**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Саратовский государственный технический университет**

**имени Гагарина Ю. А.»**

Институт прикладных информационных технологий и коммуникаций

Направление «Информационные системы и технологии»

Кафедра «Прикладные информационные технологии»

**КОНТРОЛЬНАЯ РАБОТА**

по дисциплине «Нереляционные базы данных»

Выполнил студент группы

б1-ИФСТ-21

очной формы обучения

Яновский Евгений Валерьевич

Номер зачетной книжки 210940

Проверил: старший преподаватель

Левченко Д. С.

Саратов 2023

**Оглавление**

[**Введение** 3](#_Toc136040932)

[**Раздел 1** 4](#_Toc136040933)

[Глава 1.1 – Создание БД 4](#_Toc136040934)

[Глава 1.2 – Создание классов моделей 5](#_Toc136040935)

[Глава 1.3 – Создание интерфейсов репозиториев 7](#_Toc136040936)

[Глава 1.4 – Создание сервисов 8](#_Toc136040937)

[Глава 1.5 – Создание контроллеров 9](#_Toc136040938)

[Глава 1.5 – Создание функций агрегации 10](#_Toc136040939)

[**Раздел 2 (Проверка кода)** 11](#_Toc136040940)

[Раздел 2.1 – Авторизация пользователей с разным набором прав 11](#_Toc136040941)

[Раздел 2.3 – Проверка пользователей и их ролей 14](#_Toc136040942)

[Раздел 2.4 – Проверка работы доступа по ролям 18](#_Toc136040943)

[Раздел 2.5 – Проверка агрегатных функций 20](#_Toc136040944)

[**Вывод** 22](#_Toc136040945)

# **Введение**

**В работе будет продемонстрирована работа с нереляционной базой «MongoDB».**

**Цели работы**:

* Создать базу данных «ProductBase», предназначенной для хранения записей о продуктах в магазине.
* Продемонстрировать операции чтения, удаления, записи, обновления в базе.
* Продемонстрировать, какие привилегии имеют пользователи с различным набором прав, какие действия для них разрешены, а какие запрещены
* Создать авторизацию пользователей внутри базы данных, которая позволит производить вход в систему.
* Создать индексы, которые позволят ускорить запросы к базе, которые производятся чаще всего.
* Создать агрегатные функции, которые будут являться формой отчетности.
* Создать 2 роли пользователей («Администратор», «Пользователь»)

# **Раздел 1**

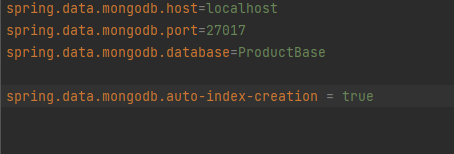
## Глава 1.1 – Создание БД

Для создания базы данных можно воспользоваться либо интерфейсом «Mongo Compass»,

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Либо в самом приложении создать **файл application properties**, в котором прописать все необходимые данные для подключения БД. Я выбрал второй вариант, так приложение, которое я пишу, включает в себя технологию «Spring Boot», **которая автоматически за меня пропишет конфигурационный файл для подключения к базе данных, используя данные для подключения из файла application properties.**



При этом необходимо **также разрешить создание индексов**, так как без этого мы не сможем их создать в дальнейшем, также при подключении, **если** указанной в файле конфигурации **базы нет**, **она создастся автоматически.**

## Глава 1.2 – Создание классов моделей

Для того чтобы было удобнее работать с данными и конвертировать их в объекты, мы создаем классы – модели, в которых описываем все необходимые поля, именно эти классы будут передаваться в репозитории – слой, который отвечает за выполнение запросов к базе.

Модель № 1 (**Product**)

**Book** (Книга) – это главная модель, с который мы будем работать, она содержит поля:

* Id
* Название
* Стоимость
* Дата добавления

Обратим внимание, что некоторые поля помечены аннотацией @Index, это значит именно по ним буде производится поиск в базе данных чаще всего. Добавлять продукты мы будем в коллекцию «products», для это над классом мы добавили аннотацию @Document. Класс модель содержит еще геттеры и сеттеры, необходимые для установки и получения значений, а также конструктор без параметров, для инициализации объекта.



Модель № 2 (**Privilege**)

**Privilege** – это вторая модель, с который мы будем работать, это объект, в котором мы будем хранить название ролей в наше базе данных, именно по ним будет определяться, какие права доступны разным пользователям.

