Second Normal Form

Normalisation

Normalisation is the process used to ensure that the logical design process has produced high-quality structures for the tables within a database. By quality, we mean structures that will perform optimally, and do not contain any duplicate information.

If a logical structure is normalised, then each attribute in the structure will have the same importance. This is useful during the physicalisation process, when the database is actually created using SQL, because indexes can be assigned to each attribute to make searching and retrieval on that attribute faster.

Normalisation is a useful verification tool, which indicates amendments to be made to the logical design process, but does not substitute for the conceptual and logical design. All design stages should be undertaken to ensure that the design covers all aspects of the required system.

The Normalisation Process

The process of normalising a logical model can be summarised as follows:

1. Convert all table structures to First Normal Form
2. Identify Functional Dependencies
3. Convert all table structures from First Normal Form to Second Normal Form
4. Convert all table structures from Second Normal form to Third Normal Form and (very rarely) into Boyce-Codd Normal Form

Functional Dependencies and Determinants

These describe some of the rules that hold between attributes of a system, detailing whether the value of one attribute depends on the value of another, so that if we know the value of the first attribute or group of attributes we can determine the value of the second.

Customer Name	**City**	**Phone**	Product ID	Date	Quantity	Salesperson	%Discount
James Smith	London	07721 121121	23	12/12/2003	50	John Brown	10
James Smith	London	07721 121121	24	15/12/2003	100	John Brown	15
Martin Jones	Manchester	01612249933	23	2/11/2002	50	Bob Jones	10
Alex Haley	London	020845522988	23	15/1/2003	150	John Brown	20

In the above table (key fields in bold), the Salesperson is always associated with a particular City. Similarly, the phone number is dependant on the customer name. These are denoted as functional dependencies:

City → Salesperson
Customer Name → Phone Number

Determinants are the left hand side of the functional dependency process – i.e. the fields whose values 'determine' the values of the other fields. If an attribute A is a **determinant** of attribute B, then B is **functionally dependant** on A.

Second Normal Form

Formally: A relation is in second normal form only if and only if it is in first normal form and every non-key attribute is fully dependant on the primary key.

The following table is in First Normal Form (key fields are in bold):

Customer Name	**City**	**Phone**	Product ID	Date	Quantity	Salesperson	%Discount
James Smith	London	07721 121121	23	12/12/2003	50	John Brown	10
James Smith	London	07721 121121	24	15/12/2003	100	John Brown	15
Martin Jones	Manchester	01612249933	23	2/11/2002	50	Bob Jones	10
Alex Haley	London	020845522988	23	15/1/2003	150	John Brown	20

To identify the problems associated with First Normal Form, we need to consider the operations that we would perform on this table:

1. **Update** – what happens if James Smith's telephone number change and only the first row is updated? – The data for James Smith becomes inconsistent, and the user will not know which number to use to contact him.

2. **Insert** – how do we add data for a salesperson in an area where no clients exist? This cannot be done as we need values for Customer Name, City and Phone to form the primary key of the table.

3. **Delete** – what happens if you delete information about the transaction carried out by Martin Jones on the 2/11/2002? This will also delete information about Martin Jones himself, as well as about the Salesperson associated with the Manchester area.

Partial Dependency

The reason for these problems is that there is a partial dependency on the primary key. This means that one or more of the other NON-KEY fields depends on PART of the key rather than the WHOLE key. In this case:

- The value in the Salesperson field depends only on the City field of the key. This is a partial dependency, as City is only part of the key.
- The value of the Product ID, Date and Quantity depend only on the Customer Name, and not on his City or Phone Number. This is another partial dependency.

The solution to this is problem to implement Second Normal Form by splitting the table in such a way to ensure that none of the fields have a partial dependency on the primary key. This may involve splitting the table into two or more tables, as follows:

SALESPEOPLE

City	Salesperson
London	John Brown
Manchester	Bob Jones

CUSTOMERS

Customer Name	City	Phone
James Smith	London	07721 121121
Martin Jones	Manchester	01612249933
Alex Hayley	London	020845522988

ORDERS

Customer Name	**Product ID**	**Date**	Quantity	%Discount
James Smith	23	12/12/2003	50	10
James Smith	24	15/12/2003	100	15
Martin Jones	23	2/11/2002	50	10
Alex Hayley	23	15/1/2003	150	20

Key fields are in bold.

Reapplying the tests above:

1. Update – we can now update James Smith's telephone number without worrying about a duplicate entry being missed.
2. Insert – we can now add a new salesperson in a new city, as the key field here is City.
3. Delete – deleting information about the purchase made by Martin Jones does not involve loss of information about Martin Jones himself.

Importantly, the NON-KEY fields are now dependant on the WHOLE key field in each table, so that if the key field changes, the values in the non-key field should also change.