МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки / специальности)

Мобильные и сетевые технологии
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка модуля генерации отчетов XLSX/DOCX/HTML на основе метаданных системы для фармацевтической компании «Озон»

Обучающийся А.А. Арутюнян
(Инициалы Фамилия) (личная подпись)

Руководитель М.А. Тренина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант И.Ю. Усатова
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

Аннотация

Выпускная квалификационная работа посвящена проектированию и разработке модуля генерации отчётов на основе метаданных системы для фармацевтической компании «Озон».

Ключевые слова: генерация, база данных, SQL, C#, тестирование.

Объектом исследования в написании ВКР послужила фармацевтическая компания «Озон».

Предметом исследования бакалаврской работы стал разработанный для фармацевтической компании «Озон» модуль генерации отчётов.

В состав работы входят разделы: введение, 3 раздела, 3 вывода по написанным разделам, заключение.

Во введении раскрывается актуальность исследования и разработки модуля генерации отчётов. Описывается цель и задачи выпускной квалификационной работы. Определяются объект и предмет исследования.

В первом разделе исследована теоретическая часть, описана приведенная в пример предметная область образовательного портала, проведен сравнительный обзор существующих программных решений для генерации отчётов, сформулирована постановка задачи подсистемы генерации отчётов, приведено обоснование выбора языка программирования и СУБД.

Во втором разделе проведено проектирование алгоритма работы программы, разработана база данных образовательного портала, изучены общие сведения о работе оператора выборки SELECT, приведено описание работы конструкторов запросов и отчётов.

В третьем разделе было проведено тестирование разработанного модуля генерации отчётности. В ходе тестирования явные в работе ошибки устранены.

Заключение посвящено основным выводам по работе.

Объём выпускной квалификационной работы составляет 49 страниц, 24 рисунка и 18 таблиц. Задействовано 20 литературных источников.

Abstract

The title of the graduation work is « Development of a module for generating reports XLSX/DOCX/HTML based on system metadata for the pharmaceutical company «Ozon»».

The senior paper consists of an introduction, three parts, a conclusion, tables, list of references including foreign sources.

The key issue of the thesis is the development of the database and the module for generating XLSX/DOCX/HTML reports based on system metadata.

The aim of the work is to study the theoretical and practical aspects of development the report generation module based on system metadata.

The graduation work may be divided into several logically connected parts which are: description of the subject area of the educational portal given as an example, conducting a comparative review of existing software solutions for generating reports, designing the program operation algorithm, developing an educational portal database.

Finally, we present the work on the testing of the developed report generation module.

In conclusion we'd like to stress the main conclusions of the work, namely the relevance of the developed report generation module, which will improve the performance of the Ozon pharmaceutical company.

Содержание

Введение

В настоящее время задача создания отчетов играет важную роль в любой сфере деятельности человека. Сбор необходимых данных для составления отчёта может занять много времени. Модуль генерации отчётности путем автоматизации позволит в разы сократить затрачиваемое время на поиск информации, её систематизацию, структурирование, а также ускорит процесс формирования отчётных документов.

В связи с этим, возникает актуальность и необходимость разработки модуля генерации отчётов.

Объектом исследования выступает фармацевтическая компания «Озон»».

Предмет исследования – процесс проектирования модуля генерации отчётов.

Цель работы – изучить теоретические и практические аспекты проектирования модуля генерации XLSX/DOCX/HTML отчётов на основе метаданных системы для фармацевтической компании «Озон».

Для достижения поставленной цели необходимо решить ряд следующих задач:

1) произвести обзор и анализ специфики предметной области генерации отчётов;

3) произвести сравнительный обзор уже готовых программных решений в области генерации отчётов для выявления требований к модулю;

2) привести обоснование выбора средств разработки;

3) на основе проведенного анализа предметной области и обзора существующих программных решений сформулировать требования к системе и постановку задачи;

4) спроектировать и реализовать базу данных;

5) спроектировать и реализовать модуль генерации отчётов;

6) произвести тестирование модуля.

1. Теоретическая часть

1.1. Описание предметной области

За последние несколько лет на первый план выдвинулась новая информационная отрасль, которая связана с разработкой компьютерных технологий и технических средств для получения данных.

Производство информационных продуктов начало доминировать над производством материальных продуктов.

В результате этого возникли несоответствия между способностью человека к обработке информации и существующими наборами хранимой и передаваемой информации. Появилось большое количество избыточной информации, в которой порой сложно сориентироваться и выбрать нужную. Для решения таких задач используются автоматизированные системы, которые стали неотъемлемой частью практически всех компьютерных систем – от промышленных до отдельных компаний [8].

Данные объектов обрабатываются и формируются в различные типы отчетов, которые систематизируют полученные сведения за определенный промежуток времени и выводят их в удобном, графическом виде в форме графиков, диаграмм [8, 10].

1.2. Сравнительный анализ существующих программных решений для генерации отчётов

Среди разработчиков программных решений задача генерации отчетов является популярной. Для генерации отчетов существует множество систем, позволяющих выполнять их автоматизированное проектирование и формирование.

В данной работе будут рассмотрены некоторые из решений для генерации отчетов, а именно: Конструктор отчетов Битрикс 24, Fast Reports, Crystal Reports и Microsoft Access.

1.2.1. Конструктор отчетов Битрикс24

Конструктор отчетов Битрикс24 позволяет пользователь создавать различные типы отчетов на основе данных CRM за счет использования нескольких уровней группировки и расчета значений по формулам.

Возможности приложения включают в себя:

– Расчет полей по формуле. Можно задать любое количество формул, получить итоги с точки зрения группировки данных.

– Группировка по любым параметрам. Реализована поддержка до четырех уровней группировки. Легко создавать иерархические отчеты, позволяющие получать информацию из разных разделов.

– Мобильность. Просмотр отчетов с любого устройства.

– Универсальность. Периодические отчеты компаний, универсальные списочные отчеты [11].

Интерфейс приложения можно увидеть на рисунке 1.

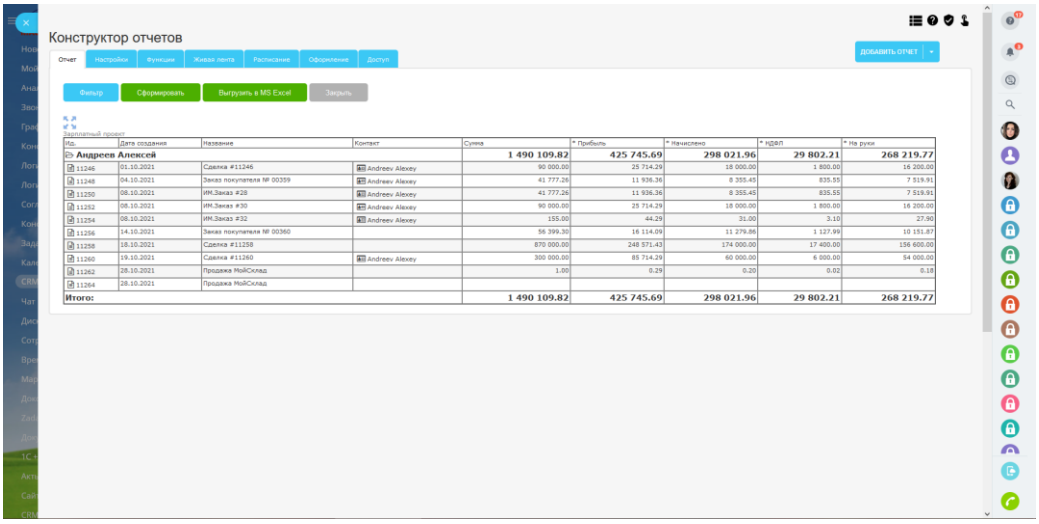


Рисунок 1 – Интерфейс решения «Конструктор отчетов Битрикс24»

1.2.2. Fast Reports

Fast Reports, Inc. – компания из России, разрабатывающая программные решения для генерации и формирования отчетности.

Fast Reports является одним из ведущих решений для генерации отчетов. Данная система имеет ряд преимуществ:

– мультиплатформенность;

– небольшой объем шаблона и высокая скорость формирования отчетов;

– имеет множество форматов для вывода.

Варианты решения включают в себя:

– генерация и формирования отчётов с любой сложностью;

– способность обслуживания сразу множества клиентов одновременно [2].

Внешний вид системы продемонстрирован на рисунке 2.

