

Sistemas Informáticos

Tema 3: Bases de datos distribuidas



Antes de meternos en materia...

- El concepto “base de datos distribuida”, o mejor aún, “base de datos como sistema distribuido” tiene varias acepciones:
 - Por una parte, es un ejemplo clásico de sistema C/S.
 - Pudiendo los datos estar físicamente en un único o distintos servidores
 - Por otra parte, se puede entender que hace referencia a BBDD que mantienen distribuidos los propios datos.
 - Aquí también sería un sistema C/S, pero ahora el servidor en sí mismo es un sistema distribuido, que puede tener una arquitectura C/S o P2P.
 - Aquí principalmente hablaremos de BBDD relacionales (datos centralizados), aunque más adelante explicaremos limitaciones de estos sistemas y las alternativas actuales (generalmente conocidas como soluciones NoSQL)



Contenido

- Introducción:
 - Gestores de bases de datos
 - Objetivos de las bases de datos
- 1. Modelo Entidad-Relación y Modelo Relacional
- 2. SQL
 - DDL
 - DML
 - Subconsultas.
 - Agrupaciones y funciones de agregación.
- 3. Procedimientos almacenados y Triggers
 - Introducción y lenguaje PLPGSQL
 - Procedimientos almacenados
 - Triggers
- 4. SQL inmerso en PHP
- 5. Optimización de consultas
- 6. Temas avanzados:
 - Tecnologías de tratamiento de Big Data
 - Bases de datos NoSQL
 - Map-reduce y métodos similares.
- Bibliografía especial del tema



Gestores de bases de datos (i)

- Un gestor de bases de datos es un sistema que se encarga de la organización, almacenamiento, gestión y recuperación eficiente de la información.
- Incluye:
 - Un lenguaje para modelar la información de acuerdo a un determinado modelo → Curiosidad: una de los aspectos que ha cambiado en la mayoría de las BBDD NoSQL
 - Estructuras de almacenamiento de la información optimizadas para trabajar con un gran volumen de datos.
 - Un lenguaje para recuperar la información almacenada mediante búsquedas dirigidas.
 - Los mecanismos adecuados que le permitan integrarse en un sistema de acceso con control transaccional.
 - Estos 2 últimos puntos siguen siendo las ventajas frente a propuestas más modernas



Gestores de bases de datos (ii)

- El modelo más habitual de gestores de bases de datos actuales es el que sigue el modelo relacional.
- El lenguaje más utilizado actualmente para modelar y recuperar la información es el *Structured Query Language, SQL*.
- Constituyen el núcleo en torno al cual se han desarrollado un gran número de las aplicaciones actualmente en producción (distribuidas y no distribuidas).



Ejemplos de gestores de bases de datos relacionales

- IBM DB/2.
- Oracle.
- Sybase.
- Computer Associates Ingress.
- IBM Informix.
- Microsoft Access.
- Microsoft SQL Server.
- PostgreSQL.
- MySQL (MaríaDB).



Sobre el acceso a los datos

- Originalmente las aplicaciones relacionadas con el manejo de datos se construían sobre un conjunto de ficheros.
- Problemas originados por esta aproximación:
 - Volumetría
 - Si manejamos muchos datos el tamaño del fichero se hace inmanejable
 - Redundancia e Inconsistencia en los datos
 - Acceso a los datos ineficiente:
 - Formatos variados e información duplicada en diferentes ficheros
 - Hay que escribir un programa nuevo para cada nueva funcionalidad que se desee añadir
 - Los datos no están aislados (hay que modificar todos los programas si se varía la estructura de los datos)
 - Problemas de Integridad
 - Las restricciones tienen que ser reforzadas en cada programa y no por la base de datos (edad >0)
 - Es difícil añadir nuevas restricciones o variar las que se establecieron inicialmente



Sobre el acceso a los datos

- Atomicidad de las modificaciones difícil de asegurar
 - La base puede quedarse en un estado inconsistente
 - por ejemplo: transferencia de dinero de una cuenta a otra
- Acceso simultáneo por varios usuarios
 - Se debe permitir el acceso simultáneo para ganar velocidad de proceso
 - Pero, ¿cómo asegurar que dos actualizaciones no son conflictivas?
 - por ejemplo dos personas sacando dinero de la misma cuenta simultáneamente
- Seguridad
 - ¿Cómo restringir parcialmente el acceso a los datos?
- El proceso de abstracción de los datos es difícil



