МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Базы данных»

Тема: Реализация базы данных с использованием ORM.

Студент гр. 1303 _____ Депрейс. А.С.

Преподаватель _____ Заславский М.М.

Санкт-Петербург
2023

Цель работы.

Создание базы данных с использованием Object-Relational Mapping (ORM).

Задание.

Вариант 7

- Описать в виде моделей Sequelize таблицы из 1-й лабораторной работы.
- Написать скрипт заполнения тестовыми данными: 5-10 строк на каждую таблицу, обязательно наличие связи между ними, данные приближены к реальности.
- Написать запросы к БД, отвечающие на вопросы из 1-й лабораторной работы с использованием ORM. Вывести результаты в консоль (или иной человеко-читабельный вывод)
- Запушить в репозиторий исходный код проекта, соблюсти .gitignore, убрать исходную базу из проекта (или иные нагенерированные данные бд если они есть).
- Описать процесс запуска: команды, зависимости
- В отчете описать цель, текст задания в соответствии с вариантом, выбранную ORM, инструкцию по запуску, скриншоты (код) моделей ORM, скриншоты на каждый запрос (или группу запросов) на изменение/таблицы с выводом результатов (ответ), ссылку на PR в приложении, вывод.

Выполнение работы.

Для работы были установлены следующие пакеты:

```
"pg": "^8.11.3",
"sequelize": "^6.34.0",
"sequelize-typescript": "^2.1.5"
```

Для запуска необходимо скомпилировать TypeScript и запустить app.js.

```
@Table( options: {
    tableName: "breed",
    timestamps: false
})
export class Breed extends Model implements BreedI{

    @PrimaryKey
    @Column(DataType.TEXT)
    breed_name: string;

    @AllowNull( allowNull: false)
    @Column( options: {
        type:DataType.INTEGER,
    })
    average_eggs_per_month: number;

    @AllowNull( allowNull: false)
    @Column( options: {
        type:DataType.REAL,
    })
    average_weight: number;

    @AllowNull( allowNull: false)
    @Column( options: {
        type:DataType.INTEGER,
    })
    recommended_diet_number: number;

    @HasMany( associatedClassGetter: () => Chicken)
    chickens: Chicken[];
}
```

Рисунок 1. – Модель породы

```
@Table( options: {
    tableName: "chicken",
    timestamps: false
})
export class Chicken extends Model implements ChickenI{

    @AutoIncrement
    @PrimaryKey
    @Column(DataType.INTEGER)
    chicken_id?: number;

    @AllowNull( allowNull: false)
    @Column( options: {
        type:DataType.REAL,
    })
    weight: number;

    @AllowNull( allowNull: false)
    @Column( options: {
        type:DataType.INTEGER,
    })
    age: number;

    @AllowNull( allowNull: false)
    @Column(DataType.INTEGER)
    eggs_per_month: number;
```

```
  @AllowNull( allowNull: false)
  @ForeignKey( relatedClassGetter: () => Breed)
  @Column
  breed_name: string;

  @BelongsTo( associatedClassGetter: () => Breed)
  breed: ReturnType<() => Breed>;

  @HasOne( associatedClassGetter: () => WatchingChickenInCage)
  watching_chicken_in_cages: WatchingChickenInCage[];
```

Рисунок 2. – Модель курицы

```
export class Worker extends Model implements WorkerI{

    @AutoIncrement
    @PrimaryKey
    @Column(DataType.INTEGER)
    worker_id?: number;

    @AllowNull( allowNull: false)
    @Column( options: {
        type:DataType.INTEGER,
        unique: "passport"
    })
    passport_id: number;

    @AllowNull( allowNull: false)
    @Column( options: {
        type:DataType.SMALLINT,
        unique: "passport"
    })
    passport_series: number;
```

```
@AllowNull( allowNull: false)
@Column(DataType.INTEGER)
salary: number;

@AllowNull( allowNull: false)
@Column(DataType.TEXT)
name: string;

@AllowNull( allowNull: false)
@Column(DataType.TEXT)
surname: string;

@AllowNull( allowNull: true)
@Column(DataType.TEXT)
patronymic: string;

@HasMany( associatedClassGetter: () => WatchingChickenInCage)
watching_chicken_in_cages: WatchingChickenInCage[];
```

Рисунок 3. – Модель рабочего.

