



# BASES DE DATOS

## Tema 5

### *Diseño Conceptual, Lógico y Físico*

*UCLM- E.S. de Informática*

*Coral Calero, Marcela Genero, Francisco Ruíz*



## Objetivos

- **Aprender a diseñar bases de datos** (BD) relacionales mediante la metodología presentada en el tema anterior.
- Conocer las diversas fases de la metodología:
  - **Modelado conceptual** (con el modelo entidad/interrelación)
  - **Diseño lógico** (con las dos subfases, estándar y específico), y
  - **Diseño físico**.



## Contenido

- Etapas del modelado conceptual
  - Análisis de requisitos
  - Generación del esquema
- Diseños ascendente y descendente
- Características deseables en un esquema conceptual
- Integración de vistas
  - Resolución de conflictos
- Etapas del diseño lógico
- Transformación desde EER a relacional
  - Entidades
  - Dominios y atributos
  - Interrelaciones
  - Dependencias en existencia y en identificación
  - Generalizaciones
  - Dimensión temporal
- Diseño lógico específico
  - Comparativa entre algunos SGBD
- Objetivos del diseño físico
  - Perspectivas de los fabricantes de SGBD
  - Actividades
- Factores que influyen en el diseño físico
- Decisiones de diseño físico



## Bibliografía

### Modelado Conceptual

- Básica
  - Piattini et al. (2006)
    - Cap. 16
- Complementaria
  - Batini et al. (1994)
    - Cap. 3 y 5
  - Conolly y Begg (2002)
    - Cap. 14
  - Elmasri y Navathe (2004)
    - Caps. 3 y 4

### Diseño Lógico

- Básica
  - De Miguel et al. (1999)
    - Caps. 5-7 y 10-11
- Complementaria
  - Batini et al. (1994)
    - Cap. 12
  - Elmasri y Navathe (2004)
    - Cap. 7 y 10-11
  - Conolly y Begg (2002)
    - Cap. 8, 13 y 15
  - Date (2004)
    - Caps. 11 y 12
- Básica
  - De Miguel et al. (1999)
    - Cap. 9

### Complementaria

- Batini et al. (1994)
  - Cap. 3 y 5
- Conolly y Begg (2002)
  - Cap. 14
- Elmasri y Navathe (2004)
  - Caps. 3 y 4

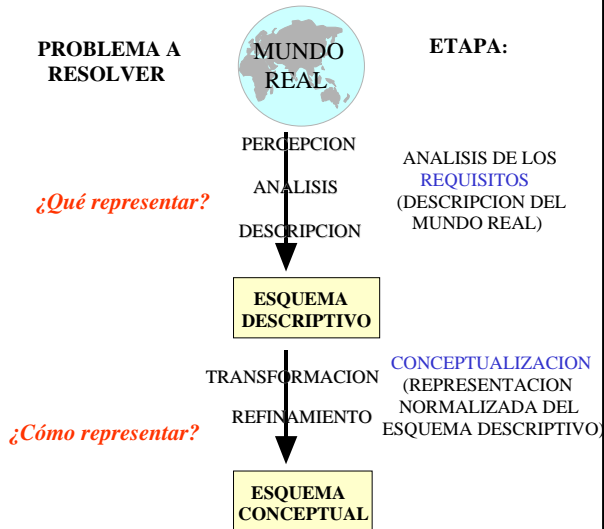
### Diseño Físico

- Básica
  - Elmasri y Navathe (2004)
    - Cap. 16
- Complementaria
  - De Miguel et al. (1999)
    - Cap. 11
  - Silberschatz et al. (2002)
    - Caps. 11, 12, 25

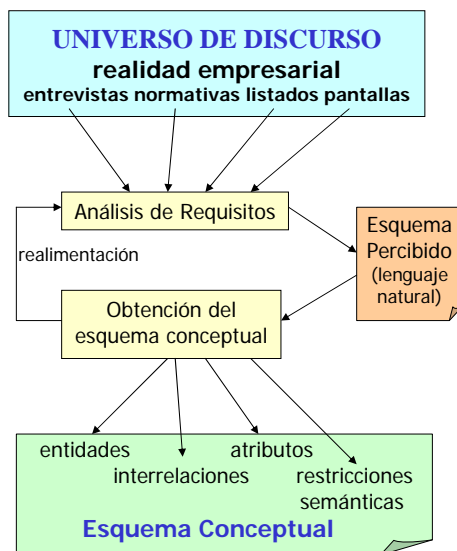


## Etapas del Modelado Conceptual

- El modelado o diseño conceptual es la primera fase del desarrollo de BD.
- Se subdivide en dos etapas:
  - **Análisis de Requisitos,**
  - **Generación del esquema conceptual (Conceptualización)**



## Etapas del Modelado Conceptual



- **Entradas:**
  - Realidad empresarial: entrevistas, normativas, listados, pantallas, etc.
- **Salidas:**
  - Esquema conceptual: entidades, interrelaciones, atributos, restricciones semánticas, etc.



## Etapas del Modelado Conceptual

### Análisis de Requisitos

- Esta primera etapa, en general común para datos y procesos, es de **percepción, identificación y descripción de los fenómenos** del mundo real a analizar.
- En el análisis de requisitos se ha de responder a la pregunta:  
“¿Qué representar?”.
- Mediante el **estudio de las reglas** de una empresa (que proveen el marco para el análisis del sistema) y de **entrevistas a los usuarios** de los diferentes niveles de la organización (que proveen los detalles sobre los datos) se llega a **elaborar un esquema descriptivo** de la realidad.
- El **esquema descriptivo** se representa utilizando el **lenguaje natural**. Con ello se ayuda a que el problema de comunicación usuarios/analistas se reduzca (aunque seguirá siendo muy importante).



## Etapas del Modelado Conceptual

### Análisis de Requisitos

#### Como ven los usuarios a los analistas

- No entienden el negocio, es decir la actividad de la empresa.
- Intentan decirnos como realizar nuestro trabajo.
- No consiguen instrumentar de manera aceptable las especificaciones del sistema.
- Dicen NO a todas nuestras sugerencias.
- Ponen demasiado énfasis en aspectos técnicos.
- Siempre piden más presupuesto.
- Siempre se retrasan.
- Nos piden tiempo y esfuerzo en detrimento de nuestro trabajo.
- No pueden responder de forma rápida y satisfactoria a los cambios necesarios en el sistema.

#### Como ven los analistas a los usuarios

- No saben lo que quieren.
- Tienen muchas necesidades “políticas”.
- Quiero todo “ya”.
- No son capaces de establecer prioridades entre las necesidades.
- Quieren poner sus necesidades específicas por delante de las de la compañía u organismo.
- Rehúsan responsabilidades sobre el sistema.
- No son capaces de dar una definición clara del sistema para que funcione.
- Son incapaces de respetar la planificación.

#### Relaciones entre Analistas y Usuarios



## Etapas del Modelado Conceptual

### Generación del Esquema

- En esta segunda etapa se transforma el esquema descriptivo, refinándolo y estructurándolo adecuadamente.
- Esta etapa responde a la pregunta:  
    **“¿Cómo representar?”**
- En esta **etapa de conceptualización** se habrá de **buscar una representación normalizada** que se apoye en un **modelo de datos** que cumpla determinadas propiedades (coherencia, plenitud, no redundancia, simplicidad, fidelidad, etc.), para llegar así al denominado **esquema conceptual**.
- Para la representación del esquema conceptual, usaremos el **Modelo E/R extendido**, además de una serie de fichas o plantillas que sirven de complemento documental al diagrama EER.



## Etapas del Modelado Conceptual

### Generación del Esquema

- Para generar el esquema conceptual es preciso **interpretar las frases del lenguaje natural** en el que está descrito el esquema percibido, convirtiéndolas en elementos del modelo E/R.
- Si bien **no existen reglas** deterministas que digan qué elemento va a ser una entidad o cuál otro una interrelación, sí es posible enunciar unos **principios generales** que, junto al buen criterio del diseñador, puedan ayudar a elaborar un primer esquema conceptual, que será sometido después a un proceso de refinamientos sucesivos.
  - Así, una **preposición** o frase preposicional entre dos nombres suele ser un **tipo de interrelación**, o también puede establecer la asociación entre una **entidad y sus atributos**.
  - **Ejemplo:** al decir, “el área del departamento”, bien podemos estar indicando la interrelación entre las entidades DEPARTAMENTO y AREA, o bien podemos estar asociando el atributo área a la entidad DEPARTAMENTO.



