BASUDEV GODABARI DEGREE COLLEGE , KESAIBAHAL

Department of Computer Science

"SELF STUDY MODULE"

Module Details :

- Class – 3rd Semester
- Subject Name : COMPUTER SCIENCE
- Paper Name : DATABASE MANAGEMENT SYSTEM

UNIT – 2 : STRUCTURE

2.1 Introduction to Database Design Theory and Normalization
2.2 Functional Dependencies
2.3 Normal Forms based on Primary Keys,
2.4 Second and third Normal Forms,
2.5 Join Dependencies and Fifth Normal Form

Learning Objective

After Learning this unit you should be able to

- Know the Basic Concept of Object Oriented Programming
- Know to write simple JAVA program
- Know how to use constructor in the program.

You Can use the Following Learning Video link related to above topic :

https://youtu.be/T7AxM7Vqvaw
https://youtu.be/vsz8PoQos_0
https://youtu.be/5GDTIUVIHB8
https://youtu.be/Tp37HXfekNo

You Can also use the following Books :

S.NO	Book Title	Author
1	Programming in JAVA	E.Balguru Swamy
2	The Complete Reference to JAVA	HERVERTSEHILDT

And also you can download any book in free by using the following website.

- https://www.pdfdrive.com/

[illegible]

DBMS – Normalization

Database Management System or **DBMS** in short refers to the technology of storing and retrieving usersí data with utmost efficiency along with appropriate security measures. This tutorial explains the basics of DBMS such as its architecture, data models, data schemas, data independence, E-R model, relation model, relational database design, and storage and file structure and much more.

Normalization

If a database design is not perfect, it may contain anomalies, which are like a bad dream for any database administrator. Managing a database with anomalies is next to impossible.

- **Update anomalies** – If data items are scattered and are not linked to each other properly, then it could lead to strange situations. For example, when we try to update one data item having its copies scattered over several places, a few instances get updated properly while a few others are left with old values. Such instances leave the database in an inconsistent state.
- **Deletion anomalies** – We tried to delete a record, but parts of it was left undeleted because of unawareness, the data is also saved somewhere else.
- **Insert anomalies** – We tried to insert data in a record that does not exist at all.

Normalization is a method to remove all these anomalies and bring the database to a consistent state.

First Normal Form

First Normal Form is defined in the definition of relations (tables) itself. This rule defines that all the attributes in a relation must have atomic domains. The values in an atomic domain are indivisible units.

Course	Content
Programming	Java, c++
Web	HTML, PHP, ASP

We re-arrange the relation (table) as below, to convert it to First Normal Form.

Course	Content
Programming	Java
Programming	C++
Web	HTML
Web	PHP
Web	ASP

Each attribute must contain only a single value from its pre-defined domain.

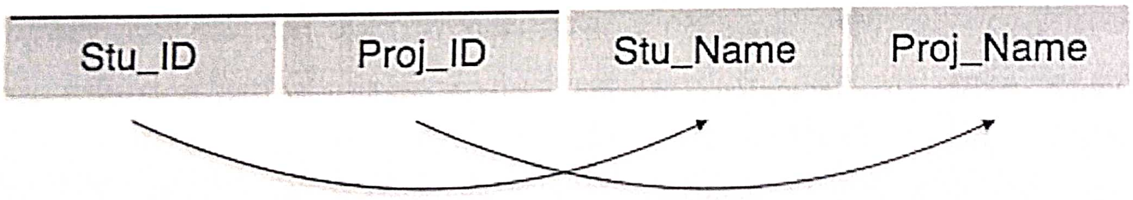
Second Normal Form

Before we learn about the second normal form, we need to understand the following –

- **Prime attribute** – An attribute, which is a part of the candidate-key, is known as a prime attribute.
- **Non-prime attribute** – An attribute, which is not a part of the prime-key, is said to be a non-prime attribute.

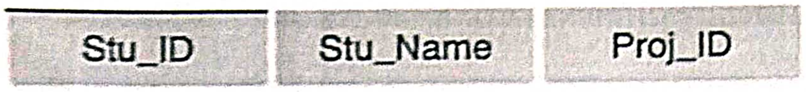
If we follow second normal form, then every non-prime attribute should be fully functionally dependent on prime key attribute. That is, if X → A holds, then there should not be any proper subset Y of X, for which Y → A also holds true.

Student_Project

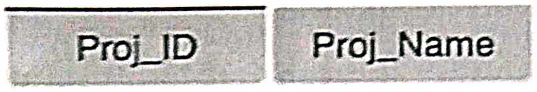


We see here in Student_Project relation that the prime key attributes are Stu_ID and Proj_ID. According to the rule, non-key attributes, i.e. Stu_Name and Proj_Name must be dependent upon both and not on any of the prime key attribute individually. But we find that Stu_Name can be identified by Stu_ID and Proj_Name can be identified by Proj_ID independently. This is called **partial dependency**, which is not allowed in Second Normal Form.

