Лабораторная работа №4,5

SEQUALIZE NODE JS

Выполнил: Нелепов Р.А.

Цель: изучить генерацию моделей, создание связей 1:1, 1:N, добавление, удаление, поиск.

Ход работы:

Настроим работу с БД, для этого зайдем в файл config.json и выставим значения показанные на рисунке 1

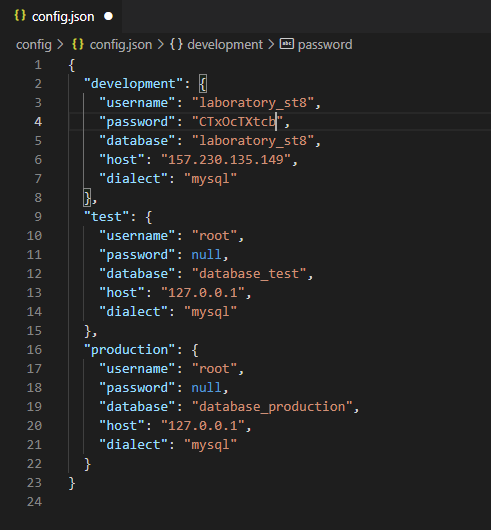


Рисунок – 1 Настройка БД

Создадим миграции и модели (рисунок 2) для таблиц приведенных на рисунке 3

Для таблицы Users

npx sequelize-cli model: generate --name User --attributes firstName: string, lastName: string, email: string

Для таблицы Comment

npx sequelize-cli model: generate --name Comment --attributes id\_user: string comment: string

Для таблицы articles

npx sequelize-cli model:generate --name articles --attributes id\_user:string, text:string

Для таблицы passwords

npx sequelize-cli model:generate --name password --attributes password:string

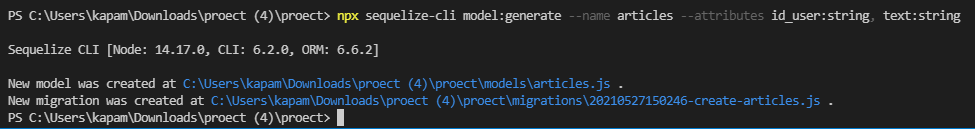


Рисунок 2 – Пример создания модели

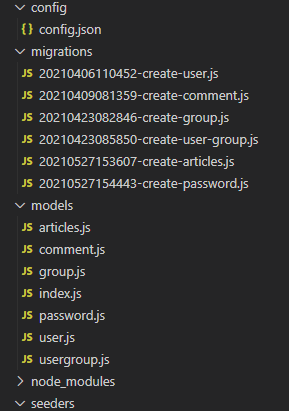


Рисунок 3 – Модели и миграции

Добавим созданные модели в БД, для этого выполнили команду (рисунок 4):

npx sequelize-cli db:migrate

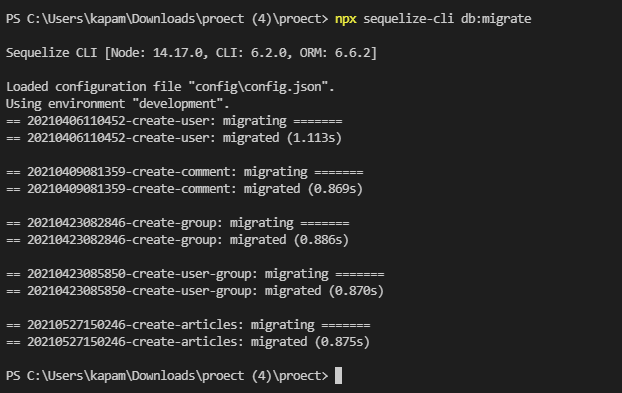


Рисунок 4 – Выполнение миграций

Проверим в phpmyadmin добавление таблиц (рисунок 5)

