MODELADO DE BASES DE DATOS

MODELADO

Es un proceso que parte en el análisis de un problema y finaliza en el diagrama de nuestra base de datos.

¿POR QUÉ ES NECESARIO EL MODELADO DE DATOS?

Es crucial para la consistencia, integridad y precisión de los datos.

Un mal modelo repercute negativamente en la fidelidad de los datos entregados, y puede complicar seriamente la construcción de una aplicación.

¿CUÁNDO SE MODELA?

Antes de empezar a crear un proyecto, se debe realizar un modelado de las diferentes partes que componen la aplicación.

Cliente nos cuenta su idea

Modelamos el problema

Programamos

MODELADO DE DATOS

Identificar el flujo de nuestra aplicación.

Decidir cual es la información que guardaremos.

EJEMPLO

Un usuario entra a el sistema y debe poder ver un listado de productos con sus precios.

¿Qué tablas y columnas tenemos que tener en la base de datos para poder construir esta parte de la aplicación?

EJEMPLO

Un usuario entra a el sistema y debe poder ver un listado de productos con sus precios.

Guardaremos

- Usuario
- Producto
- Precio

¿CUÁLES DE ESTOS DATOS DEPENDEN DE OTROS?

Usuario

Producto

Precio

¿CUÁLES DE ESTOS DATOS DEPENDEN DE OTROS?

Usuario

Producto

Precio

¿CUÁLES DE ESTOS DATOS DEPENDEN DE OTROS?

Usuario

Producto

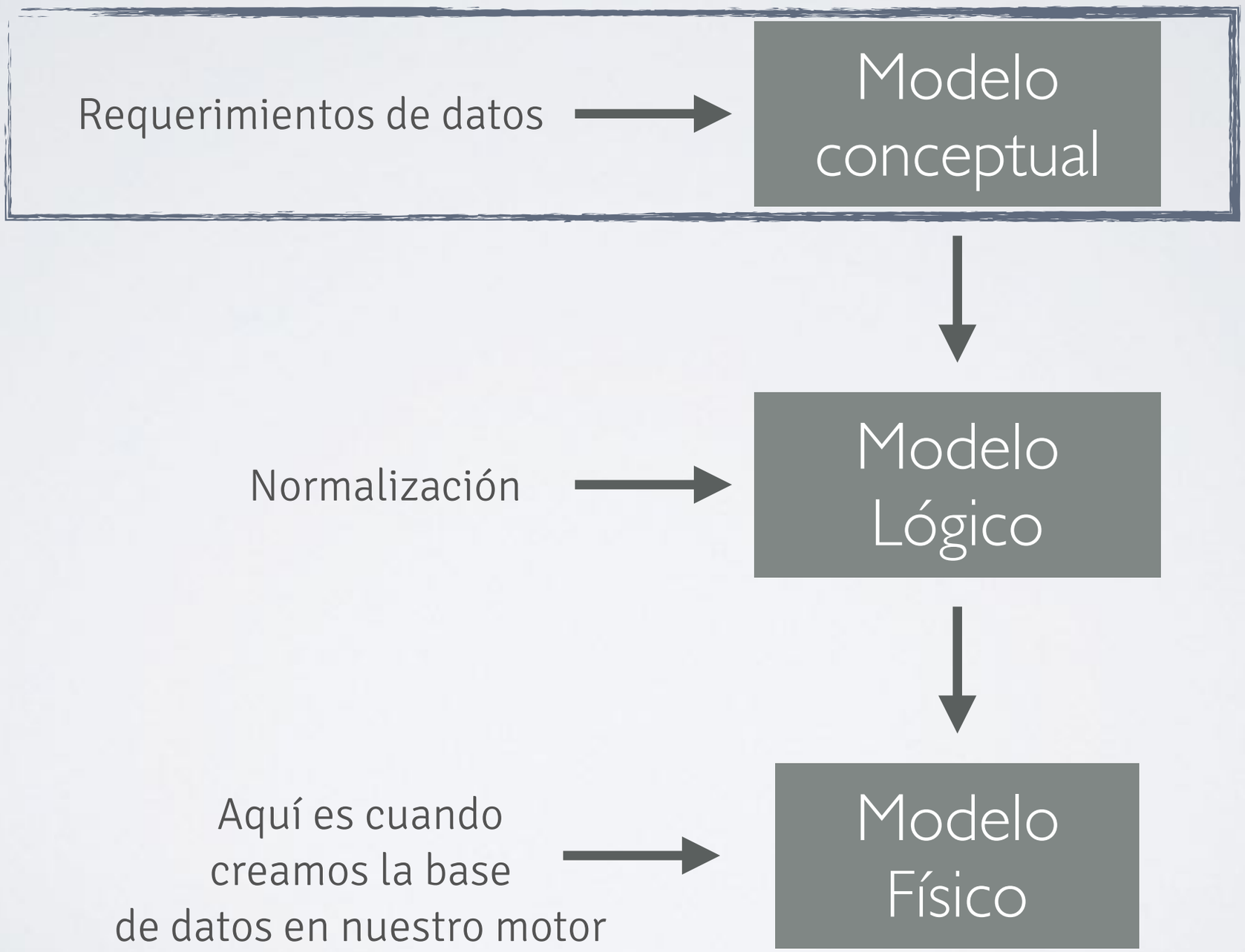
Precio

MODELADO DE DATOS

Existen diversas etapas en el modelado de datos, pondremos énfasis en las siguientes tres.

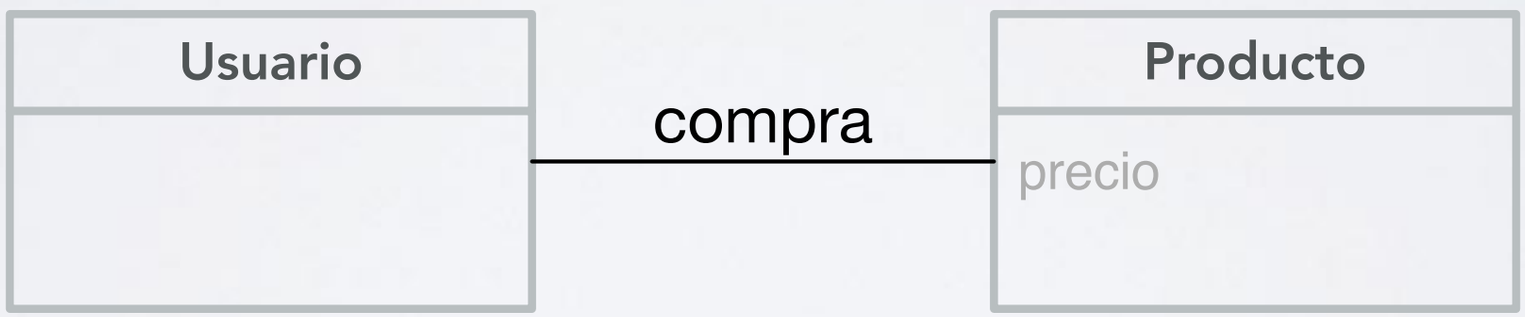
1. Modelo Conceptual
2. Modelo Lógico
3. Modelo Físico

MODELOS DE DATOS



MODELO CONCEPTUAL

Consiste en la identificación de las entidades del sistema, sus atributos y las relaciones de las entidades.



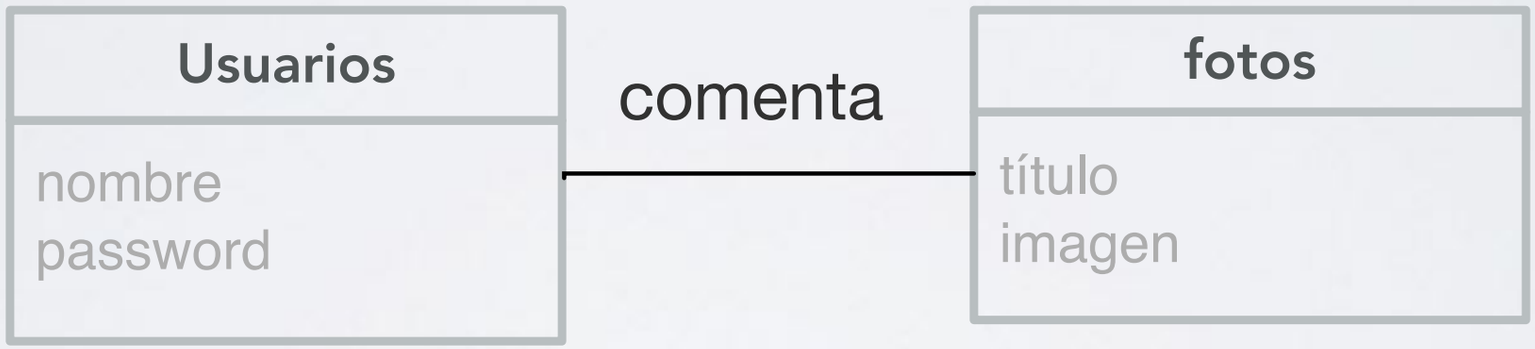
ACOTACIÓN

Existen diversas técnicas para diagrama modelos, dos de los más usados son UML, Entidad Relación (ER).

EJERCICIO

Un usuario ingresa con su nombre y password a la plataforma, aparecen distintas fotos con sus títulos y puede comentar sobre ellas.

