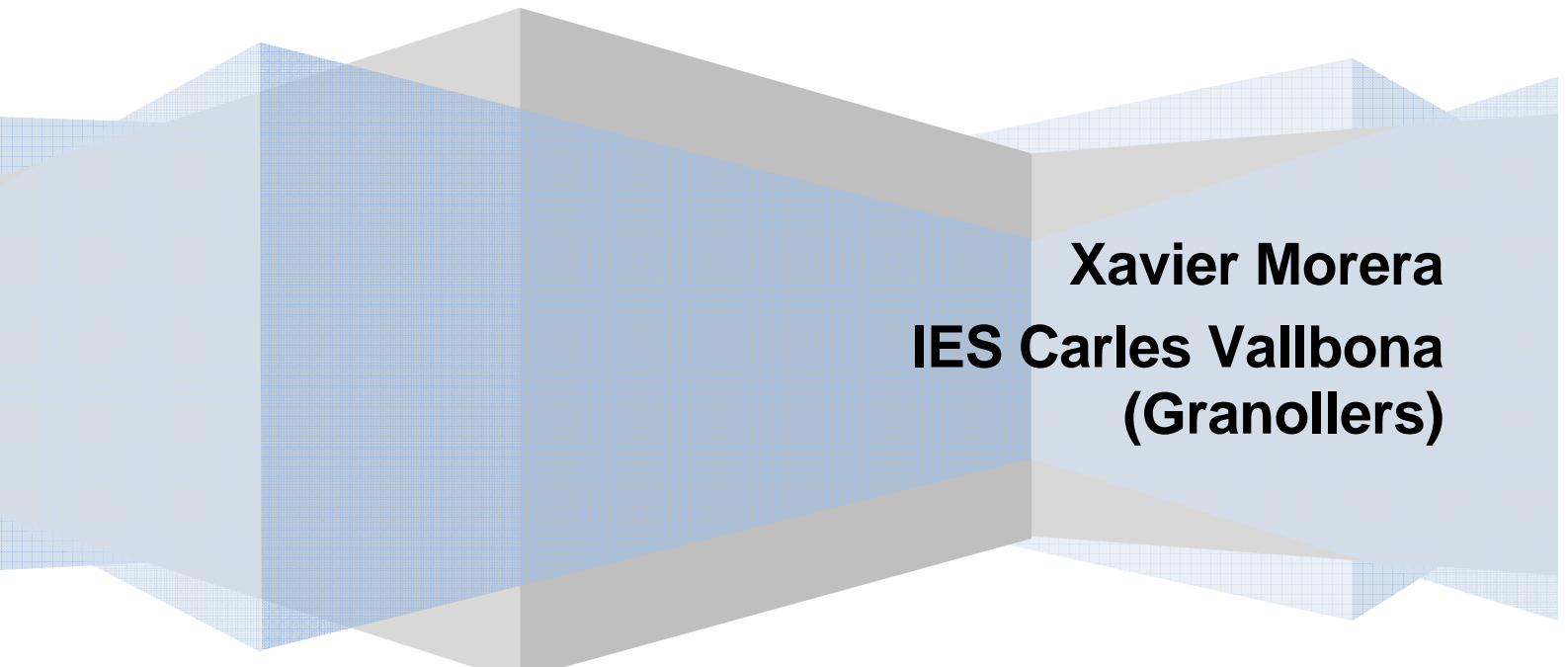


Llicència Retribuïda C – Teacher's Guide

Using English to Teach Database Design in IT Vocational Training

Teacher's Guide



Xavier Morera
IES Carles Vallbona
(Granollers)

Index

INTRODUCTION

ACTIVITIES

PART I.- DATABASE DESIGN

Lesson 1.- Database Management Systems

Lesson 2.- Relational Model Fundamentals

Lesson 3.- Entity-Relationship Model

Lesson 4.- Transformation into Tables from ER Diagrams.....

PART II.- THE SQL LANGUAGE

Lesson 1.- SQL Basics

Lesson 2.- Data Definition Language.....

Lesson 3.- Data Manipulation Language

Introduction

1. What These Teaching Materials Are About

1.1 Who Are These Materials For?

These materials provide an introductory treatment of the principles of relational databases. They are intended for students taking the 'crèdit'¹ 7 (*Relational Databases Management Systems*) of the CFGS² of "Desenvolupament d'Aplicacions Informàtiques" (*Computer Programs' Development*) as well as for those doing the 'crèdit' 6 (*Relational Database Management Systems*) of the CFGS of "Administració de Sistemes Informàtics" (*Computer Systems Administration*), in which the students are expected to develop competence in designing practical databases systems.

1.2. Speciality, Area and Education Level

As it has just been said, these materials have been designed to teach in **higher vocational training**³ within the **Information Technology speciality**.

In particular, the covered themes are concern to the **database area**.

1.3 What Are These Materials Made Up Of?

The materials consist of three elements:

- A set of **POWER POINT PRESENTATIONS** to be used by the teacher during the lessons as visual support material to explain all the concepts.

¹ Crèdit = subject.

² "Cicle Formatiu de Grau Superior" = Higher Vocational Training.

³ CFGS

These slides should also be printed out and used by the students as a reference manual.

- The **TEACHER'S GUIDE** with guidelines on how to use the materials. Here there's information about every unit such as the timing and necessary resources, objectives to be achieved and the exercises to do at any moment with their corresponding answers or suggestions to carry out them.
- The **STUDENT'S BOOK** where they will find the exercises and a bilingual English-Spanish glossary.

2. How These Materials Should be used

2.1 Power Point Presentations

To teach each lesson you should use the corresponding power point presentation, which are the linchpin of the teacher's materials.

It not only will give you a **visual support** to explain the concepts more easily, but also will guide you by telling you which activity to do at any moment; as you move forward you'll find some yellow slides with the number and name of the activity to be carried out. Besides, as the concepts are discussed the slides raise different questions about them.

On the other hand, these slides should also be printed out and handed in to the students for them to use as a **reference manual**.

2.1 Teacher's Guide

The guide is structured around lessons which one starting with an overview of the main contents.

Each lesson includes all the **activities** suggested in the power point presentation as well as their **answers** on how to complete them. Besides, fore each activity, in the teacher's guide there is a short explanation about its **objective** and a **suggested approach** that might be adopted to carry it out.

Sometimes there's not an only possible answer, so in these cases a suggested answer or some ideas about how to solve it are given. To make it easier, all

the activities are sorted in the same order that they appear in the power point presentation.

Finally, all lessons also include their corresponding **lesson plans** which gives information, among other things, about the learning objectives to be achieved, the contents to be explained, the time that should be spent and the language outcomes for each unit.

2.3 Student's Book

The student's book follows a similar structure to the teacher's guide and it's also divided into lessons, each one containing its corresponding activities.

The activities are, without a doubt, the basis of the Student's Book, but the difference with the Teacher's guide is that the approach and objectives for each activity and their answers, obviously, are not included here. Neither the lessons plans.

On the other hand, the student's book does include some extra material as a reference for the students such as a **bilingual English-Spanish glossary** for each lesson and a summary that recaps the most significant points that this lesson looks at.

3. Contents Summary

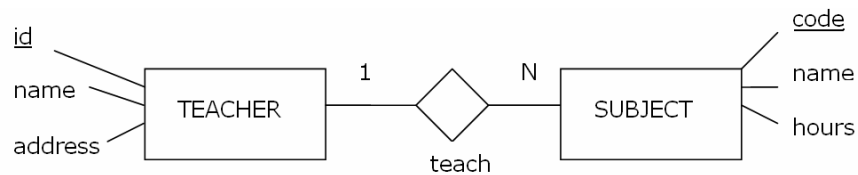
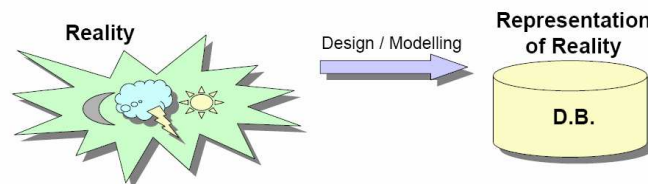
3.1 Overview

Nowadays, almost everybody needs to keep some kind of information (an address book, information about the staff of a company, information about the books in a library, etc). When using a computer we would need to store this data in a database where the information is organized in one or more tables (rows and columns). So the questions are: How many tables do we need? Which columns are needed for each table?

The database design gives an answer to these questions; it gives a methodology to obtain the necessary tables from a real situation about which we want to keep information. The most amazing thing is that, as I've said before, to do this no computers are required.

In a very simplified way, the process that allows us to get the tables can be summarized as follows:

- 1- There is a part of the real world from which we want to keep information (library, address book, etc).
- 2- By means of Entity-Relationship diagrams a representation of this information is depicted.



- 3- Applying the relational model rules, from the previous diagrams the final tables are obtained.

TEACHERS

<u>Id</u>	name	address
1	Paul	Wishaw
2	Claire	Glasgow

SUBJECTS

<u>Code</u>	Name	Hours	Teacher_id
C1	English	5	1
C2	French	3	2
C3	Spanish	2	1

Eventually, over these tables it will also be possible to make *queries* later on using the SQL language.

3.2 Objectives

The objectives of these materials are, so, to use English as a teaching and learning language to provide the following:

- An introduction to databases and to the techniques used to design and develop database systems, including:
 - The basics the principles of relational databases
 - the fundamentals of entity-relationship model to represent the real world
 - an account of the rules to obtain the final relations (tables) from the entity-relationship diagrams previously depicted
- Instruction in the use of SQL

3.3 Covered Units

The materials consist of two parts or sections:

PART I.- DATABASE DESIGN

Lesson 1.- Database Management Systems

Lesson 2.- Relational Databases Basics

Lesson 3.- Entity-Relationship Model

Lesson 4.- Transformation into Tables from ER Diagrams

PART II.- SQL LANGUAGE

Lesson 1.- SQL Basics

Lesson 2.- Data Manipulation Language

Lesson 3.- Data Definition Language

These two section units are somewhat distinct from each other:

- To do the Part I there's no computer is needed, however that doesn't mean at all that the discussed contents are theoretical only. The included activities in this section involve thinking and decision-making in a similar way as it is necessary when resolving mathematical problems.
- The Part II is a hands-on unit that does need the use of a computer room for the students to be able to execute the SQL statements on a specific DBMS such as MySQL in the aspects of the database definition, database query and database updating.

3.4 Covered Units

The contents of the lessons are indicated below:

PART I.- DATABASE DESIGN

Lesson 1.- Database Management Systems

This lesson provides an overview of database management systems which include concepts related to data, information, databases and database management systems. These concepts have to be supported the rest of the materials.

Lesson 2.- Relational Databases Basics

The basic notion of a relational database, based on the relational model, is introduced here without discussing the more formal principles underlying this model.

Lesson 3.- Entity-Relationship Model

As the first step in the process of designing a database, how to draw ER simple diagrams based on a given scenario is studied in this lesson. At this point, all ER diagrams include relationships that involve two entity-types only.

Lesson 4.- Transformation into Tables from ER Diagrams

After studying this lesson, the students will learn to convert an ER diagram to an equivalent relational model.

PART II.- SQL LANGUAGE

Lesson 1.- SQL Basics

This lesson gives an introductory description of the principal features of the SQL language.

Lesson 2.- Data Manipulation Language

As the name says, the data manipulation language (DML) statements are introduced.

- First, the lesson deals with the SQL statements that can be used to modify the data within a database.

- Next, beginning with the simplest SELECT statement, we will go into more complex statements to learn how to retrieve data from the database.

Lesson 3.- Data Definition Language

This lesson explains how to create database objects (such as tables), change the structure of existing object, or delete an object.

4. Activities

The activities included in the handbook are diverse depending on the section and the type of lesson:

PART I. DATABASE DESIGN

As it was said previously, the Part I don't need the use of computers but the activities in this section can be differentiate between theoretical and practical:

For the first lesson (*P1.L1.DBMS*), there are not practical activities but several issues about the different explained concepts. In some activities the student must choose among several options whereas others expect an open answer from the student. Although, the answer will never entail in giving a formal definition for a concept, there are plenty of "tricky" questions whose answer requires to have understood the explanation.

Some times there is an only answer for a question or activity (*P1.L1.DBMS*, *P1.L2.SQL_Basics* and *P1.L4.Transformation*) while in others the solution has to be put into a context and is quite subjective (*P1.L2.Relational_Databases* and *P1.L3.ERM*).

PART II. SQL LANGUAGE

This is basically a practical unit but the lesson 1 (*P2.L1.SQL_Basics*), since it's only an introductory lesson, can be considered to be theoretical so it raises questions relate to the syntax and the features of the language. In this case there's also one possible answer for each.

The rest of the lesson in this section (*P2.L2.DML* and *P2.L3.DML*) is totally practical and most of times the proposed queries can be solved in more than one way.

In any case, this teacher's guide gives the answers to all the proposed activities and the power point presentation guides the teacher telling him/her when each exercise must be carried out.

5. Metodology

5.1 Curriculum

The curriculum of the CFGS of DAI and ASI lays down the units that must be handled as follows:

INTRODUCTION TO DBMS

- **objectives of a DBMS and evolution**
- **ANSI/SPARC architecture**
- **physical and logical independence**
- *data models*
- **data definition and data manipulation languages**
- **users of a database**

ENTITY-RELATIONSHIP MODEL

- **Entity-types, attributes and representation**
- **Relationships, attribute, degree of a relationship and representation**
- Enhanced ER model

RELATIONAL MODEL

- Relational model structure
- **Integrity rules**
- **Transformation from ERD to tables**
- Relational algebra and calculus
- Normalisation of relations

SQL LANGUAGE

- *Operacions de consulta a la base de dades (DML)*
- *Operacions de definició de la base de dades (DDL)*
- *Operacions d'actualització de la base de dades*

(*) The units in italic and bold are totally discussed in these teaching materials while the units in italic only are partially covered.

5.2 Approaches

The materials, so, permit to teach in English the almost the whole process of database design for easy cases. The SQL language could be taught nearly entire in that language.

According to this, there are two approaches to use these materials:

- 1st APPROACH. After the introduction to DBMS and studying the basics of the relational model, the process of designing a database can be put into practise through easy scenarios, by drawing of the ER diagram and transforming them to tables/relations afterwards. Next, the same process should be continued with more difficult information systems going deeply into the relational model and applying the constructs of enhanced ER model.
- 2nd APPROACH. The units might be studied in the same order as drawn up in the curriculum (all ER model first and then all the relational model) therefore English would be only used to teach the parts these materials are comprised of and to do the included activities.

6. Needed Resources

To teach each lesson, you'll need to use the corresponding power point presentation⁴, so a projector available is required.

As it has been said before, the lessons concerning to the Part I can be done in a regular classroom, whereas a computer room is needed to teach the Part II because of their practical contents.

⁴ The lesson plan is included at the end of each lesson, shows the corresponding power point presentation to be used.

Activities

Part 1

Database Design

Part 1. Lesson 1

Database Management Systems

Introduction

As the title says, this lesson outline an overview of database management systems which include concepts related to data, information, databases and database management systems.

This lesson provides an introduction to the subject covering some of the basic principles that we need to support the rest of the subject and, so, the rest of these teaching materials.

a) The students should be aware of the importance of information as a resource for decision-making as well as the need to process and store information.

b) Next, the concept of database and database management system will be discussed and the students will learn the advantages of a database management system as opposed to a file-base system. The concept of data model and the ANSI/SPARC architecture will be also taught.

c) Eventually, the different users of a database and their functions will be covered.

Approach

This is a theoretical lesson and, therefore, there're no practical activities but several questions about the explained concepts.

As for the type of questions included, some of them refer to definitions, so to solve them all you need is to understand the meaning of these definitions, whereas others don't have so an obvious answer and imply certain reasoning. It's important to say that, some times, there's not even an only answer for the question.

With regard to how to solve the activities, the suggested approach is that the students give an answer to each activity individually or in pairs. Next these answers can be discussed and corrected by the whole class, giving the students the chance to brainstorm ideans and justify the possible solutions.

Activities



ACTIVITY 1.1 – *Information vs Data*

objective

To understand the difference between data and information.

Classify the following elements as data or information and justify your answers.

- a) not-known sounds
- b) 11092001
- c) xmp
- d) an exam without the answers
- e) a weather forecast

suggested answers

- a) We would say that they are data because, in principle, they have no meaning. However, if we put these sounds into a context (sounds coming from a printer or from a car, for instance), we can say that they are information (the printer or the car are not working well).
- b) At first sight they are data, but if we say that this element refers to a date ("11-09-2001") then we can say it's information (the day of the terrorist attack to the twin towers in NYC).
- c) It looks like data, although if I say that's the initials of my name (Xavier Morera Parra) obviously it turns into information.
- d) Even though there are no answers for the exam, the questions are information. Of course, that is only true if we're talking about a subject we know, etc.
- e) Information if it's understandable by the recipient. However, if we are given a set of maps and mathematical formulae that are too complicated to be understood by ordinary people (you need to be a meteorologist to

understand it), it would be data.



Extra information for the teacher...

In short, the students must be aware that the boundaries between data and information are weak; sometime by putting the data into a context they become information.



ACTIVITY 1.2 – *Quality of Information*

objective

To know the properties that information must fulfil to be considered high quality.

Match the following definitions of properties of information with their corresponding names.

- a) Information should be accurate when it is presented.
- b) Information should be set out in a clear, accessible format and communicated in a way that is appropriate for the user. The level of detail included and the language used should reflect the specific needs and expertise of the user.
- c) Information should have the trust of the users. If the user believes that the underlying data are inaccurate or the original source is not reliable, then the information will be useless to them. They won't be able to make a decision based on such information.
- d) Information should be made available when needed. It must be presented at the appropriate moment in the decision-making process if it is to aid that process.
- e) Information should be error free and a true reflection of what it represents. Even minor inaccuracies can lead to poor quality decisions being made.
- f) Information should not include data that has no bearing on the user's information needs. Receiving more information that one needs can lead to "information overload". The specific information that the user needs becomes lost among data that is not needed.
- g) Information should include all the data which the user needs to make a decision. There should be no significant absences; absence of information is just as misleading as inaccurate information.

h) Information should be presented to the people who need it. Giving information to people who have no need of it can contribute to "information overload". It can also create confusion and may lead to situations where information that was meant to be confidential ends up in the wrong hands.

suggested answers

- | | |
|------------------------|-----------------------|
| a) Up to date | e) Accurate |
| b) Understandable | f) Relevant |
| c) Has user confidence | g) Complete |
| d) Timely | h) Correctly targeted |



ACTIVITY 1.3 – The Role of a Database Expert

objective

To understand the duties that database experts carry out.

In this lesson we have seen that some information systems may not be automated.

In that case, what do you think the role of a computer and database expert could be?

suggested answer

The answer is quite simple: as a computer expert you can decide to computerize the current information system total or partially. As it has been said throughout the lesson, the design and use of a database also takes part of this process.



ACTIVITY 1.4 – Identifying the components of an IS

objective	<p>The objective is that the students work, almost without realising, on an exercise of analysis of information systems (that is what they will do in the next lessons to design databases) and try to get the important information from a real life context.</p>
<ul style="list-style-type: none"> - Define the working procedures, the information, the users and the physical support of a small video shop. - Analyse how the changes in one of the components affect the others. 	
suggested answer	
<p>a) <u>Users</u></p> <ul style="list-style-type: none"> - SHOP ASSISTANT (to register information about the customer and the hired films). - MANAGER (to get reports and decide whether to buy more copies) <p style="padding-left: 40px;">* customers (people that hire a film) won't use the system</p> <p>b) <u>Information</u></p> <ul style="list-style-type: none"> - FILM: title, year, director, b/w, language, ... price, copies.. - COPY: ... - CUSTOMER: name, phone, address, ... - BORROWINGS: customer, copy, date, returning date, <p style="padding-left: 40px;">* Borrowing length??</p> <p>c) <u>Working procedures</u></p> <ul style="list-style-type: none"> - Does a customer have to register with the videos shop before being able to hire a movie? - What is the registration process? Does the costumer have to show 	

some documentation?

- How long can the movie be kept?
- How many copies can be hired at the same time?
- What must be checked before hiring a copy?: The customer is registered and he doesn't have a copy awaiting to return...
- What happens if the customer returns a movie after the date? Is he penalized? How?
- What happens if a customer never returns the movie? How do we now?
- Is there any system to know what movies are going out of fashion for example?

d) Physical support

It depends on the working procedures but we should decide which parts of these procedures should be automated what implies to think about the needed software and the database to be designed.



ACTIVITY 1.5 – Advantages of a DBMS

objective

To understand the meaning of the advantages of using a database management system.

a) Match each concept with its definition.

- | | |
|--------------------------|---|
| 1. No data redundancy | a) The immunity of application programs to changes in storage structures and access techniques. |
| 2. Data integrity | b) In case of a system failure the database must remain consistent. |
| 3. Data independence | c) Different users attempting to update the same data at the same time. |
| 4. Concurrency | d) Data are stored only once. |
| 5. Authorization control | e) The data available in the database are reliable. |
| 6. Backup and recovery | f) Only authorized users can access the data. |

b) Most of concepts above refer to advantages of using a DBMS but some of them don't. In fact, they can be seen as drawbacks of DBMS's, can you identify them?

suggested answers

a)

1-e

4-c

2-e

5-f

3-a

6-b

- b) Concurrency control and authorization control are consequences of keeping all data in the same place and can be considered disadvantages.



ACTIVITY 1.6 – Drawbacks of a DBMS

objective

To understand the disadvantages and consequences of using a database management system.

1.- Give an explanation for every disadvantage of using a DBMS.

- a) Cost of using DBMS
- b) Data integrity
- c) Data quality
- d) Confidentiality, privacy and security
- e) Enterprise vulnerability

2.- Can you mention situations where it is not desirable to use DBMS?

suggested answers

1.-

a) The cost of using a DBMS

- Developing a new database system is expensive. The database structure (tables) has to be designed and the software should be "tailor made" and that costs a lot of money.
- The maintenance is also costly.
- Staff training.
- Using standardised software (Access, etc) is an alternative but it's often always less efficient than using specialised software.

b) Data Integrity

- Since a large number of users can use a database at the same time (concurrency) some techniques are necessary to ensure that the data remain correct during operation.

c) Data Quality

- With a large number of users accessing data, there are enormous opportunities for users to damage the data.

d) Confidentiality, Privacy and Security

- When the information is centralised and is available to all users, it's necessary to take measures to allow access only to the authorized users.
- The organisation is more vulnerable to data loss, so there're must be a disaster recovery policy.

d) Enterprise Vulnerability

- The survival of the company may depend on the information stored in its database. If the database is destroyed or changed without authorisation the company survival can be in danger.

2.-

- The database and applications are not expected to change.
- Data are not accessed by multiple users.



ACTIVITY 1.7 – DBMS Languages

objective

This activity must be just an introduction for the students to know to the possibilities of a data file.

These concepts will be studied deeply later on when the SQL is covered.

Consider the students data file of the lesson.

Show the operations that can be done with a data file and write the DBMS language that is involved as in the example:

ACTION	OPERATION	LANGUAGE INVOLVED
add a new student	insert new record	DML

suggested answers

ACTION	OPERATION	LANGUAGE INVOLVED
Create the students table	Create table	DDL
Add a new student	Insert a record	DML
Remove an existing student	Delete a record	DML
Modify the data of an existing student	Edit a record	DML
Store new information from each student	Add a new field to the structure of the database	DDL
Search if there's a student with a specific value in the age	Search/ Query	DML
List the students that	Query	DML

fulfil a condition		
Sort the students	Sort	DML
Destroy the table	Drop	DML

Summary

This lesson has given a brief history of databases, explaining the evolution from file storage to the current DBMS. It has also examined the main data models.

Before discussing the most important data model to date in the next lesson, the relational data model, it's important to understand the various terminology associated with databases in general and with the relational databases in particular.

- Nowadays, information is essential to decision-making.
- Only processed data gives useful information.
- All companies and organisations develop an infrastructure to manage all the necessary information: the information system.
- Some of the traditional information systems were file-based; computer files systems were used to store, manipulate, and retrieve data.
- File-based systems had limitations such as data duplications, limited data sharing, and no data independence.
- In order to overcome these limitations database approach was developed.
- A database is an integrated and organized collection of related data.
- The main objective of database management system is to store and manipulate the data in an efficient manner.
- The main advantages of DBMS approach are program-data independence, improved data sharing, and minimal data redundancy.
- There are different types of users of a DBMS.
- A data model is a representation of data
- A number of data models have been described: hierarchical, network, relational. They are used to understand the nature of information and design better databases.
- The entity-relationship model permits to represent the part of the real-world (organisation) and is the first step for designing relational databases organisation.
- The predominant form of database today is the relational databases.

- Database and design is a process that involves to analyse within the information system.

