Bases de Datos

Clase 4: Diseño de Bases de Datos II

Hasta ahora

Sabemos como transformar requisitos de usuario en modelo entidad relación (E/R), pero…
¿Qué hacemos con esto?

Hoy veremos cómo transformar el modelo E/R al modelo relacional

Llaves

Modelo Relacional y Llaves

Modelo Relacional

Los datos se almacenan como tablas:

Películas

1	Interstellar	2014	Fantasía	8.6
2	The Revenant	2015	Drama	8.1
3	The Imitation Game	2014	Biografía	8.1
4	The Theory of Everything	2014	Biografía	7.7

Distinguimos:

- **Relaciones**: a cada tabla le llamamos relación
- **Atributos**: son las columnas de la relación
- **Tuplas**: son las filas de la relación

Modelo Relacional

Una **Relacion** es un ***conjunto*** de tuplas!!!

- Quiere decir; no hay filas repetidas

1	Interstellar	2014	Fantasía	8.6
2	The Revenant	2015	Drama	8.1
3	The Imitation Game	2014	Biografía	8.1
4	The Theory of Everything	2014	Biografía	7.7
4	The Theory of Everything	2014	Biografía	7.7

Modelo Relacional

Una **Relacion** es un ***conjunto*** de tuplas, no hay filas repetidas

1	Interstellar	2014	Fantasía	8.6
2	The Revenant	2015	Drama	8.1
3	The Imitation Game	2014	Biografía	8.1
4	The Theory of Everything	2014	Biografía	7.7
4	The Theory of Everything	2014	Biografía	7.7

Llaves

Ejemplo: Una tabla con información de vinos.

Esquema:

Vinos(Productor, Cepa, Gama, Año, Grados, Orígen)

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapaca	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
…	…		…	…	…

Llaves

Restricciones de integridad

Son restricciones que imponemos a un esquema que *todas* las instancias deben satisfacer

La restricción más importante son las **llaves**

Intuitivamente una llave permite identificar, de manera única, una tupla de la relación

Hay distintas categorías de llaves!

Llaves

Super llave

La **super llave** para una relación R es:

un conjunto de atributos de R tal que no pueden existir dos tuplas en R con los mismos valores de estos atributos

Intuitivamente: si conozco los valores de atributos de la super llave, puedo identificar de **forma única** a la tupla de la relación

Llaves

Superllave

El conjunto de **TODOS** los atributos de la relación siempre forman una super llave!

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapaca	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
…	…		…	…	…

Otras super llaves: (Productor, Cepa, Gama, Año, Grados)
(Productor, Cepa, Gama, Año, Orígen)

Llaves

Superllave

Como **(Productor, Cepa, Gama, Año, Orígen)** es superllave. lo siguiente no está permitido en una instancia:

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapaca	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	13	Maipo

Llaves

Llave candidata

La **llave (o llave candidata)** para una relación R es:

un conjunto de atributos de R que es una super llave de R, y no existe un subconjunto propio de estos atributos que es una super llave

Intuitivamente: **una super llave que no se puede achicar!**

Llaves

Llave

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapaca	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
…	…		…	…	…

(Productor, Cepa, Gama, Año)

Llaves

Llave primaria

La **llave primaria** es una llave candidata que queremos destacar, y la subrayamos en el esquema.

Llaves

Llave primaria

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Tarapaca	Merlot	Reserva	2011	14	San Pedro
Viu Manent	Carmenere	Gran Reserva	2014	12	Colchagua
Perez-Cruz	Cabernet Sauvignon	Edición Especial	2014	14	Maipo
…	…		…	…	…

Vinos(Productor, Cepa, Gama, Año, Grados, Orígen)

Llaves

Si defino mal la llave primaria habrá problemas!

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2015	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2014	14	Maipo
…	…		…	…	…

Vinos(Productor, Cepa, Gama, Año, Grados, Orígen)

Un productor no puede producir vino de misma cepa y gama en distintos años!

Llaves

Si defino mal la llave primaria tendré problemas!

Productor	Cepa	Gama	Año	Grados	Orígen
Perez-Cruz	Cabernet Sauvignon	Reserva	2014	13.5	Maipo
Perez-Cruz	Carmenere	Reserva	2014	13.5	Maipo
Perez-Cruz	Cabernet Sauvignon	Gran Reserva	2014	14	Maipo
…	…		…	…	…

Vinos(Productor, Cepa, Gama, Año, Grados, Orígen)

Un productor no puede producir vinos de distinta cepa o gama en un año!

Llaves

Terminología

Super llave (superkey): cualquier conjunto de atributos qué determina a todo el resto

Llave (candidata): cualquier conjunto de atributos qué determina a todo el resto, y ninguno de sus subconjuntos es una super llave

Llave primaria: una llave candidata que queremos destacar (la subrayada en el esquema)

Modelo Relacional

Llaves

Mejor tipo de llave: ID único de la tupla!!!!
En la tabla Vinos no
existía!

Surrogate key: una llave genérica que simplifica cosas

- **id**

Vinos(id, Productor, Cepa, Gama, Año, Grados, Orígen)

Hay que tener cuidado con la lógica de la relación!

Llaves

Persona(<u>id</u>, rut, nombre)

Llave primaria: id

Ejemplo:

- Número de cliente
- Número de servicio
- SKU

Llaves

Ejemplo

Persona(<u>id</u>, rut, nombre)

Llave primaria: id

Llaves candidatas:

- id
- rut

Llaves

Personas

Persona(<u>id</u>, rut, nombre)

Superllaves:

- id
- rut
- id,rut
- id,nombre
- rut,nombre
- id,rut,nombre

Llaves candidatas:

- id
- rut

Llave primaria: id (**Surrogate key**)

Llaves

Cervezas(Nombre, Tipo, Grados, Ciudad-Origen)

Nombre	Tipo	Grados	Ciudad-Origen
Austral Lager	Lager	4.6	Punta Arenas
Austral Yagan	Ale	5.0	Punta Arenas
Kross 5	Ale	7.2	Curacavi
Kuntsmann Torobayo	Ale	5.0	Valdivia

Vinos(Nombre, Tipo, Año, Grados, Ciudad-Origen)

Nombre	Tipo	Año	Grados	Ciudad-Origen
Tarapacá	Carmenere	2014	13.5	Maipo
Tarapacá	Merlot	2014	13.5	Maipo
Gato	Merlot	2016	14.0	Maule

En-Stock(Nombre, Cantidad, Precio-unitario)

Nombre	Cantidad	Precio unitario

¿Cuál es la llave primaria más natural?

Llaves

Otro ejemplo

Cervezas(Nombre, Tipo, Grados, Ciudad-Origen)

Nombre	Tipo	Grados	Ciudad-Origen
Austral Lager	Lager	4.6	Punta Arenas
Austral Yagan	Ale	5.0	Punta Arenas
Kross 5	Ale	7.2	Curacavi
Kuntsmann Torobayo	Ale	5.0	Valdivia

Vinos(Nombre, Tipo, Año, Grados, Ciudad-Origen)

Nombre	Tipo	Año	Grados	Ciudad-Origen
Tarapacá	Carmenere	2014	13.5	Maipo
Tarapacá	Merlot	2014	13.5	Maipo
Gato	Merlot	2016	14.0	Maule

En-Stock(<u>Nombre</u>, Cantidad, Precio-unitario)

Nombre	Cantidad	Precio unitario

Llaves

Otro ejemplo -- Alternativa 1: nombre mas especifico para vinos

Cervezas(<u>Nombre</u>, Tipo, Grados, Ciudad-Origen)

Nombre	Tipo	Grados	Ciudad-Origen
Austral Lager	Lager	4.6	Punta Arenas
Austral Yagan	Ale	5.0	Punta Arenas

Vinos(<u>Nombre</u>, Tipo, Año, Grados, Ciudad-Origen)

Nombre	Tipo	Año	Grados	Ciudad-Origen
Tarapacá Carmenere 2014	Carmenere	2014	13.5	Maipo
Tarapacá Merlot 2014	Merlot	2014	13.5	Maipo

En-Stock(<u>Nombre</u>, Cantidad, Precio-unitario)

Nombre	Cantidad	Precio-Unitario
Tarapacá Carmenere 2014	600	2000

Llaves

Otro ejemplo -- Alternativa 2: crear un id

Cervezas(<u>id</u>, Nombre, Tipo, Grados, Ciudad-Origen)

id	Nombre	Tipo	Grados	Ciudad-Origen
CauL00	Austral Lager	Lager	4.6	Punta Arenas
CauY00	Austral Yagan	Ale	5.0	Punta Arenas

Vinos(<u>id</u>, Nombre, Tipo, Año, Grados, Ciudad-Origen)

id	Nombre	Tipo	Año	Grados	Ciudad-Origen
VTTC14	Tarapacá	Carmenere	2014	13.5	Maipo
VTTM14	Tarapacá	Merlot	2014	13.5	Maipo

En-Stock(<u>id</u>, Cantidad, Precio-unitario)

id	Cantidad	Precio-Unitario
CAuL00	600	2000
VTTC14	200	6000

Llaves

Otro ejemplo – Alternativa 3: tablas En-Stock separadas

Cervezas(<u>Nombre</u>, Tipo, Grados, Ciudad-Origen)

Nombre	Tipo	Grados	Ciudad-Origen
Austral Lager	Lager	4.6	Punta Arenas
Austral Yagan	Ale	5.0	Punta Arenas
…	…	…	…

Vinos(<u>Nombre</u>, <u>Tipo</u>, <u>Año</u>, Grados, Ciudad-Origen)

Nombre	Tipo	Año	Grados	Ciudad-Origen
Tarapaca	Carmenere	2014	4.6	Punta Arenas
Tarapaca	Merlot	2014	5.0	Punta Arenas
Gato	Merlot	2016	14.0	Maule

Cerveza-En-Stock(<u>Nombre</u>, Cantidad, Precio-Unitario)

Nombre	Cantidad	Precio-Unitario
Austral Lager	600	2000

Vino-En-Stock(<u>Nombre</u>, <u>Tipo</u>, <u>Año</u>, Cantidad, Precio-Unitario

Nombre	Tipo	Año	Cantidad	Precio-Unitario
Tarapaca	Merlot	2014	200	6500

Llaves

Otro ejemplo – Alternativa 4: combinar las tablas

Cervezas(<u>Nombre</u>, Tipo, Grados, Ciudad-Origen, Cantidad, Precio-Unitario)

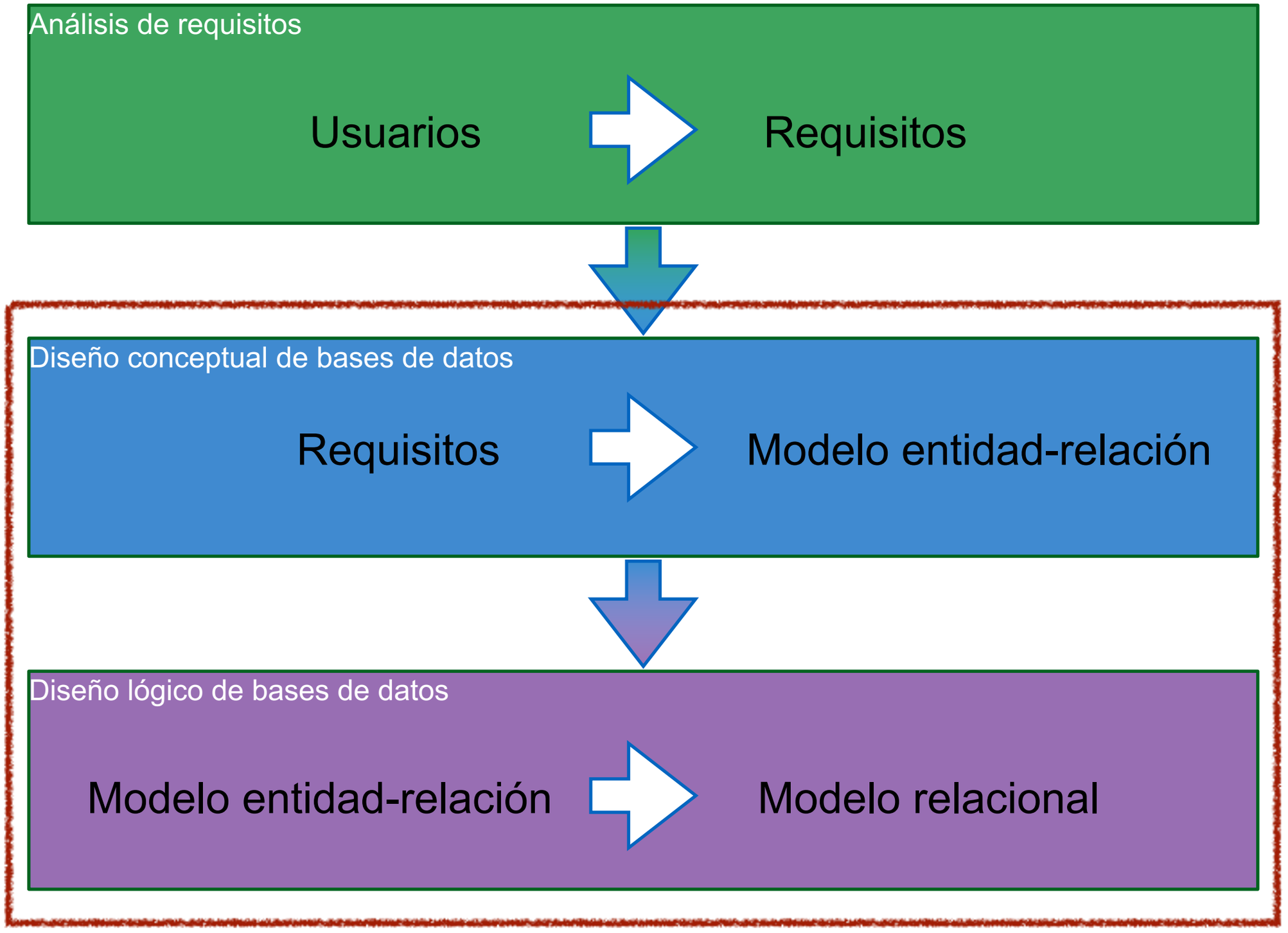
Nombre	Tipo	Grados	Ciudad-Origen	Cantidad	Precio-Unitario
Austral Lager	Lager	4.6	Punta Arenas	300	2000
Austral Yagan	Ale	5.0	Punta Arenas	600	**?**

