

Министерство образования и науки РФ
Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа программной инженерии

Отчёт по курсовой работе

по дисциплине «Базы данных»

Разработка многопользовательской автоматизированной системы управления организацией. Объект автоматизации – пивной завод.

Студент гр. 3530904/00105

Цапов Н. В.

Преподаватель

Гасанова И. А.

Санкт-Петербург

2022

Оглавление

ЗАДАНИЕ	3
Анализ предметной области	4
Описание таблиц и связей	10
Схема.....	10
Описание таблиц.....	11
Запросы.....	15
Представления.....	15
Запросы	17
Описание программы.....	27
Скриншоты системы	28
Авторизация и регистрация	28
Окно администратора ИС	29
Окно рабочего завода (Менеджер по продажам)	30
Окно менеджера по закупкам (розничные магазины).....	35
Заключение	37
Список литературы	38
Приложение (скрипт создания бд).....	39

ЗАДАНИЕ

НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА (КУРСОВОЙ РАБОТЫ)

студенту группы

3530904/00105

Цапов Никита Владимирович

(номер группы)

(фамилия, имя, отчество)

1. Тема проекта (работы)

Разработка многопользовательской

автоматизированной системы управления организацией.

Объект автоматизации - пивной завод. Задание № 19

2. Срок сдачи студентом законченного проекта (работы)

01.12.2022

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: «11» сентября 2022 г.

Руководитель

Гасанова И.А.

Задание принял к исполнению

(подпись)

(инициалы, фамилия)

Цапов Н. В.

(подпись)

(инициалы, фамилия)

11.09.2022

(дата)

Анализ предметной области

Группы пользователей разрабатываемой информационной системы (ИС)

№ гп	Наименование пользователя
1	Менеджер по продажам (завод)
2	Менеджер по закупкам (Розничные магазины)
3	Администратор ИС

Функции групп пользователей

№ гп	Выполняемая функция	Входные данные	Выходные данные	Функции, которые должны быть реализованы в ИС
Менеджер по продажам (завод)				
1	Просмотр базы товаров		<ul style="list-style-type: none"> • Наименование продукта(product_name) • Тип продукта(product_type) • UPS-код(upc_code) • Себестоимость продукта(prime_price) • Розничная цена(retail_price) • Розничная цена с учетом скидки(discount_price) • Количество товара на складе(amount) 	Просмотр всех записей и полей таблицы products , объединенной с discounts , stock
2	Добавление товара в каталог	<ul style="list-style-type: none"> • Наименование продукта(product_name) • Тип продукта(product_type) • UPC-код(upc_code) • Розничная цена(retail_price) • Скидка на товар(discount) • Количество товара на складе(amount) 	Таблица products	Добавление записи в таблицу products
3	Обновление информации о товаре	<ul style="list-style-type: none"> ○ Id продукта(product_id): • Изменяемые поля записи: • Наименование продукта(product_name) • Розничная цена(retail_price) 	Таблица products	Редактирование записи в таблице products
4	Изменение скидки на товар	<ul style="list-style-type: none"> ○ Id товара(product_id), • Скидка(discount) 	Таблицы discounts , products	Редактирование записи в таблице discounts Изменение вычисляемого значения discount_price в таблице products

5	Изменить количество товара на складе	<ul style="list-style-type: none"> ○ Id товара(product_id) • Количество(amount) 	Таблица stock	Редактирование записи в таблице stock
6	Просмотр информации о заказах	Фильтры: <ul style="list-style-type: none"> • Статус заказа(order_status) • Название организации(customer_name) • Сортировка по дате (Last week, Last month, Last year) 	<ul style="list-style-type: none"> • Номер заказа(order_id) • Название организации(customer_name) • Дата заказа(order_date) • Статус заказа(order_status) • Себестоимость заказа(total_prime_cost) • Стоимость заказа(total_cost) 	Просмотр записей таблиц orders, customers
7	Просмотр содержимого заказа		<ul style="list-style-type: none"> • Название товара(product_name) • Количество товара(amount) • Цена (price) • Стоимость заказа(cost) 	Просмотр записей таблиц orders, products, order_details, customers
8	Просмотр информации о заказчиках		<ul style="list-style-type: none"> • Название организации(customer_name) • Email(email) • Номер телефона(phone_number) • Адрес(address) 	Просмотр всех записей таблицы customers
9	Получение значения прибыли завода за все время		Вычисляемое значение	Вывод на экран прибыли завода за все время
10	Получение значения прибыли завода от конкретного заказчика	<ul style="list-style-type: none"> ○ Id заказчика(customer_id) 	Вычисляемое значение	Вывод на экран прибыли завода за все время от заказов конкретного заказчика
11	Получить информацию о недостающих товарах на складе для конкретного заказа	<ul style="list-style-type: none"> ○ Id заказа(order_id) 	Каждый товар: <ul style="list-style-type: none"> • Id товара(product_id) • Вычисляемое значение: order_details.amount – stock.amount 	Просмотр записей таблиц order_details по конкретному значению order_id и вычисление количества недостающих товаров на складе
12	Получить информацию о недостающих	<ul style="list-style-type: none"> ○ Id заказчика(customer_id) 	Каждый товар: <ul style="list-style-type: none"> • Id товара(product_id) 	Просмотр записей таблиц order_details по конкретному значению order_id для

	товарах для конкретного заказчика		<ul style="list-style-type: none"> Вычисляемое значение: <code>order_details.amount – stock.amount</code> 	каждого заказа заказчика(<code>customer_id</code>) и вычисление количества недостающих товаров на складе
13	Получить информацию о недостающих товарах для всех существующих заказов		Каждый товар: <ul style="list-style-type: none"> Id товара(<code>product_id</code>) Вычисляемое значение: <code>order_details.amount – stock.amount</code> 	Просмотр записей таблиц <code>order_details</code> по каждому значению <code>order_id</code> и вычисление количества недостающих товаров на складе
Менеджер по закупкам (Розничные магазины)				
1	Просмотр каталога товаров		<ul style="list-style-type: none"> Наименование продукта(<code>product_name</code>) Тип продукта(<code>product_type</code>) Розничная цена(<code>retail_price</code>) Розничная цена с учетом скидки(<code>discount_price</code>) 	Просмотр записей таблиц <code>products</code> , <code>discounts</code> (кроме <code>cost_price</code>)
2	Создание учетной записи заказчика	<ul style="list-style-type: none"> Название организации <code>customer_name</code> Email(<code>email</code>), Номер телефона(<code>phone_number</code>), Адрес(<code>address</code>) Пароль(<code>user_password</code>) 	Таблицы <code>customers</code> , <code>users</code>	Добавление записи в таблицы <code>customers</code> , <code>users</code>
3	Изменение данных об учетной записи заказчика	<ul style="list-style-type: none"> Id заказчика(<code>customer_id</code>): Номер телефона(<code>phone_number</code>) Адрес(<code>address</code>) 	Таблица <code>customers</code>	Редактирование записи в таблице <code>customers</code> ,
4	Формирование заказа	<ul style="list-style-type: none"> Id заказчика(<code>customer_id</code>): Id товара(<code>product_id</code>), Количество(<code>amount</code>) 	Таблицы <code>order_details</code> , <code>orders</code>	Добавление записей в таблицы <code>orders</code> , <code>order_details</code>
5	Отмена заказа	<ul style="list-style-type: none"> Id заказа(<code>order_id</code>) 	Таблица <code>orders</code>	Установить статус заказа(<code>order_state</code>) в “cancelled” в таблице <code>orders</code>

6	Просмотр своих заказов	<ul style="list-style-type: none"> ○ Id заказчика(customer_id) Фильтры: <ul style="list-style-type: none"> • Статус заказа(order_status) • Сортировка по дате (Last week, Last month, Last year) 	<ul style="list-style-type: none"> • Номер заказа(order_id) • Дата заказа(order_date) • Статус заказа(order_state) • Стоимость заказа(total_cost) 	Просмотр записей таблицы orders
7	Просмотр содержимого заказа	<ul style="list-style-type: none"> ○ Id заказа(order_id) 	<ul style="list-style-type: none"> • Название товара(product_name) • Количество товара(amount) • Цена (price) • Стоимость заказа(cost) 	Просмотр записей таблиц orders, order_details
Администратор ИС				
1	Создание учетной записи работника	<ul style="list-style-type: none"> • Адрес электронной почты(email) • Имя(first_name) • Фамилия(second_name) • Пароль(user_password) • Роль(user_role) 	Таблицы users, workers	Добавление записей в таблицы users, workers
2	Удаление учетной записи работника	<ul style="list-style-type: none"> ○ Id работника(worker_id) 	Таблицы users, workers	Удаление записей из таблиц users, workers
3	Просмотр учетных записей работников завода	Фильтр по роли (worker/admin)	<ul style="list-style-type: none"> • Ник в системе(username) • Адрес электронной почты(email) • Имя(first_name) • Фамилия(second_name) • Пароль(user_password) • Роль(user_role) 	Просмотр записей таблиц users, workers

Хранимые данные

Группы пользователей разрабатываемой информационной системы (ИС)

№ гр	Наименование пользователя	
1	Менеджер по продажам (завод)	МП
3	Менеджер по закупкам (Розничные магазины)	МЗ
4	Администратор ИС	АИС

Описание поля	Название поля	Пользователи, которым разрешен доступ на изменение	Ограничения по типу и значению
Пользователи(userss)			
ID пользователя	user_id		int primary key identity(1, 1) not null
Имя пользователя	username		Varchar(30), not null CHECK(username !="), UNIQUE
Пароль	user_password		varbinary(256), not null

			CONSTRAINT CK_USER_PASSWORD CHECK(user_password != "")
Роль	user_role		Varchar(10), not null
Рабочие завода(workers)			
ID пользователя	user_id		foreign key(user_id) references users (user_id)
ID рабочего	worker_id		int primary key identity(1, 1) not null
Электронная почта рабочего	email		Varchar(30), not null CONSTRAINT CK_WORKER_EMAIL CHECK(dbo.IsValidEmail(email)=1) CONSTRAINT UQ_WORKER_EMAIL UNIQUE
Имя рабочего	first_name		Varchar(30), not null CHECK(first_name != "")
Фамилия рабочего	second_name		Varchar(30), not null CHECK(second_name != "")
Заказчики(customers)			
ID пользователя	user_id		foreign key(user_id) references users (user_id)
ID заказчика	customer_id		int primary key identity(1, 1) not null
Имя заказчика	customer_name		Varchar(30), not null
Электронная почта заказчика	email		Varchar(30), not null CONSTRAINT CK_CUSTOMER_EMAIL CHECK(dbo.IsValidEmail(email) = 1) CONSTRAINT UQ_CUSTOMER_EMAIL UNIQUE
Номер телефона заказчика	phone_number	M3	Varchar(30), not null CONSTRAINT CK_CUSTOMER_PHONE_NUMBER CHECK(dbo.IsValidPhoneNumber(phone_number)=1) CONSTRAINT UQ_CUSTOMER_PHONE_NUMBER UNIQUE
Адрес	address	M3	Varchar(30), not null CHECK(address != "")
Заказы(orders)			
ID заказчика	customer_id		foreign key(customer_id) references customers (customer_id)
ID заказа	order_id		int primary key identity(1, 1) not null
Дата заказа	order_date		Datetime, not null
Статус заказа	order_status	МП, МЗ(отменить)	Varchar(30), not null default 'pending'
Итоговая себестоимость заказа	total_prime_cost		Decimal(8, 2)
Итоговая стоимость заказа	total_cost		Decimal(8, 2)
Состав заказа(order_details)			
ID заказа	order_id		foreign key(order_id) references orders (order_id)

ID продукта	product_id		foreign key(product_id) references products (product_id)
Количество продукта	amount		Int, not null
Цена товара	price		Decimal(8, 2)
Стоимость товаров в заказе	cost		as price * amount
Товары(products)			
ID товара	product_id		int primary key identity(1, 1) not null
Наименование продукта	product_name	МП	Varchar(30), not null UNIQUE
Тип товара	product_type	МП	Varchar(30), not null
UPC код	Upc_code	МП	Varchar(12) CONSTRAINT UQ_PRODUCT_UPC UNIQUE CONSTRAINT CK_PRODUCT_UPC CHECK(LEN(upc_code) = 12)
Себестоимость продукта	prime_price	МП	Decimal(8, 2), not null
Цена единицы продукта в розницу	retail_price		as prime_price * 1.38
Розничная цена с учетом скидки	discount_price		Decimal(8, 2), not null
Скидки(discounts)			
ID товара	product_id		foreign key(product_id) references products (product_id)
Действующая скидка на продукт	discount	МП	Decimal(3, 2), not null
Склад(stock)			
ID товара	product_id		foreign key(product_id) references products (product_id)
Количество продукта на складе	amount	МП	Int, not null

Описание таблиц и связей

Схема



Таблицы:


- users** – таблица содержит информацию о пользователях ИС (Менеджер по продажам (завод), Менеджер по закупкам (Розничные магазины)). С помощью поля user_role можно понять заказчик или сотрудник завода авторизирован.
- workers** – таблица содержит информацию о каждом работнике пивного завода (Менеджер по продажам (завод)).
- customers** – таблица содержит информацию о сторонних заказчиках (Менеджер по закупкам (Розничные магазины)).
- orders** – таблица содержит информацию о сформированных заказах от сторонних заказчиков.
- order_details** – таблица содержит информацию о составе каждого сформированного заказа из таблицы orders.
- products** – таблица содержит информацию о продукции завода.
- discounts** – таблица содержит информацию о скидках на каждый товар в формате ("0.15"=15%, "0.10"=15% и т.д).
- stock** - таблица склада, содержащая информацию о количестве каждого товара на складе

Связи таблицы:

Таблицы	Тип	Primary key	Foreign_key
users, workers	One-to-one	users.user_id	workers.user_id
users, cusotmers	One-to-one	users.user_id	customers.user_id
customers,orders	One-to-many	cutomers.customer_id	orders.customer_id
orders, order_details	One-to-many	orders.order_id	order_details.order_id
order_details, products	Many-to-many	products.product_id	order_details.product_id
stock, products	One-to-one	products.product_id	stock.product_id
discounts, products	One-to-one	products.product_id	discounts.product_id

Описание таблиц


Users

users			
	Column Name	Data Type	Allow Nulls
	user_id	int	<input type="checkbox"/>
	username	varchar(30)	<input type="checkbox"/>
	user_password	varbinary(256)	<input type="checkbox"/>
	user_role	varchar(10)	<input type="checkbox"/>

```
CREATE TABLE users (  
    [user_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,  
    [username] [varchar](30) NOT NULL CHECK(username != '') UNIQUE,  
    [user_password] varbinary(256) NOT NULL  
        CONSTRAINT CK_USER_PASSWORD CHECK(user_password != ''),  
    [user_role] [varchar](10) NOT NULL  
);
```

- user_id - ID пользователя
- username - Ник пользователя в системе
- user_password - Пароль
- user_role – роль


Workers

workers			
	Column Name	Data Type	Allow Nulls
	user_id	int	<input type="checkbox"/>
	worker_id	int	<input type="checkbox"/>
	email	varchar(30)	<input type="checkbox"/>
	first_name	varchar(30)	<input type="checkbox"/>
	second_name	varchar(30)	<input type="checkbox"/>

```
CREATE TABLE workers (  
    [user_id] [int] NOT NULL FOREIGN KEY([user_id]) REFERENCES [dbo].[users]  
    ([user_id]) on delete cascade,  
    [worker_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,  
    [email] [varchar](30) NOT NULL  
        CONSTRAINT CK_WORKER_EMAIL CHECK(dbo.isValidEmail(email) = 1)  
        CONSTRAINT UQ_WORKER_EMAIL UNIQUE,  
    [first_name] [varchar](30) NOT NULL CHECK(first_name != ''),  
    [second_name] [varchar](30) NOT NULL CHECK(second_name != '')  
);
```

