin=/binlog/binlog --log-bin-index=/binlog/binlog.index
xtrabackup version 8.0.35-30 based on MySQL server 8.0.35 Linux (x86_64) (revision id: 6beb4b49)
Open source backup tool for InnoDB and XtraDB

Copyright (C) 2009-2019 Percona LLC and/or its affiliates.
Portions Copyright (C) 2000, 2011, MySQL AB & Innobase Oy. All Rights Reserved.

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation version 2
of the License.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You can download full text of the license on http://www.gnu.org/licenses/gpl-2.0.txt

Usage: [xtrabackup [--defaults-file=#] --backup | xtrabackup [--defaults-file=#] --prepare] [OPTIONS]

Default options are read from the following files in the given order:
/etc/my.cnf /etc/mysql/my.cnf /usr/etc/my.cnf ~/.my.cnf
The following groups are read: xtrabackup mysqld
The following options may be given as the first argument:
--print-defaults        Print the program argument list and exit.
--no-defaults           Don't read default options from any option file,
                        except for login file.
--defaults-file=#       Only read default options from the given file #.
--defaults-extra-file=# Read this file after the global files are read.
--defaults-group-suffix=#
                        Also read groups with concat(group, suffix)
--login-path=#          Read this path from the login file.
  -v, --version       print xtrabackup version information
  --target-dir=name   destination directory
  --backup            take backup to target-dir
  --stats             calc statistic of datadir (offline mysqld is recommended)
  --prepare           prepare a backup for starting mysql server on the backup.
  --export            create files to import to another database when prepare.
  --apply-log-only    stop recovery process not to progress LSN after applying
                      log when prepare.
  --print-param       print parameter of mysqld needed for copyback.
  --use-memory=#      The value is used instead of buffer_pool_size
  --use-free-memory-pct=#
                      This option specifies the percentage of free memory to be
                      used by buffer pool at --prepare stage (default is 0% -
```

also you can use `man` page for xtrabackup

`man xtrabackup`

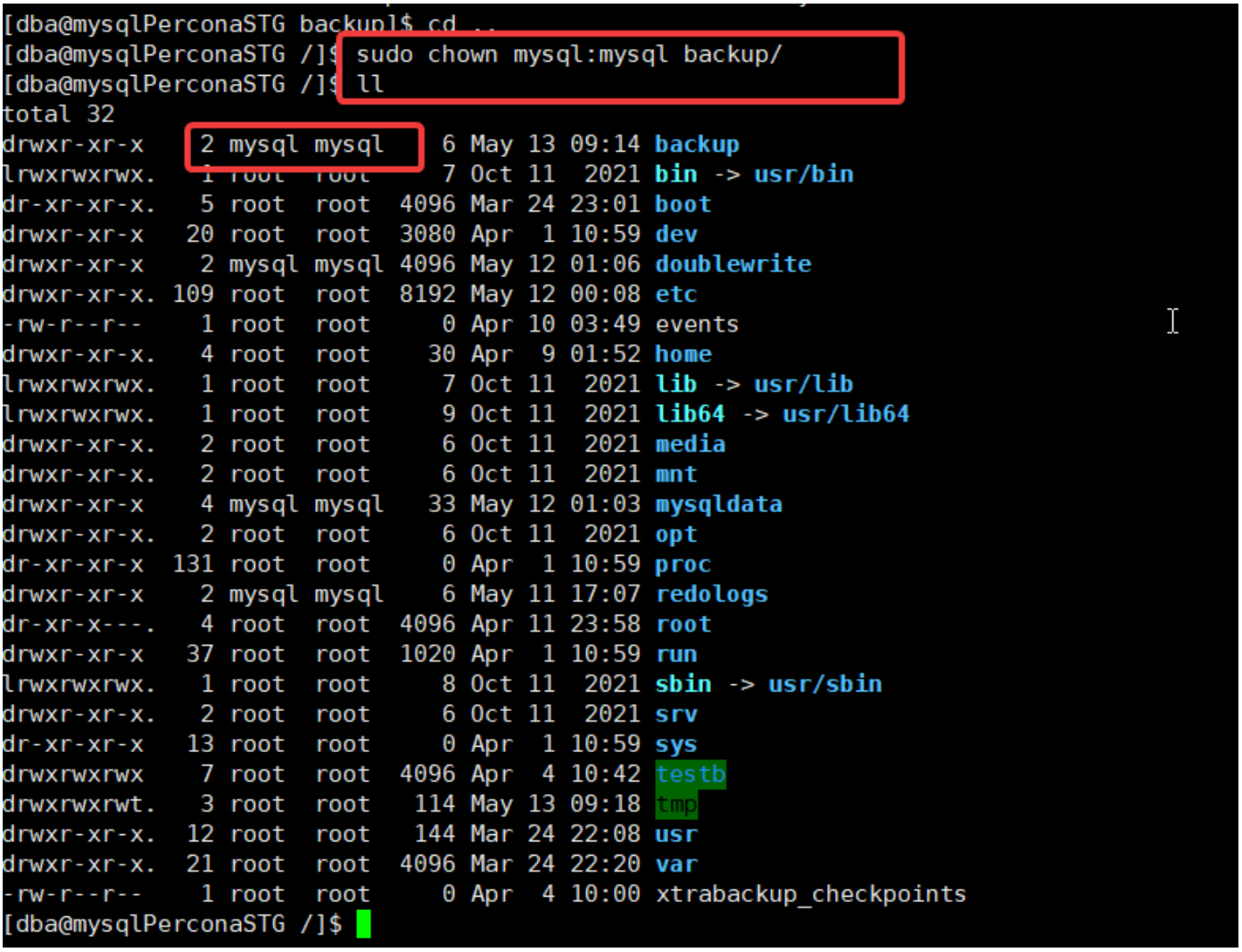


We'll proceed by creating a backup directory. While it's recommended to create this directory on a separate mount point, for testing purposes, we'll create it in the root directory.

`sudo mkdir backup`

make sure to assignee the owner to `mysql`
user

```
sudo chown mysql:mysql backup/
```



To perform a backup, we use the following syntax:

```
sudo xtrabackup  -uroot -p --backup --target-dir=backup/
```

This command will take a hot backup of the running MySQL server and place it in the `backup` directory.