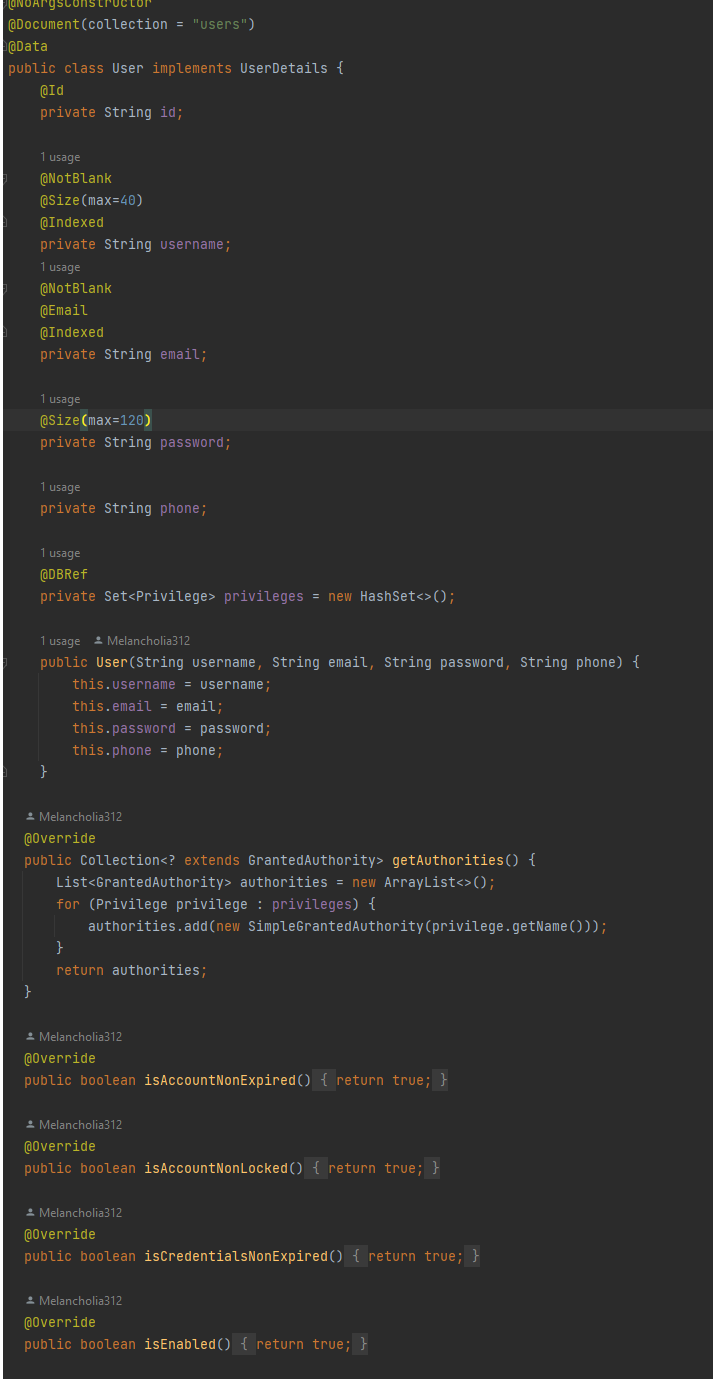
Всего в базе будет 2 роли пользователей:

* Пользователь(постоянный клиент магазина)
* Администратор



Модель № 4 (**User**)

**User** – это третья модель, с который мы будем работать, это класс , в котором мы будем хранить id пользователя, его имя, электронную почту, телефон и пароль и набор ролей, класс также содержит конструкторы с параметрами и без еще геттеры и сеттеры, необходимые для установки и получения значений.



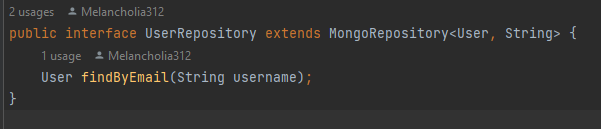
После того, как мы описали все классы модели, можно переходить к следующем шагу, а именно созданию репозиториев, которые будут непосредственно конвертировать данные из коллекций в объекты.

