Unit 33 Data Analysis and Design

Aulayna MacLeod

Table of Contents

P1.1 The Relational Data Model Compared With the Hierarchical Model & the Network Model.

The relational database consists of tables. The data within the database is stored within these tables. As long as the data within these tables are not in parent child relationship, the data within these tables can be modified or deleted without affecting the data in the other tables. A table is made of columns and rows. The column holds an attribute (such as SName) while the row contains the data for a single entity (for example a member of staff's Staff No, SName, Position, Sex, DOB, Salary, BranchNo).

The hierarchical database model uses a patent child relationship. Each parent table can have many child tables but the child table can only have one table. This results in the hierarchical model only supporting one-to-many relationships. An example of where the hierarchical model is used is within the file structure of the Windows operating system. The network database model is like the hierarchical model with improvements. It allows the child tables to have more than one parent thus allowing one-to-many relationships and many-to-many relationships like the relational database model however the relational database model also supports ad hoc relationships in addition to what the network data model can support.

The network data model is organised as graph structures, while the hierarchical data model is organised like a tree graph and the relational data model is organised with tables.

Figure 1 Relation Data Model [2]

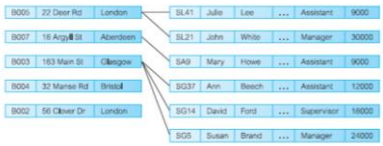


Figure 2 Network Data Model [2]

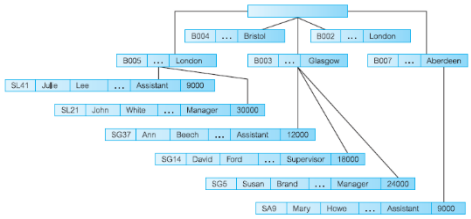


Figure 3 Hierarchical Data Model [2]

Both hierarchical and network database models require that the user has to travel through the structure to access the specific record whereas the relational database model allows the user to be able to find a single entity without having to search through the whole database first. This makes the relational database model more efficient than the other two data models. [1] The relational model also has no restrictions how many times a table can be linked to a child or a parent table as it is not restricted by the hierarchical structure. The relational database is more flexible than both the network and hierarchical models as these predefine the relations which can be used. The relational model on the other hand is based on the idea that users of the database cannot think of all of the uses for the data before the database is created. This allows users to relate records as needed instead of predefining the relations in advance like the hierarchical and network models.

P1.3 Top Down Vs Bottom Up

Top down – find all the sub parts of the problem, then find the sub parts of the sub parts. This is suited for complex database designs as the problem is broken down into smaller parts. If these parts are still too complicated then they are broken down further. This helps make a complicated problem easier to manage as it is no longer one big task but instead many smaller tasks which are more manageable and not as daunting. This means it work s from the general to the specific. The database analyst will ask

the users what the general idea of the database is before asking what data is needed to be stored within the database. In some occasions the top down approach can lead to poor results if the analyst and end user miss something important that is needed for the system. [2]

Bottom up – working up from a specific part of a problem before finding the general problem. This identifies the elements and groups them together. This is a better approach for smaller projects as it is easier to identify fewer specific parts of a smaller project rather than identifying all of the specific parts for a larger project as it would be easy to miss some of these.

