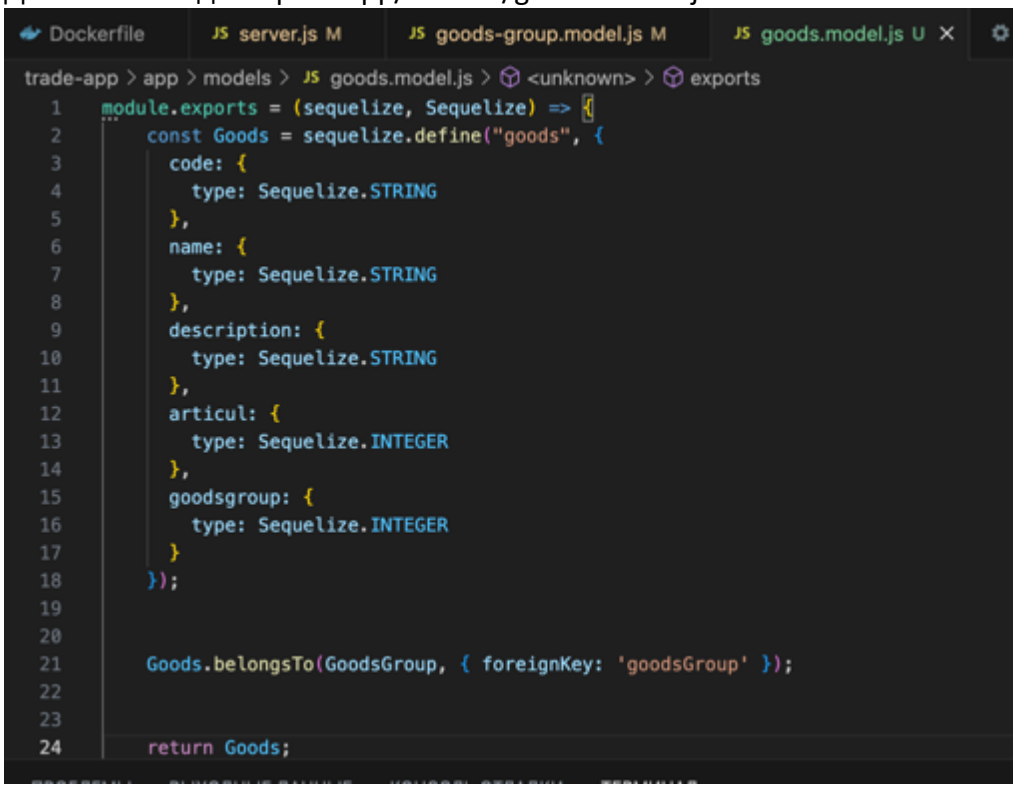
Лабораторная работа 10

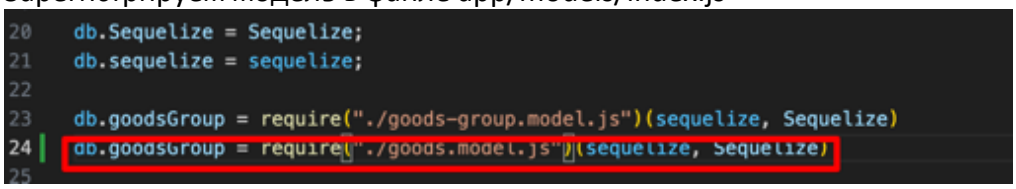
Формирование связей сущностей

2.6 Доработать остальные таблицы БД
2.6.1 Товар

Добавим в модель файл app/models/goods.model.js



Зарегистрируем модель в файле app/models/index.js



Создадим новый файл app/models/references.model.js для определения связей между сущностями приложения.

NOTE!!
Дополнительно прочитать про виды связей сущностей в фреймворке Sequelize можно по ссылке https://sequelize.org/docs/v6/core-concepts/assocs/
В частности,

Отношения «один ко многим»
Философия
Связи «один ко многим» связывают один источник с несколькими целевыми объектами, при этом все эти целевые объекты связаны только с этим единственным источником.

