Backup command rsync

The command-line utility `rsync` is the most widely used backup command in Linux for backing up or synchronizing data. This utility was developed in 1996 by Andrew Tridgell and Paul Mackerras.

This utility is mostly installed in all popular Linux distributions. If it is not installed, then run the following commands:

```
$apt-get install rsync
```

rsync is a powerful utility. It can copy or synchronize files in the same computer or across the network in another continent-based computer over the internet.

Syntax

```
$rsync -options source_folder destination_folder
```

consider that you want to copy from /home/student/data_folder to your mounted USB pen drive /media/usb_drive/data_folder. Then, the backup command would be:

```
$ sudo rsync -a /home/student/test /media/student/ name_of_drive /test
```

The preceding command will copy a test folder to your mounted USB pen drive. Of course, you will need to check the exact path for the mounted pen drive. As per the volume label of the pen drive, the exact path of the destination folder may change. We have used the -r option from recursively copying folder with all of its subfolders and files.

If we want to ensure that in the source folder, a certain file or folder is deleted, then corresponding files or folders should be deleted from the destination backup folder as well, and for that we need to use the -delete option.

If we want to backup symbolic link files along with ownership, file permissions, and time stamps, then we should use option -a.

Then, the updated command would be:

```
$ sudo rsync -a -delete /home/student/test  /media/student/ganesh/test
```

If want to observe the progress of the backup, then add the -v option.

```
$ sudo rsync -av -delete /home/student/test  /media/student/ganesh/test
```

If file sizes are very big and you want to compress the files and then take a backup, then simply add the -z option. This will save network bandwidth if you are going to transfer GB-or TB-sized data.

```
$ sudo rsync -avz -delete /home/student/test /media/student/ganesh/test
```

By default, rsync deletes any partially transferred files if the backup operation is interrupted. If we want to keep partially transferred files, then we need to add the -P option. The updated backup command will be as follows:

```
$ sudo rsync -avzP -delete /home/student/test  /media/student/ganesh/test
```

Backup across the network
For taking backup across the network, we will need to install the ssh protocol package. Normally, it will already be installed. If it is not installed, then use the following command:

```
sudo apt-get install ssh
```

The command to synchronize data from across the network to your local folder will be as follows:

```
$ rsync -avzP --delete -e ssh user@ip_address:source-folder /destination-folder

$ rsync -avzP --delete -e ssh student@192.168.10.55: /home/student/data-folder
  /home/student/data-folder
```

If we want to synchronize local folders to a remote computer, then the command would be as follows:

```
$ rsync -avzP --delete -e ssh source-folder user@ip_address:destination-folder
```

The actual command would be as follows:

```
$ rsync -avzP --delete -e ssh /home/student/data-folder  student@192.168.10.55: /home / student/
data-folder
```

You will need to replace the username and IP address of the destination PC with the required username and password.

If the remote PC has been configured with port forwarding, such as when we have to use port number 12345 while using the ssh command, then the rsync command will be as follows:

```
 $ rsync -avzP --delete -e 'ssh -p 12345' student@192.168.10.55: /home/student/data-folder
/home/student/data-folder
```

Automating backup activity
If you want to automate taking backup activity every day at 7.30 pm, then you will need to use the crontab functionality
You will need to enter the crontab -e command and enter the rsync command in it.

For regular backup at 7.30 pm every day, enter the following line in the crontab editor

```
30 19 * * * rsync -avz -delete /home/student/data-folder  /media/usb_drive/data-folder
```

Database administration using shell script (notes will be shared).

Database administration using shell scripting

Working with a MySQL Database
In this section, you will learn about automating MySQL database administration. Let's start with very basic activities, as discussed in the following topics.

Checking the version of MySQL database
We will initially start by checking which version of MySQL is installed. With this script, we will ensure that MySQL is properly installed and we are able to communicate with it with root privileges. This script will report us the database version of MySQL. It executes a SELECT VERSION() query to get the value of the database version. Let's create the script mysql_01.sh:

```
#!/bin/bash
mysql -u root -pTraining2@^ <<MY_QUERY
SELECT VERSION();
MY_QUERY
```

Save the program and execute it as follows:

```
   $ chmod +x mysql_01.sh
   $ ./mysql_01.sh
```

The output will be

```
   VERSION()
   5.7.22
```

Creating a database
In this section, you will learn about a creating new database in MySQL. We are going to use this database throughout this chapter. Create the script mysql_02.sh to create the database:

```
 $ chmod +x mysql_02.sh $ ./mysql_02.sh
```

Show databases
To see all of the databases, we are going to use the command show databases. Create the script mysql_03.sh to see all of the databases:

```
#!/bin/bash
mysql -u root -pTraining2@^ <<MY_QUERY
show databases;
MY_QUERY
```

The output will be:

```
   Database
 information_schema
 mysql
 performance_schema
 sys
 testdb
```

Creating a user

Next, you will learn about creating a new user using the MySQL command inside shell script. The CREATE USER command creates a new user and we set a password for the newly created user. The command GRANT ALL ON provides privileges to make changes to database by newly created user.

