Normalization

Normalization is the process of organizing the data in the database with two goal:

- To minimize the redundancy from a relation or set of relations.
- To ensure the data dependencies

It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies.

Normalization divides the larger table into the smaller table and links them using relationship.

Benefits of Normalization:

- Quicker updates
- Less storage space
- Clear relationship
- Flexible structure
- Less inconsistency
- Less redundancy

Types of Normalization:

Normal Form	Description
1NF	A relation is in 1NF if every attribute 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 exists.
BCNF	A relation will be in BCNF if it is in 3NF and every determinant set must the super key
4NF	A relation will be in 4NF if it is in Boyce Codd normal form and has no multi-valued dependency.
5NF	A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.

First Normal Form (INF):

- A relation will be 1NF if it contains an 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 EMPLOYEE is not in 1NF because of multi-valued attribute EMP_PHONE.

EMPLOYEE table:

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385, 9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389, 8589830302	Punjab

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

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385	UP
14	John	9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389	Punjab
12	Sam	8589830302	Punjab

Second Normal Form (2NF):

- In the 2NF, relational must be in 1NF.
- In the second normal form, every attributes are fully functional dependent on all prime attribute.

Prime attribute – Member of the candidate key
Non-Prime attribute – Non Member of the candidate key

Example: Let's assume, a school can store the data of teachers and the subjects they teach. In a school, a teacher can teach more than one subject.
TEACHER table

TEACHER_ID	SUBJECT	TEACHER_AGE

25	Chemistry	30
25	Biology	30
47	English	35
83	Math	38
83	Computer	38

In the given table, non-prime attribute TEACHER_AGE is dependent on TEACHER_ID which is a proper subset of a candidate key. That's why it violates the rule for 2NF.

To convert the given table into 2NF, we decompose it into two tables:

TEACHER_DETAIL table:

TEACHER_ID	TEACHER_AGE
25	30
47	35
83	38

TEACHER_SUBJECT table:

TEACHER_ID	SUBJECT
25	Chemistry
25	Biology
47	English
83	Math
83	Computer

Third Normal Form (3NF):

A relation will be in 3NF if it is in 2NF and not contain any transitive dependency.. If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.
3NF states that all column reference in referenced data that are not dependent on the primary key should be removed.

3NF is used to reduce the data duplication. It is also used to achieve the data integrity

Consider the following example:

TABLE_BOOK_DETAIL

Book ID	Genre ID	Genre Type	Price
1	1	Gardening	25.99
2	2	Sports	14.99
3	1	Gardening	10.00
4	3	Travel	12.99
5	2	Sports	17.99

In the table able, [Book ID] determines [Genre ID], and [Genre ID] determines [Genre Type]. Therefore, [Book ID] determines [Genre Type] via [Genre ID] and we have transitive functional dependency, and this structure does not satisfy third normal form.

To bring this table to third normal form, we split the table into two as follows:

TABLE_BOOK

Book ID	Genre ID	Price
1	1	25.99
2	2	14.99
3	1	10.00
4	3	12.99
5	2	17.99

TABLE_GENRE

Genre ID	Genre Type
1	Gardening
2	Sports
3	Travel

Boyce codd Normal Form (BCNF):

- BCNF is the advance version of 3NF. It is stricter than 3NF.
- A table is in BCNF if every functional dependency X → Y, X is the super key of the table.
- For BCNF, the table should be in 3NF, and for every FD, LHS is super key.

Example: Let's assume there is a company where employees work in more than one department.

Below we have a college enrolment table with columns student_id, subject and professor.

student_id	subject	professor
101	Java	P.Java
101	C++	P.Cpp
102	Java	P.Java2
103	C#	P.Chash
104	Java	P.Java

- One student can enrol for multiple subjects. For example, student with student_id 101, has opted for subjects - Java & C++
- For each subject, a professor is assigned to the student.
- And, there can be multiple professors teaching one subject like we have for Java.

In the table above, student_id, subject together form the primary key, because using student_id and subject, we can find all the columns of the table.

One more important point to note here is, one professor teaches only one subject, but one subject may have two different professors. Hence, there is a dependency between subject and professor here, where subject depends on the professor name.

