**NORMALIZATION**

Normalization is the process of minimizing redundancy from a relation or set of relations. Redundancy in relation may cause insertion, deletion and updation anomalies. So, it helps to minimize the redundancy in relations. Normal forms are used to eliminate or reduce redundancy in database tables.

The normal forms are:

**1. First Normal Form –**

If a relation contain composite or multi-valued attribute, it violates first normal form or a relation is in first normal form if it does not contain any composite or multi-valued attribute. A relation is in first normal form if every attribute in that relation is singled valued attribute.

Example:

ID Name Courses

------------------

1 A c1, c2

2 E c3

3 M C2, c3

1NF:

ID Name Course

------------------

1 A c1

1 A c2

2 E c3

3 M c1

3 M c2

**2. Second Normal Form –**

To be in second normal form, a relation must be in first normal form and relation must not contain any partial dependency. A relation is in 2NF if it has No Partial Dependency, i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

Partial Dependency – If the proper subset of candidate key determines non-prime attribute, it is called partial dependency.

Example-

STUD\_NO COURSE\_NO COURSE\_FEE

1 C1 1000

2 C2 1500

1 C4 2000

4 C3 1000

4 C1 1000

2 C5 2000

2NF:

Table 1 Table 2

STUD\_NO COURSE\_NO COURSE\_NO COURSE\_FEE

1 C1 C1 1000

2 C2 C2 1500

1 C4 C3 1000

4 C3 C4 2000

4 C1 C5 2000

**3. Third Normal Form –**

A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form.

A relation is in 3NF if at least one of the following condition holds in every non-trivial function dependency X –> Y

1. X is a super key.

2. Y is a prime attribute (each element of Y is part of some candidate key).



Transitive dependency – If A->B and B->C are two FDs then A->C is called transitive dependency.

Example - In relation STUDENT given in Table 4,

FD set: {STUD\_NO -> STUD\_NAME, STUD\_NO -> STUD\_STATE, STUD\_STATE -> STUD\_COUNTRY, STUD\_NO -> STUD\_AGE}

Candidate Key: {STUD\_NO}

For this relation in table 4, STUD\_NO -> STUD\_STATE and STUD\_STATE -> STUD\_COUNTRY are true. So STUD\_COUNTRY is transitively dependent on STUD\_NO. It violates the third normal form. To convert it in third normal form, we will decompose the relation STUDENT (STUD\_NO, STUD\_NAME, STUD\_PHONE, STUD\_STATE, STUD\_COUNTRY\_STUD\_AGE) as:

STUDENT (STUD\_NO, STUD\_NAME, STUD\_PHONE, STUD\_STATE, STUD\_AGE)

STATE\_COUNTRY (STATE, COUNTRY)

**4. Boyce-Codd Normal Form (BCNF) –**

A relation R is in BCNF if R is in Third Normal Form and for every FD, LHS is super key. A relation is in BCNF iff in every non-trivial functional dependency X –> Y, X is a super key.

For example consider relation R(A, B, C)

A -> BC,

B -> C

A and B both are super keys so above relation is in BCNF.

**Key Points –**

* BCNF is free from redundancy.
* If a relation is in BCNF, then 3NF is also satisfied.
* If all attributes of relation are prime attribute, then the relation is always in 3NF.
* A relation in a Relational Database is always and at least in 1NF form.
* Every Binary Relation ( a Relation with only 2 attributes ) is always in BCNF.
* If a Relation has only singleton candidate keys( i.e. every candidate key consists of only 1 attribute), then the Relation is always in 2NF( because no Partial functional dependency possible).
* Sometimes going for BCNF form may not preserve functional dependency. In that case go for BCNF only if the lost FD(s) is not required, else normalize till 3NF only.
* There are many more Normal forms that exist after BCNF, like 4NF and more. But in real world database systems it’s generally not required to go beyond BCNF.