Sequelize

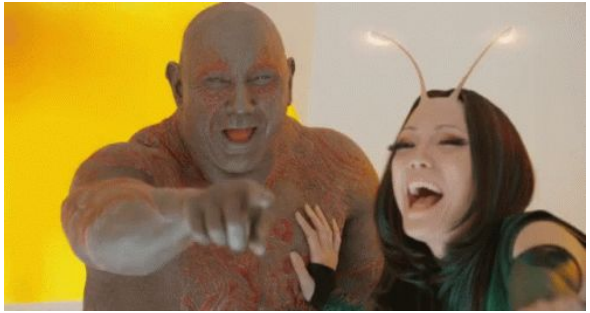
Why I never use SQL...

Object Relational Mapping ... ORM for short

1. Simplifying Database Access
    a. ORMs allow you to access and manipulate your database without needing to write SQL queries directly
2. Automatic SQL Generation
    a. ORMs generate SQL queries for you based on your Javascript data model
3. Focus on Javascript:
    a. With ORMs, particularly sequelize for our case, your main focus stays on Javascript
4. Reduce Errors and increase Speed
    a. ORMs help cut down SQL types and mistakes. This leads to faster development process and better code quality

Introduction to Sequelize

- What is Sequelize?
- Why is Sequelize?
- WHO IS SEQUELIZE?



Installing Sequelize

npm install sequelize

npm install --save pg pg-hstore

Connecting to Sequelize

Similar to pg and our pool method

```
const { Sequelize } = require('sequelize');

const sequelize = new Sequelize('postgres://user:pass@example.com:5432/dbname')
```

Models and Migrations

What is a model?

An abstract representation of your database. The model tells Sequelize several things about the entity it represents, such as the name of the table in the database and which columns it has (and their data types).

What is a migration?

- Database Migrations are like a 'Git for Databases'. They track and manage changes made to a database over time
- To smoothly transition a database from one state to another, and even reverse the transition if necessary.
- Changes are saved in 'migration files'.
- These files describe how to modify the database to reach a new state, and how to undo those changes if needed.

What the CRUD?

```
// CREATE
const jane = User.build({ firstName: "Jane", lastName: "Doe" });
await jane.save();

// READ
const users = await User.findAll();

// UPDATE
const user = await User.findByPk(1);
await user.update({firstName: 'John'});

// DELETE
const userToDelete = await User.findByPk(2);
await userToDelete.destroy();
```

We can do all the CRUD operations in sequelize:

- Create
- Read
- Update
- Delete

Associations

```
// One-to-one
User.hasOne(Profile);
Profile.belongsTo(User);

// One-to-many
User.hasMany(Post);
Post.belongsTo(User);

// Many-to-many
User.belongsToMany(Project, { through: 'UserProject' });
Project.belongsToMany(User, { through: 'UserProject' });
```

Associations are relationships between tables. There are 3 types:

1. One to One
2. One to Many
3. Many to Many

Querying in Sequelize

We can use specialized querying methods to specialize or order our data to our specific needs.

```
// WHERE
const users = await User.findAll({ where: { firstName: 'John' } });

// ORDERING
const orderedUsers = await User.findAll({ order: [['createdAt', 'DESC']]});

// PAGINATION
const paginatedUsers = await User.findAll({ offset: 10, limit: 2 });
```

Now let’s put it all together…

Let’s build a whole backend!