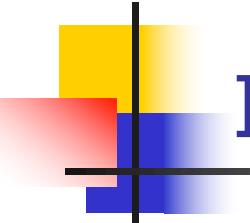


# Conceptos de Bases de Datos

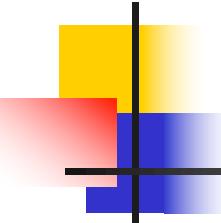
Tema 1. Apéndice A



# Indice

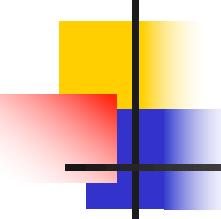
---

- Definición de Base de Datos y conceptos principales
- El Modelo E/R
- El Modelo Relacional
- Paso del E/R al Relacional
- DML
- Normalización



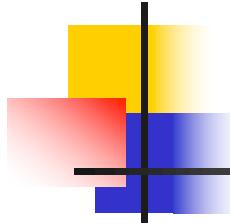
# Definición de Base de Datos

- *Colección de datos que están lógicamente relacionados entre sí, que tienen una definición y una descripción comunes y que están estructurados de una forma particular.*
- *Colección de programas que aseguran el acceso a los datos.*



## Redundancia en BD

- Un dato A es deducible de otros B, C, D, ..., si conocidos los valores de B, C, D, ... queda determinado el valor de A.
  - Si además de tener almacenados B, C, D, ... tenemos también almacenado A estamos ante un caso de **redundancia**.

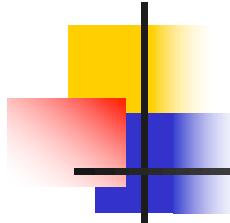


# Redundancia en BD

---

## Efectos de la redundancia:

- Almacenamiento supérfluo (problema menor)
- Inconsistencia (problema mayor)

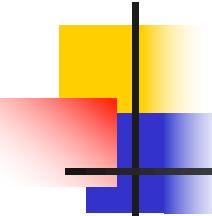


## Características de las BD

- *Integrada*
- *Compartida*

## Objetivos de Base de Datos

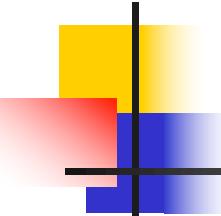
- *Almacenar*
- *Recuperar*



# El informe ANSI/SPARC

## Arquitectura en tres niveles:

- Nivel Externo
  - Cercano a los usuarios
- Nivel conceptual
  - Contenido Global
- Nivel Interno
  - Descripción a nivel físico



# El informe ANSI/SPARC

## TIPOS de ESQUEMAS:

- Conceptual: proporciona los nombres de las entidades, sus características y las relaciones que existen entre ellas.
- Externo: visión que tienen los usuarios de los datos que utilizan y operaciones sobre ellos. Por lo tanto, existirán tantos subesquemas como tipos de usuarios tenga la BD.
- Interno: descripción de las estructuras de almacenamiento, métodos de recuperación eficiente, dispositivos, etc.

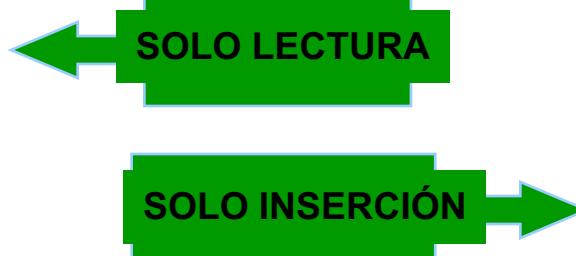
# E informe ANSI/SPARC

**SUBESQUEMA A**  
Nombre y salario  
de todos los  
empleados

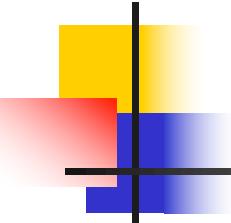
PERSONAL		
26777620	JOSE MIGUEL	1100
35699083	MARIA	1120
23990865	ESTEBAN	900
26758881	LAURA	1010
39995877	ELENA	880
33759080	LUIS	1100
33367765	FERNANDO	990

**SUBESQUEMA B**  
dni y nombre de  
empleados que  
ganan menos de  
1000

NÓMINA	
JOSE MIGUEL	1.100
MARIA	1.120
ESTEBAN	900
LAURA	1.010
ELENA	880
LUIS	1.100
FERNANDO	990



MALPAGADOS	
23990865	ESTEBAN
39995877	ELENA
33367765	FERNANDO



# El informe ANSI/SPARC

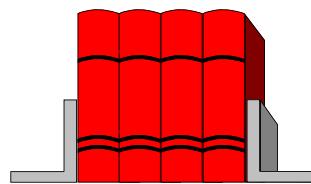
## OBJETIVO:

- Independencia de los datos: *propiedad de modificar un nivel sin que se vean afectados los niveles superiores.*
- Herramienta: correspondencias.
- Ventaja: persistencia de *Software*.

# Elementos de una Base de Datos



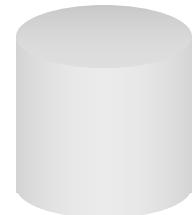
**Administrador**



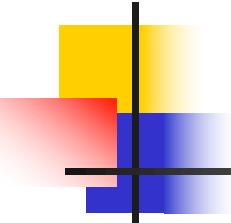
**Diccionario**



**Usuarios**



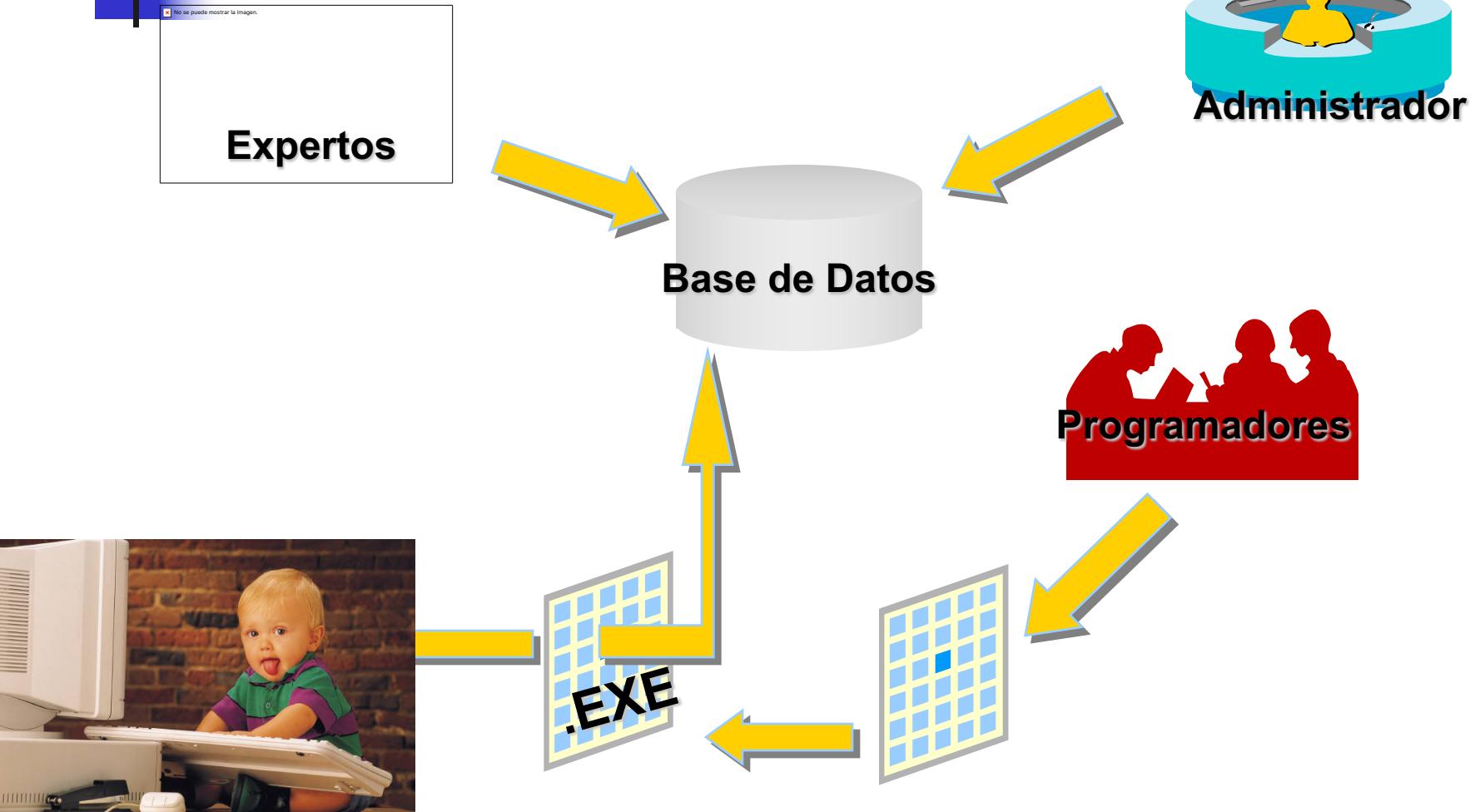
**Datos**

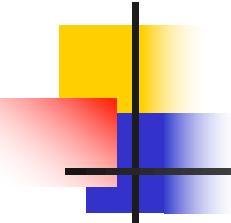


# El Gestor de la Base de Datos

*El SGBD es un conjunto coordinado de programas, procedimientos, lenguajes, etc., que suministra, tanto a los usuarios informáticos, como no informáticos y al Administrador, los medios necesarios para describir, recuperar y manipular los datos integrados en la BD, asegurando su confidencialidad y seguridad.*

# Elementos de una Base de Datos



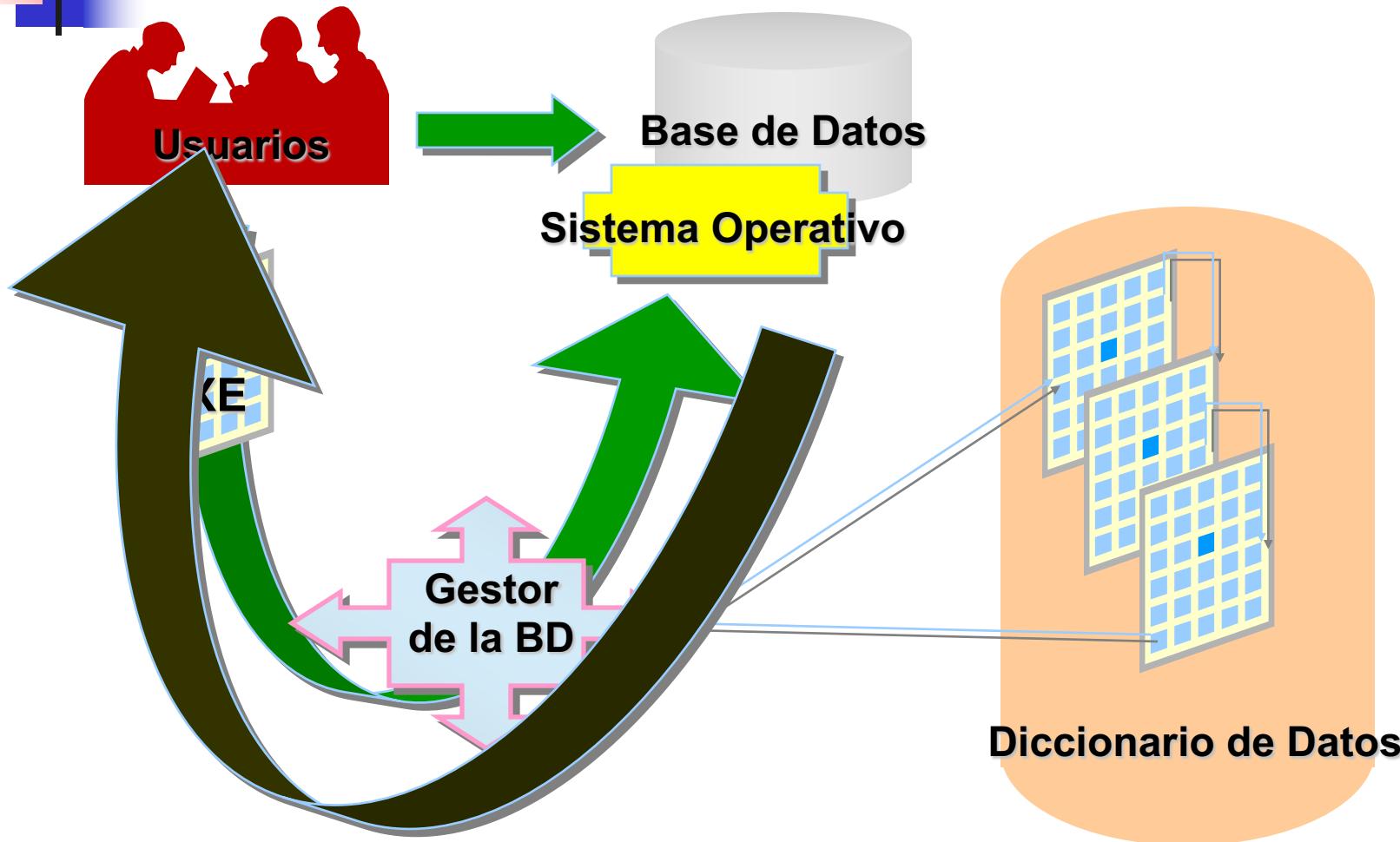


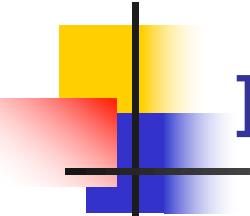
# Lenguajes de la Base de Datos

- Lenguajes de definición de datos (DDL).
  - Externo, Conceptual e Interno
- Lenguaje de manipulación de datos (DML).
- Lenguajes de Programación.



# Funcionamiento de una Base de Datos





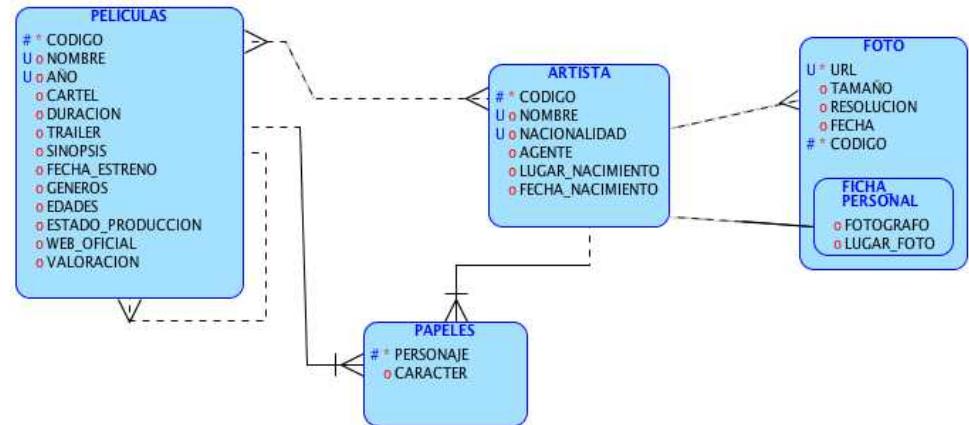
# Indice

---

- Definición de Base de Datos y conceptos principales
- El Modelo E/R
- El Modelo Relacional
- Paso del E/R al Relacional
- DML
- Normalización

# Modelo Entidad/Relación

- Diseño Lógico o Conceptual.
- Todos los datos y sus relaciones.
- Esquema global.