Это означает, что, в отличие от связи «один к одному», в которой нам приходилось выбирать, где будет расположен внешний ключ, в связях «один ко многим» есть только один вариант. Например, если у одного объекта Foo есть несколько объектов Bar (и, таким образом, каждый объект Bar принадлежит одному объекту Foo), то единственно разумная реализация — наличие столбца fooId в таблице Bar. Обратное невозможно, поскольку у одного объекта Foo есть несколько объектов Bar.

Цель
В этом примере у нас есть модели Team и Player. Мы хотим сообщить Sequelize, что между ними существует связь «один ко многим», то есть у одной команды есть несколько игроков, в то время как каждый игрок принадлежит только одной команде.

Реализация
Основной способ сделать это следующий:

```
Team.hasMany(Player); Player.belongsTo(Team);
```

Например, в PostgreSQL при выполнении sync() указанная выше конфигурация выдаст следующий SQL-запрос:

```
CREATE TABLE IF NOT EXISTS "Teams" (
/* ... */
);
CREATE TABLE IF NOT EXISTS "Players" (
/* ... */
"TeamId" INTEGER REFERENCES "Teams" ("id") ON DELETE SET NULL ON UPDATE CASCA
DE,
/* ... */
);
```

Добавим вызов файла связей в app/models/index.js

```
20  db.Sequelize = Sequelize;
21  db.sequelize = sequelize;
22
23  db.goodsGroup = require("./goods-group.model.js")(sequelize, Sequelize)
24  db.goods = require("./goods.model.js")(sequelize, Sequelize)
25
26  require('./references.model.js')(db)
27
28
```

Добавить файлы в репозиторий

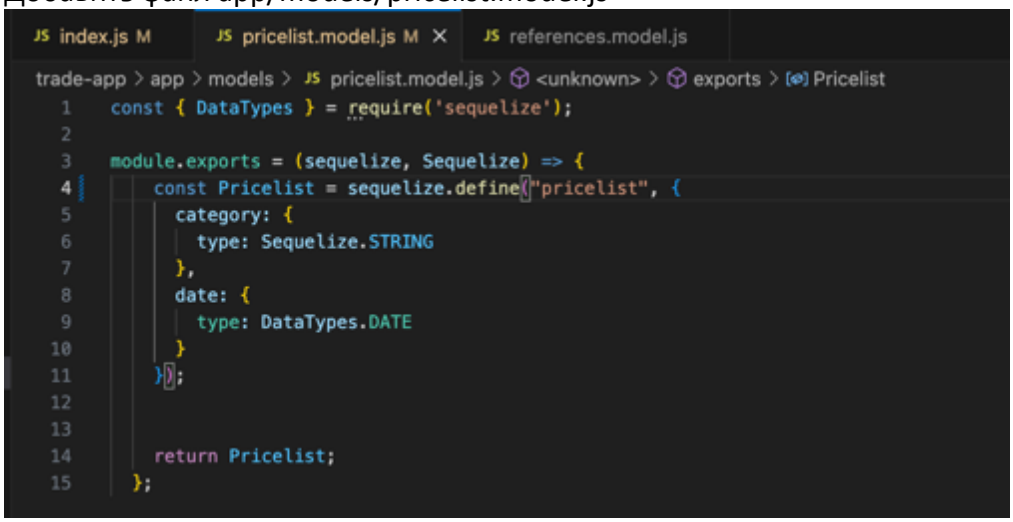
```
git add .
```

Выполнить комит изменений

```
git commit –am “goods model added”
```

6.2.2 Список цен

Добавить файл app/models/pricelist.model.js



NOTE!!
Мы использовали вспомогательный класс с описание типов данных для раоты с типом данных ДАТА DataTypes.DATE

Зарегистрировать модель в app/models.index.js

NOTE!!
Поскольку сущность pricelist не содержит внешних связей, то изменять файл app/models/references.js не нужно.