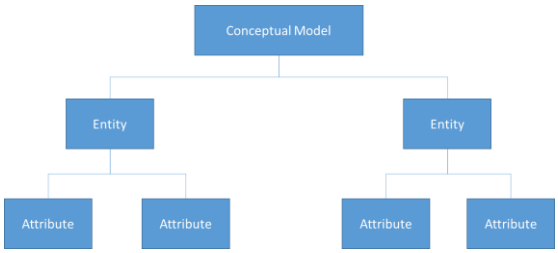


Figure 5 Top Down Approach

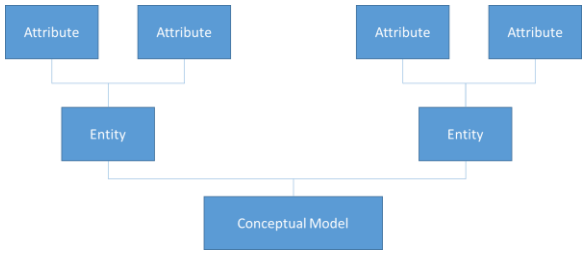


Figure 4 Bottom Up Approach

An example of the bottom up approach is someone who wants to buy a new computer. The person will tell the sales assistant what specific tasks the computer needs to be able to do, for example the customer may want the computer to be able to play games in the highest graphics with no lag and to have a high sound quality so they hear their music in the best quality possible. They may also want their computer to be able to load things quickly while not getting overheated due to working more than the hardware can handle. The sales assistant would then have a think and show the user a number of options which meets the requirements outlined by the customer.

On the other hand a customer may just walk into the shop and ask the sales assistant for a computer. The sales assistant would then use the top down approach to find out what the customer wants the computer for and will be able to suggest features and specifications which will suit the needs of the customer.

P2.1 & M2 Diagrams and Normalisation

0NF

Full Name	Address	Landline & Mobile	Email	Category	Business

1NF

tblCustomer

ID	Firstname	Surname	HouseNumber	Street	Town/City	Postcode	Email	Business	Sale	Date