```
[dba@mysqlPerconaSTG /]$
[dba@mysqlPerconaSTG /]$
[dba@mysqlPerconaSTG /]$
[dba@mysqlPerconaSTG /]$ sudo xtrabackup  -uroot -p --backup --target-dir=backup/
2024-05-13T09:23:08.739356+03:00 0 [Note] [MY-011825] [Xtrabackup] recognized server arguments: --datadir=/mysqldata/mysql --log_bin=1 --server-id=3 --innodb_flush_log_at_trx_commit=2 --i
nnodb_data_home_dir=/mysqldata/innodb/ --innodb_data_file_path=ibdata1:12M;ibdata2:10M:autoextend --innodb_undo_directory=/mysqldata/innodb
2024-05-13T09:23:08.739695+03:00 0 [Note] [MY-011825] [Xtrabackup] recognized client arguments: --user=root --password --backup=1 --target-dir=backup/
Enter password:
xtrabackup version 8.0.35-30 based on MySQL server 8.0.35 Linux (x86_64) (revision id: 6beb4b49)
240513 09:23:12  version_check Connecting to MySQL server with DSN 'dbi:mysql:;mysql_read_default_group=xtrabackup' as 'root'  (using password: YES).
240513 09:23:12  version_check Connected to MySQL server
240513 09:23:12  version_check Executing a version check against the server...

# A software update is available:
240513 09:23:14  version_check Done.
2024-05-13T09:23:14.360861+03:00 0 [Note] [MY-011825] [Xtrabackup] Connecting to MySQL server host: localhost, user: root, password: set, port: not set, socket: not set
2024-05-13T09:23:14.377172+03:00 0 [Note] [MY-011825] [Xtrabackup] Using server version 8.0.36-28
2024-05-13T09:23:14.387339+03:00 0 [Note] [MY-011825] [Xtrabackup] Executing LOCK TABLES FOR BACKUP ...
2024-05-13T09:23:14.392766+03:00 0 [Note] [MY-011825] [Xtrabackup] uses posix_fadvise().
2024-05-13T09:23:14.392840+03:00 0 [Note] [MY-011825] [Xtrabackup] cd to /mysqldata/mysql
2024-05-13T09:23:14.392873+03:00 0 [Note] [MY-011825] [Xtrabackup] open files limit requested 0, set to 1024
2024-05-13T09:23:14.394561+03:00 0 [Note] [MY-011825] [Xtrabackup] using the following InnoDB configuration:
2024-05-13T09:23:14.394613+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_data_home_dir = /mysqldata/innodb/
2024-05-13T09:23:14.394633+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_data_file_path = ibdata1:12M;ibdata2:10M:autoextend
2024-05-13T09:23:14.394719+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_log_group_home_dir = ./
2024-05-13T09:23:14.394752+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_log_files_in_group = 2
2024-05-13T09:23:14.394809+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_log_file_size = 50331648
2024-05-13T09:23:14.399063+03:00 0 [Note] [MY-011825] [Xtrabackup] inititialize_service_handles suceeded
2024-05-13T09:23:14.782745+03:00 0 [Note] [MY-011825] [Xtrabackup] Connecting to MySQL server host: localhost, user: root, password: set, port: not set, socket: not set
2024-05-13T09:23:14.795378+03:00 0 [Note] [MY-011825] [Xtrabackup] Redo Log Archiving is not set up.
2024-05-13T09:23:14.924599+03:00 0 [Note] [MY-012953] [InnoDB] Disabling background ibuf IO read threads.
2024-05-13T09:23:14.940674+03:00 1 [Note] [MY-011825] [Xtrabackup] >> log scanned up to (2104305227)
2024-05-13T09:23:15.145702+03:00 0 [Note] [MY-011825] [Xtrabackup] Generating a list of tablespaces
2024-05-13T09:23:15.145858+03:00 0 [Note] [MY-012204] [InnoDB] Scanning './'
2024-05-13T09:23:15.148712+03:00 0 [Note] [MY-012204] [InnoDB] Scanning '/mysqldata/innodb/'
2024-05-13T09:23:15.162873+03:00 0 [Note] [MY-012208] [InnoDB] Completed space ID check of 2 files.
2024-05-13T09:23:15.174995+03:00 0 [Warning] [MY-012091] [InnoDB] Allocated tablespace ID 39 for employees/departments, old maximum was 0
2024-05-13T09:23:15.219287+03:00 0 [Note] [MY-013252] [InnoDB] Using undo tablespace '/mysqldata/innodb/undo_001'.
2024-05-13T09:23:15.239112+03:00 0 [Note] [MY-013252] [InnoDB] Using undo tablespace '/mysqldata/innodb/undo_002'.
2024-05-13T09:23:15.241055+03:00 0 [Note] [MY-012910] [InnoDB] Opened 2 existing undo tablespaces.
2024-05-13T09:23:15.257611+03:00 2 [Note] [MY-011825] [Xtrabackup] Copying /mysqldata/innodb/ibdata1 to /backup/ibdata1
2024-05-13T09:23:15.367411+03:00 2 [Note] [MY-011825] [Xtrabackup] Done: Copying /mysqldata/innodb/ibdata1 to /backup/ibdata1
2024-05-13T09:23:15.375804+03:00 2 [Note] [MY-011825] [Xtrabackup] Copying /mysqldata/innodb/ibdata2 to /backup/ibdata2
2024-05-13T09:23:15.451993+03:00 2 [Note] [MY-011825] [Xtrabackup] Done: Copying /mysqldata/innodb/ibdata2 to /backup/ibdata2
2024-05-13T09:23:15.468584+03:00 2 [Note] [MY-011825] [Xtrabackup] Copying ./sys/sys_config.ibd to /backup/sys/sys_config.ibd
2024-05-13T09:23:15.469765+03:00 2 [Note] [MY-011825] [Xtrabackup] Done: Copying ./sys/sys_config.ibd to /backup/sys/sys_config.ibd
2024-05-13T09:23:15.472570+03:00 2 [Note] [MY-011825] [Xtrabackup] Copying ./test1/continents.ibd to /backup/test1/continents.ibd
2024-05-13T09:23:15.473065+03:00 2 [Note] [MY-011825] [Xtrabackup] Done: Copying ./test1/continents.ibd to /backup/test1/continents.ibd
```

