

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение высшего  
образования  
«Санкт-Петербургский политехнический университет Петра Великого»  
Институт компьютерных наук и технологий  
Высшая школа программной инженерии

## **КУРСОВОЙ ПРОЕКТ**

### **РАЗРАБОТКА МНОГОПОЛЬЗОВАТЕЛЬСКОЙ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ОРГАНИЗАЦИЕЙ по дисциплине «Базы данных»**

Выполнил  
студент гр. 3530202/80201

Григоренко С.А.

Руководитель

Гасанова И.А.

«25» декабря 2020г.

Санкт-Петербург  
2020

**ЗАДАНИЕ**  
**НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА (КУРСОВОЙ РАБОТЫ)**

студенту  
группы

3530202/80201  
(номер группы)

Григоренко Сергею Андреевичу  
(фамилия, имя, отчество)

**1. Тема проекта (работы)** Разработка многопользовательской  
автоматизированной системы управления организацией. Задание №15

Объект автоматизации – сберегательный банк.

**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: «07» сентября 2020 г.**

Руководитель

\_\_\_\_\_  
(подпись)

\_\_\_\_\_  
(инициалы, фамилия)

Задание принял к исполнению

  
\_\_\_\_\_  
(подпись)

Григоренко С.А.

\_\_\_\_\_  
(инициалы, фамилия)

07.09.2020

\_\_\_\_\_  
(дата)

## Содержание

<b>Введение.</b>	<b>4</b>
<b>Описание предметной области.</b>	<b>5</b>
<b>Схема базы данных и описание таблиц.</b>	<b>12</b>
<b>Код создания таблиц</b>	<b>14</b>
Таблица Clients	14
Таблица BankAccounts	14
Таблица Services	15
Таблица Transactions	15
Таблица Auth	16
Таблица Exchange	16
Таблица Workers	17
<b>Запросы</b>	<b>18</b>
Представления	18
Добавление	19
Изменение	19
Удаление	20
Сложные запросы	21
Хранимые процедуры	26
Триггеры	30
<b>Клиентская часть</b>	<b>35</b>
Аутентификация	35
Администрация	36
Обзор	36
Работники	37
Управление Услугами	40
Услуги	42
Транзакции	42
Оператор	43
Клиенты	43
Счета	46
Транзакции	49
Услуги	50
Клиент	51
Аккаунт	51
Транзакции	52
Услуги	53
<b>Вывод</b>	<b>54</b>
<b>Список литературы.</b>	<b>55</b>

## **Введение.**

Работа банка напрямую связана с хранением и обработкой огромного количества информации. Развитие информационных банковских систем и удобный доступ к ним открывает большие возможности для социальной и экономической мобильности. Следовательно тема представляет собой актуальную на сегодняшний день задачу. Для работы такой системы необходимо хранить данные работников банка и его клиентов. Важно правильно хранить и обрабатывать финансовые данные: счета, транзакции. Мы постараемся пройти все этапы необходимые для создания такой системы.

## Описание предметной области.

Группы пользователей разрабатываемой информационной системы (ИС)

№ пп	Наименование пользователя
1	Оператор
2	Клиент
3	Администрация

## Функции групп пользователей

№ пп	Выполняемая функция	Входные данные	Выходные данные	Функции, которые должны быть реализованы в ИС
<b>Оператор</b>				
1	Регистрация клиента	Номер паспорта, ФИО клиента, дата рождения	Изменение БД	Добавление информации о новом клиенте в базу данных
2	Просмотр данных клиента	Номер паспорта	Данные о клиенте	Выдача данных о требуемом клиенте
3	Открытие счета	Номер паспорта, Id услуги, Сумма	Изменение БД	Добавление информации о новом счете клиента в базу данных
4	Блокировка счета	Номер паспорта, номер счета	Изменение БД	Перевод счета в неактивное состояние. Если клиент хочет закрыть счет, это тоже считается блокировкой, но с предварительным снятием средств или погашением долга
5	Совершение транзакции	сумма, номер счета начисления, номер счета списания валюта	Изменение БД	Если указаны как счет списания так и счет зачисления, то транзакция это перевод между этими счетами (использовать счет кредита в качестве

				счета списания запрещено). Если не указан счет начисления, то деньги считаются выданными на руки. Если не указан счет списания, то считается, что оператор получил деньги от клиента наличными.
6	Регистрация клиента в онлайн банке	Номер паспорта, логин, пароль	Изменение БД	Регистрация клиента в онлайн банке. Считается, что оператор просит клиента ввести логин и пароль самостоятельно, так что оператор не имеет доступа к этим данным
7	Просмотр списка возможных вкладов и кредитов		Id услуг	Выдача списка всех возможных условий финансовых услуг
8	Выписка транзакций клиента	Номер паспорта, дата начала периода, дата окончания периода	Номера счетов списания, зачисления, даты, суммы, валюты	Выдает записи из таблицы транзакций в которых номер счета списания или зачисления принадлежат обслуживаемому клиенту, за определенный промежуток времени
<b>Клиент</b>				
1	Просмотр данных клиента	Логин онлайн банка, пароль	Изменение БД	Выдача данных о требуемом клиенте
2	Просмотр данных счетов	Логин онлайн банка, пароль	Счета	БД предоставляет данные о счетах клиента

3	Совершение транзакции	Логин онлайн банка, пароль, выбранный счет для списания средств, счет начисления, сумма, валюта	Изменение БД	БД уменьшает сумму выбранного счета клиента и переводит эти деньги на указанный счет начисления
4	Просмотреть список вкладов и кредитов		Id услуг	Выдача списка всех условий финансовых услуг
5	Список транзакций	Логин онлайн банка, пароль, количество транзакций которые необходимо о показать (транзакции и отсортированы по новизне)	Номера счетов списания, зачисления, даты, суммы, валюты	Выдает клиенту записи из таблицы транзакций в которых номер счета списания или зачисления принадлежат, клиенту
<b>Администрация</b>				
1	Добавить возможные условия кредита	Процентная ставка, срок в месяцах, условия досрочного погашения, условия просрочки платежа, валюта	Изменение БД, Id услуги	Администрация добавляет новый вид кредита
2	Добавить возможные условия вклада	Процентная ставка, срок в	Изменение БД, Id услуги	Администрация добавляет новый вид вклада

		месяцах, условия досрочного закрытия, валюта		
3	Заблокировать вид кредита или вклада	Id услуги	Изменение БД	Условия кредита или вклада больше не используются для создания новых счетов, но те кто уже имеют счета с данными условиями могут продолжать пользоваться ими до, окончания установленного срока пользования услугой.
4	Просмотреть список возможных вкладов и кредитов		Id услуг	Выдача списка всех возможных условий финансовых услуг
5	Просмотреть все транзакции за период времени	Дата начала, дата конца, количество транзакций которые необходим о показать (транзакции и отсортированы по новизне)	Id транзакций, Номера счетов списания, зачисления, даты, суммы, валюта	Выдает клиенту из таблицы транзакций за определенный промежуток времени
6	Регистрация оператора	Номер паспорта, ФИО оператора, дата рождения, ИНН, Справка об отсутствии судимости	Изменение БД	Добавляет данные оператора в таблицу работников и дает оператору права на исполнение его рабочих обязанностей



7	Удаление оператора	Номер паспорта	Изменение БД	Удаляет личные данные оператора из таблицы работников и отнимает его права на доступ к данным
---	--------------------	----------------	--------------	---

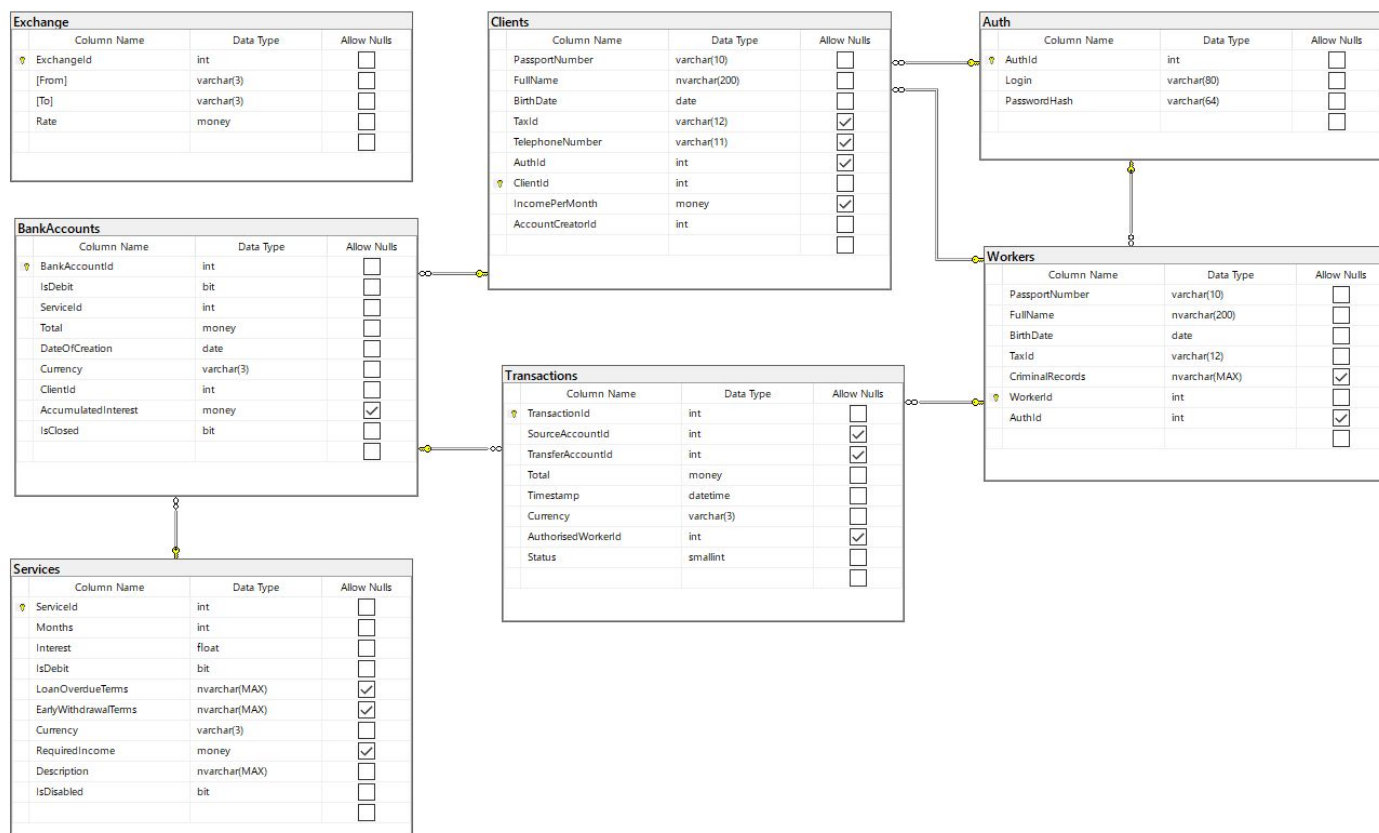
#### Хранимые данные

№ пп	Хранимые данные	Пользователи, которым разрешен доступ	Ограничения по типу и значению
1	Данные клиента (номер паспорта, ФИО, Дата рождения, массив счетов, доход в месяц, ИНН, номер телефона, Id аутентификации)	Клиент, оператор, администрация	ФИО строка символов(кириллица или латинские буквы), телефон целое положительное число из 11 цифр, номер паспорта целое положительное число из 10 цифр, дата рождения в формате dd.mm.yyyy, счета и кредиты должны быть целыми, положительными числами, доход в месяц - неотрицательное целое число, ИНН целое положительное число из 12 цифр, Id аутентификации - целое неотрицательное число
2	Счета( вклады и кредиты) (тип, номер счета, Id услуги, сумма, дата создания, валюта, Id клиента, суммарные проценты, флаг активности)	Клиент, оператор, администрация	Тип – булево значение (true – счет, false - кредит), номер счета - целое положительное число, Id услуги – целое неотрицательное число, сумма – вещественное число, дата создания - dd.mm.yyyy, валюта – enum (usd, eur, rub, jpy, cny), Id клиента – целое неотрицательное число, флаг активности - булево значение (true - счет активен, false - счет заблокирован)
3	Условия Счетов(Id услуги, срок в месяцах, процентная ставка, условия досрочного погашения, условия	Клиент, оператор, администрация	Id услуги – целое неотрицательное число, срок в месяцах – целое положительное число, условия досрочного погашения – ascii символы, условия просрочки – ascii символы, условия досрочного