## Generación del Esquema – heurísticas de Chen

- **Chen** propuso varias **heurísticas** (no son reglas absolutas):
  - Un **sustantivo** (nombre común) que actúa como sujeto o complemento directo en una frase es, en general, **un tipo de entidad**, aunque podría ser un atributo.
    - Ejemplo: en la frase “Los estudiantes solicitan becas” existen dos posibles entidades: ESTUDIANTE (sustantivo que actúa como sujeto) y BECA (que actúa como complemento directo).
  - Los **nombres propios** suelen indicar **ejemplares de un tipo de entidad**;
    - Ejemplo: La frase “el estudiante Luis Pardo” indica que Luis Pardo es un ejemplar de ESTUDIANTE.
  - Un **verbo transitivo o una frase verbal** es **un tipo de interrelación**;
    - Ejemplo: La frase “Los estudiantes solicitan becas” denota una interrelación “solicitar” entre las dos entidades ESTUDIANTE y BECA.



## Generación del Esquema – reglas basadas en roles

- **Carswell y Navathe** proponen **reglas basadas en el papel o rol** que un determinado objeto desempeña en el proceso de información:
  - Una **entidad** es un objeto de datos que tiene más propiedades que su nombre o se utiliza como operando en una sentencia de selección, borrado o inserción.
    - Ejemplos:
      - En la universidad existen profesores que poseen una serie de propiedades, como son el nombre, apellidos, DNI, dirección, etc. PROFESOR es una entidad, ya que tiene unas propiedades (nombre, apellidos, etc.).
      - Cuando un ESTUDIANTE deja de serlo es preciso darle de baja de la BD; ESTUDIANTE es una entidad, por ser un operando en una sentencia de borrado.
  - Un **atributo** es un objeto de datos al que se le asigna un valor o se utiliza como operando en una operación aritmética, lógica o cadena de caracteres.
    - Ejemplo: Se puede consultar si el nombre de un profesor es Paloma, por lo que nombre es, según este enfoque, un atributo.



## Generación del Esquema – reglas basadas en roles

- **Carswell y Navathe** (cont):
  - Una **interrelación** es un objeto de datos que hace posible la selección de una entidad por medio de una referencia a un atributo de otra entidad.
    - Ejemplo: podemos seleccionar los profesores que pertenecen a una determinada área, por lo que “pertenecer” es una interrelación, ya que nos permite seleccionar una entidad (PROFESOR) por medio de una referencia a un atributo de otra entidad (Nombre de área).



## Generación del Esquema – ‘ser’ y ‘tener’

- Los verbos **“SER”** y **“TENER”** han sido especialmente estudiados:
  - **“ES UN”** permite crear jerarquías de entidades, que corresponde al concepto de **generalización**.
    - **Ejemplo:** “...tanto un doctor como un no doctor de nuestra BD son profesores...”, que puede modelarse como una especialización de PROFESOR en DOCTOR y NO\_DOCTOR.
  - El verbo **“TIENE”** posee múltiples interpretaciones en castellano, que pueden ser más o menos específicas según la acepción del verbo en la correspondiente frase.



## Etapas del Modelado Conceptual

### Generación del Esquema – ‘ser’ y ‘tener’

- El verbo **TENER** en castellano puede significar:
  - **Interrelación general entre entidades:** cuando el verbo se utiliza como otro cualquiera, sin una acepción específica.
    - **Ejemplo:** “...los alumnos **tienen** un tutor...” establece la interrelación entre las entidades ALUMNO y TUTOR.
  - **Asociación de las entidades con sus atributos (agregación).**
    - **Ejemplo:** si decimos que “...los profesores **tienen** un nombre, un DNI y una dirección..”, estamos asociando a la entidad PROFESOR una serie de atributos: nombre, DNI, dirección ...
  - **Agregación de entidades** para formar una entidad compuesta (otro tipo de **agregación**).
    - **Ejemplo:** por ejemplo, “el curso **tiene** una parte teórica y una parte práctica”, donde CURSO es el elemento compuesto y PARTE\_TEÓRICA y PARTE\_PRÁCTICA son los elementos componentes.
  - **Dependencia en identificación** (o en existencia).
    - **Ejemplo:** al decir que “...un curso de doctorado **tiene** varias ediciones...”, en el sentido de que una edición es un ejemplar de un curso de doctorado. El identificador de la entidad que es ejemplar (EDICION) se forma con el identificador de la entidad principal (CURSO) junto a un atributo discriminante de la edición.

UCLM ESI-BDa

5.15



## Etapas del Modelado Conceptual

### Generación del Esquema – atributo vs entidad

- Muchas veces es **difícil determinar** si un objeto de datos se debe modelar **como atributo o como entidad** interrelacionada con la entidad de la que se supone que podría ser atributo.
- Brathwaite (1985) indica que **sólo es preferible** considerar el objeto de datos como **entidad**, en lugar de como atributo, en los siguientes casos:
  - Si **tiene por sí mismo asociados otros atributos**,.
    - **Ejemplo:** si la materia que imparte un profesor (que considerábamos un atributo de PROFESOR) tiene a su vez otros atributos, como número de temas, horas de práctica, horas de teoría, etc.), conviene crear la entidad MATERIA.
  - Si **está relacionado con otras entidades**:.
    - **Ejemplo:** si el área se considera como un atributo de PROFESOR, no será posible reflejar las posibles interrelaciones existentes entre las áreas y los departamentos (el departamento de Informática se compone de las áreas de Lenguajes y Sistemas Informáticos y de Ciencias de la Computación e Inteligencia Artificial).

UCLM ESI-BDa

5.16



## Características Deseables en un Esquema Conceptual

- Como resultado de la fase de modelado conceptual se obtendrá un **esquema conceptual** que debe cumplir los siguientes objetivos:
  - Captar y almacenar el universo del discurso mediante una **descripción rigurosa**, representando la información que describe a la organización y que es necesaria para su funcionamiento.
  - **Aislar la representación** de la información **de los requisitos de la máquina** y exigencias de cada usuario particular.
  - **Independizar la definición** de la información **de los SGBD** en concreto.
- En resumen, **“un esquema conceptual comprende una descripción central única de los distintos contenidos de información que pueden coexistir en una base de datos”**.
- Para ello se debe contar con un **modelo de datos adecuado**:
  - **E/R extendido (EER)**



## Características Deseables en un Esquema Conceptual

- El **Modelo EER** permite obtener esquemas conceptuales que se caracterizan por su:
  - **Claridad**, que el significado no es ambiguo.
  - **Coherencia**, que no existen contradicciones o confusiones.
  - **Plenitud**, en cuanto a que el esquema conceptual ha de representar lo esencial del fenómeno sin buscar la exhaustividad.
  - **Fidelidad**, en el sentido de que la representación del universo del discurso ha de hacerse sin desviaciones ni deformaciones.
  - **Sencillez**, será la máxima posible sin atentar contra las otras características.



## Características Deseables en un Esquema Conceptual

- La **sencillez** del esquema conceptual ha de estar basada en que:
  - El **número de componentes** básicos debe ser tan **reducido** como sea posible.
  - Ha de **separar claramente conceptos distintos**.
  - Debe **preservar la simetría**, es decir, no destruir las simetrías naturales.
  - La **redundancia** tiene que ser cuidadosamente **controlada**.



## Características Deseables en un Esquema Conceptual

- Un **mismo universo del discurso** puede ser representado mediante **múltiples esquemas EER**.
- Es importante establecer en qué medida cada solución propuesta es capaz de recoger la semántica inmersa en el universo del discurso, a fin de poder elegir aquel esquema que incorpore el mayor número de restricciones semánticas del mundo real.
- Para ello, diversos autores han establecido tres **criterios de equivalencia** entre diagramas EER que, de menos a más estrictos, son:
  - **Compatibilidad de dominios**, que asegura que los diagramas representan, en conjunto, el mismo UD.
  - **Equivalencia de dependencias de datos**, que asegura que los diagramas satisfacen las mismas restricciones entre los datos representados.
  - **Equivalencia de ejemplares**, que requiere que los esquemas E/R puedan almacenar los mismos ejemplares del universo del discurso.