```
2024-05-13T09:23:18.571994+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/memory_summary_b_163.sdi to /backup/performance_schema/memory_summary_b_163.sdi
2024-05-13T09:23:18.572218+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/memory_summary_b_163.sdi to /backup/performance_schema/memory_summary_b_163.sdi
2024-05-13T09:23:18.573479+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/memory_summary_b_164.sdi to /backup/performance_schema/memory_summary_b_164.sdi
2024-05-13T09:23:18.573637+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/memory_summary_b_164.sdi to /backup/performance_schema/memory_summary_b_164.sdi
2024-05-13T09:23:18.574957+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/memory_summary_b_165.sdi to /backup/performance_schema/memory_summary_b_165.sdi
2024-05-13T09:23:18.575193+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/memory_summary_b_165.sdi to /backup/performance_schema/memory_summary_b_165.sdi
2024-05-13T09:23:18.576599+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/memory_summary_b_166.sdi to /backup/performance_schema/memory_summary_b_166.sdi
2024-05-13T09:23:18.576831+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/memory_summary_b_166.sdi to /backup/performance_schema/memory_summary_b_166.sdi
2024-05-13T09:23:18.578233+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/table_handles_167.sdi to /backup/performance_schema/table_handles_167.sdi
2024-05-13T09:23:18.578472+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/table_handles_167.sdi to /backup/performance_schema/table_handles_167.sdi
2024-05-13T09:23:18.579711+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/metadata_locks_168.sdi to /backup/performance_schema/metadata_locks_168.sdi
2024-05-13T09:23:18.579960+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/metadata_locks_168.sdi to /backup/performance_schema/metadata_locks_168.sdi
2024-05-13T09:23:18.580984+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/data_locks_169.sdi to /backup/performance_schema/data_locks_169.sdi
2024-05-13T09:23:18.581221+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/data_locks_169.sdi to /backup/performance_schema/data_locks_169.sdi
2024-05-13T09:23:18.582389+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/data_lock_waits_170.sdi to /backup/performance_schema/data_lock_waits_170.sdi
2024-05-13T09:23:18.582625+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/data_lock_waits_170.sdi to /backup/performance_schema/data_lock_waits_170.sdi
2024-05-13T09:23:18.583894+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/replication_conn_171.sdi to /backup/performance_schema/replication_conn_171.sdi
2024-05-13T09:23:18.584341+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/replication_conn_171.sdi to /backup/performance_schema/replication_conn_171.sdi
2024-05-13T09:23:18.591542+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/replication_grou_172.sdi to /backup/performance_schema/replication_grou_172.sdi
2024-05-13T09:23:18.591797+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/replication_grou_172.sdi to /backup/performance_schema/replication_grou_172.sdi
2024-05-13T09:23:18.593289+03:00 3 [Note] [MY-011825] [Xtrabackup] Copying performance_schema/replication_conn_173.sdi to /backup/performance_schema/replication_conn_173.sdi
2024-05-13T09:23:18.593643+03:00 3 [Note] [MY-011825] [Xtrabackup] Done: Copying performance_schema/replication_conn_173.sdi to /backup/performance_schema/replication_conn_173.sdi
2024-05-13T09:23:18.660342+03:00 0 [Note] [MY-011825] [Xtrabackup] Finished backing up non-InnoDB tables and files
2024-05-13T09:23:18.660405+03:00 0 [Note] [MY-011825] [Xtrabackup] Executing FLUSH NO_WRITE_TO_BINLOG BINARY LOGS
2024-05-13T09:23:18.694255+03:00 0 [Note] [MY-011825] [Xtrabackup] Selecting LSN and binary log position from p_s.log_status
2024-05-13T09:23:18.726737+03:00 0 [Note] [MY-011825] [Xtrabackup] Copying /mysqldata/mysql/1.000028 to /backup/1.000028 up to position 157
2024-05-13T09:23:18.727005+03:00 0 [Note] [MY-011825] [Xtrabackup] Done: Copying /mysqldata/mysql/1.000028 to /backup/1.000028
2024-05-13T09:23:18.728241+03:00 0 [Note] [MY-011825] [Xtrabackup] Writing /backup/1.index
2024-05-13T09:23:18.728381+03:00 0 [Note] [MY-011825] [Xtrabackup] Done: Writing file /backup/1.index
2024-05-13T09:23:18.732213+03:00 0 [Note] [MY-011825] [Xtrabackup] Writing /backup/xtrabackup_binlog_info
2024-05-13T09:23:18.732362+03:00 0 [Note] [MY-011825] [Xtrabackup] Done: Writing file /backup/xtrabackup_binlog_info
2024-05-13T09:23:18.733872+03:00 0 [Note] [MY-011825] [Xtrabackup] Executing FLUSH NO_WRITE_TO_BINLOG ENGINE LOGS...
2024-05-13T09:23:18.736921+03:00 0 [Note] [MY-011825] [Xtrabackup] The latest check point (for incremental): '2104305227'
2024-05-13T09:23:18.736987+03:00 0 [Note] [MY-011825] [Xtrabackup] Stopping log copying thread at LSN 2104305227
2024-05-13T09:23:18.737151+03:00 1 [Note] [MY-011825] [Xtrabackup] Starting to parse redo log at lsn = 2104305167
2024-05-13T09:23:18.739307+03:00 0 [Note] [MY-011825] [Xtrabackup] Executing UNLOCK TABLES
2024-05-13T09:23:18.739583+03:00 0 [Note] [MY-011825] [Xtrabackup] All tables unlocked
2024-05-13T09:23:18.739732+03:00 0 [Note] [MY-011825] [Xtrabackup] Copying ib_buffer_pool to /backup/ib_buffer_pool
2024-05-13T09:23:18.747539+03:00 0 [Note] [MY-011825] [Xtrabackup] Done: Copying ib_buffer_pool to /backup/ib_buffer_pool
2024-05-13T09:23:18.748956+03:00 0 [Note] [MY-011825] [Xtrabackup] Backup created in directory '/backup/'
2024-05-13T09:23:18.748994+03:00 0 [Note] [MY-011825] [Xtrabackup] MySQL binlog position: filename '1.000028', position '157'
2024-05-13T09:23:18.749106+03:00 0 [Note] [MY-011825] [Xtrabackup] Writing /backup/backup-my.cnf
2024-05-13T09:23:18.749249+03:00 0 [Note] [MY-011825] [Xtrabackup] Done: Writing file /backup/backup-my.cnf
2024-05-13T09:23:18.750797+03:00 0 [Note] [MY-011825] [Xtrabackup] Writing /backup/xtrabackup_info
2024-05-13T09:23:18.750996+03:00 0 [Note] [MY-011825] [Xtrabackup] Done: Writing file /backup/xtrabackup_info
2024-05-13T09:23:19.753135+03:00 0 [Note] [MY-011825] [Xtrabackup] Transaction log of lsn (2104305227) to (2104305237) was copied.
2024-05-13T09:23:20.042054+03:00 0 [Note] [MY-011825] [Xtrabackup] completed OK!
[dba@mysqlPerconaSTG /]$
```

prepare backup

In a production environment with numerous transactions, it's essential to prepare the backup using the `--prepare` option. This ensures that the backup is fully consistent, allowing for a seamless restoration process.

The command is similar to backup, but instead of using `--backup`, we replace it with `--prepare`.