Vinos(<u>Nombre</u>, <u>Tipo</u>, <u>Año</u>, Grados, Ciudad-Origen, Cantidad, Precio-Unitario)

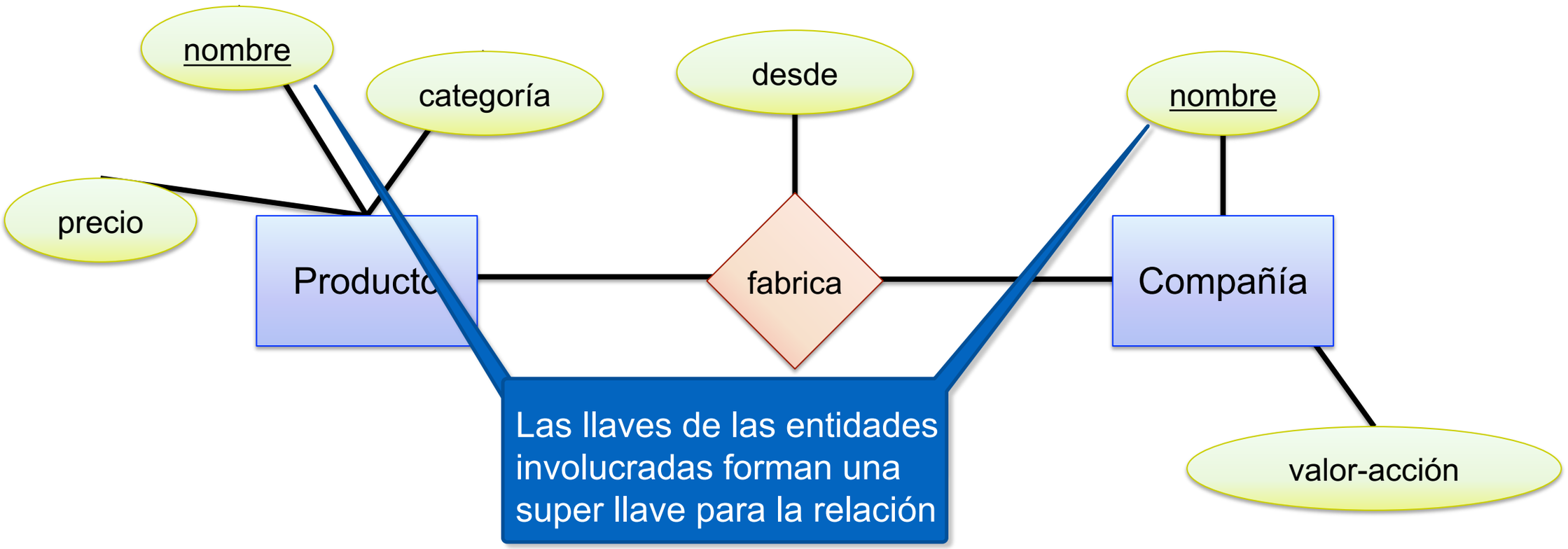
Nombre	Tipo	Año	Grados	Ciudad-Origen	Cantidad	Precio-Unitario
Tarapaca	Carmenere	2014	4.6	Punta Arenas	200	6000
Tarapaca	Merlot	2014	5.0	Punta Arenas	**?**	6500
Gato	Merlot	2016	14.0	Maule	150	**?**

Del Diagrama E/R al Modelo Relacional

Diseño de base de datos



M. E/R →M. Relacional

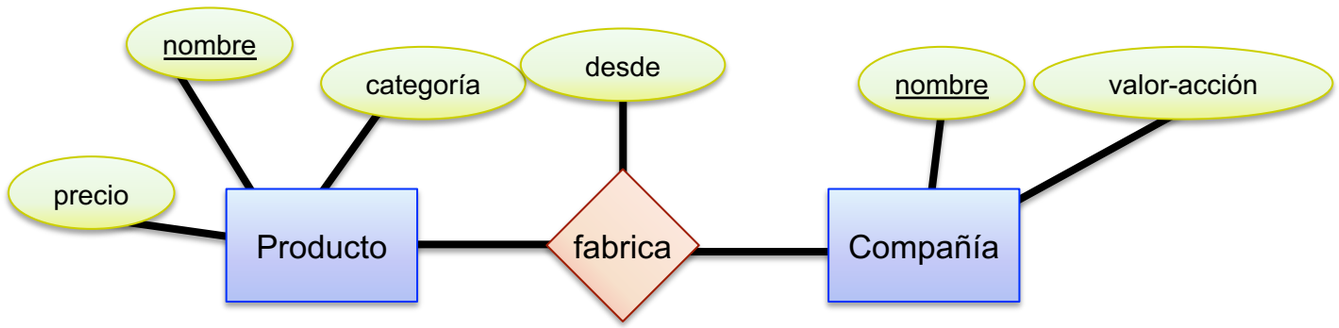


Producto(nombre: string, precio: int, categoría: string)

Compañía(nombre: string, valor-acción: int)

Fabrica(Producto.nombre: string, Compañia.nombre, desde: date)

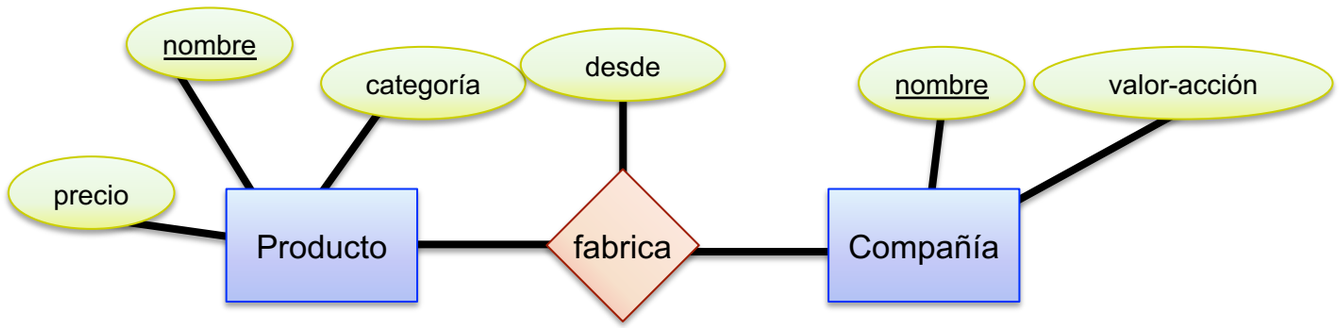
M. E/R →M. Relacional



Producto(nombre: string, precio: int, categoría: string)

```
CREATE TABLE producto(
    nombre varchar(30),
    precio int,
    categoria varchar(30),
    PRIMARY KEY (nombre)
)
```

M. E/R →M. Relacional

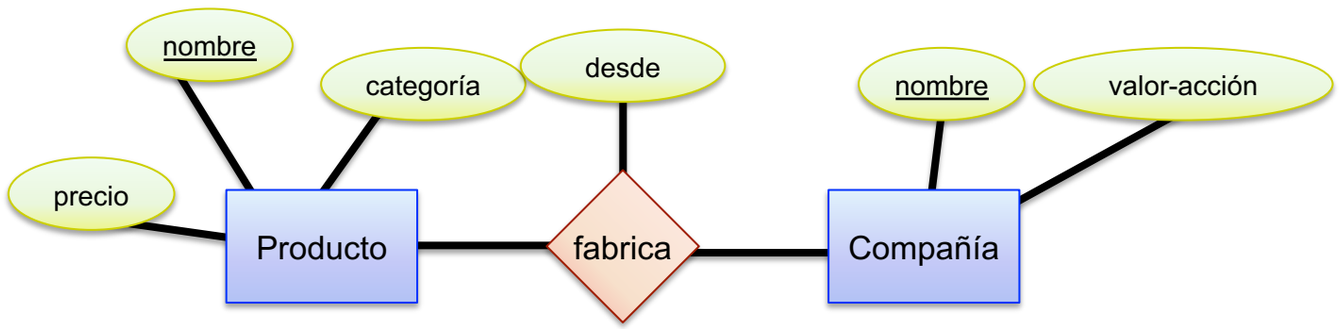


Compañía(nombre: string, valor-acción: int)

```
CREATE TABLE compania(
    nombre varchar(30),
    valor_accion int,
    PRIMARY KEY (nombre)
)
```

¿Como relacionamos la tabla compania con la tabla producto?

M. E/R →M. Relacional



Fabrica(Producto.nombre: string, Compañia.nombre, desde: date)

```
CREATE TABLE fabrica(
    p_nombre varchar(30),
    c_nombre varchar(30),
    desde date,
    PRIMARY KEY (p_nombre, c_nombre),
    FOREIGN KEY(p_nombre) REFERENCES producto(nombre),
    FOREIGN KEY(c_nombre) REFERENCES compania(nombre)
)
```

Llaves foráneas!

¡Usando llaves foraneas!

Paréntesis:

Llaves foráneas

Llaves Foráneas

¿Que es una llave foranea?

Cuando la referencia a la tabla es una llave:

$$R[A_1, \ldots, A_n] \subseteq S[B_1, \ldots, B_n]$$

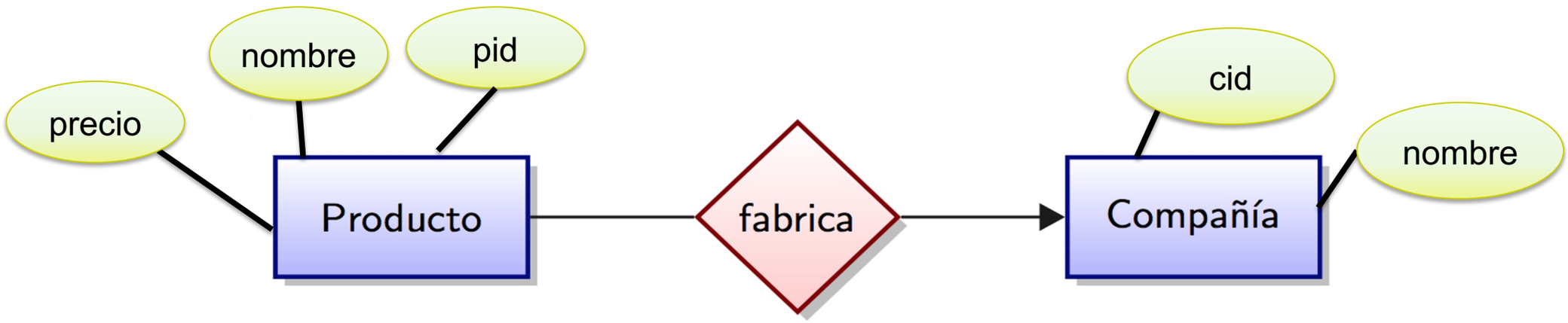
Y $B_1,\ldots,B_n$ son llave para ***S***

La relación R contiene la llave de la relación S

Pero hay que tener cuidado, por ejemplo:

Llaves foráneas

¿Qué pasa aquí?



producto

pid	nombre	precio
1	SonyXZ89	$100.000
2	SonyXperia1	$1.000.000
3	Huawei23	$1000

compania

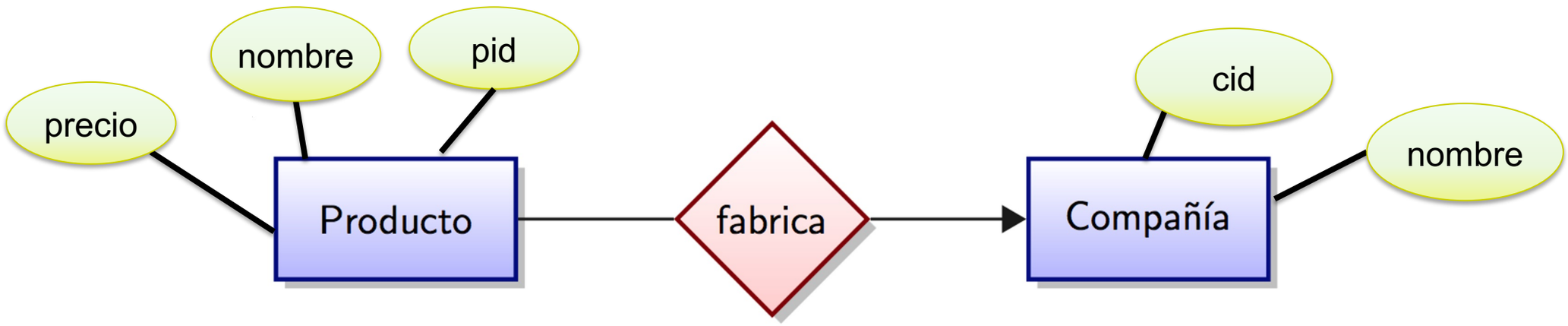
cid	nombre
C1	Sony
C2	Apple

fabrica

pid	cid
1	C1
2	C1
3	C8
89	C2

Llaves foráneas

¿Qué pasa aquí?



producto

pid	nombre	precio
1	SonyXZ89	$100.000
2	SonyXperia1	$1.000.000
3	Huawei23	$1000

compania

cid	nombre
C1	Sony
C2	Apple

fabrica

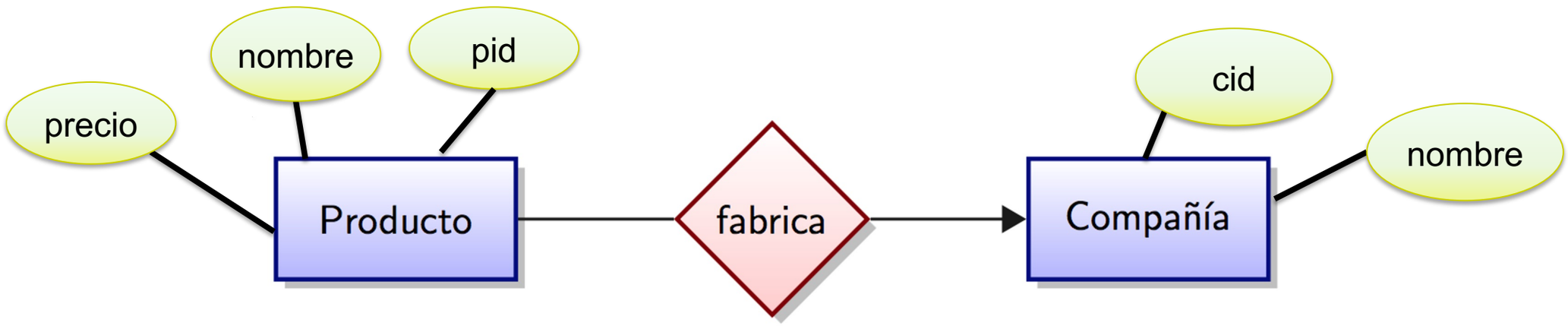
pid	cid
1	C1
2	C1
3	C8
89	C2

???

