

## 🔧 How To Create a Multi-Node MySQL Cluster on Ubuntu 16.04



Posted June 17, 2016

👁 122.3k

MYSQL

CLUSTERING

UBUNTU

UBUNTU 16.04

By: Anatoliy Dimitrov

Not using **Ubuntu 16.04**? Choose a different version:

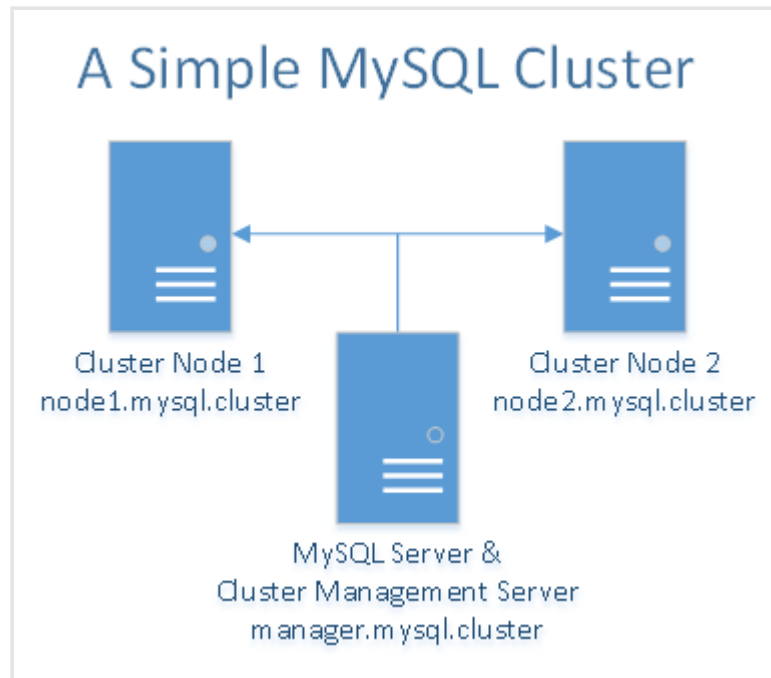
### Introduction

MySQL cluster is a software technology which provides high availability and throughput. If you are already familiar with other cluster technologies, you will find MySQL cluster similar to them. In short, there is one or more management nodes which control the data nodes (where data is stored). After consulting with the management node, clients (MySQL clients, servers, or native APIs) connect directly to the data nodes.

You may wonder how MySQL replication is related to MySQL cluster. With the cluster there is no typical replication of data, but instead there is synchronization of the data nodes. For this purpose a special data

engine must be used — NDBCluster (NDB). Think of the cluster as a single logical MySQL environment with redundant components. Thus, a MySQL cluster can participate in replication with other MySQL clusters.

MySQL cluster works best in a shared-nothing environment. Ideally, no two components should share the same hardware. For simplicity, and demonstration purposes, we'll limit ourselves to using only three Droplets. There will be two Droplets acting as data nodes which are syncing data between themselves. The third Droplet will be used for the cluster manager and at the same time for the MySQL server/client. If you have more Droplets, you can add more data nodes, separate the cluster manager from the MySQL server/client, and even add more Droplets as cluster managers and MySQL servers/clients.



## Prerequisites

You will need a total of three Droplets — one Droplet for the MySQL cluster manager and the MySQL server/client and two Droplets for the redundant MySQL data nodes.

In the **same DigitalOcean data center**, create the following Droplets with **private networking enabled**:

- Three Ubuntu 16.04 Droplets with a minimum of 1 GB RAM and private networking enabled
- Non-root user with sudo privileges for each Droplet (Initial Server Setup with Ubuntu 16.04 explains how to set this up.)

MySQL cluster stores a lot of information in RAM. Each Droplet should have at least 1GB of RAM.

As mentioned in the private networking tutorial, be sure to setup custom records for the 3 Droplets. For the sake of simplicity and convenience, we'll use the following custom records for each Droplet in the `/etc/hosts` file:

```
10.XXX.XX.X node1.mysql.cluster
10.YYY.YY.Y node2.mysql.cluster
10.ZZZ.ZZ.Z manager.mysql.cluster
```

Please replace the highlighted IPs with the private IPs of your Droplets correspondingly.

Except otherwise noted, all of the commands that require root privileges in this tutorial should be run as a non-root user with sudo privileges.

