**КУРСОВАЯ РАБОТА**

**Разработка прототипа клиент-ориентированного приложения**

**для торгового предприятия**

по дисциплине «Базы данных»

Выполнил:

Студент гр. 3530904/80105  Куксов Г.В.

Руководитель: Гасанова И.А.

**Задание**

ЗАДАНИЕ

НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА (КУРСОВОЙ РАБОТЫ)

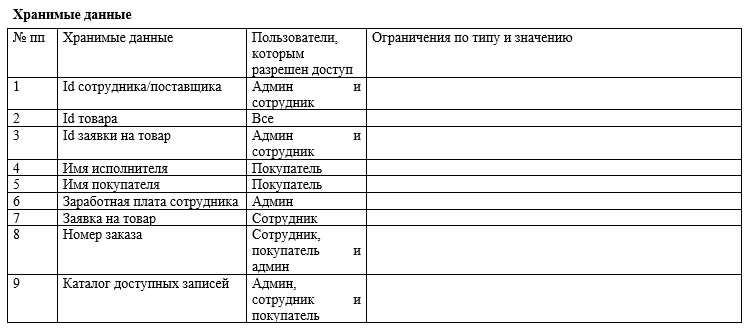
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| студенту группы | | 3530904\80105 | |  | Куксов Григорий Валерьевич | | | | | | | |
|  | | *(номер группы)* | |  | *(фамилия, имя, отчество)* | | | | | | | |
| ***1. Тема проекта (работы)*** | | | | *Разработка многопользовательской* | | | | | | | | |
| *автоматизированной системы управления организацией. Задание №10* | | | | | | | | | | | | |
| *Объект автоматизации – салон музыкальных записей* | | | | | | | | | | | | |
| ***2. Срок сдачи студентом законченного проекта (работы)*** | | | | | | | | | | | | *25.12.2020* |
| ***3. Исходные данные к проекту (работе)*** | | | | | | | | |  | | | |
| *Описание предметной области* | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| ***4.Содержание пояснительной записки***(перечень подлежащих разработке вопросов: введение, основная часть (раскрывается структура основной части), заключение, список использованных источников, приложения)*.* | | | | | | | | | | | | |
| *Введение. Анализ предметной области. Проектирование схемы данных.* | | | | | | | | | | | | |
| *Реализация базы данных в среде SQL Server. Разработка представлений и* | | | | | | | | | | | | |
| *хранимых процедур. Разработка клиентского приложения. Тестирование.* | | | | | | | | | | | | |
| *Заключение. Список использованных источников.* | | | | | | | | | | | | |
| Примерный объем пояснительной записки | | | | | | *15-20* | | | | страниц машинописного | | |
| текста | | | | | | | | | | | | |
| 5. Перечень графического материала ( с указанием обязательных чертежей и | | | | | | | | | | | | |
| плакатов) | *не предоставляется* | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| 6. Консультанты | | |  | | | | | | | | | |
|  | | | | | | | | | | | | |
| 7. Дата получения задания: «\_06\_» \_\_сентября\_\_ 2020 г. | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| Руководитель | | | | |  | | |  | | *Гасанова И.А.* | | |
|  | | | | | *(подпись)* | | |  | | *(инициалы, фамилия)* | | |
| Задание принял к исполнению | | | | |  | |  | | | *Куксов Г.В.* | | |
|  | | | | | *(подпись)* | |  | | | *(инициалы, фамилия)* | | |
|  | | | | | | | | | | | 6 сентября 2020 | |
|  | | | | | | | | | | | *(дата)* | |

# **Введение**

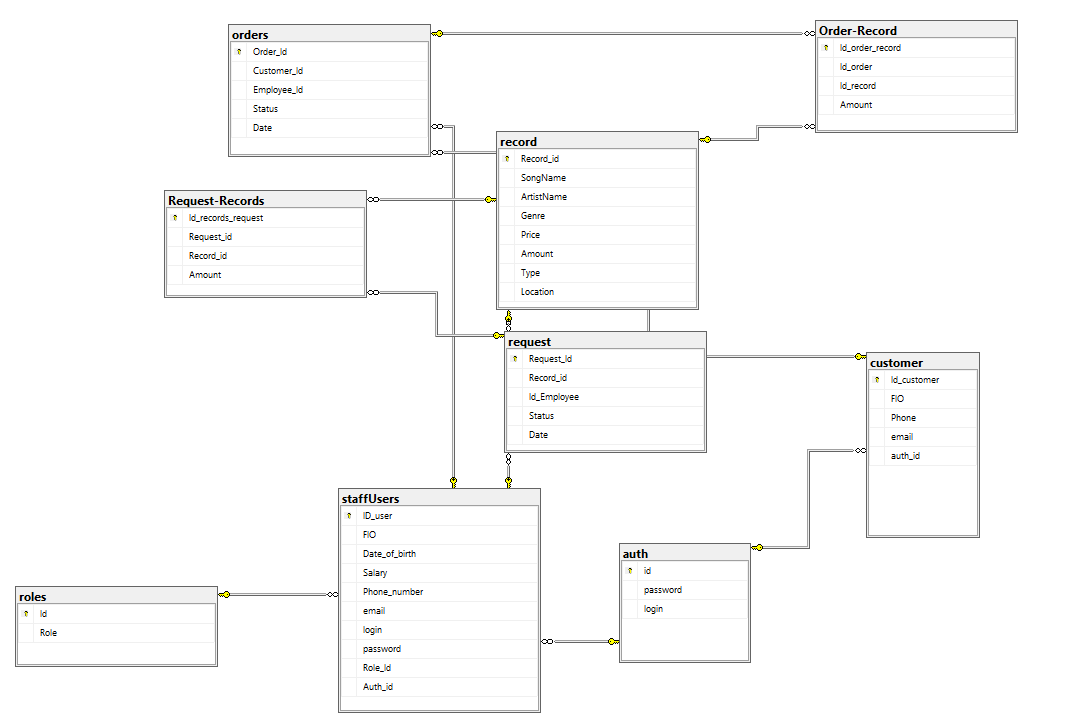
В ходе выполнения курсовой работы мной была создано клиентское приложение с базой данных. Данная «задача» является примером широкого класса задач программирования мобильных и десктопных версий ПО.

