Normalization organizes data elements in a database into tables.

Normalization avoids

* Duplication of Data – avoids same data being listed multiple times on the same table.
* Insert Anomaly – the data should not be inserted without inserting another entity.
* Update Anomaly – tables cannot be updated without change of information in other places.

## Before Normalization, all the fields are in one big table. Computed fields are not included. Any other additional attribute can be added to the field.

## **Example**:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PROJ\_NUM | PROJ\_NAME | EMP\_NUM | EMP\_NAME | JOB\_CLASS | CHG\_HOUR | HOURS |

The table would be described as follows:

ProjectDetails(proj\_num, proj\_name, emp\_num, emp\_name, job\_class, chg\_hour, hours)

* Multiple items in one attribute for a single tuple.
* No primary keys.

**Normalization: First Normal Form**

* Remove all repeating groups by creating additional row or tuple.
* Choose a primary key.

The big table is divided into two viz., a table for project and employee.

MT\_Project(**pNum**, pName)

MT\_Employee(**eNum**, eName, ejobClass, **pNum,** ehourlyRate, eHours)

MT\_Project Table: A new attribute pNum is created for each project. pNum is used as primary key.

|  |  |
| --- | --- |
| **pNum** | pName |

MT\_Employee Table: A new attribute eNum is created for each employee. eNum and pNum is used as primary key. pNum is the foreign key in employee table referenced from project table to represent that each employee belongs to a project.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **eNum** | eName | ejobClass | **pNum** | ehourlyRate | eHours |

## **Normalization: Second Normal Form**

* Remove partial data dependencies:
* Functional Dependency is the one in which the value of an attribute in a table is determined by another field in the table.
* Partial Dependency is when an attribute is functionally dependent on only part of the primary key
* Primary key must be a composite key.

MT\_Project Table: No changes in the table.

|  |  |
| --- | --- |
| **pNum** | pName |

MT\_Employee Table: eTotalWages is a new attribute calculated by multiplying ehourlyRate and eHours. eTotalWages depend on eHours and ehourlyRate. An employee can work for two projects and hence eNum and pID together (Composite key) are primary keys.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **eNum** | eName | ejobClass | **pNum** | ehourlyRate | eHours | eTotalWages |

## **Normalization: Third Normal Form**

* All non-key fields depend only on primary key.

MT\_Project(**pNum**, pName): pName depends on primary key pNum.

MT\_Employee(**eNum**, eName, ejobClass, eTotalWages): eNum is the primary key on which all other attributes depend on.

MT\_Job(**jID**, pNum, eNum, jhourlyRate, jHours): jID is the primary key on which all other attributes depend on.

The final Table would be as follows;

Project Table

|  |  |
| --- | --- |
| **pNum** | pName |

Employee Table

|  |  |  |  |
| --- | --- | --- | --- |
| **eNum** | eName, | ejobClass | eTotalWages |

Job Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **jID** | pNum | eNum | jhourlyRate | jHours |