	просрочки, условия досрочного снятия, валюта, флаг активности)		снятия – ascii символы, валюта – enum (usd, eur, rub, jpy, cny), флаг активности - булево значение (true - услуга активна, false - услуга недоступна для открытия новых счетов)
4	Транзакции (Id транзакции, номер счета списания(может быть пустым, если клиент платил наличными), номер счета зачисления (может быть пустым, если клиент получил деньги на руки), сумма, дата проведения, валюта, Id работника авторизовавшего транзакцию, статус)	Клиент, оператор, администрация	номер счета - целое положительное число (или null для счета списания если клиент платил наличными), сумма – вещественное число, дата - dd.mm.yyyy, валюта – enum (usd, eur, rub, jpy, cny), Id работника авторизовавшего транзакцию - целое неотрицательное число, статус: -1 если транзакция была отклонена, 0 если транзакция обрабатывается, 1 если транзакция выполнена
5	Работники (номер паспорта, ФИО, Дата рождения, ИНН, справка об отсутствии судимости, Id аутентификации)	администрация	ФИО строка символов(кириллица или латинские буквы), , номер паспорта целое положительное число из 10 цифр, дата рождения в формате dd.mm.yyyy, , справка - фото в base64, ИНН целое положительное число из 12 цифр, Id аутентификации - целое неотрицательное число
6	Аутентификационные данные (Логин, хеш пароля)	Клиент, оператор, администрация	логин – строка из цифр и латинских букв от 6 до 80 символов, Хеш пароля - 64 символа представляющие SHA256 хеш пароля (пароль – строка ascii символов от 10 до 80 символов)
7	Курсы валют (Валюта из которой происходит перевод, валюта в которую происходит)	Клиент, оператор, администрация	Валюта из которой происходит перевод – enum (usd, eur, rub, jpy, cny), валюта в которую происходит перевод – enum (usd, eur, rub, jpy, cny), множитель - положительное вещественное число

	перевод, множитель)		
--	------------------------	--	--

## Схема базы данных и описание таблиц.



В предлагаемой базе данных сберегательного банка присутствуют следующие сущности:

- Работники(Хранение: Id работника, номера паспорта, ФИО, даты рождения, ИНН, данных о судимостях, Id аутентификации)
- Клиенты(Хранение: Id клиента, номера паспорта, ФИО, даты рождения, ИНН, номера телефона, дохода в месяц, Id аутентификации, Id работника который добавил данные клиента)
- Услуги(Хранение: Id услуги, срока в месяцах, процентной ставки, типа услуги (кредит/вклад), условий досрочного погашения (для кредитов), условий досрочного снятия (для вкладов), валюты, необходимого дохода в месяц (для кредитов), описания, флага активности)
- Счета(Хранение: Id счета, типа счета (кредит/вклад), суммы, даты открытия, валюты, Id клиента которому принадлежит счет, сумма начисленных процентов, флага активности)

- Транзакции(Хранение: Id транзакции, счета списания, счета начисления, суммы, метки времени, валюты, Id работника который авторизировал транзакцию, статуса)
- Аутентификация (Хранение: Id аутентификации, логина, хеша пароля)
- Курсы Валют(Хранение: Валюты из которой происходит перевод, валюты в которую происходит перевод, множителя)

Существуют следующие отношения между ними:

Работники – Клиенты, Работники – Транзакции, Клиенты – Счета, Услуги – Счета, Счета – Транзакции, Аутентификация – Работники/Клиенты

## Код создания таблиц

### Таблица Clients

```
CREATE TABLE [dbo].[Clients](
    [PassportNumber] [varchar](10) NOT NULL,
    [FullName] [nvarchar](200) NOT NULL,
    [BirthDate] [date] NOT NULL,
    [TaxId] [varchar](12) NULL,
    [TelephoneNumber] [varchar](11) NULL,
    [AuthId] [int] NULL,
    [ClientId] [int] IDENTITY(1,1) NOT NULL,
    [IncomePerMonth] [money] NULL,
    [AccountCreatorId] [int] NOT NULL,
    CONSTRAINT [PK_Clients_1] PRIMARY KEY CLUSTERED
(
    [ClientId] 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
/*****CREATE UNIQUE CONSTRAINTS*****/
CREATE UNIQUE INDEX UC_Clients_PassportNumber
    ON Clients(PassportNumber)
    WHERE PassportNumber != 'DELETED'
GO
CREATE UNIQUE INDEX UC_Clients_TaxId
    ON Clients(TaxId)
    WHERE TaxId IS NOT NULL and TaxId != 'DELETED'
GO
CREATE UNIQUE INDEX UC_Clients_TelephoneNumber
    ON Clients(TelephoneNumber)
    WHERE TelephoneNumber IS NOT NULL and TelephoneNumber != 'DELETED'
GO
ALTER TABLE [dbo].[Clients] WITH CHECK ADD CONSTRAINT [FK_Clients_Auth] FOREIGN
KEY([AuthId])
REFERENCES [dbo].[Auth] ([AuthId])
GO
ALTER TABLE [dbo].[Clients] CHECK CONSTRAINT [FK_Clients_Auth]
GO
ALTER TABLE [dbo].[Clients] WITH CHECK ADD CONSTRAINT [FK_Clients_Workers] FOREIGN
KEY([AccountCreatorId])
REFERENCES [dbo].[Workers] ([WorkerId])
GO
ALTER TABLE [dbo].[Clients] CHECK CONSTRAINT [FK_Clients_Workers]
GO
```

### Таблица BankAccounts

```
CREATE TABLE [dbo].[BankAccounts](
    [BankAccountId] [int] IDENTITY(1,1) NOT NULL,
    [IsDebit] [bit] NOT NULL,
    [ServiceId] [int] NOT NULL,
    [Total] [money] NOT NULL,
    [DateOfCreation] [date] NOT NULL,
```

```

        [Currency] [varchar](3) NOT NULL,
        [ClientId] [int] NOT NULL,
        [AccumulatedInterest] [money] NULL,
        [IsClosed] [bit] NOT NULL,
    CONSTRAINT [PK_BankAccounts] PRIMARY KEY CLUSTERED
(
        [BankAccountId] 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].[BankAccounts] WITH CHECK ADD CONSTRAINT [FK_BankAccounts_Clients]
FOREIGN KEY([ClientId])
REFERENCES [dbo].[Clients] ([ClientId])
GO
ALTER TABLE [dbo].[BankAccounts] CHECK CONSTRAINT [FK_BankAccounts_Clients]
GO
ALTER TABLE [dbo].[BankAccounts] WITH CHECK ADD CONSTRAINT [FK_BankAccounts_Services]
FOREIGN KEY([ServiceId])
REFERENCES [dbo].[Services] ([ServiceId])
GO
ALTER TABLE [dbo].[BankAccounts] CHECK CONSTRAINT [FK_BankAccounts_Services]
GO

```

## Таблица Services

```

CREATE TABLE [dbo].[Services](
    [ServiceId] [int] IDENTITY(1,1) NOT NULL,
    [Months] [int] NOT NULL,
    [Interest] [float] NOT NULL,
    [IsDebit] [bit] NOT NULL,
    [LoanOverdueTerms] [nvarchar](max) NULL,
    [EarlyWithdrawalTerms] [nvarchar](max) NULL,
    [Currency] [varchar](3) NOT NULL,
    [RequiredIncome] [money] NULL,
    [Description] [nvarchar](max) NOT NULL,
    [IsDisabled] [bit] NOT NULL,
    CONSTRAINT [PK_Services] PRIMARY KEY CLUSTERED
(
        [ServiceId] 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] TEXTIMAGE_ON [PRIMARY]
GO

```

## Таблица Transactions

```

CREATE TABLE [dbo].[Transactions](
    [TransactionId] [int] IDENTITY(1,1) NOT NULL,
    [SourceAccountId] [int] NULL,
    [TransferAccountId] [int] NULL,
    [Total] [money] NOT NULL,
    [Timestamp] [datetime] NOT NULL,
    [Currency] [varchar](3) NOT NULL,

```

```

        [AuthorisedWorkerId] [int] NULL,
        [Status] [smallint] NOT NULL,
        CONSTRAINT [PK_Transactions] PRIMARY KEY CLUSTERED
    (
        [TransactionId] 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].[Transactions] WITH CHECK ADD CONSTRAINT
    [FK_Transactions_BankAccounts] FOREIGN KEY([SourceAccountId])
    REFERENCES [dbo].[BankAccounts] ([BankAccountId])
GO
ALTER TABLE [dbo].[Transactions] CHECK CONSTRAINT [FK_Transactions_BankAccounts]
GO
ALTER TABLE [dbo].[Transactions] WITH CHECK ADD CONSTRAINT
    [FK_Transactions_BankAccounts1] FOREIGN KEY([TransferAccountId])
    REFERENCES [dbo].[BankAccounts] ([BankAccountId])
GO
ALTER TABLE [dbo].[Transactions] CHECK CONSTRAINT [FK_Transactions_BankAccounts1]
GO
ALTER TABLE [dbo].[Transactions] WITH CHECK ADD CONSTRAINT [FK_Transactions_Workers]
    FOREIGN KEY([AuthorisedWorkerId])
    REFERENCES [dbo].[Workers] ([WorkerId])
GO
ALTER TABLE [dbo].[Transactions] CHECK CONSTRAINT [FK_Transactions_Workers]
GO

```

## Таблица Auth

```

CREATE TABLE [dbo].[Auth](
    [AuthId] [int] IDENTITY(1,1) NOT NULL,
    [Login] [varchar](80) NOT NULL,
    [PasswordHash] [varchar](64) NOT NULL,
    CONSTRAINT [PK_Auth] PRIMARY KEY CLUSTERED
    (
        [AuthId] 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
/*****CREATE UNIQUE CONSTRAINTS*****/
CREATE UNIQUE INDEX UC_Auth_Login
    ON Auth(Login)
GO
CREATE UNIQUE INDEX UC_Auth_PasswordHash
    ON Auth>PasswordHash)
GO

```

## Таблица Exchange

```

CREATE TABLE [dbo].[Exchange](
    [ExchangeId] [int] IDENTITY(1,1) NOT NULL,
    [From] [varchar](3) NOT NULL,

```

```

        [To] [varchar](3) NOT NULL,
        [Rate] [money] NOT NULL,
CONSTRAINT [PK_Exchange] PRIMARY KEY CLUSTERED
(
    [ExchangeId] 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

