# DATABASES

## Unit 2 **Interpretation of Entity Relationship Diagrams**

# Entity Relationship Diagram (ERD)

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases.

Peter Chen developed ERDs in 1976. Since then Charles Bachman and James Martin have added some slight refinements to the basic ERD principles.

At first glance an entity relationship diagram looks very much like a flowchart. It is the specialized symbols, and the meanings of those symbols, that make it unique.

An ER diagram is a means of visualizing how the information a system produces is related. There are six main components of an ERD: Entities, Relationships, Attriibutes, Connecting lines, Cardinality and Ordinality.

# Entities

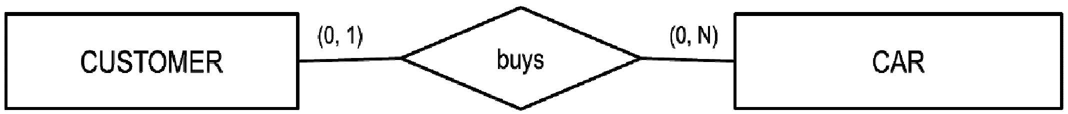
They are represented by rectangles. An entity is an object or concept about which you want to store information:


A weak entity cannot be uniquely identified by its own attributes alone. The existence of its occurrences depends on the existence of occurrences in other entities. For an entity to be weak, it must be weak with respect to all the entities with which it is related:

# Relationships

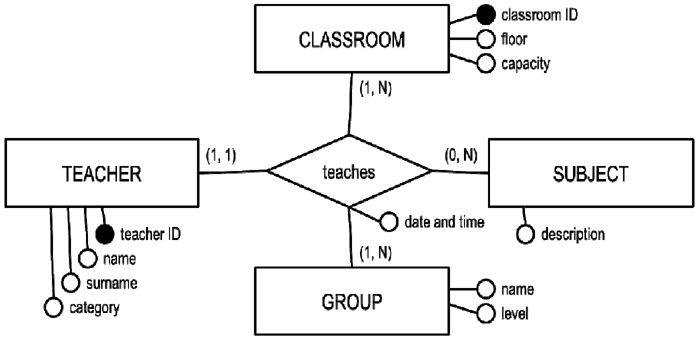
They are represented by diamond shapes, and show how two entities share information in the database.


Binary relationship

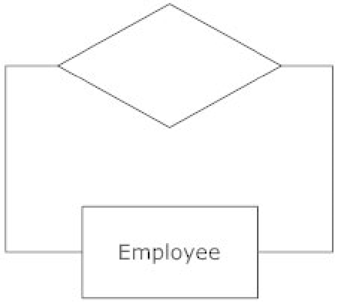


# Relationships

N-ary:




In some cases, entities can be unary (self-linked). For example, employees can supervise other employees.



# Attributes

They are represented by ovals. A key attribute is the unique, distinguishing characteristic of an entity or a relationship. For example, an employee's social security number might be the employee's key attribute.


# Connecting lines

Solid lines that connect attributes to show the relationships of entities in the diagram.

# Cardinality

Specifies how many instances of an entity relate to one instance of another entity. It can be:

- 1:N (one to N / one to many):

An occurrence of one entity can be related to several occurrences of another entity, but each occurrence of the second entity can only be related to a single occurrence of the first entity.

- M:N (M to N / many to many).

Each occurrence of an entity can be related to several of another entity, and each occurrence of the second entity can also be related to several occurrences of the first.

- 1:1 (one to one).

An occurrence of one entity is related to one and only one occurrence of another entity.