Glossary I

English-Catalan Vocabulary

English	Meaning
A set of	Un conjunt de
Accurate	Detallat/da
Approach	Enfoc, mètode
Available	Disponible
Confidence	Confiança
Guideline	Directriu, pauta
Huge	Enorme
Nowadays	Avui en dia
Programming language	Llenguatge de programació
Quiz	Concurs, prova
Relieable	De confiança
Resource	Recurs
System crash	Caiguda del sistema
To allow	Permetre
To assess	Avaluar
To be made up of	Estar format/compost de
To carry out	Portar a terme
To Fulfil	Acomplir
To guess	Adivinar
To involve	Implicar, suposar
To manage	Manipular
To perform	Desempenyar
To remove	Eliminar
To report	Informar, comunicar
To retrieve	Extraure, recuperar
To trust	Refiar-se'n
Useless	Inútil
Wastage	Desperdici
To have a bearing on somethig	Tindre a veure amb alguna cosa
To lead to	Conduir a

English	Meaning
To match	Combinar
To remain	Romandre
Drawbacks	Inconvenients

Key Vocabulary

At the end of this lesson the students must be able to recognise and understand the meaning of the following words:

ANSI/ASPARK ARCHITECTURE

COMPUTER SCIENCE

CONCURRENCY

DATA

DATA DEFINITION LANGUAGE
(DDL)

DATA DEPENDENCE

DATA MANIPULATION LANGUAGE
(DML)

DATA MODEL

DATA REDUNDANCY

DATABASE ADMINISTRATOR

DATABASE DESIGNER

DATABASE MANAGEMENT SYSTEM
(DBMS)

DATABASE USER

EXERNAL LEVEL

FILE

FORM

INFORMATION

INFORMATION SYSTEM (IS)

LOGICAL DATA INDEPENDECE

PHYSICAL DATA INDEPENDENCE

PHYSICAL SUPPORT

QUERY

REPORT

WORKING PROCEDURES

Lesson Plan		PART I - DATABASE DESIGN	
Lesson 1. Database Management Systems			
TIMING	4 hours	LEVEL	1st CFGS DAI/ASI
LEARNING OBJECTIVES	CONTENT	<ul style="list-style-type: none"> - Nature of the information - Information systems, components - File-based systems, drawbacks - Database - Database management system - Data models: hierarchical, network, relational and entity-relationship - Architecture ANSI/X3/SPARC - Data independence: logical independence and physical independence - Users of a database - Languages of a DBMS 	
	COGNITION	<ul style="list-style-type: none"> - To understand the advantages and disadvantages of using a database management system. 	
	CULTURE	<ul style="list-style-type: none"> - Importance of the information in the current society for the decision-making. 	
	COMMUNICATION	<p>Language for learning:</p> <ul style="list-style-type: none"> - To discuss and ask about the concepts. - To justify their answers and reasoning <p>Language of learning:</p> <ul style="list-style-type: none"> - Language structures needed to complete the tasks <p>Language trough learning:</p> <ul style="list-style-type: none"> - Language that comes up when carrying out the different activities 	

LEARNING OUTCOMES	<p>At the end of this lesson, students will be able to...</p> <ul style="list-style-type: none"> - Recognize the characteristics of the high quality information - Be aware of the importance of the data analysis in order to automate an information system - Identify the role of database design within the overall systems design - Understand a basic definition of a database - Understand the separation of data definition from applications - To know the advantages of using a DBMS - Identify the functions and components of a DBMS - Understand the importance of data modelling - Describe the nature of data models and their characteristics - Outline the essential principles of hierarchical, network and relational database models - Give a description of the three-layer ANSI/SPARC architecture of databases - To distinguish between DDL and DML - To identify the different users of a database 																
RESOURCES AND TIMING	<p>Lesson in power point (<i>P1.L1.DBMS.ppt</i>)</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <th colspan="2" style="background-color: #e0e0e0;">Lesson 1.- Database Management Systems (4 hours teaching + activities)</th> </tr> <tr> <td>Activity 1.1</td> <td>Information vs Data</td> </tr> <tr> <td>Activity 1.2</td> <td>Quality of Information</td> </tr> <tr> <td>Activity 1.3</td> <td>Role of a Database Expert</td> </tr> <tr> <td>Activity 1.4</td> <td>Identifying Components of an Information System</td> </tr> <tr> <td>Activity 1.5</td> <td>Advantages of a DBMS</td> </tr> <tr> <td>Activity 1.6</td> <td>Drawbacks of a DBMS</td> </tr> <tr> <td>Activity 1.7</td> <td>DBMS Languages</td> </tr> </table>	Lesson 1.- Database Management Systems (4 hours teaching + activities)		Activity 1.1	Information vs Data	Activity 1.2	Quality of Information	Activity 1.3	Role of a Database Expert	Activity 1.4	Identifying Components of an Information System	Activity 1.5	Advantages of a DBMS	Activity 1.6	Drawbacks of a DBMS	Activity 1.7	DBMS Languages
Lesson 1.- Database Management Systems (4 hours teaching + activities)																	
Activity 1.1	Information vs Data																
Activity 1.2	Quality of Information																
Activity 1.3	Role of a Database Expert																
Activity 1.4	Identifying Components of an Information System																
Activity 1.5	Advantages of a DBMS																
Activity 1.6	Drawbacks of a DBMS																
Activity 1.7	DBMS Languages																

Part 1. Lesson 2

Relational Databases Fundamentals

Introduction

This lesson gives the basis to understand the relational data model providing an intuitive description of how a relational database is composed of set of tables. With these elementary ideas in place and without getting started with the mathematical theory of sets, the lesson outlines how the relations can represent the information stored in tables.

The lesson is organised as follows:

- a) First the students will learn the concept of table as a way to represent the information of the database. They will also have to be beware of the drawbacks of using flat tables.
- b) Next, important concepts of the relational model will be discussed:
 - The meaning of key and all its sub-types: candidate keys, alternate keys, primary keys and foreign keys.
 - The concept of null value.
 - Integrity constraints that restrict the data that can be stored in the database.

- c) Finally, the relation as the main construct to represent information will be introduced.

Approach

The need of justifying the answers characterizes this lesson.

The answers to the included activities are not obvious, so to solve them the students need to have understood the meaning of the covered concepts. For some activities there is not even a simple answer and, in any case, the students have to think about the possible solutions and justify them.

The teacher should give some time for the students to think about the possible answers for each activity. Next, the students should be given the chance of discussing and justifying their answers.

Activities



ACTIVITY 2.1 – Identifying Tables

objective

The main concern is to understand that the necessary information will be likely to be stored in different tables.

Imagine that we wish to store information about the students in our college and the subjects they are enrolled as well as the grades that each student has got for each subject.

Think about the potential tables.

suggested answers

The following tables would be needed:

- SUBJECTS
- STUDENTS

Each of the items above corresponds with a table. The final tables and their contents will depend on the limits of our reality; we have to make clear what the information we are interested in is. For instance, can there be a student without any subject?, etc.

Anyway, in this example, there's also necessary a third table to keep the grades for each student.



Extra information for the teacher...

At this point the students won't be able to identify the final tables, but it's important them to understand that there won't always be an only table for each item we'd like to store information about. They have to be aware that that's the reason why we need to analyse the information system and to follow a

methodology.

**ACTIVITY 2.2 – Flat Tables****objective**

Following the points of the previous activity, the aim is again to realise that, in most cases, isn't a good idea to store all the information in a single table.

The activity can also be used to introduce the concept of foreign key later on.

Imagine that we need to keep information about cars and their owners. The information has been stored in a table as shown:

registration number	brand	model	Id	name	adress
V-30	SEAT	IBIZA	20	Xavi	Barcelona
M-81	SEAT	LEON	18	Anna	Granollers
B-32	VOLSWAG	GOLF	20	Xavi	Barcelona

What's wrong in this arrangement? What's the solution?

suggested answer

The answer to that activity is developed along the following power point slides.

There's redundancy of information (all information about an owner is repeated for each car he/she has). As a result:

- There's a waste of space
- To add a new car or to edit the information of an owner is more difficult: it takes longer to do it (time-consuming).
- It's possible to make a mistake and get inconsistency!!

The solution entails of splitting the table as it's showed in the next power point slide.



Extra information for the teacher...

The students must understand that relational databases represent the data as a set of two dimensional tables that will be connected to one another.



ACTIVITY 2.3 – Tables and Values

objective

With this activity the students must be aware of the fact that only a single value is permitted for each column.

The exercise will be also useful to stress the idea that in most cases more than one table are needed.

We have information about the students in a language school and the languages they study.

Given the following table:

name	Age	language	level
Louis	18	English	Intermediate
Anne	23	English French	Upper-intermediate Beginner
James	19	German	Advanced
Wendy	36	Italian	Pre-intermediate

a) This arrangement isn't correct. Why?

b) Can you give a solution?

suggested answer

a) By definition, a specific field (column) in a table can store a single value only. That will be studied deeply later on when the normalisation process is dealt with.

b) To solve it, we have to take the multiple values off. The solution entails dividing the table into two as follows:

STUDENTS

name	age
Louis	18
Anne	23
James	19
Wendy	36

GRADES

name	language	level
Louis	English	Intermediate
Anne	English	Upper-intermediate
Anne	French	Begginer
James	German	Advanced
Wendy	Italian	Pre-intermediate



ACTIVITY 2.4 – Identifying Candidate Keys

objective

The students have to understand that to identify a candidate key for a table is essential and they must learn to choose at least one.

We have designed a database to keep information about students in our college.

Identify possible candidate keys for them.

name	fname	phone	country	age
David	Roney	999	US	23
Jenn	Murphy	111	Scotland	32
Robert	Palmer	222	Ireland	35
David	MacPhils	444	Scotland	28

suggested answer

None of the fields of the table above can be considered a key as they can't identify a student from the rest; there could be two students with the same name, the same fname, the same phone, the same country or the same age. To use a composite key don't solve the problem either (name+fname) because, for the same reason, two students might have the same full name.

A new column to carry out this function must be added: id number, enrol number, code, etc.



Extra information for the teacher...

The students must know that to find a key for a table is not always obvious and some times it is necessary to add a new column that carries out this task.



ACTIVITY 2.5 – Composite Keys

objective

The students must be warned that a key may consist of more than one attribute.

The activity can also be used to introduce the meaning of candidate, alternate key and foreign keys.

A residents' association wishes to store information about all the flats in the block.

All needed data are stored in two tables as shown:

FLATS

floor	flat_number	area	owner

OWNERS

id	forename	surname	date_of_buy

Where:

- ♦ *floor* shows the level where the flat is located (1st, 2nd, etc.)
- ♦ *flat_number* shows the number of the flat within a specific floor (1, 2, 3, etc.)
- ♦ *area* indicates the square meters.
- ♦ *owner* gives information about the id of its proprietor.

a) Identify a candidate key for the tables above.

b) Can a flat have more than an owner?

c) Can many flats belong to the same person?

suggested answer

a) As regard the table for flats:

- *Floor* is not a key. Obviously, by knowing the floor we don't know the

flat and its owner (unless there was an only flat per floor).

- *Flat_number* is not a key, either. The same number can exist in different floors.
- However, *floor+flat_number* is key.

code	floor	flat_number	area	owner
P1	1	1	70	1000000A
P2	1	2	85	2222222E
P3	2	1	65	9999999P
P4	2	2	90	5050505M

Another possibility would be to add a new field to store a code that identifies each flat.

As regard the table for owners:

- *Id* could be a key. *Forename + surname* it's not an advisable key because two owners might have the same value for these fields.
- *Id* is the only field in the owners table that can be a key.

b) No. The field *owner* can keep a single value, an id therefore, the existing id shows the only owner of the flat.

c) Yes. The field *owner* can have repeated values, in other words several flats can belong to the same person.



ACTIVITY 2.6 – Primary and Alternate Keys I

objective

The students have to be able to identify all the possible keys for a specific table.

With this activity the teacher can take the opportunity to talk about the concept of foreign key and ask the students about the difference between primary and alternate key

Identify the candidate keys, alternate keys and primary keys for each table for the previous activity.

suggested answers

With regard to the table for flats:

- There are two possible keys (candidate keys): *floor+flat_number* or *code*. If we choose *code* as primary key, *floor+flat* is an alternate key. That also works the other way round: if *floor+flat* is believed to be the primary key, *code* will be an alternate key.

As regard the table for owners:

- There's only a possible key, which is the only candidate key and, therefore, the primary key: *id*
- That means that the field *owner* in the flats table, stores information about the id of the proprietor because both tables have to be connected by the same field type.



Extra information for the teacher...

A foreign key in a table and the primary key in another table it is connected to can have different names. However, the data type must be the same and, obviously, the meaning of their content is exactly the same for in both tables.



ACTIVITY 2.7 – Primary and Alternate Keys II

objective

As in the previous activity, the students have to be able to identify all the possible keys for a specific table.

Now consider a table that contains information about the different departments of the college.

code	Name	floor	teachers
ICT	Computing and IT	1	16
MKT	Marketing	2	8
BUS	Business and Administration	1	5
FL	Foreign Languages	2	3

Identify candidate keys for them and choose a primary key.

suggested answer

code is without a doubt a candidate key for this table (there's a different code for each department). On the other hand, if we assume that all the departments in the college have different names, *name* could also be thought to be a candidate key. (This last case will depend of the reality that is subjected to study).

One of them will have to be chosen as the primary key whereas the other would be an alternate key.



ACTIVITY 2.8 – Selecting a Primary Key

objective

The aim is to understand the importance of identifying a primary key among all the candidates.

This activity can also be used to work on the idea of composite key.

Imagine we keep information about the subjects of our technical studies curriculum (DAI or ASI).

a) Identify the primary key for that table.

code	name	hours	year
C1	Operativing systems	240	1
C2	Networks	220	1
C3	Database analysis	90	1
C4	Aplications analysis	190	2

b) Now, imagine that we store information about all subjects in the college (from the different technical studies). **Would the primary key be the same?**

suggested answers

a)

CASE 1

If we are referring only to the subjects of some specific technical studies (DAI, ASI, etc), *code* would be certainly a candidate key. If we take for granted that there are not two subjets with the same name, *name* would also be a candidate key. In that case, *name* would also be another candidate key.

In principle, any of both candidate keys would play the role of primary key but, the most sensible thing would be to choose *code* as pk.

CASE 2

Imagine now that the table keeps information about the different technical studies that are given the college.

- *code* won't be a candidate key because several subjects from different technical studies might have the same code; if we refer to *C1* we don't know what subject are we talking about.
- *name* won't be a candidate key either because, unlike case 1, we know that there are several subjects from different technical studies with the same name (e.g. FOL)

As we have said before, to identify the subject it's not enough with the code because it's not the same *C1* of DAI that *C1* of SMX. So, we also need to know the technical studies we are referring to. The solution entails adding a new column to save this information:

<u>code</u>	<u>t_studies</u>	<u>name</u>	<u>hours</u>	<u>year</u>
C4	DAI	Aplication programs analysis	240	2
C4	ASI	Programming	220	1
C3	DAI	Database analysis	90	1
C3	ASI	Aplication programs introduction	200	2

The only candidate key and, therefore, the primary key would consist of two columns: *code* + *t_studies*.



ACTIVITY 2.9 – Foreign keys

objective

The students must learn the meaning of foreign key.

Identify the possible foreign keys for the activity 2.5.

suggested answer

The attribute *owner* in the FLATS table must refer to the person who possesses the flat which is in the OWNERS table. So, *owner* is a foreign key in the FLATS table that points to *id* in the OWNERS table. Both columns contain the same type of value: the id of the owner of a flat.



Extra information for the teacher...

As it was said before, it is important to make clear that the name of the attributes can be different although they contain the same data.



ACTIVITY 2.10 – Library

objective

The idea is to work on an activity that includes all the concepts about keys learnt so far.

In an imaginary library use the following tables to manage information of books and borrowings:

BOOKS

book_code	title	publisher	language	author	editions
1	Introduction to database desing	Adisson	English	Danson	5
2	SQL para principiantes	Adisson	Spanish	Danson	3
3	Database design using ER diagrams	Prentice-Hall	English	Martin	1

COPIES

copy_number	book_code	year_edition	number_edition
1	1	2003	5
2	1	2003	5
3	2	2004	2
4	2	2004	2
5	2	2007	3
6	3	1997	1

THEMES

Book	theme
1	Database
2	Database
2	SQL
3	Database

MEMBERS

member_code	name	fname	id	address	phone	Date_registration
1	Steven	Brown	11111111	Glasgow	111	01/01/2009
2	Eric	Matew	22222222	Barcelona	222	01/01/2004
3	David	Rooney	33333333	Barcelona	333	01/01/2009

BORROWINGS

member	copy	date	deadline	returning_date
1	1	15/10/2007	30/10/2007	20/10/2007
1	4	20/11/2008	05/12/2008	05/12/2008
2	2	01/06/2007	01/07/2007	02/07/2007
2	5	15/09/2008	15/10/2008	
2	2	10/10/2008	10/11/2008	11/11/2008
3	3	30/10/2009	30/10/2009	

Identify the primary keys, alternate keys and foreign keys for the tables above.

suggested answers

BOOKS table

Primary key = book

COPIES table

Primary key = copy_number

Foreign key = book_code

THEMES table

Primary key = book+theme

Foreign key = book

MEMBERS table

Primary key = member_code

Alternate key = id

BORROWINGS table

Primary key = member+copy+date

Foreign key = member

Foreign key = copy



ACTIVITY 2.11 – Wrong Records

objective

The integrity constraints learnt in this lesson will be put into practice.

Consider the database used for the residents' association (activity 2.5)

The contents for these tables are showed next:

FLATS

<u>Code</u>	floor	flat_number	area	owner
F1	1	1	60	10000A
F2	1	2	75	50000M
F3	1	1	60	
	2	1	70	20000P
F5	2		56	10000A
F6	2	3		
F7			90	

OWNERS

<u>Id</u>	forename	surname	date_of_buy
90000B	David	Smith	11/01/2001
10000A	David	Stout	
90000B		Paterson	07/04/2008
20000P	David	Smith	28/10/2003

a) Identify the wrong values for these tables showed above.

b) Can the owner with id 10000A be removed from the database?

suggested answers

a)

1st row: correct.

2nd row: the owner doesn't exist; it doesn't fulfil the foreign key integrity constraint.

3rd row: this flat is already in the table; there are duplicate values for an alternate key.

4rd row: the primary key contains a null value; it doesn't fulfil one of the key

integrity constraints.

5th row: the alternate key contains a null value but that's not a problem.

6th row: it might be correct; the foreign key can have a null value if we assume that a flat don't need to have an owner.

7th row: it's a combination of the 5th and 6th row, so it's also correct.

b)

Yes, but all the flats for this owner (F1 and F5) should also be removed (that idea also works for updates).

Another possibility is to ban the delete of this owner while he owns some flat.



ACTIVITY 2.12 – Representing Tables I

objective

The students must learn to use relations to represent a relational database structure.

Consider the DISTRIBUTORS database used by an imaginary group of suppliers of pieces.

The database has the following three tables:

□ PIECES

p_id	P_name	P_colour	p_weight	p_city
P1	nut	Red	12	Edinburgh
P2	peg	Green	17	Glasgow
P3	screw	Blue	17	Motherwell
P4	screw	Red	14	Edinburgh
P5	washer	Blue	12	Glasgow
P6	sprocket	Red	19	Edinburgh

□ DISTRIBUTORS

D_id	d_name	d_age	d_city
D1	Smith	30	Edinburgh
D2	Jones	28	Glasgow
D3	Blake	40	Glasgow
D4	Clark	30	Edinburgh
D5	Adams	40	Stirling

□ SUPPLIES

d_id	p_id	amount
D1	P1	300
D1	P2	200
D1	P3	400
D1	P4	200
D1	P5	100
D1	P6	100
D2	P1	300
D2	P2	400
D3	P2	200
D4	P2	200
D4	P4	300
D4	P5	400

Represent the tables above using the relations from the Relational Model.

suggested answers

PIECES (p_id, p_name, p_colour, p_weight, p_city)

primary key: p_id

DISTRIBUTORS (d_id, d_name, d_age, d_city)

primary key: d_id

SUPPLIES (d_id, p_id, amount)

primary key: d_id, p_id

foreign key: id_s → DISTRIBUTORS

foreign key: id_p → PIECES



ACTIVITY 2.13 – Representing Tables II

objective

Use relations to represent a relational database structure.

Transform the tables of the activity 10 into relations in the relational model.

suggested answers

BOOKS(book_code, title, publisher, language,author, editions)

Primary key: book

COPIES (copy_number, book, year_edition, number_edition)

Primary key: copy_code

Foreign_key: book → BOOKS

THEMES(book, theme)

Primary key: book, theme

Foreign Key: book → BOOKS

MEMBERS(member_code, name, fname, id, address, phone, date_registration)

Primary key: member_code

Alternate key: id

BORROWINGS (member, copy, date, deadline, returning_date)

Primary key: member_code, copy_number, date

Foreign Key: member → MEMBERS

Foreign Key: copy → COPIES



ACTIVITY 2.14 – Understanding Relations

objective

The students have to understand the meaning of the relations being able to imagine the real world they represent.

Say if the following statements are TRUE or FALSE for the obtained relations in the previous activity:

- a) A book can have several authors.
- b) An author can write many books.
- c) A book can cover many themes.
- d) The same theme can be discussed in several books.
- e) Two members can have the same name.
- f) Two members can have the same id.
- g) Two members can have the same member_code.
- h) A member can not to have a telephone number.
- i) The library can not to have information about the id of a member.

suggested answers

- a) The attribute *author* can take a single value for each row (book) in the BOOKS table, meaning that a book has an only author.
- b) The same value for this field can appear several times in the table; the same person writes many books.
- c) The primary key for the THEMES table is book+theme, so the value for one of these fields can be repeated. If the *book* appears several times with different values in *theme*, means that this book looks at several themes.
- d) For the same reason, the same value for the theme can be in the table

for different books.

- e) *name* is not a primary key not even an alternate key, so the same value for that attribute can be repeated; two members have the same name.
- f) *member_code* is an alternate key for the MEMBERS table, so it can have repeated values.
- g) *id* is an alternate key for the MEMBERS table, so it can have repeated values either.
- h) *telephone_number* is not a primary key, so it can be a null value.
- i) *Id* is an alternate key, so its values can be null.



ACTIVITY 2.15 - *Understanding Relations II*

objective

At this point the students should be able to understand the meaning of a set of relations that represent a part of the real world and to draw their corresponding tables.

We wish to keep information about the different marriages that take place in a specific town. Our three database analysts don't come to an agreement and each of them has proposed a solution.

CASE 1.-

PERSON (dni, sex, ...)
 MARRIED_TO (dni, dni', data)
 CAlt: dni'
 CExt: dni → PERSON
 CExt: dni' → PERSON

CASE 2.-

PERSON (dni, sex, ...)
 MARRIED_TO (dni, dni', date)
 CExt: dni → PERSON
 CExt: dni' → PERSON

CASE 3.-

PERSON (dni, sex, ...)
 MARRIED_TO (dni, dni', date)
 CExt: dni → PERSON
 CExt: dni' → PERSON

Draw the tables for each case and answer the following questions about them justifying them:

- Can a person get married several times?
- Can a person marry another person several times?
- Might a person's partner be married to another person at the same time?

d) Can a person get married to himself/herself?

suggested answers

CASE 1

- a) No, *dni* and *dni'* can't get duplicated values.
- b) No, a person (*dni*) can't get married again (it doesn't matter if the partner is the same or not) because *dni* can't get repeated values.
- c) Yes!! The same value in *dni* can appear in another row in the field *dni'*.
- d) Yes!! The values for *dni* and *dni'* don't get repeated values!!

CASE 2

- a) Yes, *dni* can take repeated values.
- b) No, *dni+dni'* can take repeated values as they are the primary key. However, by swapping the *dni* and *dni'* values, the result is that they can!!
- c) Yes, as in the previous case, by swapping the *dni* and *dni'* values the same person can have two partners.
- d) No matter how weird it is, both *dni* and *dni'* are allowed to have the same values!!

CASE 3

- a) Yes, *dni* can take repeated values.
- b) Yes, *dni+dni'* can take repeated values for different dates. However, by swapping the *dni* and *dni'* values, the same couple can get married twice in different dates!!
- c) Yes, as in the previous case, by swapping the *dni* and *dni'* values the same person can have two partners even for the same date.
- d) No matter how weird it is, both *dni* and *dni'* can have the same values!!

Summary

In this lesson we have moved on to look at the concept of data modelling applied to the relational data model. We have discussed the most important data model to date and have looked briefly at its basis.

- Relational databases uses two-dimensional tables to represent information.
- The columns of a relational table represent attributes of an entity class and are also called fields.
- The rows of a relational table represent instances of the entity class.
- The relational model is based on the theory of sets.
- Using flat-tables leads to several problems such as redundancy.
- Null is used in relational databases to indicate that an attribute value is missing.
- The primary key of a relation/table is a column or columns that uniquely defines the rows of the table.
- Functional dependency expresses the concept that we can know the value of an attribute by knowing the value of another attribute.
- A candidate key is a column or columns that could potentially be a primary key.
- All the attributes in a table are functionally dependent of the attribute/s of the candidate keys.
- An alternate key is any candidate key that is not the primary key.
- Tables are associated among themselves by means of foreign keys.
- A foreign key in a table/relation is the primary key in another table/relation.
- Referential integrity is the principle that within a table a foreign key in one table must match with an existing primary key in the referenced table.
- Primary key integrity is the principle that no part of a primary key can be null.

Glossary II

English-Catalan Vocabulary

English	Meaning
Approach	Enfoc, mètode
Brand, Make	Marca (comercial)
Constraint	Restricció
Enrol number	Número de matrícula
Grades	Notes
ISBN	International Standard Book Number
Number plate	Númer de matícula (de vehicle)
Owner	Propietari
PT (principal teacher)	Cap de departament
Though	Tanmateix, encara que, però
To belong to	Pertànyer a
To enrol in	Matricular-se en
To keep	Mantenir, guardar
To make of	Estar format de
To relate to	Relacionar-se amb
To take into account	Tenir en compte
To underline	Subratllar
Value	Valor
Waste	Malbaratament
To waste	Malbaratar
To wish	Desitjar

Key Vocabulary

At the end of this lesson the students must be able to recognise and understand the meaning of the following words:

RELATIONAL DATABASE

RELATIONAL MODEL

LOGICAL SCHEMA

DOMAIN

RELATION

RECORD

FIELD

FLAT TABLE

ATTRIBUTES

TABLE

KEY

COMPOSITE KEY

CANDIDATE KEYS

PRIMARY KEY

ALTERNATE KEY

FOREIGN KEY

INTEGRITY CONSTRAINTS

KEY INTEGRITY CONSTRAINTS

FOREIGN KEY CONSTRAINT

NULL VALUE

NOT NULL VALUE

FUNCTIONAL DEPENDENCY

Lesson Plan		PART I - DATABASE DESIGN	
Lesson 2. Relational Databases Fundamentals			
TIMING	6 hours	LEVEL	1st CFGS DAI/ASI
LEARNING OBJECTIVES	CONTENT	<ul style="list-style-type: none"> - Tables - Drawbacks of using a flat table - Candidate keys - Primary keys and Alternate key. Difference - Foreign keys - Null values - Key integrity constraints - Foreign key constraint - Relations 	
	COGNITION	<ul style="list-style-type: none"> - To understand the fundamentals of the data relational model for databases. 	
	CULTURE	<ul style="list-style-type: none"> - Importance of storing the needed information to be able to retrieve it easily later on. 	
	COMMUNICATION	<p>Language for learning:</p> <ul style="list-style-type: none"> - To discuss and ask about the concepts. - To justify their answers and reasoning <p>Language of learning:</p> <ul style="list-style-type: none"> - Language structures needed to complete the tasks <p>Language through learning:</p> <ul style="list-style-type: none"> - Language that comes up when carrying out the different activities 	
LEARNING	At the end of this lesson, students will be able to...		