```

## Таблица Workers

```

CREATE TABLE [dbo].[Workers](
    [PassportNumber] [varchar](10) NOT NULL,
    [FullName] [nvarchar](200) NOT NULL,
    [BirthDate] [date] NOT NULL,
    [TaxId] [varchar](12) NOT NULL,
    [CriminalRecords] [nvarchar](max) NULL,
    [WorkerId] [int] IDENTITY(1,1) NOT NULL,
    [AuthId] [int] NULL,
CONSTRAINT [PK_Workers] PRIMARY KEY CLUSTERED
(
    [WorkerId] 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] TEXTIMAGE_ON [PRIMARY]
GO
/*****CREATE UNIQUE CONSTRAINTS*****/
CREATE UNIQUE INDEX UC_Workers_PassportNumber
    ON Workers(PassportNumber)
    WHERE PassportNumber != 'DELETED'
GO
CREATE UNIQUE INDEX UC_Workers_TaxId
    ON Workers(TaxId)
    WHERE TaxId != 'DELETED'
GO
ALTER TABLE [dbo].[Workers] WITH CHECK ADD CONSTRAINT [FK_Workers_Auth] FOREIGN
KEY([AuthId])
REFERENCES [dbo].[Auth] ([AuthId])
GO
ALTER TABLE [dbo].[Workers] CHECK CONSTRAINT [FK_Workers_Auth]
GO

```



## Запросы

### Представления

#### 1. Оператор, Клиент: Данные клиента

```
CREATE VIEW [dbo].[ClientData] as
SELECT * FROM Clients
GO
SELECT * FROM ClientData WHERE ClientId = 1000
```

	PassportNumber	FullName	BirthDate	TaxId	TelephoneNumber	AuthId	ClientId	IncomePerMonth	AccountCreatorId
1	3039533003	Stephen Harding	1945-05-10	NULL	89873812447	1100	1000	NULL	71

#### 2. Оператор, Клиент: Данные вкладов и кредитов клиента

```
CREATE VIEW [dbo].[AccountsData] as
SELECT * FROM BankAccounts
GO
SELECT * FROM AccountsData WHERE ClientId = 1000
```

	BankAccountId	IsDebit	ServiceId	Total	DateOfCreation	Currency	ClientId	AccumulatedInterest	IsClosed
1	1693	1	3	219700.00	2020-11-29	JPY	1000	0.00	0
2	20798	1	1	10000.00	2020-12-25	RUB	1000	0.00	0

#### 3. Оператор, Клиент, Администрация: Условия кредитов и вкладов

```
CREATE VIEW [dbo].[ServicesData] as
SELECT * FROM Services
GO
SELECT * FROM ServicesData WHERE IsDisabled = '0'
```

	ServiceId	Months	Interest	IsDebit	LoanOverdueTerms	EarlyWithdrawalTerms	Currency	RequiredIncome	Description	IsDisabled
1	2	60	0.01	1	NULL	NULL	RUB	0.00	Long Term Flexible Deposit	0
2	3	12	1.5	1	NULL	NULL	JPY	0.00	High Interest Deposit	0
3	4	60	0.01	1	NULL	NULL	JPY	0.00	Long Term Flexible Deposit	0
4	5	12	2.25	1	NULL	NULL	USD	0.00	High Interest Deposit	0
5	6	60	0.01	1	NULL	NULL	USD	0.00	Long Term Flexible Deposit	0
6	7	12	3.1	1	NULL	NULL	EUR	0.00	High Interest Deposit	0
7	8	60	0.01	1	NULL	NULL	EUR	0.00	Long Term Flexible Deposit	0

#### 4. Оператор, Клиент: Транзакции клиента

```
CREATE VIEW [dbo].[ClientTransactionsData] as
SELECT ClientId, TransactionId, SourceAccountId, TransferAccountId,
Transactions.Total, Timestamp, Transactions.Currency, AuthorisedWorkerId, Status
FROM Transactions
JOIN BankAccounts ON Transactions.TransferAccountId = BankAccounts.BankAccountId
OR Transactions.SourceAccountId = BankAccounts.BankAccountId
GO
SELECT * FROM ClientTransactionsData WHERE ClientId = 1000
```

	ClientId	TransactionId	SourceAccountId	TransferAccountId	Total	Timestamp	Currency	AuthorisedWorkerId	Status
1	1000	1693	NULL	1693	219700.00	2020-11-29 17:33:31.820	JPY	29	1

#### 5. Администрация: Транзакции за период времени

```
CREATE VIEW [dbo].[TransactionsData] as
SELECT * FROM Transactions
GO
SELECT TOP(10) * FROM TransactionsData
WHERE [Timestamp] >= '2020-11-01' AND [Timestamp] <= '2020-12-01'
```

	TransactionId	SourceAccountId	TransferAccountId	Total	Timestamp	Currency	AuthorisedWorkerId	Status
1	1	NULL	1	156600.00	2020-11-29 17:33:29.483	RUB	4	1
2	2	NULL	2	28100.00	2020-11-29 17:33:29.517	RUB	96	1
3	3	NULL	3	1032600.00	2020-11-29 17:33:29.517	CNY	61	1
4	4	NULL	4	51300.00	2020-11-29 17:33:29.520	RUB	83	1
5	5	NULL	5	148400.00	2020-11-29 17:33:29.520	RUB	82	1
6	6	NULL	6	13100.00	2020-11-29 17:33:29.520	RUB	12	1
7	7	NULL	7	681700.00	2020-11-29 17:33:29.523	RUB	41	1
8	8	NULL	8	9000.00	2020-11-29 17:33:29.523	RUB	20	1
9	9	NULL	9	27200.00	2020-11-29 17:33:29.523	RUB	7	1
10	10	NULL	10	97600.00	2020-11-29 17:33:29.523	RUB	81	1

## Добавление

### 1. Оператор: Добавление данных клиента

```
INSERT INTO Clients(PassportNumber, FullName, BirthDate, AccountCreatorId,
    TelephoneNumber, AuthId, IncomePerMonth)
    VALUES(9385739485, 'Gabriel Noel', '1997-11-01', 100, 89127395343, 1001,
    90000)
```

	PassportNumber	FullName	BirthDate	TaxId	TelephoneNumber	AuthId	ClientId	IncomePerMonth	AccountCreatorId
1	9385739485	Gabriel Noel	1997-11-01	NULL	89127395343	1001	10009	90000.00	100

### 2. Оператор: Добавление данных вклада или кредита клиента

```
INSERT INTO BankAccounts(IsDebit, ServiceId, Total, DateOfCreation, Currency,
    ClientId, AccumulatedInterest, IsClosed)
    VALUES (1, 1, 10000, '2020-12-25', 'RUB', 1000, '0', 0);
```

### 3. Администрация: Добавление возможных условий вклада или кредита

```
INSERT INTO Services(Months, Interest, IsDebit, LoanOverdueTerms,
    EarlyWithdrawalTerms, Currency, RequiredIncome, Description, IsDisabled)
    VALUES (60, 5, 1, NULL, NULL, 'RUB', 0, 'Some deposit', 0)
```

### 4. Администрация: Добавление данных оператора

```
INSERT INTO Workers(PassportNumber, FullName, BirthDate, TaxId, AuthId) VALUES
    (2847583753, 'Michael Bloom', '1980-11-24', 456634312653, 123);
```

### 5. Оператор, Клиент: Добавление транзакции

```
INSERT INTO Transactions(SourceAccountId, TransferAccountId, Total, Currency,
    Timestamp, AuthorisedWorkerId, Status)
    VALUES (1000, 1002, 3000, 'JPY', CURRENT_TIMESTAMP, 100, 0)
```

## Изменение

### 1. Администрация: Изменение данных оператора (например изменим имя, транслитерировав его в кириллицу)

```
UPDATE Workers
SET FullName = N'Майкл Блум'
WHERE WorkerId = 108
```

	PassportNumber	FullName	BirthDate	TaxId	CriminalRecords	WorkerId	AuthId
1	2847583753	Michael Bloom	1980-11-24	456634312653	NULL	108	123

	PassportNumber	FullName	BirthDate	TaxId	CriminalRecords	WorkerId	AuthId
1	2847583753	Майкл Блум	1980-11-24	456634312653	NULL	108	123

- Администрация: Изменение аутентификации оператора (хеш соответствует паролю: 123123123123 с AuthId = 123)

```
Update Auth
set Login = 'someLogin', PasswordHash =
'f28b17c428460d1ca9491000be9e03f75cc3e1af3e96c7ee9a81aefdf54b6720'
from Workers as wk
where wk.AuthId = Auth.AuthId and wk.WorkerId = 108
```

- Оператор: Изменение данных клиента (например, изменим ИНН и телефон)

```
UPDATE Clients
SET TaxId = 938482949292, TelephoneNumber = 89747383838
WHERE ClientId = 10000
```

- Оператор: Изменение аутентификации клиента (хеш соответствует паролю: 123123123123 с AuthId = 1001)

```
Update Auth
set Login = 'someClientLogin', PasswordHash =
'01779f10c74c7df1a377ff8f894bd1730109a08f4d3735300657e0ee77280120'
from Clients as cl
where cl.AuthId = Auth.AuthId and cl.ClientId = 10009
```

## Удаление

- Оператор: Закрытие/Блокировка счета клиента

```
UPDATE BankAccounts
SET IsClosed = 1
WHERE BankAccountId = 10000
```

	BankAccountId	IsDebit	ServiceId	Total	DateOfCreation	Currency	ClientId	AccumulatedInterest	IsClosed
1	10000	1	2	95100.00	2020-11-29	RUB	5901	0.00	0

	BankAccountId	IsDebit	ServiceId	Total	DateOfCreation	Currency	ClientId	AccumulatedInterest	IsClosed
1	10000	1	2	95100.00	2020-11-29	RUB	5901	0.00	1

## 2. Оператор: Удаление данных клиента

```
UPDATE Clients
SET PassportNumber = 'DELETED', FullName = 'DELETED', BirthDate='1970-01-01',
TaxId = NULL, TelephoneNumber = NULL, IncomePerMonth = NULL, AuthId = NULL
WHERE ClientId = 10009
```

	PassportNumber	FullName	BirthDate	TaxId	TelephoneNumber	AuthId	ClientId	IncomePerMonth	AccountCreatorId
1	9385739485	Gabriel Noel	1997-11-01	NULL	89127395343	1001	10009	90000.00	100

	PassportNumber	FullName	BirthDate	TaxId	TelephoneNumber	AuthId	ClientId	IncomePerMonth	AccountCreatorId
1	DELETED	DELETED	1970-01-01	NULL	NULL	NULL	10009	NULL	100

## 3. Администрация: Блокировка условий вклада или кредита

```
UPDATE Services
SET IsDisabled = 1
WHERE ServiceId = 21
```

## 4. Администрация: Удаление данных оператора (увольнение)

```
UPDATE Workers
SET PassportNumber = 'DELETED', FullName = 'DELETED', BirthDate='1970-01-01',
TaxId = 'DELETED', CriminalRecords = NULL, AuthId = NULL
WHERE WorkerId = 123
```

## Сложные запросы

1. Администрация: Вывести номера телефонов клиентов которые не совершали транзакции в течение указанного периода (чтобы отправить им смс с рекламой)

```
SELECT ClientId, TelephoneNumber, FullName
FROM Clients
WHERE TelephoneNumber IS NOT NULL AND ClientId NOT IN (
SELECT DISTINCT ClientId
FROM (
SELECT BankAccounts.ClientId, Transactions.Timestamp,
Transactions.TransferAccountId, Clients.TelephoneNumber
FROM Transactions
JOIN BankAccounts ON BankAccounts.BankAccountId =
Transactions.TransferAccountId
JOIN Clients ON BankAccounts.ClientId = Clients.ClientId
WHERE Transactions.Timestamp >= '2020-11-01' and Transactions.Timestamp <=
'2020-12-31'
) AS tmp)
```

