2 Be able to design, create and populate a relational database

2.1 Design

As with all creative activity, every database needs to be designed on the basis of a full analysis of the requirements. In addition, databases are always part of wider systems and as such, database design activity should be seen and be part of a systems analysis process.

In practical terms, a database consists of a number of the following objects:

- **Tables** – to store raw data. Even small commercial databases may need dozens of related tables.
- **Queries** – often described as virtual tables or views. They are designed to represent a specific view of the data as needed for a particular user or requirement. Queries can be developed from single tables or a series of related tables.
- **Data entry forms** – a mechanism by which data can be entered into the tables and often utilise a series of validation and verification techniques to ensure that the entered data is as accurate as possible.
- **Reports** – the printed outputs.

Design documentation

Although the requirements for documentation may vary between companies and projects, the core set of elements will be entity relationship diagrams (ERDs) and data dictionaries. In addition, you may have to provide what data entry screens and output screens and reports are needed with outline designs or report layouts.

When providing design documentation, data flow diagrams (DFDs) are also used frequently to show how processes and data stores work together. They, together with ways of documenting the processes themselves including flow charts, decision tables and structured English, are comprehensively covered in Unit 11 Business systems analysis.

Data dictionaries

In large or complex systems, it can be very difficult to keep track of all of the different tables and data items. If different tables are constructed over time or by different people then inconsistent naming conventions can cause errors and confusion. A data dictionary is a central store of information about all of the data in a database. Expectations by particular employers and methodologies are likely to vary but below are a core set of detail that will always be necessary.

- Table names and descriptions.
- Relationships between tables (ie entities) – often shown using ERDs.
- Field names (attributes) and the table(s) in which they appear.
- Field definitions including field types and lengths and meanings if not self evident.
- Additional properties for each field including format and validation controls.
- Aliases or alternative names for the same field as used in different tables.

2.2 Creating relationships

Normalisation

The theoretical normalisation process has been described in section 1.4. In this section the theory is translated into practical activity.

Activity: Converting to first normal form

1 Make appropriate changes to the Order table shown in Table 18.14 to turn it into first normal form. Create new table(s) as needed.

2 Make appropriate changes to the Supplier table shown in Table 18.15 on the following page to turn it into first normal form. Create new table(s) as needed.

Order reference
Customer reference
Customer telephone
Products and quantities ordered
Date of order
Estimated date of delivery
Number of days to delivery date

Table 18.14: An un-normalised Order table

Activity: Converting to first normal form continued...

Supplier reference
Supplier name
Supplier address
Supplier telephone
Product name
Qty in stock
Price
Price + VAT

Table 18.15: An un-normalised Supplier table

Activity: Converting to second normal form

Look at the list of fields in the Appointments table shown in Table 18.16. Make appropriate changes to turn it into second normal form. The primary key is a composite primary key (shown in bold). Create new table(s) as needed.

Patient ref
Date of appointment
Patient name
Patient address
Patient phone
Doctor name

Table 18.16: Appointments table

Activity: Dealing with complexity in table design

What would be the impact on your answer to the last activity if you considered that people often have two phone numbers?

Third normal form

To be in third normal form, the table must already be in first and second normal forms.

Activity: Converting to third normal form

Look at the list of field names in Table 18.17. Make appropriate changes to the table to turn it into third normal form. The Holiday type field states whether it is a beach holiday, cruise, etc. Do not assume that it is already in 1NF or 2NF.

Holiday reference
Destination
Holiday type
Hotel reference
Hotel name
Hotel address

Table 18.17: Holiday table

Remember that every time you regroup a set of field names and create additional (often smaller) tables, you have to start again and test for first normal form, then second and then third. Once you are sure that the design of the tables is in first, second and third normal forms, you can start to actually create the tables.

It is also important to note that the design of the database must be finalised before you begin to implement it. Later changes made to the underpinning design, structure of tables or field properties could mean that you may have to start again from scratch.

An ideal table:

- has a field (or pair of fields) that uniquely identifies each row (the primary key)
- does not contain unnecessary duplicate fields
- has no repetition of the same type of value
- has no fields that belong in other tables.

An ideal field:

- represents a characteristic of a table subject
- contains a single value
- is atomic (ie not multi-part)
- is not calculated
- is unique throughout the database structure
- has an appropriate name.

Activity: Normal form questions

1 Summarise what is meant by first normal, second normal and third normal forms.

2 How would you check that a table is fully normalised?

Modifying databases

Modifying relationships can be done once a database has been constructed, but this should only be undertaken with care. Once information has been added to tables and a variety of other forms, queries and reports have been designed and created, changing one of the relationships can cause significant problems. It may be necessary to first delete a relationship and then create all of them again.

Modifying a table design has the greatest potential for causing additional work as all of the relationships, queries and reports that depend upon the table may need revisiting.

Cascading updates and deletes

In Microsoft® Access®, with a relationship in which referential integrity is enforced, you can specify whether you want to automatically cascade update or cascade delete related records. If you set these options, delete and update operations that would normally be prevented by referential integrity rules are allowed. When you delete records or change primary key values in a primary table, Access® will make the necessary changes to related tables to preserve referential integrity.

If you select the Cascade Update Related Fields check box when you define a relationship, each time the primary key in a primary table is changed, Microsoft® Access® automatically updates the primary key to the new value in all related records. For example, if you change the Supplier reference field in a table called Suppliers, the Supplier reference field in the Products table will update automatically for every one of the products supplied by that supplier – this ensures that the relationship is not broken. (See Figure 18.6.)

If you select the Cascade Delete Related Records check box when you define a relationship, any time that you delete records in one table, Access® will automatically delete related records in the related table. For example, if you delete a Supplier record from the Suppliers table, all the products produced by that Supplier would be deleted in the Products table.

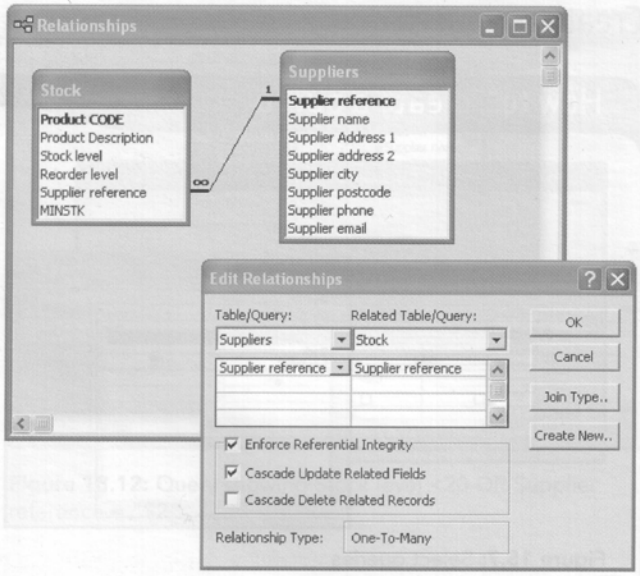

Figure 18.6: Defining the referential integrity rules between two tables

As shown in Figure 18.6, the relationship between the two tables using the Supplier reference field is defined as one in which referential integrity is enforced. The system will 'cascade update' related fields but not 'cascade delete' related records.

In practical terms, this means that if the Supplier reference is changed from, say, S2 to S002 in the Supplier table, then all S2 supplier references of the records in the Stock table will automatically change to S002. So this would be a useful setting to make.

Not checking the cascade delete option means that if the Supplier table record for S2 was deleted because the company went out of business, the related records in the Stock table would not be deleted. This is also a useful setting.