<p>OUTCOMES</p>	<ul style="list-style-type: none"> - Understand the basis of the relational data model. - To understand how, in a relational database, the information is organised in a set of tables - Be aware of the disadvantages of keeping the information in a flat table - Recognize the different types of keys in the relational model candidate key, primary key, alternate key and foreign key - Understand the meaning of a null value - To know the integrity constraints that restrict the data that can be stored in the database: entity integrity and referential integrity - To represent the structure of a database using relations - Explain, in simple terms, the meaning of relations and how these relate to the relational data model 																																
<p>RESOURCES AND TIMING</p>	<p>Lesson in power point (<i>P1.L2.Relational_db..ppt</i>)</p> <table border="1" data-bbox="488 976 1345 1630"> <tr> <th colspan="2">Lesson 2.- Relational Databases Fundamentals (6 hours teaching + activities)</th> </tr> <tr> <td>Activity 2.1</td> <td>Identifying Tables</td> </tr> <tr> <td>Activity 2.2</td> <td>Flat Tables</td> </tr> <tr> <td>Activity 2.3</td> <td>Tables and Values</td> </tr> <tr> <td>Activity 2.4</td> <td>Identifying Candidate Keys</td> </tr> <tr> <td>Activity 2.5</td> <td>Composite Keys</td> </tr> <tr> <td>Activity 2.6</td> <td>Primary and Alternate Keys I</td> </tr> <tr> <td>Activity 2.7</td> <td>Primary and Alternate Keys II</td> </tr> <tr> <td>Activity 2.8</td> <td>Selecting a Primary Key</td> </tr> <tr> <td>Activity 2.9</td> <td>Foreign Keys</td> </tr> <tr> <td>Activity 2.10</td> <td>Library</td> </tr> <tr> <td>Activity 2.11</td> <td>Wrong Records</td> </tr> <tr> <td>Activity 2.12</td> <td>Representing Tables I</td> </tr> <tr> <td>Activity 2.13</td> <td>Representing Tables II</td> </tr> <tr> <td>Activity 2.14</td> <td>Understanding Relations I</td> </tr> <tr> <td>Activity 2.15</td> <td>Understanding Relations II</td> </tr> </table>	Lesson 2.- Relational Databases Fundamentals (6 hours teaching + activities)		Activity 2.1	Identifying Tables	Activity 2.2	Flat Tables	Activity 2.3	Tables and Values	Activity 2.4	Identifying Candidate Keys	Activity 2.5	Composite Keys	Activity 2.6	Primary and Alternate Keys I	Activity 2.7	Primary and Alternate Keys II	Activity 2.8	Selecting a Primary Key	Activity 2.9	Foreign Keys	Activity 2.10	Library	Activity 2.11	Wrong Records	Activity 2.12	Representing Tables I	Activity 2.13	Representing Tables II	Activity 2.14	Understanding Relations I	Activity 2.15	Understanding Relations II
Lesson 2.- Relational Databases Fundamentals (6 hours teaching + activities)																																	
Activity 2.1	Identifying Tables																																
Activity 2.2	Flat Tables																																
Activity 2.3	Tables and Values																																
Activity 2.4	Identifying Candidate Keys																																
Activity 2.5	Composite Keys																																
Activity 2.6	Primary and Alternate Keys I																																
Activity 2.7	Primary and Alternate Keys II																																
Activity 2.8	Selecting a Primary Key																																
Activity 2.9	Foreign Keys																																
Activity 2.10	Library																																
Activity 2.11	Wrong Records																																
Activity 2.12	Representing Tables I																																
Activity 2.13	Representing Tables II																																
Activity 2.14	Understanding Relations I																																
Activity 2.15	Understanding Relations II																																

Part 1. Lesson 3

The Entity-Relationship Model

Introduction

Database design can be considered as having three phases: conceptual design, logical design and physical design.

In the conceptual design, the nature of the data are modelled without specifying any implementation details such as the DBMS that will be used or the programming languages. The final product in this phase is a set of entity-relationship diagrams (ERD).

The ER model is based on a perception of a real world that humans can easily perceive that consists of collection of basic objects called entities and relationships among these objects. This technique helps analysts and designers understand the nature and relationships that exist within the data of the system. By depicting the reality within an ER diagram, it is possible to move from an informal description of what users want from their database to a more detailed, and precise, description.

The ERD thus produced provides a graphical design which can be then used to derive a set of relational tables that model the data of the application system and that can be implemented on a DBMS later on.

This lesson deals with the basic concepts of the entity-relationship model⁵.

The treatment of this model throughout this lesson is as follows:

- a) First, the students have to be beware of the importance of the Entity-Relationship diagrams in database design as a way of modelling the reality.
- b) The main constructs of an ER diagram are covered next: entity-types, attributes and relationships⁶.
- c) Next, the meaning of maximum cardinality is studied as well as the way of representing the different types of binary relationships: one-to-one, one-to-many, and many-to-many relationships.
- d) Later on, the idea of minimum cardinality and its representation is also discussed, learning the meaning of partial participation and existence constraint too.

It's important to say that there are many variations in use to represent the above mentioned concepts. In this lesson, the nomenclature proposed by Chen has been chosen but the same ER diagrams might be represented using another.

Approach

We are, probably, in front of the most creative lesson.

At the end of it, the students should be able to draw ER diagrams based on a simple given scenario. Like most disciplines, ER is best learned by doing lots of exercises. That is the reason why the following pages provide a number of study cases sorted by difficulty so that they can practise this technique.

⁵ The Enhanced ER (EER) is not covered here.

⁶ Because of the difficulty of the task, all ER diagrams of the activities included in this lesson involve binary relationships only.

Note that there is no a single right answer to most of activities in this lesson although some answers may be better than others. Taking this into account, it is recommended the students try to resolve the activities individually for which the teacher should give them enough time to think and give an answer⁷. Later, the possible solutions to the exercise should be discussed with the whole class.

It is important to say that, because of the difficulty of the task, all ER diagrams of the activities included in this lesson involve only binary relationships.

⁷ Depending of the difficulty of the activity, the needed time to solve each can be different. The teacher must assign the time that thinks appropriate for each case.

Activities

SCENARIO 1

COLLEGE

We wish to create a database for our college, considering the following information.

Each student enrolls in a series of subjects in which they will get a mark. Many students can enroll in a subject and a student can enroll in the same subject several times.

Each subject is taught by one teacher and a teacher can teach different subjects. Each teacher belongs to a department.

We want to store each student's ID, name, family name, address, city, telephone and marital status. The marital status of a student can be one of the following:

S: single

M: married

W: widower/ widow

D: divorced

For each of the subjects we will keep a code, their name and hours.

For each teacher we know their ID, university degree that they have, name and surname.

We know the code and name of each of the departments.



ACTIVITY 3.1 – Identifying Entities and Attributes

objective

In these activity the students have to identify the entities and its attributes.

The problem domain is fairly clear in this activity at this point, so the main objective is them to difference between entities and attributes of these entities.

Identify the entities and its attributes for the information system given in the SCENARIO 1.

Give some examples of entity instances for each of these activities.

suggested answer

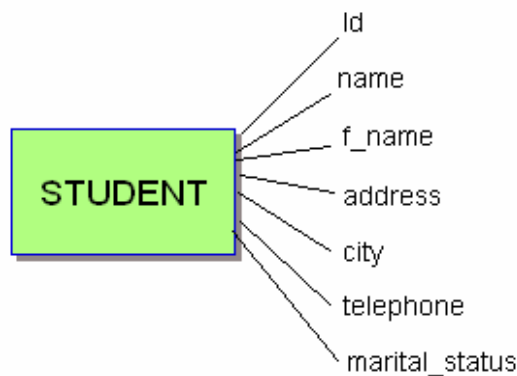
There are, clearly, four entity-types:

- STUDENT
- SUBJECT
- DEPARTMENT
- TEACHER

The attributes for each of these entities are also listed.

So, the final entities are as follows:

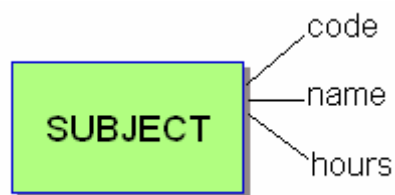
ENTITY-TYPE STUDENT



The information about the marital status is tricky in the statement of the problem; although it gives several possible values for this attribute, that doesn't have to be represented in the ER diagrama. In short, the statement says that every student has a marital status but it doesn't matter at this point which are the possible values for it. The only objective of it is to mislead the students.

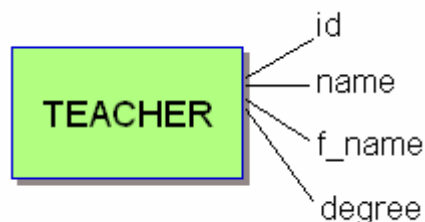
Entity instances for this subject would be the different students in the class.

ENTITY-TYPE SUBJECT



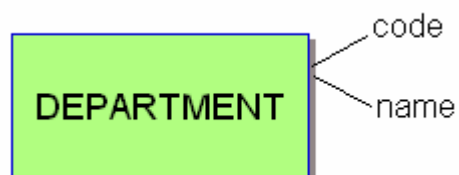
Entity instances of SUBJECT could include any subject that it taught in the college: C3 – database analysis and design..., C1- Operativng Systems...

ENTITY-TYPE TEACHER



The information about the teachers working for the college would be entity instances of this entity type: Xavier Morera, IT engineering..., Bernat Huertas, Maths, etc.

ENTITY-TYPE DEPARTMENT



As examples of departments (entity instances) we could name the different

areas in the college: Computing, Business and Administration, ...



Extra information for the teacher...

There are some attributes of different entity-types that have the same name (for instance the attribute "name"). For the time being that's not a problem but it will be when the ER diagrams are converted to their correspondent relations, since some of these attributes will be in the same relation and we'll need use different names for each to differentiate between them (teacher_name, student_name, etc. or t_name and s_name, etc.).



ACTIVITY 3.2 – Choosing Identifiers

objective

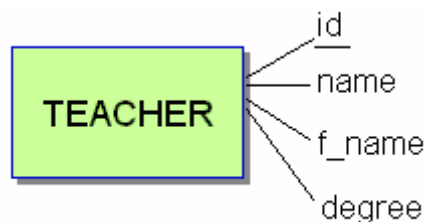
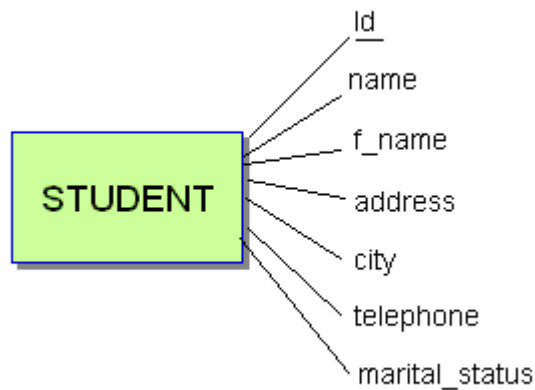
The students must be able to understand the meaning and importance of identifier. They must be able to select one or more attributes that difference each entity instance from the rest.

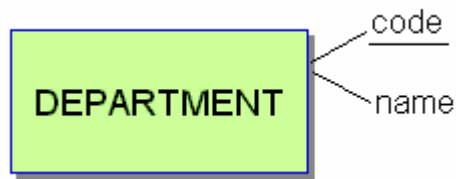
They have to be warned that the identifier can be made up for more than one attribute depending on the reality to be represented.

For each entity in the previous activity based on the SCENARIO 1, underline the attribute/s that will make up its dentifier.

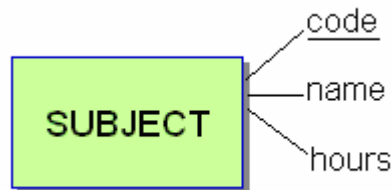
suggested answers

Most of identifiers for the previous activity are obvious:





However, there's a trick in the SUBJECT entity-type:



By choosing the code as identifier we are differentiating the subjects from some specific technical studies (DAI ASI, etc.) only.

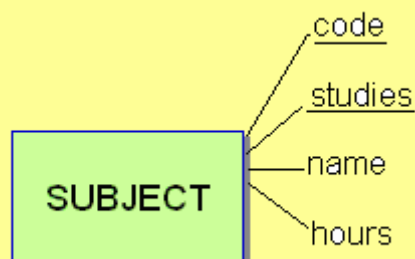


Extra information for the teacher...

It will be essential to mark out clearly the boundaries of our reality in order to represent it.

If we want to store information of the subjects from different technical studies, this last solution is not good since we can't differentiate subjects from different studies that have the same code. For instance, if we're told that the subject code is C3 we don't know yet what subject are we referring to: C3 of ASI or C3 of DAI??

To show this, we need the identifier to be made up of another attribute that shows the information of the technical studies, as follows:



That activity can also be used to introduce the meaning of weak entity.



ACTIVITY 3.3 – Identifying Relationships

objective

The students must be able to locate the existing relationships among the entity-types.

Given the SCENARIO1, locate the relationships among the entities.

Read these relationships in both directions, that’s to say use the active and the passive voices.

suggested answers

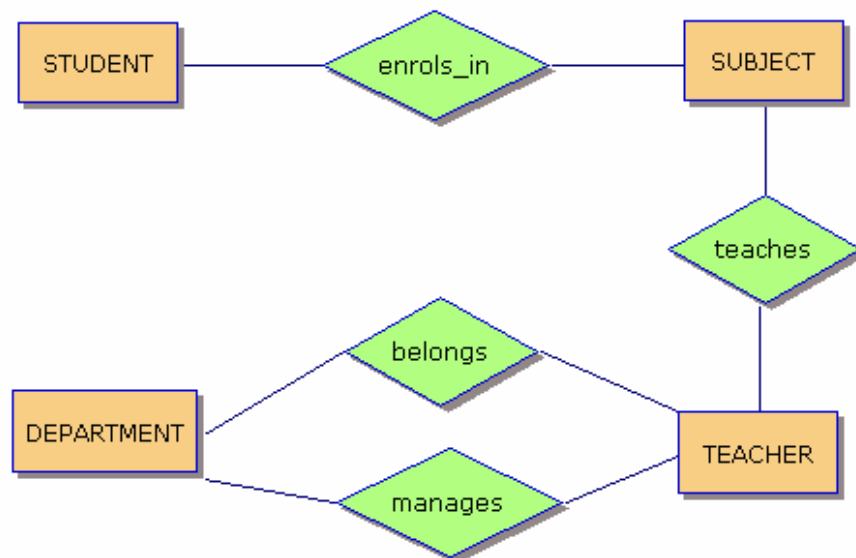
According to the statement, three relationships can be clearly found:

1. *A student enrolls in subjects./ In a subject are enrolled students.*
2. *A teacher teaches subjects./ A subject is taught by teachers.*
3. *A teacher belongs to a department./ In a department there are teachers.*

Other relationships that, although is not listed in the statement, may also be represented is:

4. *A teacher manages a department.*

For the moment, the ER diagram would be like this:



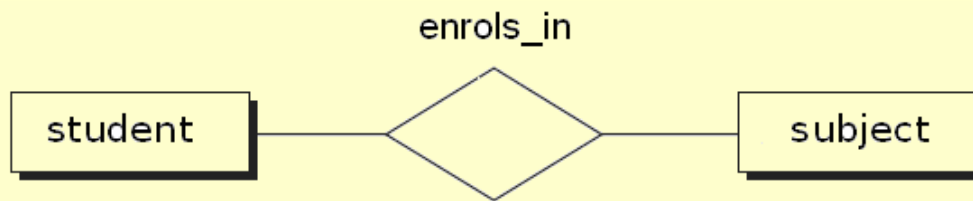


ACTIVITY 3.4 – Attributes and Relationships

objective

The students must understand when it is necessary to include an attribute in a relationship.

Following the relativity related in the SCENARIO1, imagine that now we also want to know the mark that each student has got for each subject he's enrolled in.



An attribute is needed to represent this information. Where should we put it?

suggested answer

The attribute must be put in the relationships as it is explained in the following slides.



ACTIVITY 3.5 – The Vendors Database

objective

To know how the ERD changes its meaning depending of the position of the attributes.

You wish to create a database for a group of vendors and the pieces they supply.

Every vendor will have a name, address, phone number, and a vendor number.

Pieces will have a piece code, description, and price.

suggested answer

The answer is given in the following slides:

- If *price* is thought to be an attribute of the PIECES entity, it will mean that all vendors sell that price at the same price.
- The best solution is to consider that the price of each piece depends on each the vendor, therefore the *price* should be an attribute of the relationship.



ACTIVITY 3.6 – Maximum Cardinality

objective

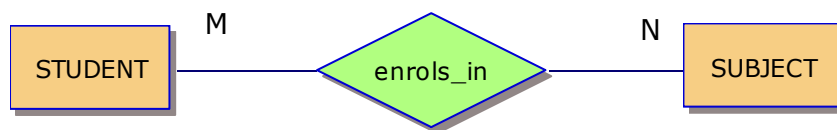
The students will put into practice the concept learnt and must try to identify the maximum cardinality of different relationships.

Represent the maximum cardinality for each relationship identified in the reality given in the SCENARIO1.

suggested answer

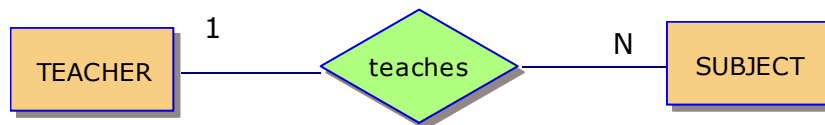
1. A student *enrols in* subjects.

Each student enrols **in a series of** subjects in which they will get a mark. **Many** students can enrol in a subject and a student can enrol in the same subject several times.



2. A teacher *teaches* subjects.

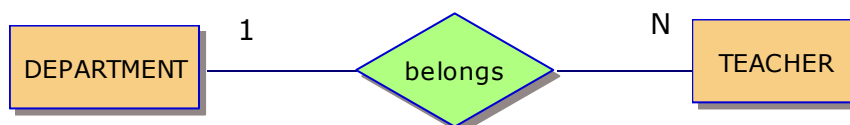
Each subject is taught by one teacher and a teacher can teach different subjects.



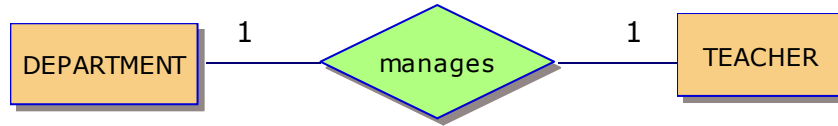
(*) In our college the reality is not like that.

3. A teacher *belongs to* a department.

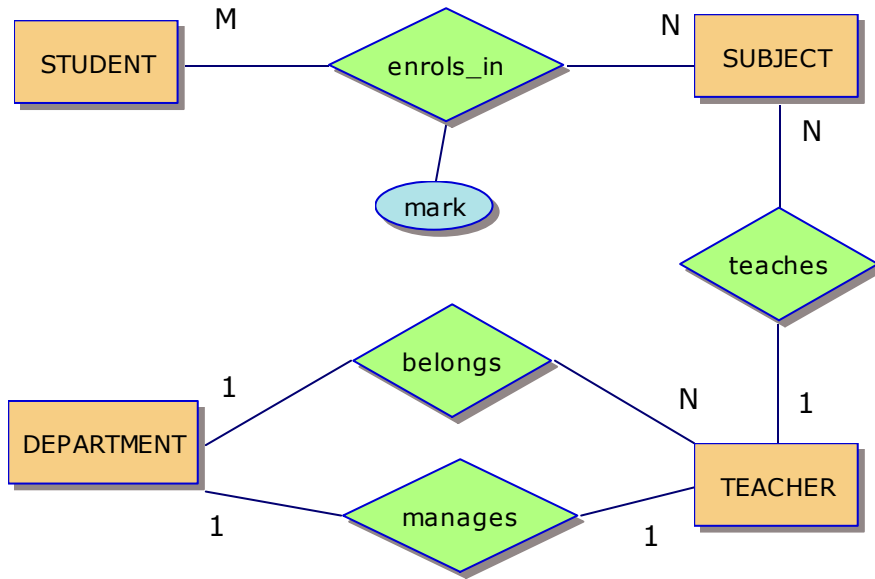
Each teacher belongs to a department.



4. A teacher *manages head of* a department.



So, the ER diagram so far, it's the next one:





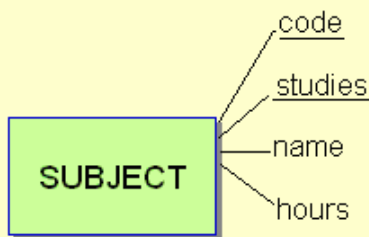
ACTIVITY 3.7 – Entity vs Attribute

objective

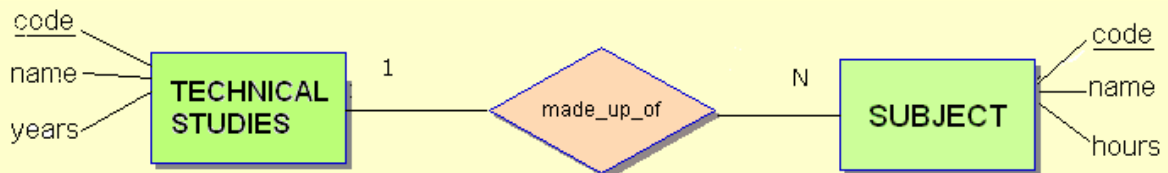
To understand the difference between identifying a property of an entity as an attribute and deciding on being another entity-type.

Explain the existing difference between these two solutions for the reality given in the SCENARIO1.

SOLUTION 1.



SOLUTION 2.



suggested answer

The decision of including *technical studies* as an attribute or as another entity-type will depend on whether we wish to keep information about this object or not.

With the first solution we know the code of the technical studies that each subject belongs to but we don't have more information about them. The second solution also says the studies of a specific subject, but also gives us more details about these studies.



ACTIVITY 3.8 – Minimum Cardinality

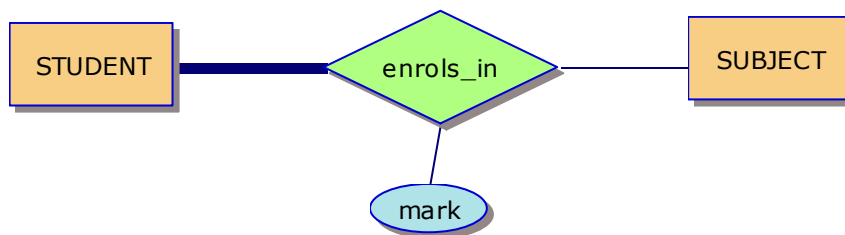
objective

In the same way that it was done for the maximum cardinality, at this point the students have to learn to identify and represent the minimum cardinalities for several relationships.

Following the reality given in the SCENARIO 1, represent now the minimum cardinality for each relationship.

suggested answer

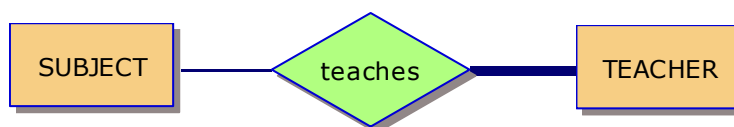
1. A student enrolls in subjects.



It's obvious that a subject can not have students; even there's no students that enrol in this subject, it will keep existing.

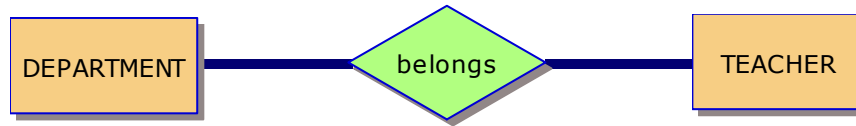
On the other hand, the minimum cardinality for the STUDENT entity depends on how we want our database to work. With a 1 minimum cardinality (total participation) we're saying that it's not possible to have a student in our database that isn't enrolled in any subject.

2. A teacher teaches subjects.

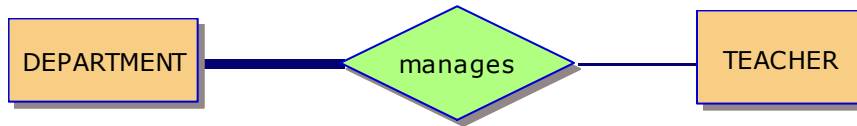


If previously we have decided that there can be a subject without students, it wouldn't make sense now to think that each subject has to have at least one teacher.

3. A teacher belongs to a department.



4. A teacher is head of a department.



Extra information for the teacher...

The students must be aware that to identify the minimum cardinalities is a more subjective process. Most of statements won't provide this information clearly, so it will be necessary to use the common sense to make a decision. In the real world the answer would be up to the client.



ACTIVITY 3.9 – Bank

objective

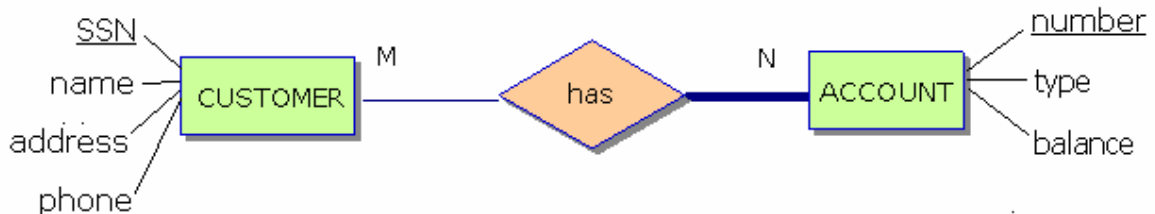
To keep working on the maximum and minimum cardinalities.

Let us design a database for a bank, including information about customers and their accounts. Information about customers includes their name, address, phone and Social Security Number. Accounts have numbers, types (e.g. savings, checking) and balances. We also need to record the customer or customers who own an account.

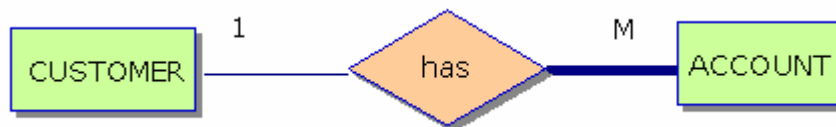
- Draw the ER diagram for this database.
- Change your diagram so that an account can have only one customer.
- Change your original diagram so that a customer can have a set of addresses (which include street, city and county) and a set of phones.

suggested answer

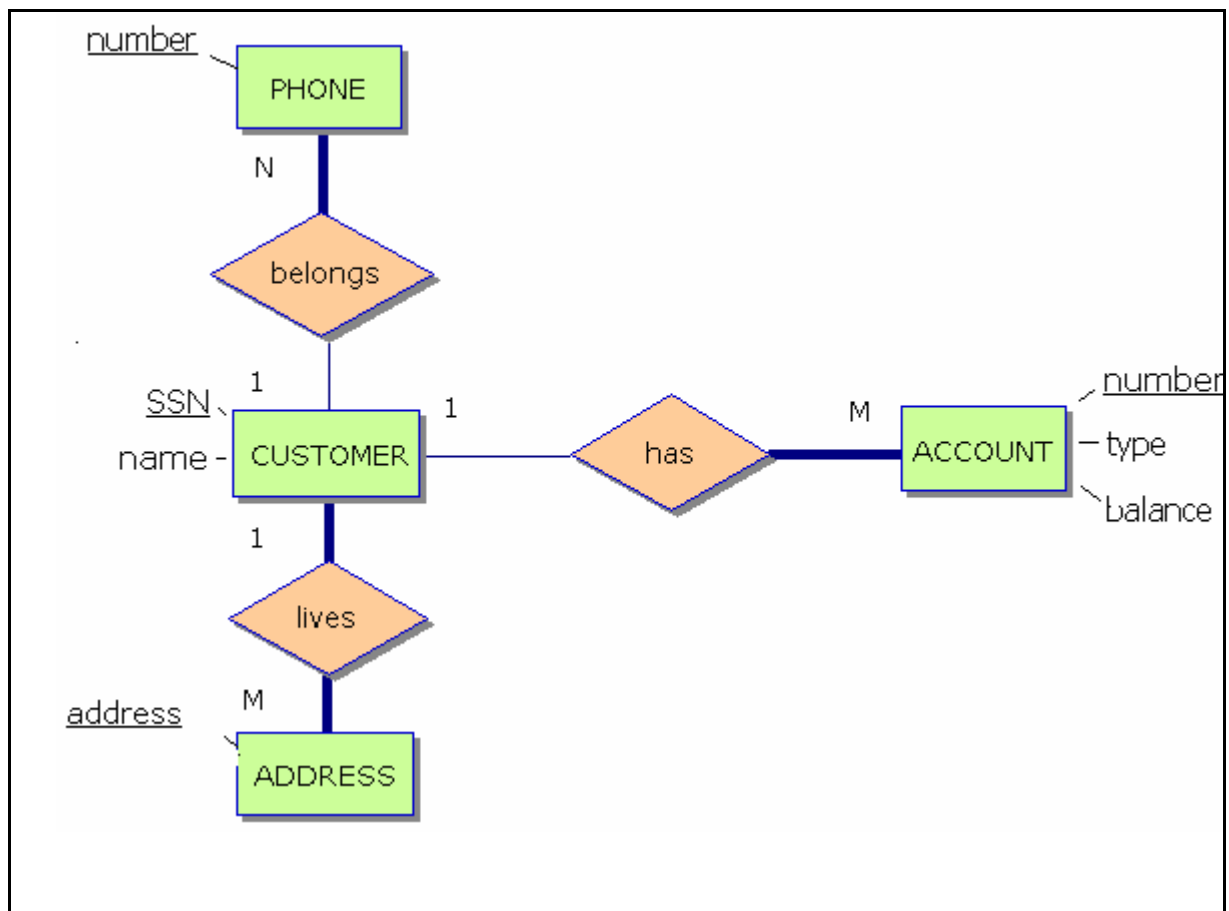
a)



b) The change affects to one maximum cardinality only:



c) Since attributes are only allowed to have atomic values, the only way there is for them to be able to take several values is them to turn into entity-types:



Extra information for the teacher...