## Step 1 — Downloading and Installing MySQL Cluster

At the time of writing this tutorial, the latest GPL version of the MySQL cluster is 7.4.11. The product is built on top of MySQL 5.6 and it includes:

- Cluster manager software
- Data node manager software
- MySQL 5.6 server and client binaries

You can download the free, Generally Available (GA) MySQL cluster release from the [official MySQL cluster download page](#). From this page, choose the Debian Linux platform package, which is also suitable for Ubuntu. Also make sure to select the 32-bit or the 64-bit version depending on the architecture of your Droplets. Upload the installation package to each of your Droplets.

The installation instructions will be the same for all Droplets, so complete these steps on all 3 Droplets.

Before you start the installation, the `libaio1` package must be installed since it is a dependency:

```
$ sudo apt-get install libaio1
```

After that, install the MySQL cluster package:

```
$ sudo dpkg -i mysql-cluster-gpl-7.4.11-debian7-x86_64.deb
```

Now you can find the MySQL cluster installation in the directory `/opt/mysql/server-5.6/`. We'll be working especially with the bin directory (`/opt/mysql/server-5.6/bin/`) where all the binaries are.

The same installation steps should be performed on all three Droplets regardless of the fact that each will have different function — manager or data node.

Next, we will configure the MySQL cluster manager on each Droplet.

## Step 2 — Configuring and Starting the Cluster Manager

In this step we'll configure the MySQL cluster manager (`manager.mysql.cluster`). Its proper configuration will ensure correct synchronization and load distribution among the data nodes. All commands should be executed on Droplet `manager.mysql.cluster`.

The cluster manager is the first component which has to be started in any cluster. It needs a configuration file which is passed as an argument to its binary file. For convenience, we'll use the file `/var/lib/mysql-cluster/config.ini` for its configuration.

On the `manager.mysql.cluster` Droplet, first create the directory where this file will reside (`/var/lib/mysql-cluster`):

```
$ sudo mkdir /var/lib/mysql-cluster
```

Then create a file and start editing it with nano:

```
$ sudo nano /var/lib/mysql-cluster/config.ini
```

This file should contain the following code:

`/var/lib/mysql-cluster/config.ini`

```
[ndb_mgmd]
# Management process options:
hostname=manager.mysql.cluster # Hostname of the manager
datadir=/var/lib/mysql-cluster # Directory for the log files

[ndbd]
hostname=node1.mysql.cluster   # Hostname of the first data node
datadir=/usr/local/mysql/data   # Remote directory for the data files

[ndbd]
hostname=node2.mysql.cluster   # Hostname of the second data node
datadir=/usr/local/mysql/data   # Remote directory for the data files

[mysqld]
# SQL node options:
hostname=manager.mysql.cluster # In our case the MySQL server/client is on the same Droplet as the c
```

For each of the above components we have defined a `hostname` parameter. This is an important security measure because only the specified hostname will be allowed to connect to the manager and participate in the cluster as per their designated role.

Furthermore, the `hostname` parameters specify on which interface the service will run. This matters, and is important for security, because in our case the above hostnames point to private IPs which we have specified in the `/etc/hosts` files. Thus, you cannot access any of the above services from outside of the private network.

In the above file you can add more redundant components such as data nodes (`ndbd`) or MySQL servers (`mysqld`) by just defining additional instances in the exactly the same manner.

Now you can start the manager for the first time by executing the `ndb_mgmd` binary and specifying the config file with the `-f` argument like this:

```
$ sudo /opt/mysql/server-5.6/bin/ndb_mgmd -f /var/lib/mysql-cluster/config.ini
```

You should see a message about successful start similar to this:

Output of `ndb_mgmd`

```
MySQL Cluster Management Server mysql-5.6.29 ndb-7.4.11
```

You would probably like to have the management service started automatically with the server. The GA cluster release doesn't come with a suitable startup script, but there are a few available online. For the beginning you can just add the start command to the `/etc/rc.local` file and the service will be automatically started during boot. First, though, you will have to make sure that `/etc/rc.local` is executed during the server startup. In Ubuntu 16.04 this requires running an additional command:

```
$ sudo systemctl enable rc-local.service
```

Then open the file `/etc/rc.local` for editing:

```
$ sudo nano /etc/rc.local
```

There add the start command before the `exit` line like this:

```
#!/etc/rc.local

...
/opt/mysql/server-5.6/bin/ndb_mgmd -f /var/lib/mysql-cluster/config.ini
exit 0
```

Save and exit the file.