tblPhone

cuID*	Type	Number

tblCategory

Business	Category	Sale Length	Date Start

2NF

tblCustomer

cuID	Firstname	Surname	HouseNumber	Street	Town/City	Postcode	Email

tblPhone

cuID*	Type	Number

tblCustomerCategoty

cuID*	Category *

tblSaleInfo

Business*
Category*
Sale Length
Date Start

tblBusiness

Business

tblCategory

Category

3NF

tblCustomer

cuID	FirstName	Surname	HouseNumber	Street	Postcode*	Email

tblPhone

cuID*	Type	Number

tblPostcode

Town/City	Postcode

tblBusiness

Business

tblSaleInfo

Business*
Category*
Sale Length
Date Start

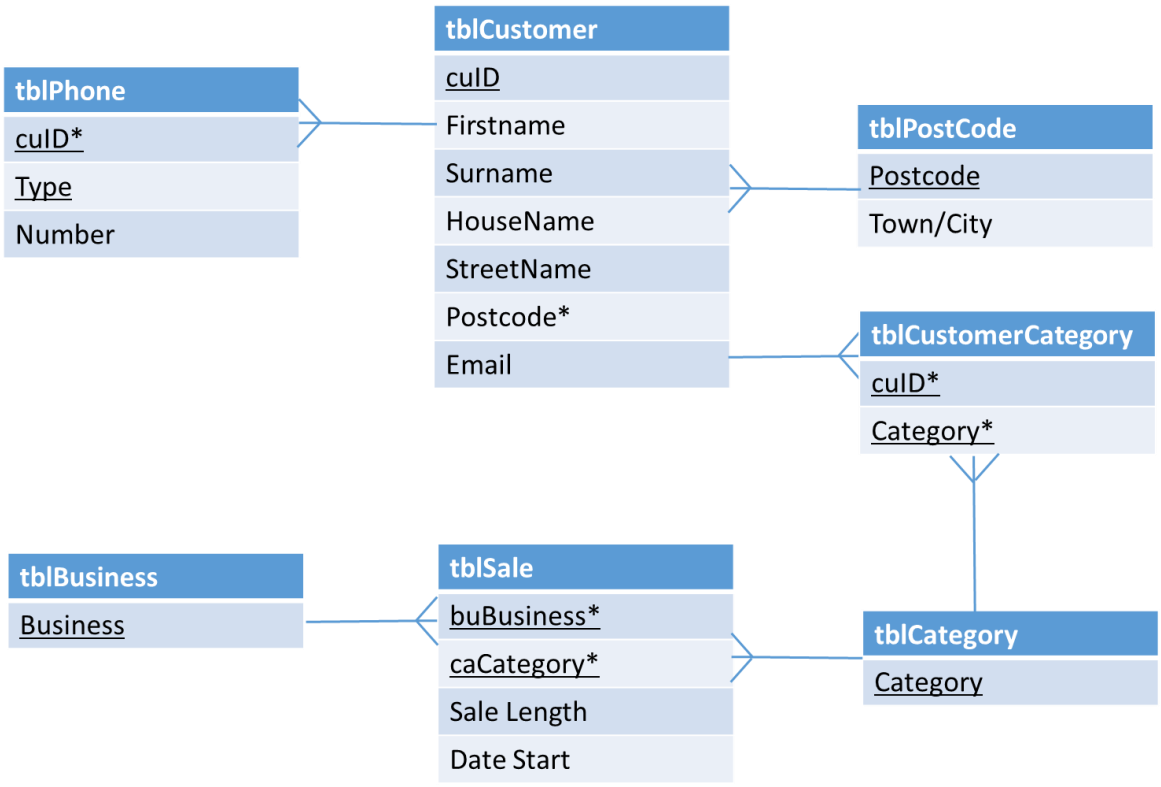
tblCategory

Category

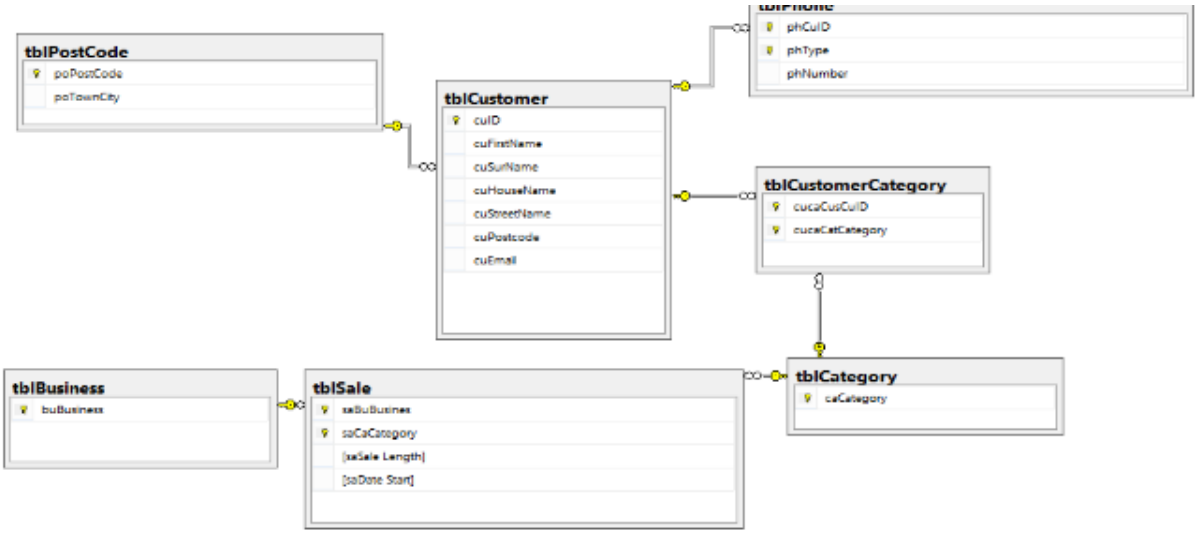
tblCustomerCategoty

cuID*	Category *

Entity Relationship Diagram



Physical Data Model



M1 Justification of Tables

There are seven tables within this database. Multiple tables are needed such as the tblCustomer table to store different types of data (all directly related to the customer) while other tables are required to break many-to-many relationships and obey the rules of normalisation. An example of a table which is needed so that the normalisation rules are not being broken is the tblPhone as the database is required to be able to store the two different types of phone number. To have these phone numbers stored within the tblCustomer table would be incorrect as this would result in repeated attributes (the landline and mobile phone numbers) which would then result in one of the rules needed to meet the criteria of the first normalisation stage being broken.