## Paradigmas para el Modelado Conceptual Diseños Ascendente y Descendente

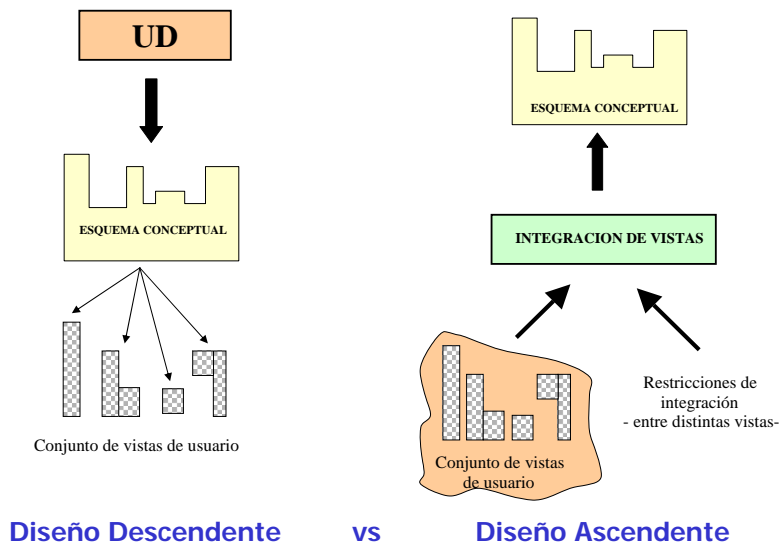
- Tradicionalmente, en el diseño de BD se han venido utilizando distintas **metodologías para elaborar el esquema conceptual**, que pueden agruparse en dos familias:
  - Descendentes (Top-Down)**: el esquema conceptual refleje directamente la visión de la empresa que se intenta modelar en la BD.
    - Se parte del estudio del universo del discurso para elaborar el esquema conceptual y, posteriormente, sobre él se definen las vistas de usuario como subconjuntos de este esquema conceptual.
    - Recomendadas para BD pequeñas.**
  - Ascendentes (Bottom-Up)**: el esquema conceptual es “el resultado de la integración de las vistas de los grupos de usuarios (subsistemas)”.
    - Se empieza construyendo las vistas de cada uno de los subsistemas (que corresponde a las aplicaciones más importantes) y, teniendo en cuenta las restricciones entre dichas vistas, se elabora el esquema conceptual mediante un proceso de **integración de vistas**.
    - Recomendadas para BD medianas y grandes.**

UCLM ESI-BDa

5.21



## Paradigmas para el Modelado Conceptual Diseños Ascendente y Descendente



UCLM ESI-BDa

5.22



## Integración de Vistas

- Al **integrar vistas** (paradigma ascendente) es necesario distinguir entre:
  - **vistas idénticas**: aquellas en las que se encuentran los mismos tipos de objetos, aunque puede que con distintos nombres; y
  - **vistas no idénticas**: poseen (en todo o en parte) distintos tipos de objetos.
    - Dentro de estas hay que distinguir las que son **equivalentes** (por ejemplo, porque lo que en una vista es un atributo en la otra está representado por una entidad) de las que no.



## Integración de Vistas

- El **proceso** consiste en:
  1. Partir de **dos vistas y obtener una nueva vista que las englobe**,
  2. Con la obtenida antes y una nueva obtener otra nueva vista.
  3. Repetir sucesivamente el paso 2 hasta llegar a un esquema global que refleje la estructura de información de la BD completa.
- En este proceso de integración de vistas existen dos etapas:
  - **Resolución de conflictos.**
  - **Análisis de redundancias en interrelaciones.**



## Integración de Vistas Resolución de Conflictos

- Al querer integrar varias vistas se pueden producir diversos problemas:
  - **Conflictos de nombres:** pueden ser tanto por **homonimia** (a dos objetos distintos se les ha asignado el mismo nombre) como por **sinonimia** (un mismo objeto que posee más de un nombre).
  - **Conflictos entre entidades:** pueden ser de varios tipos ...
    - Una entidad es un subconjunto de otra. En este caso la solución consiste en introducir un subtipo.
    - Una entidad es disjunta con respecto a otra, pero ambas poseen atributos comunes; es decir, son un subtipo de una tercera entidad. La solución es crear esa tercera entidad, esto es, el supertipo.
  - **Conflicto entre tipos de objetos en los que un atributo en una vista es una entidad en otra o viceversa:** la solución es transformar el atributo en entidad o la entidad en atributo, según convenga. Así, por ejemplo, si existen atributos de este atributo, o si éste está interrelacionado con otras entidades, convendrá considerarlo como entidad.
  - **Conflictos de cardinalidades en interrelaciones:** pueden reflejar que las dos interrelaciones son la misma, que hay dos interrelaciones distintas o que una de las entidades involucradas tiene uno o varios subtipos.



## Integración de Vistas Análisis de Redundancias en Interrelaciones

- Una vez integradas las vistas, habrá que **analizar si se producen redundancias de interrelaciones**, lo que puede reflejarse gráficamente como ciclos en el diagrama E/R. Estos ciclos se deben detectar y estudiar tal como se vio anteriormente en el tema 2.

### Repaso :

- La existencia de un ciclo no implica la existencia de interrelaciones redundantes.
- Para que una interrelación pueda ser eliminada por redundante se tiene que cumplir :
  - a) que exista un ciclo,
  - b) que las interrelaciones que componen el ciclo sean equivalentes semánticamente,
  - c) que se puedan asociar los ejemplares de las dos entidades que estaban interrelacionadas, aún habiéndose eliminado la interrelación, y
  - d) que la interrelación no tenga atributos o que éstos puedan ser transferidos a otro elemento del esquema a fin de no perder su semántica.



## Etapas del Diseño Lógico

- **Diseño Lógico Estándar:**

- A partir del esquema conceptual, y teniendo en cuenta los requisitos de proceso y de entorno, se elabora un **esquema lógico estándar** (ELS), que se apoya en un **modelo lógico estándar** (MLS), que será el mismo modelo de datos soportado por el SGBD que se vaya a utilizar, pero sin las restricciones ligadas a ningún producto comercial.
- En nuestro caso el MLS es el **modelo relacional**.
- El ELS se describirá utilizando un lenguaje estándar. En nuestra metodología se usa el **SQL**.

- **Diseño Lógico Específico:**

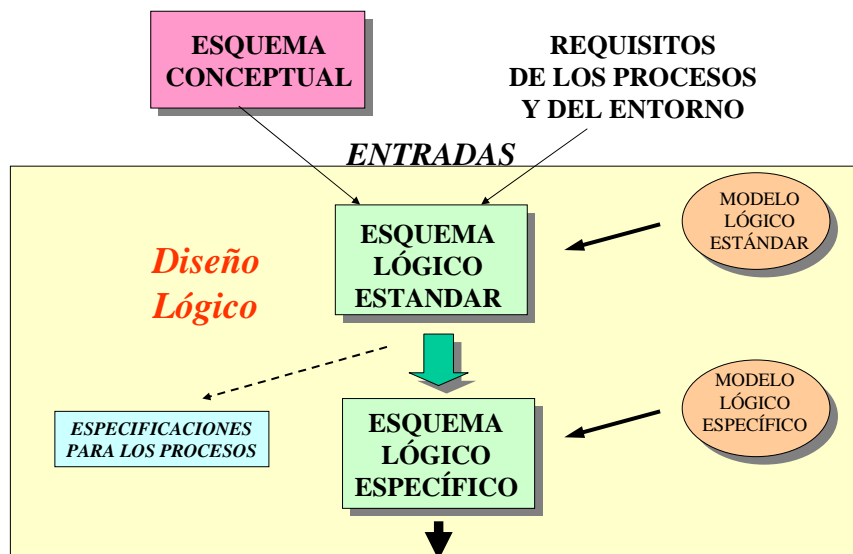
- Con el ELS, y teniendo en cuenta el **modelo lógico específico** (MLE) propio del SGBD (INGRES, SYBASE, DB2, ORACLE, INFORMIX, INTERBASE, etc.), se elabora el **esquema lógico específico** (ELE), que será descrito en el lenguaje de definición de datos (LDD) del producto comercial que estemos utilizando.