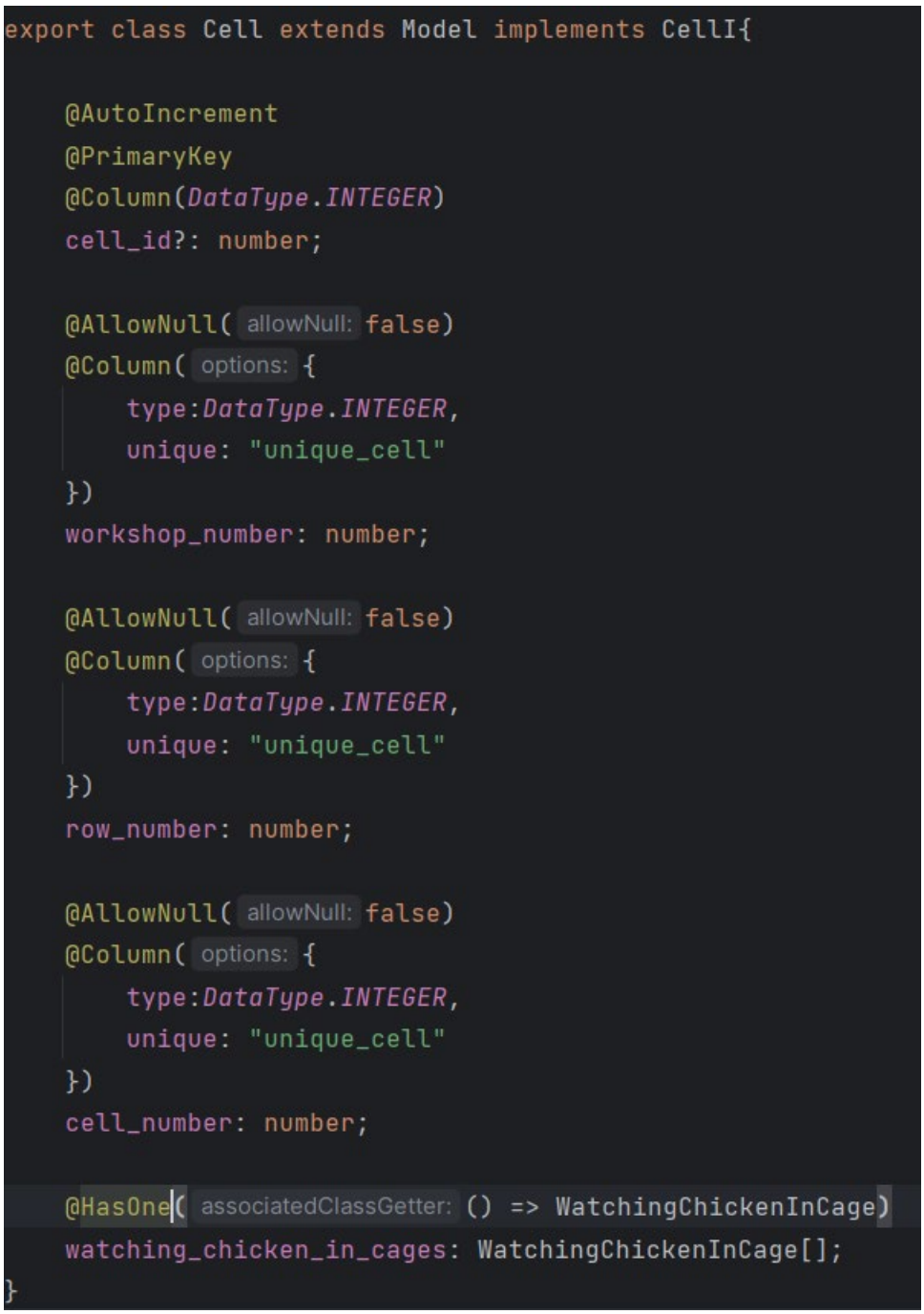

Рисунок 4. – Модель клетки.

```
export class WatchingChickenInCage extends Model implements WatchingChickenInCageI{

    @AutoIncrement
    @PrimaryKey
    @Column(DataType.INTEGER)
    watching_chicken_in_cage_id?: number;

    @AllowNull( allowNull: false)
    @ForeignKey( relatedClassGetter: () => Worker)
    @Column( options: {
        type:DataType.INTEGER,
        unique: "WatchingChickenInCage"
    })
    worker_id: number;

    @AllowNull( allowNull: false)
    @ForeignKey( relatedClassGetter: () => Cell)
    @Column( options: {
        type:DataType.INTEGER,
        unique: "WatchingChickenInCage"
    })
    cell_id: number;

    @AllowNull( allowNull: false)
    @ForeignKey( relatedClassGetter: () => Chicken)
    @Column( options: {
        type:DataType.INTEGER,
        unique: "WatchingChickenInCage"
    })
    chicken_id: number;

    @BelongsTo( associatedClassGetter: () => Worker)
    worker: ReturnType<() => Worker>;

    @BelongsTo( associatedClassGetter: () => Cell)
    cell: ReturnType<() => Cell>;

    @BelongsTo( associatedClassGetter: () => Chicken)
    chicken: ReturnType<() => Chicken>;
```

Рисунок 5. – Модель (наблюдает за курицей в клетке).

Рассмотрим запросы на заполнение:

```
await Worker.bulkCreate( records: [
    {worker_id: 1, passport_series: 1234, passport_id: 345123,
        salary: 12020, name: 'Депрейс', surname: 'Александр', patronymic: 'Сергеевич'},
    {worker_id: 2, passport_series: 1232, passport_id: 740593,
        salary: 12500, name: 'Безрукова', surname: 'Софья', patronymic: 'Ивановна'},
    {worker_id: 3, passport_series: 9365, passport_id: 376945,
        salary: 12300, name: 'Мельникова', surname: 'Маргарита', patronymic: 'Максимовна'},