In the next script, we will create a user and we will also grant all of the privileges on the database. Let's create the script mysql_04.sh to create a user:

```
#!/bin/bash
mysql -u root -pTraining2@^ <<MY_QUERY
CREATE USER 'user1'@'localhost' IDENTIFIED BY 'Test623@!@!';
GRANT ALL ON testdb.* TO 'user1'@'localhost';
select user from mysql.user;
MY_QUERY
```

Save the script and run it as follows:

```
    $ chmod +x mysql_04.sh
    $ ./mysql_04.sh
```

The output will be:

```
  user
  mysql.session
  mysql.sys
  root
  user1
```

If you want to grant the privileges on all of the databases, then you just have to put * on database name. Then, the Grant query will be as follows:

```
 GRANT ALL ON *.* TO 'user1'@'localhost';
```

In the preceding program, we have created one user named as user1 and have granted privileges to user1 on our testdb database.

From now onward, we are going to use the testdb database. Therefore, we will add use testdb; query to every shell script.

Creating a table in MySQL

We have successfully created a database named testdb and a user named user1, and have also granted all of the privileges on the testdb database.

Now we will create a table. Create the script mysql_05.sh to list the table:

```
#!/bin/bash
mysql -u user1 -pTest623@!   <<MY_QUERY
use testdb;
```

```
show tables;
MY_QUERY
```

Now save the program and run it as follows:

```
        $ chmod +x mysql_05.sh
        $ ./mysql_05.sh
```

The preceding command will not show presence of any table, as we only just have created our database.
Now, we are going to create a table named Authors. Create the script mysql_06.sh to create a table named Authors:

```
#!/bin/bash
mysql -u user1 -pTest623@! <<MY_QUERY
use testdb;
CREATE TABLE Authors(Id INT PRIMARY KEY AUTO_INCREMENT, Name VARCHAR(25));
MY_QUERY
```

Now save the program and run it as follows:

```
        $ chmod +x mysql_06.sh
        $ ./mysql_06.sh
```

The output will be:

```
   Tables_in_testdb
   Authors
```

Inserting data into table
Now we will insert some data into our Authors table. We will use insert into query to insert the data. Create the script 7_insert_into_Authors.sh to insert the data:

```
#!/bin/bash
mysql -u user1 -pTest623@! <<MY_QUERY
use testdb;
Insert into Authors(NAME)values('William Shakespeare');
Insert into Authors(NAME)values('Charles Dickens');
Insert into Authors(NAME)values('Jane Austen');
Insert into Authors(NAME)values('George Orwell');
Insert into Authors(NAME)values('Oscan Wilde');
MY_QUERY
```

Now save the program and run it as follows:

```
        $ chmod +x mysql_07.sh
        $ ./mysql_07.sh
```

After executing this script, records will have been successfully inserted. To check this you will need to use the command select.

Retrieving data from the table
To retrieve data from a table, we use the select statement. We can retrieve all of the records from the table, or we can retrieve a specific record using the select statement.

To retrieve all of the records from the table, we have to use * in the select statement. So, the query will be as follows:

```
  select * from table_name;
```

Create the script mysql_08.sh to get all of the records from the table:

```
#!/bin/bash
mysql -u user1 -pTest623@! <<MY_QUERY
use testdb;
select * from Authors;
MY_QUERY
```

Now save the program and run it as follows:

```
      $ chmod +x mysql_08.sh
      $ ./mysql_08.sh
```

The output will be:

```
   Id   Name
   1    William Shakespeare
   2    Charles Dickens
   3    Jane Austen
   4    George Orwell
   5    Oscan Wilde
```

Updating data
If we want to modify or replace the content of any row of a table, then we need to use the UPDATE command. We can modify single or multiple fields of a table.

Create mysql_09.sh to update the name of specified Id:

```
#!/bin/bash
mysql -u user1 -pTest623@! <<MY_QUERY
use testdb;
UPDATE Authors SET Name = 'Mansi Joshi' WHERE Id = 1;
select * from Authors;
MY_QUERY
```

Now, save the program and run it as follows:

```
      $ chmod +x mysql_09.sh
```

```
$ ./mysql_09.sh
```

The output will be:

```
Id   Name
1    Mansi Joshi
2    Charles Dickens
3    Jane Austen
4    George Orwell
5    Oscan Wilde
```

Deleting data
If the need arises to delete a row, or multiple rows of a table because the data has become obsolete, then we have to use the DELETE command. When we want to make large-scale deletions in a table, this command is very handy.