Sistemas Informáticos

Tema 3: Bases de datos distribuidas

3.1 Modelo E-R (entidad-relación)



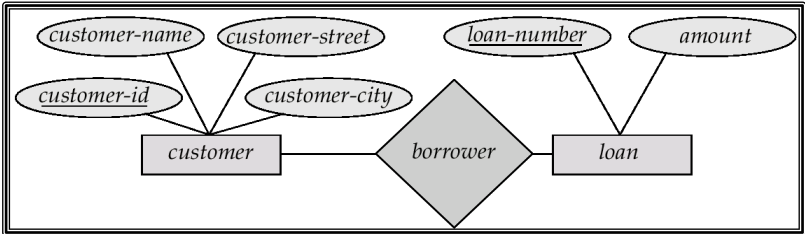
El modelo Entidad-Relación

- Modelo de datos conceptual → independiente del SGBD
 - No confundir con un modelo relacional
- Percepción del mundo real
- Una base de datos puede modelarse como una colección de entidades y relaciones entre entidades
- Elementos básicos en el modelo:
 - Entidad: “cosa” u “objeto” distinguible de otros objetos.
 - Atributo: propiedad de una entidad.
 - Relación: asociación entre entidades.
- Ejemplo:
 - Un banco desea tener almacenada la información sobre sus clientes, los préstamos que tienen éstos con el banco y los datos de sus empleados.



El modelo Entidad-Relacion

Veamos el detalle de una parte del modelo E-R del ejemplo

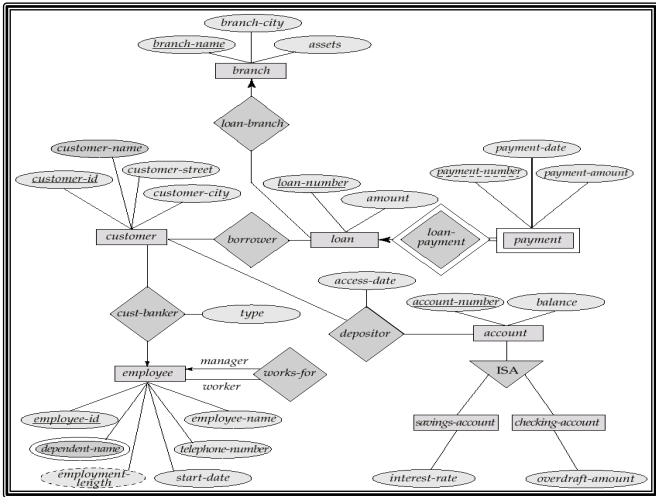


- **Rectángulos:** entidades.
- **Rombos:** relaciones.
- **Líneas:** enlaces entre entidad-atributo y entre entidad-relación.
- **Elipses:** atributos
 - **Elipses dobles** representan atributos multivalorados.
 - **Elipses punteadas** representan atributos derivados.
- **Subrayado:** el atributo que es clave primaria (más adelante)



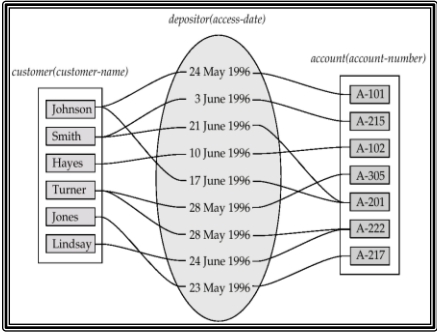
El modelo Entidad-Relación

El modelo E-R del ejemplo



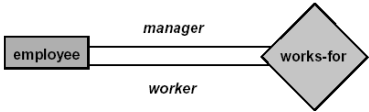
Tipos de relaciones y atributos

- Grado:
 - Normalmente relaciones binarias: p.e. entre cliente y préstamo (tener-prestamo).
 - En ocasiones relaciones ternarias o n-arias ($n > 2$).
- Cardinalidad
- Las relaciones también pueden tener atributos.
- Ejemplo: La relación *poseer* (entre cliente y cuenta) puede tener el atributo *fecha-acceso*.



