INNOPOLIS UNIVERSITY

Week 6 Lab

Database Normalization

[S25] Databases Course

Normalization 1NF

INNOPOLIS UNIVERSITY

1NF criteria

- Primary key (no duplicate tuples)
- No repeating groups
- Atomic columns (each cell has a single value)
- Values are of the same domain

1NF Example

emp_id	emp_name	emp_mobile	emp_skills
1	John Tick	9999957773	Python, JavaScript
2	Darth Trader	8888853337	HTML, CSS, JavaScript
3	Rony Shark	7777720008	Java, Linux, C++

The above table has 4 columns:

- All the columns have different names.
- All the columns hold values of the same type like emp_name has all the names, emp_mobile has all the contact numbers, etc.
- The order in which we save data doesn't matter
- But the emp_skills column holds *multiple comma-separated values*, while as per the First Normal form, each column should have a single value.

Hence the above table fails to pass the First Normal form.

So how do we fix the above table? There are two ways to do this:

1. Remove the emp_skills column from the Employee table and keep it in some other table.
2. Or add multiple rows for the employee and each row is linked with one skill.

1NF Example

Create Separate tables for Employee and Employee Skills
So the Employee table will look like this:

emp_id	emp_name	emp_mobile
1	John Tick	9999957773
2	Darth Trader	8888853337
3	Rony Shark	7777720008

emp_id	emp_skill
1	Python
1	JavaScript
2	HTML
2	CSS
2	JavaScript
3	Java
3	Linux
3	C++

Normalization 2NF

- **2NF criteria**
  - Relation in 1NF
  - No partial functional dependencies of non-prime attributes on candidate keys

so we can say: A database table is said to be in 2NF if it is in 1NF and contains only those fields/columns that are functionally dependent(means the value of the field is determined by the value of another field(s)) on the primary key. In 2NF we remove the partial dependencies of any non-key field.

Normalization 2NF Example

INNOPOLIS UNIVERSITY

Student Table

student_id	student_name	branch
1	Akon	CSE
2	Bkon	Mechanical

Subject Table

subject_id	subject_name
1	C Language
2	DSA
3	Operating System

Score Table

student_id	subject_id	marks	teacher_name
1	1	70	Miss. C
1	2	82	Mr. D
2	1	65	Miss. C

Normalization 2NF Example

Now in the **Score** table, the primary key is **student_id + subject_id**, because both these information are required to select any row of data.

But in the **Score** table, we have a column teacher_name, which depends on the subject information or just the subject_id, so we should not keep that information in the Score table.

The column teacher_name should be in the Subjects table. And then the entire system will be Normalized as per the Second Normal Form.

student_id	subject_id	marks
1	1	70
1	2	82
2	3	65

subject_id	subject_name	teacher_name
1	C Language	Miss. C
2	DSA	Mr. D
3	Operating System	Mr. Op

Normalization 3NF

INNOPOLIS UNIVERSITY

3NF criteria:

- Relation in 2NF
- No **transitive functional dependencies** of non-prime attribute on candidate key

So we can say: A table is said to be in the Third Normal Form when, It satisfies the First Normal Form and the Second Normal form. And, it doesn't have Transitive Dependency.

Let’s consider an example: We had the **Score table** in the Second Normal Form. If we have to store some extra information in it, like exam_type and total_marks

The Score table will look like this.

Normalization 3NF Example

INNOPOLIS UNIVERSITY

Score Table

student_id	subject_id	marks	exam_type	total_marks
1	1	70	Theory	100
1	2	82	Theory	100
2	1	42	Practical	50

In the table above, the column exam_type depends on both student_id and subject_id, because a student can be in the CSE branch or the Mechanical branch and based on that they may have different exam types for different subjects. The CSE students may have both Practical and Theory for Compiler Design whereas Mechanical branch students may only have Theory exams for Compiler Design.

But the column total_marks just depends on the exam_type column. And the exam_type column is not a part of the primary key. Because the primary key is student_id + subject_id, hence we have a Transitive dependency here.

How to fix it?

Normalization 3NF Example

exam_type_id	exam_type	total_marks	duration
1	Practical	50	45
2	Theory	100	180
3	Workshop	150	300

We have created a new table and we have added more related information in it like duration(duration of exam in mins.), and now we can use the exam_type_id in the Score table.

Task Instructions

Innopolis University

- Submit the solution to `Lab_06` folder on GitLab.

- You need to submit **four SQL files** as follow:
  - `task1_0NF.sql` for creating and inserting the data.
  - `task1_1NF.sql` for converting the relation into 1NF.
  - `task1_2NF.sql` for converting the relation into 2NF.
  - `task1_3NF.sql` for converting the relation into 3NF.

Normalization Task

Given the following relation, Create the table, fill in the data, convert it to 1NF, 2NF, 3NF

Project Code	Project Name	Project Manager	Project Budget	Employee No.	Employee Name	Department No.	Department Name	Hourly Rate
PC010	Reservation System	Mr. Ajay	120500	100	Mohan	D03	Database	21.00
PC010	Reservation System	Mr. Ajay	120500	101	Vipul	D02	Testing	16.50
PC010	Reservation System	Mr. Ajay	120500	102	Riyaz	D01	IT	22.00
PC011	HR System	Mrs. Charu	500500	105	Pavel	D03	Database	18.50
PC011	HR System	Mrs. Charu	500500	103	Jack	D02	Testing	17.00
PC011	HR System	Mrs. Charu	500500	104	James	D01	IT	23.50
PC012	Attendance System	Mr. Rajesh	710700	315	Raul	D03	Database	21.50
PC012	Attendance System	Mr. Rajesh	710700	218	Alex	D02	Testing	15.50
PC012	Attendance System	Mr. Rajesh	710700	109	Victor	D01	IT	20.50

INNOPOLIS UNIVERSITY

Thank you for attention

See you next week

[S25] Databases Course