# Normalization

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

In database management systems (DBMS), normal forms are a series of guidelines that help to ensure that the design of a database is efficient, organized, and free from data anomalies. There are several levels of normalization, each with its own set of guidelines, known as normal forms.

### Important Points Regarding Normal Forms in DBMS

* **First Normal Form (1NF):** This is the most basic level of normalization. In 1NF, each table cell should contain only a single value, and each column should have a unique name. The first normal form helps to eliminate duplicate data and simplify queries.
* **Second Normal Form (2NF):** 2NF eliminates redundant data by requiring that each non-key attribute be dependent on the primary key. This means that each column should be directly related to the primary key, and not to other columns.
* **Third Normal Form (3NF):** 3NF builds on 2NF by requiring that all non-key attributes are independent of each other. This means that each column should be directly related to the primary key, and not to any other columns in the same table.
* **Boyce-Codd Normal Form (BCNF):** BCNF is a stricter form of 3NF that ensures that each determinant in a table is a candidate key. In other words, BCNF ensures that each non-key attribute is dependent only on the candidate key.
* **Fourth Normal Form (4NF):** 4NF is a further refinement of BCNF that ensures that a table does not contain any multi-valued dependencies.
* **Fifth Normal Form (5NF):** 5NF is the highest level of normalization and involves decomposing a table into smaller tables to remove data redundancy and improve data integrity.

# 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 1 –** Relation STUDENT in table 1 is not in 1NF because of multi-valued attribute STUD\_PHONE.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STUD\_NO | STUD\_NAME | STUD\_PHONE | STUD\_STATE | STUD\_COUNTRY |
| 1 | RAM | 9716271721, | HARYANA | INDIA |
| 2 | RAM | 9716270001, | PUNJAB | INDIA |
| 3 | SURESH | 9716271000, | PUNJAB | INDIA |

# Second Normal Form

A relation is in 2NF if it is in 1NF and anynon-prime attribute (attributes which are not part of any candidate key) is not partially dependent on any proper subset of any candidate key of the table. In other words, we can say that, every non-prime attribute must be fully dependent on each candidate key.

A functional dependency X->Y (where X and Y are set of attributes) is said to be in **partial dependency**, if Y can be determined by any proper subset of X.

However, in 2NF it is possible for a prime attribute to be partially dependent on any candidate key, but every non-prime attribute must be fully dependent (or not partially dependent) on each candidate key of the table

|  |  |  |
| --- | --- | --- |
| STUD\_NO | COURSE\_NO | COURSE\_FEE |
| 1 | C1 | 1000 |
| 2 | C2 | 1500 |
| 1 | C4 | 2000 |
| 4 | C3 | 1000 |
| 4 | C1 | 1000 |
| 2 | C5 | 2000 |

# Boyce-Codd Normal Form (BCNF)

Application of the general definitions of 2NF and 3NF may identify additional redundancy caused by dependencies that violate one or more candidate keys. However, despite these additional constraints, dependencies can still exist that will cause redundancy to be present in 3NF relations. This weakness in 3NF resulted in the presentation of a stronger normal form calledthe**Boyce-Codd Normal Form (Codd, 1974)**.

Although, 3NF is an adequate normal form for relational databases, still, this (3NF) normal form may not remove 100% redundancy because of X−>Y in X is not a candidate key of the given relation. This can be solved by Boyce-Codd Normal Form (BCNF).

Boyce–Codd Normal Form (BCNF) is based on that take into account all candidate keys in a relation; however, BCNF also has additional constraints compared with the general definition of 3NF.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stu\_ID** | **Stu\_Branch** | **Stu\_Course** | **Branch\_Number** | **Stu\_Course\_No** |
| 101 | Computer Science & Engineering | DBMS | B\_001 | 201 |
| 101 | Computer Science & Engineering | Computer Networks | B\_002 | 202 |
| 102 | Electronics & Communication Engineering | VLSI Technology | B\_003 | 401 |
| 102 | Electronics & Communication Engineering | Mobile Communication | B\_004 | 402 |

**Stu\_Branch Table**

|  |  |
| --- | --- |
| **Stu\_ID** | **Stu\_Branch** |
| 101 | Computer Science & Engineering |
| 102 | Electronics & Communication Engineering |

Candidate Key for this table: **Stu\_ID**.

**Stu\_Course Table**

|  |  |  |
| --- | --- | --- |
| **Stu\_Course** | **Branch\_Number** | **Stu\_Course\_No** |
| DBMS | B\_001 | 201 |
| Computer Networks | B\_002 | 202 |
| VLSI Technology | B\_003 | 401 |
| Mobile Communication | B\_004 | 402 |

Candidate Key for this table: **Stu\_Course**.

**Stu\_ID to Stu\_Course\_No Table**

|  |  |
| --- | --- |
| **Branch\_Number** | **Stu\_Course\_No** |
| B\_001 | 201 |
| B\_002 | 202 |
| B\_003 | 401 |
| B\_004 | 402 |

# Fourth Normal Form (4NF)

The Fourth Normal Form (4NF) is a level of database normalization where there are no non-trivial multivalued dependencies other than a candidate key. It builds on the first three normal forms (1NF, 2NF, and 3NF) and the. It states that, in addition to a database meeting the requirements of BCNF, it must not contain more than one multivalued dependency.

**Example:**Consider the database table of a class that has two relations R1 contains student ID(SID) and student name (SNAME) and R2 contains course id(CID) and course name (CNAME).

**Table R1 Table R2**

| **CID** | **CNAME** |
| --- | --- |
| C1 | C |
| C2 | D |

| **SID** | **SNAME** |
| --- | --- |
| S1 | A |
| S2 | B |

When their cross-product is done it resulted in multivalued dependencies.

**Table R1 X R2**

| **SID** | **SNAME** | **CID** | **CNAME** |
| --- | --- | --- | --- |
| S1 | A | C1 | C |
| S1 | A | C2 | D |
| S2 | B | C1 | C |
| S2 | B | C2 | D |

## Fifth Normal Form/Projected Normal Form (5NF)

A relation R is in  if and only if everyone joins dependency in R is implied by the candidate keys of R. A relation decomposed into two relations must have  Property, which ensures that no spurious or extra tuples are generated when relations are reunited through a natural join.

**Example –** Consider the above schema, with a case as “if a company makes a product and an agent is an agent for that company, then he always sells that product for the company”. Under these circumstances, the ACP table is shown as:

**Table ACP**

| **Agent** | **Company** | **Product** |
| --- | --- | --- |
| A1 | PQR | Nut |
| A1 | PQR | Bolt |
| A1 | XYZ | Nut |
| A1 | XYZ | Bolt |
| A2 | PQR | Nut |

The relation ACP is again decomposed into 3 relations. Now, the natural Join of all three relations will be shown as:

**Table R1**

| **Agent** | **Company** |
| --- | --- |
| A1 | PQR |
| A1 | XYZ |
| A2 | PQR |

**Table R2**

| **Agent** | **Product** |
| --- | --- |
| A1 | Nut |
| A1 | Bolt |
| A2 | Nut |

**Table R3**

| **Company** | **Product** |
| --- | --- |
| PQR | Nut |
| PQR | Bolt |
| XYZ | Nut |
| XYZ | Bolt |

The result of the Natural Join of R1 and R3 over ‘Company’ and then of R13 and R2 over ‘Agent’and ‘Product’ will be **Table ACP**.

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