- user_id - ID пользователя
- worker_id - ID рабочего
- email - Электронная почта рабочего
- first_name - Имя рабочего
- second_name - Фамилия рабочего


Customers

customers			
	Column Name	Data Type	Allow Nulls
	user_id	int	<input type="checkbox"/>
	customer_id	int	<input type="checkbox"/>
	email	varchar(30)	<input type="checkbox"/>
	customer_name	varchar(30)	<input type="checkbox"/>
	phone_number	varchar(30)	<input type="checkbox"/>
	address	varchar(30)	<input type="checkbox"/>

```
CREATE TABLE customers (
    [user_id] [int] NOT NULL FOREIGN KEY([user_id]) REFERENCES [dbo].[users]
([user_id]),
    [customer_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,
    [email] [varchar](30) NOT NULL
        CONSTRAINT CK_CUSTOMER_EMAIL CHECK(dbo.isValidEmail(email) = 1)
        CONSTRAINT UQ_CUSTOMER_EMAIL UNIQUE,
    [customer_name] [varchar](30) NOT NULL CHECK(customer_name != ''),
    [phone_number] [varchar](30) NOT NULL
        CONSTRAINT CK_CUSTOMER_PHONE_NUMBER
CHECK(dbo.isValidPhoneNumber(phone_number) = 1)
        CONSTRAINT UQ_CUSTOMER_PHONE_NUMBER UNIQUE,
    [address] [varchar](30) NOT NULL CHECK(address != '')
);
```

- user_id - ID пользователя
- customer_id - ID заказчика
- customer_name – Наименование заказчика
- email - Электронная почта заказчика
- phone_number - Номер телефона заказчика
- address – Адрес

Orders

orders			
	Column Name	Data Type	Allow Nulls
	order_id	int	<input type="checkbox"/>
	customer_id	int	<input type="checkbox"/>
	order_date	datetime	<input type="checkbox"/>
	order_status	varchar(30)	<input checked="" type="checkbox"/>
	total_prime_cost	decimal(8, 2)	<input checked="" type="checkbox"/>
	total_cost	decimal(8, 2)	<input checked="" type="checkbox"/>

```
CREATE TABLE orders (
    [order_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,
    [customer_id] [int] NOT NULL FOREIGN KEY([customer_id]) REFERENCES
[dbo].[customers] ([customer_id]),
    [order_date] [datetime] NOT NULL,
    [order_status] [varchar](30) DEFAULT 'pending',
```

```

        [total_prime_cost] [decimal](8, 2) NULL,
        [total_cost] [decimal](8, 2) NULL
    );

```

- order_id - ID заказа
- customer_id - ID заказчика
- order_date - Дата заказа
- order_status - Статус заказа
- total_prime_cost - Итоговая себестоимость заказа(вычисляемое по триггеру значение)
- total_cost - Итоговая стоимость заказа(вычисляемое по триггеру значение)

Order_details

order_details

Column Name	Data Type	Allow Nulls
order_id	int	<input type="checkbox"/>
product_id	int	<input type="checkbox"/>
amount	int	<input type="checkbox"/>
price	decimal(8, 2)	<input checked="" type="checkbox"/>
cost		<input checked="" type="checkbox"/>

```


CREATE TABLE order_details (
    [order_id] [int] NOT NULL FOREIGN KEY([order_id]) REFERENCES [dbo].[orders]
([order_id]),
    [product_id] [int] NOT NULL FOREIGN KEY([product_id]) REFERENCES
[dbo].[products] ([product_id]),
    [amount] [int] NOT NULL,
    [price] [decimal](8, 2) NULL,
    [cost] as price * amount
);

```

- order_id – ID заказа
- product_id - ID продукта
- amount - Количество продукта
- price – Цена товара
- cost – Стоимость товаров

Products

products

Column Name	Data Type	Allow Nulls
 product_id	int	<input type="checkbox"/>
product_name	varchar(30)	<input type="checkbox"/>
product_type	varchar(30)	<input type="checkbox"/>
upc_code	varchar(12)	<input type="checkbox"/>
prime_price	decimal(8, 2)	<input type="checkbox"/>
retail_price		<input checked="" type="checkbox"/>
discount_price	decimal(8, 2)	<input checked="" type="checkbox"/>

```

CREATE TABLE products (
    [product_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,

```

```

[product_name] [varchar](30) NOT NULL CHECK(product_name != '')
    CONSTRAINT UQ_PRODUCT_NAME UNIQUE,
[product_type] [varchar](30) NOT NULL,
[upc_code] [varchar](12) NOT NULL
    CONSTRAINT UQ_PRODUCT_UPC UNIQUE(upc_code)
    CONSTRAINT CK_PRODUCT_UPC CHECK(LEN(upc_code) = 12),
[prime_price] [decimal](8, 2) NOT NULL,
[retail_price] as prime_price * 1.38,
[discount_price] [decimal](8, 2)
);

```

- product_id - ID товара
- product_name - Наименование товара
- product_type - Тип товара
- upc_code – UPC код товара
- prime_price - Себестоимость товара
- retail_price - Цена единицы товара в розницу
- discount_price - Розничная цена со скидкой

Discounts

discounts			
	Column Name	Data Type	Allow Nulls
	product_id	int	<input type="checkbox"/>
	discount	decimal(3, 2)	<input type="checkbox"/>

```

CREATE TABLE discounts (
    [product_id] [int] NOT NULL FOREIGN KEY([product_id]) REFERENCES
[dbo].[products] ([product_id]),
    [discount] [decimal](3, 2) NOT NULL
);

```

- product_id - ID товара
- discount - Действующая скидка на продукт

Stock

stock			
	Column Name	Data Type	Allow Nulls
	product_id	int	<input type="checkbox"/>
	amount	int	<input type="checkbox"/>

```

CREATE TABLE stock (
    [product_id] [int] NOT NULL FOREIGN KEY([product_id]) REFERENCES
[dbo].[products] ([product_id]),
    [amount] [int] NOT NULL
);

```

- product_id - ID товара
- amount - Количество товара на складе

Запросы

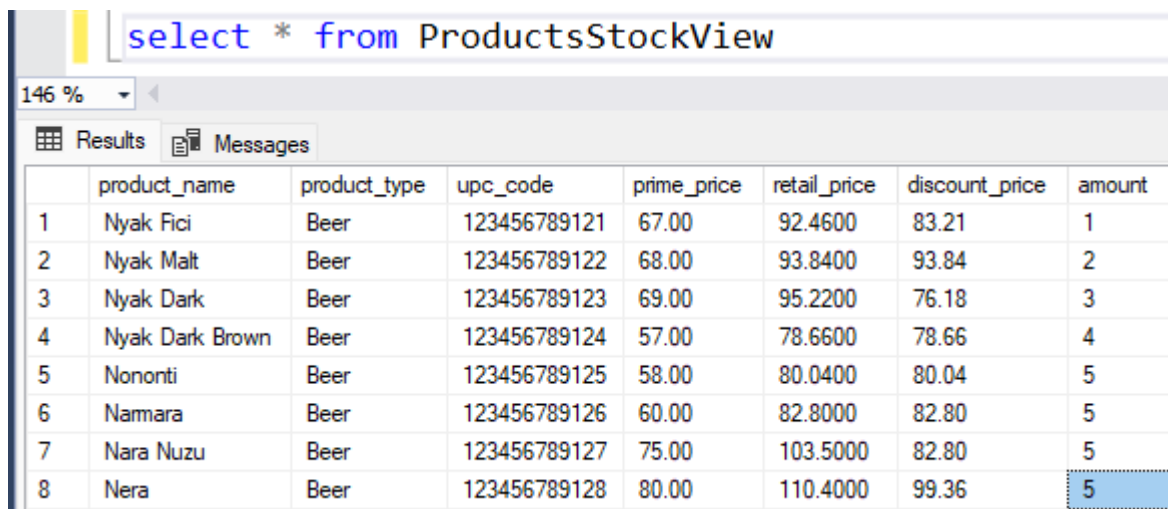
Все представления и запросы хранятся в процедурах

Представления:

1) Вывод каталога товаров, его количества на складе, и цены со скидкой (**сложный**: таблицы products, stock)

```
create view ProductsStockView as
select product_name,
       product_type,
       upc_code,
       prime_price,
       retail_price,
       discount_price,
       amount
from products
join stock on products.product_id=stock.product_id
```

GO



	product_name	product_type	upc_code	prime_price	retail_price	discount_price	amount
1	Nyak Fici	Beer	123456789121	67.00	92.4600	83.21	1
2	Nyak Malt	Beer	123456789122	68.00	93.8400	93.84	2
3	Nyak Dark	Beer	123456789123	69.00	95.2200	76.18	3
4	Nyak Dark Brown	Beer	123456789124	57.00	78.6600	78.66	4
5	Nononti	Beer	123456789125	58.00	80.0400	80.04	5
6	Namara	Beer	123456789126	60.00	82.8000	82.80	5
7	Nara Nuzu	Beer	123456789127	75.00	103.5000	82.80	5
8	Nera	Beer	123456789128	80.00	110.4000	99.36	5

2) Вывод всех заказов и их содержимого (**сложный**: таблицы orders, order_details, customers)

```
create view OrdersOrderDetailsCustomersView as
select orders.order_id,
       customer_name,
       order_date,
       orders.order_status,
       products.product_id,
       product_name,
       amount,
       price,
       cost
from orders
join order_details on orders.order_id=order_details.order_id
join products on order_details.product_id=products.product_id
```

join customers on orders.customer_id=customers.customer_id

go

select * from OrdersOrderDetailsCustomersView									
146 %									
Results Messages									
	order_id	customer_name	order_date	order_status	product_id	product_name	amount	price	cost
1	1	Красное&Белое	2022-05-23 14:25:10.000	pending	1	Nyak Fici	1	83.21	83.21
2	1	Красное&Белое	2022-05-23 14:25:10.000	pending	2	Nyak Malt	2	93.84	187.68
3	1	Красное&Белое	2022-05-23 14:25:10.000	pending	3	Nyak Dark	4	76.18	304.72
4	2	Красное&Белое	2022-05-24 14:25:10.000	processing	2	Nyak Malt	2	93.84	187.68
5	2	Красное&Белое	2022-05-24 14:25:10.000	processing	3	Nyak Dark	3	76.18	228.54
6	2	Красное&Белое	2022-05-24 14:25:10.000	processing	4	Nyak Dark Brown	4	78.66	314.64
7	3	РосАл	2022-05-25 14:25:10.000	pending	1	Nyak Fici	5	83.21	416.05
8	3	РосАл	2022-05-25 14:25:10.000	pending	3	Nyak Dark	6	76.18	457.08
9	3	РосАл	2022-05-25 14:25:10.000	pending	5	Nononti	7	80.04	560.28
10	4	РосАл	2022-05-26 14:25:10.000	completed	2	Nyak Malt	2	93.84	187.68
11	4	РосАл	2022-05-26 14:25:10.000	completed	3	Nyak Dark	4	76.18	304.72
12	4	РосАл	2022-05-26 14:25:10.000	completed	4	Nyak Dark Brown	8	78.66	629.28
13	5	РосАл	2022-05-27 14:25:10.000	cancelled	1	Nyak Fici	6	83.21	499.26
14	5	РосАл	2022-05-27 14:25:10.000	cancelled	3	Nyak Dark	24	76.18	1828.32
15	5	РосАл	2022-05-27 14:25:10.000	cancelled	4	Nyak Dark Brown	10	78.66	786.60
16	6	Beerka	2022-05-30 14:25:10.000	completed	1	Nyak Fici	4	83.21	332.84
17	6	Beerka	2022-05-30 14:25:10.000	completed	5	Nononti	2	80.04	160.08
18	6	Beerka	2022-05-30 14:25:10.000	completed	4	Nyak Dark Brown	10	78.66	786.60
19	7	Beerka	2022-07-27 14:25:10.000	completed	7	Nara Nuzu	6	82.80	496.80
20	7	Beerka	2022-07-27 14:25:10.000	completed	3	Nyak Dark	30	76.18	2285.40
21	7	Beerka	2022-07-27 14:25:10.000	completed	6	Namara	15	82.80	1242.00

Запросы

Вывод

1. Вывод базы товаров для сотрудника завода с количеством товара на складе и себестоимости продукта(сложный)

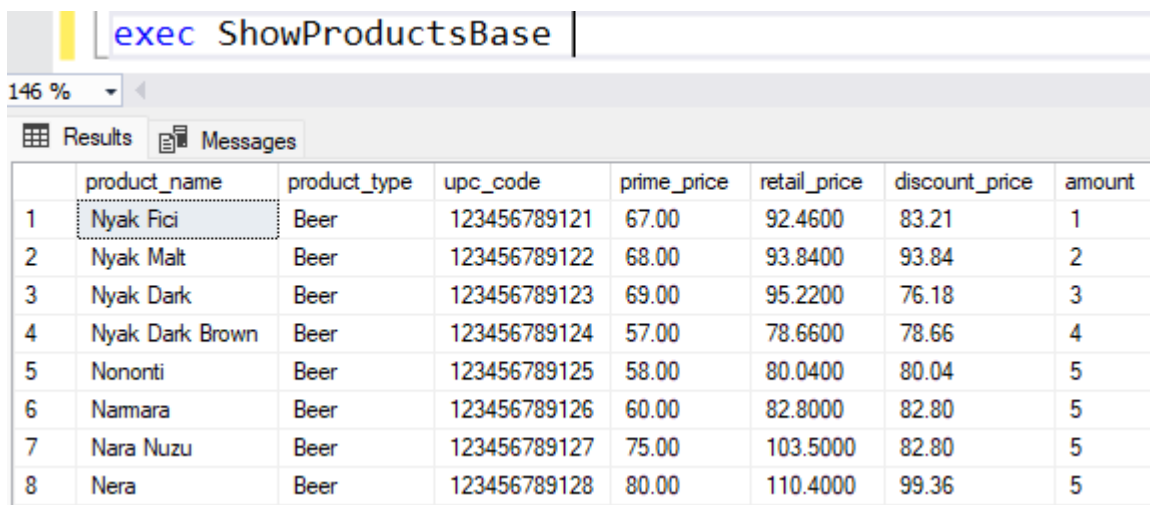
```
create or alter procedure ShowProductsBase
```

```
as
```

```
select product_name,  
       product_type,  
       upc_code,  
       prime_price,  
       retail_price,  
       discount_price,  
       amount
```

```
from ProductsStockView
```

```
go
```



	product_name	product_type	upc_code	prime_price	retail_price	discount_price	amount
1	Nyak Fici	Beer	123456789121	67.00	92.4600	83.21	1
2	Nyak Malt	Beer	123456789122	68.00	93.8400	93.84	2
3	Nyak Dark	Beer	123456789123	69.00	95.2200	76.18	3
4	Nyak Dark Brown	Beer	123456789124	57.00	78.6600	78.66	4
5	Nononti	Beer	123456789125	58.00	80.0400	80.04	5
6	Namara	Beer	123456789126	60.00	82.8000	82.80	5
7	Nara Nuzu	Beer	123456789127	75.00	103.5000	82.80	5
8	Nera	Beer	123456789128	80.00	110.4000	99.36	5

2. Вывод каталога товаров для заказчика(сложный)