Добавить файлы в репозиторий

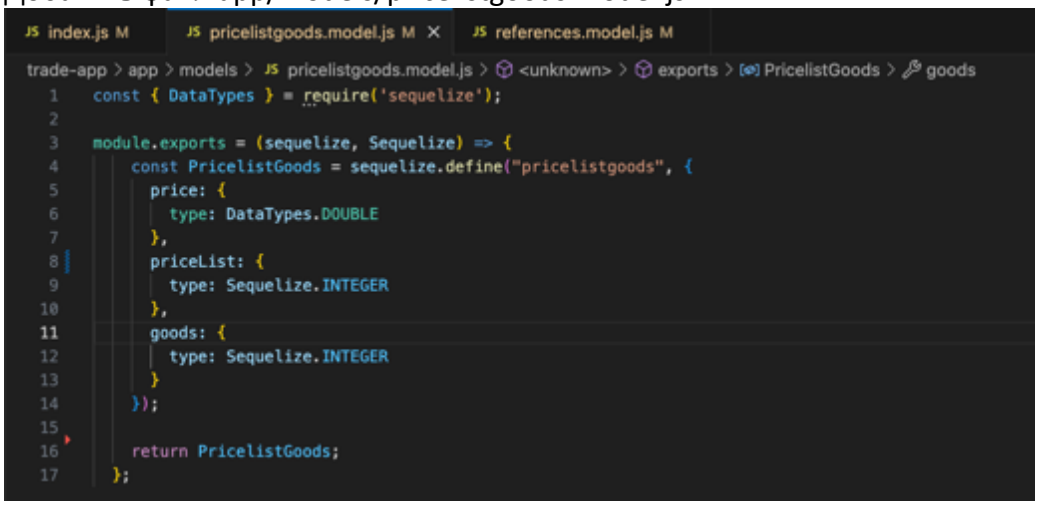
```
git add .
```

Выполнить комит изменений

```
git commit –am “pricelist model added”
```

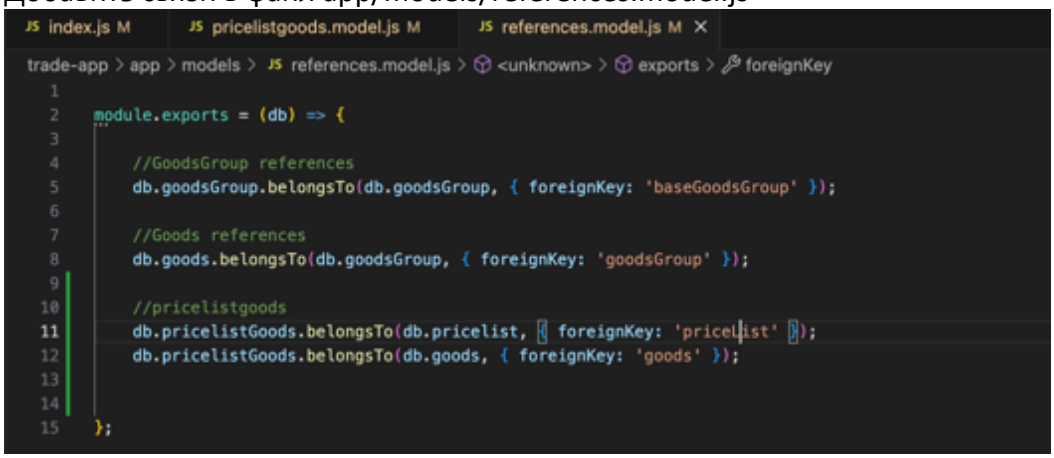
6.2.3 Товар в списке цен

Добавить файл app/models/pricelistgoods.model.js



Зарегистрировать файл модели в app/models/index.js

Добавить связи в файл app/models/references.model.js



Добавить файлы в репозиторий

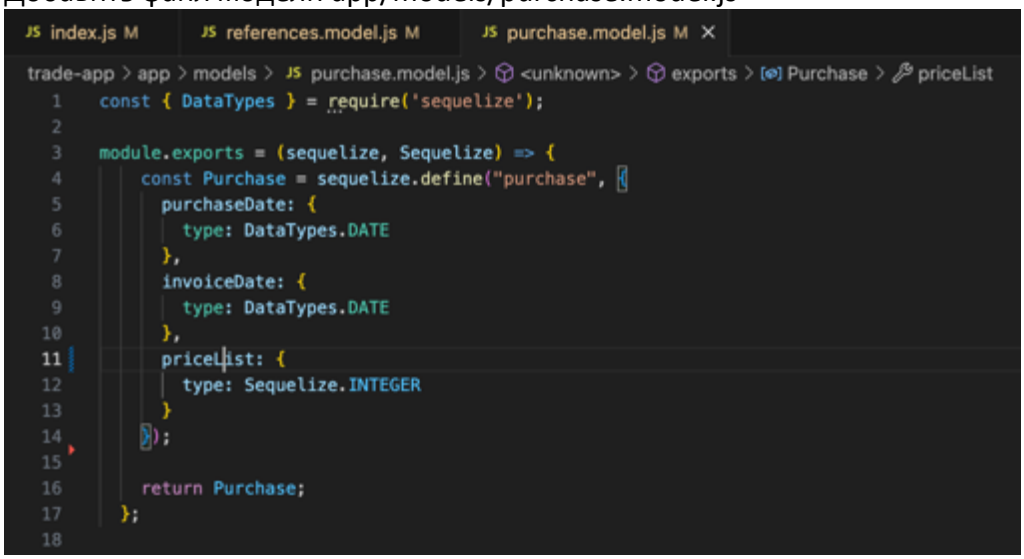
```
git add .
```