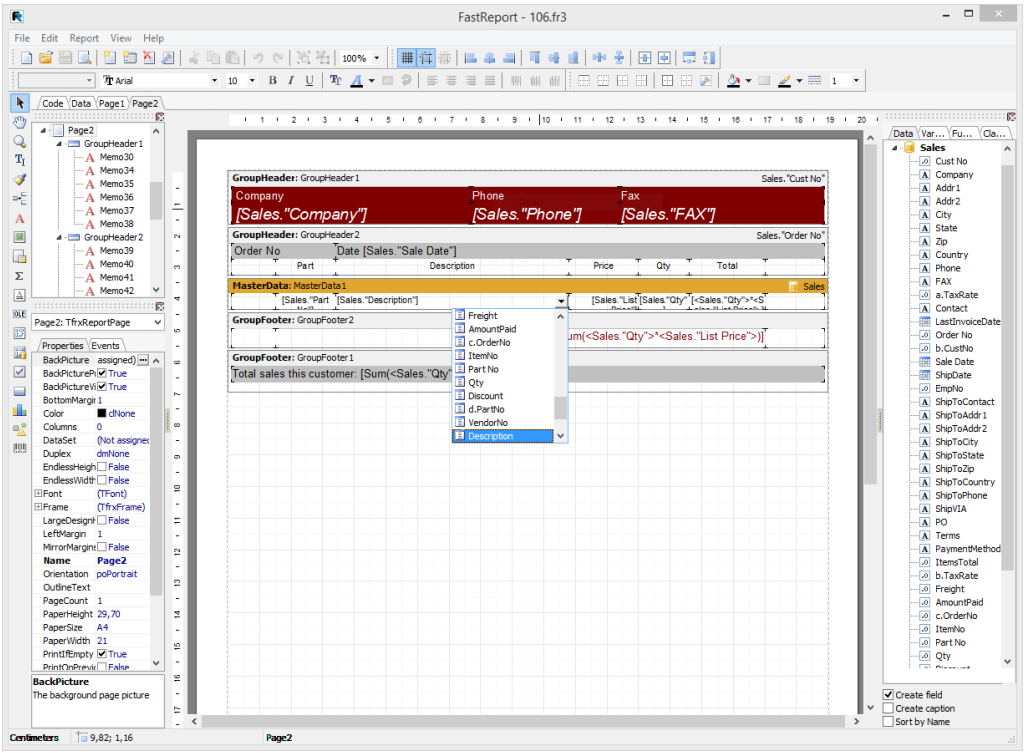


Рисунок 2 – Интерфейс решения «Fast Reports»

1.2.3. Crystal Reports

Crystal Reports – без преувеличения популярный и ведущий продукт в области формирования и генерации отчётов.

Данная система включает в себя следующие возможности:

– поддержка более чем 35 различных драйверов данных;

– гибкая настройках данных;

– возможна работа с документами XML;

– отчеты строятся при помощи визуального конструктора;

– возможность проектирования диаграмм и схем;

– возможность создания и настройки пользовательских отчетных шаблонов;

– экспорт в различные форматы [1].

Внешний вид системы продемонстрирован на рисунке 3.

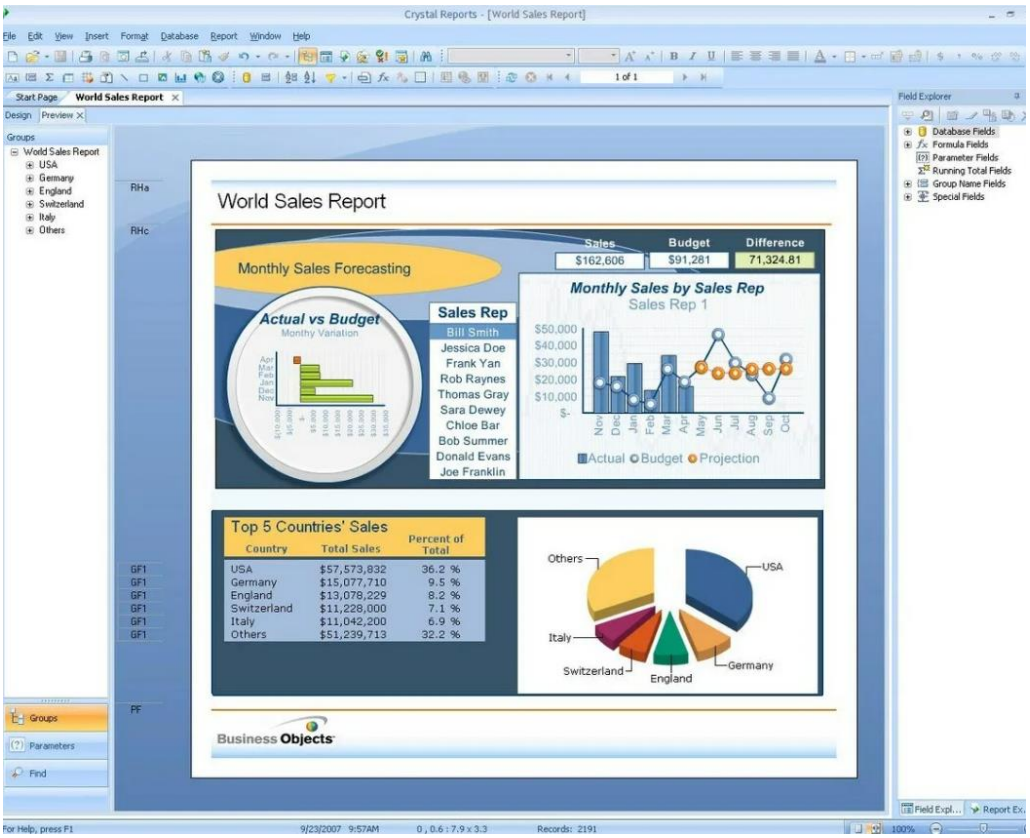


Рисунок 3 – Интерфейс решения «Crystal Reports»

1.2.4. MS Access

Microsoft Access – СУБД реляционного типа, разработанная корпорацией Microsoft.

На текущий момент MS Access активно обновляется и поставляется в составе пакета Office. Последняя версия СУБД была обновлена в 2021 году.

MS Access представляет из себя СУБД с графическим интерфейсом, который позволяет пользователям разрабатывать не только базы данных, но и клиентские приложения внутри одного файла.

Для разработки баз данных Access имеет следующие встроенные инструменты:

- создание таблицы в режиме таблицы;
- создание таблицы при помощи конструктора таблиц;
- импорт таблиц из различных источников Excel, Access, текстовых файлов и прочих.

Access также поддерживает функции по созданию пользовательских запросов на выборку, добавление, изменение и удаление данных.

Для разработки клиентских приложений Access имеет следующие встроенные инструменты:

- создание формы в режиме макета;
- создание формы при помощи конструктора форм;
- автоматическое создание формы при помощи генерации на основе данных таблицы или запроса.

Access поддерживает встроенный язык программирования Visual Basic для создания подпрограмм (макросов), позволяющий гибко настраивать и реализовывать функционал приложения [3].

Внешний вид системы продемонстрирован на рисунке 4.

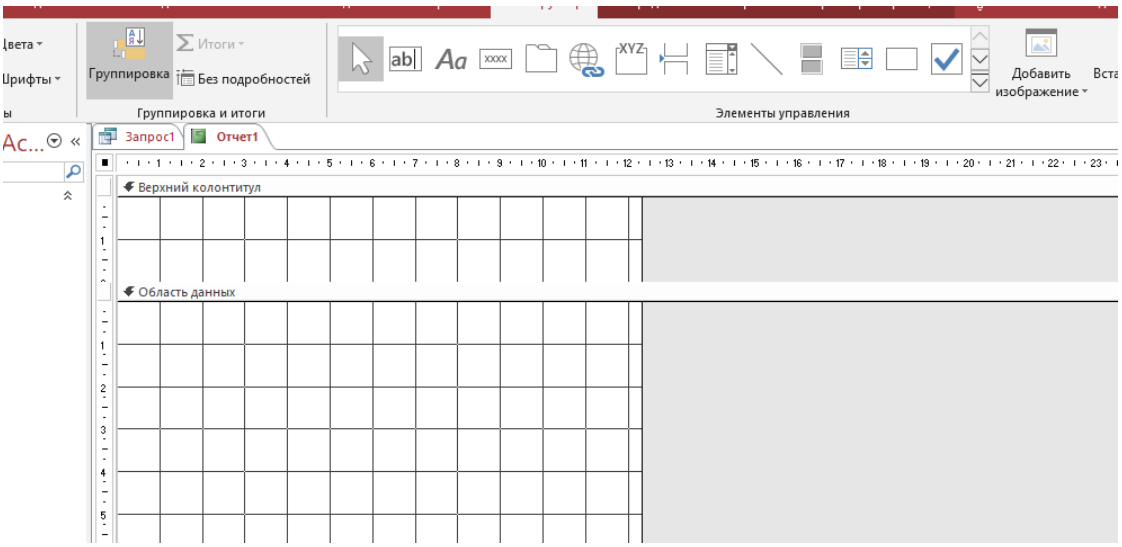


Рисунок 4 – Интерфейс решения «MS Access»

В ходе проведения анализ готовых программных решений в области генерации отчётов была составлен сравнительная таблица, представленная ниже в таблице 1.

Таблица 1 – Сравнительная характеристика программных решений

Программа	Конструктор запросов	Печать отчетов	Визуальный конструктор отчетов	Русский язык	Стоимость
Конструктор отчетов Битрикс24	–	+	–	+	Бесплатно
Fast Reports	+	+	+	+	2170 рублей
Crystal Reports	+	+	+	–	27250 рублей
MS Access	+	+	+	+	3300 рублей

В результате анализа рассмотренных систем генерации отчетов, можно сделать следующий вывод. Функционал систем, практически, одинаковый, но их стоимость резко отличается друг от друга в силу своей популярности и распространенности.

1.3. Постановка задачи подсистемы генерации отчётов

Цель данной работы – разработка модуля генерации отчетов на основе метаданных системы для фармацевтической компании «Озон».

Функционал системы будет тестироваться на примере работы образовательного портала. Поэтому, дальнейшее проектирование будет связано именно с данной предметной областью.

Возможности программы должны позволять:

– генерацию SQL-запросов посредством графического интерфейса пользователя;

– выбирать таблиц базы данных из предложенного списка;

– выбирать метаданные из таблиц;

– задавать условия для выборки;

– задавать параметры группировки данных;

– формировать отчет на основе построенного SQL-запроса;

– вывод отчета в следующих форматах: XLSX/DOCX/HTML.

Генератор отчетов позволит снизить временные затраты на формирование отчетности.