Roles (en relaciones)

- La función que una entidad juega en una relación es llamado rol (role)
- Normalmente los roles son obvios
 - Ejemplo: *cliente* y *préstamo* en *tener-prestamo*
- Otras veces no son obvios los roles
 - Ejemplo: *trabaja-para* en un gráfico sobre la organización de una empresa
 - *Trabaja-para* *empleado* X *empleado*
 - ¿Quién es el empleado jefe y quien el trabajador?



Claves

- Superclave: uno o más atributos que permiten identificar de forma única a una entidad en el conjunto de entidades.
 - La combinación de *nombre-cliente* e *id-cliente* es una superclave del conjunto de entidades *cliente*.
- Clave candidata: superclaves mínimas
 - *id-cliente* es una clave candidata de *cliente*.
- Clave primaria: la clave candidata elegida para identificar de forma unívoca a una entidad en el conjunto de entidades.
 - No puede tener valor nulo (NULL), no se puede repetir.
 - Preferiblemente que sus valores no suelen cambiar.



Cardinalidad de asignación

- Restricción *cardinalidad de asignación*: el número de entidades con las que puede asociarse otra entidad mediante una relación.
- **Una a una (1-1)**: Una entidad en A está asociada a lo sumo con una entidad en B, una entidad en B está asociada a lo sumo con una entidad en A.
- **Muchas a una (∞ -1)**: Una entidad en A puede estar asociada a lo sumo con una entidad en B, una entidad en B está asociada con un número cualquiera de entidades en A.
- **Muchas a muchas (∞ - ∞)**: Una entidad en A puede estar asociada con un número cualquiera de entidades en B, una entidad en B puede estar asociada con un número cualquiera de entidades en A.

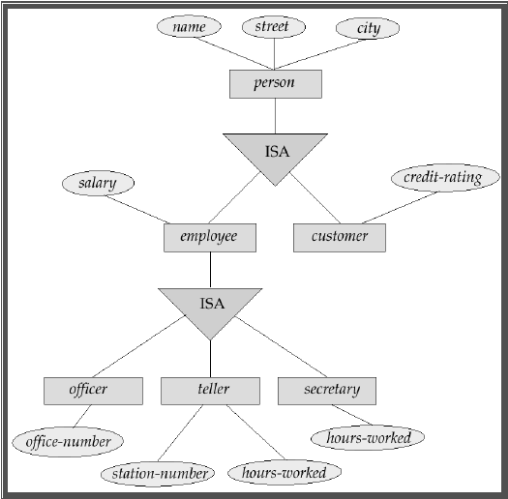


Especialización y generalización

- Proceso de diseño de arriba abajo (top-down): Un conjunto de entidades puede incluir subgrupos de entidades.
- El proceso de designación de subgrupos dentro de un conjunto de entidades se denomina **especialización**.
- Proceso de diseño de abajo a arriba (down-top): varios conjuntos de entidades se sintetizan en un conjunto de entidades de más alto nivel basándose en características comunes.
- Proceso de **generalización**. La generalización es una inversión simple de la especialización.
- Los conjuntos de entidades de nivel más alto: **superclase**
- Los conjuntos de entidades de nivel más bajo: **subclase**.
- **Herencia de atributos**: un conjunto de entidades de más bajo nivel hereda todos los atributos y la participación en las relaciones del conjunto de entidades de más alto nivel con la que está enlazada.