Для решения задач такого типа существуют множество средств для различных языков

программирования. В качестве среды я использовал Intellij IDEA. В качестве СУБД выбрана H2.

**** **** 

**Схема БД + описание таблиц + код создания таблиц.**



**Таблицы**

1. **Users** – пользователи

Id int primary key

FIO varchar(50)

Date\_of\_birth date

Salary float

Phone\_number varchar(15)

Email varchar(50)

Role\_id int foreign key

Auth\_id int foreign key

CREATE TABLE [dbo].[staffUsers](

[ID\_user] [int] NOT NULL,

[FIO] [varchar](50) NULL,

[Date\_of\_birth] [date] NULL,

[Salary] [float] NULL,

[Phone\_number] [varchar](15) NULL,

[email] [varchar](50) NULL,

[Role\_Id] [int] NULL,

[Auth\_id] [int] NULL,

CONSTRAINT [PK\_users] PRIMARY KEY CLUSTERED

(

[ID\_user] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[staffUsers] WITH CHECK ADD CONSTRAINT [FK\_staffUsers\_auth1] FOREIGN KEY([Auth\_id])

REFERENCES [dbo].[auth] ([id])

GO

ALTER TABLE [dbo].[staffUsers] CHECK CONSTRAINT [FK\_staffUsers\_auth1]

GO

ALTER TABLE [dbo].[staffUsers] WITH CHECK ADD CONSTRAINT [FK\_staffUsers\_roles] FOREIGN KEY([Role\_Id])

REFERENCES [dbo].[roles] ([Id])

GO

ALTER TABLE [dbo].[staffUsers] CHECK CONSTRAINT [FK\_staffUsers\_roles]

GO

1. **Roles** – роли в системе

Id int primary key

Role varchar(50)

CREATE TABLE [dbo].[roles](

[Id] [int] NOT NULL,

[Role] [varchar](50) NULL,

CONSTRAINT [PK\_USER] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

1. **Request**-**Records** – вспомогательная связь между заявкой и записями

Id int primary key

Request\_id int

Record\_id int

Amount int

CREATE TABLE [dbo].[Request-Records](

[Id\_records\_request] [int] NOT NULL,

[Request\_id] [int] NULL,

[Record\_id] [int] NULL,

[Amount] [int] NULL,

CONSTRAINT [PK\_Request-Records] PRIMARY KEY CLUSTERED

(

[Id\_records\_request] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Request-Records] WITH CHECK ADD CONSTRAINT [FK\_Request-Records\_record1] FOREIGN KEY([Record\_id])

REFERENCES [dbo].[record] ([Record\_id])

GO

ALTER TABLE [dbo].[Request-Records] CHECK CONSTRAINT [FK\_Request-Records\_record1]

GO

ALTER TABLE [dbo].[Request-Records] WITH CHECK ADD CONSTRAINT [FK\_Request-Records\_request1] FOREIGN KEY([Request\_id])

REFERENCES [dbo].[request] ([Request\_Id])

GO

ALTER TABLE [dbo].[Request-Records] CHECK CONSTRAINT [FK\_Request-Records\_request1]

GO

1. **Request** – заявка на товар

Id int primary key

Record\_id int foreign key

Id\_Employee int foreign key

Status bit

CREATE TABLE [dbo].[request](

[Request\_Id] [int] NOT NULL,

[Id\_Employee] [int] NULL,

[Status] [bit] NULL,

[Date] [date] NULL,

CONSTRAINT [PK\_request] PRIMARY KEY CLUSTERED

(

[Request\_Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[request] WITH CHECK ADD CONSTRAINT [FK\_request\_staffUsers] FOREIGN KEY([Id\_Employee])

REFERENCES [dbo].[staffUsers] ([ID\_user])

GO

ALTER TABLE [dbo].[request] CHECK CONSTRAINT [FK\_request\_staffUsers]

GO

1. **Record** – музыкальная запись

Record\_id int primary key

SongName varchar(50)

ArtistName varchar(50)

Genre varchar(50)

Price float

Amount int

Type varchar(50)

Location varchar(50)

CREATE TABLE [dbo].[record](

[Record\_id] [int] NOT NULL,

[SongName] [varchar](50) NULL,

[ArtistName] [varchar](50) NULL,

[Genre] [varchar](50) NULL,

[Price] [float] NULL,

[Amount] [int] NULL,

[Type] [varchar](50) NULL,

[Location] [varchar](50) NULL,

CONSTRAINT [PK\_record] PRIMARY KEY CLUSTERED

(

[Record\_id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

1. **Orders** – заказы

Order\_id int primary key

Customer\_id int foreign key

Employee\_id int foreign key

Status bit

CREATE TABLE [dbo].[orders](

[Order\_Id] [int] NOT NULL,

[Customer\_Id] [int] NULL,

[Employee\_Id] [int] NULL,

[Status] [bit] NULL,

CONSTRAINT [PK\_orders] PRIMARY KEY CLUSTERED

(

[Order\_Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[orders] WITH CHECK ADD CONSTRAINT [FK\_orders\_customer)1] FOREIGN KEY([Customer\_Id])

REFERENCES [dbo].[customer] ([Id\_customer])

GO

ALTER TABLE [dbo].[orders] CHECK CONSTRAINT [FK\_orders\_customer)1]

GO

ALTER TABLE [dbo].[orders] WITH CHECK ADD CONSTRAINT [FK\_orders\_staffUsers] FOREIGN KEY([Employee\_Id])

REFERENCES [dbo].[staffUsers] ([ID\_user])

GO

ALTER TABLE [dbo].[orders] CHECK CONSTRAINT [FK\_orders\_staffUsers]

GO

1. **Order**-**Record** – вспомогательная связь заказов и записей

Id\_order\_record int primary key

Id\_order int foreign key

Id\_record int foreign key

Amount int

CREATE TABLE [dbo].[Order-Record](

[Id\_order\_record] [int] NOT NULL,

[Id\_order] [int] NOT NULL,

[Id\_record] [int] NULL,

[Amount] [int] NULL,

CONSTRAINT [PK\_Order-Record] PRIMARY KEY CLUSTERED

(

[Id\_order\_record] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Order-Record] WITH CHECK ADD CONSTRAINT [FK\_Order-Record\_orders1] FOREIGN KEY([Id\_order])