    {worker_id: 4, passport_series: 2754, passport_id: 745788,
        salary: 13000, name: 'Волков', surname: 'Александр', patronymic: 'Андреевич'},
    {worker_id: 5, passport_series: 3437, passport_id: 845375,
        salary: 15200, name: 'Калашников', surname: 'Никита', patronymic: 'Николаевич'},
    {worker_id: 6, passport_series: 9374, passport_id: 883654,
        salary: 12023, name: 'Кондратьев', surname: 'Никита', patronymic: null},
    {worker_id: 7, passport_series: 6439, passport_id: 234666,
        salary: 12490, name: 'Спиридонова', surname: 'София', patronymic: null},
    {worker_id: 8, passport_series: 4386, passport_id: 636485,
        salary: 12850, name: 'Афанасьева', surname: 'Милана', patronymic: 'Данииловна'},
    {worker_id: 9, passport_series: 3567, passport_id: 528473,
        salary: 11500, name: 'Исаев', surname: 'Георгий', patronymic: 'Максимович'},
    {worker_id: 10, passport_series: 8756, passport_id: 235747,
        salary: 12050, name: 'Овсянни ков', surname: 'Михаил', patronymic: 'Серафимович'}
]);
```

Рисунок 6. – Запрос на заполнение.

```
await Cell.bulkCreate( records: [
    {cell_id:null, workshop_number:1,row_number:1,cell_number:1},
    {cell_id:null, workshop_number:1,row_number:2,cell_number:1},
    {cell_id:null, workshop_number:1,row_number:2,cell_number:2},
    {cell_id:null, workshop_number:1,row_number:2,cell_number:3},
    {cell_id:null, workshop_number:2,row_number:1,cell_number:1},
    {cell_id:null, workshop_number:2,row_number:2,cell_number:1},
    {cell_id:null, workshop_number:2,row_number:3,cell_number:1},
    {cell_id:null, workshop_number:2,row_number:3,cell_number:2},
    {cell_id:null, workshop_number:3,row_number:1,cell_number:1},
    {cell_id:null, workshop_number:3,row_number:1,cell_number:2}
]);
```

