Installing, Securing & Logging Into MySQL

AiCE Software Engineering Foundations — MySQL Programming

You now know what data, databases, MySQL, and SQL are. This lesson gets MySQL running on your own machine — installed, secured, and ready for the mysql> prompt where every command you write from here on will land.

1. Two Paths: Ubuntu or PowerShell

You already know from the Terminal series that you are working in either the Ubuntu terminal or PowerShell on Windows. MySQL installs differently on each, so follow the section that matches your machine. If you already installed and logged in on one, you do not need to repeat the process on the other.

2. Installing MySQL on Ubuntu

Open your terminal. First, update your system's package information:

```
sudo apt update
```

Let it run, then install MySQL itself:

```
sudo apt install mysql-server
```

Type y and press Enter when it asks you to confirm, and wait for the installation to finish. Confirm it installed correctly:

```
mysql --version
```

You should see a version number printed — MySQL is installed. Now start it:

```
sudo systemctl start mysql
```

Confirm it is running:

```
sudo systemctl status mysql
```

You should see active (running). Press q to exit that screen. Finally, make sure MySQL starts automatically every time your machine boots:

```
sudo systemctl enable mysql
```

3. Installing MySQL on PowerShell

If you already installed and logged in on Ubuntu above, you can skip this section entirely — you do not need MySQL installed twice. Otherwise, open PowerShell as administrator and type:

```
scoop install mysql
```

From this point on, whether you are on Ubuntu or PowerShell, everything is identical — once you see the mysql> prompt, the two operating systems disappear and only SQL remains.

4. Securing Your Installation

MySQL ships with insecure defaults out of the box, so before doing anything else, run the built-in security script:

```
sudo mysql_secure_installation
```

When it asks about a password validation component, type n. For the remaining questions — removing anonymous users, disabling remote root login, removing the test database, and reloading privilege tables — answer yes (Y) to each one. These defaults exist purely for demos and testing, and every one of them is a security risk on a real machine.

5. Setting a Real Root Password

Log in without a password for one last time to set a proper one:

```
sudo mysql
```

Then, at the mysql> prompt, set the password:

```
ALTER USER 'root'@'localhost' IDENTIFIED BY 'new_password';
FLUSH PRIVILEGES;
```

> **Remember This Password**
> *You will use this exact password every single time you log in to MySQL from now on. Write it down somewhere safe — there is no autofill for it.*

6. Logging In

On Ubuntu, log in with:

```
sudo mysql -u root -p
```

On PowerShell, the sudo prefix is not needed:

```
mysql -u root -p
```

Enter the password you just set. Your terminal prompt changes to mysql> — that is MySQL telling you it is listening and ready for SQL commands.

7. Your First Command

Let's run something. Type:

```
SHOW DATABASES;
```

You will see a short list of default databases that ship with MySQL itself — leave these alone, as MySQL uses them internally to manage itself. Notice the semicolon at the end of the command; every SQL statement you write in MySQL ends with one, a habit you will build quickly over the next few lessons.

Key Takeaways

You installed MySQL — on Ubuntu with apt, or on Windows with Scoop — started the service, and ran mysql_secure_installation to close off its insecure defaults. You set a real root password with ALTER USER, logged in with mysql -u root -p, and confirmed everything works with your first real command, SHOW DATABASES;. From here, the

theory is behind you — the next unit puts MySQL to work, building a real database and table for the AiCE Books project and performing every CRUD operation against it: Create, Read, Update, and Delete.