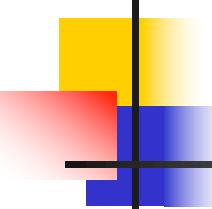


# Diseño Lógico: modelado relacional.

- Tablas.
- SQL.
- Reglas de transformación (algoritmo)

```
1  CREATE TABLE DIMENSION TIEMPO (
2      CLAVE TIEMPO      INTEGER PRIMARY KEY,
3      FECHA           DATE NOT NULL,
4      DIA_DE_SEMANA   VARCHAR(9) NOT NULL,
5      DIA_DE_MES       INTEGER NOT NULL,
6      DIA_TOTAL        INTEGER NOT NULL,
7      SEMANA_ANIO     INTEGER NOT NULL,
8      MES              INTEGER NOT NULL,
9      MES_TOTAL        INTEGER NOT NULL,
10     TRIMESTRE       INTEGER NOT NULL,
11     ESTACION         VARCHAR(50) NOT NULL
12 );
13
```





# Normalización

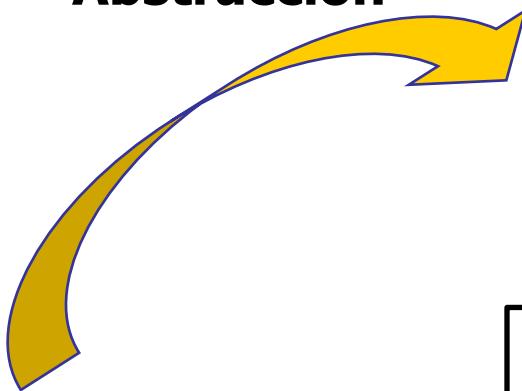
- Técnica eliminación anomalías.
- Disponer de BD sin redundancia y fáciles de mantener.



# Proceso de Diseño



**Abstracción**



**Modelo E/R**

**ENTIDADES  
ATRIBUTOS  
RELACIONES**



**Modelo Relacional**

`CREATE TABLE veterinarios`

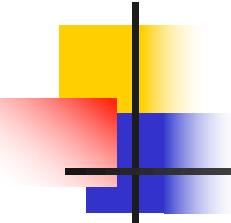
`CREATE TABLE vacunas`



**Base de Datos**

## “Seres distinguibles del mundo real”

- Un objeto de interés
- Real o abstracto
- Un nombre o sustantivo
- Algo sobre lo que la organización necesita información

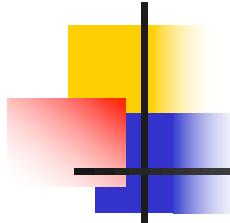


## Atributo

---

### “Propiedades de las entidades”

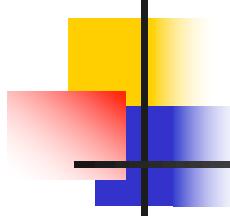
- Se usan para describir las entidades
- Especifican elementos de información:  
DATOS
- Una entidad posee muchos atributos



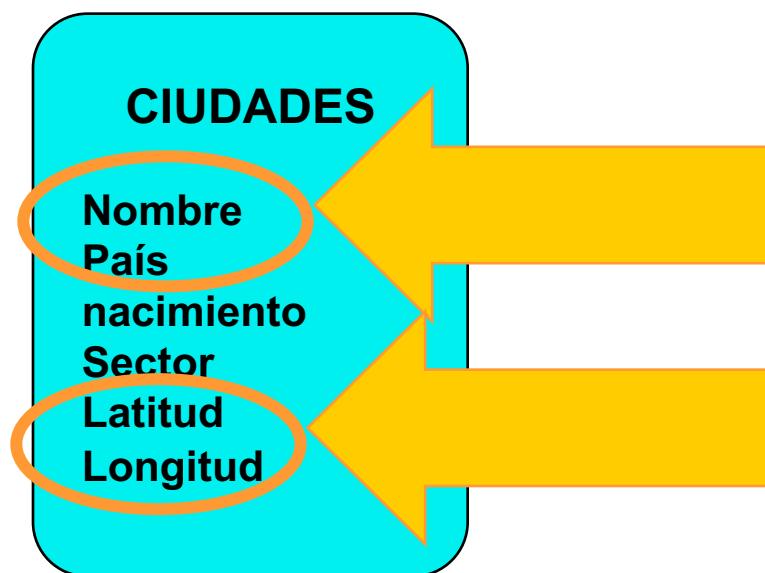
# Diagrama de Entidades

- Rectángulo 'suave'
- Un nombre único en mayúsculas
- Atributos en minúsculas

EMPLEADO  
nombre  
apellidos  
fecha nac.

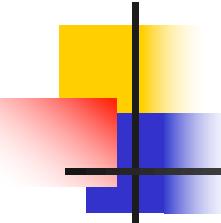


# Restricción de identificación: CLAVE



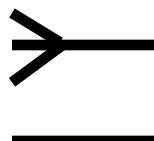
## “Conexiones entre las entidades”

- Esquema binario
- Doble dirección
- Nombradas
- Orden
- Obligatoriedad



# Diagrams

- Una línea que conecta dos entidades
- Nombres de la relación en minúsculas



- **Opcionalidad**

Obligatoria - *debe ser*

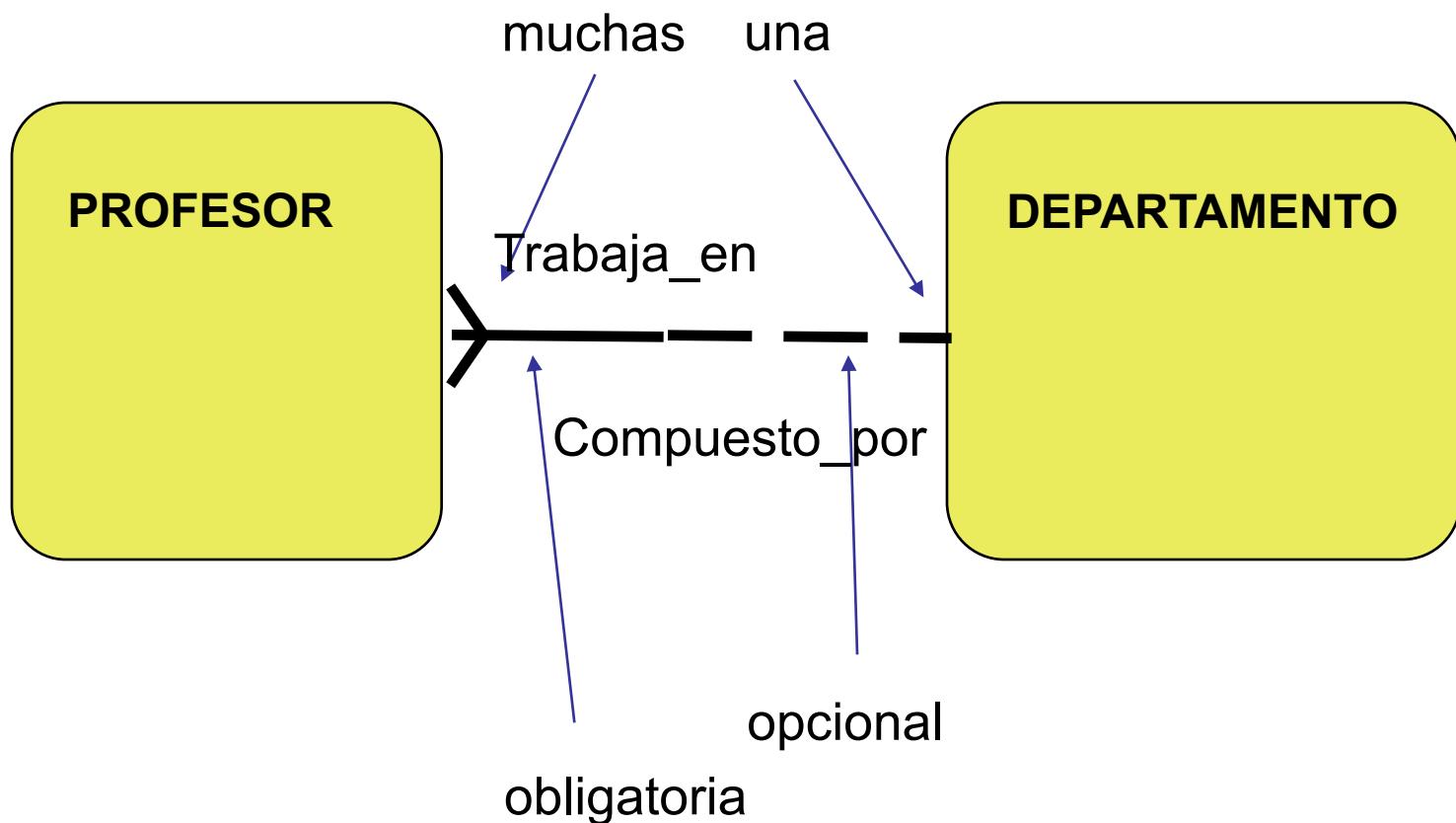
Opcional - *puede ser*

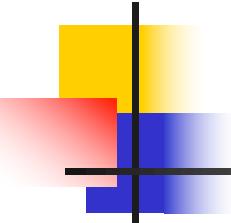
- **Grado**

Una o más

Una y sólo una

# Diagrams





# Semántica de las Relaciones

Cada

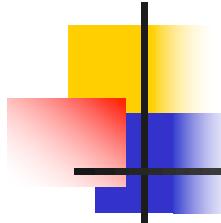
entidad 1

( Debe ser  
o  
puede ser )

Nombre de  
la relación

( Una o más  
o  
una y sólo una )

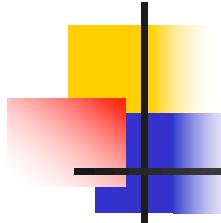
entidad 2



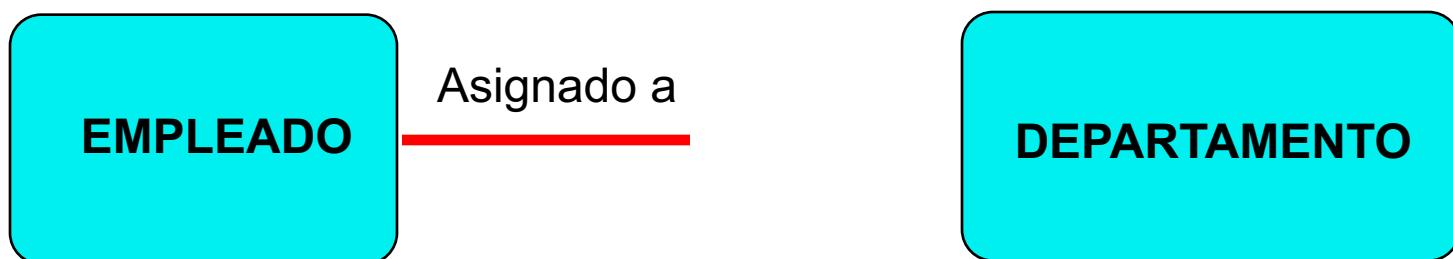
# Ejemplo

---

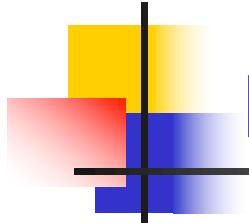




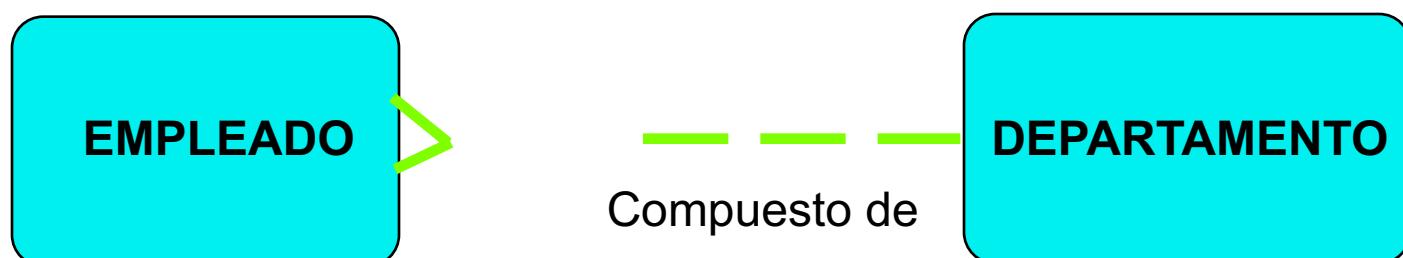
# Ejemplo



Cada **EMPLEADO** debe ser asignado a uno y sólo un **DEPARTAMENTO**



# Ejemplo

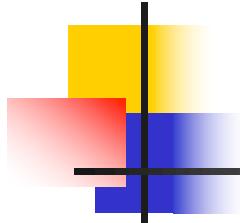


Cada DEPARTAMENTO puede estar compuesto de uno o más EMPLEADOS

# Ejemplo



Cada DEPARTAMENTO puede ser responsable de uno o más EMPLEADOS



# Tipos de Relación



Muchos a uno  
M:1

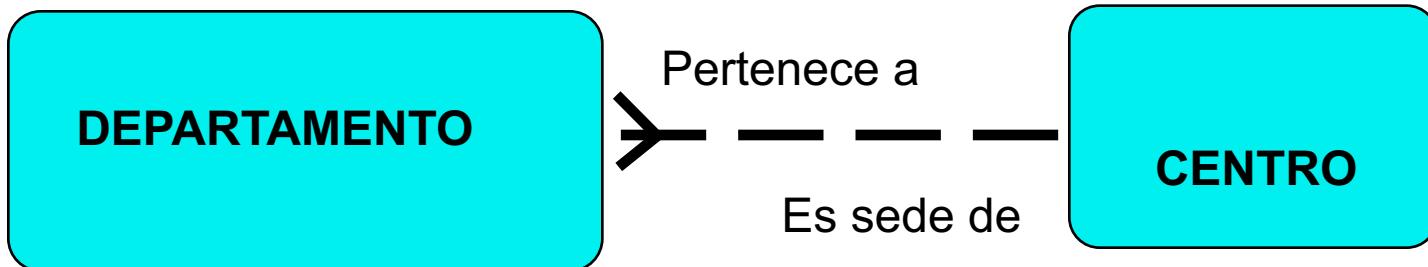
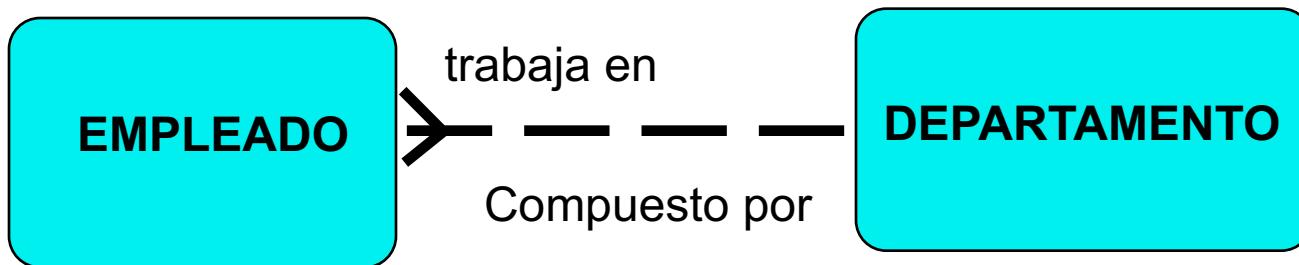