SOLUCIÓN



"Entendemos modelos conceptuales cuando somos capaces de distinguir entidades de atributos y sus relaciones"

Paulo Coelho

EJERCICIO

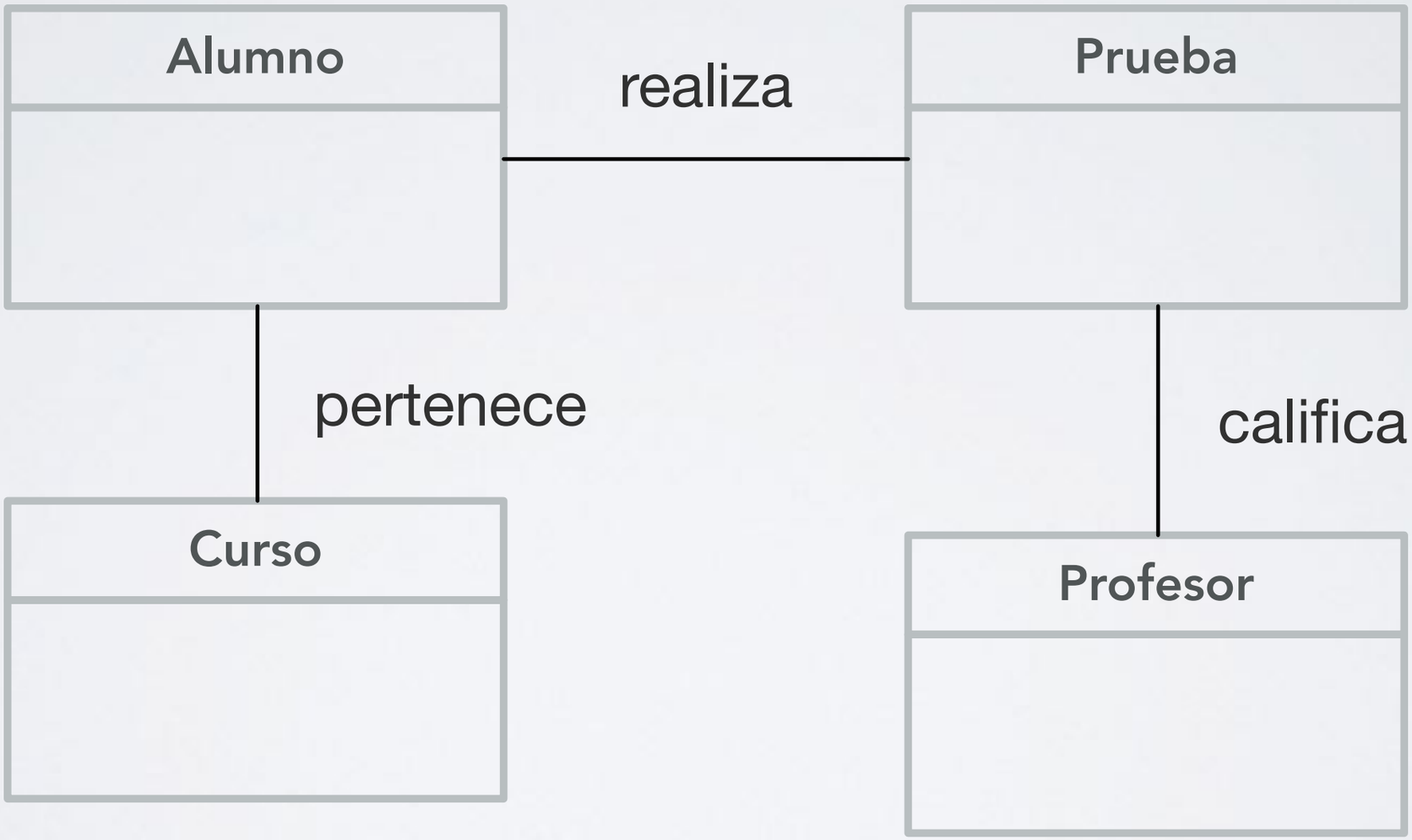
En una red social de citas un usuario puede hacer match con otro usuario.

SOLUCIÓN

EJERCICIO

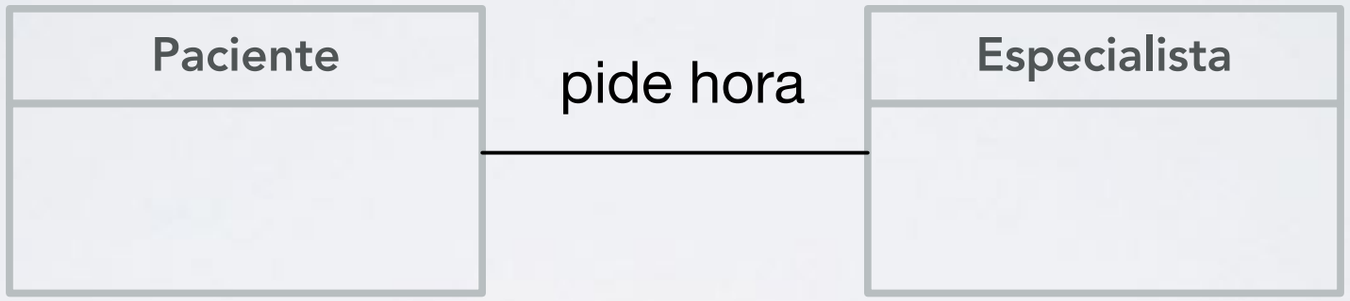
En un curso los alumnos toman una prueba que luego es calificada por el profesor.

SOLUCIÓN



Se necesita una plataforma para que un paciente pueda pedir hora a un especialista en un centro médico.

SOLUCIÓN



MODELO LÓGICO

El propósito del diseño lógico es la creación de un esquema conceptual que luego pueda ser usado en cualquier motor de base de datos.

El motor de base de datos tiene que ser del mismo tipo (relacional / NOSQL / Grafos / etc)

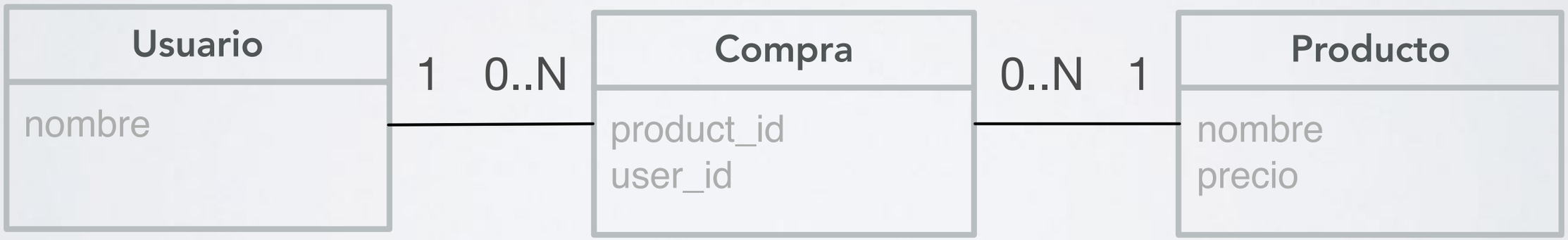
NOSOTROS TRABAJAREMOS CON BASES DE DATOS RELACIONALES

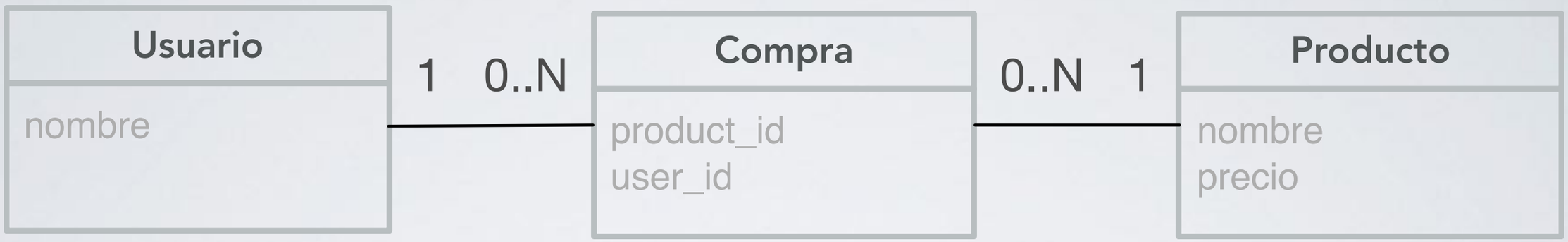
RESTRICCIÓN IMPORTANTE

Toda nuestra información tiene que estar en tablas, podemos relacionar esas tablas a través de claves foráneas.

POR EJEMPLO LA INFORMACIÓN DE LA COMPRA TIENE QUE ESTAR EN UNA TABLA

POR EJEMPLO LA INFORMACIÓN DE LA COMPRA TIENE QUE ESTAR EN UNA TABLA





En el modelo lógico deben aparecer todos los atributos, además se introducen la cardinalidad, las claves primarias y las claves foráneas.