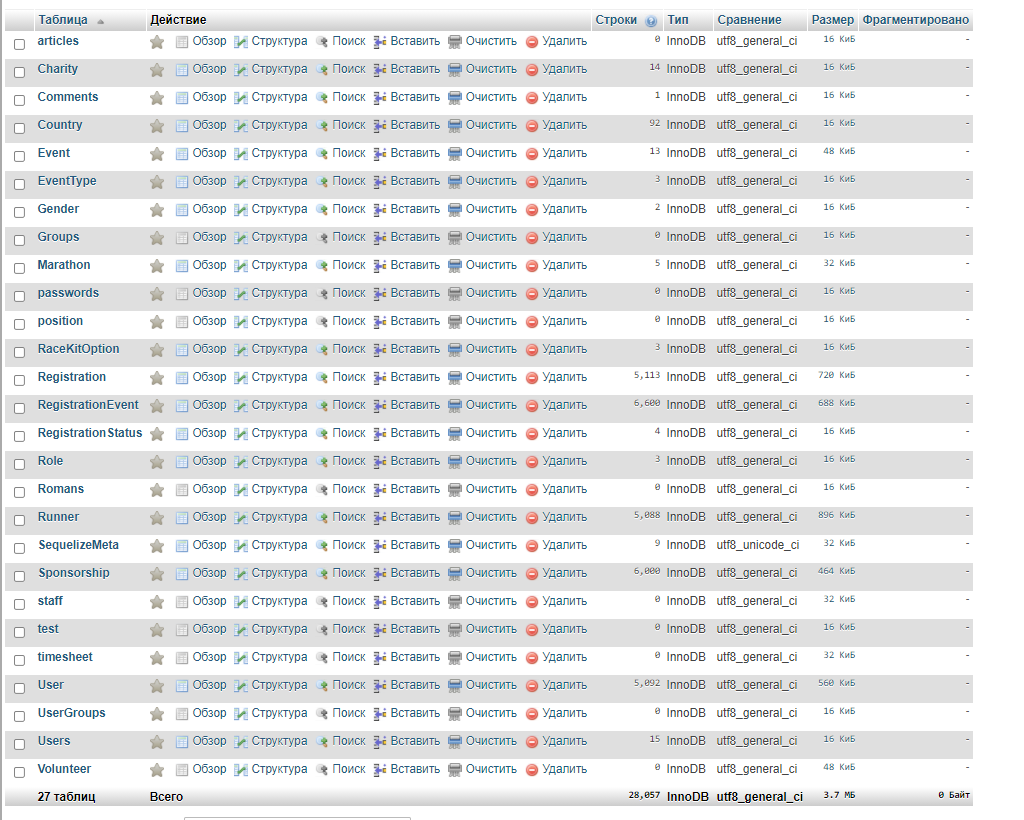


Рисунок 5 – Проверка добавления таблиц

Сделаем связь один к одному

User.create({firstName: "Roman",lastName: "Nelepov", email: "nelepoman@mail.ru"})



Рисунок 6 – Добавление пользователя

const user = await User.findByPk(33);

const user\_Articles = await Articles.create({id\_user:user.id, text:"text text roman namor txet txet"});

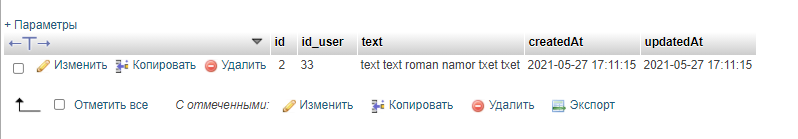


Рисунок 7 – Добавление articles (статьи) для пользователя

User.destroy({

where: {id: 28}

});



Рисунок 8 – Удаление пользователя

Найдем все articles с 2 атрибутами 'createdAt' и 'updatedAt' (рисунок 9)

Articles.findAll({

attributes:{ exclude: ['createdAt','updatedAt'] }

}).then((users)=>{

console.log(users)

});;

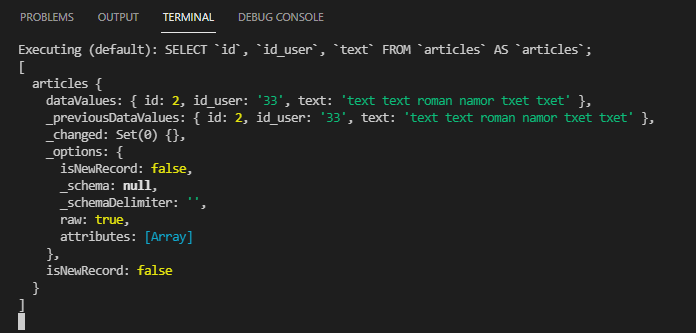


Рисунок 9 – Вывод результата поиска

Сделаем связь один ко многим

const Roman = await User.findByPk(33);

const comment = await Articles.create({id\_user:Roman.id, text: "Самый лучший день" });

const comment\_1 = await Articles.create({id\_user:Roman.id, text:"Copatel online" });

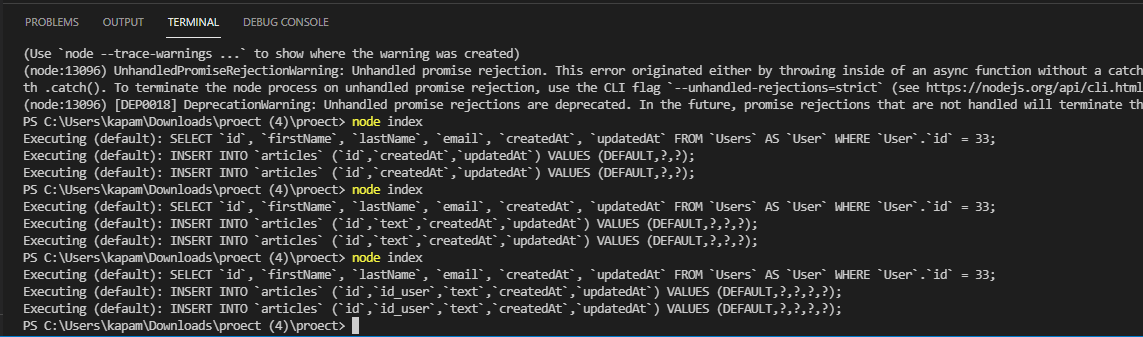


Рисунок 10 – Результат терминала

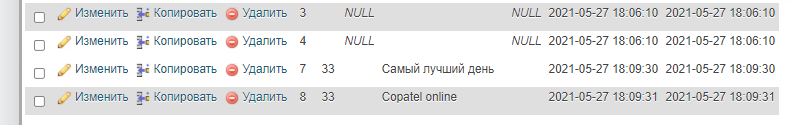


Рисунок 11 – Связь 1:N

Сделаем свзяь многие ко многим

Создадим модель связку articles\_user

npx sequelize-cli model:generate --name articles\_user --attributes id\_user:string,id\_articles:string

В модель user.js добавим:

User.belongsToMany(models. Articles,{through:'enty'});

В модель course.js добавим:

Articles.belongsToMany(models.User,{through:'enty'});

Добавим данные в таблицу связку

User.findOne({where: {id:33}})

.then(user=>{

if(!user) return;

// добавим Тому курс по JavaScript

Articles.findOne({where: {id:7}})

.then(Articles=>{

if(!Articles) return;

User.addArticles(Articles, {through: { price:"1000"}});

});

});



Рисунок 12 – Добавление данных в таблицу связку