```
CREATE USER 'newuser'@'localhost' IDENTIFIED BY 'password';

```

Note: When adding users within the MySQL shell in this tutorial, we will specify the user's host as `localhost` and not the server's IP address. `localhost` is a hostname which means "this computer," and MySQL treats this particular hostname specially: when a user with that host logs into MySQL it will attempt to connect to the local server by using a Unix socket file. Thus, `localhost` is typically used when you plan to connect by SSHing into your server or when you're running the local `mysql` client to connect to the local MySQL server.

At this point newuser has no permissions to do anything with the databases. In fact, even if newuser tries to login (with the password, password), they will not be able to reach the MySQL shell.
Therefore, the first thing to do is to provide the user with access to the information they will need.

```
GRANT ALL PRIVILEGES ON * . * TO 'newuser'@'localhost';
Grant select on test.emp to 'newuser'@'localhost'
```

The asterisks in this command refer to the database and table (respectively) that they can access—this specific command allows to the user to read, edit, execute and perform all tasks across all the databases and tables.

Please note that in this example we are granting newuser full root access to everything in our database. While this is helpful for explaining some MySQL concepts, it may be impractical for most use cases and could put your database's security at high risk.
Once you have finalized the permissions that you want to set up for your new users, always be sure to reload all the privileges.

```
FLUSH PRIVILEGES;

```

Your changes will now be in effect.

How To Grant Different User Permissions

Here is a short list of other common possible permissions that users can enjoy.

- ALL PRIVILEGES- as we saw previously, this would allow a MySQL user full access to a designated database (or if no database is selected, global access across the system)

- CREATE- allows them to create new tables or databases
- DROP- allows them to them to delete tables or databases
- DELETE- allows them to delete rows from tables
- INSERT- allows them to insert rows into tables
- SELECT- allows them to use the `SELECT` command to read through databases
- UPDATE- allow them to update table rows
- GRANT OPTION- allows them to grant or remove other users' privileges

To provide a specific user with a permission, you can use this framework:

```
GRANT type_of_permission ON database_name.table_name TO 'username'@'localhost';
Grant select,insert on ‘mydb.dept’ to user2@localhost’
Grant select,insert on ‘mydb.dept’ to user3@localhost’ with grant option
```

If you want to give them access to any database or to any table, make sure to put an asterisk (*) in the place of the database name or table name.

Each time you update or change a permission be sure to use the Flush Privileges command.

If you need to revoke a permission, the structure is almost identical to granting it:

```
REVOKE type_of_permission ON database_name.table_name FROM 'username'@'localhost';
Revoke insert on test.dept from 'username'@'localhost';
```

Note that when revoking permissions, the syntax requires that you use `FROM`, instead of `TO` as we used when granting permissions.

You can review a user's current permissions by running the following:

```
SHOW GRANTS FOR 'username'@'localhost';
```

Just as you can delete databases with DROP, you can use DROP to delete a user altogether:

```
DROP USER 'username'@'localhost';
```

To test out your new user, log out by typing:

```
quit
```

and log back in with this command in terminal:

```
mysql -h localhost -u [username] -p
```