МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Вятский государственный университет»
Факультет автоматики и вычислительной техники
Кафедра электронных вычислительных машин

Допущено к защите
Руководитель проекта
_____/_____/
(подпись) (Ф.И.О.)
«___»_____2021г.

РАЗРАБОТКА ПРОГРАММЫ ДЛЯ ЭКСПОРТА ДАННЫХ ИЗ EXCEL-ФАЙЛА В БАЗУ ДАННЫХ

Пояснительная записка курсового проекта по дисциплине
«Технология разработки программного обеспечения»

ТПЖА.090301.869 ПЗ

Разработал студент группы ИВТм-11_____/Баташев П.А./

Руководитель, доцент кафедры ЭВМ _____/Долженкова М.Л/

Работа защищена с оценкой «_____» _____
(оценка) (дата)

Члены комиссии _____/ _____/
(подпись) (Ф.И.О)

_____/ _____/
(подпись) (Ф.И.О)

_____/ _____/
(подпись) (Ф.И.О)

Киров 2021

Реферат

Баташев П.А. Разработка программы для экспорта данных из Excel-файла в базу данных. ТПЖА.090301.869 ПЗ: Курс. проект / ВятГУ, каф. ЭВМ; рук. Долженкова М.Л. - Киров, 2021. – ПЗ 40 с , 17 рис., 4 табл., 7 источников, 6 прил.

ЭКСПОРТ ДАННЫХ, EXCEL, EXCEL-ФАЙЛ, ПАРСИНГ, ПАРСИНГ EXCEL-ФАЙЛА, ДЕСКРИПТОР, БАЗА ДАННЫХ, SQL-ЗАПРОС, ГЕНЕРАТОР SQL-ЗАПРОСОВ, МАСТЕР ИМПОРТА И ЭКСПОРТА SQL SERVER, МАКРОСЫ

Цель курсовой работы – разработать программы для экспорта данных из Excel-файла в базу данных.

Ввод данных из файла в базу данных – это рутинный процесс, который при ручном выполнении может приводить к различным ошибкам. Автоматизация данного процесса позволит ускорить перенос данных и избежать различных ошибок.

Средства для экспорта данных из Excel-файла уже существуют, но все они имеют ограничения к структуре файла, например, Microsoft SQL Server Management не способен вычленять отдельные поля на странице, а также работать с несколькими страницами или с несколькими таблицами на одной странице.

Из-за отсутствия аналогов, позволяющих экспортировать из предоставленного Excel-файла с особой структурой, в базу данных, при этом производить не только вставку, но и различные проверки при экспорте, было принято решение разработать программу, имеющую перечисленные возможности.

Разработанная программа позволяет экспортировать данные из Excel-файла с любой структурой данных, при этом достаточно просто указать в дескрипторе файла свойства данных, необходимых для извлечения, а также создать SQL-запрос, в котором могут быть произведены различные действия с извлеченными данными перед вставкой в базу.

Содержание

Введение

Несмотря на то, что базы данных активно вошли в нашу жизнь и применяются в различных отраслях, есть случаи, когда данные из-за отсутствия построенной базы данных, либо из-за отсутствия доступа к базе данных, либо из-за отсутствия удобного интерфейса взаимодействия с базой данных, необходимо хранить в отдельных файлах. И в тот момент, когда база данных была создана, либо, когда доступ к базе данных появился напрямую или через посредника, то необходимо все накопленные данные, хранящиеся во множестве файлов, экспортировать из файлов в базу данных.

Процесс переноса данных из файла в базу данных рутинный, требующий много времени, внимания и концентрации, поэтому, если файлов много, или они приходят регулярно, то этот процесс необходимо автоматизировать.

При автоматизации процесса экспорта файлов одинаковой структуры в базу данных, может понадобиться возможность работать с различными страницами, с различными представлением данных, таких как одиночные поля в файле, или данные, представленные в виде таблиц, которых в свою очередь может быть несколько на одной странице. Все это требует определенного описания файла (дескриптора), по которому будет происходить поиск определенных полей или таблиц на определенных страницах.

Так же при автоматизации процесса экспорта файлов одинаковой структуры в базу данных, может понадобиться проверка извлеченных данных из файла перед вставкой в базу данных, и это может быть проверка не только ошибок в данных, но еще и сверка с данными из самой базы данных, распределение данных по разным таблицам и так далее. Все это требует определенного запроса, который одинаков для всех файлов и отличается только данными, с которыми работает данный запрос.

1 Анализ существующих аналогов

1.1 Экспорт данных с помощью средств в SQL Server

В СУБД Microsoft SQL Server существует отличный функционал по импорту и экспорту данных, причем в разные форматы и разные базы данных. Его можно также использовать для простого переноса данных из одной базы в другую или с одного сервера на другой, рисунок 1.1

Данный способ отлично подходит, если данные представлены в виде таблицы, но данный способ не подходит в том случае, если данные располагаются в различных частях файла. Так же данный способ требует наличие установленной среды разработки Microsoft SQL Server Management Studio, поэтому данный способ подойдет системным администраторам, нежели обычным пользователям.

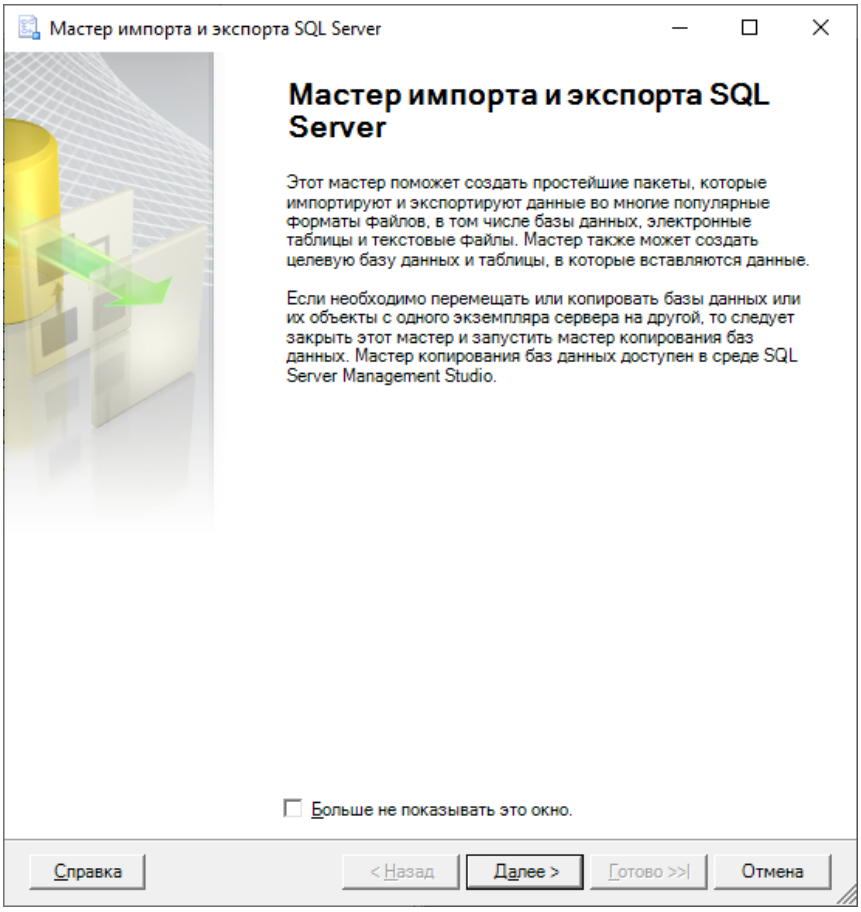


Рисунок 1.1 – Стартовое окно мастера импорта и экспорта SQL-Server

1.2 Экспорт данных с помощью средств в Excel

Экспорт данных можно производить из самого Excel, для этого необходимо написать соответствующий макрос для обработки на языке Visual Basic. С помощью макроса можно производить поиск данных в файле, подключение к базе данных и выполнение написанных SQL-запросов.

Данный способ подойдет как к данным, представленным в виде таблицы, так и данным, располагающимся в разных частях файла. Но данный способ имеет ряд недостатков. Во-первых, необходимо производить действия для подключения макросов. Во-вторых, требуется установка дополнительных библиотек для Visual Basic, если они были использованы для написания макросов. В-третьих, отсутствует пользовательский интерфейс.

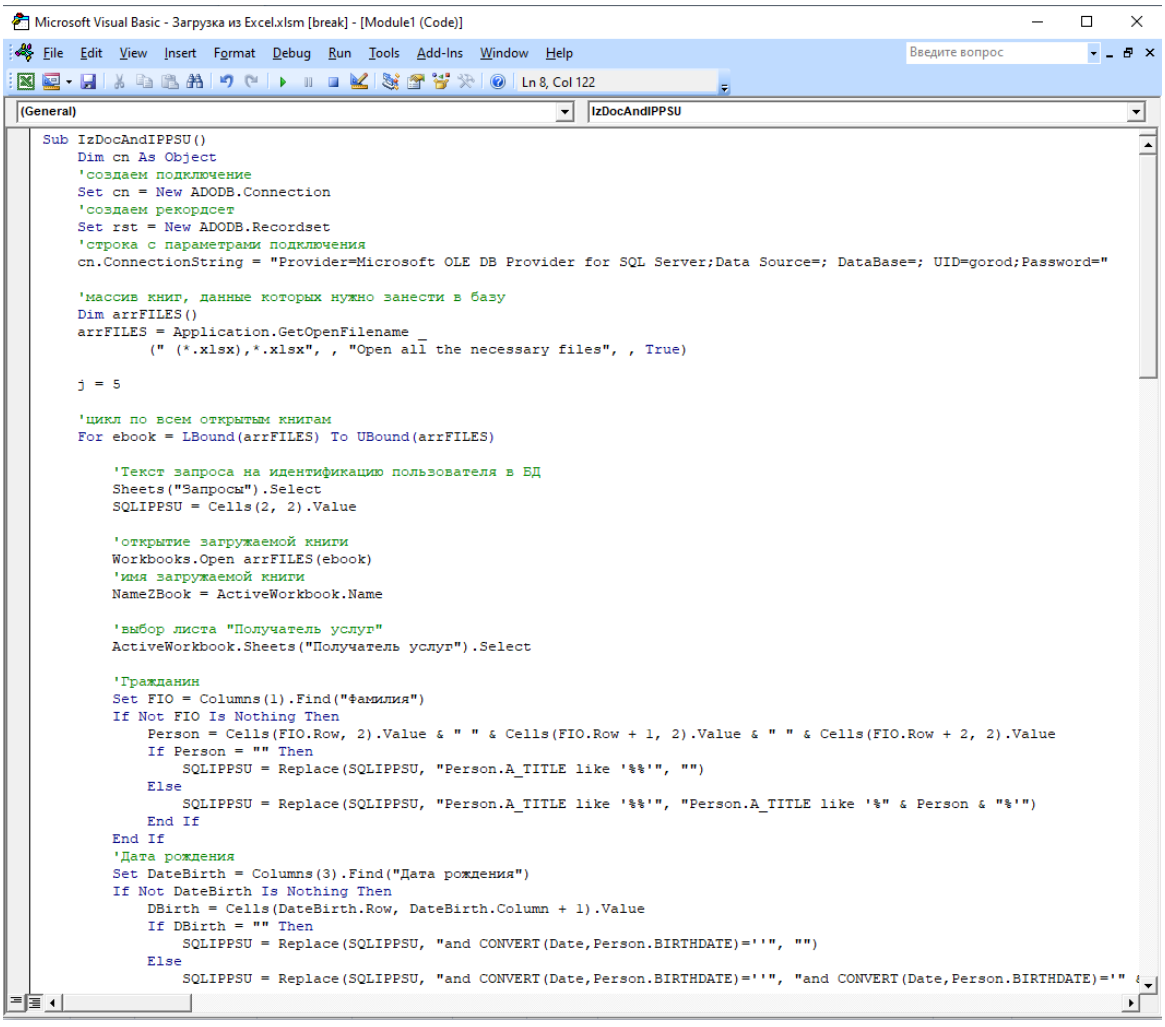


Рисунок 1.2 – Окно для написания макросов в Visual Basic

1.3 Сравнение аналогов

В таблице 1.1 представлено сравнение рассмотренных аналогов с разрабатываемой программой. Из таблицы видно, что разрабатываемая программа имеет ряд преимуществ, относительно рассмотренных аналогов.

Недостатками мастера импорта и экспорта SQL-сервер является то, что отсутствует пакетная обработка файлов, из-за чего требуется вручную проделывать ряд шагов (выбор файла, выбор базы данных и т.д.) перед экспортом для каждого файла. Так же что файлы должны иметь определенный шаблон: на странице должна располагаться одна таблица; заголовки столбцов должны располагаться в первой строке файла.

Недостатками макросов из Excel является отсутствие пользовательского интерфейса и необходимость установки дополнительных средств, чтобы макросы можно было запускать.

Таблица 1.1 – Сравнение аналогов

Параметр	Мастер импорта и экспорта SQL-сервер	Исполнение макросов из Excel	Разрабатываемая программа
экспорт данных из Excel-файла в базу данных	+	+	+
возможность обработки Excel-файла любого шаблона	-	+	+
наличие пакетной обработки файлов	-	+	+
возможность подключения собственного SQL-запроса при экспорте	+	+	+
импорт сразу в несколько различных таблиц	-	+	+
запуск не требует установки дополнительных средств	-	-	+
удобный пользовательский интерфейс	+	-	+

2 Техническое задание

2.1 Введение

Требуется написать программу, которая предоставляет средства для экспорта данных из Excel-файла в базу данных, с возможностью изменения шаблона файла и шаблона SQL-запроса.

В качестве языка программирования выбран C#, в качестве среды программирования выбрана Visual Studio, так как она содержит все необходимые компоненты для создания программы с пользовательским интерфейсом.

2.2 Основание для разработки

Основанием для разработки является необходимость переноса информации об оказанных социальных услугах, предоставляемых в Excel-файлах, шаблон которого изображен в приложении А, общественными некоммерческими организациями, у которых отсутствует доступ к базе данных ЭСРН.

2.3 Назначение

Основным назначением программы является автоматизация переноса данных из Excel-файла в базу данных, с возможностью изменять шаблон Excel-файла и SQL-запрос, без изменения программного кода.

2.4 Требования к программе

Требования к функциональным характеристикам:

- экспорт данных из Excel-файла в базу данных;
- наличие пакетной обработки файлов;
- возможность изменения шаблона Excel-файла без изменения программного кода;
- возможность изменения SQL-запроса без изменения программного кода;
- подробное описание ошибок на всех этапах экспорта данных.