UCLM ESI-BDa

5.27



## Etapas del Diseño Lógico



UCLM ESI-BDa

5.28



## Etapas del Diseño Lógico Diseño Lógico Estándar

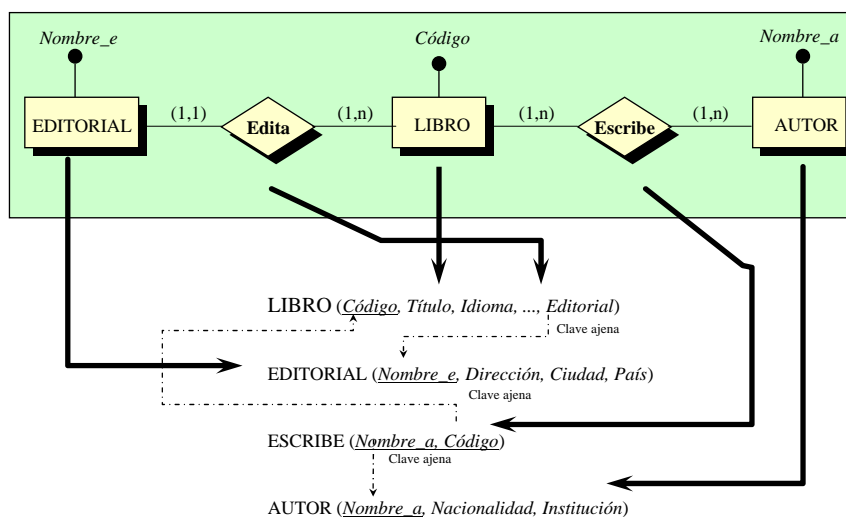
- Las **tres reglas básicas** para convertir un esquema en el modelo EER al relacional son las siguientes:
  - Todo tipo de entidad se convierte en una relación.
  - Todo tipo de interrelación N:M se transforma en una relación.
  - Para todo tipo de interrelación 1:N se realiza lo que se denomina propagación de clave (regla general), o se crea una nueva relación.
- Como el modelo relacional no distingue entre entidades e interrelaciones, ambos conceptos deben representarse mediante relaciones. Esto implica una **pérdida de semántica** con respecto al esquema EER:
  - Las **interrelaciones N:M no se distinguen de las entidades** (ambas se transforman en tablas)
  - Las **interrelaciones 1:N** se suelen representar mediante una **propagación de clave**, desapareciendo incluso el nombre de la interrelación.

UCLM ESI-BDa

5.29



## Etapas del Diseño Lógico Diseño Lógico Estándar – ejemplo de transformación



UCLM ESI-BDa

5.30



## Etapas del Diseño Lógico

### Diseño Lógico Específico

- A partir del ELS obtenido en la etapa anterior, y teniendo en cuenta el modelo lógico específico (MLE) en el que se va a implementar la BD, se elabora el esquema lógico específico (ELE), que será descrito en el LDD del SGBDR elegido.
- El paso del ELS al ELE conlleva un **conocimiento del SGBD** para:
  - Ver en que grado soporta el MLS;
  - Adaptar el ELS a las características específicas del SGBD que se va a utilizar (a su MLE); y
  - Definir el ELE en la sintaxis propia del SGBD.
- Al estudiar la **correspondencia entre** los conceptos del **MLS** y los del **SGBD** concreto pueden darse los siguientes casos:
  - **El SGBD soporta todos los conceptos del MLS** sin restricciones. La transformación del ELS al ELE es prácticamente directa, sólo se ha de transcribir a la sintaxis propia del SGBD utilizado (normalmente SQL).
  - **El SGBD no soporta ciertos conceptos**, o los soporta con restricciones. Se deberán utilizar nuevos objetos, realizar una programación complementaria o, en último caso, transferir a los programas restricciones no soportadas convenientemente por el MLE.



## Etapas del Diseño Lógico

### Diseño Lógico Específico

- Algunos aspectos en los que es necesaria una **transformación adicional**, debido a que los conceptos del MLS no son soportados por el MLE de los SGBDR actuales, son:
  - **Dominios**: Si el SGBDR no permite sentencias para la definición de dominios, existen dos opciones:
    - Al definir la columna de una tabla especificar el tipo de dato, la longitud y (si el esquema lo permite) las restricciones pertinentes (**CHECK**).
    - Construir un procedimiento que compruebe que los valores que se pretenden insertar o modificar se encuentran en una tabla de una sola columna (llamada **tabla de dominio**) que no sea susceptible de inserción, borrado o modificación (tabla estática que sólo el ABD podrá modificar).



## Etapas del Diseño Lógico

### Diseño Lógico Específico

- **Transformaciones Adicionales** (cont)
  - **Claves Primarias:** Si el SGBDR no permite la cláusula **PRIMARY KEY**, el procedimiento sustitutorio es:
    - No admitir nulos en los atributos de la clave primaria (asignar NOT NULL).
    - Definir una restricción de unicidad (UNIQUE) para el conjunto de atributos de la clave primaria.
    - Asegurar la existencia del índice asociado a la restricción de unicidad anterior: crearlo al crear la tabla y no borrarlo nunca.
    - Añadir un comentario en el esquema /catálogo indicando cuál debería ser la clave primaria.



## Etapas del Diseño Lógico

### Diseño Lógico Específico

- **Transformaciones Adicionales** (cont)
  - **Claves Ajenas:** Si el SGBDR no permite la cláusula **FOREIGN KEY**, o no incorpora toda la semántica que conlleva, los pasos a seguir son los siguientes:
    - Introducir las restricciones de clave ajena (integridades referenciales) como requisitos en la especificación de los programas.
    - Asignar la cláusula NOT NULL a los atributos de la clave ajena que no admiten nulos.
    - Mantener la definición de clave ajena como un comentario en el catálogo.
    - Utilizar los mecanismos de seguridad del SGBDR para prohibir las operaciones de los usuarios que pueden violar la restricción de integridad.
    - Realizar un programa que permita el chequeo periódico automático de la integridad referencial.
    - Para mejorar la eficiencia en las opciones anteriores, se puede crear un índice formado por todos los atributos de la clave ajena.



## Etapas del Diseño Lógico

### Diseño Lógico Específico

- **Transformaciones Adicionales** (cont)
  - **Otros conceptos:** Los demás conceptos del modelo relacional que no existan en el MLE suelen necesitar el uso de **disparadores** (triggers) o **procedimientos almacenados** que comprueben las restricciones de integridad definidas en el ELS.



## Transformación desde EER a Relacional

- Con las tres **reglas básicas** comentadas la **calidad** del esquema lógico obtenido puede ser **baja**
  - => Baja normalización
  - => Problemas de redundancias
- Para mejorar esta calidad se utiliza una colección de **reglas de transformación más elaboradas**
  - => Esquemas relacionales que requieren después **poca optimización** (normalización)



## Transformación desde EER a Relacional - reglas

### Reglas de transformación detalladas:

- De entidades.
- De dominios.
- De atributos de entidades.
  - Identificadores principales.
  - Identificadores alternativos.
  - Atributos no identificadores.
  - Atributos multivaluados.
  - Atributos derivados.
- De interrelaciones.
  - Binarias N:M.
  - Binarias 1:N.
  - Binarias 1:1.
  - De grado > 2.
- De atributos de interrelaciones.
- De dependencias en identificación y en existencia.
- De restricciones.
- De restricciones entre interrelaciones.
- De generalizaciones / especializaciones.
- De la dimensión temporal.



## Transformación desde EER a Relacional Entidades

- **Transformación de Entidades.**
  - Cada **tipo de entidad** se convierte en una **tabla**. La tabla se llamará igual que el tipo de entidad de donde proviene.
  - Para su definición en **SQL** se utiliza la sentencia **CREATE TABLE**.



- **Transformación de Dominios.**

- Cada dominio EER se transforma en otro dominio relacional, representado por una sentencia **CREATE DOMAIN** en SQL:

```
CREATE DOMAIN e_civil AS CHAR(1)
CHECK (VALUE IN ('S', 'C', 'V', 'D'))
```

**EER**



**MR (SQL)**

```
DOMINIO E_CIVIL CHAR(1) CHECK
(VALUE IN ('S','C','V','D'))
```



- **Transformación de Atributos de Entidades:**

- Cada atributo de una entidad se transforma en una **columna de la tabla** a la que ha dado lugar la entidad. Teniendo en cuenta los identificadores, esta regla se divide en tres subreglas:
  - **Atributos Identificadores:**
    - Los atributos que son identificadores principales (AIP) pasan a formar la clave primaria de la tabla.
    - En SQL se representan con la cláusula **PRIMARY KEY** dentro de la orden CREATE TABLE.
  - **Atributos Identificadores Alternativos:**
    - Se representan en SQL por medio de la cláusula **UNIQUE** dentro de la orden CREATE TABLE.
  - **Atributos No Identificadores:**
    - Se representan como **columnas de la tabla** correspondiente.



