First Normal Form (1st NF)

In 1st NF

- The table cells must be of a single value.
- Eliminate repeating groups in individual tables.
- Create a separate table for each set of related data.
- Identify each set of related data with a primary key.

Definition: An entity is in the first normal form if it contains no repeating groups. In relational terms, a table is in the first normal form if it contains no repeating columns. Repeating columns make your data less flexible, waste disk space, and makes it more difficult to search for data.

IMP: In 1NF relation, the order of tuples (rows) and attributes (columns) does not matter.

Example

Order	Customer	Contact Person	Total
1	Rishabh	Manish	134.23
2	Preeti	Rohan	521.24
3	Rishabh	Manish	1042.42
4	Rishabh	Manish	928.53

The above relation satisfies the properties of relation and is said to be in first normal form (or 1NF). Conceptually it is convenient to have all the information in one relation since it is then likely to be easier to query the database.

Second Normal Form (2nd NF)

In 2nd NF

- Remove Partial Dependencies.
- Functional Dependency: The value of one attribute in a table is determined entirely by the value of another.
- Partial Dependency: A type of functional dependency where an attribute is functionally dependent on only part of the primary key (primary key must be a composite key).
- Create a separate table with the functionally dependent data and the part of the key on which it depends. The tables created at this step will usually contain descriptions of resources.

Definition: A relation is in 2NF if it is in 1NF and every non-key attribute is fully dependent on each candidate key of the relation.

Example

The following relation is not in Second Normal Form:

Order	Customer	Contact Person	Total
1	Rishabh	Manish	134.23
2	Preeti	Rohan	521.24
3	Rishabh	Manish	1042.42
4	Rishabh	Manish	928.53

In the table above, the order number serves as the primary key. Notice that the customer and total amount are dependent upon the order number -- this data is specific to each order. However, the contact person is dependent upon the customer. An alternative way to accomplish this would be to create two tables:

Customer	Contact Person
Rishabh	Manish
Preeti	Rohan

Order	Customer	Total
1	Rishabh	134.23
2	Preeti	521.24
3	Rishabh	1042.42
4	Rishabh	928.53

The creation of two separate tables eliminates the dependency problem. In the first table, contact person is dependent upon the primary key -- customer name. The second table only includes the information unique to each order. Someone interested in the contact person for each order could obtain this information by performing a Join Operation.

Third Normal Form (3rd NF)

In 3rd NF

- Remove transitive dependencies.
- Transitive Dependency A type of functional dependency where an attribute is functionally dependent on an attribute other than the primary key. Thus its value is only indirectly determined by the primary key.
- Create a separate table containing the attribute and the fields that are functionally dependent on it. The tables created at this step will usually contain descriptions of either resources or agents. Keep a copy of the key attribute in the original file.

A relation is in third normal form if it is in 2NF and every non-key attribute of the relation is non-transitively dependent on each candidate key of the relation.

Non-transitive dependency

Let A, B, and C be three attributes of a relation R such that A◊B and B◊C. From these FDs, we may derive A-C. This dependence A-C is transitive.

Example

Company	City	State	ZIP
ABC Ltd.	Mumbai	MH	10169
XYZ Ltd.	Noida	UP	33196
ASD Ltd.	Chennai	TN	21046

The above table is not in the 3NF.

In this example, the city and state are dependent upon the ZIP code. To place this table in 3NF, two separate tables would be created -- one containing the company name and ZIP code and the other containing city, state, ZIP code pairings.

Company	ZIP
ABC Ltd.	10169
XYZ Ltd.	33196
ASD Ltd.	21046

City	State	ZIP
Mumbai	MH	10169
Noida	UP	33196
Chennai	TN	21046

This may seem overly complex for daily applications and indeed it may be. Database designers should always keep in mind the tradeoffs between higher level normal forms and the resource issues that complexity creates.