Требования к минимальной конфигурации:

- минимальное разрешение экрана 800x600;
- наличие компьютерной мыши;
- наличие клавиатуры.

Требования к программной совместимости:

- операционная система: Windows 7,8,10.

2.5 Требования к документации

Документация должна содержать:

- тексты программ со всеми необходимыми комментариями;
- пояснительную записку, содержащей описание разработки;
- руководство пользователя.

3 Разработка парсера Excel-файла

Парсинг – это метод индексирования информации с последующей конвертацией ее в иной формат или даже иной тип данных. Парсинг позволяет взять файл в одном формате и преобразовать его данные в более удобоваримую форму, которую можно использовать в своих целях.

В нашем случае парсинг это процесс обработки данных из Excel-файла. Он подразумевает анализ текста, вычленение оттуда необходимых материалов и их преобразование в подходящий вид. Благодаря парсингу можно находить на страницах файла небольшие клочки полезной информации и в автоматическом режиме их оттуда извлекать, чтобы потом переиспользовать.

Вычленяемая информация в нашем случае это будет та информация, которая нужна для вставки в SQL-скрипт. Представлена она может быть как отдельными полями, так и табличными значениями.

3.1 Дескриптор Excel-файла общая информация

Под дескриптором Excel-файла будем понимать ту информацию, которая описывает извлекаемые объекты из файла. Дескриптор состоит из объектов и их атрибутов. На данный момент есть три вида объектов, которые заключаются в открывающий и закрывающий тег:

- <singleValue>…</singleValue> - информация об значении, хранившемся в одном поле;
- <table>…</table> - информация об таблице, в которой хранятся значения;
- <column>…</column> - является вложенным тегом в тег table и является информацией о столбце таблицы, в котором хранятся данные.

Атрибуты в свою очередь представляются в виде токенов. Под токеном подразумевается значение и короткого описания этого значения (имя токена). Токен представляется следующим образом:

Имя токена : значение токена;

Каждый токен объекта должен быть отделен точкой с запятой, при этом, если имя или значение имеют пробелы и табуляцию, то они должны быть заключены в кавычки. Таким образом на рисунке 3.1 представлен дескриптор объекта, который имеет N атрибутов и один вложенный тег. А в приложении Б представлен дескриптор шаблона Excel-файла, представленного в приложении А.

```
<tag>
    nameToken_1 : valueToken_1;
    …
    nameToken_N : valueToken_N;
    <nestedObject>
        nameToken_1 : valueToken_1;
        …
        nameToken_N : valueToken_N;
        <nestedObject>
            nameToken_1 : valueToken_1;
            …
            nameToken_N : valueToken_N;
        </nestedObject>
    </nestedObject>
</tag>
```

Рисунок 3.1 – Пример оформления дескриптора объекта

В таблице 3.1 представлены возможные имена атрибутов тега singleValue.

Таблица 3.1 – Возможные атрибуты тега singleValue

Коды атрибутов	Допустимые значения	Описание	Если значение не указано
1	2	3	4
SHEET_NUMBER	Целочисленные числа, больше 0	Номер страницы, на которой производится поиск (отсчет страниц ведется с 1).	Поиск производится на всех страницах
SHEET_NAME	Любая строка	Имя страницы, на которой производится поиск.	Поиск производится на всех страницах
SECTION_NAME	Любая строка	Раздел на странице, относительно которого производится поиск (для полей с одинаковыми надписями).	Поиск производится на всей странице
SECTION_BOTTOM_LEFT	1 или 0	При условии, что атрибут SECTION_NAME определен. Область снизу слева от раздела включать в поиск, рисунок 3.2.	Поиск не производится в левой нижней части от раздела (Аналогично значению 0).
SECTION_BOTTOM_RIGHT	1 или 0	При условии, что атрибут SECTION_NAME определен. Область снизу справа от раздела включать в поиск, рисунок 3.2.	Поиск не производится в правой нижней части от раздела (Аналогично значению 0).
SECTION_UP_LEFT	1 или 0	При условии, что атрибут SECTION_NAME определен. Область сверху слева от раздела включать в поиск, рисунок 3.2.	Поиск не производится в левой верхней части от раздела (Аналогично значению 0).

Продолжение таблицы 3.1

1	2	3	4
SECTION_UP_RIGHT	1 или 0	При условии, что атрибут SECTION_NAME определен. Область сверху справа от раздела включать в поиск, рисунок 3.2.	Поиск не производится в правой верхней части от раздела (Аналогично значению 0).
FIELD	Любая строка	Текст ячейки, относительно которой ищется значение.	Аналогично пустой ячейке.
CODE	Любая строка	Код значения, по которому можно идентифицировать данное поле.	Значение данного объекта будет получено, но обратиться к нему будет нельзя в SQL-скрипте.
OFFEST_ROW	Любое целочисленное число	Смещение по строке относительно FIELD.	Аналогично значению 0
OFFEST_COLUMN	Любое целочисленное число	Смещение по столбцу относительно FIELD.	Аналогично значению 0

В таблице 3.2 представлены возможные имена атрибутов тега table.

Таблица 3.2– Возможные атрибуты тега table

Коды атрибутов	Допустимые значения	Описание	Если значение не указано
1	2	3	4
SHEET_NUMBER	Целочисленные числа, больше 0	Номер страницы, на которой производится поиск (отсчет страниц ведется с 1).	Поиск производится на всех страницах
SHEET_NAME	Любая строка	Имя страницы, на которой производится поиск.	Поиск производится на всех страницах

Продолжение таблицы 3.2

1	2	3	4
SECTION_NAME	Любая строка	Раздел на странице, относительно которого производится поиск (для полей с одинаковыми надписями).	Поиск производится на всей странице
SECTION_BOTTOM_LEFT	1 или 0	При условии, что атрибут SECTION_NAME определен. Область снизу слева от раздела включать в поиск, рисунок 3.2.	Поиск не производится в левой нижней части от раздела (Аналогично значению 0).
SECTION_BOTTOM_RIGHT	1 или 0	При условии, что атрибут SECTION_NAME определен. Область снизу справа от раздела включать в поиск, рисунок 3.2.	Поиск не производится в правой нижней части от раздела (Аналогично значению 0).
SECTION_UP_LEFT	1 или 0	При условии, что атрибут SECTION_NAME определен. Область сверху слева от раздела включать в поиск, рисунок 3.2.	Поиск не производится в левой верхней части от раздела (Аналогично значению 0).
SECTION_UP_RIGHT	1 или 0	При условии, что атрибут SECTION_NAME определен. Область сверху справа от раздела включать в поиск, рисунок 3.2.	Поиск не производится в правой верхней части от раздела (Аналогично значению 0).
CODE	Любая строка	Код таблицы, по которому можно идентифицировать данную таблицу.	Значения данной таблицы будут получены, но обратиться к ним будет нельзя в SQL-скрипте.
INCLUDE_FINAL_ROW	1 или 0	Записывать последнюю строку таблицы, которая определяется во вложенных объектах column.	Аналогично значению 0

В таблице 3.3 представлены возможные имена атрибутов тега column.

Таблица 3.3– Возможные атрибуты тега table

Коды атрибутов	Допустимые значения	Описание	Если значение не указано
NAME	Любая строка	Заголовок столбца.	Поиск производится на всех страницах
CODE	Любая строка	Код столбца, по которому можно идентифицировать данный столбец.	Значения данного столбца будут получены, но обратиться к ним будет нельзя в SQL-скрипте.
FINAL_CELL	Любая строка	Значение, говорящее, что данная ячейка последняя. Последней строкой в таблице считается та строка, когда у всех столбцов соответствующая ячейка равняется ее конечному значению. Если нужно указать, что пустая ячейка, то нужно указать двойные кавычки, идущие подряд без пробела.	Любой текст в ячейке является сигналом конечной строки.

На рисунке 3.2 изображены области для поиска относительно раздела при соответственно установленных флагов.

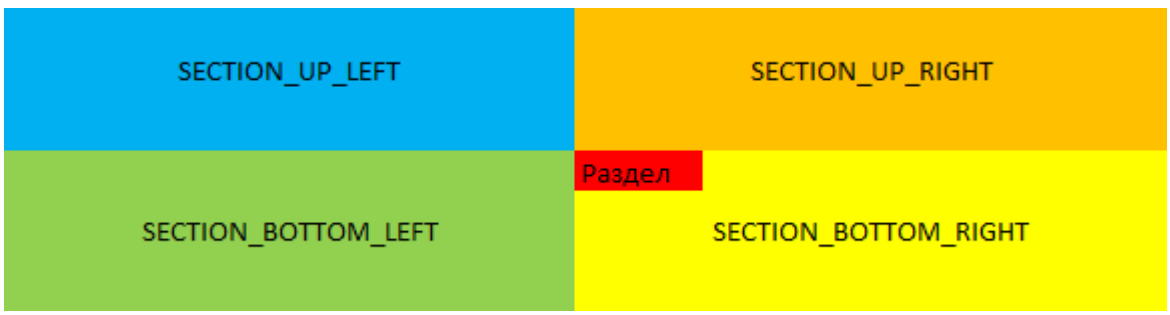


Рисунок 3.2 – Области для поиска относительно раздела

3.2 Парсинг дескриптора Excel-файла

Парсинг дескриптора Excel-файла происходит следующим образом. Происходит проход по дескриптору, пока не будет найден значащий символ (любой символ кроме табуляции и пробелов). Далее определяется, является ли данный символ началом тега. Если нет, то выдается ошибка, иначе читается имя тега и происходит дальнейший проход. После тега так же ищется первый значащий символ. Далее определяется, является ли данный символ началом вложенного тега или окончанием текущего тега. Если ни тем, ни другим не является, то считается, что встретили атрибут объекта. Для атриубта берется подстрока от текущей позиции до символа точки с запятой, и из данной подстроки извлекается пара имени и значения токена, после чего прочитанный токен помещается в список токенов текущего объекта. Если же встретился вложенный тег, то обработка его происходит аналогично тому, как было описано выше, только после встречи закрывающего тега вложенного объекта происходит добавление прочитанного вложенного объекта в список вложенных объектов того объекта, в котором этот вложенный объект был встречен. Если же встречен закрывающий тег текущего объекта, то данный объект помещается в список дескриптора объекта и происходит дальнейший поиск тега, либо окончания дескриптора Excel-файла. Описанный алгоритм отображен на рисунке 3.3.

4 Разработка исполнителя SQL-запросов

Задача исполнителя SQL-запросов заключается в том, чтобы занести прочитанные данные из Excel-файла в готовый SQL-запрос и отправить на сервер отредактированный запрос на выполнение.

Место для вставки одиночных данных в SQL-запрос должно представлять из себя код одиночного значения, заключенного в символы

решетки. Например, была получена дата из Excel файла, которой был присвоен код dateReport. Место, в которое должно быть вставлено это значение, должно выглядеть следующим образом:

DECLARE @date = CONVERT(DATE, '#dateReport#')

И тогда, перед тем, как исполнить SQL-запрос, данный код будет заменен прочитанным значением, например 01-01-2021:

DECLARE @date DATE = CONVERT(DATE, '01-01-2021')

Для табличных значений необходимо на место, где они должны быть вставлены, вставить шаблон строки, которая будет копироваться и вставляться на место данного шаблона. При этом шаблон должен быть заключен в символы «<» и «>» и в начале иметь код таблицы, а далее в скобках указаны столбцы таблицы, заключенные в символы решетки. Допустим была прочитана таблица, код для которой был указан dataForInsert. Данная таблица имеет два столбца с кодами dateInfo и Info, тогда шаблон для вставки данных значений в SQL-запрос будет выглядеть следующим образом:

<dataForInsert (CONVERT(DATE, '#dateInfo#'), '#Info#')>

И тогда, перед тем, как исполнить SQL-запрос, данный шаблон будет заменен прочитанными значениями, например:

(CONVERT(DATE, '01-01-2021'), 'Новый год')

(CONVERT(DATE, '23-02-2021'), 'День защитника отечества')

При составлении SQL-запроса нужно учитывать, что при вставки значений не учитываются типы данных, поэтому, если данные должны быть вставлены как строка, то необходимо код, заключенный в символы решетки, заключить еще в одинарные кавычки. А если, к примеру, данные должны быть вставлены как дата, то нужно еще поместить шаблон в конструкцию по конвертации строки в дату, как это было показано выше.

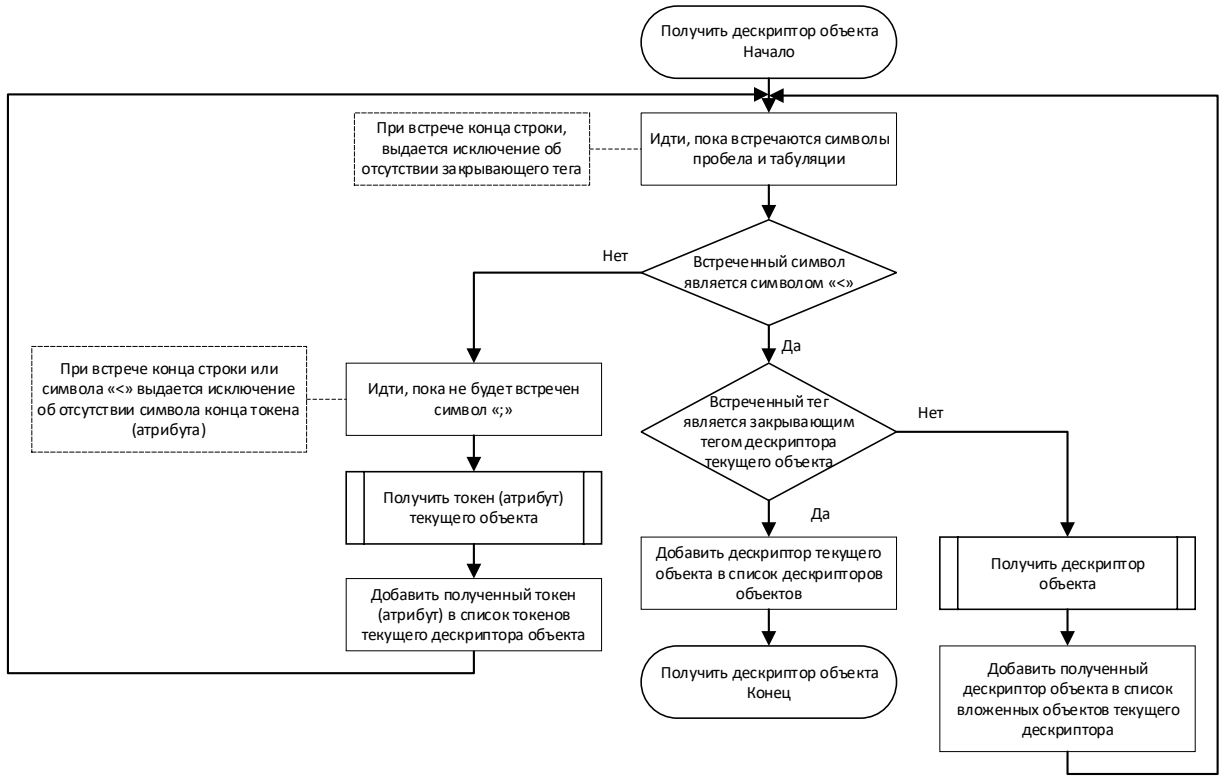


Рисунок 3.3 – Алгоритм получения дескриптора объекта

5 Разработка пользовательского интерфейса

Главное окно разработанной программы содержит в себе две основные секции – меню и панель обработки файлов, рисунок 5.1. При загрузке файлов отображается прогресс загрузки файлов, а также статус. При ошибке чтения фала в поле дополнительной информации отображается причина ошибки, рисунок 5.2.