Ejemplo de especialización y generalización



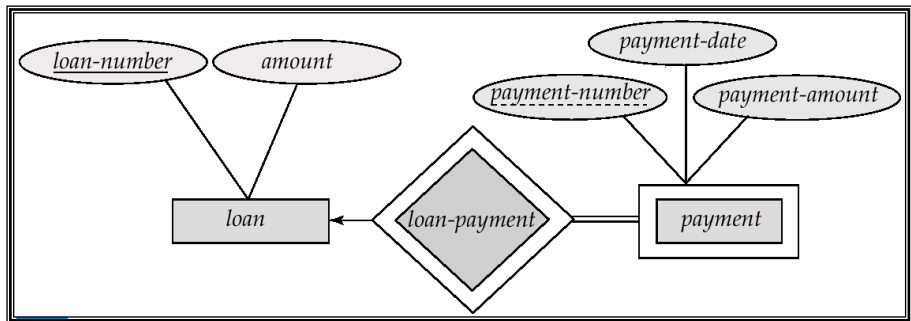
Entidades Débiles

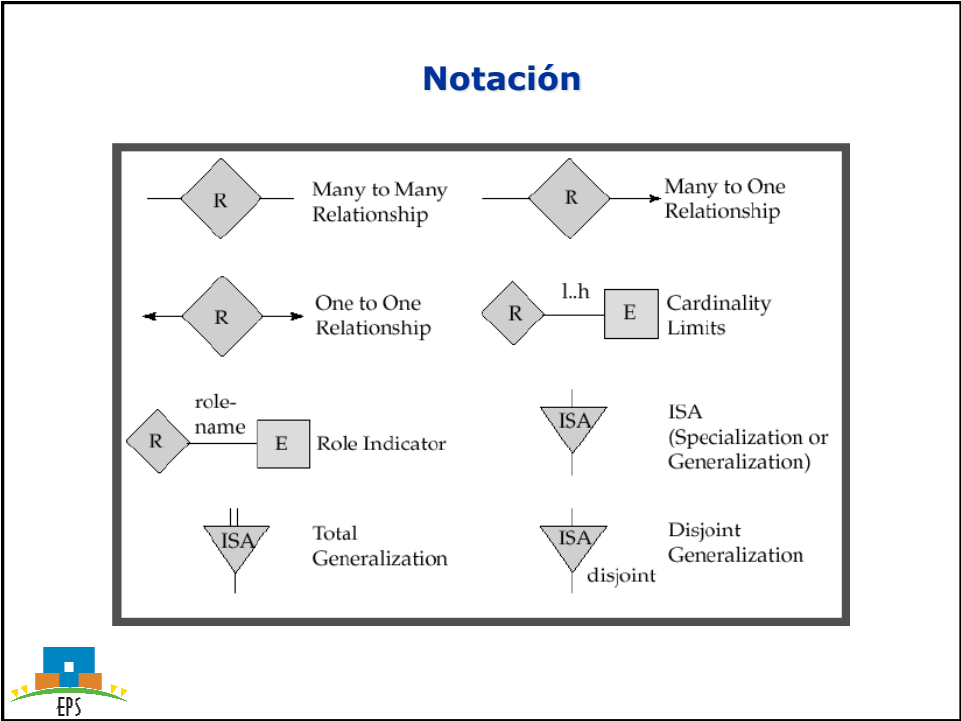
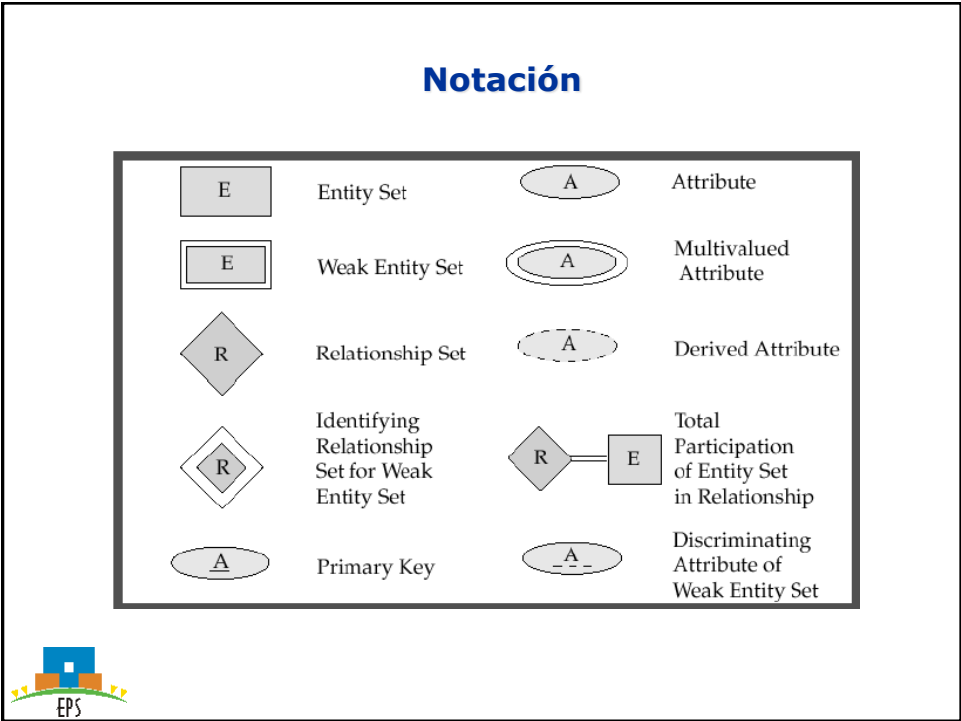
- Una entidad que no tiene clave primaria es una entidad débil.
- Las entidades débiles no están bien definidas sino es con relación a otra entidad
 - la relación tiene que ser muchos (lado de la entidad débil) a uno
- Existe un/os atributo/s discriminante/s (o *clave parcial*) que diferencia todas las entidades débiles relacionadas a la misma entidad (fuerte).
- La clave primaria de una entidad débil se forma unión la clave primaria de la entidad fuerte (asociada con la entidad débil) y los atributos discriminantes.