Программа должна иметь простой и понятные графический интерфейс, который не вызовет трудностей в его изучении и освоении.

1.4. Обоснование выбора языка программирования

На сегодняшний день существуют множество языков программирования. Рассмотрим некоторые из тех, которые поддерживаются средой Visual Studio.

C# является объектно-ориентированным языком программирования. Он ориентирован не на действия, а на объекты и данные. C# может применяться на любой из платформ, благодаря тому, что стал кроссплатформенным с приходом NET Core.

С помощью C# разработка стартует быстрее, что позволяет скорее получить нужное решение. На начальном этапе скорость разработки на языке C# значительно выше, чем на любом другом. C# изучается легче, синтаксис просто в понимании и на рынке существует уже более 23 лет.

Следующий, не менее популярный язык программирования, который будет рассмотрен – C++.

C++ является языком программирования общего назначения. В сравнении с C# обладает большей производительностью. Также, может быть использован на любой из платформ.

В простоте освоения C++ является противоположностью языку C#. Но разница в сложности синтаксиса сказывается на доходе разработчика.

Исходя из описания двух языков программирования C# и C++, можно сделать следующий вывод: C# проще изучается, порог вхождения ниже. C++ популярен, но сложнее в освоении. При разработке небольших проектов, особенно для новичков, предпочтительнее C#.

Таким образом, оценив сложность данного проекта, было принято решение выбрать в качестве инструмента программирования язык C#.

1.5. Обоснование выбора СУБД

Управление базами данных осуществляется при помощи систем управления базами данных.

Комплексом программных средств, предназначенных для создания, поддержки и обеспечения доступа множества пользователей к базе называется СУБД.

Реляционные СУБД на текущий момент являются наиболее предпочтительными, так как база данных в них представляется в виде набор таблиц, связанных между собой. Действия над данными таблиц производятся с помощью языка SQL. Для сравнения рассмотрим некоторых из известных СУБД.

MySQL – это постоянно обновляющийся бесплатный продукт компании Oracle. Для работы с MySQL существуют как платное, так и бесплатное издание. Бесплатное издание, в целом, обладает всеми необходимыми функциями для начинающего разработчика.

MySQL поддерживает набор пользовательских интерфейсов MySQL Workbench, позволяющий быстро и удобно вести разработку баз данных, практически, не прибегая к SQL [4].

Достоинства и недостатки СУБД MySQL приведены в таблице 2.

Таблица 2 – Достоинства и недостатки MySQL

Достоинства	Недостатки
Бесплатное распространение	Требовательна к компьютеру
Множество функций (создание триггеров, функций, курсоров, процедур и так далее)	Есть проблемы с интеграцией файлов
Наличие MySQL Workbench	–
Поддержка множества различных типов данных	–

Следующая СУБД – Microsoft SQL Server, разработанная корпорацией Microsoft.

Microsoft SQL Server представляет из себя СУБД, позволяющую работать как с облачными технологиями, так и на локальных устройствах. Обеспечивает возможность работы под операционными системами Windows, Linux, MacOS.

MS SQL Server распространяется как в платном издании, так и бесплатном Express Edition. Express версия СУБД поддерживает весь спектр необходимых функций для начинающего разработчика (создание таблиц, триггеров, хранимых процедур, пользовательских функций, индексов, проверочных выражений и другие) [7].

Достоинства и недостатки СУБД MS SQL приведены в таблице 3.

Таблица 3 – Достоинства и недостатки MS SQL Server

Достоинства	Недостатки
Полная и подробная документация с примерами работы	Высокие требования к системе
Бесплатное распространение	–
Поддержка пользовательского интерфейса на русском языке MS SQL Management Studio	–
Наличие большого числа инструментов для защиты	–
Множество функций (создание триггеров, функций, курсоров, процедур и так далее)	–
Поддержка взаимодействия с другими продуктами Microsoft	–
Поддержка множества различных типов данных	–

Последней рассмотренной СУБД в этом разделе будет Microsoft Access, разработанная той же корпорацией Microsoft. Данная СУБД объединяет ядро базы данных с пользовательским интерфейсом. То есть, в одном файле MS Access могут храниться как таблицы БД, так и формы, и отчеты клиентского приложения. Интерфейса системы поддерживает русский язык.

MS Access предлагает разработку приложений с помощью автоматизированных инструментов для создания форм, отчетов, запросов, таблиц, макросов. Макрос – своего рода участок программы, отвечающий за определенную функцию. Макросы пишутся на встроенном в MS Access языке Visual Basic [3].

К недостаткам Access можно отнести отсутствие возможности создания триггеров на языке SQL.

Достоинства и недостатки СУБД MS Access приведены в таблице 4.

Таблица 4 – Достоинства и недостатки MS Access

Достоинства	Недостатки
Поддержка пользовательского интерфейса на русском языке	Бесплатно не распространяется
Поддержка загрузки и выгрузки данных во внешние источники данных (Excel, Word)	Средства защиты почти отсутствуют
–	Нехватка документации
–	Отсутствуют некоторые функции, свойственные для СУБД (например, триггеры, курсоры)

Изучив возможности, достоинства и недостатки приведенных выше реляционных СУБД, было принято решение выбрать в качестве инструмента разработки базы данных MS SQL, так как она распространяется бесплатно с Express версией, поддерживает набор пользовательских интерфейсов MS SQL Management Studio, включает множество функций (от создания таблиц до триггеров и транзакций), поддерживает огромное множество различных типов данных.

1.6. Выводы по разделу

В результате работы над первым разделом ВКР была исследована теоретическая часть, описана приведенная в пример предметная область образовательного портала, проведен сравнительный обзор существующих программных решений для генерации отчётов, сформулирована постановка задачи подсистемы генерации отчётов, приведено обоснование выбора языка программирования и СУБД.

2. Практическая часть

2.1. Описание алгоритма работы программы

Алгоритм работы модуля генерации отчета заключается в следующем:

– пользователь, запустив модуль, задаёт свойства подключения к серверу Microsoft SQL Server и базе данных. Подключается к базе данных;

– модуль получает перечень объектов базы данных (таблицы, атрибуты и связи между ними);

– открывается окно «Конструктора запросов»;

– пользователь системы создает новый запрос путем выборки нужных ему таблиц и атрибутов базы данных;

– пользователь может задать условия на выборку, порядок сортировки, правила вывода атрибутов на экран;

– на основе выбранных пользователем объектов, модуль генерирует SQL-запрос и в режиме реального времени выводит его текст на экран;

– пользователь может выполнить сформированный SQL-запрос и получить его результат в виде таблицы;

– полученную выборку пользователь системы может экспортировать в один из следующих форматов: «XLSX», «DOCX», «HTML».

2.2. Разработка базы данных

2.2.1. Проектирование концептуальной модели данных

Концептуальная модель данных – это модель, позволяющая описать смысл информации, которой обмениваются участники какой-либо системы, с ее помощью они могут интерпретировать смысл (семантику) данных. В концептуальной модели данных факты описываются с помощью бинарных отношений между элементами данных.

Концептуальная модель предполагает не только выделение основных сущностей с указанием первичных ключей, но и определение взаимосвязей между ними [9].

Для концептуальной модели данных предметной области «Образовательный портал» можно выделить несколько информационных объектов:

– преподаватель – содержит сведения о преподавателях курсов на портале (будет использоваться в качестве справочника);

– курс – содержит информацию о курсах, по которым проходят обучение студенты портала;

– раздел курса – включает сведения о разделах курса (какие темы освещаются в заданном курсе);

– категория – содержит сведения о категориях курсов (будет использоваться в качестве справочника-классификатора);

– запись на курс – включает сведения о регистрации и записи студентов на курс;

– студент – содержит информацию о студентах, обучающихся на портале (будет использоваться в качестве справочника);

– преподаватель курса – содержит сведения о том, какой преподаватель и на каким курсах осуществляет свою деятельность.

Для объекта «Преподаватель» можно выделить следующие характеристики: уникальный код преподавателя (является первичным ключом и однозначно идентифицирующим объект полем), ФИО.

Объект «Курс» может быть охарактеризован следующим списком атрибутов: уникальный код курса (является первичным ключом и однозначно идентифицирующим объект полем), наименование (наименование курса также может быть уникальным), код категории (ссылка на объект «Категория»), длительность (в месяцах), описание курса и его стоимость.

Объекту «Раздел курса» соответствуют следующие характеристики: уникальный код раздела (является первичным ключом и однозначно

идентифицирующим объект полем), наименование, код курса (ссылка на объект «Курс»), описание раздела.

Для объекта «Категория» можно выделить следующие характеристики: уникальный код категории (является первичным ключом и однозначно идентифицирующим объект полем), наименование (наименование категории также может быть уникальным).

Объект «Запись на курс» может быть охарактеризован следующим списком атрибутов: уникальный код записи (является первичным ключом и однозначно идентифицирующим объект полем), код студента (ссылка на объект «Студент»), код курса (ссылка на объект «Курс»), дата записи и отметка об оплате курса.

Объекту «Студент» соответствуют следующие характеристики: уникальный код студента (является первичным ключом и однозначно идентифицирующим объект полем), ФИО, номер телефона (должно быть уникальным, так как номер телефона не может повторяться), адрес электронной почты (является однозначно идентифицирующим объект полем, так как адрес электронной почты не может повторяться), пароль для доступа к системе.

Для объекта «Преподаватель курса» можно выделить следующие характеристики: код курса (ссылка на объект «Курс»), код преподавателя (ссылка на объект «Преподаватель»). Первичный ключ составной.