Рисунок 7. – Запрос на заполнение.

```
await Breed.bulkCreate( records: [
    {breed_name:'Адлерская серебристая курица',
        average_eggs_per_month: 15,average_weight: 2.65,recommended_diet_number: 1},
    {breed_name:'Леггорн',average_eggs_per_month: 16,average_weight: 1.75,recommended_diet_number: 2},
    {breed_name:'Доминант',average_eggs_per_month: 25,average_weight: 2.15,recommended_diet_number: 2},
    {breed_name:'Ломан Браун',average_eggs_per_month: 26,average_weight: 1.8,recommended_diet_number: 2},
    {breed_name:'Орловские куры',average_eggs_per_month: 13,average_weight: 2.5,recommended_diet_number: 1},
    {breed_name:'Хай Лайн',average_eggs_per_month: 21,average_weight: 1.32,recommended_diet_number: 1},
    {breed_name:'Маран', average_eggs_per_month:12,average_weight: 2.9,recommended_diet_number: 3},
    {breed_name:'Легбар',average_eggs_per_month: 17,average_weight: 2.35,recommended_diet_number: 3},
    {breed_name:'Араукан',average_eggs_per_month: 13,average_weight: 1.6,recommended_diet_number: 3},
    {breed_name:'Русская белая',average_eggs_per_month: 18,average_weight: 2.1,recommended_diet_number: 1}
]);
```

Рисунок 8. – Запрос на заполнение.

```
await Chicken.bulkCreate( records: [
    {chicken_id:null,weight: 2.65,age: 1,eggs_per_month: 13,breed_name: 'Адлерская серебристая курица'},
    {chicken_id:null,weight: 2.33,age: 3,eggs_per_month: 14,breed_name: 'Адлерская серебристая курица'},
    {chicken_id:null,weight: 1.65,age: 3,eggs_per_month: 23,breed_name: 'Ломан Браун'},
    {chicken_id:null,weight: 2.65,age: 2,eggs_per_month: 12,breed_name: 'Орловские куры'},
    {chicken_id:null,weight: 2.13,age: 5,eggs_per_month: 15,breed_name: 'Орловские куры'},
    {chicken_id:null,weight: 2.48,age: 2,eggs_per_month: 11,breed_name: 'Орловские куры'},
    {chicken_id:null,weight: 2.8,age: 2,eggs_per_month: 13,breed_name: 'Маран'},
    {chicken_id:null,weight: 2.9,age: 2,eggs_per_month: 11,breed_name: 'Маран'},
    {chicken_id:null,weight: 1.95,age: 1,eggs_per_month: 18,breed_name: 'Русская белая'},
    {chicken_id:null,weight: 1.95,age: 1,eggs_per_month: 21,breed_name: 'Русская белая'}
]);
```

Рисунок 9. – Запрос на заполнение.

```
await  WatchingChickenInCage.bulkCreate( records: [
    {watching_chicken_in_cage_id: null,worker_id: 1,cell_id:1,chicken_id:9},
    {watching_chicken_in_cage_id: null,worker_id: 2,cell_id:2,chicken_id:3},
    {watching_chicken_in_cage_id: null,worker_id: 2,cell_id:3,chicken_id:7},
    {watching_chicken_in_cage_id: null,worker_id: 2,cell_id:4,chicken_id:8},
    {watching_chicken_in_cage_id: null,worker_id: 3,cell_id:5,chicken_id:1},
    {watching_chicken_in_cage_id: null,worker_id: 4,cell_id:6,chicken_id:2},
    {watching_chicken_in_cage_id: null,worker_id: 5,cell_id:7,chicken_id:6},
    {watching_chicken_in_cage_id: null,worker_id: 6,cell_id:8,chicken_id:4},
    {watching_chicken_in_cage_id: null,worker_id: 6,cell_id:9,chicken_id:5},
    {watching_chicken_in_cage_id: null,worker_id: 7,cell_id:10,chicken_id:10}
]);
```

Рисунок 10. – Запрос на заполнение.