The cluster manager does not have to run all the time. It can be started, stopped, and restarted without downtime for the cluster. It is required only during the initial startup of the cluster nodes and the MySQL server/client.

## Step 3 — Configuring and Starting the Data Nodes

Next we'll configure the data nodes (`node1.mysql.cluster` and `node2.mysql.cluster`) to store the data files and support properly the NDB engine. All commands should be executed on both nodes. You can start first with `node1.mysql.cluster` and then repeat exactly the same steps on `node2.mysql.cluster`.

The data nodes read the configuration from the standard MySQL configuration file `/etc/my.cnf` and more specifically the part after the line `[mysql_cluster]`. Create this file with nano and start editing it:

```
$ sudo nano /etc/my.cnf
```

Specify the hostname of the manager like this:

```
                                /etc/my.cnf

[mysql_cluster]
ndb-connectstring=manager.mysql.cluster
```

Save and exit the file.

Specifying the location of the manager is the only configuration needed for the node engine to start. The rest of the configuration will be taken from manager directly. In our example the data node will find out that its data directory is `/usr/local/mysql/data` as per the manager's configuration. This directory has to be created on the node. You can do it with the command:

```
$ sudo mkdir -p /usr/local/mysql/data
```

After that you can start the data node for the first time with the command:

```
$ sudo /opt/mysql/server-5.6/bin/ndbd
```

After a successful start you should see a similar output:

Output of `ndbd`

```
2016-05-11 16:12:23 [ndbd] INFO      -- Angel connected to 'manager.mysql.cluster:1186'
2016-05-11 16:12:23 [ndbd] INFO      -- Angel allocated nodeid: 2
```

You should have the `ndbd` service started automatically with the server. The GA cluster release doesn't come with a suitable startup script for this either. Just as we did for the cluster manager, let's add the startup command to the `/etc/rc.local` file. Again, you will have to make sure that `/etc/rc.local` is executed during the server startup with the command:

```
$ sudo systemctl enable rc-local.service
```

Then open the file `/etc/rc.local` for editing:

```
$ sudo nano /etc/rc.local
```

Add the start command before the `exit` line like this:

```
/etc/rc.local  
  
...  
/opt/mysql/server-5.6/bin/ndbd  
exit 0
```

Save and exit the file.

Once you are finished with the first node, repeat exactly the same steps on the other node , which is `node2.mysql.cluster` in our example.

## Step 4 — Configuring and Starting the MySQL Server and Client

A standard MySQL server, such as the one that is available in Ubuntu's default apt repository, does not support the MySQL cluster engine NDB. That's why you need a custom MySQL server installation. The cluster package which we already installed on the three Droplets comes with a MySQL server and a client too. As already mentioned, we'll use the MySQL server and client on the management node (`manager.mysql.cluster`).

The configuration is stored again the default `/etc/my.cnf` file. On `manager.mysql.cluster` , open the configuration file:

```
$ sudo nano /etc/my.cnf
```

Then add the following to it:

```
/etc/my.cnf  
  
[mysqld]  
ndbcluster # run NDB storage engine  
...
```

Save and exit the file.

As per the best practices, the MySQL server should run under its own user (`mysql`) which belongs to its own group (again `mysql`). So let's create first the group:

```
$ sudo groupadd mysql
```

Then create the `mysql` user belonging to this group and make sure it cannot use shell by setting its shell path to `/bin/false` like this:

```
$ sudo useradd -r -g mysql -s /bin/false mysql
```

The last requirement for the custom MySQL server installation is to create the default database. You can do it with the command:

```
$ sudo /opt/mysql/server-5.6/scripts/mysql_install_db --user=mysql
```

For starting the MySQL server we'll use the startup script from `/opt/mysql/server-5.6/support-files/mysql.server`. Copy it to the default init scripts directory under the name `mysqld` like this:

```
$ sudo cp /opt/mysql/server-5.6/support-files/mysql.server /etc/init.d/mysqld
```

Enable the startup script and add it to the default runlevels with the command:

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

Now we can start the MySQL server for the first time manually with the command:

```
$ sudo systemctl start mysqld
```

