Lesson 7: Updating and Deleting Data

Duration: 20 minutes
Deliverable: `lesson7_modifications.sql`

🎯 Learning Objectives

By the end of this lesson, you will be able to:

- Update existing records with UPDATE
- Delete records safely with DELETE
- Understand the critical importance of WHERE clauses
- Use transactions for safety
- Apply database constraints
- Recover from mistakes

⚠️ IMPORTANT WARNING

UPDATE and DELETE are powerful and potentially dangerous!

Without a WHERE clause, these commands affect EVERY row in the table!

- `UPDATE characters SET name = 'Bob'` → All characters named Bob!
- `DELETE FROM characters` → ALL characters deleted!

Always:

1. Write SELECT with WHERE first to test your condition
2. Then change SELECT to UPDATE or DELETE
3. Double-check before executing!

🛠️ Part 1: The UPDATE Statement (8 minutes)

UPDATE modifies existing data in a table.

UPDATE Syntax

```
UPDATE table_name
SET column1 = value1, column2 = value2
WHERE condition;
```

Step 1: Create Your SQL File

1. Create: `lesson7_modifications.sql`
2. Add header:

```
-- Lesson 7: Updating and Deleting Data
-- Student Name: [Your Name]
-- Date: [Today's Date]
--
-- This script demonstrates UPDATE and DELETE operations
--
-- ⚠️  WARNING: Always use WHERE with UPDATE and DELETE!
```

Step 2: Update a Single Record

Let's fix R2-D2's affiliation:

```
-- First, check current data
SELECT id, name, affiliation FROM characters WHERE name = 'R2-D2';

-- Update R2-D2's affiliation
UPDATE characters
SET affiliation = 'Rebel Alliance'
WHERE name = 'R2-D2';

-- Verify the change
SELECT id, name, affiliation FROM characters WHERE name = 'R2-D2';
```

Best Practice: SELECT before and after to verify!

Step 3: Update Multiple Columns

```
-- Update multiple columns at once
UPDATE characters
SET species = 'Human (Cyborg)',
    affiliation = 'Galactic Empire'
WHERE name = 'Darth Vader';

-- Verify
SELECT name, species, affiliation FROM characters WHERE name = 'Darth Vader';
```

Step 4: Update Multiple Records

```
-- Update all droids to a new affiliation
UPDATE characters
SET affiliation = 'No Affiliation'
WHERE species = 'Droid';

-- Check how many were updated
SELECT name, species, affiliation FROM characters WHERE species = 'Droid';
```

Step 5: Update Using Calculations

```
-- Add 5 cm to everyone's height (growth spurt!)
UPDATE characters
SET height = height + 5
WHERE height IS NOT NULL;

-- View updated heights
SELECT name, height FROM characters ORDER BY height;
```

Step 6: Conditional Updates with CASE

```
-- Update affiliations based on species
UPDATE characters
SET affiliation = CASE
    WHEN species = 'Droid' THEN 'No Affiliation'
    WHEN species = 'Wookiee' THEN 'Rebel Alliance'
    WHEN species LIKE '%Jedi%' OR name LIKE '%Obi-Wan%' THEN 'Jedi Order'
    ELSE affiliation
END;

-- View results
SELECT name, species, affiliation FROM characters ORDER BY species;
```

Explanation: CASE allows different values based on conditions, like an if-else statement.

🗑️ Part 2: The DELETE Statement (7 minutes)

DELETE removes rows from a table permanently.

DELETE Syntax

```
DELETE FROM table_name
WHERE condition;
```

⚠️ Critical Warning About DELETE

```
-- DANGEROUS! Deletes EVERYTHING:
DELETE FROM characters;

-- SAFE: Deletes specific row:
DELETE FROM characters WHERE id = 99;
```

Step 7: Delete a Single Record

```
-- First, check which record you're targeting
SELECT * FROM characters WHERE name = 'Test Character';

-- Delete the record
DELETE FROM characters
WHERE name = 'Test Character';

-- Verify it's gone
SELECT COUNT(*) FROM characters;
```

Step 8: Delete with Multiple Conditions

```
-- Let's add a test character first
INSERT INTO characters (name, species, homeworld) VALUES ('Temporary', 'Test', 'Nowhe

-- Verify it exists
SELECT * FROM characters WHERE name = 'Temporary';

-- Delete it
DELETE FROM characters
WHERE name = 'Temporary' AND species = 'Test';

-- Confirm deletion
SELECT * FROM characters WHERE name = 'Temporary';
```