REFERENCES [dbo].[orders] ([Order\_Id])

GO

ALTER TABLE [dbo].[Order-Record] CHECK CONSTRAINT [FK\_Order-Record\_orders1]

GO

ALTER TABLE [dbo].[Order-Record] WITH CHECK ADD CONSTRAINT [FK\_Order-Record\_record1] FOREIGN KEY([Id\_record])

REFERENCES [dbo].[record] ([Record\_id])

GO

ALTER TABLE [dbo].[Order-Record] CHECK CONSTRAINT [FK\_Order-Record\_record1]

GO

1. **Customer** – покупатель

Id int primary key

FIO varchar(50)

Phone varchar(15)

Email varchar(50)

Auth\_id int foreign key

CREATE TABLE [dbo].[customer](

[Id\_customer] [int] NOT NULL,

[FIO] [varchar](50) NULL,

[Phone] [varchar](15) NULL,

[email] [varchar](50) NULL,

[auth\_id] [int] NULL,

CONSTRAINT [PK\_customer)] PRIMARY KEY CLUSTERED

(

[Id\_customer] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[customer] WITH CHECK ADD CONSTRAINT [FK\_customer)\_auth1] FOREIGN KEY([auth\_id])

REFERENCES [dbo].[auth] ([id])

GO

ALTER TABLE [dbo].[customer] CHECK CONSTRAINT [FK\_customer)\_auth1]

GO

1. **Auth** – авторизация

Id int primary key

Login varchar(50)

Password varchar(50)

CREATE TABLE [dbo].[auth](

[id] [int] NOT NULL,

[password] [varchar](50) NULL,

[login] [varchar](50) NULL,

CONSTRAINT [PK\_auth] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

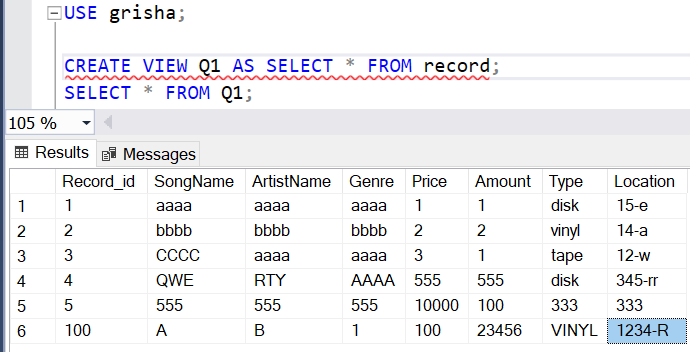
) ON [PRIMARY]

GO

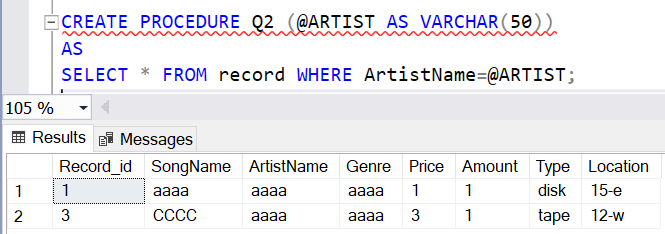
**Код запросов с описанием.**

ПРЕДСТАВЛЕНИЯ

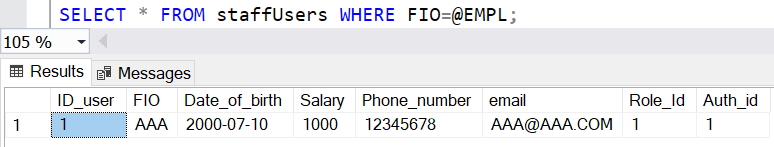
1. Просмотр каталога\заказов\заявок



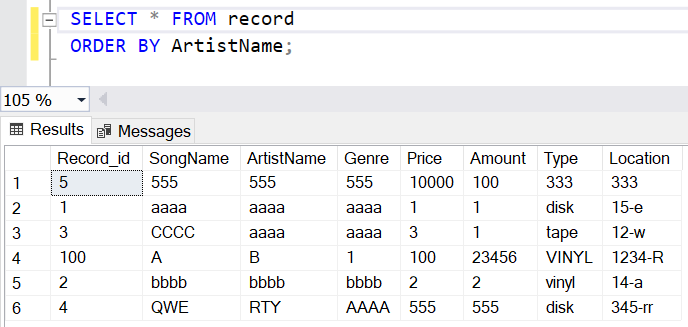
1. Просмотр музыкальных записей определенного исполнителя



1. Просмотр информации о сотруднике

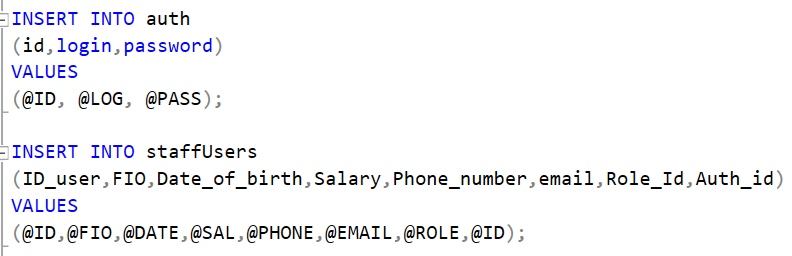


1. Фильтрация записей по имени исполнителя

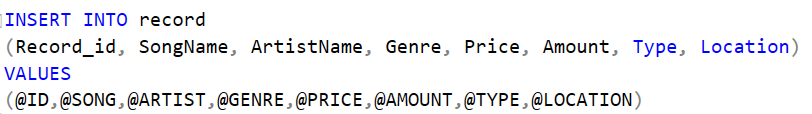


ДОБАВЛЕНИЕ

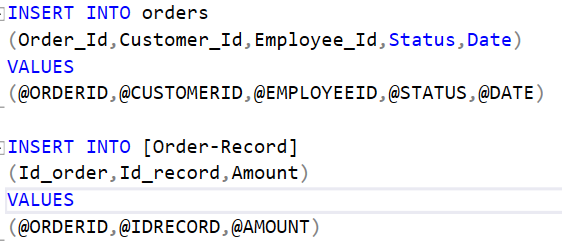
1. **Добавление учетной записи сотрудника**