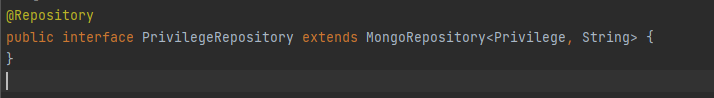
## Глава 1.3 – Создание интерфейсов репозиториев

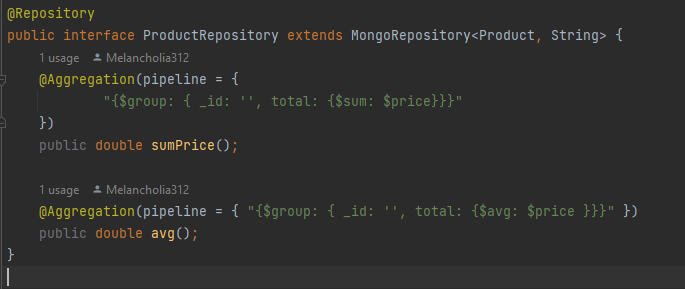
Как было сказано ранее, именно репозитории отвечают за представление данных из базы в виде объектов, с которыми дальше мы и будем работать.

Но для того, чтобы их реализовать, необходимы классы модели, именно в объекты этих классов и будет производиться запись данных.

У нас будет 3 репозитория, которые предоставляют помимо стандартных методов еще и функции агрегирования.