После загрузки файла можно посмотреть данные, которые были извлечены парсером. Одна вкладка с одиночными данными, рисунок 5.5, и вкладки с табличными значениями, рисунок 5.6. Так же после загрузки файла можно посмотреть сгенерированный запрос со вставленными значениями, рисунок 5.7.

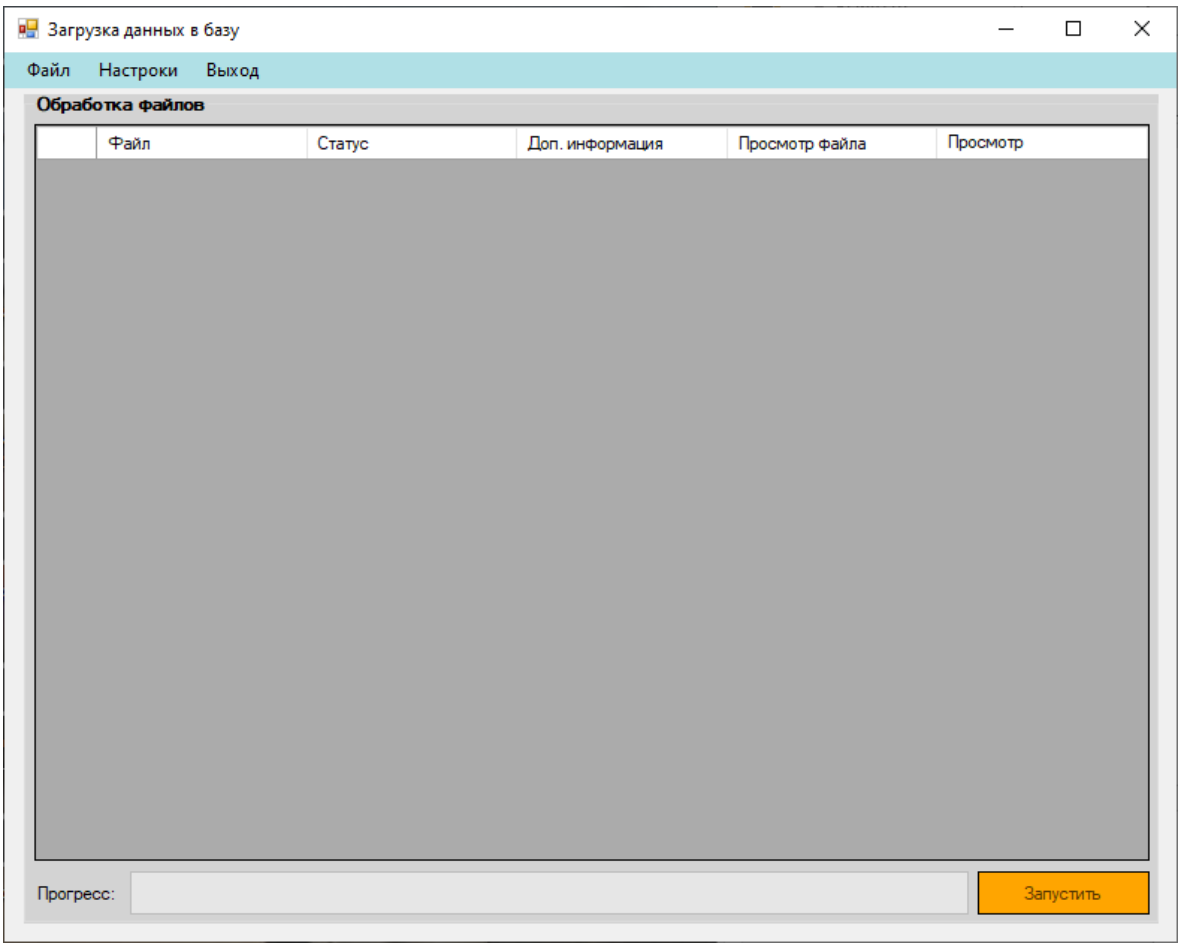


Рисунок 5.1 – Главное окно программы

На меню присутствуют следующие кнопки:

- Файл – при нажатии выпадает подпункты, где можно выбрать один файл или целую папку для обработки;
- Настройки – при нажатии на которую можно в открывшемся окне посмотреть результат парсинга дескриптора Excel-файла и шаблон SQL-запроса, а так же установить собственные файлы. По умолчанию файлы берутся из папки Source, которая располагается в папке с исполняющим файлом. По умолчанию файлы берутся с названием descriptor.txt и Query.sql;
- Выход – выход из программы.

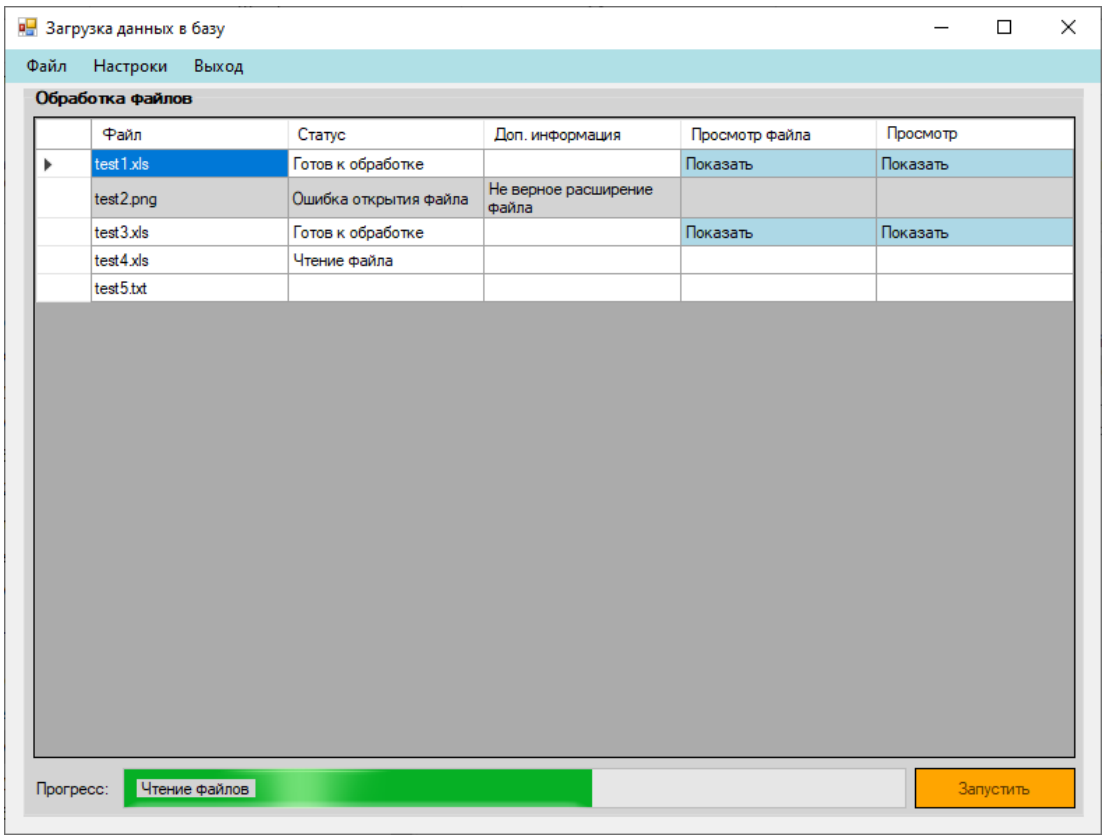


Рисунок 5.2 – Загрузка файлов

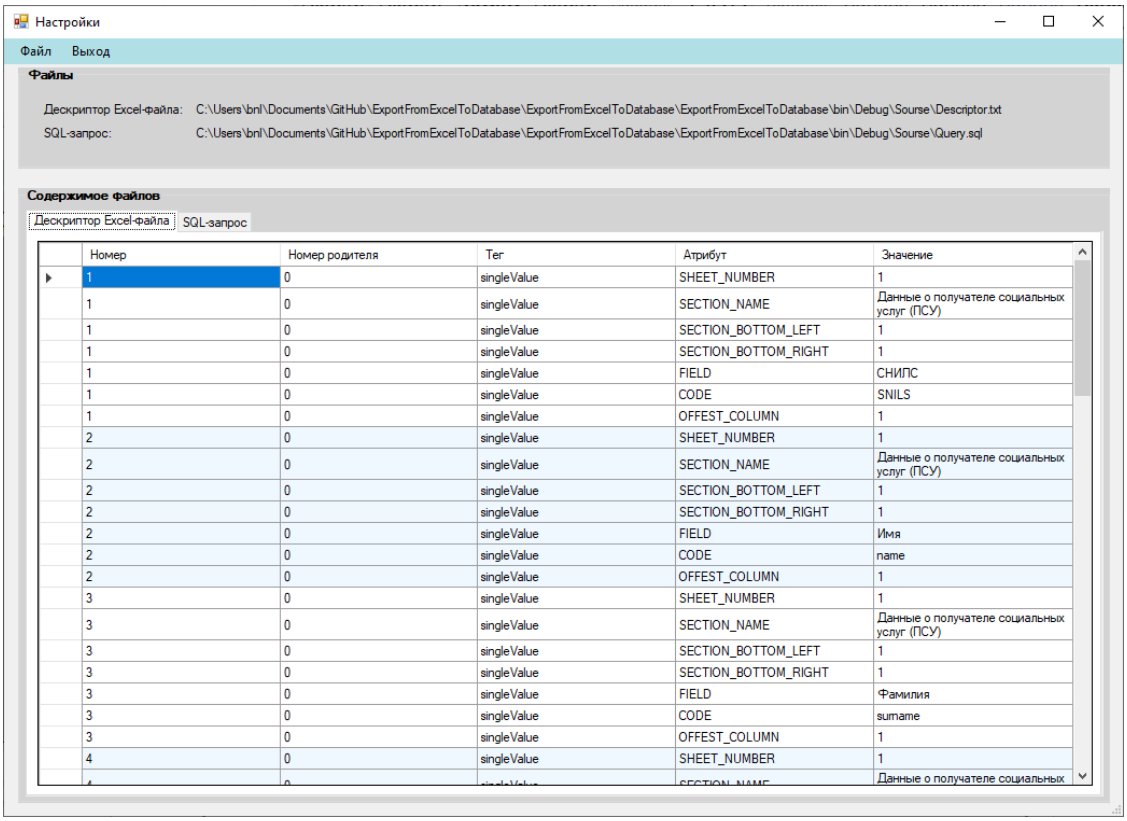


Рисунок 5.3 – Настройки (вкладка с отображением дескриптора)