The students must know that sometimes the cardinalities are not obvious and any answer is possible; it is necessary so, to understand the reality to be represented.



ACTIVITY 3.10 – Nursery

objective

At this point the objective is the students to be able to draw an ER diagram based on a given simple scenario.

We want to computerize the activities in a nursery.

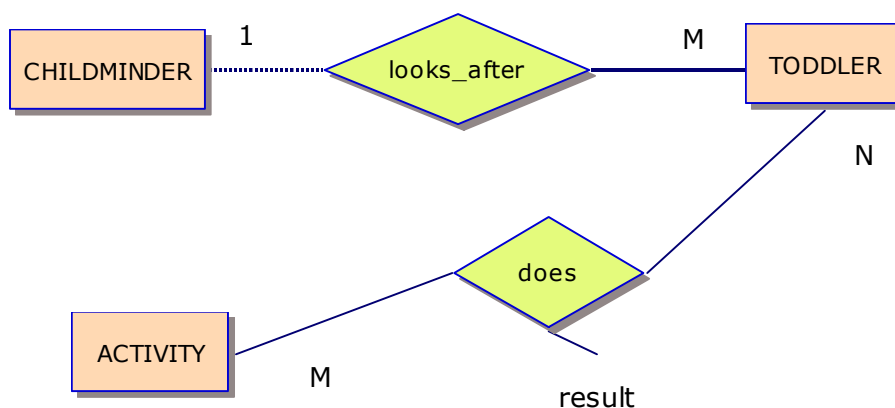
For each childminder we have information about their ID, forename and surname. Every toddler is assigned a unique code and data are also stored about their name, family name, date of birth and sex.

Each childminder takes care of a group of toddlers, whereas an only child is been looked after by an only childminder.

On the other hand, there are activities that the children do in the nursery from which a code and name is stored. Each child can do many activities and we want to keep the result that the children get on each one. It must been taken into account that every activity can be done by several children.

Draw the ER diagram for the case above.

suggested answer



Extra information for the teacher...

The first thing to be aware of when tackling an ER design is that it is not a mechanistic process: there is no fixed set of steps to proceed through that will

automatically yield the “correct” solution. In other words, it is an iterative process that requires successive passes through the design.



ACTIVITY 3.11 – Garage

objective

To draw an ER diagram based on a given simple scenario.

A garage wants to keep a historical record of the information about the repairs done to the cars that arrive there.

For each owner they store their ID and full name. For the cars they keep the number plate, the brand, model and type (1 = petrol, 2 = unleaded, 3=Diesel). In the garage the different kind of repairs that can be done are given a code. So, for each repair a unique code and a repair description are stored.

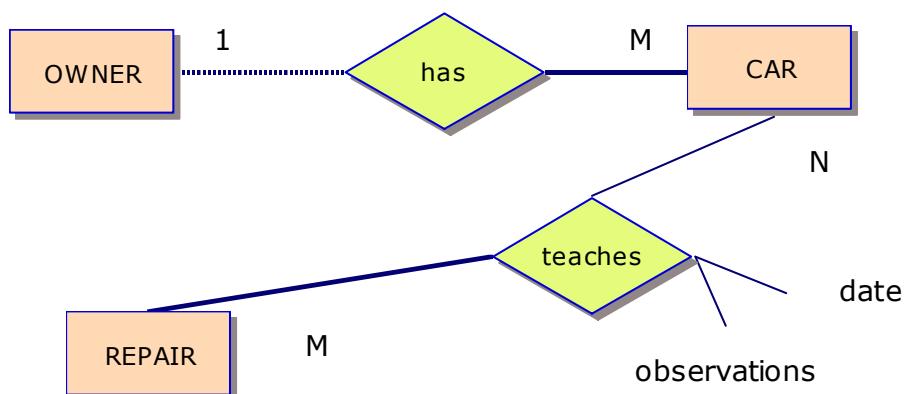
An owner can have many cars whereas a car belongs to an owner only.

A car can receive several repairs and a specific repair can be done to many cars.

Every time that a car is fixed, they store the starting date and a field with observations.

Draw the ER diagram for the information system above set out

suggested answer





ACTIVITY 3.12 – Model Agencies

objective

To draw an ER diagram based on a given simple scenario.

The company “Consulting S.A.” is asked to carry out a study on how successful a set of model agencies are.

Each agency has a team of models as part of its staff, so a model only works for a specific agency. Some companies contact the agencies asking for models to do an advertising campaign about its products. A company can ask for several models and the same model can take part in advertising campaigns for different companies.

They want to store the result or impact that the adverts made by each model have had on the consumers. This impact is represented by a number as follows:

- 1: excellent
- 2: very good
- 3: good
- 4: so-so
- 5: bad

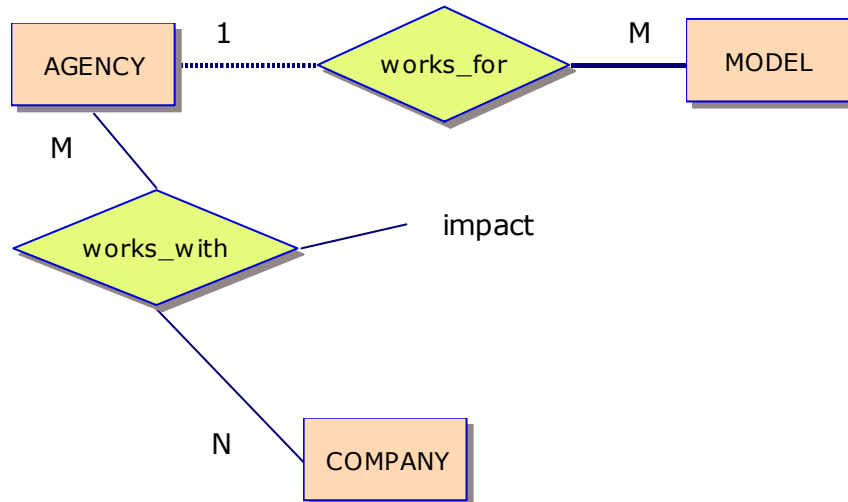
Each models agency has their CIF, name and the country where they are located.

For the companies they keep information about their CIF, company name, full name of the manager, and the field they belong to and which is represented by a letter (F: food sector, M: motor industry, C: cosmetics and fashion sector).

As regard to the models, they have their ID, name and surname, nationality and date of birth.

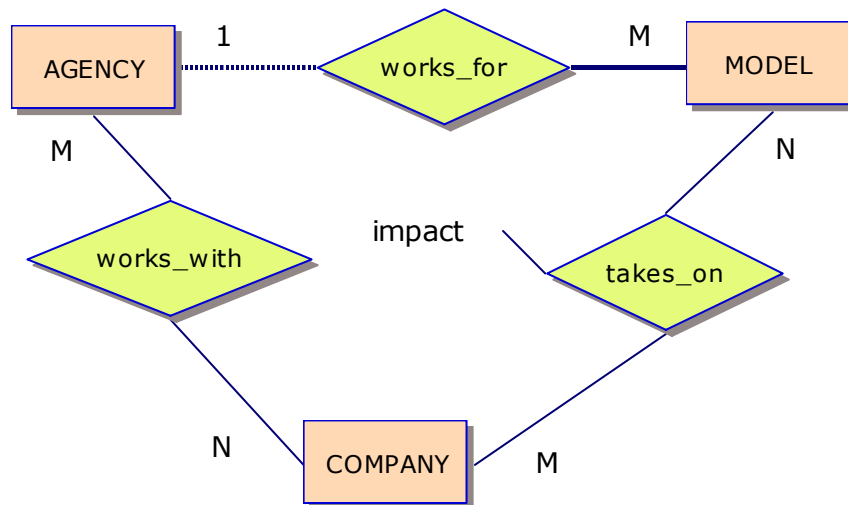
Draw the ER diagram for the information system above described.

suggested answer



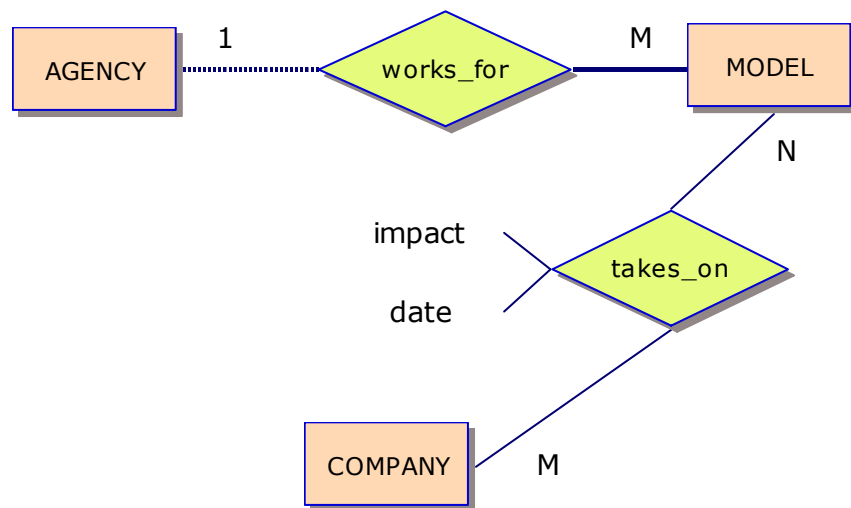
This solution isn't valid because we don't know what specific model did the advert.

SOLUTION 2



This solution is not correct either because there's redundancy of information: if we know the model that has been taken on the company to do an advert, we also know the agency that model works for.

SUGGESTED SOLUTION:



This solution gives us information about the model as well as the company that did the advert. Beside, it's liked to know the date when the advert took place as the same model can do several adverts for the same company in different times.



Extra information for the teacher...

That last case might be used to explain the notion of "historical record"



ACTIVITY 3.13 – *General elections*

objective

To draw an ER diagram based on a given simple scenario.

A company of statistics services makes surveys to know the way people interviewee intend to vote about the different presidential candidates.

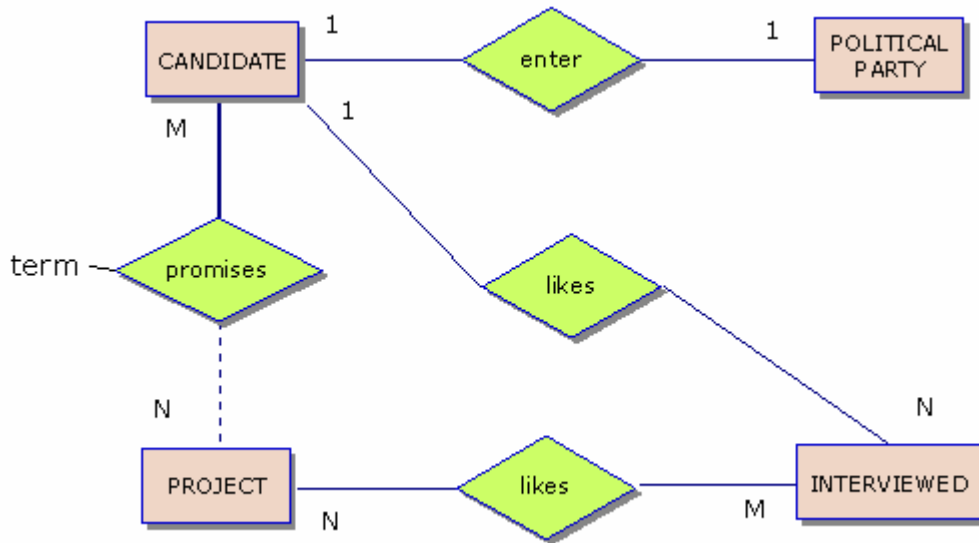
The known information about the candidates is their name, age and unique code. They also have information about the different political parties that take part in the elections such their name, founding year and political tendencies. Each party can only enter a candidate.

The interviewee are asked their age, cultural level (1: without an education, 2: primary education, 3: secondary education, 4: higher education) and have a code.

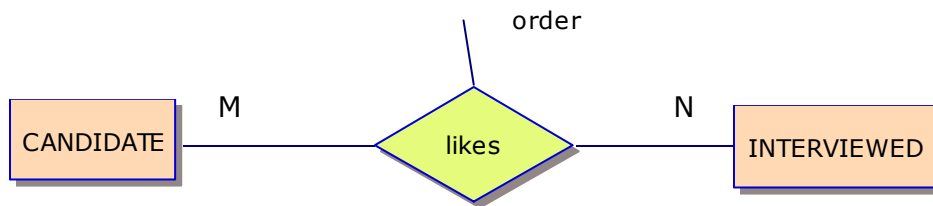
The company is also interested in knowing their taste about the different projects that make up the election manifesto of each candidate. So, it has information about the description of all projects (which sometimes are included in the election manifesto of different candidates), the approximate execution period and a code for each one.

Draw the ER diagram for the information system above stated.

suggested answer

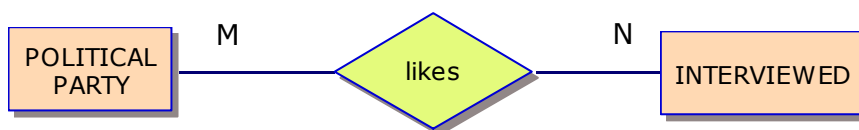


It's also possible to show their tastes about the candidates in order of preference doing some changes in the relationship between candidate and interviewee:



This idea can also be used to sort the preferences about the different projects.

A relationship between political party and interviewee could also be added as follows:





ACTIVITY 3.14 – DVD Hire Shop

objective

To draw an ER diagram based on a given simple scenario.

A DVD rental shop requires a database system to assist in managing the hire and return of DVDs.

The database maintains details of customers, films, DVD copies of films and rental transactions. The system must record transactions (i.e. when a DVD is rented and when it is returned), calculate how much rental cost and determine which rentals are overdue. Additionally it must record the arrival in stock of new DVDs and the disposal of obsolete and faulty DVDs. DVDs are grouped into rental price categories; the category of a DVD will normally change during its lifetime in the shop – it will become cheaper as it ages. The cost of rental is based on the rental price times the number of days it has been rented. The standard period of rental is two days; additional days are charged at a reduced rate. A DVD is considered overdue if it is not returned within one week.

Reports are required showing rental histories for selected clients, usage reports for selected DVDs and popularity ranking for current films available for rent.

Draw the ER diagram for the information system above set out.

suggested answer

Summary

This chapter has provided the fundamentals of ER modeling of data.

- The Entity-Relationship model in database design it's used to develop an initial database design, thus the conceptual design can be considered the first phase of database design.
- The ER model provides a conceptual representation of data and was introduced by Peter Chen in 1976.
- An ER model is usually expressed in the form of ER diagrams.
- ER diagrams (ERD) are used to model the nature of data within the domain of a database application.
- The ERD provide a graphical design wich can be used to derive a set of relational tables that model the data of the application system.
- The basic constructs of ER model are entity types, relationships, and attributes.
- Entities are identifiable things or concepts that are relevant to the application. These are modelled in the ERD by boxes.
- Entity-types have associated attributes (properties).
- Relationships express associations between entities. They are drawn as lines connecting the related entity boxes.
- According to their maximum cardinalities, the types of bynary relationships are one-to-one, one-to-many, and many-to-many relationship. Most relationships are binary.
- With regard as their mininum cardinality, relationships can be mandatory (with an existence conrstraint) or optional, also called full and partial participation.
- Multi-valued attributes and time-varying attributes can be represented by an additional entity-type related to the base entity-type in a one-to-many relationship.

Glossary III

English-catalan vocabulary

English	Meaning
Among	Entre (més de dos)
At least	Al menys
At most	Com a molt
Grade	Nota
Guideline	Guia
Later on	Més endavant
Mark	Nota
Number plate	Número de matrícula (de cotxe)
To be involved in	Estar implicat en
To denote	Denotar, indicar
To draw	Dibuixar
To enrol in	Matricular-se en
To extend	Ampliar
To model	Modelar
To remove	Esborrar, eliminar
To take into account	Tenir en compte
To take up	Prendre, acceptar
To underline	Subratllar

Key Vocabulary

At the end of this lesson the students must be able to recognise and understand the meaning of the following words:

ATTRIBUTE

CARDINALITY

CONNECTIVITY

DEGREE

DESCRIPTOR

ENTITY INSTANCE

ENTITY OCCURRENCE

ENTITY-RELATIONSHIP (ER)

ENTITY-TYPE

ER DIAGRAM (ERD)

ER MODEL (ERM)

EXISTENCE CONSTRAINT

IDENTIFIER

KEY

MANDATORY RELATIONSHIP

MAXIMUM CARDINALITY

MINIMUM CARDINALITY

OPTIONAL RELATIONSHIP

PARTIAL PARTICIPATION

TOTAL PARTICIPATION

Lesson Plan	PART I - DATABASE DESIGN		
Lesson 3. Entity-Relationship Model			
TIMING	14 hours	LEVEL	1st CFGS DAI/ASI
OBJECTIVES	CONTENT	<ul style="list-style-type: none"> - Entity-Relationship diagrams (ERD) - Entity-types and entity instances, meaning and representation - Attributes, meaning and representation - Relationships, meaning and representation - Descriptor and key attributes - Identifier, meaning and representation - Relationships, meaning and representation - Degree of a relationship - Cardinality, meaning and types - Maximum cardinality, types and representation - Minimum cardinality, representation - Total participation (existence constraint) and partial participation - Historical records 	
	COGNITION	<ul style="list-style-type: none"> - To understand the principles of the ER model to be able to draw simple ER diagrams. 	
	CULTURE	<ul style="list-style-type: none"> - Importance of modelling 	
	COMMUNICATION	Language for learning:	

	<ul style="list-style-type: none"> - To discuss and ask about the concepts - To justify their answers and reasoning <p>Language of learning:</p> <ul style="list-style-type: none"> - Language structures needed to complete the tasks <p>Language trough learning:</p> <ul style="list-style-type: none"> - Language that comes up when carrying out the different activities 																										
<p>LEARNING OUTCOMES</p>	<p>At the end of this lesson, students will be able to...</p> <ul style="list-style-type: none"> - Understand the difference between an entity-type and an entity instance - Show how the concepts of entity-type and "relationship are represented in ER modelling - Understand the term cardinality and the possible cardinality values - Explain the meaning of entity-constraint, partial participation and optionality - Draw simple ER diagrams based on a given scenario 																										
<p>RESOURCES AND TIMING</p>	<p>Lesson in power point (<i>P1.L3.ERM.ppt</i>)</p> <table border="1" data-bbox="488 1223 1345 1765"> <tr> <th colspan="2">Lesson 3.- Entity-Relationship Model (14 hours teaching + activities)</th> </tr> <tr> <td>Activity 3.1</td> <td>Identifying Entities and Attributes</td> </tr> <tr> <td>Activity 3.2</td> <td>Choosing Identifiyers</td> </tr> <tr> <td>Activity 3.3</td> <td>Identifying relationships</td> </tr> <tr> <td>Activity 3.4</td> <td>Attributes and Relationships</td> </tr> <tr> <td>Activity 3.5</td> <td>The Vendors Database</td> </tr> <tr> <td>Activity 3.6</td> <td>Maximum Cardinality</td> </tr> <tr> <td>Activity 3.7</td> <td>Entities vs Attributtes</td> </tr> <tr> <td>Activity 3.8</td> <td>Minimum Cardinality</td> </tr> <tr> <td>Activity 3.9</td> <td>Bank</td> </tr> <tr> <td>Activity 3.10</td> <td>Nusery</td> </tr> <tr> <td>Activity 3.11</td> <td>Garage</td> </tr> <tr> <td>Activity 3.12</td> <td>Model Agencies</td> </tr> </table>	Lesson 3.- Entity-Relationship Model (14 hours teaching + activities)		Activity 3.1	Identifying Entities and Attributes	Activity 3.2	Choosing Identifiyers	Activity 3.3	Identifying relationships	Activity 3.4	Attributes and Relationships	Activity 3.5	The Vendors Database	Activity 3.6	Maximum Cardinality	Activity 3.7	Entities vs Attributtes	Activity 3.8	Minimum Cardinality	Activity 3.9	Bank	Activity 3.10	Nusery	Activity 3.11	Garage	Activity 3.12	Model Agencies
Lesson 3.- Entity-Relationship Model (14 hours teaching + activities)																											
Activity 3.1	Identifying Entities and Attributes																										
Activity 3.2	Choosing Identifiyers																										
Activity 3.3	Identifying relationships																										
Activity 3.4	Attributes and Relationships																										
Activity 3.5	The Vendors Database																										
Activity 3.6	Maximum Cardinality																										
Activity 3.7	Entities vs Attributtes																										
Activity 3.8	Minimum Cardinality																										
Activity 3.9	Bank																										
Activity 3.10	Nusery																										
Activity 3.11	Garage																										
Activity 3.12	Model Agencies																										

Part 1. Lesson 4

Transformation into Tables

Introduction

The logical design can be considered the second phase of database design.

In this phase, the ER diagrams produced in the conceptual design are used to develop a set of relational tables that represent the application data.

In this lesson we will study the topic of deriving a relational model from the previous ER design. The end result of this process should be the design of a set of interconnected relational database tables which can then physically implemented on a chosen database product.

Approach

For the purposes of illustration, we will use two simple examples. To explain the conversion of one-to-many and many-to-many relationships will use an example based on a scenario in which Engineers are assigned to work on Projects

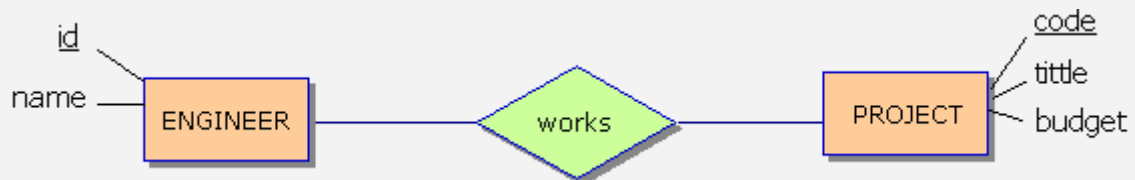
To understand how to derive a table design from a one-to-one relationship, we will use an example based on the marriages between a man and a woman.

In both cases, various alternatives for the minimum cardinality will be examined as part of the process explanation.

Activities

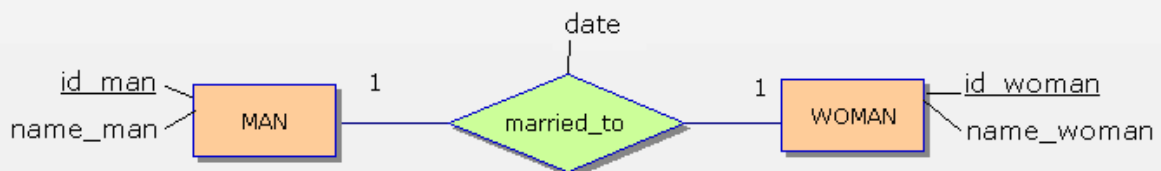
SCENARIO 2 PROJECTS

Imagine that in a company *engineers* are assigned to work on *projects*. Without taking into account the cardinalities, the corresponding ER diagram for this scenario is showed bellow:



SCENARIO 3 MARRIAGES

We wish to keep information about the marriages that have taken place in our city. For this, we start of the ER diagram bellow.





ACTIVITY 4.1 – Transforming Entities

objective

To transform entity-types into relations/tables.

Given the SCENARIO 2

- a) Transform the the entity-types into relations.
- b) Draw corresponding tables for the obtained relations and fill them with instances

suggested answer

The attributes of the entity set become the column attributes of the table.

ENGINEERS (id, name)

PROJECTS (code, title, budget)

It's necessary to adopt one or more columns as a primary key for each relation. In this case, the choice is obvious and the primary key is the same as the identifiers.

With regard to the tables, the essential requirement, of course, is that the primary key must have a unique value for each row of the table:

ENGINEERS

<u>id</u>	name
12	Kelly
34	Ross
56	Simth

PROJECTS

<u>code</u>	title	budget
S03	Mercury	£5000
X99	Venus	£10000
Z22	Apollo	£2300



Extra information for the teacher...

Singular is usually used to describe entity-types whereas relations/tables are normally named in plural.

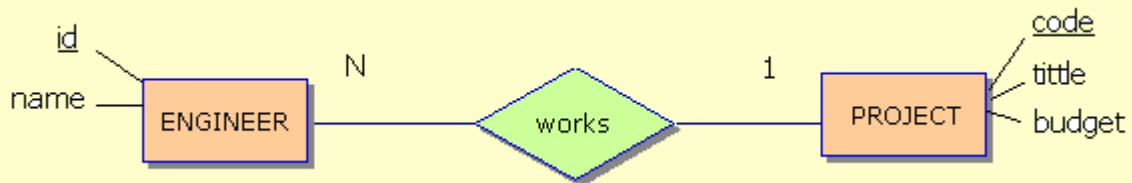


ACTIVITY 4.2 – Transforming 1:N Relationships

objective

To convert one-to-many relationships into relations/tables.

Let's consider now that the maximum cardinality for the case in SCENARIO 2 is one-to-many. The ER will be as follows:



- Incorporate the ER relationships into the relations and tables.
- Consider separately the different cases of minimum cardinality.

suggested answer

Since any one engineer is only associated with one project, it's possible to adopt a foreign key of project code for the engineer relation:

ENGINEERS (id, name, code)

Foreign key: code → PROJECTS

PROJECTS (code, title, budget)

ENGINEERS

<u>id</u>	name	code
12	Kelly	X99
29	Brown	S03
34	Ross	S03
56	Smith	Z22

PROJECTS

<u>code</u>	title	budget
S03	Mercury	£5000

	X99	Venus	£10000	
	Z22	Apollo	£2300	



Extra information for the teacher...

Notice that for any tables linked by a one-to-many relationship, the foreign key will be in the relation/table at the "many" end, and the referenced primary key at the "one" end.