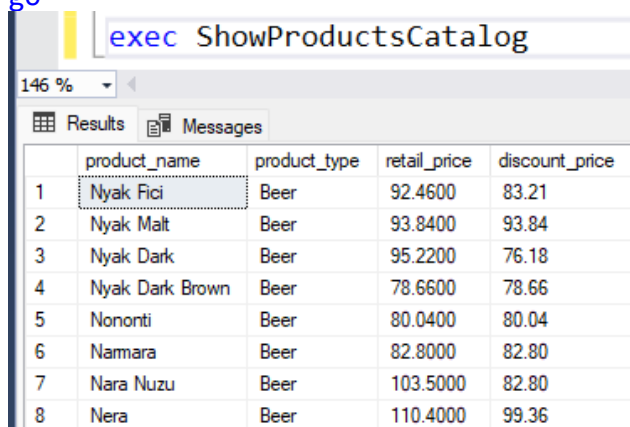
```
create or alter procedure ShowProductsCatalog
```

```
as
```

```
select product_name,  
       product_type,  
       retail_price,  
       discount_price
```

```
from ProductsStockView
```

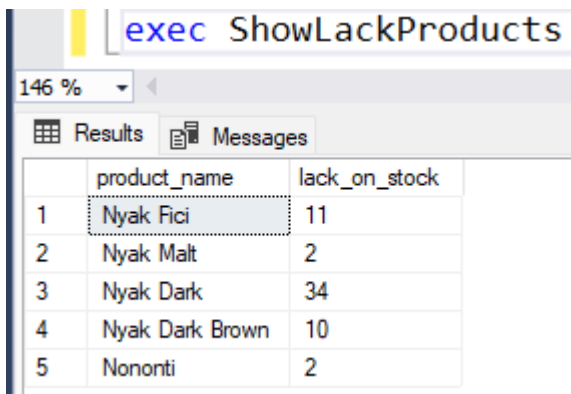
```
go
```



	product_name	product_type	retail_price	discount_price
1	Nyak Fici	Beer	92.4600	83.21
2	Nyak Malt	Beer	93.8400	93.84
3	Nyak Dark	Beer	95.2200	76.18
4	Nyak Dark Brown	Beer	78.6600	78.66
5	Nononti	Beer	80.0400	80.04
6	Namara	Beer	82.8000	82.80
7	Nara Nuzu	Beer	103.5000	82.80
8	Nera	Beer	110.4000	99.36

3. Получить информацию о недостающих товарах на складе для существующих заказов(сложный)

```
create or alter procedure ShowLackProducts
as
select product_name,
       abs(amount_on_stock - ordered) as lack_on_stock
from (select product_name,
            sum(OrdersOrderDetailsCustomersView.amount) as ordered,
            stock.amount as amount_on_stock
      from OrdersOrderDetailsCustomersView
      join stock
        on OrdersOrderDetailsCustomersView.product_id=stock.product_id
     where order_status != 'completed'
     group by product_name, stock.amount)
as a
where (amount_on_stock - ordered) < 0
go
```

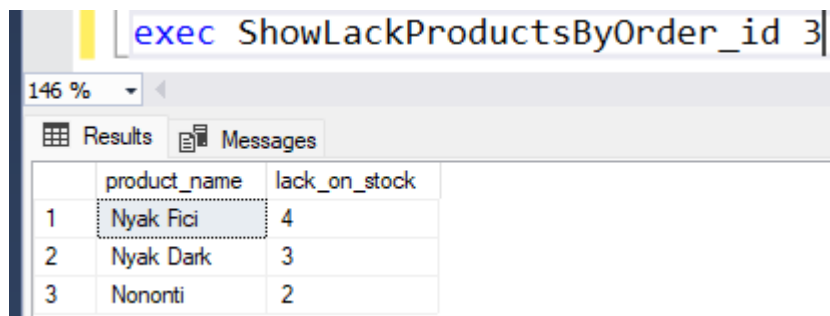


	product_name	lack_on_stock
1	Nyak Fici	11
2	Nyak Malt	2
3	Nyak Dark	34
4	Nyak Dark Brown	10
5	Nononti	2

4. Вывод информации о недостающих товарах на складе для существующих заказов для конкретного заказа (сложный)

```
create or alter procedure ShowLackProductsByOrder_id
@order_id int
as
select product_name,
       abs(amount_on_stock - ordered) as lack_on_stock
from
  (select product_name,
          sum(OrdersOrderDetailsCustomersView.amount) as ordered,
          stock.amount as amount_on_stock
    from OrdersOrderDetailsCustomersView
    join stock on OrdersOrderDetailsCustomersView.product_id=stock.product_id
   where order_status != 'completed' and order_id = @order_id
   group by product_name, stock.amount)
as a
where (amount_on_stock - ordered) < 0
```

go



	product_name	lack_on_stock
1	Nyak Fici	4
2	Nyak Dark	3
3	Nononti	2

5. Вывод информации о недостающих товарах на складе для существующих заказов для конкретного заказчика (сложный)

```
create or alter procedure ShowLackProductsByCustomer_name
```

```
@customer_name varchar(30)
```

```
as
```

```
select product_name,  
       abs(amount_on_stock - ordered) as lack_on_stock
```

```
from
```

```
(select product_name,  
       sum(OrdersOrderDetailsCustomersView.amount) as ordered,  
       stock.amount as amount_on_stock
```

```
from OrdersOrderDetailsCustomersView join stock on
```

```
OrdersOrderDetailsCustomersView.product_id=stock.product_id
```

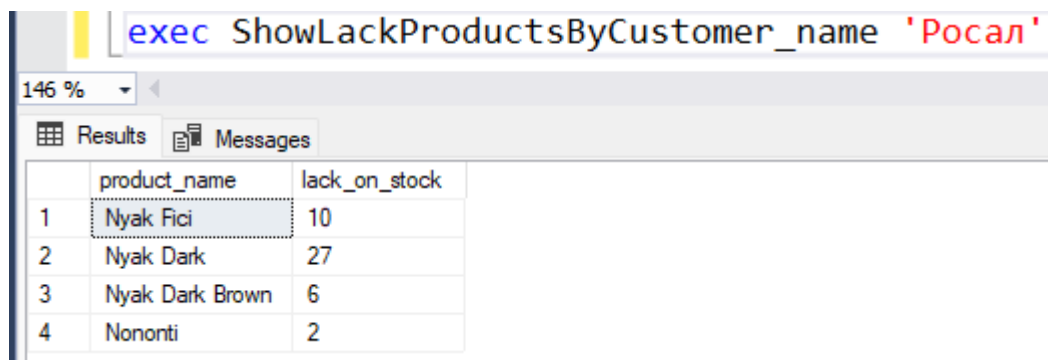
```
where order_status != 'completed' and customer_name = @customer_name
```

```
group by product_name, stock.amount)
```

```
as a
```

```
where (amount_on_stock - ordered) < 0
```

```
go
```



	product_name	lack_on_stock
1	Nyak Fici	10
2	Nyak Dark	27
3	Nyak Dark Brown	6
4	Nononti	2

6. Вывод информации обо всех заказах (сложный)

```
create or alter procedure ShowAllOrders
```

```
as
```

```
select order_id,  
       customer_name,  
       order_date,  
       order_status,  
       total_prime_cost,  
       total_cost
```

```
from Orders
```

```

join customers
on orders.customer_id=customers.customer_id
go

```

exec ShowAllOrders

146 %

Results Messages

	order_id	customer_name	order_date	order_status	total_prime_cost	total_cost
1	1	Красное&Белое	2022-05-23 14:25:10.000	pending	479.00	575.61
2	2	Красное&Белое	2022-05-24 14:25:10.000	processing	571.00	730.86
3	3	РосАл	2022-05-25 14:25:10.000	pending	1155.00	1433.41
4	4	РосАл	2022-05-26 14:25:10.000	completed	868.00	1121.68
5	5	РосАл	2022-05-27 14:25:10.000	cancelled	2628.00	3114.18
6	6	Beerka	2022-05-30 14:25:10.000	completed	954.00	1279.52
7	7	Beerka	2022-07-27 14:25:10.000	completed	3420.00	4024.20

7. Вывод информации обо всех заказах по различным фильтрам (customer_name, order_status, date)
(сложный)

```

ShowOrdersByCustomer_nameAndOrder_status_ForWorker,
ShowOrdersByOrder_statusAnd_Period_ForWorker,
ShowOrdersByOrder_status_ForWorker,
ShowOrdersByCustomer_nameAnd_Period_ForWorker,
ShowOrdersByCustomer_name_ForWorker,
ShowAllOrders_Period_ForWorker,

```

Запросы практически идентичны, отличия только в фильтре

create or alter procedure

ShowOrdersByCustomer_nameAndOrder_statusAnd_Period_ForWorker

```

@customer_name varchar(30),
@order_status varchar(30),
@period int

```

as

```

select order_id,
customer_name,
order_date,
order_status,
total_prime_cost,
total_cost

```

from Orders

```

join customers

```

```

on orders.customer_id=customers.customer_id

```

```

where customer_name = @customer_name and order_status = @order_status and DATEDIFF(day,
order_date, GETDATE()) < @period

```

go

exec ShowOrdersByCustomer_nameAndOrder_statusAnd_Period_ForWorker 'Росал', 'pending', 365

146 %

Results Messages

	order_id	customer_name	order_date	order_status	total_prime_cost	total_cost
1	3	РосАл	2022-05-25 14:25:10.000	pending	1155.00	1433.41

8. Вывод информации о своих заказах для заказчика

```
create or alter procedure ShowOrdersByCustomer_id_ForCustomer
    @customer_id int
as
select order_id,
       order_date,
       order_status,
       total_cost
from Orders
where customer_id = @customer_id
go
```

exec ShowOrdersByCustomer_id_ForCustomer 2

146 %

Results Messages

	order_id	order_date	order_status	total_cost
1	3	2022-05-25 14:25:10.000	pending	1433.41
2	4	2022-05-26 14:25:10.000	completed	1121.68
3	5	2022-05-27 14:25:10.000	cancelled	3114.18

9. Вывод информации о своих заказах для заказчика по различным фильтрам (order_status, date)
 ShowOrdersByCustomer_idAndOrder_status_ForCustomer,
 ShowOrdersByCustomer_id_Period_ForCustomer

Запросы практически идентичны, отличия только в фильтре

```
create or alter procedure ShowOrdersByCustomer_idAndOrder_status_Period_ForCustomer
    @customer_id int,
    @order_status varchar(30),
    @period int
as
select order_id,
       order_date,
       order_status,
       total_cost
from Orders
where customer_id = @customer_id and order_status = @order_status and DATEDIFF(day,
order_date, GETDATE()) < @period
go
```

exec ShowOrdersByCustomer_idAndOrder_status_Period_ForCustomer 2, pending, 365

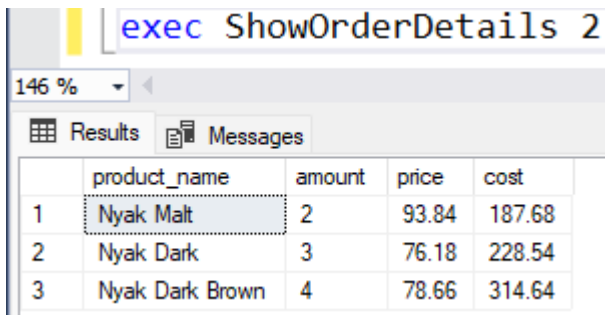
146 %

Results Messages

	order_id	order_date	order_status	total_cost
1	3	2022-05-25 14:25:10.000	pending	1433.41

10. Вывод содержимого заказа (сложный)

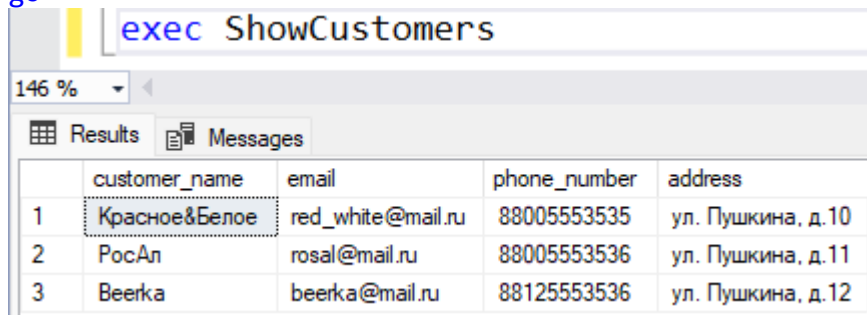
```
create or alter procedure ShowOrderDetails
@order_id int
as
select product_name,
        amount,
        price,
        cost
from OrdersOrderDetailsCustomersView
where order_id = @order_id
GO
```



	product_name	amount	price	cost
1	Nyak Malt	2	93.84	187.68
2	Nyak Dark	3	76.18	228.54
3	Nyak Dark Brown	4	78.66	314.64

11. Вывод информации о заказчиках

```
create or alter procedure ShowCustomers
as
select customer_name,
        email,
        phone_number,
        address
from customers
go
```



	customer_name	email	phone_number	address
1	Красное&Белое	red_white@mail.ru	88005553535	ул. Пушкина, д.10
2	РосАл	rosal@mail.ru	88005553536	ул. Пушкина, д.11
3	Beerka	beerka@mail.ru	88125553536	ул. Пушкина, д.12

12. Вывод значения прибыли завода за все время

```
create or alter procedure ShowAllIncome
as
select sum(total_cost - total_prime_cost) as income
from Orders
where order_status='completed'
go
```

exec ShowAllIncome

146 %

Results Messages

	income
1	1183.40

13. Вывод значения прибыли завода от конкретного заказчика

```
create or alter procedure ShowIncomeByCustomer_name
    @customer_name varchar(30)
as
select sum(total_cost - total_prime_cost) as income
    from Orders
        join customers
            on orders.customer_id=customers.customer_id
where order_status='completed' and customer_name=@customer_name
go
```

exec ShowIncomeByCustomer_name 'Росал'

146 %

Results Messages

	income
1	253.68

14. Вывод информации о сотрудниках завода

```
create or alter procedure ShowWorkers
as
select username,
    email,
    first_name,
    second_name,
    user_role
from workers
    join users on workers.user_id=users.user_id
go
```

exec ShowWorkers

146 %

Results Messages

	username	email	first_name	second_name	user_role
1	admin	admin@brewery.com	Nikita	Tsapov	admin
2	v.pupkin	v.pupkin@brewery.com	Василий	Пупкин	worker
3	n.pechkin	n.pechkin@brewery.com	Николай	Печкин	worker

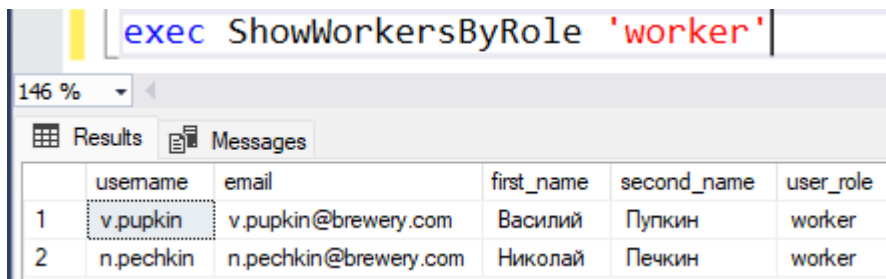
15. Вывод информации о сотрудниках завода по фильтру роли

```
create or alter procedure ShowWorkersByRole
    @user_role varchar(10)
as
```

```

select username,
       email,
       first_name,
       second_name,
       user_role
from workers
join users on workers.user_id=users.user_id
where user_role = @user_role
go

```



	username	email	first_name	second_name	user_role
1	v.pupkin	v.pupkin@brewery.com	Василий	Пупкин	worker
2	n.pechkin	n.pechkin@brewery.com	Николай	Печкин	worker