# Ordinality

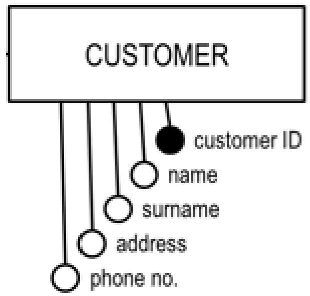
It defines the minimum and maximum number of occurrences of an entity that may be related to an occurrence of another or other entities, identifying optional relationships (in which there doesn't have to be correspondence). The ordinality is indicated on both sides of the relation, and its maximum value coincides with the value of the cardinality corresponding to the side of the relation in which we find ourselves. It can be of the following types:

- 0..1 (zero to one). Each occurrence of the entity can relate to an occurrence of the second entity or not. It can not relate to several.
- 1..1 (one to one). Each occurrence of the entity must necessarily relate to one and only one occurrence of the second entity.
- 1..N (one to N/ one to many). Each occurrence of the entity must necessarily relate to at least one occurrence of the second entity. It can relate to several.
- 0..N (zero to N / zero to many). Each occurrence of the entity has not limited its relation to occurrences of the second entity. It can be related to one, several or none.

# Primary key and candidate keys

In every entity there is at least one attribute or set of attributes whose values identify each of the occurrences unambiguously. These attributes are called candidate keys, and the one we choose to identify the occurrences of the entity will be the primary key. For reasons of performance of the resulting database, we always have to choose the smallest possible primary key.

In an entity-relationship diagram, the primary keys are represented by underlining the name of the attribute if you have chosen to draw the attributes inside an oval, or darkening the circle if you have decided to use the other type of graphic representation.



# Extended Entity-Relationship (EE-R) Model

Some authors added restrictions to the entity-relationship diagram that, in some cases, expanded its semantic magnitude. Although there is no unanimity on the need and suitability of these additions or on the notation to be used, these are some of the most recognizable elements of the extended entity-relationship diagram:

- **Exclusiveness (XOR).** The existence of a relationship prevents the existence of another. Suppose an educational center where a teacher can teach or work on a research project, but not do both at the same time.

- **Hierarchies.** The occurrences of an entity (supertype) are related to those of several entities (subtypes) that, in addition to sharing the attributes of the supertype, have their own attributes.

- **Aggregation**. Occurrences of several entities represent the parts of a whole represented in an occurrence of another entity.

# The relational model

Defined in 1970 by Edgar F. Codd at IBM Research Center (San José, CA). Its main element is the **relation**: a bidimensional structure that represents a group of data objects. Each data object, constituting a row, is a **tuple**, and attributes go by the name **attribute**.

Within the database scope, relations are named **tables**, tuples **rows** or **records** and attributes keep their original name.

attribute

value

tuple (row = record)

| ID | Name | Surname | Phone no. | Birthdate | Ins. date |
|---|---|---|---|---|---|
| 007895211 | Elizabeth | Jones | 2036785241 | 1983/09/13 | 2012/09/20 |
| 506874522 | Julian | Chapman | 2039876455 | 1963/02/05 | 2011/09/14 |
| 385469983 | Ian | McAlister | 2051123542 | 1978/10/24 | 2009/05/05 |
| 096538014 | Ruth | Hutchinson | 2038900017 | 1981/05/15 | 2013/09/04 |

# Normalization

Also to Codd is the definition of a series of rules whose application eliminates the redundancies of information in a relational solution. The technique is known as normalization, and involves carrying all relationships to certain states called normal forms. These normal forms are described below and how they are reached.

**First normal form (1NF):** A relation is in 1FN if all its values are atomic, that is, each value of the domains of all the attributes is unique. In the following example (primary key in bold) we see an attribute, "Phone", whose values are not atomic but repetitive:

| **NIF** | Nombre | Apellidos | Teléfono |
|---|---|---|---|
| 00789521T | Paula | Sanz González | 619554687<br>915196347 |
| 09653801B | José Luis | García Viñals | 667859621<br>914079880<br>913200121 |
| 50687452Y | Ruth | Lázaro Cardenal | 689330247 |

# Normalization

**First normal form (1NF): (cont.)**

A first solution to achieve the 1FN is to atomize the attribute "Phone", as follows:

| NIF | Nombre | Apellidos | Teléfono |
|---|---|---|---|
| 00789521T | Paula | Sanz González | 619554687 |
| 00789521T | Paula | Sanz González | 915196347 |
| 09653801B | José Luis | García Viñals | 667859621 |
| 09653801B | José Luis | García Viñals | 914079880 |
| 09653801B | José Luis | García Viñals | 913200121 |
| 50687452Y | Ruth | Lázaro Cardenal | 689330247 |

This solution implies a strong redundancy ("Name" and "Surnames" are repeated for each telephone), and invalidates "NIF" as the primary key, forcing the extension of said primary key with the "Telephone" attribute.

