

Extending the scope of the Student Record System: A brief look at The Entity Relation Model.

Aims and Objectives

Students should be able to:

- Describe the basic elements of the Entity Relation model
- Use the Entity Relational model to design a simple relational database

To date we have a Student Record System which will record students details and car details. This is shown in figure 8.1.

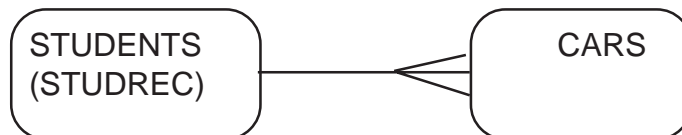


Figure 8.1 Students can own many cars

Figure 8.1 needs some explanation. The terms used are common in the field of Systems Analysis and in particular SSADM these topics are properly covered in detail in Systems Analysis course.

Systems Analysis:

The work done in studying a problem to decide how best it can be handled by a computer. It can cover a wide area, e.g. the way a company works, how data moves from one part of the company to another, the design of forms and reports etc.

Longman Illustrated Dictionary of Computer Science - p 191.

SSADM

Structured Systems Analysis & Design Method. Launched in 1981 as a bid to standardise IT projects across government departments.



Entity



Figure 8.2: Entity

This rounded box represents an entity. An entity is a 'thing' about which our database system holds information. It has the potential to occur more than once and has a number of data items associated with it to describe each occurrence.

e.g.. Cars can occur many times in the CARS table. They are identified by REGNO and have other data items associated with them (COLOUR, MAKE etc.)

Relations



A relationship is a logical connection between two entities, e.g. a student can own a car, or even two or three cars. This is known as a one to many relation and is depicted by a "crows foot" symbol

The student/cars relationship may be depicted as shown in Figure 8.3:

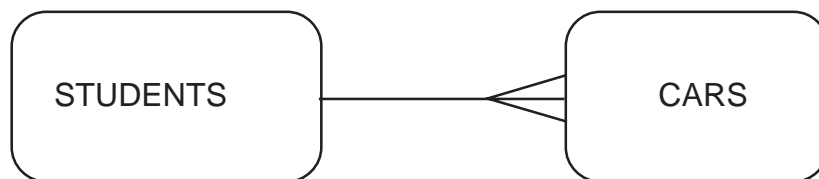


Figure 8.3: Logical Data Structure of Students and Cars

This is known as a Logical Data Structure, it models the relationships between entities.



Extending the Database to Include Course Details.

Assume that we are asked to extend the database to include details of the courses taught and which students are on which course. Whilst there are a number of accepted techniques involving Structured Systems Analysis and Design Method (SSADM), these are properly the subject area of the Systems Analysis Module (second semester), therefore we shall adopt a looser, practical approach for the purpose of this exercise.

The third element in our model is an entity called courses. What possible attributes can a course have? Some attributes are listed below:-

- (i) Course Title
- (ii) Length of Course
- (iii) Accreditation
- (iv) Subject Area

Note: We are not including course tutors at this stage as they are a separate entity.

The design is shown in Figure 8.4.



Notes

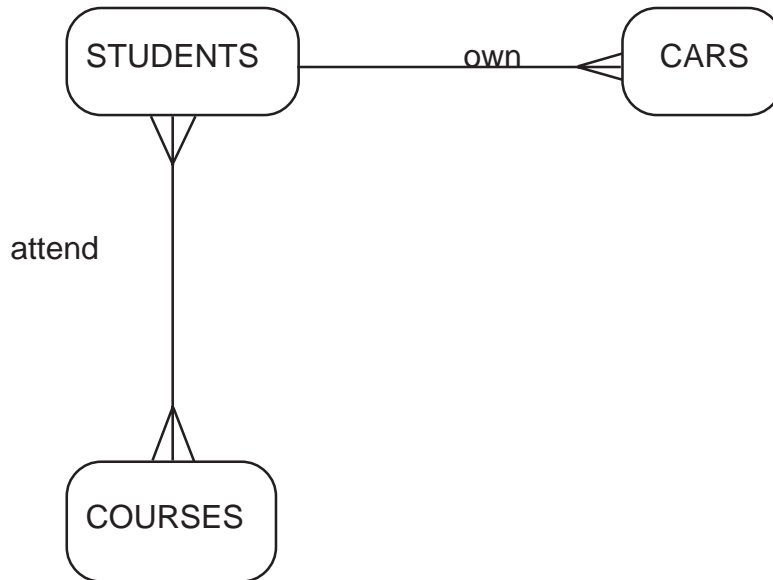


Figure 8.4: Entity Model for Students, Cars & Classes (ver. 1)

This is showing us that:

- (i) A student may own 1 or more cars.
- (ii) Many Students may attend many courses.

