Updating & Deleting Data Safely

AiCE Software Engineering Foundations — MySQL Programming

You can create books and read them back, filtered, sorted, and limited exactly how you want. This lesson finishes CRUD with the two operations that actually change what's stored — Update and Delete — and the single habit that protects you from both of them going badly wrong.

1. The Shape of UPDATE

UPDATE changes existing data in a table. Let's say AiCE Books just restocked "Sapiens", and the quantity needs to go from 15 up to 40:

```
UPDATE books SET quantity = 40 WHERE title = 'Sapiens';
```

Press Enter and you will see Query OK, 1 row affected. Confirm it worked:

```
SELECT title, quantity FROM books WHERE title = 'Sapiens';
```

quantity is now 40.

2. Why WHERE Is Never Optional

UPDATE books SET quantity = 40; with no WHERE clause does not update one book — it updates every single row in the table, setting every book's quantity to 40 at once. There is no undo button and no confirmation prompt. MySQL does exactly what you asked, immediately.

> **Always Use WHERE With UPDATE**
> *Before you press Enter on any UPDATE, ask yourself: does this WHERE clause identify exactly the row or rows I mean to change, and nothing else? If you're not sure, run the matching SELECT first to see exactly which rows would be affected.*

3. Updating More Than One Column at Once

SET can take more than one column, separated by commas, so you can change several fields in a single statement:

```
UPDATE books SET quantity = 45, price = 19.99 WHERE title =
'Sapiens';
```

Both quantity and price change together, in one query, against the same WHERE condition.

4. The Shape of DELETE

DELETE removes rows entirely. Let's say "The Alchemist" is out of print, and AiCE Books is no longer stocking it. Before deleting anything, always SELECT first, to confirm you're about to remove the right record:

```
SELECT * FROM books WHERE title = 'The Alchemist';
```

You can see the exact record you're about to remove. Now delete it:

```
DELETE FROM books WHERE title = 'The Alchemist';
```

This produces Query OK, 1 row affected. Confirm the result:

```
SELECT * FROM books;
```

"The Alchemist" is gone, and five books remain.

5. The Habit That Keeps You Safe

The exact same rule from UPDATE applies to DELETE — never run it without a WHERE clause, unless you genuinely mean to empty the entire table. DELETE FROM books; with nothing after it deletes every book, permanently, in one line.

Before You Run It	Ask Yourself
UPDATE ... SET ...	Does the WHERE clause match only the row(s) I actually mean to change?
DELETE FROM ...	Have I run the matching SELECT first, to confirm exactly what I'm about to remove?
Either statement, no WHERE at all	Am I certain I want this applied to every single row in the table?

Key Takeaways

UPDATE changes existing rows with SET, and DELETE removes rows entirely — both always paired with a WHERE clause that identifies exactly which rows are affected, since leaving it off applies the change to the whole table at once.
SELECT-before-DELETE is a habit worth keeping permanently: confirm what you're about to remove before you remove it. With all four CRUD operations now covered directly in MySQL — Create, Read, Update, and Delete — the next unit takes this exact same aice_books database and connects it to Python, so these same operations can be triggered from application code instead of typed by hand at the mysql> prompt.