Entidades Débiles

- Se representan mediante líneas dobles
- El discriminador se marca con una línea de puntos.
- *numero-pago* es el discriminador de “pago”
- Clave primaria (*numero_prestamo, numero_pago*)





Sistemas Informáticos

Tema 3.1: Modelo E-R y modelo relacional

Modelo Relacional



Modelo Relacional

- Todas las bases de datos que se pueden modelar siguiendo el modelo entidad-relación pueden implementarse siguiendo el modelo relacional.
- Una base de datos que se ajusta al modelo relacional puede representarse como un conjunto de tablas
- Convertir un diagrama E-R a tablas es el primer paso para obtener una base de datos relacional
- Normalmente cada entidad y cada relación muchos a muchos da lugar a una tabla
- Cada tabla tienen un conjunto de columnas que suelen corresponderse con los atributos



Definición Formal de Relación

- Dados los conjuntos D_1, D_2, \dots, D_n una **relación** r es un subconjunto de $D_1 \times D_2 \times \dots \times D_n$
Esto es, una relación es un subconjunto de n-tuples (a_1, a_2, \dots, a_n) donde cada $a_i \in D_i$
- Ejemplo: si
 nombre-cliente = {Jones, Smith, Curry, Lindsay}
 direccion-cliente = {Main, North, Park}
 ciudad-cliente = {Harrison, Rye, Pittsfield}
Entonces $r = \{$ (Jones, Main, Harrison),
 (Smith, North, Rye),
 (Curry, North, Rye),
 (Lindsay, Park, Pittsfield) $\}$
es una relación sobre nombre-cliente \times direccion-cliente \times ciudad-cliente



Instancia de una Relación

- Los valores actuales (instancia) de una relación se especifican mediante una tabla.
- Un elemento t de r es una *tupla*, se representa mediante una fila en una tabla

campos (o columnas)		
nombre-cliente	Direccion-cliente	Ciudad-cliente
Jones	Main	Harrison
Smith	North	Rye
Curry	North	Rye
Lindsay	Park	Pittsfield

cliente

tupla
(o filas)



nombre

apellidos

id_cliente


DNI

domicilio

CLIENTE

↓

TABLA CLIENTE



• Del modelo E-R al modelo relacional:
esquemas de sus tablas, sus claves primarias
y sus claves externas

• Los atributos son los campos de la tabla.

• Cada entidad se convierte en una tabla.

• Las relaciones ∞-∞ se convierten en una tabla nueva.

CAMPOS

id_cliente	DNI	Nombre	Apellidos	Domicilio
001	50529234	María	López	C/ Mayor 1
002	47219689	Juan	Castro	C/ Alcalá 5
...

REGISTROS

Modelo Relacional y Tablas

• Tablas

- Una BB.DD. relacional consta de un conjunto de tablas.
- Cada tabla se compone de columnas con nombre.


• Filas

- Cada fila representa una relación entre un conjunto de valores
- El orden de las filas es irrelevante

• Relación

• Subconjunto del conjunto cartesiano de los dominios de los atributos

- El dominio de los atributos debe ser atómico (no se puede subdividir)