In the table above, student_id, subject form primary key, which means subject column is a prime attribute.
But, there is one more dependency, professor → subject.

And while subject is a prime attribute, professor is a non-prime attribute, which is not allowed by BCNF.

To make this relation(table) satisfy BCNF, we will decompose this table into two tables, student table and professor table.

Student Table

student_id	p_id
101	1
101	2
102	3
103	4
104	1

And, Professor Table

p_id	professor	subject
1	P.Java	Java
2	P.Cpp	C++
3	P.Java2	java
4	P.hash	C#

COMPARISION OF 3NF AND BCNF

BASIS FOR COMPARISON	3NF	BCNF
Concept	No non-prime attribute must be transitively dependent on the Candidate key.	For any trivial dependency in a relation R say X->Y, X should be a super key of relation R.
Dependency	3NF can be obtained without sacrificing all dependencies.	Dependencies may not be preserved in BCNF.
Decomposition	Lossless decomposition can be achieved in 3NF.	Lossless decomposition is hard to achieve in BCNF.

Fourth normal form (4NF)

A relation will be in 4NF if it is in Boyce Codd normal form and has no multi-valued dependency. If the table has two or more independent one – many relationships then the relation is said to be in multivalued dependency. i.e, For a dependency A → B, if for a single value of A, multiple values of B exists, then the relation will be a multi-valued dependency.

Example

STUDENT

STU_ID	COURSE	HOBBY
21	Computer	Dancing
21	Math	Singing
34	Chemistry	Dancing
74	Biology	Cricket
59	Physics	Hockey

The given STUDENT table is in 3NF, but the COURSE and HOBBY are two independent entity. Hence, there is no relationship between COURSE and HOBBY.

In the STUDENT relation, a student with STU_ID, **21** contains two courses, **Computer** and **Math** and two hobbies, **Dancing** and **Singing**. So there is a Multi-valued dependency on STU_ID, which leads to unnecessary repetition of data.
So to make the above table into 4NF, we can decompose it into two tables:

STUDENT_COURSE

STU_ID	COURSE
21	Computer
21	Math
34	Chemistry
74	Biology

59	Physics

STUDENT_HOBBY

STU_ID	HOBBY
21	Dancing
21	Singing
34	Dancing
74	Cricket
59	Hockey

Fifth Normal Form (5NF):

- A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.
- 5NF is satisfied when all the tables are broken into as many tables as possible in order to avoid redundancy.
- 5NF is also known as Project-join normal form (PJ/NF).

Agent	Company	Product_Name
Suneet	ABC	Nut
Raj	ABC	Bolt
Raj	ABC	Nut
Suneet	CDE	Bolt
Suneet	ABC	Bolt

If the above table is decomposed as:

P1

Agent	Company
Suneet	ABC
Suneet	CDE
Raj	ABC

P2

Agent	Product_Name
Suneet	Nut
Suneet	Bolt
Raj	Bolt
Raj	Nut

P3

Company	Product_Name
ABC	Nut
ABC	Bolt
CDE	Bolt

if the natural join of PI and P2 IS taken, the result is:

Agent	Company	Product_Name
Suneet	ABC	Nut
Suneet	ABC	Bolt
Suneet	CDE	Nut*
Suneet	CDE	Bolt
Raj	ABC	Bolt
Raj	ABC	Nut

The spurious row as asterisked. Now, if this result is joined with P3 over the column 'company 'product_name' the following table is obtained:

Agent	Company	Product_name
Suneet	ABC	NUT
Suneet	ABC	BOLT
Suneet	CDE	BOLT
Raj	ABC	BOLT
Raj	ABC	NUT

Hence, in this example, all the redundancies are eliminated, and the decomposition of ACP is a lossless join decomposition. Therefore, the relation is in 5NF as it does not violate the property of lossless join.

Denormalization:

The process of introducing redundancy in the relations is called as denormalization.

Example: Auditing

DKNF:

DKNF is a normal form used in database normalization which requires that the database contains no constraints other than domain constraints and key constraints.

Reason to use DKNF are as follows:

1. To avoid general constraints in the database that are not clear key constraints. Most database can easily test or check key constraints on attributes