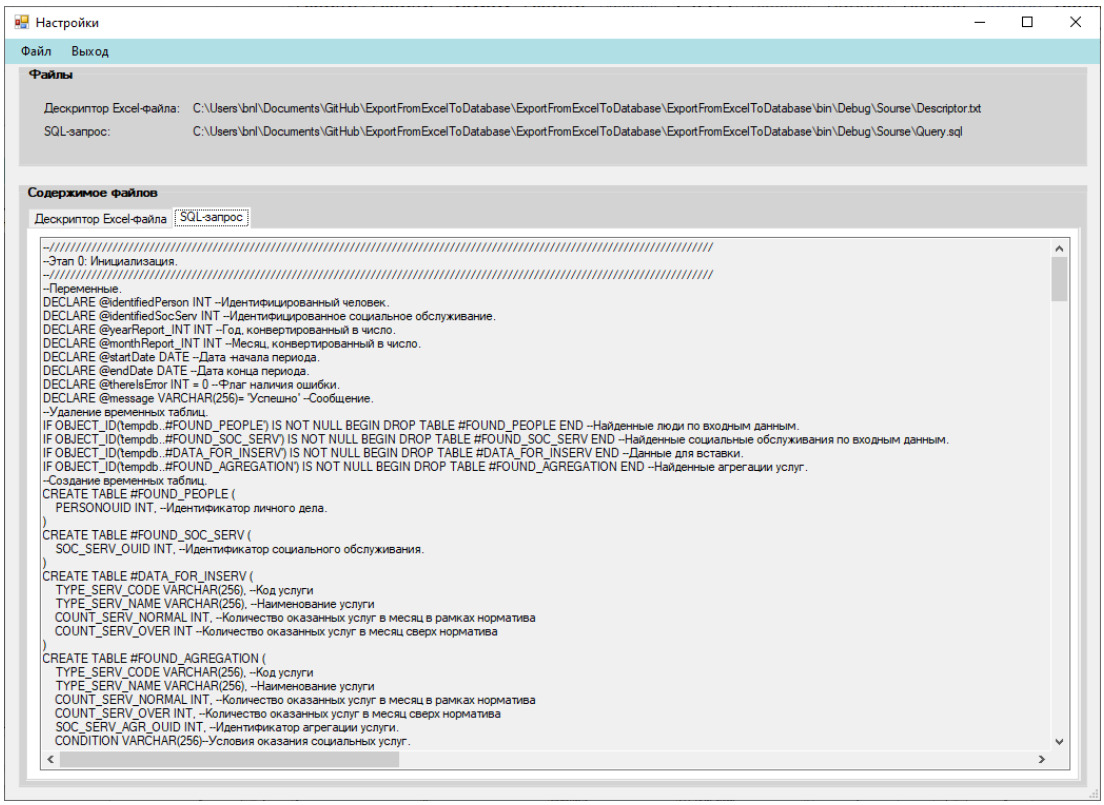
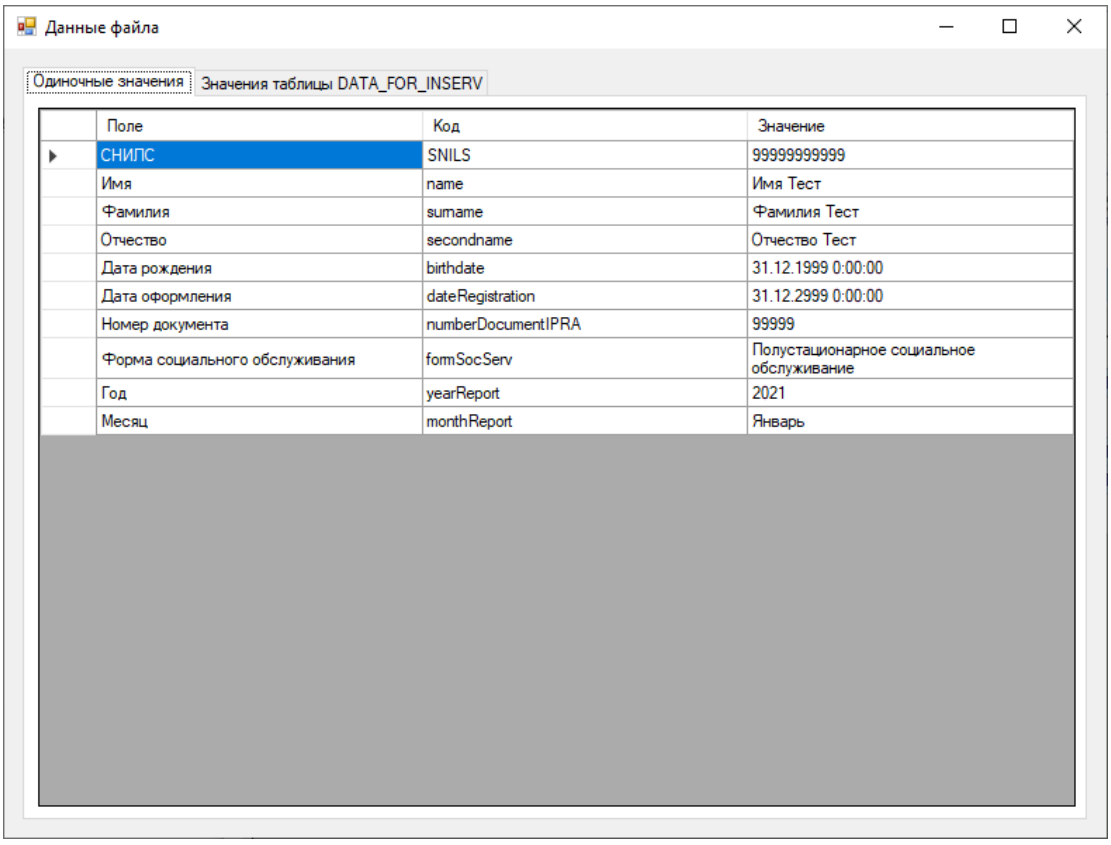
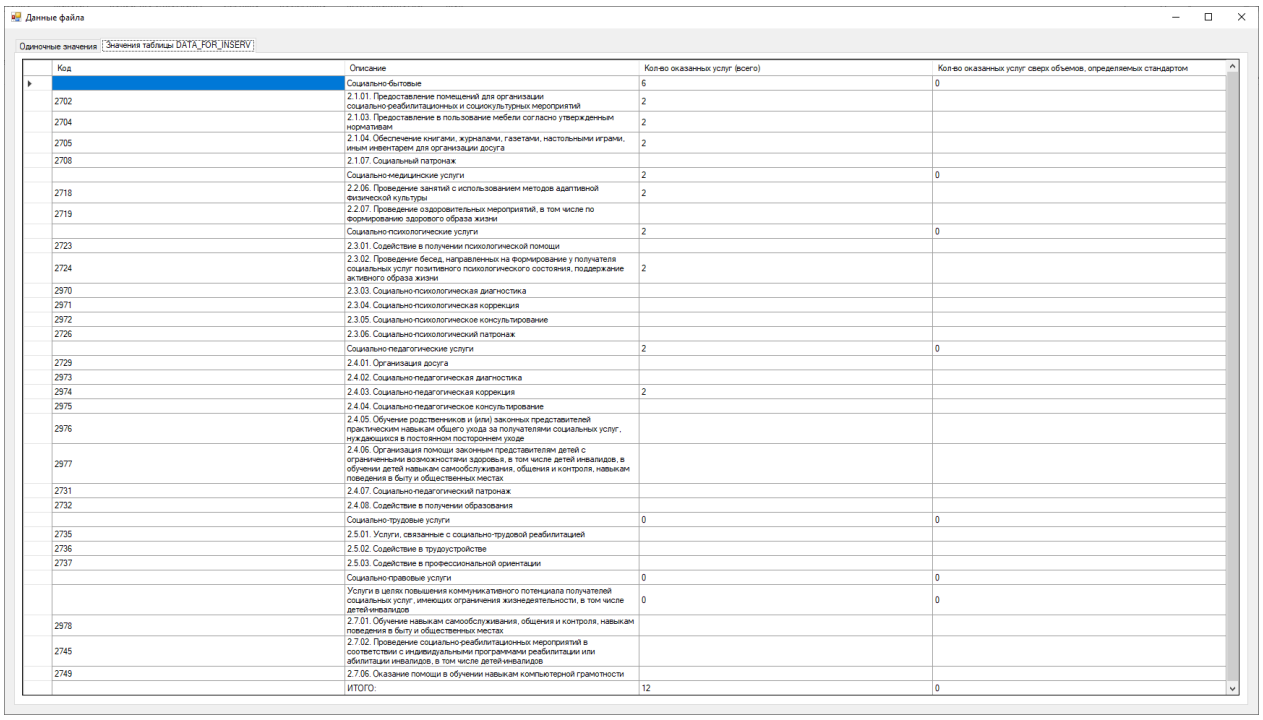


Рисунок 5.4 – Настройки (вкладка с отображением запроса)



Поле	Код	Значение
СНИЛС	SNILS	99999999999
Имя	name	Имя Тест
Фамилия	sumame	Фамилия Тест
Отчество	secondname	Отчество Тест
Дата рождения	birthdate	31.12.1999 0:00:00
Дата оформления	dateRegistration	31.12.2999 0:00:00
Номер документа	numberDocumentIPRA	99999
Форма социального обслуживания	formSocServ	Полустационарное социальное обслуживание
Год	yearReport	2021
Месяц	monthReport	Январь

Рисунок 5.5 – Просмотр извлеченных данных (вкладка с одиночными значениями)



Код	Описание	Кол-во оказанных услуг (всего)	Кол-во оказанных услуг сверх объемов, определяемых стандартом
	Социально-бытовые	6	0
2702	2.1.01. Предоставление помещений для организации социально-реабилитационных и социокультурных мероприятий	2	
2704	2.1.03. Предоставление в пользование мебели согласно утвержденным нормативам	2	
2705	2.1.04. Обеспечение книгами, журналами, газетами, настольными играми, иным инвентарем для организации досуга	2	
2708	2.1.07. Социальный патронаж		
	Социально-медицинские услуги	2	0
2718	2.2.06. Проведение занятий с использованием методов адаптивной физической культуры	2	
2719	2.2.07. Проведение оздоровительных мероприятий, в том числе по формированию здорового образа жизни		
	Социально-психологические услуги	2	0
2723	2.3.01. Содействие в получении психологической помощи		
2724	2.3.02. Проведение бесед, направленных на формирование у получателя социальных услуг позитивного психологического состояния, поддержание активного образа жизни	2	
2970	2.3.03. Социально-психологическая диагностика		
2971	2.3.04. Социально-психологическая коррекция		
2972	2.3.05. Социально-психологическое консультирование		
2726	2.3.06. Социально-психологический патронаж		
	Социально-педагогические услуги	2	0
2729	2.4.01. Организация досуга		
2973	2.4.02. Социально-педагогическая диагностика		
2974	2.4.03. Социально-педагогическая коррекция	2	
2975	2.4.04. Социально-педагогическое консультирование		
2976	2.4.05. Обучение родственников и (или) законных представителей практическим навыкам общего ухода за получателями социальных услуг, нуждающихся в постоянном постороннем уходе		
2977	2.4.06. Организация помощи законным представителям детей с ограниченными возможностями здоровья, в том числе детей инвалидов, в обучении детей навыкам самообслуживания, общения и контроля, навыкам поведения в быту и общественных местах		
2731	2.4.07. Социально-педагогический патронаж		
2732	2.4.08. Содействие в получении образования		
	Социально-трудовые услуги	0	0
2735	2.5.01. Услуги, связанные с социально-трудовой реабилитацией		
2736	2.5.02. Содействие в трудоустройстве		
2737	2.5.03. Содействие в профессиональной ориентации		
	Социально-правовые услуги	0	0
	Услуги в целях повышения коммуникативного потенциала получателей социальных услуг, имеющих ограничения жизнедеятельности, в том числе детей-инвалидов	0	0
2978	2.7.01. Обучение навыкам самообслуживания, общения и контроля, навыкам поведения в быту и общественных местах		
2745	2.7.02. Проведение социально-реабилитационных мероприятий в соответствии с индивидуальными программами реабилитации или абилитации инвалидов, в том числе детей-инвалидов		
2749	2.7.06. Оказание помощи в обучении навыкам компьютерной грамотности		
	ИТОГО:	12	0

Рисунок 5.6 – Просмотр извлеченных данных (вкладка с табличными значениями)

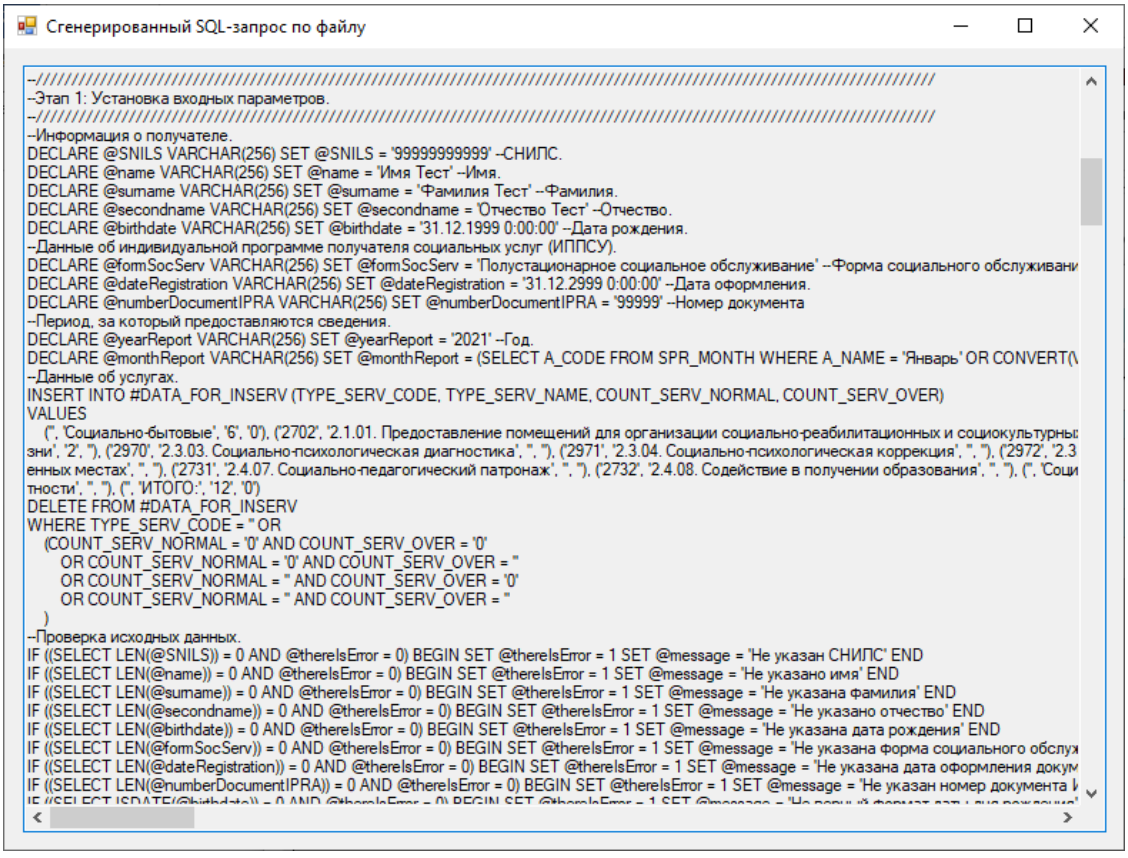


Рисунок 5.7 – Просмотр сгенерированного запроса

После прочтения всех файлов в папке или одиночного файла, откроется доступ к исполнению успешно сгенерированных SQL-запросов. В поле статуса будет информация об успешном выполнении запроса, и при наличии ошибки, в поле дополнительной информации будет информация об ошибке.

ЗАКЛЮЧЕНИЕ

В рамках данной курсовой работы была разработана программа для экспорта данных из Excel-файла в базу данных.

Обстоятельством для разработки послужило то, что у имеющихся аналогов отсутствует нужный функционал для решения поставленной задачи. Например, у мастера импорта и экспорта SQL-сервер отсутствует пакетная обработка файлов, а также ограничение к шаблону файла: на странице должна располагаться одна таблица; заголовки столбцов должны располагаться в первой строке файла; экспорт может осуществляться только в отдельную таблицу. Решение проблемы с помощью макросов из Excel имеет так же ряд недостатков: отсутствие пользовательского интерфейса и необходимость установки дополнительных средств, чтобы макросы можно было запускать.

В ходе выполнения курсового проекта был разработан класс-парсер дескриптора Excel-файла, который собирает информацию об объектах, располагаемых в Excel-файле. По данной информации, которая помогает найти область считывания данных, класс-парсер Excel-файла извлекает данные из файла. Далее извлеченные данные вставляются классом-генератором запросов в специальные места SQL-запроса, которые отмечены специальными символами и кодами. Сгенерированный SQL-запрос отправляется на исполнение классом-исполнителем SQL-запросов.