C8 No existe en la tabla compania!

Llaves foráneas

¿Qué pasa aquí?



producto

pid	nombre	precio
1	SonyXZ89	$100.000
2	SonyXperia1	$1.000.000
3	Huawei23	$1000

compania

cid	nombre
C1	Sony
C2	Apple

???

89 No existe en la tabla producto!

fabrica

pid	cid
1	C1
2	C1
3	C8
89	C2

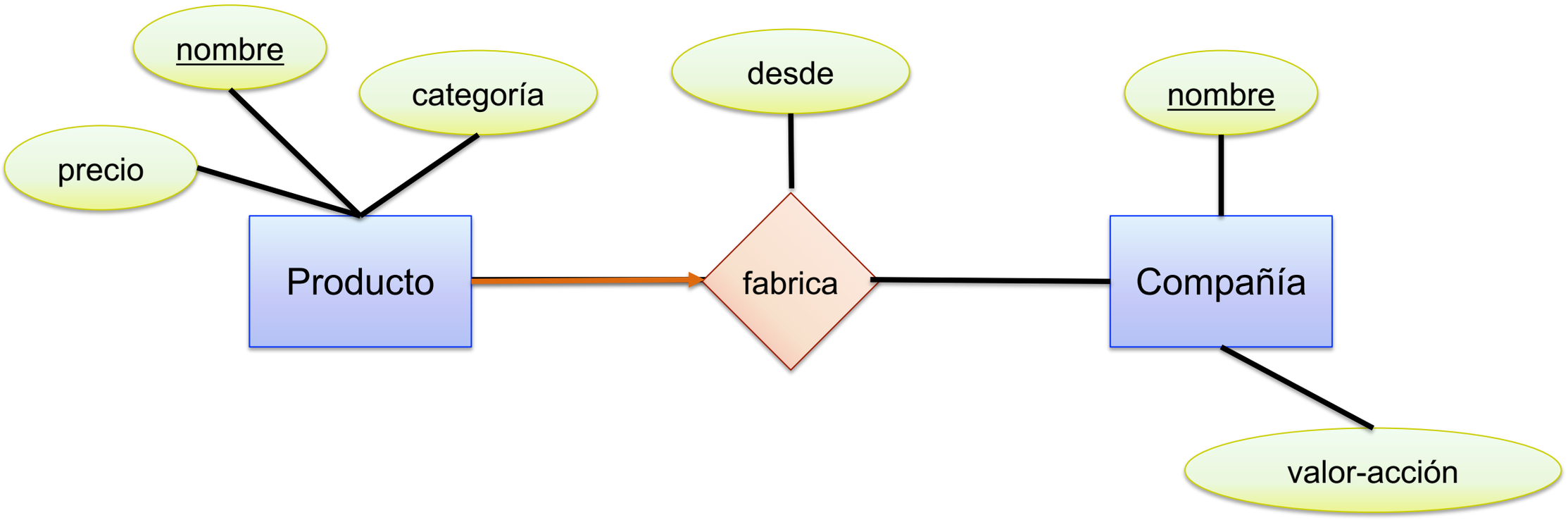
???

C8 No existe en la tabla compania!

¿Cómo representar E/R con llaves foraneas?

M. E/R →M. Relacional

¿Que hacemos en este caso?



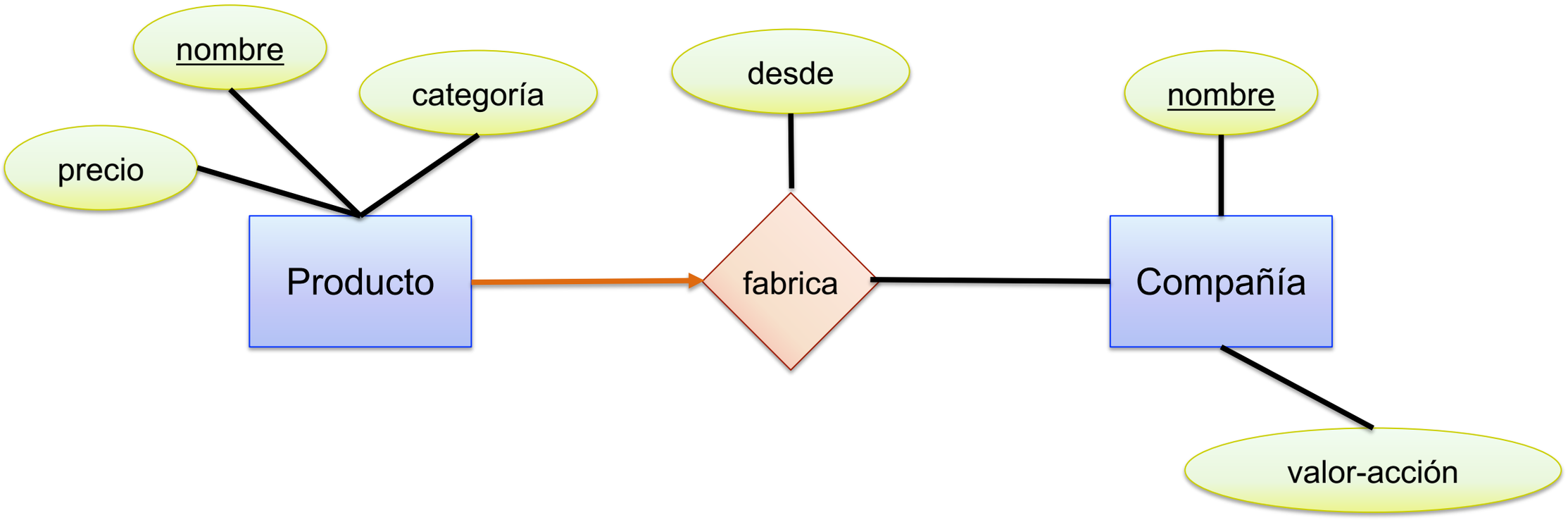
Producto(nombre: string, precio: int, categoría: string)

Compañía(nombre: string, valor-acción: int)

Fabrica(Producto.nombre: string, Compañia.nombre, desde: date)

M. E/R →M. Relacional

¿Que hacemos en este caso?



Producto(nombre: string, precio: int, categoría: string)

Compañía(nombre: string, valor-acción: int)

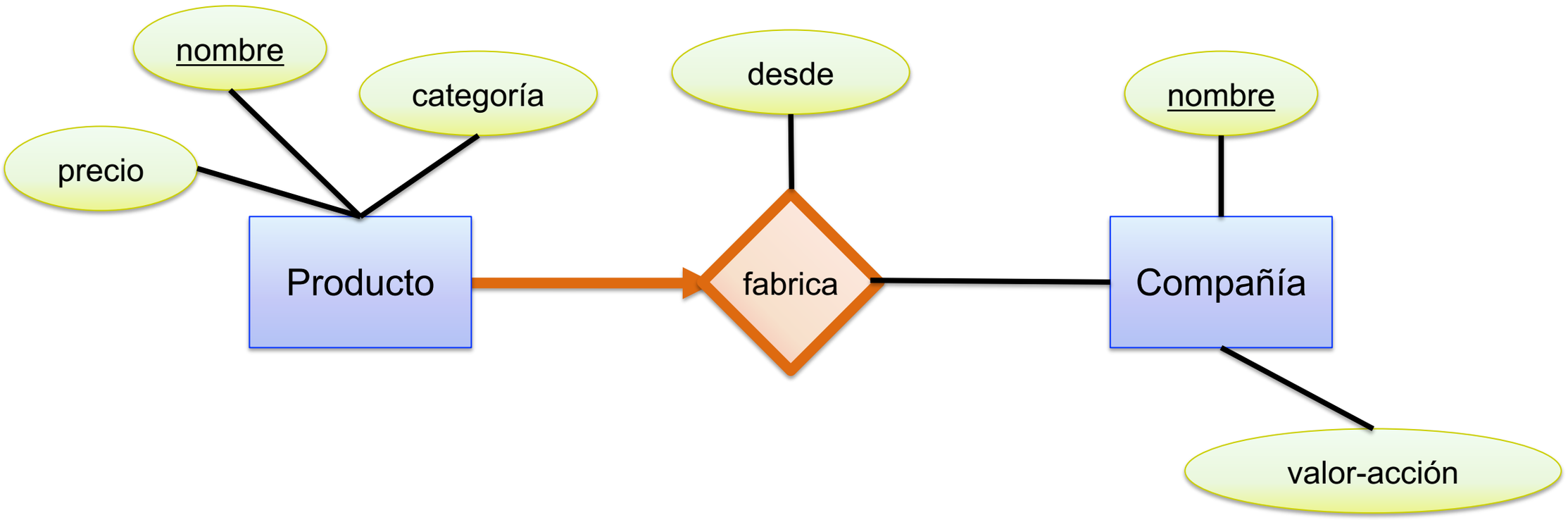
Fabrica(Producto.nombre: string, Compañia.nombre, desde: date)

Producto.nombre forma una llave candidata

No se necesita que Compañia.nombre sea llave

M. E/R →M. Relacional

¿Y ahora?



Producto(nombre: string, precio: int, categoría: string,
Compañía.nombre: string, desde: date)

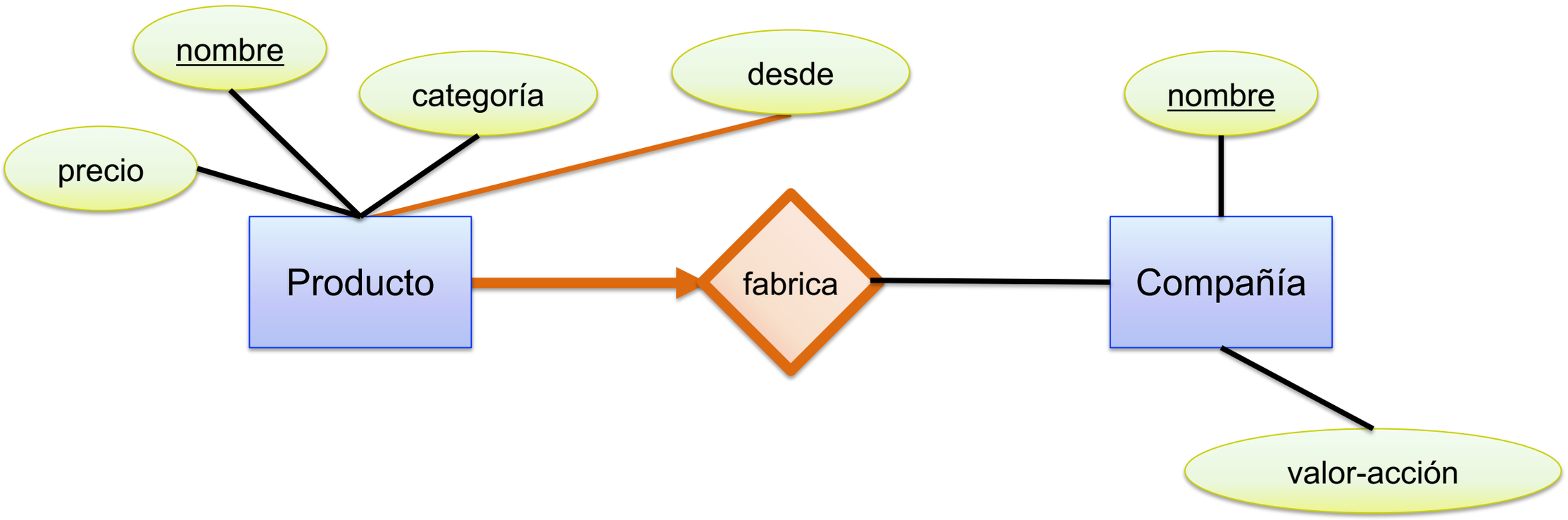
Compañía(nombre: string, valor-acción: int)

Sólo necesitamos una llave foránea en Producto.

Agregamos también el atributo de la relación.