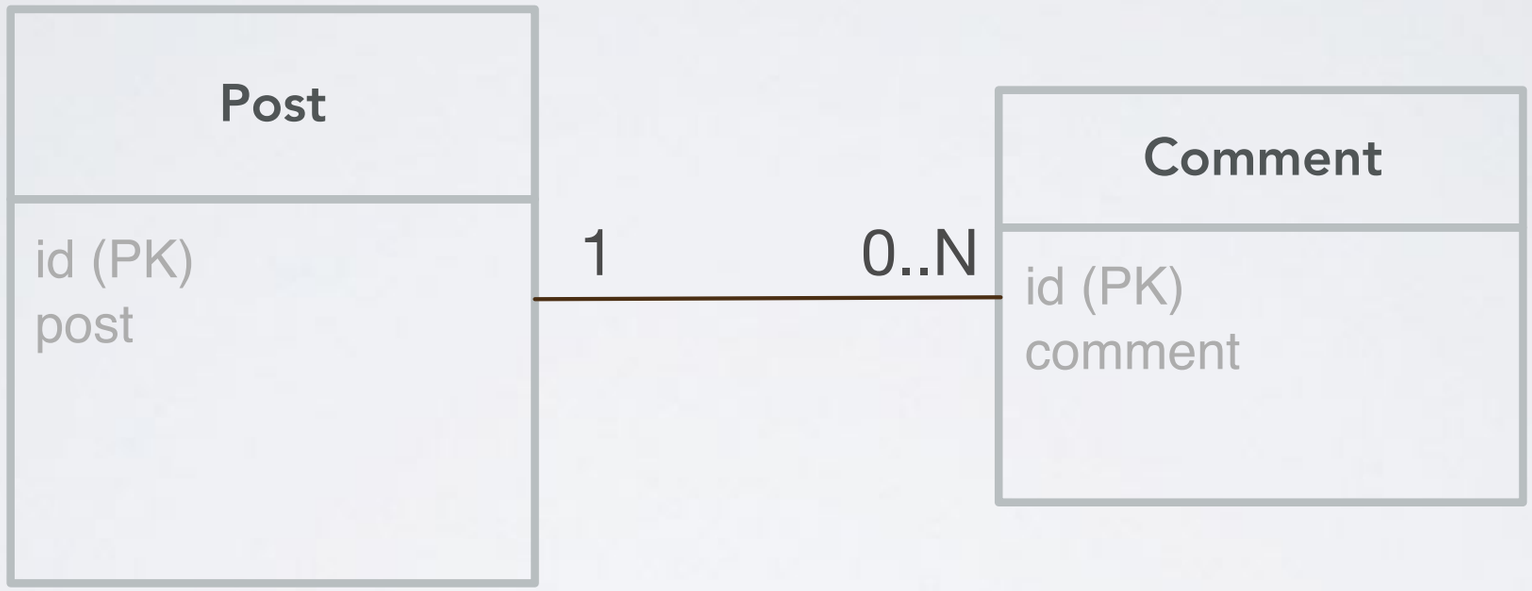
PARA ENTENDER COMO CONSTRUIR ESTOS MODELOS NECESITAMOS CONOCER LOS TIPOS DE RELACIONES QUE PODEMOS HACER.

TIPOS DE RELACIONES

1 a 1	1 a N	N a N

MODELADO

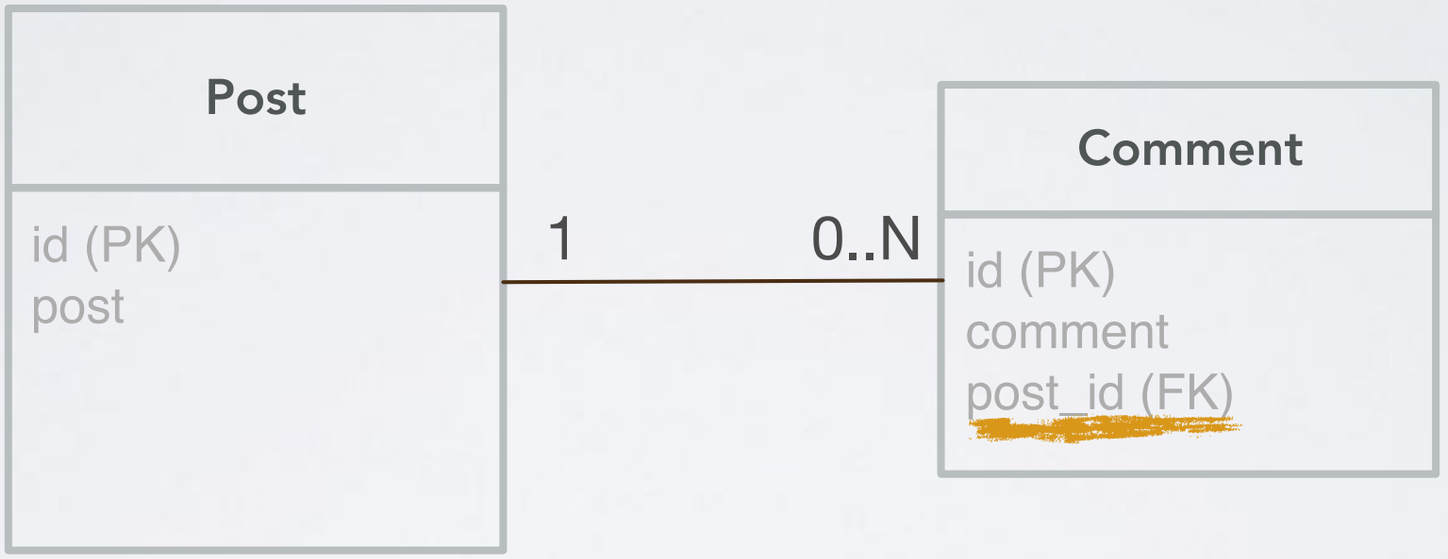
Relaciones 1 a N



Este tipo de relaciones es el mas frecuente, en este ejemplo permite que un post tenga múltiples comentarios, mientras que un comentario ingresado debe pertenecer a un post.

LA CLAVE QUE RELACIONA

Relaciones 1 a N



La clave Foránea siempre va por el lado de las N

PONIENDO LA FK

Posts

id	post
1	Lorem Ipsum
2	Spoiler de #GOT

Comments

id	comment	post_id
1	No mas Spoilers	2
2	Yo todavía no lo veo	2

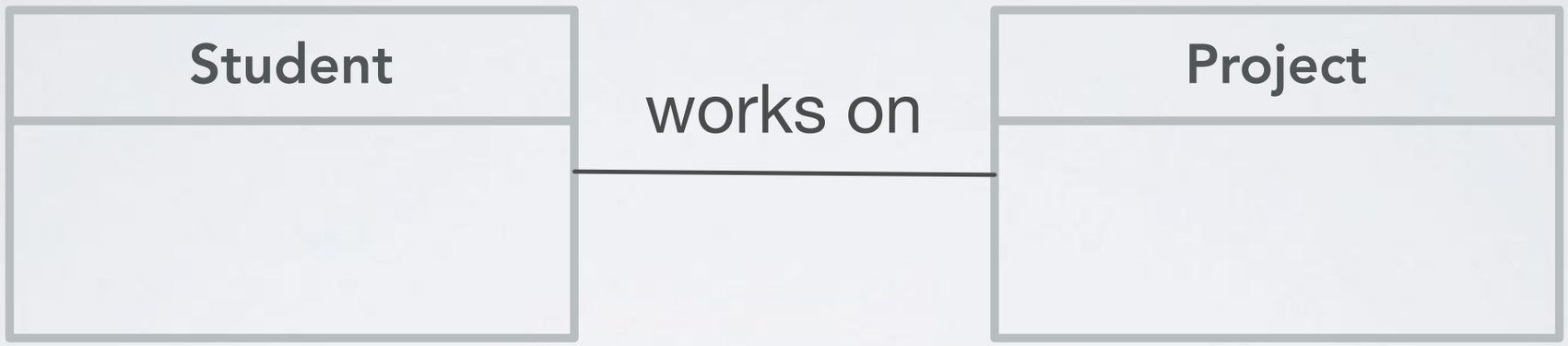
Gracias a que está la FK en el lado de los comentarios cada post puede tener múltiples comentarios.

EJERCICIO:
REALIZAR EL MODELO LÓGICO

Al entrar a la plataforma se ve un listado de países junto a sus ciudades respectivas, se puede agregar y borrar ciudades y países.