# Normalization

**First normal form (1NF): (cont.)**

For this reason, a more elaborate solution is proposed consisting of dividing the original relationship into two (one with the people and the other with the telephones), linking them through the values of the original primary key:

| NIF | Nombre | Apellidos |
|---|---|---|
| 00789521T | Paula | Sanz González |
| 09653801B | José Luis | García Viñals |
| 50687452Y | Ruth | Lázaro Cardenal |

| NIF | Teléfono |
|---|---|
| 00789521T | 619554687 |
| 00789521T | 915196347 |
| 09653801B | 667859621 |
| 09653801B | 914079880 |
| 09653801B | 913200121 |
| 50687452Y | 689330247 |

# Normalization

**Second normal form (2NF):**

A relationship is in 2NFif you meet the following rules:

- It is in 1NF.
- All the attributes that are not part of the primary key depend on it completely.

The following relationship illustrates the inventory of a library:

| Cód. libro | Cód. tienda | Cantidad | Dirección |
|---|---|---|---|
| 342 | 12 | 3 | C/Luchana, 34 |
| 268 | 9 | 1 | Pº Castellana, 132 |
| 250 | 10 | 5 | C/General Ricardos, 145 |
| 181 | 9 | 7 | Pº Castellana, 132 |

# Normalization

**Second normal form (2NF):**

| Cód. libro | Cód. tienda | Cantidad | Dirección |
|---|---|---|---|
| 342 | 12 | 3 | C/Luchana, 34 |
| 268 | 9 | 1 | Pº Castellana, 132 |
| 250 | 10 | 5 | C/General Ricardos, 145 |
| 181 | 9 | 7 | Pº Castellana, 132 |

The primary key is composed of two ("Store Code"), but the "Address" attribute does not depend on the entire key, but only on the attribute "Store Code". That is why the address of store 9 is repeated, with the redundancy of additional information. In 2FN it will be like this:

| Cód. libro | Cód. tienda | Cantidad |
|---|---|---|
| 342 | 12 | 3 |
| 268 | 9 | 1 |
| 250 | 10 | 5 |
| 181 | 9 | 7 |

| Cód. tienda | Dirección |
|---|---|
| 12 | C/Luchana, 34 |
| 9 | Pº Castellana, 132 |
| 10 | C/General Ricardos, 145 |

# Normalization

**Third normal form (3NF):**

A relationship is in 3NF if you meet the following rules:

- It is in 2NF.
- All attributes that are not part of the primary key are independent of each other, that is, they do not give any information about other attributes of the relationship.

The following example illustrates a relationship with information about employees. All attributes depend directly on the primary key ("Employee Code") except "Department Name", which depends on "Department Code":

| Cód. emp. | Nombre | Apellidos | Dirección | Cód. dpto. | Nombre dpto. | Fecha nac. |
|---|---|---|---|---|---|---|
| 12 | Paula | Sanz González | C/Mtnez. Izqdo., 40 | 3 | Financiero | 13/09/1983 |
| 268 | José Luis | García Viñals | Pº Melancólicos, 1 | 2 | Informática | 05/02/1963 |
| 250 | Javier | Peinado Martín | C/Guitarra, 7 | 5 | RRHH | 24/10/1978 |
| 181 | Ruth | Lázaro Cardenal | C/Torrelaguna, 64 | 3 | Financiero | 15/05/1981 |