As a MySQL client we'll use again the custom binary which comes with the cluster installation. It has the following path: `/opt/mysql/server-5.6/bin/mysql`. For convenience let's create a symbolic link to it in the default `/usr/bin` path:

```
$ sudo ln -s /opt/mysql/server-5.6/bin/mysql /usr/bin/
```

Now you can start the client from the command line by simply typing `mysql` like this:

```
$ mysql
```

You should see an output similar to:

Output of `ndb_mgmd`

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
```

```
Your MySQL connection id is 3
```

```
Server version: 5.6.29-ndb-7.4.11-cluster-gpl MySQL Cluster Community Server (GPL)
```

To exit the MySQL prompt, simply type `quit` or press simultaneously `CTRL-D`.

The above is the first check to show that the MySQL cluster, server, and client are working. Next we'll go through more detailed tests to confirm the cluster is working properly.



# Testing the Cluster

At this point our simple MySQL cluster with one client, one server, one manager, and two data nodes should be complete. From the cluster manager Droplet (`manager.mysql.cluster`) open the management console with the command:

```
$ sudo /opt/mysql/server-5.6/bin/ndb_mgm
```

Now the prompt should change to the cluster management console. It looks like this:

```
Inside the ndb_mgm console
-- NDB Cluster -- Management Client --
ndb_mgm>
```

Once inside the console execute the command `SHOW` like this:

```
ndb_mgm> SHOW
```

You should see output similar to this one:

```
Output of ndb_mgm
Connected to Management Server at: manager.mysql.cluster:1186
Cluster Configuration
-----
[ndbd(NDB)]      2 node(s)
id=2      @10.135.27.42  (mysql-5.6.29 ndb-7.4.11, Nodegroup: 0, *)
id=3      @10.135.27.43  (mysql-5.6.29 ndb-7.4.11, Nodegroup: 0)

[ndb_mgmd(MGM)]  1 node(s)
id=1      @10.135.27.51  (mysql-5.6.29 ndb-7.4.11)

[mysqld(API)]    1 node(s)
id=4      @10.135.27.51  (mysql-5.6.29 ndb-7.4.11)
```

The above shows that there are two data nodes with ids 2 and 3. They are active and connected. There is also one management node with id 1 and one MySQL server with id 4. You can find more information about each id by typing its number with the command `STATUS` like this:

```
ndb_mgm> 2 STATUS
```

The above command would show you the status of node 2 along with its MySQL and NDB versions:

```
Output of ndb_mgm
```

Node 2: started (mysql-5.6.29 ndb-7.4.11)

To exit the management console type `quit`.

The management console is very powerful and gives you many other options for managing the cluster and its data, including creating an online backup. For more information check the [official documentation](#).

Let's have a test with the MySQL client now. From the same Droplet, start the client with the `mysql` command for the MySQL root user. Please recall that we have created a symlink to it earlier.

```
$ mysql -u root
```

Your console will change to the MySQL client console. Once inside the MySQL client, run the command:

```
mysql> SHOW ENGINE NDB STATUS \G
```

Now you should see all the information about the NDB cluster engine starting with the connection details:

Output of `mysql`

```
***** 1. row *****
  Type: ndbcluster
  Name: connection
Status: cluster_node_id=4, connected_host=manager.mysql.cluster, connected_port=1186, number_of_data_
...
```

The most important information from above is the number of ready nodes — 2. This redundancy will allow your MySQL cluster to continue operating even if one of the data nodes fails while. At the same time your SQL queries will be load balanced to the two nodes.

You can try shutting down one of the data nodes in order to test the cluster stability. The simplest thing would be just to restart the whole Droplet in order to have a full test of the recovery process. You will see the value of `number_of_ready_data_nodes` change to 1 and back to 2 again as the node is restarted.

## Working with the NDB Engine

To see how the cluster really works, let's create a new table with the NDB engine and insert some data into it. Please note that in order to use the cluster functionality, the engine must be NDB. If you use InnoDB (default) or any other engine other than NDB, you will not make use of the cluster.

First, let's create a database called `cluster` with the command:

```
mysql> CREATE DATABASE cluster;
```

Next, switch to the new database:

```
mysql> USE cluster;
```

Now, create a simple table called `cluster_test` like this:

```
mysql> CREATE TABLE cluster_test (name VARCHAR(20), value VARCHAR(20)) ENGINE=ndbcluster;
```

We have explicitly specified above the engine `ndbcluster` in order to make use of the cluster. Next, we can start inserting data with a query like this:

```
mysql> INSERT INTO cluster_test (name,value) VALUES('some_name','some_value');
```

To verify the data has been inserted, run a select query like this:

```
mysql> SELECT * FROM cluster_test;
```

When you are inserting and selecting data like this, you are load-balancing your queries between all the available data node, which are two in our example. With this scaling out you benefit both in terms of stability and performance.

## Conclusion

As we have seen in this article, setting up a MySQL cluster can be simple and easy. Of course, there are many more advanced options and features which are worth mastering before bringing the cluster to your production environment. As always, make sure to have an adequate testing process because some problems could be very hard to solve later. For more information and further reading please go to the [official documentation for MySQL cluster](#).

By: Anatoliy Dimitrov

♡ Upvote (25)

📄 Subscribe

🔗 Share



Editor:  
Tammy Fox



We just made it easier for you to deploy faster.

[TRY FREE](#)

Related Tutorials

[How To Set Up a Remote Database to Optimize Site Performance with MySQL](#)

[How To Migrate a MySQL Database Between Two Servers](#)

[How To Set Up Master Slave Replication in MySQL](#)

[How To Import and Export Databases and Reset a Root Password in MySQL](#)


[How To Create a New User and Grant Permissions in MySQL](#)

28 Comments

Leave a comment...

[Log In to Comment](#)

 [TinoTokic](#) June 19, 2016

 0 One of the most useful tuts yet! Everyone needs a good hard to kill DB. Thanks!

^ [sdpagent](#) June 20, 2016



- 2 It is worth mentioning the limitations of using a cluster compared to a single node. This is primarily the lack of distributed table locking. Developers may think that their applications never make use of table locks and have managed to use transactions for everything, but table locking is a key part of ALTER TABLE queries when running an application's database migrations. Also, some queries, such as TRUNCATE run an alter table query implicitly without developers perhaps realizing. The MySQL docs for the limitations of running multiple MySQL cluster nodes can be found here: <https://dev.mysql.com/doc/refman/5.7/en/mysql-cluster-limitations-multiple-nodes.html>

---

^ [jmilam104](#) June 22, 2016



- 0 Can NDB do Full text searching yet, all the info I see on it is old? Would like to experiment with this but not sure, any other recommendations for getting around it if not?

---

^ [Toli](#) June 25, 2016

- 0 Hi, I haven't tried it but according to the documentation for MySQL 5.6, on which the latest cluster software is based, full text searching is not supported for NDB. It works only with InnoDB or MyISAM:

<http://dev.mysql.com/doc/refman/5.6/en/fulltext-search.html>

---

^ [julien5304fdae3](#) July 4, 2016



- 0 Hello, i have got 2 server running mysql databse (2 differents database) im not sure when the connect them to the cluster. I have to connect them on the manager server after settin gup the 2 node server & the manager one?

---

^ [selmanguney](#) August 19, 2016



- 0 Great tutorial,  
In a disaster scenario, does a single cluster node work as single regular DB server? if all other nodes and MySQL cluster manager is not accessible or down.  
Thanks

---

^ [Toli](#) August 20, 2016



- 0 Hi,

In most cases in a typical disaster scenario - yes, it can be regarded as one DB environment. For DR you'd wish to have the environments also geographically separated so the usual replication would work better.

---

^ [Pinche](#) August 25, 2016



- 0 Hi  
I have 2 Problems and maybe somebody here can help me with them.  
1 My notes are stuck in "starting" on the manager but on the notes they say they are connected.

- No firewall
- Configs are ok
- Logs just show that they try to connect over and over again

2 Also for some reason my mysqld(API) is not getting connected either


- I have the Part in the config.ini
- and mysql > SHOW ENGINE NDB STATUS \G is giving me a proper result

I hope somebody can help me

with best regards  
Pinche


---

^ [stirol](#) September 7, 2016

0  Heh... What people tend to do, instead of simply using Galera Replication. ..


---

^ [audrino](#) September 21, 2016

0  i can't fine /opt/mysql/server-5.7/scripts/mysqlinstalldb .  
can some one show me how the mysqlinstalldb

---

^ [mahesh1010logic](#) September 22, 2016

0  Great tutorial. I'm configuring Master - Slave MySQL Cluster replication between two MySQL Clusters .  
Someone brief me about what configuration changes required and where it is need to do. Thanks in advance.

---

^ [Karam](#) November 3, 2016

0  \*\*\*\*\***Question Regarding MySQL Cluster Ports**\*\*\*\*\*

Hello,

I am on Linux platform with MySQL NDB 5.7. I am trying to monitor all traffic related to MySQL clustering - between data nodes, management node and sql nodes. To that end, I used netstat to list all open ports listening on my machine before starting MySQL cluster. Then, I started MySQL cluster and ran netstat again. I assumed that the ports that were listening the second time around, but not the first time, were related to MySQL clustering.

But there are two problems with this. First, there could be ports opened by other processes between the two netstat runs. Second, MySQL might open other ports after I ran the netstat command the second time.

**\*\*What is the best way to go about finding all ports being used by MySQL for clustering purposes? \*\***

I believe ephemeral ports are picked dynamically, so perhaps if I knew all the MySQL clustering related processes that would be running, I can figure out every port that they are using. Pointers will be very welcome.

---

^ [rspadar](#) November 8, 2016

0 I try to configure but on STEP4 sudo systemctl start mysqld have error  
MySql not start.  
I follow all STEP correctly  
Can you help me?

---

^ [sheikh303](#) October 5, 2018

0 I am having the same issue. But Fixed it.  
the actual issue was this error:  
2018-10-05 11:00:30 0 [ERROR] Failed to access directory for --secure-file-priv. Please make sure that  
directory exists and is accessible by MySQL Server. Supplied value : /var/lib/mysql-files  
  
--secure-file-priv needed to be disabled or set a accisible directory to start mysql. I disabled it my adding  
the following line in /etc/my.cnf  
  
[mysqld]  
secure-file-priv = ""

---

^ [stevenquan](#) December 2, 2016

0 Hi ,  
  
thanks for the tutorial I have followed this to a point where trying to start mysql server on the management  
node it is not starting with the following error  
  
2016-12-02T16:36:29.025637Z 0 [Warning] TIMESTAMP with implicit DEFAULT value is deprecated. Please use  
--explicitdefaultsfor\_timestamp server option (see documentation for more details).  
2016-12-02T16:36:29.025800Z 0 [ERROR] Failed to access directory for --secure-file-priv. Please make sure  
that directory exists and is accessible by MySQL Server. Supplied value : /var/lib/mysql-files  
2016-12-02T16:36:29.025810Z 0 [ERROR] Aborting  
2016-12-02T16:36:29.025833Z 0 [Note] Binlog end  
  
please help

some more details:

- ubuntu 14.04
- mysql-cluster-gpl-7.5.4-debian8-x86\_64.deb
- 3 node cluster (2 data / 1 management)
- /var/lib/mysql-files does not exist
- I have ran "/opt/mysql/server-5.7/bin/mysqlinstalldb --user=mysql --datadir=/data/mysql"

/etc/my.cnf on management node