SOLUCIÓN

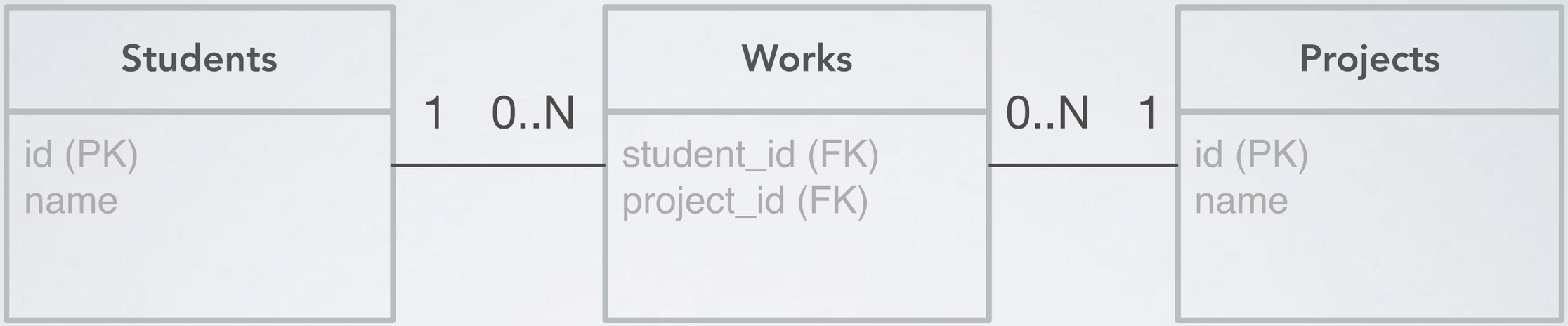
RELACIONES N A N



TRABAJANDO CON N A N

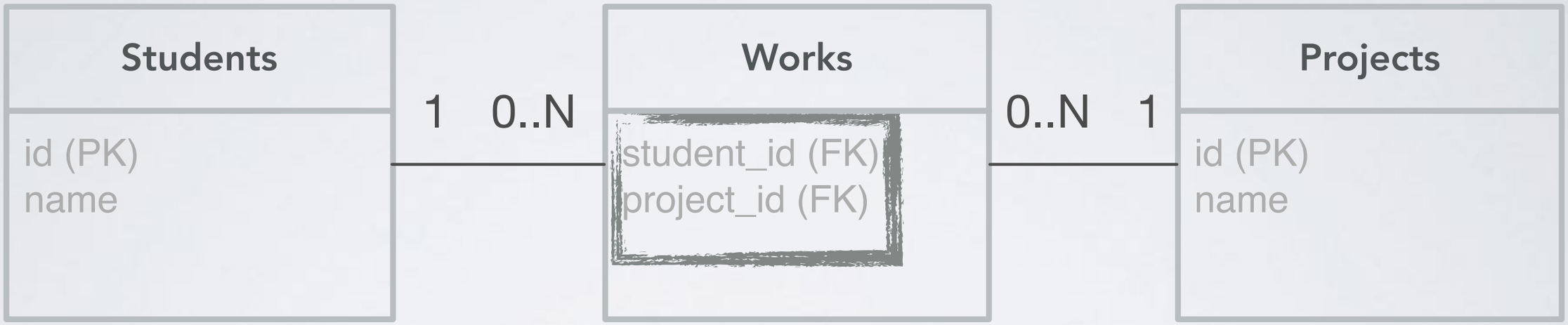
Los motores de datos relacionales no pueden almacenar directamente relaciones N a N, pero podemos desacoplarlas en dos relaciones de 1 a n

TRABAJANDO CON N A N



Las claves foráneas siempre van en la tabla intermedia

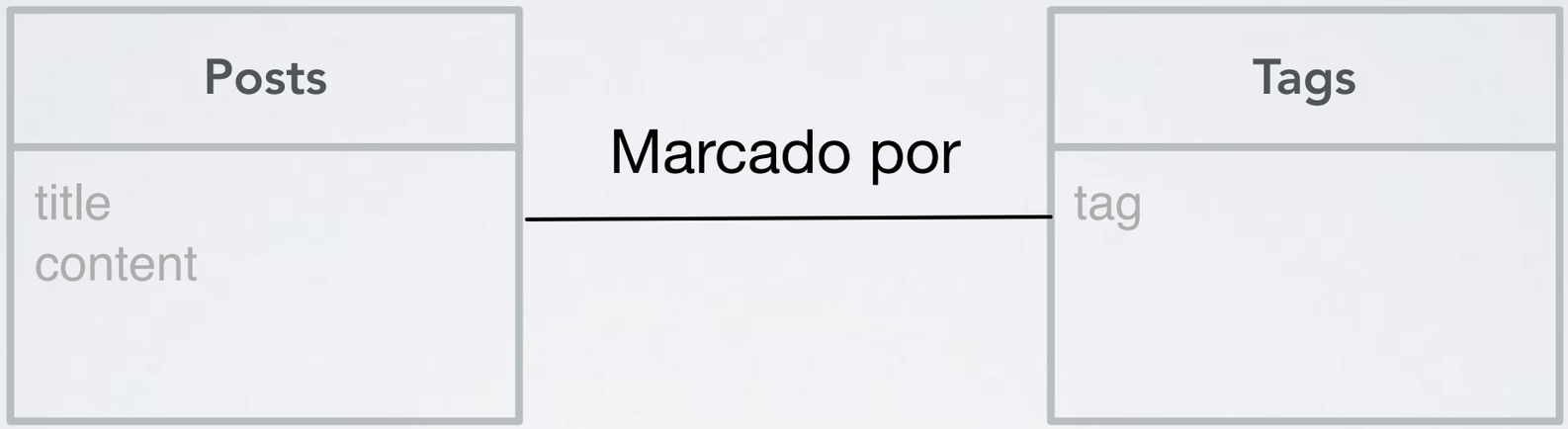
TRABAJANDO CON N A N



Las claves foráneas siempre van en la tabla intermedia

VEAMOS OTROS EJEMPLO

Tenemos un blog



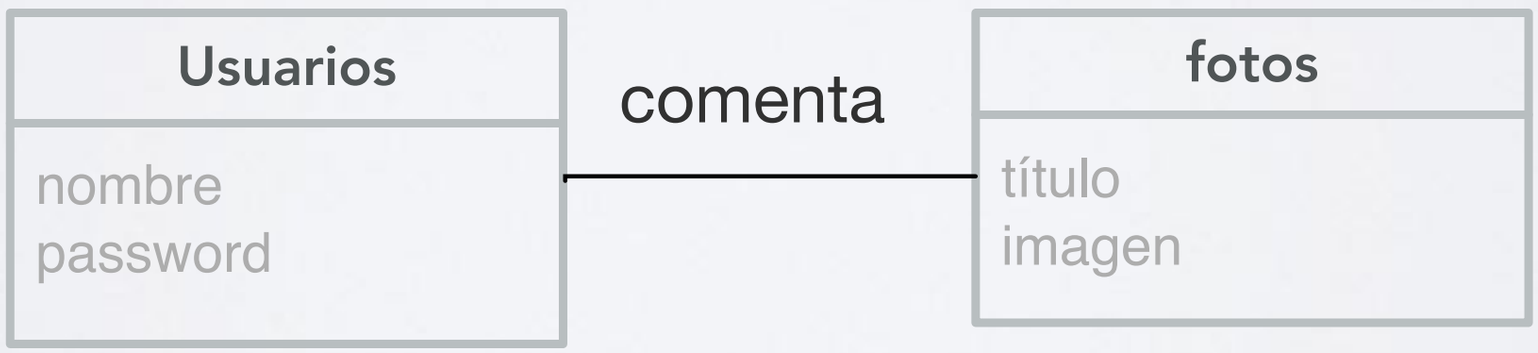
Un post puede tener múltiples tags, ejemplo tecnología, e internet y el mismo tag (ej: tecnología) puede estar en múltiples post

SOLUCIÓN


Posts
id (PK)
title
content
1
0..N
TaggedPost
post_id (FK)
tag_id (FK)
0..N
1
Tags
id (PK)
tag

UN CASO LIGERAMENTE DISTINTO

Se necesita una plataforma donde haya diversas fotos con sus títulos e imagen y los usuarios puedan comentar sobre ellas.