## Transformación desde EER a relacional Dominios y Atributos

**EER**



**MR**

**PROFESOR**

<i>Cod_prof</i>	<i>Nombre</i>	<i>DNI</i>	<i>Dirección</i>	<i>Teléfono</i>	<i>Materia</i>
00001	Juan	12223433	Rios Rosas, 23	670123123	Ing. Software
00002	Coral	54656754	Alarcos, 8	567983456	Bases de datos
00003	Belén	53567523	Getafe, 4	6º9267854	Orientación objetos
00004	Goyo	97856757	Pez, 102	679345763	Sistemas operativos
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
03568	Roberto	34534522	Fundación, 10	639456239	Redes

← CLAVE PRIMARIA

**Ejemplo de transformación de un tipo de entidad y sus atributos**

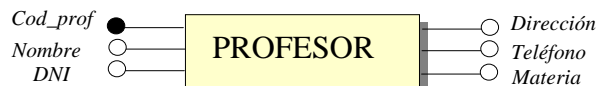
UCLM ESI-BDa

5.41



## Transformación desde EER a relacional Dominios y Atributos

**EER**



**MR-SQL**

```
CREATE TABLE Profesor (
    Cod_Profesor Códigos,
    Nombre Nombres,
    DNI DNIS, NOT NULL
    Dirección Lugares,
    Teléfono Nos_Teléfono,
    Materia Materias,
    PRIMARY KEY (Cod_Profesor),
    UNIQUE (DNI));
```

**Ejemplo de transformación de un tipo de entidad y sus atributos**

UCLM ESI-BDa

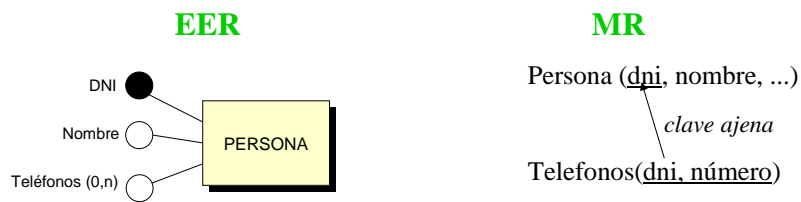
5.42



## Transformación desde EER a relacional Dominios y Atributos

### • Transformación de Atributos Multivaluados.

- Puesto que el MR no permite dominios multivaluados, deberá crearse una nueva tabla cuyos únicos atributos (y clave primaria) serán la concatenación de la clave primaria de la entidad original y el atributo multivaluado. Además, se debe crear una clave ajena referenciando a la tabla primera.



UCLM ESI-BDa

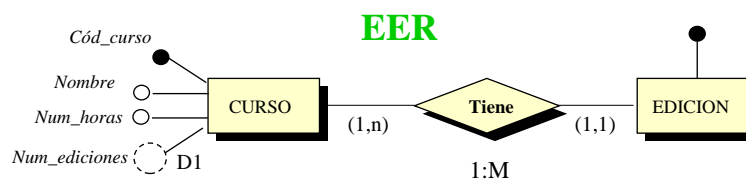
5.43



## Transformación desde EER a relacional Dominios y Atributos

### • Transformación de Atributos Derivados

- No existe para los atributos derivados una representación directa y concreta en el MLE. Por tanto, **se deben tratar como atributos normales**, que pasarán a ser columnas de la tabla correspondiente. Además, se deberá construir un **disparador que calcule el valor del atributo** derivado cada vez que se inserten o borren las ocurrencias de los atributos que intervienen en el cálculo y añadir la restricciones correspondientes.



UCLM ESI-BDa

5.44



## Transformación desde EER a Relacional Interrelaciones – binarias n:m

### • Transformación de Interrelaciones N:M.

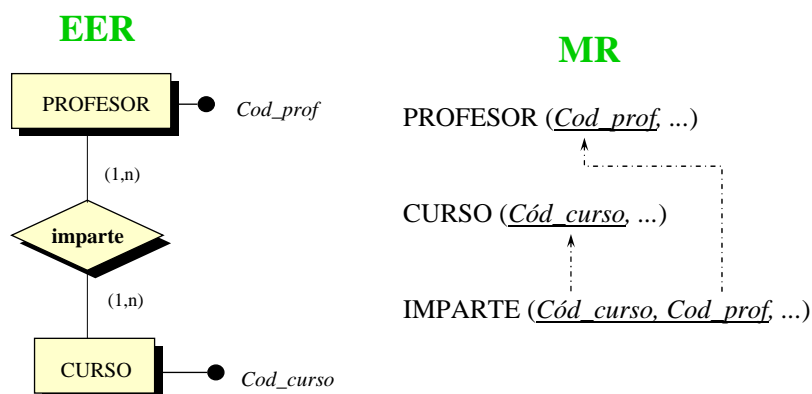
- Un tipo de **interrelación N:M** se transforma en una **tabla** que tendrá como clave primaria la concatenación de los AIP de los tipos de entidad que asocia.
- Además, cada uno de los atributos que forman la clava primaria de esta tabla también son claves ajenas que referencian a las tablas en que se han convertido las entidades interrelacionadas (claves primarias):
  - En **SQL** se representan con la cláusula **FOREIGN KEY** dentro de la sentencia de creación de la tabla.
- Para cada clave ajena así obtenida deberá estudiarse cuales son los **modos de borrado y modificación** adecuados (opciones **ON DELETE** y **ON UPDATE** en SQL). Las opciones permitidas en SQL son:
  - Operación restringida (en caso de no especificar la acción o poner **NO ACTION**)
  - Puesta a nulo (**SET NULL**)
  - Puesta a valor por defecto (**SET DEFAULT**)
  - Operación en cascada (**CASCADE**).
- Las **cardinalidades mínimas** de las entidades participantes en la interrelación se pueden modelar utilizando restricciones **CREATE ASSERTION**.

UCLM ESI-BDa

5.45



## Transformación desde EER a Relacional Interrelaciones – binarias n:m



### Transformación de una interrelación binaria N:M

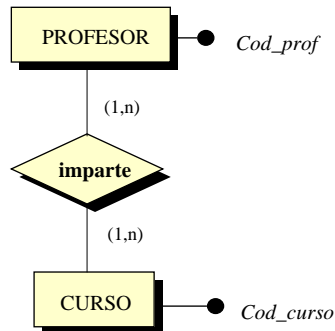
UCLM ESI-BDa

5.46



## Transformación desde EER a Relacional Interrelaciones – binarias n:m

### EER



### MR-SQL

```
CREATE TABLE Imparte
(Cod_Profesor Codigos_P,
Cod_Curso Codigos_C, .... ,
PRIMARY KEY (Cod_Profesor, Cod_Curso),
FOREIGN KEY (Cod_Profesor)
REFERENCES Profesor
ON DELETE CASCADE
ON UPDATE CASCADE,
FOREIGN KEY (Cod_Curso)
REFERENCES Curso
ON DELETE CASCADE
ON UPDATE CASCADE)
```

### Transformación de una interrelación binaria N:M

UCLM ESI-BDa

5.47



## Transformación desde EER a Relacional Interrelaciones – binarias 1:n

### • Transformación de Interrelaciones 1:N.

- Existen **dos soluciones**:
  - a) Propagar los AIP del tipo de entidad que tiene de cardinalidad máxima 1 a la que tiene N (**propagación de clave**). Esta es la regla habitual.
  - b) Transformar la interrelación en una **tabla** como si se tratara de una interrelación N:M; pero ahora la clave primaria de la tabla creada es sólo la clave primaria de la tabla a la que le corresponde la cardinalidad máxima n.
- La opción b) se utiliza sólo cuando
  - El número de ejemplares interrelacionados de la entidad que propaga su clave es muy pequeño y, por tanto, existirían muchos valores nulos en la clave propagada.
  - Se prevé que la interrelación acabará convirtiéndose en una N:M.
  - La interrelación tiene atributos propios y no es deseable propagarlos (a fin de conservar la semántica).