```
[mysqld]
ndbcluster # run NDB storage engine
datadir=/data/mysql
```

Regards  
Steven

---

^ [stevenquan](#) December 2, 2016

o looks like I managed to resolve my issue with mysql server

created a directory in /var/lib/ called "mysql-files" and changed the ownership and group of "mysql-files" to mysql

so you should have a directory like the following: /var/lib/mysql-files

thanks again for the great Tutorial

---

^ [sheikh303](#) October 5, 2018

o add the following line in my.cnf

```
[mysqld]
secure-file-priv = ""
```

---

^ [john306e2514c683cc52811851](#) December 23, 2016

o

```
$ sudo /opt/mysql/server-5.7/scripts/mysql_install_db --user=mysql
```

cant seem to find this in mysql 5.7

---

^ [fernandomm8e564](#) December 26, 2016

o Stay in

```
$sudo /opt/mysql/server-5.7/bin/mysql_install_db --user=mysql
```

---

^ [dan424152](#) February 28, 2017

o Ive followed the tutorial step by step (a couple of times with no luck) but I am geting this error: **Job for mysqld.service failed because the control process exited with error code. See "systemctl status mysql.service" and "journalctl -xe" for details.** when I run **systemctl start mysqld**

then the results of journalctl -xe are:

```
Feb 28 14:51:31 manager sudo[4592]:      root : TTY=pts/1 ; PWD=/root ; USER=root ; COMMAND=/
Feb 28 14:51:31 manager sudo[4592]: pam_unix(sudo:session): session opened for user root by
Feb 28 14:51:31 manager systemd[1]: Starting LSB: start and stop MySQL...
-- Subject: Unit mysqld.service has begun start-up
-- Defined-By: systemd
-- Support: http://lists.freedesktop.org/mailman/listinfo/systemd-devel
```