Вставка

1. Вставка товара в базу товара

```

create or alter procedure InsertProduct
    @product_name varchar(30),
    @product_type varchar(30),
    @upc_code varchar(12),
    @prime_price decimal(8, 2),
    @discount decimal(8, 2),
    @amount int
as
BEGIN TRANSACTION
    INSERT INTO products(product_name, product_type, upc_code, prime_price)
        VALUES (@product_name, @product_type, @upc_code, @prime_price)
    Declare @product_id INT
    Set @product_id=(select product_id from products where product_name=@product_name)
    EXEC UpdateDiscount @product_id, @discount
    EXEC UpdateAmountOnStock @product_id, @amount
    IF (@@error <> 0)
        ROLLBACK
COMMIT
GO

```

2. Вставка учетной записи в таблицу сотрудников завода(сложный)

```

create or alter procedure InsertWorker
    @email varchar(30),
    @first_name varchar(30),
    @second_name varchar(30),
    @user_password varchar(30)
as

```



```

BEGIN
BEGIN TRANSACTION
    INSERT INTO users(username, user_password, user_role)
        VALUES (left(@email, charindex('@', @email) - 1), HASHBYTES('SHA2_256',
@user_password), 'worker');
    INSERT INTO workers(user_id, email, first_name, second_name)
        VALUES (
            (select user_id from users
                where username=left(@email, charindex('@', @email) - 1)),
            @email,
            @first_name,
            @second_name
        )
COMMIT
END
GO

```

3. Вставка учетной записи в таблицу заказчиков(сложный)

```

create or alter procedure InsertCustomer
    @customer_name varchar(30),
    @email varchar(30),
    @phone_number varchar(30),
    @address varchar(30),
    @user_password varchar(30)
as
BEGIN TRANSACTION
    INSERT INTO users(username,
        user_password,
        user_role)
        VALUES (left(@email, charindex('@', @email) - 1),
            HASHBYTES('SHA2_256', @user_password),
            'customer')
    INSERT INTO customers(user_id,
        email,
        customer_name,
        phone_number,
        address)
        VALUES ((select user_id from users
            where username=left(@email, charindex('@', @email) - 1)),
            @email,
            @customer_name,
            @phone_number,
            @address
        )
    IF (@@error <> 0)

```

ROLLBACK

COMMIT

GO

4. Вставка нового заказа(сложный)

a. В таблицу orders

create or alter procedure InsertOrder

@customer_id INT,

@order_date DATETIME

as

INSERT INTO orders(customer_id, order_date)

VALUES (@customer_id, @order_date)

go

b. В таблицу order_details

create or alter procedure InsertOrder_details

@order_id INT,

@product_id INT,

@amount INT

as

INSERT INTO order_details(order_id, product_id, amount)

VALUES (@order_id, @product_id, @amount)

GO

Описание программы

Задание заключается в разработке многопользовательской автоматизированной системы управления организацией. Объект автоматизации – пивной завод.

В качестве СУБД была предложена Microsoft SQL Server.

Microsoft SQL Server — система управления реляционными базами данных, разработанная корпорацией Microsoft. Основным используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

Для реализации логики системы было подготовлено множество запросов на языке Transact-SQL. “Обертка” была написана на языке Python. Связь приложения с базой была реализована с помощью модуля QSql библиотеки PyQt6, в качестве драйвера выступает драйвер SQL Server ODBC.

Python - высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным в том плане, что всё является объектами. Необычной особенностью языка является выделение блоков кода пробельными отступами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов. Недостатками языка являются зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как C или C++.

Графический интерфейс для системы был так же написан на языке Python с использованием библиотеки PyQt6.

PyQt — набор расширений (биндингов) графического фреймворка Qt для языка программирования Python, выполненный в виде расширения Python.

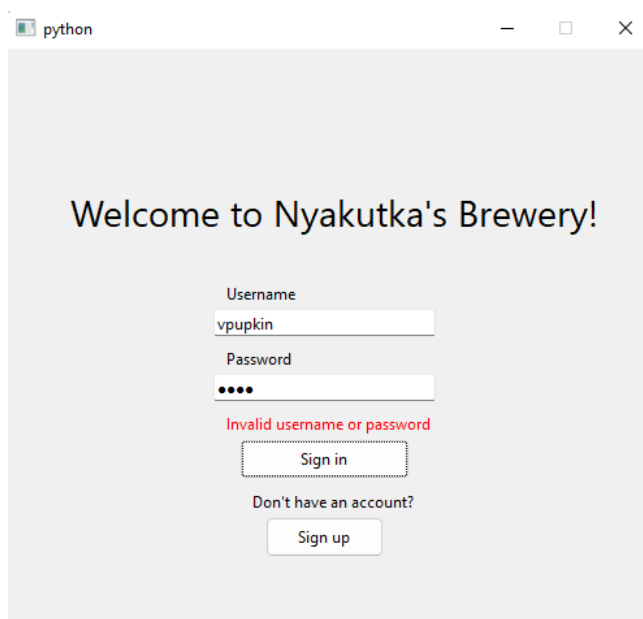
PyQt разработан британской компанией Riverbank Computing. PyQt работает на всех платформах, поддерживаемых Qt: Linux и другие UNIX-подобные ОС, macOS и Windows. Существует 3 версии: PyQt6, PyQt5 и PyQt4, поддерживающие соответствующие версии Qt. PyQt распространяется под лицензиями GPL (2 и 3 версии) и коммерческой.

В качестве среды разработки был использован многофункциональный инструмент Visual Studio Code.

Visual Studio Code (VS Code) — текстовый редактор, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией.

Скриншоты системы

Авторизация и регистрация



Welcome to Nyakutka's Brewery!

Username
vpupkin

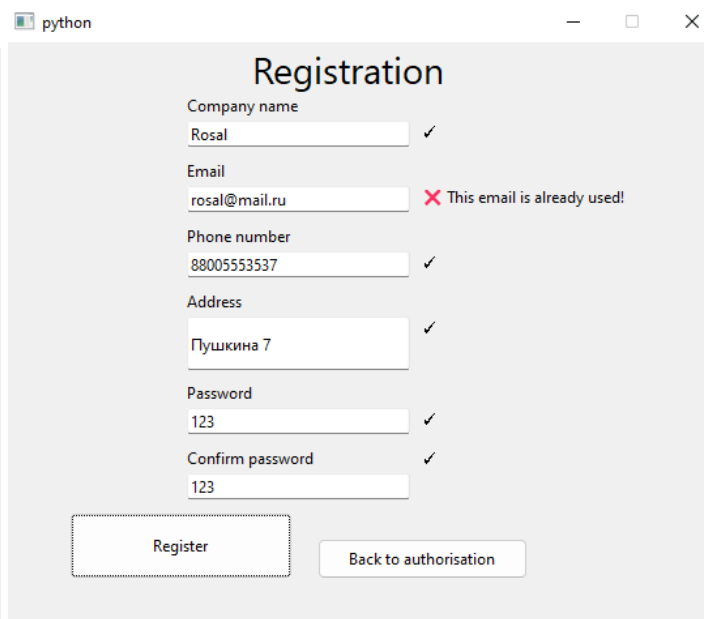
Password
••••

Invalid username or password

Sign in

Don't have an account?
Sign up

Рисунок 1. Неуспешная попытка авторизации



Registration

Company name
Rosal ✓

Email
rosal@mail.ru ✗ This email is already used!

Phone number
88005553537 ✓

Address
Пушкина 7 ✓

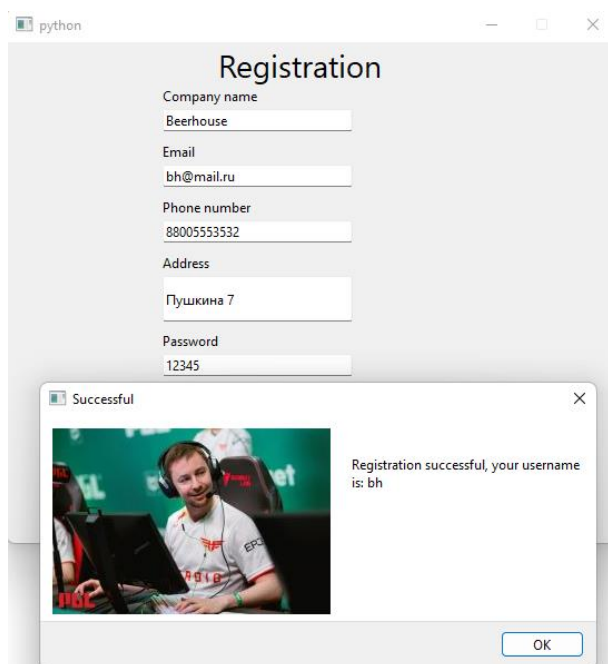
Password
123 ✓

Confirm password
123 ✓

Register

Back to authorisation

Рисунок 2. Неуспешная попытка регистрации



Registration

Company name
Beerhouse

Email
bh@mail.ru

Phone number
88005553532

Address
Пушкина 7

Password
12345

Successful

Registration successful, your username is: bh

OK

Рисунок 3. Успешная регистрация

Окно администратора ИС

python

Nyakutka's Brewery

You are signed in as admin: admin [Sign out](#)

Workers

	Username	Email	First Name	Second Name	Role	
1	v.pupkin	v.pupkin@brewery.com	Василий	Пупкин	worker	delete
2	n.pechkin	n.pechkin@brewery.com	Николай	Печкин	worker	delete

Role

[New worker](#) [Submit](#) [Cancel](#)

Email


First Name

Second Name

Password

Рисунок 4. Окно администратора системы в режиме создания учетной записи рабочего

Successful



Registration of new worker successful,
his username is: n.tsapov

[OK](#)

[New worker](#) [Submit](#) [Cancel](#)

Email

First Name

Second Name

Password

Рисунок 5. Успешное создание учетной записи рабочего

Окно рабочего завода (Менеджер по продажам)

python

Nyakutka's Brewery

You are signed in as worker: v.pupkin

Sign out

CatalogOrdersMy accountCustomers

	Name	Type	UPC Code	Prime price	Retail price	Discount price	Amount on stock	
1	Nyak Fici	Beer	123456789121	67.0	92.46	83.21	1	update info
2	Nyak Malt	Beer	123456789122	68.0	93.84	93.84	2	update info
3	Nyak Dark	Beer	123456789123	69.0	95.22	76.18	3	update info
4	Nyak Dark Brown	Beer	123456789124	57.0	78.66	78.66	4	update info
5	Nononti	Beer	123456789125	58.0	80.04	80.04	5	update info
6	Narmara	Beer	123456789126	60.0	82.8	82.8	5	update info
7	Nara Nuzu	Beer	123456789127	75.0	103.5	82.8	5	update info
8	Nera	Beer	123456789128	80.0	110.4	99.36	5	update info

New product

Products lack on stock

all

	Product Name	Lack on stock
1	Nyak Fici	11
2	Nyak Malt	2
3	Nyak Dark	34
4	Nyak Dark Brown	10
5	Nononti	2

Рисунок 6. Окно рабочего завода, вкладка Catalog

python

Nyakutka's Brewery

You are signed in as worker: v.pupkin

Sign out

CatalogOrdersMy accountCustomers

Successful

Product Baltica 7 was added to catalog

OK

	Name	Type	UPC Code	Prime price	Retail price	Discount price	Amount on stock	
6	Narmara	Beer	123456789126	60.0	82.8	82.8	5	update info
7	Nara Nuzu	Beer	123456789127	75.0	103.5	82.8	5	update info
8	Nera	Beer	123456789128	80.0	110.4	99.36	5	update info

New product

Submit

Cancel

Product name

Baltica 7

Type

Beer

UPC

123456789986

Prime price

60,00

Discount

0,00

Amount on stock

10

Retail price:

82.8

Discount price:

82.8

Products lack on stock

all

	Product Name	Lack on stock
1	Nyak Fici	11
2	Nyak Malt	2
3	Nyak Dark	34
4	Nyak Dark Brown	10
5	Nononti	2