M. E/R →M. Relacional

Un mejor diagrama…



Producto(nombre: string, precio: int, categoría: string,
Compañía.nombre: string, desde: date)

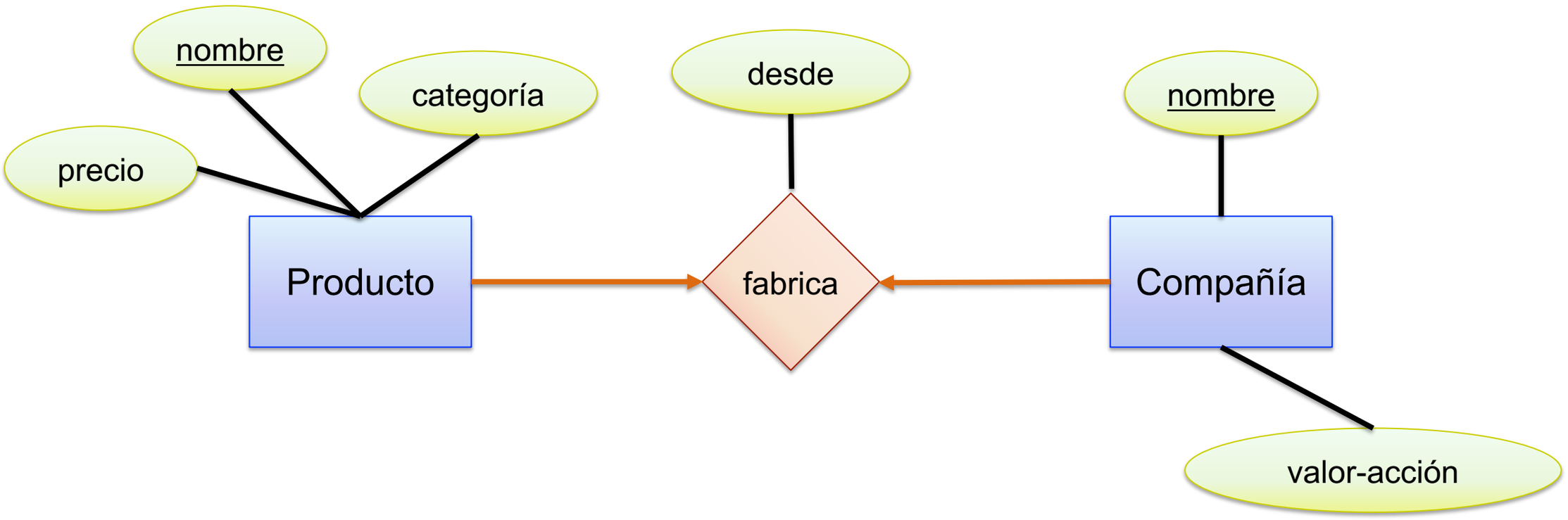
Compañía(nombre: string, valor-acción: int)

Sólo necesitamos una llave foránea en Producto.

Agregamos también el atributo de la relación.

M. E/R →M. Relacional

¿Y en este caso?



Producto(nombre: string, precio: int, categoría: string)

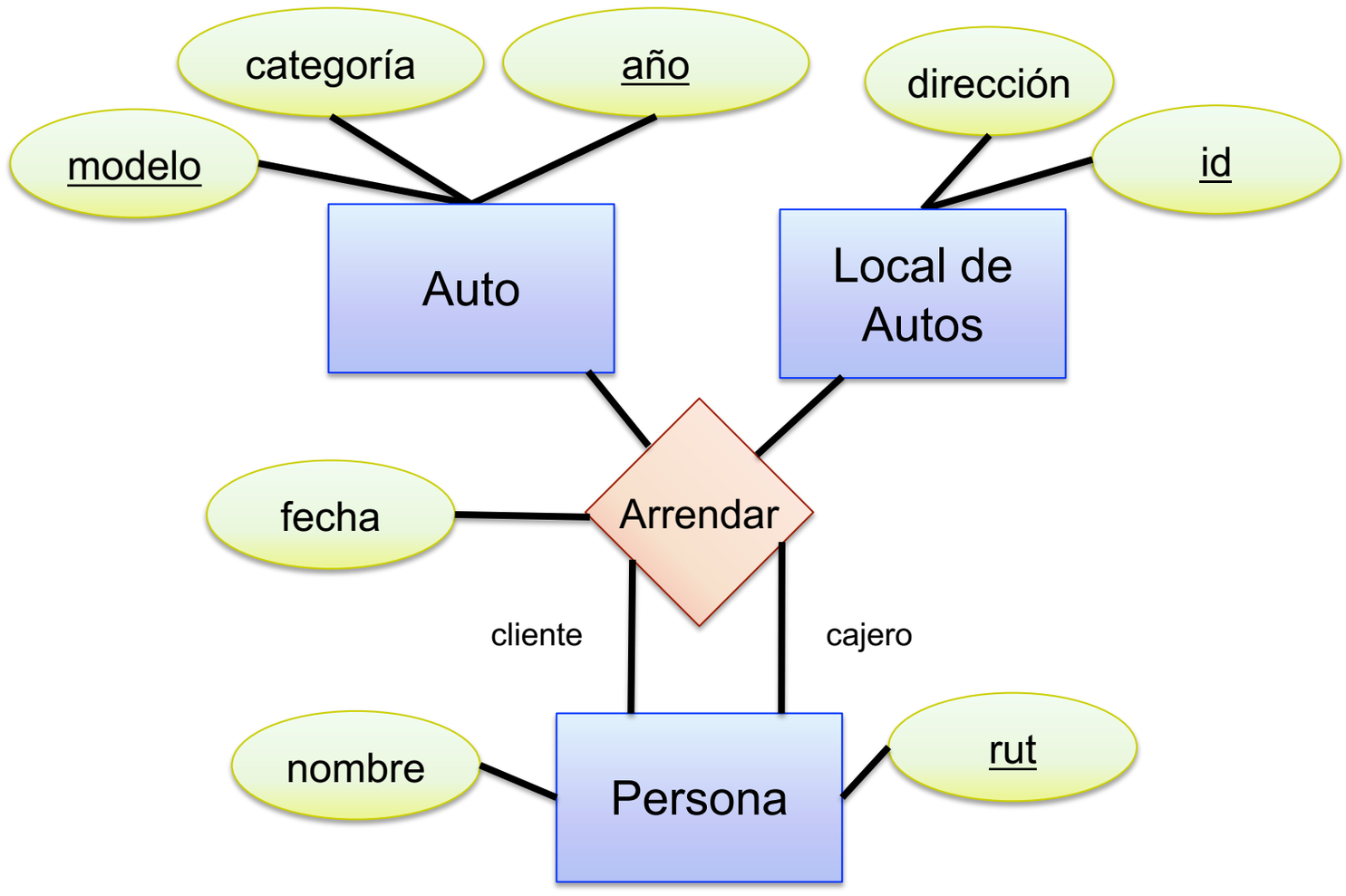
Compañía(nombre: string, valor-acción: int)

Fabrica(Producto.nombre: string, Compañia.nombre, desde: date)

Podemos hacer llave a Producto.nombre o a Compañía.nombre

M. E/R →M. Relacional

¿Que hacemos en este caso?



Auto(modelo: string, año: int, categoría: string)

Local de Autos(id: int, dirección: string)

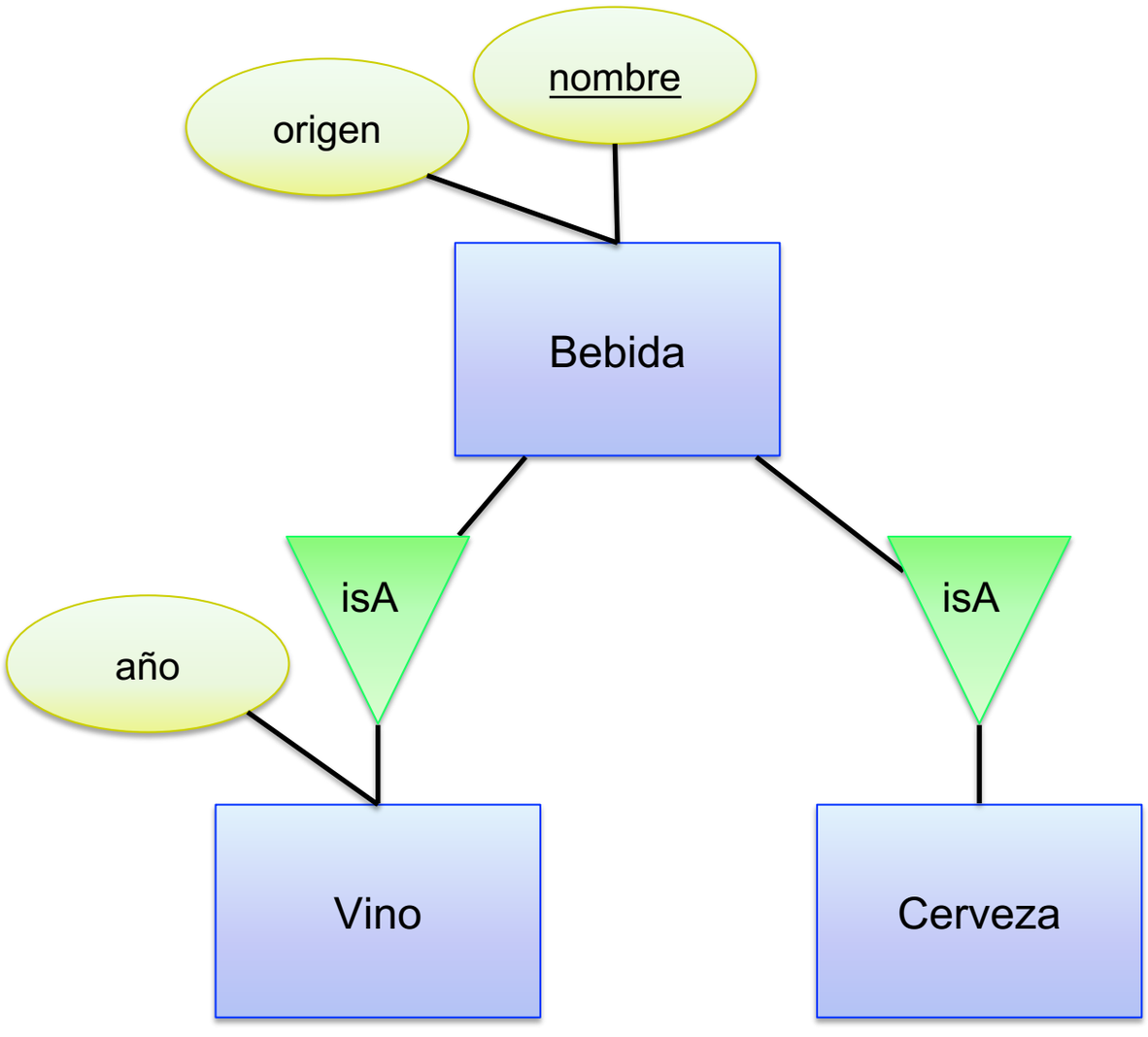
Persona(rut: string, nombre: string)

Arriendo(A.modelo: string, A.año: int, Pr.rut-cl: string, Pr.rut-ca: string, L.id: int, fecha: date)

¿Cómo representar un M. E/R en un M. Relacional con jerarquia de clases?

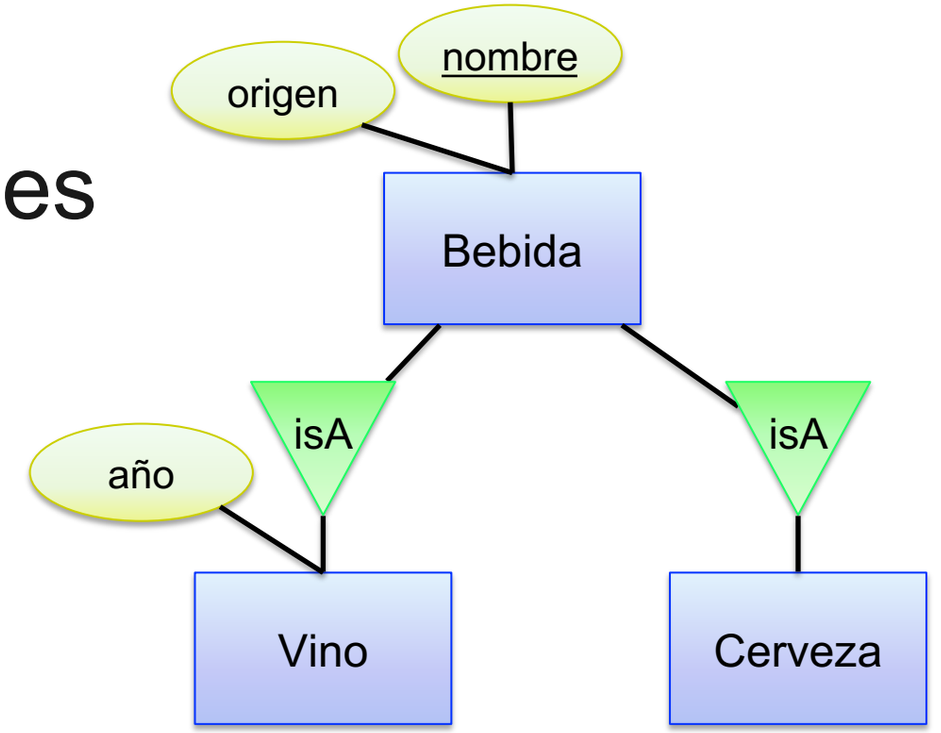
Ejemplo

Jerarquía de clases



M. E/R →M. Relacional

Jerarquía de clases



Opción 1: Tablas solo para las subclases

Vino(nombre: string, origen: string, año: string)
Cerveza(nombre: string, origen: string)

Opción 2: Tabla para la superclase

Bebida(nombre: string, origen: string)
Vino(nombre: string, año: string)
Cerveza(nombre: string)

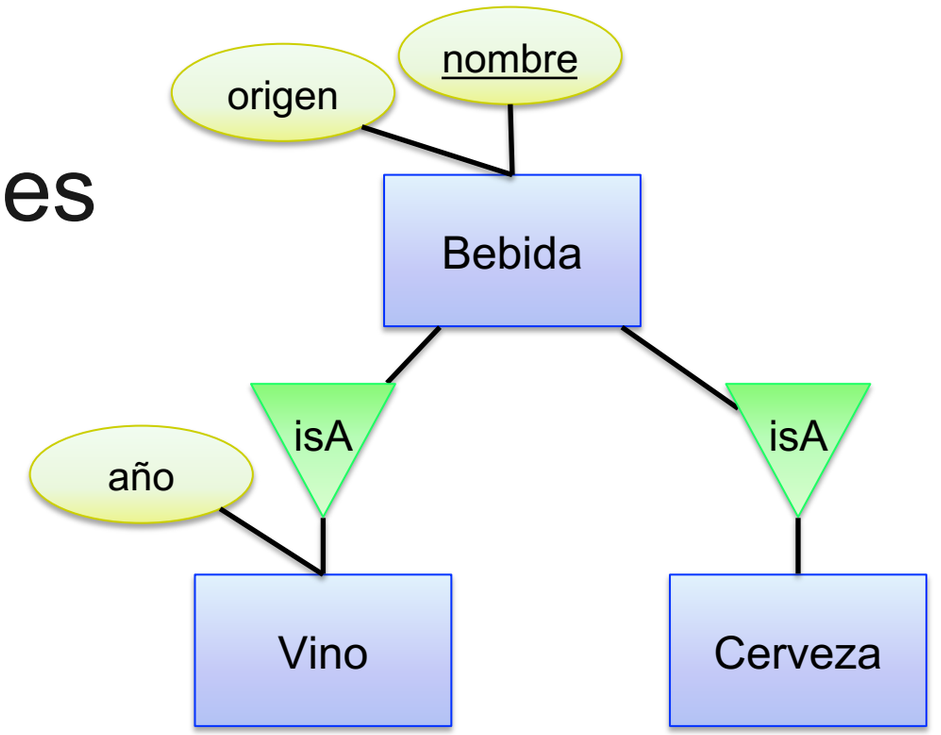
Se requieren joins para acceder a todos los datos

¿Cuál es mejor?

