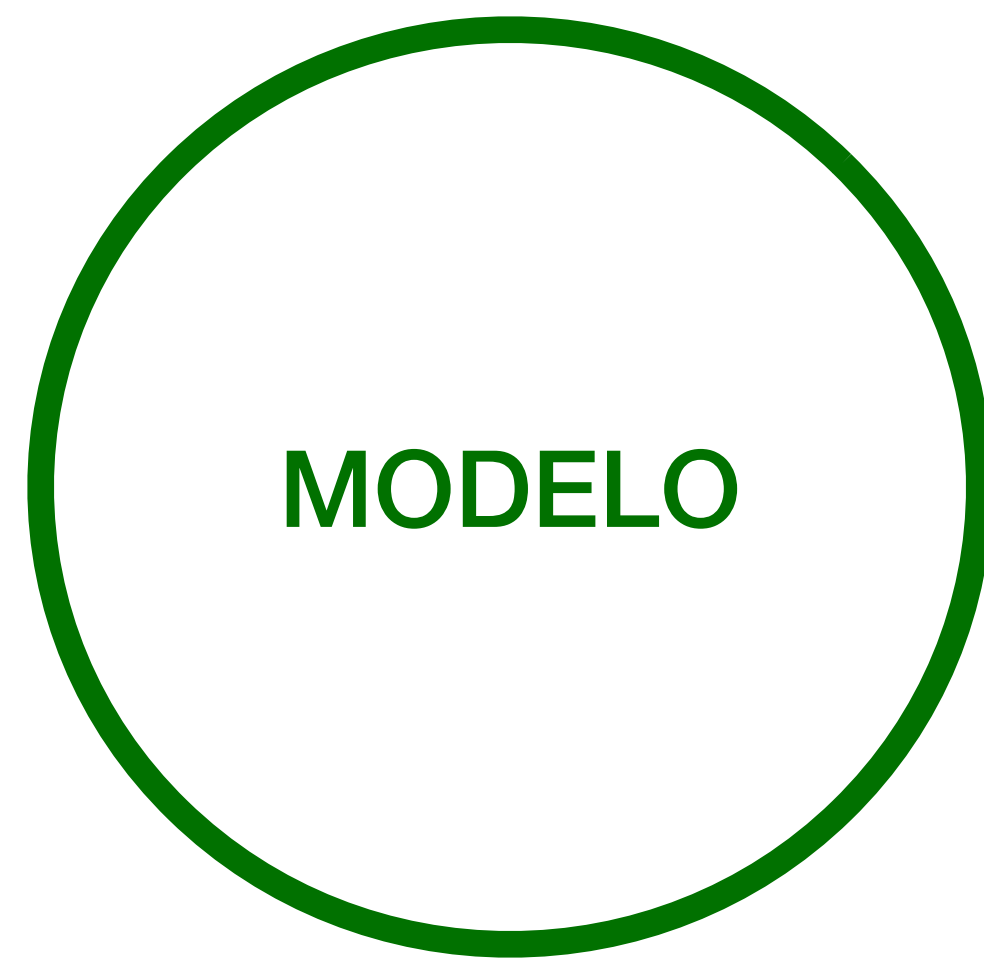


Ingeniería de Software 2021

Laboratorio 1 - Primera app

MVC: Model View Controller

Capas de la aplicación



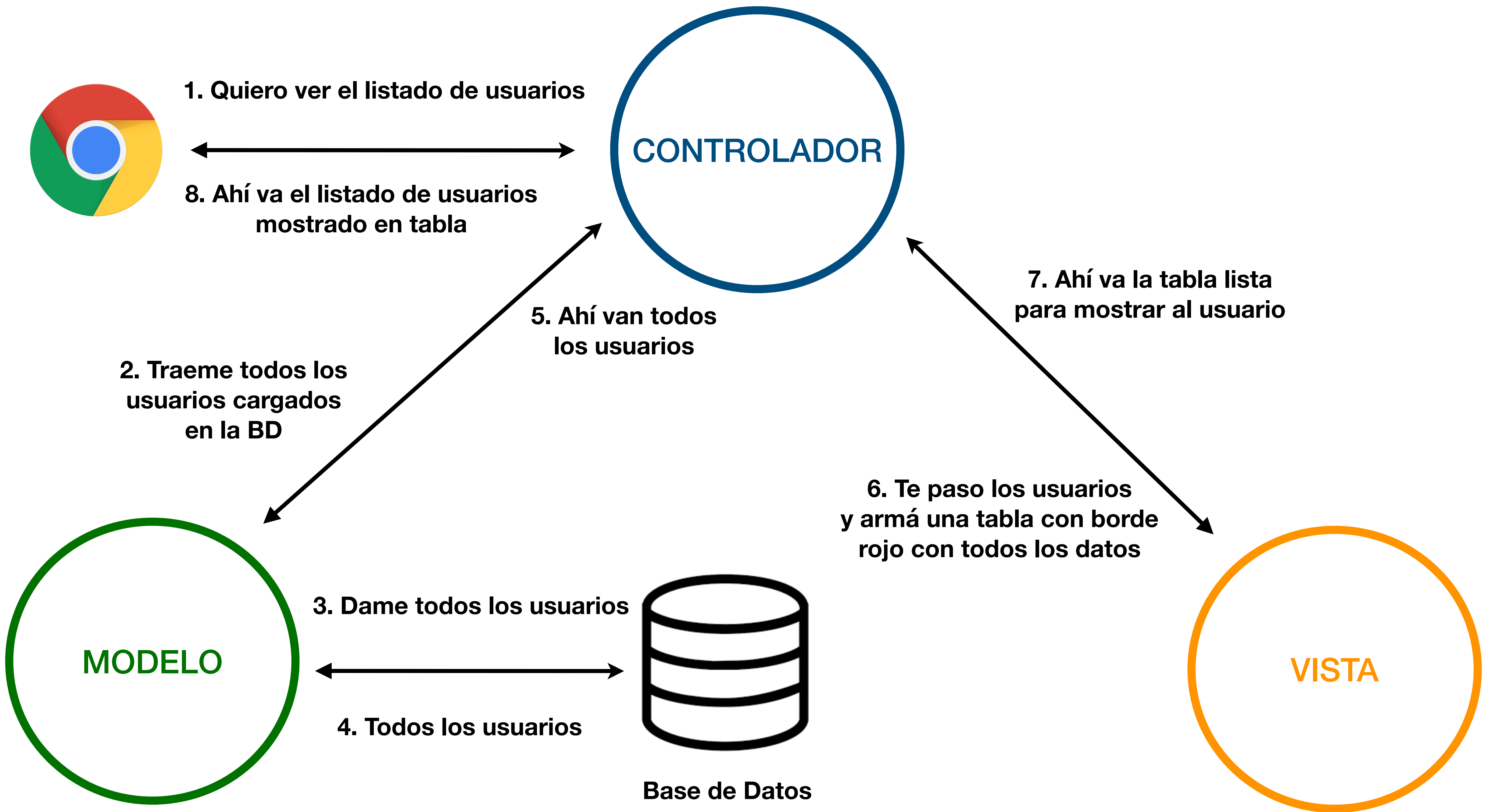
↓
Accesos a la BD



↓
**Funciona de intermediario entre
el modelo y las vistas**



↓
**Representación visual de la
información (usuario final)**





Practiquemos comandos...

- rails

- rails new lab1

- cd lab1

- rails

Practiquemos comandos...

- rails

- rails new lab1



BUNDLER: Se utiliza para instalar e incluir las gemas (gems) necesarias por la aplicación. Se especifican en el archivo Gemfile.

- cd lab1

- rails

Practiquemos comandos...

- rails

- rails new lab1



BUNDLER: Se utiliza para instalar e incluir las gemas (gems) necesarias por la aplicación. Se especifican en el archivo Gemfile.

- cd lab1

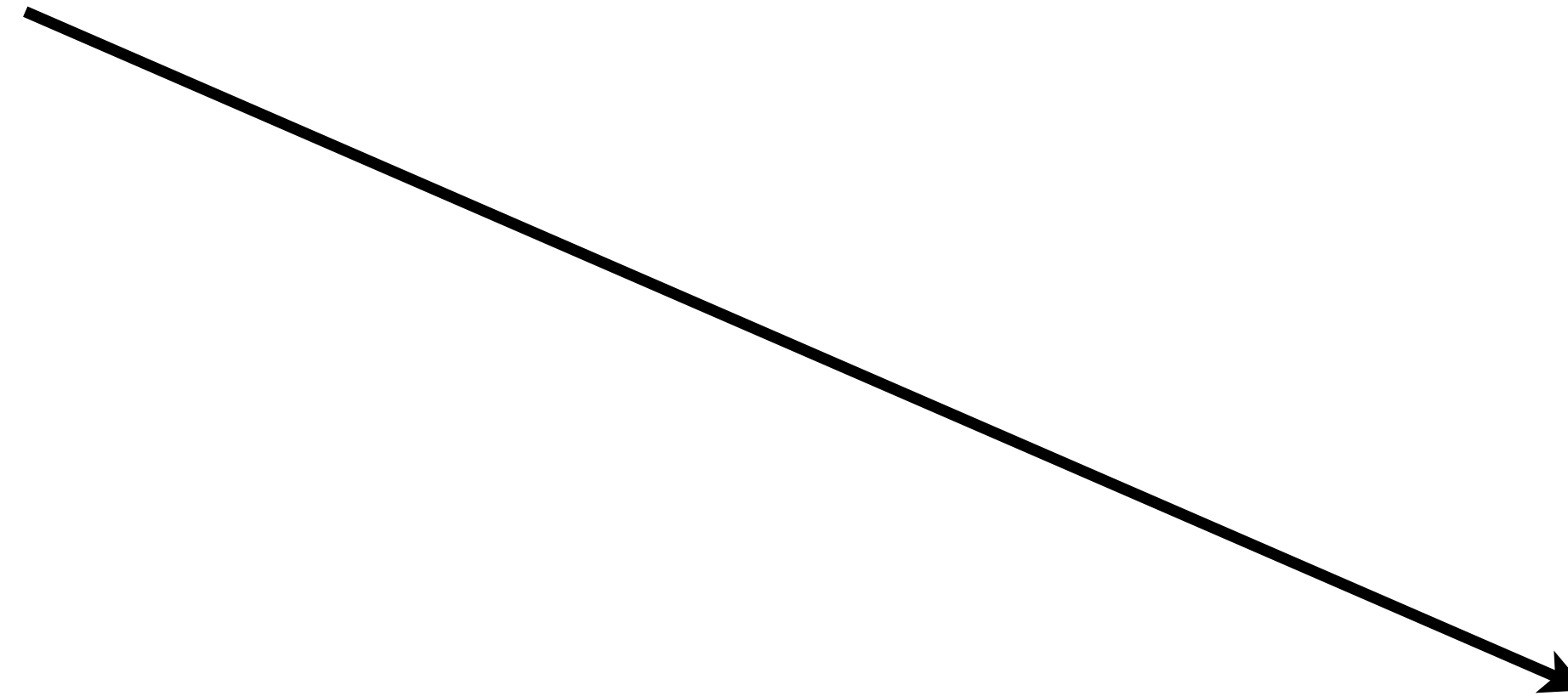
- rails



server (s) | console (c) | generate (g)

Corriendo la aplicación

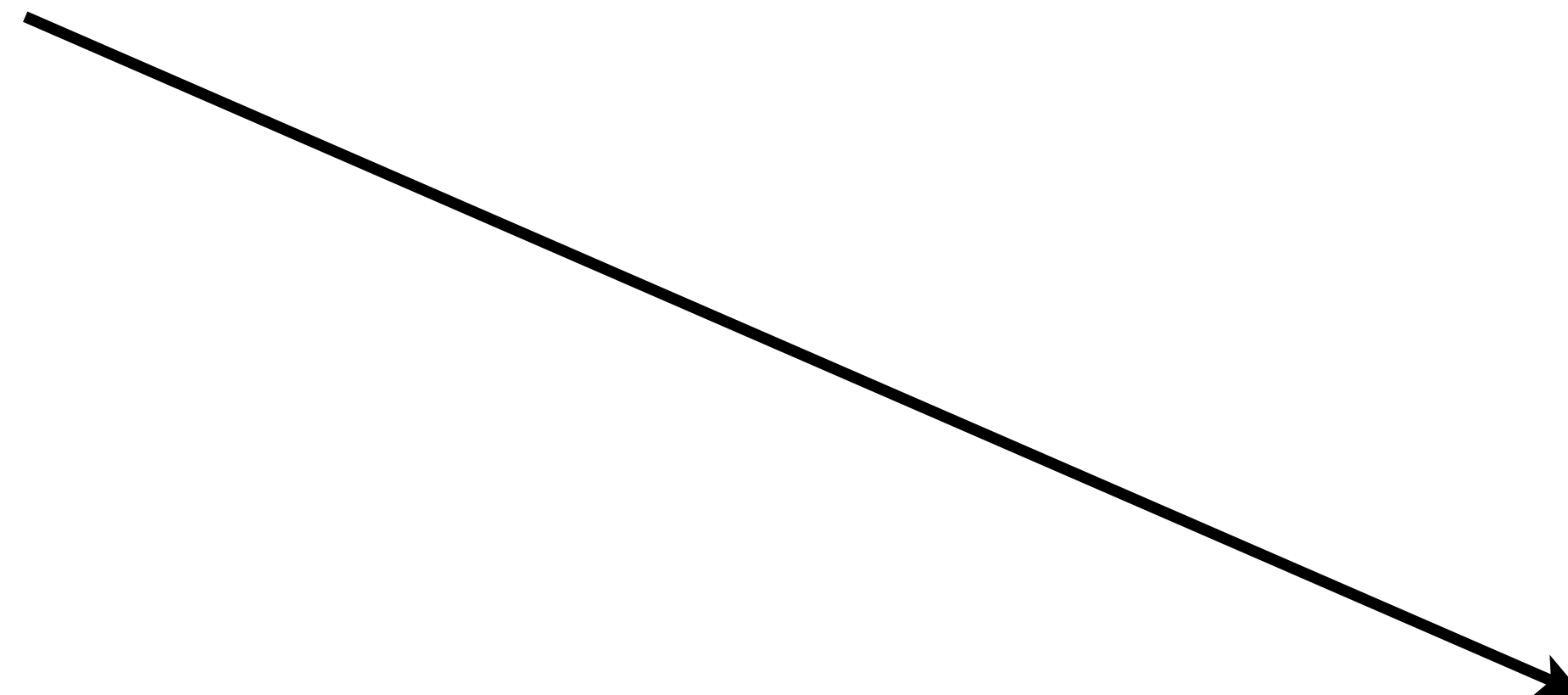
rails s



localhost:3000

Depurando la aplicación

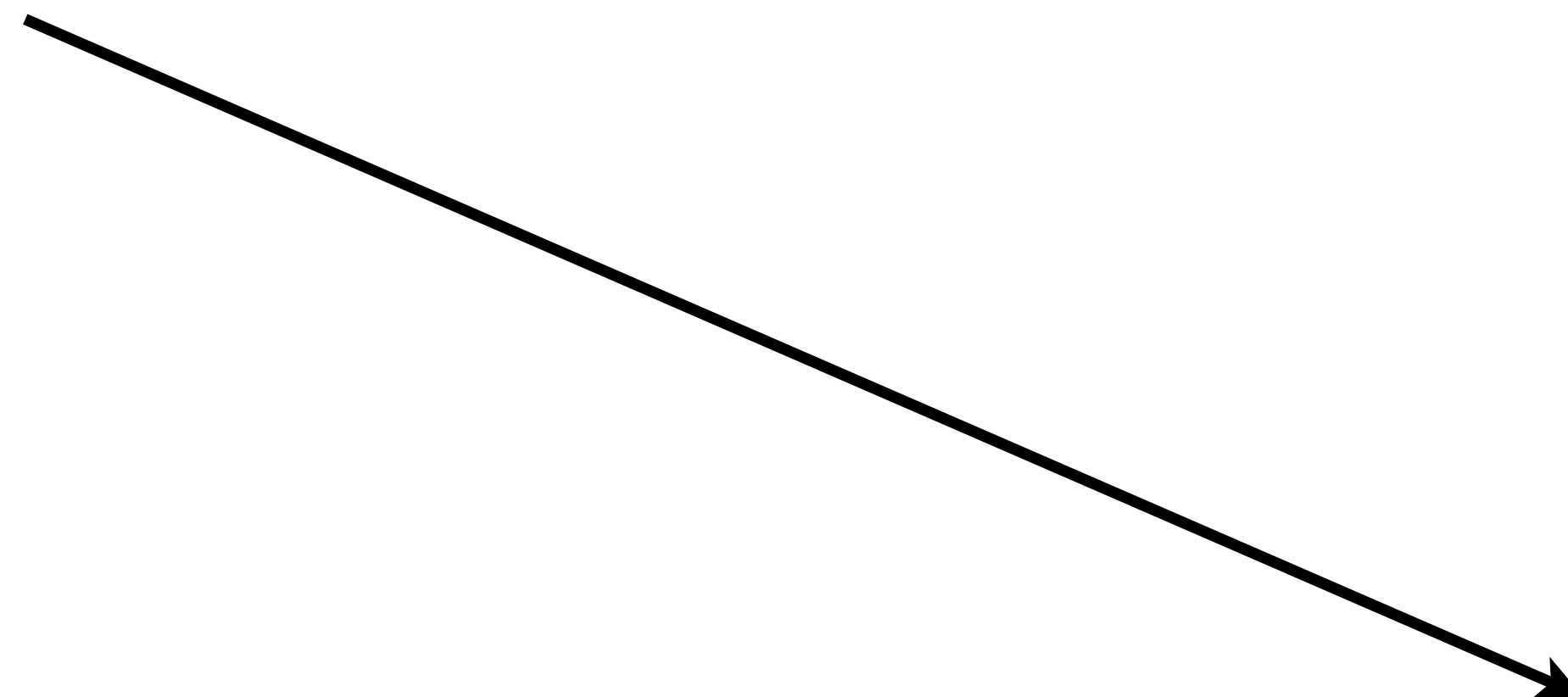
rails c



Habilita una consola en la que
podemos correr comandos de la BD

Generando los archivos

rails g



Genera archivos en base a un parámetro



```
$ rails generate controller Welcome index
```



git

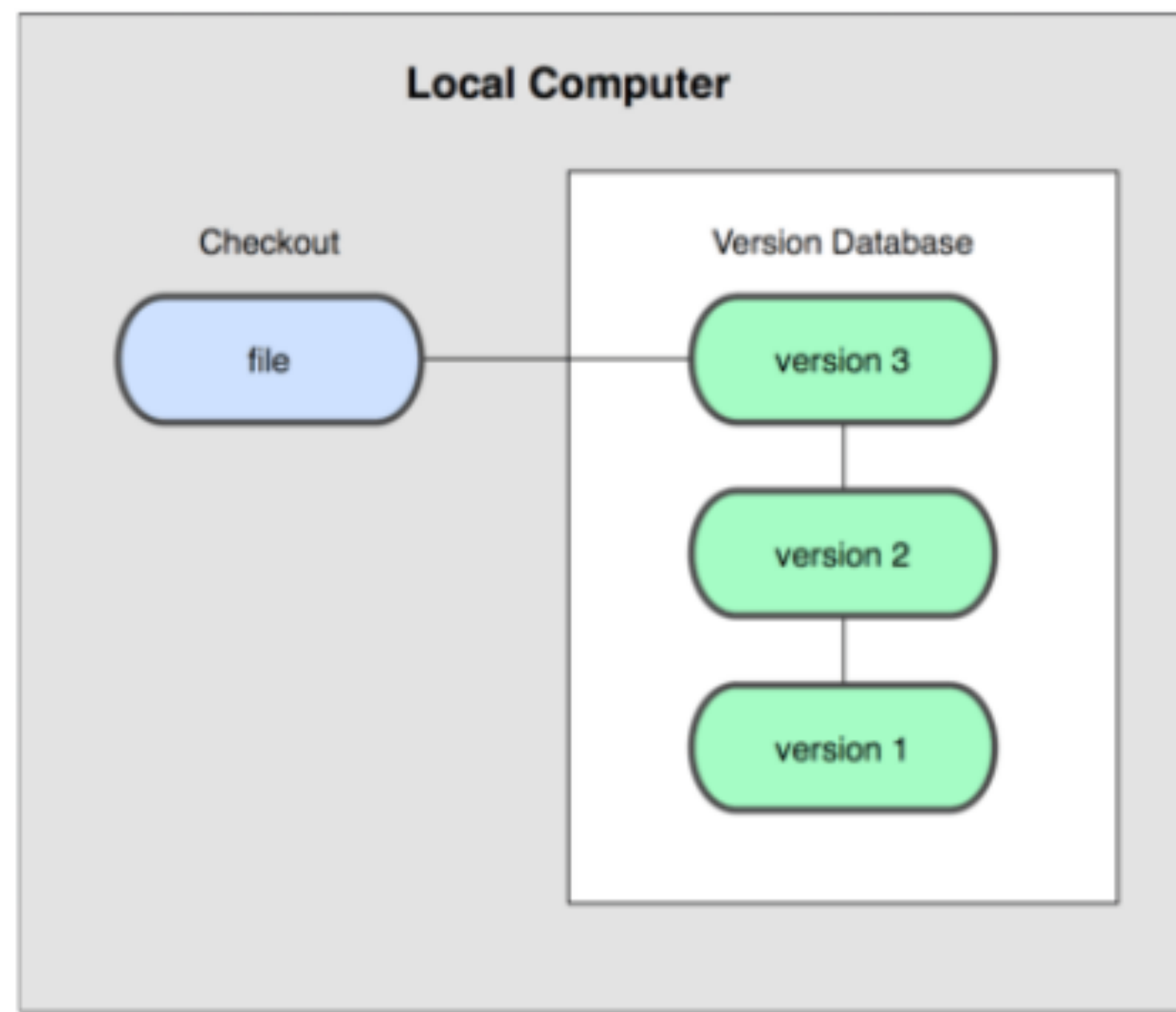
Manteniendo versiones de los archivos

El método de control de versiones usado por mucha gente es copiar los archivos a otro directorio (quizás indicando la fecha y hora en que lo hicieron).

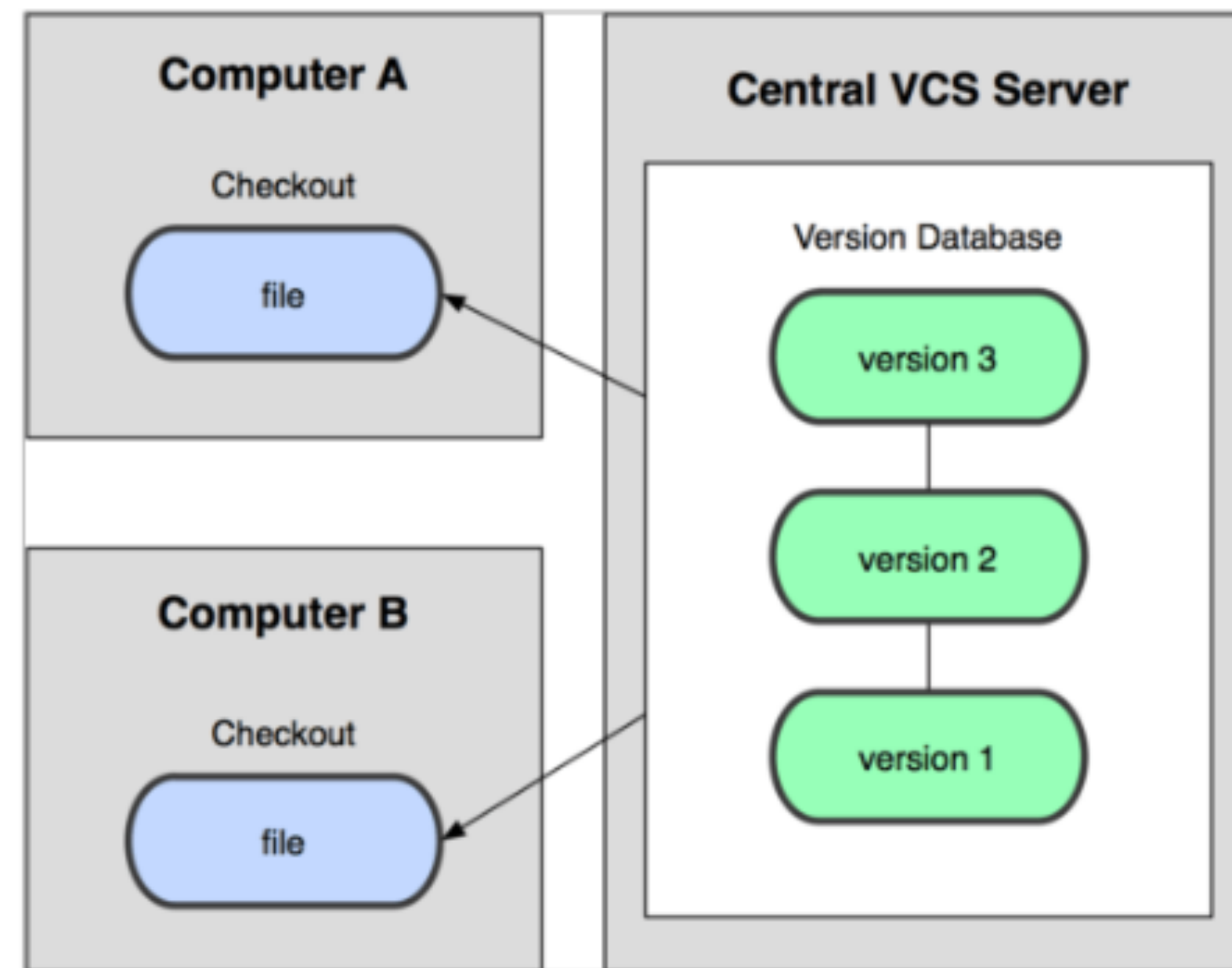
Sistema de control de versiones

El control de versiones es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de modo que se puedan recuperar versiones específicas más adelante.

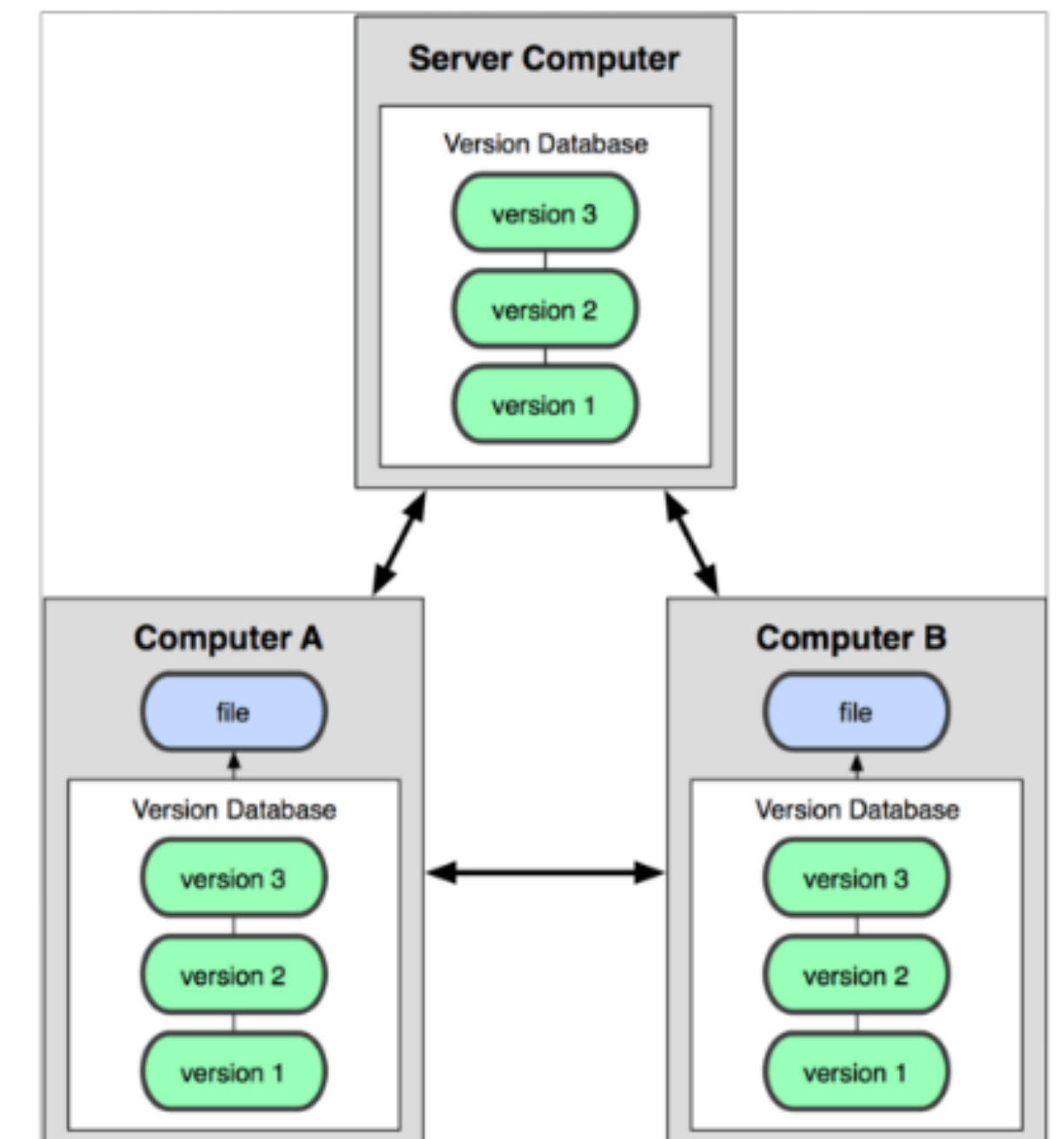
Tipos de sistemas de control de versiones



Local

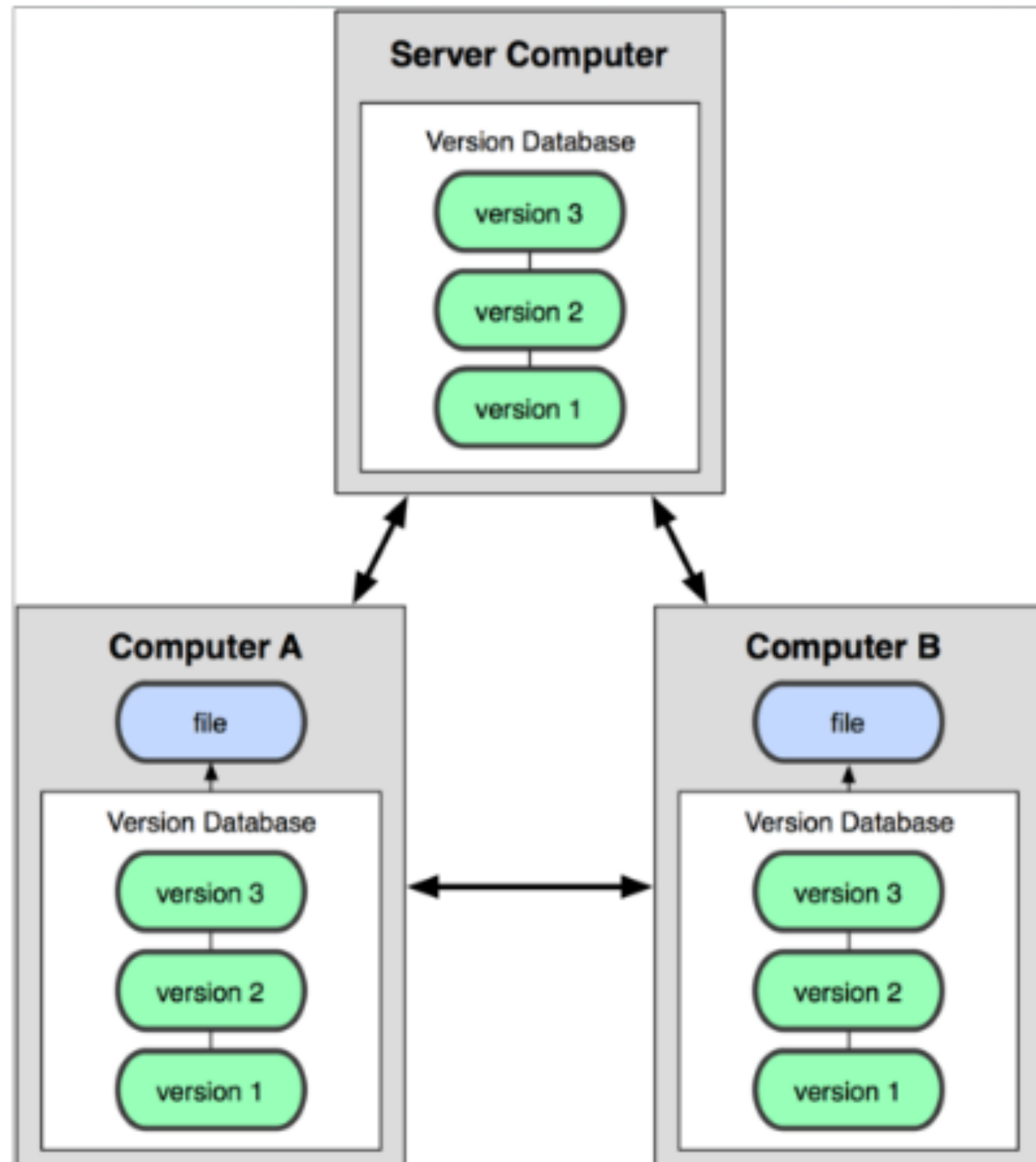


Centralizado



Distribuido

GIT



git init Para inicializar un repositorio git

git status Ver el estado de los archivos

git add Para agregar los archivos para el próximo commit

git commit Para confirmar localmente los archivos

git pull Para bajar los cambios realizados en el repositorio remoto

git push Para enviar los cambios al repositorio remoto

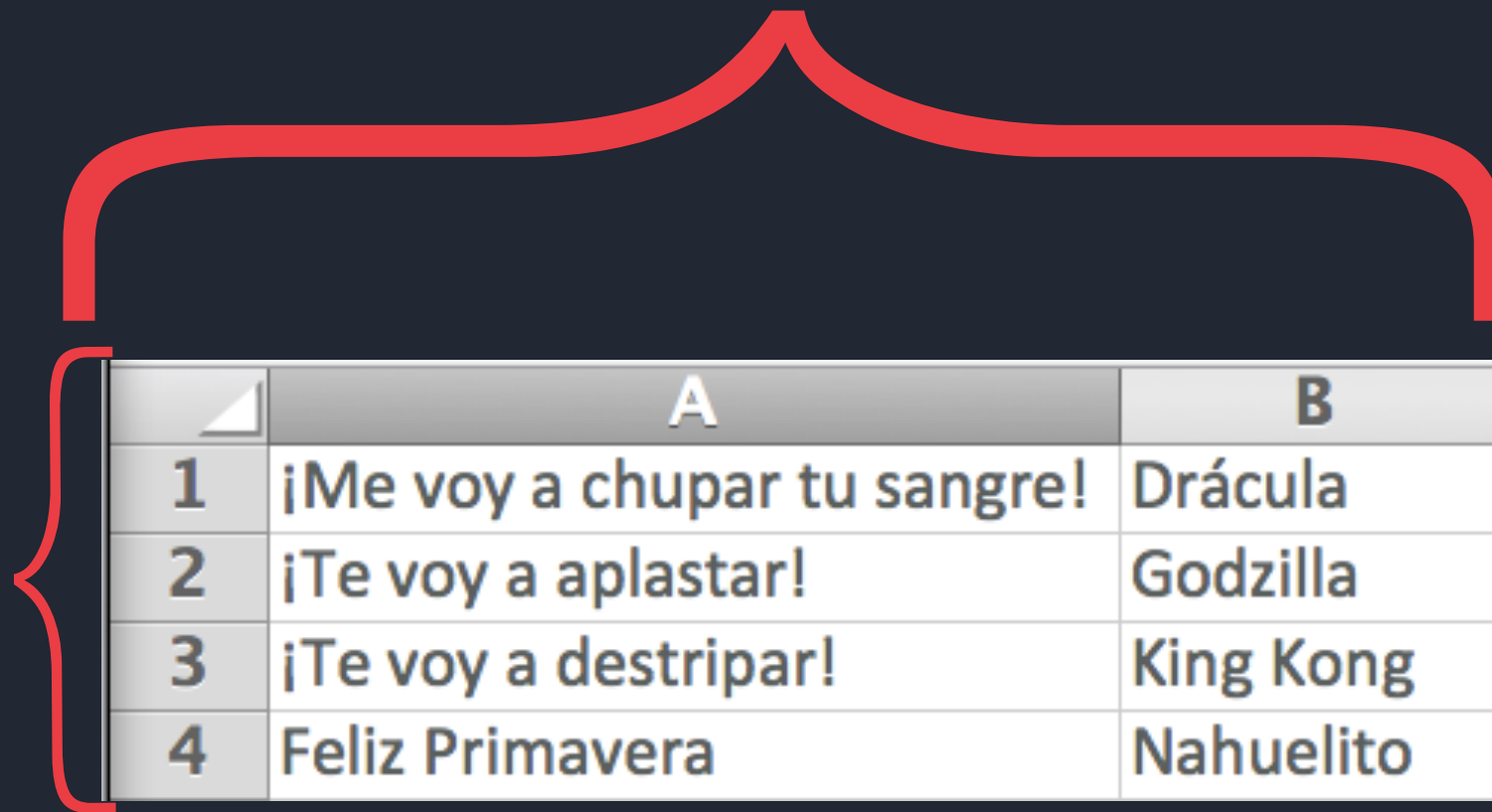
Twitter para monstruos



COLUMNAS

(tenemos 3)

FILAS
(tenemos 4)



| | A | B |
|---|-----------------------------|-----------|
| 1 | ¡Me voy a chupar tu sangre! | Drácula |
| 2 | ¡Te voy a aplastar! | Godzilla |
| 3 | ¡Te voy a destripar! | King Kong |
| 4 | Feliz Primavera | Nahuelito |

tweets



id



estado



monstruo

| id | estado | monstruo |
|----|---------------------------|-----------|
| 1 | Me voy a chupar tu sangre | Drácula |
| 2 | ¡Te voy a aplastar! | Godzilla |
| 3 | ¡Te voy a destripar! | King Kong |
| 4 | Feliz primavera | Nahuelito |

**Queremos obtener
el tweet con id = 3**



**Iterar sobre la tabla hasta
encontrar el tweet con id 3**



Obtener el tweet con id = 3

RESPUESTA

```
t = Tweet.find(3)
```



RESULTADO

```
t = #<Tweet id: 3,  
estado: "¡Te voy a destripar!",  
monstruo: "King Kong" >
```

tweets

Plural y en minúsculas

| id | estado | monstruo |
|----|---------------------------|-----------|
| 1 | Me voy a chupar tu sangre | Drácula |
| 2 | ¡Te voy a aplastar! | Godzilla |
| 3 | ¡Te voy a destripar! | King Kong |
| 4 | Feliz primavera | Nahuelito |

RESPUESTA

t = Tweet.find(3)

*El nombre de la tabla
singular y en mayúscula*

tweets

| id | estado | monstruo |
|----|---------------------------|-----------|
| 1 | Me voy a chupar tu sangre | Drácula |
| 2 | ¡Te voy a aplastar! | Godzilla |
| 3 | ¡Te voy a destripar! | King Kong |
| 4 | Feliz primavera | Nahuelito |

t = *Tweet.find*(3)

t.id

=> 3

t.estado

=> ¡Te voy a destripar!

t.monstruo

=> King Kong

Create

```
t = Tweet.new  
t.estado = "Tengo hambre"  
t.save
```

Read

```
t = Tweet.find(3)
```

Update


```
t = Tweet.find(3)  
t.monstruo = "Hombre lobo"  
t.save
```

Delete

```
t = Tweet.find(3)  
t.destroy
```

Create

```
t = Tweet.new  
t.estado = "Tengo hambre"  
t.monstruo = "Hombre lobo"  
t.save
```



El id se setea automáticamente

```
t = Tweet.new (  
  estado: "Tengo hambre",  
  monstruo: "Hombre lobo")  
t.save
```

```
t = Tweet.create (  
  estado: "Tengo hambre",  
  monstruo: "Hombre lobo")
```


Read

Tweet.find(3)

=> *Retorna el tweet con id 3*

Tweet.find(3, 4, 5)

=> *Retorna un array con los tweets con id 3, 4 y 5*

Tweet.first

=> *Retorna el primer tweet*

Tweet.last

=> *Retorna el último tweet*

Tweet.all

=> *Retorna un array con todos los tweets*

Read

Tweet.count

=> *Retorna la cantidad de tweets*

Tweet.order(:monstruo) # *Tweet.order('monstruo')*

=> *Retorna todos los tweets ordenados por monstruo*

Tweet.limit(10)

=> *Retorna los primeros 10 tweets*

Tweet.where(monstruo: "Drácula")

=> *Retorna los tweets de Drácula*

Se pueden combinar los métodos

Read

READ

Tweet.where(monstruo:“Drácula”).order(:estado).limit(10)

=> Retorna los primeros 10 tweets de Drácula ordenados por estado

Tweet.where(monstruo:“Drácula”).first

=> Retorna el primer tweet de Drácula

Update

UPDATE

```
t = Tweet.find(3)
t.monstruo = "Hombre lobo"
t.save
```

```
t = Tweet.find(3)
t.attributes = {
  estado: "¿Y Caperucita?",
  monstruo: "Hombre lobo"
}
t.save
```

```
t = Tweet.find(3)
t.update (
  estado: "Tengo hambre",
  monstruo: "Hombre lobo"
)
```

Delete

DELETE

```
t = Tweet.find(3)  
t.destroy
```

```
t = Tweet.find(3).destroy
```

```
t = Tweet.where("monstruo = 'Dracula'").destroy_all
```