```
sudo xtrabackup -uroot -p --prepare --target-dir=backup/
```

```
[root@mysqlPerconaSTG /]# sudo xtrabackup -uroot -p --prepare --target-dir=backup/
2024-05-13T09:31:26.470426+03:00 0 [Note] [MY-011825] [Xtrabackup] recognized server arguments: --innodb_checksum_algorithm=crc32 --innodb_log_checksums=1 --innodb_data_file_path=ibdata1:
12M;ibdata2:10M:autoextend --innodb_log_file_size=50331648 --innodb_page_size=16384 --innodb_undo_directory=/mysqldata/innodb --innodb_undo_tablespaces=2 --server-id=3 --innodb_log_checks
ums=ON --innodb_redo_log_encrypt=0 --innodb_undo_log_encrypt=0
2024-05-13T09:31:26.470683+03:00 0 [Note] [MY-011825] [Xtrabackup] recognized client arguments: --user=root --password --prepare=1 --target-dir=backup/
Enter password:
xtrabackup version 8.0.35-30 based on MySQL server 8.0.35 Linux (x86_64) (revision id: 6beb4b49)
2024-05-13T09:31:30.698766+03:00 0 [Note] [MY-011825] [Xtrabackup] cd to /backup/
2024-05-13T09:31:30.699161+03:00 0 [Note] [MY-011825] [Xtrabackup] This target seems to be not prepared yet.
2024-05-13T09:31:30.748219+03:00 0 [Note] [MY-011825] [Xtrabackup] xtrabackup_logfile detected: size=8388608, start_lsn=(2104305227)
2024-05-13T09:31:30.750984+03:00 0 [Note] [MY-011825] [Xtrabackup] using the following InnoDB configuration for recovery:
2024-05-13T09:31:30.751030+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_data_home_dir = .
2024-05-13T09:31:30.751063+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_data_file_path = ibdata1:12M;ibdata2:10M:autoextend
2024-05-13T09:31:30.751192+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_log_group_home_dir = .
2024-05-13T09:31:30.751265+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_log_files_in_group = 1
2024-05-13T09:31:30.751302+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_log_file_size = 8388608
2024-05-13T09:31:30.752194+03:00 0 [Note] [MY-011825] [Xtrabackup] inititialize_service_handles suceeded
2024-05-13T09:31:30.752805+03:00 0 [Note] [MY-011825] [Xtrabackup] using the following InnoDB configuration for recovery:
2024-05-13T09:31:30.752843+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_data_home_dir = .
2024-05-13T09:31:30.752862+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_data_file_path = ibdata1:12M;ibdata2:10M:autoextend
2024-05-13T09:31:30.752897+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_log_group_home_dir = .
2024-05-13T09:31:30.752916+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_log_files_in_group = 1
2024-05-13T09:31:30.752958+03:00 0 [Note] [MY-011825] [Xtrabackup] innodb_log_file_size = 8388608
2024-05-13T09:31:30.752992+03:00 0 [Note] [MY-011825] [Xtrabackup] Starting InnoDB instance for recovery.
2024-05-13T09:31:30.753013+03:00 0 [Note] [MY-011825] [Xtrabackup] Using 104857600 bytes for buffer pool (set by --use-memory parameter)
2024-05-13T09:31:30.753170+03:00 0 [Note] [MY-012932] [InnoDB] PUNCH HOLE support available
2024-05-13T09:31:30.753230+03:00 0 [Note] [MY-012944] [InnoDB] Uses event mutexes
2024-05-13T09:31:30.753252+03:00 0 [Note] [MY-012945] [InnoDB] GCC builtin __atomic_thread_fence() is used for memory barrier
2024-05-13T09:31:30.753286+03:00 0 [Note] [MY-012948] [InnoDB] Compressed tables use zlib 1.2.13
2024-05-13T09:31:30.762194+03:00 0 [Note] [MY-012951] [InnoDB] Using software crc32.
2024-05-13T09:31:30.763138+03:00 0 [Note] [MY-012203] [InnoDB] Directories to scan './'
2024-05-13T09:31:30.763252+03:00 0 [Note] [MY-012204] [InnoDB] Scanning './'
2024-05-13T09:31:30.769010+03:00 0 [Note] [MY-012208] [InnoDB] Completed space ID check of 12 files.
2024-05-13T09:31:30.770112+03:00 0 [Note] [MY-012955] [InnoDB] Initializing buffer pool, total size = 128.000000M, instances = 1, chunk size =128.000000M
2024-05-13T09:31:30.799826+03:00 0 [Note] [MY-012957] [InnoDB] Completed initialization of buffer pool
2024-05-13T09:31:30.827263+03:00 0 [Note] [MY-011951] [InnoDB] page_cleaner coordinator priority: -20
2024-05-13T09:31:30.836448+03:00 0 [Note] [MY-011954] [InnoDB] page_cleaner worker priority: -20
2024-05-13T09:31:30.862242+03:00 0 [Note] [MY-011954] [InnoDB] page_cleaner worker priority: -20
2024-05-13T09:31:30.873142+03:00 0 [Note] [MY-011954] [InnoDB] page_cleaner worker priority: -20
2024-05-13T09:31:31.005088+03:00 0 [Note] [MY-013883] [InnoDB] The latest found checkpoint is at lsn = 2104305227 in redo log file ./#innodb_redo/#ib_redo0.
2024-05-13T09:31:31.005230+03:00 0 [Note] [MY-012560] [InnoDB] The log sequence number 1427248675 in the system tablespace does not match the log sequence number 2104305227 in the redo lo
g files!
2024-05-13T09:31:31.005305+03:00 0 [Note] [MY-012551] [InnoDB] Database was not shutdown normally!
2024-05-13T09:31:31.005348+03:00 0 [Note] [MY-012552] [InnoDB] Starting crash recovery.
2024-05-13T09:31:31.005881+03:00 0 [Note] [MY-013086] [InnoDB] Starting to parse redo log at lsn = 2104305167, whereas checkpoint_lsn = 2104305227 and start_lsn = 2104305152
2024-05-13T09:31:31.005964+03:00 0 [Note] [MY-012550] [InnoDB] Doing recovery: scanned up to log sequence number 2104305227
2024-05-13T09:31:31.015300+03:00 0 [Note] [MY-013083] [InnoDB] Log background threads are being started...
2024-05-13T09:31:31.027041+03:00 0 [Note] [MY-012532] [InnoDB] Applying a batch of 0 redo log records ...
2024-05-13T09:31:31.027099+03:00 0 [Note] [MY-012535] [InnoDB] Apply batch completed!
2024-05-13T09:31:31.127531+03:00 0 [Note] [MY-013084] [InnoDB] Log background threads are being closed...
2024-05-13T09:31:31.132641+03:00 0 [Note] [MY-013888] [InnoDB] Upgrading redo log: 1032M, LSN=2104305227.
2024-05-13T09:31:31.132685+03:00 0 [Note] [MY-012968] [InnoDB] Starting to delete and rewrite redo log files.
2024-05-13T09:31:31.132758+03:00 0 [Note] [MY-011825] [InnoDB] Removing redo log file: ./#innodb_redo/#ib_redo0
2024-05-13T09:31:31.239398+03:00 0 [Note] [MY-011825] [InnoDB] Creating redo log file at ./#innodb_redo/#ib_redo0_tmp with file_id 0 with size 33554432 bytes
2024-05-13T09:31:31.242839+03:00 0 [Note] [MY-011825] [InnoDB] Renaming redo log file from ./#innodb_redo/#ib_redo0_tmp to ./#innodb_redo/#ib_redo0
2024-05-13T09:31:31.244354+03:00 0 [Note] [MY-012893] [InnoDB] New redo log files created, LSN=2104305676
2024-05-13T09:31:31.244492+03:00 0 [Note] [MY-013083] [InnoDB] Log background threads are being started...
2024-05-13T09:31:31.276542+03:00 0 [Note] [MY-013252] [InnoDB] Using undo tablespace './undo_001'.
2024-05-13T09:31:31.277292+03:00 0 [Note] [MY-013252] [InnoDB] Using undo tablespace './undo_002'.
2024-05-13T09:31:31.280031+03:00 0 [Note] [MY-012910] [InnoDB] Opened 2 existing undo tablespaces.
2024-05-13T09:31:31.280518+03:00 0 [Note] [MY-011980] [InnoDB] GTID recovery trx_no: 30153
2024-05-13T09:31:31.395614+03:00 0 [Note] [MY-013776] [InnoDB] Parallel initialization of rseg complete
2024-05-13T09:31:31.395687+03:00 0 [Note] [MY-013777] [InnoDB] Time taken to initialize rseg using 2 thread: 115542 ms.
2024-05-13T09:31:31.409176+03:00 0 [Note] [MY-012923] [InnoDB] Creating shared tablespace for temporary tables
2024-05-13T09:31:31.409310+03:00 0 [Note] [MY-012265] [InnoDB] Setting file './ibtmp1' size to 12 MB. Physically writing the file full; Please wait ...
2024-05-13T09:31:31.451022+03:00 0 [Note] [MY-012266] [InnoDB] File './ibtmp1' size is now 12 MB.
2024-05-13T09:31:31.451510+03:00 0 [Note] [MY-013627] [InnoDB] Scanning temp tablespace dir:'./#innodb_temp/'
2024-05-13T09:31:31.472603+03:00 0 [Note] [MY-013018] [InnoDB] Created 128 and tracked 128 new rollback segment(s) in the temporary tablespace. 128 are now active.
2024-05-13T09:31:31.473282+03:00 0 [Note] [MY-012976] [InnoDB] 8.0.35 started; log sequence number 2104305686
2024-05-13T09:31:31.473748+03:00 0 [Warning] [MY-012091] [InnoDB] Allocated tablespace ID 1 for sys/sys_config, old maximum was 0
2024-05-13T09:31:31.480116+03:00 0 [Note] [MY-011825] [Xtrabackup] Completed loading of 10 tablespaces into cache in 0.00675774 seconds
2024-05-13T09:31:31.599117+03:00 0 [Note] [MY-011825] [Xtrabackup] Time taken to build dictionary: 0.118933 seconds
2024-05-13T09:31:31.605732+03:00 0 [Note] [MY-011825] [Xtrabackup] starting shutdown with innodb_fast_shutdown = 1
2024-05-13T09:31:31.606216+03:00 0 [Note] [MY-012330] [InnoDB] FTS optimize thread exiting.
2024-05-13T09:31:32.601617+03:00 0 [Note] [MY-013072] [InnoDB] Starting shutdown...
2024-05-13T09:31:32.627301+03:00 0 [Note] [MY-013084] [InnoDB] Log background threads are being closed...
2024-05-13T09:31:32.657161+03:00 0 [Note] [MY-012980] [InnoDB] Shutdown completed; log sequence number 2104305686
2024-05-13T09:31:32.662811+03:00 0 [Note] [MY-011825] [Xtrabackup] completed OK!
[root@mysqlPerconaSTG /]#
```

XtraBackup Backup Files

in this section we will demonstrate what xtrabackup file has been create .

below is the whole xtrabackup output

