### Normalization of Database

Normalization is a systematic approach of decomposing tables to eliminate data redundancy and undesirable characteristics like Insertion, Update and Deletion Anamolies. It is a two step process that puts data into tabular form by removing duplicated data from the relation tables.

Normalization is used for mainly two purpose,

* Eliminating reduntant(useless) data.
* Ensuring data dependencies make sense i.e data is logically stored.

#### Problem Without Normalization

Without Normalization, it becomes difficult to handle and update the database, without facing data loss. Insertion, Updation and Deletion Anamolies are very frequent if Database is not Normalized. To understand these anomalies let us take an example of **Student** table.

|  |  |  |  |
| --- | --- | --- | --- |
| **S\_id** | **S\_Name** | **S\_Address** | **Subject\_opted** |
| 401 | Adam | Noida | Bio |
| 402 | Alex | Panipat | Maths |
| 403 | Stuart | Jammu | Maths |
| 404 | Adam | Noida | Physics |

* **Updation Anamoly :** To update address of a student who occurs twice or more than twice in a table, we will have to update **S\_Address** column in all the rows, else data will become inconsistent.
* **Insertion Anamoly :** Suppose for a new admission, we have a Student id(S\_id), name and address of a student but if student has not opted for any subjects yet then we have to insert **NULL** there, leading to Insertion Anamoly.
* **Deletion Anamoly :** If (S\_id) 401 has only one subject and temporarily he drops it, when we delete that row, entire student record will be deleted along with it.

#### Normalization Rule

Normalization rule are divided into following normal form.

1. First Normal Form
2. Second Normal Form
3. Third Normal Form
4. BCNF 3.5

#### First Normal Form (1NF)

A row of data cannot contain repeating group of data i.e each column must have a unique value. Each row of data must have a unique identifier i.e **Primary key**. For example consider a table which is not in First normal form

**Student Table :**

|  |  |  |
| --- | --- | --- |
| **S\_id** | **S\_Name** | **subject** |
| 401 | Adam | Biology |
| 401 | Adam | Physics |
| 402 | Alex | Maths |
| 403 | Stuart | Maths |

You can clearly see here that student name **Adam** is used twice in the table and subject **math** is also repeated. This violates the **First Normal form**. To reduce above table to **First Normal form** break the table into two different tables

**New Student Table :**

|  |  |
| --- | --- |
| **S\_id** | **S\_Name** |
| 401 | Adam |
| 402 | Alex |
| 403 | Stuart |

**Subject Table :**

|  |  |  |
| --- | --- | --- |
| **subject\_id** | **student\_id** | **subject** |
| 10 | 401 | Biology |
| 11 | 401 | Physics |
| 12 | 402 | Math |
| 12 | 403 | Math |

In Student table concatenation of subject\_id and student\_id is the **Primary key**. Now both the Student table and Subject table are normalized to first normal form

#### Second Normal Form (2NF)

A table to be normalized to **Second Normal Form** should meet all the needs of **First Normal Form** and there must not be any partial dependency of any column on primary key. It means that for a table that has concatenated primary key, each column in the table that is not part of the primary key must depend upon the entire concatenated key for its existence. If any column depends oly on one part of the concatenated key, then the table fails **Second normal form**. For example, consider a table which is not in Second normal form.

**Customer Table :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **customer\_id** | **Customer\_Name** | **Order\_id** | **Order\_name** | **Sale\_detail** |
| 101 | Adam | 10 | order1 | sale1 |
| 101 | Adam | 11 | order2 | sale2 |
| 102 | Alex | 12 | order3 | sale3 |
| 103 | Stuart | 13 | order4 | sale4 |

In **Customer** table concatenation of Customer\_id and Order\_id is the primary key. This table is in **First Normal form** but not in **Second Normal form** because there are partial dependencies of columns on primary key. Customer\_Name is only dependent on customer\_id, Order\_name is dependent on Order\_id and there is no link between sale\_detail and Customer\_name.

To reduce **Customer** table to **Second Normal form** break the table into following three different tables.

**Customer\_Detail Table :**

|  |  |
| --- | --- |
| **customer\_id** | **Customer\_Name** |
| 101 | Adam |
| 102 | Alex |
| 103 | Stuart |

**Order\_Detail Table :**

|  |  |
| --- | --- |
| **Order\_id** | **Order\_Name** |
| 10 | Order1 |
| 11 | Order2 |
| 12 | Order3 |
| 13 | Order4 |

**Sale\_Detail Table :**

|  |  |  |
| --- | --- | --- |
| **customer\_id** | **Order\_id** | **Sale\_detail** |
| 101 | 10 | sale1 |
| 101 | 11 | sale2 |
| 102 | 12 | sale3 |
| 103 | 13 | sale4 |

Now all these three table comply with **Second Normal form**.

#### Third Normal Form (3NF)

**Third Normal form** applies that every non-prime attribute of table must be dependent on primary key. The *transitive functional dependency* should be removed from the table. The table must be in **Second Normal form**. For example, consider a table with following fields.

**Student\_Detail Table :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student\_id** | **Student\_name** | **DOB** | **Street** | **city** | **State** | **Zip** |

In this table Student\_id is Primary key, but street, city and state depends upon Zip. The dependency between zip and other fields is called **transitive dependency**. Hence to apply **3NF**, we need to move the street, city and state to new table, with **Zip** as primary key.

**New Student\_Detail Table :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student\_id** | **Student\_name** | **DOB** | **Zip** |

**Address Table :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Zip** | **Street** | **city** | **state** |

The advantage of removing transtive dependency is,

* Amount of data duplication is reduced.
* Data integrity achieved.

#### Boyce and Codd Normal Form (BCNF)

**Boyce and Codd Normal Form** is a higher version of the Third Normal form. This form deals with certain type of anamoly that is not handled by 3NF. A 3NF table which does not have multiple overlapping candidate keys is said to be in BCNF.