Выполнить комит изменений

```
git commit –am “pricelistgoods model added”
```

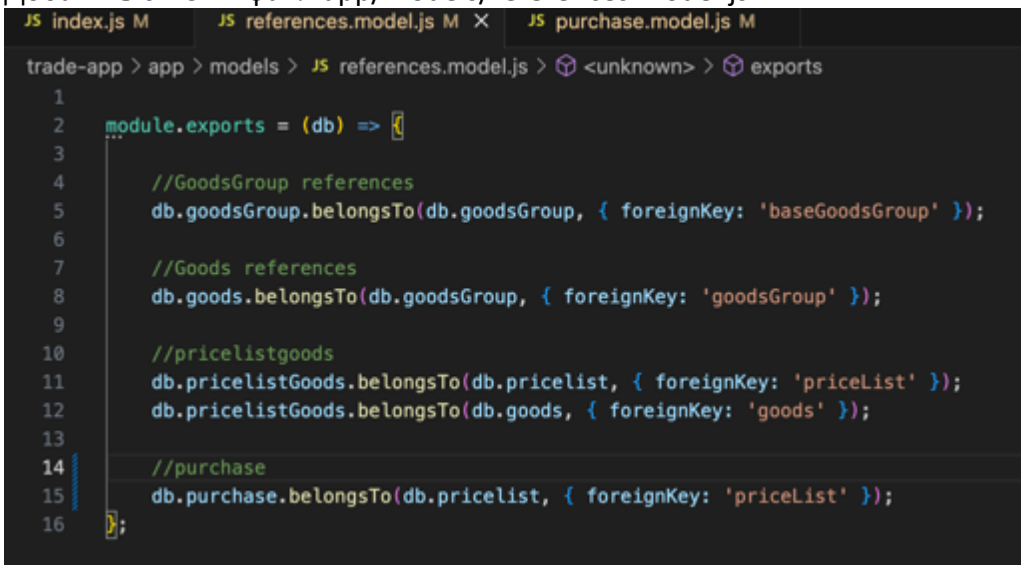
6.2.4 Продажа

Добавить файл модели app/models/purchase.model.js



Зарегистрировать модель продажи app/models/index.js

Добавить связи в файл app/models/references.model.js



Добавить файлы в репозиторий

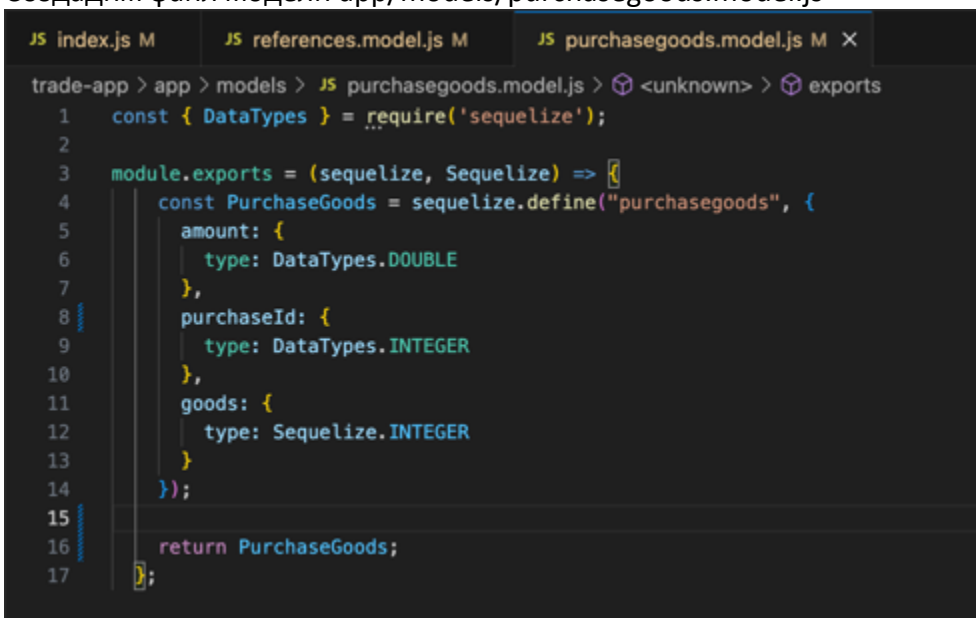
```
git add .
```

Выполнить комит изменений

```
git commit –am “purchase model added”
```

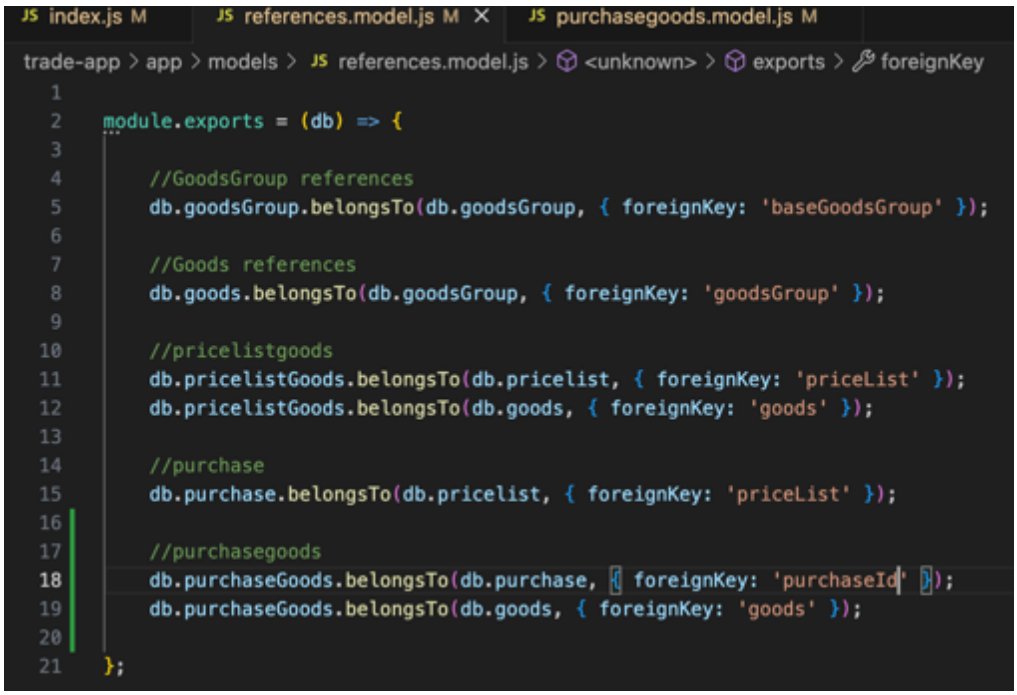