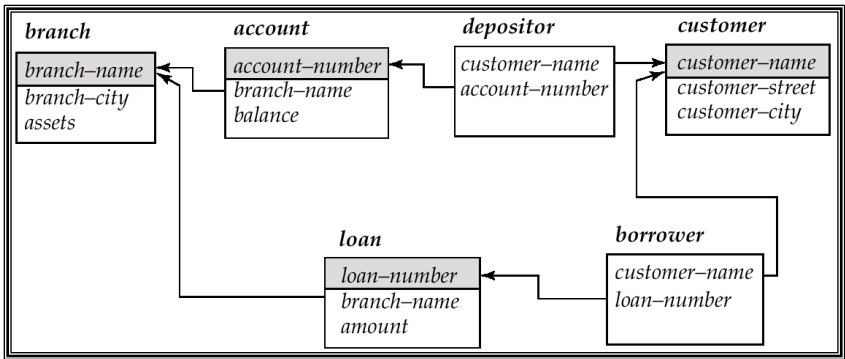


Esquema de la base de Datos

- Esquema BB.DD. = diseño lógico (descripción de la base de datos)
- Esquema BB.DD != datos
- Ejemplo:
Esquema-cliente = (nombre-cliente, direccion-cliente, ciudad-cliente)
 - Define Esquema para la relación cliente
- Indicamos que cliente es una tabla/relación sobre el esquema Esquema-cliente
 - cliente(esquema-cliente)
- convenciones:
 - Nombres de los esquemas en mayúsculas
 - Nombres de las instancias/relaciones en minúsculas



Esquema de la base de datos del ejemplo del Banco



- Notación para los esquemas de las relaciones:
Cuenta(número-cuenta, nombre-sucursal, saldo)

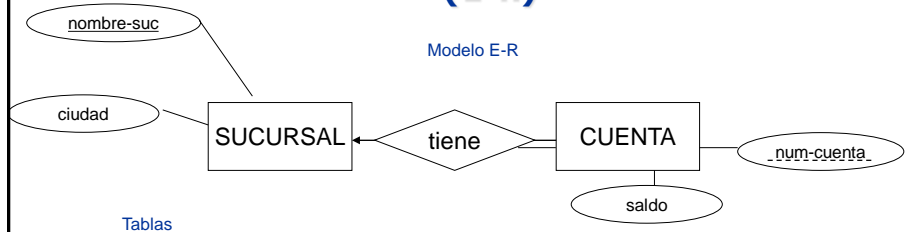


Resumen: Modelo E-R a Tablas

- Hallar claves primarias
- Identificar atributos multivalorados/compuestos y convertirlos en entidades
- Identificar entidades débiles
- Identificar atributos/entidades redundantes
- Identificar relaciones que darán lugar a tablas n a n (y relaciones que no darán lugar a tablas)
- Todas las entidades producen una tabla



Modelo E-R a Tablas Entidades no débiles, atributos y relaciones (1-n)



- Sucursal (nombre-suc, ciudad)
- Cuenta (num-cuenta, nombre-suc, saldo)
- Entidades -> tablas
- Atributos -> campos.
- La relación $\infty-1$: la clave primaria del conjunto de entidades dominante pasa a ser un campo en la tabla.
- La relación 1-1: se añade un atributo a una de las tablas.
- La clave primaria de una entidad débil se forma con la unión de la clave primaria de la entidad fuerte (asociada con la entidad débil) y los atributos discriminantes



Modelo E-R a Tablas

Atributos compuestos y multivaluados

Modelo E-R

Tablas

- Cliente (id-cliente, nombre, apellido1, apellido2)
- Telefono (numero-telefono, id-cliente)
- Los atributos compuestos (no atómicos): se descomponen en partes atómicas.
- Atributo multivalorado: tabla nueva
 - La nueva tabla tendrá una/s columna/s correspondiente/s a la clave primaria del entidad original.

Modelo E-R a Tablas

Entidades débiles

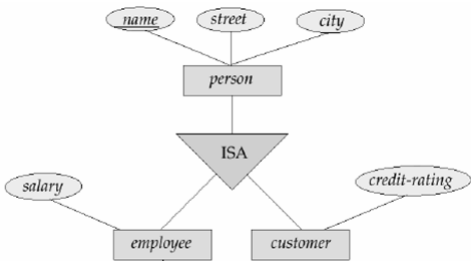
Modelo E-R

Tablas

- Prestamo (num-prestamo, importe)
- Pago (num-pago, num-prestamo, saldo)
- Conjunto de entidades débiles: la tabla contiene:
 - Atributos del conjunto de entidades débiles
 - La clave primaria del conjunto de entidades fuertes del que depende el conjunto de entidades débiles

Modelo E-R a Tablas

Representando especialización como tablas



Propuesta de tablas a crear:

- Persona (nombre, calle, ciudad)
- Empleado (nombre[↑], sueldo)
- Cliente (nombre[↑], credito)