Si hay mucho solapamiento: opción 2.
De lo contrario tendríamos mucha repetición de datos.

M. E/R →M. Relacional

Jerarquía de clases



Opción 1: Tablas solo para las subclases

Vino(nombre: string, origen: string, año: string)
Cerveza(nombre: string, origen: string)

Opción 2: Tabla para la superclase

Bebida(nombre: string, origen: string)
Vino(nombre: string, año: string)
Cerveza(nombre: string)

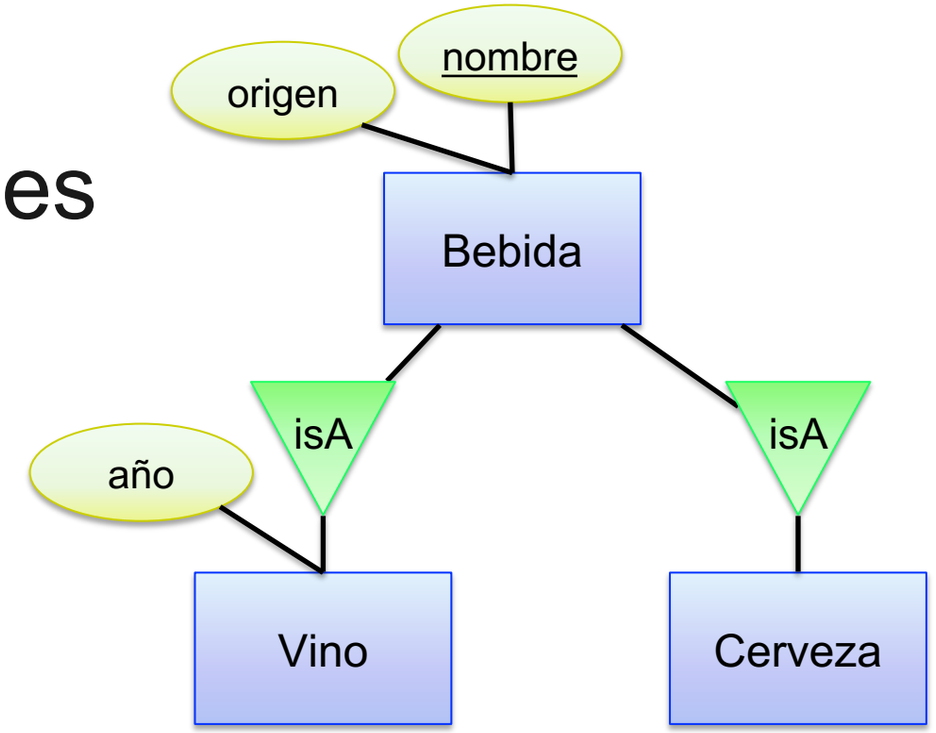
¿Cuál es mejor?

Se requieren joins para acceder a todos los datos

Si no hay cobertura: opción 2.
No hay otra opción o no podríamos guardar el whisky :(

M. E/R →M. Relacional

Jerarquía de clases



Opción 1: Tablas solo para las subclases

Vino(nombre: string, origen: string, año: string)
Cerveza(nombre: string, origen: string)

Opción 2: Tabla para la superclase

Bebida(nombre: string, origen: string)
Vino(nombre: string, año: string)
Cerveza(nombre: string)

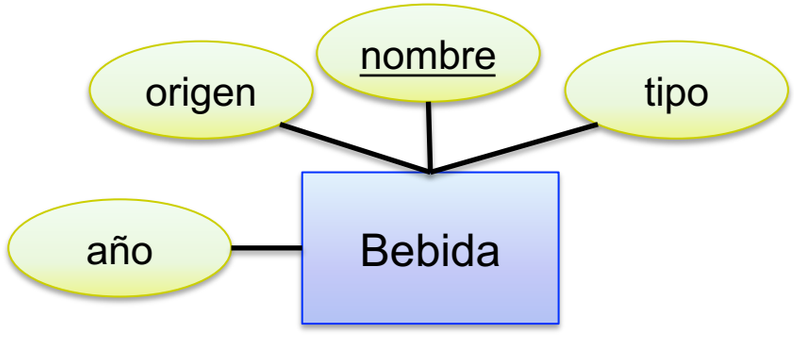
Se requieren joins para acceder a todos los datos

¿Cuál es mejor?

Si hay muchas consultas por **nombre**: opción 2.
Con la opción 1 tendríamos que consultar dos tablas.

M. E/R →M. Relacional

Jerarquía de clases



Opción 3: Quitar la jerarquía

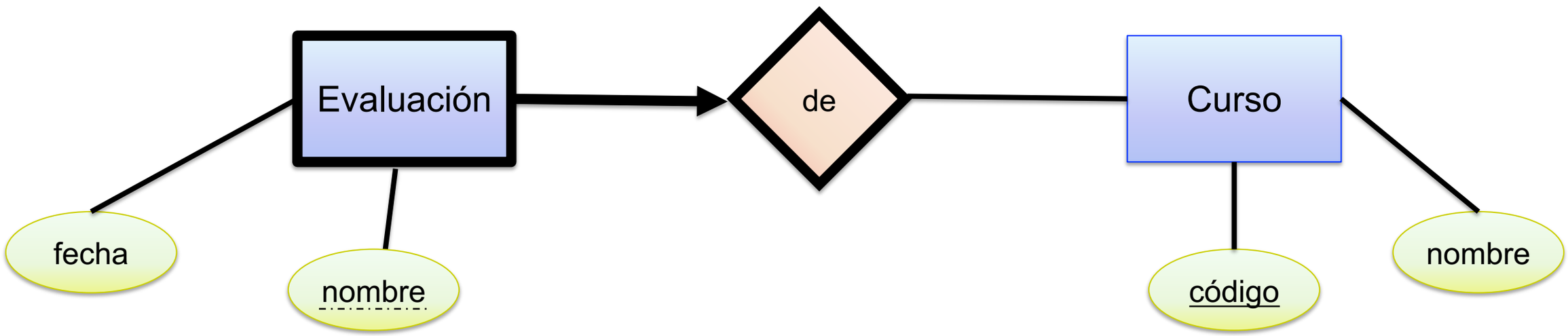
Bebida(nombre: string, origen: string, año: string, tipo: string)

- Muchas repeticiones de la columna tipo.
- Puede que no se conozca el tipo (nulls).
- Pero más sencillo (y comprimible)

¿Cómo representar un M. E/R en un M. Relacional con entidades debiles?

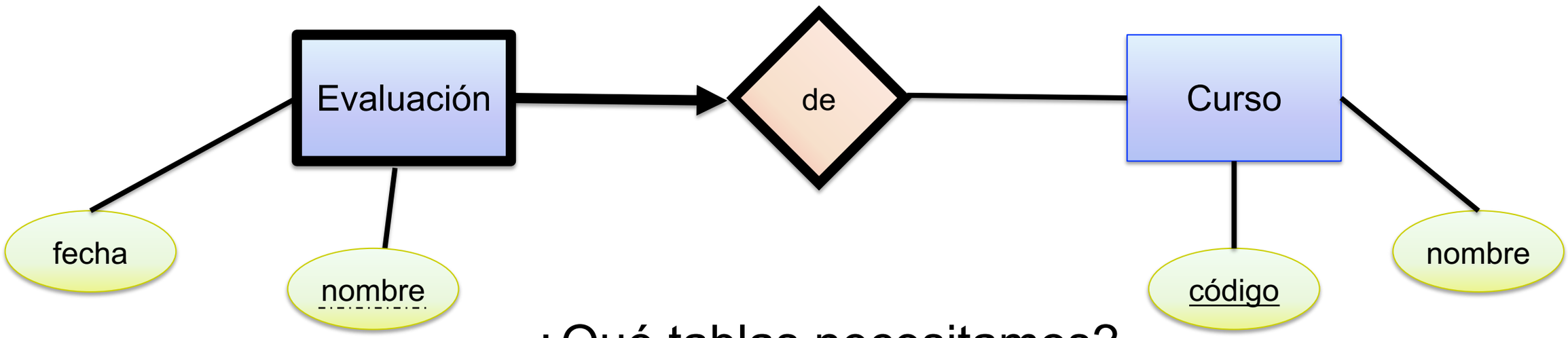
Ejemplo

Entidades débiles



M. E/R →M. Relacional

Entidades débiles



¿Qué tablas necesitamos?

Curso(codigo: string, nombre: string)

Evaluación(nombre: string, C.código: string, fecha: date)

De(E.nombre: string, C.código: string) ✘

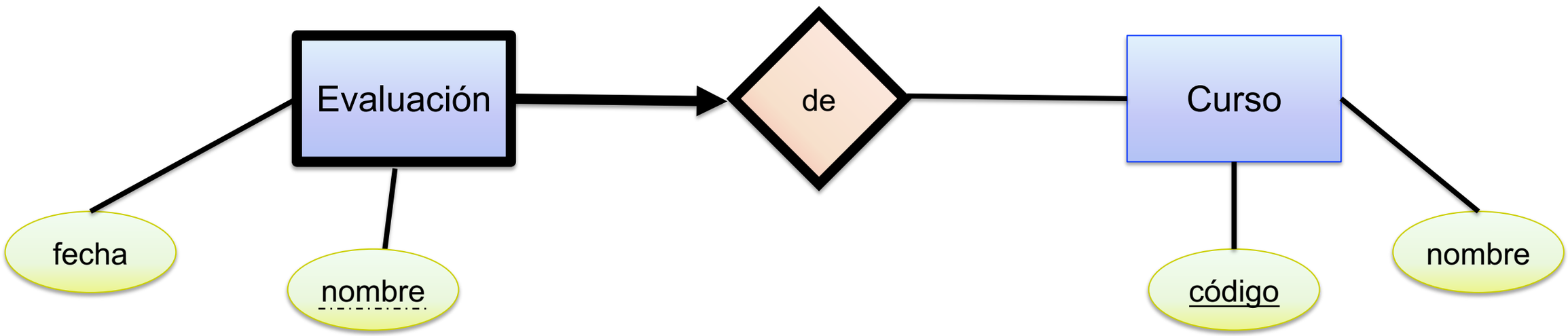
¿Está bien esto?