6.2.5 Товар в продаже

Создадим файл модели app/models/purchasegoods.model.js



Зарегистрировать модель в файле app/models/index.js

Добавить связи в файл app/models/references.model.js



Добавить файлы в репозиторий

```
git add .
```

Выполнить комит изменений

```
git commit –am “purchasegoods model added”
```

6.3 Запуск приложения

6.3.1 Синхронизировать данные в github

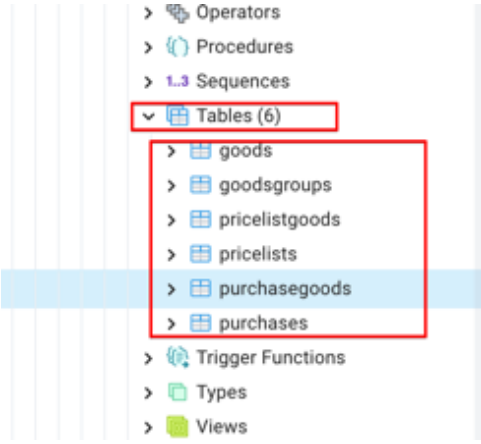
```
git push –u origin main
```

6.3.2 Запуск приложения

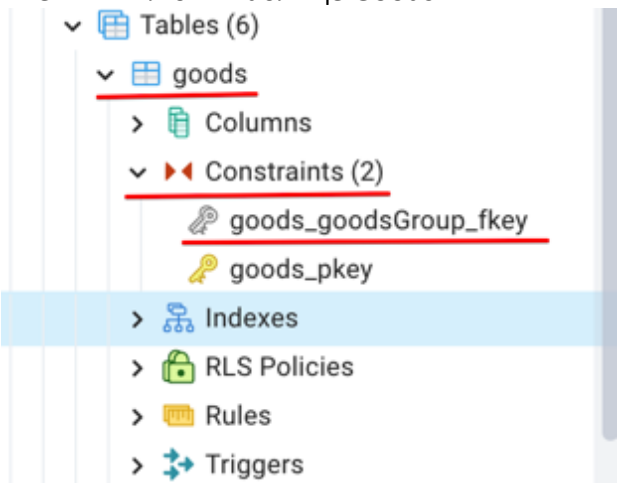
```
docker–compose up –d –build
```

6.4 Проверка корректности созданных сущностей и связей в БД

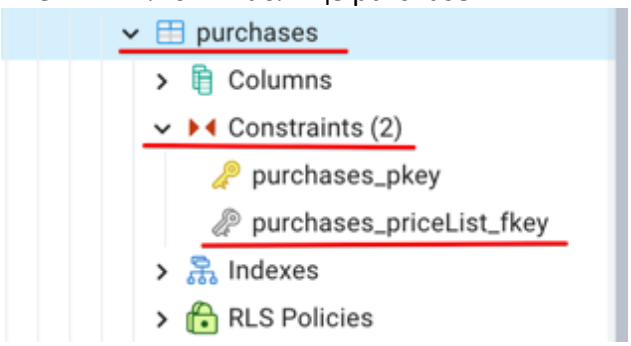
Подключиться клиентом БД к самой БД и проверить наличие сущностей и связей



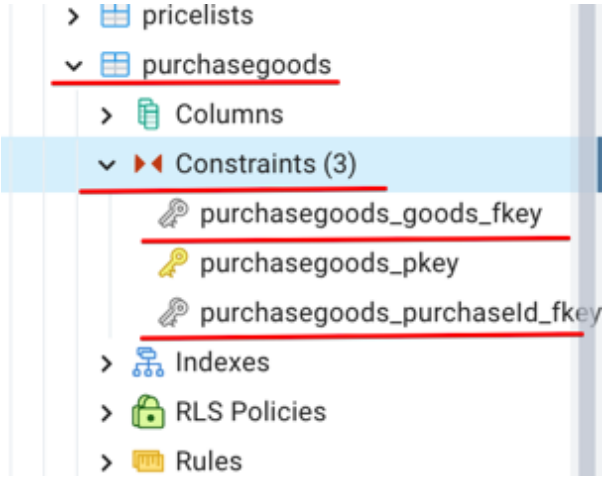
Проверяем связи
Внешний ключ в таблице Goods



Внешний ключ в таблице purchase



Внешние ключи в таблице purchasegoods



Задание для самостоятельного выполнения:

Определить связи для сущностей своего проекта