Student



Project



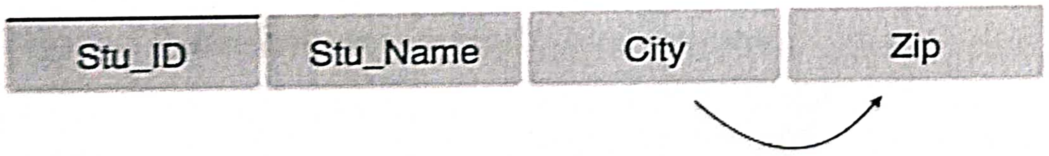
We broke the relation in two as depicted in the above picture. So there exists no partial dependency.

Third Normal Form

For a relation to be in Third Normal Form, it must be in Second Normal form and the following must satisfy –

- No non-prime attribute is transitively dependent on prime key attribute.
- For any non-trivial functional dependency, X → A, then either –
  - X is a superkey or,
  - A is prime attribute.

Student_Detail



We find that in the above Student_detail relation, Stu_ID is the key and only prime key attribute. We find that City can be identified by Stu_ID as well as Zip itself. Neither Zip is a superkey nor is City a prime attribute. Additionally, Stu_ID → Zip → City, so there exists **transitive dependency**.

To bring this relation into third normal form, we break the relation into two relations as follows –

Student_Detail

Stu_ID	Stu_Name	Zip

ZipCodes

Zip	City

Boyce-Codd Normal Form

Boyce-Codd Normal Form (BCNF) is an extension of Third Normal Form on strict terms. BCNF states that –

- For any non-trivial functional dependency, X → A, X must be a super-key.

In the above image, Stu_ID is the super-key in the relation Student_Detail and Zip is the super-key in the relation ZipCodes. So,

Stu_ID → Stu_Name, Zip

and

Zip → City

Which confirms that both the relations are in BCNF

What is Join Dependency?

If a table can be recreated by joining multiple tables and each of this table have a subset of the attributes of the table, then the table is in Join Dependency. It is a generalization of Multivalued Dependency

Join Dependency can be related to 5NF, wherein a relation is in 5NF, only if it is already in 4NF and it cannot be decomposed further.

Example

<Employee>

EmpName	**EmpSkills**	**EmpJob** (Assigned Wor
Tom	Networking	EJ001

Harry	Web Development	EJ002
Katie	Programming	EJ002

The above table can be decomposed into the following three tables; therefore it is not in 5NF:

<EmployeeSkills>

EmpName	**EmpSkills**
Tom	Networking
Harry	Web Development
Katie	Programming

<EmployeeJob>

EmpName	**EmpJob**
Tom	EJ001
Harry	EJ002
Katie	EJ002

<JobSkills>

EmpSkills	**EmpJob**

Networking	EJ001
Web Development	EJ002
Programming	EJ002

Our Join Dependency –

{(EmpName, EmpSkills), (EmpName, EmpJob), (EmpSkills, EmpJob)}

The above relations have join dependency, so they are not in 5NF. That would mean that a join relation of the above three relations is equal to our original relation **<Employee**

Join Dependency

- Join decomposition is a further generalization of Multivalued dependencies.
- If the join of R1 and R2 over C is equal to relation R, then we can say that a join dependency (JD) exists.
- Where R1 and R2 are the decompositions R1(A, B, C) and R2(C, D) of a given relations R (A, B, C, D).
- Alternatively, R1 and R2 are a lossless decomposition of R.
- A JD ⋈ {R1, R2,..., Rn} is said to hold over a relation R if R1, R2,....., Rn is a lossless-join decomposition.
- The *(A, B, C, D), (C, D) will be a JD of R if the join of join's attribute is equal to the relation R.
- Here, *(R1, R2, R3) is used to indicate that relation R1, R2, R3 and so on are a JD of R.

- **QUESTION BANK :-**

1. Define redundancy?
2. Define functional dependency?
3. Explain the problems with Redundancy?
4. What is decomposition? Explain the properties of Decomposition?
5. Discuss normalization? 6. Illustrate functional dependency with example?
7. Illustrate fully functional dependency with example?
8. Demonstrate transitive dependency? Give an example?
9. Define First Normal Form?
10. Define Second Normal Form?
11. Define Third Normal Form?
12. Explain about Loss Less Join Decomposition?
13. Describe Dependency Preserving Decomposition?
14. What is multi valued Dependency?
15. Define Fourth Normal Form?
16. Define Join Dependency?
17. Define BCNF?
18. Explain Fifth Normal Form?
19. Explain about Inclusion Dependency?