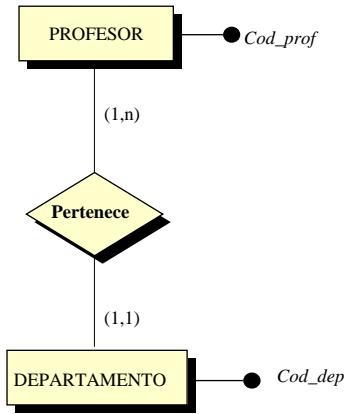
UCLM ESI-BDa

5.48

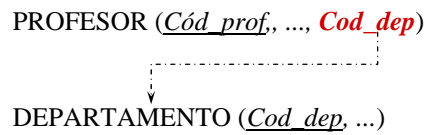


Transformación desde EER a Relacional  
Interrelaciones – binarias 1:n

**EER**



**MR**

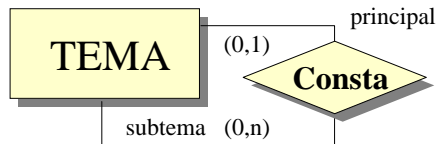


**Transformación de una interrelación 1:N por propagación de clave**



Transformación desde EER a Relacional  
Interrelaciones – binarias 1:n

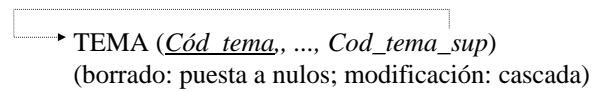
**EER**



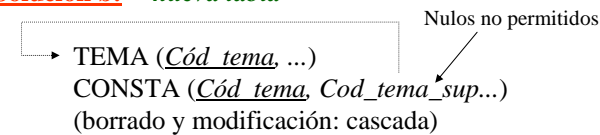
Interrelación reflexiva que modela una jerarquía

**MR**

**Solución a:** *propagación de clave*



**Solución b:** *nueva tabla*

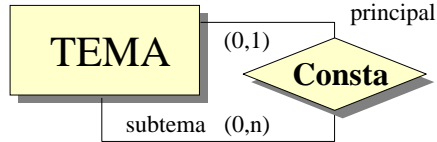


**Transformación de una interrelación 1:N reflexiva**



## Transformación desde EER a Relacional Interrelaciones – binarias 1:n

**EER**



Interrelación reflexiva que modela una jerarquía

**MR-SQL**

```
CREATE TABLE Consta (
  Cod_Tema Codigos,
  Cod_Tema Sup Codigos,
  PRIMARY KEY (Cod_tema)
  FOREIGN KEY (Cod_Tema) REFERENCES Tema
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  FOREIGN KEY (Cod_Tema_SUP) REFERENCES Tema
    ON DELETE CASCADE
    ON UPDATE CASCADE
);
```

**Transformación de una interrelación 1:N reflexiva**

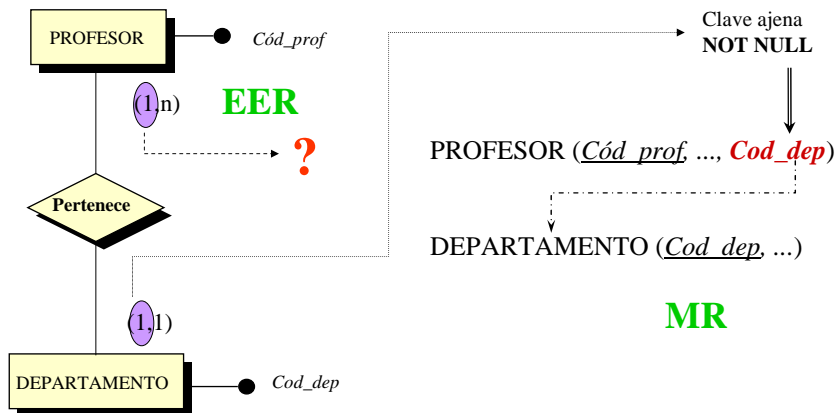
UCLM ESI-BDa

5.51



## Transformación desde EER a Relacional Interrelaciones – binarias 1:n

- Al hacer **propagación de clave**, la **cardinalidad mínima** de la interrelación con máxima 1 se puede modelar usando NOT NULL para el valor 1. Para la interrelación de cardinalidad máxima N es necesaria una restricción (CHECK, ASSERTION o TRIGGER).



UCLM ESI-BDa

5.52



## Transformación desde EER a Relacional Interrelaciones – binarias 1:1

- Una interrelación de tipo 1:1 es un **caso particular de una 1:N**, por lo que se pueden aplicar las dos opciones ya comentadas:
  - crear una nueva tabla, o
  - propagación de clave, teniendo en cuenta que ahora la propagación de la clave puede efectuarse en ambos sentidos.
- Los **criterios** para aplicar una u otra regla de transformación se basan en:
  - las cardinalidades mínimas,
  - recoger la mayor cantidad de semántica posible,
  - evitar los valores nulos, y
  - Aumentar la eficiencia.

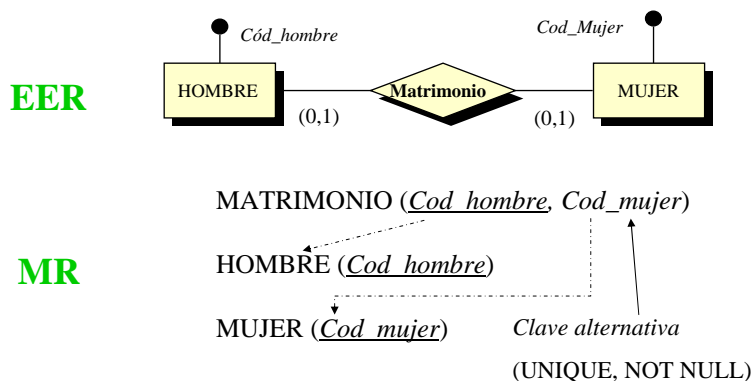
UCLM ESI-BDa

5.53



## Transformación desde EER a Relacional Interrelaciones – binarias 1:1

- Si las entidades que se asocian poseen cardinalidades **(0,1)**, suele ser conveniente transformar la interrelación en una tabla.



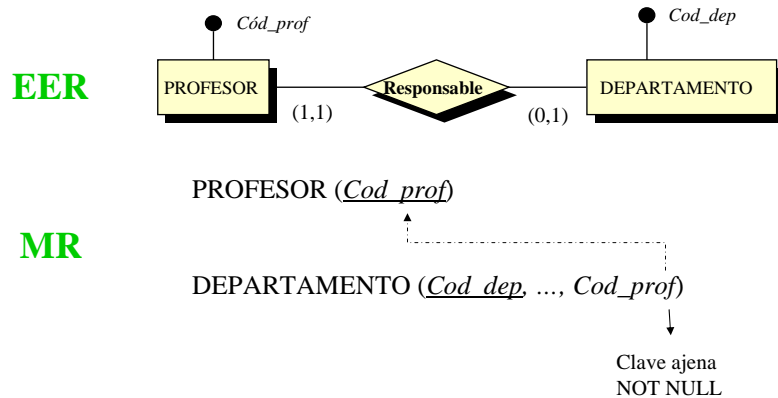
UCLM ESI-BDa

5.54



## Transformación desde EER a Relacional Interrelaciones – binarias 1:1

- Si las entidades que participan en la interrelación poseen cardinalidades (0,1) y (1,1), conviene propagar la clave de la entidad con (1,1) a la tabla resultante de la entidad con (0,1).



UCLM ESI-BDa

5.55



## Transformación desde EER a Relacional Interrelaciones – binarias 1:1

- En el caso de que ambas entidades presenten cardinalidades (1,1), se puede **propagar la clave de cualquiera de ellas** a la tabla resultante de la otra, teniendo en cuenta en este caso los accesos más frecuentes y prioritarios a los datos de las tablas.
- Algunas veces, si la semántica no se perjudica, es conveniente unir las tablas de ambas entidades en una única tabla con todos los atributos.

UCLM ESI-BDa

5.56



## Transformación desde EER a Relacional Interrelaciones – grado $n > 2$

- Se podrían inventar reglas similares a las anteriores para las interrelaciones ternarias o superiores, pero serían demasiadas y muy complicadas.
- Por eso, las interrelaciones de grado mayor que 2 se representan **igual que las interrelaciones N:M**, es decir, creando una nueva tabla cuya clave primaria será la concatenación de las claves primarias de los tipos de entidades participantes.
- Para controlar las **cardinalidades mínimas y máximas** de cada entidad participante deberá recurrirse a restricciones (**CHECK, CREATE ASSERTION, ...**).