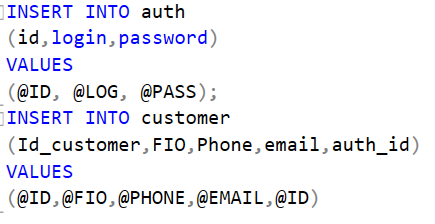
1. **Добавление товара в каталог**



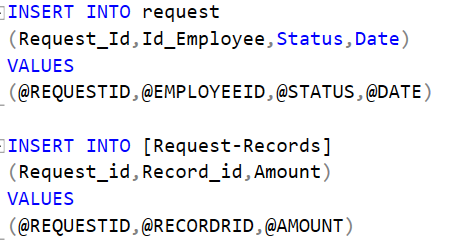
1. **Создание заказа**



1. **Добавление покупателя в систему**

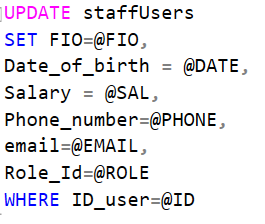


1. **Оформление заявки**

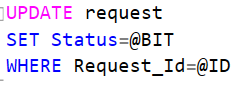


ИЗМЕНЕНИЕ

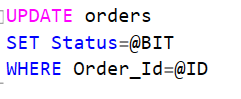
1. **Изменение учетной записи о сотруднике**

****

1. **Обработка заявки на товар**

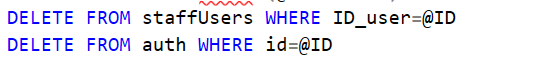
****

1. **Обработка заказа**



УДАЛЕНИЕ

1. **Удаление учетной записи сотрудника**



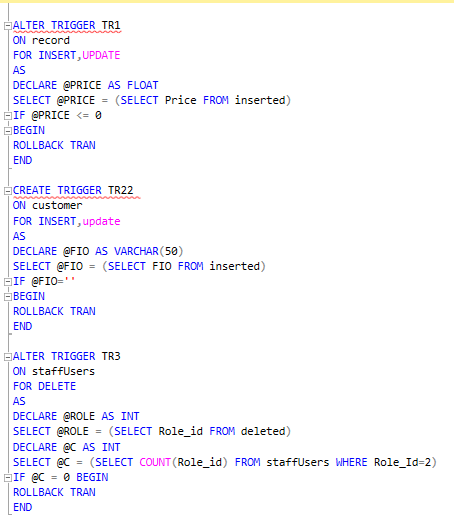
1. **Удаление товара из каталога**



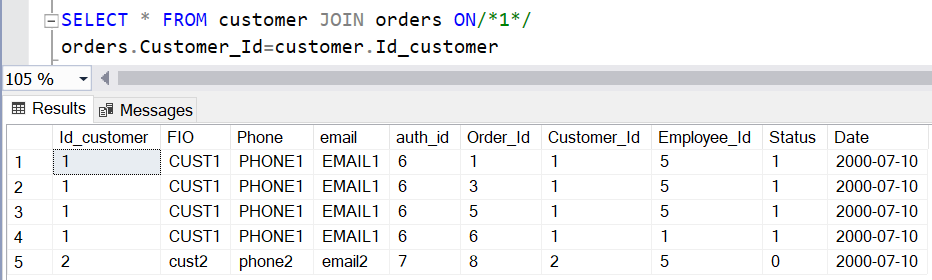
1. **Удаление покупателя из системы**



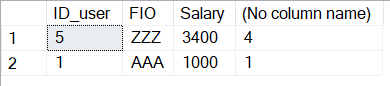
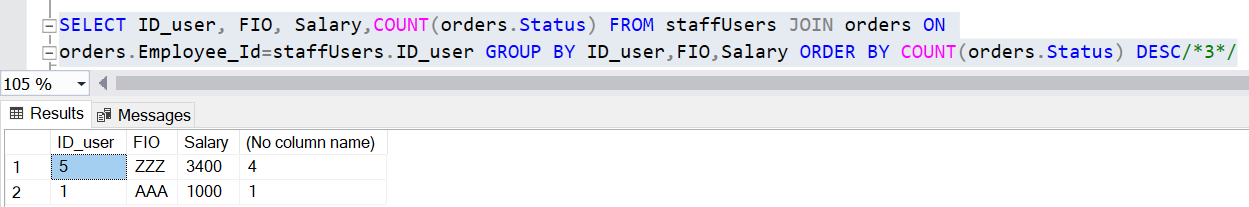
ТРИГГЕРЫ



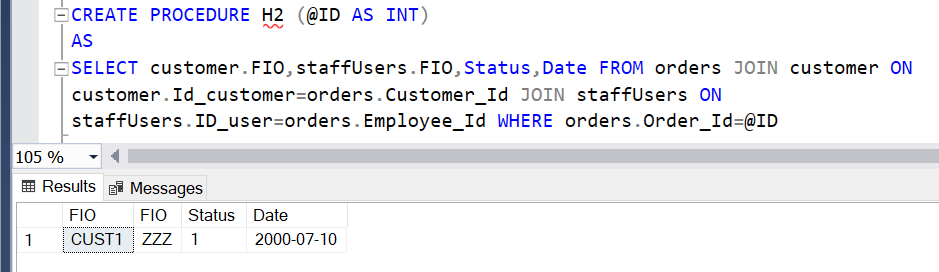
**Просмотр клиентов, сделавших заказы**



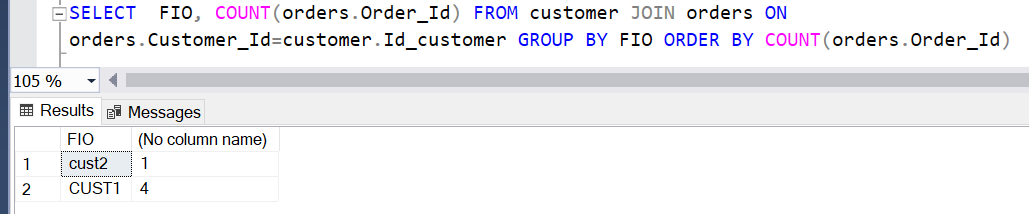
**Сортировка работников по количеству сделок и просмотр их зарплаты**