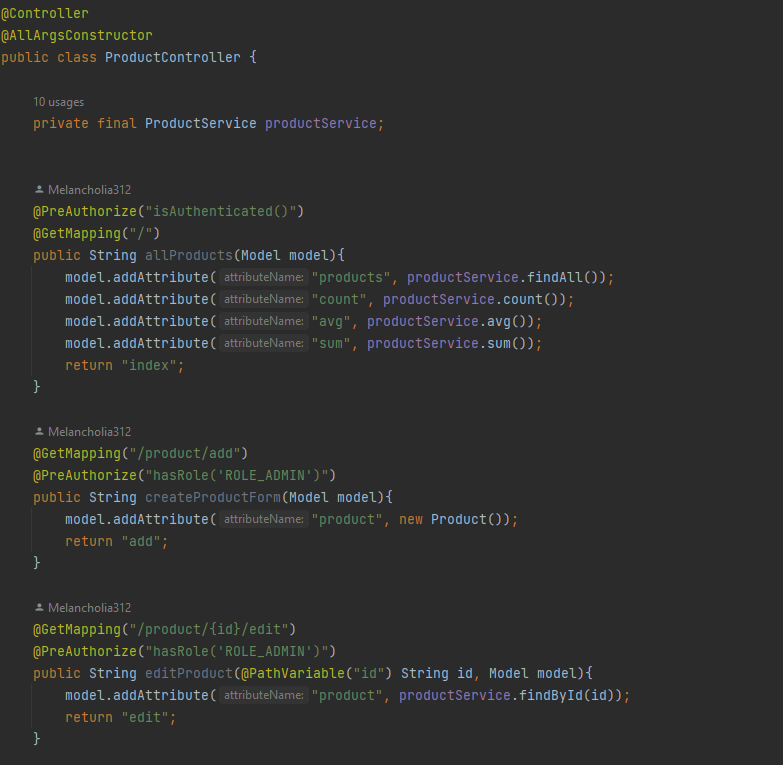


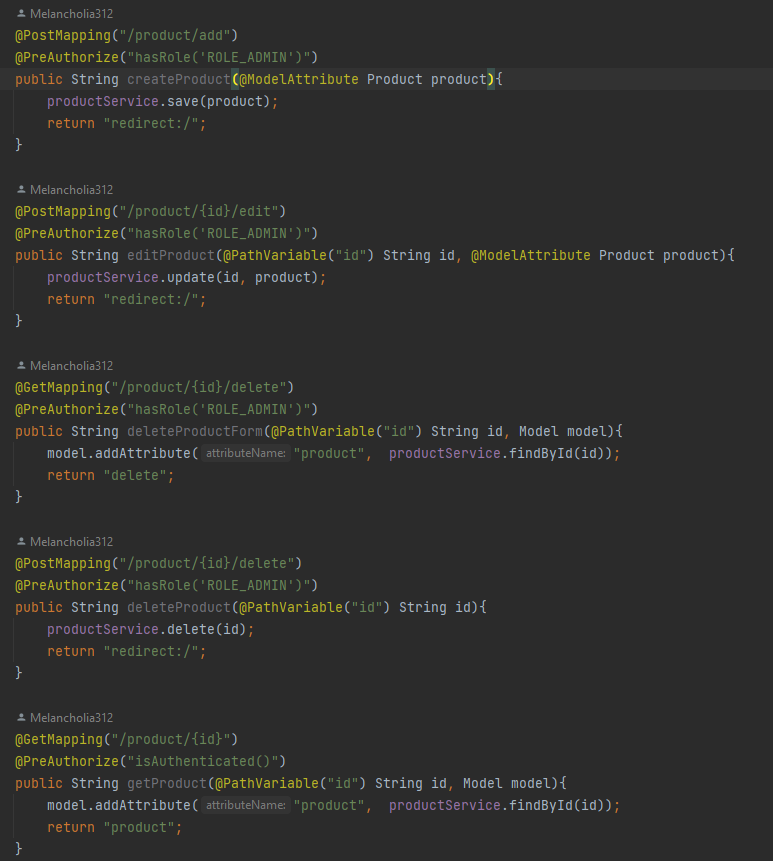
## Глава 1.4 – Создание сервисов



## Глава 1.5 – Создание контроллеров

В данной работе будет реализовано один контроллер, который реализует набор стандартных методов.

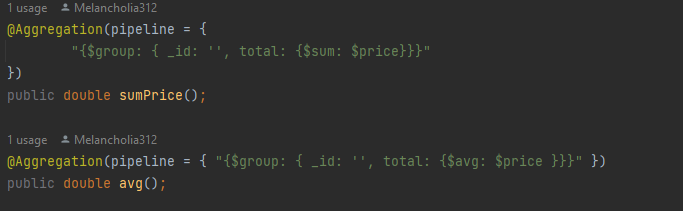




## Глава 1.5 – Создание функций агрегации

Все функции созданы для получения дополнительной информации о продуктах, будет предоставлено несколько примеров агрегатных функций.

Все функции создаются с помощью аннотации @Aggregation.



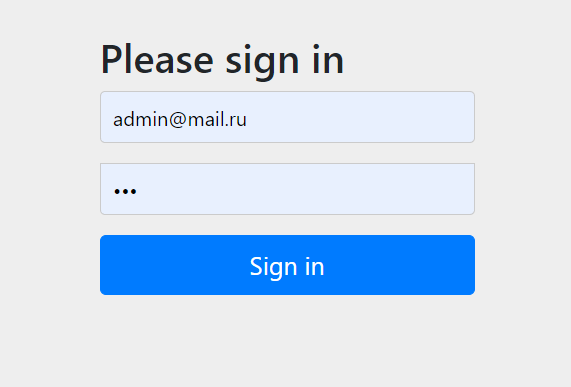
Метод sumPrice считает общую стоимость всех продуктов, а метод avg считает среднюю стоимость продукта.

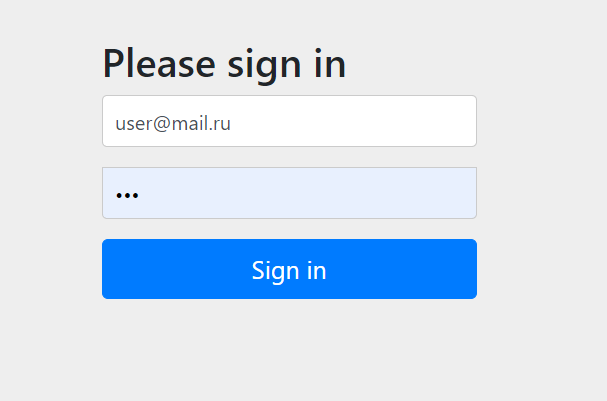
# **Раздел 2 (Проверка кода)**

Проверка будет осуществляться в браузере.

## Раздел 2.1 – Авторизация пользователей с разным набором прав

Авторизуем одного пользователя с ролью пользователя, а второго с ролью администратора.





При нажатии на кнопку нас переносит на главную страницу.