Step 9: Delete Based on JOIN

Sometimes you need to delete based on related table data:

```
-- Delete characters from unknown planets
DELETE FROM characters
WHERE homeworld_id IN (SELECT id FROM planets WHERE name = 'Unknown');

-- This is safer than trying to use JOIN in DELETE
```

Step 10: Safe DELETE Practice

Always follow this process:

```
-- Step 1: SELECT to see what WOULD be deleted
SELECT * FROM characters WHERE species = 'Test Species';

-- Step 2: If results look correct, change SELECT to DELETE
-- DELETE FROM characters WHERE species = 'Test Species';

-- Step 3: Verify deletion
-- SELECT * FROM characters WHERE species = 'Test Species';  -- Should return 0 rows
```

🔒 Part 3: Data Integrity and Constraints (5 minutes)

Constraints ensure data quality and prevent errors.

Common Constraints

Constraint	Purpose	Example
`PRIMARY KEY`	Unique identifier	`id INTEGER PRIMARY KEY`
`NOT NULL`	Must have a value	`name TEXT NOT NULL`
`UNIQUE`	No duplicates allowed	`email TEXT UNIQUE`
`CHECK`	Must meet condition	`CHECK(height > 0)`
`FOREIGN KEY`	Must reference valid record	`FOREIGN KEY (homeworld_id)`
`DEFAULT`	Default value if none provided	`DEFAULT 'Unknown'`

Step 11: Understanding Constraints

```
-- Try to insert a character without a name (should fail with NOT NULL)
INSERT INTO characters (species, homeworld) VALUES ('Human', 'Earth');

-- Try to set height to negative (if CHECK constraint exists)
UPDATE characters SET height = -100 WHERE name = 'Yoda';
```

Expected: These should fail! Constraints protect your data.

Step 12: Foreign Key Constraints

```
-- Try to reference a planet that doesn't exist
UPDATE characters
SET homeworld_id = 9999
WHERE name = 'Luke Skywalker';
```

Expected: Should fail if foreign key constraints are enabled.

Note: SQLite foreign keys are disabled by default. Enable with:

```
PRAGMA foreign_keys = ON;
```

🔄 Part 4: Transactions (Bonus - 2 minutes)

Transactions let you group multiple operations and rollback if something goes wrong.

Transaction Syntax

```
BEGIN TRANSACTION;
    -- Your SQL statements here
    -- If successful:
    COMMIT;
    -- If error:
    -- ROLLBACK;
```

Step 13: Using Transactions

```
-- Start a transaction
BEGIN TRANSACTION;

-- Make changes
UPDATE characters SET affiliation = 'Test' WHERE species = 'Human';
UPDATE characters SET height = height * 2 WHERE species = 'Human';

-- Check the changes
SELECT name, affiliation, height FROM characters WHERE species = 'Human';

-- If happy with changes: COMMIT
-- If not happy: ROLLBACK
-- For this practice, let's undo:
ROLLBACK;

-- Verify rollback worked
SELECT name, affiliation, height FROM characters WHERE species = 'Human';
```

Use Transactions For:

- Multiple related updates
- When testing risky operations
- Ensuring data consistency

🎓 Practice Exercises

Exercise 1: Safe Update

```
-- Exercise 1: Add 10 years to Yoda's experience
-- (We'll pretend height represents experience level for this exercise)

-- Step 1: Check current value
SELECT name, height FROM characters WHERE name = 'Yoda';

-- Step 2: Update
UPDATE characters
SET height = height + 10
WHERE name = 'Yoda';

-- Step 3: Verify
SELECT name, height FROM characters WHERE name = 'Yoda';
```

Exercise 2: Conditional Update

```
-- Exercise 2: Update all characters from Tatooine to be affiliated with 'Desert Nati
UPDATE characters
SET affiliation = 'Desert Natives'
WHERE homeworld_id = (SELECT id FROM planets WHERE name = 'Tatooine');

-- Verify
SELECT c.name, p.name AS homeworld, c.affiliation
FROM characters c
JOIN planets p ON c.homeworld_id = p.id
WHERE p.name = 'Tatooine';
```

Exercise 3: Safe Deletion

```
-- Exercise 3: Add and then delete a test character

-- Add test character
INSERT INTO characters (name, species, homeworld) VALUES ('Test Delete', 'Test', 'Now

-- Verify it exists
SELECT * FROM characters WHERE name = 'Test Delete';

-- Delete it
DELETE FROM characters WHERE name = 'Test Delete';

-- Confirm deletion
SELECT * FROM characters WHERE name = 'Test Delete';
```

🐛 Common Errors & Troubleshooting

Error: "syntax error"

Problem: Missing WHERE, incorrect column name, or missing quotes.

