Creating Databases & Tables

AiCE Software Engineering Foundations — MySQL Programming

MySQL is installed, secured, and you know your way to the mysql> prompt. Now it's time to put it to work. In this lesson you will build a small real database for AiCE Books, an online bookstore, and use it to learn every core MySQL skill — starting with the two things every project needs before it can hold a single row of data: a database, and a table.

1. All Roads Lead to CRUD

Almost everything you will ever do with a database comes down to one thing: CRUD. That stands for Create, Read, Update, and Delete — the four operations behind every login form, shopping cart, and dashboard you have ever used. This lesson gets your database and table ready; the next lesson starts putting CRUD into practice.

2. Creating the aice_books Database

Log back in to MySQL — sudo mysql -u root -p on Ubuntu, or mysql -u root -p on PowerShell. Before creating anything, check what already exists:

```
SHOW DATABASES;
```

You will see only the default databases MySQL ships with. Now create your own:

```
CREATE DATABASE aice_books;
```

Press Enter and you will see Query OK — the database has been created. Run SHOW DATABASES; again, and aice_books now appears in the list.

> **Naming Rule: No Spaces**
> *When naming anything in programming — a database, a table, a column — never use spaces. Use an underscore or a hyphen instead. aice_books, not aice books.*

3. Selecting a Database to Work In

Creating a database is only the first step. MySQL can hold many databases at once, so before you can create tables or insert data, you have to tell it which one you mean to work in:

```
USE aice_books;
```

You will see Database changed. Every command you run from this point in the lesson happens inside aice_books, until you USE a different database or log out.

4. Creating Your First Table: books

Recall that a spreadsheet has columns and rows, with headers naming each column. MySQL saves data the exact same way — you just describe the table's structure with a SQL query instead of typing headers into cells.

Let's create a table named books, with the columns id, title, author, genre, price, quantity, and date_added:

```
CREATE TABLE books (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(100) NOT NULL,
    author VARCHAR(100) NOT NULL,
    genre VARCHAR(50),
    price DECIMAL(6,2) NOT NULL,
    quantity INT DEFAULT 0,
    date_added DATE
);
```

Press Enter, and you will see Query OK — the table has been created. A few things worth pointing out:

Piece	What It Means
id INT AUTO_INCREMENT PRIMARY KEY	id is the Primary Key — a unique identifier for every row — and AUTO_INCREMENT means MySQL generates the next number automatically; you never type it yourself.
title VARCHAR(100) NOT NULL	title stores text up to 100 characters, and NOT NULL means every row must have one — it cannot be left empty.
genre VARCHAR(50)	genre has no NOT NULL, so it is allowed to be left empty for a row if needed.
price DECIMAL(6,2) NOT NULL	price stores an exact decimal number with up to 6 total digits and 2 after the decimal point — the correct type for money.
quantity INT DEFAULT 0	quantity has a DEFAULT of 0, so if you forget to specify it when inserting a row, MySQL fills it in automatically.

5. Inspecting a Table's Structure: DESCRIBE

Now that books exists, you can ask MySQL to show you its structure at any time:

```
DESCRIBE books;
```

This lists all seven columns, along with each one's type, whether it allows NULL, its key status, and its default value — a fast way to double-check a table's shape without scrolling back to find the original CREATE TABLE statement.

The table exists, but it holds nothing yet except its headers — no books have been added. That is exactly where the next lesson picks up.

Key Takeaways

You created a real database with CREATE DATABASE, told MySQL which one to work in with USE, and built your first table with CREATE TABLE — defining a primary key,

text columns, a decimal column for prices, and a default value for quantity. DESCRIBE lets you inspect any table's structure on demand. The books table exists but is completely empty — the next lesson fills it with real data and starts putting all four CRUD operations to work: inserting new books, reading them back, updating stock levels, and safely deleting a record.