Для создания связей между таблицами используются первичные и внешний ключи, а также дополнительные параметры обеспечения целостности.

В результате связывания объектов были получены следующие типы связей:

– одному преподавателю может быть назначено множество курсов. Следовательно, между объектами «Преподаватель» и «Преподаватель курса» необходимо установить связь типа «один-ко-многим»;

– один курс могут вести несколько преподавателей. Следовательно, между объектами «Курс» и «Преподаватель курса» необходимо установить связь типа «один-ко-многим»;

– один курс может иметь множество разделов. Следовательно, между объектами «Курс» и «Раздел курса» необходимо установить связь типа «один-ко-многим»;

– к одной категории может относиться множество курсов. Следовательно, между объектами «Категория» и «Курс» необходимо установить связь типа «один-ко-многим»;

– на один курс может записаться множество студентов. Следовательно, между объектами «Курс» и «Запись на курс» необходимо установить связь типа «один-ко-многим»;

– один студент может быть записан на прохождение множества курсов. Следовательно, между объектами «Студент» и «Запись на курс» необходимо установить связь типа «один-ко-многим».

С помощью CASE-средства Microsoft Visio построена схема концептуальной модели данных в нотации Питера Чена. Полученная диаграмма представлена на рисунке 5.

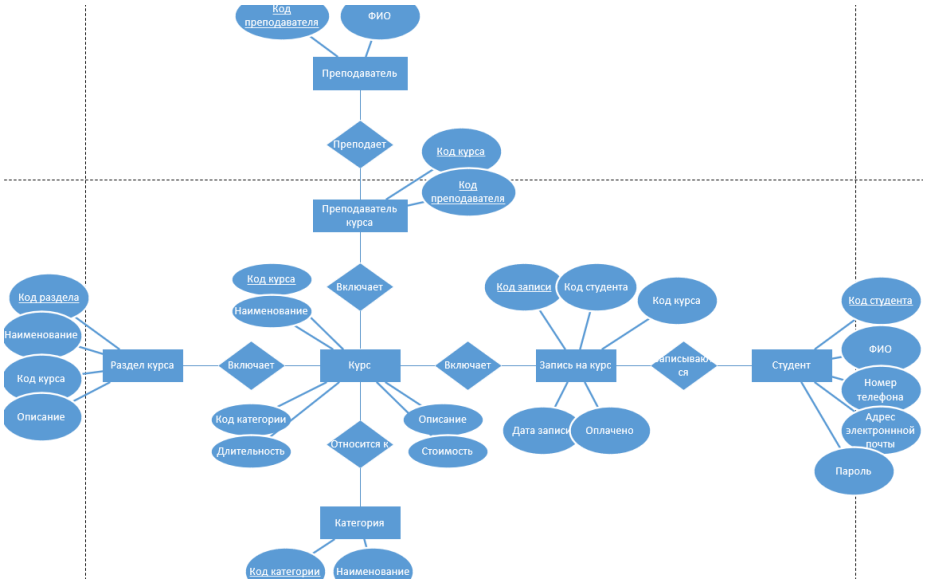


Рисунок 5 – Схема концептуальной модели данных

Таким образом, была спроектирована и получена концептуальная модель данных для генерации отчетов на основе метаданных образовательного портала.

2.2.2. Проектирование логической модели данных

Логическая модель данных – это представление структуры БД, независимое от специфики конкретной СУБД или платформы [13]. Для разработки логических моделей данных существует множество нотаций. В данной работе была использована нотаций IDEF1X.

Структура сущностей в нотации IDEF1X приведена в таблицах 5 – 11.

Таблица 5 – Атрибуты сущности «Преподаватель»

Имя атрибута	Тип данных	Ключ	Обязательность	Уникальность
Код преподавателя	Целое число	Первичный	Да	Да
ФИО	Текст	–	Да	Нет

Таблица 6 – Атрибуты сущности «Курс»

Имя атрибута	Тип данных	Ключ	Обязательность	Уникальность
Код курса	Целое число	Первичный	Да	Да
Наименование	Текст	–	Да	Нет
Код категории	Целое число	Внешний	Да	Нет
Длительность	Целое число	–	Да	Нет
Описание	Текст	–	Нет	Нет
Стоимость	Денежный	–	Да	Нет

Таблица 7 – Атрибуты сущности «Преподаватель курса»

Имя атрибута	Тип данных	Ключ	Обязательность	Уникальность
Код курса	Целое число	Первичный Внешний	Да	Да
Код преподавателя	Целое число	Первичный Внешний	Да	Да

Таблица 8 – Атрибуты сущности «Раздел курса»

Имя атрибута	Тип данных	Ключ	Обязательность	Уникальность
Код раздела	Целое число	Первичный	Да	Нет
Наименование	Текст	–	Да	Да
Код курса	Целое число	Внешний	Да	Нет
Описание	Текст	–	Нет	Нет

Таблица 9 – Атрибуты сущности «Запись на курс»

Имя атрибута	Тип данных	Ключ	Обязательность	Уникальность
Код записи	Целое число	Первичный	Да	Да
Код студента	Целое число	Внешний	Да	Нет
Код курса	Целое число	Внешний	Да	Нет
Дата записи	Дата	–	Да	Нет
Оплачено	Логический	–	Да	Нет

Таблица 10 – Атрибуты сущности «Категория»

Имя атрибута	Тип данных	Ключ	Обязательность	Уникальность
Код категории	Целое число	Первичный	Да	Да
Наименование	Текст	–	Да	Да

Таблица 11 – Атрибуты сущности «Студент»

Имя атрибута	Тип данных	Ключ	Обязательность	Уникальность
Код студента	Целое число	Первичный	Да	Да
ФИО	Текст	–	Да	Нет
Номер телефона	Текст	–	Да	Да
Адрес электронной почты	Текст	–	Да	Да
Пароль	Текст	–	Да	Нет

На рисунке 6 представлена схема логической модели базы данных «Образовательный портал», построенная в среде Erwin Data Modeler в нотации IDEF1X.

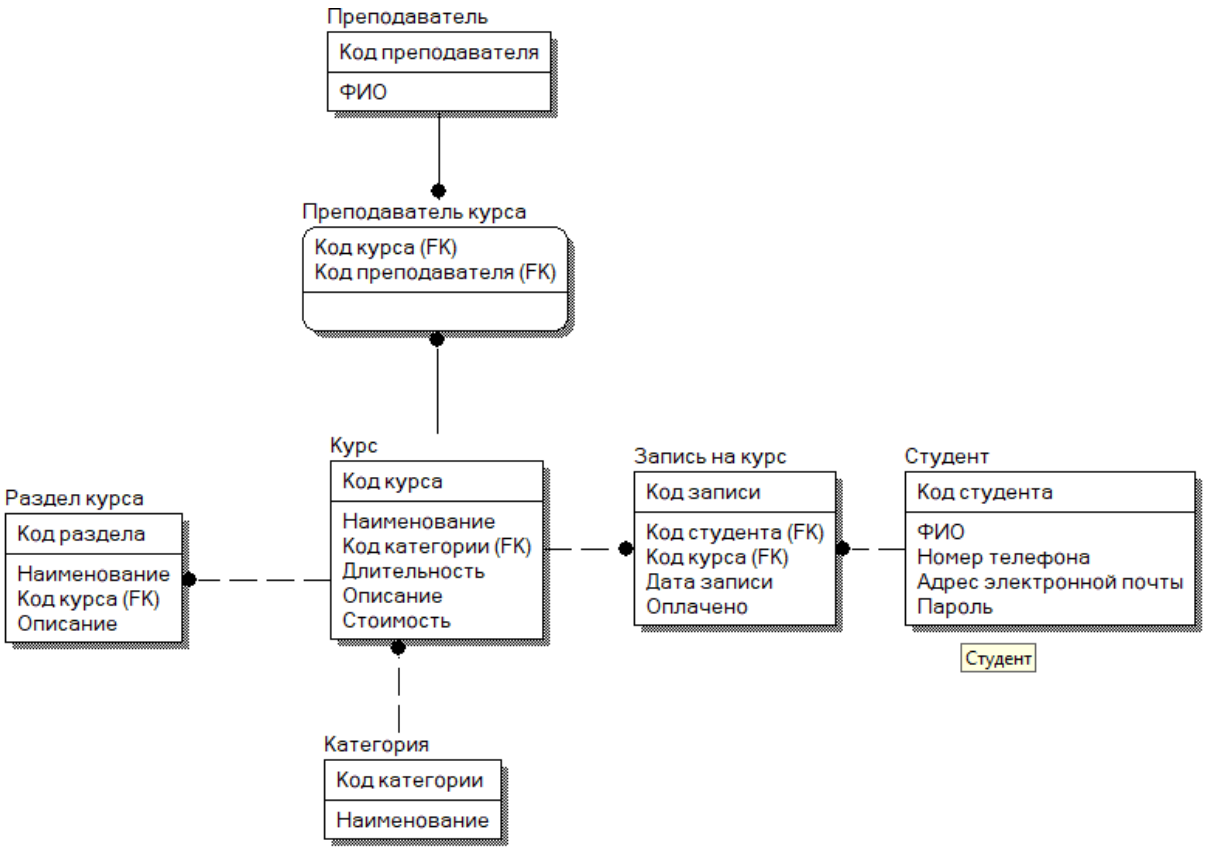


Рисунок 6 – Схема логической модели базы данных

2.2.3. Проектирование физической модели данных

Физическая модель базы данных строится с учетом специфики конкретной СУБД. В данной работе в качестве СУБД используется Microsoft SQL Server.