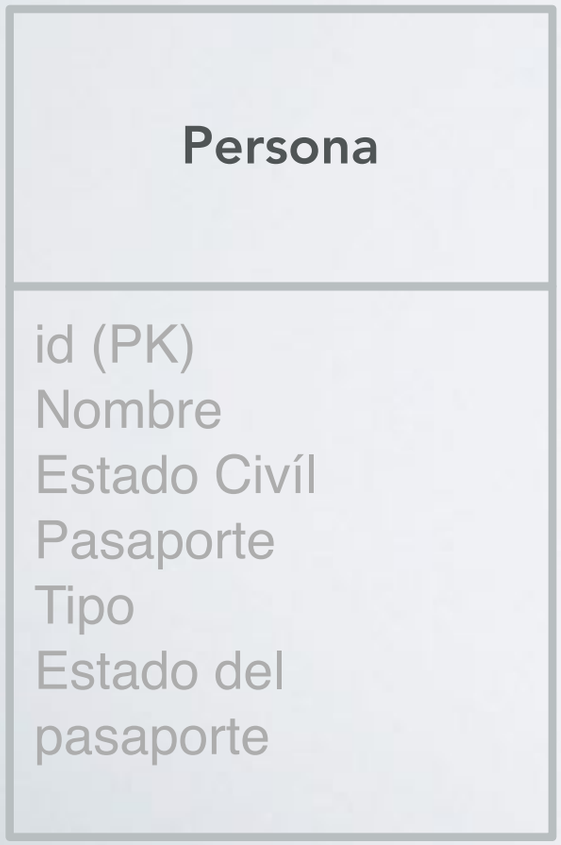


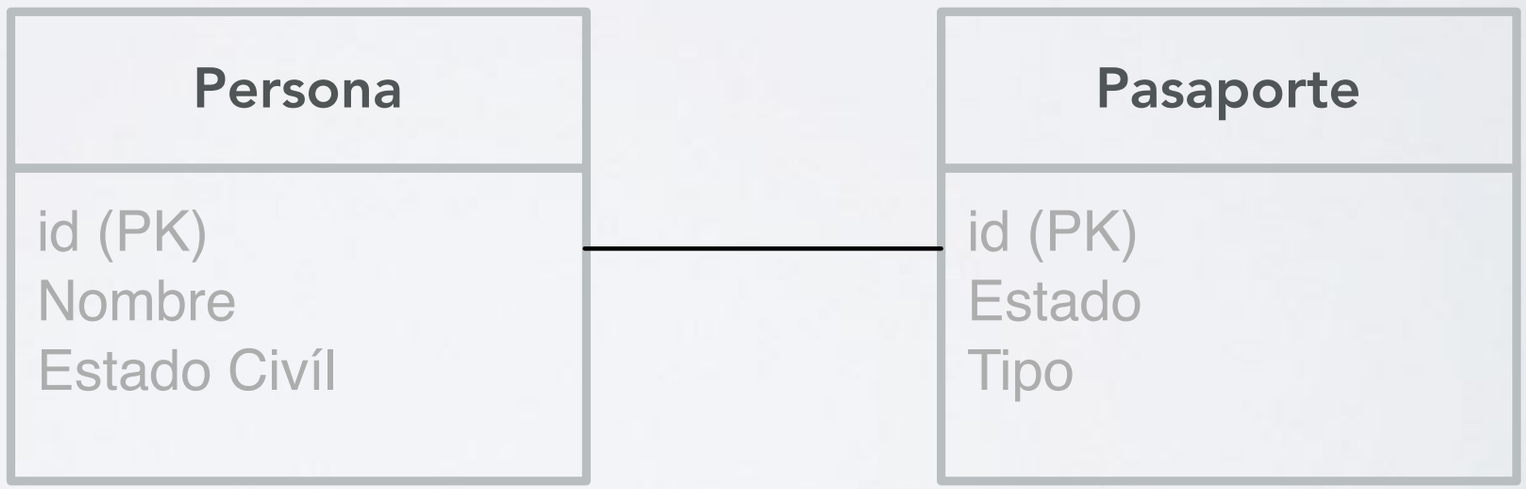
SOLUCIÓN


usuario
id (PK)
name
1
0..N
commentario
id (PK)
comment
foto_id
usuario_id
0..N
1
foto
id (PK)
title
imagen

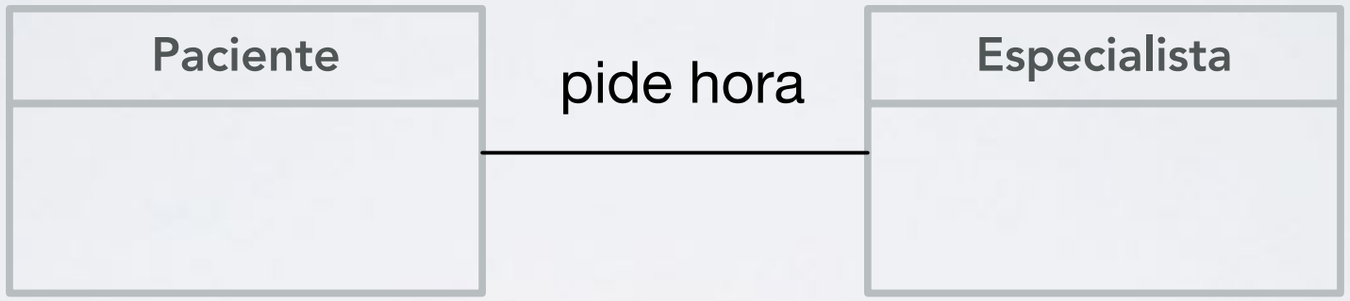
RELACIONES 1 A 1

Las relaciones 1 a 1 son mucho menos utilizadas que las anteriores, nos permiten separar una tabla que contiene muchos valores nulos en dos tablas.

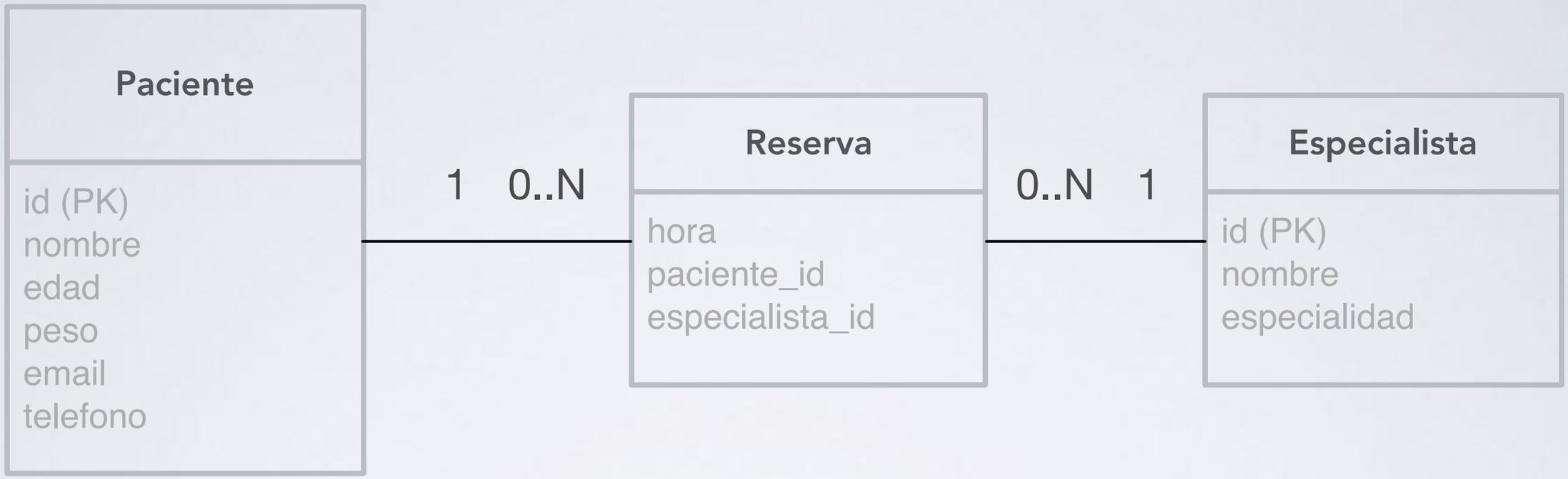




EJERCICIO:
PASAR A MODELO LÓGICO



SOLUCIÓN



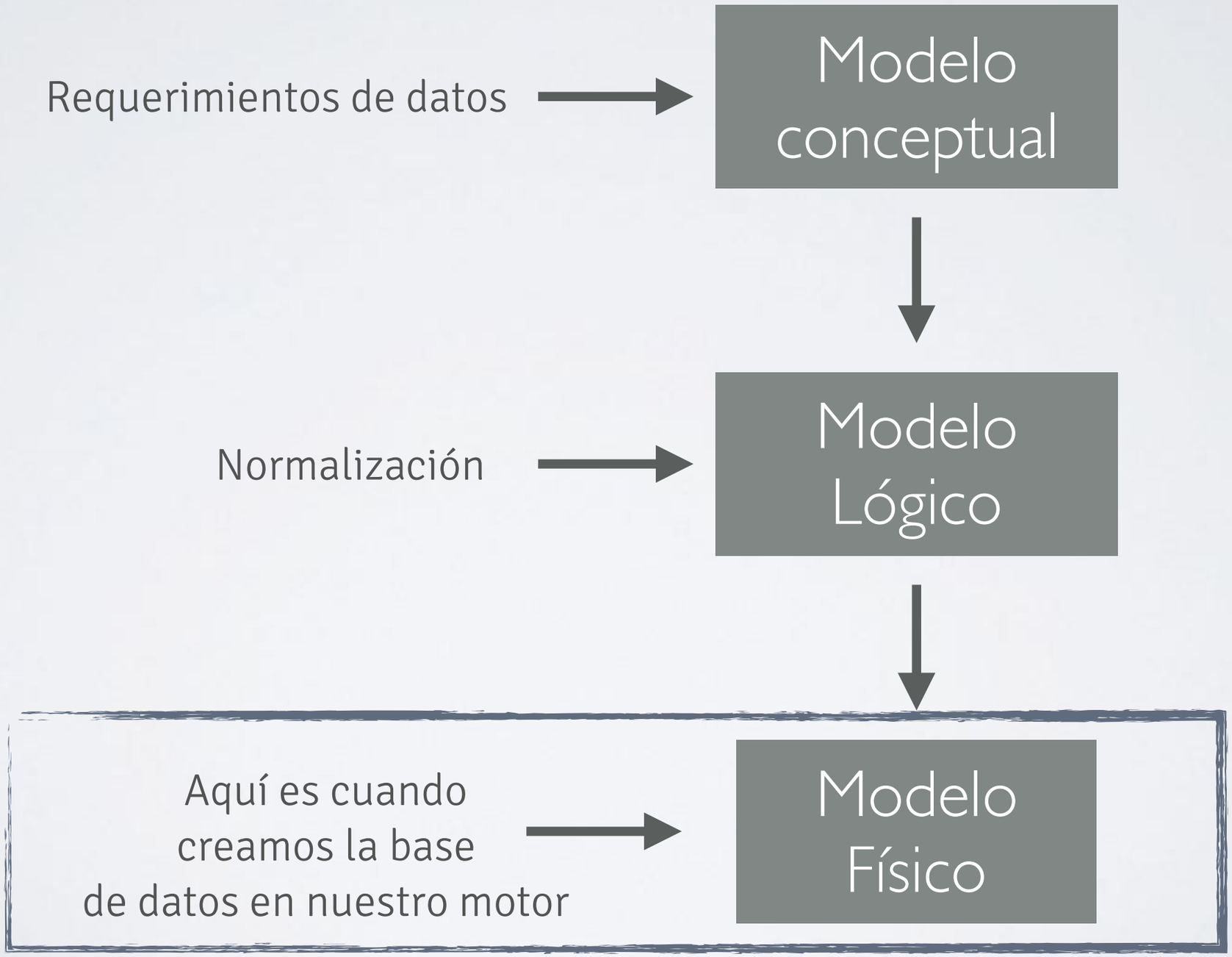
Hagamos otra mejora, evitemos la redundancia en el campo especialidad.

SOLUCIÓN


Paciente
id (PK)
nombre
edad
peso
email
telefono
1
0..N
Reserva
hora
paciente_id (FK)
especialista_id (FK)
0..N
1
Especialista
id (PK)
nombre
especialidad_id (FK)
0..N
1
Especialidad
id (PK)
nombre

PROFUNDIZAREMOS EN EL MODELO LÓGICO DESPUÉS DE CONOCER EL MODELO FÍSICO

MODELO FÍSICO



MODELOS FÍSICO

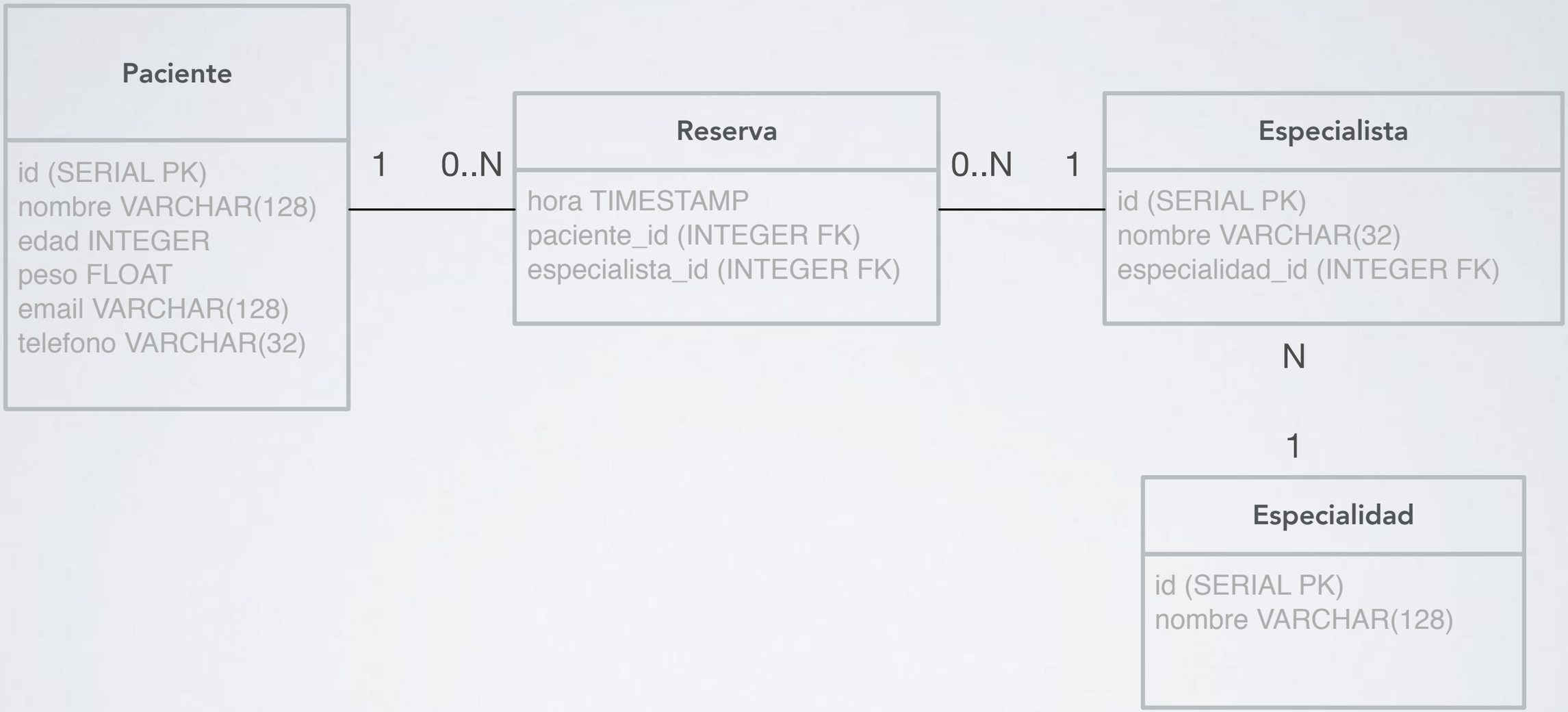
Aquí se adapta el modelo lógico
al motor de la base de datos

PASOS IMPORTANTES

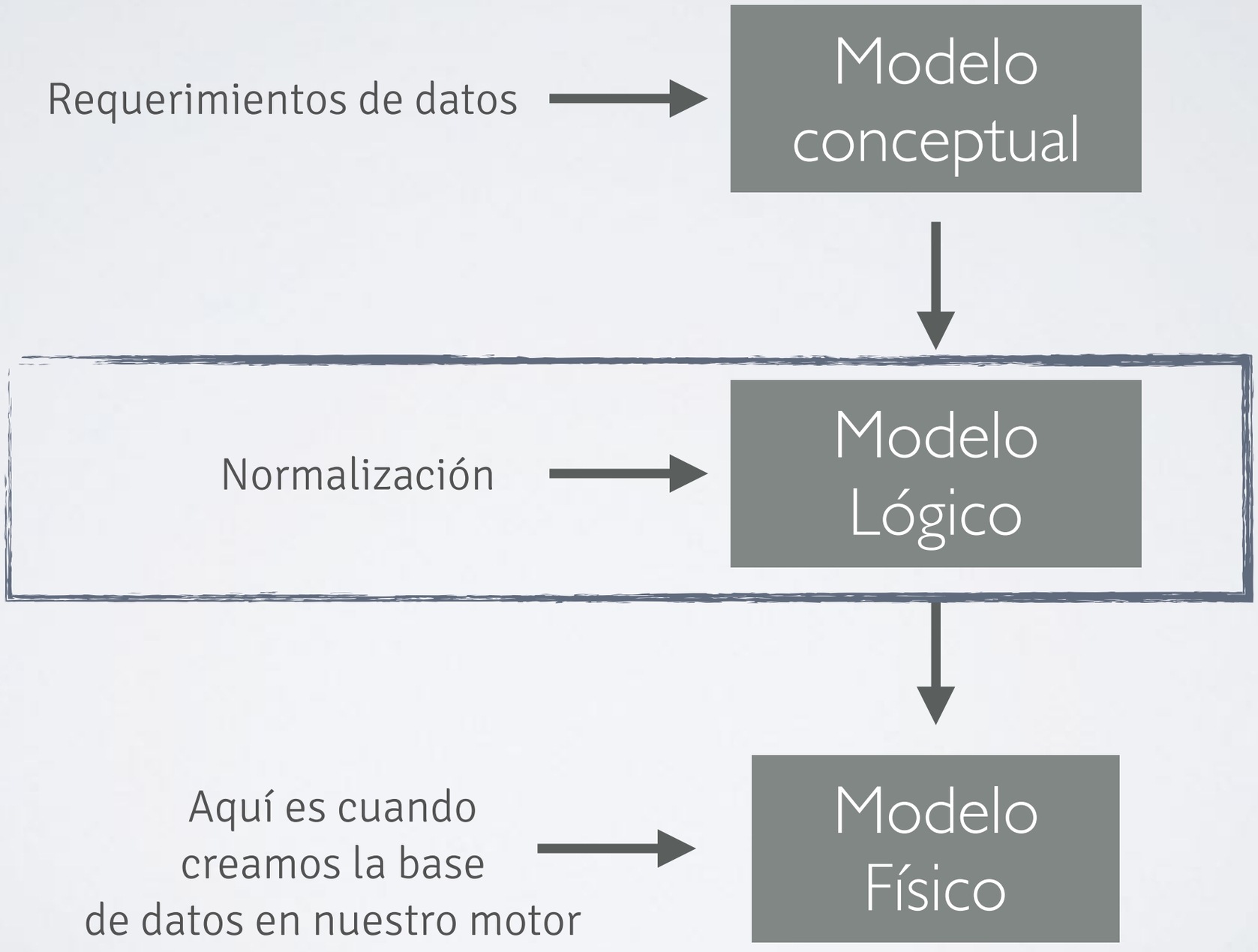
- Especificar tipos de datos para las columnas
- Especificar ciertos índices necesarios
- Especificar los campos que no pueden ser nulos
- Se crean los stored procedures (funciones que son creadas dentro del motor de base de datos)

EJERCICIO:
PASAR A MODELO FÍSICO

SOLUCIÓN


Paciente
id (SERIAL PK)
nombre VARCHAR(128)
edad INTEGER
peso FLOAT
email VARCHAR(128)
telefono VARCHAR(32)
1
0..N
Reserva
hora TIMESTAMP
paciente_id (INTEGER FK)
especialista_id (INTEGER FK)
0..N
1
Especialista
id (SERIAL PK)
nombre VARCHAR(32)
especialidad_id (INTEGER FK)
N
1
Especialidad
id (SERIAL PK)
nombre VARCHAR(128)

NORMALIZACIÓN



¿QUÉ ES NORMALIZACIÓN?

- El concepto introducido por Edgar F. Codd (1972).
- Se utiliza para verificar esquemas relacionales.
- Buscar eliminar la redundancia de datos

¿CUÁNDO NORMALIZAR?

Cuando tenemos una tabla de la cual no tenemos certeza si hay redundancia.

¿PARA QUÉ SE NORMALIZAN LAS BASES DE DATOS?

Evitar la redundancia de datos.

Disminuir problemas en inserción, modificación y eliminación de registros en la base de datos.

¿POR QUÉ SE NORMALIZAN LAS BASES DE DATOS?

Para proteger la integridad de los datos

¿CÓMO SE NORMALIZA UNA BASE DE DATOS?

DESAFIO LATAM

Aplicando las **Formas Normales**

- Las formas normales son aplicadas a las tablas de una base de datos.
- Decir que una base de datos está en la forma normal N, es decir que todas sus tablas están en la forma normal N.
- En general, las primeras tres formales normales son suficientes para cubrir las necesidades de la mayoría de las bases de datos.

PRIMERA FORMA NORMAL

Se deben cumplir 3 reglas.

1. Cada campo debe tener un solo valor
2. No pueden haber grupos repetitivos
3. Identificar cada grupo de datos relacionados con una clave primaria (PK).

¿ESTÁ EN PRIMERA FORMA NORMAL?

Customer ID	First Name	Surname	Telephone Number
123	Pooja	Singh	555-861-2025, 192-122-1111
456	San	Zhang	(555) 403-1659 Ext. 53; 182-929-2929
789	John	Doe	555-808-9633

UN CAMPO PUEDE TENER UN SOLO VALOR

Customer ID	First Name	Surname	Telephone Number
123	Pooja	Singh	555-861-2025, 192-122-1111
456	San	Zhang	(555) 403-1659 Ext. 53; 182-929-2929
789	John	Doe	555-808-9633

REPASEMOS LAS REGLAS

1. Cada campo debe tener un solo valor
2. No pueden haber grupos repetitivos
3. Identificar cada grupo de datos relacionados con una clave primaria (PK).

¿Y AHORA ESTÁ EN PRIMERA FORMA NORMAL?

Customer ID	First Name	Surname	Telephone Number1	Telephone Number2
123	Pooja	Singh	555-861-2025	192-122-1111
456	San	Zhang	(555) 403-1659 Ext. 53	182-929-2929
789	John	Doe	555-808-9633	

NO PUEDEN HABER GRUPOS REPETITIVOS

Customer ID	First Name	Surname	Telephone Number1	Telephone Number2
123	Pooja	Singh	555-861-2025	192-122-1111
456	San	Zhang	(555) 403-1659 Ext. 53	182-929-2929
789	John	Doe	555-808-9633	

GRUPOS REPETITIVOS

nombre	empresa	direccion_empresa	url1	url2
David	ABC	Avenida ABC 111	abc.com	abc.cl
Gonzalo	LMN	Calle LMN 222	lmn.com	lmn.cl
Magdalena	XYZ	Paseo XYZ 333	xyz.com	xyz.cl

Los **atributos o grupos repetitivos** son atributos que son inherentemente iguales. En este caso los atributos **url1** y **url2** son esencialmente lo mismo

¿Y AHORA ESTÁ EN PRIMERA FORMA NORMAL?

Customer ID	First Name	Surname	Telephone Number
123	Pooja	Singh	555-861-2025
123	Pooja	Singh	192-122-1111
456	San	Zhang	182-929-2929
456	San	Zhang	(555) 403-1659 Ext. 53
789	John	Doe	555-808-9633

3º REGLA

Identificar cada grupo de datos relacionados con una clave primaria (PK).

Customer ID	First Name	Surname	Telephone Number
123	Pooja	Singh	555-861-2025
123	Pooja	Singh	192-122-1111
456	San	Zhang	182-929-2929
456	San	Zhang	(555) 403-1659 Ext. 53
789	John	Doe	555-808-9633

Tenemos dos opciones: Creamos una PK que sea una combinación entre el ID y el teléfono, o, separamos la columna de teléfono en otra tabla

REPASEMOS CON OTRO EJEMPLO

Se pide dejar en **primera** forma normal.

nombre	empresa	direccion_empresa	url1	url2
David	ABC	Avenida ABC 111	abc.com	abc.cl
Gonzalo	LMN	Calle LMN 222	lmn.com	lmn.cl
Magdalena	XYZ	Paseo XYZ 333	xyz.com	xyz.cl

nombre	empresa	direccion_empresa	url1	url2
David	ABC	Avenida ABC 111	abc.com	abc.cl
Gonzalo	LMN	Calle LMN 222	lmn.com	lmn.cl
Magdalena	XYZ	Paseo XYZ 333	xyz.com	xyz.cl

Observar los campos **url1** y **url2**

¿Qué haremos cuando en nuestra aplicación necesitemos una tercera url ?

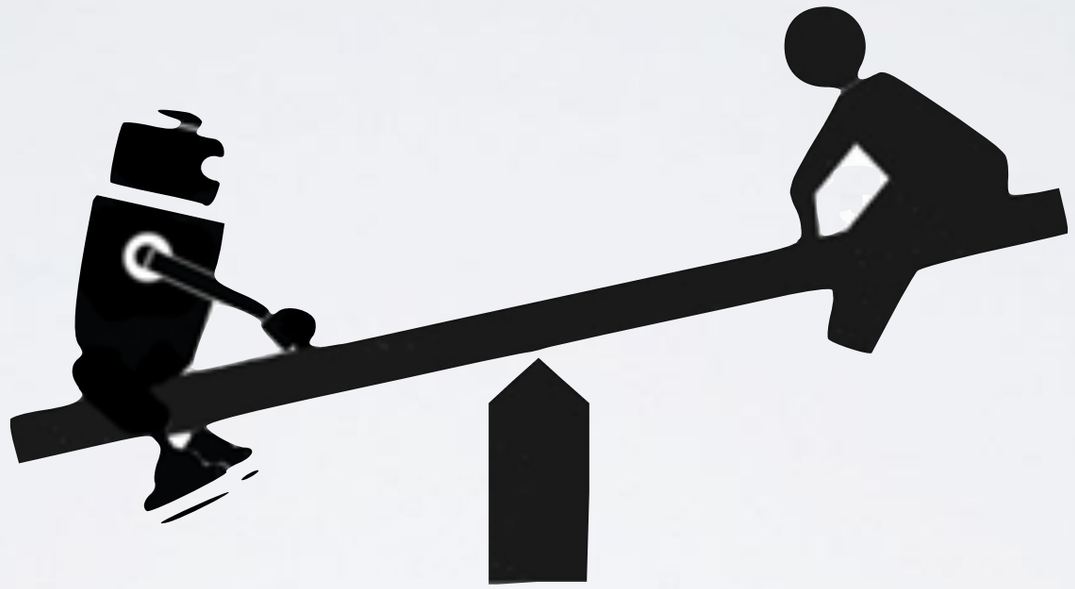
¿ Agregaremos otro campo/columna a nuestra tabla y tendremos que re-programar toda la entrada de datos de nuestro código ?

PRIMERA FORMA NORMAL (1FN)

usuarios

userId	nombre	empresa	direccion_empresa	url
1	David	ABC	Avenida ABC 111	abc.com
1	David	ABC	Avenida ABC 111	abc.cl
2	Gonzalo	LMN	Calle LMN 222	lmn.com
2	Gonzalo	LMN	Calle LMN 222	lmn.cl
3	Magdalena	XYZ	Paseo XYZ 333	xyz.com
3	Magdalena	XYZ	Paseo XYZ 333	xyz.cl

Antes de pasar a la segunda forma normal, debemos entender el concepto de **Dependencia Funcional**



EL COMPORTAMIENTO DE UN ATRIBUTO DEPENDE DE OTRO

DEPENDENCIA FUNCIONAL

numOrden	fechaOrden	numProducto	nombreProducto	cantidad
30311	11/06/2017	101	Martillo	15
30311	11/06/2017	202	Sierra	10
30311	11/06/2017	303	Guantes	7
30312	11/06/2017	404	Clavo	40
30313	12/06/2017	505	Tornillo	13

La fecha de la orden depende del número de la orden, es decir, **por cada número de orden, solo habrá una fecha.**

El nombre del producto depende del número del producto, es decir, **por cada número de producto, sólo habrá un nombre.**

SEGUNDA FORMA NORMAL (2FN)

- Se cumple 1FN.
- Cualquier atributo que no sea clave primaria, es **dependiente de toda la clave primaria** en vez de sólo una parte de ella.
- Los atributos que **dependan de manera parcial de la clave** primaria deben ser **eliminados o almacenados en una nueva entidad.**

¿ESTÁ EN SEGUNDA FORMA NORMAL?

PK compuesta

numOrden	fechaOrden	numProducto	nombreProducto	cantidad
30311	11/06/2017	101	Martillo	15
30311	11/06/2017	202	Sierra	10
30311	11/06/2017	303	Guantes	7
30312	11/06/2017	404	Clavo	40
30313	12/06/2017	505	Tornillo	13

El atributo ***fechaOrden*** no depende de la clave primaria completa, sólo parcialmente de **numOrden**

PK compuesta

numOrden	fechaOrden	numProducto	nombreProducto	cantidad
30311	11/06/2017	101	Martillo	15
30311	11/06/2017	202	Sierra	10
30311	11/06/2017	303	Guantes	7
30312	11/06/2017	404	Clavo	40
30313	12/06/2017	505	Tornillo	13

El campo ***nombreProducto*** no depende de la clave primaria completa, sólo parcialmente de **numProducto**

REPASEMOS LAS REGLAS

- Se cumple 1FN.
- Cualquier atributo que no sea clave primaria, es **dependiente de toda la clave primaria** en vez de sólo una parte de ella.
- Los atributos que **dependan de manera parcial de la clave** primaria deben ser **eliminados o almacenados en otra tabla** y relacionados mediante una FK.

¿ESTÁ EN SEGUNDA FORMA NORMAL?

PK compuesta

numOrden		numProducto		cantidad
30311		101		15
30311		202		10
30311		303		7
30312		404		40
30313		505		13

El atributo **cantidad** es el único que depende de la clave primaria en su totalidad y nos indica **la cantidad de un producto contenido en una orden**

¿ESTÁ EN SEGUNDA FORMA NORMAL?

PK compuesta

numOrden	numProducto	cantidad
30311	101	15
30311	202	10
30311	303	7
30312	404	40
30313	505	13

El atributo **cantidad** es el único que depende de la clave primaria en su totalidad y nos indica **la cantidad de un producto contenido en una orden**

Hemos eliminado los atributos que no dependen de la clave primaria en su totalidad, pero…

¿Cómo almacenaremos la fecha de la orden y el nombre del producto?

"Los atributos que **dependen de manera parcial de la clave** primaria deben ser **eliminados o almacenados en una nueva entidad**"

PK

orden

numOrden	fechaOrden
30311	11/06/2017
30311	11/06/2017
30311	11/06/2017
30312	11/06/2017
30313	12/06/2017

PK

producto

numProducto	nombreProducto
101	Martillo
202	Sierra
303	Guantes
404	Clavo
505	Tornillo

PK compuesta

numOrden	numProducto	cantidad
30311	101	15
30311	202	10
30311	303	7
30312	404	40
30313	505	13

detalle_orden

REPASEMOS CON OTRO EJEMPLO

Se pide dejar en **segunda** forma normal.

codLibro	titulo	Autor	Editorial	codLector	nombre	fechaDevolucion
1001	Intro a Ruby	Homero Simpson	McGraw Hill	223	David Barrientos	abc.com
1002	Intro a Rails	Chuck Norris	McGraw Hill	224	Gonzalo Sanchez	abc.cl
1003	Modelamiento de datos	David Heinemeier	Oracle Corp.	225	Magdalena Claro	lmn.com
1004	Programación	Steve Jobs	Anaya	226	Cristobal Dominguez	lmn.cl

PK compuesta

codLibro	titulo	autor	editorial	codLector	nombre	fechaDevolucion
1001	Intro a Ruby	Homero Simpson	McGraw Hill	223	David Barrientos	20/06/2017
1002	Intro a Rails	Chuck Norris	McGraw Hill	224	Gonzalo Sanchez	04/08/2017
1003	Modelamiento de datos	David Heinemeier	Oracle Corp.	225	Magdalena Claro	08/08/2017
1004	Programación	Steve Jobs	Anaya	226	Cristobal Dominguez	11/09/2017

Los campos **titulo**, **autor** y **editorial**, sólo dependen de una parte de la PK: **codLibro**

El campo **nombre**, sólo dependen de una parte de la PK: **codLector**

SEGUNDA FORMA NORMAL (2FN)

codLibro	titulo	autor	editorial
1001	Intro a Ruby	Homero Simpson	McGraw Hill
1002	Intro a Rails	Chuck Norris	McGraw Hill
1003	Modelamiento de datos	David Heinemeier	Oracle Corp.
1004	Programación	Steve Jobs	Anaya

codLector	nombre
223	David Barrientos
224	Gonzalo Sanchez
225	Magdalena Claro
226	Cristobal Dominguez

codLibro	codLector	fechaDevolucion
1001	223	20/06/2017
1002	224	04/08/2017
1003	225	08/08/2017
1004	226	11/09/2017

Cuando tenemos una entidad en **1FN** con una clave primaria simple, automáticamente estamos aplicando **2FN**

Tabla usuario en 2FN

PK

userId	nombre	empresa	direccion_empresa
1	David	ABC	Avenida ABC 111
1	David	ABC	Avenida ABC 111
2	Gonzalo	LMN	Calle LMN 222
2	Gonzalo	LMN	Calle LMN 222
3	Magdalena	XYZ	Paseo XYZ 333
3	Magdalena	XYZ	Paseo XYZ 333

TERCERA FORMA NORMAL (3FN)

- Se cumple 2FN.
- **Todo atributo debe depender de la clave primaria**, de toda la clave primaria, y no de otro atributo.

¿ESTÁ EN TERCERA FORMA NORMAL?

Tabla usuario en 2FN

PK

userId	nombre	empresa	direccion_empresa
1	Daniela	ABC	Avenida ABC 111
2	Marcelo	ABC	Avenida ABC 111
3	Gonzalo	LMN	Calle LMN 222
4	David	LMN	Calle LMN 222
5	Magdalena	XYZ	Paseo XYZ 333
6	Juan Pablo	XYZ	Paseo XYZ 333

El atributo ***direccion_empresa*** no depende de la clave primaria **userId**, pero sí depende del atributo **empresa**

Tabla usuario en 2FN

PK

userId	nombre	empresa	direccion_empresa
1	Daniela	ABC	Avenida ABC 111
2	Marcelo	ABC	Avenida ABC 111
3	Juan Pablo	LMN	Calle LMN 222
4	David	LMN	Calle LMN 222
5	Magdalena	XYZ	Paseo XYZ 333
6	Gonzalo	XYZ	Paseo XYZ 333

REPASEMOS LAS REGLAS

- Se cumple 2FN.
- **Todo atributo debe depender de la clave primaria**, de toda la clave primaria, y no de otro atributo.

¿ESTÁ EN TERCERA FORMA NORMAL?

PK

userId	nombre
1	Daniela
2	Marcelo
3	Gonzalo
4	David
5	Magdalena
6	Juan Pablo

Los atributos que **no** dependan únicamente de la clave primaria deben ser almacenados en una nueva entidad y **relacionados mediante una clave foránea (FK)**

TERCERA FORMA NORMAL (3FN)

PK FK

userId	nombre	relEmpresaId
1	Daniela	11
2	Marcelo	11
3	Gonzalo	22
4	David	22
5	Magdalena	33
6	Juan Pablo	33

El atributo **nombre** de la entidad usuario depende de la PK

Los atributos **empresa** y **direccion_empresa** dependen de la PK

PK

empresaId	empresa	direccion_empresa
11	ABC	Avenida ABC 111
22	LMN	Calle LMN 222
33	XYZ	Paseo XYZ 333

DESNORMALIZACIÓN

duplicación intencionada de datos en algunas tablas

¿PARA QUÉ SIRVE?

- Las reglas de normalización **no** consideran el rendimiento.

- En algunos casos, es necesario **considerar la desnormalización para mejorar el rendimiento**.

EJEMPLO DE DESNORMALIZACIÓN

Se necesita una plataforma para almacenar componentes, la cantidad de cada uno y la dirección física donde se están almacenando.

componenteId	cantidad	almacenId
1	10	11
2	20	22
3	30	33

almacenId	almacenDir
11	Vicuña Mackenna #20
22	Providencia #1000
33	Los Leones #300

componenteId	cantidad	almacenId
1	10	11
2	20	22
3	30	33

almacenId	almacenDir
11	Vicuña Mackenna #20
22	Providencia #1000
33	Los Leones #300

PODEMOS RECUPERAR LA INFORMACIÓN DE LAS COMPONENTES CON LOS ALMACENES UTILIZANDO **JOIN**

EL PROBLEMA DE JOIN

- La premisa de las sentencias SQL es que pueden recuperar la información uniendo tablas (**join**).

- El problema es que, en algunos casos, se pueden producir **problemas de rendimiento como resultado de una normalización**.

- Al relacionar muchas tablas obtenemos como resultado demasiadas uniones en las consultas y, por ende, pueden aumentar los costes de acceso (**tardar mucho tiempo**)

SOLUCIÓN:

componentes (tabla desnormalizada)

componenteId	cantidad	almacenId	almacenDir
1	10	11	Vicuña Mackenna #20
2	20	22	Providencia #1000
3	30	33	Los Leones #300
4	40	11	Vicuña Mackenna #20
5	50	22	Providencia #1000

Al no ser necesarias operaciones de unión, podría ser que la redundancia valga la pena. Las direcciones de almacenes no cambian a menudo y, si cambia alguna, podemos utilizar SQL para actualizar todos los registros con bastante facilidad.

TIPS

- En muchos casos, la **normalización** y las uniones son el **método de acceso más eficaz**, a pesar de la sobrecarga que suponen.

- No desnormalizar las tablas a menos que se tenga una buena **comprensión de los datos y la lógica de negocios.**

- No optimizar antes de tiempo.