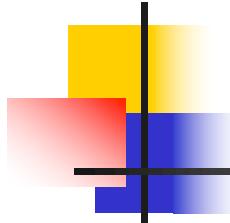
Muchos a muchos  
M:M



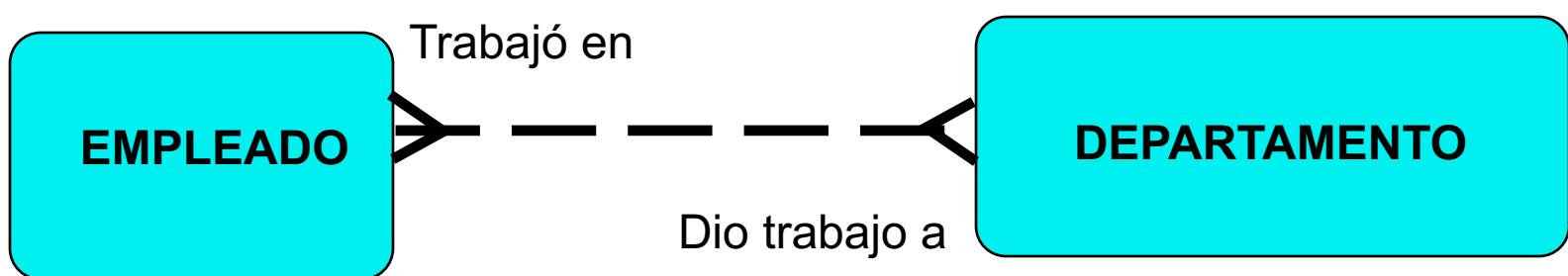
Uno a uno  
1:1

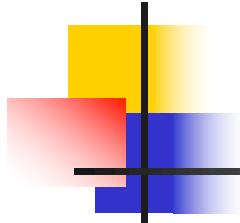
# Relaciones Muchos a Muchos



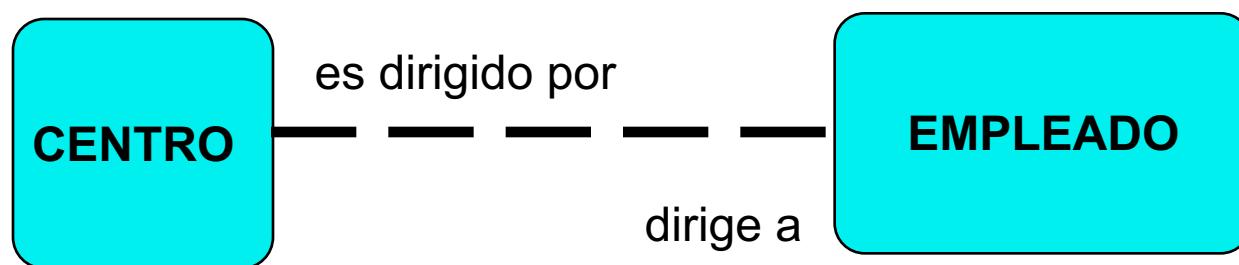


# Relaciones Muchos a Muchos

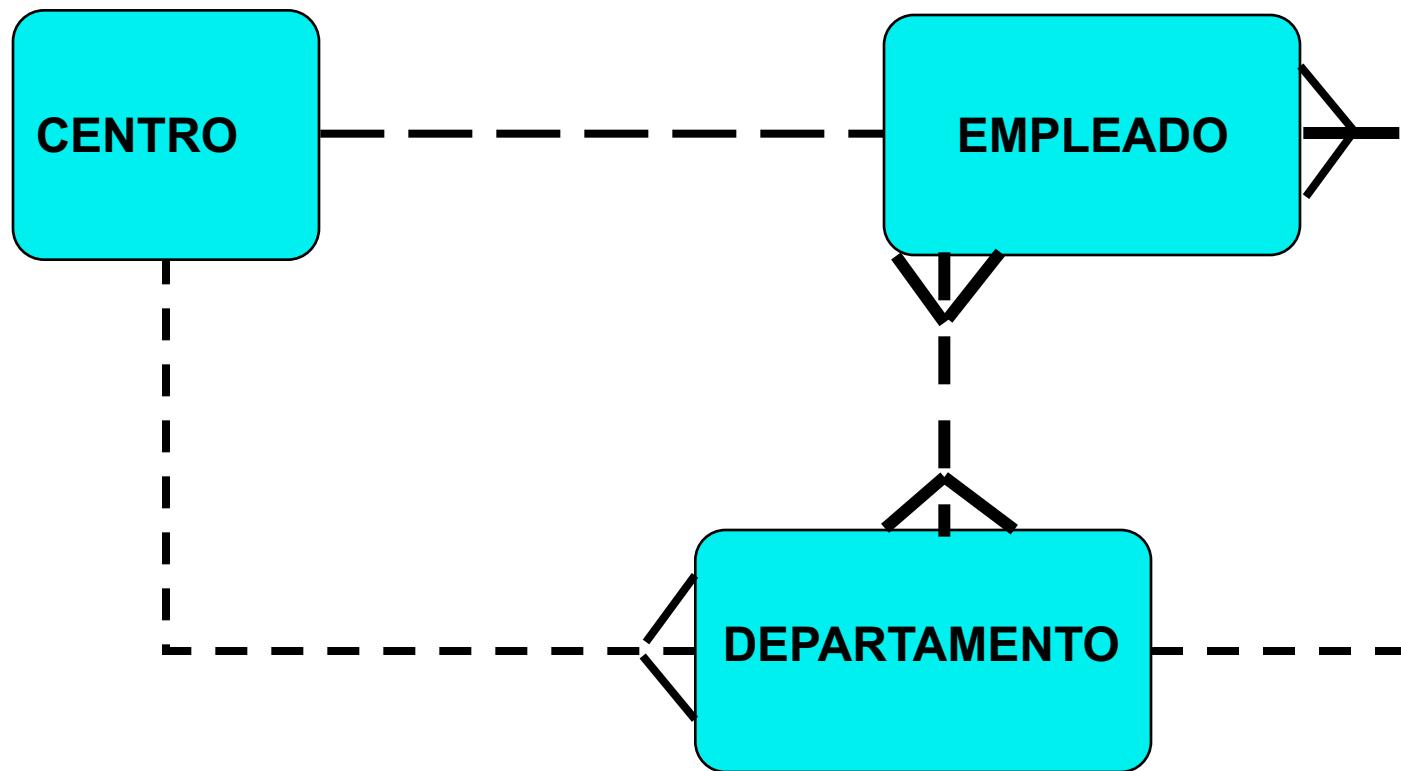


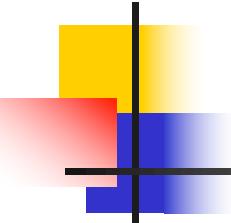


# Relaciones Uno a Uno



# Todas las Relaciones Juntas

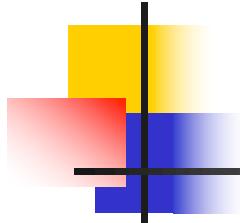




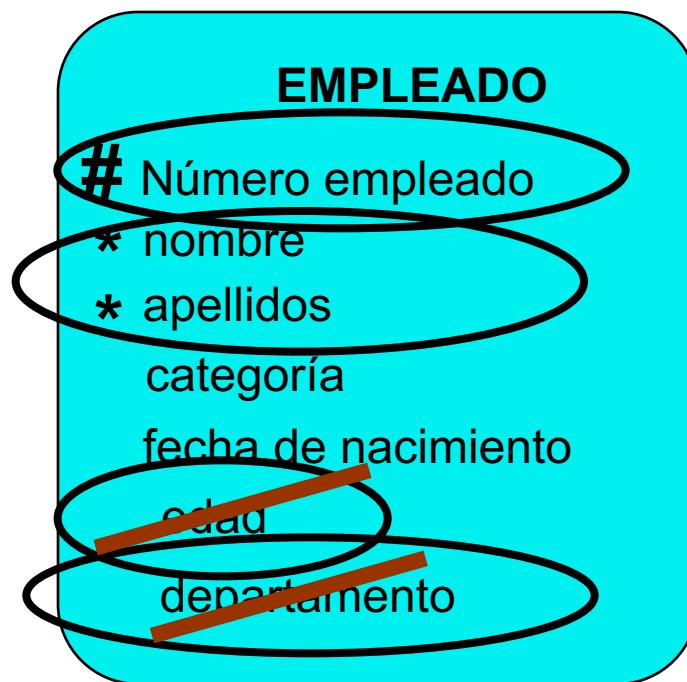
# Atributos

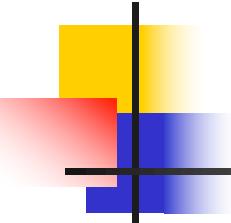
---

- Los atributos son los datos del mundo real
- No tienen estructura. Atomicidad.
- No son computables a partir de otros atributos
- Son mono-valuados
- Pueden ser requeridos
- **NUNCA REPRESENTA OTRA ENTIDAD !!!!**

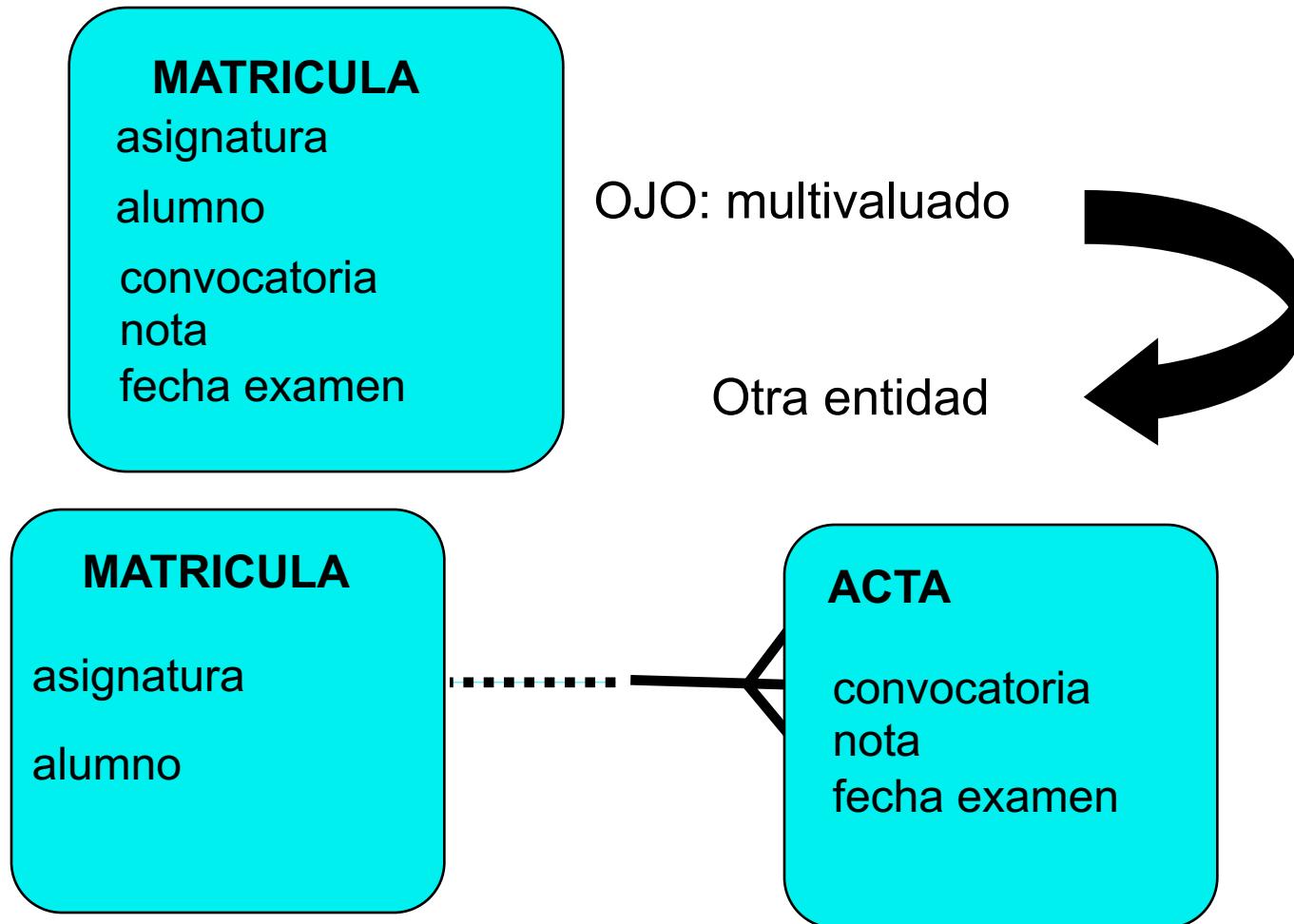


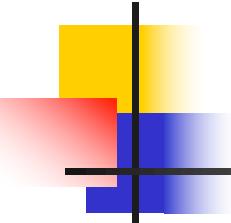
# Atributos





# Valores Simples



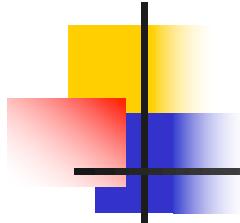


# Claves

Cada instancia de una entidad debe ser identificada de manera única



Regla de completitud: siempre puede determinarse un subconjunto de atributos que la identifica.



# Claves Simples y Compuestas

**PROFESOR**

# Número registro personal

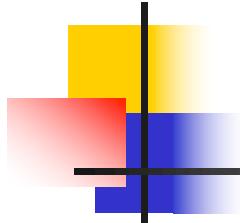
Atributo simple

**CIUDAD**

# latitud

# longitud

Atributo compuesto



# Claves Primarias y Candidatas

## CIUDAD

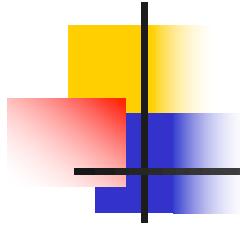
# latitud

# longitud

U\* nombre ciudad

U\* nombre país

U página web



# Claves Candidatas y Obligatoriedad

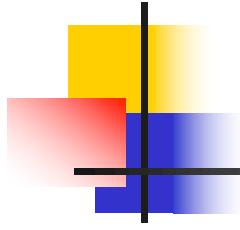
## CIUDAD

# latitud

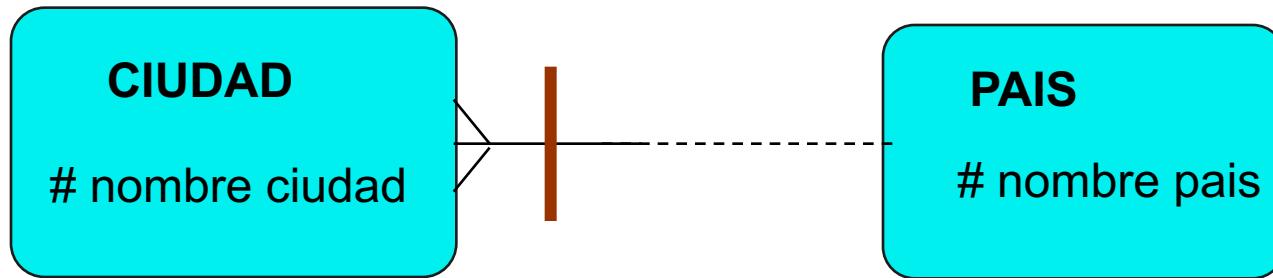
# longitud

U\* nombre ciudad

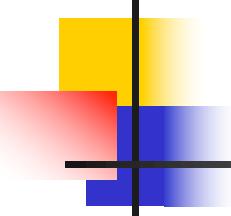
U\* nombre país



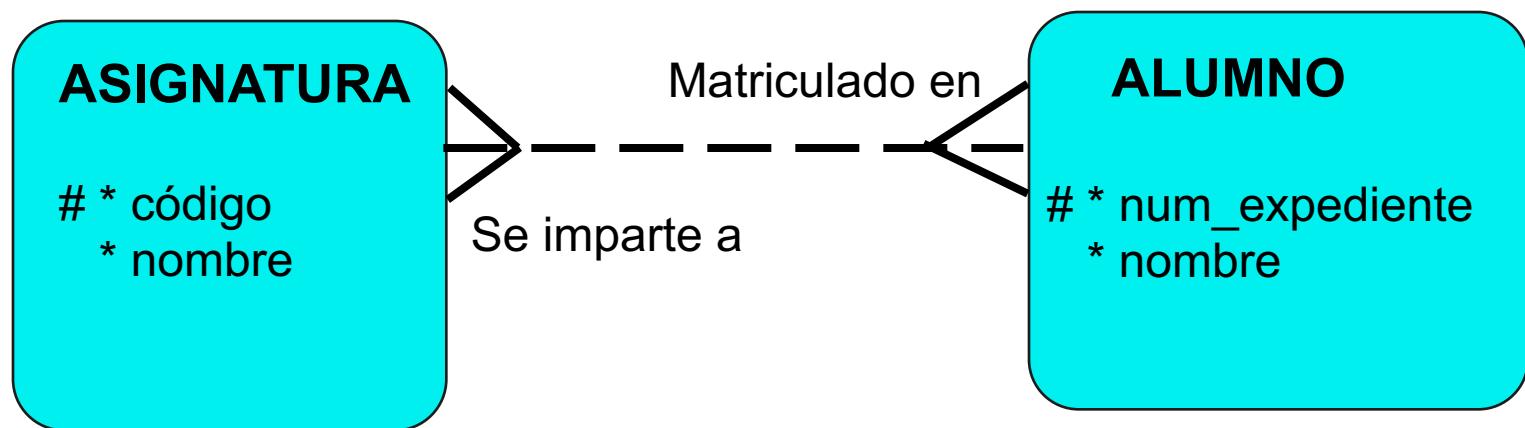
# Entidad débil. Clave prestada



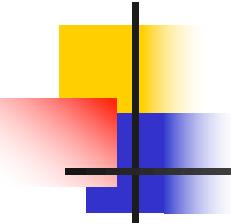
¿cómo se identifica la ciudad?