Таким образом, на основе полученной логической модели данных была спроектирована структура физической модели данных, приведенная в таблицах 12 – 18.

Таблица 12 – Атрибуты сущности «Преподаватель»

Имя атрибута	Описание	Тип данных	Размер	Ключ	Обязательность	Уникальность
id	Код преподавателя	INTEGER	11	Первичный	Да	Да
fullname	ФИО	VARCHAR	100		Да	Нет

Таблица 13 – Атрибуты сущности «Курс»

Имя атрибута	Описание	Тип данных	Размер	Ключ	Обязательность	Уникальность
id	Код курса	INTEGER	11	Первичный	Да	Да
name	Наименование	VARCHAR	100	–	Да	Нет
category_id	Код категории	INTEGER	11	Внешний	Да	Нет
duration	Длительность	INTEGER	11	–	Да	Нет
description	Описание	VARCHAR	MAX	–	Нет	Нет
cost	Стоимость	MONEY	19,2	–	Да	Нет

Таблица 14 – Атрибуты сущности «Преподаватель курса»

Имя атрибута	Описание	Тип данных	Размер	Ключ	Обязательность	Уникальность
course_id	Код курса	INTEGER	11	Первичный Внешний	Да	Да
teacher_id	Код преподавателя	INTEGER	11	Первичный Внешний	Да	Да

Таблица 15 – Атрибуты сущности «Раздел курса»

Имя атрибута	Описание	Тип данных	Размер	Ключ	Обязательность	Уникальность
id	Код раздела	INTEGER	11	Первичный	Да	Нет
name	Наименование	VARCHAR	100	–	Да	Да
course_id	Код курса	INTEGER	11	Внешний	Да	Нет
description	Описание	VARCHAR	MAX	–	Нет	Нет

Таблица 16 – Атрибуты сущности «Категория»

Имя атрибута	Описание	Тип данных	Размер	Ключ	Обязательность	Уникальность
id	Код категории	INTEGER	11	Первичный	Да	Да
name	Наименование	VARCHAR	100	–	Да	Да

Таблица 17– Атрибуты сущности «Запись на курс»

Имя атрибута	Описание	Тип данных	Размер	Ключ	Обязательность	Уникальность
id	Код записи	INTEGER	11	Первичный	Да	Да
student_id	Код студента	INTEGER	11	Внешний	Да	Нет
course_id	Код курса	INTEGER	11	Внешний	Да	Нет
enrollment_date	Дата записи	DATE	8	–	Да	Нет
paid	Оплачено	BIT	1	–	Да	Нет

Таблица 18 – Атрибуты сущности «Студент»

Имя атрибута	Описание	Тип данных	Размер	Ключ	Обязательность	Уникальность
id	Код студента	INTEGER	11	Первичный	Да	Да
fullname	ФИО	VARCHAR	100	–	Да	Нет
phone_number	Номер телефона	VARCHAR	16	–	Да	Да
email	Адрес электронной почты	VARCHAR	100	–	Да	Да
password	Пароль	VARCHAR	100	–	Да	Нет

На рисунке 7 представлена схема физической модели базы данных «Образовательный портал», построенная в среде Erwin Data Modeler.

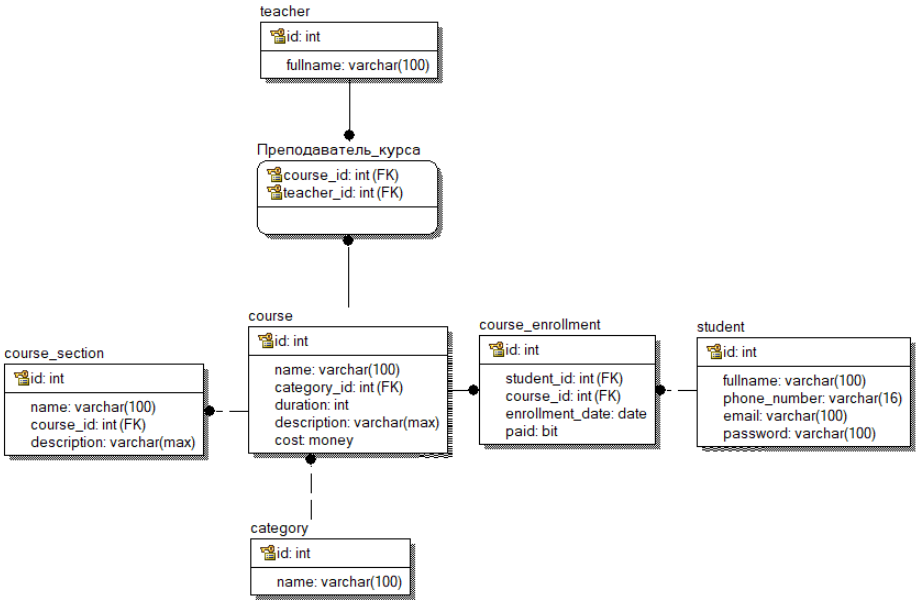


Рисунок 7 – Схема физической модели базы данных

На основе спроектированной физической модели данных средствами Microsoft SQL Server была реализована база данных [5] (рисунок 8)

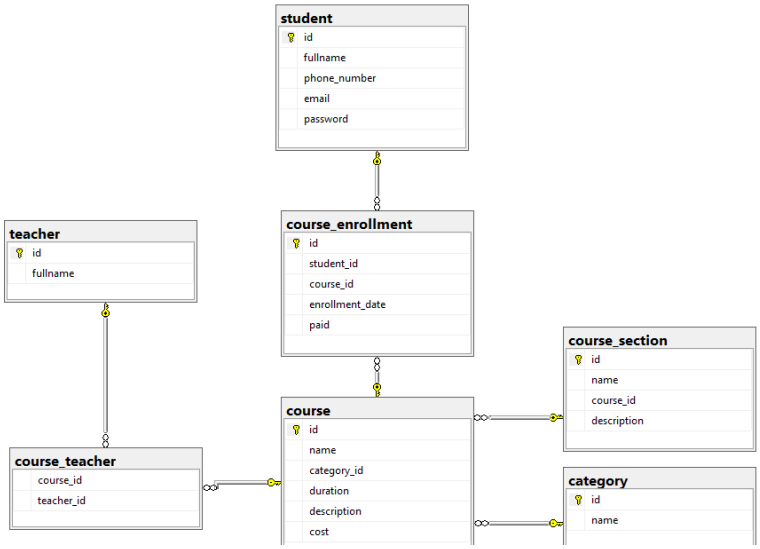


Рисунок 8 – Итоговая диаграмма базы данных «Образовательный портал»

Таким образом, была спроектирована и создана БД образовательного портала для генерации отчетов на основе метаданных.

2.3. Разработка программного модуля генерации отчётов

2.3.1. Описание работы оператора выборки SELECT

Таблица или запрос, построенный на основе нескольких таблиц, являются источниками данных. С помощью оператора SELECT производится запрос на выборку данных из источника данных.

Синтаксис оператора SELECT в общем случае имеет следующий синтаксис (листинг 2.1) [7]:

Листинг 2.1 – Синтаксис оператора SELECT в общем случае

```
SELECT <Атрибуты>
FROM <Таблицы>
[WHERE <Условие отбора>]
[GROUP BY <Атрибуты группировки>]
[HAVING <Фильтр после группировки>]
[ORDER BY <Атрибуты сортировки> [ASC | DESC] ].
```

Проектируемый модуль генерации отчетов будет использовать такие конструкции оператора SELECT, как: WHERE, FROM, AND, ORDER BY.

При помощи оператора WHERE модуль генерации отчетов получает и составляет связи между выбранными таблицами (вместо оператора JOIN).

Команда AND отвечает за соединение таблиц и реализацию условий отбора.

ORDER BY реализует порядок сортировки данных по выбранным пользователем полям.

2.3.2. Конструктор запросов

Для конструктора запросов свойственна задача создания SQL-запроса при помощи визуального интерфейса и графических элементов на форме.

Разработка и проектирование отчета реализуется на основании данных некоторого источника (таблица или запрос). Схема конструктора запросов модуля генерации отчетов представлена на рисунке 9.

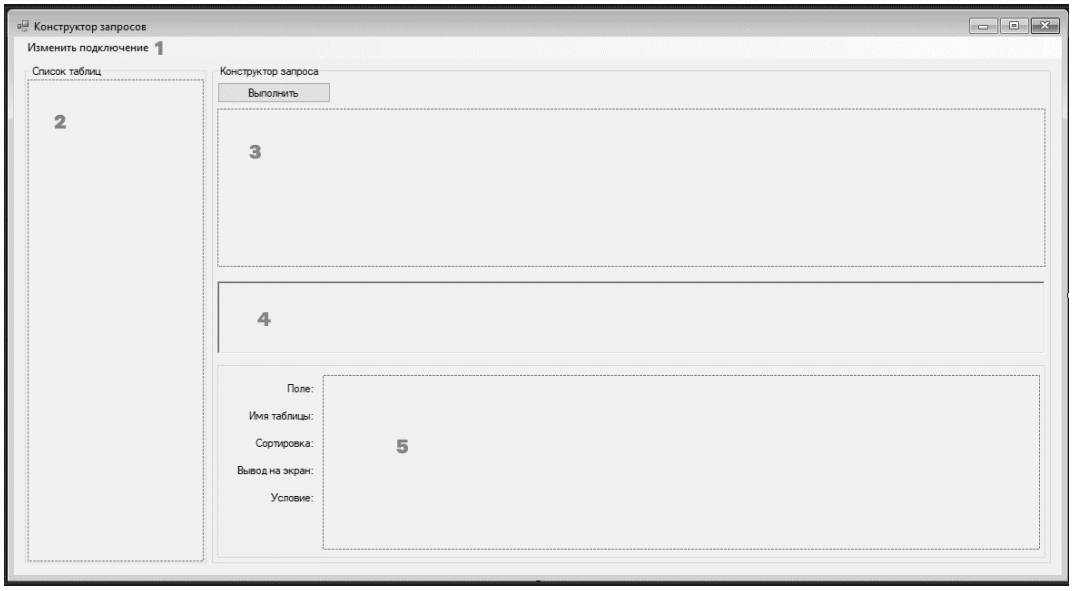


Рисунок 9 – Схема конструктора запросов

На рисунке 9 под цифрой 1 размещено меню программы, содержащее один пункт – изменить подключение. Нажав на данную кнопку, пользователь возвращается на форму, где может изменить свойства подключения и выбрать нужную ему базу данных для генерации отчетов.