La tabla De es redundante (1-a-algo)

(y mal nombre para una tabla)

M. E/R →M. Relacional

Entidades débiles



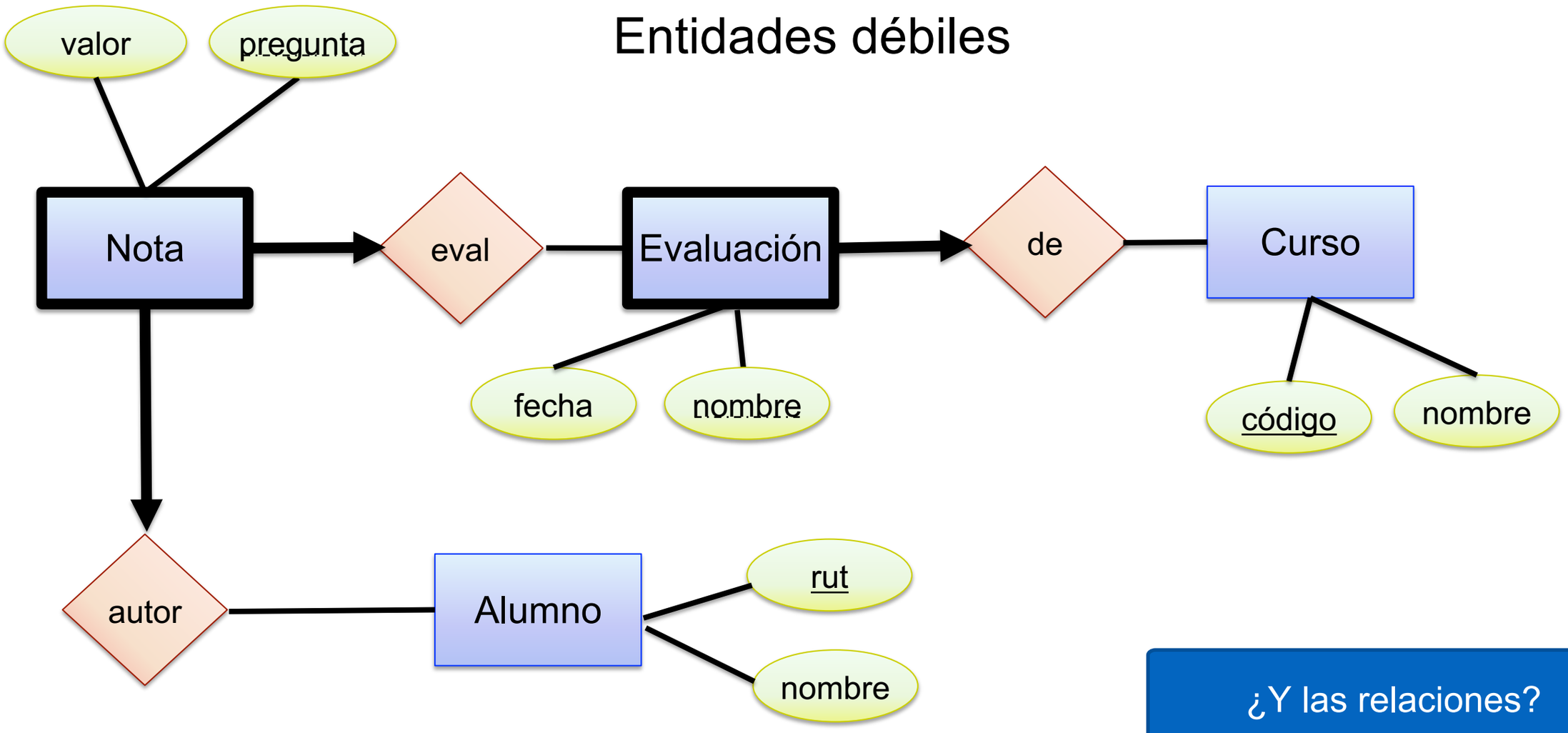
Curso(<u>codigo</u>: string, nombre: string)
Evaluación(<u>nombre</u>: string, C.<u>código</u>: string, fecha: date)

¡Ahora sí!

```
CREATE TABLE evaluacion(
    nombre varchar(30) NOT NULL,
    codigo varchar(30) NOT NULL,
    fecha date,
    PRIMARY KEY (nombre, codigo)
    FOREIGN KEY(codigo) REFERENCES curso(codigo) ON DELETE CASCADE
)
```

M. E/R →M. Relacional

Entidades débiles



¿Y las relaciones?

Curso(codigo: string, nombre: string)

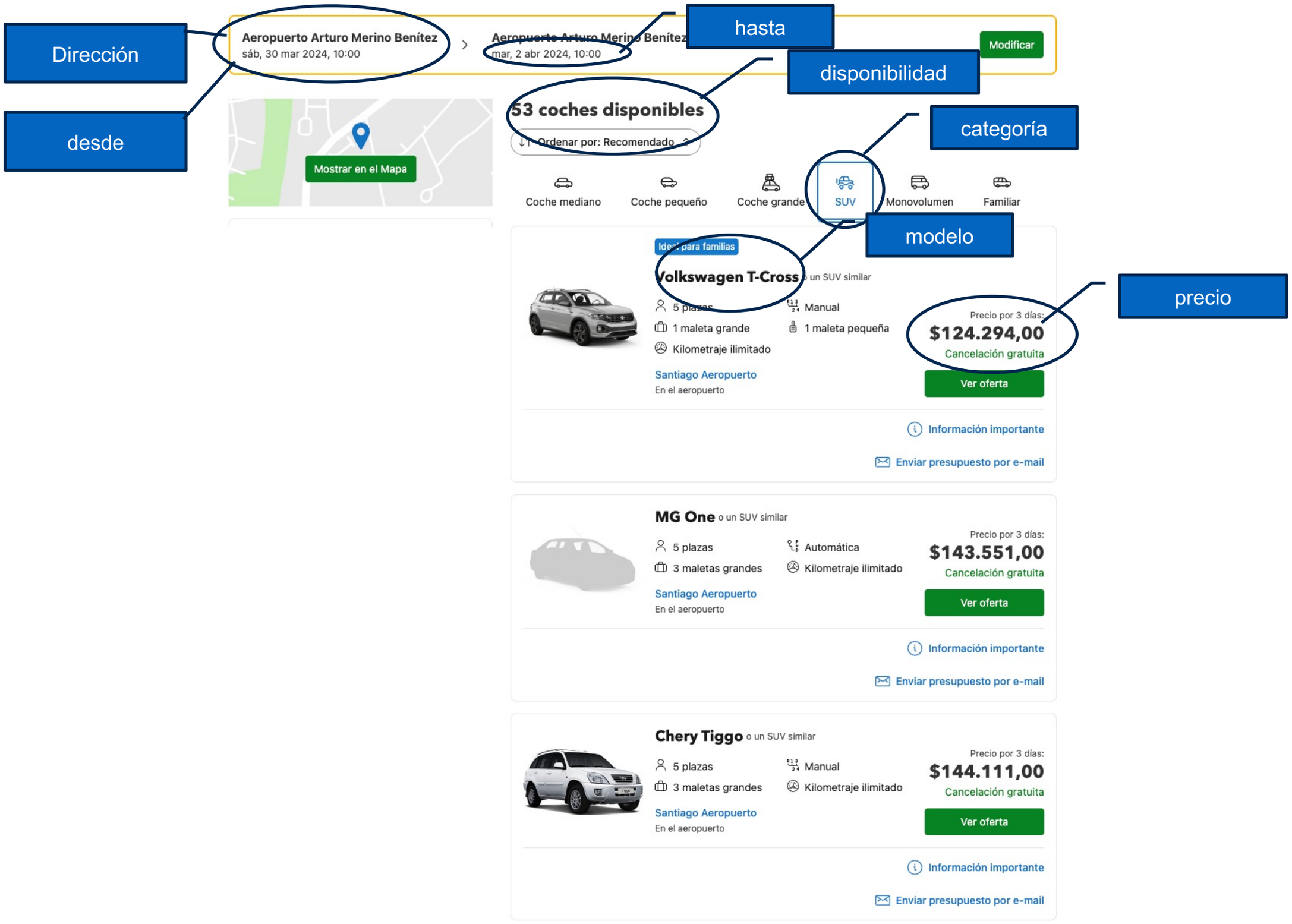
Evaluación(nombre: string, C.código: string, fecha: date)

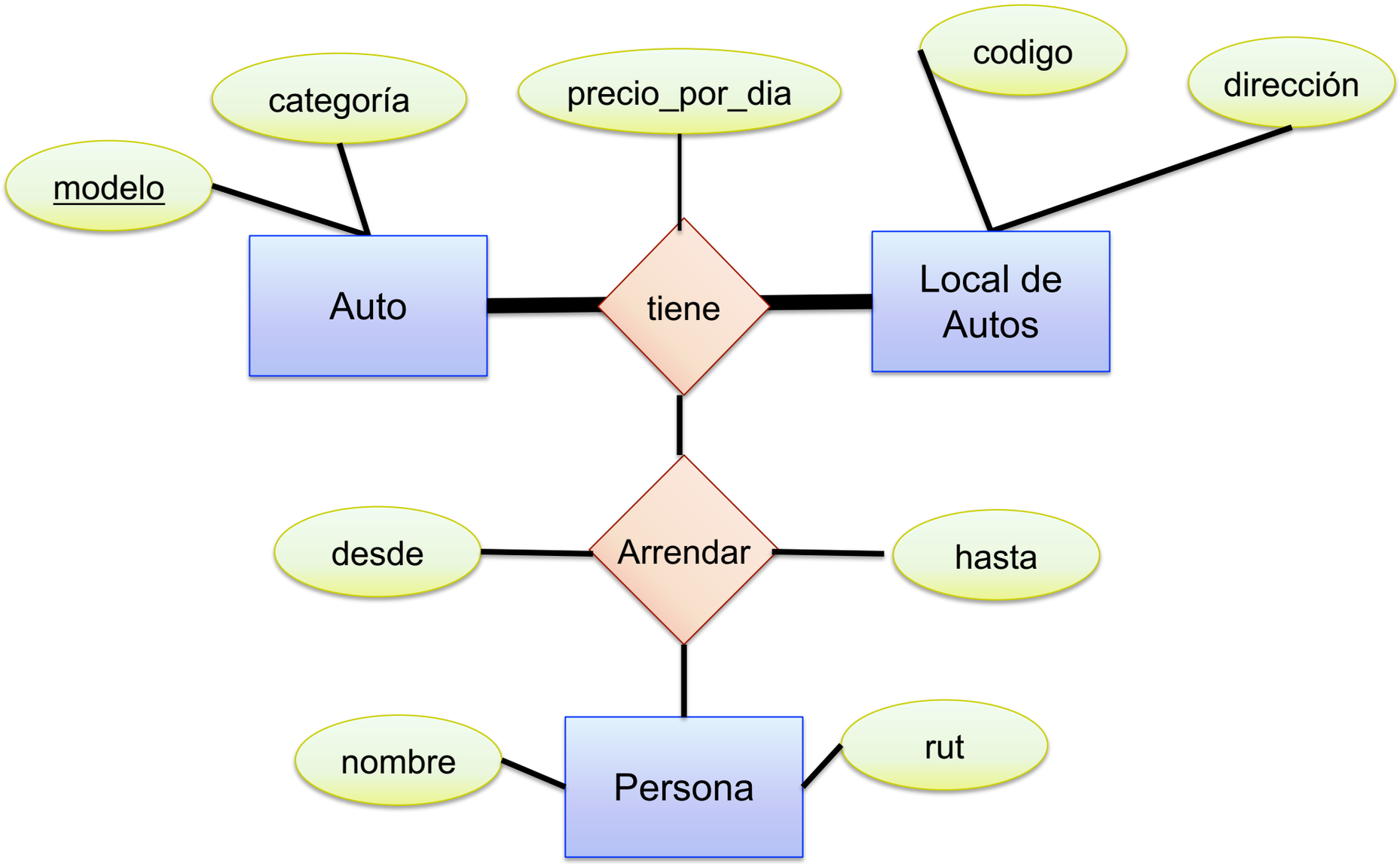
Nota(pregunta: int, E.nombre: string, C.código: string, A.rut: string, valor: float)

Alumno(rut: string, nombre: string)

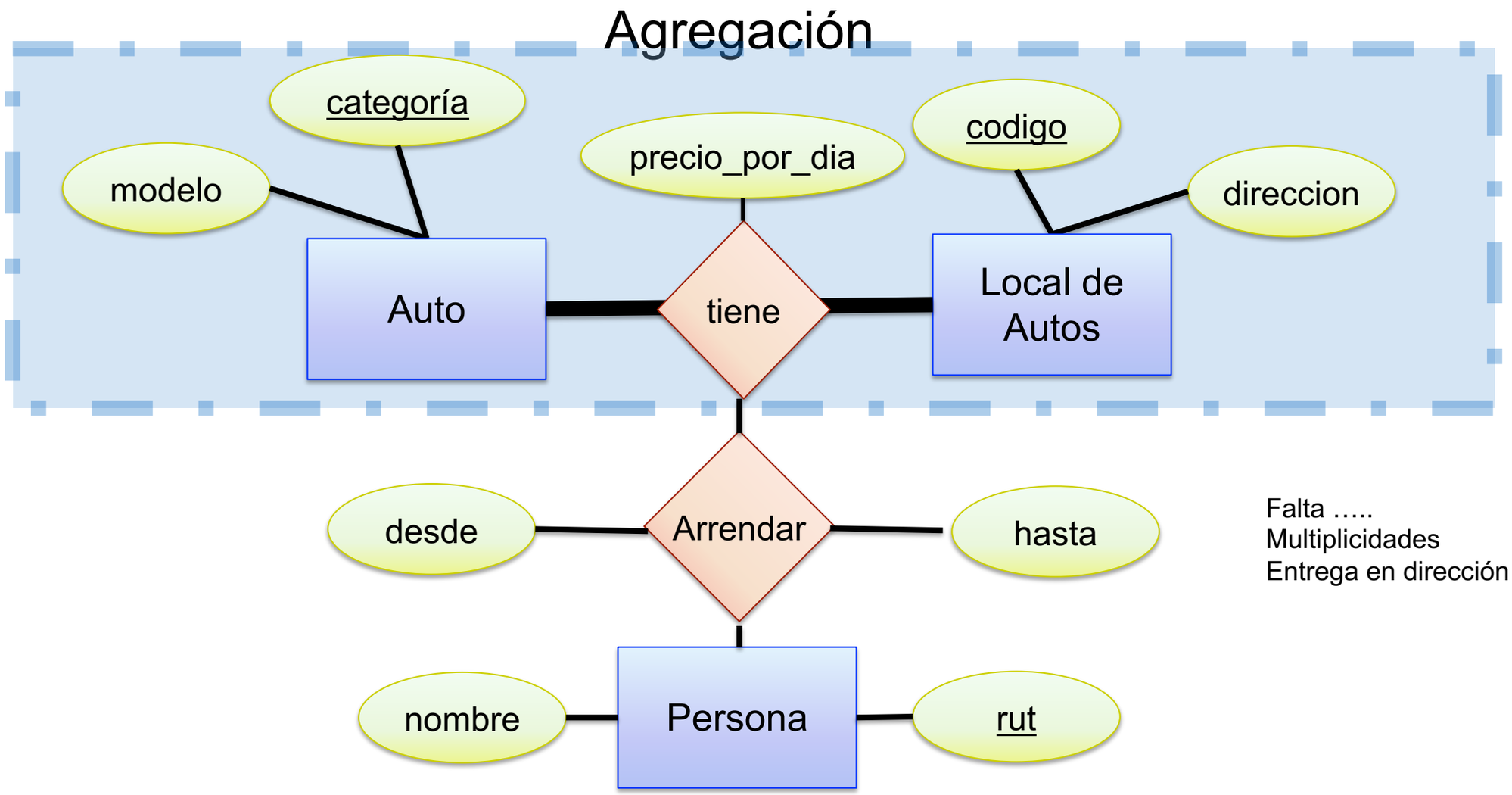
¿Cómo representar un M. E/R en un M. Relacional con agregacion?