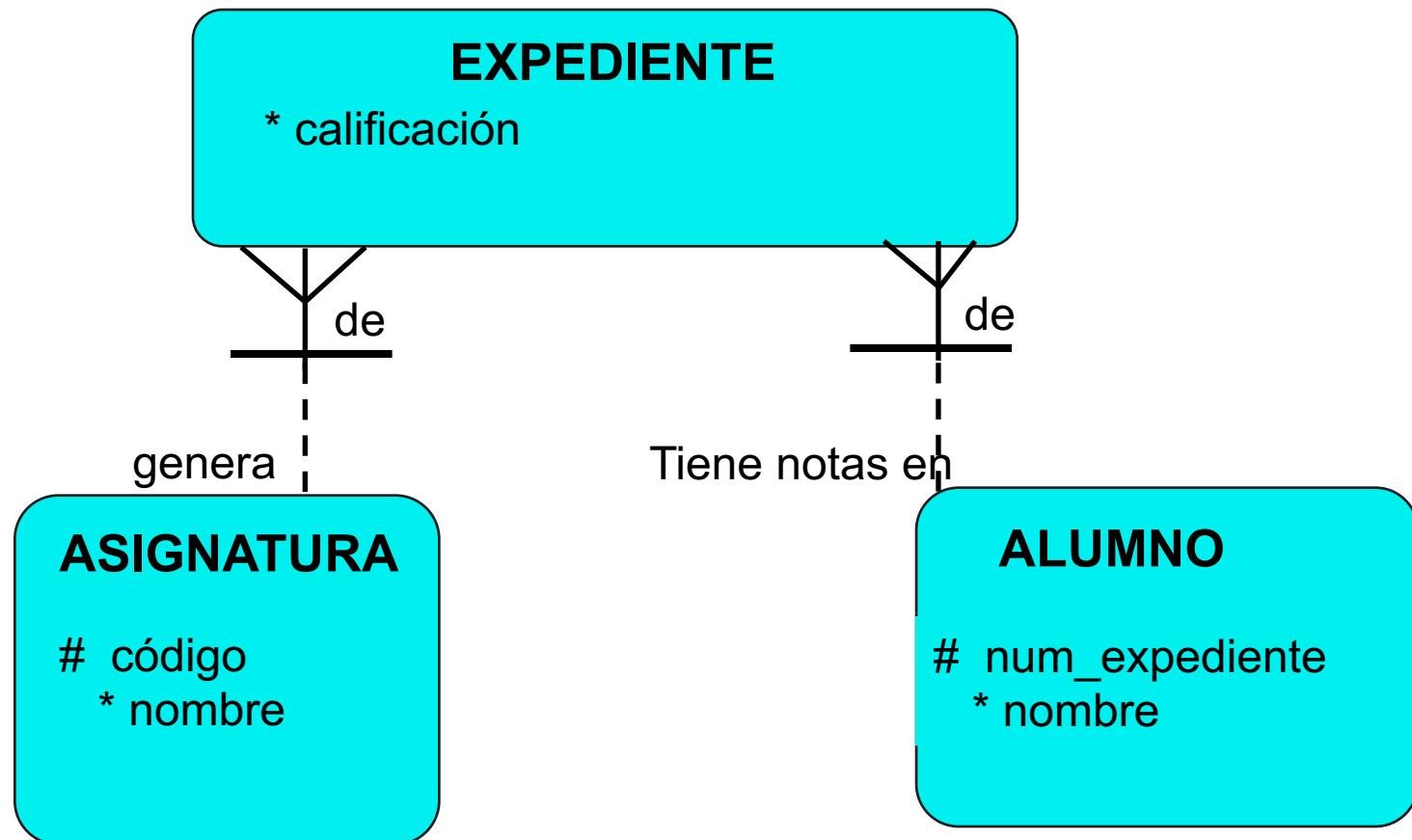
# Entidad débil y relación M:M

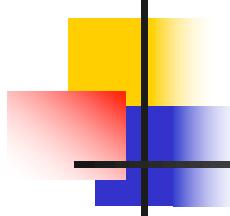


¿A quién pertenece el atributo  
**CALIFICACIÓN?**

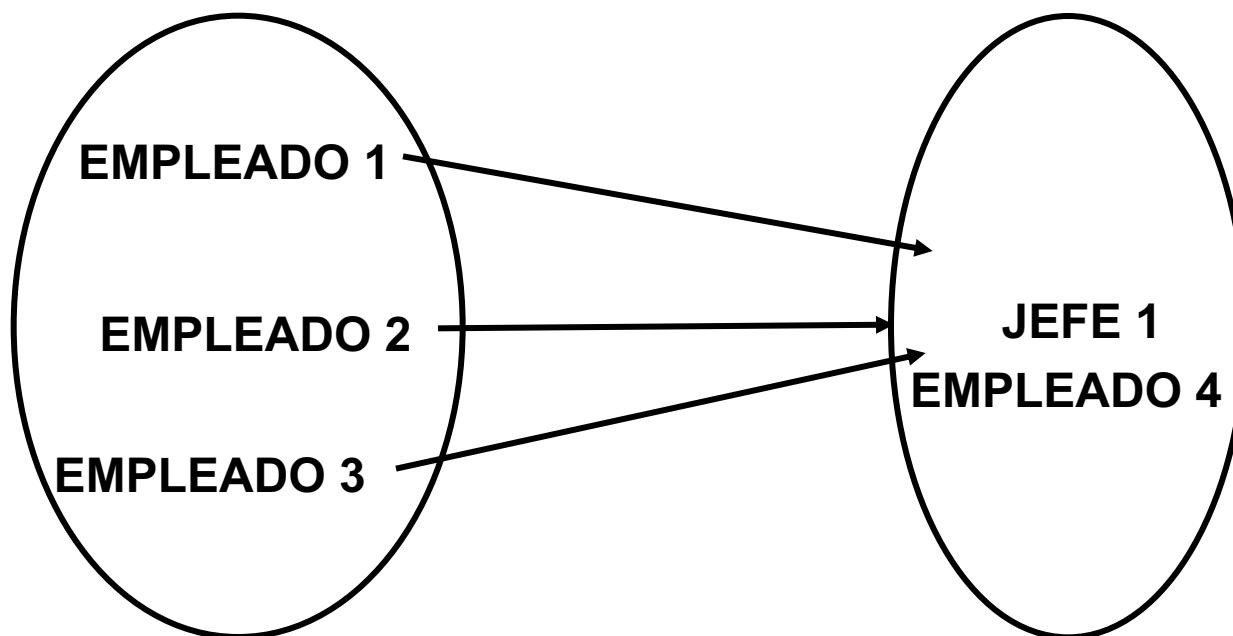


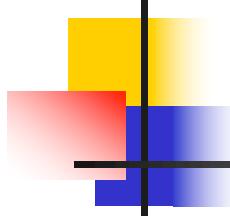
# Entidad débil y relación M:M



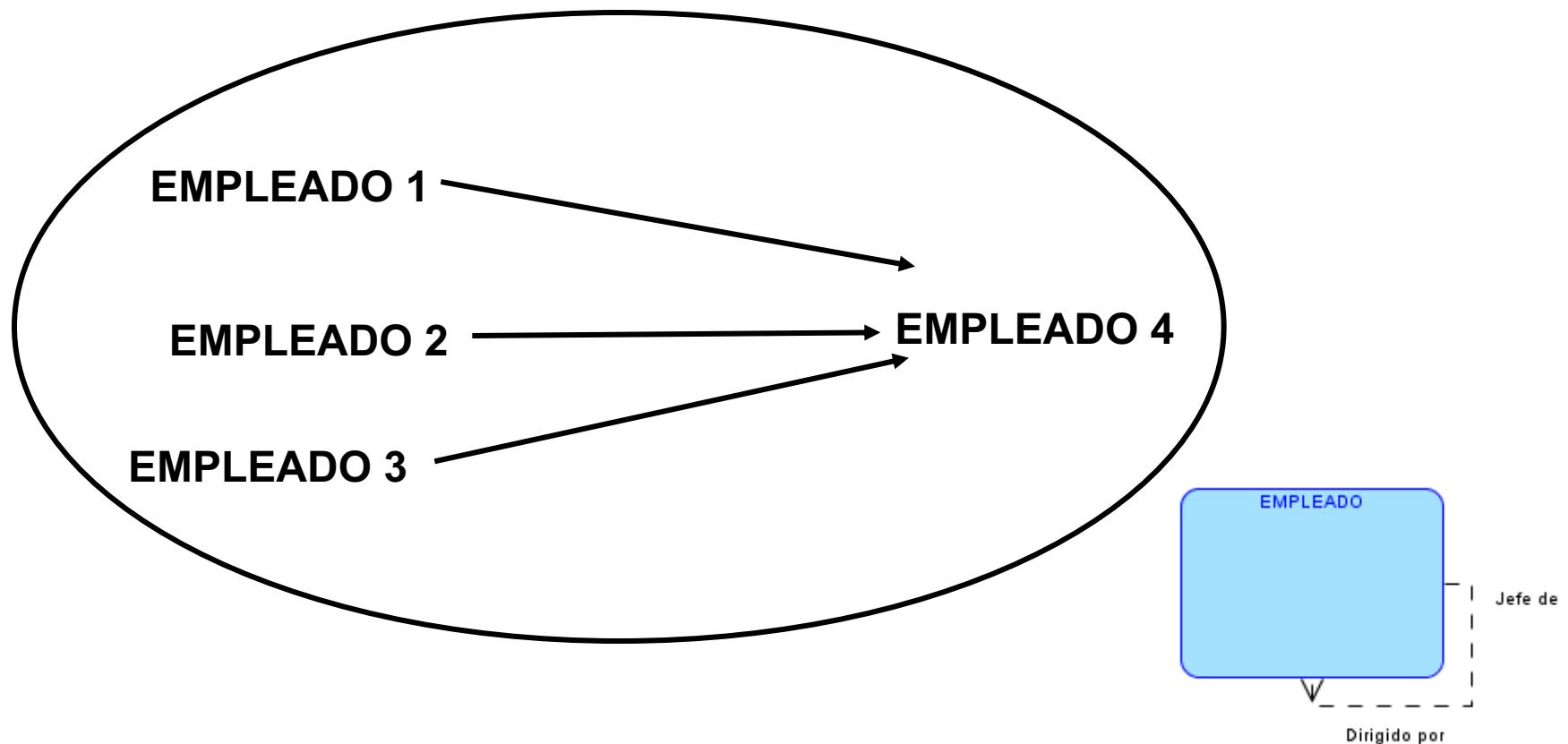


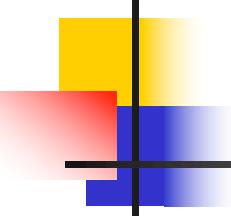
# Relaciones Reflexivas





# Relaciones Reflexivas

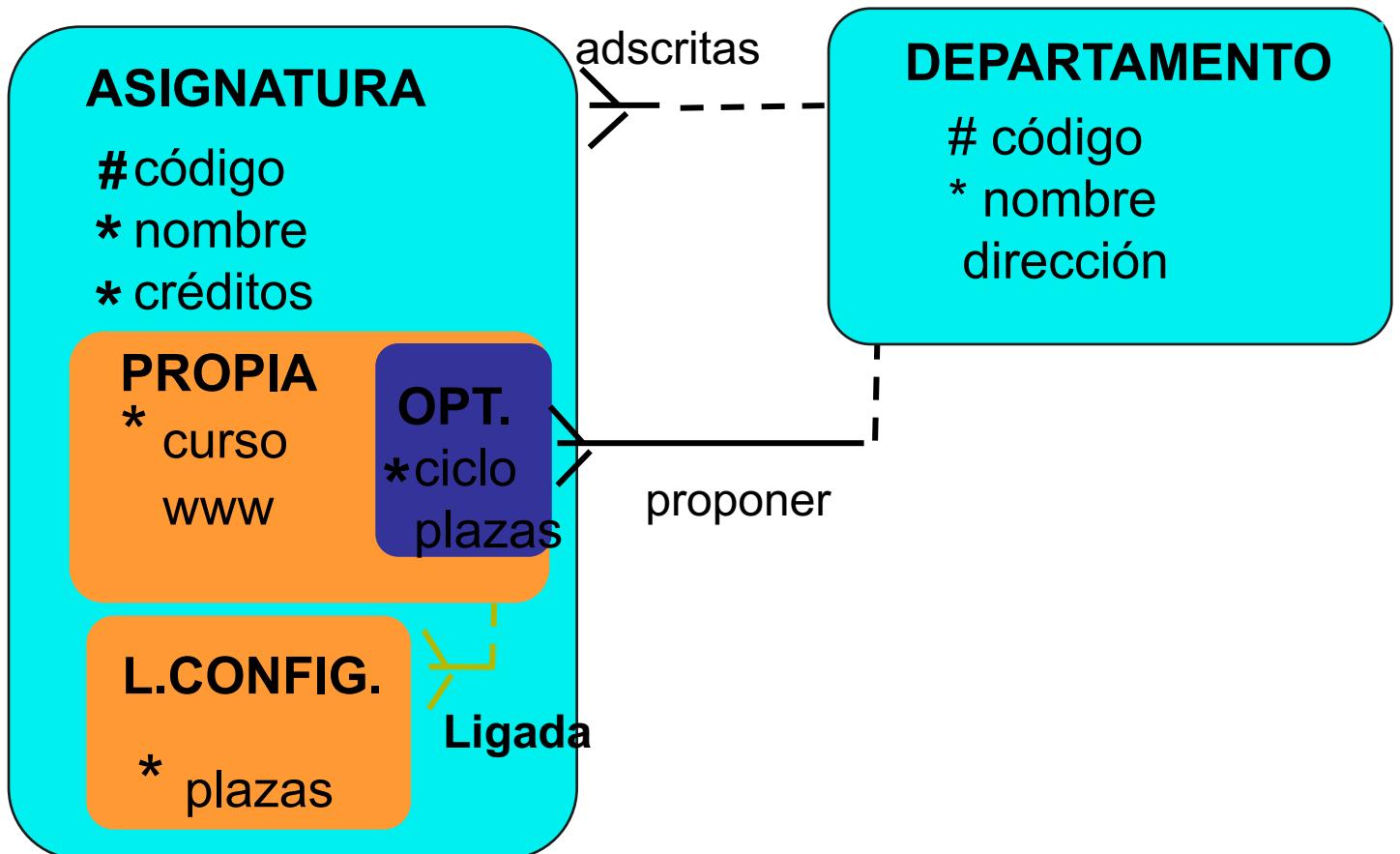


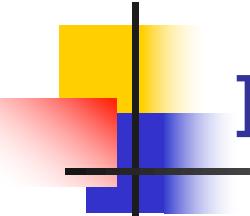


# Relación ES UN. Subentidad



# Relación ES UN. Subentidad

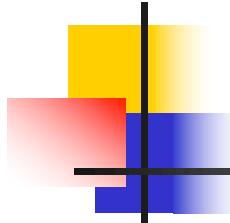




# Indice

---

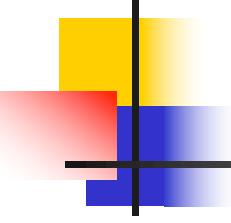
- Definición de Base de Datos y conceptos principales
- El Modelo E/R
- El Modelo Relacional
- Paso del E/R al Relacional
- DML
- Normalización



## Modelo Relacional

Una *relación* consta de:

- Un esquema: conjunto de pares (atributo,dominio).
- Un cuerpo: conjunto de tuplas de pares (atributo,valor).