Рассмотрим запросы из задания:

```
await Chicken.findAll( options: {
    attributes: ["chicken_id", "eggs_per_month"],
    where: {
        weight: {
            [Op.and]: {
                [Op.lt]: 1.95 + 0.0001,
                [Op.gt]: 1.95 - 0.0001,
            }
        }
    }
}).then((res : Chicken[] ) : void  => {
```

Рисунок 11. – Какое количество яиц получают от каждой курицы данного веса, породы, возраста?

```
await Cell.findAll( options: {
    attributes:["workshop_number"],
    include:[
        {
            model: WatchingChickenInCage,
            required: true,
            attributes: [],
            include:[
                {
                    model: Chicken,
                    required: true,
                    attributes:[],
                    where:{ breed_name: 'Орловские куры' }
                }
            ]
        },
    ],

    group: ['workshop_number'],
    order: [[sequelize.fn( fn: 'COUNT', sequelize.col( col: 'workshop_number')), 'DESC']],
    limit: 1,
}).then((res : Cell[] ) : void  => {
```

Рисунок 12. – В каком цехе наибольшее количество кур определенной породы?

Рисунок 13. – В каких клетках находятся куры указанного возраста с заданным номером диеты?

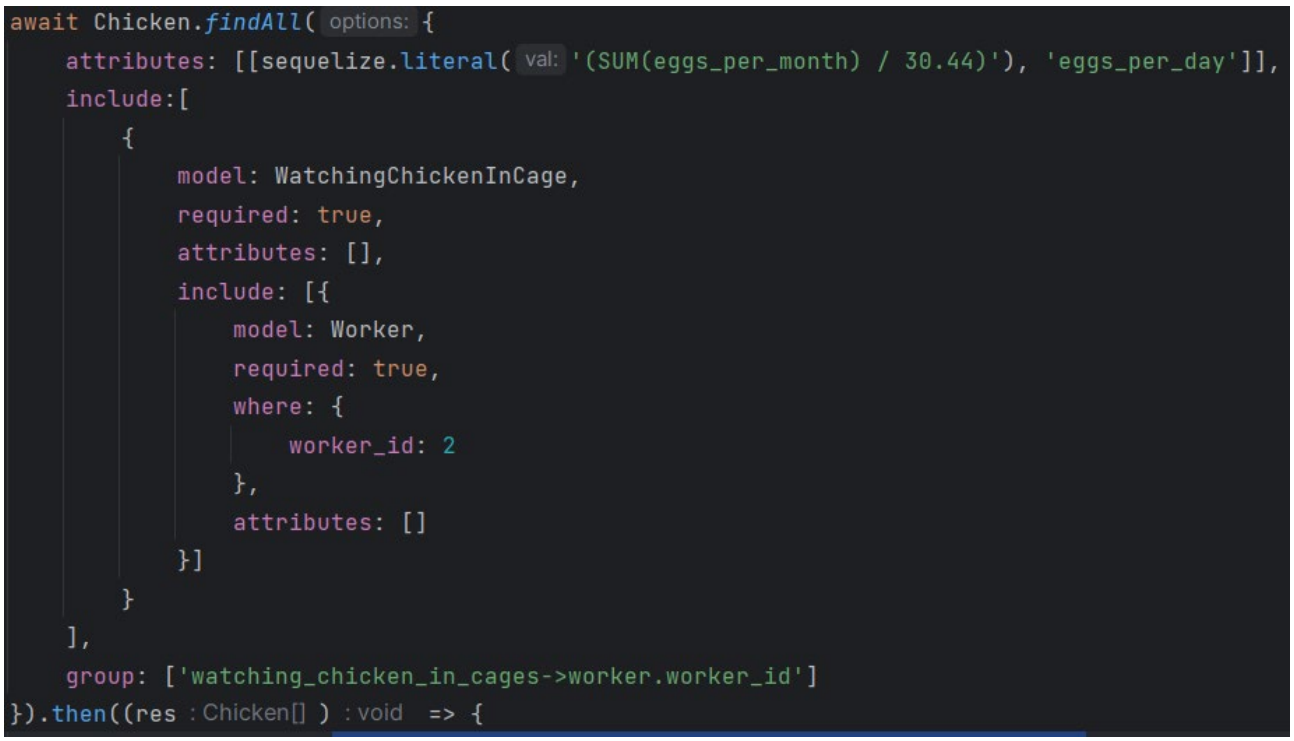

Рисунок 14. – Сколько яиц в день приносят куры указанного работника?

