**2nd NF :**-

2NF allows non-prime attributes to be functionally dependent on non-prime attributes

**3rd NF :-**

1. In 3NF non-prime attributes are only allowed to be functionally dependent on Super key of relation.

2.when a table is in 3NF it is in 2NF and 3NF is stricter than 2NF

Example of 2nf :

Suppose a school wants to store the data of teachers and the subjects they teach. They create a table that looks like this: Since a teacher can teach more than one subjects, the table can have multiple rows for a same teacher.

|  |  |  |
| --- | --- | --- |
| Teacher id | Subject | Teacher age |
| 111 | Maths | 38 |
| 111 | Physics | 38 |
| 222 | Biology | 38 |
| 333 | Physics | 40 |
| 333 | Chemistry | 40 |

**Candidate Keys**: {teacher\_id, subject}  
**Non prime attribute**: teacher\_age

However, it is not in 2NF because non prime attribute teacher\_age is dependent on teacher\_id alone which is a proper subset of candidate key. This violates the rule for 2NF as the rule says “**no** non-prime attribute is dependent on the proper subset of any candidate key of the table”

To make the table complies with 2NF we can break it in two tables like this:-

**teacher\_details table:-**

|  |  |
| --- | --- |
| Teacher\_id | Teacher\_age |
| 111 | 38 |
| 222 | 38 |
| 333 | 40 |

**teacher\_subject table:-**

|  |  |
| --- | --- |
| Teacher\_id | Subject |
| 111 | Maths |
| 111 | Physics |
| 222 | Biology |
| 333 | Physics |
| 333 | Chemistry |

3rd Nf Example :-

1.Table must be in 2NF

2.[Transitive functional dependency](https://beginnersbook.com/2015/04/transitive-dependency-in-dbms/" \t "_blank) of non-prime attribute on any super key should be removed

* X is a [super key](https://beginnersbook.com/2015/04/super-key-in-dbms/) of table
* Y is a prime attribute of table

Suppose a company wants to store the complete address of each employee, they create a table named employee\_details that looks like this:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Emp\_id | Emp\_name | Emp\_Zip | Emp\_state | Emp\_city | Emp\_dist |
| 1001 | John | 282005 | UP | AGRA | Dayal Bagh |
| 1002 | Ajit | 222008 | TN | CHENNAI | M-city |
| 1006 | Lora | 282007 | TN | CHENNAI | Urrapakkam |
| 1101 | Lilly | 292008 | UK | PAURI | Bhagwan |
| 1201 | Steve | 222999 | MP | GWALIOR | ratan |

**Super keys**: {emp\_id}, {emp\_id, emp\_name}, {emp\_id, emp\_name, emp\_zip}…so on  
**Candidate Keys**: {emp\_id}  
**Non-prime attributes**: all attributes except emp\_id are non-prime as they are not part of any candidate keys

Here, emp\_state, emp\_city & emp\_district dependent on emp\_zip. And, emp\_zip is dependent on emp\_id that makes non-prime attributes (emp\_state, emp\_city & emp\_district) transitively dependent on super key (emp\_id). This violates the rule of 3NF.

To make this table complies with 3NF we have to break the table into two tables to remove the transitive dependency:

**employee table:-**

|  |  |  |
| --- | --- | --- |
| Emp\_id | Emp\_name | Emp\_zip |
| 1001 | John | 282005 |
| 1002 | Ajeet | 222008 |
| 1006 | Lora | 282007 |
| 1101 | Lilly | 292008 |
| 1201 | Steve | 222999 |

**employee\_zip table:-**

|  |  |  |  |
| --- | --- | --- | --- |
| **Emp\_zip** | **Emp\_state** | **Emp\_city** | **Emp\_dist** |
| 282005 | **UP** | **AGRA** | **Dalyal Bagh** |
| **222008** | **TN** | **CHENNAI** | **M-city** |
| **282007** | **TN** | **CHENNAI** | **Urrapakkam** |
| **292008** | **UK** | **PAURI** | **Bhagwan** |
| **222999** | **MP** | **GWALIOR** | **Ratan** |