# Ejemplo de relación:

Tabla: DEPARTAMENTOS

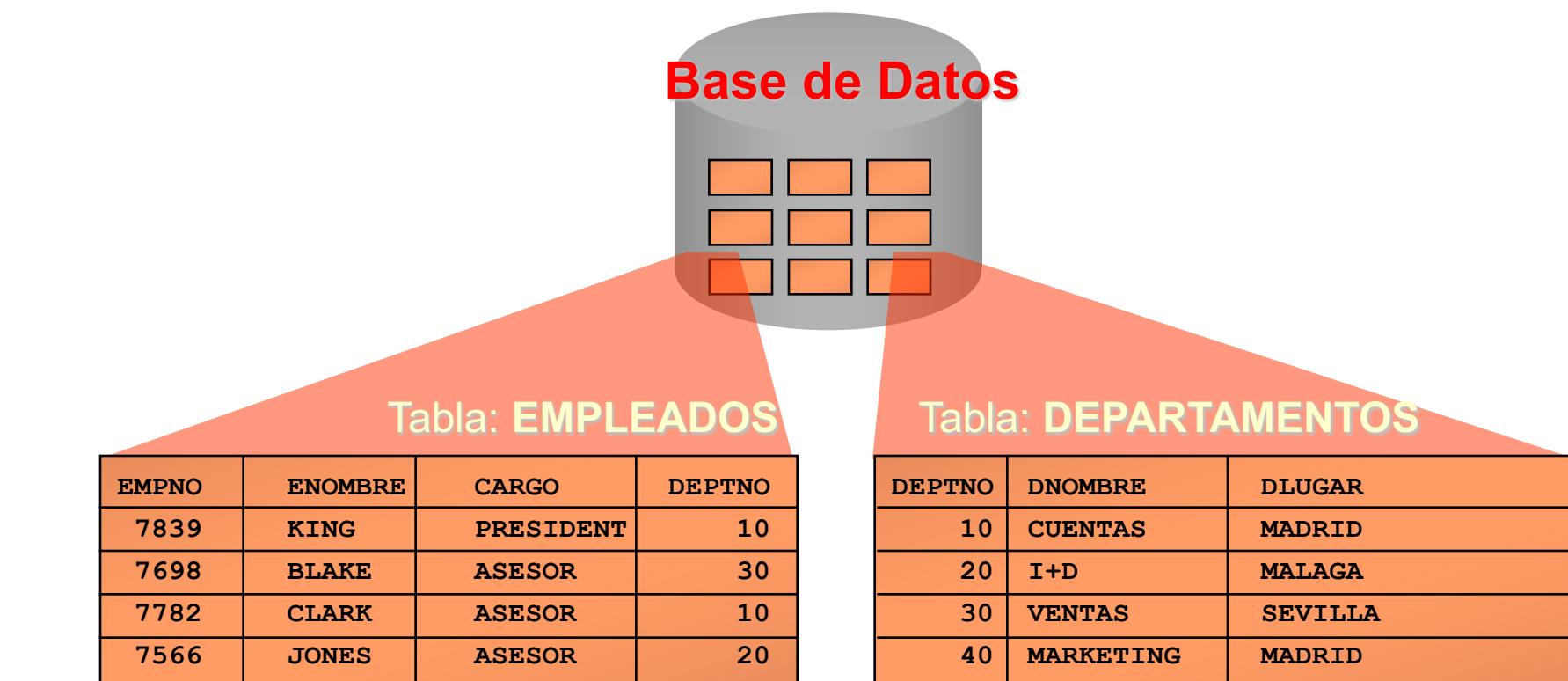
DEPTNO	DNOMBRE	DLUGAR
01	MANTENIMIENTO	MÁLAGA
10	CUENTAS	MADRID
20	I+D	MALAGA
30	VENTAS	SEVILLA
40	MARKETING	MADRID

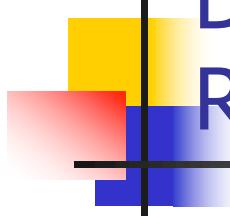
**ESQUEMA**

**CUERPO**

# Definición de una Base de Datos Relacional

Una base de datos relacional es un conjunto de relaciones (o tablas).





# Definición de una Base de Datos Relacional

---

- Los dominios se completan con el valor NULL.
- Claves candidatas.
  - Clave minimal.
  - Clave Primaria. Criterios de selección.
- Clave foránea.

## **MODELO ORIENTADO A VALORES**

# Relaciones. Funcionamiento.

- Cada fila de una tabla se identifica mediante la Clave Primaria (PK).
- Se pueden relacionar datos de varias tablas mediante las Claves Foráneas (FK).

Tabla: **EMPLEADOS**

NÚMERO	NOMBRE	CARGO	DEPTNO
7839	KING	PRESIDENT	10
7698	BLAKE	ASRSOR	30
7782	CLARK	ASESOR	10
7566	JONES	ASESOR	20



Clave Primaria

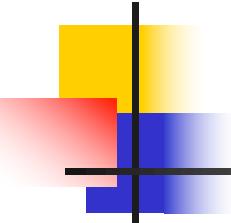
Tabla: **DEPARTAMENTOS**

DEPTNO	NOMBRE	LUGAR
10	CUENTAS	MADRID
20	I + D	MALAGA
30	VENTAS	SEVILLA
40	MARKETING	MADRID



Clave Foránea

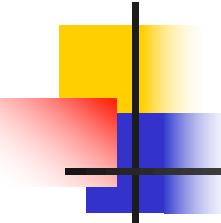
Clave Primaria



# Reglas del Modelo Relacional

## Propiedad de la clave primaria: IDENTIFICAR

Primera Regla de Integridad (de la *Entidad*): Las componentes de una clave primaria no pueden ser nulos.



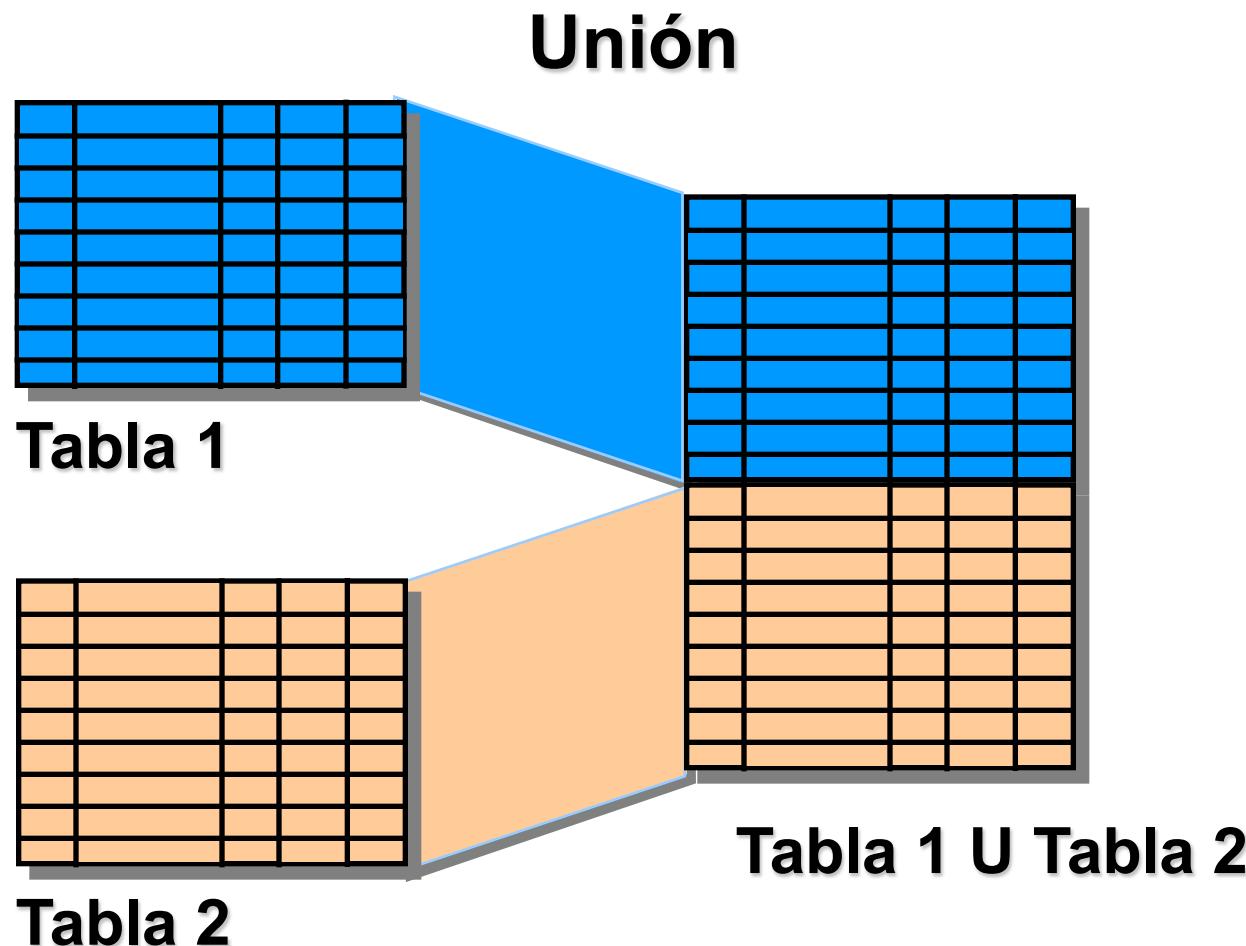
# Reglas del Modelo Relacional

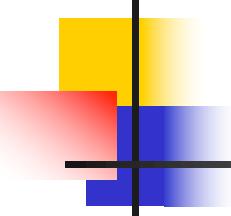
## Clave Foránea: relaciones entre tablas

**Segunda Regla de Integridad (de *Referencia*):**  
**Las componentes de una clave foránea son nulas o son iguales que el valor de alguna una primaria en una tabla del modelo.**

# Operaciones relacionales.

# Operaciones de Conjuntos





# Operaciones relacionales.

# Operaciones de Conjuntos

## **Unión Compatibles:**

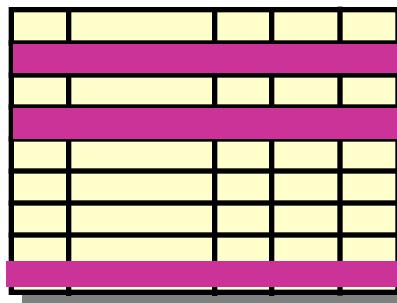
- **Unión.**
- **Intersección.**
- **Diferencia.**

## **Libres:**

- **Producto Cartesiano.**

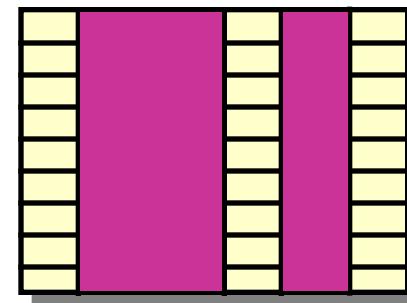
# Operaciones relacionales

# Selección



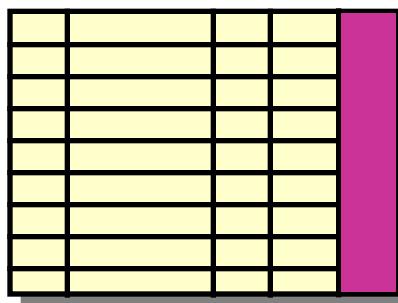
**Tabla 1**

# Proyección

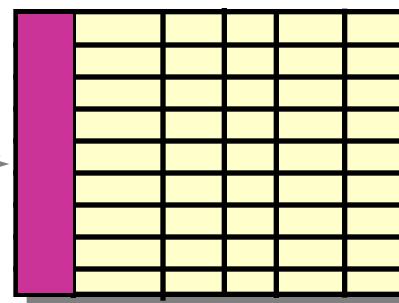


## Tabla 1

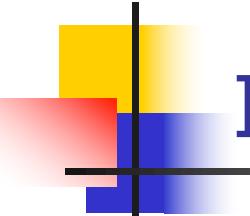
## Reunión (join)



**Tabla 1**



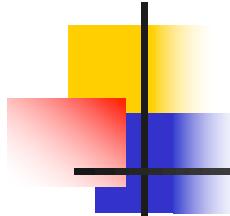
**Tabla 2**



# Indice

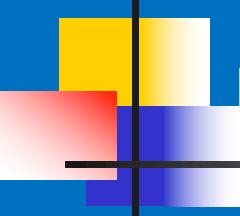
---

- Definición de Base de Datos y conceptos principales
- El Modelo E/R
- El Modelo Relacional
- **Paso del E/R al Relacional**
- DML
- Normalización



# Paso del Modelo E/R al Relacional

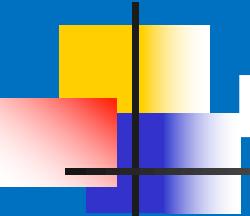
- Equivalencia entre el Modelo E/R y el Modelo Relacional
- Generación de un modelo conceptual de bases de datos relacionales a partir del modelo E/R



# Ejercicio I

---

- Se va a modelar un portal de juegos online. Así, contaremos con **empresas** que se encargarán de desarrollar **juegos**. La empresa tendrá un código único que servirá como identificador, un nombre (también único), pertenecerá a un país y dispondrá de una página web. Cada empresa podrá desarrollar un conjunto de juegos, donde cada juego contendrá un nombre (único), la fecha de lanzamiento y el precio. Un juego sólo puede ser desarrollado por una única empresa. Un juego a su vez puede tener una o varias secuelas del mismo (con los mismos campos que el juego), con lo que será necesario almacenar esta relación.
- Los **usuarios** que quieran jugar mediante este portal de juegos online deberán proporcionar un *nick* (valor único), un nombre y una fecha de nacimiento. También se le asociará un código numérico que servirá para su identificación. Un usuario podrá registrarse en varios juegos y un juego tendrá varios usuarios registrados.