# Normalization

**Third normal form (3NF): (cont.)**

The information about the department will constitute a new relationship. In 3FN then:

| Cód. emp. | Nombre | Apellidos | Dirección | Cód. dpto. | Fecha nac. |
|---|---|---|---|---|---|
| 12 | Paula | Sanz González | C/Mtnez. Izqdo., 40 | 3 | 13/09/1983 |
| 268 | José Luis | García Viñals | Pº Melancólicos, 1 | 2 | 05/02/1963 |
| 250 | Javier | Peinado Martín | C/Guitarra, 7 | 5 | 24/10/1978 |
| 181 | Ruth | Lázaro Cardenal | C/Torrelaguna, 64 | 3 | 15/05/1981 |

| Cód. dpto. | Nombre dpto. |
|---|---|
| 3 | Financiero |
| 2 | Informática |
| 5 | RRHH |

# ER Model to Relational Model

**Nomenclature: Hungarian notation**

Defined in the 70s by Charles Simonyi, Hungarian programmer of Xerox.

The use of characters from local alphabets, such as ñ, ç, vowels accented or with diaeresis, etc., will be avoided. Only the following characters will be accepted as valid:

- Anglo-Saxon alphabet letters in uppercase and lowercase.
- Numbers
- Underline sign ("_").

The first character of a name will always be a letter.

# ER Model to Relational Model

**Nomenclature: Hungarian notation**

➢ **Names of tables.**

They will be prefixed with a capital "T" (TEmployee, TClient).

➢ **Field names**

The prefix, a lowercase letter, will indicate the type of field data, according to the following basic types:

- Numbers: "n" (nQuantity, nTotal), of number.
- Chains of characters: "c" (cName, cCity), char.

# ER Model to Relational Model

**Nomenclature: Hungarian notation**

➢ **Field names (cont.)**

- Dates: "d" (dNacimiento, dAlta), of date. Note that it is no longer necessary to specify the word "date" (d Date Birth would be redundant).
- Logical values: "l" (lSexo, lPensionista).
- Objects: "o" (oFoto, oDocumentoXML).

➢ **Identifiers**

Any field created to uniquely identify the records of a table (customer code, book code, country code) will carry as suffix the letters "ID" (identifier) in upper case (nClienteID, nLibroID, nPaisID).

# ER Model to Relational Model

**Transformation rules**

- Tables are represented as a rectangle with the name of the table at the top and the list of fields at the bottom. The fields that make up the primary key will be underlined


- Cardinality will be implicit in the termination of the lines that relate tables:

# ER Model to Relational Model

**Transformation rules**

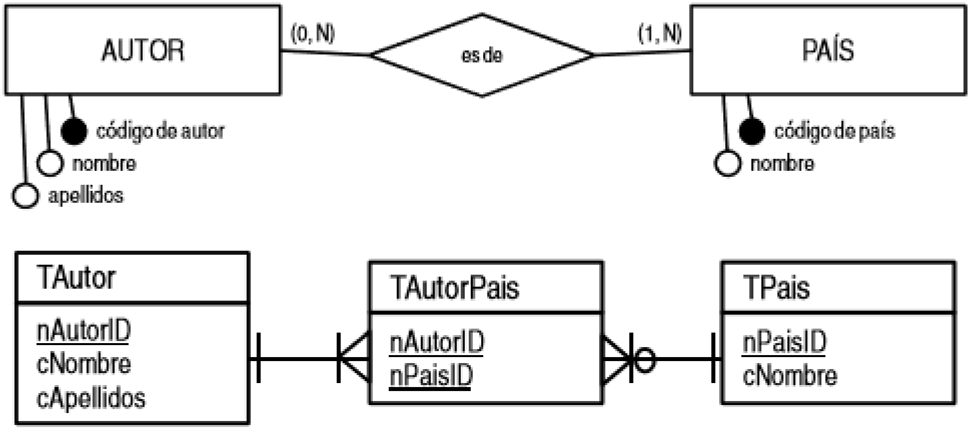
The transformation of components of the entity-relationship diagram into elements of the physical data model follows the following rules:

- Every entity becomes a table.
- Every attribute becomes a field.
- The attributes marked as part of the primary key are converted into primary key fields of the new table. Relationships present a casuistic based on their cardinality.

