University of Bristol Information Services

Designing your database

Practical workbook

Aims and Learning Objectives

By the end of this document, you will be able to:

- explain the three types of logical relationship that can be identified in database design and point out the only relationship that should be used in logical design;
- explain how to resolve many-to-many relationships;
- explain where primary and foreign keys must be located in any logical design;
- capture data requirements from paper forms or spreadsheets;
- draw simple data structure models with correctly-placed primary and foreign keys.

Document information

Course files

This document and any associated practice files (if needed) are available on the web. To find these, go to www.bristol.ac.uk/it-services/learning/resources and in the Keyword box, type the document code given in brackets at the top of this page.

Related documentation

Other related documents are available from the web at:

http://www.bristol.ac.uk/it-services/learning/resources

Introduction

This document provides a framework for planning and designing a simple database. Once you have taken on board the few simple rules introduced here, you will find that these rules apply to the logical design of any relational database, whatever its size. However, you will have to apply and practice your new skills.

Why do I need to make plans before I start to build my database?

Too many databases fail to do what was planned because simple questions were not asked and simple tasks were not done beforehand. Poor planning results in time-wasting throughout development and often leads to an unusable database. Yes, parts might work, but it could take longer to do the job using the database than it did to do it manually - and the database might provide less accurate results to boot.

Prerequisites

A clear understanding of what you want to get out of your database once it is completed. If you know what outputs you want, then you are more than half way toward knowing what inputs you need: and if you know what inputs you need, then you should have little trouble working out what tables your database needs. At which point, this document will fill in the gaps to help you design your database correctly.

Contents

Task 1 Understanding relationships

Objectives To understand the three main types of relationship that can be identified when designing database tables; to know how to resolve a many-to-many relationship.

Comments Databases are made up of tables (also called entities) and each of these tables is related to one or more of the other tables in the database.

Database tables

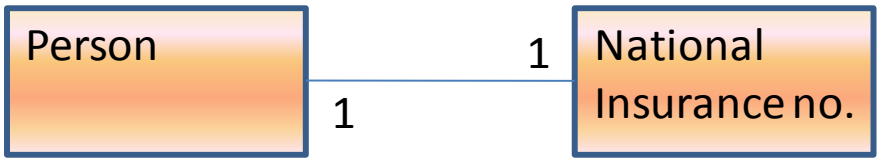
Before trying to design your database, you should know how to go about designing the database on paper first. The most common model is called a logical data structure (LDS) and is also known as an entity-relationship model (ER) because it shows the entities (tables) and the relationships between them.

In a database, each table should hold data about one specific item. For example, a Person table would hold data about people but would not hold data about jobs, because data about jobs would be held in a Job table. The two tables would then be linked by a relationship so that you can tell which person has which job (or which people have which jobs).

Once we have worked out what tables the database needs to hold we need to work out what the relationships between these tables are.

Types of relationship

One-to-one relationship



This is where there is only one occurrence of something for each individual occurrence of the related item, e.g. each person has only one National Insurance number.

One-to-one relationships, generally speaking, are not very useful and the one-to-one related tables can generally be merged into one table, e.g. in the above example, there is no need for both a Person table and a National Insurance number table – the person's National Insurance number can easily be held in the Person table.

(In the logical design of a database there is no place for one-to-one relationships. However, when it comes to building the database, there may be reasons for putting some of the data fields (i.e. those not often used) into a separate table and linking with the same primary key in both tables. But my advice is don't worry about this unless you are needing to build a massive corporate database – in which case you probably won't be reading this cheap and cheerful guide to relational design issues!)