Результатом работы является программа, которая имеет пакетную обработку файлов, шаблон которых можно задать с помощью дескриптора Excel-файла, и SQL-запрос для которых можно задать.

СПИСОК ЛИТЕРАТУРЫ

1. Herbert S. C#: The Complete Reference [Текст] / Herbert S. – McGraw-Hill Osborne Media, 2010. – 949 с.
2. Weisfeld M. The Object-Oriented Thought Process [Текст] / Weisfeld M. – Addison-Wesley, 2009. – 304 с.
3. Автоматизация рутины в Microsoft Excel при помощи VBA [Электронный ресурс]. URL: https://habr.com/ru/post/112080/
4. Средство импорта и экспорта данных в Microsoft SQL Server [Электронный ресурс]. URL: https://info-comp.ru/sisadminst/357-import-export-data-mssql.html
5. Паттерны/шаблоны проектирования [Электронный ресурс]. URL: https://refactoring.guru/ru/design-patterns
6. 90 рекомендаций по стилю написания программ C++ [Электронный ресурс]. URL: https://habr.com/ru/post/172091/
7. 10 приемов, разрушающих хрупкую красоту кода [Электронный ресурс]. URL: https://habr.com/ru/post/59570/

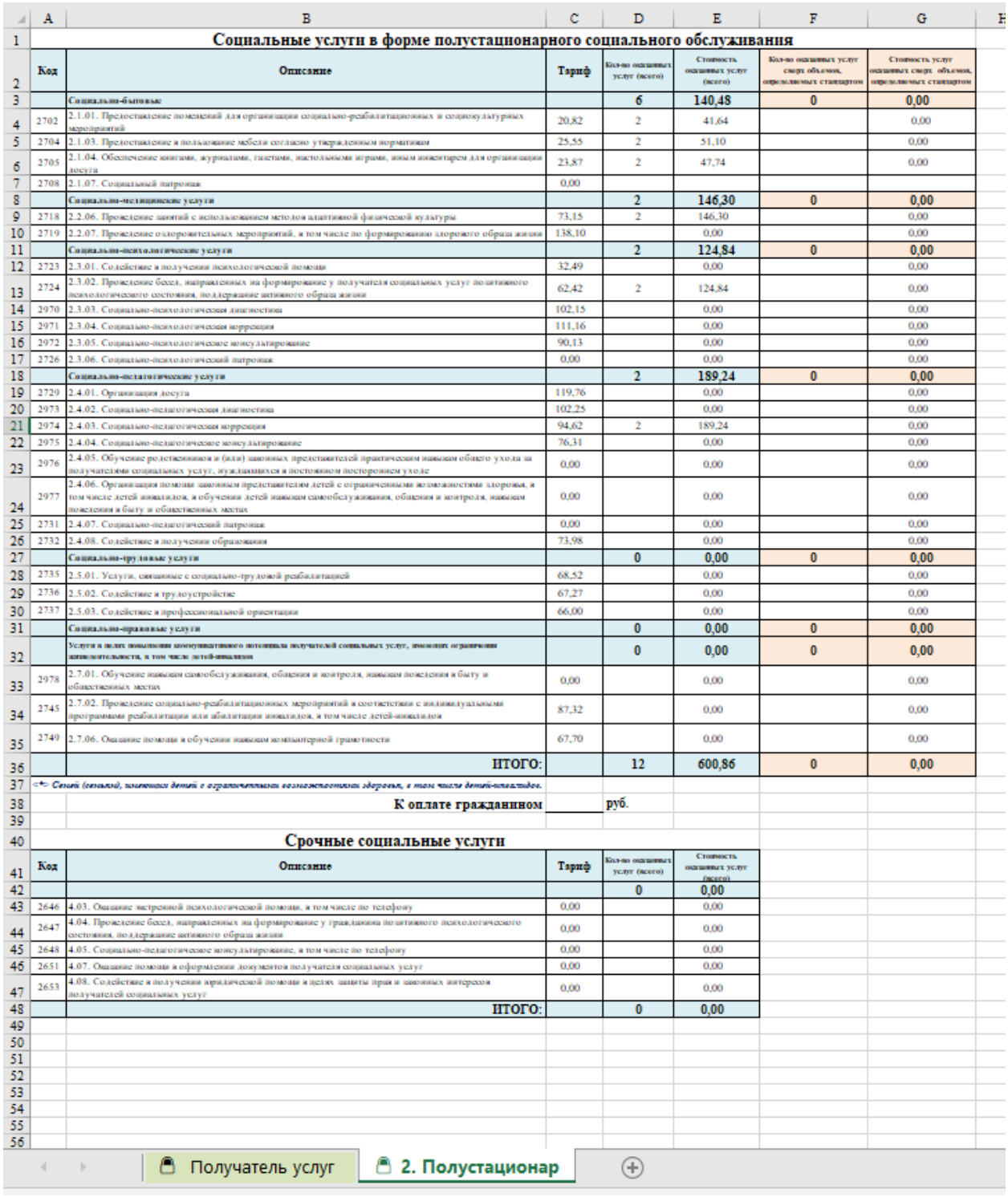
ПРИЛОЖЕНИЕ А

Шаблон Excel-файла

	A	B	C	D
1	**Поставщик социальных услуг:**	**Поставщик социальных услуг Тест**		
2	Регистрационный номер поставщика в Реестре поставщиков:		**999**	
3	Период за который предоставляются сведения		Год	2021
4			Месяц	Январь
5				
6	**Данные о получателе социальных услуг (ПСУ)**			
7	Номер личного дела		СНИЛС	99999999999
8	Фамилия	Фамилия Тест		
9	Имя	Имя Тест	Пол	М
10	Отчество	Отчество Тест	Дата рождения	31.12.1999
11				
12	**Адрес регистрации ПСУ**			
13	Код КЛАДР (если известен)		Номер дома	999
14	Район субъекта федерации		Корпус	
15	Населенный пункт	Населенный пункт Тест	Квартира	999
16	Улица	Улица Тест	Комната	
17	Телефон	Телефон тест		
18				
19	**Документ удостоверяющий личность**			
20	Вид удостоверяющего документа	Паспорт гражданина России		
21	Серия документа	Серия Тест		
22	Номер документа	Номер Тест		
23	Дата выдачи документа	31.12.1999		
24	Организация, выдавшая документ	Организация Тест		
25				
26	**Данные об индивидуальной программе получателя социальных услуг (ИППСУ)**			
27	Дата оформления	31.12.2999		
28	Номер документа	99999		
29				
30	**Данные о договоре (доп.соглашении)**			
31	Дата обращения	31.12.1999		
32	Форма социального обслуживания	Полустационарное социальное обслуживание		
33	Тип (договор/доп.соглашение)	Договор		
34	Номер договора	999		
35	Дата договора	31.12.1999		
36	Дата окончания действия договора	31.12.2999		
37	Местность распространения договора	Городская местность		
38	***Если тип "Дополнительное соглашение к договору":***			
39	Номер документа			
40	Дата документа			
41	Дата окончания действия			
42				
43	**Сумма по договору**			
44				
45				

Получатель услуг | 2. Полустационар

Страница с информацией о получателе



	A	B	C	D	E	F	G
1	**Социальные услуги в форме полустационарного социального обслуживания**						
2	**Код**	**Описание**	**Тариф**	**Кол-во оказанных услуг (всего)**	**Стоимость оказанных услуг (всего)**	**Кол-во оказанных услуг сверх объемов, определяемых стандартом**	**Стоимость услуг оказанных сверх объемов, определяемых стандартом**
3		**Социально-бытовые**		**6**	**140,48**	**0**	**0,00**
4	2702	2.1.01. Предоставление помещений для организации социально-реабилитационных и социокультурных мероприятий	20,82	2	41,64		0,00
5	2704	2.1.03. Предоставление и пользование мебели согласно утвержденным нормативам	25,55	2	51,10		0,00
6	2705	2.1.04. Обеспечение книгами, журналами, газетами, настольными играми, иным инвентарем для организации досуга	23,87	2	47,74		0,00
7	2708	2.1.07. Социальный патронаж	0,00				
8		**Социально-медицинские услуги**		**2**	**146,30**	**0**	**0,00**
9	2718	2.2.06. Проведение занятий с использованием методов адаптивной физической культуры	73,15	2	146,30		0,00
10	2719	2.2.07. Проведение оздоровительных мероприятий, в том числе по формированию здорового образа жизни	138,10		0,00		0,00
11		**Социально-психологические услуги**		**2**	**124,84**	**0**	**0,00**
12	2723	2.3.01. Содействие в получении психологической помощи	32,49		0,00		0,00
13	2724	2.3.02. Проведение бесед, направленных на формирование у получателя социальных услуг позитивного психологического состояния, поддержание активного образа жизни	62,42	2	124,84		0,00
14	2970	2.3.03. Социально-психологическая диагностика	102,15		0,00		0,00
15	2971	2.3.04. Социально-психологическая коррекция	111,16		0,00		0,00
16	2972	2.3.05. Социально-психологическое консультирование	90,13		0,00		0,00
17	2726	2.3.06. Социально-психологический патронаж	0,00		0,00		0,00
18		**Социально-педагогические услуги**		**2**	**189,24**	**0**	**0,00**
19	2729	2.4.01. Организация досуга	119,76		0,00		0,00
20	2973	2.4.02. Социально-педагогическая диагностика	102,25		0,00		0,00
21	2974	2.4.03. Социально-педагогическая коррекция	94,62	2	189,24		0,00
22	2975	2.4.04. Социально-педагогическое консультирование	76,31		0,00		0,00
23	2976	2.4.05. Обучение родственников и (или) законных представителей практическим навыкам общего ухода за получателями социальных услуг, нуждающихся в постоянном постороннем уходе	0,00		0,00		0,00
24	2977	2.4.06. Организация помощи законным представителям детей с ограниченными возможностями здоровья, в том числе детей инвалидов, в обучении детей навыкам самообслуживания, общения и контроля, навыкам поведения в быту и общественных местах	0,00		0,00		0,00
25	2731	2.4.07. Социально-педагогический патронаж	0,00		0,00		0,00
26	2732	2.4.08. Содействие в получении образования	73,98		0,00		0,00
27		**Социально-трудовые услуги**		**0**	**0,00**	**0**	**0,00**
28	2735	2.5.01. Услуги, связанные с социально-трудовой реабилитацией	68,52		0,00		0,00
29	2736	2.5.02. Содействие в трудоустройстве	67,27		0,00		0,00
30	2737	2.5.03. Содействие в профессиональной ориентации	66,00		0,00		0,00
31		**Социально-правовые услуги**		**0**	**0,00**	**0**	**0,00**
32		**Услуги в целях повышения коммуникативного потенциала получателей социальных услуг, имеющих ограничения жизнедеятельности, в том числе детей-инвалидов**		**0**	**0,00**	**0**	**0,00**
33	2978	2.7.01. Обучение навыкам самообслуживания, общения и контроля, навыкам поведения в быту и общественных местах	0,00		0,00		0,00
34	2745	2.7.02. Проведение социально-реабилитационных мероприятий в соответствии с индивидуальными программами реабилитации или абилитации инвалидов, в том числе детей-инвалидов	87,32		0,00		0,00
35	2749	2.7.06. Оказание помощи в обучении навыкам компьютерной грамотности	67,70		0,00		0,00
36		**ИТОГО:**		**12**	**600,86**	**0**	**0,00**
37	*<= Сений (взамен), имеющих детей с ограниченными возможностями здоровья, в том числе детей-инвалидов.*						
38		**К оплате гражданином**		**руб.**			
39							
40	**Срочные социальные услуги**						
41	**Код**	**Описание**	**Тариф**	**Кол-во оказанных услуг (всего)**	**Стоимость оказанных услуг (всего)**		
42				**0**	**0,00**		
43	2646	4.03. Оказание экстренной психологической помощи, в том числе по телефону	0,00		0,00		
44	2647	4.04. Проведение бесед, направленных на формирование у гражданина позитивного психологического состояния, поддержание активного образа жизни	0,00		0,00		
45	2648	4.05. Социально-педагогическое консультирование, в том числе по телефону	0,00		0,00		
46	2651	4.07. Оказание помощи в оформлении документов получателя социальных услуг	0,00		0,00		
47	2653	4.08. Содействие в получении юридической помощи в целях защиты прав и законных интересов получателей социальных услуг	0,00		0,00		
48		**ИТОГО:**		**0**	**0,00**		
49							
50							
51							
52							
53							
54							
55							
56							