```
[root@mysqlPerconaSTG backup]# ll
total 132140
-rw-r----- 1 root root      157 May 13 09:23  1.000028
-rw-r----- 1 root root       11 May 13 09:23  1.index
-rw-r----- 1 root root      474 May 13 09:23  backup-my.cnf
drwxr-x--- 2 root root       20 May 13 09:23  database
drwxr-x--- 2 root root      149 May 13 09:23  employees
-rw-r----- 1 root root     3539 May 13 09:23  ib_buffer_pool
-rw-r----- 1 root root 12582912 May 13 09:31  ibdata1
-rw-r----- 1 root root 10485760 May 13 09:23  ibdata2
-rw-r----- 1 root root 12582912 May 13 09:31  ibtmp1
drwxr-x--- 2 root root        6 May 13 09:31 '#innodb_redo'
drwxr-x--- 2 root root      143 May 13 09:23  mysql
-rw-r----- 1 root root 32505856 May 13 09:23  mysql.ibd
drwxr-x--- 2 root root     8192 May 13 09:23  performance_schema
drwxr-x--- 2 root root       28 May 13 09:23  sys
drwxr-x--- 2 root root       28 May 13 09:23  test1
-rw-r----- 1 root root 16777216 May 13 09:23  undo_001
-rw-r----- 1 root root 16777216 May 13 09:23  undo_002
-rw-r----- 1 root root       13 May 13 09:23  xtrabackup_binlog_info
-rw-r----- 1 root root      140 May 13 09:31  xtrabackup_checkpoints
-rw-r----- 1 root root      460 May 13 09:23  xtrabackup_info
-rw-r----- 1 root root 33554432 May 13 09:31  xtrabackup_logfile
-rw-r----- 1 root root       39 May 13 09:31  xtrabackup_tablespaces
[root@mysqlPerconaSTG backup]#
```

You'll discover our database stored as directories within the data directory. Additionally, xtrabackup backs up the system tablespace contained in `ibdata1` and `ibdata2`, along with the redo log files. It also includes backups of the undo tablespaces `undo_002` and `undo_001`, and points to the latest binary log `1.000028`. Furthermore, it captures an index file pointing to the latest binary log and takes a backup of the `my.cnf` configuration file.

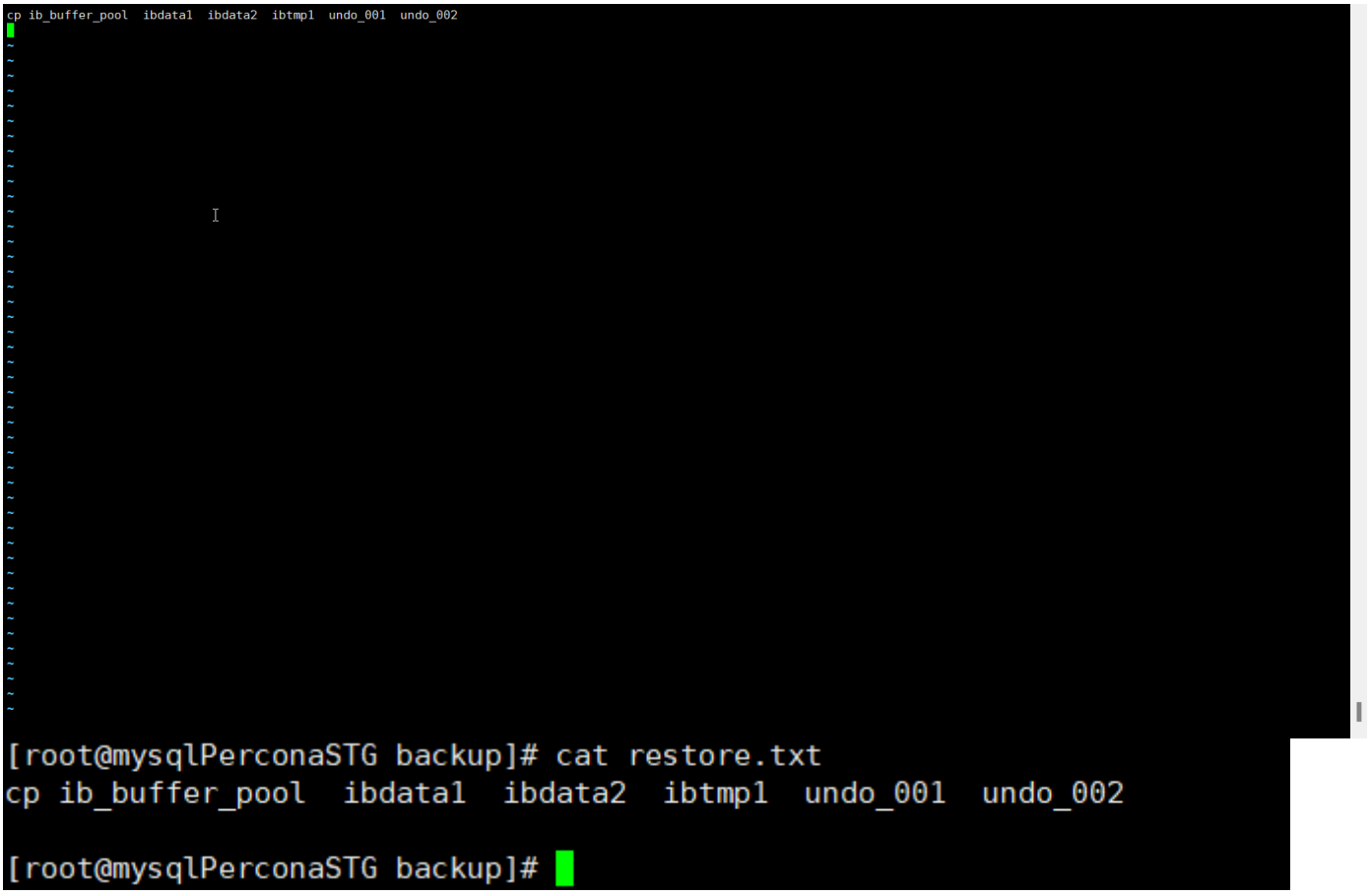
Preparing Hot Backup Restore

before we restore percona xtrabackup it is very important to prepare for the restore .
first we match double buffer file and system tablespace and see if we have backup of them