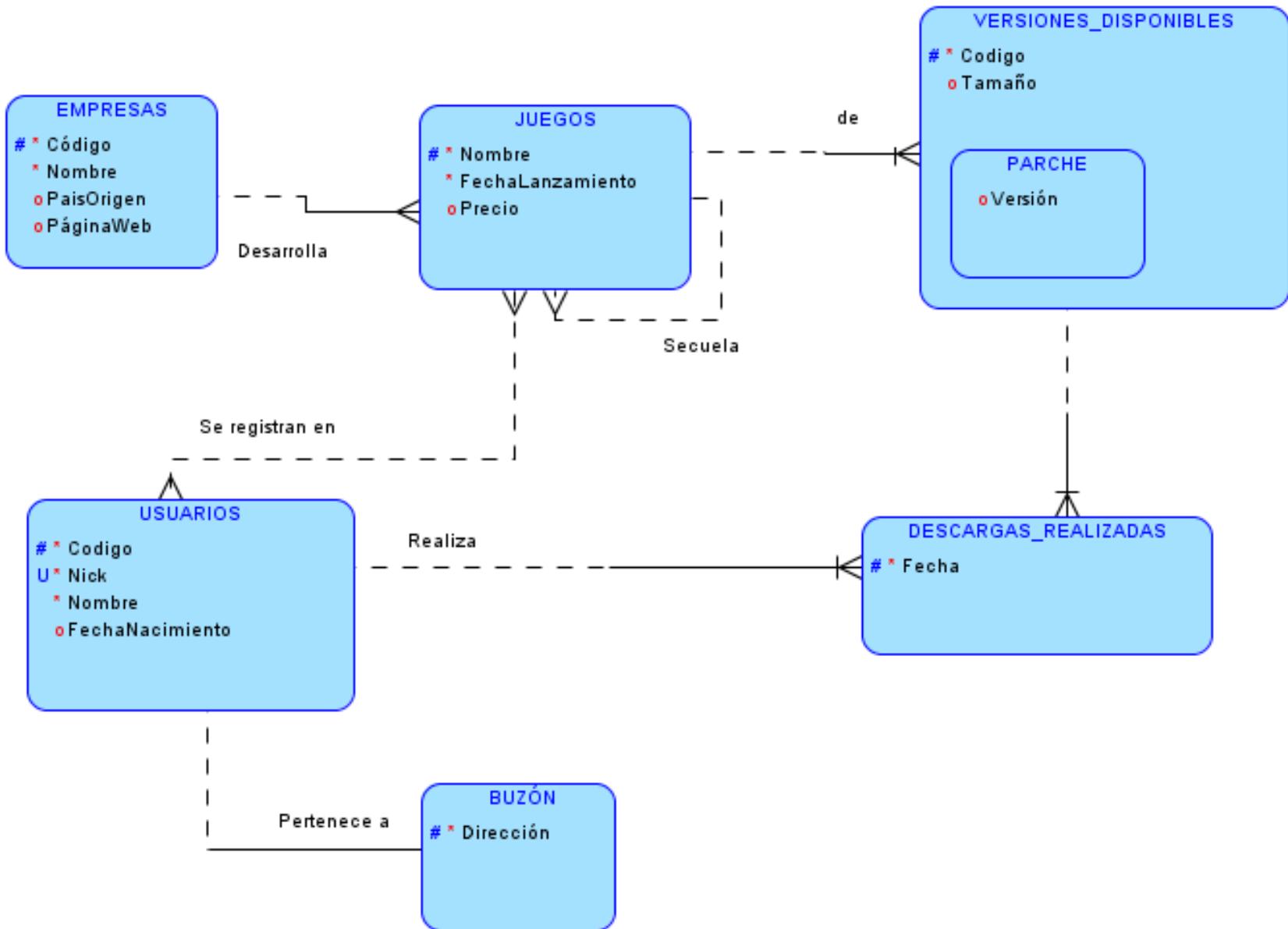
## Ejercicio II

---

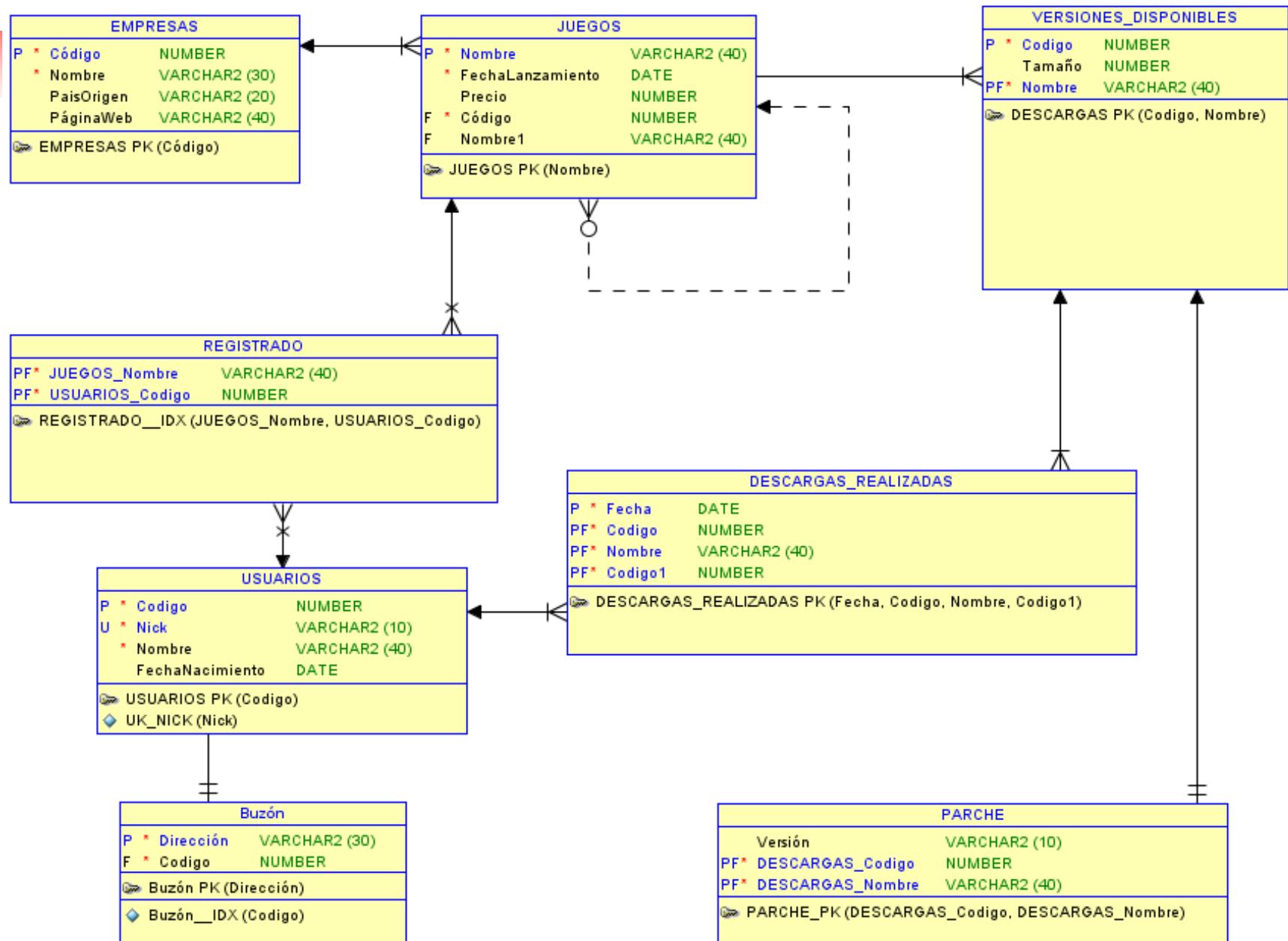
- Además, tendremos una entidad **buzón** que estará asociada estrictamente a un usuario. El usuario podrá definir esta dirección o dejarla vacía.
- Un juego podrá tener una o varias **versiones disponibles** del juego que se identificarán por el código y el tamaño de la versión. Además, cada versión disponible podría tener asociado un **parche**, que no es más que una actualización de dicha versión para corregir errores encontrados tras su publicación. El parche, además de la información de la versión disponible tendrá un atributo versión que lo distinguirá.
- Por último, los usuarios podrán realizar descargas de las versiones disponibles de cada juego. Por cada **descarga realizada** se almacenará la fecha. Es importante indicar que un usuario podrá realizar muchas descargas y la versión de un juego podrá ser descargada por varios usuarios.

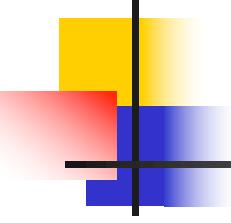
Modela el sistema anterior mediante un diagrama E/R

# EJEMPLO: Diagrama E/R Portal de Juegos Online



# Diagrama Relacional Resultante

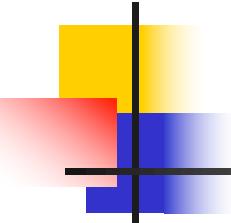




# Entidad Fuerte

---

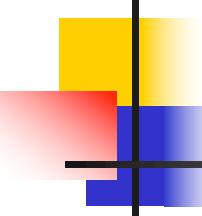
- Por cada Entidad Fuerte o principal se crea una Tabla
- Se crean las claves:
  - Primarias: Primary Key
  - Candidatas: Unique Key (opcionalmente NOT NULL)
- Se crean los atributos. Los que sean obligatorios serán NOT NULL



# Entidad Fuerte

```
CREATE TABLE EMPRESAS #  
  ( Código NUMBER PRIMARY KEY,  
    Nombre VARCHAR2 (30) NOT NULL, *  
    PaísOrigen VARCHAR2 (20) ,  
    PáginaWeb VARCHAR2 (40)  
 );
```

Código	Nombre	PaísOrigen	PáginaWeb
1	Electronic Arts	EEUU	www.ea.com
2	Infinity Ward	EEUU	www.infinityward.com

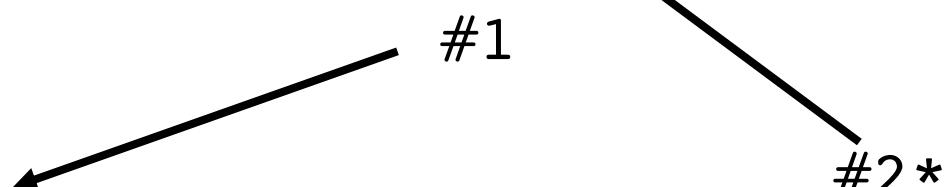


# Entidad Fuerte

```
CREATE TABLE USUARIOS
```

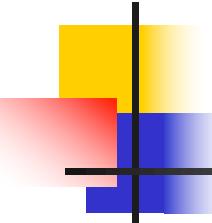
```
( Código NUMBER NOT NULL ,  
  Nick VARCHAR2 (10) NOT NULL ,  
  Nombre VARCHAR2 (40) NOT NULL ,  
  FechaNacimiento DATE  
);
```

```
ALTER TABLE USUARIOS
```



```
ALTER TABLE USUARIOS
```

```
ADD CONSTRAINT UK_NICK UNIQUE ( Nick );
```



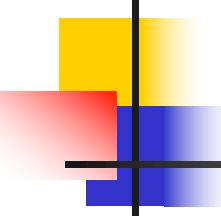
# Entidad Fuerte

```
CREATE TABLE JUEGOS  
  ( Nombre VARCHAR2 (40) NOT NULL ,  
    FechaLanzamiento DATE NOT NULL ,  
    Precio NUMBER ,  
    Código NUMBER NOT NULL ,  
    Nombre1 VARCHAR2 (40)  
  );
```

#1

```
ALTER TABLE JUEGOS  
  ADD CONSTRAINT "JUEGOS PK" PRIMARY KEY ( Nombre );
```





## Relación 1:M

---

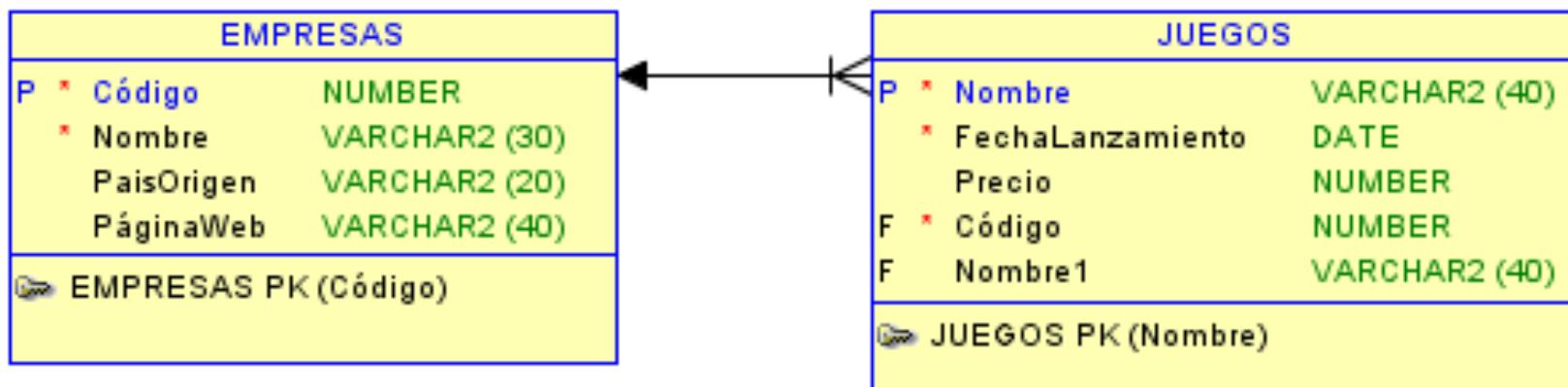
- Las Relaciones de Uno a Muchos entre 2 entidades  $A$  y  $B$  se implementan en la tabla generada para la entidad  $B$ , creando un atributo nuevo que referencia a la clave de la otra Entidad  $A$  (FOREIGN KEY)
- Si es obligatoria en el lado  $B$ , el campo creado debe ser NOT NULL para obligar a que exista la referencia
- Si es obligatoria en el lado de  $A$  se puede hacer mediante programación de bases de datos (Triggers, Procedimientos Almacenados, etc.)
- Las relaciones obligatorias por el lado  $A$  no son frecuentes

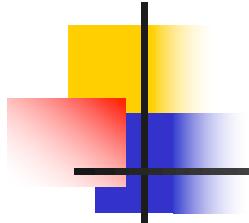
# Relación 1:M

ALTER TABLE JUEGOS

ADD CONSTRAINT Desarrolla FOREIGN KEY

( Código ) REFERENCES EMPRESAS ( Código ) ;





## Relación 1:M Reflexiva

- La relación reflexiva de una Entidad  $A$  consigo misma se implementa añadiendo un campo del mismo tipo que la clave de  $A$ , con una referencia a la propia tabla generada para  $A$

