

Daryl D. Silva

BSCpE-3B2

Chapter 3: Database Design and Modeling

Laboratory Activity 5:

Laboratory Title: Normalization - First Normal Form (1NF)

Chapter No. and Topic: Chapter 3 - Database Design and Modeling

Discussions:

This activity demonstrates how to normalize a table to the First Normal Form (1NF).

Activity Description:

Given a sample non-normalized table, convert it to 1NF by ensuring that all columns contain atomic values.

Objectives:

- Understand how to apply 1NF to a database design.
- Convert a table into 1NF.

Result:

The table is now in 1NF with atomic values for each column.

The screenshot shows the SQL Server Management Studio interface. The left pane displays the 'libraymanagement' schema with tables like books, books_1nf, members, transactions, and unnormalizedbooks. The right pane shows a query window with T-SQL code. The code first creates a non-normalized table 'UnNormalizedBooks' with columns BookID, Title, Authors, and Genre. It then inserts two rows of data: (1, 'Book A', 'Author1, Author2', 'Fiction') and (2, 'Book B', 'Author3', 'Non-Fiction'). Following this, a normalized table 'Books_INF' is created with columns BookID, Title, Author, and Genre. It also inserts the same two rows of data. The bottom pane shows the execution log with successful messages for each query.

```
8
9     Genre VARCHAR(50)
10
11 );
12
13 • INSERT INTO UnNormalizedBooks (BookID, Title, Authors, Genre)
14
15     VALUES
16
17     (1, 'Book A', 'Author1, Author2', 'Fiction'),
18
19     (2, 'Book B', 'Author3', 'Non-Fiction');
20
21 • CREATE TABLE Books_INF (
22
23     BookID INT,
24
25     Title VARCHAR(100),
26
27     Author VARCHAR(100),
28
29     Genre VARCHAR(50)
30
31 );
32
33 • INSERT INTO Books_INF (BookID, Title, Author, Genre)
34
35     VALUES
36
37     (1, 'Book A', 'Author1', 'Fiction'),
38
39     (1, 'Book A', 'Author2', 'Fiction'),
40
41     (2, 'Book B', 'Author3', 'Non-Fiction');
```

#	Time	Action	Message
16	23:10:13	INSERT INTO Members (FirstName, LastName, Email) VALUES ('John', 'Doe', 'john.doe@example.com), ('Jane', 'Smith', 'jane.smith@example.com)	45 row(s) affected Records: 45 Duplicates: 0 Warnings: 0
17	23:11:05	INSERT INTO Transactions (MemberID, BookID, IssueDate, ReturnDate) VALUES (1, 1, '2025-01-10', '2025-01-17), (2, 2, '2025-01-10', '2025-01-17)	50 row(s) affected Records: 50 Duplicates: 0 Warnings: 0
18	23:31:07	CREATE TABLE UnNormalizedBooks (BookID INT, Title VARCHAR(100), Authors VARCHAR(100), Genre VARCHAR(50))	0 row(s) affected
19	23:32:14	INSERT INTO UnNormalizedBooks (BookID, Title, Authors, Genre) VALUES (1, 'Book A', 'Author1, Author2, Fiction'), (2, 'Book B', 'Author3', 'Non-Fiction')	2 row(s) affected Records: 2 Duplicates: 0 Warnings: 0
20	23:32:49	CREATE TABLE Books_INF (BookID INT, Title VARCHAR(100), Author VARCHAR(100), Genre VARCHAR(50))	0 row(s) affected
21	23:33:54	INSERT INTO Books_INF (BookID, Title, Author, Genre) VALUES (1, 'Book A', 'Author1', 'Fiction'), (1, 'Book A', 'Author2', 'Fiction'), (2, 'Book B', 'Author3', 'Non-Fiction')	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0

The screenshot shows the MySQL Workbench interface. In the Navigator pane on the left, under the 'librarymanagement' schema, there are tables like 'books', 'books_1nf', 'members', 'transactions', and 'unnormalizedbooks'. The 'unnormalizedbooks' table is selected. In the Query Editor pane on the right, there is a block of SQL code:

```

8     Genre VARCHAR(50)
9
10    );
11
12
13 • INSERT INTO UnNormalizedBooks (BookID, Title, Authors, Genre)
14
15   VALUES
16
17   (1, 'Book A', 'Author1, Author2', 'Fiction'),
18
19   (2, 'Book B', 'Author3', 'Non-Fiction');
20
21 • CREATE TABLE Books_1NF (
22
23   BookID INT,
24

```

Additional Questions/Discussions:

1. How does 1NF improve data integrity?

Answer: **First Normal Form (1NF)** ensures that each column in a table contains only atomic (indivisible) values and that each row is uniquely identifiable. This improves data integrity by:

- **Eliminating duplicate data:** Each piece of information is stored in only one place, reducing redundancy.
- **Ensuring consistency:** Atomic values prevent issues where a single field contains multiple pieces of data that may become inconsistent.
- **Simplifying queries:** Queries and updates become more efficient because data is organized into individual, well-defined fields.

2. What are atomic values, and why are they important?

Answer: An **atomic value** is a single, indivisible piece of data that cannot be broken down further while retaining its meaning. For example, storing FullName as a single column ("John Doe") is **not atomic** because it contains two pieces of data (first name and last name). Instead, it should be split into FirstName and LastName.

Importance of atomic values:

- **Prevents anomalies:** Atomic values help prevent update, insert, and delete anomalies by ensuring that each field stores only one piece of data.
- **Improves database flexibility:** If data is stored atomically, it can be more easily sorted, searched, and updated without affecting other records.
- **Enhances data retrieval:** Queries become more straightforward and efficient when dealing with atomic values.

Conclusions:

Applying 1NF to a database structure ensures that:

- Each column contains atomic values.
- Each row is uniquely identifiable (with a primary key).
- There are no repeating groups or redundant data.

By normalizing the database to 1NF, we improve data integrity, efficiency, and ease of maintenance, making the system more scalable and reliable.