**MySQL:**

1. **Install mysql**

sudo apt-get update

sudo apt-get install mysql-server

mysql\_secure\_installation

systemctl status mysql.service

1. **setup root password**

Stop the MySQL Server: sudo /etc/init.d/mysql stop

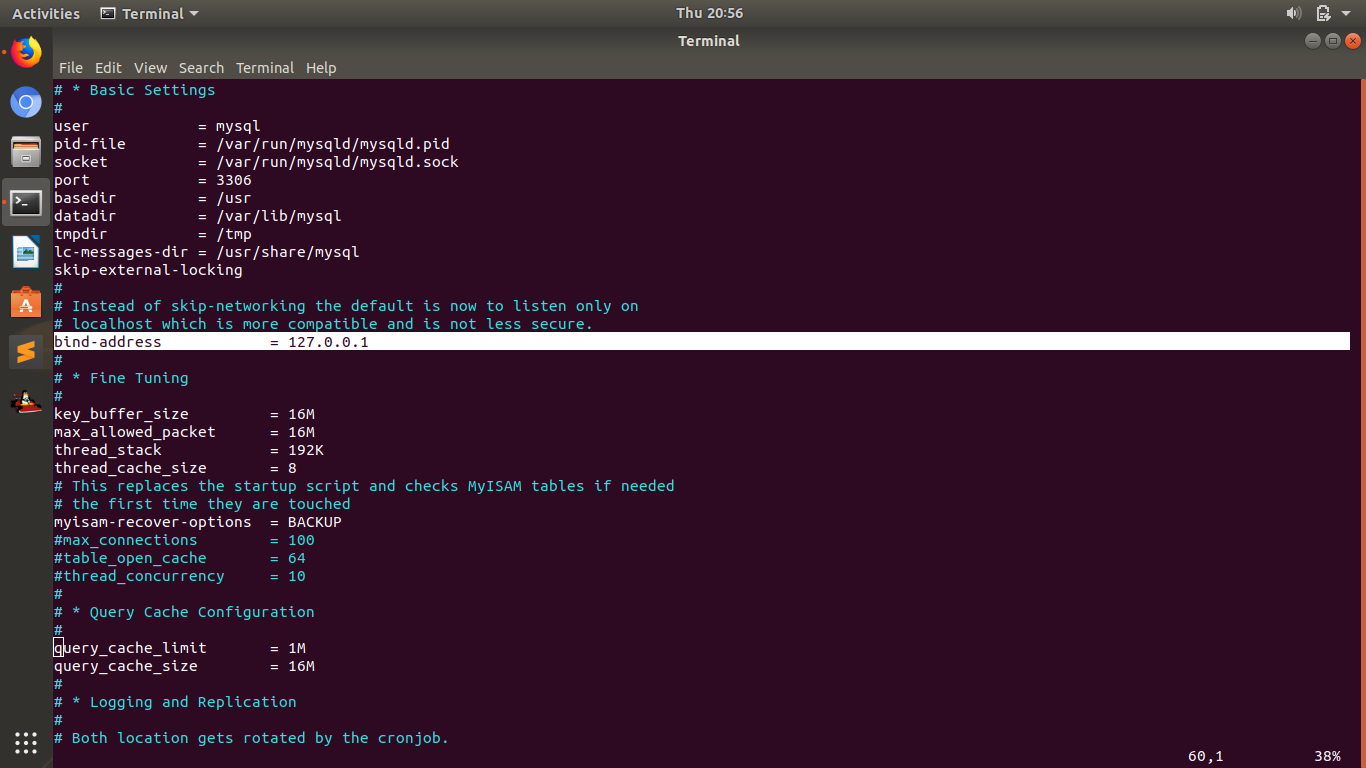
Start the mysqld configuration: sudo mysqld --skip-grant-tables &

Login to MySQL as root: mysql -u root mysql

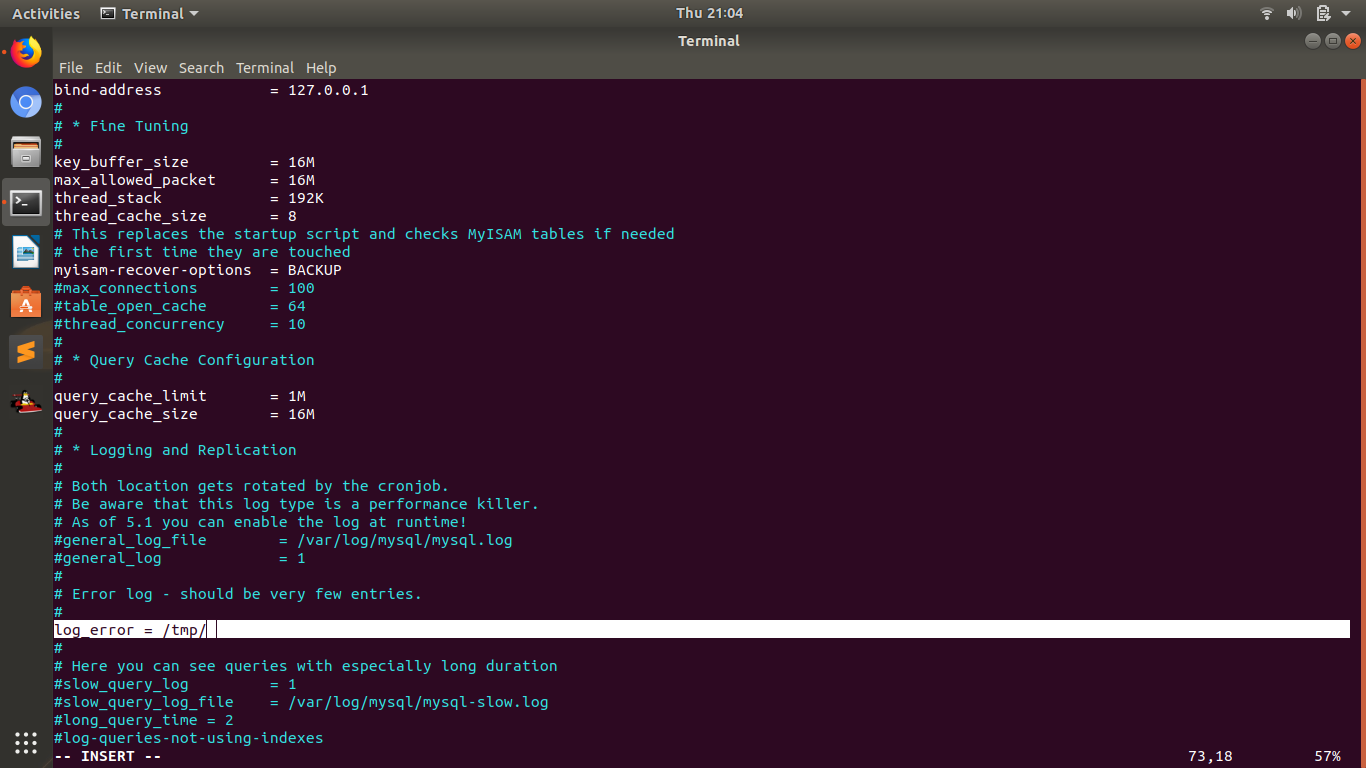
Replace YOURNEWPASSWORD with your new password:

UPDATE  
 mysql.user  
SET  
 Password = PASSWORD('YOURNEWPASSWORD')  
WHERE  
 User = 'root';  
FLUSH PRIVILEGES;  
exit;

1. **set the bind address to 127.0.0.1**



1. **change the log file location to /tmp/**



1. **reset the root password without using your current root password**

$ service mysql stop

$ mysqld\_safe --skip-grant-tables &

Log into MySQL using the following command:

$ mysql

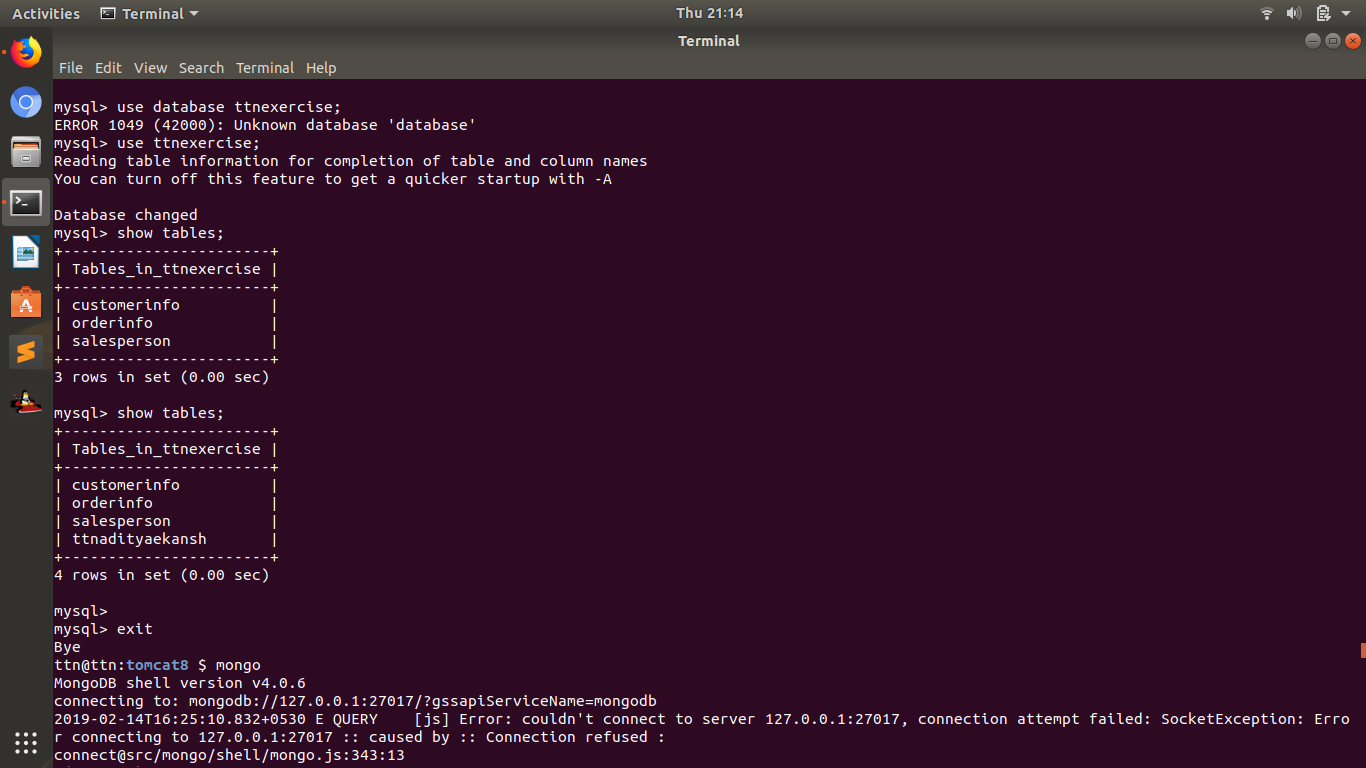
> UPDATE mysql.user SET Password=PASSWORD('NEW-PASSWORD') WHERE User='root';