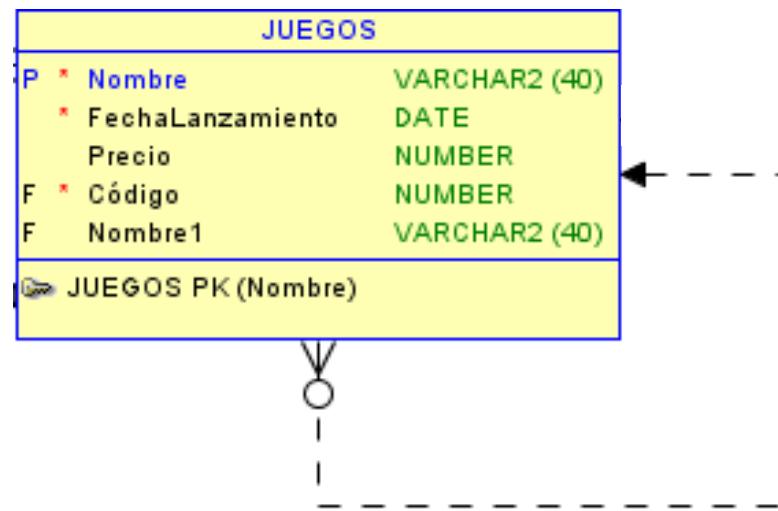
# Relación 1:M Reflexiva

ALTER TABLE JUEGOS

ADD CONSTRAINT Secuela FOREIGN KEY

( Nombre1 ) REFERENCES JUEGOS ( Nombre )

ON DELETE SET NULL ;



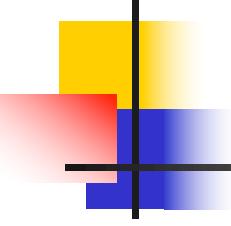
# EJEMPLO DE DATOS EN LAS TABLAS

## JUEGOS

Nombre	FechaLanzamiento	Precio	Codigo	Nombre1
FIFA 2015	20/09/14	50	1	
FIFA 2016	29/09/15	69	1	FIFA 2015
CALL OF DUTY	29/10/03		2	

## EMPRESAS

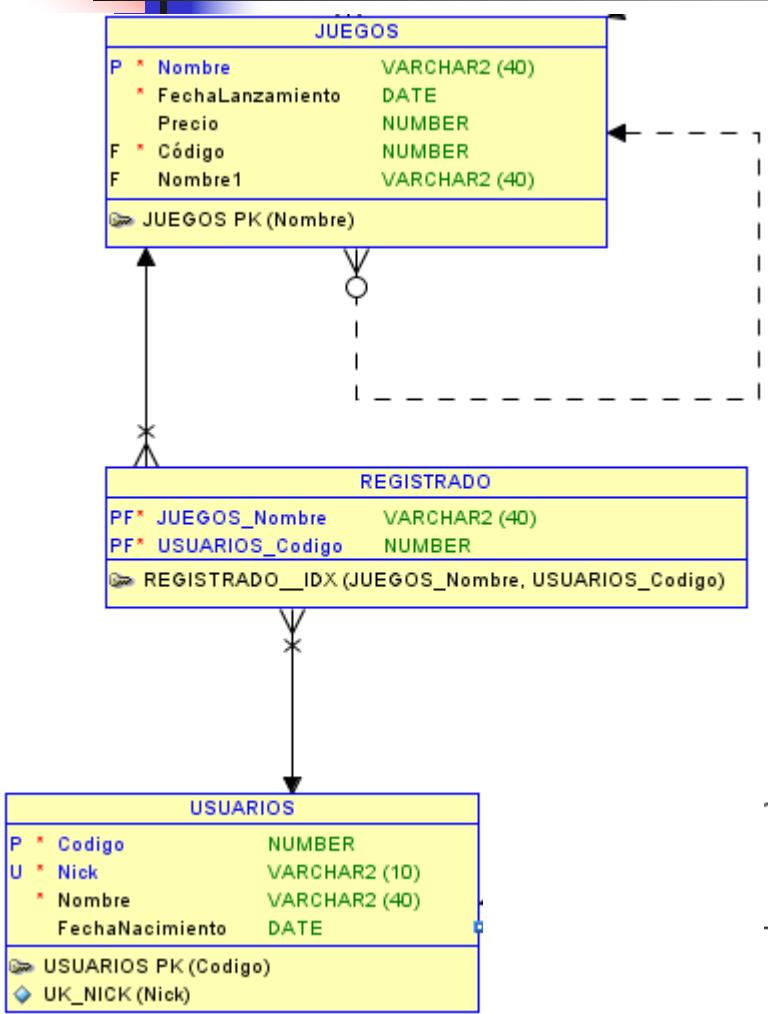
Código	Nombre	PaisOrigen	PaginaWeb
1	Electronic Arts	EEUU	www.ea.com
2	Infinity Ward	EEUU	www.infinityward.com



# Relación M:M

- Una Relación R de muchos a muchos (M:M) entre 2 entidades  $A$  y  $B$  se implementa creando una tabla nueva  $T_R$ .
- $T_R$  tendrá una referencia a cada una de las tablas generadas para las entidades  $A$  y  $B$ .
- La clave primaria de  $T_R$  está formada por los atributos que referencian a las tablas A y B.

# Relación M:M



CREATE TABLE REGISTRADO (

JUEGOS\_Nombre REFERENCES JUEGOS,

USUARIOS\_Codigo REFERENCES

USUARIOS (Codigo),

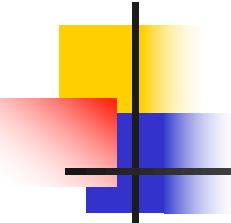
CONSTRAINT REGISTRADO\_IDK PRIMARY KEY ( JUEGOS\_Nombre,

USUARIOS\_Codigo));

relaciona

USUARIOS tiene 2 índices

Clave compuesta



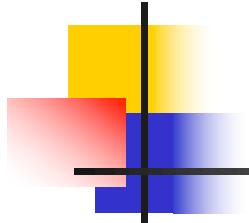
## Relación M:M

### REGISTRADO

JUEGOS_NOMBRE	USUARIOS_CODIGO
FIFA 2015	1
CALL OF DUTY	1
CALL OF DUTY	2

El Usuario 1 está Registrado en los juegos FIFA 2015 y  
CALL OF DUTY

El Usuario 2 sólo en CALL OF DUTY



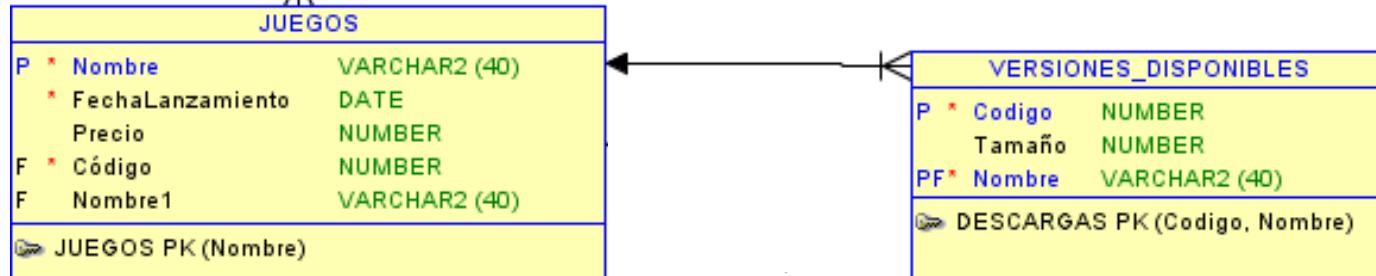
## Entidad débil

---

Una entidad débil  $D$  de una fuerte  $F$  se implementa incluyendo una referencia en  $D$  hacia  $F$

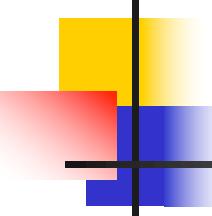
La clave Primaria de  $D$  será la suya propia junto con la clave de la entidad fuerte  $F$

# Entidad débil



CREATE TABLE VERSIONES\_DISPONIBLES

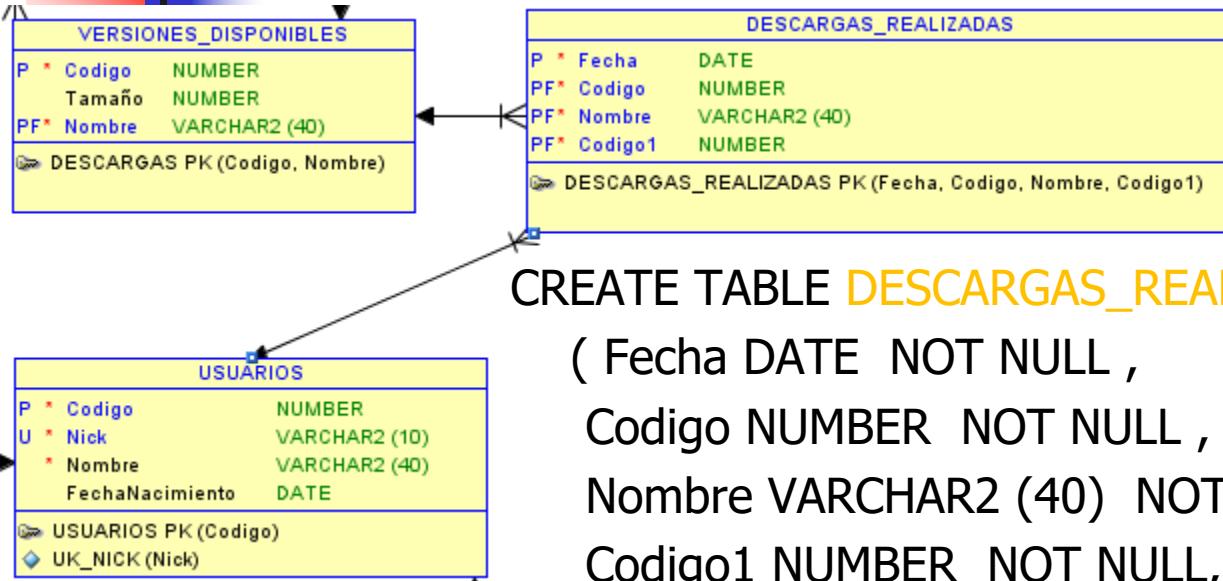
```
(  
    Código NUMBER NOT NULL ,  
    Tamaño NUMBER ,  
    Nombre VARCHAR2 (40) NOT NULL REFERENCES JUEGOS ( Nombre ) ,  
    CONSTRAINT "DESCARGAS PK" PRIMARY KEY ( Código, Nombre )  
) ;
```



## Entidad débil (m:m)

- Las Relaciones M:M entre 2 relaciones  $A$  y  $B$  que contiene atributos generan una tabla con referencia a las claves primarias de  $A$  y  $B$ .
- La clave primaria de la tabla es la suya propia junto con las claves de  $A$  y  $B$

# Entidad débil (m:m)



Clave prestada

CREATE TABLE DESCARGAS\_REALIZADAS

( Fecha DATE NOT NULL ,

Código NUMBER NOT NULL ,

Nombre VARCHAR2 (40) NOT NULL ,

Código1 NUMBER NOT NULL ,

CONSTRAINT "DESCARGAS\_REALIZADAS PK" PRIMARY KEY

( Fecha, Código, Nombre, Código1 ) ,

CONSTRAINT Juego FOREIGN KEY (Código, Nombre)

REFERENCES VERSIONES\_DISPONIBLES

( Código, Nombre ) ,

CONSTRAINT "Se descarga" FOREIGN KEY

( Código1 ) REFERENCES USUARIOS ( Código ) );

# Entidad débil (m:m)

CODIGO	TAMAÑO	NOMBRE
1	100000000	FIFA 2015
2	50000000	FIFA 2015
3	60000000	FIFA 2015
1	200000000	FIFA 2016

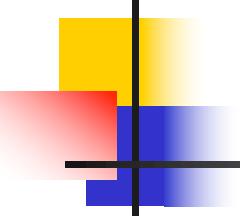
**VERSIONES DISPONIBLES**

**USUARIOS**

CODIGO	NICK	NOMBRE	FechaNacimiento
1	friki	Victor Mento	04/10/91
2	frika	Lola Mento	15/02/93

**DESCARGAS**

FECHA	CODIGO	NOMBRE	CODIGO1
03/05/11	1	FIFA 2015	1
04/05/11	3	FIFA 2015	2



# Relación 1:1

---

- Una Relación 1 a 1 entre 2 Entidades  $A$  y  $B$  se implementa según la obligatoriedad:
- Si es obligatoria en  $A$  y no en  $B$  se implementa en  $A$  agregando un atributo que referencia a la clave primaria de  $B$ . Ese atributo debe tener la restricción UNIQUE para impedir que varias filas de  $A$  referencia a la misma de  $B$ . Además, el atributo debe ser NOT NULL. Si la clave de  $B$  es compuesta, se añadirán tantos atributos como atributos tenga la clave.
- Si es opcional en ambas entidades, se puede implementar en cualquiera de las 2. Por motivos de eficiencia, se suele hacer en la que se prevea que tenga menos filas.
- Es poco frecuente que sea obligatoria en ambas. Suelen ser la misma entidad y, por tanto, se implementan en una sola tabla.

# Relación 1:1

USUARIOS	
P * Código	NUMBER
U * Nick	VARCHAR2 (10)
* Nombre	VARCHAR2 (40)
FechaNacimiento	DATE
PK USUARIOS	PK (Código)
UK_NICK	(Nick)

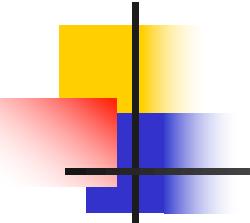
Buzón	
P * Dirección	VARCHAR2 (30)
F * Código	NUMBER
PK Buzón	Dirección
IDX Buzón	Código

CREATE TABLE Buzón ( Dirección VARCHAR2 (30) PRIMARY KEY,  
Código NUMBER NOT NULL UNIQUE,  
CONSTRAINT Proporciona FOREIGN KEY  
( Código ) REFERENCES USUARIOS );

obligatoria

uno a uno

relaciona



## Relación ES\_UN o SUBENTIDAD

---

- Una subentidad  $S$  de una fuerte  $F$  se implementa incluyendo una referencia en  $S$  hacia  $F$
- La clave Primaria de  $S$  será la misma que la clave de la entidad fuerte  $F$
- Esa es la diferencia con la relación de debilidad. Las entidades débiles tienen otro atributo más como parte de la clave. Las subentidaes, no.
- Existen otras posibilidades de Implementación:

# Relación ES\_UN o SUBENTIDAD

Entity Properties - PARCHE

**General**

Name:	PARCHE
Short Name:	
Synonyms:	
Synonym to display:	
Preferred Abbreviation:	
Long Name:	PARCHE
FWD Engineer Strategy:	Single Table
Based on Structured Type:	Single Table Table per child Table for each entity PARCHE_DIFERENCIAS
Super Type:	
Source:	
Classification Type:	
Scope:	

# Relación ES\_UN o SUBENTIDAD



CREATE TABLE PARCHE (

    Versión VARCHAR2 (10) ,  
    VERSIONES\_DISPONIBLES\_Código NUMBER NOT NULL ,  
    VERSIONES\_DISPONIBLES\_Nombre VARCHAR2 (40) NOT NULL );

ALTER TABLE PARCHE

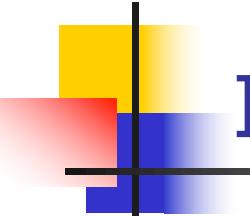
    ADD CONSTRAINT PARCHE\_PK PRIMARY KEY ( VERSIONES\_DISPONIBLES\_Código,  
    VERSIONES\_DISPONIBLES\_Nombre ) ;

ALTER TABLE PARCHE ADD CONSTRAINT FK\_ASS\_4 FOREIGN KEY (

    VERSIONES\_DISPONIBLES\_Código, VERSIONES\_DISPONIBLES\_Nombre )  
REFERENCES VERSIONES\_DISPONIBLES (Código, Nombre) ;

# Resumen

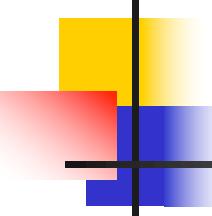
	FK	PK	Unique	Not NULL
DÉBIL				
SUBENTIDAD		exclusivamente		
1:1				Si obligatoria
M:1				Si obligatoria
M:M	(2)	(2) exclusivamente		Disparadores



# Indice

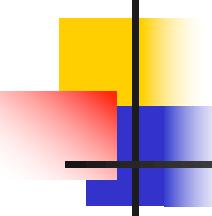
---

- Definición de Base de Datos y conceptos principales
- El Modelo E/R
- El Modelo Relacional
- Paso del E/R al Relacional
- **DML**
- Normalización



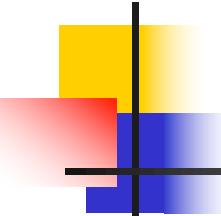
# DML. SELECT

- **SELECT** <Select\_list>
  - Expresiones a recuperar (atributos, funciones, operaciones...).
- **FROM** <Table\_list>
  - Tablas necesarias para la consulta (incluyen tablas de reunión, subconsultas...).
- [ **WHERE** <Condición> ]
  - Condición de selección de tuplas, incluyendo condiciones de reunión.
- [ **GROUP BY** <Atributos\_para\_Agrupar> ]
  - Atributos por los que agrupar el resultado (cada grupo tendrá los mismos valores en estos atributos). Las funciones de grupo o de agregación se aplicarán sobre cada uno de estos grupos. Se aplicarán sobre todas las tuplas si no existe esta cláusula.
- [ **HAVING** <Condición\_de Grupo> ]
  - Condición sobre los grupos (no sobre las tuplas).
- [ **ORDER BY** <Atributos\_para\_Ordenar> ]
  - Atributos por los que ordenar. El orden de estos atributos influye.