Wrong:

```
UPDATE characters SET name = Luke WHERE id = 1;  -- Missing quotes
```

Correct:

```
UPDATE characters SET name = 'Luke' WHERE id = 1;
```

Accidentally Updated/Deleted Everything

Problem: Forgot WHERE clause!

Wrong:

```
UPDATE characters SET species = 'Unknown';  -- Updates ALL rows!
```

Recovery Options:

1. If using transactions: `ROLLBACK`
2. Restore from backup
3. Re-run your INSERT statements
4. Learn the lesson: Always use WHERE!

Can't Delete Due to Foreign Key

Problem: Trying to delete a record referenced by another table.

Example:

```
DELETE FROM planets WHERE name = 'Tatooine';
-- Fails because characters reference this planet
```

Solutions:

1. Delete referencing records first (characters from Tatooine)
2. Update foreign keys to NULL or different value
3. Use CASCADE delete (advanced)

UPDATE Affects Wrong Rows

Problem: WHERE condition too broad.

Prevention:

```
-- Always test with SELECT first
SELECT * FROM characters WHERE species = 'Human';
-- Check this returns ONLY the rows you want to update

-- Then UPDATE
UPDATE characters SET affiliation = 'Changed' WHERE species = 'Human';
```

NULL Value Comparisons

Problem: Using = NULL instead of IS NULL.

Wrong:

```
SELECT * FROM characters WHERE affiliation = NULL;
```

Correct:

```
SELECT * FROM characters WHERE affiliation IS NULL;
```

✅ Checkpoint: What You've Learnt

Before moving on, make sure you can:

- ✅ Update single and multiple records with UPDATE
- ✅ Update multiple columns in one statement
- ✅ Delete records safely with DELETE
- ✅ Always use WHERE clause (or risk disaster!)
- ✅ Test with SELECT before UPDATE/DELETE
- ✅ Understand database constraints
- ✅ Use transactions for safety

🎯 Challenge Problem (Optional)

Task: Create a series of UPDATE statements that:

1. Change all Rebel Alliance members to "New Republic"
2. Add 5 cm to the height of all characters over 180 cm tall
3. Set affiliation to "Retired" for Obi-Wan Kenobi and Yoda

Requirements:

- Use transactions
- Test with SELECT first
- Verify all changes

Click to reveal the solution

```
BEGIN TRANSACTION;

-- Check current affiliations
SELECT name, affiliation FROM characters WHERE affiliation = 'Rebel Alliance';

-- 1. Update Rebel Alliance to New Republic
UPDATE characters
SET affiliation = 'New Republic'
WHERE affiliation = 'Rebel Alliance';

-- 2. Add height to tall characters
UPDATE characters
SET height = height + 5
WHERE height > 180;

-- 3. Retire the old Jedi
UPDATE characters
SET affiliation = 'Retired'
WHERE name IN ('Obi-Wan Kenobi', 'Yoda');

-- Verify all changes
SELECT name, affiliation, height FROM characters ORDER BY affiliation;

-- If satisfied, commit; otherwise rollback
COMMIT;
-- ROLLBACK;
```

💾 Save Your Work with Git

```
git status
git add lessons/lesson7_modifications.sql database/starwars.db
git commit -m "Completed Lesson 7: UPDATE and DELETE operations with safety practices
git push
```

📖 Key SQL Commands Learnt

Command	Purpose	Example
`UPDATE`	Modify existing records	`UPDATE characters SET height = 180`
`DELETE`	Remove records	`DELETE FROM characters WHERE id = 5`
`SET`	Specify new values	`SET name = 'New Name', species = 'Human'`
`CASE`	Conditional logic	`CASE WHEN species = 'Droid' THEN...`
`BEGIN TRANSACTION`	Start transaction	`BEGIN TRANSACTION;`
`COMMIT`	Save changes	`COMMIT;`
`ROLLBACK`	Undo changes	`ROLLBACK;`
`PRAGMA`	Configure database	`PRAGMA foreign_keys = ON;`

⚠️ Safety Checklist

Before running UPDATE or DELETE:

- [ ] Have I included a WHERE clause?
- [ ] Did I test with SELECT first?
- [ ] Am I sure this targets the right rows?
- [ ] Do I have a backup (or using transactions)?
- [ ] Have I double-checked the conditions?

🎉 Well Done!

You can now safely modify and delete data! In the next lesson, you'll learn about advanced queries using subqueries.

Ready to continue? Move on to `lesson8_instructions.md`

Need Help?

- Always SELECT before DELETE/UPDATE
- Use transactions when testing
- Check row counts before and after
- Ask your instructor
- Don't panic if you make a mistake - it's a learning opportunity!