Рисунок 7. Успешное добавление нового продукта в базу

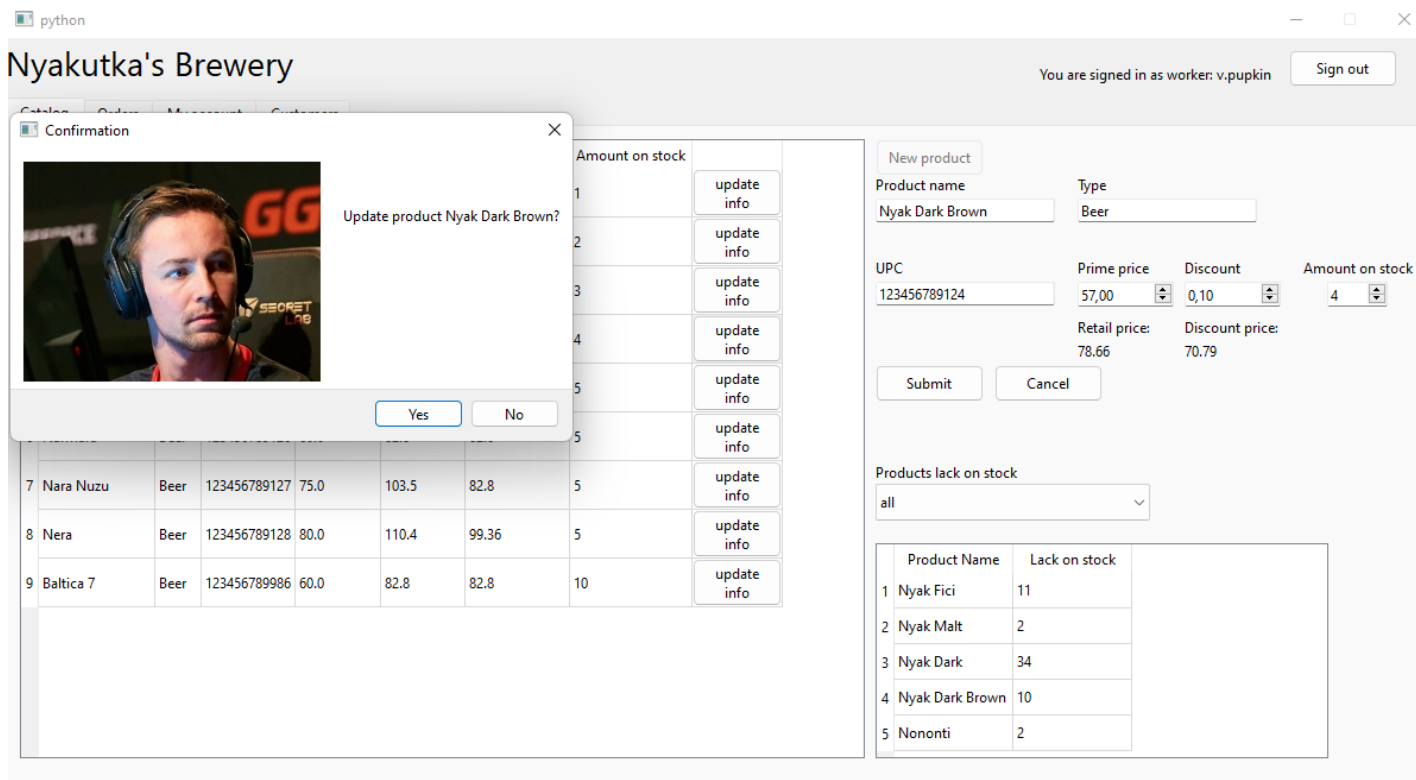


Рисунок 8. Обновление данных о продукте, запрос на подтверждение

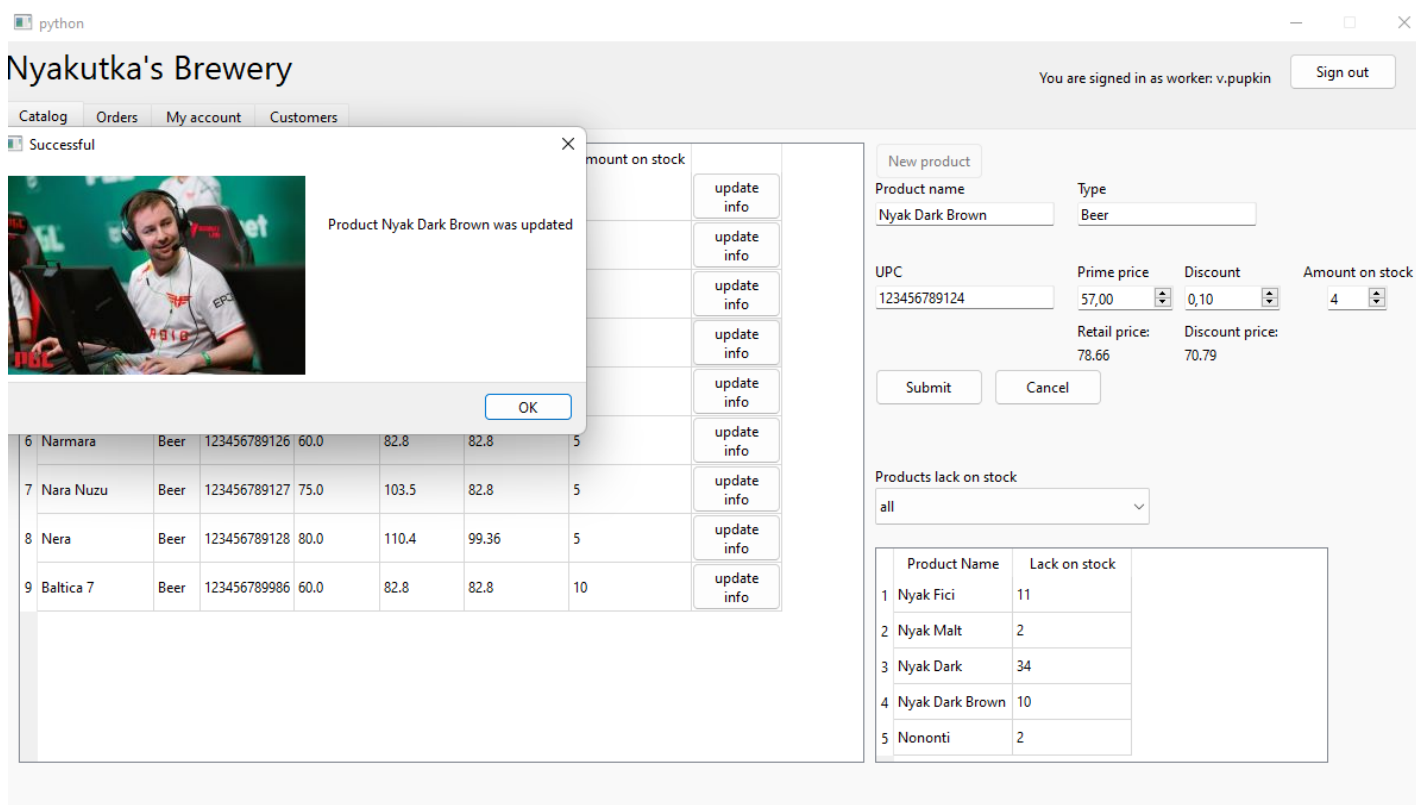


Рисунок 9. Обновление данных о продукте прошло успешно

python

Nyakutka's Brewery

You are signed in as worker: v.pupkin Sign out

Catalog

Orders

My account

Customers

	Order number	Customer name	Date	Status	Prime cost	Cost		
1	1	Красное&Белое	Mon May 23 14:25:10 2022	pending	479.0	575.61	show details	Set status ▾
2	2	Красное&Белое	Tue May 24 14:25:10 2022	processing	571.0	730.86	show details	Set status ▾
3	3	РосАл	Wed May 25 14:25:10 2022	pending	1155.0	1433.41	show details	Set status ▾
4	4	РосАл	Thu May 26 14:25:10 2022	completed	868.0	1121.68	show details	Set status ▾
5	5	РосАл	Fri May 27 14:25:10 2022	cancelled	2628.0	3114.18	show details	Set status ▾
6	6	Beerka	Mon May 30 14:25:10 2022	completed	954.0	1279.52	show details	Set status ▾
7	7	Beerka	Wed Jul 27 14:25:10 2022	completed	3420.0	4024.2	show details	Set status ▾

Filter by status:

all ▾

Filter by date:

all ▾

Filter by customer:

all ▾

Details of Order: 3

Hide

	Product Name	Amount	Price	Cost
1	Nyak Fici	5	83.21	416.05
2	Nyak Dark	6	76.18	457.08
3	Nononti	7	80.04	560.28

Products lack on stock

	Product Name	Lack on stock
1	Nyak Fici	4
2	Nyak Dark	3
3	Nononti	2

Рисунок 10. Окно рабочего завода, вкладка Orders

python

Nyakutka's Brewery

You are signed in as worker: v.pupkin Sign out

Catalog

Orders

My account

Customers

	Order number	Customer name	Date	Status	Prime cost	Cost		
1	1	Красное&Белое	Mon May 23 14:25:10 2022	pending	479.0	575.61	show details	Set status ▾
2	2	Красное&Белое	Tue May 24 14:25:10 2022	processing	571.0	730.86	show details	Set status ▾
3	3	РосАл	Wed May 25 14:25:10 2022	pending	1155.0	1433.41	show details	Set status ▾
4	4	РосАл	Thu May 26 14:25:10 2022	completed	868.0	1121.68	show details	Set status ▾
5	5	РосАл	Fri May 27 14:25:10 2022	cancelled	2628.0	3114.18	show details	Set status ▾
6	6	Beerka	Mon May 30 14:25:10 2022	completed	954.0	1279.52	show details	Set status ▾
7	7	Beerka	Wed Jul 27 14:25:10 2022	completed	3420.0	4024.2	show details	Set status ▾

Filter by status:

all ▾

Filter by date:

all ▾

Filter by customer:

all ▾

Details of Order: 7

Hide

	Product Name	Amount	Price	Cost
1	Nara Nuzu	6	82.8	496.8
2	Nyak Dark	30	76.18	2285.4
3	Narmara	15	82.8	1242.0

Products lack on stock

	Product Name	Lack on stock
--	--------------	---------------

Set status

Set status

cancelled

pending

processing

completed

Рисунок 11. Окно рабочего завода, вкладка Catalog, изменение статуса заказа

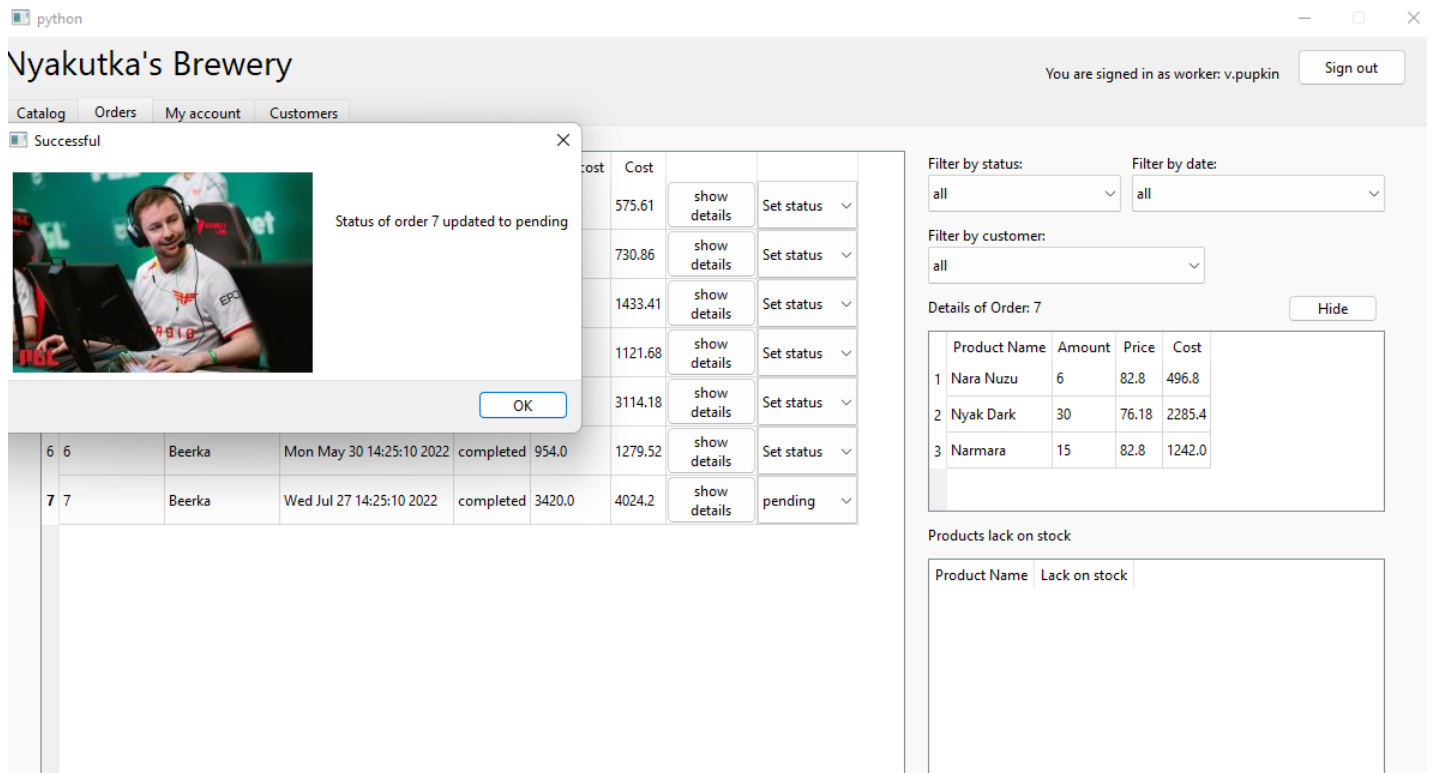


Рисунок 12. Окно рабочего завода, вкладка Catalog, изменение статуса заказа прошло успешно

