# Lesson 19

## Objectives

- Case Studies
    - Invoice
    - StaffatBranch
    - ArtGallary
    - IssuedBook

**Invoice**

| OrderNo | OorderDate | CustormerID | CustomerName | CustomerAdd | ProductID | ProductDesc | ProductPrice | ProductQuantity |
|---|---|---|---|---|---|---|---|---|
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi | 1 | Shampoo | 50 | 2 |
| | | | | | 2 | Soap | 40 | 3 |
| | | | | | 3 | Detergent | 110 | 1 |
| | | | | | 4 | Milk | 89 | 2 |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad | 1 | Shampoo | 50 | 2 |
| | | | | | 6 | LemonMax | 33 | 3 |
| | | | | | 3 | Detergent | 110 | 1 |
| | | | | | 5 | Chocolate | 29 | 3 |

We have to normalize the above up to 3NF.

Above relation of Invoice is un-normalized; due to violating the property, atomic value in relation.

This problem of atomicity can be solved by just filling the values as below.

Invoice

| OrderNo | OrderDate | CustormerID | CustomerName | CustomerAdd | ProductID | ProductDesc | ProductPrice | ProductQuantity |
|---|---|---|---|---|---|---|---|---|
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi | 1 | Shampoo | 50 | 2 |
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi | 2 | Soap | 40 | 3 |
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi | 3 | Detergent | 110 | 1 |
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi | 4 | Milk | 89 | 2 |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad | 1 | Shampoo | 50 | 2 |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad | 6 | LemonMax | 33 | 3 |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad | 3 | Detergent | 110 | 1 |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad | 5 | Chocolate | 29 | 3 |

Functional dependencies in the above table is as

orderNo $\longrightarrow$ orderDate
orderNo $\longrightarrow$ customerID
orderNo $\longrightarrow$ customerName
orderNo $\longrightarrow$ customerAdd
orderNo, ProductID $\longrightarrow$ orderQuantity
customerID $\longrightarrow$ customerName
customerID $\longrightarrow$ customerAdress
productID $\longrightarrow$ productPrice
productID $\longrightarrow$ productDesc

There is no single attribute that has an ability to determine values of the remaining attribute. From orderNo we cannot determine the value of productID uniquely. Therefore a composite key is made by combining orderNo and productID. This combination has capability to determine the values of remaining attributes uniquely. In other words we can say that all of the remaining attributes are functionally dependent on orderNo and productID. So the primary key will be ProductID and orderNo.

1NF

| OrderNo | OrderDate | CustormerID | CustomerName | CustomerAdd | ProductID | ProductDesc | ProductPrice | ProductQuantity |
|---|---|---|---|---|---|---|---|---|
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi | 1 | Shampoo | 50 | 2 |
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi | 2 | Soap | 40 | 3 |
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi | 3 | Detergent | 110 | 1 |
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi | 4 | Milk | 89 | 2 |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad | 1 | Shampoo | 50 | 2 |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad | 6 | LemonMax | 33 | 3 |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad | 3 | Detergent | 110 | 1 |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad | 5 | Chocolate | 29 | 3 |

In the above relation there is a composite key (productID&orderNo); orderDate, customerID, customerName, customerAdress can be determined by using only orderNo that is a part of composite key.

In this scenario; orderDate, customerID, customerName, customerAdd are partially dependent on primary key. Therefore we say that there is a partial dependency in relation Invoice.

Above relation is not in 2NF due to having partial dependencies.

Partial dependencies can be resolved by placing all those attributes in separate relation with copy of determinant: orderDate, customerID, customerName, customerAdd will be placed in separate relation with determinant orderNo as

Order

| OrderNo | OrderDate | CustormerID | CustomerName | CustomerAdd |
|---|---|---|---|---|
| 1 | 23/5/2012 | 1 | Ali | Rawalpindi |
| 2 | 23/5/2012 | 2 | Ahmad | Islamabad |

And productPrice and ProductDesc are partially dependent and can be resolved as

Product

| ProductID | ProductDesc | ProductPrice |
|---|---|---|
| 1 | Shampoo | 50 |
| 2 | Soap | 40 |
| 3 | Detergent | 110 |
| 4 | Milk | 89 |
| 6 | LemonMax | 33 |
| 5 | Chocolate | 29 |

Invoice

| OrderNo | ProductID | ProductQuantity |
|---|---|---|
| 1 | 1 | 2 |
| 1 | 2 | 3 |
| 1 | 3 | 1 |
| 1 | 4 | 2 |
| 2 | 1 | 2 |
| 2 | 6 | 3 |
| 2 | 3 | 1 |
| 2 | 5 | 3 |

Given table is decomposed into three tables (Order, Product and Invoice). In all of three there is no partial dependency so all of three relations are in 2NF.

3NF

To check whether the obtained tables are in 3NF or not. To do this we have to check transitive dependencies in all three relations.

In Invoice and Product relation there is no transitive dependency so these two are in 3NF.

There is a transivtive dependency in Order table because customerAdd and customerName can be determined by customerID; so customerAdd and customerName are transitively dependent on OrderNo.

To resolve this problem we have to place customerAdd and customerName in separate relation with their determinant customerID as follow

Customer

| CustormerID | CustomerName | CustomerAdd |
|---|---|---|
| 1 | Ali | Rawalpindi |
| 2 | Ahmad | Islamabad |

Now the resultant relations are:

Order

| OrderNo | OrderDate | CustormerID |
|---|---|---|
| 1 | 23/5/2012 | 1 |
| 2 | 23/5/2012 | 2 |

Product

| ProductID | ProductDesc | ProductPrice |
|---|---|---|
| 1 | Shampoo | 50 |
| 2 | Soap | 40 |
| 3 | Detergent | 110 |
| 4 | Milk | 89 |
| 6 | LemonMax | 33 |
| 5 | Chocolate | 29 |

Invoice

| OrderNo | ProductID | ProductQuantity |
|---|---|---|
| 1 | 1 | 2 |
| 1 | 2 | 3 |
| 1 | 3 | 1 |
| 1 | 4 | 2 |
| 2 | 1 | 2 |
| 2 | 6 | 3 |
| 2 | 3 | 1 |
| 2 | 5 | 3 |

Customer

| CustormerID | CustomerName | CustomerAdd |
|---|---|---|
| 1 | Ali | Rawalpindi |
| 2 | Ahmad | Islamabad |

All of above four (Invoice, order, product and customer) are in 3NF

Hence given relation have been normalized up to 3NF