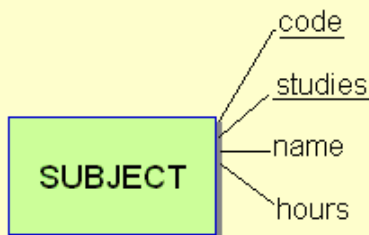
ACTIVITY 4.3 – Entity vs Attribute

objective

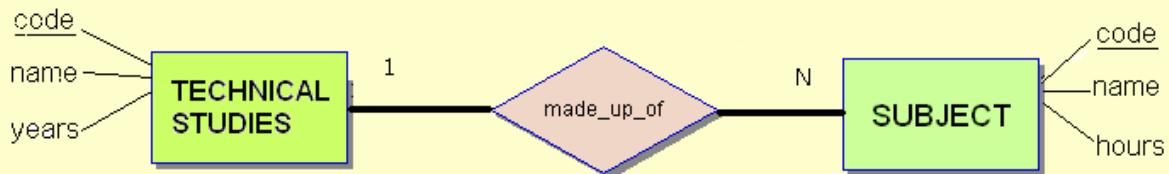
To understand the difference between treating something as an entity or an attribute.

Given the two solutions for the reality of the SCENARIO1 set in the activity 3.7:

SOLUTION 1.



SOLUTION 2.



- Obtain the corresponding relations for each case.
- Draw the corresponding tables with examples to see the difference. Fill them with instances and, in case there's some aspect of the reality that can't be represented, make up examples to prove it.

suggested answer

SOLUTION 1.

The result is a single relation:

SUBJECTS (code, studies, name, hours)

Which is equivalent to the following table:

SUBJECTS

<u>code</u>	<u>studies</u>	name	hours
C3	DAI	Database analysis	90
C3	ASI	Applications introduction	180
C4	DAI	Applications analysis	240

SOLUTION 2.

1. In the first stage, the entity-types are transformed:

SUBJECTS (code, name, hours)

STUDIES (code, name, years)

2. We apply the process to the one-to-many relationship:

SUBJECTS (subject_code, subject_name, subject_hours, **studies_code**)

NNV: studies_code

STUDIES (studies_code, studies_name, studies_years)

(* It can't be represented that all technical studies are made up of at least one subject.

The final tables would be:

SUBJECTS

<u>subject_code</u>	subject_name	subject_hours	code_studies
C3	Applications introduction	180	ASI
C4	Applications analysis	240	ASI

STUDIES

<u>studies_code</u>	studies_name	studies_years
DAI	Applications developpement	2
ASI	Systems administration	2



Extra information for the teacher...

Pay attention that the names of some attributes must be changed to avoid confusion.



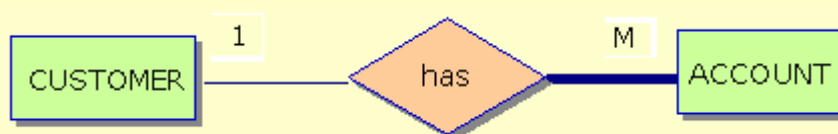
ACTIVITY 4.4 – Bank

objective

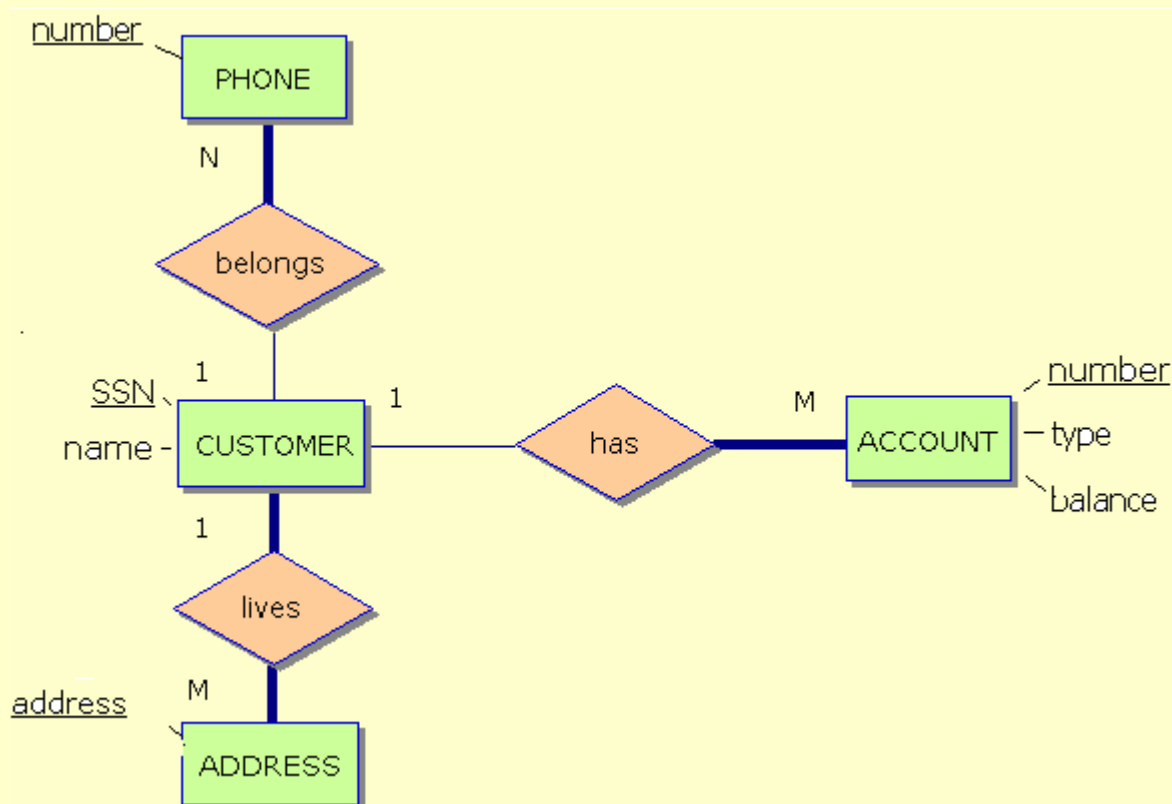
To practise the transformation of simple ER diagrams based on one-to-many relationships.

Recall the ER diagrams obtained for the scenario given in the activity 3.9

- a) The ER diagram for this database assuming that an account belongs to one customer only, is:



- b) The corresponding diagram so that a customer can have a set of addresses (which include number house and street) and a set of phones is:



- Transform the ER diagrams above into relations and see the difference among them.

- Draw the tables and give examples.

suggested answer

a)

CUSTOMERS (SSN, name, phone, address)

ACCOUNTS (number, type, balance, SSN)

Foreign key: SSN → CUSTOMERS

NNV: SSN

That can be visualised in the following example:

CUSTOMERS

<u>SSN</u>	name	phone	address
20	Jackson	444	3 Earnbank
92	Stephan	111	25 Gordon
55	Scully	333	8 Sachiehall

ACCOUNTS

<u>number</u>	type	balance	SSN
12345	savings	£13000	55
68000	current	£380	20
20220	savings	£6491	55

Notice that there's a customer without account but all account have an owner.

b)

CUSTOMERS (SSN, name)

ACCOUNTS (number, type, balance)

Foreign key: SSN → CUSTOMERS

NNV: SSN

PHONES (phone_number, SSN)

Foreign key: SSN → CUSTOMERS

NNV: SSN

ADDRESSES (address, SSN)

Foreign key: SSN → CUSTOMERS

NNV: SSN

ACCOUNTS (number, type, balance, SSN)

Foreign key: SSN → CUSTOMER

(*) It can't be represented that it there can't be a customer without address.

CUSTOMERS

<u>SSN</u>	name
20	Jackson
92	Stephan
55	Scully

ADDRESSES

<u>address</u>	SSN
3 Earnbank	20
8 Sachiehall	20
25 Gordon	92

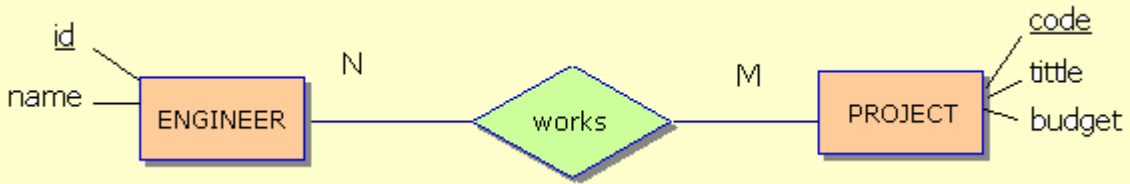
Jackson has two addresses, wich is not a problem, but Scully hasn't any.



ACTIVITY 4.5 – Transforming N:M Relationships

Objective	To transform many-to-many relationships into relations/tables.
------------------	--

Resume the case set in the activity 4.1 and imagine that an engineer can work on several projects.



- a) Make the appropriate changes over the resulting relations and tables.
- b) Consider separately the different cases of minimum cardinality.

suggested answer

In a many-to-many relationship we have no choice: we must form a third table to represent the relationship.

ENGINEERS (id, name, code)

Foreign key: code → PROJECTS

PROJECTS (code, title, budget)

ASSIGNAMENT (id, code)

Foreign key: id → ENGINEERS

Foreign key: code → PROJECTS

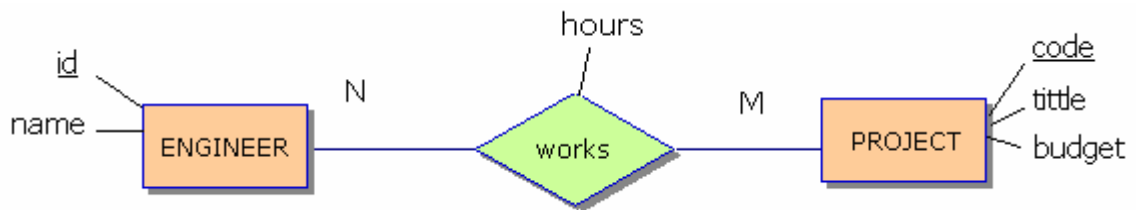
ASSIGNAMENTS

<u>id</u>	<u>code</u>
12	S03
29	X99

29	S03
34	X99
34	S03

However, this relation/table permits projects without engineers and engineers without project.

If there was an attribute in the relationship, this will be included in the same table as shown:



ASSIGNMENTS

<u>id</u>	<u>code</u>	hours
12	S03	200
29	X99	35
29	S03	150
34	X99	28
34	S03	90



ACTIVITY 4.6 – Nursery

Objective

To practise the transformation of simple ER diagrams based on one-to-many and many-to-many relationships.

Given the scenario of the activity 3.10:

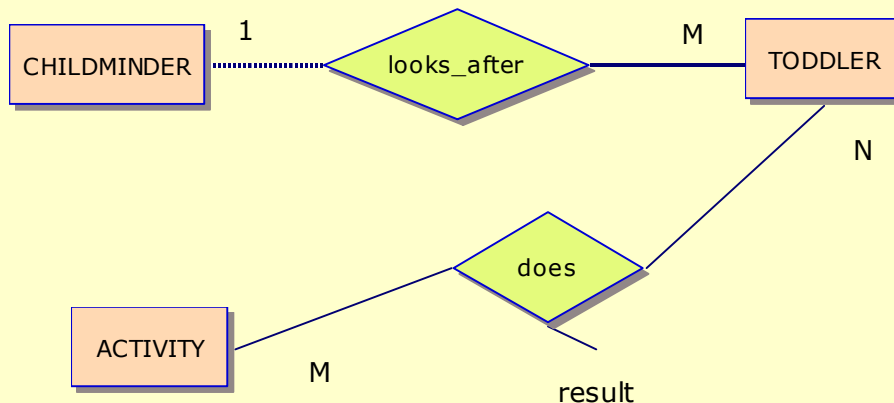
We want to computerize the activities in a nursery.

For each childminder we have information about their ID, forename and surname. Every toddler is assigned a unique code and data are also stored about their name, family name, date of birth and sex.

Each childminder takes care of a group of toddlers, whereas an only child is been looked after by an only childminder.

On the other hand, there are activities that the children do in the nursery from which a code and name is stored. Each child can do many activities and we want to keep the result that the children get on each one. It must be taken into account that every activity can be done by several children.

The ER diagram for the case above is:



Tranform the ER diagram into the corresponding relations.

suggested answer

CHILDMINDERS (id, forename, surname)

TODDLERS (code, name, fname, date_birth, sex, chilmind_id)

Alternate key: childminder_id → CHILDMINDERS

NNV: childminder

ACTIVITIES (code, name)

ACTIVITIES-TODDLERS (activity_code, toddler_code, result)

Alternate key: activity_code → ACTIVITIES

Alternate key:toddler_code → ACTIVITIES



Extra information for the teacher...

We will normally name relations using the label used in the relationship in the ER diagram. However, when this name is not clear enough, it is quite common to use the name of all the entity-types that are linked.

CARS (registration_number, brand, model, type, id_owner)

Alternate key::id_owner → OWNERS

NNV: id_owner

REPAIRS (code, description)

CARS-REPAIRS (registration_number, repair_code, date, observation)

This solutions has a problem though:

REPAIRS

<u>code</u>	description
R1	check breaks
R2	change spark plugs
R3	change tires

CARS-REPAIRS

<u>registration_number</u>	<u>repair_code</u>	date	observation
2020	R3	2009	
1192	R3	2009	very old
2020	R3	2009	

According to this, the third row is not correct what means that the same car cannot have the same kind of repair more than once.

The solution would be include the date as part of the primary key, as shown:

<u>registration_number</u>	<u>repair_code</u>	<u>date</u>	observation
2020	R3	2009	
1192	R3	2009	very old
2020	R3	2009	



Extra information for the teacher...

When keeping a history the date will have to be part of the primary key.



ACTIVITY 4.8 – Model Agencies

objective

To practise the transformation of simple ER diagrams based on one-to-many and many-to-many relationships.

To understand the meaning of redundant relationships.

Given the scenario of the activity 3.12:

The company “Consulting S.A.” is asked to carry out a study on how successful a set of model agencies are.

Each agency has a team of models as part of its staff, so a model only works for a specific agency. Some companies contact the agencies asking for models to do an advertising campaign about its products. A company can ask for several models and the same model can take part in advertising campaigns for different companies.

They want to store the result or impact that the adverts made by each model have had on the consumers. This impact is represented by a number as follows: 1: excellent, 2: very good, 3: good, 4: so-so, 5: bad.

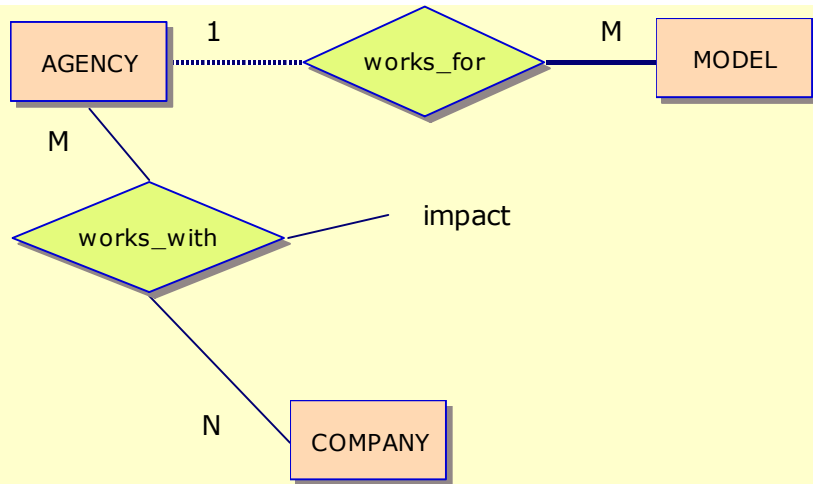
Each models agency has their CIF, name and the country where they are located.

For the companies they keep information about their CIF, company name, full name of the manager, and the field they belong to and which is represented by a letter (F: food sector, M: motor industry, C: cosmetics and fashion sector).

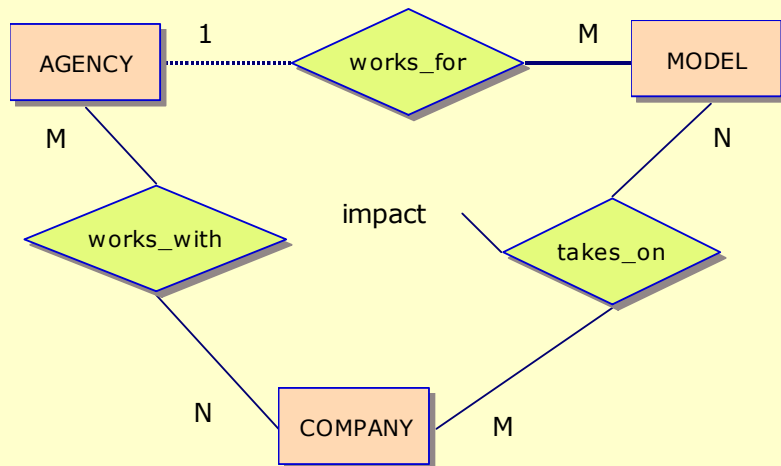
As regard to the models, they have their ID, name and surname, nationality and date of birth.

The possible ER diagram for the information system above described are:

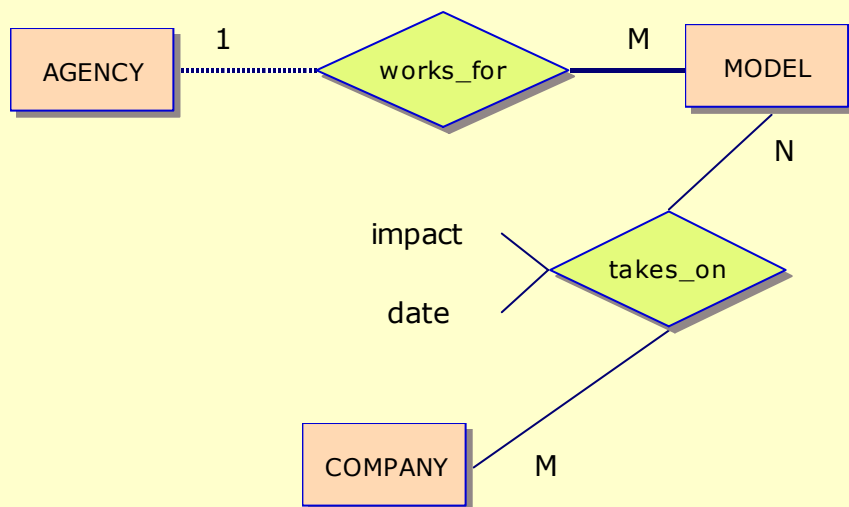
SOLUTION 1



SOLUTION 2



SOLUTION 3



Transform the three solutions above into relations and compare the results.

suggested answer

In all cases we will have the following relations:

AGENCIES (CIF, name, country)

MODELS (id, name, nationality, date_birth, Agency_cif)

Agency_cif → AGENCIES

NNV: Agency_cif

COMPANIES (CIF, name, manager, sector)

AGENCIES

<u>cif</u>	agency
10	Omarish
22	Bcn fashion
73	5stars

MODELS

<u>id</u>	name	nationality	date birth	agency_cif
550	Rachel M.	British	1991	10
992	Maurine D.	Irish	1982	73
171	Cristina S	Colombian	1989	10
909	Kathy O.	British	1992	73

COMPANIES

<u>cif</u>	name	manager	sector
88	Chanel	D.Gilmour	C
11	Valentino	A.Di Pietro	C
44	Audi	J.Peters	M

The SOLUTION 1 leads to a new relation:

RESULTS (agency_cif, company_cif, impact)

RESULTS

<u>agency_cif</u>	<u>cif_company</u>	impact
73	88	2
73	44	1
10	44	3

Which tells us for what companies has worked each agency. However we have no information about the model who took part in the advert.

In the SOLUTION 2 and SOLUTION 3 this problem is fixed with using this new relation.

IMPACTS (model_id, company_cif, impact)

RESULTS

<u>model_id</u>	<u>cif_ company</u>	impact
992	88	2
909	44	1
550	44	3

Since a model works for an agency only, implicitly this relation/table also gives us information about the agency that made the advert. That means, however, that if we have this relation/table, the previous one is redundant and should be rulled out.

So, the best solution it the SOLUTION 3. We must take into account that a model can take part in different adverts for the same company, meaning that she works in different momement throughout the time. We need a column to refer to the date, that will have to be part of the primary key as you can see next:

IMPACTS (model_id, company_cif, date, impact)

RESULTS

<u>model_id</u>	<u>cif_ company</u>	<u>date</u>	impact
992	88	2008	2
909	44	2009	1
550	44	2008	3

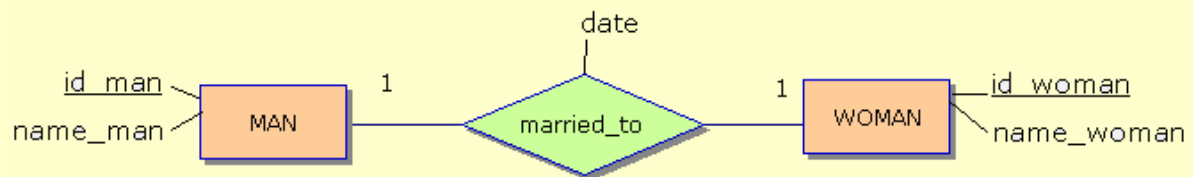


ACTIVITY 4.9 – Transforming 1:1 Relationships

objective

To transform one-to-one entity-types with no existence constraint into relations/tables.

Given the SCENARIO 3 where the relationship is optional in both directions.



- Transform the the entity-types into relations.
- Draw corresponding tables for the obtained relations and fill them with instances

suggested answer

Each entity will be transformed into a relation and any of them will get the identifier of the other one as an alternate key. So, there are two possibilities:

SOLUTION 1

MEN (id_man, name_man, id_woman)

Alternate key: id_woman

WOMEN (id_woman, name_woman)

MEN

<u>id_man</u>	<u>name_man</u>	id_woman	date
14	Peter	78	2000
29	Ben	90	2003
45	Jason		

WOMAN

<u>id_woman</u>	<u>name_woman</u>
78	Mary
55	Rachel
90	Julie

Where Julie and Jason are single.

SOLUTION 2

Like the previous one but it is the women table that absorbs the primary key of the men table.

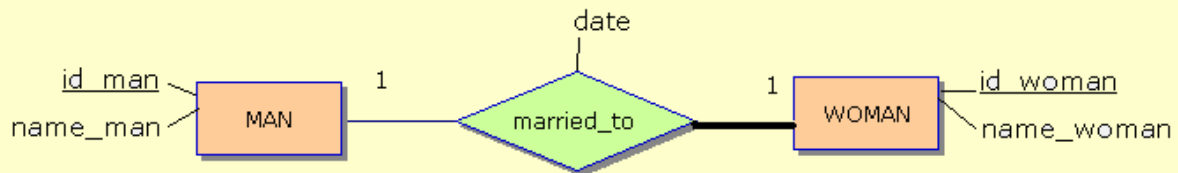


ACTIVITY 4.10 – Transforming 1:1 Relationships II

objective

To transform one-to-one entity-types with one existence constraint into relations/tables.

Given the reality of the SCENARIO 3, consider now that all women, as soon they are born, are forced to marry a man, that's to say that there is a total participation in the woman's side.



- e) Transform the the entity-types into relations.
- f) Draw corresponding tables for the obtained relations and fill them with instances

suggested answer

The result of transforming this relationship is like the one we have obtained before, except that in this case a not null value is required to express the restriction. Since are the woman who have to be married, it is that relation/table that takes the value of the men.

SOLUTION 1

WOMEN (id_woman, name_woman, id_man)

Alternate key: id_man

NNV: Id_woman → WOMEN

WOMEN (id_man, name_man)

WOMEN

<u>id_woman</u>	<u>name_woman</u>	id_man	date
78	Mary	14	2000

55	Rachel	66	2008
90	Julie	45	2003

MAN

<u>id_man</u>	<u>name_man</u>
14	Peter
29	Ben
45	Jason
66	Stuard

All women are married but, there can be single men.



ACTIVITY 4.11 – College

objective

To transform a simple ER diagram with any type of relationships into relations/tables.

Given the ER diagram obtained for the SCENARIO 1:

We wish to create a database for our college, considering the following information.

Each student enrolls in a series of subjects in which they will get a mark. Many students can enroll in a subject and a student can enroll in the same subject several times.

Each subject is taught by one teacher and a teacher can teach different subjects. Each teacher belongs to a department.

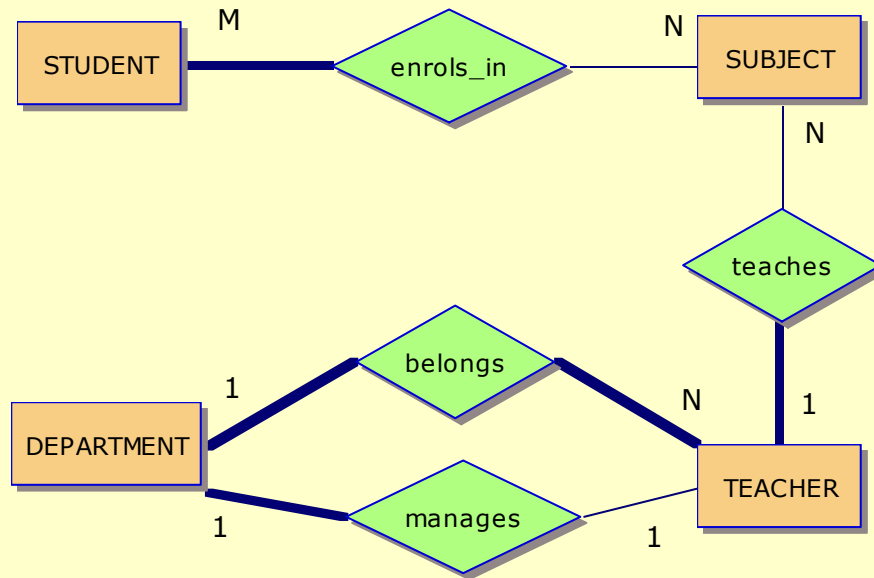
We want to store each student's ID, name, family name, address, city, telephone and marital status. The marital status of a student can be one of the following: S: single, M: married, W: widower/ widow, D: divorced.

For each of the subjects we will keep a code, their name and hours.

For each teacher we know their ID, university degree that they have, name and surname.

We know the code and name of each of the departments.

The corresponding ER diagram is:



- a) Transform the ER diagram above into the corresponding relations.
 - b) Draw the corresponding tables for the previous relations and fill them with entity instances of the activity 3.1.
 - c) Make up examples to prove that there are aspects of the reality that can't be represented with the final relations.
- (*) NOTE: To know the attributes of each entity-type, refer to the activity 3.1. The activity 3.2 gives us the identifiers.

suggested answer

a)

1.- The first stage entails transforming the entities:

STUDENTS(id, name, f_name, address, city, telephone, marital_status

SUBJECTS (code, name, hours,

TEACHERS (id, name, f_name, degree

DEPARTMENTS (code, name

2.- The one-to-many relationships are transformed next:

STUDENTS(id, name, f_name, address, city, telephone, marital_status

SUBJECTS (code, name, hours, **teacher_id**)

Foreign key: teacher_id → TEACHERS

TEACHERS (id, name, f_name, degree, **department_code**)

Foreign key: department_code → DEPARTMENTS

VNN: department_code

DEPARTMENTS (code, name,

3.- Then, we convert the many-to-many relationships:

STUDENTS(id, name, f_name, address, city, telephone, marital_status)

SUBJECTS (code, name, hours, teacher_id)

Foreign key: teacher_id → TEACHERS

TEACHERS (id, name, f_name, degree, department_code)

Foreign key: department_code → DEPARTMENTS

VNN: department_code

DEPARTMENTS (code, name,

ENROLS_IN (id_student, subject_code, mark)

Foreign key: id_student → STUDENTS

Foreign key: subject_code → SUBJECTS

(*) it can't be represented that a student has to enrol in at least one subject.