The tblPostcode table is required as a town/city is based upon the postcode and is not directly related to the customer information and needs its own table.

The tblBusiness is needed so that new businesses can be added to the database easily. As the business does not relate directly to the customer the businesses need their own table.

The tblCategory needed so that new categories can be added to the database easily. As the category which a business belongs to does not relate directly to the customer the categories need their own table.

The tblSale is required to store the details which relate to the sale event of the company. It also acts as an intermediate table between the tblCategory and tblBusiness as a many-to-many relationship is nor permitted to exist within a relational database.

tblCustomerCategory is used to provide a link between the customer and the category table. Without this link the database would be unable to like the categories which the customer has chosen to follow with the customer without breaking the rules of normalisation. In addition to this is acts as an intermediate table between the tblCustomer and the tblCategory tables as mentioned above many-to-many relationships are not allowed in a relation database.

P4.2 Technical Documentation

Function of Each Table

tblCustomer used to store the data which directly relates to the customer. This includes the first name, surname, house name, street name, postcode, email and what category the customer has chosen to follow. The foreign keys in this table is cuPostcode. The primary key of this table is cuID.

tblPhone used to store the phone number related to each customer as well as the type of phone number which is used. The foreign key for this table is coCuID. The primary key is phCuID & phType.

tblPostcode is used to store the postcode and the town/city the postcode belongs to. The primary key of this table is coPostCode. There is no foreign key.

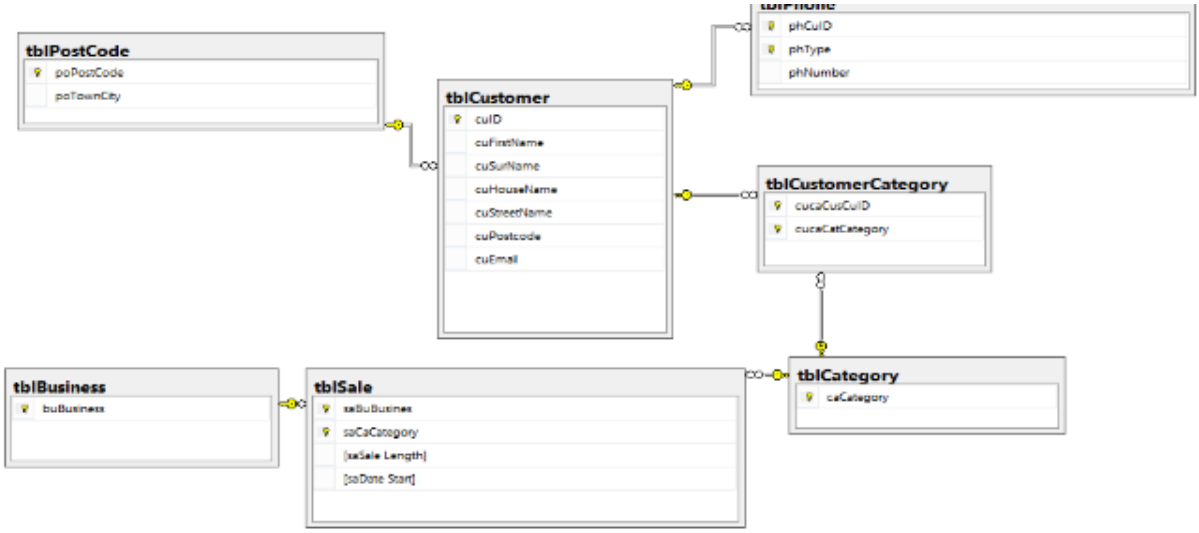
tblCategory stores the different categories which each business can belong to. The primary key is caCategory. There is no foreign key.

tblBusiness stores the different businesses. The primary key of this table is buBusiness. There is no foreign key.

tblSale is used to store the information related to the sale. It also acts as an intermediate table between tblCategory and tblBusiness. The primary key of this table is suBuBusiness & saCaCategory. The foreign keys are suBuBusiness & saCaCategory.

tblCustomerCategory acts as an intermediate table between the tblCustomer and the tblCategory tables. The primary keys for this table are cucaCuID and cucaCaCategory. The forgein keys are also cucaCuID and cucaCaCategory.