```
[root@mysqlPerconaSTG backup]# ll
total 132140
-rw-r----- 1 root root      157 May 13 09:23  1.000028
-rw-r----- 1 root root       11 May 13 09:23  1.index
-rw-r----- 1 root root      474 May 13 09:23  backup-my.cnf
drwxr-x--- 2 root root       20 May 13 09:23  database
drwxr-x--- 2 root root      149 May 13 09:23  employees
-rw-r----- 1 root root     3539 May 13 09:23  ib_buffer_pool
-rw-r----- 1 root root 12582912 May 13 09:31  ibdata1
-rw-r----- 1 root root 10485760 May 13 09:23  ibdata2
-rw-r----- 1 root root 12582912 May 13 09:31  ibtmp1
drwxr-x--- 2 root root        6 May 13 09:31 '#innodb_redo'
drwxr-x--- 2 root root      143 May 13 09:23  mysql
-rw-r----- 1 root root 32505856 May 13 09:23  mysql.ibd
drwxr-x--- 2 root root     8192 May 13 09:23  performance_schema
drwxr-x--- 2 root root       28 May 13 09:23  sys
drwxr-x--- 2 root root       28 May 13 09:23  test1
-rw-r----- 1 root root 16777216 May 13 09:23  undo_001
-rw-r----- 1 root root 16777216 May 13 09:23  undo_002
-rw-r----- 1 root root       13 May 13 09:23  xtrabackup_binlog_info
-rw-r----- 1 root root      140 May 13 09:31  xtrabackup_checkpoints
-rw-r----- 1 root root      460 May 13 09:23  xtrabackup_info
-rw-r----- 1 root root 33554432 May 13 09:31  xtrabackup_logfile
-rw-r----- 1 root root       39 May 13 09:31  xtrabackup_tablespaces
[root@mysqlPerconaSTG backup]# ls /mysqldata/innodb/
ib_buffer_pool  ibdata1  ibdata2  ibtmp1  undo_001  undo_002
[root@mysqlPerconaSTG backup]#
```

so we create restore file and past cp command in it

```
cp ib_buffer_pool ibdata1 ibdata2 ibtmp1 undo_001 undo_002
```



Restore From Hot Backup

reference link

here are the steps to restore a backup using xtrabackup:

1. Stop MySQL services.
2. Remove the contents of the data directory.
3. Copy everything related to the data directory from the xtrabackup to the data directory.
4. Recheck the ownership of the data directory.
5. Remove any existing binary logs.
6. Start MySQL services.

1. stop MySQL services

