**PRIMARY KEY:**

* A Primary Key in SQL is a field in a table which uniquely identifies each row/record in that table.
* Primary keys must contain unique values, and they cannot contain NULL values.
* Each table can have only one primary key, which can consist of single or multiple columns.

**FOREIGN KEY:**

* A foreign key is a field or collection of fields in one table that uniquely identifies a row of another table.
* In simpler terms, it's a reference in one table that points to the primary key in another table.
* The primary purpose of a foreign key is to maintain data integrity and allow navigation between two different instances of an entity.

**REFERENTIAL INTEGRITY:**

* Referential Integrity is a concept in relational databases that ensures consistency and correctness of data by maintaining proper relationships between tables.
* It is implemented through the use of Foreign Keys and ensures that the relationships between tables remain valid and data integrity is preserved.

**PRIME ATTRIBUTES:**

**Candidate Key**

* A candidate key is a set of one or more columns that can uniquely identify every row in a table.
* Each table can have multiple candidate keys.

**Prime Attributes**

* These are the columns that make up a candidate key. If a column is part of any candidate key, it is considered a prime attribute.
* Columns that are part of any candidate key.

**Non-Prime Attributes**

* Columns that are not part of any candidate key.

**NORMALIZATION:**

Normalization is a process in SQL to structure a relational database in a way that reduces redundancy and dependency by organizing fields and tables.

The main aim is to divide large tables into smaller ones and ensure that they are connected through relationships.

**First Normal Form (1NF):**

* The table should contain only atomic (indivisible) values,i.e., each cell must contain a single value.
* There are no repeating groups or arrays.

***Example:***

Consider a table with students and their subjects (before 1NF):

| Student\_id | Name | Subjects |
| --- | --- | --- |
| 1 | Alice | Maths,Physics |
| 2 | Bob | Maths,Chemistry |

This table is not in 1NF because the Subjects column contains multiple values.

To convert it to 1NF, we split it into multiple rows:

| Student\_id | Name | Subject |
| --- | --- | --- |
| 1 | Alice | Maths |
| 1 | Alice | Physics |
| 2 | Bob | Maths |
| 2 | Bob | Chemistry |

**Second Normal Form (2NF):**

A table should:

* Already be in 1NF.
* Has no partial dependency, ie., non-prime attributes must depend on the entire primary key, not just part of it.

***Example:***

| Student\_id | Subject | Student\_name | Professor |
| --- | --- | --- | --- |
| 1 | Maths | Alice | John |
| 1 | Physics | Alice | Park |
| 2 | Maths | Bob | John |
| 2 | Chemistry | Bob | Kim |

This table is not in 2NF because Student\_name depends only on Student\_ID (a part of the composite key) and not on the entire candidate key (Student\_ID, Subject).

To bring it to 2NF, we decompose the table:

Student Table:

| Student\_id | Student\_name |
| --- | --- |
| 1 | Alice |
| 2 | Bob |

Enrollment Table:

| Student\_id | Subject | Professor |
| --- | --- | --- |
| 1 | Maths | John |
| 1 | Physics | Park |
| 2 | Maths | John |
| 2 | Chemistry | Kim |

**Third Normal Form (3NF):**

A table should:

* Be already in \*\*2NF\*\*.
* Have no transitive dependencies, i.e., no non-prime attribute depends on another non-prime attribute.

***Example:***

| Studet\_id | Subject | Professor | Department |
| --- | --- | --- | --- |
| 1 | Maths | John | Science |
| 1 | Physics | Park | Science |
| 2 | Chemistry | Kim | Science |

This table is not in 3NF because Department depends on Professor (not on the primary key directly).

To bring it to 3NF, we split it:

Student-Subject Table:

| Student\_id | Subject | Professor |
| --- | --- | --- |
| 1 | Maths | John |
| 1 | Physics | Park |
| 2 | Chemistry | Kim |

Professor Table:

| Professor | Department |
| --- | --- |
| John | Science |
| Park | Science |
| Kim | Science |

**DIFFERENCE BETWEEN 2NF AND 3NF:**

| **Aspect** | **2NF** | **3NF** |
| --- | --- | --- |
| Dependency | * Eliminates Partial Dependency. | * Eliminates Transitive Dependency. |
| Main Difference | * Ensures that non-prime attributes depend on the entire primary/candidate key. * Not just a part of the candidate key. | * Ensures that a non-prime attribute is not dependent on any other non-prime attributes. |
| Application | * Mainly applied on tables with candidate keys. | * Applies to any table. |