> FLUSH PRIVILEGES;  
> exit;

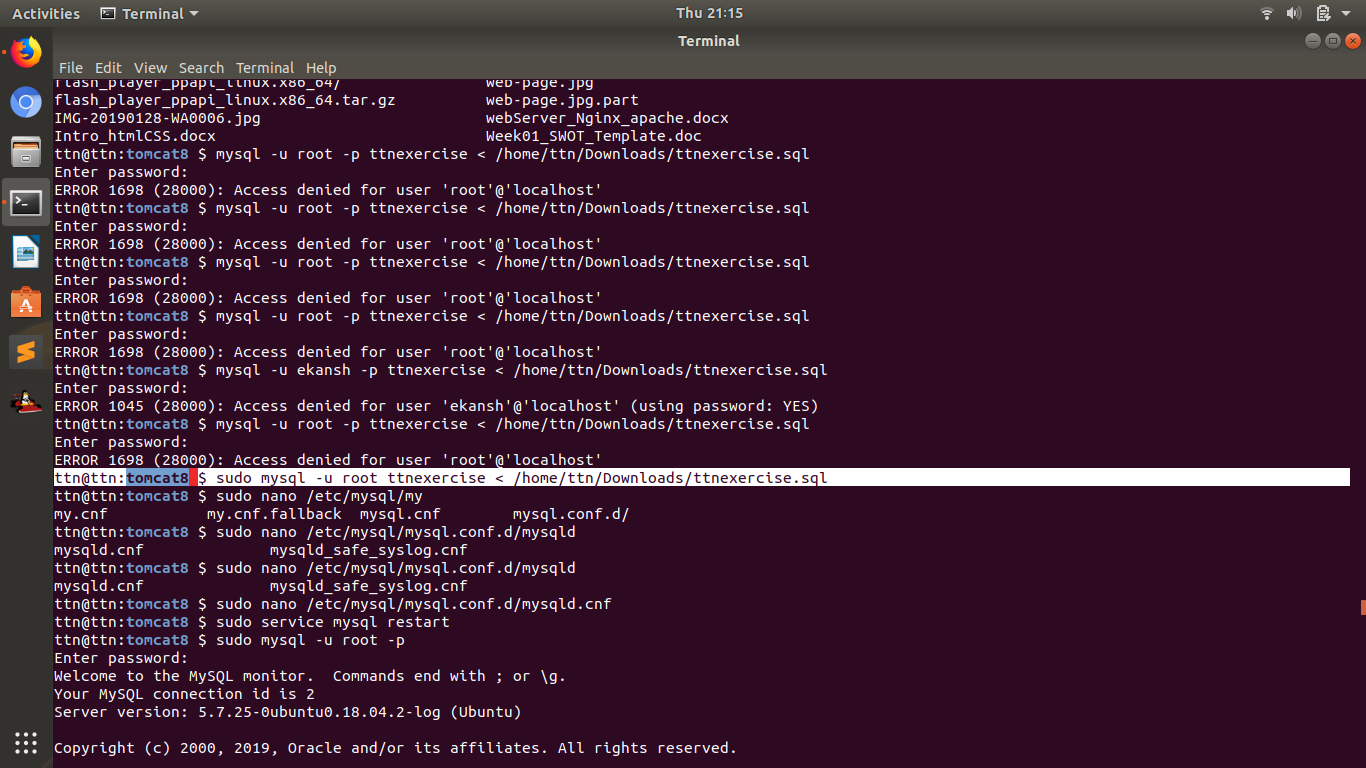
$ mysqladmin -u root -p shutdown

$ service mysql start

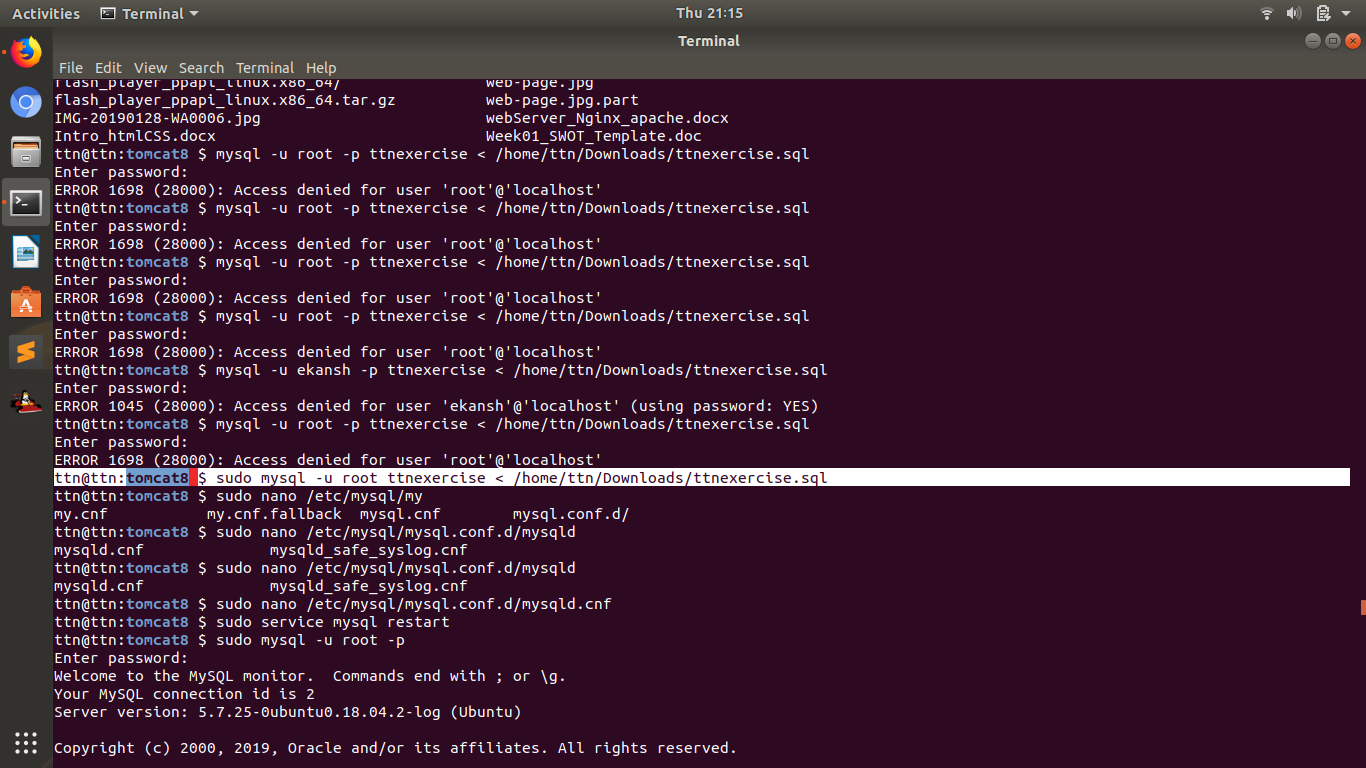
1. **create a dummy database with some tables**



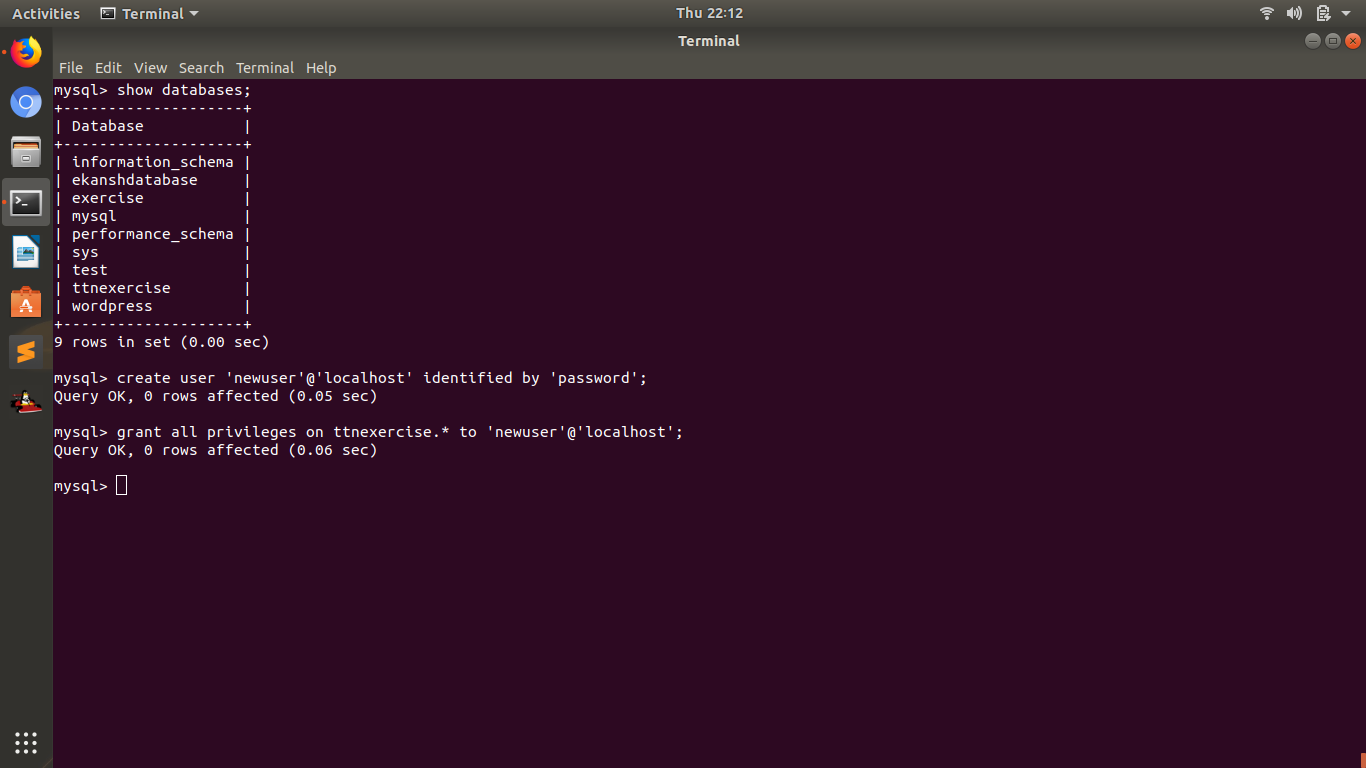
1. **take the dump of the database created above**