4.- Finally, the one-to-one relationship is transformed:

STUDENTS(id, name, f_name, address, city, telephone, marital_status)

SUBJECTS (code, name, hours, teacher_id)

Foreign key: teacher_id → TEACHERS

TEACHERS (id, name, f_name, degree, department_code)

Foreign key: department_code → DEPARTMENTS

VNN: department_code

DEPARTMENTS (code, name, **teacher_id**)

Alternate key: teacher_id

NNV: teacher_id

ENROLS_IN (id_student, subject_code, mark)

Foreign key: id_student → STUDENTS

Foreign key: subject_code → SUBJECTS

(*) it can't be represented that a student has to enrol in at least one subject.



ACTIVITY 4.12 – *General elections*

objective

To derive a simple relational design from a simple ER diagram.

Given the scenario of the activity 3.13:

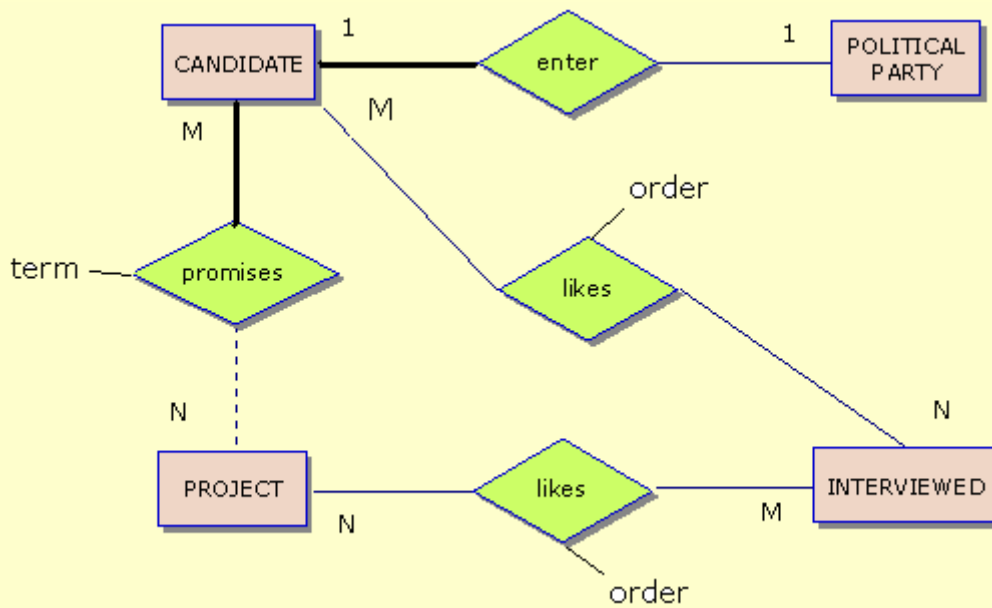
A company of statistics services makes surveys to know the way people interviewee intend to vote about the different presidential candidates

The known information about the candidates is their name, age and unique code. They also have information about the different political parties that take part in the elections such their name, founding year and political tendencies. Each party can only enter a candidate.

The interviewee are asked their age, cultural level (1: without an education, 2: primary education, 3: secondary education, 4: higher education) and have a code.

The company is also interested in knowing their taste about the different projects that make up the election manifesto of each candidate. So, it has information about the description of all projects (which sometimes are included in the election manifesto of different candidates), the approximate execution term and a code for each one.

The ER diagram for the information system above described is:



- Transform the ER above into relations for the ER above

suggested answer

The final relations will be:

CANDIDATES (code, name, age, party_code)

Alternate key: party_code

Foreign key: party_code → PARTIES

NNV: party_code

PARTIES (code, name, f_year, tendencies)

PROJECTS (code, description)

INTERVIEWEES (code, age, c_level)

CANDIDATES-INTERVIEWEE (candidate_code, inter_code, order)

Foreign key: candidate_code → CANDIDATES

Foreign key: inter_code → INTERVIEWEE

PROJECTS-INTERVIEWEES (project_code, inter_code, order)

Foreign key: project_code → PROJECTS

Foreign key: inter_code → INTERVIEWEES

PROMISES (project_code, candidate_code, term)

Foreign key: project_code → PROJECTS

Foreign key: candidate_code → CANDIDATES

Summary

- The logical design is the second phase of database design.
- An ER diagram representing a database application can be converted to a set of tables.
- In general, an ERD entity-type converts to a relational table.
- Relationships conversion will depend on their maximum cardinality and optionality. Some relationships will generate another table, while others can be represented within the entity tables.
- Relationships between entities can be represented either by foreign key additions to entity tables or by additional tables, depending on the cardinality and optionality.
- For any tables linked by a one-to-many relationship, the foreign key will be in the relation/table at the “many” end, and the referenced primary key at the “one” end.

Glossary IV

English-Catalan Vocabulary

English	Meaning
To transpose... to	Transportar, canviar
Stage	Pas
Linkage	Connexió
To tend to	Tenir tendència a
To depict	Representar
Budget	Pressupost
Marriage	Matrimoni
To convert... Into	Convertir.... En
To name	Nomenar
Tor treat	Tractar
To make up	Inventar
Balance	Saldo
To be forced to	Ser obligat/da a
To draw	Dibuixar

Key Vocabulary

At the end of this lesson the students must be able to recognise and understand the meaning of the following words:

EXISTENCE CONSTRAINT

FOREIGN KEY

ALTERNATE KEY

PRIMARY KEY

NNV = NOT NUL VALUE

REDUNDANT RELATIONSHIP

MAXIMUM CONNECTIVITY

MINIMUM CONNECTIVITY

PARTIAL PARTICIPATION

TOTAL PARTICIPATION

RELATION

RELATIONSHIP

TABLE

Lesson Plan	PART I - DATABASE DESIGN		
Lesson 4. Transformation into Tables			
TIMING	6 hours	LEVEL	1st CFGS DAI/ASI
OBJECTIVES	CONTENT	<ul style="list-style-type: none"> - Principles of conversion process from ER diagrams to a set of relational tables - Entities transformation - 1:N relationships transformation - N:M relationships transformation - 1:1 relationships transformation 	
	COGNITION	<ul style="list-style-type: none"> - To understand the process of deriving a table design from an ER diagram 	
	CULTURE	-	
	COMMUNICATION	<p>Language for learning:</p> <ul style="list-style-type: none"> - To discuss and ask about the concepts - To justify their answers and reasoning <p>Language of learning:</p> <ul style="list-style-type: none"> - Language structures needed to complete the tasks <p>Language trough learning:</p> <ul style="list-style-type: none"> - Language that comes up when carrying out the different activities 	
LEARNING OUTCOMES	<p>At the end of this lesson, students will be able to...</p> <ul style="list-style-type: none"> - Transpose an ER diagram to an equivalent set of relations - Identify columns that must be added as part of the primary to the table meaning express the information of the reality. 		

RESOURCES AND TIMING	Lesson in power point (<i>P1.L4.Tranformation.ppt</i>)	
	Lesson 4.- Transformation into Tables (8 hours teaching + activities)	
	Activity 4.1	Tranforming Entities
	Activity 4.2	Transforming 1:N relationships
	Activity 4.3	Entity vs attribute
	Activity 4.4	Bank
	Activity 4.5	Transforming N:M relationships
	Activity 4.6	Nursery
	Activity 4.7	Garage
	Activity 4.8	Model agencies
	Activity 4.9	Transforming 1:1 relationships
	Activity 4.10	Transforming 1:1 relationships II
	Activity 4.11	College
	Activity 4.12	General elections

Part 2

SQL Language

Part 2. Lesson 1

SQL Basics

Introduction

As the title says, this lesson deals with the fundamentals of SQL only. At the end of this lesson, the students must be able to:

- a) Understand the possibilities of SQL as a language of data definition (DDL) and data manipulation (DML).
- b) Identify the functions, syntax and basic commands of SQL to define, retrieve and update data.

So, this lesson will be used as a “springboard” to start with the language in the following lessons.

Approach

Although SQL is totally practical, this is a theoretical lesson where the students don't have to create SQL statements. So, no computers are necessary to learn most of the concepts covered in this lesson.

The activities of this lesson should be solved individually or in pairs. Then the answers could be discussed and corrected by the whole class given the students the chance to express their ideas and possible solutions.

Activities



ACTIVITY 1.1 – Vocab Workshop

objective

To get used to the things that can be done with SQL.

a) Choose the right words from the list below to fill the gaps.

insert	create	delete	administrative
destroy	programming language	update	retrieve
relational database management systems (RDBMS)			management

SQL is an abbreviation for *Structured Query Language*. SQL is a standard _____ specifically designed for communicating with _____.

With SQL you can _____ and _____ databases, as well as _____ their structure. SQL also allows _____, _____, _____, and _____ data. A database management system also includes _____ and _____ functions.

suggested answer

SQL is an abbreviation for *Structured Query Language*. SQL is a standard **programming language** specifically designed for communicating with **relational database management systems (RDBMS)**.

With SQL you can **create** and **destroy** databases, as well as **modify** their structure. SQL also allows **retrieve**, **insert**, **update**, and **delete** data. A database management system also includes **management** and **administrative** functions.

b) Choose synonymous for each verb.	
word	suggested synonymous
to manipulate	to operate, to handle, to work, to control, to use, to employ, to utilize...
to retrieve	to recover, to get out, to recuperate, to obtain, to take out...
to insert	to put, to introduce, to enter, to inset, to add..
to delete	to remove, to erase, to rub out, ...
to update	to improve, to upgrade, ...
to manage	to organize, to administer, to take care of, to regulate...



ACTIVITY 1.2 - Quiz

objective

To understand the importance of learning SQL.

Answer the following questions about SQL.

- a) Why should IT students be concerned about SQL?
- b) Why do we need to know something about relational the database theory to use SQL?
- c) If each vendor has its own version of SQL, is it worth to learn ANSI SQL?

suggested answer

a)

SQL is a standard language to manipulate databases.

Programming languages today require a working knowledge of SQL to develop applications that access to databases.

Retrieve information from databases is also fundamental for computers expert.

b)

SQL was developed to service relational databases. Without a minimal understanding of relational database theory, you will not be able to use SQL effectively except in the most trivial cases.

c)

Of course, Although each vendor's implementation will differ slightly from the others, you should be able to use SQL with very few adjustments. ANSI SQL is a standard version of the language.



ACTIVITY 1.3 – Quiz

objective	To diference among the different SQL languages.	
<p>Classify the following SQL operations as part of DML, DDL or DCL</p> <ul style="list-style-type: none"> a) insert records in a database b) set permissions on tables, procedures, and views c) retrieve data from a database d) create new tables in a database e) update records in a database f) delete records from a database g) create new databases h) execute queries against a database 		
suggested answer		
<ul style="list-style-type: none"> a) DML b) DCL c) DML d) DDL 	<ul style="list-style-type: none"> e) DML f) DML g) DDL h) DML 	



ACTIVITY 1.4 – Quiz

objective

To understand the syntax of SQL statements.

Is there any difference in the statements bellow?

- a) SELECT * FROM students;
- b) select * from students;
- c) SELECT *
FROM students;

suggested answer

All of them work and give the same result but the letter c) is the most advisable because:

- commands and clauses are written in capital letters
- reserved words are broken up in different lines which makes them easier to understand and read.

**ACTIVITY 1.5 - Quiz****objective**

To revise some aspects of the SQL syntax.

The queries below don't work. Why?

- a) `SELECT * FROM people WHERE name= Mark;`
- b) `SELECT * FROM people`
- c) `SELECT * FROM ;`
- d) `SELECT * FROM 123;`
- e) `SELECT * FROM where;`

suggested answer

- a) The data type of the field name is string, so it must be enclosed between inverted commas.
- b) The statement must finish with a semicolon.
- c) The table/s from which the data are retrieved must be shown.
- d) A table can't be called 123.
- e) You're not allowed to call a table using the name of a clause or command (where in this example).

Glossary I

English-Catalan Vocabulary

English	Meaning
Almost	Quasibé
Avalaible	Disponible
Compulsory	Obligatori
To delete	Esborrar
Facility/ facilities	Prestació/s
Feature	Característica
Following	Següent
Remove	Esborrar, eliminar
Semicolon	Punt i coma
To store	Emmagatzemar
To allow	Permetre
To break up	Dividir, descomposar
To deal wih	Tractar amb
To embed	Encastar
To handle	Manipula
To be made up of	Estar format/composat de
To update	Actualitzar
To perform	Desempeñar
Optional	Opcional
To retrieve	Recuperar
To type	Teclejar
Blank	Espai en blanc
To debug	Depurar
Case-sensitivity	Sensibilitat a les lletres majúscules
Clause	Clàusula

Field	Camp
Host language	Llenguatge anfitrió
Iso	International standards organisation
Null value	Valor nul
Programming language	Llenguatge de programació
Query	Consulta
Rdbms = relational data base management system	Sgbdr = sistema gestor de bases de dades relacional
Screen	Pantalla
Sql = structured query language	Sql = llenguatge estructurat de consulta
Statement	Sentència
To query	Consultar
To type	Teclejar (escriure a màquina)
Command	Ordre, comanda
Ansi	American national standards institut
Field	Camp
Dbms = data base management system	Sgbd = sistema gestor de bases de dades
Ddl = data definition language	Ddl = llenguatge de denició de dades
Dml = data manipulation language	Dml = llenguatge de manipulació de dades
Embedded sql	Sql encastat, hostatjat

Key Vocabulary

At the end of this lesson the students must be able to recognise and understand the meaning of the following words:

ANSI

COMMAND

DBMS = DATA BASE

MANAGEMENT SYSTEM

DDL = DATA DEFINITION

LANGUAGE

DML = DATA MANIPULATION
LANGUAGE

FIELD

FIELD

HOST LANGUAGE

ISO

NULL VALUE

PROGRAMMING LANGUAGE

QUERY

RDBMS = RELATIONAL DATA BASE
MANAGEMENT SYSTEM

SQL = STRUCTURED QUERY
LANGUAGE

STATEMENT

TO QUERY

TO RETRIEVE

EMBEDDED SQL

Lesson Plan	PART II - STRUCTURED QUERY LANGUAGE		
Lesson 1. SQL BASICS			
TIMING	2 hours	LEVEL	1 st CFGS DAI/ASI
OBJECTIVES	CONTENT	<ul style="list-style-type: none"> - History - How SQL works. - Syntax. - Components: Data Manipulation Language, Data Definition Language and Data Control Language. - How SQL can be used: interactive and embedded SQL - Null values. 	
	COGNITION	<ul style="list-style-type: none"> - To understand the possibilities of SQL as a language of data definition and data manipulation - To know the syntax fo the SQL statements 	
	CULTURE	The importance of store and retrieve information.	
	COMMUNICATION	<p>Language for learning:</p> <ul style="list-style-type: none"> - To discuss and ask about the concepts - To justify their answers and reasoning <p>Language of learning:</p> <ul style="list-style-type: none"> - Language structures needed to complete the tasks <p>Language trough learning:</p> <ul style="list-style-type: none"> - Language to carry out the different activities 	
LEARNING	At the end of this lesson, students will be able to...		

OUTCOMES	<ul style="list-style-type: none"> - To know what be can done with SQL. - Identify the functions, syntax and basic commands of SQL to define, retrieve and update data 												
RESOURCES AND TIMING	<p>Lesson in power point (<i>P1.L1.SQL_Basics.ppt</i>)</p> <table border="1" data-bbox="440 450 1297 757"> <tr> <td colspan="2" data-bbox="440 450 1297 573">Lesson 1.- SQL Basics (2 hours teaching + activities)</td> </tr> <tr> <td data-bbox="440 573 715 611">Activity 1.1</td> <td data-bbox="715 573 1297 611">Vocab Workshop</td> </tr> <tr> <td data-bbox="440 611 715 649">Activity 1.2</td> <td data-bbox="715 611 1297 649">Quiz I</td> </tr> <tr> <td data-bbox="440 649 715 687">Activity 1.3</td> <td data-bbox="715 649 1297 687">Quiz II</td> </tr> <tr> <td data-bbox="440 687 715 725">Activity 1.4</td> <td data-bbox="715 687 1297 725">Quiz III</td> </tr> <tr> <td data-bbox="440 725 715 757">Activity 1.5</td> <td data-bbox="715 725 1297 757">Quiz IV</td> </tr> </table>	Lesson 1.- SQL Basics (2 hours teaching + activities)		Activity 1.1	Vocab Workshop	Activity 1.2	Quiz I	Activity 1.3	Quiz II	Activity 1.4	Quiz III	Activity 1.5	Quiz IV
Lesson 1.- SQL Basics (2 hours teaching + activities)													
Activity 1.1	Vocab Workshop												
Activity 1.2	Quiz I												
Activity 1.3	Quiz II												
Activity 1.4	Quiz III												
Activity 1.5	Quiz IV												

Part 2. Lesson 2

Data Manipulation Language

Introduction

The main purpose of storing data on a computer is to be able to retrieve specific elements of the data when you need them. However, as databases grow in size, the difficulty to retrieve specific data also increases.

You may want to retrieve the contents of one row out of thousands in a table. You may want to retrieve all the rows that satisfy a condition or a combination of conditions. You may even want to retrieve all the rows in the table.

One particular SQL statement, the SELECT statement, performs all these tasks. In its simplest form, with one modifying clause (FROM clause), it retrieves everything from a table. By adding more modifying clauses, you can refine what it retrieves.

Approach

As the best way to learn SQL is to try it, from next lesson we're going to use several databases which test the SQL statements:

- On one hand, we will use simple databases as exemple to explain new statements and clauses.
- On the other hand, throughout the lesson we will use a real database called "distributors" for the students to put into practice what they are learning as they go along.
- Once all DML statements are discussed the students should resolve, individually or in pairs, the activities coming up next.

In all cases the queries must be solved following the given order and with the help of the teacher who will clarify students' doubts. The students can check if their answers are correct by comparing the obtained results with the lists given for each query.



The DISTRIBUTORS Database

The following tables are used by an imaginary group of distributors of pieces to perform several tasks

In a simplified way, three tables are required:

- PIECES The pieces table contains the pieces catalogue, one piece per row.
- DISTRIBUTOR The distributors table stores all information about pieces vendors.
- SUPPLIES This table keeps information about the different pieces supplied by the distributors.

Relational Schema

PIECES (p_id, p_name, p_colour, p_weight, p_city)

primary key: p_id

DISTRIBUTORS (d_id, d_name, d_age, d_city)

primary key: d_id

SUPPLIES (d_id, p_id, amount)

primary key: d_id, p_id

foreign key: id_s → DISTRIBUTORS

foreign key: id_p → PIECES

Table Descriptions

PIECES

Columns	description
p_id	Unice piece ID
p_name	piece name
p_col	piece colour
p_wg	piece weight
p_city	city where the piece is manufactured

DISTRIBUTORS

Columns	description
d_id	Unique distributor ID
d_name	distributor name
d_age	distributor age
d_city	city where the distributor is based

❑ SUPPLIES

Columns	description
d_id	Unique distributor ID
p_id	Uniquese piece ID
amount	units that this distributor has supplied



ACTIVITY 2.1 – Inserting data into the "distributors" database

objective

Work with the INSERT command to put new records into the database tables

- Using the SQL statement INSERT, fill the tables with the data given below.
- Save the data when you finish.

PIECES

p_id	P_name	P_colour	p_weight	p_city
P1	nut	Red	12	Edinburgh
P2	peg	Green	17	Glasgow
P3	screw	Blue	17	Motherwell
P4	screw	Red	14	Edinburgh
P5	washer	Blue	12	Glasgow
P6	sprocket	Red	19	Edinburgh

DISTRIBUTORS

D_id	d_name	d_age	d_city
D1	Smith	30	Edinburgh
D2	Jones	28	Glasgow
D3	Blake	40	Glasgow
D4	Clark	30	Edinburgh
D5	Adams	40	Stirling

SUPPLIES

d_id	p_id	amount
D1	P1	300
D1	P2	200
D1	P3	400
D1	P4	200
D1	P5	100
D1	P6	100
D2	P1	300
D2	P2	400
D3	P2	200
D4	P2	200
D4	P4	300
D4	P5	400

suggested answer

```
INSERT INTO pieces VALUES('P1', 'nut', 'red', 12, 'Edinburgh');
INSERT INTO pieces VALUES('P2', 'peg', 'green', 17, 'Glasgow');
INSERT INTO pieces VALUES('P3', 'screw', 'blue', 17, 'Motherwell');
INSERT INTO pieces VALUES('P4', 'screw', 'red', 14, 'Edinburgh');
INSERT INTO pieces VALUES('P5', 'washer', 'blue', 12, 'Glasgow');
INSERT INTO pieces VALUES('P6', 'sprocket', 'red', 19, 'Edinburgh');
```

```
INSERT INTO distributors VALUES('D1', 'Smith', 30, 'Edinburgh');
INSERT INTO distributors VALUES('D2', 'Jones', 28, 'Glasgow');
INSERT INTO distributors VALUES('D3', 'Blake', 40, 'Glasgow');
INSERT INTO distributors VALUES('D4', 'Clark', 30, 'Edinburgh');
INSERT INTO distributors VALUES('D5', 'Adams', 40, 'Stirling');
```

```
INSERT INTO supplies VALUES('D1', 'P1', 300);
INSERT INTO supplies VALUES('D1', 'P2', 200);
INSERT INTO supplies VALUES('D1', 'P3', 400);
INSERT INTO supplies VALUES('D1', 'P4', 200);
INSERT INTO supplies VALUES('D1', 'P5', 100);
INSERT INTO supplies VALUES('D1', 'P6', 100);
INSERT INTO supplies VALUES('D2', 'P1', 300);
INSERT INTO supplies VALUES('D2', 'P2', 400);
INSERT INTO supplies VALUES('D3', 'P2', 200);
INSERT INTO supplies VALUES('D4', 'P4', 300);
INSERT INTO supplies VALUES('D4', 'P5', 400);
```



ACTIVITY 2.2 – Querying the “distributors” database

objective

To carry out queries of different difficulty level on a simple database.

Consider the DISTRIBUTORS database from the lesson 4

Write sentences to retrieve data:

Queries one a single table

1. The codes of all distributors.

d_id
D1
D2
D3
D4
D5

suggested answer

```
SELECT d_id
FROM distributors;
```

2. The codes of the distributors that provide a specific piece.

d_id
D1
D1
D1
D1
D1
D1

D2
D2
D3
D4
D4
D4

suggested answer

```
SELECT d_id
FROM supplies;
```

3. The name and the city of the distributors sorted in descending order.

d_name	d_city
Smith	Edinburgh
Jones	Glasgow
Clark	Edinburgh
Blake	Glasgow
Adams	Stirling

suggested answer

```
SELECT d_name, d_city
FROM distributors
ORDER BY d_name;
```

4. The codes of the distributors that provide a specific piece without repeating the obtained values.

d_id
D1
D2
D3
D4

suggested answer

```
SELECT DISTINCT d_id
FROM supplies;
```

5. The codes of distributors from Glasgow younger than 30 years old.

d_id
D2

suggested answer

```
SELECT d_id
FROM distributors
WHERE d_city = 'Glasgow' AND d_age <30;
```

6. The codes of distributors and the codes of pieces for the supplies where the amount is 100.

d_id	p_id
D1	P5
D1	P6

suggested answer

```
SELECT d_id, p_id
FROM supplies
WHERE amount=100;
```

7. Code and name of the pieces starting with an S.

p_id	p_name
P3	screw
P4	screw
P6	sprocket

suggested answer

```
SELECT p_id, p_name
FROM pieces
WHERE p_name LIKE 'S%';
```

8. Code and name of the pieces containing the letter S.

p_id	p_name
P3	screw
P4	screw
P5	washer
P6	sprocket

suggested answer

```
SELECT p_id, p_name
FROM pieces
WHERE p_name LIKE '%S%';
```

9. Total amount of distributors.

amount_sup
5

suggested answer

```
SELECT COUNT( DISTINCT d_id) AS amount_sup
FROM distributors;
```

10. Amount of distributors that supply at least one piece.

amount_sup
4

suggested answer

```
SELECT COUNT (DISTINCT d_id) AS amount_sup
FROM supplies;
```

11. Amount of supplies for the piece P2.

total_supplies_P2
4

suggested answer

```
SELECT COUNT (*)
FROM supplies
WHERE p_id= 'P2';
```

12. Total amount supplied for the piece P2.

total_supplied_P2
1000

suggested answer

```
SELECT MAX(amount) AS total_supplied_P2
FROM supplies
WHERE p_id= 'P2';
```

13. For each piece supplied, list the code, the maximum amount and the minimum amount supplied for this piece.

d_id	max	min
D1	400	100
D2	400	300
D3	200	200
D4	400	200

suggested answer

```
SELECT d_id, MAX(amount), MIN(amount)
FROM supplies
GROUP BY p_id';
```

14. For each piece supplied, list the code, the maximum amount and the minimum amount supplied for this piece. Exclude all the sales of the distributor D1.

d_id	max	min
D2	400	300
D3	200	200
D4	400	200

suggested answer

```
SELECT d_id, MAX(amount), MIN(amount)
FROM supplies
WHERE d_id='D1'
GROUP BY p_id';
```

15. Write a query to retrieve the codes of pieces supplied by more than one distributor.

p_id
P1
P2
P4
P5

suggested answer

```
SELECT p_id
FROM supplies
GROUP BY p_id
```


HAVING COUNT(*) >1;

Queries on several tables

16. Code of pieces that have been supplied in the same or greater amount than 400.

p_id	amount
P2	400
P3	400
P5	400

suggested answer

```
SELECT p_id
FROM supplies
WHERE amount >=400;
```

17. Name of pieces that have been supplied in the same or greater amount than 400.

p_name	amount
peg	400
Screw	400
washer	400

suggested answer

```
SELECT P.p_name, SP.amount
FROM pieces P, supplies SP
WHERE P.p_id= SP.p_id AND
amount >=400;
```

18. Name of the red pieces supplied by Clark.

p_id
P2

suggested answer

```

SELECT p_id
FROM pieces P, supplies SP, distributors D
WHERE P.p_id=SP.p_id AND D.d_id=SP.d_id AND
      P.colour='red'
      D.d_name='Clark';

```

Subqueries

19. Distributors younger than the maximum age in the distributors table.

d_id	d_name	d_age
D1	Smith	30
D2	Jones	28
D4	Clark	n30

suggested answer

```

SELECT d_id, d_name, d_age
FROM distributors
WHERE age <
      (SELECT MAX(age)
      FROM distributors);

```

20. Distributors based in the same city than as the distributor D1.

d_id	d_name	d_city
D1	Smith	Edinburgh
D4	Clark	Edinburgh

suggested answer

```

SELECT d_id, d_name, d_city
FROM distributors
WHERE d_city =

```

```
(SELECT d_city
FROM distributors
WHERE d_id);
```

21. Name of pieces that have been supplied in the same or greater amount than 400. (Query 17 using subqueries)

p_name	amount
peg	400
Screw	400
washer	400

suggested answer

```
SELECT p_name, amount
FROM pieces
WHERE p_id IN
      (SELECT p_id
FROM supplies
      WHERE amount >=400);
```

22. Names of distributors that supply the piece P2.

d_name
Smith
Jones

suggested answer

SOLUTION 1

```
SELECT d_name
FROM distributors
WHERE d_id IN
```

```
(SELECT d_id  
FROM supplies  
WHERE p_name='P2');
```

SOLUTION 2

```
SELECT d_name  
FROM distributors D, supplies SP  
WHERE D.d_id=SP.s.id AND  
SP.p_name='P2';
```

23. Name of the distributors that supply at least one red piece.

d_id
D1
D2
D4

suggested answer

SOLUTION 1

```
SELECT d_name  
FROM distributors  
WHERE d_id IN  
      (SELECT d_id  
      FROM supplies  
      WHERE p_id IN  
            (SELECT p_id  
            FROM pieces  
            WHERE colour='Red'));
```

SOLUTION 2

```

SELECT d_name
FROM distributors S, supplies SP, pieces P
WHERE D.d_id= SP.d_id AND
      SP.p_id = P.p_id AND
      P.colour='Red'

```

24. Codes of the distributors that supply at least one of the pieces supplied by D2.

d_id
D1
D2
D3
D4

suggested answer

```

SELECT DISTINCT
FROM supplies SP1
WHERE SP1.p_id IN
      (SELECT SP2.p_id
      FROM supplies SP2
      WHERE SP2.d_id='D2');

```

25. Name of distributors that supply at least one piece

d_name
Smith
Jones
Blake
Clark

suggested answer

SOLUTION 1

```
SELECT d_name
FROM distributors D
WHERE EXISTS
    (SELECT *
    FROM supplies SP
    WHERE SP.d_id = D.d_id);
```

SOLUTION 2

```
SELECT d_name
FROM distributors
WHERE d_id IN
    (SELECT DISTINCT d_id
    FROM supplies);
```

26. Name of distributors that supply all the pieces.

d_name
Smith

suggested answer

```
SELECT d_name
FROM distributors D
WHERE NOT EXISTS
    (SELECT *
    FROM pieces P
    WHERE NOT EXISTS
        (SELECT *
        FROM supplies SP
        WHERE SP.d_id = D.d_id AND
            SP.p_id= P.p_id);
```

27. Codes of distributors that supply at least one piece in an amount greater than

any amount supplied by D1.

d_id

suggested answer

SOLUTION 1

```
SELECT DISTINCT SP1.d_id
FROM distributors SP1
WHERE NOT EXISTS
    (SELECT *
     FROM distributors SP2
     WHERE SP2.d_id='S1' AND
     SP2.amount > SP1.amount);
```

SOLUTION 2

```
SELECT DISTINCT (d_id)
FROM supplies
WHERE amount >
    (SELECT MAX(amount) >
     FROM supplies
     WHERE d_id='D1'
     GROUP BY d.id);
```



The PROJECTS Database

A company uses the following database to keep information about its employees as well as about the projects they work on and the departments they are assigned to.