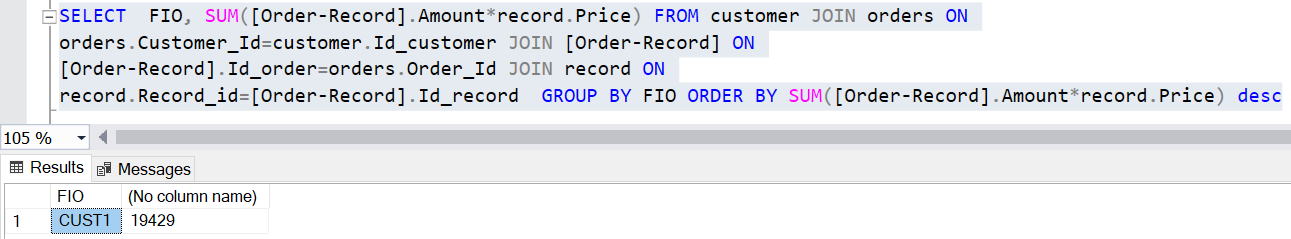
**Вывод содержимого отдельного заказа у клиента, по его айди**



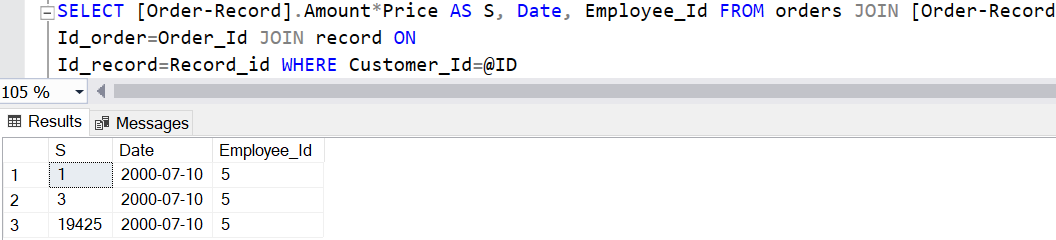
**Сортировка клиентов по количеству заказов**



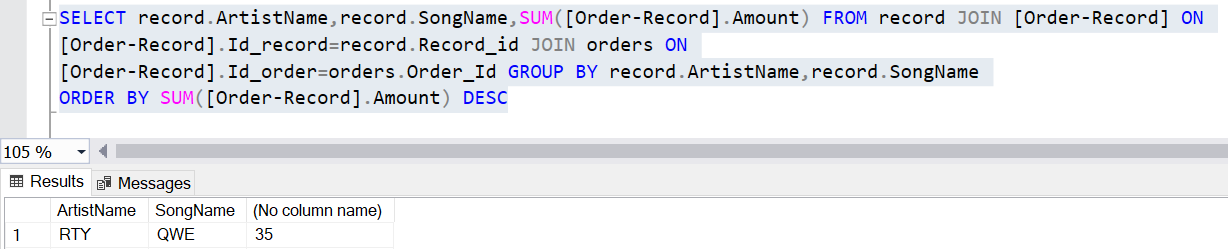
**Сортировка клиентов по сумме заказов**



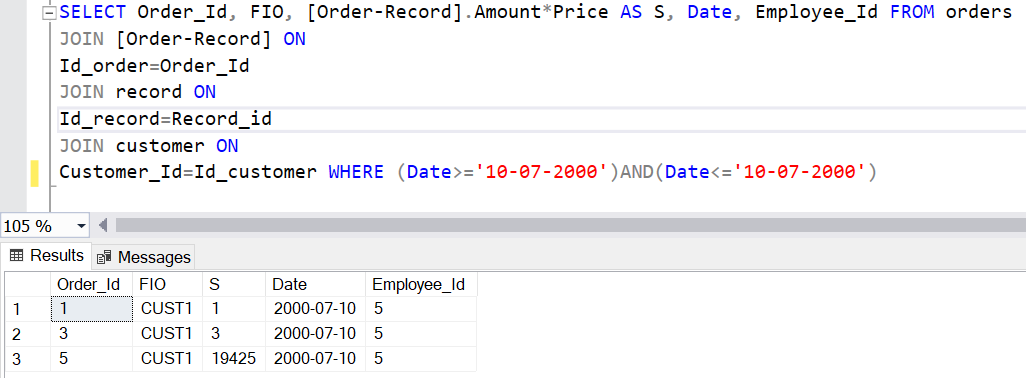
**Выбрать информацию о заказах (сумма, даты и статусы) для отдельного клиента**



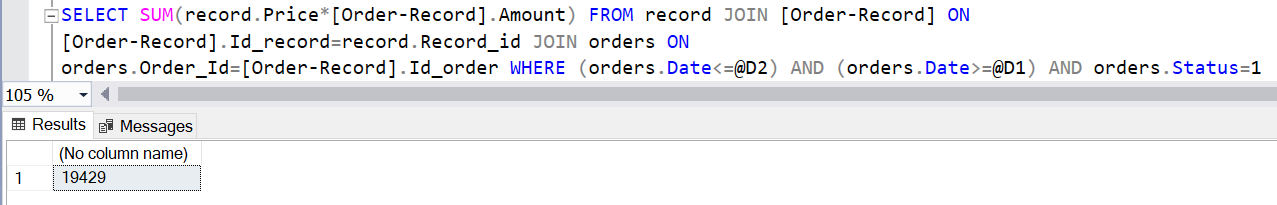
**Выбрать самые популярные муз. Записи**