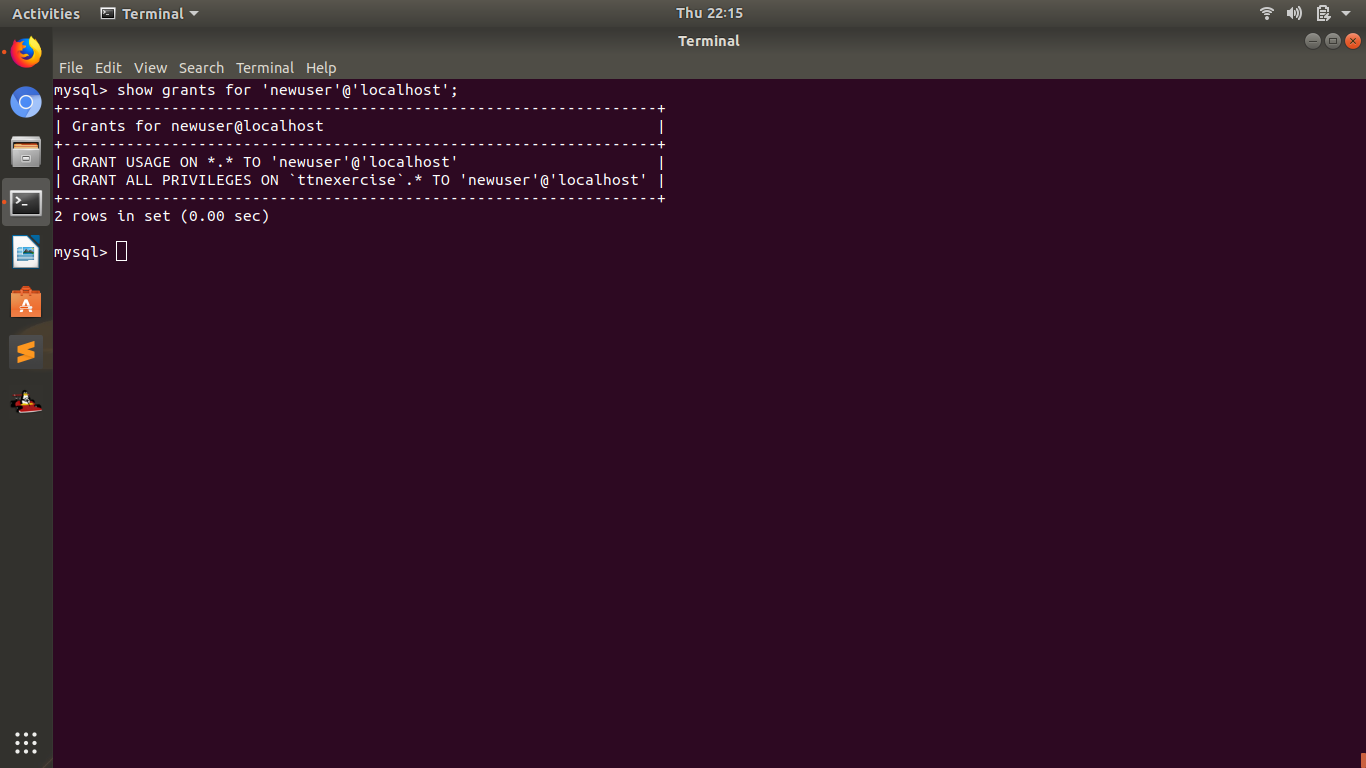
1. **restore the backedup database to another database**



1. **create a new user and grant access only to the new database**



1. **show grants/privileges of the newly created user**



1. **Setup master-slave for MySql.**

sudo apt-get install mysql-server mysql-client

## **Step One—Configure the Master**

sudo vim /etc/mysql/mysql.conf.d/mysqld.cnf

Replace the bind address with your private ip

bind-address = 10.1.254.24

Make sure this line is uncommented.

server-id = 1

log\_bin = /var/log/mysql/mysql-bin.log

binlog\_do\_db = ttnexercise

Refresh MySQL.

sudo service mysql restart

Open up the MySQL shell.

mysql -u root -p

We need to grant privileges to the slave. You can use this line to name your slave and set up their password. The command should be in this format:

GRANT REPLICATION SLAVE ON \*.\* TO 'ekansh'@'%' IDENTIFIED BY 'Ekansh@1234';

Follow up with:

FLUSH PRIVILEGES;

## **Step Two—Configure the Slave Database**

Log into your slave server, open up the MySQL shell and create the new database that you will be replicating from the master (then exit):

CREATE DATABASE ttnexercise;

EXIT;

Import the database that you previously exported from the master database.

mysql -u root -p ttnexercise < /path/to/exercise.sql

Now we need to configure the slave configuration in the same way as we did the master:

sudo vim /etc/mysql/mysql.conf.d/mysqld.cnf

server-id = 2

Following that, make sure that your have the following three criteria appropriately filled out:

relay-log = /var/log/mysql/mysql-relay-bin.log

log\_bin = /var/log/mysql/mysql-bin.log

binlog\_do\_db = ttnexercise

sudo service mysql restart

CHANGE MASTER TO MASTER\_HOST='10.1.254.24',MASTER\_USER='ekansh', MASTER\_PASSWORD='Ekansh@1234', MASTER\_LOG\_FILE='mysql-bin.000077', MASTER\_LOG\_POS= XXX;

Activate the slave server:

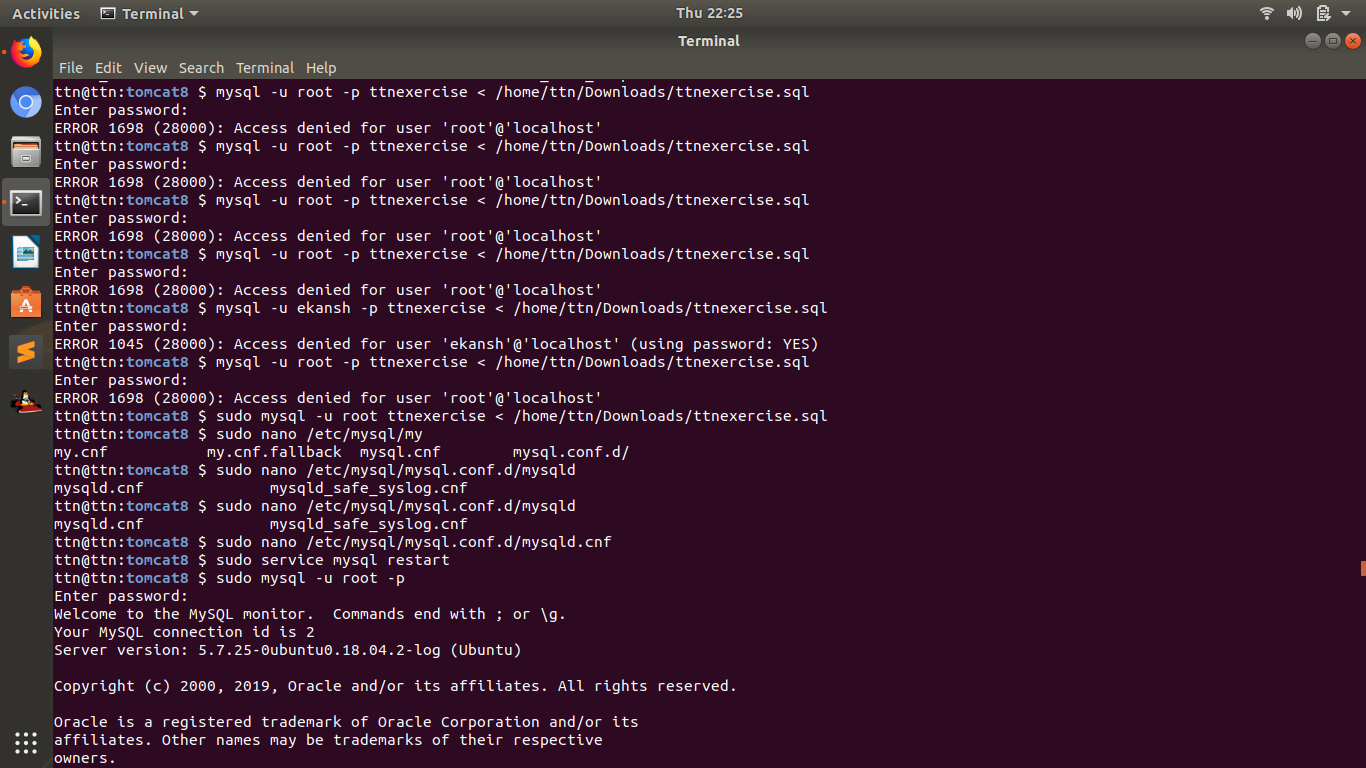
START SLAVE;

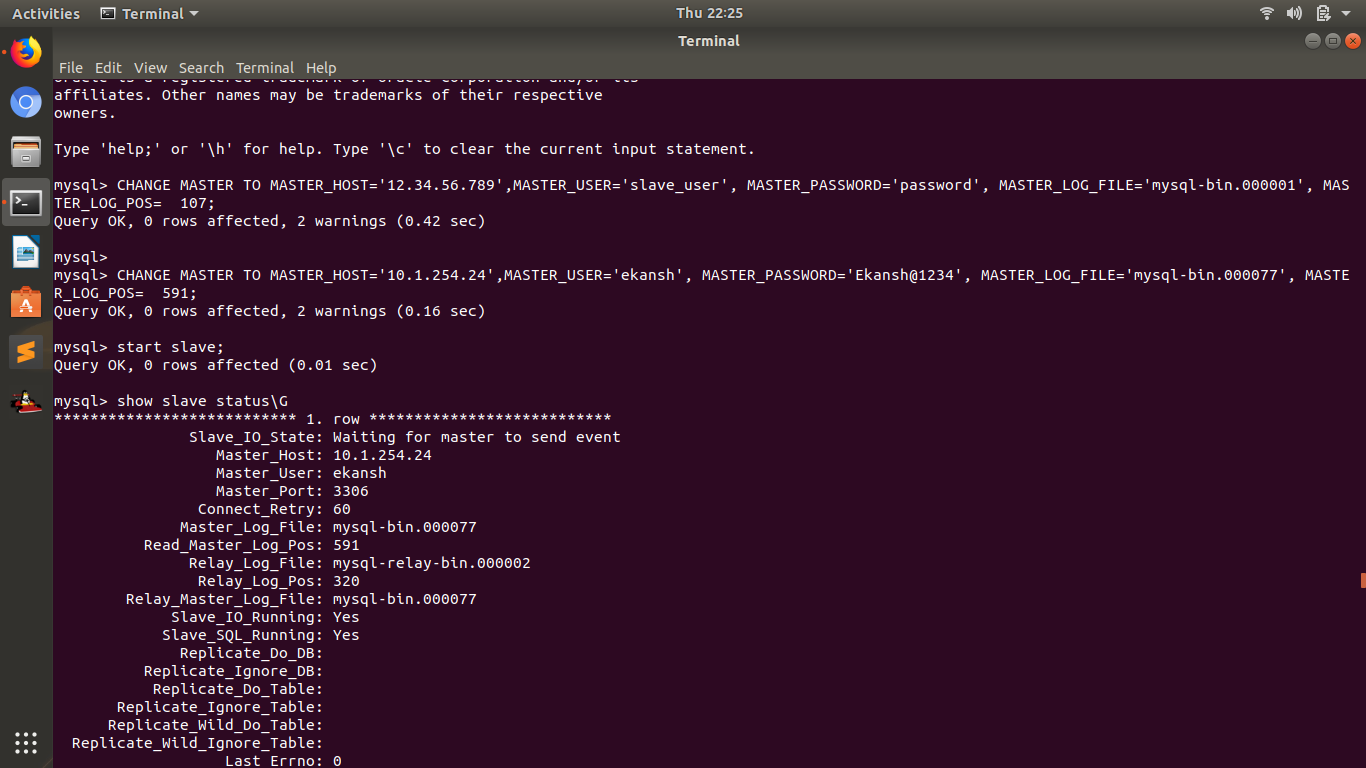
You be able to see the details of the slave replication by typing in this command. The \G rearranges the text to make it more readable.