	ClientId	TelephoneNumber	FullName
1	152	89159285410	Kari Pergola
2	698	89417747779	Ronald Bryant
3	789	89137927658	Richard Baker
4	888	89173751340	Jonathan Ratcliff
5	1178	89659821459	Dina Hollomon

2. Администрация: Вывести номера телефонов клиентов с днем рождения соответствующим указанной дате

```
SELECT Clients.ClientId, Clients.FullName, Clients.TelephoneNumber FROM Clients
WHERE DAY(Clients.BirthDate) = '25'
AND MONTH(Clients.BirthDate) = '12'
```

	ClientId	FullName	TelephoneNumber
1	63	Charisse Clark	89862200348
2	377	Jeffery Marotta	89245542581
3	1003	Hailey Biderman	89492857336
4	1122	Larry Fields	89247162591
5	1778	Jeffrey Caswell	89479040237

3. Администрация: Вывести транзакции больше 600 тысяч рублей (для анализа службой безопасности в соответствии с 115-ФЗ)

```
SELECT Transactions.TransactionId, Transactions.Total, Transactions.Currency,
Transactions.Timestamp, Transactions.Status FROM Transactions
WHERE Transactions.Total > 600000 and Transactions.Currency = 'RUB' and
Transactions.Timestamp >= '2020-11-01' and Transactions.Timestamp <=
'2020-12-31'
```

	TransactionId	Total	Currency	Timestamp	Status
1	7	681700.00	RUB	2020-11-29 17:33:29.523	1
2	25	1274400.00	RUB	2020-11-29 17:33:29.537	1
3	57	699900.00	RUB	2020-11-29 17:33:29.563	1
4	69	686700.00	RUB	2020-11-29 17:33:29.580	1
5	88	615500.00	RUB	2020-11-29 17:33:29.630	1

4. Администрация: Вывести список операторов в соответствии с количеством авторизованных транзакций за указанный период времени

```
SELECT Transactions.AuthorisedWorkerId, Workers.FullName, Count(*) AS 'Count'
FROM Transactions
JOIN Workers ON Transactions.AuthorisedWorkerId = Workers.WorkerId
WHERE Transactions.Timestamp >= '2020-11-01' and Transactions.Timestamp <=
'2020-12-31'
GROUP BY AuthorisedWorkerId, FullName
ORDER BY 'Count' desc
```



	AuthorisedWorkerId	FullName	Count
1	64	Andrea Renfrew	254
2	18	Ruth Ellis	242
3	50	Pauline Palomaki	239
4	24	Margaret Hinkle	237
5	29	Esteban Pritchard	237

5. Администрация: Вывести общую прибыль с кредитов для каждой валюты

```

SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) AS 'Sum'
FROM BankAccounts
WHERE BankAccounts.IsDebit = 'False' and Currency = 'RUB'
GROUP BY BankAccounts.Currency
UNION
SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) FROM
BankAccounts
WHERE BankAccounts.IsDebit = 'False' and Currency = 'USD'
GROUP BY BankAccounts.Currency
UNION
SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) FROM
BankAccounts
WHERE BankAccounts.IsDebit = 'False' and Currency = 'EUR'
GROUP BY BankAccounts.Currency
UNION
SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) FROM
BankAccounts
WHERE BankAccounts.IsDebit = 'False' and Currency = 'JPY'
GROUP BY BankAccounts.Currency
UNION
SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) FROM
BankAccounts
WHERE BankAccounts.IsDebit = 'False' and Currency = 'CNY'
GROUP BY BankAccounts.Currency

```

	Currency	Sum
1	RUB	7710996432.209
2	USD	1387330.108
3	EUR	793581.474
4	JPY	88397630.722
5	CNY	21130593.762

6. Администрация: Вывести общую суммы выплат процентов по вкладам для каждой валюты

```

SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) AS 'Sum'
FROM BankAccounts
WHERE BankAccounts.IsDebit = 'True' and Currency = 'RUB'
GROUP BY BankAccounts.Currency
UNION

```

```

SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) FROM
BankAccounts
WHERE BankAccounts.IsDebit = 'True' and Currency = 'USD'
GROUP BY BankAccounts.Currency
UNION
SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) FROM
BankAccounts
WHERE BankAccounts.IsDebit = 'True' and Currency = 'EUR'
GROUP BY BankAccounts.Currency
UNION
SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) FROM
BankAccounts
WHERE BankAccounts.IsDebit = 'True' and Currency = 'JPY'
GROUP BY BankAccounts.Currency
UNION
SELECT BankAccounts.Currency, SUM(BankAccounts.AccumulatedInterest) FROM
BankAccounts
WHERE BankAccounts.IsDebit = 'True' and Currency = 'CNY'
GROUP BY BankAccounts.Currency

```

	Currency	Sum
1	RUB	0.00
2	USD	0.00
3	EUR	0.00
4	JPY	0.00
5	CNY	0.00

7. Администрация: Вывести список условий вкладов и кредитов в соответствии с количеством открытых с этими условиями счетов

```

SELECT tmp.Count, tmp.ServiceId, Services.Interest, Services.IsDebit,
Services.Months, Services.RequiredIncome, Services.Currency
FROM (
SELECT BankAccounts.ServiceId , COUNT(*) as 'Count'
FROM BankAccounts
GROUP BY BankAccounts.ServiceId
) AS tmp
JOIN Services ON Services.ServiceId = tmp.ServiceId
ORDER BY tmp.Count DESC

```

	Count	ServiceId	Interest	IsDebit	Months	RequiredIncome	Currency
1	8997	1	8.4	1	12	0.00	RUB
2	4386	2	0.01	1	60	0.00	RUB
3	2090	11	10.2	0	60	80000.00	RUB
4	1274	12	9.66	0	240	50000.00	RUB
5	600	3	1.5	1	12	0.00	JPY

8. Администрация: Вывести общий денежный оборот транзакций с определенной валютой (CNY) за указанный период времени.

```

SELECT Transactions.Currency, SUM(Transactions.Total) AS 'Sum' FROM Transactions

```

```
WHERE Transactions.Currency = 'CNY' AND Transactions.Status = 1 AND
Transactions.Timestamp >= '2020-11-01' AND Transactions.Timestamp <=
'2020-12-31'
GROUP BY Transactions.Currency
```

	Currency	Sum
1	CNY	227257301.00

9. Администрация: Вывести список клиентов с самыми большими накоплениями в указанной валюте (EUR)

```
SELECT Clients.FullName, Clients.ClientId, SUM(BankAccounts.Total) as 'Sum' from
BankAccounts
JOIN Clients on BankAccounts.ClientId = Clients.ClientId
WHERE BankAccounts.IsDebit = 'True' and BankAccounts.Currency = 'EUR'
GROUP BY Clients.FullName, Clients.ClientId
ORDER BY 'Sum' DESC
```

	FullName	ClientId	Sum
1	Peter Ray	3649	4192600.00
2	Sheila Larson	4094	2747000.00
3	Cecile Alber	3374	2739100.00
4	Delores Jones	2043	2723000.00
5	Stephen Mattern	7078	2106700.00

10. Администрация: Вывести список клиентов с самым большим количеством долгов в указанной валюте (JPY)

```
SELECT Clients.FullName, Clients.ClientId, SUM(BankAccounts.Total) as 'Sum'
from BankAccounts
JOIN Clients on BankAccounts.ClientId = Clients.ClientId
WHERE BankAccounts.IsDebit = 'False' and BankAccounts.Currency = 'JPY'
GROUP BY Clients.FullName, Clients.ClientId
ORDER BY 'Sum' DESC
```

	FullName	ClientId	Sum
1	Terri Tomasino	8070	16832376.768
2	Victor Rodregez	5339	16303511.40
3	William Rogers	2090	15852010.032
4	Jessica Oleary	5622	15023834.76
5	Chad Hastings	7867	13708646.808



## Хранимые процедуры

### 1. Добавить данные клиента

Принимает номер паспорта, фиио, дату рождения, id работника добавившего клиента, а также необязательные поля: инн, номер телефона, доход в месяц. Также хп добавляет запись с временными логином и паролем в таблицу аутентификации (просто чтобы закрепить какой-то authId за клиентом)

```
CREATE PROCEDURE [dbo].[Add_Client](@PassportNumber AS VARCHAR(10), @FullName AS NVARCHAR(200), @BirthDate AS DATE, @AccountCreatorId AS INT, @TaxId VARCHAR(12) = NULL, @TelephoneNumber VARCHAR(11) = NULL, @IncomePerMonth MONEY = NULL) AS
    IF (@IncomePerMonth != NULL AND @IncomePerMonth < 0)
    BEGIN
        RAISERROR('ERROR: Income can not be negative',15,1);
        RETURN;
    END;
    ELSE
    BEGIN
        Declare @PromiseLogin as varchar(64)
        Declare @PromiseHash as varchar(64)
        Set @PromiseLogin = (SELECT
CONVERT(varchar(64),LEFT(REPLACE(NEWID(),'-',''),64)))--Random data
        Set @PromiseHash = (SELECT
CONVERT(varchar(64),LEFT(REPLACE(NEWID(),'-',''),64)))--Random data
        insert into Auth(Login,PasswordHash) Values (@PromiseLogin, @PromiseHash);
        Declare @AuthId as int
        set @AuthId = SCOPE_IDENTITY()
        INSERT INTO Clients(PassportNumber,FullName,BirthDate, AccountCreatorId,
TaxId, TelephoneNumber, AuthId, IncomePerMonth)
        VALUES (@PassportNumber, @FullName, @BirthDate, @AccountCreatorId, @TaxId,
@TelephoneNumber, @AuthId, @IncomePerMonth);
    END;
GO
EXEC Add_Client 2342367856, N'Дональд Чэмбэрлин', '12-21-1944', 100, NULL, 18552700615
```

	PassportNumber	FullName	BirthDate	TaxId	TelephoneNumber	AuthId	ClientId	IncomePerMonth	AccountCreatorId
1	2342367856	Дональд Чэмбэрлин	1944-12-21	NULL	18552700615	10118	10011	NULL	100

### 2. Добавить данные работника

Принимает номер паспорта, фиио, дату рождения, инн, логин и пароль

```
CREATE PROCEDURE [dbo].[Add_Worker](@PassportNumber AS VARCHAR(10), @FullName AS NVARCHAR(200), @BirthDate AS DATE, @TaxId as varchar(12), @Login as varchar(80), @Password as varchar(80)) AS
    --In case of an error, rollback will be issued automatically.
    set xact_abort on
    begin transaction

    Declare @AuthId as int
    Declare @WorkerId as int
    insert into Auth(Login,PasswordHash) Values (@Login, 'promise'); --because of a circular dependency
```

```

set @AuthId = SCOPE_IDENTITY()
INSERT INTO Workers(PassportNumber,FullName,BirthDate,TaxId, AuthId) VALUES
(@PassportNumber, @FullName, @BirthDate, @TaxId, @AuthId);
set @WorkerId = SCOPE_IDENTITY()
--we still need to store real password's hash
Update Auth set PasswordHash = CONVERT(varchar(64), HASHBYTES('SHA2_256',
@Password + '' + CONVERT(varchar(10), @AuthId)),2) --salted hash
from Workers as wk
where wk.AuthId = Auth.AuthId and wk.WorkerId = @WorkerId
--the real password's hash is stored, promise is fulfilled, circular dependency is
resolved. And everyone lived happily ever after...