```
sudo systemctl stop mysql
sudo systemctl status  mysql
```

```
[root@mysqlPerconaSTG backup]# sudo systemctl stop mysql
[root@mysqlPerconaSTG backup]# sudo systemctl status  mysql
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: inactive (dead) since Mon 2024-05-13 10:14:06 +03; 5s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 131692 ExecStart=/usr/sbin/mysqld $MYSQLD_OPTS (code=exited, status=0/SUCCESS)
  Process: 131658 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 131692 (code=exited, status=0/SUCCESS)
   Status: "Server shutdown complete"

May 12 01:06:13 mysqlPerconaSTG systemd[1]: Starting MySQL Server...
May 12 01:06:21 mysqlPerconaSTG systemd[1]: Started MySQL Server.
May 13 10:14:05 mysqlPerconaSTG systemd[1]: Stopping MySQL Server...
May 13 10:14:06 mysqlPerconaSTG systemd[1]: mysqld.service: Succeeded.
May 13 10:14:06 mysqlPerconaSTG systemd[1]: Stopped MySQL Server.
[root@mysqlPerconaSTG backup]#
```

2. remove the content of Data Dir

```
rm -rf /mysqldata/mysql/*
rm -rf /mysqldata/innodb/*
```

```
[root@mysqlPerconaSTG /]# rm -rf /mysqldata/mysql/*
[root@mysqlPerconaSTG /]# ll  /mysqldata/mysql/
total 0
[root@mysqlPerconaSTG /]# rm -rf /mysqldata/innodb/*
[root@mysqlPerconaSTG /]# ll  /mysqldata/innodb/
total 0
[root@mysqlPerconaSTG /]#
```

3. Copy everything related to the data directory from the xtrabackup to the data directory.

first list content of percona xtrabackup files

`ll backup/`

not all the file are needed some files are related to perocna xtrabackup

so only copy the file that are highlighted

```
[root@mysqlPerconaSTG /]# ll backup/
total 132144
-rw-r----- 1 root root      157 May 13 09:23  1.000028
-rw-r----- 1 root root       11 May 13 09:23  1.index
-rw-r----- 1 root root      474 May 13 09:23  backup-my.cnf
drwxr-x--- 2 root root       20 May 13 09:23  database
drwxr-x--- 2 root root      149 May 13 09:23  employees
-rw-r----- 1 root root     3539 May 13 09:23  ib_buffer_pool
-rw-r----- 1 root root 12582912 May 13 09:31  ibdata1
-rw-r----- 1 root root 10485760 May 13 09:23  ibdata2
-rw-r----- 1 root root 12582912 May 13 09:31  ibtmp1
drwxr-x--- 2 root root        6 May 13 09:31 '#innodb_redo'
drwxr-x--- 2 root root      143 May 13 09:23  mysql
-rw-r----- 1 root root 32505856 May 13 09:23  mysql.ibd
drwxr-x--- 2 root root     8192 May 13 09:23  performance_schema
-rw-r--r-- 1 root root       65 May 13 10:01  restore.txt
drwxr-x--- 2 root root       28 May 13 09:23  sys
drwxr-x--- 2 root root       28 May 13 09:23  test1
-rw-r----- 1 root root 16777216 May 13 09:23  undo_001
-rw-r----- 1 root root 16777216 May 13 09:23  undo_002
-rw-r----- 1 root root       13 May 13 09:23  xtrabackup_binlog_info
-rw-r----- 1 root root      140 May 13 09:31  xtrabackup_checkpoints
-rw-r----- 1 root root      460 May 13 09:23  xtrabackup_info
-rw-r----- 1 root root 33554432 May 13 09:31  xtrabackup_logfile
-rw-r----- 1 root root       39 May 13 09:31  xtrabackup_tablespaces
[root@mysqlPerconaSTG /]#
```

```
cp 1.000028 1.index backup-my.cnf ib_buffer_pool employees  database ibdata1 ibdata2 ibtmp1 mysql.ibd  mysql  '#innodb_redo' sys  performance_schema test1  undo_001  undo_002 /mysqldata/mysql/
```

```
[root@mysqlPerconaSTG backup]# cd  /mysqldata/mysql/
[root@mysqlPerconaSTG mysql]# ll
total 99356
-rw-r----- 1 root root      157 May 13 11:01  1.000028
-rw-r----- 1 root root       11 May 13 11:01  1.index
-rw-r----- 1 root root      474 May 13 11:01  backup-my.cnf
drwxr-x--- 2 root root       20 May 13 11:01  database
drwxr-x--- 2 root root      149 May 13 11:01  employees
-rw-r----- 1 root root     3539 May 13 11:01  ib_buffer_pool
-rw-r----- 1 root root 12582912 May 13 11:01  ibdata1
-rw-r----- 1 root root 10485760 May 13 11:01  ibdata2
-rw-r----- 1 root root 12582912 May 13 11:01  ibtmp1
drwxr-x--- 2 root root        6 May 13 11:01 '#innodb_redo'
drwxr-x--- 2 root root      143 May 13 11:01  mysql
-rw-r----- 1 root root 32505856 May 13 11:01  mysql.ibd
drwxr-x--- 2 root root     8192 May 13 11:01  performance_schema
drwxr-x--- 2 root root       28 May 13 11:01  sys
drwxr-x--- 2 root root       28 May 13 11:01  test1
-rw-r----- 1 root root 16777216 May 13 11:01  undo_001
-rw-r----- 1 root root 16777216 May 13 11:01  undo_002
[root@mysqlPerconaSTG mysql]#
```

copy the doube buffer system tablespace and undo log to correct directory , also insure to remove them after copying

```
[root@mysqlPerconaSTG mysql]# ll
total 99356
-rw-r----- 1 root root      157 May 13 11:01  1.000028
-rw-r----- 1 root root       11 May 13 11:01  1.index
-rw-r----- 1 root root      474 May 13 11:01  backup-my.cnf
drwxr-x--- 2 root root       20 May 13 11:01  database
drwxr-x--- 2 root root      149 May 13 11:01  employees
-rw-r----- 1 root root     3539 May 13 11:01  ib_buffer_pool
-rw-r----- 1 root root 12582912 May 13 11:01  ibdata1
-rw-r----- 1 root root 10485760 May 13 11:01  ibdata2
-rw-r----- 1 root root 12582912 May 13 11:01  ibtmp1
drwxr-x--- 2 root root        6 May 13 11:01 '#innodb_redo'
drwxr-x--- 2 root root      143 May 13 11:01  mysql
-rw-r----- 1 root root 32505856 May 13 11:01  mysql.ibd
drwxr-x--- 2 root root     8192 May 13 11:01  performance_schema
drwxr-x--- 2 root root       28 May 13 11:01  sys
drwxr-x--- 2 root root       28 May 13 11:01  test1
-rw-r----- 1 root root 16777216 May 13 11:01  undo_001
-rw-r----- 1 root root 16777216 May 13 11:01  undo_002
[root@mysqlPerconaSTG mysql]# cp -r undo_001 undo_002 ibdata1 ibdata2  /mysqldata/innodb/
[root@mysqlPerconaSTG mysql]# ll /mysqldata/innodb/
total 55296
-rw-r----- 1 root root 12582912 May 13 11:04 ibdata1
-rw-r----- 1 root root 10485760 May 13 11:04 ibdata2
-rw-r----- 1 root root 16777216 May 13 11:04 undo_001
-rw-r----- 1 root root 16777216 May 13 11:04 undo_002
[root@mysqlPerconaSTG mysql]#
```