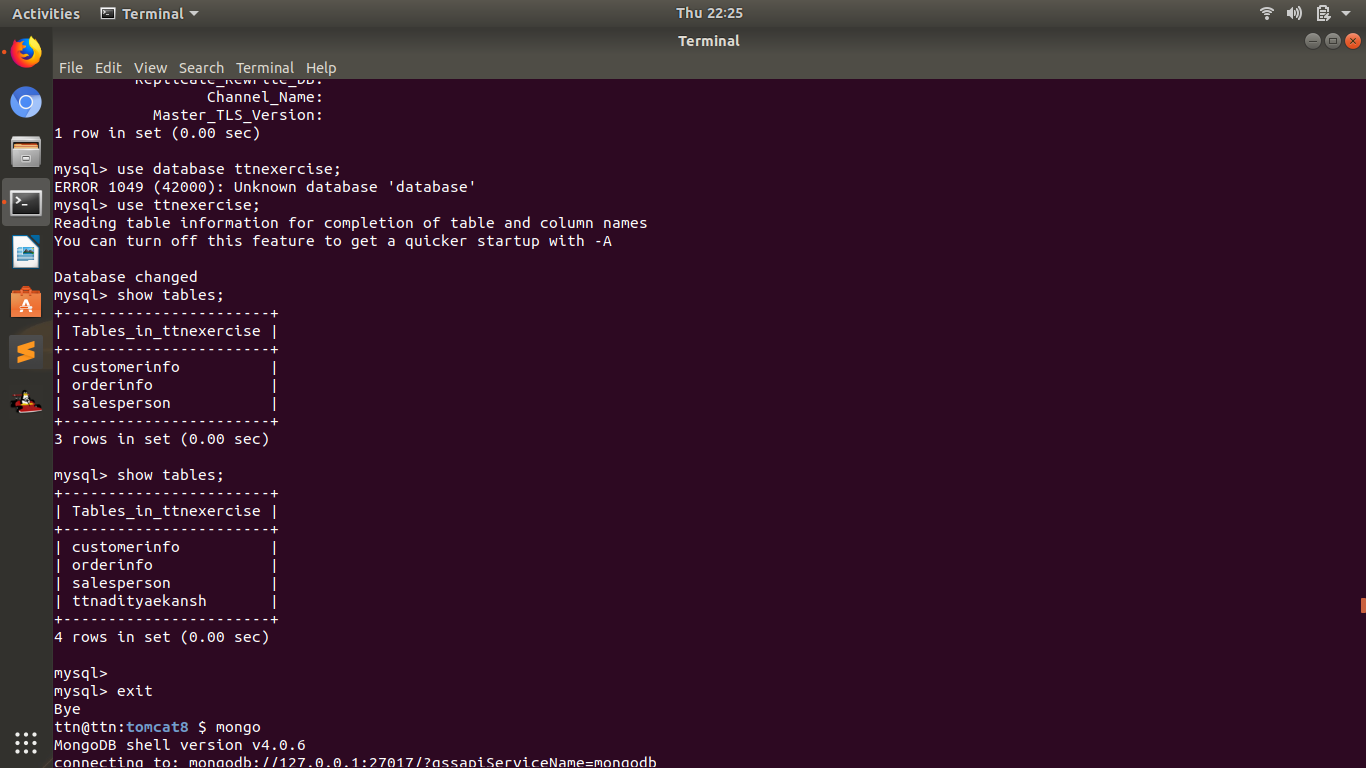
SHOW SLAVE STATUS\G

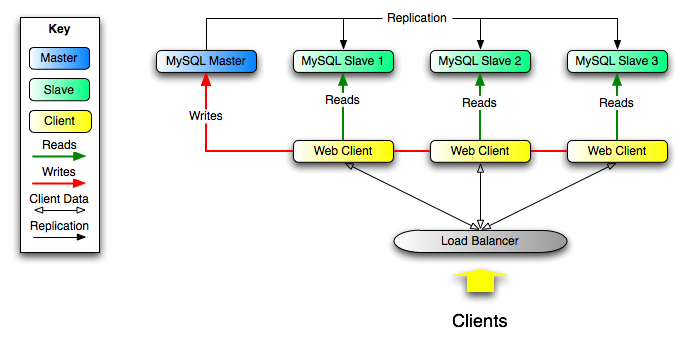
If there is an issue in connecting, you can try starting slave with a command to skip over it:

SET GLOBAL SQL\_SLAVE\_SKIP\_COUNTER = 1; SLAVE START;









**MongoDB:**

1. **Install latest version of MongoDB from apt-get repository**

### Import the public key used by the package management system.

sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 9DA31620334BD75D9DCB49F368818C72E52529D4

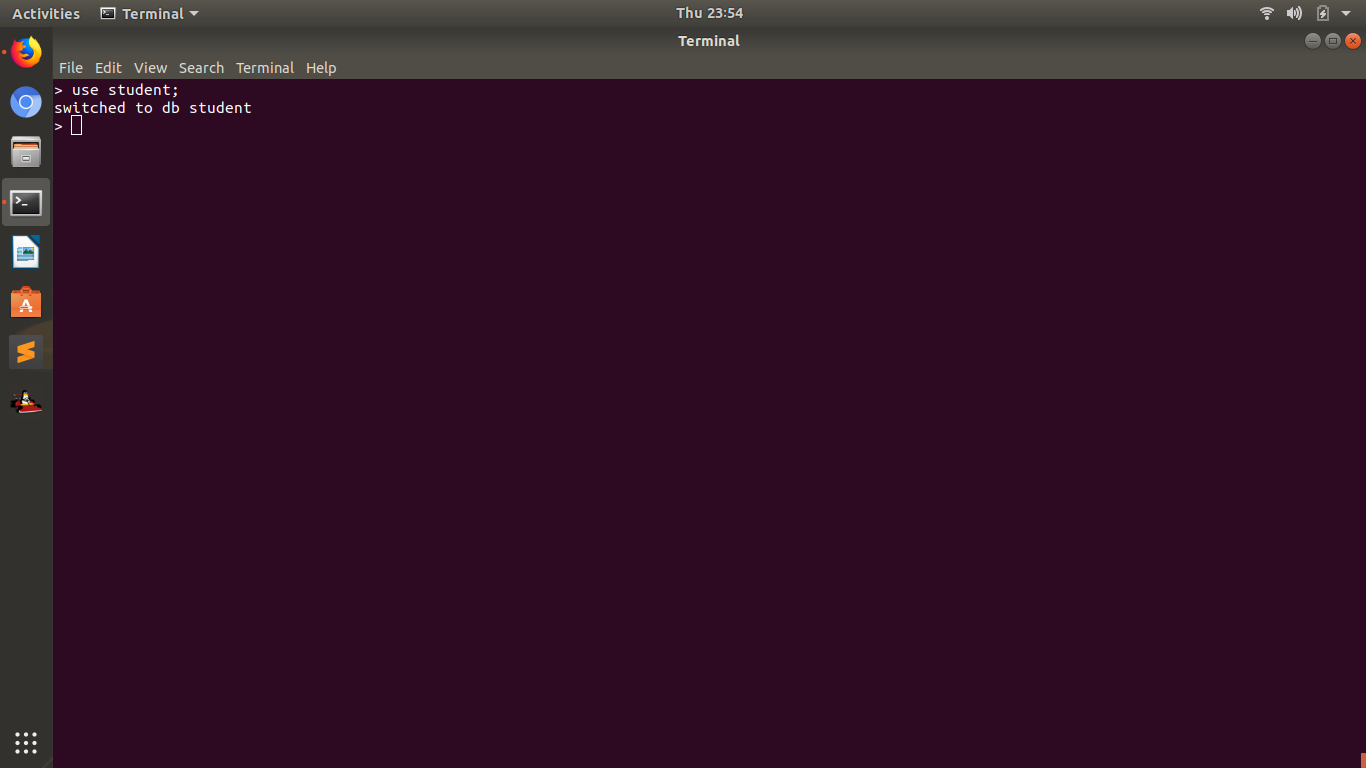
Create a list file for MongoDB.

echo "deb [ arch=amd64 ] https://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.0 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.0.list