Relationships



This diagram shows the relationships between each table.

The tblPostcode table is linked to the tblCustomer table via coPostcode. The cuPostcode column in the tblCustomer table is a foreign key from the tblPostcode table.

The tblPhone tables is linked to the tblCustomer table via the cuID which is the primary key of the tblCustomer table. The phCuID is a foreign key from the tblCustomer table but also acts as part of the primary key with the phType column in the tblPhone table.

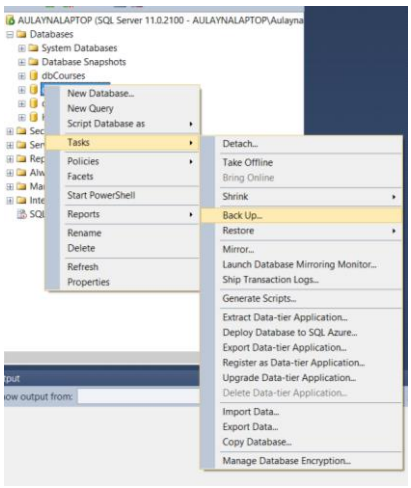
tblCategory is linked to the the tblCustomerCateoory table as well as the tblSale table. This allows the two be linked together. The cucaCuCategory column in the tblCustomerCategory table is a foreign key which is connected to the caCategoty column in the tblCategory table.

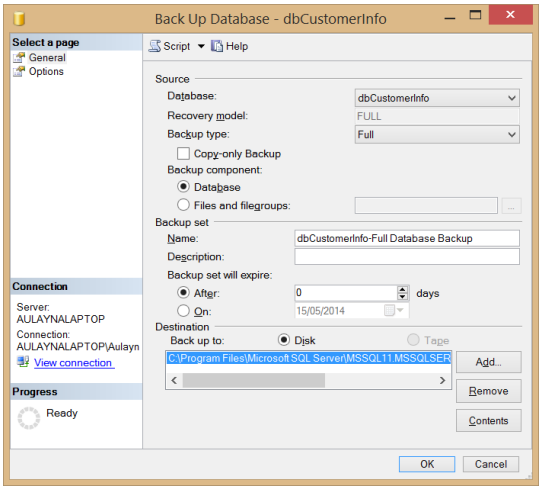
The caCategory column is also linked to the saCaCategory column within the tblSale table as part of the primary key in the tblSale table. The buBusiness column is also linked to the tblSale table as the other part of the primary key. Even though these two columns make up the primary key they are both foreign keys in this table.

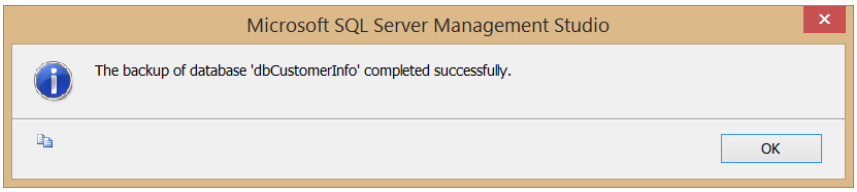
Back Up

The database should be backed up regularly in case something happens to the database or the drive which it is stored on. These backups should be done weekly and stored away from the database (preferably in another building) to keep them safe.

Backups can be done both manually and can be automated. To manually backup the database right click on the database to be backed up, select Tasks and click Back Up... Ensure that what is needed to be selected is selected and after this has been done choose the file path to save the back up to and press ok.







This message should appear after the backup has been completed successfully.

References

[1] Andrew J. Oppel (2009) *Databases: a beginner's guide*, San Francisco, CA: McGraw-Hill Companies.

[2] (2014) *Top-down vs. Bottom-Up Object Database Design,* Available at: *http://www.dba-oracle.com/t_object_top_down_bottom_up.htm* (Accessed: 13/04/2014).