Получатель услуг | 2. Полустационар

Страница с информацией об оказанных услугах

ПРИЛОЖЕНИЕ Б

Дескриптор шаблона Excel-файла

<singleValue>
SHEET_NUMBER: 1;
SECTION_NAME: "Данные о получателе социальных услуг (ПСУ)";
SECTION_BOTTOM_LEFT: 1;
SECTION_BOTTOM_RIGHT: 1;
FIELD: СНИЛС;
CODE: SNILS;
OFFEST_COLUMN: 1;
</singleValue>
<singleValue>
SHEET_NUMBER: 1;
SECTION_NAME: "Данные о получателе социальных услуг (ПСУ)";
SECTION_BOTTOM_LEFT: 1;
SECTION_BOTTOM_RIGHT: 1;
FIELD: Имя;
CODE: name;
OFFEST_COLUMN: 1;
</singleValue>
<singleValue>
SHEET_NUMBER: 1;
SECTION_NAME: "Данные о получателе социальных услуг (ПСУ)";
SECTION_BOTTOM_LEFT: 1;
SECTION_BOTTOM_RIGHT: 1;
FIELD: Фамилия;
CODE: surname;
OFFEST_COLUMN: 1;
</singleValue>
<singleValue>
SHEET_NUMBER: 1;
SECTION_NAME: "Данные о получателе социальных услуг (ПСУ)";
SECTION_BOTTOM_LEFT: 1;
SECTION_BOTTOM_RIGHT: 1;
FIELD: Отчество;
CODE: secondname;
OFFEST_COLUMN: 1;
</singleValue>
<singleValue>
SHEET_NUMBER: 1;
SECTION_NAME: "Данные о получателе социальных услуг (ПСУ)";
SECTION_BOTTOM_LEFT: 1;
SECTION_BOTTOM_RIGHT: 1;
FIELD: "Дата рождения";
CODE: birthdate;
OFFEST_COLUMN: 1;
</singleValue>
<singleValue>
SHEET_NUMBER: 1;
SECTION_NAME: "Данные об индивидуальной программе получателя социальных услуг (ИППСУ)";
SECTION_BOTTOM_LEFT: 1;
SECTION_BOTTOM_RIGHT: 1;
FIELD: "Дата оформления";
CODE: dateRegistration;
OFFEST_COLUMN: 1;
</singleValue>
<singleValue>
SHEET_NUMBER: 1;
SECTION_NAME: "Данные об индивидуальной программе получателя социальных услуг (ИППСУ)";
SECTION_BOTTOM_LEFT: 1;
SECTION_BOTTOM_RIGHT: 1;
FIELD: "Номер документа ";
CODE: numberDocumentIPRA;
OFFEST_COLUMN: 1;
</singleValue>
<singleValue>
SHEET_NUMBER: 1;
SECTION_NAME: "Данные о договоре (доп.соглашении)";
SECTION_BOTTOM_LEFT: 1;
SECTION_BOTTOM_RIGHT: 1;
FIELD: "Форма социального обслуживания";

```
            CODE: formSocServ;
            OFFEST_COLUMN: 1;
    </singleValue>
    <singleValue>
            SHEET_NUMBER: 1;
            SECTION_NAME: "Период за который предоставляются сведения";
            SECTION_BOTTOM_LEFT: 1;
            SECTION_BOTTOM_RIGHT: 1;
            FIELD: "Год";
            CODE: yearReport;
            OFFEST_COLUMN: 1;
    </singleValue>
    <singleValue>
            SHEET_NUMBER: 1;
            SECTION_NAME: "Период за который предоставляются сведения";
            SECTION_BOTTOM_LEFT: 1;
            SECTION_BOTTOM_RIGHT: 1;
            FIELD: "Месяц";
            CODE: monthReport;
            OFFEST_COLUMN: 1;
    </singleValue>
    <table>
            SHEET_NUMBER: 2;
            CODE: "DATA_FOR_INSERV";
            INCLUDE_FINAL_ROW: "1";
            <column>
                    NAME: "Код";
                    CODE: "TYPE_SERV_CODE";
            </column>
            <column>
                    NAME: "Описание";
                    CODE: "TYPE_SERV_NAME";
                    FINAL_CELL: "ИТОГО:";
            </column>
            <column>
                    NAME: "Кол-во оказанных услуг (всего)";
                    CODE: "COUNT_SERV_NORMAL";
            </column>
            <column>
                    NAME: "Кол-во оказанных услуг сверх объемов, определяемых стандартом";
                    CODE: "COUNT_SERV_OVER";
            </column>
    </table>
```

ПРИЛОЖЕНИЕ В

SQL-запрос для вставки данных

```
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 0: Инициализация.
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Переменные.
DECLARE @identifiedPerson INT --Идентифицированный человек.
DECLARE @identifiedSocServ INT --Идентифицированное социальное обслуживание.
DECLARE @yearReport_INT INT --Год, конвертированный в число.
DECLARE @monthReport_INT INT --Месяц, конвертированный в число.
DECLARE @startDate DATE --Дата -начала периода.
DECLARE @endDate DATE --Дата конца периода.
DECLARE @thereIsError INT = 0 --Флаг наличия ошибки.
DECLARE @message VARCHAR(256)= 'Успешно' --Сообщение.
--Удаление временных таблиц.
IF OBJECT_ID('tempdb..#FOUND_PEOPLE') IS NOT NULL BEGIN DROP TABLE #FOUND_PEOPLE END --Найденные люди по
входным данным.
IF OBJECT_ID('tempdb..#FOUND_SOC_SERV') IS NOT NULL BEGIN DROP TABLE #FOUND_SOC_SERV END --Найденные
социальные обслуживания по входным данным.
IF OBJECT_ID('tempdb..#DATA_FOR_INSERV') IS NOT NULL BEGIN DROP TABLE #DATA_FOR_INSERV END --Данные для
вставки.
IF OBJECT_ID('tempdb..#FOUND_AGREGATION') IS NOT NULL BEGIN DROP TABLE #FOUND_AGREGATION END --Найденные
агрегации услуг.
--Создание временных таблиц.
CREATE TABLE #FOUND_PEOPLE (
   PERSONOUID INT, --Идентификатор личного дела.
)
CREATE TABLE #FOUND_SOC_SERV (
   SOC_SERV_OUID INT, --Идентификатор социального обслуживания.
)
CREATE TABLE #DATA_FOR_INSERV (
   TYPE_SERV_CODE VARCHAR(256), --Код услуги
   TYPE_SERV_NAME VARCHAR(256), --Наименование услуги
   COUNT_SERV_NORMAL INT, --Количество оказанных услуг в месяц в рамках норматива
   COUNT_SERV_OVER INT --Количество оказанных услуг в месяц сверх норматива
)
CREATE TABLE #FOUND_AGREGATION (
   TYPE_SERV_CODE VARCHAR(256), --Код услуги
   TYPE_SERV_NAME VARCHAR(256), --Наименование услуги
   COUNT_SERV_NORMAL INT, --Количество оказанных услуг в месяц в рамках норматива
   COUNT_SERV_OVER INT, --Количество оказанных услуг в месяц сверх норматива
   SOC_SERV_AGR_OUID INT, --Идентификатор агрегации услуги.
   CONDITION VARCHAR(256)--Условия оказания социальных услуг.
)
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 1: Установка входных параметров.
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Информация о получателе.
DECLARE @SNILS VARCHAR(256) SET @SNILS = '#SNILS#' --СНИЛС.
DECLARE @name VARCHAR(256) SET @name = '#name#' --Имя.
DECLARE @surname VARCHAR(256) SET @surname = '#surname#' --Фамилия.
DECLARE @secondname VARCHAR(256) SET @secondname = '#secondname#' --Отчество.
DECLARE @birthdate VARCHAR(256) SET @birthdate = '#birthdate#' --Дата рождения.
--Данные об индивидуальной программе получателя социальных услуг (ИППСУ).
DECLARE @formSocServ VARCHAR(256) SET @formSocServ = '#formSocServ#' --Форма социального обслуживания.
DECLARE @dateRegistration VARCHAR(256) SET @dateRegistration = '#dateRegistration#' --Дата оформления.
DECLARE @numberDocumentIPRA VARCHAR(256) SET @numberDocumentIPRA = '#numberDocumentIPRA#' --Номер документа
--Период, за который предоставляются сведения.
DECLARE @yearReport VARCHAR(256) SET @yearReport = '#yearReport#' --Год.
DECLARE @monthReport VARCHAR(256) SET @monthReport = (SELECT A_CODE FROM SPR_MONTH WHERE A_NAME =
'#monthReport#' OR CONVERT(VARCHAR, A_CODE) = '#monthReport#') --Месяц.
--Данные об услугах.
INSERT INTO #DATA_FOR_INSERV (TYPE_SERV_CODE, TYPE_SERV_NAME, COUNT_SERV_NORMAL, COUNT_SERV_OVER)
VALUES
   <DATA_FOR_INSERV ('#TYPE_SERV_CODE#', '#TYPE_SERV_NAME#', '#COUNT_SERV_NORMAL#', '#COUNT_SERV_OVER#')>
DELETE FROM #DATA_FOR_INSERV
WHERE TYPE_SERV_CODE = '' OR
   (COUNT_SERV_NORMAL = '0' AND COUNT_SERV_OVER = '0'
      OR COUNT_SERV_NORMAL = '0' AND COUNT_SERV_OVER = ''
      OR COUNT_SERV_NORMAL = '' AND COUNT_SERV_OVER = '0'
      OR COUNT_SERV_NORMAL = '' AND COUNT_SERV_OVER = ''
      OR COUNT_SERV_NORMAL = '0' AND COUNT_SERV_OVER = '0'
   )
```

```
--Проверка исходных данных.
IF ((SELECT LEN(@SNILS)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не указан СНИЛС' END
IF ((SELECT LEN(@name)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не указано имя' END
IF ((SELECT LEN(@surname)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не указана фамилия' END
IF ((SELECT LEN(@secondname)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не указано отчество' END
IF ((SELECT LEN(@birthdate)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не указана дата рождения'
END
IF ((SELECT LEN(@formSocServ)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не указана форма
социального обслуживания' END
IF ((SELECT LEN(@dateRegistration)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не указана дата
оформления документа ИППСУ' END
IF ((SELECT LEN(@numberDocumentIPRA)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не указан
номер документа ИППСУ' END
IF ((SELECT ISDATE(@birthdate)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не верный формат даты
дня рождения' END
IF ((SELECT ISDATE(@dateRegistration)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не верный формат
даты оформления ИППСУ' END
IF ((SELECT ISNUMERIC(@yearReport)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не верный формат
года' END
IF ((SELECT ISNUMERIC(@monthReport)) = 0 AND @thereIsError = 0) BEGIN SET @thereIsError = 1 SET @message = 'Не верный
формат месяца' END
IF ((SELECT COUNT(*) FROM #DATA_FOR_INSERV) = 0) BEGIN SET @thereIsError = 1 SET @message = 'Нет не нулевых услуг' END
--Конвертация исходных данных.
IF (@thereIsError = 0) BEGIN
   SET @yearReport_INT = CONVERT(INT, @yearReport)
   SET @monthReport_INT = CONVERT(INT, @monthReport)
   SET @startDate = CAST(@yearReport + '-' + @monthReport + '-01' AS DATE)
   SET @endDate = DATEADD(MONTH, ((YEAR(@startDate) - 1900) * 12) + MONTH(@startDate), -1)
END
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 2: Идентификация человека.
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
IF (@thereIsError = 0) BEGIN
   --Выбор людей, удовлетворяющих условиям.
   INSERT #FOUND_PEOPLE (PERSONOUID)
   SELECT DISTINCT
      personalCard.OUID AS PERSONOUID
   FROM WM_PERSONAL_CARD personalCard --Личное дело гражданина.
   ----Фамилия.
      LEFT JOIN SPR_FIO_SURNAME fioSurname
         ON fioSurname.OUID = personalCard.SURNAME
   ----Имя.
      LEFT JOIN SPR_FIO_NAME fioName
         ON fioName.OUID = personalCard.A_NAME
   ----Отчество.
      LEFT JOIN SPR_FIO_SECONDNAME fioSecondname
         ON fioSecondname.OUID = personalCard.A_SECONDNAME
   WHERE personalCard.A_STATUS = 10 --Статус личного дела в БД "Действует".
      AND personalCard.A_PCSTATUS = 1 --Статус личного дела "Действует".
      AND personalCard.A_SNILS = @SNILS --СНИЛС совпадает.
      AND ISNULL(personalCard.A_NAME_STR, fioName.A_NAME) = @name --Имя совпадает.
      AND ISNULL(personalCard.A_SURNAME_STR, fioSurname.A_NAME) = @surname --Фамилия совпадает.
      AND CONVERT(DATE, personalCard.BIRTHDATE) = CONVERT(DATE, @birthdate) --Дата рожденяи совпадает.
      AND ISNULL(personalCard.A_SECONDNAME_STR, fioSecondname.A_NAME) = @secondname --Отчество совпадает.
   --Подсчет людей.
   DECLARE @countFoundPeople INT
   SET @countFoundPeople = (SELECT COUNT(*) FROM #FOUND_PEOPLE)
   --Результат 2 этапа.
   IF (@countFoundPeople > 1) BEGIN
      SET @thereIsError = 1
      SET @message = 'Найдено более одного человека, удовлетворяющего условиям'
   END
   ELSE IF (@countFoundPeople = 0) BEGIN
      SET @thereIsError = 1
      SET @message = 'Не найден человек, удовлетворяющий условиям'
   END
   ELSE BEGIN
      SET @identifiedPerson = (SELECT TOP 1 PERSONOUID FROM #FOUND_PEOPLE)
   END
END
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 3: Идентификация социального обслуживания.
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
IF (@thereIsError = 0) BEGIN
   --Выбор назначений на социальной обслуживание.
   INSERT #FOUND_SOC_SERV(SOC_SERV_OUID)
   SELECT DISTINCT
```

```
    socServ.OUID AS SOC_SERV_OUID
  FROM ESRN_SOC_SERV socServ --Назначение социального обслуживания.
  ----Период предоставления МСП.
    INNER JOIN SPR_SOCSERV_PERIOD period
      ON period.A_STATUS = 10 --Статус в БД "Действует".
        AND period.A_SERV = socServ.OUID --Связка с назначением.
        AND @yearReport_INT BETWEEN YEAR(period.STARTDATE) AND ISNULL(YEAR(period.A_LASTDATE), 3000) --Год
отчета входит в период действия назначения.
        AND (@yearReport_INT <> YEAR(period.STARTDATE) AND @yearReport_INT <> ISNULL(YEAR(period.A_LASTDATE),
3000) --Год отчета не равен крайнему.
          OR @yearReport_INT = YEAR(period.STARTDATE) AND @monthReport_INT >= MONTH(period.STARTDATE) --Или равен
начальному, но месяц позже начала.
          OR @yearReport_INT = YEAR(period.A_LASTDATE) AND @monthReport_INT <= ISNULL(MONTH(period.A_LASTDATE),
12) --Или равен конечному, но месяц раньше конца.
        )
  ----Индивидуальная программа.
    INNER JOIN INDIVID_PROGRAM individProgram
      ON individProgram.A_OUID = socServ.A_IPPSU
        AND individProgram.A_STATUS = 10 --Статус индивидуальной программы в БД "Действует".
  ----Действующие документы.
    INNER JOIN WM_ACTDOCUMENTS actDocuments
      ON actDocuments.OUID = individProgram.A_DOC
        AND actDocuments.A_STATUS = 10 --Статус документа в БД "Действует".
        AND actDocuments.DOCUMENTSNUMBER = @numberDocumentIPRA --Номер документа совпадает с требуемым.
  ----Форма социального обслуживания.
    INNER JOIN SPR_FORM_SOCSERV formSocServ
      ON formSocServ.A_OUID = individProgram.A_FORM_SOCSERV
        AND formSocServ.A_NAME = @formSocServ --Совпадает с формой отчета.
  ----Органы социальной защиты.
    INNER JOIN SPR_ORG_BASE organization
      ON organization.OUID = socServ.A_ORGNAME
  WHERE socServ.A_STATUS = 10 --Статус назначения в БД "Действует".
    AND socServ.A_PERSONOUID = @identifiedPerson --Льготодержатель - идентифицированный человек.
  --Подсчет назначений.
  DECLARE @countFoundSocServ INT
  SET @countFoundSocServ = (SELECT COUNT(*) FROM #FOUND_SOC_SERV)
  --Результат 3 этапа.
  IF (@countFoundSocServ > 1) BEGIN
    SET @thereIsError = 1
    SET @message = 'Найдено более одного социального обслуживания, удовлетворяющего условиям'
  END
  ELSE IF (@countFoundSocServ = 0) BEGIN
    SET @thereIsError = 1
    SET @message = 'Не найдено социальное обслуживание, удовлетворяющего условиям'
  END
  ELSE BEGIN
    SET @identifiedSocServ = (SELECT TOP 1 SOC_SERV_OUID FROM #FOUND_SOC_SERV)
  END
END
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 4: Идентификация агрегаций по услугам.
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
IF (@thereIsError = 0) BEGIN
  --Выбор агрегации по услуге.
  INSERT INTO #FOUND_AGREGATION(TYPE_SERV_CODE, TYPE_SERV_NAME, COUNT_SERV_NORMAL,
COUNT_SERV_OVER, SOC_SERV_AGR_OUID, CONDITION)
  SELECT DISTINCT
    forInsert.TYPE_SERV_CODE        AS TYPE_SERV_CODE,
    forInsert.TYPE_SERV_NAME        AS TYPE_SERV_NAME,
    forInsert.COUNT_SERV_NORMAL     AS COUNT_SERV_NORMAL,
    forInsert.COUNT_SERV_OVER       AS COUNT_SERV_OVER,
    socServAGR.A_ID                 AS SOC_SERV_AGR_OUID,
    condition.A_COND_SOC_SERV
  FROM ESRN_SOC_SERV socServ --Назначение социального обслуживания.
  ----Агрегация по социальной услуге.
    INNER JOIN WM_SOC_SERV_AGR socServAGR
      ON socServAGR.ESRN_SOC_SERV = socServ.OUID
        AND socServAGR.A_STATUS = 10 --Статус агрегации в БД "Действует".
  ----Тарифы на социальные услуги.
    INNER JOIN SPR_TARIF_SOC_SERV socServTarif
      ON socServTarif.A_ID = socServAGR.A_SOC_SERV
        AND socServTarif.A_STATUS = 10 --Статус тарифа в БД "Действует".
  ----Социальные услуги.
    INNER JOIN SPR_SOC_SERV typeSocServ
      ON typeSocServ.OUID = socServTarif.A_SOC_SERV
        AND typeSocServ.A_STATUS = 10 --Статус социальной услуги в БД "Действует".
  ----Данные для вставки.
```