**Выбрать заказы, сделанные в указанный период времени**



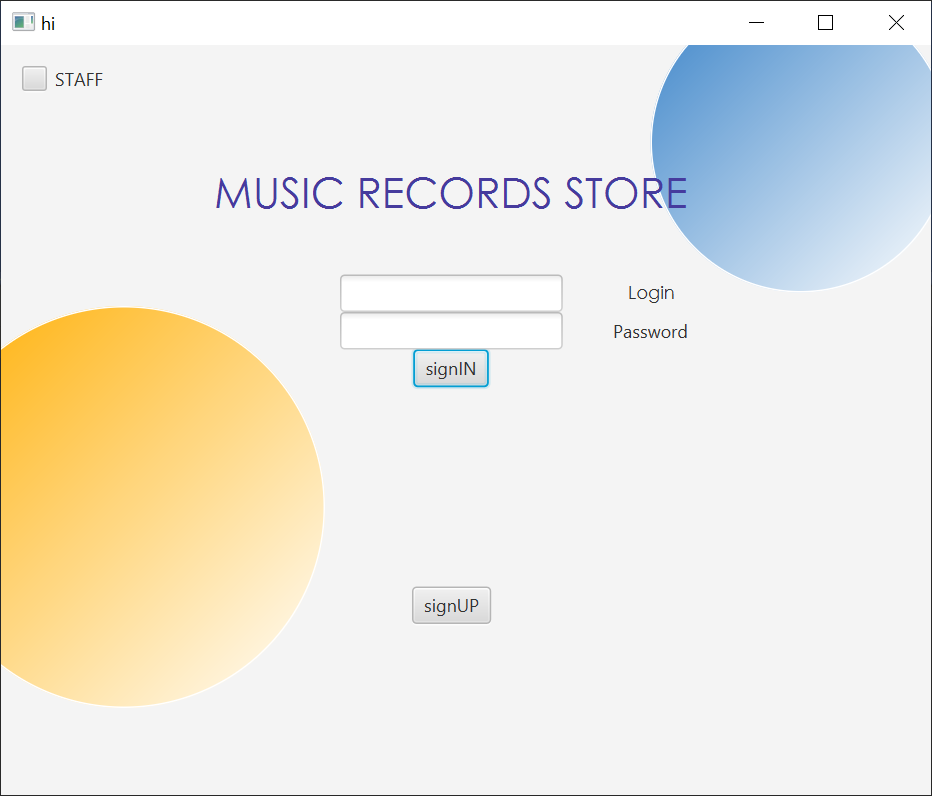
**Посмотреть сумму продаж закрытых заказов за определенный период**