The three tables needed are:

- ❑ DEPARTMENTS The departments table keeps the data about the different departments of the company.
- ❑ PROJECTS The projects table contains information about the projects that are being developed by the company.
- ❑ EMPLOYEES This table stores all information about the employees that work for the company. An employee belongs to a department and is assigned to a project.

Relational Schema

DEPARTMENTS (dpt_code, dpt_name, dpt_city, building, floor)

primary key: dpt_code

PROJECTS (proj_code, proj_name, product, budget)

primary key: proj_code

EMPLOYEES (emp_code, emp_name, salary, emp_city, dpt_code, proj_code)

primary key: emp_code

foreign key: dpt_code → DEPARTMENTS

foreign key: proj_code → PROJECTS

NNV: dpt_code

NNV: proj_code

Table Descriptions

- ❑ DEPARTMENTS

columns	datatype	description
dpt_code	varchar(3)	unique department ID
dpt_name	varchar(20)	department name
dpt_city	varchar(20)	city where the department is based

building	varchar(20)	building where the department is located
floor	varchar(2)	floor of the department

❑ PROJECTS

columns	datatype	description
proj_code	varchar(3)	unique project ID
proj_name	varchar(10)	project name
product	varchar(20)	name of the product that is being developed in the project
budget	integer	amount of money that can be spent

❑ EMPLOYEES

columns	datatype	description
emp_code	varchar(3)	unique distributor ID
emp_name	varchar(30)	employee name
salary	integer	salary that the employee is paid
emp_city	varchar(20)	city where the employee was born
dpt_code	varchar(3)	department that the employee works on
emp_code	varchar(3)	units that this distributor has supplied

Table Content

❑ DEPARTMENTS

dpt_code	dpt_name	dpt_city	building	floor
1	management	Barcelona	Pau Claris	10
2	management	London	Princess	8
3	marketing	Barcelona	Pau Claris	1
4	marketing	London	Queen St.	3
5	sales	Barcelona	Muntaner	1
6	sales	London	Dunbarton	1
7	sales	Glasgow	Argyl St.	3
8	sales	Seville	De las Sierpes	1
9	administration	Barcelona	Muntaner	7

❑ PROJECTS

proj_code	proj_name	product	Budget
1	ibdtel	television	10000
2	ibdvid	video	5000
3	ibdtel	telephone	2000

4	Ibdcom	cd	20000
---	--------	----	-------

□ EMPLOYEES

emp_code	emp_name	salary	emp_city	dpt_cod	proj_code
1	Claire	4000	Aberdeen	1	1
2	Eugenia	3500	Toledo	2	2
3	Paul	2500	Paisley	3	1
4	Richy	2500	London	4	2
5	Eulàlia	1500	Barcelona	5	1
6	Miquel	1250	Badalona	5	1
7	Mary	1750	London	6	2
8	Steven	1500	London	6	2
9	Laura	1250	Glasgow	7	3
10	Antonio	1500	Seville	8	3



ACTIVITY 2.3 – Querying the “projects” database

objective

To carry out queries of different difficulty level on a simple database.

Consider the PROJECTS database:

Write sentences to retrieve data from the database:

1. Name and salary of the employees with dpt_code 1, 2 or 3.

Emp_name	Salary
Claire	4000
Eugenia	3500
Paul	2500

suggested answer

```
SELECT emp_name, salary
FROM EMPLOYEES
WHERE DPT_CODE IN (1,2,3);
```

2. Name of employees working at the department 5 and the building where they work.

Emp_name	Building
Eulàlia	Muntaner
Miquel	Muntaner

suggested answer

```
SELECT E.emp_name, D.building
FROM employees E, departments D
WHERE E.dpt_code=D.dpt_code AND E.dpt_code='5';
```

3. Codes and names of the departments from London that have some employee that earns more than 2000€.

dpt_code	dpt_name	salary
2	Management	3500
4	Marketing	2500

suggested answer

```
SELECT D.dpt_code, D.dpt_name, E.salary
FROM departments D, employees E
WHERE D.dpt_code=E.dpt_code AND
      D.dpt_city='London' AND
      E.salary>2000;
```

4. Name of the employees that earn more than 2000€. List them in descendant alphabetical order.

emp_name
Claire
Eugenia
Paul
Richy

suggested answer

```
SELECT emp_name
FROM employees
WHERE salary > 200000
ORDER BY emp_name DESC;
```

5. For each department list its name and the highest salary.

dpt_code	MaxSalary
1	400000
2	350000
3	250000
4	250000
5	150000
6	175000
7	125000
8	150000

suggested answer

```
SELECT dpt_code, Max(salary) As MaxSalary
FROM employees
GROUP BY dpt_code;
```

6. Codes and name of the employees that don't belong to the department 1 and that work in Barcelona.

emp_code	emp_name	emp_city
5	Eulàlia	Barcelona

suggested answer

```
SELECT emp_code, emp_name, emp_city
FROM employees E, departments D
WHERE (E.dpt_code=D.Dpt_code) AND
(E.dpt_code <> '1') AND
(E.emp_city = 'Barcelona');
```

7. Amount of people that work in London.

EmployeesFromLondon
3

suggested answer

```
SELECT Count(*) As EmployeesFromLondon
FROM employees E, departments D
WHERE E.dpt_code=D.dpt_code AND Dpt_city='London!';
```

8. Name of the employees that earn more than the employee 3.

emp_name
Claire
Eugenia

suggested answer

```
SELECT emp_name
FROM employees
WHERE Salary >
      (SELECT salary
      FROM employees
      WHERE emp_code = '3');
```

9. Name of the employees that earn the highest salary.

emp_name
Claire

suggested answer

```
SELECT emp_name
FROM employees
WHERE salary =
      (SELECT Max(salary)
      FROM employees);
```

10. Codes and names of projects without any assigned employee.

proj_code	proj_name
4	lbdcom

suggested answer

SOLUTION 1:

```
SELECT proj_code, proj_name
FROM projects
WHERE proj_code NOT IN
      (SELECT DISTINCT proj_code
       FROM employees);
```

SOLUTION 2:

```
SELECT proj_code, proj_name
FROM projects P
WHERE NOT EXISTS (SELECT *
                  FROM employees E
                  WHERE E.proj_code = P.proj_code);
```

11. Names of departments with employees that work on the ibdtel project.

dpt_name
Management
Marketing
Sales

suggested answer

```
SELECT DISTINCT D.dpt_name
FROM employees E, projects P, departments D
WHERE (D.dpt_code = E.dpt_code)
      AND (E.proj_code = P.proj_code)
      AND (P.proj_name = 'ibdtel');
```

12. Names of the employees that don't work on Project 2.

emp_name
Claire
Paul
Eulàlia
Miquel
Laura
Antonio

suggested answer

```
SELECT emp_name
FROM employees
WHERE proj_code <> '2';
```

13. Codes and names of the departments with two or more employees working in project 1.

dpt_code	dpt_name
5	Sales

suggested answer

SOLUTION 1:

```

SELECT dpt_code, dpt_name
FROM departments
WHERE dpt_code IN
        (SELECT dpt_code
        FROM employees
        WHERE proj_code = '1'
        GROUP BY dpt_code, proj_code
        HAVING COUNT(*)>=2);

```

SOLUTION 2:

```

SELECT E.dpt_code, D.dpt_name
FROM departments D, employees E
WHERE D.dpt_code=E.dpt_code AND
        E.proj_code = '1'
GROUP BY E.dpt_code, D.dpt_name
HAVING COUNT(*)>=2;

```

14. Codes and names of the projects that have two or more employees assigned.

proj_code	proj_name
1	lbdtel
2	lbdvid

suggested answer

SOLUTION 1:

```
SELECT proj_code, proj_name
FROM projects
WHERE proj_code IN
        (SELECT proj_code
        FROM employees
        GROUP BY proj_code
        HAVING COUNT(*)>2);
```

SOLUTION 2:

```
SELECT E.proj_code, P.proj_name
FROM projects P, employees E
WHERE E.proj_code = P.proj_code
GROUP BY E.proj_code, P.proj_name
HAVING COUNT(*)>2);
```

15. Projects assigned to the employees of department 1.

products
television

suggested answer

SOLUTION 1:

```
SELECT products
FROM employees E, projects P
WHERE (E.proj_code = P.proj_code)
AND (E.dpt_code = '1');
```

SOLUTION 2:

```
SELECT DISTINCT P.products
FROM employees E, projects P
```

```
WHERE E.proj_code = P.proj_code AND
      E.dpt_code='1';
```

16. Name of the department where the employee 2 works and the name of the project assigned to this employee.

dpt_name	proj_name
Management	ibdvid

suggested answer

```
SELECT D.dpt_name, P.proj_name
FROM employees E, departments D, projects P
WHERE (E.emp_code = 2) AND
      (E.dpt_code = D.dpt_code) AND
      (E.proj_code = P.proj_code);
```

17. Code and name of the employees that live in the same city where the department they work for is located.

emp_code	emp_name
4	Richy
7	Maria
8	Steven
9	Laura

suggested answer

```
SELECT emp_code, emp_name
FROM employees E, departments D
WHERE (E.dpt_code = D.dpt_code)
      AND (E.emp_city = D.dpt_city);
```

18. Code and name of the projects for each project with more of one employee working in the same building.

proj_code	proj_name	building	EmpProj
1	lbdtel	Muntaner	2
1	lbdtel	Pau Claris	2
2	lbdvid	Dubarton	2
2	lbdvid	Princess	2

suggested answer

```
SELECT E.proj_code, P.proj_name, D.building
FROM employees E, departments D, projects P
WHERE (E.dpt_code = D.dpt_code) AND
      (E.proj_code = P.proj_code)
GROUP BY E.proj_code, P.proj_name, D.building
HAVING COUNT(*) > 1;
```

19. Projects that have smaller budget than the the project 1. List the code and the name of the projects and the average salary of the employees assigned to each project.

proj_code	proj_name	AverageSalary
2	lbdvid	
3	lbdtef	

suggested answer

```
SELECT P.proj_code, P.proj_name, AVG(E.salary) As AverageSalary
FROM projects P, employees E
WHERE (P.proj_code = E.proj_code)
      AND (P.budget <
           (SELECT budget
            FROM projects
            WHERE proj_code = '1'))
GROUP BY P.proj_code, P.proj_name;
```

20. Code and name of the projects that have a smaller budget than 6000 and for which all the assigned employees earn a salary greater of 1500 or over.

Proj_code	Proj_name
2	lbdvid

suggested answer

SOLUTION 1

```
SELECT proj_code, proj_name
FROM projects
WHERE (budget < 6000) AND
      proj_code IN
          (SELECT proj_code
           FROM employees
           GROUP BY proj_code
           HAVING MIN(salary)>=1500));
```

SOLUTION 2

```
SELECT P.proj_code, P.proj_name
FROM employees E, projects P
WHERE E.proj_code=P.proj_code AND budget<6000
GROUP BY proj_code, P.proj_name
HAVING MIN(E.salary)>=1500;
```

SOLUTION 3

```
SELECT P.proj_code, P.proj_name
FROM projects P
WHERE P.budget < 6000 AND
      NOT EXISTS (SELECT *
                  FROM employees E
                  WHERE E.proj_code=P.proj_code AND
                        E.salary<1500);
```

SOLUTION 4

```
SELECT proj_code , proj_name
FROM projects
WHERE budget <6000 AND
      proj_code NOT IN (SELECT proj_code
                       FROM employees
                       WHERE salary<1500);
```

21. Name of the cities where some employees live but where there isn't any department.

emp_city
Badalona
Aberdeen
Seville
Paisley
Toled

suggested answer

SOLUTION 1:

```
SELECT DISTINCT emp_city
FROM employees
WHERE emp_city NOT IN
      (SELECT DISTINCT dpt_city
```

FROM departments);

SOLUTION 2:

```
SELECT DISTINCT E.emp_city
FROM employees E
WHERE NOT EXISTS (SELECT *
                  FROM departments D
                  WHERE D.dpt_city=E.emp_city);
```

22. Code and name of the departments with more employees than the department 1.

dpt_code	dpt_name
5	sales
6	sales

suggested answer

SOLUTION 1:

```
SELECT dpt_code, dpt_name
FROM departments
WHERE dpt_code IN
      (SELECT DISTINCT dpt_code
       FROM employees
       WHERE dpt_code <> '1'
       GROUP BY dpt_code
       HAVING COUNT(*) >
              (SELECT COUNT(*)
               FROM employees
               WHERE (dpt_code = '1')));
```

SOLUTION 2:

```
SELECT D.dpt_code, D.dpt_name
```

```

FROM employees E, departments D
WHERE D.dpt_code=E.dpt_code
GROUP BY D.dpt_code, D.dpt_name
HAVING COUNT(*) > (SELECT *
                    FROM employees E
                    WHERE E.dpt_code='1');

```

23. Code and name of the department with no employees and that are located in Barcelona.

dpt_code	dpt_name
9	administration

suggested answer

SOLUTION 1:

```

SELECT dpt_code, dpt_name
FROM departments
WHERE (dpt_city = 'Barcelona')
AND (dpt_code NOT IN
      (SELECT dpt_code
       FROM employees
       GROUP BY dpt_code));

```

SOLUTION 2:

```

SELECT D.dpt_code, D.dpt_name
FROM departments D
WHERE NOT EXIST (SELECT *
                FROM employees E
                WHERE E.dpt_code=D.dpt_code AND
                D.dpt_city='Barcelona');

```


24.Code, name and budget of the projects with more than two employees assigned in the same city.

proj_code	proj_name	budget
2	ibdvid	5000

suggested answer

SOLUTION 1:

```
SELECT proj_code, proj_name, budget
FROM projects
WHERE proj_code IN
        (SELECT proj_code
        FROM employees
        GROUP BY proj_code, emp_city
        HAVING (COUNT(*) >1));
```

SOLUTION 2:

```
SELECT DISTINCT P.proj_code, P.proj_name, P.budget
FROM projects P, employees E
WHERE P.proj_code = E.proj_code
GROUP BY P.proj_code, P.budget, P.proj_name, E.emp_city
HAVING COUNT(*)>2;
```



The LIBRARY Database

In an imaginary library they use the following tables to manage the information of books and loans

The three tables needed are:

- BOOKS The books table keeps information about the different books that can be borrowed.
- COPIES The copies table stores data about the different copies that there're of each book
- THEMES It contains information about the different themes that the books deal with.
- MEMBERS This table keeps data about the people that can borrow books at the library.
- LOANS It stores information about the copies borrowed by a member.

Relational Schema

BOOKS (book_code, title, publisher, language, author, editions)

primary key: book_code

COPIES (copy_number, book_code, year_edition, number_edition)

primary key: copy_code

THEMES(book_code, theme)

Primary key: book_code, theme

Foreign Key: book_code → BOOKS

MEMBERS(member_code, name, fname, id, address, phone, date_registration)

Primary key: member_code

LOANS(member_code, copy_number, date, deadline, returning_date)

Primary key: member_code, copy_number, date

Table Descriptions

❑ BOOKS

Columns	datatype	description
book_code	varchar(3)	unique book ID
title	varchar(50)	title of the book
publisher	varchar(20)	name of the publisher
language	varchar(20)	original language
author	varchar(25)	writer
editions	integer	amount of editions

❑ COPIES

Columns	datatype	description
copy_number	varchar(3)	unique copy ID
book_code	varchar(10)	code of the book the copy refers to
year_edition	interger	year when the copy was edited
number_edition	integer	number of the edition

❑ THEMES

columns	datatype	description
book_code	varchar(3)	unique book ID
theme	varchar(30)	description of the theme the book deals with

❑ MEMBERS

columns	datatype	description
member_code	varchar(3)	unique member ID
name	varchar(15)	name of the member
fname	varchar(20)	family name of the member
id	varchar(8)	id number of the member
address	varchar(50)	address of the member
date_registration	date	date when the member registered to the library as a member

❑ LOANS

columns	datatype	description
member_code	varchar(2)	unique member ID
copy_number	varchar(3)	unique copy ID
date	date	date when the loan has taken place
deadline	date	date when the copy should be

returning_date date returned to the library
 real date when the copy is returned

Table Contents

☐ BOOKS

book_code	title	publisher	language	author	editions
1	Introduction to database desing	Adisson	English	Danson	5
2	SQL para principiantes	Adisson	Spanish	Danson	3
3	Database design using ER diagrams	Prentice-Hall	English	Martin	1

☐ COPIES

copy_number	book_code	year_edition	number_edition
1	1	2003	5
2	1	2003	5
3	2	2004	2
4	2	2004	2
5	2	2007	3
6	3	1997	1

☐ THEMES

book_code	theme
1	Database
2	Database
2	SQL
3	Database

☐ MEMBERS

member_code	name	fname	id	address	phone	Date_registration
1	Steven	Brown	11111111	Glasgow	111	01/01/2009
2	Eric	Matew	22222222	Barcelona	222	01/01/2004
3	David	Rooney	33333333	Barcelona	333	01/01/2009

☐ LOANS

member_code	copy_number	date	deadline	returning_date
1	1	15/10/2007	30/10/2007	20/10/2007
1	4	20/11/2008	05/12/2008	05/12/2008
2	2	01/06/2007	01/07/2007	02/07/2007
2	5	15/09/2008	15/10/2008	
2	2	10/10/2008	10/11/2008	11/11/2008
3	3	30/10/2009	30/10/2009	



ACTIVITY 2.4 – Querying the “library” database

objective

To carry out queries of different difficulty level on a more complex database.

Consider the LIBRARY database:

Write sentences to retrieve data from the it:

1. Retrieve all the books from the database.

book_code	title	publisher	language	author	editions
1	Introduction to database desing	Adisson	English	Dason	5
2	SQL para principiantes	Adisson	Spanish	Dason	3
3	Database design using ER diagrams	Prentice-Hall	English	Martin	1

suggested answer

```
SELECT *
FROM books;
```

2. Retrieve the title and the author of all the books.

title	author
Introduction to database desing	Dason
SQL para principiantes	Dason
Database design using ER diagrams	Martin

suggested answer

```
SELECT title, author
FROM books;
```

3. Retrieve the title and the author of all the books sorted by author (in descending order) and title.

title	author
Database design using ER diagrams	Martin
Introduction to database desing	Dason
SQL para principiantes	Dason

suggested answer

```
SELECT title, author
FROM books
ORDER BY author DESC, title;
```

4. For each loan, retrieve the number of copy, the code of the member and the number of days that it has taken to return the book, sorted by the code of member and the number of days

member_code	copy_number	days
1	1	5
1	4	15
2	5	
2	2	32
2	2	31
3	3	

SOLUTION:

suggested answer

```
SELECT copy_number, member_code, returning_date – date AS days
FROM loans
ORDER BY member_code, days;
```

5. Retrieve the title of all books written by Dason.

book_code	title	publisher	language	author	editions
1	Introduction to database desing	Adisson	English	Dason	5

2	SQL para principiantes	Adisson	Spanish	Dason	3				
suggested answer									
<pre>SELECT title FROM books WHERE author='Dason';</pre>									
<p>6. Name and family name of member that live in Barcelona or London and that have registered to the library in the last year.</p>									
<table border="1"> <thead> <tr> <th>name</th> <th>fname</th> </tr> </thead> <tbody> <tr> <td>David</td> <td>Rooney</td> </tr> </tbody> </table>						name	fname	David	Rooney
name	fname								
David	Rooney								
suggested answer									
<pre>SELECT name, fname FROM members WHERE address IN ('Barcelona', 'London') AND date >= '01/01/09';</pre>									
<p>7. Retrieve the number of copy and the code of the member for all loans that where made in 2008 and that haven't been returned yet.</p>									
<table border="1"> <thead> <tr> <th>member_code</th> <th>copy_number</th> </tr> </thead> <tbody> <tr> <td>2</td> <td>5</td> </tr> </tbody> </table>						member_code	copy_number	2	5
member_code	copy_number								
2	5								
suggested answer									
<pre>SELECT copy_number, member_code FROM loans WHERE date BETWEEN '01/01/08 AND '31/12/08' AND returning_date IS NULL;</pre>									

8. Books containing the expression "database desing".

book_code	title	publisher	language	author	editions
1	Introduction to database desing	Adisson	English	Dason	5
3	Database design using ER diagrams	Prentice-Hall	English	Martin	1

suggested answer

```
SELECT *
FROM books
WHERE title LIKE '%database design%';
```

9. Retrieve the number of copies and de code of members of the loans done in 2008 and that have been returned after the returning date or that haven't been returned yet and they are also after the returning date.

member_code	copy_number	date	Deadline	returning_date
2	5	15/09/2008	15/10/2008	
2	2	10/10/2008	10/10/2008	11/01/2008

suggested answer

```
SELECT copy_number, member_code
FROM loans
WHERE date >= '01/01/2008' AND
(returning_date > deadline) Or
(returning_date IS NULL AND CURRENT_DATE >
deadline);
```

10. Themes of the books written by Dason.

theme
Database

suggested answer

```
SELECT DISTINCT T.theme
FROM themes T, books B
WHERE T.book_code = L.book_code AND
```


L.author = 'Dason';

11. Retrieve the not returning loans. List the code of books and the names and family names of the members, sorted by family name and name.

book_code	name	fname
2	Eric	Matew
2	David	Rooney

SOLUTION:

suggested answer

```
SELECT C.book_code, M.name, M.fname
FROM copies C, loans L, members M
WHERE C.copy_number = L.copy_number AND
      L.returning_date IS NULL
ORDER BY M.fname, M.name;
```

12. Names and family name of members that, in 2008, have borrowed some of the books written by Dason

member_code	copy_number	name	fname
1	4	Steven	Brown
2	2	Eric	Matew

suggested answer

```
SELECT DISTINCT (M.name, M.fname)
FROM member M, loans L, copies C, books B
WHERE M.member_code = L.member_code AND
      L.copy_number = C.copy_number AND
      C.copy_number = B.book_code
      L.date BETWEEN 01/01/05 AND 31/12/2005;
```

13. Codes of books that are about themes discussed in the book 1.

book_code
2
3

suggested answer

```
SELECT DISTINCT book_code
FROM themes T1, themes T2
WHERE T1.book_code = '1' AND
      T1.theme = T1.theme AND
      T2.book_code <> T1.book_code;
```

14. Amount of books written by Dason and the average of editions of these books.

books	av_editions
2	7

suggested answer

```
SELECT COUNT (book_code) AS books, AVG (editions) AS
      av_editions
FROM books
WHERE author = 'Dason';
```

15. Amount of days of the longest loan of 2008.

days
32

suggested answer

```
SELECT MAX (returning_date - date) AS days
FROM loans
WHERE date BETWEEN '1/1/08' AND 31/12/08;
```

16. Amount of books, amount of copies and average of copies per book.

books	copies	average
3	6	2

suggested answer

```
SELECT COUNT (DISTINCT book_code) AS a_books,
      COUNT (copy_number) AS a_copies,
      A_copies / a_books AS average
FROM loans;
```

17. Retrieve the total amount of loans made in 2008, the amount of returned loans and the amount of non returned loans.

total	returned	non_returned
4	2	1

suggested answer

```
SELECT COUNT (date) AS total,
       COUNT (returning_date) AS returned,
       total – returned AS non_returned
FROM loans
WHERE date BETWEEN '1/1/08' AND '31/12/08';
```

18. For each member, retrieve the amount of copies that has borrowed from the library.

member_code	copies
1	2
2	3
3	1

suggested answer

```
SELECT member_code, COUNT (copy_number) AS n_copies
FROM Loans
GROUP BY member_code;
```

19. Amount of *different* books borrowed by each member.

member_code	copies
1	2
2	3
3	1

suggested answer

```
SELECT L.member_code, COUNT (DISTINCT C.book_code)
FROM Loans L, copies C
WHERE L.copy_number = C.copy_number
GROUP BY L.member_code ;
```