## 

## Раздел 2.3 – Проверка пользователей и их ролей

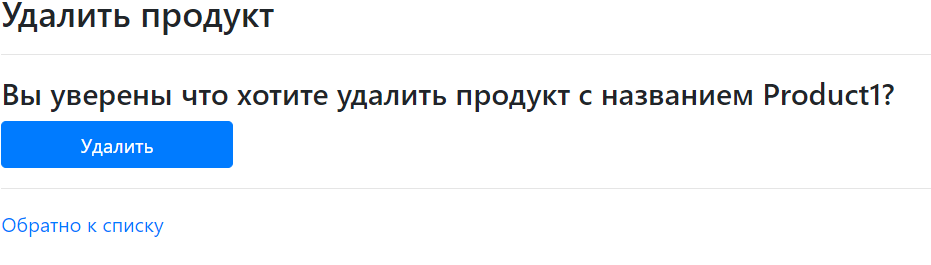


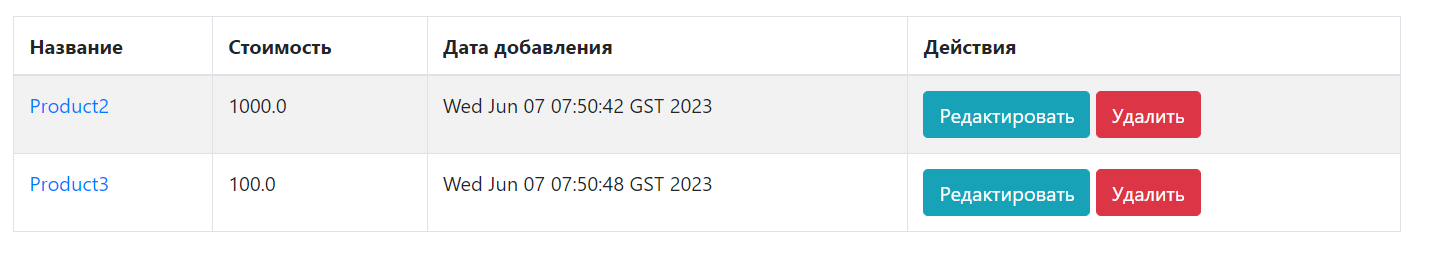
## Раздел 2.4 – Проверка работы доступа по ролям

Здесь мы попытались выполнить удаление продукта под ролью Пользователь, но так как это действие доступно лишь администратору, то получили 403 ошибку, которая означает, что у нас нет прав на это действие.

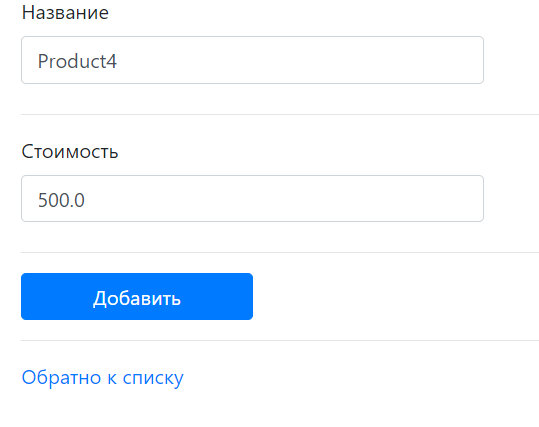


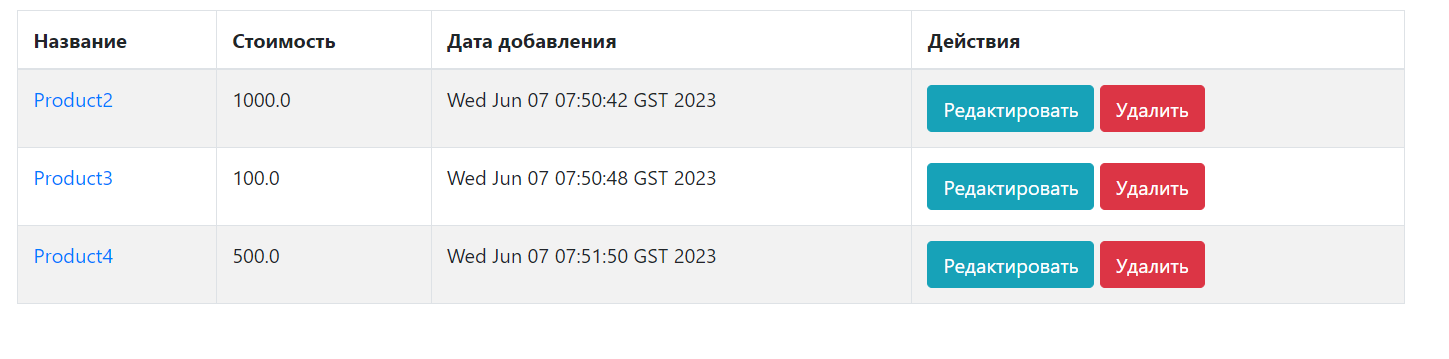
Здесь мы выполнили удаление продукта под ролью Админ, и все получилось.



