Paper Name: Database Management Systems

Topic: Normalization

Paper Code: BCA CC410

By Ms. Manisha Prasad

Head, Assistant Professor,

Department of Computer Science

Patna Women's College

Why Normalization

.

We know that in Relational Model, Data is organized in the form of a Table or Relation. Each row in the table is known as a Record or a Tuple. Each column denotes a Field or an attribute. This type of model designs the data in the form of rows and columns within a table. Thus, a relational model uses tables for representing data and in-between relationships. This model was initially described by Edgar F. Codd, in 1969. The relational data model is the widely used model which is primarily used by commercial data processing applications

Let us take an example of a Relation called STUDENT with following Fields:-

1. Stud-id ----------→ student roll number
2. Stud-name----------→ student name
3. Address----------→ student address
4. Course-code ---------→ course ID in which he has taken admission **for example BCA**
5. Course-name---------→ course name **for example : Bachelor in Computer Applications**
6. Course-HOD----------→ Name of Head of the department of that course **for example : XYZ**
7. Phone ----------→Phone number of the HOD
8. Email----------→ email id of HOD

Stud-id	Stud-name	Address	Course-code	Course-name	Couse-HOD	Phone	Email
S1	Amit	Patna	BCA	Computer Applications	ABC	1234	abc@yahoo.com
S2	Roma	Gaya	BBA	Business Administration	XYZ	4567	xyz@yahoo.com
S3	Seema	Mumbai	BCA	Computer Applications	ABC	1234	abc@yahoo.com
S4	Gunja	Gaya	BCA	Computer Applications	ABC	1234	abc@yahoo.com
S5	Rina	Patna	IMB	Industrial Microbiology	PQR	7890	pqr@yahoo.com
S6	Sumit	Mumbai	IMB	Industrial Microbiology	PQR	7890	pqr@yahoo.com
S7	Ramesh	Delhi	BBA	Business Administration	XYZ	4567	xyz@yahoo.com
S8	Ajita	Delhi	FD	Fashion Designing	LMN	6666	lmn@yahoo.com

In the above table, we see that a considerable amount of redundancy (duplication of data) exists. For example :- due to every admitted student in a particular course, the details of the course i.e. is **Course-code** and **Course-name** and the details of HOD i.e **HOD-name, Phone, Email** is repeated in the table.

Now also consider following situations:-

a. If we want to add details of a new course, say **Bachelor in Mass Communications – BMC** in the above table. Is it possible without having a detail of any student admitted in this course. The answer is **NO**. This is known as **Insertion Anomaly**.

b. Suppose student **S8** leaves the course and we delete the details of this student from the Table, then we also lose the details of the course **FD** as well as the details of the HOD. This is known as **Deletion Anomaly**.

c. If we want to change the phone number of an HOD in the above table, for example the phone no of **Mr ABC**. We have to make sure that this change is made in all the rows (Tuples) where his name is mentioned. If by mistake even at one place it is left out, then we will have inconsistency (wrong data) in the table. This is known as **Update Anomaly**.

Seeing the above problems, the concept of Normalization developed

Normalization is the process of organizing the data in the database. It is used to minimize the redundancy from a relation and is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies. This process divides the larger table into the smaller table and links them using relationship. The basic three Normal forms are :-

1. First Normal Form denoted as 1NF
2. Second Normal Form denoted as 2NF
3. Third Normal Form denoted as 3NF

Normal Form	Description
1NF	A relation is in 1NF if it contains an atomic value.
2NF	A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.
3NF	A relation will be in 3NF if it is in 2NF and no transitive dependency.

First Normal Form (1NF)

A Relation R will be in 1NF if all attributes contain atomic value. It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute. First Normal Form disallows the multi-valued attribute, composite attribute, and their combinations.

Example: Relation STUD_DETAIL is not in 1NF because of multi-valued attribute STUD_PHONE.
In Relational Model, we can denote the schema of Relation **STUD_DETAIL** with following attributes STUD_ID, STUD_NAME, STUD_PHONE and ADDRESS attribute as R (A B C D) Where R denotes STUD_DETAIL Table, A denotes STUD_ID, B denotes STUD_NAME, C denotes STUD_PHONE and D denotes ADDRESS attribute respectively.

STUD_DETAIL **table:**

STUD_ID	STUD_NAME	STUD_PHONE	ADDRESS
S1	Amit	7272826385, 9064738238	Patna
S2	Roma	8574783832 6789543210	Gaya
S3	Seema	7390372389, 8589830302	Mumbai

The decomposition of the STUD_DETAIL table into 1NF has been shown below:

STUD_ID	STUD_NAME	STUD_PHONE	ADDRESS
S1	Amit	7272826385	Patna
S1	Amit	9064738238	Patna
S2	Roma	8574783832	Gaya
S2	Roma	6789543210	Gaya
S3	Seema	7390372389	Mumbai
S3	Seema	8589830302	Mumbai

Second Normal Form (2NF)

A Table is said to be in Second Normal form if it is in First Normal Form and there is no Partial Dependency in it.

What is Partial Dependency ????

In a table if any Non prime attribute is dependent on a part of the Primary key(Prime attributes) and not on the entire key, this is known as Partial dependency.
For example : See the following table **student** with attributes Roll, Course_code, Name, Course_name, Grade.