Рисунок 15. – Среднее количество яиц, которое получает в день каждый работник от обслуживаемых им кур?

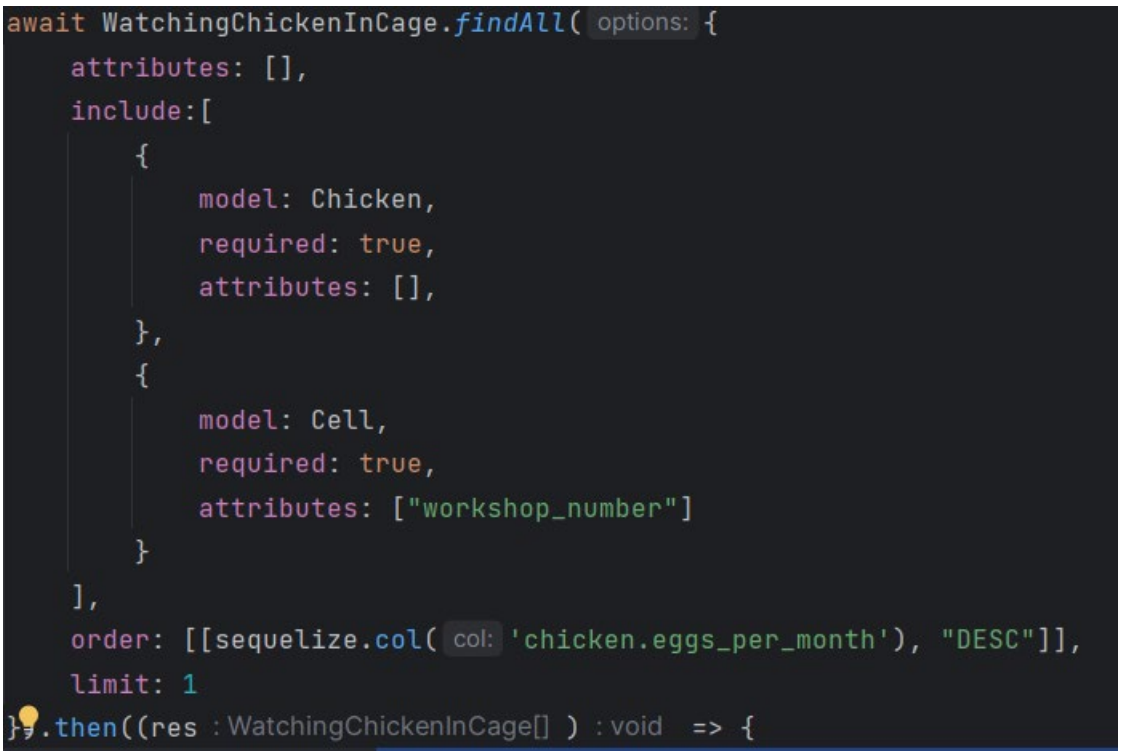

Рисунок 16. – В каком цехе находится курица, от которой получают больше всего яиц?

Рисунок 17. – Сколько кур каждой породы в каждом цехе?

```
await Worker.findAll( options: {
    attributes: ["worker_id",
        [sequelize.fn( fn: "COUNT", sequelize.col( col: "watching_chicken_in_cage_id")), "count"]],
    include:[
        {
            model: WatchingChickenInCage,
            required: false,
            attributes: []
        }
    ],
    group: ["Worker.worker_id"],
    order: [[sequelize.col( col: 'Worker.worker_id'), "ASC"]]
}).then((res : Worker[] ) : void  => {
```

Рисунок 18. – Какое количество кур обслуживает каждый работник?