The many to many relationship between students and courses will cause problems (the data will not be in 3NF). Usually a design which includes a many to many relation is hiding a further entity. We may have a file design as shown below:

STUDREC

SREF	INIT	SNAME	DOB	GENDER	RES	KIDS	HTOWN
------	------	-------	-----	--------	-----	------	-------

COURSES

CREF	CTITLE	CLEN	CACCRED	CSUBJECT
------	--------	------	---------	----------

Figure 8.5: Initial Table Design for Classes & Students



The problem with the design in figure 8.5 is that we don't know which students are on which course. This gives us a clue as to the missing entity. The entity should allow us to link courses to students and is known as a "Link-Entity"; the entity will give details about individual course bookings, we will call it "CLASS LIST". The entity relations are now as shown below:

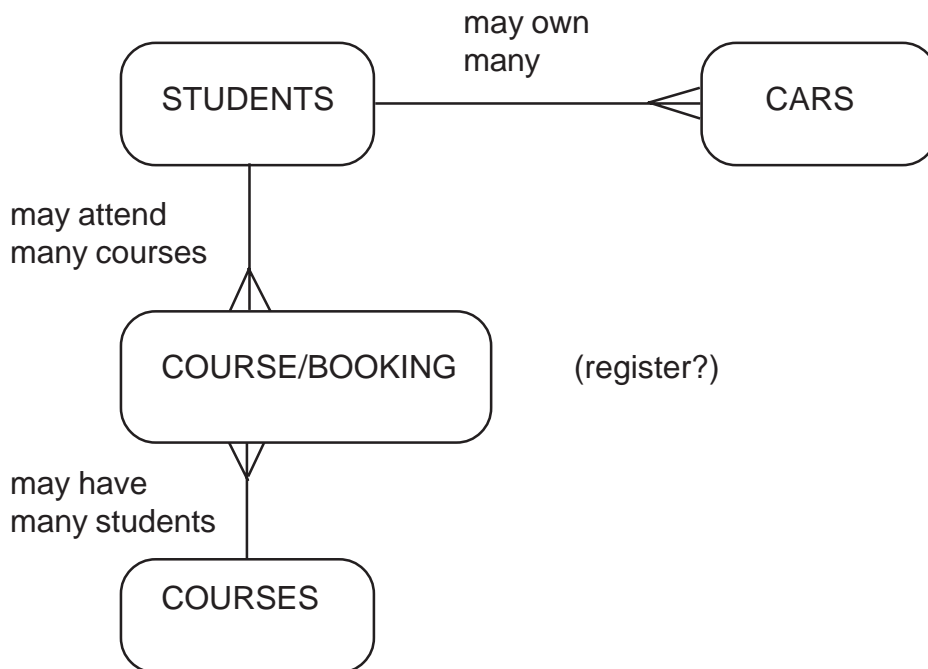


Figure 8.6: A Working design for Classes and Students

If we redesign our database using figure 8.6 as our starting point we arrive at the design shown in figure 8.7 (we'll omit CARS for the sake of clarity).



Notes

STUDREC

SREF	INIT	SNAME	DOB	GENDER	RES	KIDS	HTOWN
------	------	-------	-----	--------	-----	------	-------

CLASS LIST

SREF	CREF
------	------

CLASSES

CREF	CTITLE	CLEN	CACCRED	CSUBJECT
------	--------	------	---------	----------

Figure 8.7: A working physical table design for Students and Classes

We know from our previous work on the tables in the physical design outlined in figure 8.7, that if we establish the correct relations between them we can obtain a set of data that identifies all classes and which students are taking which classes.

Figure 8.8 shows how we could envisage the DBMS processing each of the records in CLASS LIST and cross reference them with CLASSES and STUDREC

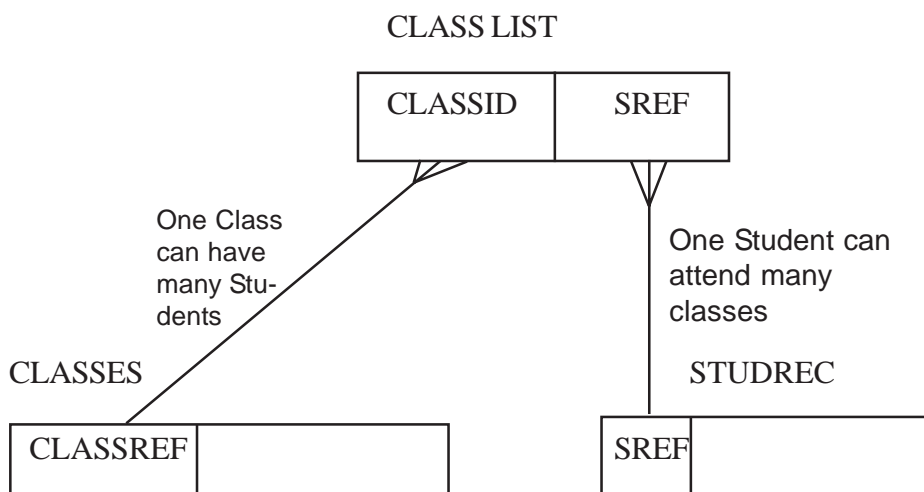


Figure 8.8: Physical design for Students, Classes and Class Lists



Exercise 8: Entity Relationships

This exercise is purely a theoretical exercise. You won't have to go near a computer to complete it.

However, as an optional extra, you might want to see if you can implement the design you come up with as your answer to exercise 8.

Amend the student record system as shown in Figure 8.6 to allow us to store details of the course tutors. Assume the following data is required:-

- (i) Tutor's Name
- (ii) Tutor's specialist subject
- (iii) Tutor's highest Qualification
- (iv) University where Qual. gained
- (v) Date Qual. gained

You should provide as your answer:-

- (i) An amended version of the entity relationship diagram (Figure 8.6). Assume that team teaching is common, i.e. more than one tutor may teach on a course. One tutor may teach on many courses. Your design should resolve any many to many relations that may crop up.
- (ii) Show how you would relate the tables so that you can obtain a list of all tutors and the courses they teach on. Your answer should be in the form given in figure 8.8.
- (iii) Provide a brief written account of how you arrived at your design (guide length 300 - 400 words).