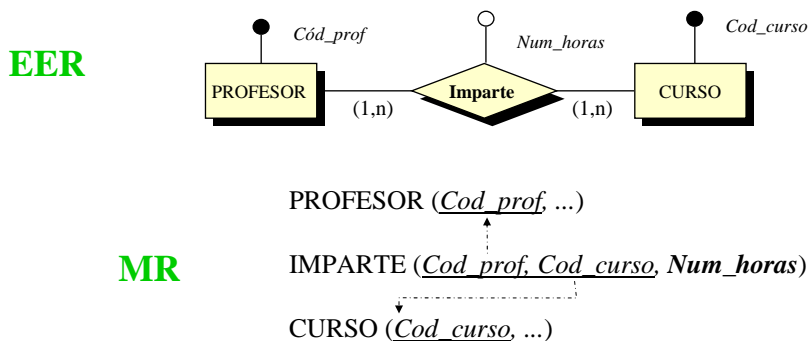
UCLM ESI-BDa

5.57



## Transformación desde EER a Relacional Interrelaciones – atributos

- Si la interrelación se transforma en una **tabla**, todos sus atributos pasan a ser columnas de la tabla.
- En caso de que la interrelación se transforme mediante **propagación de clave**, sus atributos migran junto a la clave a la tabla correspondiente.



UCLM ESI-BDa

5.58



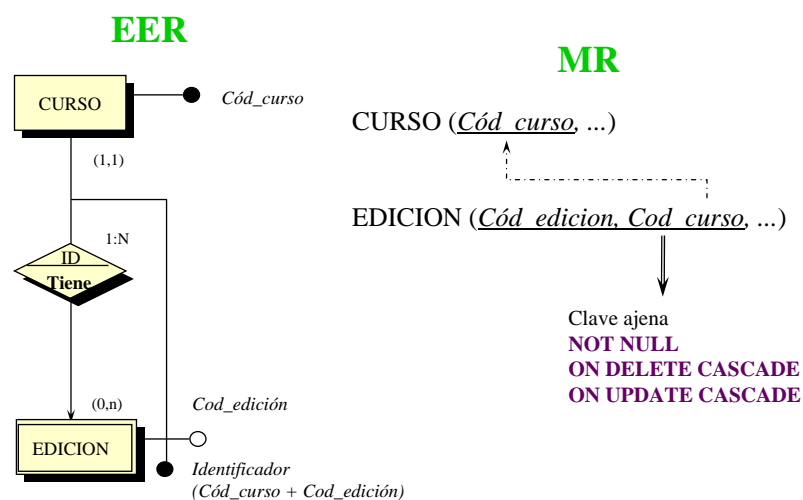
## Dependencias en Existencia y en Identificación

### • Transformación de Dependencias en Existencia y en Identificación

- No son recogidas directamente en el MLS. La manera de transformarlas es utilizar **propagación de clave**, creando una clave ajena, con nulos no permitidos, en la tabla de la entidad dependiente, con modos de **modificación y borrado en cascada**.
- Además, en el caso de **dependencia en identificación**, la clave primaria de la tabla en la que se ha transformado la entidad débil debe estar formada por la concatenación de las claves de las dos entidades participantes en la interrelación.



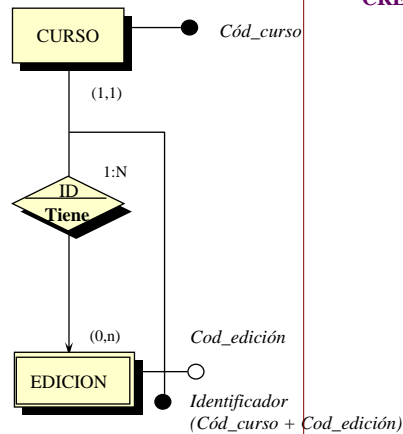
## Dependencias en Existencia y en Identificación





## Transformación desde EER a Relacional Dependencias en Existencia y en Identificación

### EER



UCLM ESI-BDa

### MR-SQL

```
CREATE TABLE Curso (
    Cod_Curso
    Codigos_cursos, ...,
    PRIMARY KEY
    (Cod_Curso));

CREATE TABLE Edicion (
    Cod_Curso
    Codigos_Cursos,
    Cod_Edicion
    Codigos_Ediciones, ...,
    PRIMARY KEY
    (Cod_Curso, Cod_Edicion)
    FOREIGN KEY
    (Cod_Curso)
    REFERENCES Curso
    ON DELETE
    CASCADE
    ON UPDATE
    CASCADE);
```

5.61



## Transformación desde EER a Relacional Restricciones

### • Transformación de Restricciones:

- Existen opciones en el LLS (SQL) que pueden recoger ciertos tipos de restricciones de usuario:
  - Se pueden restringir a un rango determinado los valores de un dominio a través de la cláusula BETWEEN.
  - Se pueden determinar por enumeración los valores que puede tomar una columna en una tabla con la cláusula IN.
  - Otra posibilidad es utilizar la cláusula CHECK dentro de la descripción de una tabla para expresar una condición que deben cumplir un conjunto de atributos de la tabla.
  - También se puede utilizar la sentencia CREATE ASSERTION si la comprobación afecta a atributos de más de una tabla.
  - Además, se pueden utilizar disparadores (CREATE TRIGGER) cuando las opciones anteriores no sean suficientes.

UCLM ESI-BDa

5.62



## Transformación desde EER a Relacional Restricciones

- **Transformación de Restricciones:**

- **Ejemplo:**

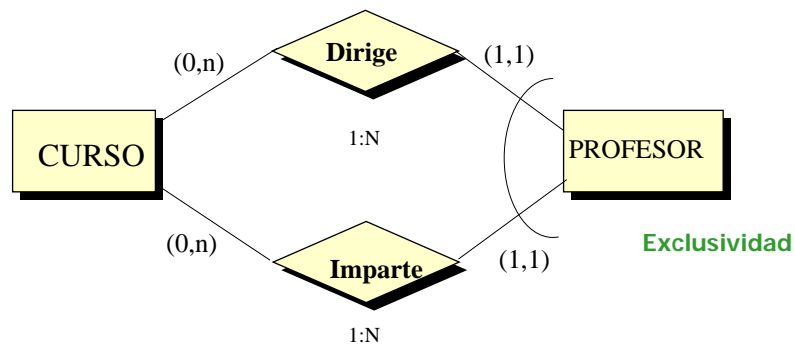
```
CREATE TABLE Curso (  
  Cod_Curso Cursos,  
  Nombre Nombres,  
  Num_Horas Horas  
  Fecha_I Fechas,  
  Fecha_F Fechas,  
  PRIMARY KEY (Cod_Curso),  
  CHECK ( Fecha_I < Fecha_F));
```



## Transformación desde EER a Relacional Restricciones – entre interrelaciones

- **Transformación de Restricciones entre Interrelaciones**

- Para representar restricciones de interrelaciones en el MLS (exclusión, inclusión, etc.) es necesario utilizar las **reglas de transformación de interrelaciones** comentadas junto con la definición de las **restricciones (CHECK, ASSERTION)** pertinentes en cada caso.





## Transformación desde EER a Relacional

### Restricciones – entre interrelaciones

- **Transformación de Restricciones entre Interrelaciones**

```
CREATE TABLE Curso (  
  Cod_Curso Codigos_Cursos,  
  Nombre Nombres, .... ,  
  Cod_prof_dirige Cods_profesores,  
  Cod_prof_imparte Cods_profesores,  
  PRIMARY KEY (Cod_Curso)  
  FOREIGN KEY (Cod_prof_dirige) REFERENCES Profesor  
    ON UPDATE CASCADE  
  FOREIGN KEY (Cod_prof_imparte) REFERENCES Profesor  
    ON UPDATE CASCADE  
  CHECK (( Cod_prof_dirige NOT IN (SELECT Cod_prof_imparte FROM Curso))  
    AND  
    ( Cod_prof_imparte NOT IN (SELECT Cod_prof_dirige FROM Curso)));
```



**Exclusividad**

UCLM ESI-BDa

5.65



## Transformación desde EER a Relacional

### Generalizaciones

- **Transformación de Generalizaciones (tipos y subtipos)**

**Existen 3 soluciones de transformación al MR:**

- **A)** Englobar todos los atributos de la entidad y sus subtipos en una sola tabla.
- **B)** Crear una tabla para el supertipo y tantas tablas como subtipos haya, con sus atributos correspondientes.
- **C)** Considerar tablas distintas para cada subtipo, que contengan, además de los atributos propios, los atributos comunes.
  
- La opción B) es la más completa semánticamente.
- Las opciones A) y C) se justifican por razones de eficiencia en algunos casos.

