**Experiment-1**

**Case study of Normalizations**

**NORMALIZATION:**

Normalization is the process of organizing data into a related table; it also eliminates redundancy and increases the integrity which improves performance of the query. To normalize a database, we divide the database into tables and establish relationships between the tables.

**Normalization Avoids:**

* **Duplication of Data** - The same data is listed in multiple lines of the database
* **Insert Anomaly** - A record about an entity cannot be inserted into the table without first inserting information about another entity - Cannot enter a customer without a sales order
* **Delete Anomaly** - A record cannot be deleted without deleting a record about a related entity. Cannot delete a sales order without deleting all of the customer's information.
* **Update Anomaly** - Cannot update information without changing information in many places. To update customer information, it must be updated for each sales order the customer has placed.

Consider an example to understand normalisation

**ONLINE BUS BOOKING SYSTEM**

**Unnormalised Normal Form**

Booking\_date: userid:

Username: email:

Ticket No: Seatno:

Departuredate: Departuretime:

PassengerId: PassengerName:

PassengerAddress: DOB:

GENDER: Phoneno:

Busno: Busname:

Capacity: type:

Routeno: Route-name:

Source: Destination:

Distance: Fare:

Mode of payment:

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

* The table cells must be of single value.
* Eliminate repeating groups in individual tables.
* Create a separate table for each set of related data.
* Identify each set of related data with a primary key.

**Definition:** An entity is in the first normal form if it contains no repeating groups. In relational terms, a table is in the first normal form if it contains no repeating columns. Repeating columns make your data less flexible, waste disk space, and make it more difficult to search for data.

USER TABLE-(userid->username,u\_email)

|  |  |  |
| --- | --- | --- |
| **Userid** | **Username** | **U\_email** |
|  |  |  |

PASSENGER TABLE-(*Pid,Phno*->pname,paddress,Dob,gender)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Pid** | **phno** | **pname** | **paddress** | **DOB** | **gender** |
|  |  |  |  |  |  |

BUS\_ROUTE(routeno,busno,routename,source,destination,distance,fare,dept\_time,bname,capacity,type,dept\_date)

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| route  no | Bus  no | route  name | sourc  e | desti  nation | dista  nce | fare | Dept  \_time | Dept  \_date | Bus  name | capac  ity | type |
|  |  |  |  |  |  |  |  |  |  |  |  |

RESERVATION TABLE-(seatno,busno->status,bookingdate,ticketno,mode\_of\_payment)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| seatno | busno | status | bookingdate | ticketno | mode\_of\_payment |
|  |  |  |  |  |  |

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

In 2nd NF:

* Remove Partial Dependencies.
* Functional Dependency: The value of one attribute in a table is determined entirely by the value of another.
* Partial Dependency: A type of functional dependency where an attribute is functionally dependent on only part of the primary key (primary key must be a composite key).
* Create separate table with the functionally dependent data and the part of the key on which it depends. Tables created at this step will usually contain descriptions of resources.

Definition: A relation is in 2NF if it is in 1NF and every non-key attribute is fully dependent on each candidate key of the relation.

1)USER

(userid->username,u\_email)

|  |  |  |
| --- | --- | --- |
| **Userid** | **Username** | **U\_email** |
|  |  |  |

2)PASSENGER

(pid-> pname,paddress, DOB, gender,userid)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Pid** | **pname** | **paddress** | **DOB** | **gender** | userid |
|  |  |  |  |  |  |

3)CONTACTS

(pid,phid->phno)

|  |  |  |
| --- | --- | --- |
| **pid** | **phid** | **phno** |
|  |  |  |

4)BUS\_ROUTE(routeno,busno->bname,capacity,type,

source,destination,distance,fare,Dept\_time,dept\_date)

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **route**  **no** | **Bus**  **no** | **route**  **name** | **sourc**  **e** | **desti**  **nation** | **dista**  **nce** | **fare** | **Dept**  **\_time** | **Dept**  **\_date** | **Bus**  **name** | **capac**  **ity** | **type** |
|  |  |  |  |  |  |  |  |  |  |  |  |

5)RESERVATION

(seatno,busno->status,

bookingdate,ticketno, mode\_of\_payment)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **seatno** | **busno** | **status** | **bookingdate** | **ticketno** | **mode\_of\_payment** |
|  |  |  |  |  |  |

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

* Remove transitive dependencies.
* Transitive Dependency A type of functional dependency where an attribute is functionally dependent on an attribute other than the primary key. Thus its value is only indirectly determined by the primary key.
* Create a separate table containing the attribute and the fields that are functionally dependent on it. Tables created at this step will usually contain descriptions of either resources or agents. Keep a copy of the key attribute in the original file.

A relation is in third [normal](http://www.watchesn.com) form, if it is in 2NF and every non-key attribute of the relation is non-transitively dependent on each candidate key of the relation.

**1)USER**

(***userid->***username,u\_email)

2)**PASSENGER**

(***pid***->pname,paddress,DOB, gender,userid)

**3)CONTACTS**

(***pid,phno***->phno)

***(As, busno->routeno and routeno->distance)***

Break **BUS\_ROUTE relation** into 2 tables

**a)BUS**

(***busno***->bname,capacity,type,routeno)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***busno*** | **bname** | **capacity** | **type** | **routeno** |
|  |  |  |  |  |

**b)ROUTE**

(routeno->routename,source,destination,

dept\_date,dept\_time,distance,fare)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **routeno** | **routename** | **source** | **destination** | **dept\_date** | **dept\_time** | **distance** | **fare** |
|  |  |  |  |  |  |  |  |

**Boyce-Codd Normal Form (BCNF)**  
  
In BCNF:

* When a relation has more than one candidate key, anomalies may result even though the relation is in 3NF.
* 3NF does not deal satisfactorily with the case of a relation with overlapping candidate keys
* i.e. composite candidate keys with at least one attribute in common.
* BCNF is based on the concept of a determinant.
* A determinant is any attribute (simple or composite) on which some other attribute is fully functionally dependent.
* A relation is in BCNF is, and only if, every determinant is a candidate key.

**Definition:** A relation is in Boyce-Codd Normal Form (BCNF) if every determinant is a candidate key. (See the links in the box at right for definitions of determinant and candidate key.)

1)BOOKING

(seatno-> pid,busno,status)

|  |  |  |  |
| --- | --- | --- | --- |
| **seatno** | **pid** | **busno** | **status** |
|  |  |  |  |

**4TH Normal Form**

* A table is in 4NF if it is in BCNF and if it has no multi-valued dependencies.

**5TH Normal Form**

* A table is in 5NF, also called "ProjectionJoin Normal Form" (PJNF), if it is in 4NF and if every join dependency in the table is a consequence of the candidate keys of the table.