



Curso de
**Fundamentos de
Bases de Datos**

Israel Vázquez

Introducción

Introducción y bienvenida

Israel Vázquez

Senior Web Developer en SF startup YouNoodle. Organizer del GDG Cloud Mexico. Seminarista de Bases de Datos relacionales. Data engineering enthusiast.

Historia de la persistencia de la información

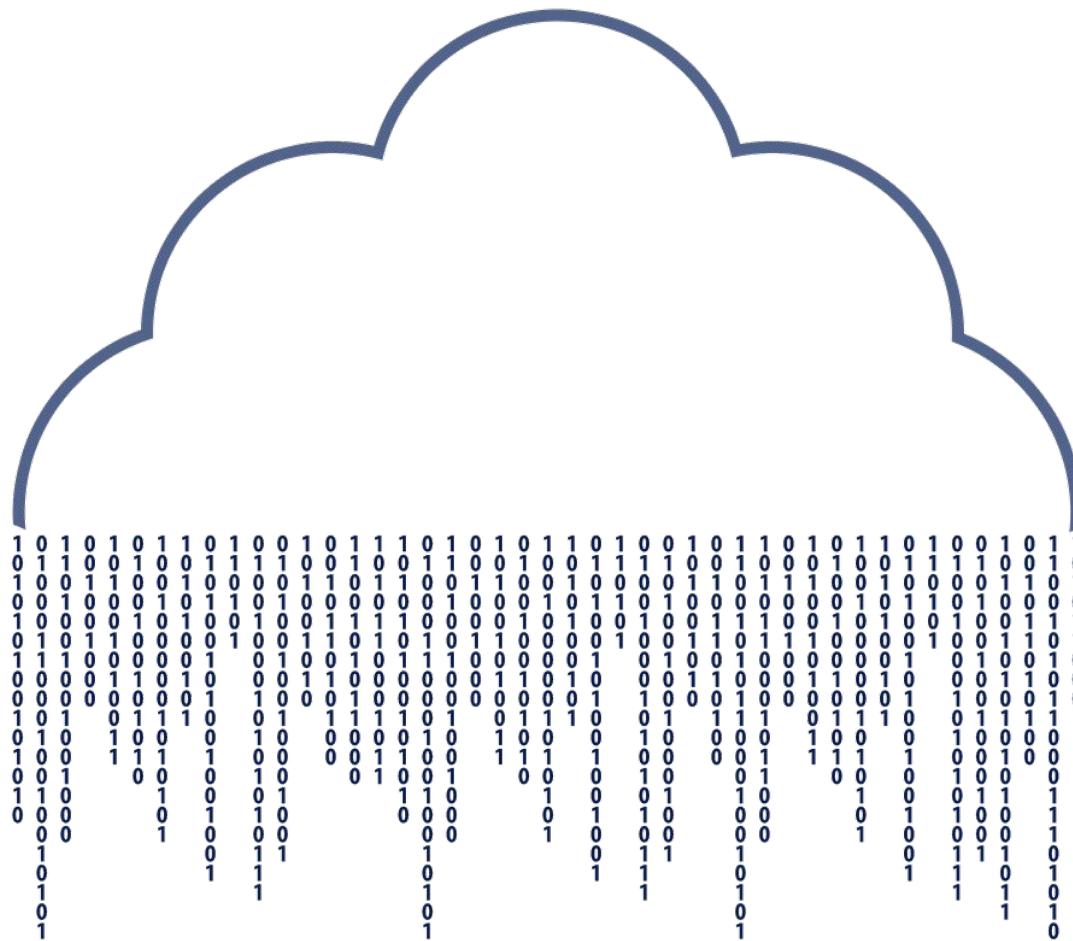
¿Por qué necesitamos persistencia?



A close-up, slightly blurred photograph of two pages from a handwritten notebook. The pages contain dense, cursive handwriting in black ink on aged, yellowish paper. The perspective is from above, looking down at the open pages.







**Entonces,
¿Que *&%)\$@
son las Bases
de Datos?**



Tipos de bases de datos

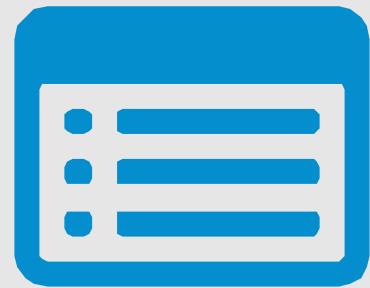


Relacionales

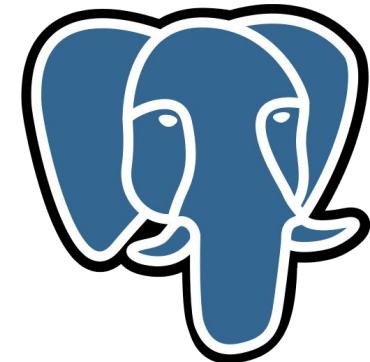
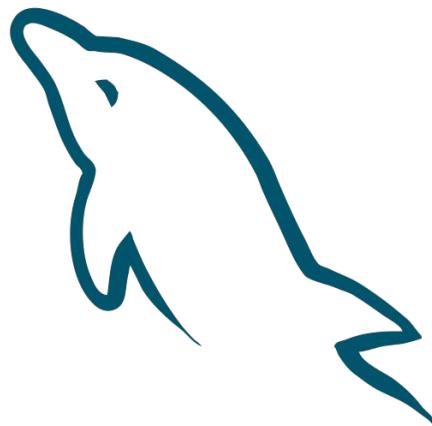


No relacionales

Bases de datos relacionales



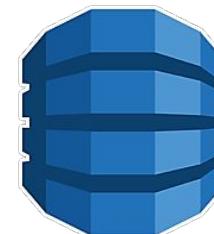
Ejemplos de bases de datos relacionales



Bases de datos no relacionales



Bases de datos no relacionales



elasticsearch

Bases de datos no relacionales



Servicios



Auto
administrados



Administrados



Introducción a la bases de datos relacionales

Historia



No me hables de álgebra



$$\begin{aligned} & \text{... en paín, sandaani,} \\ & \log_2 45 = \log_{15^2} 45^2 = \log_{15^2} (15^2 \cdot 3^2) \\ & = \log_{15^2} 15^2 + \log_{15^2} 3^2 \\ & = 2 + \log_{15^2} 3^2 \\ & = 2 + \log_5 3^2 \\ & = 2 + 2 \log_5 3 \\ & = 2 + 2(0.477) \\ & = 2 + 0.954 \\ & = 2.954 \end{aligned}$$

Entidades

laptops



Atributos



Atributos

no de serie	color	año	pantalla
LKJ789JKAS	gris	2017	AX4829i
KCO3100KJH	negro	2019	AX4930i
NSDJOIH128	negro	2018	AX4930i
09KSIHBD71	gris	2017	AX4829i

Entidades débiles



Entidades débiles: identidad

Libros

id	título	...
LKJ789JKAS	Viaje al cent...	...
KCO3100KJH	El señor de
NSDJOIH128	De la tierra...	...
09KSIHBD71	Amor en tie...	...

Ejemplares

libro_id	localización	edición
LKJ789JKAS	pasillo 1	1
KCO3100KJH	pasillo 1	1
NSDJOIH128	pasillo 1	3
09KSIHBD71	pasillo 1	1

Entidades débiles: existencia

Libros

id	título	...
LKJ789JKAS	Viaje al cent...	...
KCO3100KJH	El señor de
NSDJOIH128	De la tierra...	...
09KSIHBD71	Amor en tie...	...

Ejemplares

id	localización	edición
JKE7823CLK	pasillo 1	1
JKFE1093JD	pasillo 1	1
82938ISHDIK	pasillo 1	3
838439JHDUI	pasillo 1	1

Diagrama ER: Platziblog

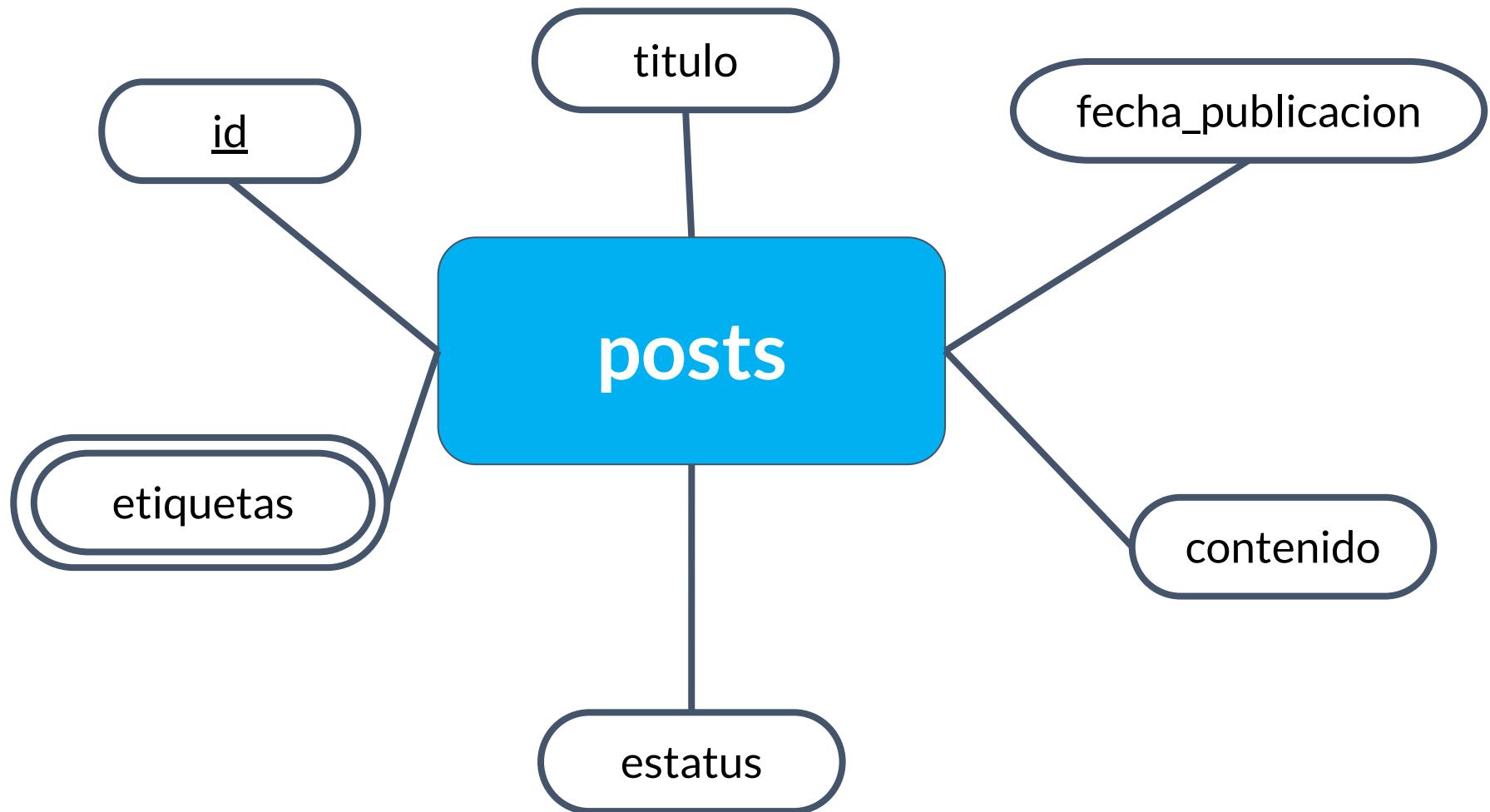
usuarios

comentarios

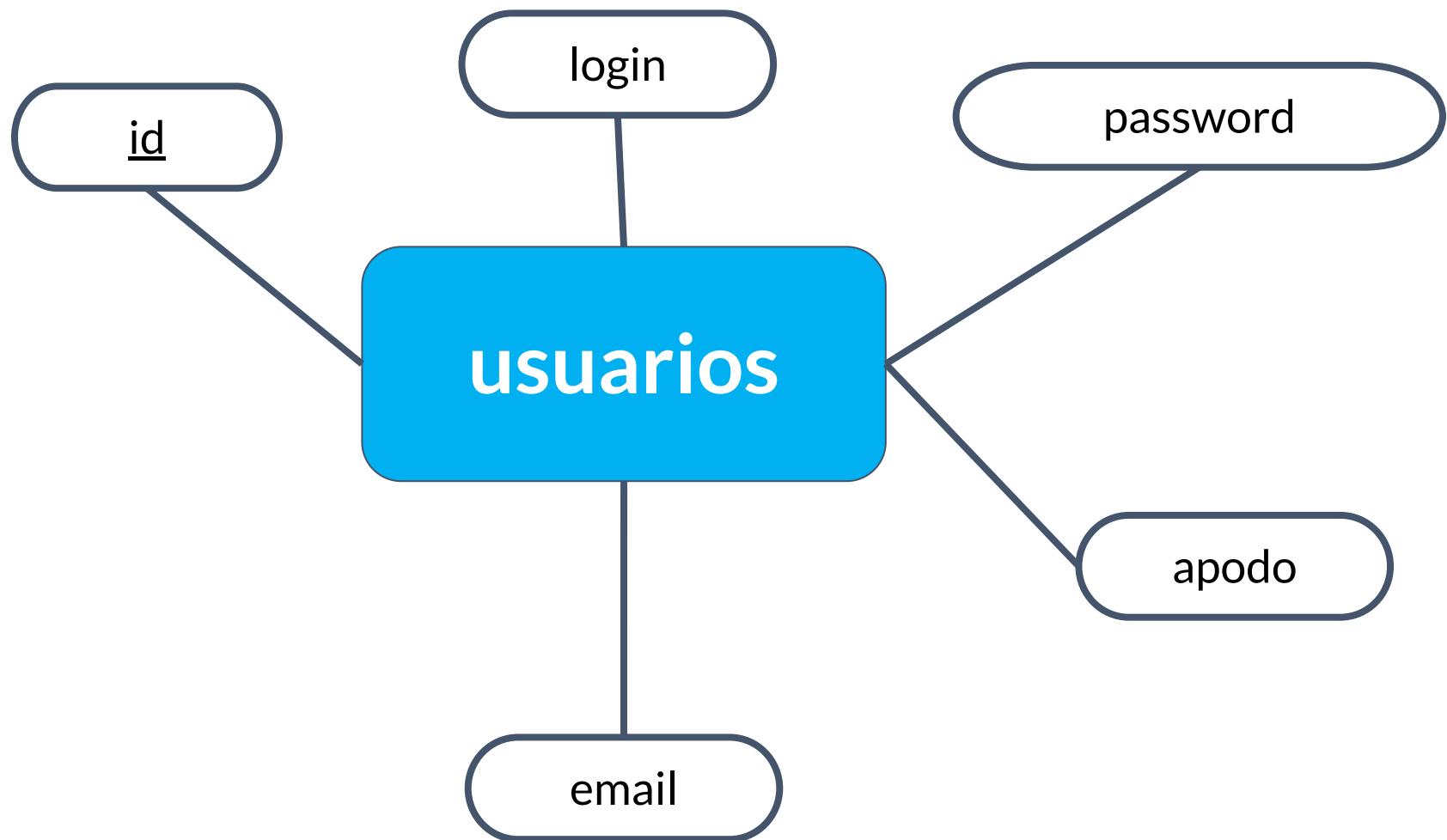
posts

categorías

Entidades Platziblog



Entidades Platziblog

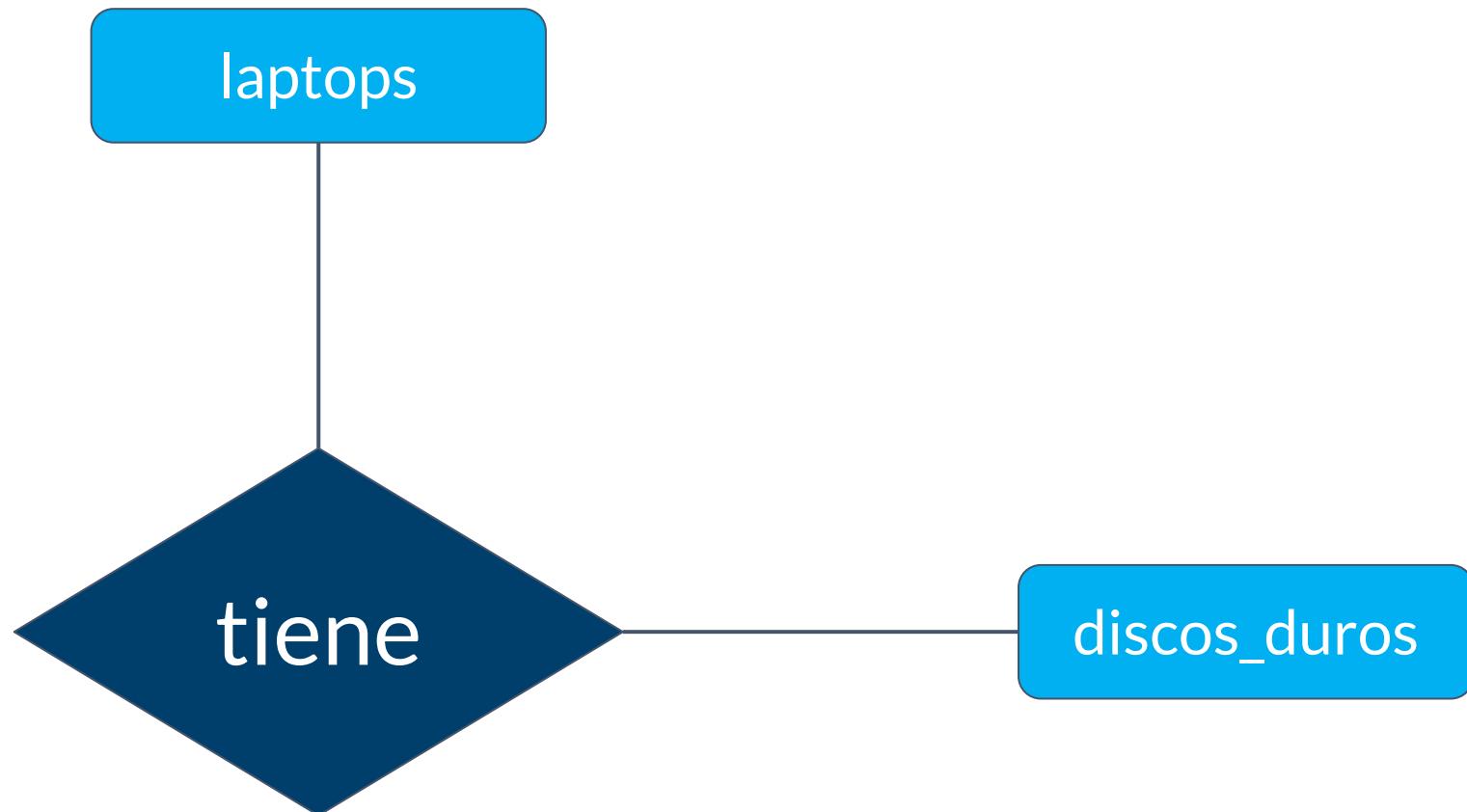


Relaciones

tiene



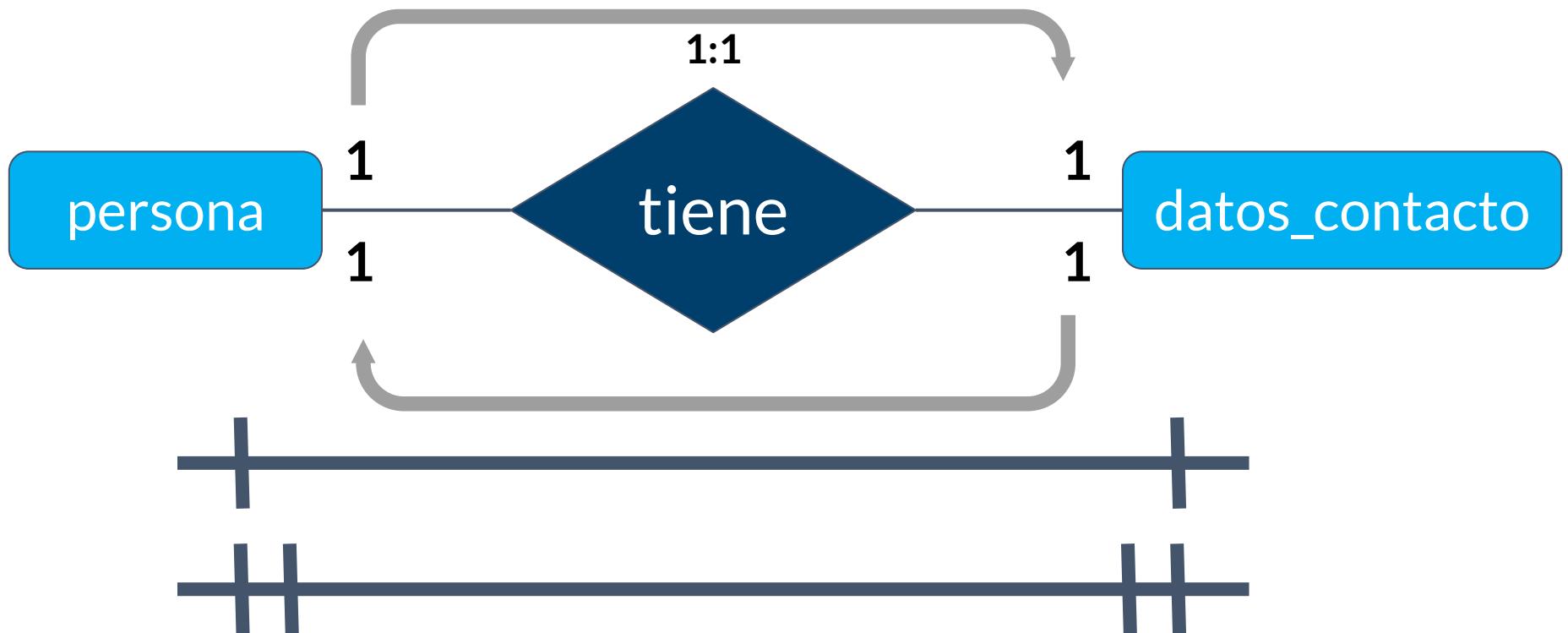
Relaciones



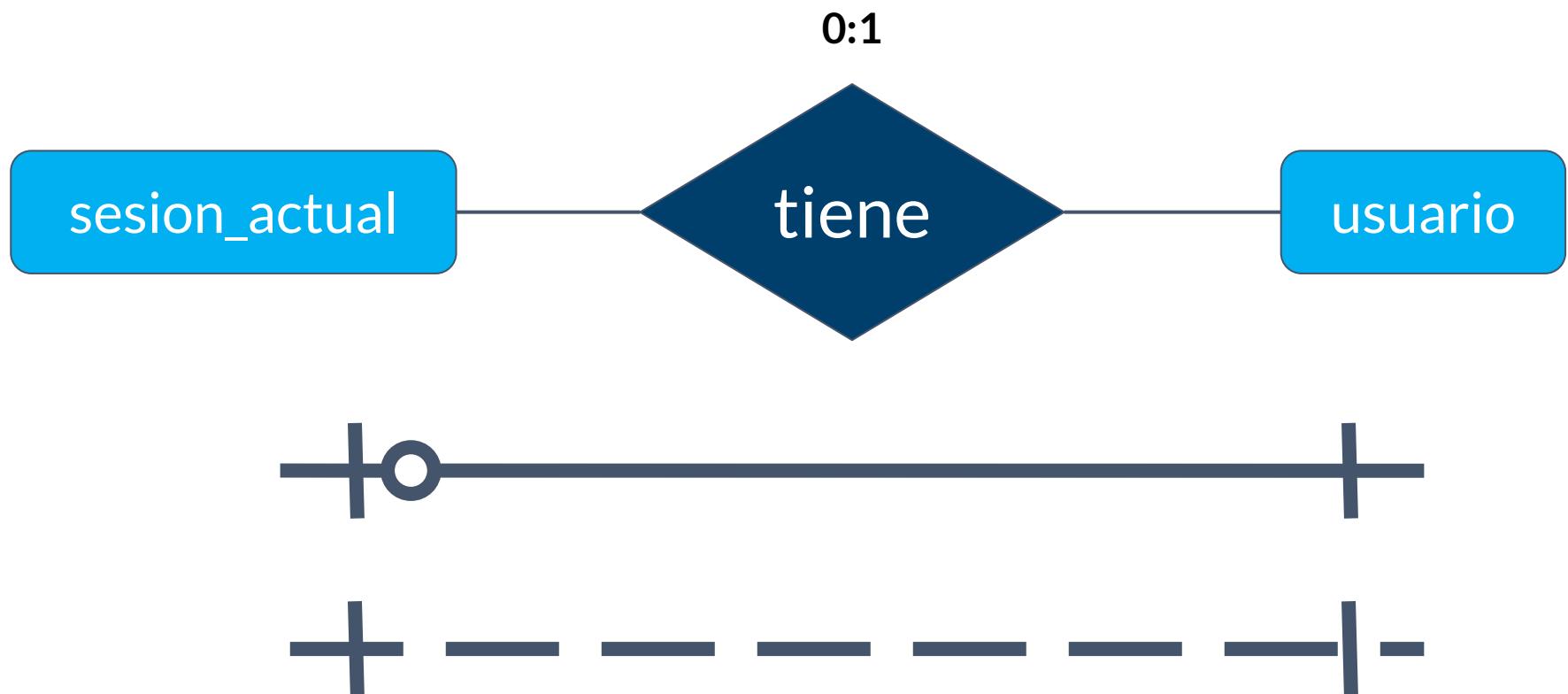
Cardinalidad



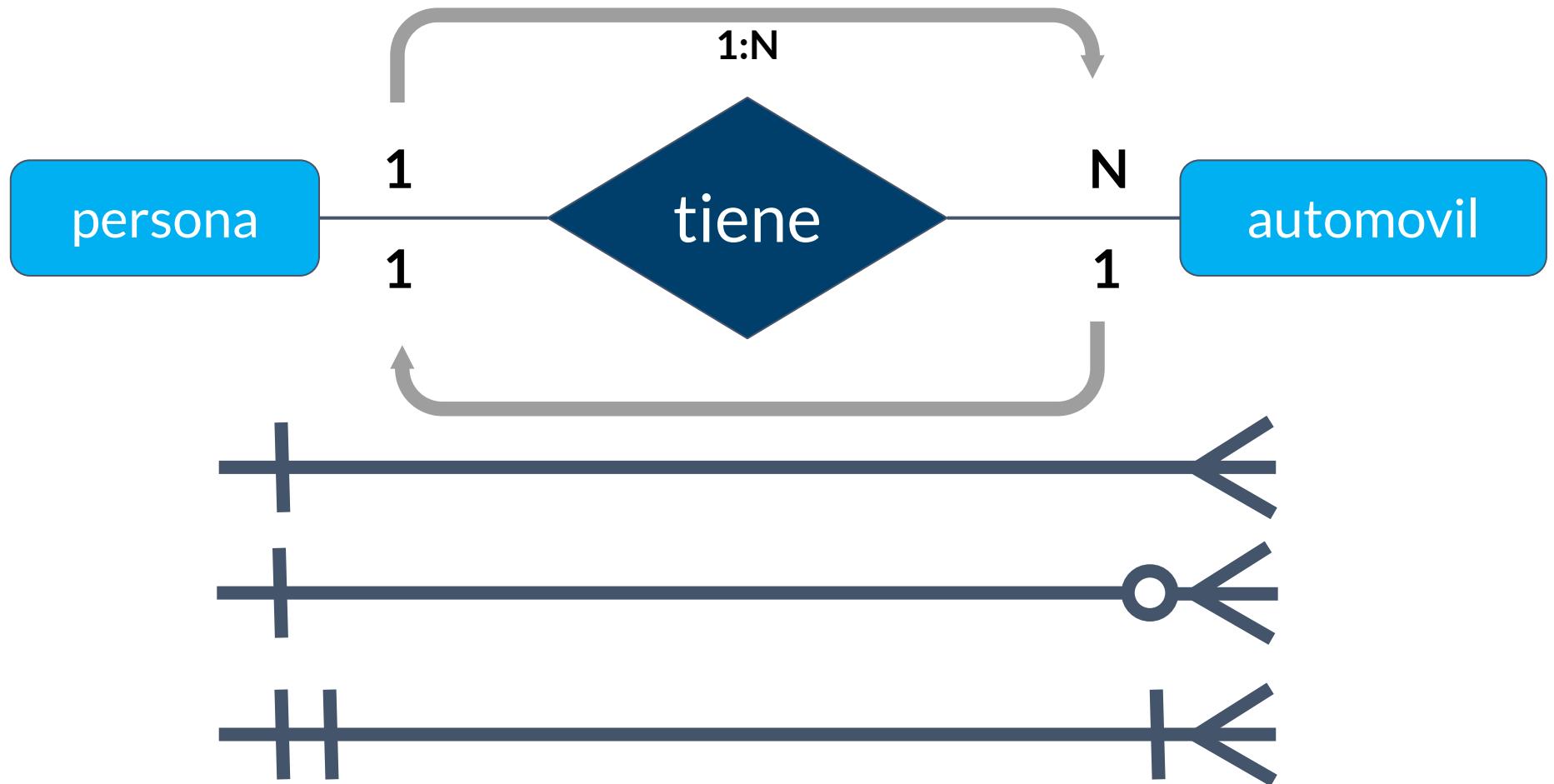
Cardinalidad: 1 a 1



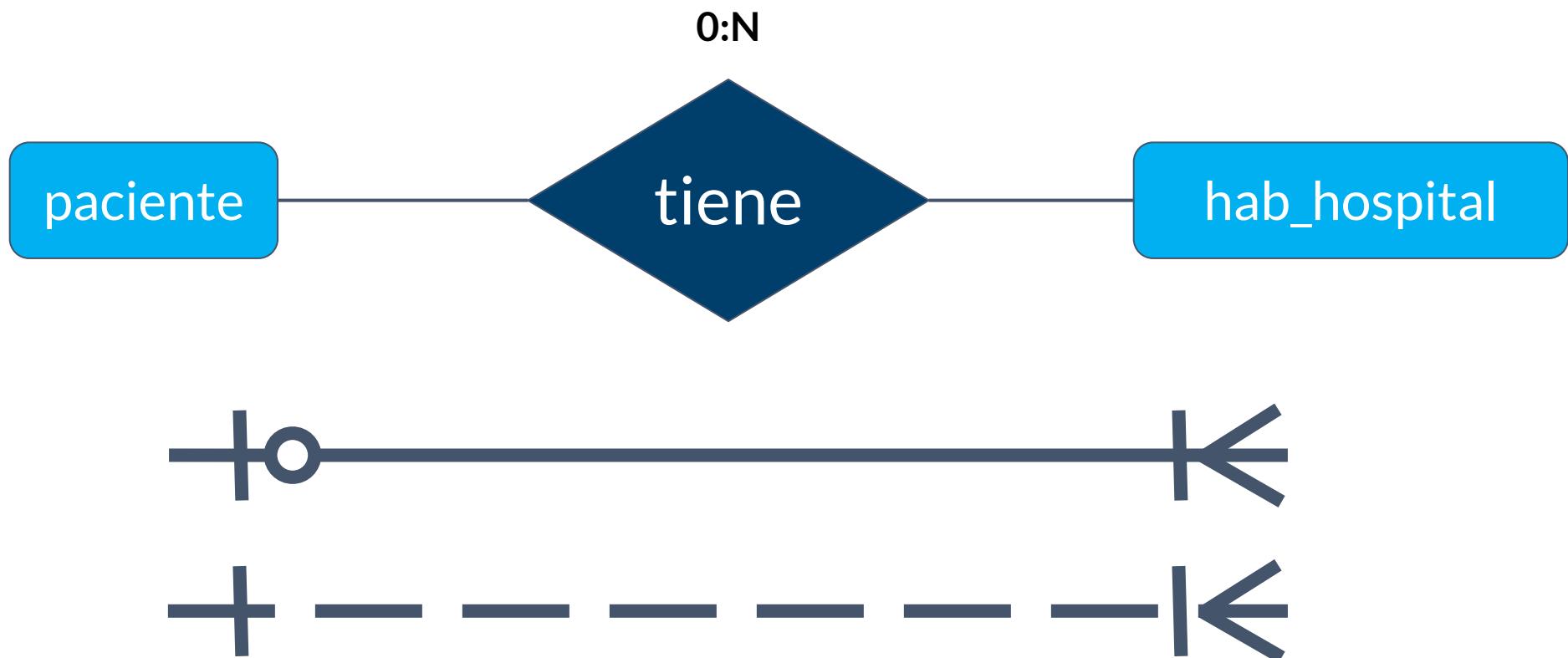
Cardinalidad: 0 a 1



Cardinalidad: 1 a N



Cardinalidad: 0 a N



Cardinalidad: N a N

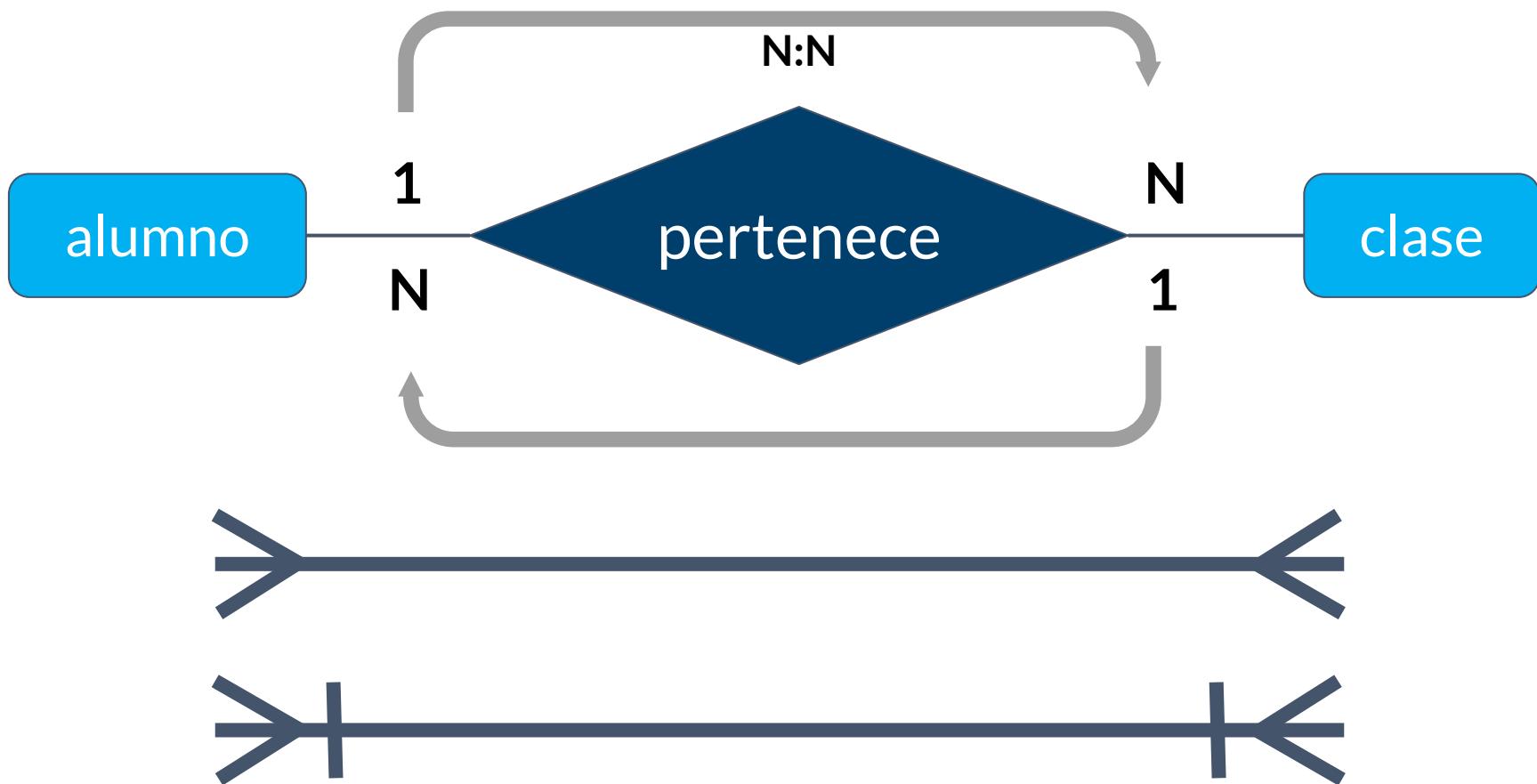


Diagrama ER

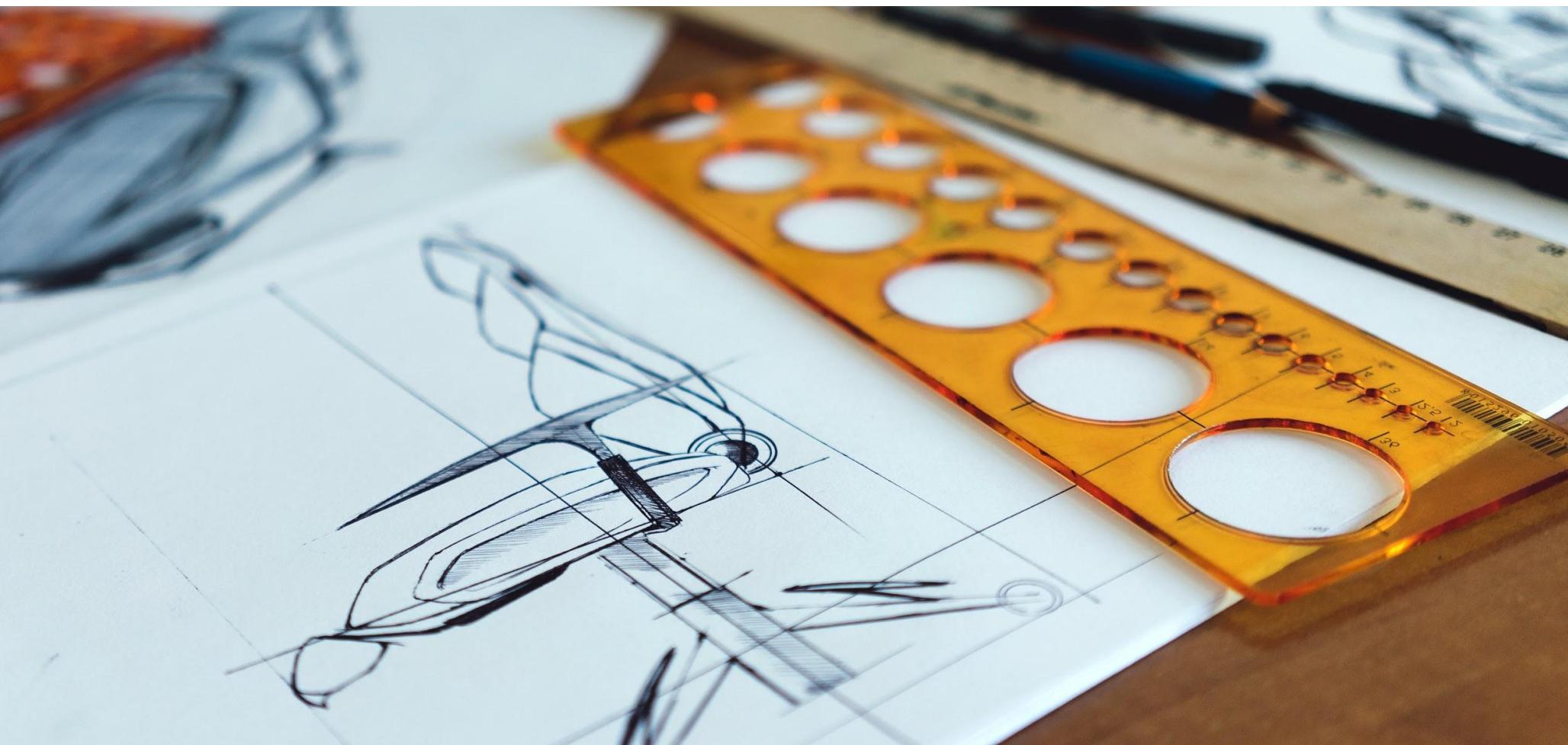


Diagrama ER: Platziblog

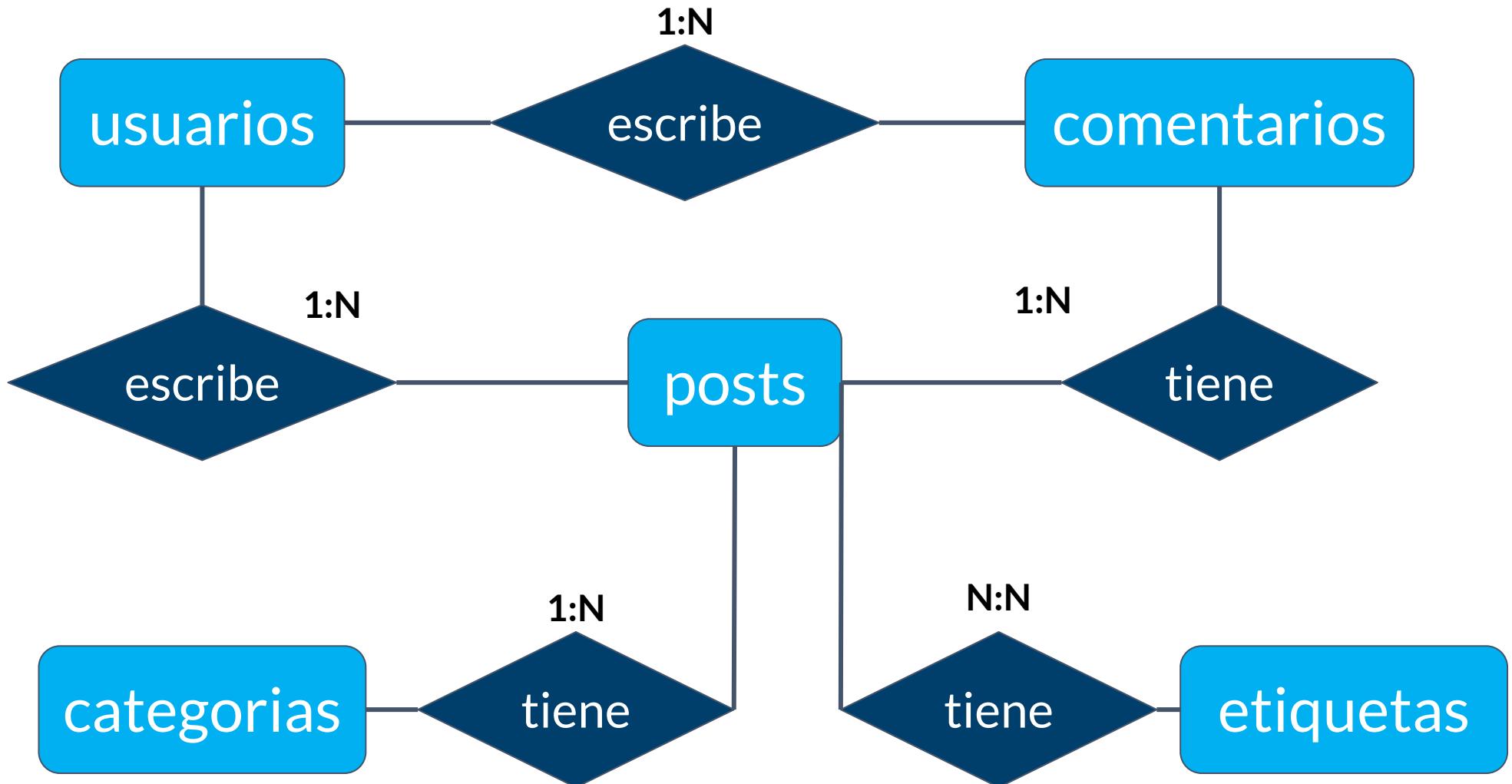
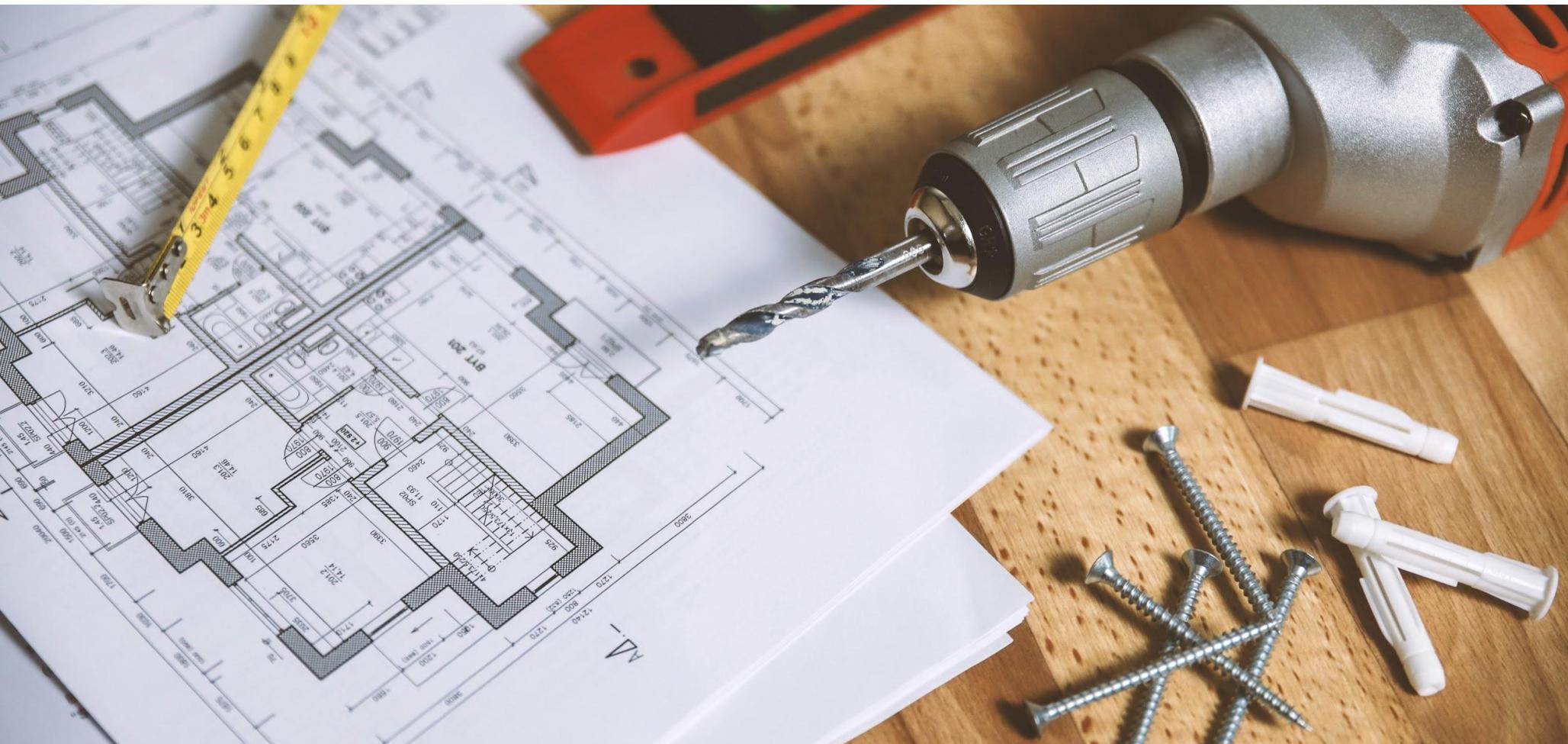


Diagrama Físico



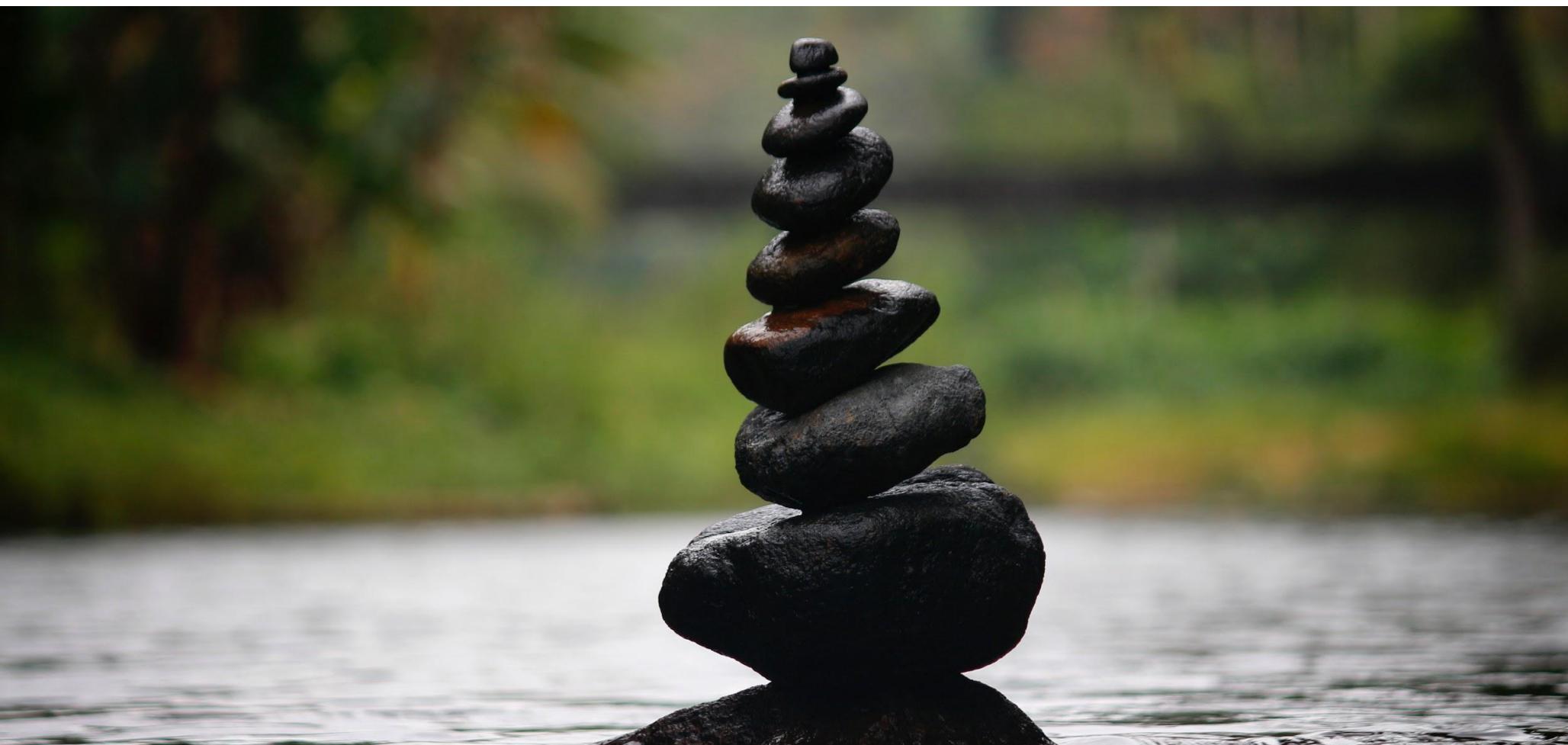
Tipos de dato

Texto	Números	Fecha/hora	Lógicos
CHAR(n)	INTEGER	DATE	BOOLEAN
VARCHAR(n)	BIGINT	TIME	
TEXT	SMALLINT	DATETIME	
	DECIMAL(n, s)	TIMESTAMP	
	NUMERIC (n, s)		

Constraints (Restricciones)

Constraint	Descripción
NOT NULL	Se asegura que la columna no tenga valores nulos
UNIQUE	Se asegura que cada valor en la columna no se repita
PRIMARY KEY	Es una combinación de NOT NULL y UNIQUE
FOREIGN KEY	Identifica de manera única una tupla en otra tabla
CHECK	Se asegura que el valor en la columna cumpla una condición dada
DEFAULT	Coloca un valor por defecto cuando no hay un valor especificado
INDEX	Se crea por columna para permitir búsquedas más rápidas

Normalización



Sin normalizar

alumno	nivel_curso	nombre_curso	materia_1	materia_2
Juanito	Maestría	Data engineering	MySQL	Python
Pepito	Licenciatura	Programación	MySQL	Python

Primera forma normal (1FN)

Atributos atómicos (Sin campos repetidos)

alumnos				
alumno_id	alumno	nivel_curso	nombre_curso	materia
1	Juanito	Maestría	Data engineering	MySQL
1	Juanito	Maestría	Data engineering	Python
2	Pepito	Licenciatura	Programación	MySQL
2	Pepito	Licenciatura	Programación	Python

Segunda forma normal (2FN)

Cumple 1FN y Cada campo de la tabla debe
depender de una clave única

alumnos			
alumno_id	alumno	nivel_curso	nombre_curso
1	Juanito	Maestría	Data engineering
2	Pepito	Licenciatura	Programación

materias		
materia_id	alumno_id	materia
1	1	MySQL
2	1	Python
3	2	MySQL
4	2	Python

Tercera forma normal (3FN)

Cumple 1FN y 2FN y los campos que NO son clave NO deben tener dependencias.

alumnos		
alumno_id	alumno	curso_id
1	Juanito	1
2	Pepito	2

cursos		
curso_id	nivel_curso	nombre_curso
1	Maestría	Data engineering
2	Licenciatura	Programación

materias		
materia_id	alumno_id	materia
1	1	MySQL
2	1	Python
3	2	MySQL
4	2	Python

Cuarta forma normal (4FN)

Cumple 1FN, 2FN y 3FN los
campos multivaluados se
identifican por una clave única.

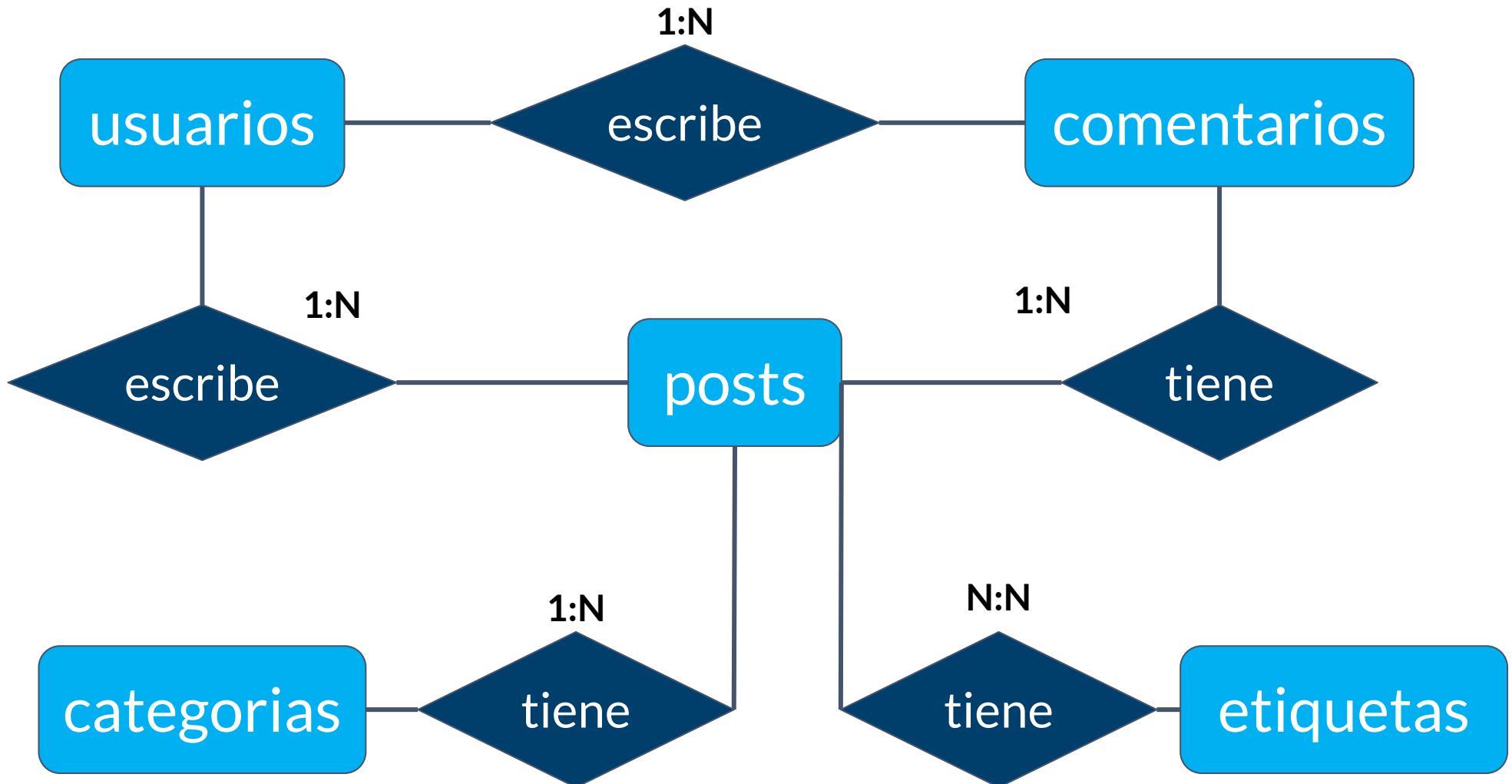
alumnos		
alumno_id	alumno	curso_id
1	Juanito	1
2	Pepito	2

cursos		
curso_id	nivel_curso	nombre_curso
1	Maestría	Data engineering
2	Licenciatura	Programación

materias	
materia_id	materia
1	MySQL
2	Python

materias_por_alumno		
mpa_id	materia_id	alumno_id
1	1	1
2	2	1
3	1	2
4	2	2

Diagrama ER: Platziblog



usuarios

id: INTEGER (PK)
login: VARCHAR(30) NN
password: VARCHAR(32) NN
nickname: VARCHAR(40) NN
email: VARCHAR(40) NN UNIQUE

comentarios

id: INTEGER (PK)
comentario: TEXT
usuarios_id: INTEGER FK
posts_id: INTEGER FK

posts

id: INTEGER (PK)
titulo: VARCHAR (150)
fecha_publicacion: TIMESTAMP
contenido: TEXT
estatus: CHAR(8) CHECK(IN ('activo', 'inactivo')
usuarios_id: INTEGER FK
categorias_id: INTEGER FK

categorias

id: INTEGER (PK)
categoria: VARCHAR(30)

etiquetas

id: INTEGER (PK)
nombre_etiqueta VARCHAR(30)

posts

id: INTEGER (PK)

titulo: VARCHAR (150)

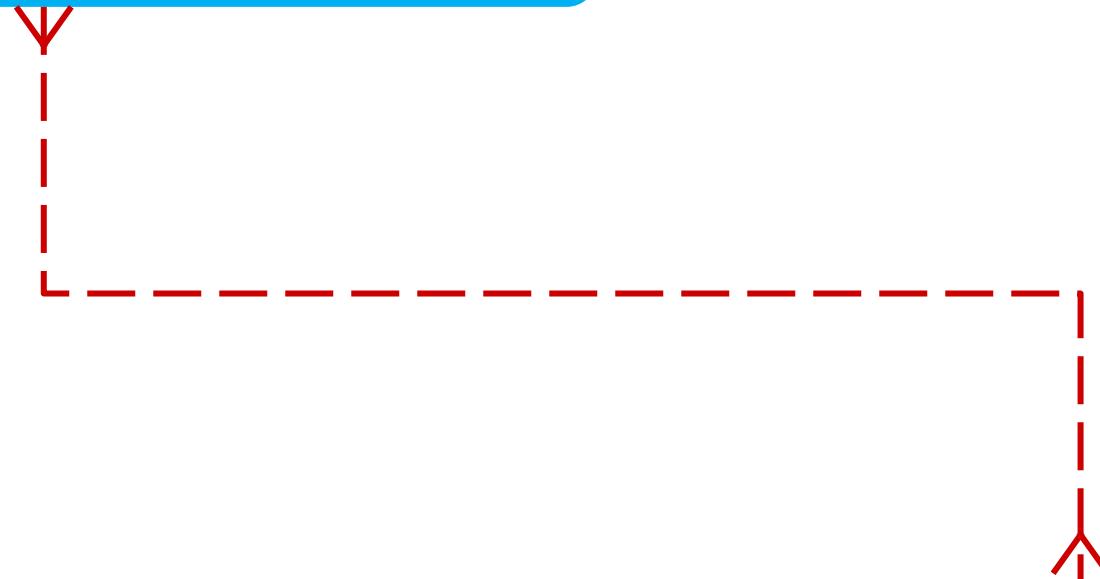
fecha_publicacion: TIMESTAMP

contenido: TEXT

estatus: CHAR(8) CHECK(IN ('activo', 'inactivo')

usuarios_id: INTEGER FK

categorias_id: INTEGER FK



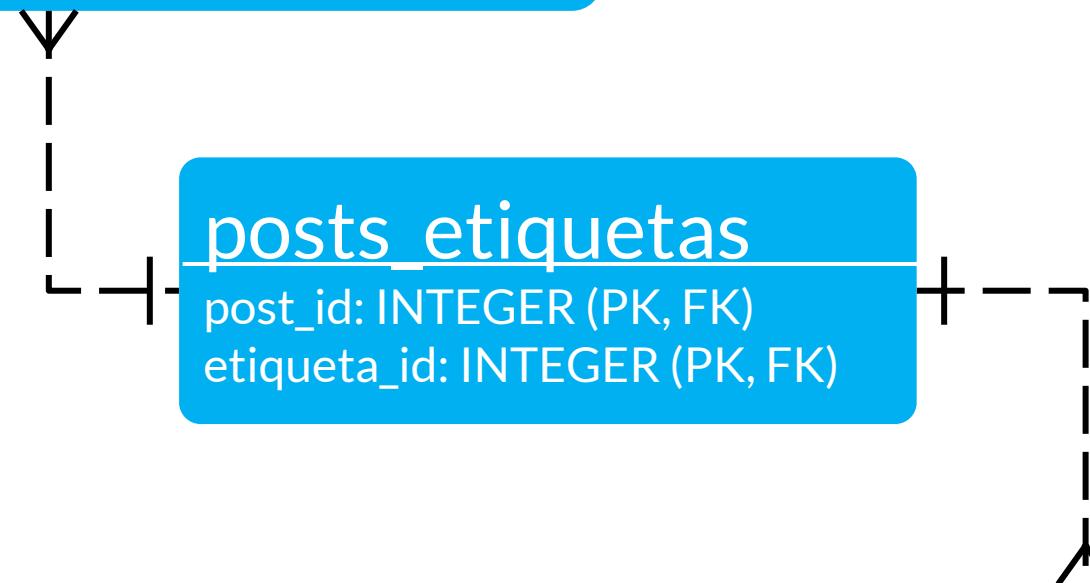
etiquetas

id: INTEGER (PK)

nombre_etiqueta VARCHAR(30)

posts

id: INTEGER (PK)
titulo: VARCHAR (150)
fecha_publicacion: TIMESTAMP
contenido: TEXT
estatus: CHAR(8) CHECK(IN ('activo', 'inactivo')
usuarios_id: INTEGER FK
categorias_id: INTEGER FK



etiquetas

id: INTEGER (PK)
nombre_etiqueta VARCHAR(30)

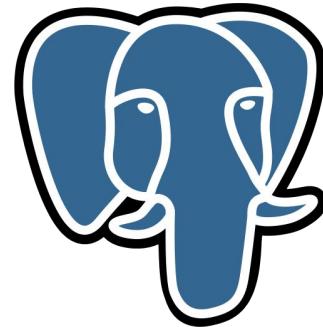
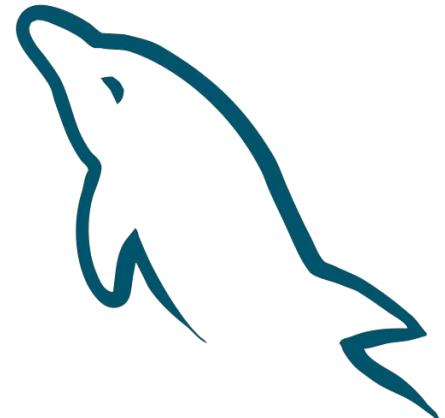
RDBMS o cómo hacer lo anterior de manera práctica

RDB ¿Qué?



RDBMS

Relational
Database
Management
System



ORACLE®

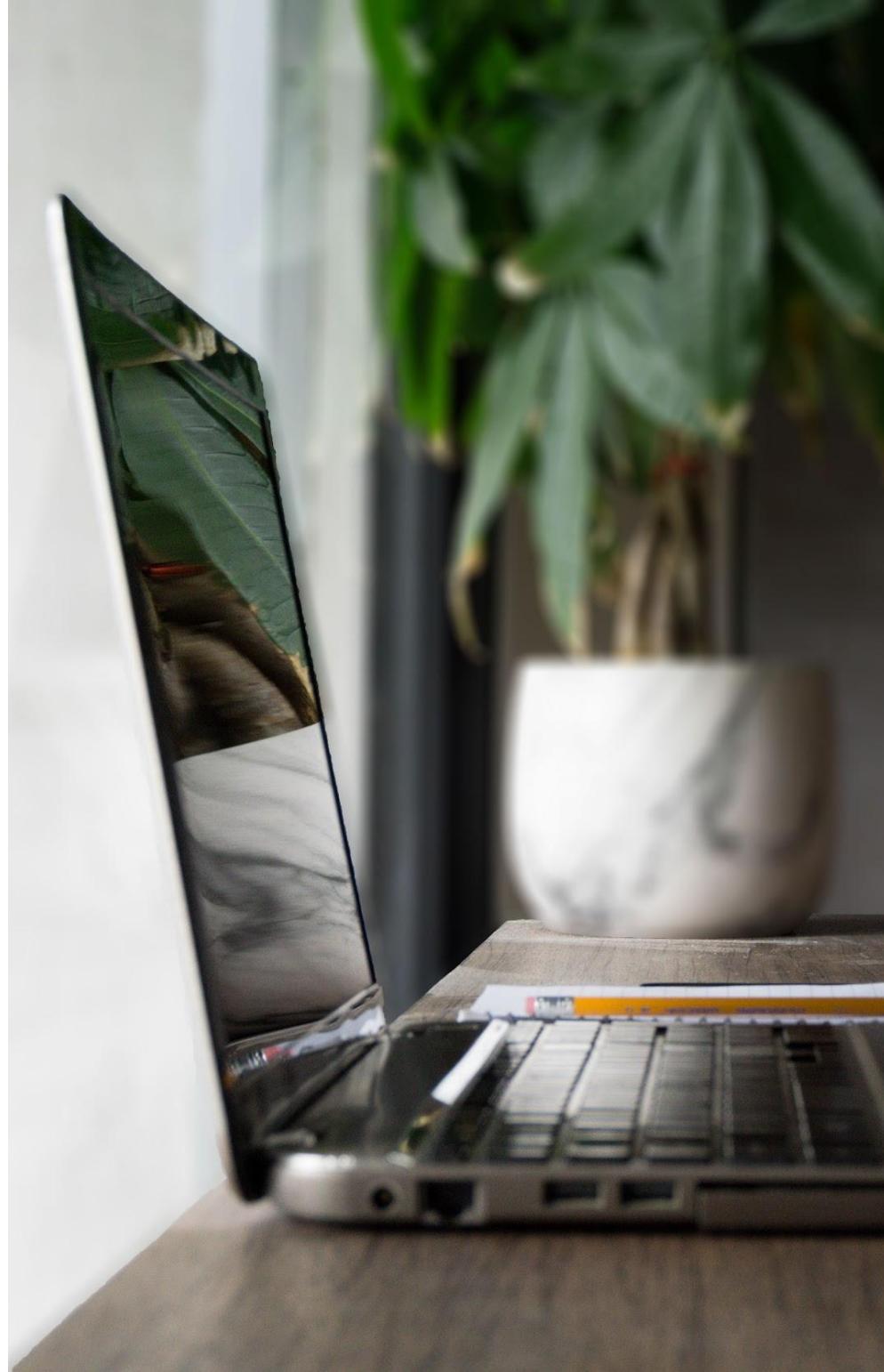
Instalación local de un RDBMS (Mac)



Descargar el instalador

<https://dev.mysql.com/downloads/mysql/5.6.html>

Instalación local de un RDBMS (Windows)



Descargar el instalador

<https://dev.mysql.com/downloads/mysql/5.6.html>

Servicios administrados



Creadores gráficos

```
te['sort_order'],
rror'];

{
_order'];

C, $quotes);

g_methods'] = $quotes;
'] = $address;

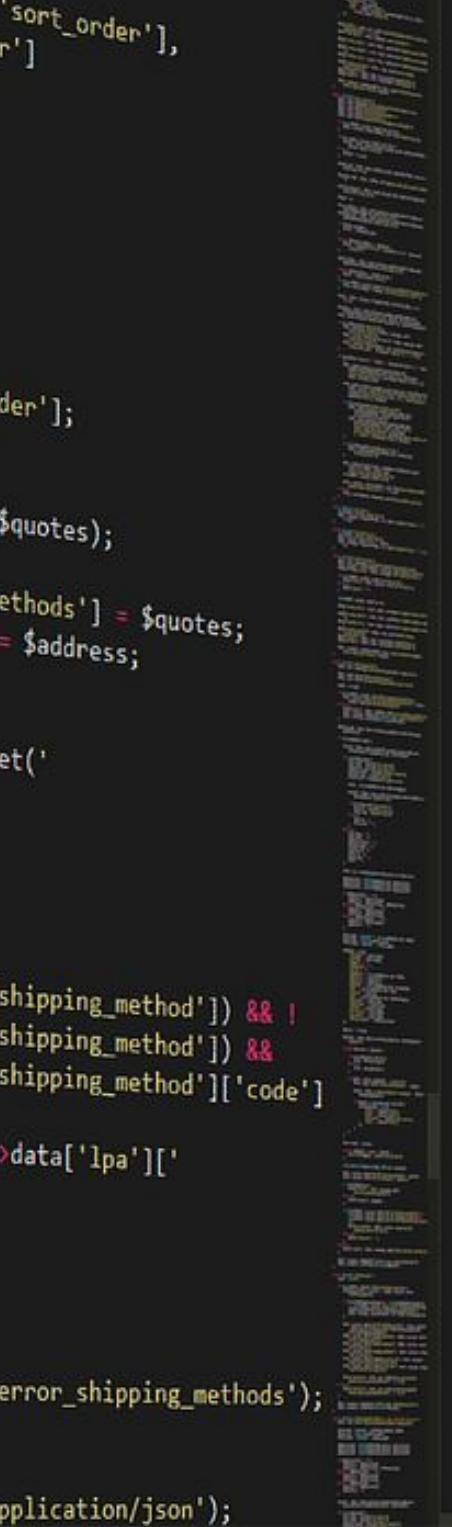
->get('

][['shipping_method']] && !
][['shipping_method']] &&
][['shipping_method']]['code']

on->data['lpa']['

t('error_shipping_methods');

: application/json');
```



```
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
` || (this.paused = function()
if (this.$element.find('.next',
this.$element.trigger($.support
} cycle(true)

this.interval = clearInterval(th
return this
}

Carousel.prototype.next = function()
if (this.sliding) return
return this.slide('next')

Carousel.prototype.prev = function()
if (this.sliding) return
return this.slide('prev')

Carousel.prototype.slide = function()
var $active   = this.$element.fi
var $next    = next || this.get
var isCycling = this.interval
var direction = type == 'next' ?
var fallback = type == 'next' ?
var that     = this

if (!$next.length) {
  if (!this.options.wrap) return
  $next = this.$element.find('.i
}

if ($next.hasClass('active')) re
var relatedTarget = $next[0]
var slideEvent = $.Event('slide
  relatedTarget: relatedTarget,
  direction: direction
})
this.$element.trigger(slideEvent)
```

Descargar el instalador

<https://dev.mysql.com/downloads/workbench/>

SQL hasta en la sopa

Historia SQL



SQL

Structured

Query

Language

NOSQL

Not
Only
Structured
Query
Language

SQL Hoy



DDL

Data Definition Language

Create

Alter

Drop

Create

Database

Table

View

Create Database

```
CREATE DATABASE test_db;
```

```
USE test_db;
```

Create Table

```
CREATE TABLE people (
    person_id int,
    last_name varchar(255),
    first_name varchar(255),
    address varchar(255),
    city varchar(255)
) ;
```

Create View

```
CREATE VIEW v_brasil_customers AS  
    SELECT customer_name,  
contact_name  
    FROM customers  
    WHERE country = "Brasil";
```

Alter Table

```
ALTER TABLE people  
ADD date_of_birth date;
```

```
ALTER TABLE people  
ALTER COLUMN date_of_birth year;
```

```
ALTER TABLE people  
DROP COLUMN date_of_birth;
```

Drop

```
DROP TABLE people;
```

```
DROP DATABASE test_db;
```

DML

Data Manipulation Language

Insert

Update

Delete

Select

Insert

```
INSERT INTO people (last_name,  
first_name, address, city)  
VALUES ('Hernández', 'Laura',  
'Calle 21', 'Monterrey');
```

Update

```
UPDATE people  
SET last_name = 'Chávez', city= 'Mérida'  
WHERE person_id = 1;
```

```
UPDATE people  
SET first_name = 'Juan'  
WHERE city = 'Mérida';
```

```
UPDATE people  
SET first_name = 'Juan';
```

Delete

```
DELETE FROM people  
WHERE person_id = 1;
```

```
DELETE FROM people;
```

Select

```
SELECT fist_name, last_name  
FROM people;
```



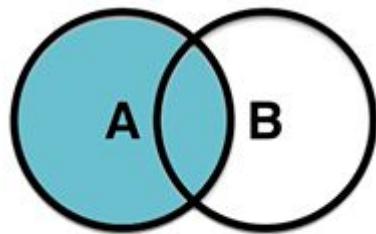
¿Qué tan standard es?

Estructura básica de un Query

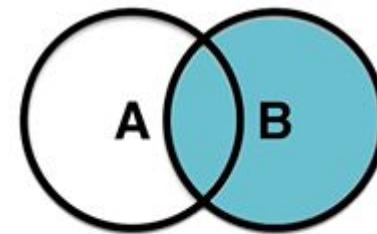
```
SELECT city, count(*) AS total  
FROM people  
WHERE active = true  
GROUP BY city  
ORDER BY total DESC  
HAVING total >= 2;
```

JOIN

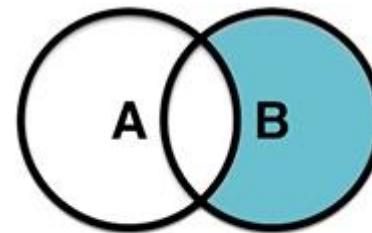
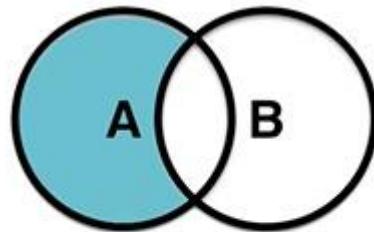
Diferencia



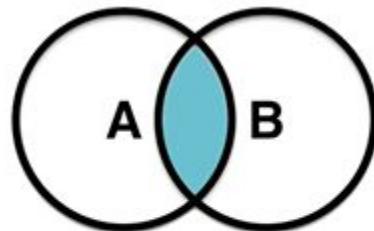
LEFT JOIN



RIGHT JOIN

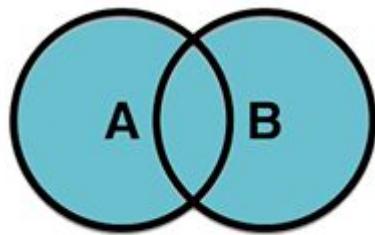


JOIN



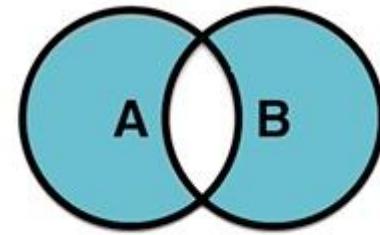
Intersección

INNER JOIN



Unión

OUTER JOIN



Diferencia simétrica

De pregunta a Query

Lo que quieres mostrar = SELECT

De donde voy a tomar los datos = FROM

Los filtros de los datos que quieres mostrar = WHERE

Los rubros por los que me interesa agrupar la información =
GROUP BY

El orden en que quiero presentar mi información ORDER BY

Los filtros que quiero que mis datos agrupados tengan HAVING

Introducción a las bases de datos NO relacionales

¿Qué son las bases de datos no relacionales?

```
6-03T18:42:18.018", "deltaStartTimeMillis": "14",  
"method": "handle", "sessionID": "14",  
"handler": "RequestHandler", "durationMillis": "14",  
"message": "Duration Log", "class": "com.org",  
"analyze": "webParams": "null", "sessionID": "14",  
}{"timestamp": "2017-06-03T18:43:335.030", "de  
rtda_new.json", "class": "com.orgmanager.han  
, "level": "INFO", "sizeChars": "48455", "mes  
bfa8-4e7d-8047-498454af885d", "sessionID": "14",  
"timestamp": "2017-06-03T18:46:921.000", "de  
r.handlers.RequestHandler", "method": "handl  
message": "Duration Log", "durationMillis": "14",  
"on/file", "webParams": "file=chartdata_new.j  
19e2-4a60-88d7-6ead86e273d1", "sessionID": "14",  
{"timestamp": "2017-06-03T18:42:18.018", "de  
r.handlers.RequestHandler", "method": "handl  
message": "Duration Log", "durationMillis": "56",  
"analyze": "webParams": "null", "class": "com.org",  
fd8-46ac-9745-839146a20f09", "sessionID": "14",  
} {"timestamp": "2017-06-03T18:43:335.030", "de  
tdata_new.json", "class": "com.orgmanager.han  
, "level": "INFO", "sizeChars": "48455", "mes  
bfa8-4e7d-8047-498454af885d", "sessionID": "14",  
"timestamp": "2017-06-03T18:46:921.000", "de  
r.handlers.RequestHandler", "method": "handl
```



- Clave - valor**

- Basadas en documentos**

- Basadas en grafos**

- En memoria**

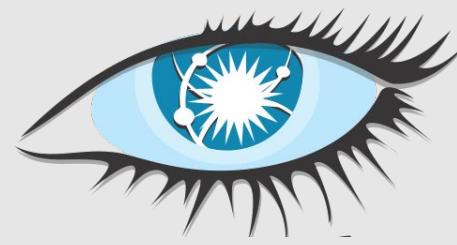
- Optimizadas para búsquedas**

Clave - valor

Son ideales para almacenar y extraer datos con una clave única. Manejan los diccionarios de manera excepcional.



DynamoDB



Cassandra

Basados en documentos

Son una implementación de clave valor que varía en la forma semiestructurada en que se trata la información. Ideal para almacenar datos JSON y XML.



MongoDB



Firestore

Basadas en grafos

Basadas en teoría de grafos sirven para entidades que se encuentran interconectadas por múltiples relaciones.
Ideales para almacenar relaciones complejas.

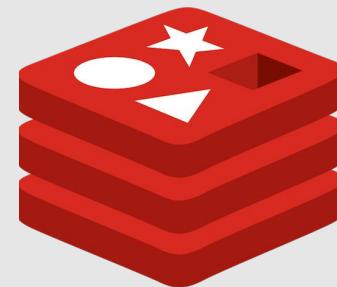


En memoria

Pueden ser de estructura variada, pero su ventaja radica en la velocidad, ya que al vivir en memoria la extracción de datos es casi inmediata.



Memcached



Redis

Optimizadas para búsqueda

Pueden ser de diversas estructuras, su ventaja radica en que se pueden hacer queries y búsquedas complejas de manera sencilla.



BigQuery



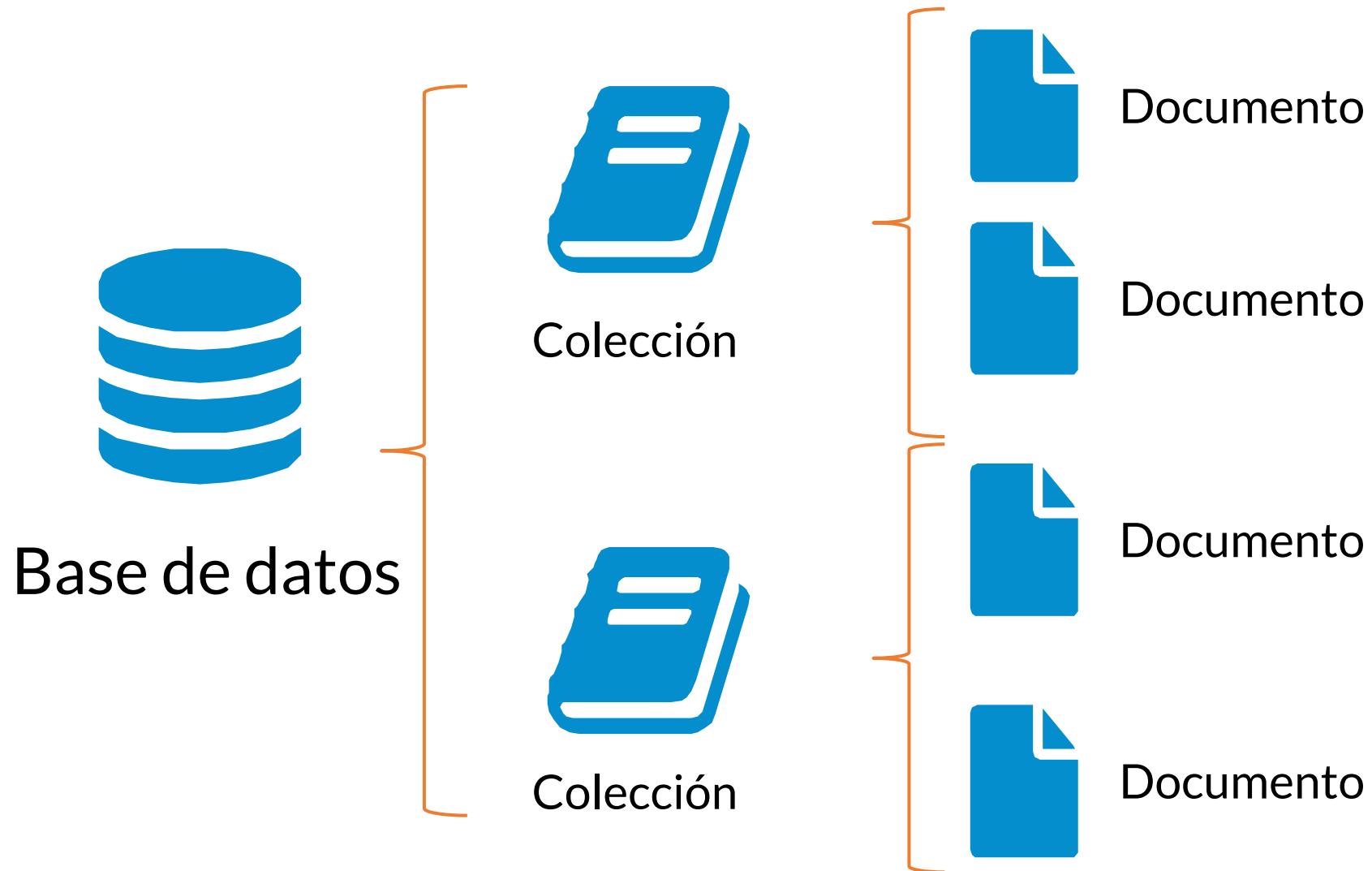
Elasticsearch

Servicios administrados



Firestore

Jerarquía de datos en firestore



Manejo de modelos de datos en bases de datos no relacionales

Top level collections



Creando documentos



Subcolecciones



Colecciones vs subcolecciones



Recreando Platziblog



usuarios

id: INTEGER (PK)
login: VARCHAR(30) NN
password: VARCHAR(32) NN
nickname: VARCHAR(40) NN
email: VARCHAR(40) NN UNIQUE

comentarios

id: INTEGER (PK)
comentario: TEXT
usuarios_id: INTEGER FK
posts_id: INTEGER FK

posts

id: INTEGER (PK)
titulo: VARCHAR (150)
fecha_publicacion: TIMESTAMP
contenido: TEXT
estatus: CHAR(8) CHECK(IN ('activo', 'inactivo')
usuarios_id: INTEGER FK
categorias_id: INTEGER FK

categorias

id: INTEGER (PK)
categoria: VARCHAR(30)

etiquetas

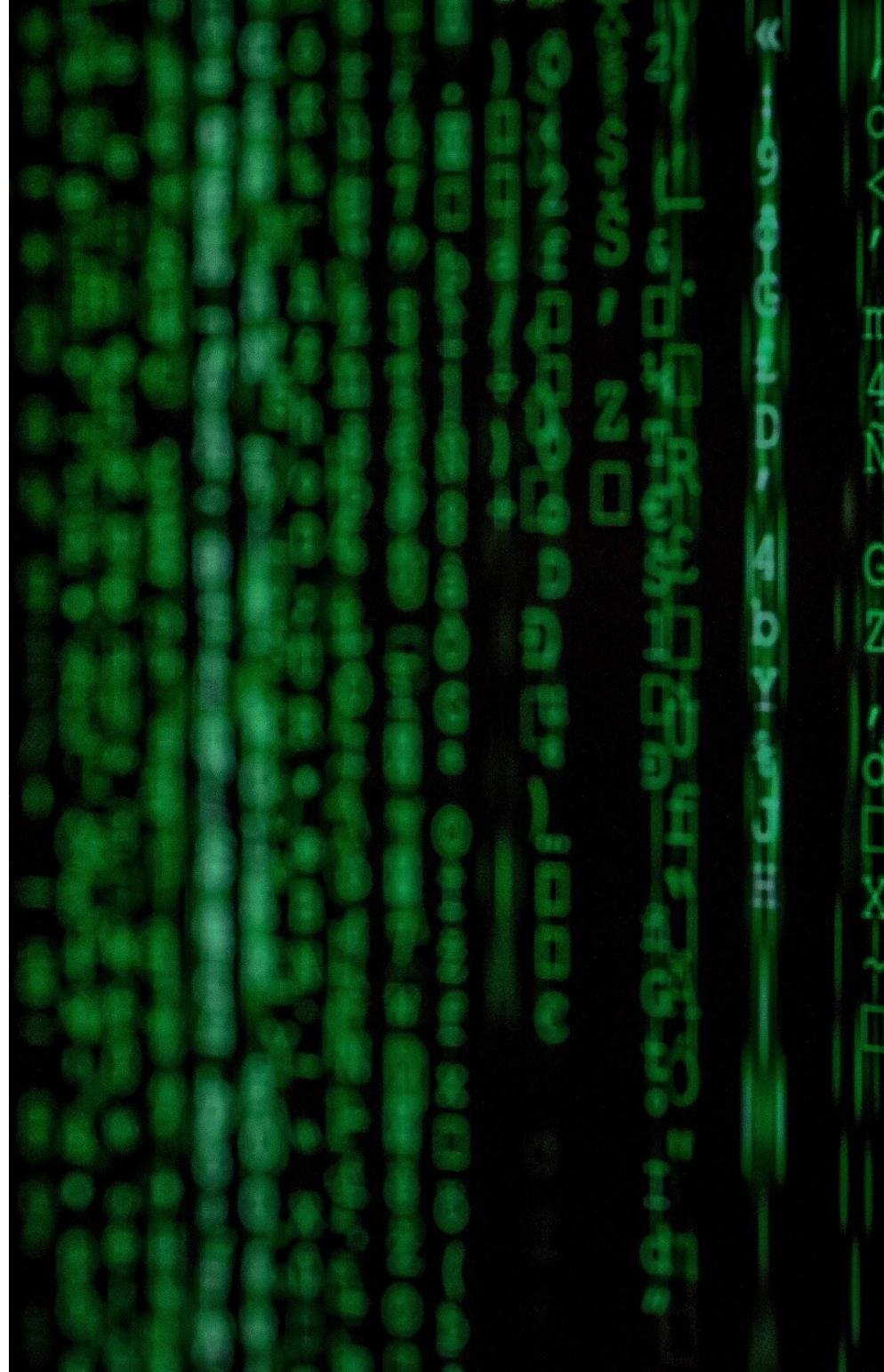
id: INTEGER (PK)
nombre_etiqueta VARCHAR(30)

Bases de Datos en la vida real

Diferentes soluciones de datos para aplicaciones modernas



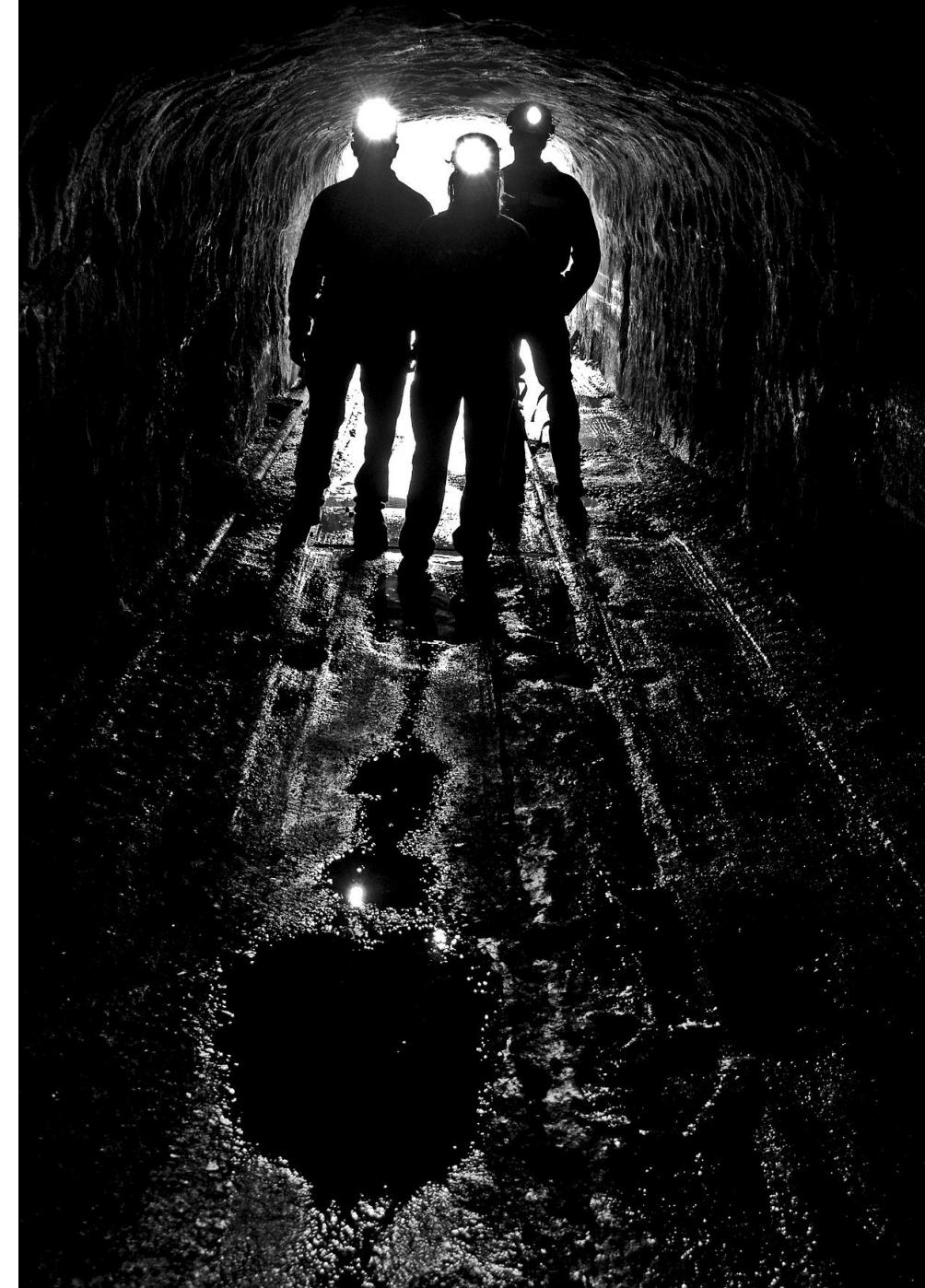
Big Data



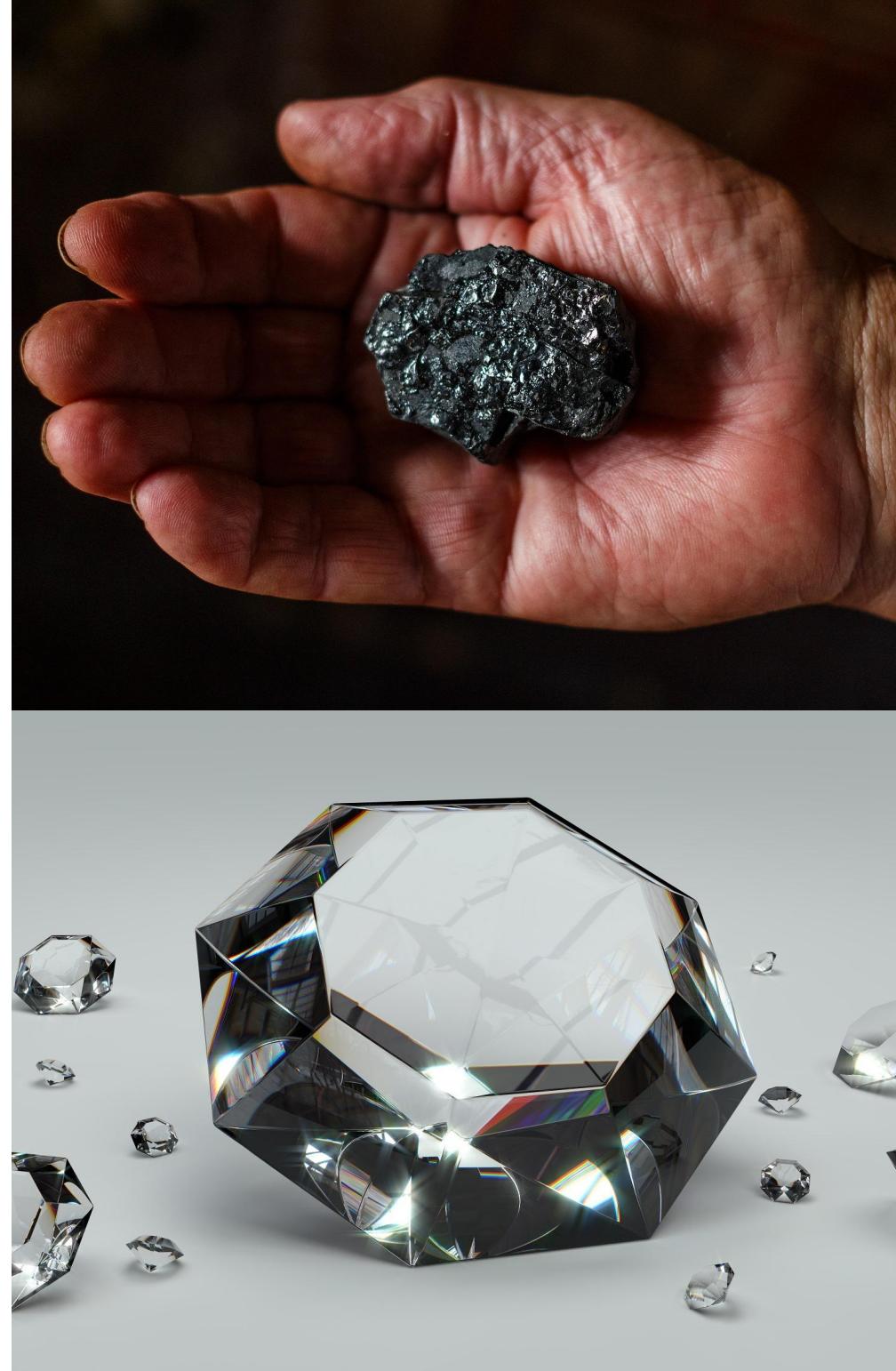
Data Warehouse



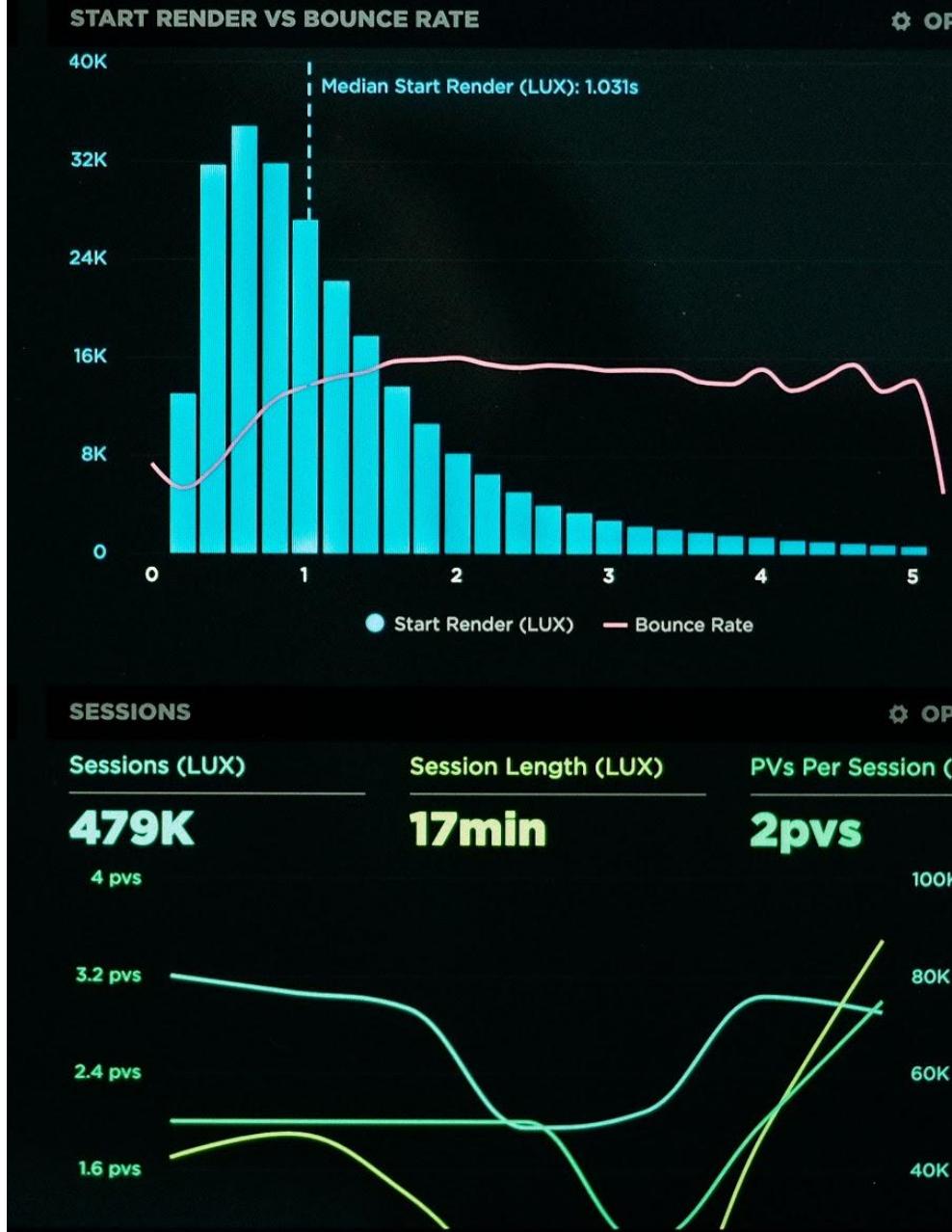
Data Mining



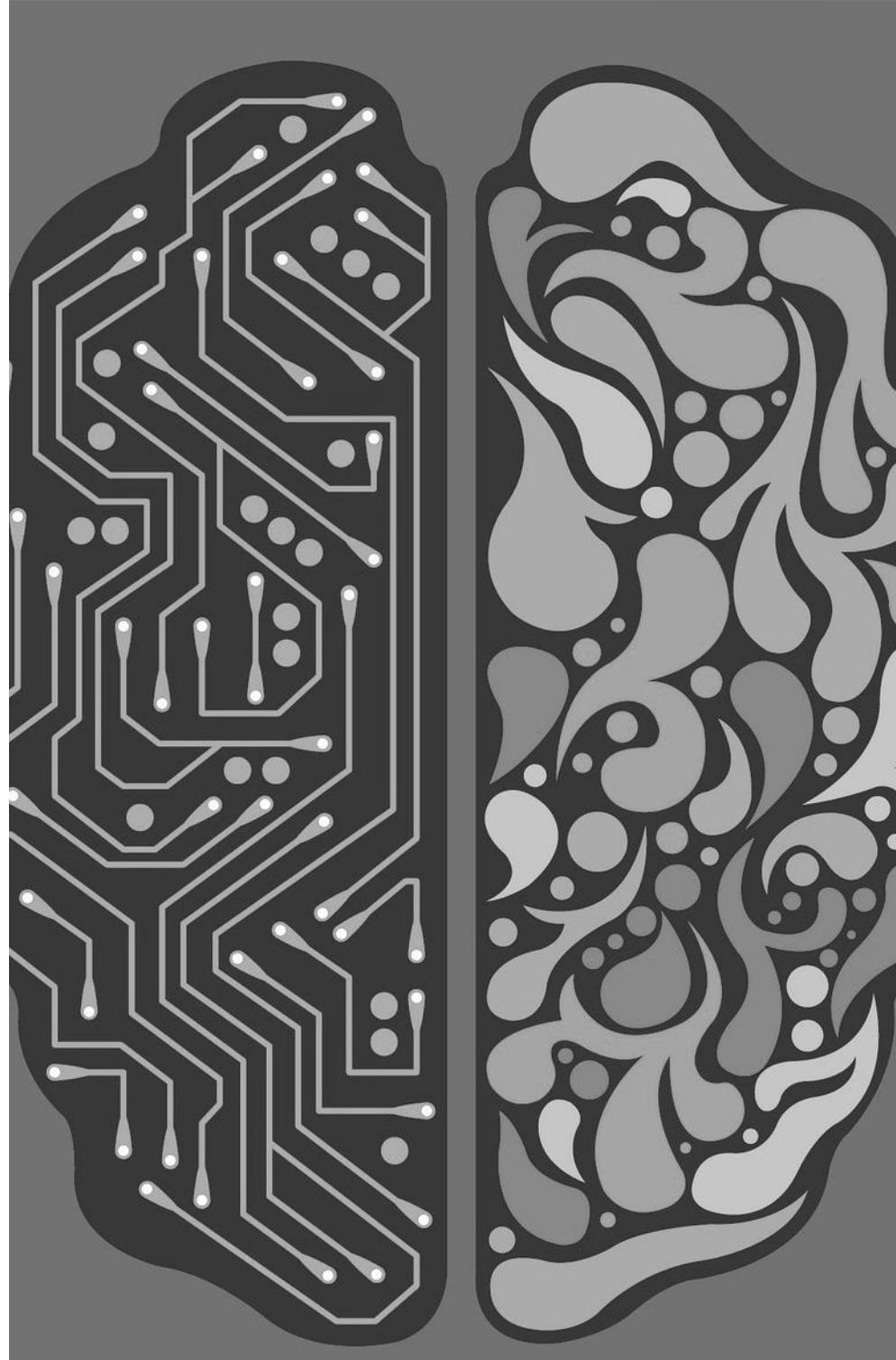
ETL



Business Intelligence



Machine Learning



Data Science



¿Por qué
aprender Bases
de Datos hoy?