```
--
-- Unit mysqld.service has begun starting up.
Feb 28 14:51:31 manager mysqld[4595]: Starting MySQL
Feb 28 14:51:31 manager mysqld[4595]: .Logging to '/opt/mysql/server-5.6/data/manager.err'.
Feb 28 14:51:32 manager mysqld[4595]: * The server quit without updating PID file (/opt/mys
Feb 28 14:51:32 manager systemd[1]: mysqld.service: Control process exited, code=exited stat
Feb 28 14:51:32 manager sudo[4592]: pam_unix(sudo:session): session closed for user root
Feb 28 14:51:32 manager systemd[1]: Failed to start LSB: start and stop MySQL.
-- Subject: Unit mysqld.service has failed
-- Defined-By: systemd
-- Support: http://lists.freedesktop.org/mailman/listinfo/systemd-devel
--
-- Unit mysqld.service has failed.
--
-- The result is failed.
Feb 28 14:51:32 manager systemd[1]: mysqld.service: Unit entered failed state.
Feb 28 14:51:32 manager systemd[1]: mysqld.service: Failed with result 'exit-code'.
```

There are no other error logs or anything anywhere, this is as much info as I can get out of it.

---

^ [jonasmarquez](#) March 18, 2017

0 Hi Anatoliy,

I have some problems to take a perfect installation follow this tutorial, with step 4 with the `mysqlinstalldb` command because is deprecated, I attach a log code in pastebin with the error to you can take a look!

Thanks in advance!

Pastebin: <http://pastebin.com/OnXdS7gv>

---

^ [rsrinath](#) May 11, 2017

0 Hi,

Now you can start the client from the command line by simply typing mysql like this:

mysql

As i followed the same in my server, after entered mysql, showing error as  
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)

Please help me to get out from this.

---

^ [fehimb](#) November 19, 2017

1 Hi @all,

I was following this tutorial and when it came to the point where I had to start the MySQL I got this Error Message:

```
Nov 19 18:42:22 MySQL-Server-01 mysqld[25546]: Starting MySQL
Nov 19 18:42:22 MySQL-Server-01 mysqld[25546]: .Logging to '/opt/mysql/server-5.6/data/MySQL
Nov 19 18:42:23 MySQL-Server-01 mysqld[25546]: * The server quit without updating PID file
Nov 19 18:42:23 MySQL-Server-01 systemd[1]: mysqld.service: Control process exited, code=exi
Nov 19 18:42:23 MySQL-Server-01 systemd[1]: Failed to start LSB: start and stop MySQL.
-- Subject: Unit mysqld.service has failed
-- Defined-By: systemd
-- Support: http://lists.freedesktop.org/mailman/listinfo/systemd-devel
--
-- Unit mysqld.service has failed.
--
-- The result is failed.
Nov 19 18:42:23 MySQL-Server-01 systemd[1]: mysqld.service: Unit entered failed state.
Nov 19 18:42:23 MySQL-Server-01 systemd[1]: mysqld.service: Failed with result 'exit-code'
```



Any one can help or has the same error?

bg

---

^ [Porthorian](#) March 22, 2018

0 Did you ever get this working? Encountering same problem.

---

^ [ccor1412](#) May 11, 2018

0 Hi,  
When I need to run `ndbmcmd` command the next error showed "`ndbmcmd` command not found", Does somebody why this is happened ?

---

^ [ccor1412](#) May 12, 2018

0 How Can I install and connect with phpmyadmin to my mysql database in this tutorial?

---

^ [juanmendieta21](#) May 16, 2018

0 Hi, I have changed the config.ini and the MGM seems not to read it. I have deleted all the information on config.ini and the MGM doesn't read it neither. Do I have to delete some cache or something? Sorry for my english.

[Load More Comments](#)



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.



Copyright © 2019 DigitalOcean™ Inc.

[Community](#) [Tutorials](#) [Questions](#) [Projects](#) [Tags](#) [Newsletter](#) [RSS](#) 

---

[Distros & One-Click Apps](#) [Terms, Privacy, & Copyright](#) [Security](#) [Report a Bug](#) [Write for DOnations](#) [Shop](#)