Now, create the script mysql_10.sh to delete the record:

```
#!/bin/bash
mysql -u user1 -pTest623@! <<MY_QUERY
use testdb;
DELETE FROM Authors WHERE Name = 'Mansi Joshi';
select * from Authors;
MY_QUERY
```

Now, save the program and run it as follows:

```
$ chmod +x mysql_10.sh
$ ./mysql_10.sh
```

The output will be:

```
Id   Name
2    Charles Dickens
3    Jane Austen
4    George Orwell
5    Oscan Wilde
```

Altering a table
If we want to modify a table's basic structure, such as, adding an extra column, then we need to use the Alter command. When we need to put more information related to table objective, using this command, we can add extra column. We can use this command to modify table as well as database also.

In the next script, we are going to alter the table. Create the script mysql_11.sh to alter the table definition:

```
#!/bin/bash
mysql -u user1 -pTest623@! <<MY_QUERY
use testdb;
ALTER TABLE Authors
ADD ADDRESS VARCHAR(25);
MY_QUERY
```

In this shell script, we have added a new field named ADDRESS using ALTER command.

Now save the program and run it as follows:

```
$ chmod +x mysql_11.sh
$ ./mysql_11.sh
```

Describing a table
If we need to know more about the overall information or query the execution plan of an entire table, we need to use the DESCRIBE command.

We will create the script mysql_12.sh to obtain the table structure:

```
#!/bin/bash
mysql -u user1 -pTest623@!  <<MY_QUERY
use testdb;
desc Authors;
MY_QUERY
```

Save the script and run it as follows:

```
$ chmod +x mysql_12.sh
$ ./mysql_12.sh
```

The output will be:

```
Field Type  Null  Key   Default          Extra
Id    int(11)     NO    PRI   NULL  auto_increment
Name  varchar(25) YES   NULL
ADDRESS varchar(25)     YES   NULL
```

Drop the table
If we want to remove any particular table from a database, then we need to use the DROP TABLE command. This command will remove the table definition along with all related partition information.

Now we will create the script mysql_13.sh to drop the table:

```
#!/bin/bash
```

```
mysql -u user1 -pTest623@! <<MY_QUERY
use testdb;
DROP TABLE Authors;
MY_QUERY
```

Save the program and run it as follows:

```
    $ chmod +x mysql_13.sh
    $ ./mysql_13.sh
```

After executing this script, your table will have been deleted. To check whether your table has been deleted, run this same script again, and you will get the following error message:

```
ERROR 1051 (42S02) at line 3: Unknown table 'testdb.Authors'
```

This means that your table has successfully been deleted.

Drop the database

If you want to delete the database itself, then the DROP DATABASE command will delete the complete database along with all of the tables inside it.

Now, we will create the script mysql_14.sh to drop the database:

```
#!/bin/bash
mysql -u user1 -pTest623@! <<MY_QUERY
DROP DATABASE testdb;
MY_QUERY
```

Now, save the program and run it as follows:

```
    $ chmod +x mysql_14.sh
    $ ./mysql_14.sh
```

After executing this script, your database will have been deleted. To check this, run this same script again, and you will get the following error message:

```
ERROR 1008 (HY000) at line 2: Can't drop database 'testdb'; database doesn't exist
```

Executing commands on remote server, executing multiple commands on remote server in password based authentication environment systems and password (commands were demonstrated in the class and the same will be sent soon).

Connecting to a remote server:
We can connect to a remote server using ssh from our local/working server.
    They are two ways to connect with remote server using ssh.
        Using password or Password Authentication)
        Using ssh-keys (Password-less Authentication)

Executing commands on remote server:
Two ways (for both password and password less Authentication)
First way:

```
ssh user_name@remote_server
```

Provide the password if it is password authentication connection.
Now run the command and see the result
Run exit command to close remote session
Note: This is not good for automation

Second way:

```
ssh user_name@remote_server “command”
```

Provide the password if it is password authentication connection.
This is good for automation, if the connection is password less authentication
Note: ssh -t user_name@remote_server “command”

```
ssh -t -o StrictHostKeyChecking=No user_name@remote_server “command”
```

Executing command on remote server without logging into remote server:

```
ssh -t -o StrictHostKeyChecking=No user_name@remote_server “command”
```

Executing multiple commands on remote server without logging into remote server

```
ssh -t -o StrictHostKeyChecking=No user_name@remote_server “cmd1;cmd2;cmd3”
```

Providing password for ssh using sshpass:
sshpass is a Non-interactive ssh password authentication

Infrastructure monitoring and sending mail alerts based on infrastructure status (commands and script will be sent)

Docker
https://www.youtube.com/watch?v=pTFZFxd4hOI

CuRL
https://www.youtube.com/watch?v=I6id1Y0YuNk

Curl in HINDI

https://www.youtube.com/watch?v=mR5GLQWx8DY