Рассмотрим ответы на запросы:

```
Какое количество яиц получают от каждой курицы данного веса, породы, возраста?
[
  {
    "chicken_id": 9,
    "eggs_per_month": 18
  },
  {
    "chicken_id": 10,
    "eggs_per_month": 21
  }
]

В каком цехе наибольшее количество кур определенной породы?
[
  {
    "workshop_number": 2
  }
]
```

Рисунок 19. – Ответы на запросы

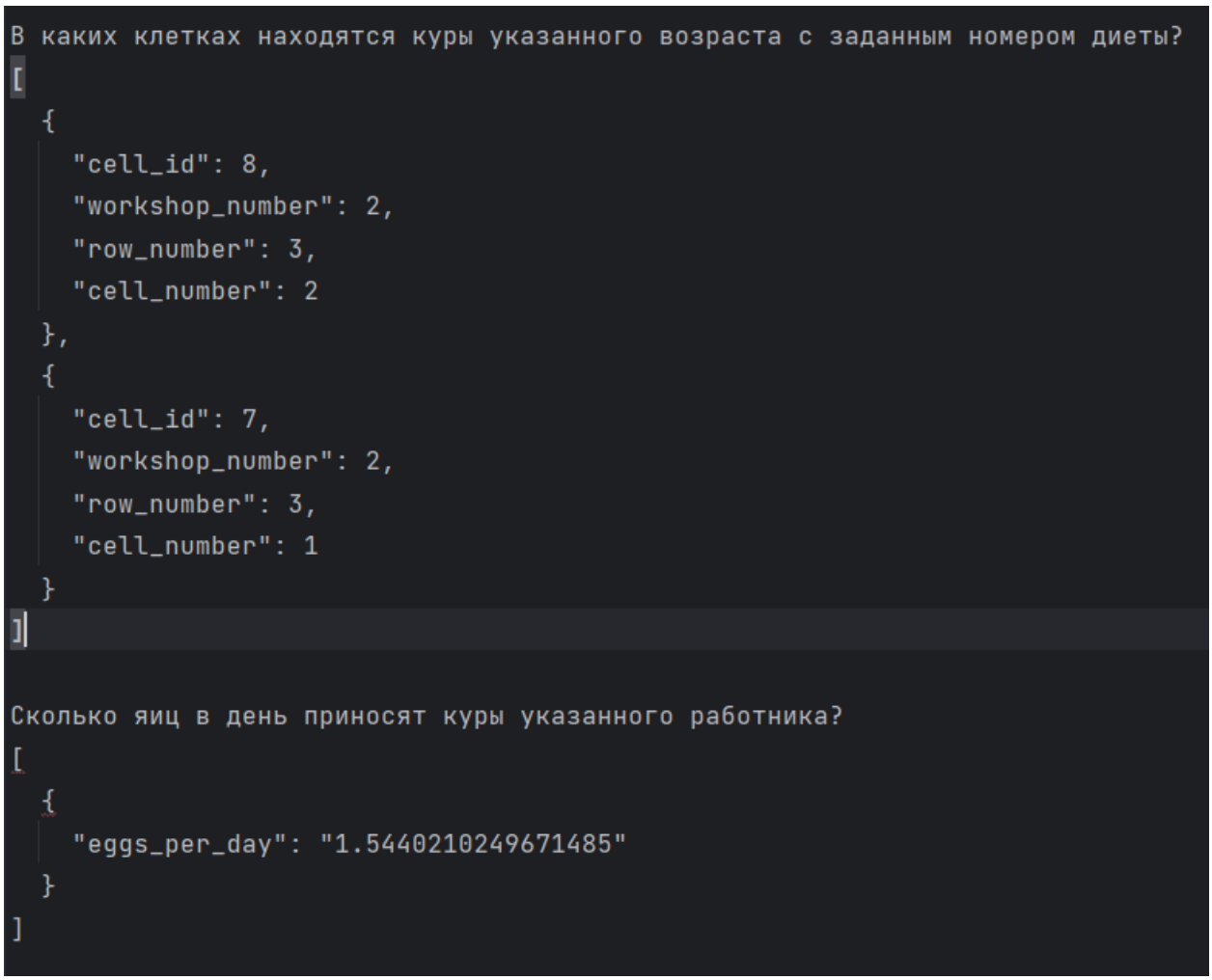

Рисунок 20. – Ответы на запросы

```
Среднее количество яиц, которое получает в день каждый работник от обслуживаемых им кур?
[
  {
    "worker_id": 3,
    "average_eggs_per_day": "0.49277266754270696452"
  },
  {
    "worker_id": 5,
    "average_eggs_per_day": "0.42706964520367936925"
  },
  {
    "worker_id": 4,
    "average_eggs_per_day": "0.49277266754270696452"
  },
  {
    "worker_id": 6,
    "average_eggs_per_day": "0.85413929040735873850"
  },
  {
    "worker_id": 2,
    "average_eggs_per_day": "1.6425755584756899"
  },
  {
    "worker_id": 7,
    "average_eggs_per_day": "0.59132720105124835742"
  },
  {
    "worker_id": 1,
    "average_eggs_per_day": "0.59132720105124835742"
  }
]
```


Рисунок 21. – Ответы на запросы


```
В каком цехе находится курица, от которой получают больше всего яиц?
[
  {
    "cell": {
      "workshop_number": 1
    }
  }
]
```


Рисунок 22. – Ответы на запросы

Сколько кур каждой породы в каждом цехе?


```
[
  {
    "count": "1",