20. Average of days that each book has been borrowed.

member_code	copies
1	2
2	3
3	1

suggested answer

```
SELECT C.book_code, AVG (P.returning_date – P.date)
FROM Loans L, Copies C
WHERE L.copy_number = C.copy_number AND
      L.returning_date IS NOT NULL
GROUP BY C.book_code;
```

21. Amount of copies for each book edited by Addison-Wesley. Retrieve the title, language, author and amount of copies for each book sorted by amount of copies and author.

title	language	author	copies
Introduction to database desing	English	Danson	2
SQL para principiantes	Spanish	Danson	3

suggested answer

```
SELECT B.title, B.language, B.author, COUNT (copy_number) As
      a_copies
FROM Books B, Copies C
WHERE B.book_code = C.book_code AND B.publisher= 'Addison-
      Wesley'
GROUP BY B.title, B.language, B.author
ORDER BY copies DESC, B.author;
```

22. Retrieve the code of members than have borrowed more than one book in 2008.

member_code
2

suggested answer

```

SELECT member_code
FROM Loans
WHERE date BETWEEN '1/1/08' AND '31/12/08'
GROUP BY member_code
HAVING COUNT (*) > 2;

```

23. Retrieve the code of books that cover more than one theme.

book_code
2

suggested answer

```

SELECT book_code
FROM Temes
GROUP BY book_code
HAVING COUNT (*) > 1;

```

24. For each author that has published more than one book with Addison, retrieve the amount of copies of all his/her books existing at the library, sorted by amount of copies and author.

author	copies
Dason	5

suggested answer

```

SELECT B.author, COUNT (*) AS copies
FROM Books B, Copies C
WHERE B.publisher = 'Addison' AND
B.book_code = C.book_code
GROUP BY B.author
HAVING COUNT (DISTINCT B.book_code) > 1
ORDER BY copies DESC, B.author;

```

25. Amount of loans made by the members that have borrowed some book in 2008.

member_code	total_loans
1	2
2	3

suggested answer

```
SELECT member_code, COUNT (*) as total_loans
FROM Loans
GROUP BY member_code
HAVING MAX (date) >= '1/1/01';
```

26. Title of books that have been borrowed in 2008.

title
Introduction to database design
SQL para principiantes

suggested answer

```
SELECT title
FROM Books
WHERE book_code IN (SELECT book_code
                    FROM Loans
                    WHERE date > '1/1/08');
```

27. Title of the book with more editions.

title
Introduction to database design

suggested answer

```
SELECT title
FROM Books
WHERE editions >= ALL (SELECT editions
                      FROM Books);
```

28. Name and family name of members that have some non-returned book. (Try to solve the query in at least three different ways).

name	fname
Eric	Matew
David	Rooney

suggested answer

SOLUTION1:

```

SELECT M.name , M.fname
FROM members M
WHERE EXISTS (SELECT *
              FROM Loans L
              WHERE returning_date IS NULL AND
              L.member_code = M.member_code);

```

SOLUTION 2:

```

SELECT nom, cognoms
FROM Members M
WHERE EXISTS
      ( SELECT *
        FROM Loans L
        WHERE P.member_code = S.codi AND
              P.returning_date IS NULL);

```

SOLUTION 3:

```

SELECT nom, cognoms
FROM Socis
WHERE codi IN
      ( SELECT member_code
        FROM Loans
        WHERE returning_date IS NULL);

```

SOLUTION 4:

```

SELECT S.nom, S.cognoms
FROM Members M
WHERE 0 <
      ( SELECT COUNT (*)
        FROM Loans L
        WHERE P.member_code = S.codi AND
              P.returning_date IS NULL);

```

SOLUTION 5:

```

SELECT nom, cognoms
FROM Socis
WHERE codi = ANY
      ( SELECT member_code
        FROM Loans

```

```
WHERE returning_date IS NULL);
```

SOLUTION 6:

```
SELECT DISTINCT S.nom, S.cognoms  
FROM Members M, Loans L  
WHERE S.codi = P.member_code AND  
P.returning_date IS NULL ;
```

29. Insert all books of Dason as database books.

suggested answer

```
INSERT INTO Temes  
VALUES (SELECT L.codi, "Bases de Dades"  
FROM Books B  
WHERE B.author = "Date") ;
```

30. Record the information that the member of id 333 have returned today to the library the book Introduction to database design.

suggested answer

```
UPDATE Loans  
SET returning_date = CURRENT_DATE  
WHERE member_code = (SELECT codi  
FROM Socis  
WHERE DNI = "33445566")  
  
AND  
copy_number IN (SELECT E.copy_number  
FROM Books B, Copies C  
WHERE (L.title = "Introducción a los  
Sistemas de Bases de  
Datos") AND  
(L.codi=C.book_code)) ;
```


31. Delete all the returned loans made before the 1st of January of 2000.

suggested answer

```
DELETE
FROM Loans
WHERE date < '1/1/200' AND
       returning_date IS NOT NULL ;
```



The CYCLE RACING Database

We want to maintain some information about a cyclist race (such as the Tour de France, Il Giro di Italia, La Vuelta a España, etc.). In order to do this, we have defined a relational database whose schema is shown next:

The tables needed are:

- TEAMS Information about the teams that take part in the cycle racing
- CICLISTS Information about the racing cyclists that compete. Each cyclist belongs to a team.
- STAGES Information about the different stages which the competition is divided into.
- MOUNTAINS Information about the mountain passes that must be climbed.
- JERSEYS Information about the maillots that can be won as a prize.
- WEAR Information about the different maillots that a cyclist has worn.

Relational Schema

TEAMS (team, coach)

primary key: team

CYCLISTS (dorsal, name, age, team)

primary key: dorsal

foreign key: name → TEAMS

NNV: team

STAGES (stage, km, departure, arrival, dorsal)

primary key: stage

foreign key: dorsal → CICLIST

MOUNTAINS (mountain, height, category, slope, stage, dorsal)

primary key: mountainpass
 foreign key: stage → STAGES
 foreign key: dorsal → CICLISTS
 NNV: stage

MAILLOTS (maillot, type, color)
 primary key: maillot

WEAR (dorsal, stage, maillot)
 primary key: stage, maillot
 foreign key: stage → STAGES
 foreign key: dorsal → CICLISTS
 foreign key: maillot → MAILLOTS
 NNV: dorsal

Table Descriptions

□ TEAMS

columns	datatype	description
team	varchar(25)	name of the team
coach	varchar(30)	name of the coach of the team

□ CYCLISTS

columns	datatype	description
dorsal	integer	cyclist number assigned to the cyclist during the racing
name	varchar(30)	name of the cyclist
age	integer	age of the cyclist
team	varchar(25)	name of the team which the cyclist belongs to

□ STAGES

columns	datatype	description
stage	integer	stage number in the race
km	integer	how many kilometres the stage has
departure	varchar(35)	name of the city where the stage starts
arrival	varchar(35)	name of the city where the stage finishes
dorsal	integer	number of the cyclist who has won the

stage

❑ MOUNTAINS

columns	datatype	description
mountain	varchar(30)	name of the mountain pass
height	integer	maximum height in the pass
category	char	category of the pass
slope	integer	mean slope of the pass (in %)
stage	integer	stage number where mountain pas is climbed
dorsal	integer	number of the cyclist who has won the mountain pass

❑ MAILLOTS

datatype	description
maillot	varchar(3) code of the maillot
type	varchar(20) indicates the prize level of the maillot
colour	varchar(15) colour of the prize

❑ WEAR

columns	datatype
dorsal	integer
stage	integer
maillot	varchar(3)

(*) The cyclist with number *dorsal* who has worn the maillot identified by *maillot* at the stage with number *stage*.



ACTIVITY 2.5 – Querying the “cycle rycing” database

objective

To practise doing queries of different difficulty level on a complex database.

Write the following queries:

Queries on one single table

1. List the code, the type, the colour and the prize of all the jerseys (maillots) in the database.
2. List the cyclist number (*dorsal*) and the name of the cyclists who are 25 or over.
3. List the name and the height of all category 'E' (special) mountain passes (“puerto”).
4. List the value of the stage attribute for those stages with departure and arrival in the same city.
5. How many cyclists are there in the database?
6. How many cyclists are there over 25 are there?
7. How many teams are there?
8. List the average age of all cyclists.
9. List the minimum and maximum height of the mountain passes (“puerto”).

Queries on several tables

10. List the name and the category of the mountain passes (“puerto”) won by cyclists from the 'Banesto' team.
11. List the name of each mountain pass, also showing the number (stage) and the kilometres of the stage in which the mountain pass is (“puerto”).
12. List the name and the coach of the teams having at least one cyclist of

more than 33 years old.

13. List the name of the cyclists with the colour of each jersey (maillot) they have worn.

14. List the name of a cyclist and the number of the stage such that the cyclist has won the stage and has worn the yellow jersey ('maillot' with colour = 'Amarillo') at least once.

15. List the value of the stage attribute of the stages which do not start in the same city where the previous stage finished.

Queries with subqueries

16. List the value of the attribute stage and the departure city for those stages with no mountain passes.

17. List the average age of the cyclists who have won a stage.

18. Select the name of the mountain passes with a height greater than the average height of all the mountain passes.

19. List the name of the departure and the arrival of the stages where the steepest mountain passes are located.

20. List the cyclist number (dorsal) and the name of the cyclists who have won the highest mountain passes.

21. List the name of the youngest cyclist.

22. List the name of the youngest cyclist who has won at least one stage.

23. List the name of the cyclists who have won more than one mountain pass.

24. List the value of the stage attribute for those stages such that all the mountain passes in them are more than 700 metres high.

25. List the name and the coach of the teams such that all their cyclists are more than 25 years old.

26. List the cyclist number and the name of the cyclists such that all the stages they have won are more than 170 km long (i.e. they have only won stages longer than 170 km).

27. List the name of the cyclists who have won all the mountain passes in one

stage and have also won the stage.

28. List the name of the teams such that all their cyclists have worn some jersey ('maillot') or have won some mountain pass.

29. List the code and the colour of those jerseys ('maillots') which have only been worn by cyclists of the same team.

30. List the name of those teams such that their cyclists have only won mountain passes of category = 1.

Queries with Group By

31. List the value of the stage attribute of those stages which have mountain passes, also indicating how many it has.

32. List the name of the teams which have cyclists, indicating how many cyclists there are in the team.

33. List the name of all the teams, indicating how many cyclists there are in each team.

34. List the coach and the name of the teams which have more than 3 cyclists and with an average age lower or equal to 30.

35. List the name of the cyclists who belong to a team which has more than five cyclists and have also won one or more stages. Please also indicate how many stages he has won.

36. List the name of the teams and the average age of the cyclists of those teams who have the highest average age of all the teams.

37. List the coach of the teams whose cyclists have worn jerseys (of any type) more days than the rest. Note: each tuple in the Llevar relation indicate that a cyclist has worn a jersey one day.

General queries

38. List the coach of the teams whose cyclists have worn jerseys (of any type) more days than the rest. Note: each tuple in the Llevar relation indicate that a cyclist has worn a jersey one day.

39. List the value for the 'stage' attribute, the departure city and the arrival city of the stages longer than 190 km. and with at least two mountain

passes.

40. List the cyclist number and the name of the cyclists who have not worn all the jerseys (maillots) worn by the cyclist with number 20.
41. List the cyclist number and the name of the cyclists who have worn at least one of the jerseys (maillot) worn by the cyclist with number 20.
42. List the cyclist number and the name of the cyclists who have not worn any of the jerseys worn by the cyclist with number 20.
43. List the cyclist number and the name of the cyclists who have worn all the jerseys (maillots) worn by the cyclist with number 20.
44. List the cyclist number and the name of the cyclists who have worn exactly the same jerseys (maillots) as the cyclist with number 20.
45. List the cyclist number and the name of the cyclist who has worn the same jersey during more kilometres than any other cyclist, and also indicate the colour of this jersey.
46. List the cyclist number and the name of the cyclists who have worn three types of jersey less than the jerseys worn by the cyclist with number 1.
47. List the value of the stage attribute and the length of the stages (in km) which have mountain passes.

Glossary II

English – Catalan Vocabulary

English	Meaning
Amount	Quantitat
As follows	Com segueix, com es veu a continuació
Challenge	Repte
Column	Columna
Distinct	Distint, diferent
Distributor	Subministrador, distribuïdor
Either... or	O... O
Field	Camp
Highlighted	Resaltat
Instead of...	En lloc de..
Pattern	Patró
Row	Fila
Several	Diverses
To check	Comprovar
To delete	Esborrar
To display	Mostrar per pantalla, visualitzar
To enclose	Tancar
To fulfil	Acomplir (una condició)
To match	Coincidir
To pay attention	Prestar atenció
To provide	Proveir, proporcionar
To remove	Esborrar, eliminar
To retrieve	Recuperar
To return	Retornar, tornar
To search	Cercar
To sort	Ordenar
To update	Actualitzar
Within	Dins de

English	Meaning
Asterisk	Asterisc
Between	Entre
Comma	Coma
Field	Camp
Label	Etiqueta
Prefixe	Prefix
Quotes	Cometes
Single quotes	Cometes simples
To concatenate	Concatenar
To enable	Fer
To join	Ajuntar
To make up	Formar
To perform	Realitzar
To relate to	Relacionar-se amb
To report	Presenter, mostrar
To supply	Subministrar
Wildcard	Comodí
Contents	Contingut

Key vocabulary

At the end of this lesson the students must be able to recognise and understand the meaning of the following words, commands, clauses and functions:

AGGREGATE FUNCTIONS

ALL OPERATOR

ANY OPERATOR

AVG()

BASIC JOIN

BETWEEN OPERATOR

CARTESIAN PRODUCT

COUNT()

EXISTS OPERATOR

FULL OUTER JOIN

GROUP BY

GROUP BY.. HAVING

IN OPERATOR

INNER JOIN

INSERT... SELECT

INSERT...VALUES

IS NULL

JOIN

LEFT OUTER JOIN

LIKE OPERATOR

MATCHING CONDITION

MAX()

MIN()

NESTED QUERIES

NOT OPERATOR

NULL VALUE

ORDER BY

OUTER JOIN

QUERY

RIGHT OUTER JOIN

SELECT... FROM

SELECT... WHERE

SUBQUERIES

SUM()

UPDATE... VALUES

Lesson Plan		PART II - STRUCTURED QUERY LANGUAGE	
Lesson 2. Data Manipulation Language			
TIMING	26 hours	LEVEL	1 st CFGS DAI/ASI
OBJECTIVES	CONTENT	<ul style="list-style-type: none"> □ Altering the contents of the database: <ul style="list-style-type: none"> - INSERT command. - DELETE command. - UPDATE command. □ Retrieving data from the database: <ul style="list-style-type: none"> - SELECT command and FROM, WHERE and DISTINCT clauses. - ORDER BY clause. - WHERE operators: BETWEEN, IN, LIKE, NOT, IS NULL. - Calculated fields. - Aggregate functions: COUNT(), MAX(), MIN(), AVG(), SUM() - GROUP BY and HAVING clauses. - JOINS. - Subqueries. - ALL, ANY and EXISTS operators. 	
	COGNITION	<ul style="list-style-type: none"> - To decide the tables from where the information has to be retrieved, edited, or removed. - To decide the information to be retrieved, edited, removed or inserted. - To identify the operations to be carried out to retrieve the wanted data. - To decide the commands and clauses to be used to carry out the wanted operations. 	

	CULTURE	The importance of retrieve the correct information when need it.																					
	COMMUNICATION	<p>Language for learning:</p> <ul style="list-style-type: none"> - To discuss and ask about the concepts - To justify their answers and reasoning <p>Language of learning:</p> <ul style="list-style-type: none"> - Language structures needed to complete the tasks <p>Language trough learning:</p> <ul style="list-style-type: none"> - Language to carry out the different activities 																					
LEARNING OUTCOMES	<p>At the end of this lesson, students will be able to...</p> <ul style="list-style-type: none"> - To write complex queries to retrieve data from the database using interactive SQL. - To add new records in a table. - To remove an existing record. - To edit data of an existing record. 																						
RESOURCES AND TIMING	<p>Lesson in power point (<i>P2.L3.DDL.ppt</i>)</p> <table border="1" style="margin-left: 20px;"> <tr> <td colspan="3">Lesson 2.- Data Manipulation Language</td> </tr> <tr> <td colspan="3">(6 hours teaching + activity 2.1.</td> </tr> <tr> <td colspan="3">20 hours activities 2.2 to 2.4)</td> </tr> <tr> <td>Activity 2.1</td> <td>Querying the "distributors" database</td> <td>----</td> </tr> <tr> <td>Activity 2.2</td> <td>Querying the "projects" database</td> <td>6 hours</td> </tr> <tr> <td>Activity 2.3</td> <td>Queryin the "library" database</td> <td>6 hours</td> </tr> <tr> <td>Activity 2.4</td> <td>Querying de "cycle rycing" database</td> <td>8 hours</td> </tr> </table>		Lesson 2.- Data Manipulation Language			(6 hours teaching + activity 2.1.			20 hours activities 2.2 to 2.4)			Activity 2.1	Querying the "distributors" database	----	Activity 2.2	Querying the "projects" database	6 hours	Activity 2.3	Queryin the "library" database	6 hours	Activity 2.4	Querying de "cycle rycing" database	8 hours
Lesson 2.- Data Manipulation Language																							
(6 hours teaching + activity 2.1.																							
20 hours activities 2.2 to 2.4)																							
Activity 2.1	Querying the "distributors" database	----																					
Activity 2.2	Querying the "projects" database	6 hours																					
Activity 2.3	Queryin the "library" database	6 hours																					
Activity 2.4	Querying de "cycle rycing" database	8 hours																					

Part 3. Lesson 3

Data Definition Language

Introduction

Before using any of the data manipulation statements, it is also important to create a good database.

The Data Definition Language (DDL) is the part of SQL you use to create, change, or destroy the basic elements of a relational database. Basic elements include tables, views and other things as well. In this lesson, we will discuss the the commands that operate on these elements.

Approach

DDL is more mechanical than DML and, therefore, easier.

In this lesson the students will use DDL commands to create and manipulate the structure of some of the databases they have worked with in the previous lesson.



ACTIVITY 3.1 – Creating the “distributors database”

objective

To work on the CREATE command.

- Point out the data type you would use for each field in the result tables.
- Run MySQL and create a new database. Save it as “distributors”.
- Create the tables using the SQL *CREATE TABLE* statements.

suggested answer

```
CREATE TABLE pieces
(p_id      varchar(2),
 p_name    varchar(20),
 p_colour  varchar(10),
 p_weight  integer,
 p_city    varchar(30),
 CONSTRAINT pk PRIMARY KEY (p_id) );
```

```
CREATE TABLE distributors
(d_id      varchar(2)
 d_name    varchar(10),
 d_age     integer,
 d_city    varchar(20),
 CONSTRAINT pk PRIMARY KEY (d_id) );
```

```
CREATE TABLE supplies
(d_id      varchar(2),
 p_id      varchar(3),
 amount    integer
 CONSTRAINT fk1 FOREIGN KEY (d_id) REFERENCES distributors,
 CONSTRAINT fk2 FOREIGN KEY (p_id) REFERENCES pieces,
 CONSTRAINT pk PRIMARY KEY (d_id, p_id) );
```



Extra information for the teacher...

Since that's the first exercise about creating structures, eventually the teacher should also take care of discussing about the solutions and explaining the more difficult regarded aspects.



ACTIVITY 3.2 – Modifying the “distributors” database

objective

To modify a simple database structure.

To do this exercise, work with a copy of the “distributors” copy (don’t do it over the original database).

Carry out the following modifications over the tables:

a) Change the city of P3 to Hamilton.

suggested answer

```
UPDATE pieces
SET p_city= 'Hamilton'
WHERE p_id='P3';
```

b) Change the distributor code for the first supply to D6

suggested answer

```
UPDATE supplies
SET d_id= 'D6
WHERE d_id='D1' AND p_id='P1';
```

The above modification will result in an error. Why?

suggested answer

Referential integrity is being broken since there isn’t any distributor with the code D6.

c) Delete record of distributor Adams from distributors table.

suggested answer

```
DELETE FROM distributors
```

WHERE d_name= 'Adams';
d) Delete record of distributor Smith from distributors table.
suggested answer
DELETE FROM distributors WHERE d_name= 'Smith';
b) The above modification will result in an error. Why?
suggested answer
Referential integrity is being broken since, after deleting the record, there would be supplies of a non-existing distributor.
e) Change the name of the pieces table to products.
f) Add a column price of type integer to the products table.
g) Alter the price column so that the default price is 0.
h) How could you remove (drop) price again?



ACTIVITY 3.3 – *Creating the "library database"*

objective

To think about the data type that will contain the fields as well as the necessary constraints that the tables must have.

- a) Point out the data type you would use for each field in the result tables as well as the column constraints you think are needed.
- b) Run MySQL and create a new database. Save it as "library".

suggested answer

First we will create the database:

```
CREATE DATABASE library
```

Then, we'll create the tables:

```
CREATE TABLE books
(book_code varchar(3) PRIMARY KEY,
title varchar(50),
publisher varchar(20),
language varchar(20)
author varchar(30),
editions integer DEFAULT (1)
);
```

```
CREATE TABLE copies
(copy_number varchar(3) PRIMARY KEY,
book_code varchar(3),
year_edition integer,
number_edition varchar(20),
```

```
CONSTRAINT fk FOREIGN KEY (book_code) REFERENCES books
(book_code)
```

```
);
```

```
CREATE TABLE themes
```

```
( book_code varchar(2),
  theme      varchar(3),
```

```
  CONSTRAINT pk PRIMARY KEY (book_code, theme),
```

```
  CONSTRAINT fk FOREIGN KEY (book_code) REFERENCES books
  (book_code)
```

```
);
```

```
CREATE TABLE members
```

```
( member_code varchar(2), PRIMARY KEY
```

```
  name          varchar(15),
```

```
  fname         varchar(20),
```

```
  id            varchar(8), NOT NULL UNIQUE
```

```
  address       varchar(50),
```

```
);
```

```
CREATE TABLE loans
```

```
( member_code varchar(3),
```

```
  copy_number  varchar(3),
```

```
  date         date,          DEFAULT CURRENT_DATE()
```

```
  deadline     date,          NOT NULL UNIQUE
```

```
  returning_date date,
```

```
  CONSTRAINT pk PRIMARY KEY (member_code, copy_number, date),
```

```
  CONSTRAINT fk1 FOREIGN KEY (member_code) REFERENCES
  members,
```

```
  CONSTRAINT fk2 FOREIGN KEY (copy_number) REFERENCES copies
```

```
);
```



ACTIVITY 3.4 – Modifying the “library” database

objective

To use the necessary commands to make modifications on the table structures.

To do this exercise, work with a copy of the “library” database (don’t do it over the original database).

Make the following changes over the tables:

a) Add a new field in the members table to store the telephone number.

suggested answer

```
ALTER TABLE member {ADD COLUMN phone varchar(9)};
```

b) Prevent the title of books from having the same name.

suggested answer

```
ALTER TABLE books {ALTER COLUMN title varchar(50) UNIQUE };
```

c) Change the address of the member 1 to ‘Edinburgh’.

suggested answer

```
UPDATE members
SET address= ‘Edinburgh’
WHERE member_code=’1’;
```

d) - Change the code of the member for the first loan to 4.

suggested answer

```
UPDATE loans
SET member_code= ‘4’
WHERE member_code=’1’ AND copy_number=’1’
```

AND date = '15/10/2008';
- The above modification will result in an error. Why?
Referencial integrity is being broken since there isn't any member with code 1.
e) Delete record of copy 1 from copies table.
suggested answer
DELETE FROM copies WHERE copy_number= '1';
f) 1. Delete record of book 1 from books table.
<u>Suggested answer:</u>
DELETE FROM books WHERE book_number= '1';
2. The above modification will result in an error. Why? How could have this problem been avoided?
suggested answer
Referencial integrity is being broken since, after deleting the record, there would be copies of an non-existing book. We could have included cascading delete and update actions on the foreign keys in the copies table.. CREATE TABLE copies (copy_number varchar(3) PRIMARY KEY, book_code varchar(3), year_edition integer, number_edition varchar(20), CONSTRAINT fk FOREIGN KEY (book_code) REFERENCES books (book_code) ON DELETE CASCADE ON UPDATE CASCADE);

g) Can you delete the loan for the member 3?

suggested answer

You can do:

```
DELETE FROM loans  
WHERE member_code='3';
```

But afterwards you'll have removed the information of a loan for a non-returned book. This person might not return the book ever.

h) How could you remove the phone again of the members table?

suggested answer

ALTER TABLE command can be used only to add or modify columns within a table. To remove columns, a new table should be created with the desired format, and then the records from the old table should be selected into that new table.

i) Remove the loans table.

suggested answer

```
DROP TABLE loans;
```


Glossary III

English - Catalan Vocabulary

English	Meaning
Constraint	restricció
To disallow	No permetre, impedir
To fulfil	acomplir
Primary key	Clau primària
Foreign key	Clau externa, forània
To check	comprovar
To enable	Permetre, fer possible
To type	teclatjar
View	vista
To drop	rebutjar

Key Vocabulary

At the end of this lesson the students must be able to recognise and understand the meaning of the following words, commands, clauses and functions:

CREATE DATABASE

COLUMN CONSTRAINT

TABLE CONSTRAINT

NOT NULL

UNIQUE

PRIMARY KEY

DEFAULT

ALTER TABLE

DROP TABLE

CREATE VIEW

DROP VIEW

CHECK

FOREIGN KEY

Lesson Plan	PART II - STRUCTURED QUERY LANGUAGE		
Lesson 3. Data Definition Language			
TIMING	4 hours	LEVEL	1 st CFGS DAI/ASI
OBJECTIVES	CONTENT	<ul style="list-style-type: none"> - Creation of a database. CREATE DATABASE. - Creation, change and elimination of tables. CREATE TABLES, ALTER TABLE and DROP TABLE. - Creation and destruction of views. CREATE VIEW and DROP VIEW 	
	COGNITION	<ul style="list-style-type: none"> - To decide the structure of the tables we need to store the information. - To identify the operations to be carried out to create the needed structures. - To decide the commands and clauses to be used to carry out the wanted operations. 	
	CULTURE		
	COMMUNICATION	<p>Language for learning:</p> <ul style="list-style-type: none"> - To discuss and ask about the concepts - To justify their answers and reasoning <p>Language of learning:</p> <ul style="list-style-type: none"> - Language structures needed to complete the tasks <p>Language trough learning:</p> <ul style="list-style-type: none"> - Language to carry out the different activities 	
LEARNING OUTCOMES	<p>At the end of this lesson, students will be able to...</p> <ul style="list-style-type: none"> - To create tables and build relationships using SQL datadefinitions commands 		

RESOURCES AND TIMING	Lesson in power point (<i>P2.L3.DDL.ppt</i>)	
	Lesson 3.- Data Definition Language (6 hours teaching + activity 4.1)	
	Activity 3.1	Creating the distributors database
	Activity 3.2	Modifying the distributors database
	Activity 3.3	Revision Activity. The library database.