Ejemplo: Arrendar un auto




categoría
precio_por_dia
codigo
dirección
modelo
Auto
tiene
Local de Autos
desde
Arrendar
hasta
nombre
Persona
rut

M. E/R →M. Relacional



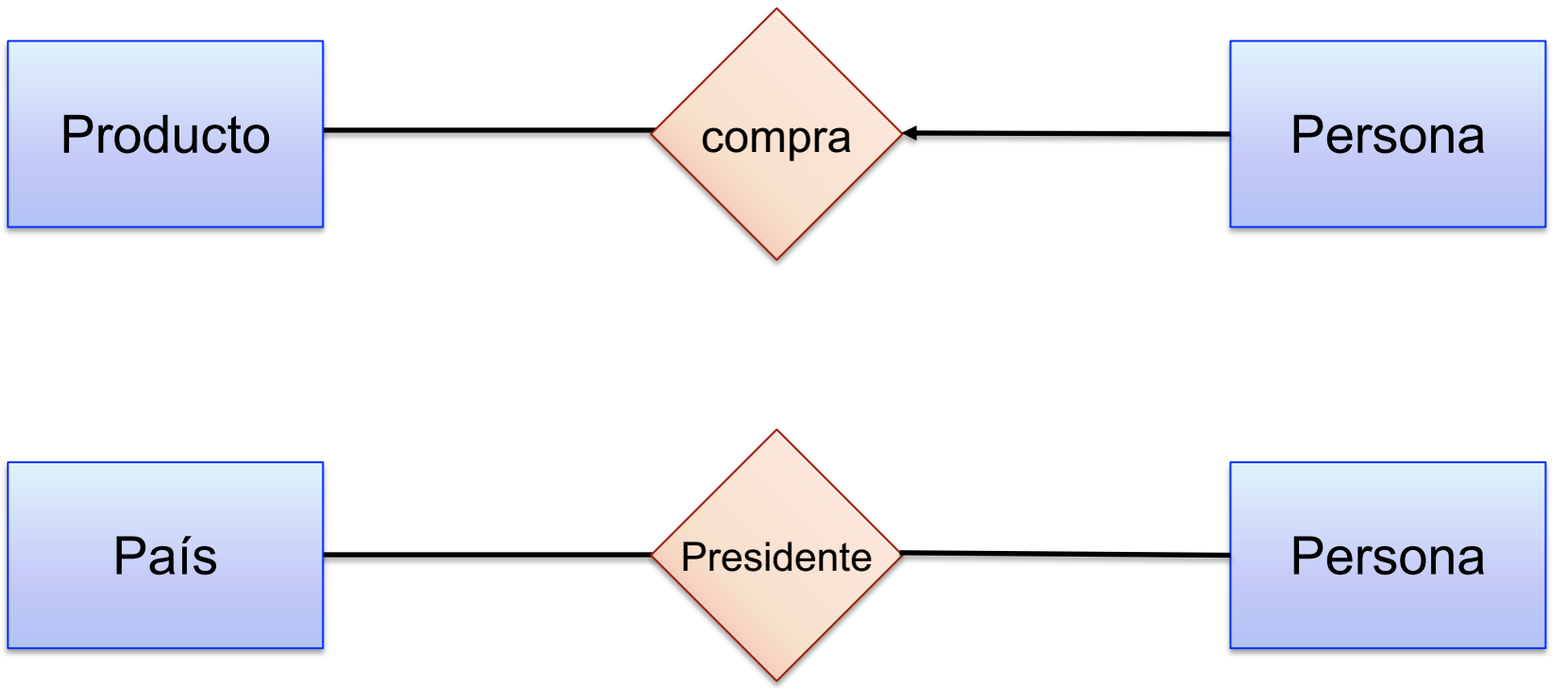
Falta …..
Multiplicidades
Entrega en dirección diferente

Auto(categoria: string, modelo: string) as A
LocalDeAutos(codigo: string, direccion: string) as L
Tiene(A.categoría: string, L.código: string, precio_por_dia: int) as T
Persona(rut: string, nombre: string) as P
Arrendar(T.a.categoría: string, T.l.código: string, P.rut, desde: date, hasta: date)

Principios básicos del diseño

Principios básicos del diseño

1. Fidelidad al problema

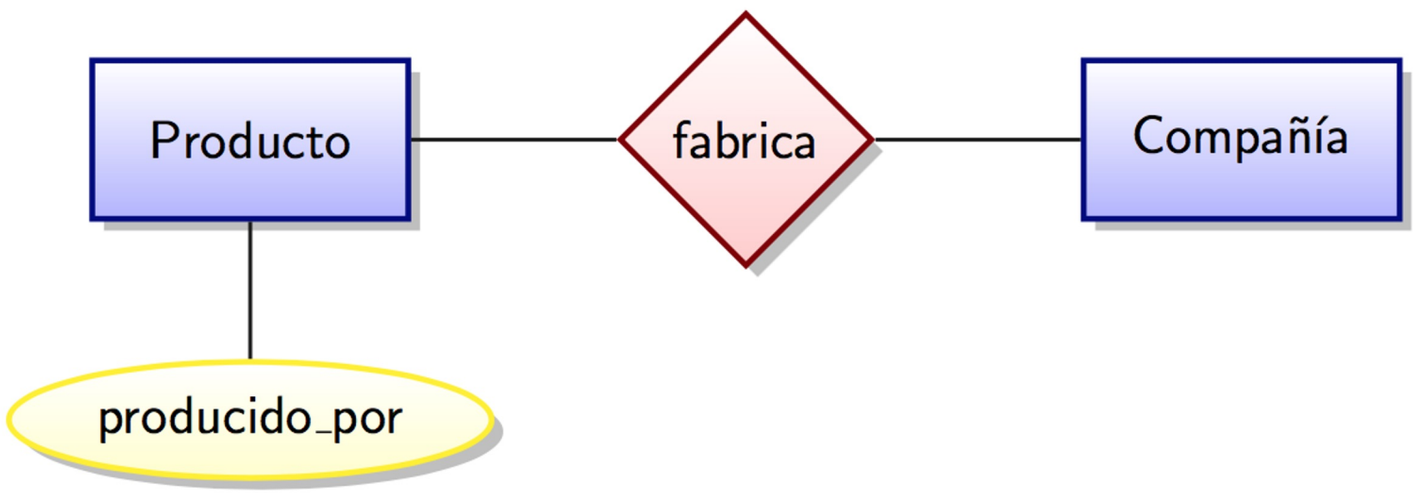


Las relaciones no tienen nada que ver entre si

Principios básicos del diseño

2. Evitar redundancia

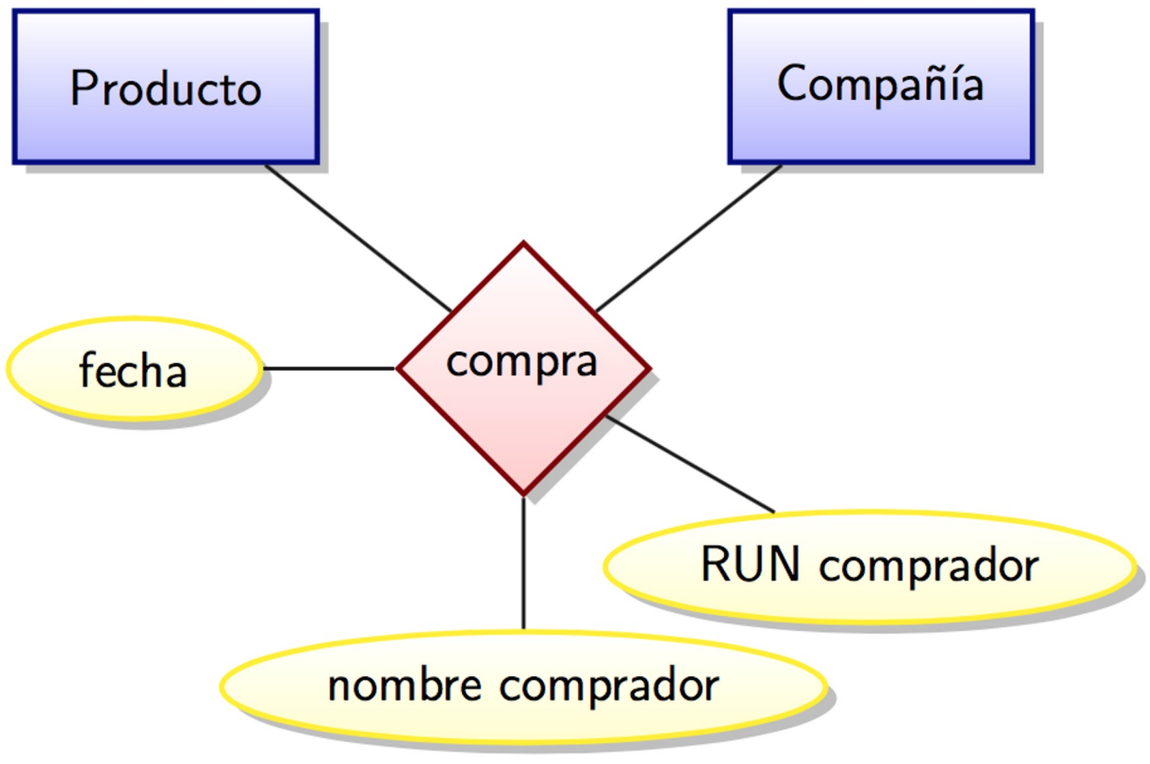
¿Qué está mal?



El atributo producido_por es redundante y puede generar **anomalías**

Principios básicos del diseño

3. Elegir entidades y relaciones correctamente

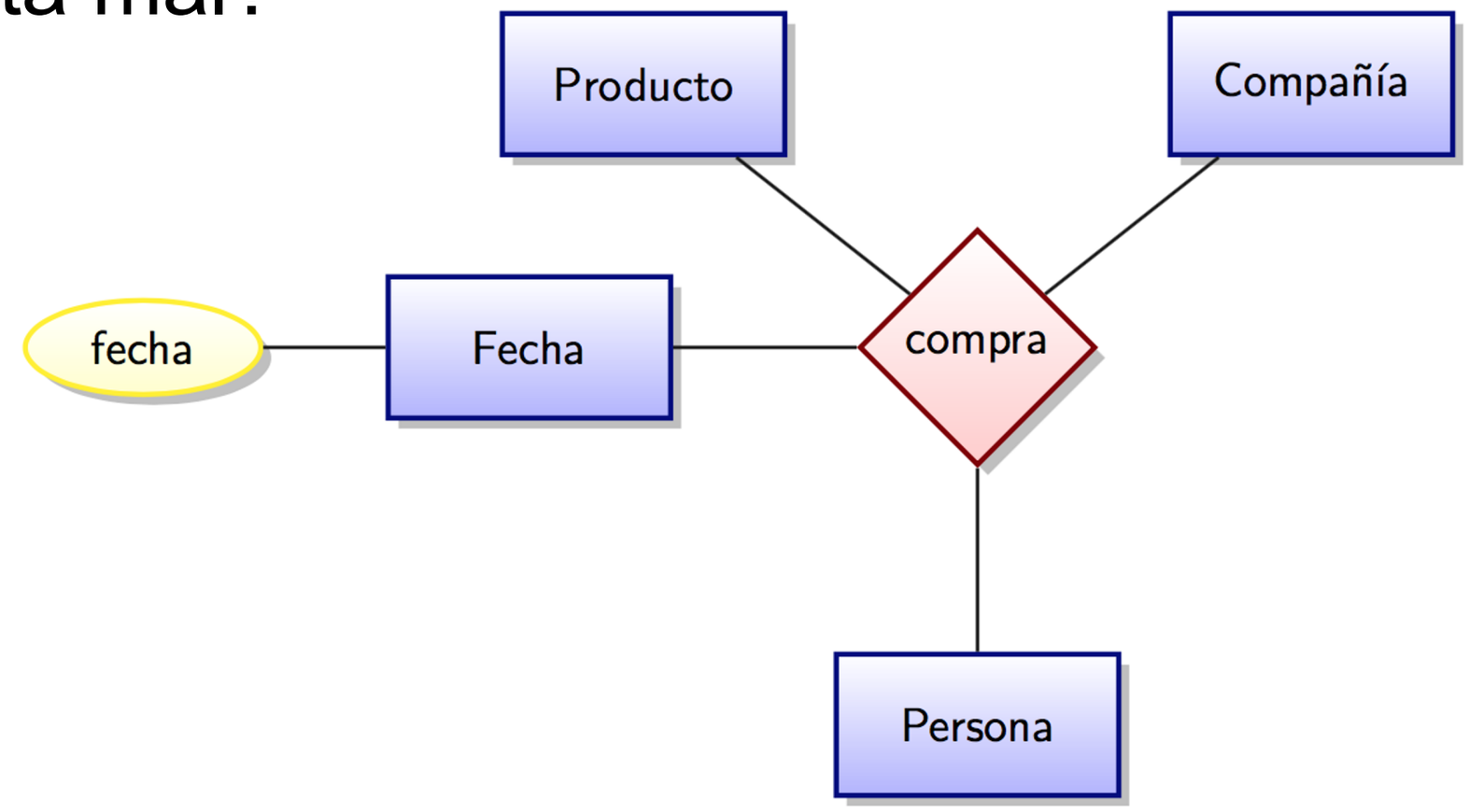


Seria mejor crear una entidad persona la cual relacione el producto a traves de la compra

Principios básicos del diseño

4. No complicar más de lo necesario

¿Qué está mal?