    "workshop_number": 1,
    "breed": {
      "breed_name": "Ломан Браун"
    }
  },
  {
    "count": "2",
    "workshop_number": 1,
    "breed": {
      "breed_name": "Маран"
    }
  },
  {
    "count": "1",
    "workshop_number": 1,
    "breed": {
      "breed_name": "Русская белая"
    }
  },
  {
    "count": "2",
    "workshop_number": 2,
    "breed": {
      "breed_name": "Адлерская серебристая курица"
    }
  },
```

```
[
  {
    "count": "2",
    "workshop_number": 2,
    "breed": {
      "breed_name": "Орловские куры"
    }
  },
  {
    "count": "1",
    "workshop_number": 3,
    "breed": {
      "breed_name": "Орловские куры"
    }
  },
  {
    "count": "1",
    "workshop_number": 3,
    "breed": {
      "breed_name": "Русская белая"
    }
  }
]
```


Рисунок 23. – Ответы на запросы

```
Какое количество кур обслуживает каждый работни
[
  {
    "worker_id": 1,
    "count": "1"
  },
  {
    "worker_id": 2,
    "count": "3"
  },
  {
    "worker_id": 3,
    "count": "1"
  },
  {
    "worker_id": 4,
    "count": "1"
  },
  {
    "worker_id": 5,
    "count": "1"
  },
  {
    "worker_id": 6,
    "count": "2"
  },
```

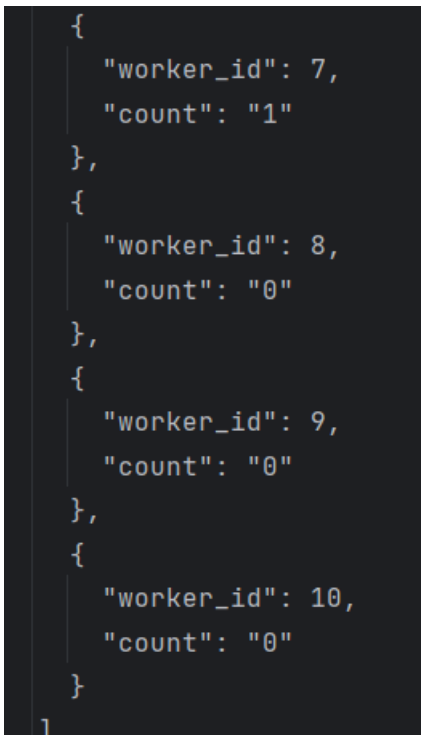

Рисунок 24. – Ответы на запросы

Выводы.

В данной лабораторной работе освоена работа с ORM для Node.js – Sequelize.

ПРИЛОЖЕНИЕ А

ССЫЛКИ

Pull Request: https://github.com/moevm/sql-2023-1303/pull/40