```

```
commit
```

```
GO
```

```
EXEC Add_Worker 8374629573, N'Датaбейсов Баздан', '11-11-1991', 123456567234, 'DBboy',
'databasesbasesbases'
```

	PassportNumber	FullName	BirthDate	TaxId	CriminalRecords	WorkerId	AuthId
1	8374629573	Датaбейсов Баздан	1991-11-11	123456567234	NULL	110	10116

	AuthId	Login	PasswordHash
1	10116	DBboy	DAB6B70C86BE19C646D6CA55E4A31877250D7F389804D27C7A3E81AFDA156280

### 3. Открыть счет(выдать кредит)

Принимает Id услуги, Сумму, Id клиента и Id работника, который открывает счет. Проверяет, существует ли услуга, правильно ли указана сумма, достаточен ли доход клиента(для кредита).

```

CREATE PROCEDURE [dbo].[Add_BankAccount](@ServiceId AS INT, @Total AS MONEY,
@ClientId AS INT, @AuthorisedWorkerId AS INT)
AS
    IF NOT EXISTS(SELECT * FROM Services WHERE ServiceId = @ServiceId and IsDisabled =
0 )
        BEGIN
            RAISERROR('ERROR: Illegal Service Id',15,1);
            RETURN;
        END;
    IF (@Total < 0)
        BEGIN
            RAISERROR('ERROR: Total can not be negative',15,1);
            RETURN;
        END;
    IF NOT EXISTS (SELECT * FROM Workers WHERE WorkerId = @AuthorisedWorkerId)
        BEGIN
            RAISERROR('ERROR: The is no worker with this
AuthorisedWorkerId',15,1);
            RETURN;
        END;
    --Getting some additional information
    DECLARE @Months AS INT
    SET @Months = (SELECT Months FROM [dbo].Services WHERE ServiceId = @ServiceId)
    DECLARE @Interest AS FLOAT
    SET @Interest = (SELECT Interest FROM [dbo].Services WHERE ServiceId = @ServiceId)
    DECLARE @IsDebit AS BIT

```

```

SET @IsDebit = (SELECT IsDebit FROM [dbo].Services WHERE ServiceId = @ServiceId)
DECLARE @Currency AS VARCHAR(3)
SET @Currency = (SELECT Currency FROM [dbo].Services WHERE ServiceId = @ServiceId)
DECLARE @CurrentDate AS DATE
SET @CurrentDate = CAST( GETDATE() AS DATE );

IF (@IsDebit = 1) --Adding a Saving Account/Deposit
BEGIN
    INSERT INTO BankAccounts(IsDebit, ServiceId, Total, DateOfCreation,
Currency, ClientId, AccumulatedInterest, IsClosed)
    VALUES (@IsDebit, @ServiceId, @Total, @CurrentDate, @Currency, @ClientId,
'0', 0);
    INSERT INTO Transactions(TransferAccountId, Total, Timestamp, Currency,
AuthorisedWorkerId, Status)
    VALUES (SCOPE_IDENTITY(), @Total, CURRENT_TIMESTAMP, @Currency,
@AuthorisedWorkerId, '1');
    RETURN;
END;
ELSE --Adding a loan
BEGIN
    DECLARE @IncomePerMonth AS MONEY
    SET @IncomePerMonth = (SELECT IncomePerMonth FROM [dbo].Clients WHERE
ClientId = @ClientId)
    DECLARE @RequiredIncome AS MONEY
    SET @RequiredIncome = (SELECT RequiredIncome FROM [dbo].Services WHERE
ServiceId = @ServiceId)
    DECLARE @MonthlyPayment AS MONEY
    EXECUTE Get_LoanPaymentPerMonth @Total, @Interest, @Months, @MonthlyPayment
OUTPUT
    IF (@IncomePerMonth IS NULL)
    BEGIN
        RAISERROR('ERROR: This client did not provide income per month
information',15,1);
        RETURN;
    END;
    --Converting to RUB if necessary
    IF (@Currency != 'RUB')
    BEGIN
        DECLARE @Rate AS MONEY
        SET @Rate = (SELECT Rate FROM [dbo].Exchange WHERE [From] =
@Currency AND [To] = 'RUB')
        SET @RequiredIncome *= @Rate
        SET @MonthlyPayment *= @Rate
    END;
    --Cheking if Client has enough income per month
    IF (@IncomePerMonth < @RequiredIncome)
    BEGIN
        RAISERROR('ERROR: Income per month of this client is insufficient
for this loan',15,1);
        RETURN;
    END;
    --Checking if total is lower than loan cap
    IF (@MonthlyPayment > @IncomePerMonth)
    BEGIN
        RAISERROR('ERROR: The calculated monthly payment is larger than
income per month of this client',15,1);
    END;

```

```

        RETURN;
    END;
    EXECUTE Get_LoanPaymentPerMonth @Total, @Interest, @Months,
@MonthlyPayment OUTPUT
    DECLARE @AccumulatedInterest AS MONEY
    SET @AccumulatedInterest = @MonthlyPayment*@Months-@Total
    INSERT INTO BankAccounts(IsDebit, ServiceId, Total, DateOfCreation,
Currency, ClientId, AccumulatedInterest, IsClosed)
    VALUES (@IsDebit, @ServiceId, @AccumulatedInterest+@Total,
@CurrentDate, @Currency, @ClientId, @AccumulatedInterest, 0);
    INSERT INTO Transactions(SourceAccountId, Total, Timestamp,
Currency, AuthorisedWorkerId, Status)
    VALUES (SCOPE_IDENTITY(), @Total, CURRENT_TIMESTAMP, @Currency,
@AuthorisedWorkerId, '1');
    END;

```

GO

EXEC Add\_BankAccount 1000,-1000,10009,999

```

Msg 50000, Level 15, State 1, Procedure Add_BankAccount, Line 7 [Batch Start Line 166]
ERROR: Illegal Service Id

```

EXEC Add\_BankAccount 2,-1000,10009,999

```

Msg 50000, Level 15, State 1, Procedure Add_BankAccount, Line 12 [Batch Start Line 166]
ERROR: Total can not be negative

```

EXEC Add\_BankAccount 2,1000,10009,999

```

Msg 50000, Level 15, State 1, Procedure Add_BankAccount, Line 17 [Batch Start Line 166]
ERROR: The is no worker with this AuthorisedWorkerId

```

EXEC Add\_BankAccount 2,1000,10009,99

	BankAccountId	IsDebit	ServiceId	Total	DateOfCreation	Currency	ClientId	AccumulatedInterest	IsClosed
1	20799	1	2	1000.00	2020-12-27	RUB	10009	0.00	0

#### 4. Рассчитать ежемесячную выплату за кредит (формула амортизированного платежа)

```

CREATE PROCEDURE [dbo].[Get_LoanPaymentPerMonth] (@Total AS MONEY, @Interest AS FLOAT,
@Months AS INT, @MonthlyPayment MONEY OUTPUT)
AS
BEGIN
    SELECT @MonthlyPayment = @Total*(@Interest/1200)*POWER(1+@Interest/1200,
@Months)/(POWER(1+@Interest/1200, @Months)-1)
END
GO

```

DECLARE @Payment as Money

EXEC Get\_LoanPaymentPerMonth 1000000, 6.3, 60, @Payment OUTPUT

SELECT @Payment as 'Monthly Payment'

	Monthly Payment
1	19472.6049

## Триггеры

### 1. Исполнение транзакции.

После создания новой записи в таблице транзакций, этот триггер проверяет, что: указанная валюта правильна, сумма это неотрицательное число, мы не указали кредит как счет списания, счет не закрыт, на счете списания достаточно средств. Если все проверки прошли, то триггер изменяет суммы счетов списания и начисления и устанавливает статус транзакции равным 1, в противном случае статус устанавливается как -1 и выводится ошибка.

```
CREATE TRIGGER [dbo].[Account_Transfer]
ON [dbo].[Transactions]
for insert
AS
BEGIN
    IF((select Status from inserted) != 0)
    BEGIN
        return
    END

    DECLARE @IsIncorrect AS BIT
    SET @IsIncorrect = 0

    IF EXISTS(SELECT Currency FROM inserted where Currency != 'RUB' and Currency !=
'JPY' and Currency != 'USD' and Currency != 'EUR' and Currency != 'CNY' and Status = 0)
    BEGIN
        SET @IsIncorrect = 1 -- We can't have random currencies
    END
    IF EXISTS(SELECT Total FROM inserted where Total <= 0 and Status = 0)
    BEGIN
        SET @IsIncorrect = 1 -- We can only have positive transactions
    END
    IF EXISTS(SELECT SourceAccountId FROM inserted where (SELECT IsDebit FROM
BankAccounts where BankAccountId = SourceAccountId) = 0 and Status = 0)
    BEGIN
        SET @IsIncorrect = 1 -- Taking money from Loan's Total is a no no
    END
    IF EXISTS(SELECT SourceAccountId FROM inserted where (SELECT IsClosed FROM
BankAccounts where BankAccountId = SourceAccountId) = 1 and Status = 0)
    BEGIN
        SET @IsIncorrect = 1 -- Account is closed, so no transacitons
    END
    IF EXISTS(SELECT TransferAccountId FROM inserted where (SELECT IsClosed FROM
BankAccounts where BankAccountId = TransferAccountId) = 1 and Status = 0)
    BEGIN
        SET @IsIncorrect = 1 -- Account is closed, so no transacitons
    END
    IF (@IsIncorrect = 1)
    BEGIN
        Update Transactions set Transactions.Status = -1
        where Transactions.TransactionId = (select TransactionId from inserted)
        RAISERROR('WARNING: INCORRECT TRANSACTION HAS BEEN REJECTED',15,1);
        return
    END

    DECLARE @SourceRate AS MONEY
    SET @SourceRate = 1
    DECLARE @TransferRate AS MONEY
```

```

SET @TransferRate = 1
DECLARE @Multiplier AS MONEY -- -1 For Loans, 1 for Deposits
SET @Multiplier = 1
DECLARE @SourceCurrency AS VARCHAR(3) --Currency of Source Bank Account
SET @SourceCurrency = (select Currency from BankAccounts where BankAccountId =
(select SourceAccountId from inserted))
DECLARE @TransferCurrency AS VARCHAR(3) --Currency of Transfer Bank Account
SET @TransferCurrency = (select Currency from BankAccounts where BankAccountId =
(select TransferAccountId from inserted))
DECLARE @TransactionCurrency AS VARCHAR(3) --Currency of Transaction
SET @TransactionCurrency = (select Currency from inserted)

IF (@SourceCurrency != @TransactionCurrency)
BEGIN
    SET @SourceRate = (SELECT Rate FROM [dbo].Exchange WHERE [From] =
@TransactionCurrency AND [To] = @SourceCurrency)
    END;
    IF (@TransferCurrency != @TransactionCurrency)
    BEGIN
        SET @TransferRate = (SELECT Rate FROM [dbo].Exchange WHERE [From] =
@TransactionCurrency AND [To] = @TransferCurrency)
        END;
        IF ((select IsDebit from BankAccounts where BankAccountId = (select
TransferAccountId from inserted)) = 0)
        BEGIN
            SET @Multiplier = -1
            END;

            IF ((SELECT Total FROM inserted)*@SourceRate > (SELECT Total FROM BankAccounts
where BankAccountId = (select SourceAccountId from inserted)))
            BEGIN
                Update Transactions set Transactions.Status = -1
                where Transactions.TransactionId = (select TransactionId from inserted) --
Insufficient funds
                RAISERROR('WARNING: INCORRECT TRANSACTION HAS BEEN REJECTED',15,1);
                return
            END

            --In case of an error, rollback will be issued automatically.
            set xact_abort on
            begin transaction
                Update BankAccounts set BankAccounts.Total += ((select Total from
inserted)*@TransferRate*@Multiplier)
                where BankAccounts.BankAccountId = (select TransferAccountId from
inserted)
                Update BankAccounts set BankAccounts.Total -= ((select Total from
inserted)*@SourceRate)
                where BankAccounts.BankAccountId = (select SourceAccountId from inserted)
                Update Transactions set Transactions.Status = 1
                where Transactions.TransactionId = (select TransactionId from inserted)
            commit
        END;
        GO
        ALTER TABLE [dbo].[Transactions] ENABLE TRIGGER [Account_Transfer]
        GO
        INSERT INTO Transactions (SourceAccountId, TransferAccountId, Total, Timestamp,
Currency, AuthorisedWorkerId, Status)
        VALUES(1,2,1000,CURRENT_TIMESTAMP, 'BAD', 12,0)