**КЛИЕНТСКАЯ ЧАСТЬ**

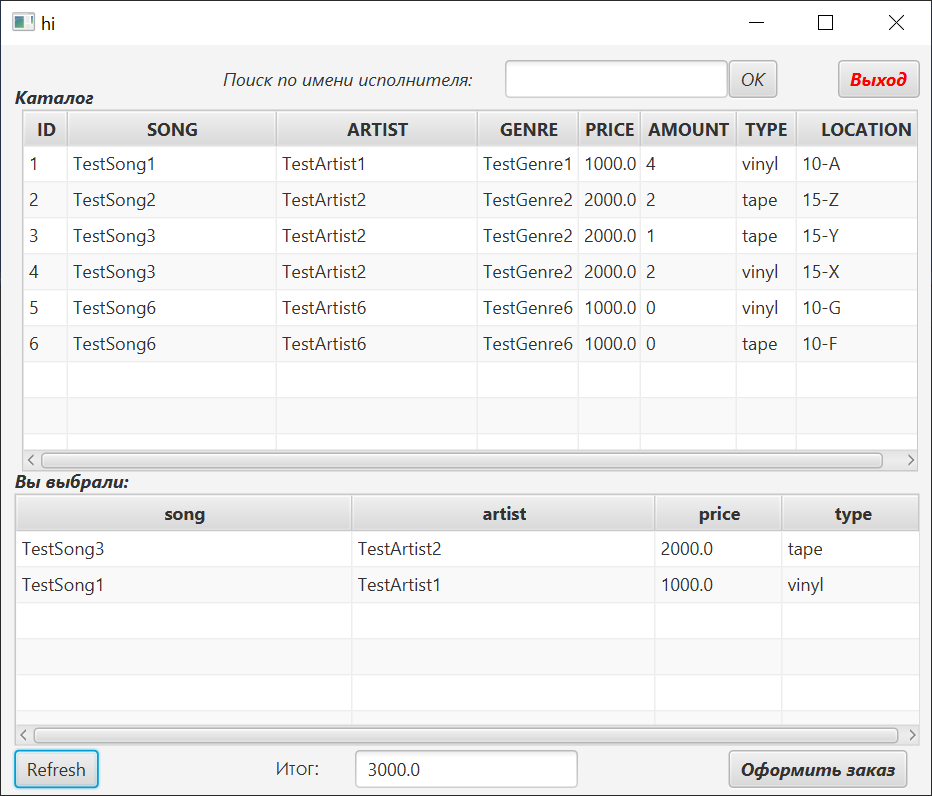
Авторизация

* Вход для пользователя\администратора\продавца
* Регистрация покупателя



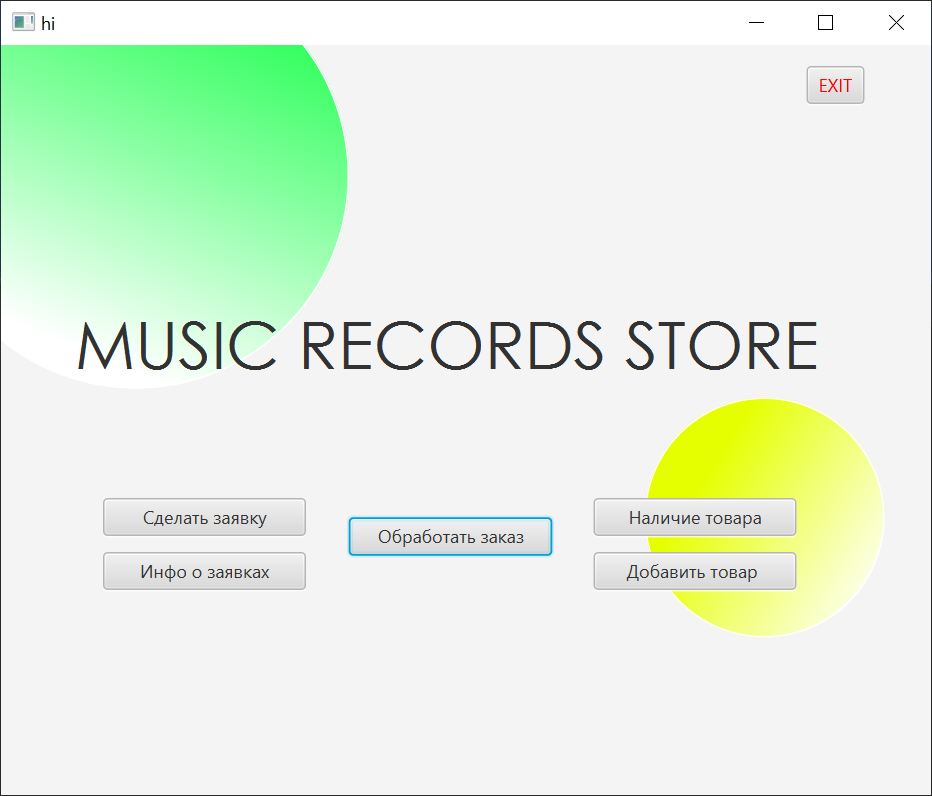
Покупатель

* Оформление заказа
* Просмотр корзины
* Поиск по имени исполнителя



Продавец

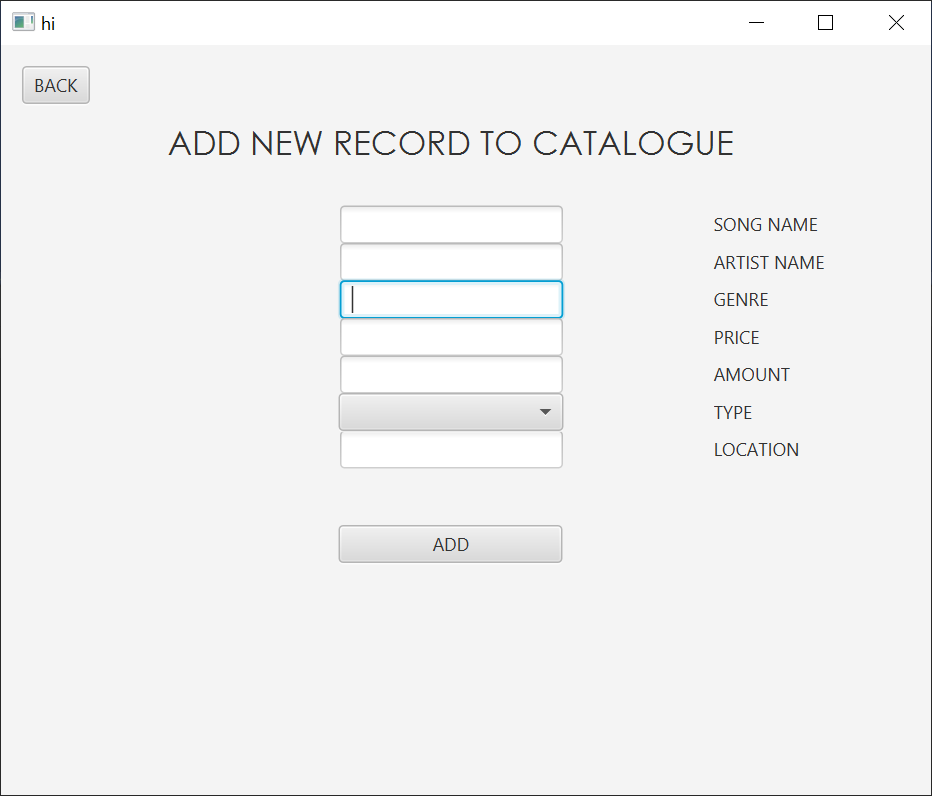
* Оформление заявки
* Просмотр заявок
* Обработка заказа
* Проверка наличия на складе
* Добавление товара в каталог