Fecha puede ser un atributo de compra

Principios básicos del diseño

5. Buena elección de llave primaria.

Al momento de diseñar siempre queremos identificar todas los atributos de las entidades que son candidatos a ser llave de la tabla, a estos les llamamos *natural key*, porque son columnas que naturalmente tienen el comportamiento de una llave. Por ejemplo de la siguiente tabla:

Usuario(email, rut, username, nombre, tipo, fecha_de_inscripcion)

rut, email y username son posibles *natural keys*.

...pero en la práctica el 99% de las veces es mejor usar una columna inventada, sin significado que sea autogenerada por el RDBMS. A esto le llamamos *surrogate key*.

Principios básicos del diseño

Elección de llave primaria.

Bueno en realidad es algo medio opinionado...

Surrogate vs. natural/business keys [closed]

Asked 12 years, 7 months ago Active 2 months ago Viewed 71k times

Closed. This question is opinion-based. It is not currently accepting answers.

Principios básicos del diseño

Elección de llave primaria.

Bueno en realidad es algo medio opinionado...

Pero en la práctica los frameworks de desarrollo web modernos esperan una *surrogate key* llamada *id* como llave primaria e incluso la generan por defecto.

La tabla anterior deberíamos generarla así:

```
CREATE TABLE usuario(
    id SERIAL,
    email nombre varchar(30) UNIQUE NOT NULL,
    RUT varchar(30) UNIQUE NOT NULL,
    fecha date,
    PRIMARY KEY (id)
)
```

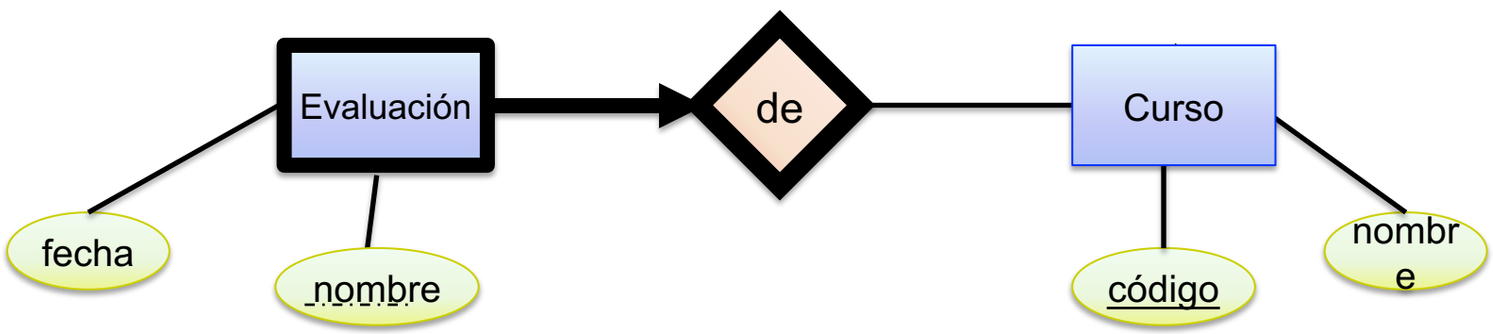
Restricciones de integridad

Restricciones de integridad

Son restricciones formales que imponemos a un esquema que todas sus instancias deben satisfacer. Algunas son:

- **De valores nulos:** El valor puede o no ser nulo.
- **Unicidad**: Dado un atributo, no pueden haber dos tuplas con el mismo valor.
- **De llave**: el valor es único y no puede ser null.
- **De referencia**: si se trabaja en una compañía, esta debe existir (Llaves foráneas).
- **De dominio**: la edad de las personas debe estar entre 0 y 150 años.

M. E/R →M. Relacional

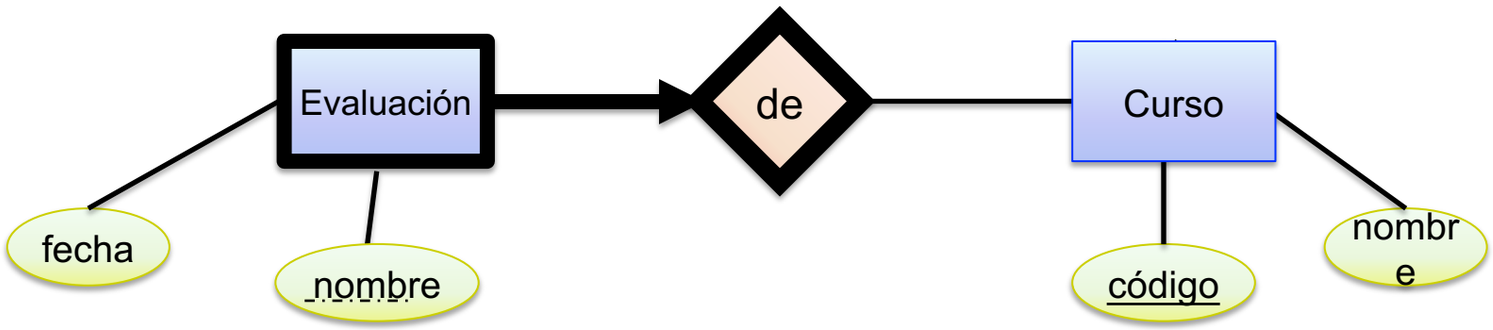


Curso(codigo: string, nombre: string)
Evaluación(nombre: string, C.código: string, fecha: date)

```
CREATE TABLE evaluacion(
    nombre varchar(30) NOT NULL,
    codigo varchar(30) NOT NULL,
    fecha date DEFAULT NOW(),
    PRIMARY KEY (nombre, codigo)
    FOREIGN KEY(codigo) REFERENCES curso(codigo) ON DELETE CASCADE
)
```

¿Que restricciones ves aquí?

M. E/R →M. Relacional



Curso(codigo: string, nombre: string)

Evaluación(nombre: string, C.código: string, fecha: date)

¿Que restricciones ves aquí?

```
CREATE TABLE evaluacion(
    nombre varchar(30) NOT NULL,
    codigo varchar(30) NOT NULL,
    fecha date DEFAULT NOW(),
    PRIMARY KEY (nombre, codigo)
    FOREIGN KEY(codigo) REFERENCES curso(codigo) ON DELETE CASCADE
)
```

No puede ser nulo

Tiene que tener a lo más 30 caracteres

No puede ser nulo

Tiene que ser una fecha, y su valor por defecto es la fecha actual

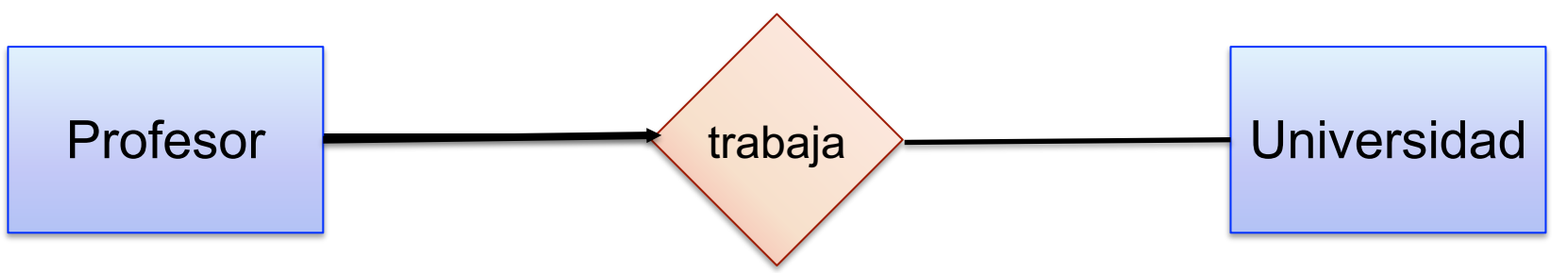
La llave son “nombre” y “código”

“codigo” es una llave foránea a la tabla curso

Borrar las tuplas de evaluación que dependan de un código en la tabla curso que fue eliminado.

Restricciones de integridad

Integridad de la entidad



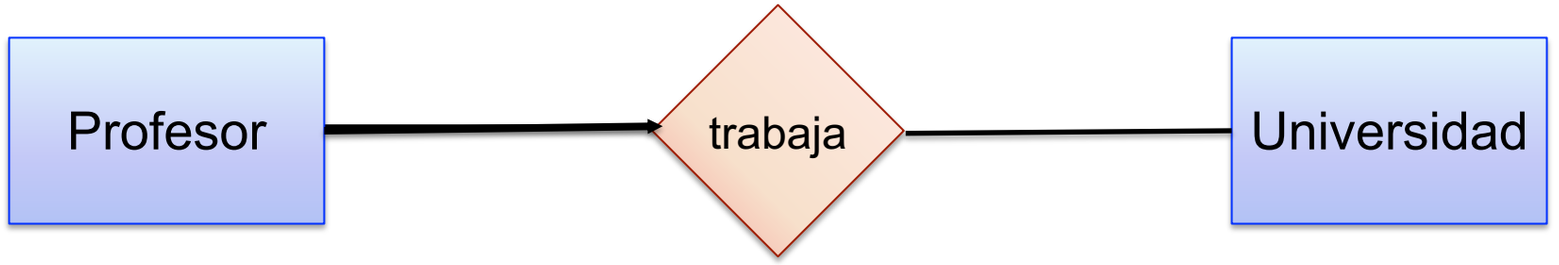
Un ejemplo de tabla de profesor con algunas restricciones:

```
CREATE TABLE Profesor(
    id int PRIMARY KEY,
    nombre varchar(30) NOT NULL,
    apellidos varchar(30) NOT NULL,
    telefono varchar(30) NOT NULL,
    id_universidad int,
    nivel varchar(20) DEFAULT 'Pregrado'
    FOREIGN KEY(id_universidad) REFERENCES Universidad(id)
)
```

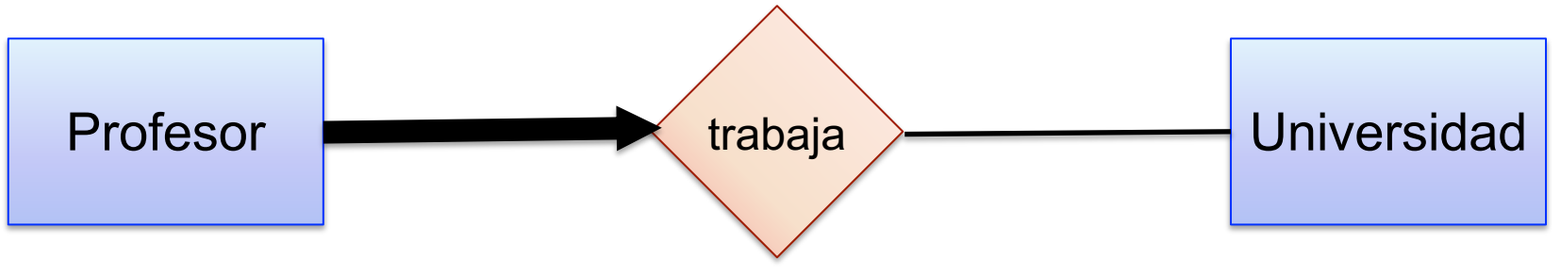
Restricciones de integridad

Participación

Cada profesor puede trabajar en una única universidad (pero puede estar sin trabajo!):



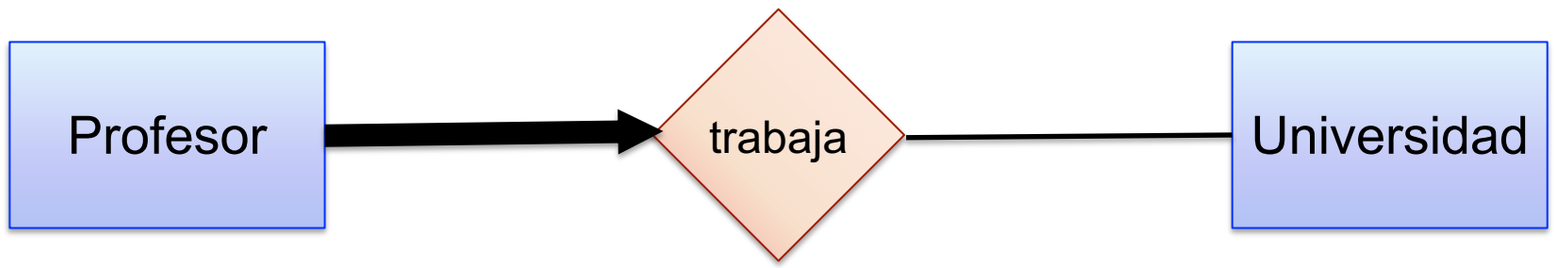
Cada profesor necesariamente trabaja en una única universidad:



Restricciones de integridad

Participación

Cada profesor necesariamente trabaja en una única universidad




```
CREATE TABLE Profesor(
    id int PRIMARY KEY,
    nombre varchar(30) NOT NULL,
    apellidos varchar(30) NOT NULL,
    telefono varchar(30) NOT NULL,
    id_universidad int NOT NULL,
    nivel varchar(20) DEFAULT 'Pregrado'
    FOREIGN KEY(id_universidad) REFERENCES Universidad(id)
)
```

Restricciones de integridad

Dominio

Queremos restringir el dominio de las columnas. Una forma simple de hacer esto en SQL es con CHECK:

```
CREATE TABLE Festival(
    id int PRIMARY KEY,
    nombre varchar(30) NOT NULL,
    fecha_inicio date NOT NULL,
    fecha_fin date NOT NULL,
    precio int NOT NULL,
    CHECK( precio BETWEEN 10000 AND 1000000 ),
    CHECK( fecha_fin > fecha_inicio)
)
```