**Keys in DBMS**

**1. Primary Key**

A **Primary Key** is a unique identifier for each record in a table. It ensures that no two rows have the same value for this key.

* **Unique:** No two rows can have the same value for the primary key.
* **Non-null:** A primary key cannot have NULL values. Every record must have a value assigned to the primary key.
* **Only one per relation:** A table can have only one primary key, which may consist of a single column or a combination of multiple columns (composite key).

✅ **Example:**  
In a **Students** table, Student\_ID can be a primary key as it uniquely identifies each student.

CREATE TABLE Students (

Student\_ID INT PRIMARY KEY,

Name VARCHAR(100),

Age INT

);

**2. Super Key**

A **Super Key** is a set of one or more attributes that uniquely identify a row in a table. However, it may contain **redundant** attributes that are not necessary for uniqueness.

* It is a **superset** of a candidate key.
* Every **candidate key** is a super key, but not every **super key** is a candidate key.
* Super keys can have **additional** attributes that are not needed for uniqueness.

✅ **Example:**  
In the **Students** table, the following are super keys:

* {Student\_ID} (Minimal & unique)
* {Student\_ID, Name} (Unique but not minimal)
* {Student\_ID, Name, Age} (Unique but has redundant attributes)

**3. Candidate Key**

A **Candidate Key** is a **minimal** subset of a super key that uniquely identifies a row in a table.

* **Unique:** No two rows have the same value for this key.
* **Minimal:** Removing any attribute from the candidate key will make it non-unique.
* A table can have **multiple candidate keys**, but one is chosen as the **primary key**.

✅ **Example:**  
Consider a **Students** table with attributes:

* Student\_ID
* Email\_ID

Both Student\_ID and Email\_ID can uniquely identify a student. So, they are candidate keys. However, only one is selected as the **primary key**.

**4. Foreign Key**

A **Foreign Key** is an attribute (or a set of attributes) in a table that **refers to** the **Primary Key** of another table. It is used to establish relationships between tables.

* **Can be NULL:** A foreign key can have NULL values (if the relationship is optional).
* **Can be non-unique:** Multiple rows in a table can have the same foreign key value.

✅ **Example:**  
If we have a **Students** table and a **Courses** table, we can establish a relationship using a foreign key:

CREATE TABLE Students (

Student\_ID INT PRIMARY KEY,

Name VARCHAR(100)

);

CREATE TABLE Courses (

Course\_ID INT PRIMARY KEY,

Student\_ID INT,

FOREIGN KEY (Student\_ID) REFERENCES Students(Student\_ID)

);

Here, Student\_ID in the **Courses** table is a **foreign key** referencing Student\_ID in the **Students** table.

**5. Alternate Key**

An **Alternate Key** is a **candidate key** that is **not chosen as the primary key**.

* Since a table can have multiple **candidate keys**, one of them is selected as the **primary key**, while the remaining ones are alternate keys.

✅ **Example:**  
In a **Students** table,

* {Student\_ID} and {Email\_ID} are both **candidate keys**.
* If Student\_ID is selected as the **primary key**, then Email\_ID becomes an **alternate key**.

CREATE TABLE Students (

Student\_ID INT PRIMARY KEY,

Email\_ID VARCHAR(100) UNIQUE,

Name VARCHAR(100)

);

Here, Email\_ID is an **alternate key** since it is a candidate key but not the primary key.

**Comparison Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type** | **Uniqueness** | **Nullable** | **Multiple per Table?** | **Example** |
| **Primary Key** | Unique | Not NULL | Only one | Student\_ID |
| **Super Key** | Unique | Can be NULL | Many | {Student\_ID, Name, Age} |
| **Candidate Key** | Unique | Not NULL | Many | {Student\_ID}, {Email\_ID} |
| **Foreign Key** | Can be Non-Unique | Can be NULL | Many | Student\_ID in Courses table referencing Students table |
| **Alternate Key** | Unique | Not NULL | Many | Email\_ID |

Would you like me to add more diagrams or provide additional clarifications?