Изображение выглядит как стол

Автоматически созданное описаниеИзображение выглядит как стол

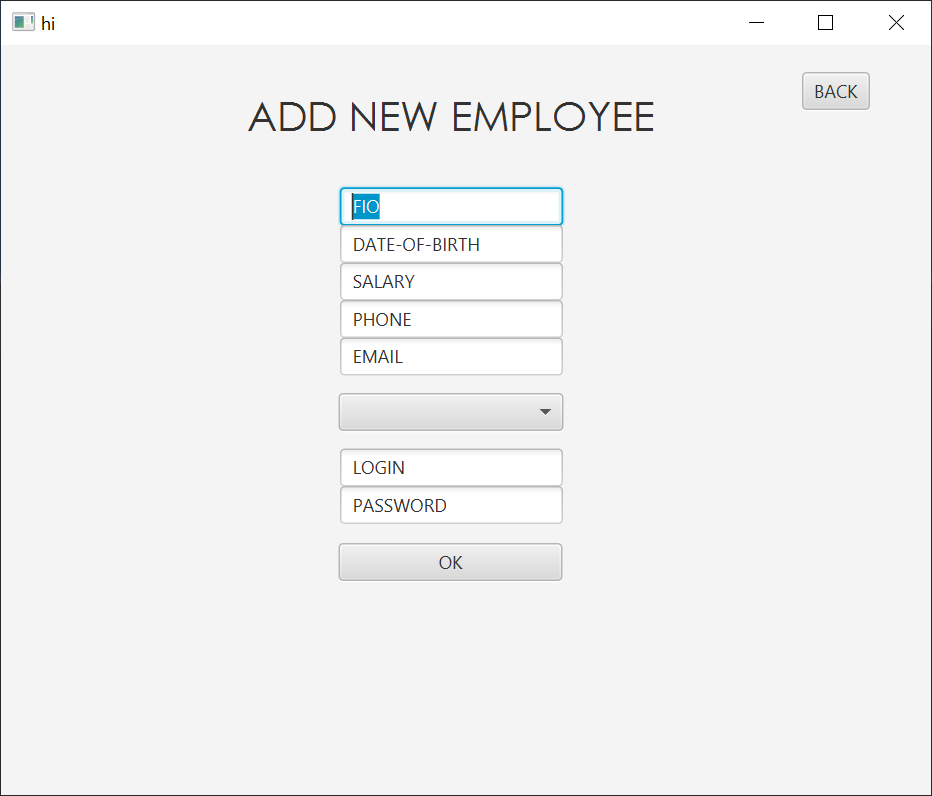
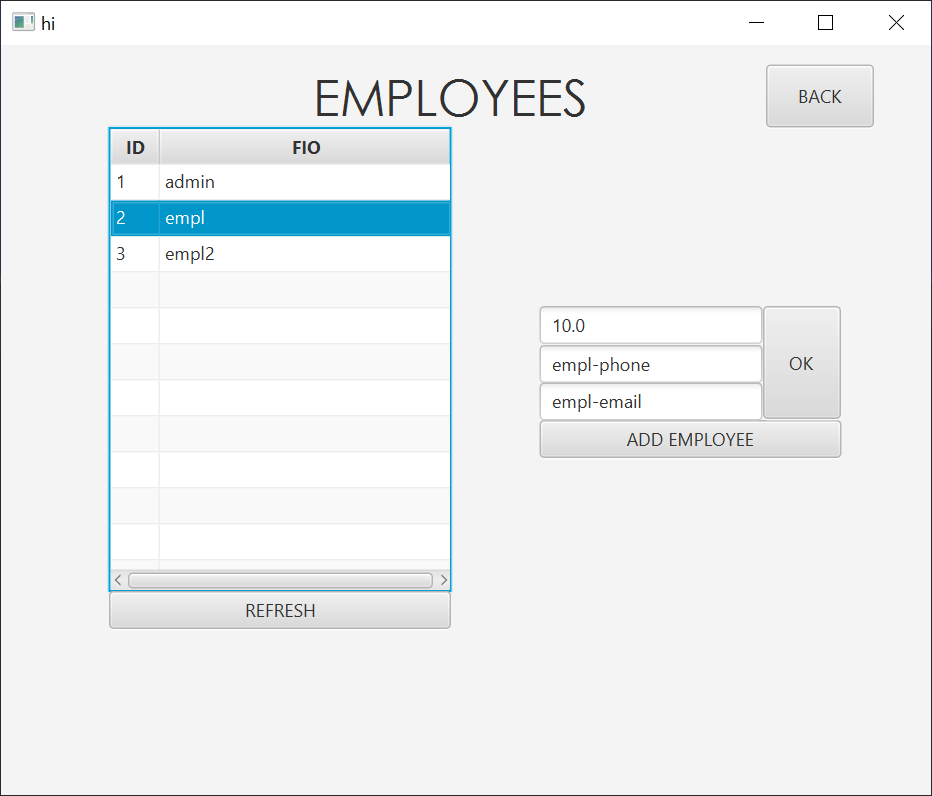
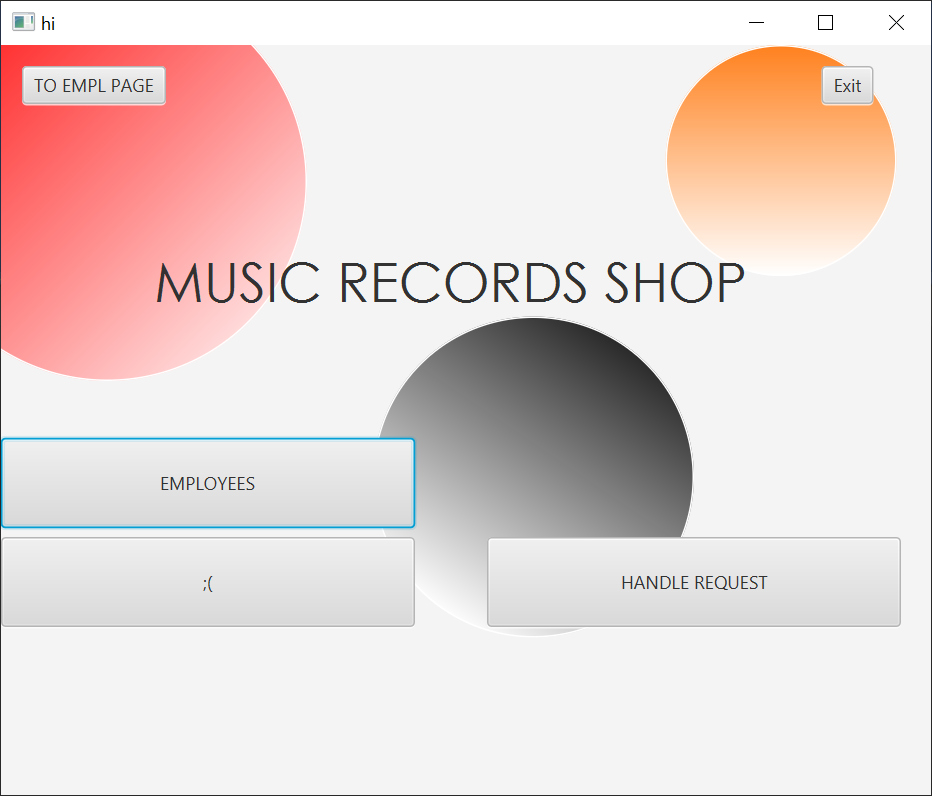
Автоматически созданное описаниеИзображение выглядит как стол

Автоматически созданное описаниеИзображение выглядит как стол

Автоматически созданное описание

Администратор

* Доступ к функциям продавца
* Просмотр\изменение информации о сотрудниках
* Регистрация нового сотрудника
* Обработка заявки

Изображение выглядит как стол

Автоматически созданное описание

**Вывод**

В результате работы было разработано клиент-ориентированное приложение на Java с базой данных H2

Были встречены трудности в виду выбранной базы данных (невозможность использовать DELETE вместе с JOIN) и загруженности языка

Из преимуществ можно выделить относительно интуитивно-понятный интерфейс, созданный с помощью JavaFX и SceneBuilder

**Список литературы**

1. Документация к JavaFx

<https://openjfx.io/javadoc/15/javafx.fxml/javafx/fxml/doc-files/introduction_to_fxml.html>

1. Документация к Intellij IDEA

<https://www.jetbrains.com/ru-ru/idea/resources/>

1. Документация к H2

https://www.h2database.com/html/tutorial.html

1. Рекомендации к оформлению