По цифрой 2 в виде кнопок размещается список таблиц, полученных из базы данных.

Под цифрой 3 размещается визуальная схема базы данных, полученная из выбранных пользователем таблиц (сравнимо с тем, как это делает встроенный построитель диаграмм Microsoft SQL Server).

Под цифрой 4 расположено текстовое поле, в которое записывается сформированный SQL-запрос на текущий момент.

Под цифрой 5 размещаются выбранные пользователем атрибуты. Для атрибутов можно задать следующие параметры: сортировка (выпадающий список), вывод на экран (логическое поле), условие (текстовое поле с условием

на выборку). Для каждого атрибута указываются также имя и таблица, которой он принадлежит.

2.3.2. Конструктор отчётов

На рисунке 10 приведен макет конструктора отчётов.

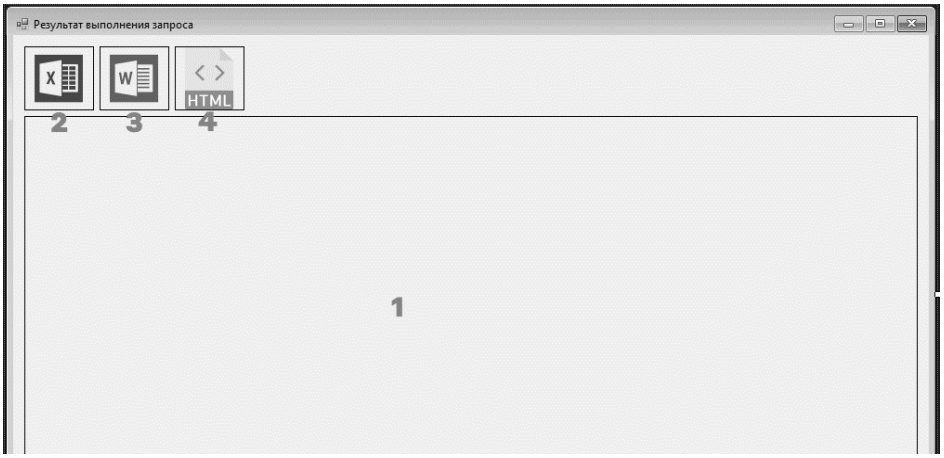


Рисунок 10 – Макет конструктора отчётов

На форме «Результат построения запроса» размещены следующие элементы:

– под цифрой 1 – визуальное представление выборки из базы данных;

– под цифрой 2 – кнопка для экспорта выборки в документ EXCEL в формате XLSX;

– под цифрой 3 – кнопка для экспорта выборки в документ WORD в формате DOCX;

– под цифрой 4 – кнопка для экспорта выборки в файл HMTL.

2.4. Описание работы программы

Пользователь, запуская модуль генерации отчетов, попадает на форму «Свойства подключения». Она позволяет определить, к какой серверу и какой

базе данных осуществляется подключение. Алгоритм работы подключения к базе данных следующий:

– ввод имени сервера и к шагу 2;

– выбор проверки подлинности. Если имя пользователя и пароль не требуются, то к шагу 4, иначе к шагу 3;

– ввод имени пользователя и пароля. К шагу 4;

– если введенные данные верны, то к шагу 5, иначе к шагу 1;

– получить список баз данных. Выбрать из списка. К шагу 6;

– создать подключение к базе данных. Открыть форму «Конструктор запросов» и к шагу 7;

– конец.

Внешний вид формы «Свойства подключения» можно увидеть на рисунке 11.

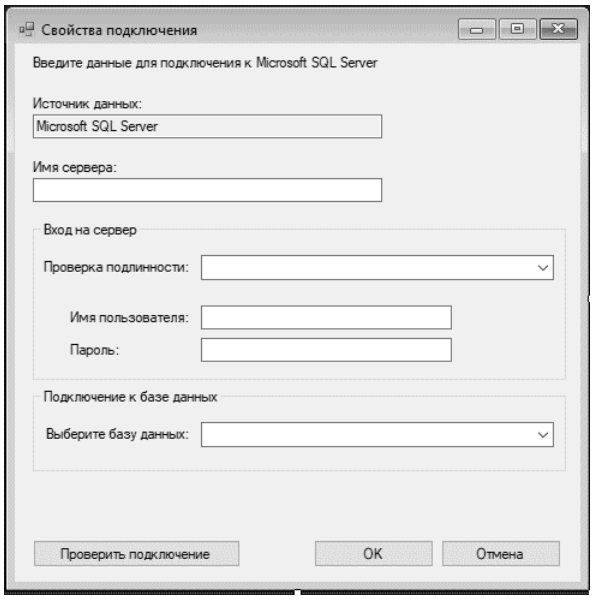


Рисунок 11 – Внешний вид формы «Свойства подключения»

После открытия формы «Конструктор запросов» происходит выборка таблицы и атрибутов из базы данных. Делается это следующим образом:

– выполнить запрос к базе данных, чтобы получить список таблиц;

– выполнить запрос к базе данных, чтобы получить атрибуты таблиц;

– выполнить запрос к базе данных, чтобы получить связи между таблицами.

Для таблиц и атрибутов базы данных созданы отдельные классы для сериализации. Код класса «Table» приведен ниже в листинге 2.2.

Листинг 2.2 – Код класса «Table»

```
public class Table
{
    public string Name { get; set; }
    public Field[] Fields { get; set; }
    public List<Table> Parents { get; set; }

    public Table()
    {
        Parents = new List<Table>();
    }
}
```

Код класса «Field» приведен ниже в листинге 2.3.

Листинг 2.3 – Код класса «Field»

```
public enum SortingOrder
    {
        Asc,
        Desc,
        None
    }

    public class Field
    {
        public string Name { get; set; }
        public string DataType { get; set; }
        public int Length { get; set; }
        public bool IsNullable { get; set; }
        public Table Parent { get; set; }
        /// <summary>
        /// Атрибут, на который ссылаются
        /// </summary>
        public bool IsReferenced { get; set; }
        /// <summary>
        /// Атрибут, который ссылается
        /// </summary>
```

```
        public bool IsReferencing { get; set; }
        public SortingOrder SortingOrder { get; set; }
        public bool IsOutputOnScreen { get; set; }

        public Field()
        {
            IsReferenced = false;
            IsReferencing = false;
            SortingOrder = SortingOrder.None;
            IsOutputOnScreen = true;
        }

        public override string ToString()
        {
            return $"{Name} : {DataType} ({Length}) {(IsNullable
? "NULL" : "NOT NULL")}";
        }
    }
```

Сериализация таблиц базы данных производится в соответствии с листингом 2.4.

Листинг 2.4 – Сериализация таблиц базы данных

```
var tFields = table as IDictionary<string, object>;
                string                  tableName                =
tFields["TABLE_NAME"].ToString();

                Table newTable = new Table() { Name = tableName };
```

Сериализация атрибутов таблиц производится в соответствии с листингом 2.5.

Листинг 2.5 – Сериализация атрибутов таблиц

```
var cFields = column as IDictionary<string, object>;
                    Field field = new Field()
                    {
                        Name = cFields["COLUMN_NAME"].ToString(),
                        DataType                                    =
cFields["DATA_TYPE"].ToString(),
                        Length                                      =
cFields["CHARACTER_MAXIMUM_LENGTH"]    ==    null    ?    0    :
Convert.ToInt32(cFields["CHARACTER_MAXIMUM_LENGTH"]),
                        IsNullable                                  =
cFields["IS_NULLABLE"].ToString() == "YES" ? true : false,
                        Parent = newTable };
```

После успешного получения объектов из базы данных, пользователь начинает работу с конструктором запросов. Выбрав таблицу из списка, она добавляет на форму в следующем виде (рисунок 12).

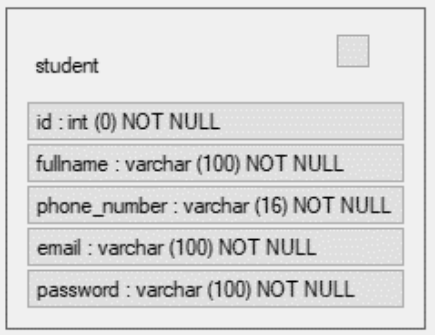


Рисунок 12 – Таблица в конструкторе запросов

Добавление таблицы в конструктор запросов производится в соответствии с листингом 2.6.

Листинг 2.6 – Код для добавления таблицы в конструктор запросов

```
private void BtnTable_Click(object sender, EventArgs e)
        {
            Table table = (sender as Button).Tag as Table;

            AddTable(table);
        }
```

Код метода «AddTable» приведен в приложении А.