# ER Model to Relational Model

**Transformation rules. M:N Relationships**

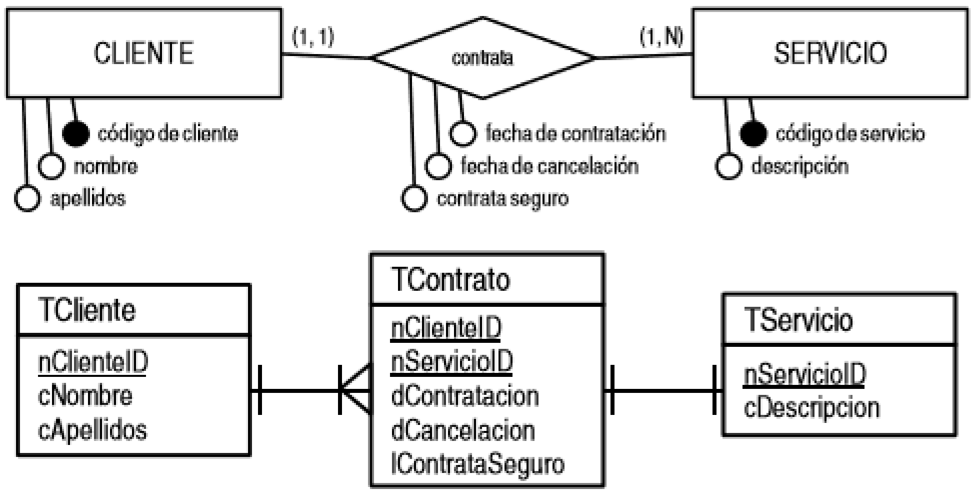
They become a table. Its primary key will be the concatenation of the primary keys of the entities that relate.



# ER Model to Relational Model

**Transformation rules. 1:N Relationships**

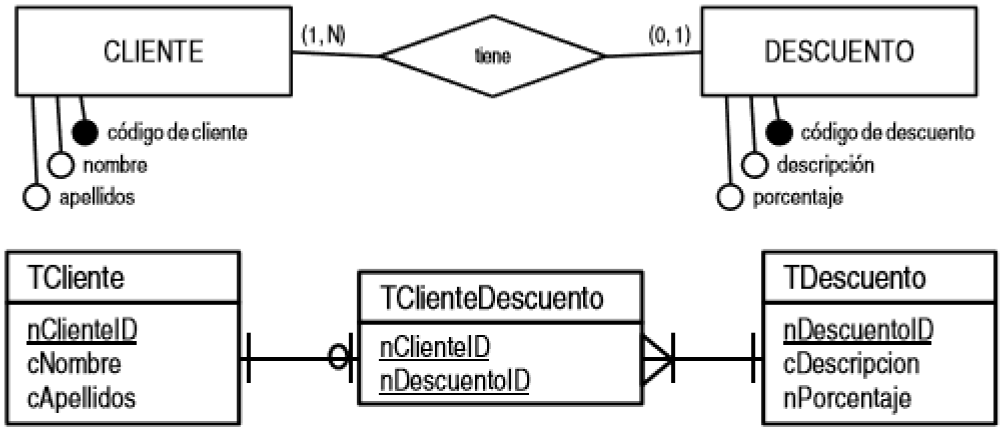
1) If the relationship has an attribute, it becomes a table:



# ER Model to Relational Model

**Transformation rules. 1:N Relationships**

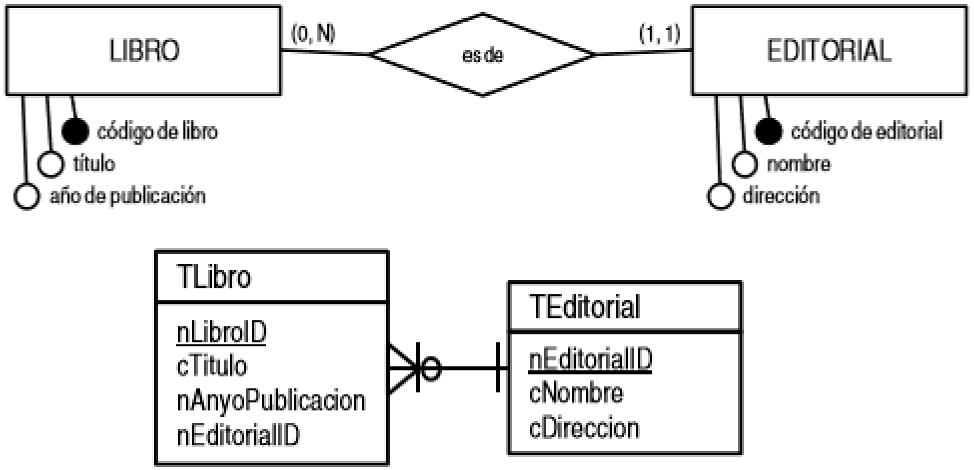
2) If there is a minimum ordinality of 0 on the entity whose cardinality is 1, the relationship becomes a table.



# ER Model to Relational Model

**Transformation rules. 1:N Relationships**

3) If there is a minimum ordinality of 1 on the entity whose cardinality is 1, and there are no attributes, the relationship doesn't become a table. The attributes that make up the primary key of the maximum entity with cardinality 1 will be propagated to the entity with cardinality N as foreign key/s:



# ER Model to Relational Model

**Transformation rules. 1:1 Relationships**

It is the most complex situation. The objective is to avoid at all costs that there are no value fields. It usually requires a detailed study of the specific circumstances of each case, but some rules can be defined for certain situations:

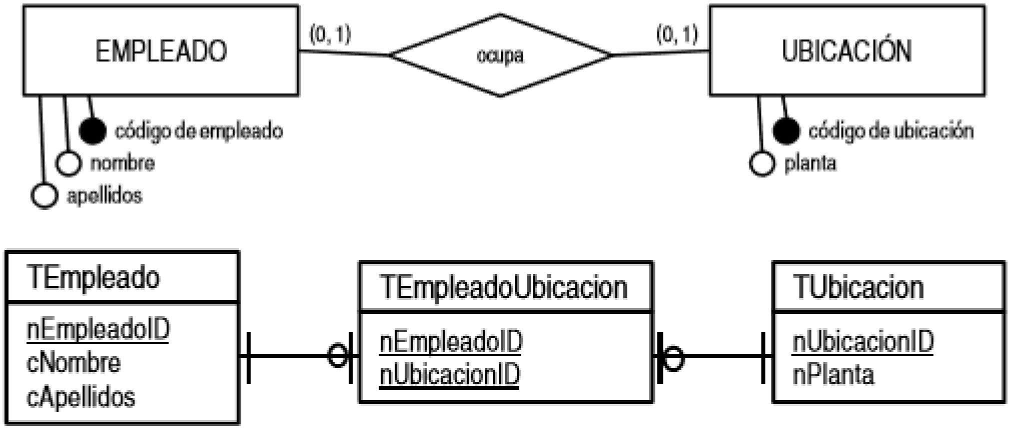
1) If both ordinalities are (1, 1) both entities can be transformed to a single table.

2) If the relationship has an attribute, it becomes a table.

# ER Model to Relational Model

**Transformation rules. 1:1 Relationships**

If both modalities are (0, 1) the relation generates a table:



# ER Model to Relational Model

**Transformation rules. 1:1 Relationships**

3) If one ordinality is (0,1) and the other (1, 1) the entity primary key with minimum ordinality 1 is propagated to the entity with minimum ordinality 0: