Express & Sequelize

Rounding Out

Express & Sequelize

An assortment of tips, tricks, time savers and conversation starters...

What we will cover

• Express

- Custom error handler
- 404 Not Found
- Template engines

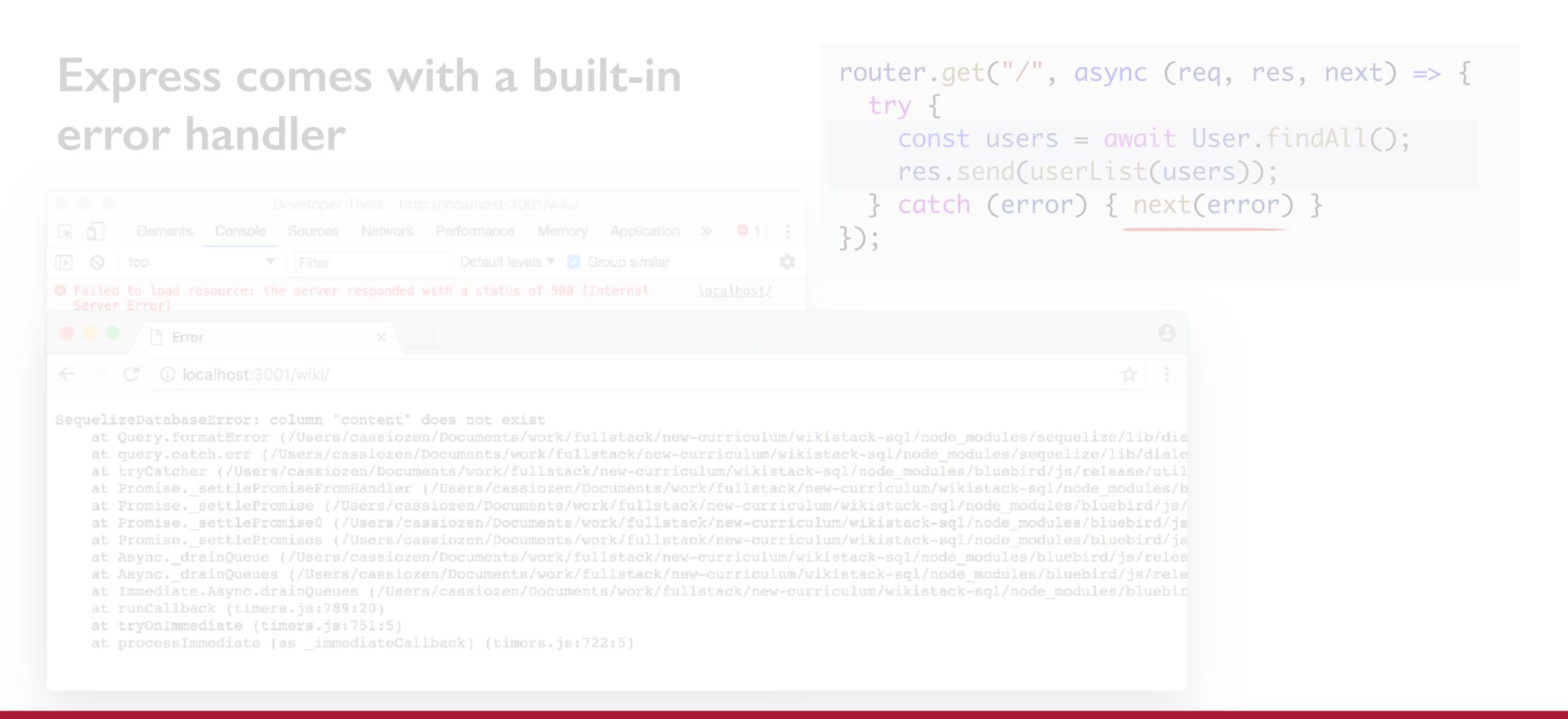
Sequelize

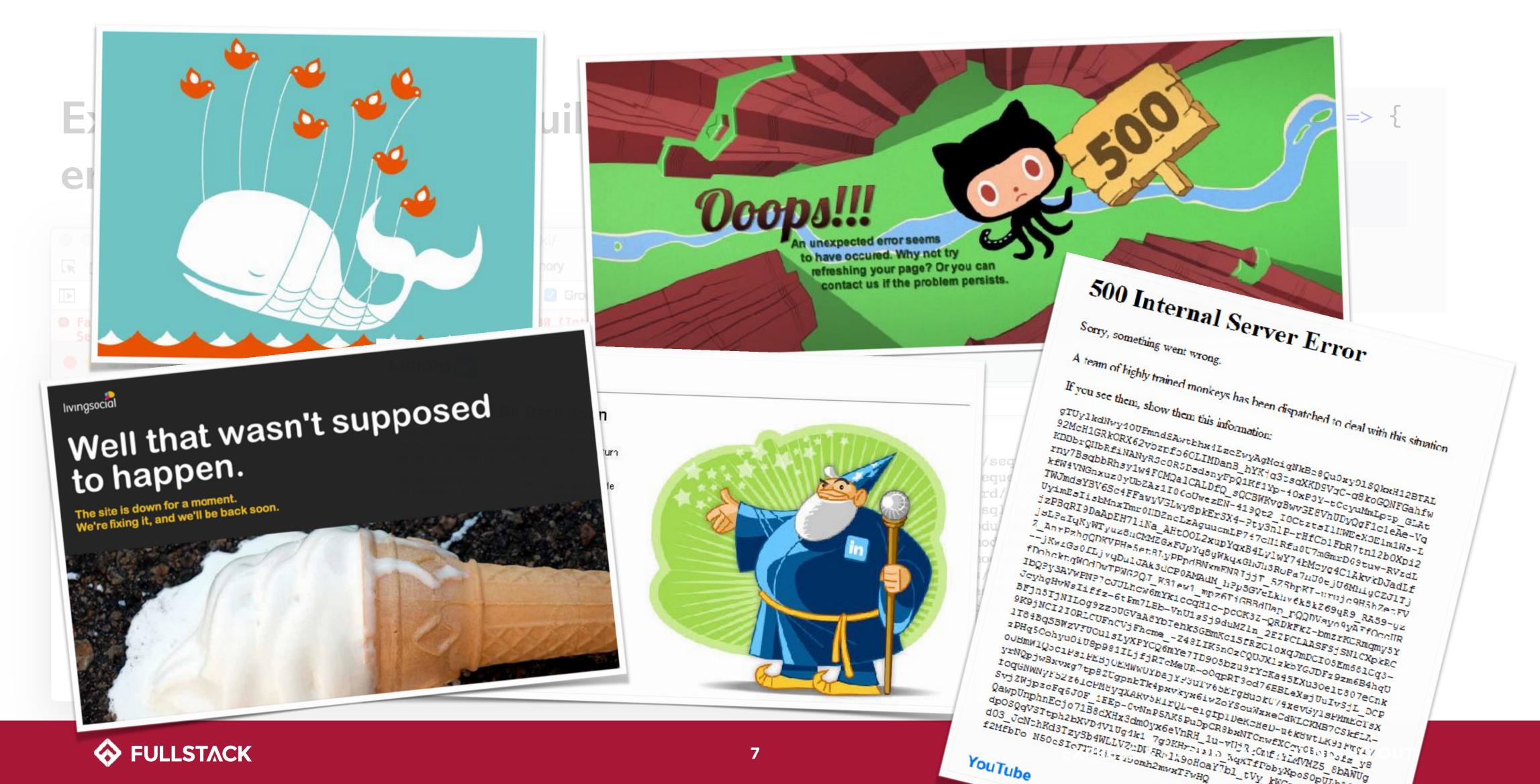
- Eager Loading
- Class methods / instance methods
- Many-Many Relationships

Express comes with a built-in error handler

```
router.get("/", async (req, res, next) => {
   try {
     const users = await User.findAll();
     res.send(userList(users));
   } catch (error) { next(error) }
});
```

Express comes with a built-in router.get("/", async (req, res, next) => { try { error handler const users = await User.findAll(); res.send(userList(users)); } catch (error) { next(error) } Developer Tools - http://localhost:3001/wiki/ Network Performance Console }); ▼ | Filter Default levels ▼ ✓ Group similar Failed to load resource: the server responded with a status of 500 (Internal localhost/ Server Error) localhost:3001/wiki/ ₩ SequelizeDatabaseError: column "content" does not exist at Query.formatError (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node_modules/sequelize/lib/dia at query.catch.err (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node_modules/sequelize/lib/diale at tryCatcher (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node_modules/bluebird/js/release/util at Promise._settlePromiseFromHandler (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node_modules/b at Promise._settlePromise (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node_modules/bluebird/js/ at Promise. settlePromise0 (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node modules/bluebird/js at Promise. settlePromises (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node_modules/bluebird/js at Async._drainQueue (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node_modules/bluebird/js/relea at Async. drainQueues (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node modules/bluebird/js/rele at Immediate.Async.drainQueues (/Users/cassiozen/Documents/work/fullstack/new-curriculum/wikistack-sql/node modules/bluebir at runCallback (timers.js:789:20) at tryOnImmediate (timers.js:751:5) at processImmediate [as _immediateCallback] (timers.js:722:5)





- Define error-handling middleware just like other middleware
- except error-handling functions have four arguments instead of three: (err, req, res, next)

```
app.use((err, req, res, next) => {
  console.error(err.stack)
  res.status(500)
    .send(/* Some friendly content */)
})
```

Express: 404 Not Found

Express: 404 Not Found

```
// All routes and middlewares above

// Last, Handle 404:
app.use((req, res) => {
   res.status(404).send(/*Not found*/);
});
```

- JavaScript template literals # template engine.
- Templates: static text files with special annotations.
- Template engines:
 - Interprets the document;
 - Replaces variables with actual values
 - Send them as HTML in the HTTP response.

userList.js const html = require("html-template-tag"); const layout = require("./layout"); module.exports = (users) => layout(html` <h3>Users</h3> <hr> <l \${users.map(user => html`> \${user.name} `)}

userList.html

userList.js const html = require("html-template-tag"); const layout = require("./layout"); module.exports = (users) => layout(html` <h3>Users</h3> <hr> <l \${users.map(user => html`> \${user.name} `)}

userList.html

Sequelize: Eager Loading



Sequelize: not Eager Loading

```
const pages = await Page.findAll();
for (let i = 0; i < pages.length; i++) {
  const author = await pages[i].getAuthor();
}</pre>
```



Sequelize: not Eager Loading

```
const pages
                  t Page
for (let i = 0;
                        .length; i++) {
                       hages[i].getAuthor();
  const author =
```

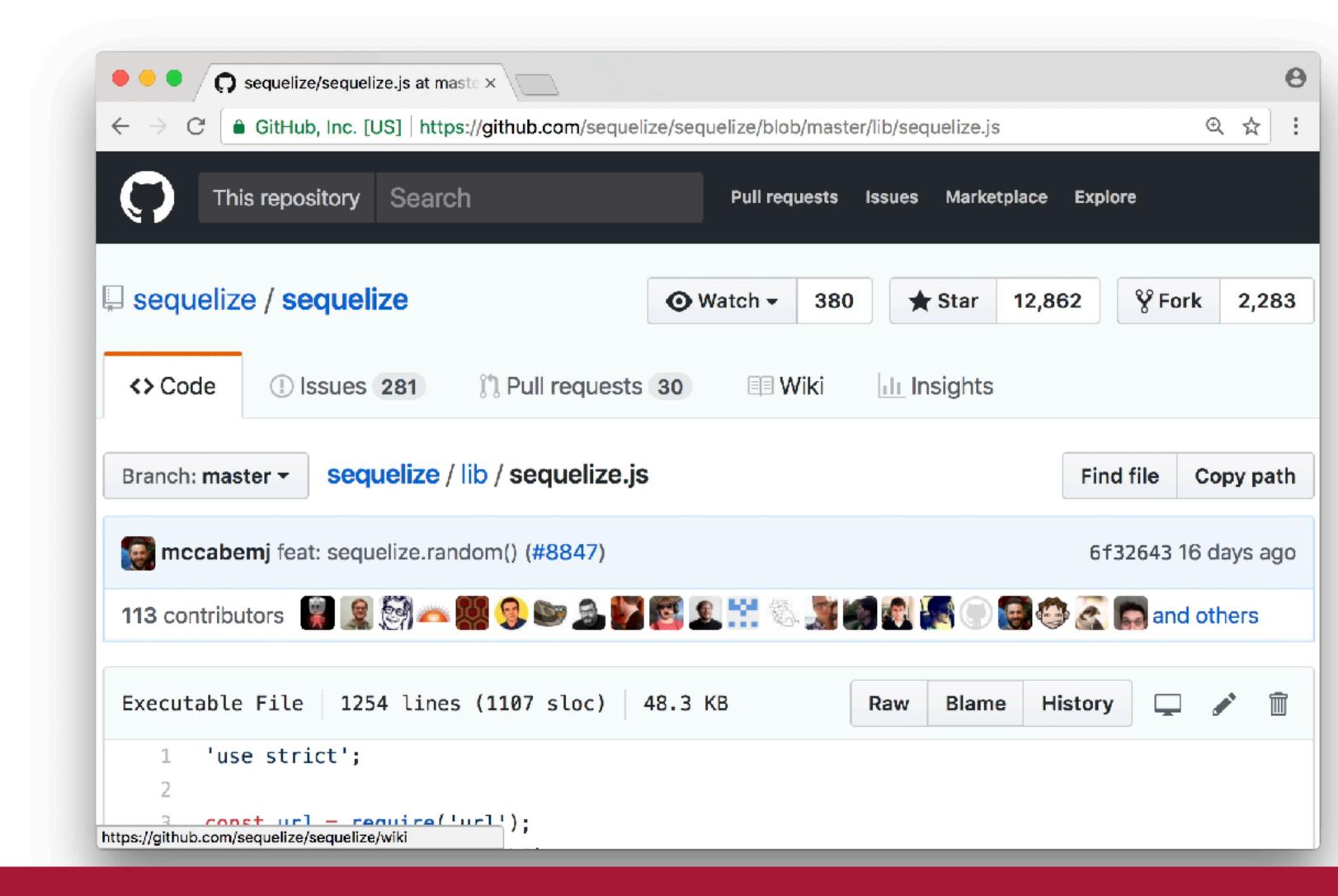
Sequelize: Eager Loading

- In raw SQL queries, we have JOIN
- In Sequelize it just goes by the name of "eager loading"
 - Usually a LEFT OUTER JOIN
- Don't get hung up on the terminology
 when you see "eager loading", think "join two tables".

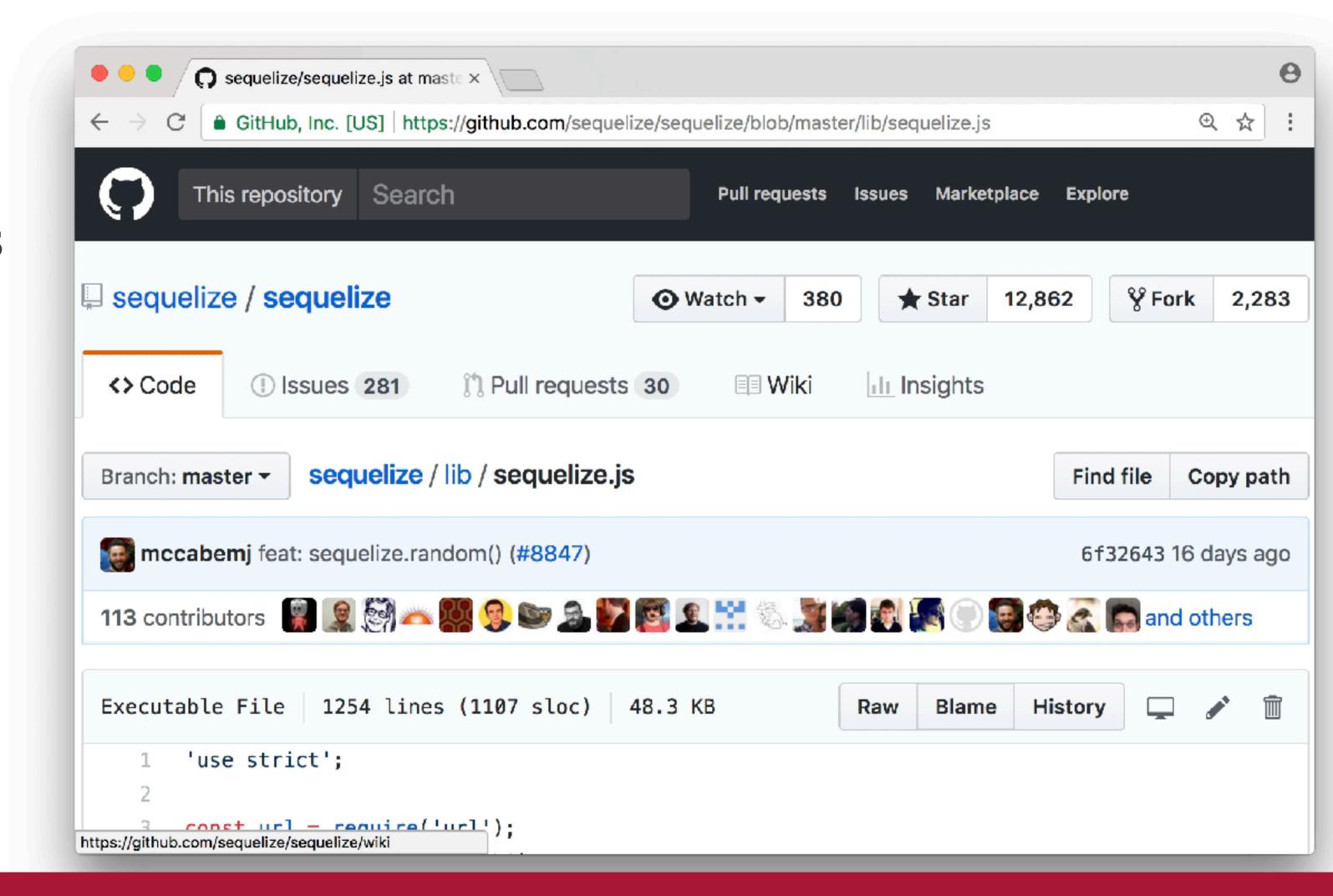


Sequelize: Eager Loading

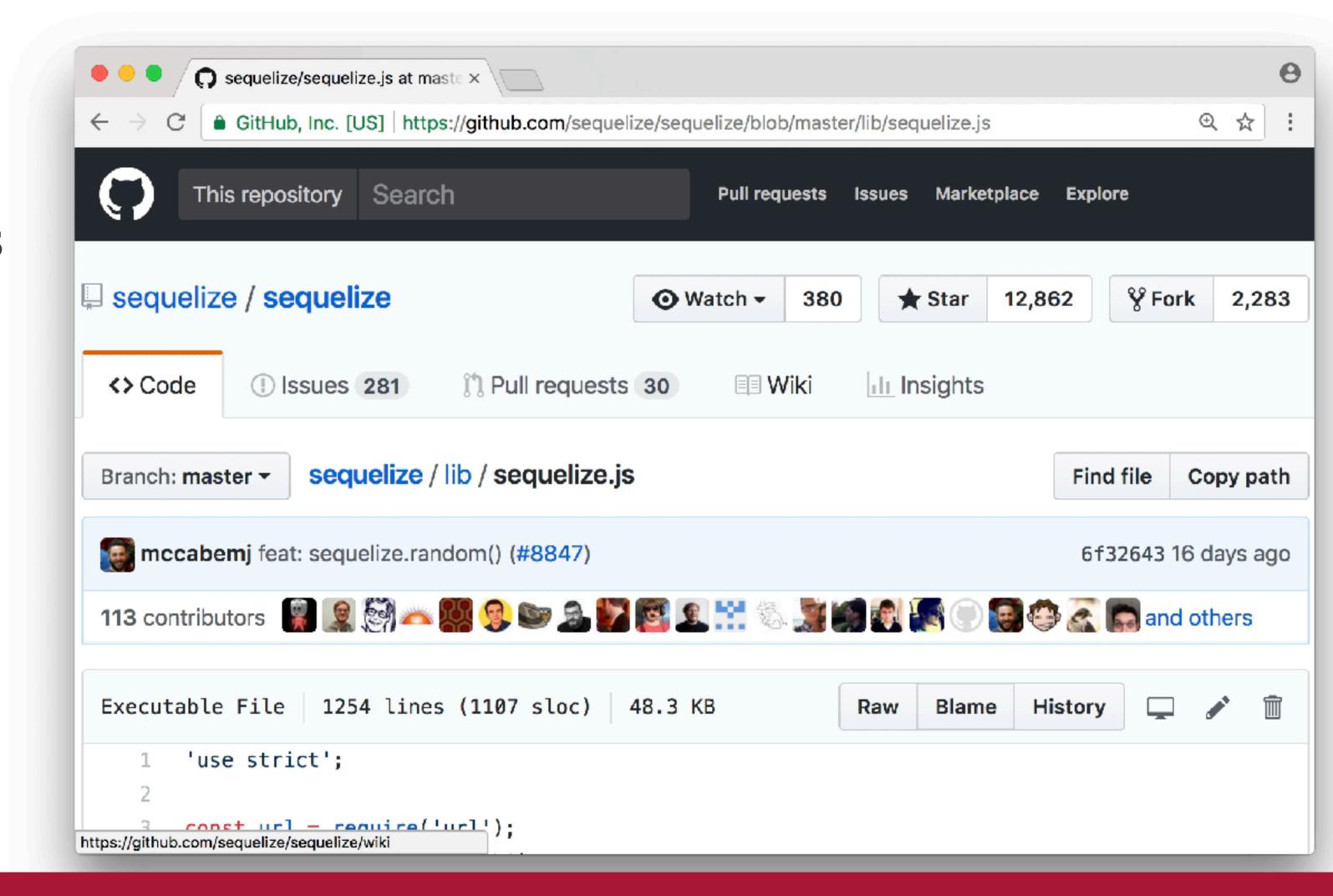
```
const pages = await Page.findAll({
  include: [
     {model: User, as: 'author'}
  ]
});
```



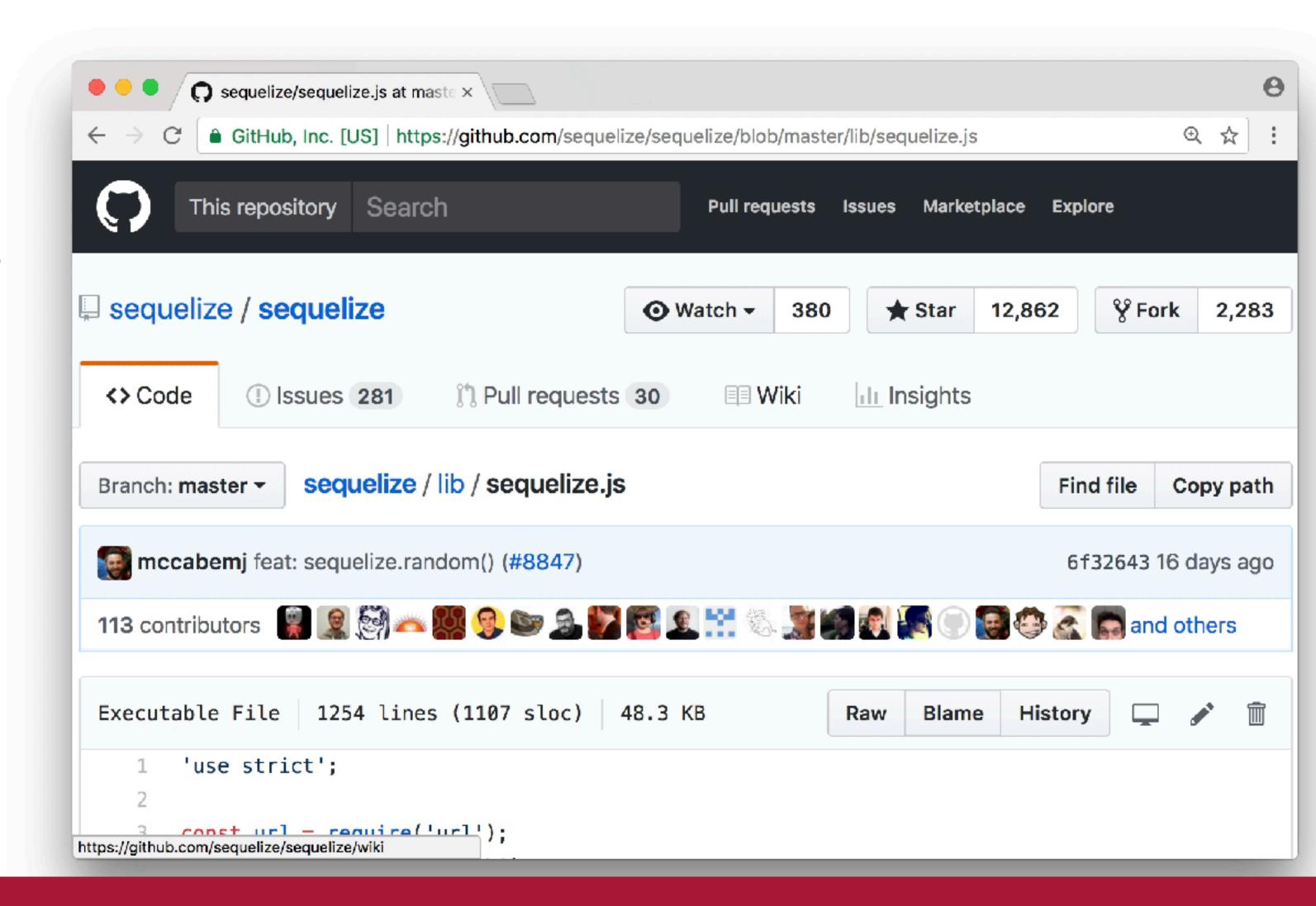
Common/Convenient
 ways to add functionality
 to your Sequelize models



Common/Convenient
 ways to add functionality
 to your Sequelize models



- Common/Convenient
 ways to add functionality
 to your Sequelize models
- It's just Javascript:
 We can add functions to the constructor function
 OR to its prototype.





```
const Dog = db.define('dog', {/* etc*/})
```



```
const Dog = db.define('dog', {/* etc*/})
Dog.findPuppies = function () {
 // 'this' refers directly back to the model
  return this.findAll({ // Dog.findAll
   where: {
     age: {$lte: 1}
```



```
const Dog = db.define('dog', {/* etc*/})
Dog.findPuppies = function () {
 // 'this' refers directly back to the model
  return this.findAll({ // Dog.findAll
   where: {
     age: {$lte: 1}
```



```
const Dog = db.define('dog', {/* etc*/})
Dog.findPuppies = function () {
 // 'this' refers directly back to the model
  return this.findAll({ // Dog.findAll
    where: {
                         In your routes, for example...
      age: {$lte: 1}
                          const foundPuppies = await Dog.findPuppies()
```



```
const Dog = db.define('dog', {/* etc*/})
```



```
const Dog = db.define('dog', {/* etc*/})
Dog.prototype.isBirthday = function () {
 const today = new Date()
 // 'this' refers to the instance itself
 if (this.birthDay === today.getDate()) {
    return true;
 } else {
    return false;
```



```
const Dog = db.define('dog', {/* etc*/})
Dog.prototype.isBirthday = function () {
  const today = new Date()
 // 'this' refers to the instance itself
 if (this.birthDay === today.getDate()) {
    return true;
 } else {
    return false;
```



```
const Dog = db.define('dog', {/* etc*/})
Dog.prototype.isBirthday = function () {
  const today = new Date()
  // 'this' refers to the instance itself
  if (this.birthDay === today.getDate()) {
    return true;
                            In your routes, for example...
  } else {
                             const createdDog = new Dog({name: 'Pork Chop'})
    return false;
                             // the instance method is invoked *on the instance*
                             if(createdDog.isBirthday) console.log("Happy birthday!")
```



Sequelize: Many-Many Relationships

ONE-ONE

PUGS

id	name	ownerld	
I	Cody		
2	Doug	NULL	
3	Maisy	4	
4	Frank	NULL	

Pug.belongsTo(Owner) Owner.hasOne(Pug)

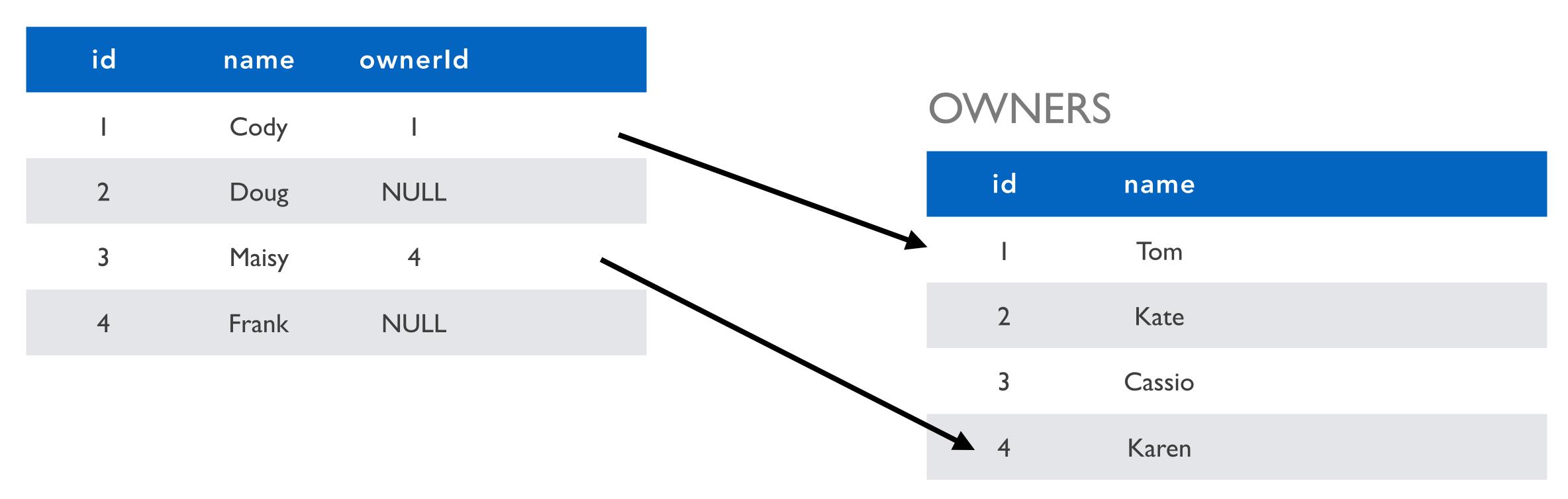
OWNERS

id	name	
I	Tom	
2	Kate	
3	Cassio	
4	Karen	

ONE-ONE

Pug.belongsTo(Owner) Owner.hasOne(Pug)

PUGS



ONE-MANY

Pug.belongsTo(Owner) Owner.hasMany(Pug)

PUGS

id	name	ownerld	
I	Cody	I	
2	Doug	ı	
3	Milo	2	
4	Frank	NULL	

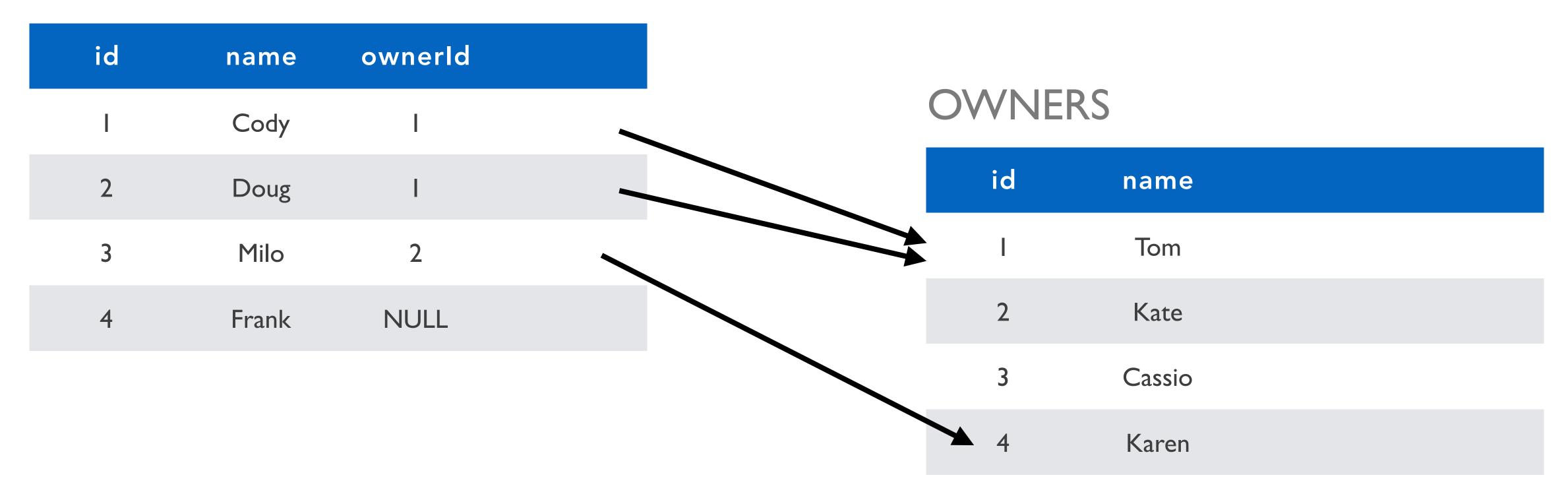
OWNERS

id	name	
I	Tom	
2	Kate	
3	Cassio	
4	Karen	

ONE-MANY

Pug.belongsTo(Owner) Owner.hasMany(Pug)

PUGS



Pug.belongsToMany(Friend, {through: "pug_friend"})
Friend.belongsToMany(Pug, {through: "pug_friend"})

PUG FRIEND

pug_id	friend_id
I	I
I	2
2	

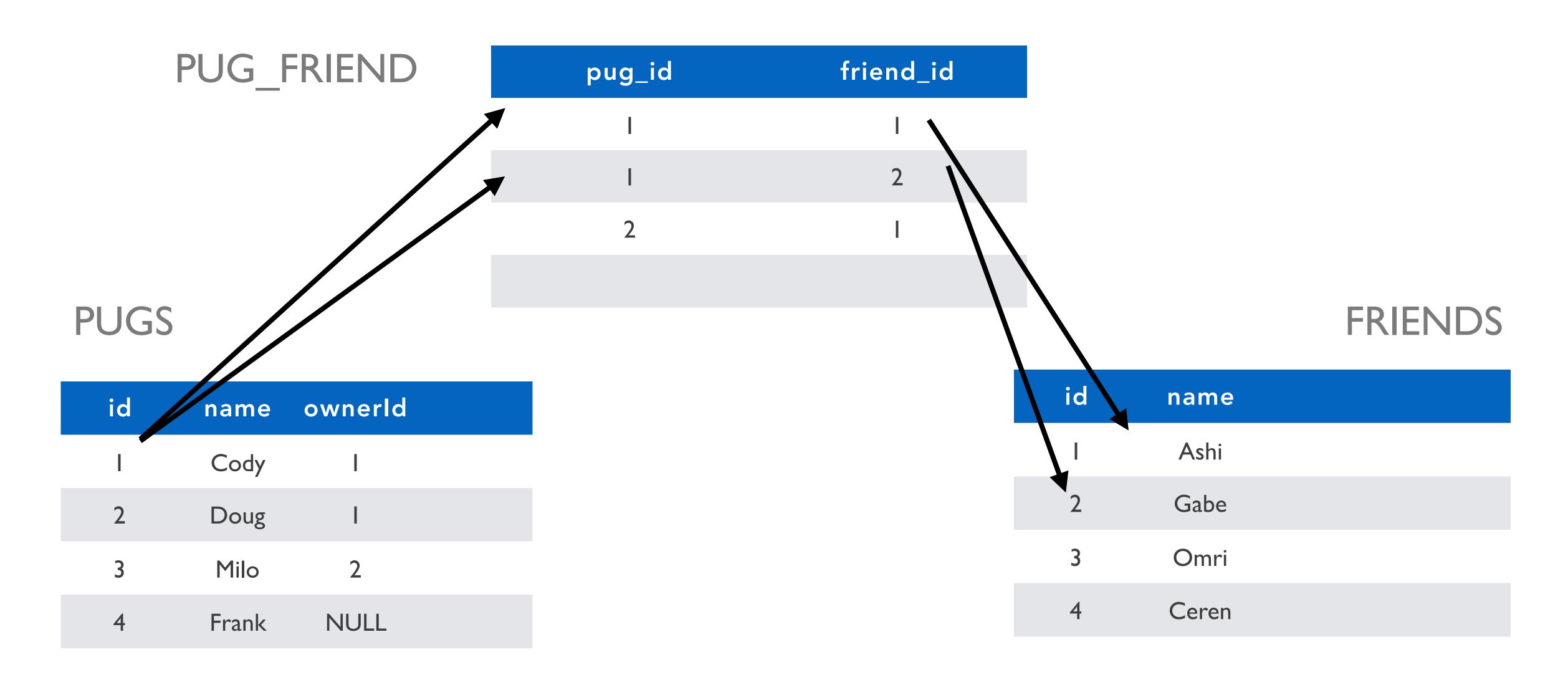
PUGS

FRIENDS

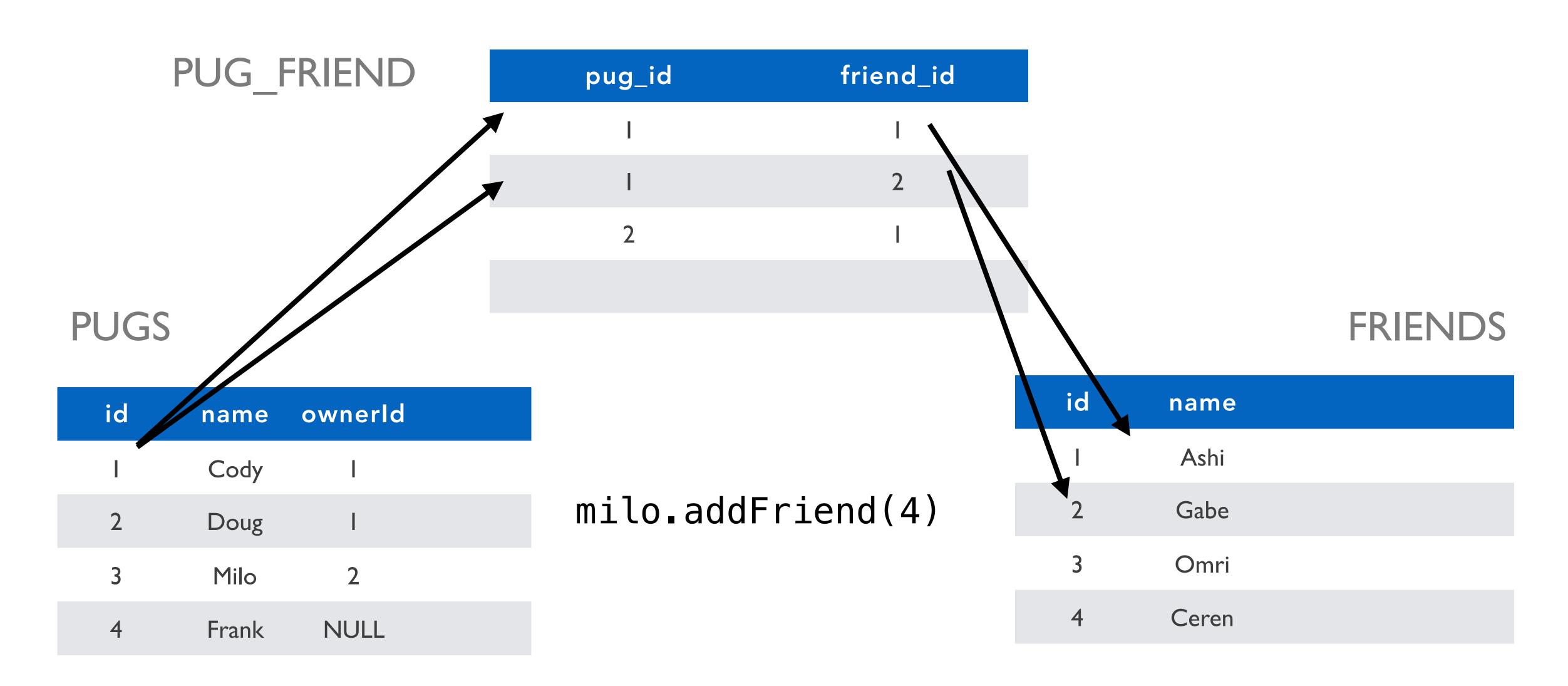
id	name	ownerld	
I	Cody	I	
2	Doug	I	
3	Milo	2	
4	Frank	NULL	

id	name	
I	Ashi	
2	Gabe	
3	Omri	
4	Ceren	

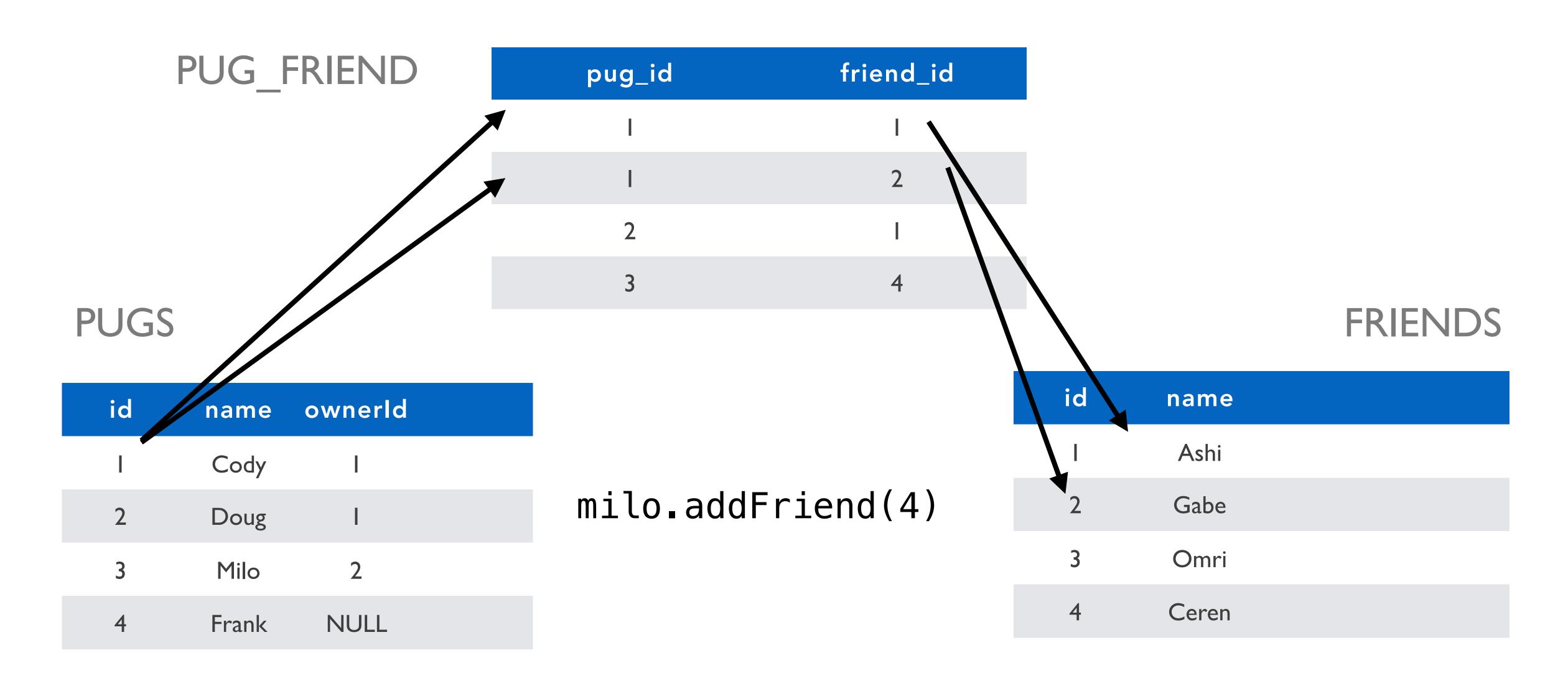
Pug.belongsToMany(Friend, {through: "pug_friend"})
Friend.belongsToMany(Pug, {through: "pug_friend"})



Pug.belongsToMany(Friend, {through: "pug_friend"})
Friend.belongsToMany(Pug, {through: "pug_friend"})



Pug.belongsToMany(Friend, {through: "pug_friend"})
Friend.belongsToMany(Pug, {through: "pug_friend"})



Workshop

• Express

- Custom error handler
- 404 Not Found
- Template engines

Sequelize

- Eager Loading
- Class methods / instance methods
- Many-Many relationships