# DML. INSERT

- **INSERTA** una o varias tuplas en una relación. Formato:  
`INSERT INTO <Tabla> VALUES (a1, a2, ..., an);`
  - **Lista de valores:** En el mismo orden en el que fue creada la tabla con  
Ej.: `INSERT INTO pieza VALUES (4,'Teja',50,1000);`
- Puede especificarse sólo un **subconjunto de atributos**. Formato:  
`INSERT INTO <Tabla>(<Lista_Atribs>) VALUES (...);`
  - **Lista de atributos:** Nombres de los atributos en los que se desea insertar algún valor. Los valores deben estar en ese orden.
    - Atributos ausentes: Se les asigna su valor por defecto o NULL.
    - Deben estar todos los atributos que no admiten NULL y que además no tienen valor por defecto (restr. `NOT NULL` y `DEFAULT`).
- **Insertar varias tuplas:** Separadas por comas y entre paréntesis.
  - Se puede sustituir la cláusula `VALUES (...)` por una subconsulta:  
Se insertarán TODAS las tuplas recuperadas por esa subconsulta



# DML. DELETE

- **BORRA** una o varias tuplas en una relación. Formato:

**DELETE [FROM] <Tabla> [<Alias>] [WHERE <Cond>];**

- **Borra** las tuplas que cumplan la condición.
- **Puede implicar el borrado de otras tuplas en otras tablas**, para que se cumpla una restricción de integridad referencial.
- **Si se omite la cláusula WHERE** (y su condición), se borran todas las tuplas de la tabla: La tabla quedará vacía (pero sigue existiendo).
- **Ejemplos:**

- **DELETE** Suministrador **WHERE** S# **IN** (2, 4, 8);

- **DELETE** Suministros SP

**WHERE** S# **IN** (**SELECT** S# **FROM** Suministrador

**WHERE** Ciudad='Tegucigalpa');

- **DELETE** Pieza

**WHERE**

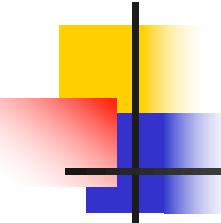
Peso-250 > (**SELECT** AVG(Peso) **FROM** Pieza

**WHERE** Peso >

(**SELECT** AVG(Peso)

**FROM**

Pieza) );

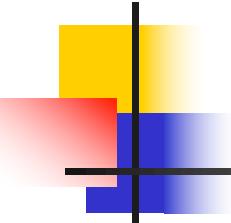


# DML. UPDATE

- **ACTUALIZA** o **MODIFICA** tuplas de una relación.

Formato: **UPDATE** <Tabla> [<Alias>] **SET** <Actualizaciones>  
[ **WHERE** <Cond> ];

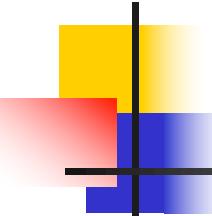
- <Actualizaciones> puede ser:
  - **1.** Una lista de asignaciones separadas por coma del tipo:  
<Atributo> = <Expresión> (expresión puede ser una subconsulta que recupere un valor).
  - **2.** Una lista de atributos entre paréntesis a los que se le asigna una subconsulta:  
(<A<sub>1</sub>>, . . . , <A<sub>m</sub>>) = (<Subconsulta>)
- La cláusula **WHERE** selecciona las tuplas a modificar.
- **UPDATE** Pieza **SET** Nombre='Eje', Peso=9 **WHERE** P#=5;
- **UPDATE** Pieza **SET** Cantidad=Cantidad+100  
**WHERE** P# **IN** (**SELECT** P# **FROM** Suministros  
**WHERE** S# **IN** (3,7));
- **UPDATE** Pieza **SET** (Peso,Cantidad)=(**SELECT** Peso, Cantidad **FROM** Pieza **WHERE** P#=6) **WHERE** P#=5;



# Indice

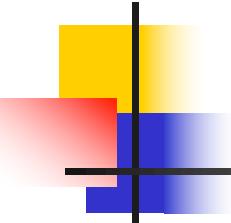
---

- Definición de Base de Datos y conceptos principales
- El Modelo E/R
- El Modelo Relacional
- Paso del E/R al Relacional
- DML
- Normalización



# Información redundante y anomalías en tuplas

- Mezclar atributos de distintas entidades no es adecuado
- Provoca redundancia en el almacenamiento
- Provoca anomalías de inserción, modificación y borrado
- Imaginemos la relación EMP\_PROJ ( Empleado#, Proyecto#, ENombre, PNombre, N\_horas)
  - Cambiar el nombre del proyecto 1 de 'AEROESPACIAL' a 'MANTENIMIENTO' produce que cambie para los 100 empleados trabajando en el proyecto 1. **Anomalía de modificación.**
  - No podemos insertar un proyecto a menos que asignemos al menos un empleado. **Anomalía de inserción.**
    - A la inversa, no podemos insertar un empleado a menos que exista un proyecto para él.
    - Cuando un proyecto se elimina, se eliminará la información de todos los empleados asignados a ese proyecto. Además, si se elimina el único empleado de un proyecto, eliminaremos también ese proyecto. **Anomalía de borrado.**

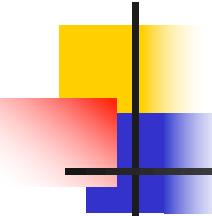


# Bah, pero esto pasa sólo si mezclas atributos de distintas entidades

## Anomalía de modificación

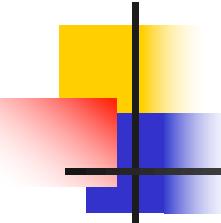
¿Qué ocurre si el gobierno modifica el porcentaje de un tipo de IVA en las facturas vigentes?





# Semántica de la relación de atributos

- En el diseño de una base de datos, cada tupla en una relación (tabla) debe representar una entidad o instancia de la relación.
  - Los atributos de diferentes entidades (EMPLOYEEs, DEPARTMENTs, PROJECTs) no debería mezclarse en una misma relación.
  - Utilizar claves foráneas para referenciar otras entidades
  - La entidad y sus atributos deben formar una unidad tan independiente como sea posible.
- Por tanto, el diseño que realicemos debe estar, en la medida de lo posible, libre de redundancias y anomalías.
- El objetivo de la **normalización** es proporcionar un procedimiento sistemático para ello.



# Dependencias funcionales

Indicativo	Numero	País	Abonado
34	952131000	España	Universidad de Málaga
34	951299316	España	Turismo Andaluz
34	951299300	España	Turismo Andaluz
1	800-275-2273	EEUU	Apple
1	800-426-9400	EEUU	Microsoft
1	951299300	Canadá	Peter Smith
1	800-555-0077	EEUU	Turismo Andaluz
...	...	...	...

País, Número → Abonado

País → Indicativo

Son dependencias funcionales. ¿Hay algunas más?

**Preguntas:**

¿Hay valores repetidos en la tabla? ¿Hay redundancia en la tabla? ¿Dónde? ¿Es mala la redundancia?

**Preguntas:** ¿Es igual  $A \rightarrow B, C$  que  $A \rightarrow B, A \rightarrow C$ ?

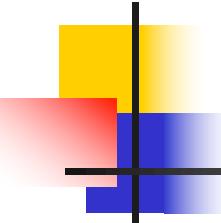
## Definición

Entre los atributos  $A_1, A_2, \dots, A_n$  y  $B$  de una relación  $R$  hay una Dependencia Funcional (DF) si siempre que dos tuplas de  $R$  coincidan en sus valores de  $A_1, A_2, \dots, A_n$ , coinciden también en sus valores de  $B$ .

## Notación

$A_1, A_2, \dots, A_n \rightarrow B$

Generalizamos la definición:  $A_1, A_2, \dots, A_n \rightarrow B_1, B_2, \dots, B_m$



## Definiciones previas

- Se define un **atributo primario** como aquel que es miembro de alguna clave candidata. Cualquier otro atributo es considerado un atributo no primario.
- Una DF  $A \rightarrow B$  es **plena** si no existe otra DF  $C \rightarrow B$  donde  $C$  es un subconjunto propio de  $A$ .
- Una DF  $A \rightarrow B$  es **transitiva** si existe un atributo  $C$  con  $C \not\subseteq A$  y  $B \not\subseteq C$  tal que existen las dependencias funcionales  $A \rightarrow C$  y  $C \rightarrow B$ .

# 1FN

- La primera forma normal es aquella que no permite atributos compuestos o multivalor; es decir, atributos cuyos valores para un tupla individual no son atómicos.
  - Se considera que cualquier tabla, dada las restricciones del DDL está en 1FN.

Indicativo	Numero	País	Abonado
34	952131000	España	Universidad de Málaga
34	951299316	España	Turismo Andaluz
	951299300		
1	800-275-2273	EEUU	Apple
1	800-426-9400	EEUU	Microsoft
1	951299300	Canadá	Peter Smith
1	800-555-0077	EEUU	Turismo Andaluz
...	...	...	...



Indicativo	Numero	País	Abonado
34	952131000	España	Universidad de Málaga
34	951299316	España	Turismo Andaluz
34	951299300	España	Turismo Andaluz
1	800-275-2273	EEUU	Apple
1	800-426-9400	EEUU	Microsoft
1	951299300	Canadá	Peter Smith
1	800-555-0077	EEUU	Turismo Andaluz
...	...	...	...

# 2FN

- Un esquema de relación R se encuentra en 2FN si todo atributo no primario de R se encuentra en una DF plena con la clave primaria.
- Si R no cumple la 2FN tiene que descomponerse en varias relaciones 2FN, por cada clave parcial con sus atributos dependientes.

normalización

Indicativo	<u>Numero</u>	<u>País</u>	Abonado
34	952131000	España	Universidad de Málaga
34	951299316	España	Turismo Andaluz
34	951299300	España	Turismo Andaluz
1	800-275-2273	EEUU	Apple
1	800-426-9400	EEUU	Microsoft
1	951299300	Canadá	Peter Smith
1	800-555-0077	EEUU	Turismo Andaluz
...	...	...	...



<u>Numero</u>	<u>País</u>	Abonado
952131000	España	Universidad de Málaga
951299316	España	Turismo Andaluz
951299300	España	Turismo Andaluz
800-275-2273	EEUU	Apple
800-426-9400	EEUU	Microsoft
951299300	Canadá	Peter Smith
800-555-0077	EEUU	Turismo Andaluz
...	...	...

Indicativo	<u>País</u>
34	España
1	EEUU
1	Canadá
...	...

# 3FN

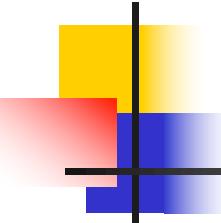
- Un esquema de relación R se encuentra en 3FN si se está en 2FN y ningún atributo no primario de R es transitivamente dependiente de la clave primaria a través de atributos no primarios. Si la dependencia transitiva es a través de una clave, no hay problema.
- Si R no cumple la 3FN tiene que descomponerse en varias relaciones 3FN que incluyan los atributos no primarios que determinen funcionalmente otros atributos no primarios.
- **EMPLEADO (SSN, NumEmp#, Salario)**, ¿está en 3FN?.

normalización

Nacimiento	Numero	País	Abonado
1974	952131000	España	Francisca
1974	951299316	España	Lucas
1964	951299300	España	Lolo
1998	800-275-2273	EEUU	Antonio
1998	800-426-9400	EEUU	Miguel
1981	951299300	Canadá	Peter Smith
1971	800-555-0077	EEUU	Tomás
...	...	...	...



Numero	País	Abonado
952131000	España	Francisca
951299316	España	Lucas
951299300	España	Lolo
800-275-2273	EEUU	Antonio
800-426-9400	EEUU	Miguel
951299300	Canadá	Peter Smith
800-555-0077	EEUU	Tomás
...	...	...



# Definiciones generales de las FN

- Las definiciones anteriores consideran sólo las claves primarias, aunque también pueden definirse en función de cualquier clave candidata.
- Un esquema de relación R se encuentra en 2FN si cualquier atributo no primario de R se encuentra en una DF plena con *todas las claves* de R.
- Un esquema de relación R se encuentra en 3FN si cuando existe una DF  $X \rightarrow A$  en R entonces se cumple una de las siguientes condiciones:
  - a) X es una superclave de R
  - b) A es un atributo primario de R
- La FNBC se puede definir como una 3FN en la que no se permite b).

- Un esquema de relación R se encuentra en FNBC si cuando existe una DF  $X \rightarrow A$  en R entonces se cumple que X es una superclave de R.
- Dicho de otro modo, una relación está en FNBC si y solo si está en 3FN y cada determinante es una clave candidata
- Cada forma normal es estrictamente más fuerte que la anterior
- Existen relaciones que están en 3FN pero no en FBC. El objetivo es conseguir que todas las relaciones de nuestro diseño estén en FNBC.

# Ejemplo. Direcciones Postales:

<b>CPost</b>	<b>Dir</b>	<b>Ciudad</b>
3000	C/ Las Flores N°17	Merida
4858	Av. Bolívar este N°72	Maracay

A Ciudades diferentes le corresponden códigos postales distintos.

En este caso hay dependencia entre el Código Postal y la Ciudad, ya que, conocido el Código Postal se puede conocer la Ciudad, y conocida la Dirección y la Ciudad, se conoce el Código Postal.

Para transformar la tabla en una tabla en FNBC se crea una tabla de Códigos Postales y Ciudades, eliminando de la tabla original la Ciudad, obteniéndose dos tablas, una con los atributos Dirección y Código Postal y otra con el Código Postal y la Ciudad

Tablas en FNBC:

## CÓDIGOS\_CIUDADES

<b>CPost</b>	<b>Ciudad</b>
3000	Merida
4858	Maracay

## CÓDIGOS\_DIRECCIONES

<b>CPost</b>	<b>Dir</b>
3000	C/ Las Flores N°17
4858	Av. Bolívar este N°72