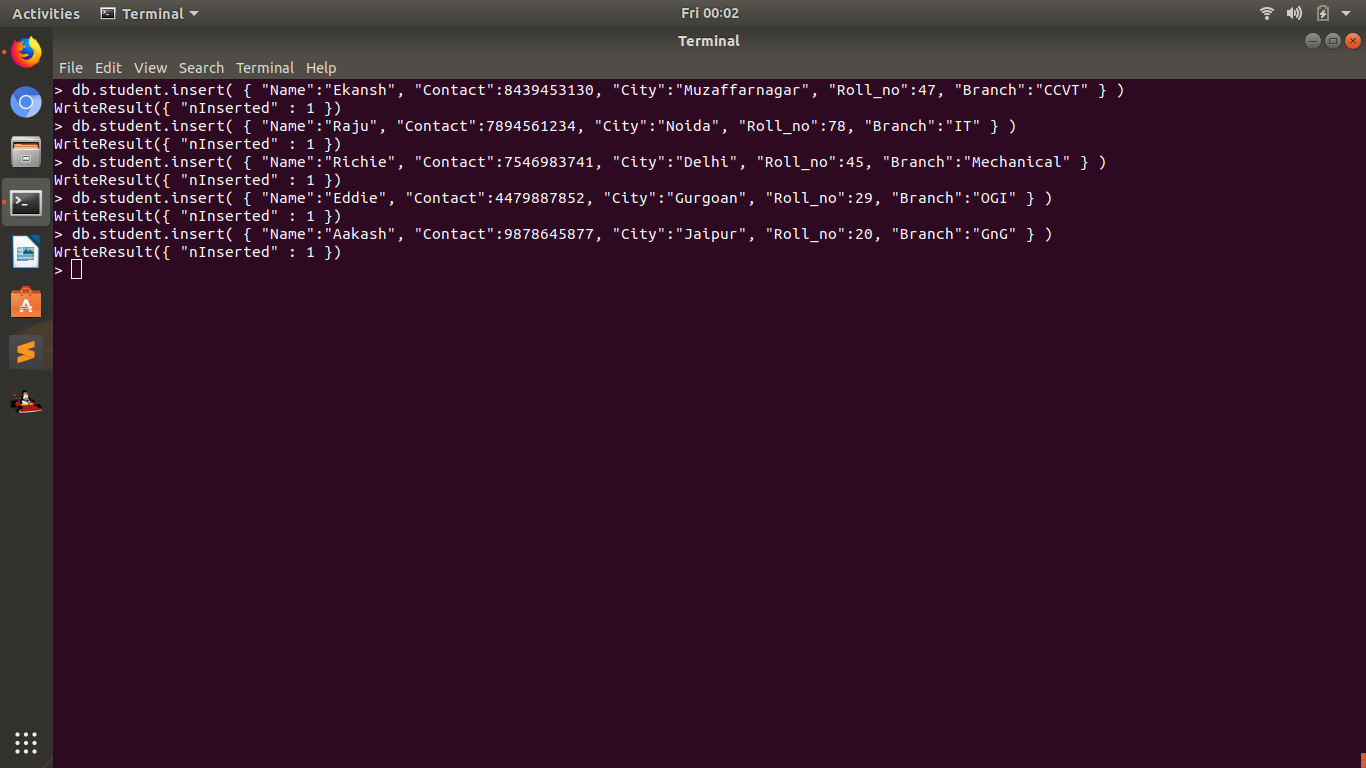
sudo apt-get update

sudo apt-get install -y mongodb-org

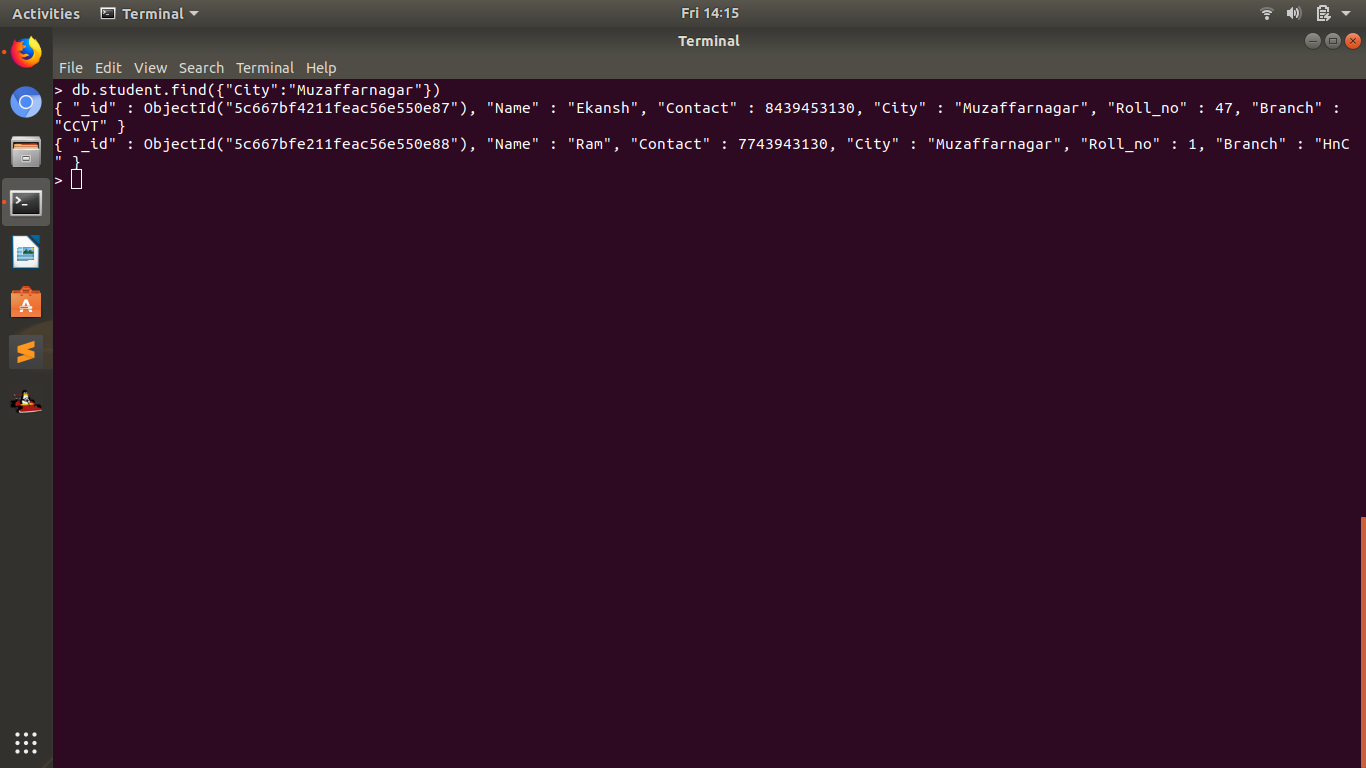
1. **Create a database student**



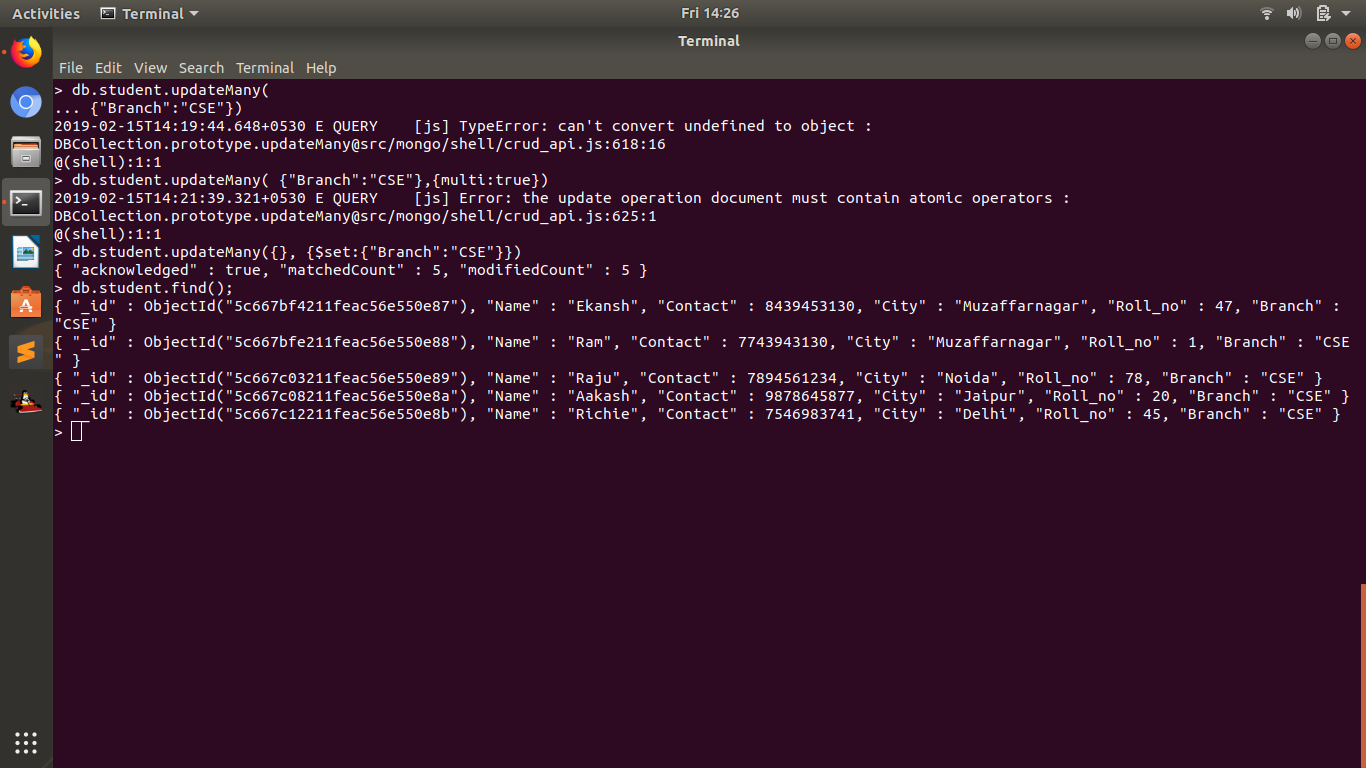
1. **Insert operation : 5 students data (Name, Contact, City, Roll No, Branch)**



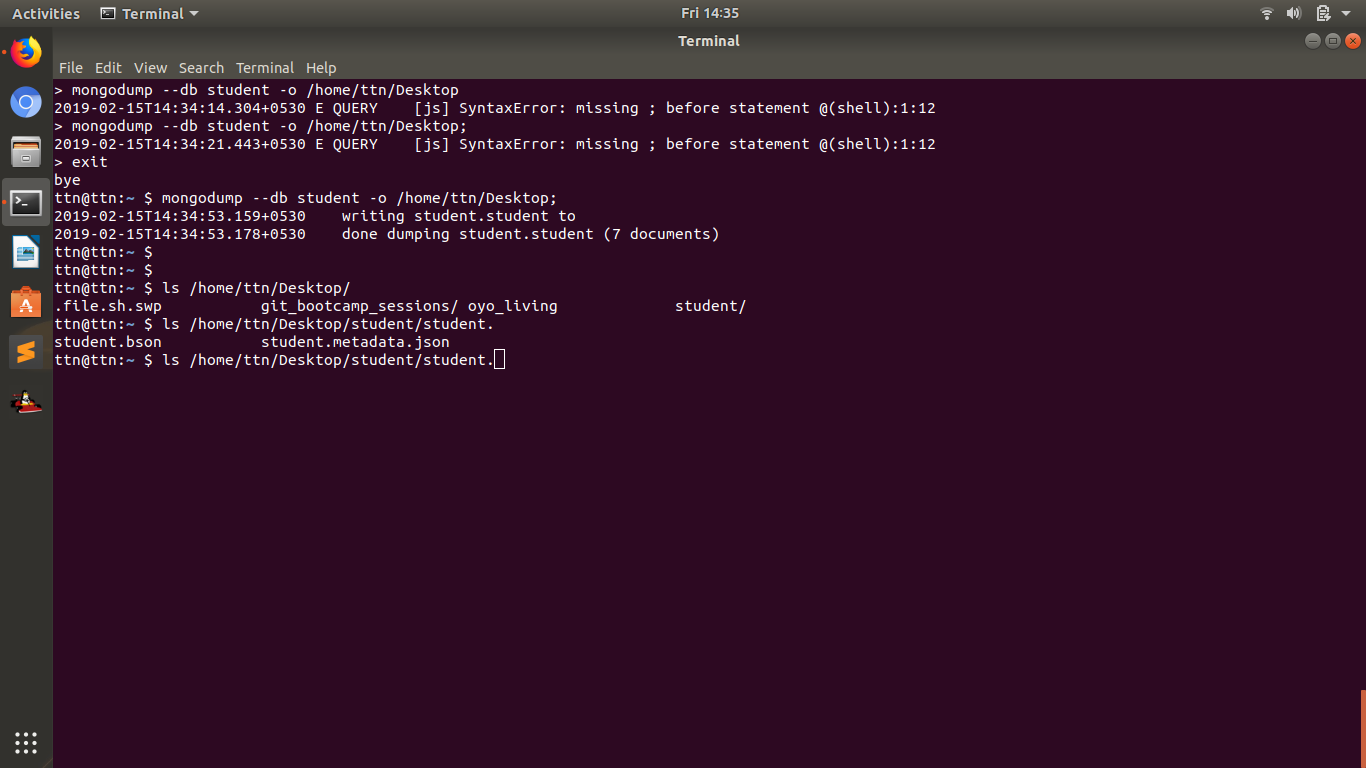
1. **Read operation : All the students belong to a particular city**