Table : student

Roll A	Course_code **B**	Name **C**	Grade **D**	Course_name **E**
1	BCA	Amita	A	Computer Applications
1	BBA	Neha	A	Business Administration
2	BCA	Neha	A	Computer Applications
2	BBA	Gita	B	Business Administration

In the above example, **the Candidate key** (Primary Key) is Roll + Course_code together, therefore these two attributes are **Prime attributes**. All the remaining attributes are **Non Prime attributes**.

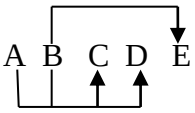
Now if we observe closely, we realize that the attribute **Name** is fully functionally dependent on the key and the attribute **Grade** is also fully dependent on the key. However the attribute **Course_name** is dependent only on the attribute **Course_code.** It has got nothing to do with the attribute **Roll**. So this is an example of Partial dependency. In other words when any non prime attribute is not fully functionally dependent on the key attributes and is only dependent on a part of key attribute, it is known as Partial dependency.

Now in terms of Relational Model, the above **Table student** can be represented as Relation R with five attributes **A denoting Roll, B denoting Course_code, C denoting Name, D denoting Grade, E denoting Course_name as R (A B C D E)** with following two functional dependencies**.**

1. **A B ->C D** because using Roll and Course_code we can find Name and Grade of the students uniquely.
2. **B->E** because using Course_code we can find Course_name.

How to check Partial Dependency ????

Step 1: Draw edge diagram to find candidate key.



Since A and B do not have any incoming edge, they are essential attributes. We see that using **A** and **B** we can find **C** and **D**, and using **B** we can find **E** (as mentioned in the above functional dependencies. Hence we can derive all the attributes of the **relation R** using **A** and **B**. So **AB** is the Candidate key.

Step 2: Now when we closely observe the above mentioned two Functional dependencies, nmwe see that:-
in the First dependency **A B -> C D, the attributes C and D** are fully dependent on key attributes **A and B** because we can get correct information about **Name(C) and Grade(D)** when we give the value of **Roll(A) and Course_code(B).**

However in the second dependency B->E, only Course_code (B) is sufficient to give information about **Course_name (E).** This means that attribute **A** which is a prime attribute is not required along with **B** to find the value of **E**. This is Partial Dependency.

Hence we say that the above table is not in the Second Normal Form.

How to convert the table into Second Normal Form ????

Using the above example R (A B C D E) with following functional dependency:-

a. **A B ->C D**
b. **B->E (Partial Dependency)**

We will decompose the above relation **R** into two relations **R1** and **R2**

R1(A B C D) because **in Relation R1**, **AB** is candidate key and **C** and **D** are fully dependant on **A** and **B**.

R2 (B E) because in **Relation R2, B** becomes key attribute, and **E** is fully dependant on **B**.

Third Normal Form (3NF)

A Table is said to be in Third Normal Form if it is in Second Normal Form and there is no Transitive dependency in it.

What is Transitive Dependency ????
In a table if any non prime attribute is dependent on any other non prime attribute, this is known as Transitive dependency.
For example : See the following table **student** with attributes Roll, Course_code, Name, Course_name, Grade and Status. In this table, we have added one more attribute **Status** denoted by **F**.
Here we have assumed that the student will get only either "a" or "b" grade. He is Pass only if he gets "a" Grade and he is Fail if he gets "b" Grade.

This means that attribute **F** is functionally dependent on attribute **D**. This is an example of Transitive dependency because both **D** and **F** are Non prime attributes. So in this table the functional dependencies are :-

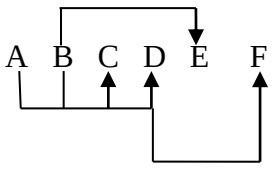
a. **A B ->C D**
b. **B->E**
c. **D->F (Transitive dependency)**

Table : student

Roll A **B**	Course_code **C**	Name **D**	Grade **E**	Course_name F	Status
1	BCA	Amita	**a**	Computer Applications	Pass
1	BBA	Neha	**a**	Business Administration	Pass
2	BCA	Neha	**a**	Computer Applications	Pass
2	BBA	Gita	**b**	Business Administration	Fail

Step 1:

Draw edge diagram to find candidate key.



- We see that using **A** and **B** we can find **C** and **D.**
- Using **B** we can find **E.**
- Since using **A B** we have derived **D** , so using this **D** we can find **F** indirectly (as mentioned in the above functional dependencies.

Hence we can derive all the attributes of the **relation R** using **A** and **B**. So **AB** is the Candidate key.

How to convert the table into Third Normal Form ????

Using the above example R (A B C D E F) with following functional dependency:-

a. **A B ->C D**
b. **B->E**
c. **D->F**

We will decompose the above relation **R** into two relations **R1, R2 and R3**

R1(A B C D) because **in Relation R1**, **AB** is candidate key and **C** and **D** are fully dependant on **A** and **B**.

R2 (B E) because in **Relation R2,** B becomes key attribute, and **E** is fully dependant on **B**.

R3 (D F) because in **Relation R3,** D becomes key attribute, and **F** is fully dependant on **D**.