После того, как таблицы были добавлены в конструктор, пользователь может выбрать нужные ему атрибуты в результативной выборке. Для добавления атрибутов на форму должно произойти событие клика по нему (листинг 2.7).

Листинг 2.7 – Код для добавления атрибута в конструктор запросов

```
private void BtnField_Click(object sender, EventArgs e)
        {
            Field field = (sender as Button).Tag as Field;
            AddField(field);
        }
```

Листинг метода «AddField» приведен в приложении Б.

Итоговым этапом работы конструктора запросов является формирование SQL-запроса. Для этого был создан отдельный метод «BuildSqlQuery», листинг которого представлен в приложении В.

Выполнив полученный SQL-запрос, пользователь может экспортировать полученные данные в один из форматов: XLSX, DOCX, HTML. Для каждого формата вывода создан отдельный метод в отдельном классе «Exporter». Его структура представлена на рисунке 13.

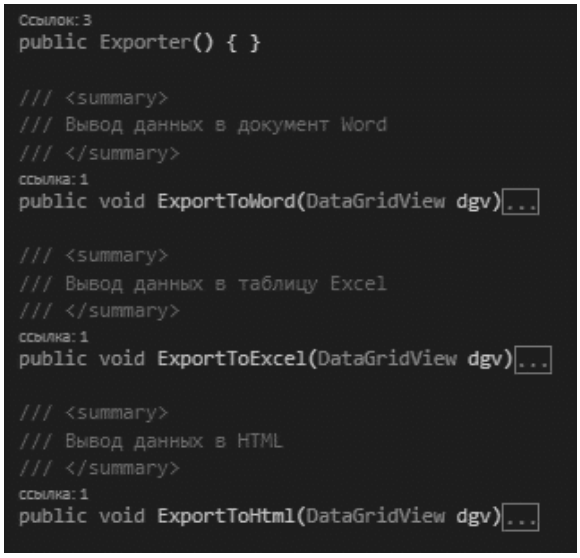


Рисунок 13 – Структура класса «Exporter»

2.5. Выводы по разделу

В результате работы над вторым разделом ВКР было проведено проектирование и разработка алгоритма работы программы, разработана база данных образовательного портала, изучены общие сведения о работе оператора выборки SELECT, приведено описание работы конструкторов запросов и отчётов.

3. Тестирование работы модуля генерации отчётности

В результате тестирования модуля генерации отчётов были произведены следующие тесты.

Тест 1. Подключение к базе данных образовательного портала и получение списка объектов. Результаты теста 1 изображены на рисунке 14, 15.

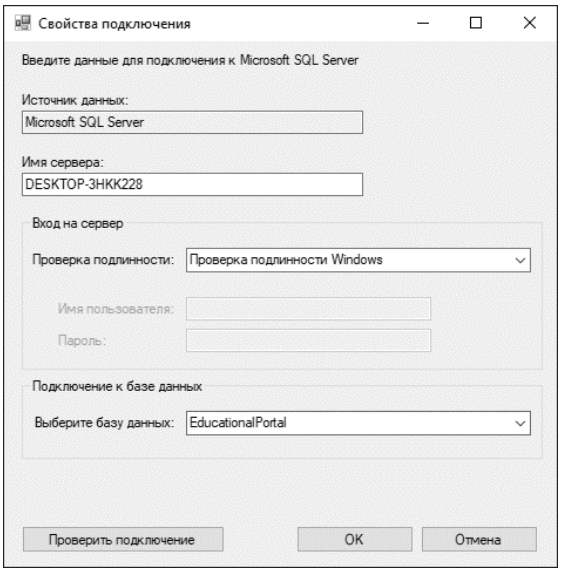


Рисунок 14 – Результат теста 1 (подключение к базе данных)

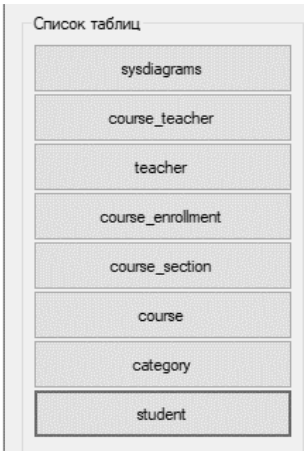


Рисунок 15 – Результат теста 1 (получение списка объектов)

Тест 2. Размещение выбранной таблицы в конструкторе запросов. Процесс размещения выбранной таблицы изображен на рисунке 16.

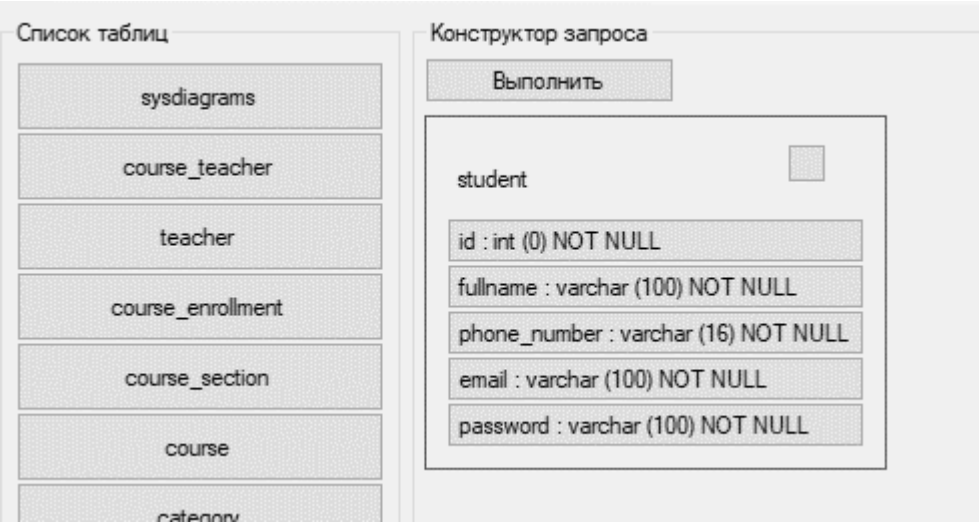


Рисунок 16 – Результат теста 2

Тест 3. Размещение нескольких таблиц в конструкторе запросов. Результаты 3 теста изображены на рисунке 17.

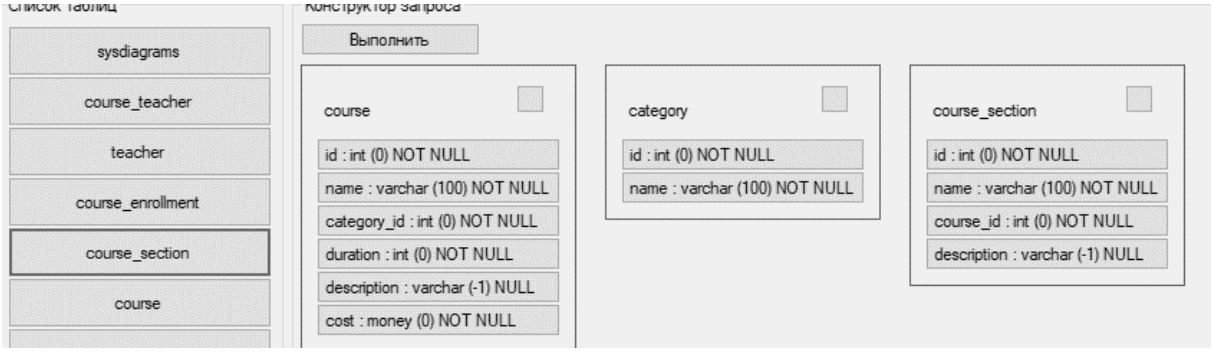


Рисунок 17 – Результат теста 3

Тест 4. Добавление атрибутов в конструктор запросов.

Полученный результат изображен на рисунке 18.

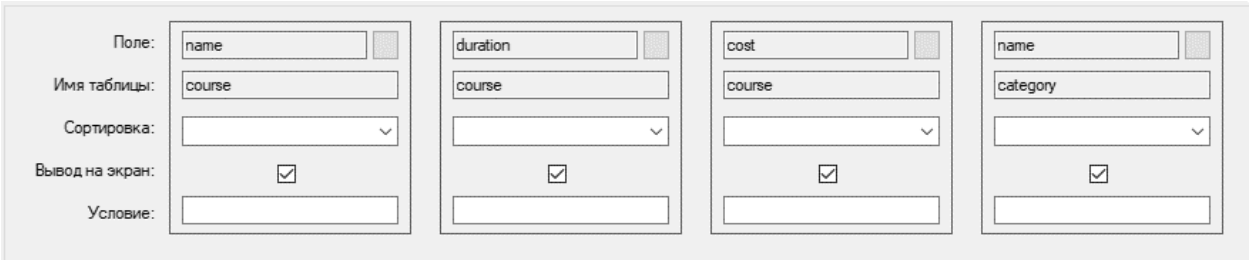


Рисунок 18 – Результат теста 4

Тест 5. Выбор порядка сортировки атрибутов, добавление условия на выборку. Результаты теста 5 изображены на рисунке 19.