```

Msg 50000, Level 15, State 1, Procedure Account\_Transfer, Line 39 [Batch Start Line 173]  
WARNING: INCORRECT TRANSACTION HAS BEEN REJECTED

	TransactionId	SourceAccountId	TransferAccountId	Total	Timestamp	Currency	AuthorisedWorkerId	Status
1	20838	1	2	1000.00	2020-12-27 08:19:27.287	BAD	12	-1

```
INSERT INTO Transactions (SourceAccountId, TransferAccountId, Total, Timestamp,
Currency, AuthorisedWorkerId, Status)
VALUES(1,2,1000,CURRENT_TIMESTAMP, 'RUB', 12,0)
```

	BankAccountId	IsDebit	ServiceId	Total	DateOfCreation	Currency	ClientId	AccumulatedInterest	IsClosed
1	1	1	2	79917.03	2020-11-29	RUB	1	0.00	0
2	2	1	2	104788.97	2020-11-29	RUB	1	0.00	0

	BankAccountId	IsDebit	ServiceId	Total	DateOfCreation	Currency	ClientId	AccumulatedInterest	IsClosed
1	1	1	2	78917.03	2020-11-29	RUB	1	0.00	0
2	2	1	2	105788.97	2020-11-29	RUB	1	0.00	0

	TransactionId	SourceAccountId	TransferAccountId	Total	Timestamp	Currency	AuthorisedWorkerId	Status
1	20840	1	2	1000.00	2020-12-27 08:21:33.333	RUB	12	1

## 2. Отмена подозрительной транзакции

Если сумма исполненной транзакции превышает 600000 рублей (сумма транзакций в других валютах переводится в рубли) и клиент совершил её сам (через онлайн банк), то средства возвращаются с аккаунта начисления на аккаунт списания, статус транзакции ставится равным -1 и выводится ошибка.

```
CREATE TRIGGER [dbo].[Reject_Suspicious]
ON [dbo].[Transactions]
after update
AS
BEGIN
    DECLARE @IsSus AS BIT
    SET @IsSus = 0
    DECLARE @Threshold AS MONEY --RUB
    SET @Threshold = 600000
    IF EXISTS(SELECT Total FROM Transactions WHERE Currency = 'RUB'
        AND Total > @Threshold and AuthorisedWorkerId is null and Status = 1)
    BEGIN
        SET @IsSus = 1
    END;
    IF EXISTS (SELECT Total FROM Transactions WHERE Currency = 'JPY'
        AND Total*(SELECT Rate FROM [dbo].Exchange WHERE [From] = 'JPY' AND [To] =
'RUB') > @Threshold
        AND AuthorisedWorkerId IS NULL AND Status = 1)
    BEGIN
        SET @IsSus = 1
    END;
    IF EXISTS (SELECT Total FROM Transactions WHERE Currency = 'USD'
        AND Total*(SELECT Rate FROM [dbo].Exchange WHERE [From] = 'USD' AND [To] =
'RUB') > @Threshold
        AND AuthorisedWorkerId IS NULL AND Status = 1)
    BEGIN
        SET @IsSus = 1
    END;
    IF EXISTS (SELECT Total FROM Transactions WHERE Currency = 'EUR'
        AND Total*(SELECT Rate FROM [dbo].Exchange WHERE [From] = 'EUR' AND [To] =
'RUB') > @Threshold
        AND AuthorisedWorkerId IS NULL AND Status = 1)
```



```

BEGIN
    SET @IsSus = 1
END;
IF EXISTS (SELECT Total FROM Transactions WHERE Currency = 'CNY'
           AND Total*(SELECT Rate FROM [dbo].Exchange WHERE [From] = 'CNY' AND [To] =
'RUB') > @Threshold
           AND AuthorisedWorkerId IS NULL AND Status = 1)
BEGIN
    SET @IsSus = 1
END;

IF (@IsSus = 1)
BEGIN
    DECLARE @SourceRate AS MONEY
    SET @SourceRate = 1
    DECLARE @TransferRate AS MONEY
    SET @TransferRate = 1
    DECLARE @Multiplier AS MONEY -- -1 For Loans, 1 for Deposits
    SET @Multiplier = 1
    DECLARE @SourceCurrency AS VARCHAR(3) --Currency of Source Bank Account
    SET @SourceCurrency = (select Currency from BankAccounts where
BankAccountId = (select SourceAccountId from inserted))
    DECLARE @TransferCurrency AS VARCHAR(3) --Currency of Transfer Bank Account
    SET @TransferCurrency = (select Currency from BankAccounts where
BankAccountId = (select TransferAccountId from inserted))
    DECLARE @TransactionCurrency AS VARCHAR(3) --Currency of Transaction
    SET @TransactionCurrency = (select Currency from inserted)

    IF (@SourceCurrency != @TransactionCurrency)
    BEGIN
        SET @SourceRate = (SELECT Rate FROM [dbo].Exchange WHERE [From] =
@TransactionCurrency AND [To] = @SourceCurrency)
    END;
    IF (@TransferCurrency != @TransactionCurrency)
    BEGIN
        SET @TransferRate = (SELECT Rate FROM [dbo].Exchange WHERE [From] =
@TransactionCurrency AND [To] = @TransferCurrency)
    END;
    IF ((select IsDebit from BankAccounts where BankAccountId = (select
TransferAccountId from inserted)) = 0)
    BEGIN
        SET @Multiplier = -1
    END;

    RAISERROR('WARNING: SUSCICIOUS TRANSACTION HAS BEEN IDENTIFIED',15,1);
    --In case of an error, rollback will be issued automatically.
    set xact_abort on
    begin transaction
        Update BankAccounts set BankAccounts.Total -= ((select Total from
inserted)*@TransferRate*@Multiplier)
        where BankAccounts.BankAccountId = (select TransferAccountId from
inserted)
        Update BankAccounts set BankAccounts.Total += ((select Total from
inserted)*@SourceRate)

```



```

        where BankAccounts.BankAccountId = (select SourceAccountId from inserted)
        Update Transactions set Transactions.Status = -1
        where Transactions.TransactionId = (select TransactionId from inserted)
    commit
    RAISERROR('ATTENTION: SUSCICIOUS TRANSACTION HAS BEEN REVERSED',15,1);
END;
END;
GO
ALTER TABLE [dbo].[Transactions] ENABLE TRIGGER [Reject_Suspicious]
GO
INSERT INTO Transactions (SourceAccountId, TransferAccountId, Total, Timestamp,
Currency, Status)
VALUES(1,2,700000,CURRENT_TIMESTAMP, 'RUB',0)

(1 row affected)

(1 row affected)
Msg 50000, Level 15, State 1, Procedure Reject_Suspicious, Line 70 [Batch Start Line 174]
WARNING: SUSCICIOUS TRANSACTION HAS BEEN IDENTIFIED

(1 row affected)

(1 row affected)

(1 row affected)
Msg 50000, Level 15, State 1, Procedure Reject_Suspicious, Line 81 [Batch Start Line 174]
ATTENTION: SUSCICIOUS TRANSACTION HAS BEEN REVERSED

(1 row affected)

(1 row affected)

```

## Клиентская часть

В данной БД реализовано три основных пользователя: Администратор, Работник, Клиент, каждый со своим функционалом.

Клиентская часть написана как single page web application с использованием React JS и не имеет прямого доступа к базе данных из соображений безопасности. Связь с базой данных происходит через REST API, который абстрагирует все запросы к бд, get запросами к нему.

## Аутентификация

Войти как:

Администратор

Логин

Пароль

ВОЙТИ

ЗАБЫЛ ПАРОЛЬ

ЗАРЕГИСТРИРОВАТЬСЯ

В данном окне пользователь вводит тип своего аккаунта, логин и пароль, пароль скрывается значками \* при вводе символов в окно. Используя тип аккаунта, логин и пароль в клиентском приложении генерируется salted hash используя алгоритм sha256, после чего API проверяет его правильность, и в случае ошибки, показывает её пользователю

❗ Неверный пароль

Войти как:

Оператор

Логин

mildMussel982

Пароль

.....

ВОЙТИ

ЗАБЫЛ ПАРОЛЬ

ЗАРЕГИСТРИРОВАТЬСЯ

Обзор

ОбзорРаботникиУправление УслугамиУслугиТранзакции

Выйти

Начало периода  
01/01/2020

Конеч периода  
12/27/2020

День рождения  
12/27

Количество  
10

Валюта:  
RUB

ОБНОВИТЬ

Общий оборот транзакций с валютой RUB за указанный период времени : 8939595569

Транзакции больше 600тыс. рублей за этот период:

Сумма начисленных процентов:

ID Транзакции	Сумма	Статус	Метка Времени	Валюта	По кредитам	По вкладам
7	681700	исполнено	29.11.2020, 17:33:29	RUB	7710996432.209	0
25	1274400	исполнено	29.11.2020, 17:33:29	JPY	1387330.108	0
57	699900	исполнено	29.11.2020, 17:33:29	USD	793581.474	0
69	686700	исполнено	29.11.2020, 17:33:29	EUR	88397630.722	0
88	615500	исполнено	29.11.2020, 17:33:29	CNY	21130593.762	0

1-5 of 10

1-5 of 5

Топ операторов по транзакциям:

Популярные услуги::

ID Операто...	ФИО	Количество	ID Услуги	Количество	Тип	Валюта	Ставка	Период	Необх. доход
18	Ruth Ellis	242	1	8997	вклад	RUB	8.4	12 мес.	—
20	Zachary Ro...	227	2	4386	вклад	RUB	0.01	60 мес.	—
24	Margaret H...	237	3	600	вклад	JPY	1.5	12 мес.	—

Не совершали транзакций в этот период:

День рождения 12/27:

ID	ФИО	Телефон	ID	ФИО	Телефон
152	Kari Pergola	89159285410	262	Brandon Grisham	89712399022
698	Ronald Bryant	89417747779	1240	Elisa Han	89295680742
789	Richard Baker	89137927658	2741	Krista Wheeler	89135047996
888	Jonathan Ratcliff	89173751340	3243	Kevin Ridgeway	89451463609
1178	Dina Hollomon	89659821459	3704	Marvin Reiter	89843891950

1-5 of 10

1-5 of 10

Топ клиентов по накоплениям в валюте: RUB

Топ клиентов по долгам в валюте: RUB

ID	ФИО	Сумма	ID	ФИО	Сумма
1516	Eloise Hawk	4586800	240	Priscilla Meagher	34422396.288
1836	Zenaida Davis	3712500	1453	Elizabeth Link	36628090.032
2994	Richard Penton	4066100	2206	Randall Nelson	38799850.056
3169	Sarah Aldridge	4434500	4052	John Mcguire	43437054.792
3935	Judith Vargas	3687400	4714	William Walton	34732324.512

1-5 of 10

1-5 of 10

В данной вкладке администратор может получить общие данные о состоянии банковской системы, здесь используются все сложные запросы. Панель настроек позволяет изменить период за который выводится информация и количество выводимых записей каждого типа. А также выбрать валюту и дату для которой выводится список клиентов которые родились в неё.