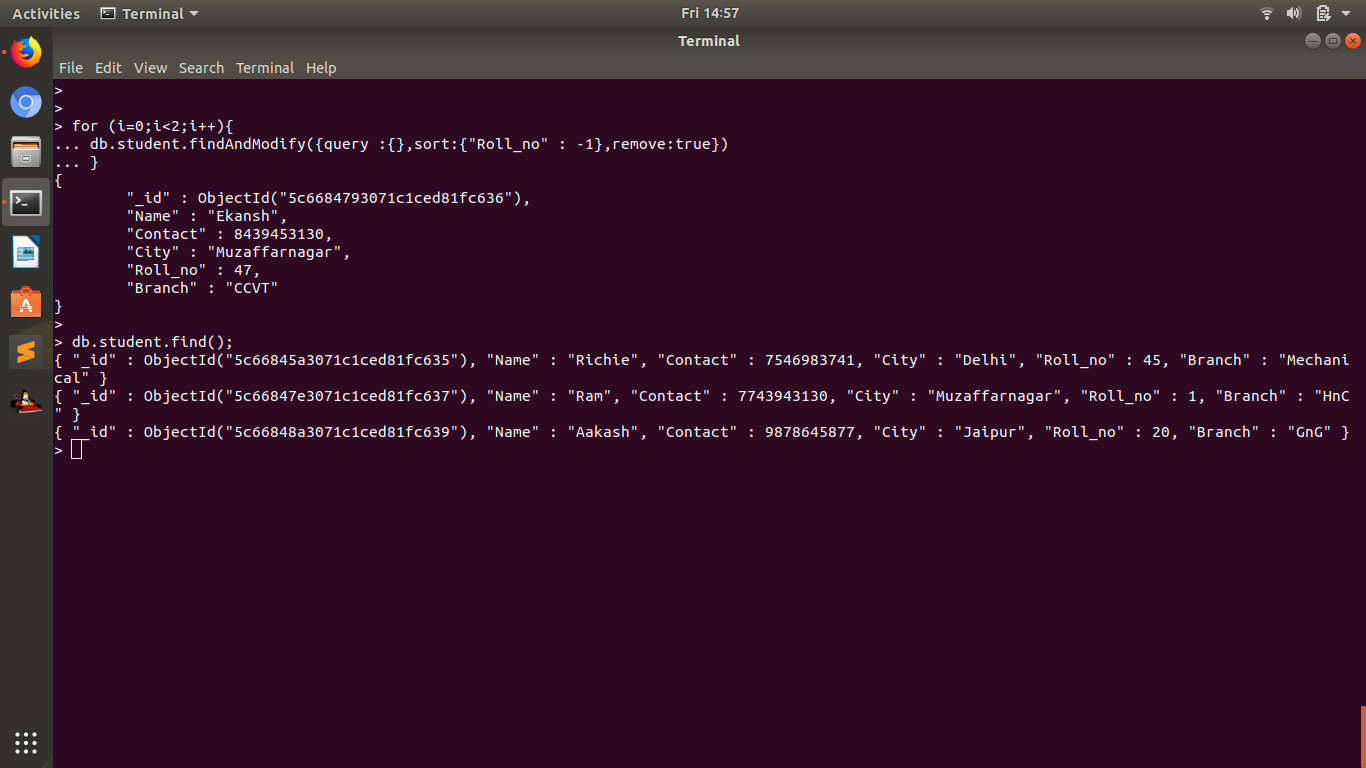
1. **Update operation : Update the branch of all the students to CSE**



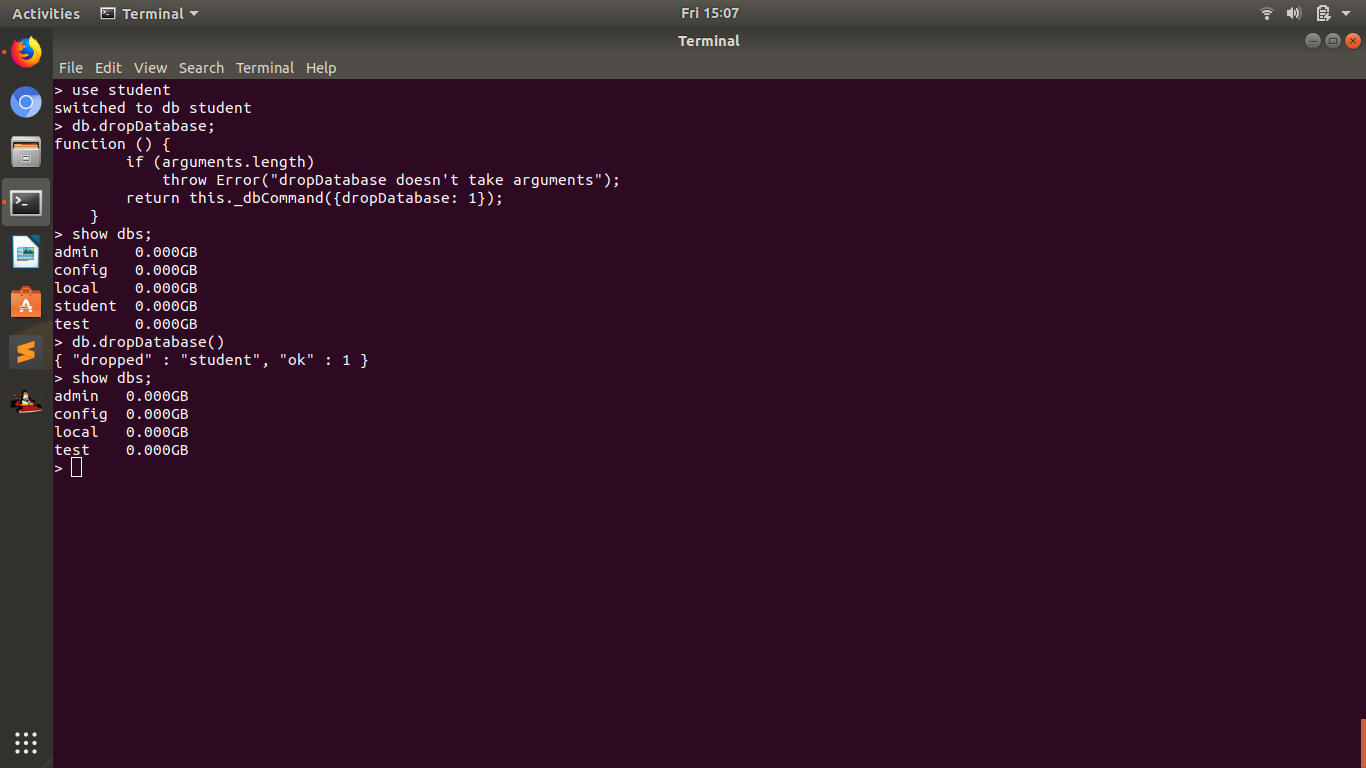
1. **Take dump of the database**



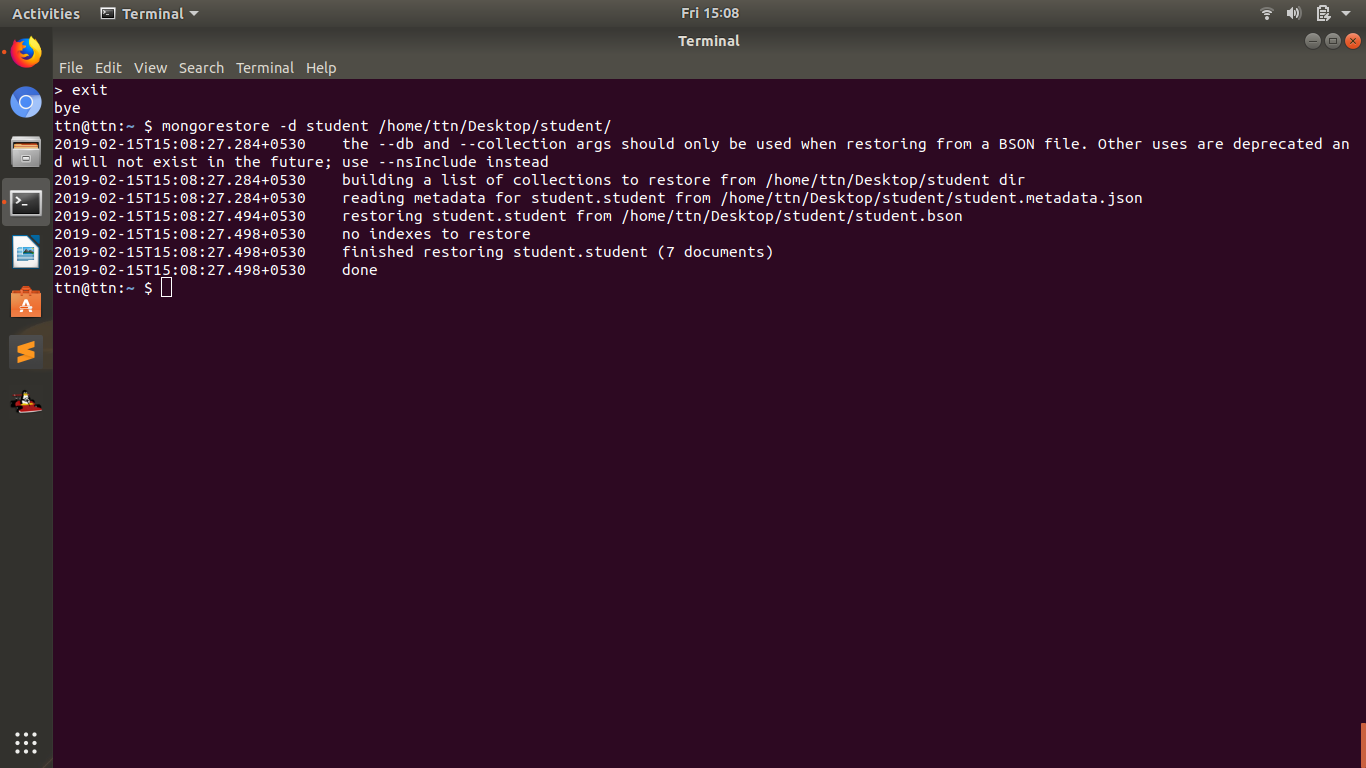
1. **Delete operation : Delete the record of last 2 students according to the roll number**