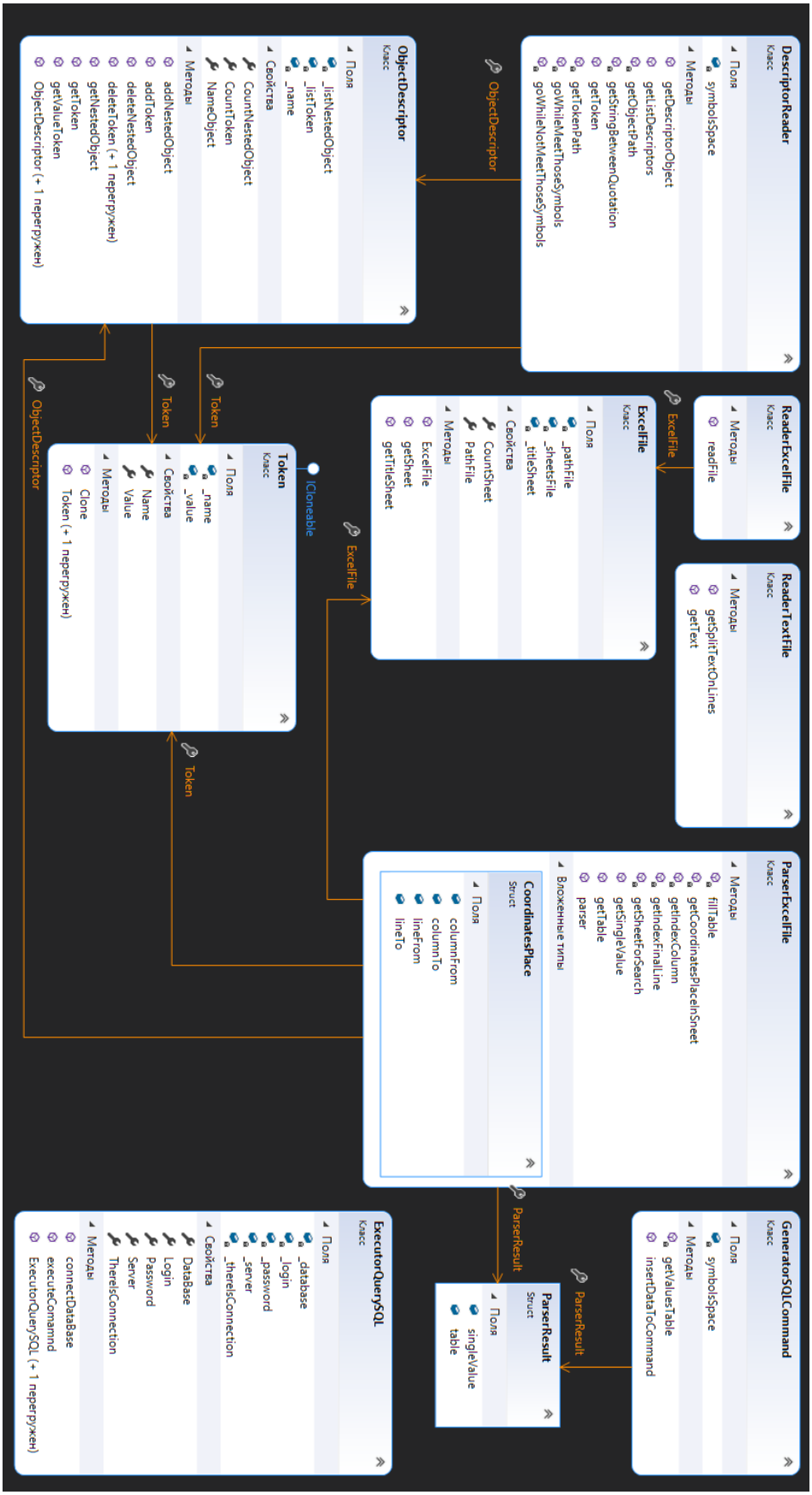
```
      INNER JOIN #DATA_FOR_INSERV forInsert
         ON CHARINDEX(forInsert.TYPE_SERV_CODE, typeSocServ.A_CODE) > 0
         --AND forInsert.TYPE_SERV_NAME = typeSocServ.A_NAME
      INNER JOIN WM_COND_SOC_SERV_ONE condition
         ON condition.A_SOC_SERV_AGR = socServAGR.A_ID --Агрегация отчета.
            AND condition.A_STATUS = 10
            AND (@yearReport_INT BETWEEN YEAR(A_STARTDATE) AND ISNULL(YEAR(A_LASTDATE), 9999) --Год отчета входит
в период действия назначения.
            AND (@yearReport_INT <> YEAR(A_STARTDATE) AND @yearReport_INT <> ISNULL(YEAR(A_LASTDATE), 9999) --Год
отчета не равен крайнему.
               OR @yearReport_INT = YEAR(A_STARTDATE) AND @monthReport_INT >= MONTH(A_STARTDATE) --Или равен
начальному, но месяц позже начала.
               OR @yearReport_INT = YEAR(A_LASTDATE) AND @monthReport_INT <= MONTH(A_LASTDATE) --Или равен
конечному, но месяц раньше конца.
            )
         )
   WHERE socServ.A_STATUS = 10 --Статус назначения в БД "Действует".
      AND socServ.OUID = @identifiedSocServ --Идентифицированное назначение.
   --Подсчет услуг.
   DECLARE @countTypeServForInsert INT
   SET @countTypeServForInsert = (SELECT COUNT(*) FROM #DATA_FOR_INSERV)
   DECLARE @countFoundTypeServ INT
   SET @countFoundTypeServ = (SELECT COUNT(*) FROM #FOUND_AGREGATION)
   --Результат 4 этапа.
   IF (@countTypeServForInsert <> @countFoundTypeServ) BEGIN
      SET @thereIsError = 1
      SET @message = 'Не все услуги были найдены в социальном обслуживании'
   END
END
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 4: Проверка введенных данных.
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
IF (@thereIsError = 0) BEGIN
   --Проверка отсутствия уже введенных данных.
   DECLARE @alreadyThereCount INT
   SET @alreadyThereCount = (
      SELECT
         COUNT(DISTINCT cosSocServ.A_ID)
      FROM #FOUND_AGREGATION foundAgregation
         INNER JOIN WM_COST_SOC_SERV cosSocServ
            ON cosSocServ.A_AGR_SOC_SERV = foundAgregation.SOC_SERV_AGR_OUID
               AND cosSocServ.A_STATUS = 10
               AND cosSocServ.A_DATE_START = @startDate
               AND cosSocServ.A_DATE_LAST = @endDate
      )
   IF (@alreadyThereCount = @countTypeServForInsert) BEGIN
      SET @thereIsError = 1
      SET @message = 'Данные уже есть за данный период'
   END
   IF (@alreadyThereCount <> @countTypeServForInsert AND @alreadyThereCount <> 0) BEGIN
      SET @thereIsError = 1
      SET @message = 'Частично данные уже есть за данный период'
   END
END
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 5: Вставка стоимости и количество оказанных социальных услуг.
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
IF (@thereIsError = 0) BEGIN
   INSERT INTO WM_COST_SOC_SERV(A_EMPLOYEE, A_EDITOWNER, A_STATUS, GUID, TS, SYSTEMCLASS,
A_CREATEDATE, A_CROWNER, A_AGR_SOC_SERV, A_SUM_SOC_SERV_PERIOD, A_DATE_START, A_DATE_LAST,
A_ACT_VOLUME, A_COMMENT, A_ACT_EXCESS_QUANT, A_COST_DOP_SOC_SERV, A_ACT_QUANT_NORM)
   SELECT
      CAST(NULL AS INT)          AS A_EMPLOYEE,          --Сотрудники.
      CAST(NULL AS INT)          AS A_EDITOWNER,         --Изменил.
      10                         AS A_STATUS,            --Статус.
      NEWID()                    AS GUID,                --Глобальный идентификатор.
      GETDATE()                  AS TS,                  --Дата модификации.
      10284209                   AS SYSTEMCLASS,         --Класс объекта.
      GETDATE()                  AS A_CREATEDATE,        --Дата создания.
      10314303                   AS A_CROWNER,           --Автор.
      SOC_SERV_AGR_OUID          AS A_AGR_SOC_SERV,      --Социальная услуга (Агрегация по социальной услуге).
      CAST(NULL AS FLOAT)        AS A_SUM_SOC_SERV_PERIOD, --Сумма по услуге за период, руб.
      @startDate                 AS A_DATE_START,        --Дата начала периода.
      @endDate                   AS A_DATE_LAST,         --Дата окончания периода.
      CAST(NULL AS FLOAT)        AS A_ACT_VOLUME,        --Фактический объем работ на 1 услугу.
      CAST(NULL AS VARCHAR)      AS A_COMMENT,           --Примечание.
      COUNT_SERV_OVER            AS A_ACT_EXCESS_QUANT, --Количество оказанных услуг в месяц сверх норматива.
```

```
    CAST(NULL AS FLOAT)             AS A_COST_DOP_SOC_SERV,    --Стоимость оказанных услуг, руб.
    COUNT_SERV_NORMAL               AS A_ACT_QUANT_NORM        --Количество оказанных услуг в месяц в рамках норматива.
  FROM #FOUND_AGREGATION foundAgregation
END
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 6: Вставка суммы по услуге за календарный месяц.
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
IF (@thereIsError = 0) BEGIN
  INSERT INTO WM_COST_SOC_SERV_MONTH(A_YEAR, A_MONTH, A_SOC_SERV_MONTH, A_SUM_SOC_SERV_MONTH,
A_NORM_EX, A_EDITOWNER, A_TS, A_GUID, A_STATUS, A_CREATEDATE, A_CROWNER, A_AGR_SOC_SERV,
A_SYSTEMCLASS, A_FULL_COST, A_PERCENT_PART_PAY, A_SUM_NORM_EX, A_IS_PART_PAY, A_COND_SOC_SERV,
A_NORMSOCSERV, A_ACT_EXCESS_QUANT)
  SELECT
    @yearReport_INT                 AS A_YEAR,                 --Год.
    @monthReport_INT                AS A_MONTH,                --Месяц.
    COUNT_SERV_NORMAL + COUNT_SERV_OVER AS A_SOC_SERV_MONTH,   --Количество оказанных услуг за месяц (Сумма
норматива и превышения норматива).
    0                               AS A_SUM_SOC_SERV_MONTH,
    CASE COUNT_SERV_OVER
      WHEN 0 THEN 0
      ELSE 1
    END                             AS A_NORM_EX,          --Норматив превышен (Если есть сумма превышения норматива).
    CAST(NULL AS INT)               AS A_EDITOWNER,            --Изменил.
    GETDATE()                       AS A_TS,                   --Дата модификации.
    NEWID()                         AS A_GUID,                 --Глобальный идентификатор.
    10                              AS A_STATUS,               --Статус.
    GETDATE()                       AS A_CREATEDATE,           --Дата создания.
    10314303                        AS A_CROWNER,              --Автор.
    SOC_SERV_AGR_OUID               AS A_AGR_SOC_SERV,         --Социальная услуга (Агрегация социальной услуги)
    10284209                        AS A_SYSTEMCLASS,          --Класс объекта.
    --CASE @condition
    --   WHEN 'free' THEN @costServOver
    --   WHEN 'part' THEN @costServNormal / 50 + @costServOver
    --   WHEN 'full' THEN @costServNormal + @costServOver
    --END
    0                               AS A_FULL_COST,            --Полная стоимость оказанных услуг, руб.
    CASE CONDITION
      WHEN 'part' THEN 50
      ELSE 100
    END                             AS A_PERCENT_PART_PAY,  --Размер частичной оплаты услуг, %
    0                               AS A_SUM_NORM_EX,          --Сумма превышения норматива, руб.
    0                               AS A_IS_PART_PAY,          --Без учета частичной оплаты
    CONDITION                       AS A_COND_SOC_SERV,        --Условие оказания социальных услуг.
    COUNT_SERV_NORMAL               AS A_NORMSOCSERV,          --Количество оказанных услуг в рамках норматива.
    COUNT_SERV_OVER                 AS A_ACT_EXCESS_QUANT --Количество оказанных услуг в месяц сверх норматива.
  FROM #FOUND_AGREGATION
END
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 7: Вставка стоимости всех оказанных услуг за календарный месяц
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
IF (@thereIsError = 0) BEGIN
  --Условия оказания социальных услуг.
  DECLARE @condition VARCHAR(256) SET @condition = (
    SELECT
      A_COND_SOC_SERV
    FROM WM_COND_SOC_SERV WHERE A_SOC_SERV = @identifiedSocServ --Назначение
      AND (@yearReport_INT BETWEEN YEAR(A_STARTDATE) AND ISNULL(YEAR(A_LASTDATE), 9999) --Год отчета входит в
период действия назначения.
        AND (@yearReport_INT <> YEAR(A_STARTDATE) AND @yearReport_INT <> ISNULL(YEAR(A_LASTDATE), 9999) --Год
отчета не равен крайнему.
          OR @yearReport_INT = YEAR(A_STARTDATE) AND @monthReport_INT >= MONTH(A_STARTDATE) --Или равен
начальному, но месяц позже начала.
          OR @yearReport_INT = YEAR(A_LASTDATE) AND @monthReport_INT <= MONTH(A_LASTDATE) --Или равен
конечному, но месяц раньше конца.
        )
    )
  )
  INSERT INTO
WM_FACT_COST_SOC_SERV(TS,SYSTEMCLASS,GUID,A_CREATEDATE,A_CROWNER,A_SERV,A_SUM_PERIOD,A_FACT_PAY,
A_EDITOWNER,A_STATUS,A_PAY_DATE,A_YEAR,A_MONTH,A_FULL_COST_MONTH,A_SUM_PAY,A_PART_PAY,A_SUMM_E
XT_SERV,A_SUM_NORM_EX,A_SUMM_DOG,ESRN_SOC_SERV,A_COND_SOC_SERV,A_COMMENT,A_SUM_PERIOD_BUDGET,
A_FULL_COUNT,A_SOC_COUNT,A_SOC_COST_MONTH,A_DOP_COUNT,A_DOP_COST_MONTH,A_OTHER_COUNT,A_OTHER_
COST_MONTH)
  SELECT
    GETDATE()                       AS TS,                     --Дата модификации
    10284266                        AS SYSTEMCLASS,            --Класс объекта
    NEWID()                         AS GUID,                   --Глобальный идентификатор
```

```
    GETDATE()                       AS A_CREATEDATE,          --Дата создания
    10314303                        AS A_CROWNER,             --Автор
    CAST(NULL AS INT)               AS A_SERV,                --Назначение
    0                               AS A_SUM_PERIOD,          --Всего к оплате гражданином, руб.
    'withoutPay'                    AS A_FACT_PAY,             --Статус оплаты
    CAST(NULL AS INT)               AS A_EDITOWNER,           --Изменил
    10                              AS A_STATUS,              --Статус
    CAST(NULL AS DATE)              AS A_PAY_DATE,            --Дата оплаты
    @yearReport_INT                 AS A_YEAR,                --Год
    @monthReport_INT                AS A_MONTH,               --Месяц
    0                               AS A_FULL_COST_MONTH,   --Полная стоимость оказанных услуг, руб.
    0                               AS A_SUM_PAY,              --Оплачено гражданином, руб.
    0                               AS A_PART_PAY,            --Остаток к оплате, руб.
    0                               AS A_SUMM_EXT_SERV,       --Сумма по дополнительным услугам, руб.
    0                               AS A_SUM_NORM_EX,         --Сумма превышения норматива, руб.
    CAST(NULL AS FLOAT)             AS A_SUMM_DOG,            --Размер частичной оплаты по договору, руб.
    @identifiedSocServ              AS ESRN_SOC_SERV,         --Назначение на социальное обслуживание
    @condition                      AS A_COND_SOC_SERV,       --Условие оказания социальных услуг
    CAST(NULL AS VARCHAR)           AS A_COMMENT,              --Комментарий
    0                               AS A_SUM_PERIOD_BUDGET, --Всего к оплате из средств бюджета
    SUM(t.TOTAL_COUNT_NORMAL)       AS A_FULL_COUNT,          --Количество оказанных услуг
    SUM(t.TOTAL_COUNT_NORMAL)       AS A_SOC_COUNT,           --Количество оказанных соц. услуг
    0                               AS A_SOC_COST_MONTH,    --Полная стоимость оказанных соц. услуг, руб.
    CAST(NULL AS INT)               AS A_DOP_COUNT,           --Количество оказанных доп. услуг
    CAST(NULL AS FLOAT)             AS A_DOP_COST_MONTH,    --Полная стоимость оказанных доп. услуг, руб.
    CAST(NULL AS INT)               AS A_OTHER_COUNT,         --Количество оказанных иных услуг
    CAST(NULL AS FLOAT)             AS A_OTHER_COST_MONTH   --Полная стоимость оказанных иных услуг, руб.
  FROM (
    SELECT
      socServAGR.A_ID                 AS SOC_SERV_AGR_OUID,
      SUM(costSocServMonth.A_NORMSOCSERV) AS TOTAL_COUNT_NORMAL
    FROM ESRN_SOC_SERV socServ --Назначение социального обслуживания.
    ----Агрегация по социальной услуге.
      INNER JOIN WM_SOC_SERV_AGR socServAGR
        ON socServAGR.ESRN_SOC_SERV = socServ.OUID
          AND socServAGR.A_STATUS = 10 --Статус агрегации в БД "Действует".
    ----Суммы по услуге за календарный месяц.
      INNER JOIN WM_COST_SOC_SERV_MONTH costSocServMonth
        ON costSocServMonth.A_AGR_SOC_SERV = socServAGR.A_ID
          AND costSocServMonth.A_STATUS = 10
          AND costSocServMonth.A_YEAR = @yearReport_INT
          AND costSocServMonth.A_MONTH = @monthReport_INT
    WHERE socServ.A_STATUS = 10 --Статус назначения в БД "Действует".
      AND socServ.OUID = @identifiedSocServ --Требуемое назначение.
    GROUP BY socServAGR.A_ID
  ) t
END
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Этап 8: Завершение
--////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
--Вывод результата.
SELECT @thereIsError AS thereIsError, @message AS message
--Удаление временных таблиц.
IF OBJECT_ID('tempdb..#FOUND_PEOPLE') IS NOT NULL BEGIN DROP TABLE #FOUND_PEOPLE END --Найденные люди по
входным данным.
IF OBJECT_ID('tempdb..#FOUND_SOC_SERV') IS NOT NULL BEGIN DROP TABLE #FOUND_SOC_SERV END --Найденные
социальные обслуживания по входным данным.
IF OBJECT_ID('tempdb..#DATA_FOR_INSERV') IS NOT NULL BEGIN DROP TABLE #DATA_FOR_INSERV END --Данные для
вставки.
```