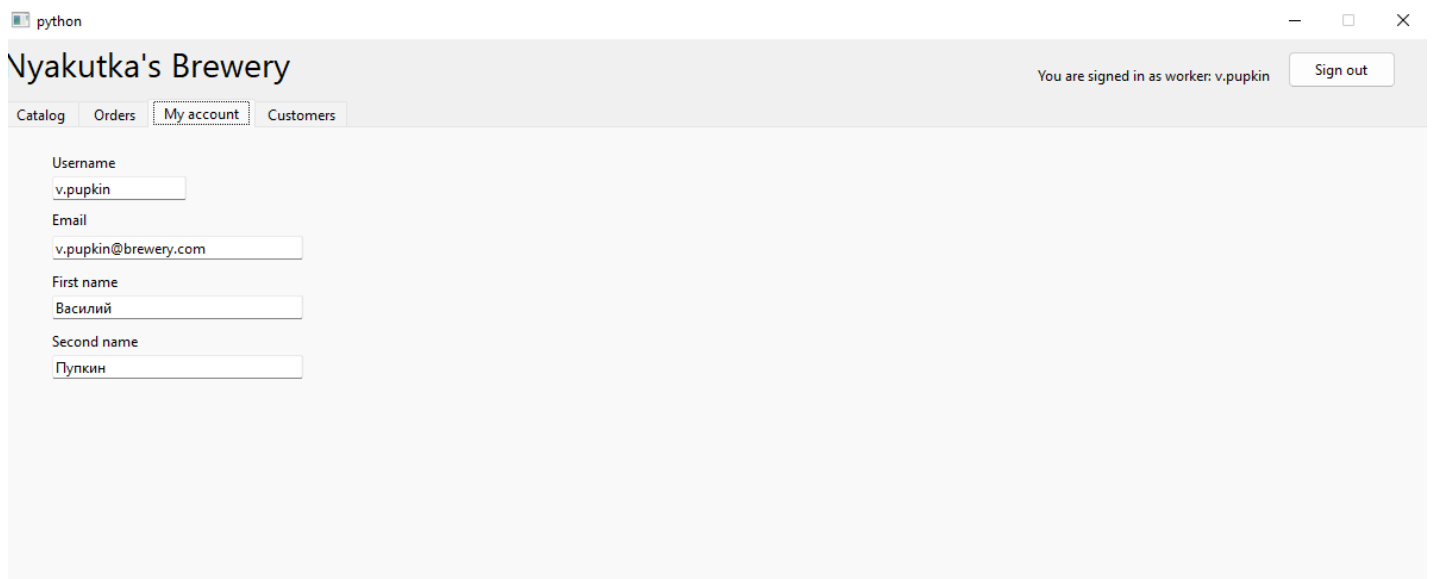


Рисунок 13. Окно рабочего завода, вкладка My account

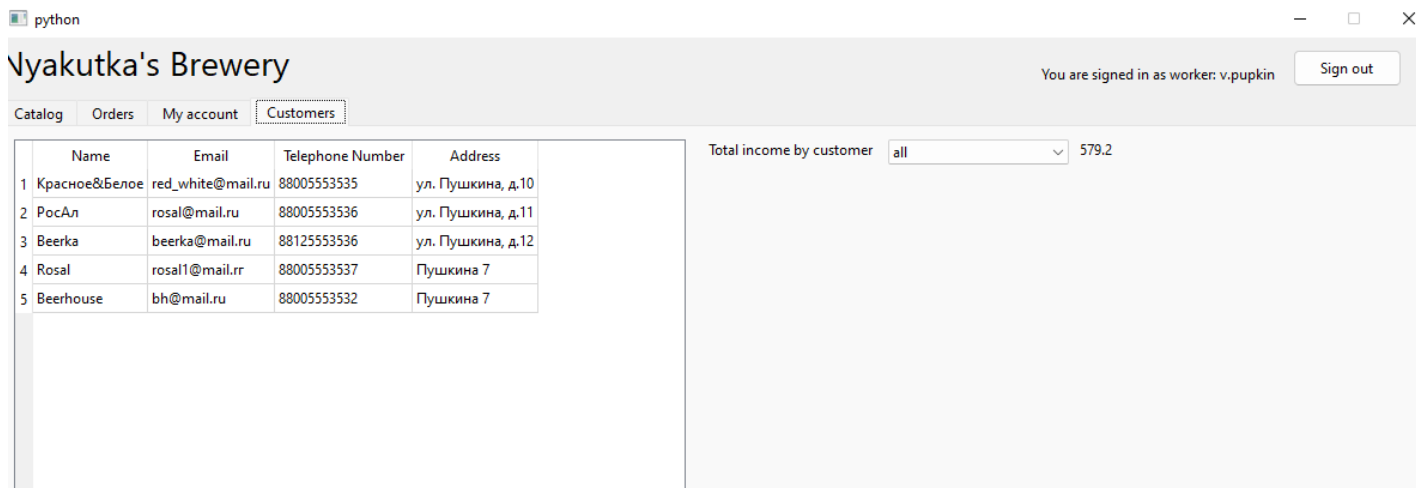


Рисунок 14. Окно рабочего завода, вкладка Customers

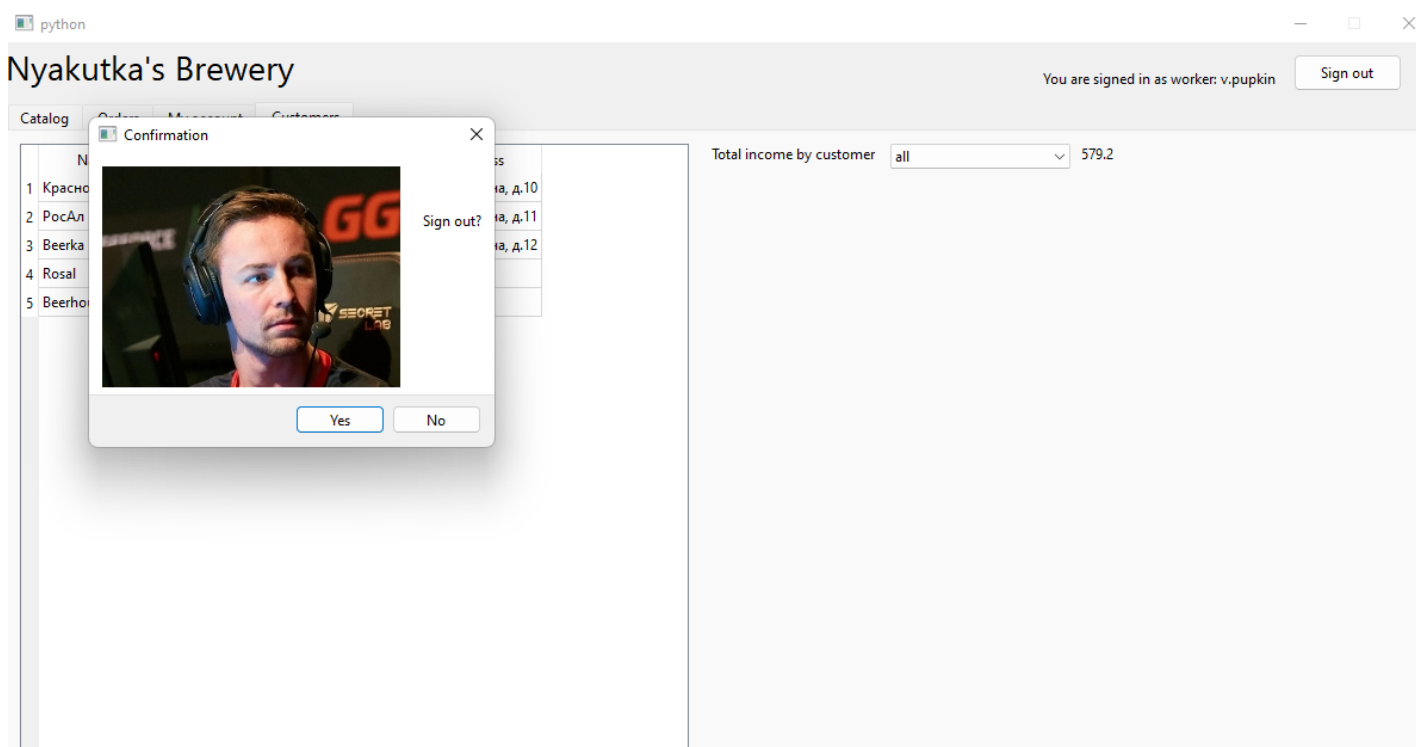


Рисунок 15. Подтверждение выхода из системы

Окно менеджера по закупкам (розничные магазины)

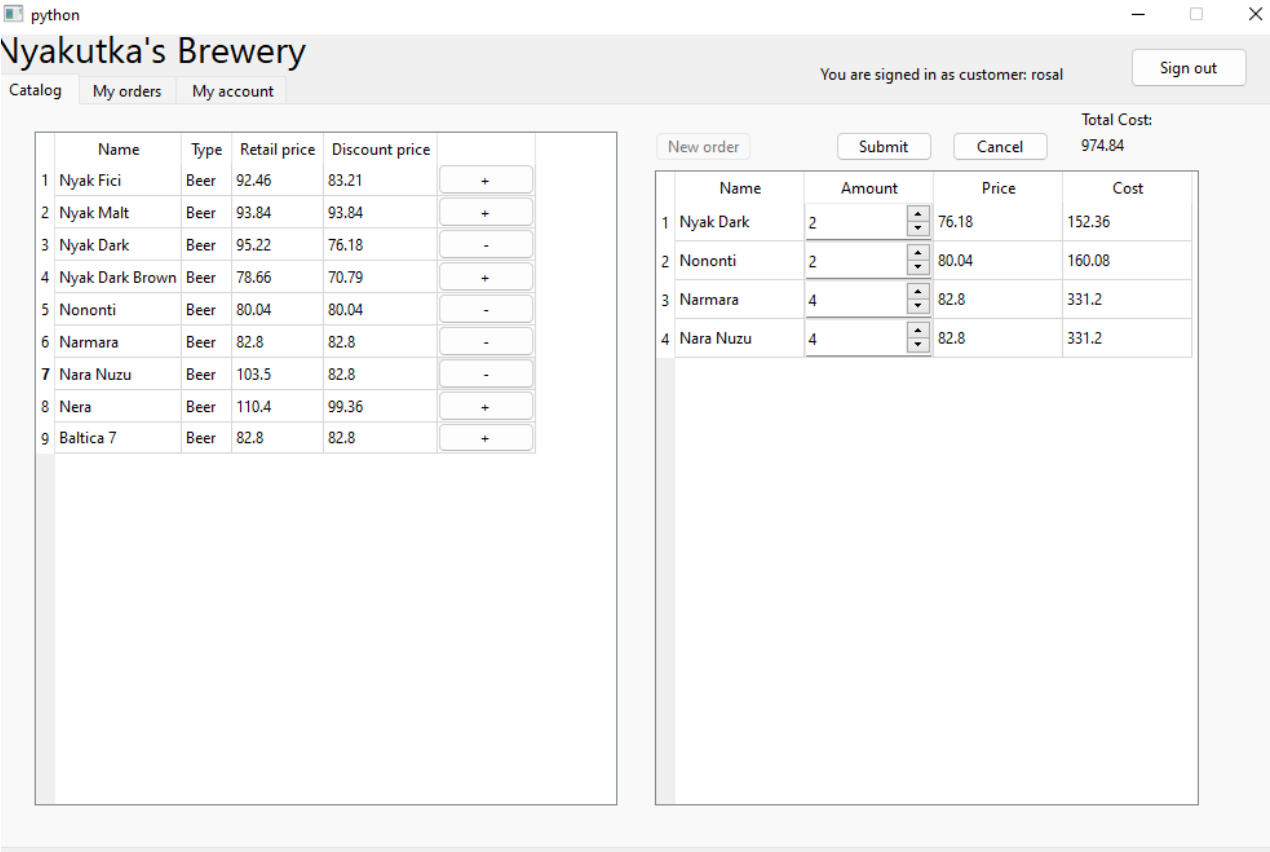


Рисунок 16. Окно пользователя, вкладка Catalog, формирование нового заказа

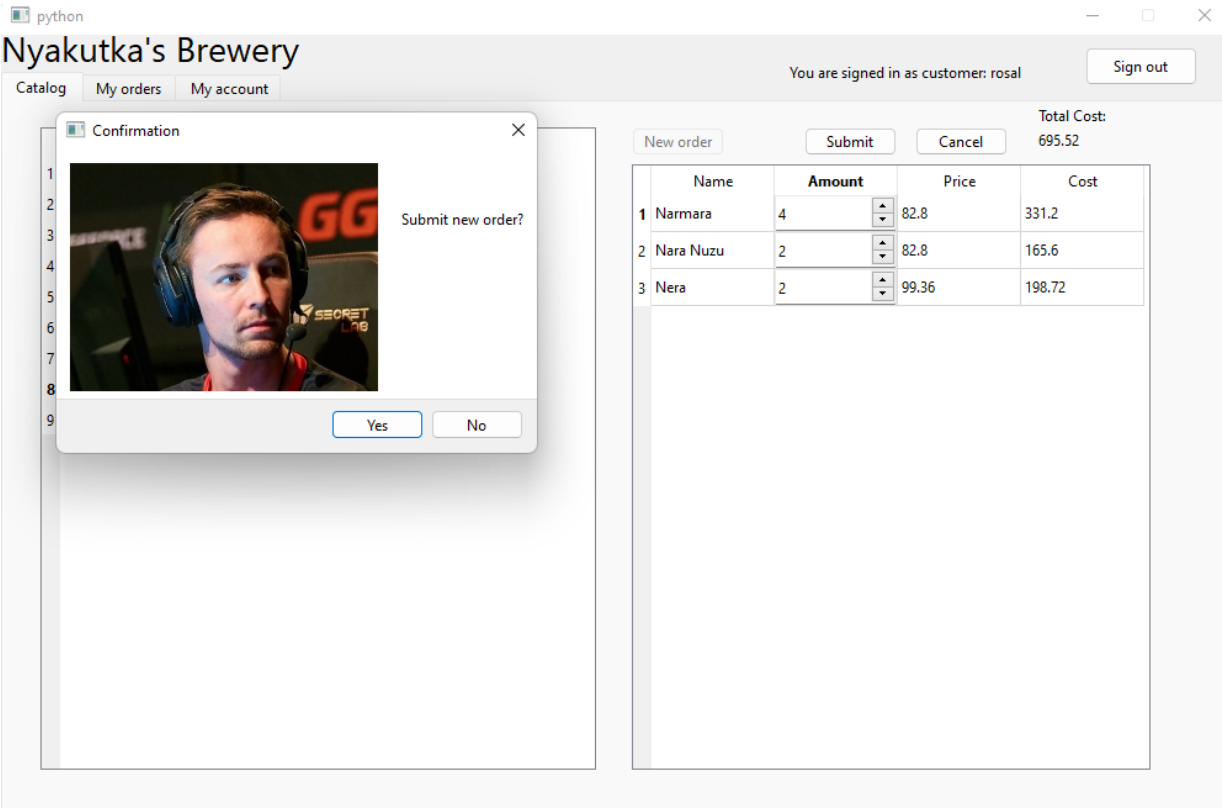


Рисунок 17. Окно пользователя, вкладка Catalog, формирование нового заказа, подтверждение

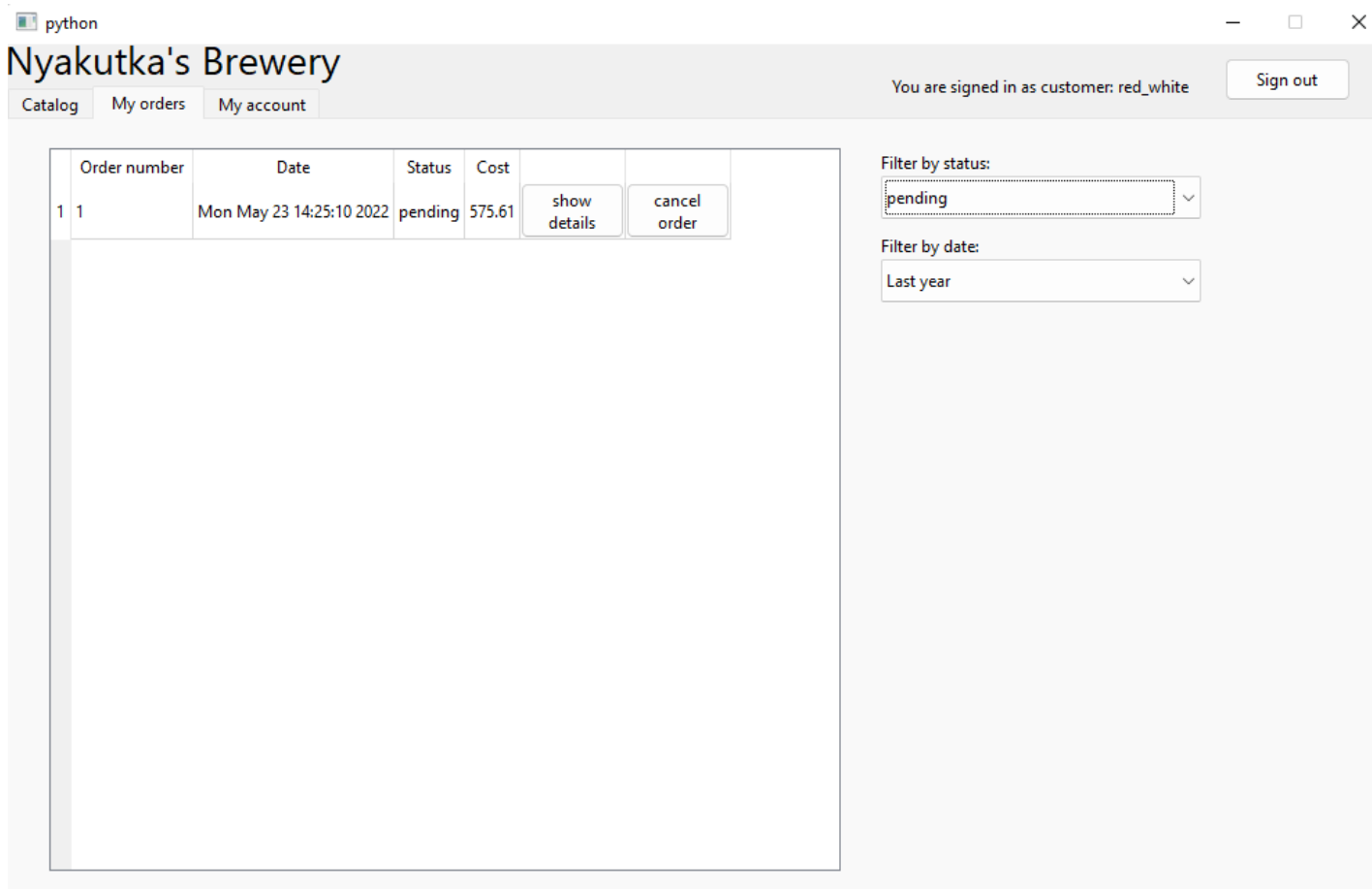


Рисунок 18. Окно пользователя, вкладка My orders, фильтры по статусу заказа и по date