- **Работники**

В данной вкладке администратор может добавить работника:

Обзор

Работники

Управление Услугами

Услуги

Транзакции

ВЫЙТИ

ID Работника

ФИО

НАЙТИ

Нет данных 🐾

Управление персоналом:

Выбранный ID Работника

ОБНОВИТЬ ДАННЫЕ

ОБНОВИТЬ ДОСТУП

УВОЛИТЬ

❗ Введите ФИО

Номер паспорта  
3453456756

ФИО

День Рождения  
12/27/1968

ИНН  
357745011459

Логин  
Ivanov68

Пароль  
\*\*\*\*\*

ДОБАВИТЬ

После исправления ошибок и нажатия кнопки добавить форма очищается

Успех 🎉

Номер паспорта

ФИО

День Рождения  
12/27/2020

ИНН

Логин

Пароль

ДОБАВИТЬ

Также можно найти работника по Id или ФИО и изменить его данные, сменить его логин или пароль, либо уволить его

ID Работника

23

ФИО

НАЙТИ

Найденные работники:

ID	ФИО	Номер Паспорта
23	Joan Strange	4538806921

1 row selected1-1 of 1<>

## Управление персоналом:

Выбранный ID Работника

23

ОБНОВИТЬ ДАННЫЕ

ОБНОВИТЬ ДОСТУП

УВОЛИТЬ

После нажатия на кнопку “ОБНОВИТЬ ДАННЫЕ” форма справа меняется на эту:

Изменение данных работника

Id работника

23

Номер паспорта

4538806921

ФИО

Joan Strange

День Рождения

03/09/1973

ИНН

251719974626

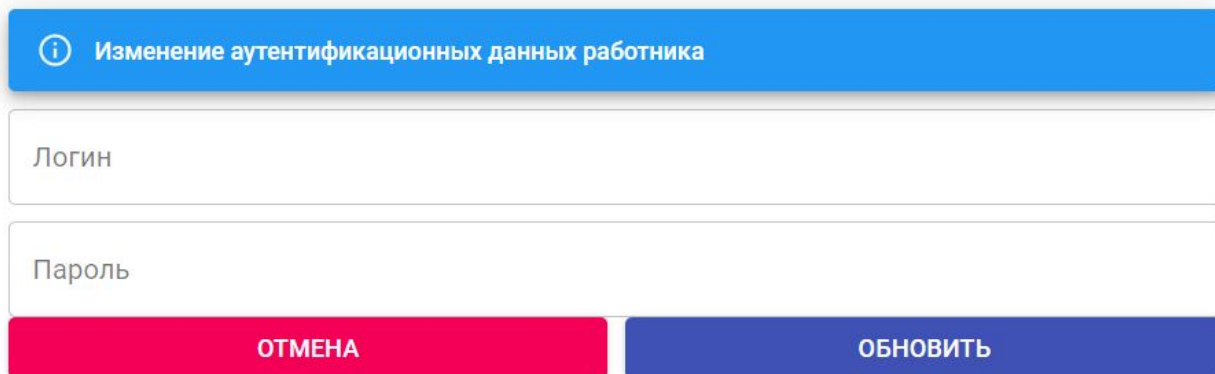
Судимости

null

ОТМЕНА

ОБНОВИТЬ

После нажатия на кнопку “ОБНОВИТЬ ДОСТУП” форма справа меняется на эту:



The form is titled "Изменение аутентификационных данных работника" (Change employee authentication data) in a blue header bar. It contains two input fields: "Логин" (Login) and "Пароль" (Password). At the bottom, there are two buttons: "ОТМЕНА" (Cancel) in a red bar and "ОБНОВИТЬ" (Update) in a blue bar.

После нажатия на кнопку “УВОЛИТЬ” личные данные работника удаляются и выводится сообщение:



A white dialog box with a light gray border. The title bar text is "localhost:3000 の内容". The main text reads "Работник: Joan Strange с Id:23 уволен!" (Employee: Joan Strange with Id:23 is dismissed!). There is a blue "ОК" button in the bottom right corner.

- **Управление Услугами**

В данной вкладке администратор может добавить условия вклада или кредита:

Обзор

Работники

Управление Услугами

Услуги

Транзакции

ВЫЙТИ

ID Услуги

Валюта:  
RUB

НАЙТИ

Нет данных 🤔

Управление услугами:

Выбранный ID Услуги

РАЗБЛОКИРОВАТЬ

ЗАБЛОКИРОВАТЬ

Добавление услуги

Продолжительность в месяцах  
36

Процентная Ставка  
3.8

Валюта:  
RUB

Тип:  
кредит

Условия просрочки  
Я не знаю, что тут написать

Необходимый доход в месяц  
20000

Описание  
Какой-то кредит

ДОБАВИТЬ

После нажатия кнопки добавить форма очищается (если нет ошибок)

Успех 🎉

Продолжительность в месяцах

Процентная Ставка

Валюта:  
RUB

Тип:  
вклад

Условия досрочного снятия средств

Описание

ДОБАВИТЬ

Также можно найти услугу по Id, либо вывести все услуги с выбранной валютой и заблокировать/разблокировать её.

Валюта: RUB

Найденные услуги:

ID	Тип	Валюта	Описание	Блокировка
1	вклад	RUB	High Interest Deposit	нет
2	вклад	RUB	Long Term Flexible Deposit	нет

1 row selected 1-2 of 7 < >

## Управление услугами:

Выбранный ID Услуги  
1

После нажатия на кнопку “ЗАБЛОКИРОВАТЬ” с данной услугой больше нельзя открывать новые счета, выводится сообщение:

localhost:3000 の内容

Услуга: с Id:1 заблокированна!

После нажатия на кнопку “РАЗБЛОКИРОВАТЬ” с данной услугой вновь можно открывать новые счета, выводится сообщение:

localhost:3000 の内容

Услуга: с Id:1 разблокированна!



## ● Услуги

В данной вкладке можно посмотреть возможные условия кредитов и вкладов

Обзор

Работники

Управление Услугами

Услуги

Транзакции

Выйти

Количество

10

ОБНОВИТЬ

Услуги:

ID	Срок в мес.	Ставка	Тип	Условия Просрочки	Условия Снятия	Валюта	Необх. Доход	Описание
2	60	0.01	вклад	N/A	N/A	RUB	0	Long Term Flexible Deposit
3	12	1.5	вклад	N/A	N/A	JPY	0	High Interest Deposit
4	60	0.01	вклад	N/A	N/A	JPY	0	Long Term Flexible Deposit
5	12	2.25	вклад	N/A	N/A	USD	0	High Interest Deposit
6	60	0.01	вклад	N/A	N/A	USD	0	Long Term Flexible Deposit
7	12	3.1	вклад	N/A	N/A	EUR	0	High Interest Deposit
8	60	0.01	вклад	N/A	N/A	EUR	0	Long Term Flexible Deposit
9	12	4.12	вклад	N/A	N/A	CNY	0	High Interest Deposit
10	60	0.01	вклад	N/A	N/A	CNY	0	Long Term Flexible Deposit
11	60	10.2	кредит	N/A	N/A	RUB	80000	Regular Loan

- **Транзакции**

В данной вкладке администратор может просмотреть все транзакции за указанный период времени

Обзор

Работники

Управление Услугами

Услуги

Транзакции

Выйти

Начало периода  
01/01/2020

Конеч период  
12/27/2020

Количество  
100

ОБНОВИТЬ

Транзакции за указанный период:

ID Транзакции	Счет Списания	Счет Зачисления	Сумма	Валюта	Статус	ID Оператора	Метка Времени
1	Наличные	1	156600	RUB	исполнено	4	29.11.2020, 17:33:29
2	Наличные	2	28100	RUB	исполнено	96	29.11.2020, 17:33:29
3	Наличные	3	1032600	CNY	исполнено	61	29.11.2020, 17:33:29
4	Наличные	4	51300	RUB	исполнено	83	29.11.2020, 17:33:29
5	Наличные	5	148400	RUB	исполнено	82	29.11.2020, 17:33:29

1-5 of 100 < >

## Оператор

- Клиенты

В данной вкладке оператор может добавить данные клиента

Клиенты

Счета

Транзакции

Услуги

ВЫЙТИ

ID Клиента

ФИО

Номер Паспорта

НАЙТИ

Нет данных 😊

Введите корректный номер паспорта (10 цифр)

Номер паспорта

84635545932

ФИО

Иванов Иван Иванович

День Рождения

08/22/1980

ИНН

Номер телефона

Доход в месяц

ДОБАВИТЬ

Управление клиентами:

Выбранный ID Клиента

ОБНОВИТЬ ДАННЫЕ

ОБНОВИТЬ ДОСТУП

УДАЛИТЬ

После исправления ошибок и нажатия кнопки добавить форма очищается

Успех 🎉

Номер паспорта

ФИО

День Рождения

ИНН

Номер телефона

Доход в месяц

ДОБАВИТЬ

Также можно найти клиента по Id, ФИО или номеру паспорта. Изменить его данные, сменить его логин или пароль, либо удалить его личные данные

Клиенты

Счета

Транзакции

Услуги

Выйти

ID Клиента

ФИО

Номер Паспорта  
8463554599

НАЙТИ

Успех

Найденные клиенты:

ID	ФИО	Номер Паспорта
10002	名前	8463554599

1 row selected1-1 of 1

Управление клиентами:

Выбранный ID Клиента  
10002

ОБНОВИТЬ ДАННЫЕ

ОБНОВИТЬ ДОСТУП

УДАЛИТЬ

Номер паспорта

ФИО

День Рождения

ИНН

Номер телефона

Доход в месяц

ДОБАВИТЬ

После нажатия на кнопку “ОБНОВИТЬ ДАННЫЕ” форма справа меняется на эту:

Изменение данных клиента

Id клиента  
10002

Номер паспорта  
8463554599

ФИО  
名前

День Рождения  
11/29/2020

ИНН  
null

Номер телефона  
null

Доход в месяц  
null

ОТМЕНА

ОБНОВИТЬ

После нажатия на кнопку “ОБНОВИТЬ ДОСТУП” форма справа меняется на эту:

Изменение аутентификационных данных клиента

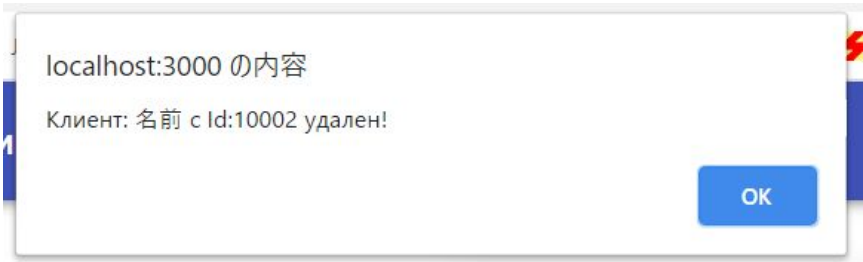
Логин

Пароль

ОТМЕНА

ОБНОВИТЬ

После нажатия на кнопку “УДАЛИТЬ” личные данные клиента удаляются и выводится сообщение:



Также на этой вкладке оператор может посмотреть данные клиента, счета и транзакции

Данные Клиента с ID 1222:

Количество  
100

ОБНОВИТЬ

Данные:

ID	Номер Паспорта	ФИО	Дата Рождения	ИНН	Телефон	Доход
1222	1761451121	Deborah Mendoza	10.06.2000	-	89911111952	-

1-1 of 1 < >

Счета:

ID	Тип	ID Услуги	Сумма	Дата Открытия	Валюта	Сумма Процентов	Закрит?
2054	вклад	3	11700	29.11.2020	JPY	0	нет
2055	вклад	2	6200	29.11.2020	RUB	0	нет

1-2 of 2 < >

Транзакции:

ID Транзакции	Счет Списания	Счет Начисления	Сумма	Валюта	Статус	Метка Времени
2054	Наличные	2054	11700	JPY	исполнено	29.11.2020, 17:33:32
2055	Наличные	2055	6200	RUB	исполнено	29.11.2020, 17:33:32

1-2 of 2 < >

- **Счета**

На этой вкладке оператор может управлять существующими счетами клиента и добавлять новые

Клиенты

Счета

Транзакции

Услуги

Выйти

ID Клиента

123

ФИО

Номер Паспорта

Найти

Найденные клиенты:

ID	ФИО	Номер Паспорта
123	Donald Tateer	9701920133

1 row selected1-1 of 1

ID Услуги

Валюта: RUB

Найти

Нет данных

Открытие счета

ID Услуги

Сумма

ID Клиента

123

Добавить

Управление счетами:

Выбранный ID Счета

Разблокировать

Заблокировать

Перевести с

Перевести на

Выбрав один из счетов клиента можно произвести с ним ряд действий

Найденные счета:

ID	Тип	Валюта	ID услуги	Сумма	Блокировка
209	вклад	RUB	1	243600	да

1 row selected1-1 of 1

## Управление счетами:

Выбранный ID Счета

209

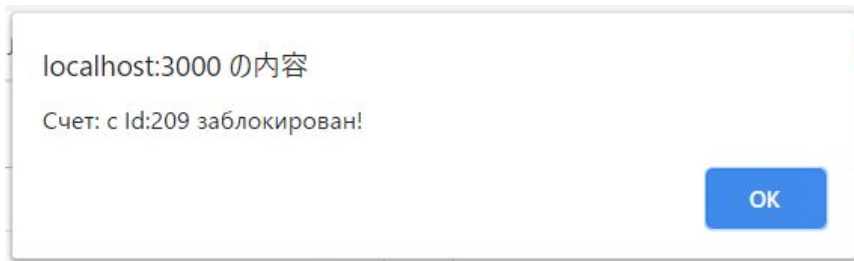
Разблокировать

Заблокировать

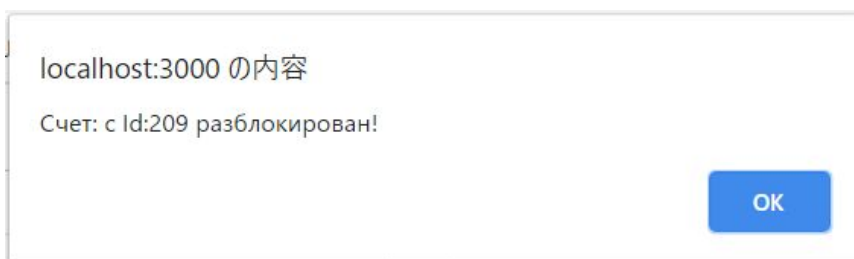
Перевести с

Перевести на

После нажатия на кнопку “ЗАБЛОКИРОВАТЬ” с данным счетом больше нельзя проводить транзакции, выводится сообщение:



После нажатия на кнопку “РАЗБЛОКИРОВАТЬ” с данным счетом вновь можно проводить транзакции, выводится сообщение:



После нажатия на кнопку “ПЕРЕВЕСТИ С” открывается вкладка Транзакции, где выбранный счет установлен как счет списания:

КлиентыСчетаТранзакцииУслуги

Выйти

Выполнение транзакции

Со счета с номером:  
209

На счет с номером:

Сумма:

Валюта:  
RUB

Выполнить

После нажатия на кнопку “ПЕРЕВЕСТИ НА” открывается вкладка Транзакции, где выбранный счет установлен как счет начисления:

КлиентыСчетаТранзакцииУслуги

Выйти

Выполнение транзакции

Со счета с номером:

На счет с номером:  
209

Сумма:

Валюта:  
RUB

Выполнить

Для добавления нового счета, после того как клиент выбран в форме слева, в форме справа необходимо найти услугу по Id или валюте, выбрать её и указать сумму. После чего можно добавить счет нажав на соответствующую кнопку.

ID Услуги

1

Валюта:

JPY

НАЙТИ

Найденные услуги:

ID	Тип	Валюта	Описание	Блокировка
4	вклад	JPY	Long Term Flexible Deposit	нет
13	кредит	JPY	Regular Loan	нет

1 row selected

3-4 of 6

<

>

Открытие счета

ID Услуги

4

Сумма

10000

ID Клиента

123

ДОБАВИТЬ

- **Транзакции**

На данной вкладке оператор может совершить транзакцию. Для этого необходимо указать хотя бы один номер счета (если счет списания отсутствует, то деньги принимаются от клиента наличными, если нет счета начисления, то деньги выдаются клиенту на руки), сумму (она должна быть неотрицательной) и выбрать валюту. Также нужно помнить, что на кредит не может быть счетом списания.

Клиенты

Счета

Транзакции

Услуги

Выйти

Выполнение транзакции

Со счета с номером:

На счет с номером:

Сумма:

Валюта:  
RUB

ВЫПОЛНИТЬ

Если не указать оба номера счета, то покажется ошибка:

Необходимо указать хотя бы один номер счета!

Если указать некорректную сумму, то покажется ошибка:

Введите корректную сумму (неотрицательное вещественное число)

Если указать кредит счетом списания, то сработает триггер и покажется ошибка:

WARNING: INCORRECT TRANSACTION HAS BEEN REJECTED

Если ошибок нет, то форма покажется сообщение об успешном выполнении и форма очистится:

Клиенты

Счета

Транзакции

Услуги

Выйти

Успех

Со счета с номером:

На счет с номером:

Сумма:

Валюта:  
RUB

ВЫПОЛНИТЬ



● Услуги

В данной вкладке можно посмотреть возможные условия кредитов и вкладов

Клиенты

Счета

Транзакции

Услуги

Выйти

Количество  
5

Обновить

Услуги:

ID	Срок в мес.	Ставка	Тип	Условия Просрочки	Условия Снятия	Валюта	Необх. Доход	Описание
2	60	0.01	вклад	N/A	N/A	RUB	0	Long Term Flexible Deposit
3	12	1.5	вклад	N/A	N/A	JPY	0	High Interest Deposit
4	60	0.01	вклад	N/A	N/A	JPY	0	Long Term Flexible Deposit
5	12	2.25	вклад	N/A	N/A	USD	0	High Interest Deposit
6	60	0.01	вклад	N/A	N/A	USD	0	Long Term Flexible Deposit

● Аккаунт

В данной вкладке клиент может посмотреть свои данные, счета и транзакции

Аккаунт

Транзакции

Услуги

Выйти

Мои Данные:

Количество  
100

Обновить

Данные:

ID	Номер Паспорта	ФИО	Дата Рождения	ИНН	Телефон	Доход
1	1367103727	Mary Edwards	11.06.1997	-	89909401483	12000

1-1 of 1

Счета:

ID	Тип	ID Услуги	Сумма	Дата Открытия	Валюта	Сумма Процентов	Закрыт?
1	вклад	2	155911.03	29.11.2020	RUB	0	нет
2	вклад	2	28796.97	29.11.2020	RUB	0	нет
20779	вклад	1	9	29.11.2020	RUB	0	нет
20780	вклад	4	12	29.11.2020	JPY	0	нет
20781	вклад	1	999	29.11.2020	RUB	0	нет

1-5 of 7

Транзакции:

ID Транзакции	Счет Списания	Счет Начисления	Сумма	Валюта	Статус	Метка Времени
1	Наличные	1	156600	RUB	исполнено	29.11.2020, 17:33:29
2	Наличные	2	28100	RUB	исполнено	29.11.2020, 17:33:29
20779	Наличные	20779	10	RUB	исполнено	29.11.2020, 20:25:04
20780	Наличные	20780	12	JPY	исполнено	29.11.2020, 20:55:56
20781	Наличные	20781	999	RUB	исполнено	29.11.2020, 22:54:09

1-5 of 19

## • Транзакции

В данной вкладке клиент может совершить транзакцию. Для этого необходимо указать свой счет как счет списания, указать счет начисления, сумму (она должна быть неотрицательной) и выбрать валюту. Также нужно помнить, что на кредит не может быть счетом списания.

[Аккаунт](#) > [Транзакции](#) > [Услуги](#) ВЫЙТИ

① Выполнение транзакции

Со счета с номером:

На счет с номером:

Сумма:

Валюта:  
RUB

ВЫПОЛНИТЬ

Если не указать счет списания, то покажется ошибка:

① Предоставьте корректный номер счета с которого будут списаны средства

Если не указать счет начисления, то покажется ошибка:

① Предоставьте корректный номер счета на который поступят средства

Если указать счет, не принадлежащий клиента, то покажется ошибка:

① Необходимо указать свой счет как счет списания!

Если указать некорректную сумму, то покажется ошибка:

① Введите корректную сумму (неотрицательное вещественное число)

Если указать кредит счетом списания, то сработает триггер и покажется ошибка:

① WARNING: INCORRECT TRANSACTION HAS BEEN REJECTED

Если ошибок нет, то форма покажется сообщение об успешном выполнении и форма очистится:

Аккаунт

Транзакции

Услуги

Выйти

Успех

Со счета с номером:

На счет с номером:

Сумма:

Валюта:  
RUB

Выполнить

Услуги

В данной вкладке можно посмотреть возможные условия кредитов и вкладов

Аккаунт

Транзакции

Услуги

Выйти

Количество  
10

Обновить

Услуги:

ID	Срок в мес.	Ставка	Тип	Условия Просрочки	Условия Снятия	Валюта	Необх. Доход	Описание
2	60	0.01	вклад	N/A	N/A	RUB	0	Long Term Flexible Deposit
3	12	1.5	вклад	N/A	N/A	JPY	0	High Interest Deposit
4	60	0.01	вклад	N/A	N/A	JPY	0	Long Term Flexible Deposit
5	12	2.25	вклад	N/A	N/A	USD	0	High Interest Deposit
6	60	0.01	вклад	N/A	N/A	USD	0	Long Term Flexible Deposit
7	12	3.1	вклад	N/A	N/A	EUR	0	High Interest Deposit
8	60	0.01	вклад	N/A	N/A	EUR	0	Long Term Flexible Deposit
9	12	4.12	вклад	N/A	N/A	CNY	0	High Interest Deposit
10	60	0.01	вклад	N/A	N/A	CNY	0	Long Term Flexible Deposit
11	60	10.2	кредит	N/A	N/A	RUB	80000	Regular Loan

## **Вывод**

В данной курсовой работе было реализовано клиентское приложение для работы сберегательного банка с использованием SQL Server Management Studio, React JS для написания веб приложения и Node JS для написания REST API, необходимого для работы веб приложения.

Главной трудностью был объем работы необходимый для написания API и веб приложения, исходный код которых суммарно превышает 5000 строк. У приложения есть некоторые недостатки, например отсутствие возможности изменить пароль без помощи администратора для оператора и оператора для клиента.

В дальнейшем планируется доработать все недостатки.

## Список литературы.

1. Кляйн К., Кляйн Д., Хант Б. Справочник SQL (3-е издание).: Пер. с англ.—М.: Символ-Плюс—652с.
2. Петкович Д. Microsoft® SQL Server™ 2012. Руководство для начинающих: Пер. с англ.—СПб.: БХВ-Петербург, 2013.—816 с.
3. Руководство пользователя SQL Management Studio for SQL Server // EMS Database Management Solutions, Ltd. - 1999-2015