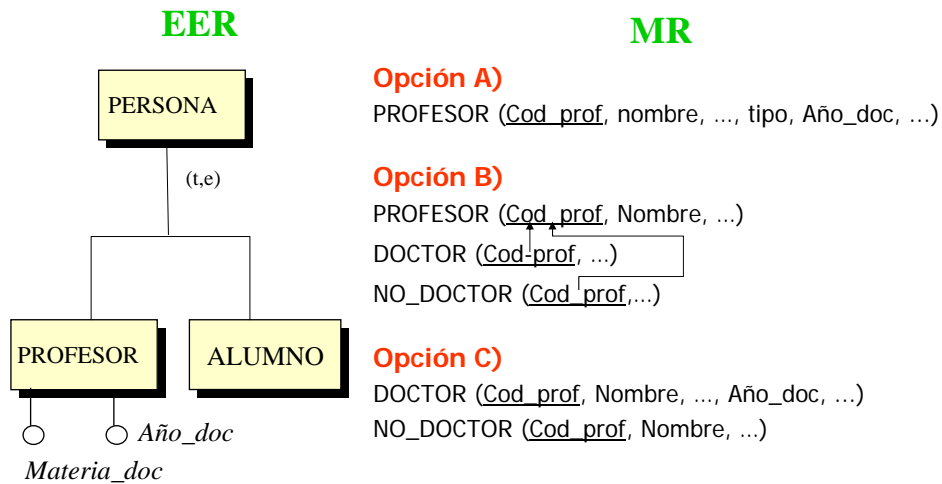
UCLM ESI-BDa

5.66



## Transformación desde EER a Relacional Generalizaciones

- Transformación de Generalizaciones (tipos y subtipos)



UCLM ESI-BDa

5.67



## Transformación desde EER a Relacional Generalizaciones

- Aunque es posible elegir cualquiera de las tres estrategias para la transformación de un tipo y sus subtipos al modelo relacional,
  - desde un punto de vista exclusivamente **semántico** la opción b es la mejor; y
  - desde el punto de vista de la **eficiencia** deberá tenerse en cuenta que:
    - Opción a:** El acceso a una fila que refleje toda la información de una determinada entidad es mucho más rápido (no hace falta combinar varias tablas).
    - Opción b:** La menos eficiente aunque es la mejor desde un punto de vista exclusivamente semántico.
    - Opción c:** Con esta solución se aumenta la eficiencia ante determinadas consultas (las que afecten a todos los atributos, tanto comunes como propios, de un subtipo) pero se disminuye ante otras. Esta solución es en la que se pierde más semántica; además si existe solapamiento se introduce redundancia que debe ser controlada si se quieren evitar inconsistencias.
- Se deberá elegir una estrategia u otra dependiendo de que sea la **semántica o la eficiencia** la que prime para el usuario en un momento determinado.

UCLM ESI-BDa

5.68



## Generalizaciones

- También se deben especificar las **restricciones semánticas** correspondientes
  - Por ejemplo:  
CHECK (  
    (Tipo = 'NO\_DOCTOR' AND Año\_doc IS NULL AND Materia\_doc IS NULL)  
    OR (Tipo= "DOCTOR" AND Año\_doc IS NOT NULL  
        AND Materia\_doc IS NOT NULL) )
- El **atributo discriminante** de la generalización podrá admitir valores nulos en el caso de que haya **recubrimiento parcial** y deberá declararse como **NOT NULL** si el **recubrimiento es total**.
- El atributo discriminante constituirá un grupo repetitivo (multivaluado), si los subtipos **solapan**, debiendo, en este caso, separar este atributo en una tabla aparte según indica la regla de transformación de atributos multivaluados.



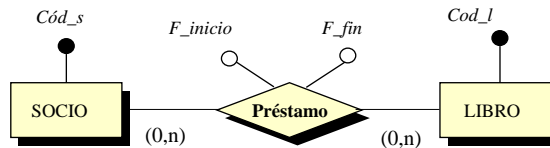
## Dimensión Temporal

- **Transformación de la Dimensión Temporal**
  - En el caso de que en el esquema EER aparezca el tiempo **como un tipo de entidad**, se tratará como otro tipo de entidad cualquiera y, por tanto, se creará una tabla más:  
**TIEMPO (Fecha\_I, Fecha\_F, Hora\_I, Hora\_F, Minutos\_I, ...)**
  - Cuando la dimensión temporal se ha representado a través de **atributos de interrelación de tipo FECHA**, la transformación en el MLS consiste en pasarlos a columnas de la tabla que corresponda, pero teniendo especial **cuidado** a la hora de **elegir la clave primaria** de la tabla resultante, dependiendo de los supuestos semánticos del entorno.



## Transformación desde EER a Relacional Dimensión Temporal

**EER**



**MR**

PRESTA (Cod\_s, Cod\_l, **F\_inicio**, F\_fin)

SOCIO (Cod\_s, ...)

LIBRO (Cod\_l, ...)

**!OJO!**  
con la clave primaria

UCLM ESI-BDa

5.71



## Diseño Lógico Especifico Comparativa entre Algunos SGBD

- En el DEL se deben tener en cuenta las **particularidades de cada versión SQL** de cada fabricante.
- **Ejemplos:**
  - **MS Access 2003** no permite **CREATE ASSERTION** (SQL 2003 sí).
  - Los **tipos de datos** de **MS ACCESS 2003** son **distintos** que en el SQL estándar.
  - **Oracle 9i** soporta **varrays**, que son arrays de longitud variable (SQL 2003 no).
  - **Oracle 9i** permite que la **funciones de agregación definidas por el usuario**, se puedan utilizar en la instrucciones SQL de la misma forma que las funciones incorporadas tales como sum y count (SQL 2003 no).

UCLM ESI-BDa

5.72



## Objetivos del Diseño Físico

- La última etapa de la metodología de diseño de BD es el **diseño físico**, cuyo **objetivo general es satisfacer los requisitos del sistema optimizando la relación costes/beneficios**.
- Esto se concreta en los siguientes **objetivos concretos**:
  - Disminuir los tiempos de respuesta,
  - Minimizar el espacio de almacenamiento,
  - Evitar las reorganizaciones periódicas,
  - Proporcionar la máxima seguridad, y
  - Optimizar el consumo de recursos.



## Objetivos del Diseño Físico

- Las **entradas** de esta etapa son:
  - Lista de objetivos de diseño físico con sus correspondientes prioridades y cuantificación (a ser posible);
  - Esquema lógico específico;
  - Recursos de máquina disponibles;
  - Recursos de software disponibles (sistema operativo, middleware, ...);
  - Información sobre las aplicaciones que utilizarán la BD; y
  - Políticas de seguridad de datos.
- A partir de estas entradas, se producirán las siguientes **salidas**:
  - Estructura interna (esquema interno);
  - Especificaciones para el afinamiento (*tunning*) de la BD; y
  - Normas de seguridad.



## Objetivos del Diseño Físico

### Perspectivas de los Fabricantes de SGBD

- Los fabricantes de SGBDR abordan el problema del diseño físico desde tres perspectivas diferentes:
  - a) El **SGBD impone una estructura interna** y deja muy poca flexibilidad al diseñador. Suele suponer una mayor independencia físico/lógica a costa de menor eficiencia.
  - b) El **ABD diseña la estructura interna**. Supone más trabajo y un perjuicio para la independencia de datos, aunque puede mejorar la eficiencia.
  - c) El **SGBD proporciona una estructura interna inicial** a partir de algunos parámetros dados por el diseñador. El ABD puede ir afinándolos (tunning) para mejorar el rendimiento.
- En general, la mejor opción es la C porque:
  - La BD puede empezar a funcionar inmediatamente;
  - La eficiencia va aumentando al ir realizando ajustes posteriores;
  - La independencia físico/lógica se mantiene.
  - Además, es la estrategia de la mayoría de los SGBDR actuales y también la que mejor se adapta a la metodología propuesta.



## Objetivos del Diseño Físico

### Actividades

- Algunos de las **técnicas más importantes** que se pueden considerar en el diseño físico son:
  - Determinación de los índices secundarios y sus características (compresión, orden, etc.);
  - Tipo de registros físicos;
  - Uso de punteros;
  - Direccionamiento calculado (*Hashing*);
  - Agrupamientos (*Clustering*) de tablas;
  - Bloqueos (*Locking*) y compresión de datos;
  - Definición de tamaños de memorias intermedias (*Buffers*);
  - Asignación de conjuntos de datos a particiones y/o a dispositivos físicos; y
  - Redundancia de datos.
- **NO existe un modelo formal general** para el diseño físico, sino que depende mucho de cada producto comercial concreto.

**Reparar estructuras de ficheros e índices de búsqueda  
(árboles B, hashing, ...)**