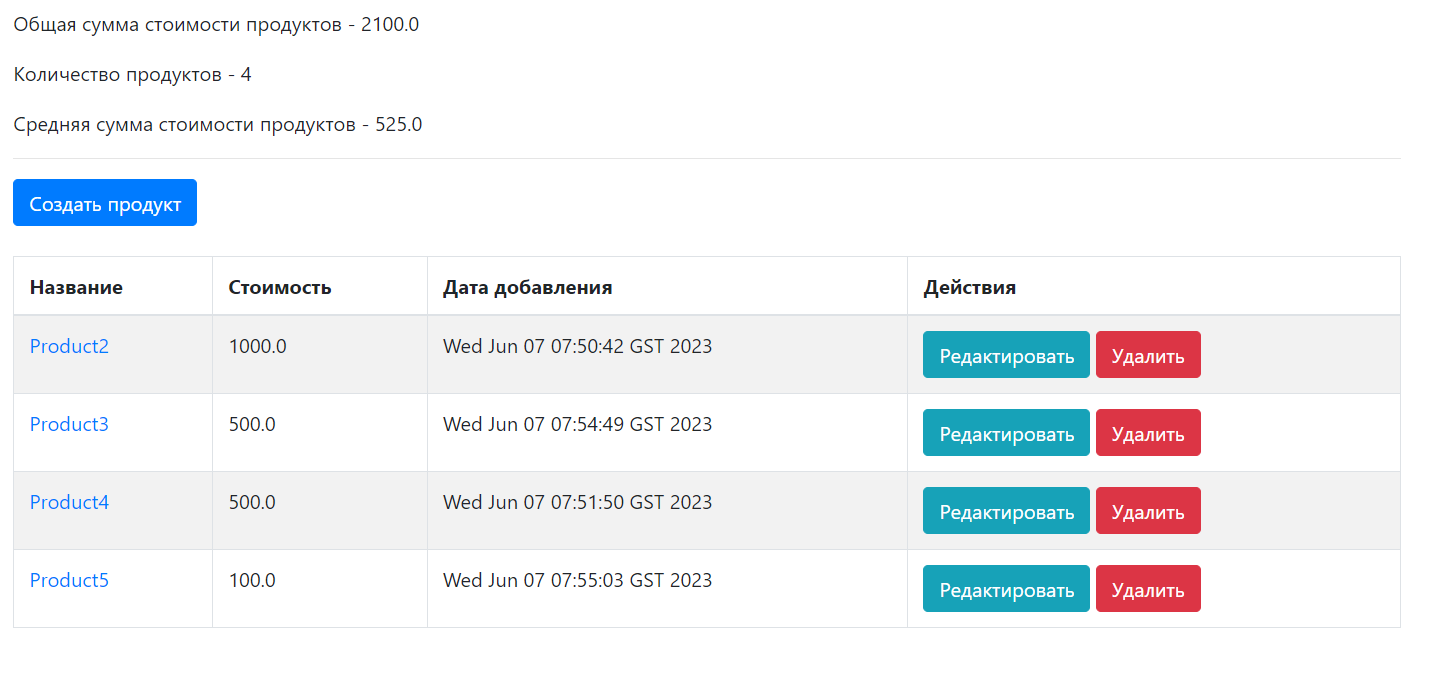


Здесь мы выполнили добавление продукта под ролью Админ, и все сработало

.



## Раздел 2.5 – Проверка агрегатных функций



Агрегатные функции работают корректно. Сумма, количество и средняя цена высчитаны верно.

# **Вывод**

Пользовательские роли отработали корректно. Все агрегатные функции выполнись успешно.