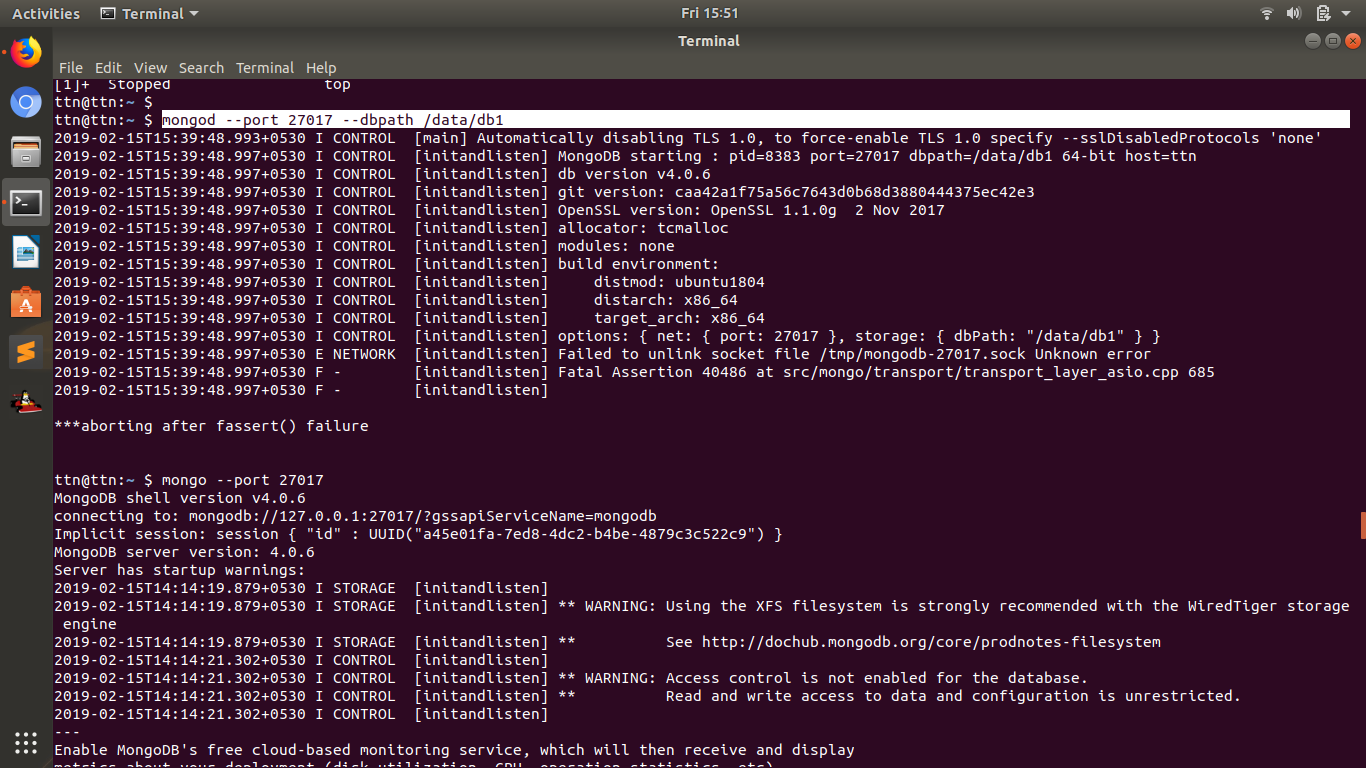
1. **Drop the database**

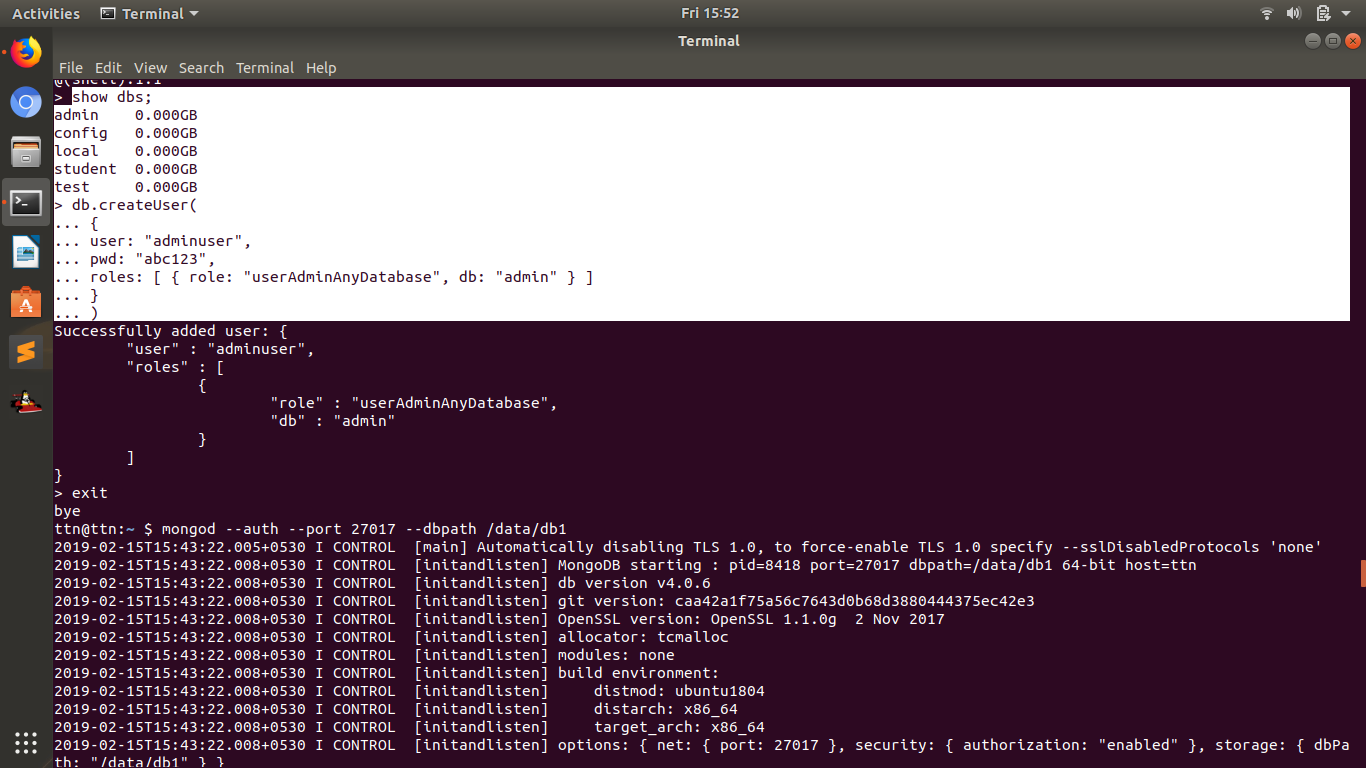


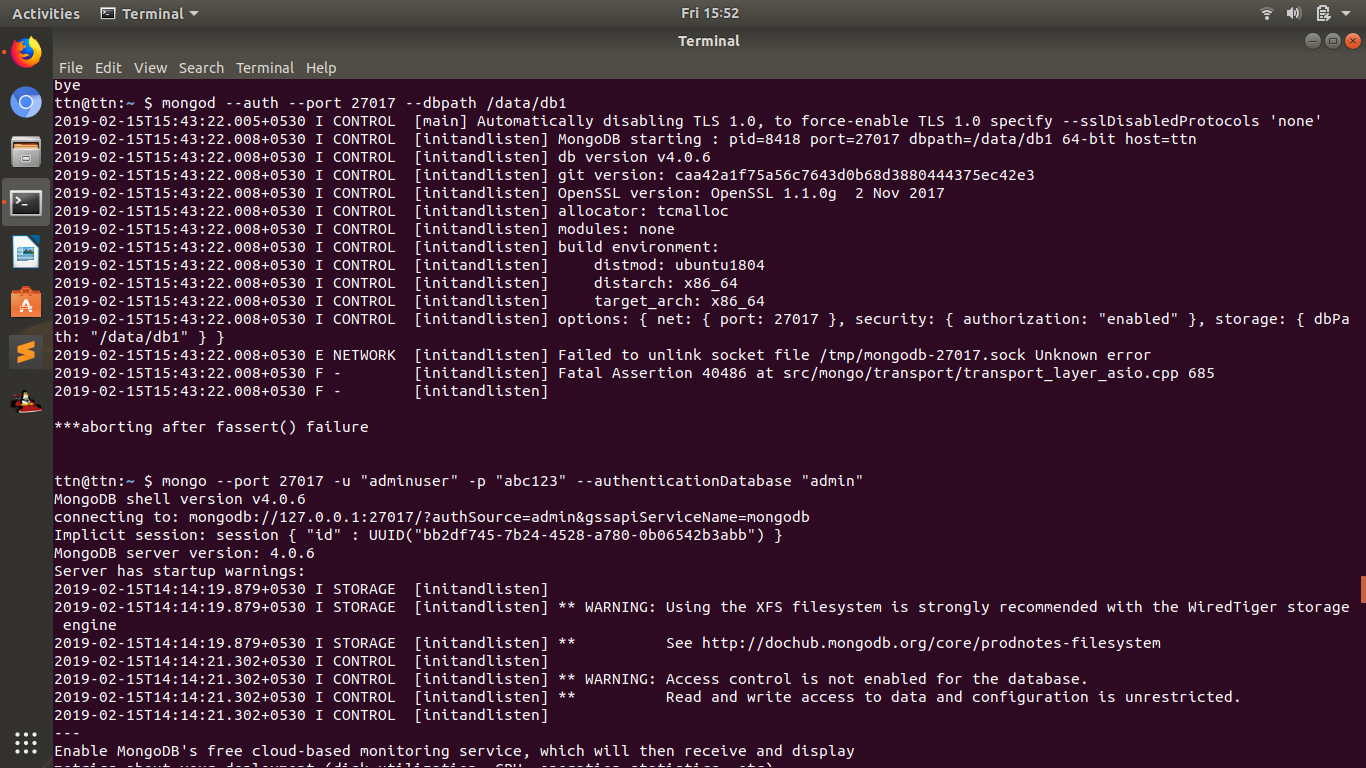
1. **Restore the database again to have the full data**