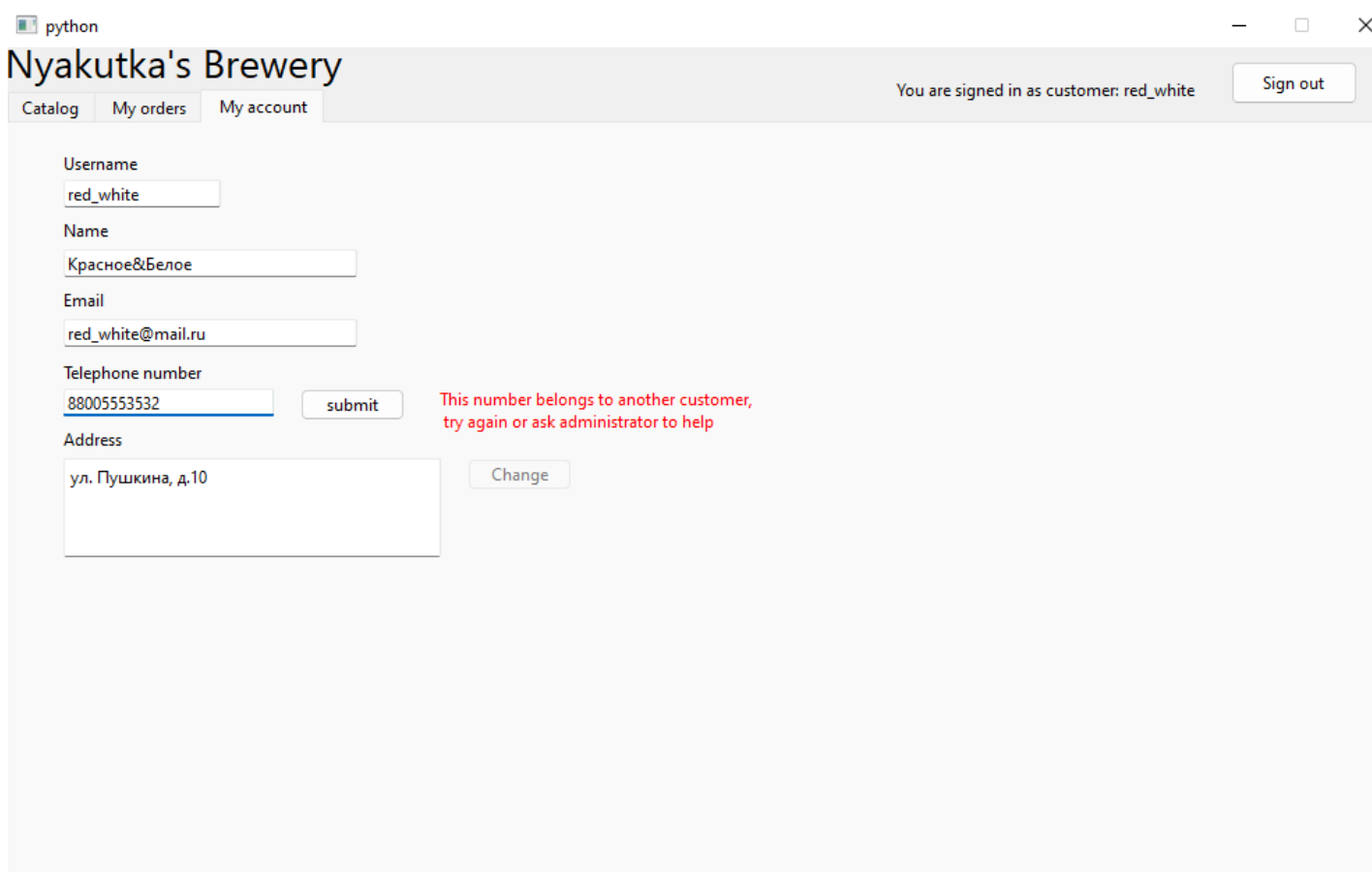


Рисунок 19. Окно пользователя, вкладка My account, неуспешная попытка изменить данные

Заключение

Задача была решена в полной мере, а именно:

- 1) Сформирован анализ предметной области, где были объявлены все функции и особенности приложения
- 2) Составлена структура базы данных
- 3) Реализованы запросы к базе данных
- 4) Создано приложение с графическим интерфейсом пользователя

В процессе написания запросов убедился, что хоть инструменты ORM во многих случаях удобны, но существуют такие задачи, которые требуют сырых запросов на языке SQL. Так же я познакомился с инструментами для создания графического интерфейса, которые в начале работы давались мне нелегко.

Ссылка на код проекта в github: <https://github.com/Nyakutka/brewery>

Список литературы

- 1) Введение в MS SQL Server и T-SQL // METANIT URL:
<https://metanit.com/sql/sqlserver/1.1.php> (дата обращения: 21.10.2022).
- 2) Справочник по Transact-SQL (ядро СУБД) // Microsoft learn URL:
<https://learn.microsoft.com/ru-ru/sql/t-sql/language-reference?view=sql-server-ver16> (дата обращения: 21.10.2022).
- 3) Qt for Python // PyQt Documentation URL: <https://doc.qt.io/qtforpython/> (дата обращения: 21.10.2022).

Приложение (скрипт создания бд)

```
USE [master]
GO
/***** Object:  Database [brewery]      Script Date: 06.12.2022 19:04:16 *****/
CREATE DATABASE [brewery]
    CONTAINMENT = NONE
    ON PRIMARY
( NAME = N'brewery', FILENAME = N'D:\Programs\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\DATA\brewery.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED,
FILEGROWTH = 65536KB )
    LOG ON
( NAME = N'brewery_log', FILENAME = N'D:\Programs\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\DATA\brewery_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB ,
FILEGROWTH = 65536KB )
    WITH CATALOG_COLLATION = DATABASE_DEFAULT
GO
ALTER DATABASE [brewery] SET COMPATIBILITY_LEVEL = 150
GO
IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))
begin
EXEC [brewery].[dbo].[sp_fulltext_database] @action = 'enable'
end
GO
ALTER DATABASE [brewery] SET ANSI_NULL_DEFAULT OFF
GO
ALTER DATABASE [brewery] SET ANSI_NULLS OFF
GO
ALTER DATABASE [brewery] SET ANSI_PADDING OFF
GO
ALTER DATABASE [brewery] SET ANSI_WARNINGS OFF
GO
ALTER DATABASE [brewery] SET ARITHABORT OFF
GO
ALTER DATABASE [brewery] SET AUTO_CLOSE OFF
GO
ALTER DATABASE [brewery] SET AUTO_SHRINK OFF
GO
ALTER DATABASE [brewery] SET AUTO_UPDATE_STATISTICS ON
GO
ALTER DATABASE [brewery] SET CURSOR_CLOSE_ON_COMMIT OFF
GO
ALTER DATABASE [brewery] SET CURSOR_DEFAULT GLOBAL
GO
ALTER DATABASE [brewery] SET CONCAT_NULL_YIELDS_NULL OFF
```

```

GO
ALTER DATABASE [brewery] SET NUMERIC_ROUNDABORT OFF
GO
ALTER DATABASE [brewery] SET QUOTED_IDENTIFIER OFF
GO
ALTER DATABASE [brewery] SET RECURSIVE_TRIGGERS OFF
GO
ALTER DATABASE [brewery] SET  ENABLE_BROKER
GO
ALTER DATABASE [brewery] SET AUTO_UPDATE_STATISTICS_ASYNC OFF
GO
ALTER DATABASE [brewery] SET DATE_CORRELATION_OPTIMIZATION OFF
GO
ALTER DATABASE [brewery] SET TRUSTWORTHY OFF
GO
ALTER DATABASE [brewery] SET ALLOW_SNAPSHOT_ISOLATION OFF
GO
ALTER DATABASE [brewery] SET PARAMETERIZATION SIMPLE
GO
ALTER DATABASE [brewery] SET READ_COMMITTED_SNAPSHOT OFF
GO
ALTER DATABASE [brewery] SET HONOR_BROKER_PRIORITY OFF
GO
ALTER DATABASE [brewery] SET RECOVERY FULL
GO
ALTER DATABASE [brewery] SET  MULTI_USER
GO
ALTER DATABASE [brewery] SET PAGE_VERIFY CHECKSUM
GO
ALTER DATABASE [brewery] SET DB_CHAINING OFF
GO
ALTER DATABASE [brewery] SET FILESTREAM( NON_TRANSACTED_ACCESS = OFF )
GO
ALTER DATABASE [brewery] SET TARGET_RECOVERY_TIME = 60 SECONDS
GO
ALTER DATABASE [brewery] SET DELAYED_DURABILITY = DISABLED
GO
ALTER DATABASE [brewery] SET ACCELERATED_DATABASE_RECOVERY = OFF
GO
EXEC sys.sp_db_vardecimal_storage_format N'brewery', N'ON'
GO
ALTER DATABASE [brewery] SET QUERY_STORE = OFF
GO
USE [brewery]

```



```

GO

/***** Object:  UserDefinedFunction [dbo].[isValidEmail]      Script Date: 06.12.2022 19:04:16
*****/

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

--TABLES
CREATE FUNCTION [dbo].[isValidEmail](@EMAIL varchar(30))
RETURNS bit as
BEGIN
    DECLARE @bitRetVal as Bit
    IF (@EMAIL = '' OR @EMAIL NOT LIKE '_%@__%.__%')
        SET @bitRetVal = 0  -- Invalid
    ELSE
        SET @bitRetVal = 1  -- Valid
    RETURN @bitRetVal
END
GO

/***** Object:  UserDefinedFunction [dbo].[isValidPhoneNumber]  Script Date: 06.12.2022
19:04:16 *****/

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

CREATE FUNCTION [dbo].[isValidPhoneNumber](@PHONE_NUMBER varchar(30))
RETURNS bit as
BEGIN
    DECLARE @bitRetVal as Bit
    IF (LEN(@PHONE_NUMBER) != 11 or ISNUMERIC(@PHONE_NUMBER) = 0)
        SET @bitRetVal = 0  -- Invalid
    ELSE
        SET @bitRetVal = 1  -- Valid
    RETURN @bitRetVal
END
GO

/***** Object:  Table [dbo].[products]      Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[products](

```

```

[product_id] [int] IDENTITY(1,1) NOT NULL,
[product_name] [varchar](30) NOT NULL,
[product_type] [varchar](30) NOT NULL,
[upc_code] [varchar](12) NOT NULL,
[prime_price] [decimal](8, 2) NOT NULL,
[retail_price] AS ([prime_price]*(1.38)),
[discount_price] [decimal](8, 2) NULL,
PRIMARY KEY CLUSTERED
(
    [product_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],
    CONSTRAINT [UQ_PRODUCT_NAME] UNIQUE NONCLUSTERED
(
    [product_name] 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],
    CONSTRAINT [UQ_PRODUCT_UPC] UNIQUE NONCLUSTERED
(
    [upc_code] 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
/***** Object: Table [dbo].[stock]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[stock](
    [product_id] [int] NOT NULL,
    [amount] [int] NOT NULL
) ON [PRIMARY]
GO
/***** Object: View [dbo].[ProductsStockView]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

--VIEWS
create view [dbo].[ProductsStockView] as
    select product_name,
           product_type,

```

```

        upc_code,
        prime_price,
        retail_price,
        discount_price,
        amount
    from products
        join stock on products.product_id=stock.product_id
GO
/***** Object: Table [dbo].[customers]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[customers](
    [user_id] [int] NOT NULL,
    [customer_id] [int] IDENTITY(1,1) NOT NULL,
    [email] [varchar](30) NOT NULL,
    [customer_name] [varchar](30) NOT NULL,
    [phone_number] [varchar](30) NOT NULL,
    [address] [varchar](30) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [customer_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],
    CONSTRAINT [UQ_CUSTOMER_EMAIL] UNIQUE NONCLUSTERED
(
    [email] 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],
    CONSTRAINT [UQ_CUSTOMER_PHONE_NUMBER] UNIQUE NONCLUSTERED
(
    [phone_number] 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
/***** Object: Table [dbo].[orders]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[orders](
    [order_id] [int] IDENTITY(1,1) NOT NULL,

```

```

[customer_id] [int] NOT NULL,
[order_date] [datetime] NOT NULL,
[order_status] [varchar](30) NULL,
[total_prime_cost] [decimal](8, 2) NULL,
[total_cost] [decimal](8, 2) NULL,
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
/***** Object: Table [dbo].[order_details]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[order_details](
    [order_id] [int] NOT NULL,
    [product_id] [int] NOT NULL,
    [amount] [int] NOT NULL,
    [price] [decimal](8, 2) NULL,
    [cost] AS ([price]*[amount])
) ON [PRIMARY]
GO
/***** Object: View [dbo].[OrdersOrderDetailsCustomersView]    Script Date: 06.12.2022
19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

create view [dbo].[OrdersOrderDetailsCustomersView] as
    select orders.order_id,
           customer_name,
           order_date,
           orders.order_status,
           products.product_id,
           product_name,
           amount,
           price,
           cost
    from orders
        join order_details on orders.order_id=order_details.order_id

```

```

        join products on order_details.product_id=products.product_id
        join customers on orders.customer_id=customers.customer_id
GO
/***** Object: Table [dbo].[discounts]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[discounts](
    [product_id] [int] NOT NULL,
    [discount] [decimal](3, 2) NOT NULL
) ON [PRIMARY]
GO
/***** Object: Table [dbo].[users]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[users](
    [user_id] [int] IDENTITY(1,1) NOT NULL,
    [username] [varchar](30) NOT NULL,
    [user_password] [varbinary](256) NOT NULL,
    [user_role] [varchar](10) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [user_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],
UNIQUE NONCLUSTERED
(
    [username] 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
/***** Object: Table [dbo].[workers]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[workers](
    [user_id] [int] NOT NULL,
    [worker_id] [int] IDENTITY(1,1) NOT NULL,
    [email] [varchar](30) NOT NULL,

```

```

[first_name] [varchar](30) NOT NULL,
[second_name] [varchar](30) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [worker_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],
    CONSTRAINT [UQ_WORKER_EMAIL] UNIQUE NONCLUSTERED
(
    [email] 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] ADD DEFAULT ('pending') FOR [order_status]
GO
ALTER TABLE [dbo].[customers] WITH CHECK ADD FOREIGN KEY([user_id])
REFERENCES [dbo].[users] ([user_id])
GO
ALTER TABLE [dbo].[discounts] WITH CHECK ADD FOREIGN KEY([product_id])
REFERENCES [dbo].[products] ([product_id])
GO
ALTER TABLE [dbo].[order_details] WITH CHECK ADD FOREIGN KEY([order_id])
REFERENCES [dbo].[orders] ([order_id])
GO
ALTER TABLE [dbo].[order_details] WITH CHECK ADD FOREIGN KEY([product_id])
REFERENCES [dbo].[products] ([product_id])
GO
ALTER TABLE [dbo].[orders] WITH CHECK ADD FOREIGN KEY([customer_id])
REFERENCES [dbo].[customers] ([customer_id])
GO
ALTER TABLE [dbo].[stock] WITH CHECK ADD FOREIGN KEY([product_id])
REFERENCES [dbo].[products] ([product_id])
GO
ALTER TABLE [dbo].[workers] WITH CHECK ADD FOREIGN KEY([user_id])
REFERENCES [dbo].[users] ([user_id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[customers] WITH CHECK ADD CHECK (([address]<>''))
GO
ALTER TABLE [dbo].[customers] WITH CHECK ADD CHECK (([customer_name]<>''))
GO
ALTER TABLE [dbo].[customers] WITH CHECK ADD CONSTRAINT [CK_CUSTOMER_EMAIL]
CHECK ((([dbo].[isValidEmail]([email])=(1))))

```

```

GO
ALTER TABLE [dbo].[customers] CHECK CONSTRAINT [CK_CUSTOMER_EMAIL]
GO
ALTER TABLE [dbo].[customers] WITH CHECK ADD CONSTRAINT [CK_CUSTOMER_PHONE_NUMBER]
CHECK (([dbo].[isValidPhoneNumber]([phone_number])=(1)))
GO
ALTER TABLE [dbo].[customers] CHECK CONSTRAINT [CK_CUSTOMER_PHONE_NUMBER]
GO
ALTER TABLE [dbo].[products] WITH CHECK ADD CHECK (([product_name]<>''))
GO
ALTER TABLE [dbo].[products] WITH CHECK ADD CONSTRAINT [CK_PRODUCT_UPC]
CHECK ((len([upc_code])=(12)))
GO
ALTER TABLE [dbo].[products] CHECK CONSTRAINT [CK_PRODUCT_UPC]
GO
ALTER TABLE [dbo].[users] WITH CHECK ADD CHECK (([username]<>''))
GO
ALTER TABLE [dbo].[users] WITH CHECK ADD CONSTRAINT [CK_USER_PASSWORD]
CHECK (([user_password]<>''))
GO
ALTER TABLE [dbo].[users] CHECK CONSTRAINT [CK_USER_PASSWORD]
GO
ALTER TABLE [dbo].[workers] WITH CHECK ADD CHECK (([first_name]<>''))
GO
ALTER TABLE [dbo].[workers] WITH CHECK ADD CHECK (([second_name]<>''))
GO
ALTER TABLE [dbo].[workers] WITH CHECK ADD CONSTRAINT [CK_WORKER_EMAIL]
CHECK (([dbo].[isValidEmail]([email])=(1)))
GO
ALTER TABLE [dbo].[workers] CHECK CONSTRAINT [CK_WORKER_EMAIL]
GO
/***** Object: StoredProcedure [dbo].[DeleteWorker] Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--DELETE--
create procedure [dbo].[DeleteWorker]
    @user_id int
as
    DELETE from users
        where user_id=@user_id
GO