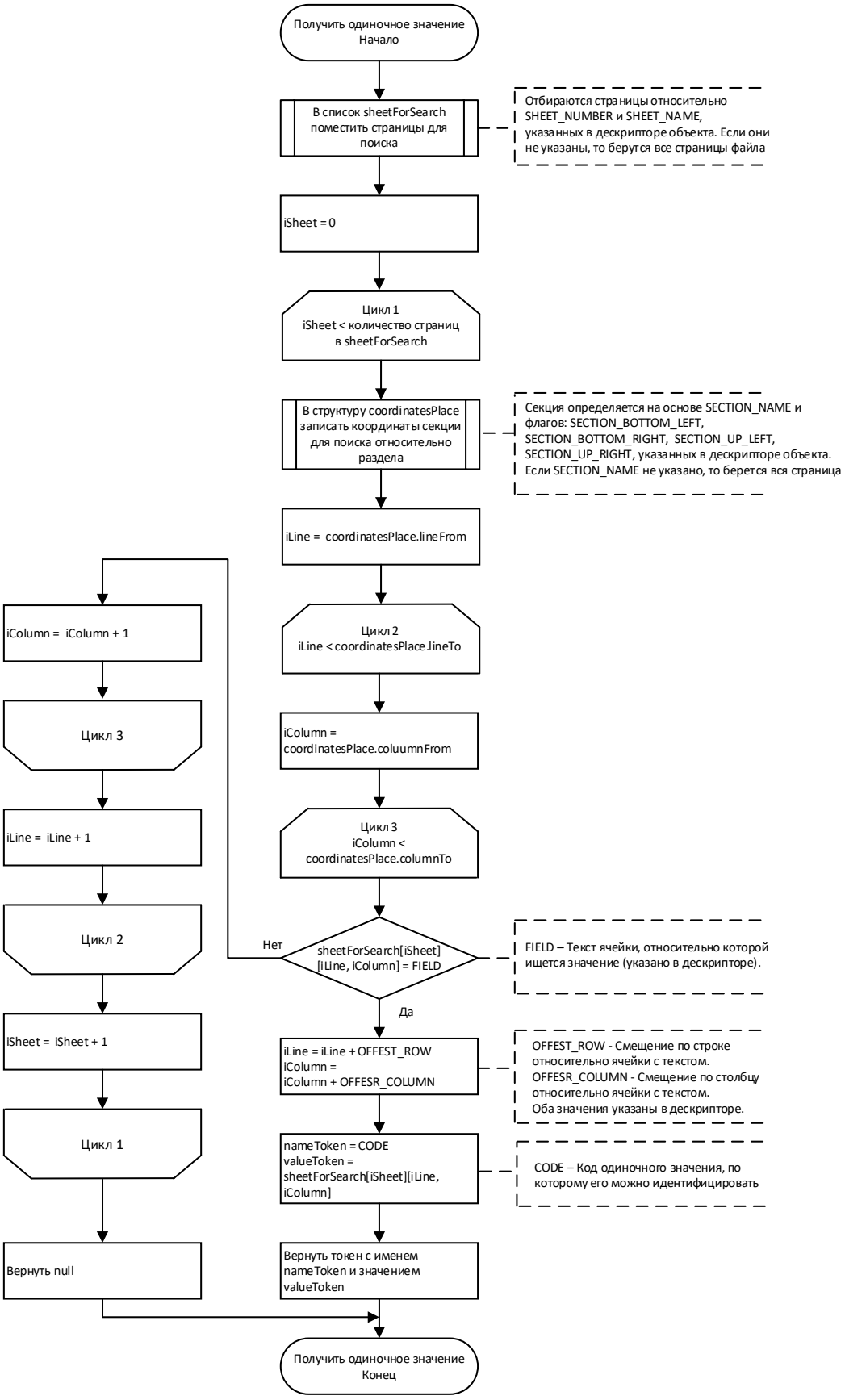
ПРИЛОЖЕНИЕ Г

Диаграмма классов



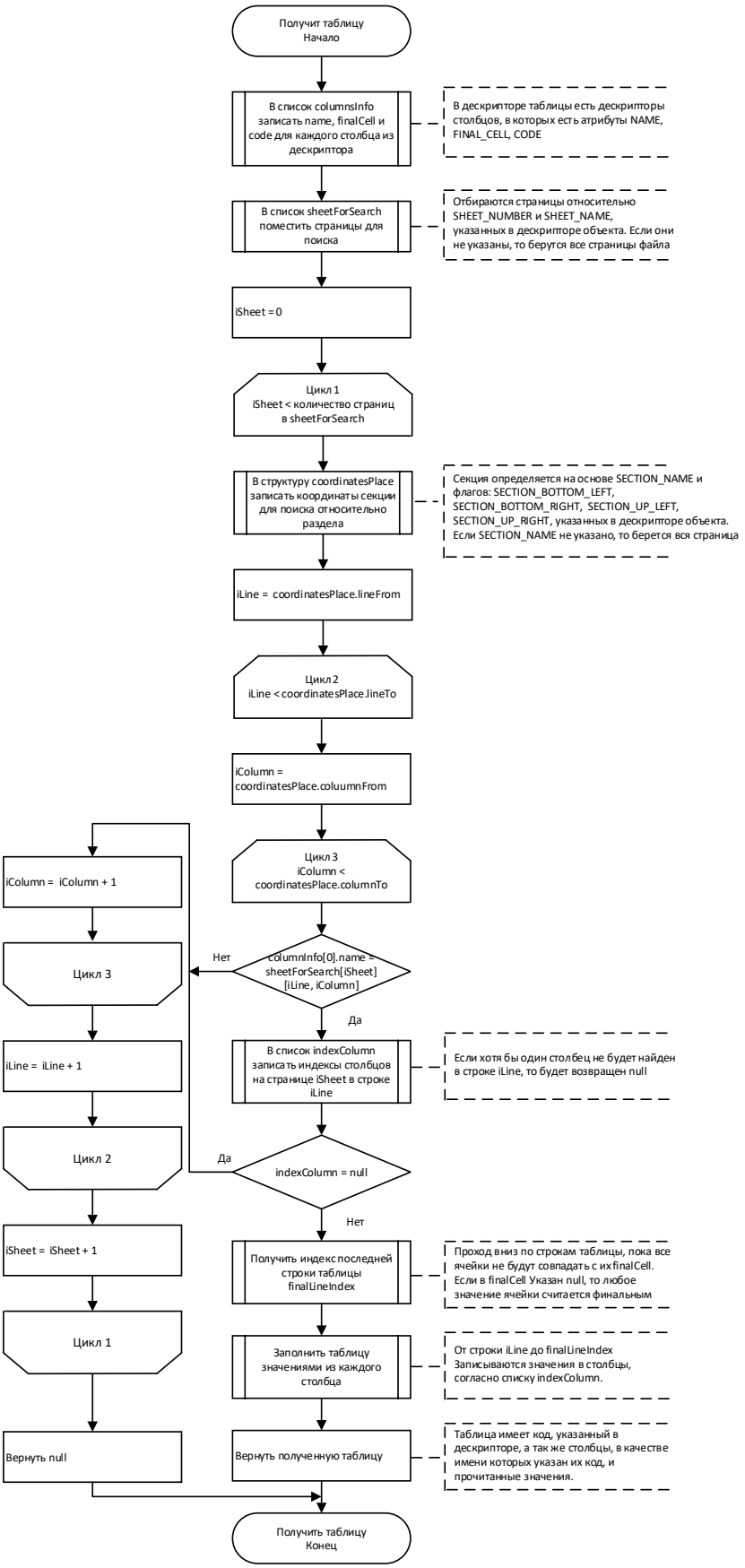
ПРИЛОЖЕНИЕ Д

Алгоритм извлечения одиночного значения из файла



ПРИЛОЖЕНИЕ Е

Алгоритм извлечения таблицы из файла



ПРИЛОЖЕНИЕ Ж

Пользовательский интерфейс