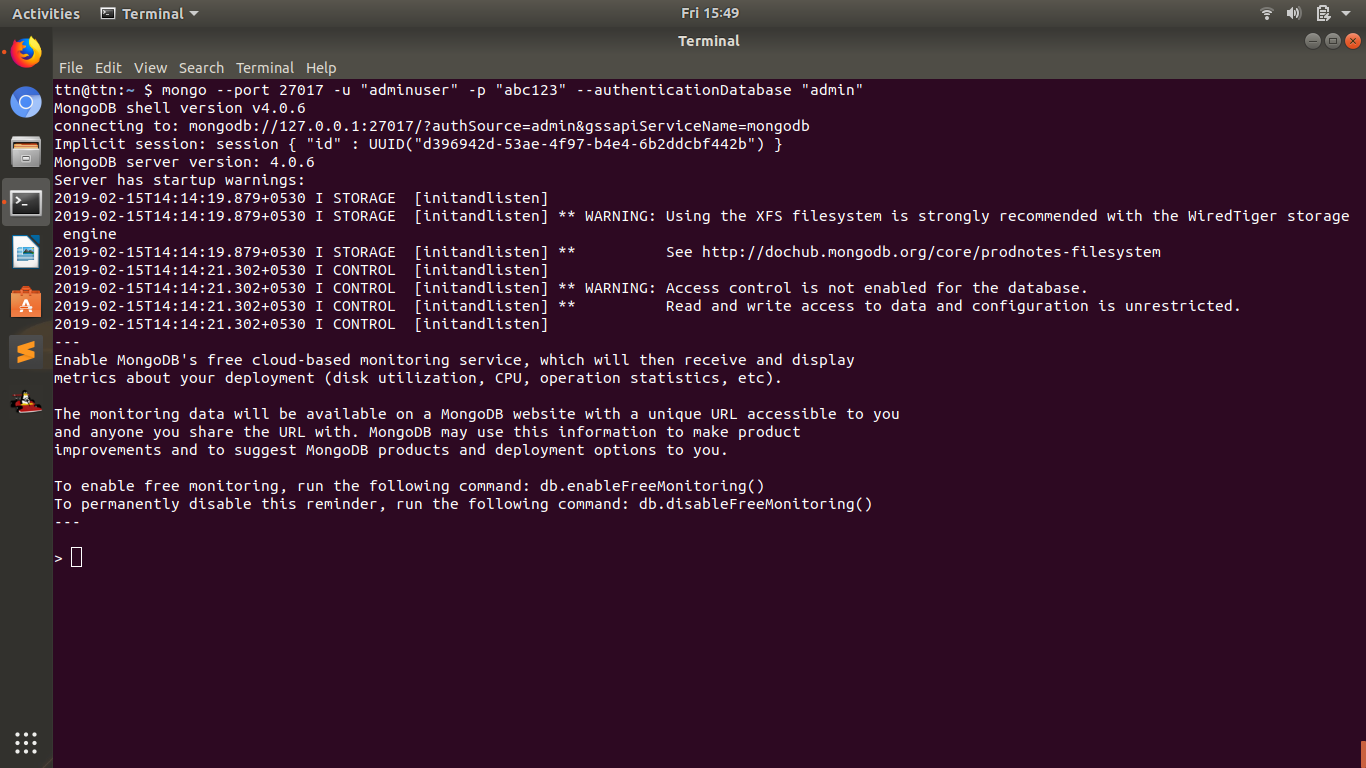


1. **Enable authentication on the Mongo**

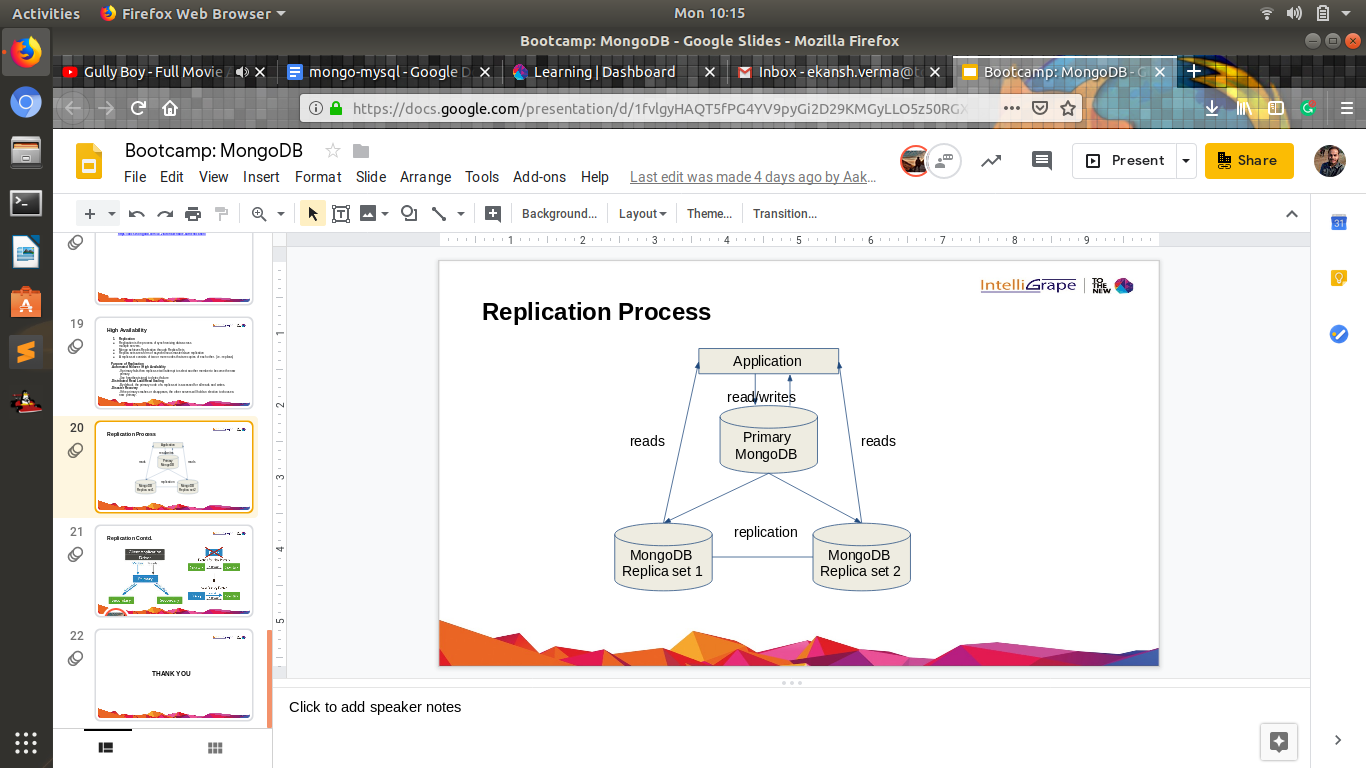


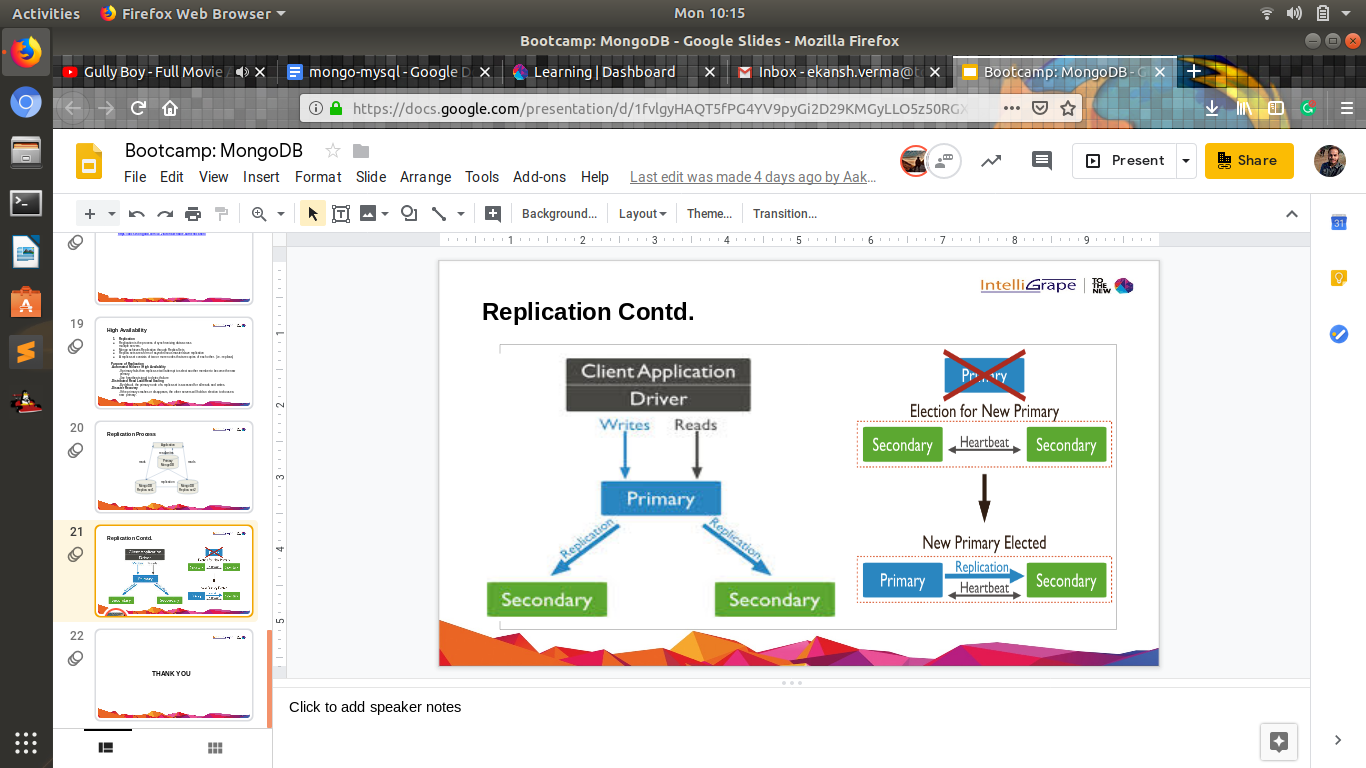






1. **Setup replica set**





### **1.Start each member of the replica set with the appropriate options.**

For each member, start a mongod instance with the following settings:

* Set replication.replSetName option to the replica set name,
* If your application connects to more than one replica set, each set should have a distinct name. Some drivers group replica set connections by replica set name.
* Set net.bindIp option to the hostname/ip or a comma-delimited list of hostnames/ips, and
* Set any other settings as appropriate for your deployment.

$mongod --replSet "rs0" --bind\_ip localhost,<hostname(s)|ip address(es)>

For <hostname(s)|ip address(es)>, specify the hostname(s) and/or ip address(es) for your mongod instance that remote clients (including the other members of the replica set) can use to connect to the instance.

Alternatively, you can also specify the replica set name and the ip addresses in a configuration file:

replication:

replSetName: "rs0"

net:

bindIp: localhost,<hostname(s) | ip address(es)>

To start mongod with a configuration file, specify the configuration file path with the --config option:

$mongod --config <path-to-config>

### **2.Connect a mongo shell to one of the mongod instances.**

$mongo

### **3.Initiate the replica set**

From the mongo shell, run rs.initiate() on replica set member 0.

rs.initiate( {

\_id : "rs0",

members: [

{ \_id: 0, host: "mongodb0.example.net:27017" },

{ \_id: 1, host: "mongodb1.example.net:27017" },

{ \_id: 2, host: "mongodb2.example.net:27017" }

]

})

### **4. View the replica set configuration.**

Use rs.conf() to display the replica set configuration object:

$rs.conf()

### **5.Ensure that the replica set has a primary.**

Use rs.status() to identify the primary in the replica set.