```

/***** Object: StoredProcedure [dbo].[InsertCustomer] Script Date: 06.12.2022 19:04:16

*****/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

create procedure [dbo].[InsertCustomer]

@customer_name varchar(30),

@email varchar(30),

@phone_number varchar(30),

@address varchar(30),

@user_password varchar(30)

as

BEGIN TRANSACTION;

INSERT INTO users(username,
user_password,
user_role)

VALUES (left(@email, charindex('@', @email) - 1),
HASHBYTES('SHA2_256', @user_password),
'customer')

IF (@@error <> 0)

ROLLBACK

INSERT INTO customers(user_id,
email,
customer_name,
phone_number,
address)

VALUES ((select user_id from users
where username=left(@email, charindex('@', @email) - 1)),
@email,
@customer_name,
@phone_number,
@address
)

COMMIT TRANSACTION;

GO

/***** Object: StoredProcedure [dbo].[InsertOrder] Script Date: 06.12.2022 19:04:16 *****/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

create procedure [dbo].[InsertOrder]


```

    @customer_id INT,
    @order_date DATETIME
as
    INSERT INTO orders(customer_id, order_date)
        VALUES (@customer_id, @order_date)
GO

/***** Object:  StoredProcedure [dbo].[InsertOrder_details]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

create  procedure [dbo].[InsertOrder_details]
    @order_id INT,
    @product_id INT,
    @amount INT
as
    INSERT INTO order_details(order_id, product_id, amount)
        VALUES (@order_id, @product_id, @amount)
GO

/***** Object:  StoredProcedure [dbo].[InsertProduct]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---
--INSERT--
create  procedure [dbo].[InsertProduct]
    @product_name varchar(30),
    @product_type varchar(30),
    @upc_code varchar(12),
    @prime_price decimal(8, 2),
    @discount decimal(8, 2),
    @amount int
as
BEGIN TRANSACTION
    INSERT INTO products(product_name, product_type, upc_code, prime_price)
        VALUES (@product_name, @product_type, @upc_code, @prime_price)
    Declare @product_id INT
    Set @product_id=(select product_id from products where product_name=@product_name)
    EXEC UpdateDiscount @product_id, @discount
    EXEC UpdateAmountOnStock @product_id, @amount

```

```

    IF (@@error <> 0)
        ROLLBACK
COMMIT
GO
/***** Object:  StoredProcedure [dbo].[InsertWorker]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

---
create  procedure [dbo].[InsertWorker]
    @email varchar(30),
    @first_name varchar(30),
    @second_name varchar(30),
    @user_password varchar(30)
as
BEGIN
BEGIN TRANSACTION
    INSERT INTO users(username, user_password, user_role)
        VALUES (left(@email, charindex('@', @email) - 1), HASHBYTES('SHA2_256', @user_password),
'worker');
    INSERT INTO workers(user_id, email, first_name, second_name)
        VALUES (
            (select user_id from users
                where username=left(@email, charindex('@', @email) - 1)),
            @email,
            @first_name,
            @second_name
        )
COMMIT
END
GO
/***** Object:  StoredProcedure [dbo].[ShowAllIncome]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

---
create  procedure [dbo].[ShowAllIncome]
as
select sum(total_cost - total_prime_cost) as income

```

```

        from Orders
where order_status='completed'
GO
/***** Object:  StoredProcedure [dbo].[ShowAllOrders]      Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--
create  procedure [dbo].[ShowAllOrders]
as
select order_id,
        customer_name,
        order_date,
        order_status,
        total_prime_cost,
        total_cost
from Orders
    join customers
        on orders.customer_id=customers.customer_id
GO
/***** Object:  StoredProcedure [dbo].[ShowAllOrders_Period_ForWorker]      Script Date:
06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--
create  procedure [dbo].[ShowAllOrders_Period_ForWorker]
    @period int
as
select order_id,
        customer_name,
        order_date,
        order_status,
        total_prime_cost,
        total_cost
from Orders
    join customers
        on orders.customer_id=customers.customer_id
where DATEDIFF(day, order_date, GETDATE()) < @period
GO

```

```

/***** Object:  StoredProcedure [dbo].[ShowCustomers]      Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---
create  procedure [dbo].[ShowCustomers]
as
select customer_name,
        email,
        phone_number,
        address
from customers
GO

/***** Object:  StoredProcedure [dbo].[ShowIncomeByCustomer_name]      Script Date: 06.12.2022
19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---
create  procedure [dbo].[ShowIncomeByCustomer_name]
        @customer_name varchar(30)
as
select sum(total_cost - total_prime_cost) as income
        from Orders
        join customers
        on orders.customer_id=customers.customer_id
where order_status='completed' and customer_name=@customer_name
GO

/***** Object:  StoredProcedure [dbo].[ShowLackProducts]      Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--
create  procedure [dbo].[ShowLackProducts]
as
select product_name,
        abs(amount_on_stock - ordered) as lack_on_stock
from (select product_name,
        sum(OrdersOrderDetailsCustomersView.amount) as ordered,

```

```

        stock.amount as amount_on_stock
    from OrdersOrderDetailsCustomersView
        join stock
            on OrdersOrderDetailsCustomersView.product_id=stock.product_id
    where order_status != 'completed'
    group by product_name, stock.amount)
as a
where (amount_on_stock - ordered) < 0
GO

/***** Object:  StoredProcedure [dbo].[ShowLackProductsByCustomer_name]    Script Date:
06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---
create procedure [dbo].[ShowLackProductsByCustomer_name]
    @customer_name varchar(30)
as
select product_name,
    abs(amount_on_stock - ordered) as lack_on_stock
from
    (select product_name,
        sum(OrdersOrderDetailsCustomersView.amount) as ordered,
        stock.amount as amount_on_stock
    from OrdersOrderDetailsCustomersView join stock on
OrdersOrderDetailsCustomersView.product_id=stock.product_id
    where order_status != 'completed' and customer_name = @customer_name
    group by product_name, stock.amount)
as a
where (amount_on_stock - ordered) < 0
GO

/***** Object:  StoredProcedure [dbo].[ShowLackProductsByOrder_id]    Script Date: 06.12.2022
19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---
create procedure [dbo].[ShowLackProductsByOrder_id]
    @order_id int
as
select product_name,
    abs(amount_on_stock - ordered) as lack_on_stock

```

```

from
    (select product_name,
        sum(OrdersOrderDetailsCustomersView.amount) as ordered,
        stock.amount as amount_on_stock
    from OrdersOrderDetailsCustomersView
        join stock on OrdersOrderDetailsCustomersView.product_id=stock.product_id
    where order_status != 'completed' and order_id = @order_id
group by product_name, stock.amount)
as a
where (amount_on_stock - ordered) < 0
GO
/***** Object:  StoredProcedure [dbo].[ShowOrderDetails]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---
create procedure [dbo].[ShowOrderDetails]
@order_id int
as
select product_name,
    amount,
    price,
    cost
from OrdersOrderDetailsCustomersView
where order_id = @order_id
GO
/***** Object:  StoredProcedure [dbo].[ShowOrdersByCustomer_id_ForCustomer]    Script Date:
06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--
create procedure [dbo].[ShowOrdersByCustomer_id_ForCustomer]
    @customer_id int
as
select order_id,
    order_date,
    order_status,
    total_cost
from Orders
where customer_id = @customer_id

```

```

GO
/***** Object:  StoredProcedure [dbo].[ShowOrdersByCustomer_id_Period_ForCustomer]    Script
Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---
create procedure [dbo].[ShowOrdersByCustomer_id_Period_ForCustomer]
    @customer_id int,
    @period int
as
select order_id,
       order_date,
       order_status,
       total_cost
from Orders
where customer_id = @customer_id and DATEDIFF(day, order_date, GETDATE()) < @period
GO
/***** Object:  StoredProcedure
[dbo].[ShowOrdersByCustomer_idAndOrder_status_ForCustomer]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--
create procedure [dbo].[ShowOrdersByCustomer_idAndOrder_status_ForCustomer]
    @customer_id int,
    @order_status varchar(30)
as
select order_id,
       order_date,
       order_status,
       total_cost
from Orders
where customer_id = @customer_id and order_status = @order_status
GO
/***** Object:  StoredProcedure
[dbo].[ShowOrdersByCustomer_idAndOrder_status_Period_ForCustomer]    Script Date: 06.12.2022
19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON

```

```

GO
---
create procedure [dbo].[ShowOrdersByCustomer_idAndOrder_status_Period_ForCustomer]
    @customer_id int,
    @order_status varchar(30),
    @period int
as
select order_id,
    order_date,
    order_status,
    total_cost
from Orders
where customer_id = @customer_id and order_status = @order_status and DATEDIFF(day, order_date,
GETDATE()) < @period
GO
/***** Object:  StoredProcedure [dbo].[ShowOrdersByCustomer_name_ForWorker]    Script Date:
06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--
create procedure [dbo].[ShowOrdersByCustomer_name_ForWorker]
    @customer_name varchar(30)
as
select order_id,
    customer_name,
    order_date,
    order_status,
    total_prime_cost,
    total_cost
from Orders
    join customers
        on orders.customer_id=customers.customer_id
where customer_name = @customer_name
GO
/***** Object:  StoredProcedure [dbo].[ShowOrdersByCustomer_nameAnd_Period_ForWorker]    Script
Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---
create procedure [dbo].[ShowOrdersByCustomer_nameAnd_Period_ForWorker]

```



```

        @customer_name varchar(30),
        @period int
as
select order_id,
        customer_name,
        order_date,
        order_status,
        total_prime_cost,
        total_cost
from Orders
    join customers
        on orders.customer_id=customers.customer_id
where customer_name = @customer_name and DATEDIFF(day, order_date, GETDATE()) < @period
GO

/***** Object:  StoredProcedure
[dbo].[ShowOrdersByCustomer_nameAndOrder_status_ForWorker]    Script Date: 06.12.2022 19:04:16
*****/

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--
create procedure [dbo].[ShowOrdersByCustomer_nameAndOrder_status_ForWorker]
    @customer_name varchar(30),
    @order_status varchar(30)
as
select order_id,
        customer_name,
        order_date,
        order_status,
        total_prime_cost,
        total_cost
from Orders
    join customers
        on orders.customer_id=customers.customer_id
where customer_name = @customer_name and order_status = @order_status
GO

/***** Object:  StoredProcedure
[dbo].[ShowOrdersByCustomer_nameAndOrder_statusAnd_Period_ForWorker]    Script Date: 06.12.2022
19:04:16 *****/

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

```

```

---
create procedure [dbo].[ShowOrdersByCustomer_nameAndOrder_statusAnd_Period_ForWorker]
    @customer_name varchar(30),
    @order_status varchar(30),
    @period int
as
select order_id,
        customer_name,
        order_date,
        order_status,
        total_prime_cost,
        total_cost
from Orders
    join customers
        on orders.customer_id=customers.customer_id
where customer_name = @customer_name and order_status = @order_status and DATEDIFF(day,
order_date, GETDATE()) < @period
GO

/***** Object:  StoredProcedure [dbo].[ShowOrdersByOrder_status_ForWorker]      Script Date:
06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--
create procedure [dbo].[ShowOrdersByOrder_status_ForWorker]
    @order_status varchar(30)
as
select order_id,
        customer_name,
        order_date,
        order_status,
        total_prime_cost,
        total_cost
from Orders
    join customers
        on orders.customer_id=customers.customer_id
where order_status = @order_status
GO

/***** Object:  StoredProcedure [dbo].[ShowOrdersByOrder_statusAnd_Period_ForWorker]      Script
Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON

```

```

GO
---
create procedure [dbo].[ShowOrdersByOrder_statusAnd_Period_ForWorker]
    @order_status varchar(30),
    @period int
as
select order_id,
        customer_name,
        order_date,
        order_status,
        total_prime_cost,
        total_cost
from Orders
    join customers
        on orders.customer_id=customers.customer_id
where order_status = @order_status and DATEDIFF(day, order_date, GETDATE()) < @period
GO
/***** Object:  StoredProcedure [dbo].[ShowProductsBase]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--PROCEDURES
---Просмотр базы товаров
create procedure [dbo].[ShowProductsBase]
as
select product_name,
        product_type,
        upc_code,
        prime_price,
        retail_price,
        discount_price,
        amount
from ProductsStockView
GO
/***** Object:  StoredProcedure [dbo].[ShowProductsCatalog]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---
create procedure [dbo].[ShowProductsCatalog]

```

```

as
select product_name,
        product_type,
        retail_price,
        discount_price
from ProductsStockView
GO

/***** Object:  StoredProcedure [dbo].[ShowWorkers]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---

---

---
create procedure [dbo].[ShowWorkers]
as
select username,
        email,
        first_name,
        second_name,
        user_role
from workers
    join users on workers.user_id=users.user_id
GO

/***** Object:  StoredProcedure [dbo].[ShowWorkersByRole]    Script Date: 06.12.2022 19:04:16 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
---

create procedure [dbo].[ShowWorkersByRole]
    @user_role varchar(10)
as
select username,
        email,
        first_name,
        second_name,
        user_role
from workers
    join users on workers.user_id=users.user_id

```

```

        where user_role = @user_role
GO

/***** Object:  StoredProcedure [dbo].[UpdateAmountOnStock]    Script Date: 06.12.2022 19:04:16
*****/

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

create procedure [dbo].[UpdateAmountOnStock]
    @product_id int,
    @amount int
as
    UPDATE stock
        set amount = @amount
    where product_id=@product_id
GO

/***** Object:  StoredProcedure [dbo].[UpdateCustomer]    Script Date: 06.12.2022 19:04:16
*****/

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

create procedure [dbo].[UpdateCustomer]
    @customer_id int,
    @phone_number varchar(30),
    @address varchar(30)
as
    UPDATE customers
        set phone_number = @phone_number,
            address = @address
    where customer_id=@customer_id
GO

/***** Object:  StoredProcedure [dbo].[UpdateDiscount]    Script Date: 06.12.2022 19:04:16
*****/

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

create procedure [dbo].[UpdateDiscount]
    @product_id int,
    @discount decimal(3, 2)

```

```

as
    UPDATE discounts
        set discount = @discount
    where product_id=@product_id
GO
/***** Object:  StoredProcedure [dbo].[UpdateOrderStatus]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

create  procedure [dbo].[UpdateOrderStatus]
    @order_id int,
    @order_status varchar(30)
as
    UPDATE orders
        set order_status = @order_status
    where order_id=@order_id
GO
/***** Object:  StoredProcedure [dbo].[UpdateProduct]    Script Date: 06.12.2022 19:04:16
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--UPDATE--
create  procedure [dbo].[UpdateProduct]
    @product_id int,
    @product_name varchar(30),
    @product_type varchar(30),
    @upc_code varchar(30),
    @prime_price varchar(30),
    @discount decimal(8, 2),
    @amount int
as
begin TRANSACTION
    UPDATE products
        set product_name = @product_name,
            product_type = @product_type,
            upc_code = @upc_code,
            prime_price = @prime_price
    where product_id=@product_id
    EXEC UpdateDiscount @product_id = @product_id, @discount = @discount

```

```
EXEC UpdateAmountOnStock @product_id = @product_id, @amount = @amount
IF (@@error <> 0)
    ROLLBACK
COMMIT
GO
USE [master]
GO
ALTER DATABASE [brewery] SET READ_WRITE
GO
```