```
[root@mysqlPerconaSTG backup]# cd  /mysqldata/mysql/
[root@mysqlPerconaSTG mysql]# ll
total 99356
-rw-r----- 1 root root      157 May 13 11:01  1.000028
-rw-r----- 1 root root       11 May 13 11:01  1.index
-rw-r----- 1 root root      474 May 13 11:01  backup-my.cnf
drwxr-x--- 2 root root       20 May 13 11:01  database
drwxr-x--- 2 root root      149 May 13 11:01  employees
-rw-r----- 1 root root     3539 May 13 11:01  ib_buffer_pool
-rw-r----- 1 root root 12582912 May 13 11:01  ibdata1
-rw-r----- 1 root root 10485760 May 13 11:01  ibdata2
-rw-r----- 1 root root 12582912 May 13 11:01  ibtmp1
drwxr-x--- 2 root root        6 May 13 11:01 '#innodb_redo'
drwxr-x--- 2 root root      143 May 13 11:01  mysql
-rw-r----- 1 root root 32505856 May 13 11:01  mysql.ibd
drwxr-x--- 2 root root     8192 May 13 11:01  performance_schema
drwxr-x--- 2 root root       28 May 13 11:01  sys
drwxr-x--- 2 root root       28 May 13 11:01  test1
-rw-r----- 1 root root 16777216 May 13 11:01  undo_001
-rw-r----- 1 root root 16777216 May 13 11:01  undo_002
[root@mysqlPerconaSTG mysql]# cp -r undo_001 undo_002 ibdata1 ibdata2  /mysqldata/innodb/
[root@mysqlPerconaSTG mysql]# ll /mysqldata/innodb/
total 55296
-rw-r----- 1 root root 12582912 May 13 11:04 ibdata1
-rw-r----- 1 root root 10485760 May 13 11:04 ibdata2
-rw-r----- 1 root root 16777216 May 13 11:04 undo_001
-rw-r----- 1 root root 16777216 May 13 11:04 undo_002
[root@mysqlPerconaSTG mysql]# cp -r ib_buffer_pool
cp: missing destination file operand after 'ib_buffer_pool'
Try 'cp --help' for more information.
[root@mysqlPerconaSTG mysql]# cp -r ib_buffer_pool ibtmp1 /mysqldata/innodb/
[root@mysqlPerconaSTG mysql]#
[root@mysqlPerconaSTG mysql]# ll
total 99360
-rw-r----- 1 mysql mysql      157 May 13 11:01  1.000028
-rw-r----- 1 mysql mysql       11 May 13 11:01  1.index
-rw-r----- 1 mysql mysql       56 May 13 11:09  auto.cnf
-rw-r----- 1 mysql mysql      474 May 13 11:01  backup-my.cnf
drwxr-x--- 2 mysql mysql       20 May 13 11:01  database
drwxr-x--- 2 mysql mysql      149 May 13 11:01  employees
-rw-r----- 1 mysql mysql     3539 May 13 11:01  ib_buffer_pool
-rw-r----- 1 mysql mysql 12582912 May 13 11:01  ibdata1
-rw-r----- 1 mysql mysql 10485760 May 13 11:01  ibdata2
-rw-r----- 1 mysql mysql 12582912 May 13 11:01  ibtmp1
drwxr-x--- 2 mysql mysql        6 May 13 11:01 '#innodb_redo'
drwxr-x--- 2 mysql mysql      143 May 13 11:01  mysql
-rw-r----- 1 mysql mysql 32505856 May 13 11:01  mysql.ibd
drwxr-x--- 2 mysql mysql     8192 May 13 11:01  performance_schema
drwxr-x--- 2 mysql mysql       28 May 13 11:01  sys
drwxr-x--- 2 mysql mysql       28 May 13 11:01  test1
-rw-r----- 1 mysql mysql 16777216 May 13 11:01  undo_001
-rw-r----- 1 mysql mysql 16777216 May 13 11:01  undo_002
[root@mysqlPerconaSTG mysql]# rm -rf ibdata1
[root@mysqlPerconaSTG mysql]# rm -rf ibdata2
[root@mysqlPerconaSTG mysql]# rm -rf ibtmp1
[root@mysqlPerconaSTG mysql]# rm -rf undo_001
[root@mysqlPerconaSTG mysql]# rm -rf undo_002
```

remove the double write buffer files

```
[root@mysqlPerconaSTG /]# ls
backup  bin  boot  dev  doublewrite  etc  events  home  lib  lib64  media  mnt  mysqldata  opt  proc  redologs  root  run  sbin  srv  sys  testb  tmp  usr  var  xtrabackup_checkpoints
[root@mysqlPerconaSTG /]# rm -rf doublewrite/*
[root@mysqlPerconaSTG /]#
```

4. Recheck the ownership of the data directory.

`sudo chown -R mysql:mysql /mysqldata/mysql`

`sudo chown -R mysql:mysql /mysqldata/innodb`

```
[root@mysqlPerconaSTG /]# sudo chown -R mysql:mysql /mysqldata/mysql
[root@mysqlPerconaSTG /]# sudo chown -R mysql:mysql /mysqldata/innodb
[root@mysqlPerconaSTG /]# ll mysqldata/
total 0
drwxr-xr-x 2 mysql mysql 104 May 13 11:05 innodb
drwxr-xr-x 9 mysql mysql 289 May 13 11:01 mysql
[root@mysqlPerconaSTG /]#
```

6. Start MySQL services.

```
sudo systemct   start mysqld
```

```
[root@mysqlPerconaSTG mysql]# sudo systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Mon 2024-05-13 11:11:21 +03; 7s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 133420 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 133454 (mysqld)
   Status: "Server is operational"
    Tasks: 49 (limit: 24168)
   Memory: 544.8M
   CGroup: /system.slice/mysqld.service
           └─133454 /usr/sbin/mysqld

May 13 11:11:15 mysqlPerconaSTG systemd[1]: Starting MySQL Server...
May 13 11:11:21 mysqlPerconaSTG systemd[1]: Started MySQL Server.
[root@mysqlPerconaSTG mysql]#
```