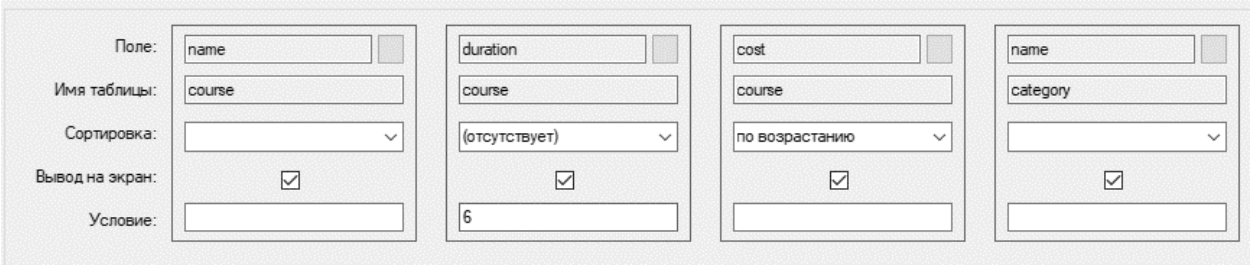


Рисунок 19 – Результат теста 5

Тест 6. Сформировать SQL-выражение на основе конструктора запроса, полученная в тесте 5. Сформированное SQL-выражение изображено на рисунке 20.

```
SELECT course.name as course_name,course.duration as course_duration,course.cost as
course_cost,category.name as category_name FROM course,category WHERE category.id = course.category_id
AND course.duration = '6' ORDER BY course.cost ASC
```

Рисунок 20 – Результат теста 6

Тест 7. Выполнение запроса на выборку, полученного в тесте 6. Результат теста 7 изображен на рисунке 21.

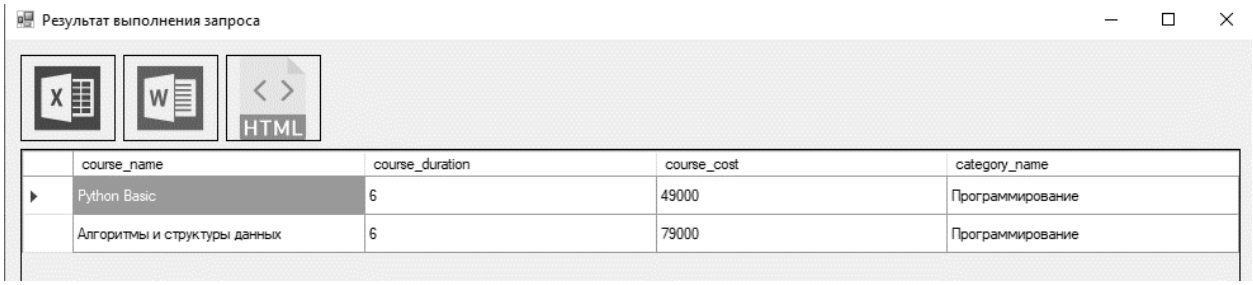
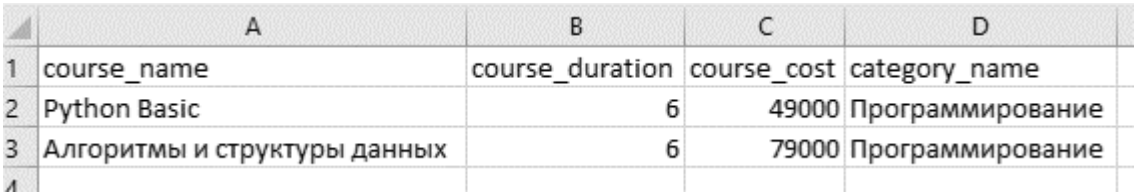


Рисунок 21 – Результат теста 7

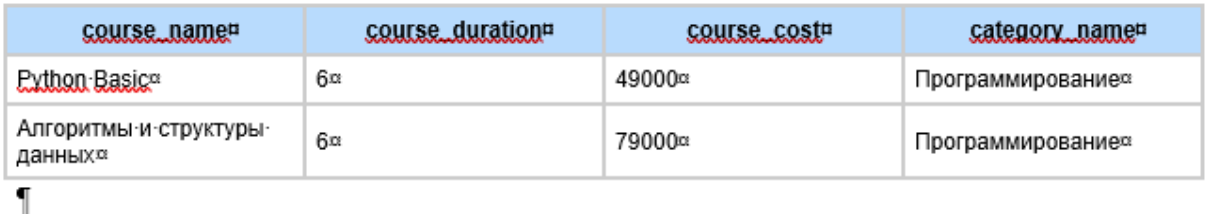
Тест 8. Сформировать отчет XLSX на основе выборки, полученной в тесте 7. Сформированный отчет XLSX изображен на рисунке 22.



	A	B	C	D
1	course_name	course_duration	course_cost	category_name
2	Python Basic	6	49000	Программирование
3	Алгоритмы и структуры данных	6	79000	Программирование

Рисунок 22 – Результат теста 8

Тест 9. Сформировать отчет DOCX на основе выборки, полученной в тесте 7. Сформированный отчет DOCX изображен на рисунке 23.



course_name¤	course_duration¤	course_cost¤	category_name¤
Python·Basic¤	6¤	49000¤	Программирование¤
Алгоритмы·и·структуры·данных¤	6¤	79000¤	Программирование¤

¶

Рисунок 23 – Результат теста 9

Тест 10. Сформировать отчет HTML на основе выборки, полученной в тесте 7. Сформированный отчет HTML изображен на рисунке 24.

course_name	course_duration	course_cost	category_name
Python Basic	6	49000	Программирование
Алгоритмы и структуры данных	6	79000	Программирование

Рисунок 24 – Результат теста 10

3.2. Выводы по разделу

В результате работы над третьим разделом ВКР было произведено тестирование разработанного модуля генерации отчётности. В ходе тестирования явные в работе ошибки устранены.

Заключение

В рамках выполнения данной выпускной квалификационной работы была спроектирована и реализована база данных и программный модуль, имеющий графический интерфейс пользователя, для генерации XLSX/DOCX/HTML отчетов на основе метаданных системы для фармацевтической компании «Озон».

В ходе работы над первой задачей был проведен анализ предметной области генерации отчётов, проведено сравнение готовых программных решений, выполняющих задачу генерации отчётов. В результате данного сравнения были выявлены необходимые требования к проектируемому программному модулю. Проведены анализ и сравнение средств реализации модуля, в результате которых были выбраны язык программирования и система для работы с базой данных.

Второй задачей проекта было проектирование и реализация базы данных, и модуля генерации XLSX/DOCX/HTML отчетов. В ходе исследования данных задач была построена концептуальная модель базы данных, выделены основные информационные объекты и атрибуты. На основании концептуальной модели построены логическая и физическая модели. Итогом работы над вторым разделом ВКР стал реализованный в среде Visual Studio интерфейс модуля. Также было составлено описание алгоритма работы модуля генерации отчетов.

В ходе работы над третьей задачей ВКР было проведено тщательное тестирование модуля генерации отчётов. В результате чего, были составлены, а затем пройдены 10 пользовательских тестов. Исходя из результатов тестирования, можно сделать вывод о том, что все явные ошибки в работе приложения устранены. Модуль генерации отчетов работает корректно и выполняет весь заявленный функционал.

Разработанный модуль генерации отчетов, при желании, может быть доработан и улучшен.

Список используемой литературы

1) Бедердинова О. И., Минеева Т. А., Водовозова Ю. А. Создание приложений баз данных в среде Visual Studio. Инфра-М., 2021. – 94 с. Электронный ресурс. Режим доступа: https://znanium.com/catalog/document?id=373660.

2) Бондаренко И. С. Базы данных: создание баз данных в среде SQL Server. ИД МИСиС., 2019. – 39 с. Электронный ресурс. Режим доступа: https://znanium.com/catalog/document?id=371206

3) Винкоп, Стефан Использование Microsoft SQL Server 7.0. Специальное издание / Стефан Винкоп. - М.: Вильямс, 2017. - 816 с.

4) Жилинский, А. Самоучитель Misrosoft SQL Server 2008 / А. Жилинский. - М.: БХВ-Петербург, 2018. - 240 с.

5) Заботина Н.Н. Проектирование информационных систем. Электронный ресурс. Учебное пособие /Н.Н. Заботина - М.: НИЦ ИНФРА-М, 2016. - 331 с. – ЭБС Znanium.com. – Режим доступа: http://znanium.com/catalog/product/542810

6) Инфологическая модель данных. StudFiles. Электронный ресурс. Режим доступа: https://studfile.net/preview/720297/

7) Коваленко В.В. Проектирование информационных систем Электронный ресурс: учебное пособие / В.В. Коваленко. — М.: ФОРУМ: ИНФРА-М, 2018. — 320 с. – ЭБС Znanium.com. – Режим доступа: http://znanium.com/catalog/product/980117.

8) Конструктор отчетов Битрикс24. Электронный ресурс. Режим доступа: https://www.bitrix24.ru/apps/?app=htmls.reports

9) Кригель, А. SQL. Библия пользователя / А. Кригель. - М.: Диалектика / Вильямс, 2019. - 318 с.

10) Кузнецов, С. Д. Основы баз данных / С.Д. Кузнецов. - М.: Бином. Лаборатория знаний, Интернет-университет информационных технологий, 2017. - 488 с.

11) Оппель, Эндрю Дж. SQL. Полное руководство / Оппель Эндрю Дж. - М.: Диалектика / Вильямс, 2017;

12) Проектирование баз данных. Википедия. Электронный ресурс. Режим доступа: https://ru.wikipedia.org/wiki/Проектирование_баз_данных;

13) Тестирование программного обеспечения. Википедия. Электронный ресурс. Режим доступа: https://ru.wikipedia.org/wiki/Тестирование_программного_обеспечения

14) Уолтерс, Роберт SQL Server 2008. Ускоренный курс для профессионалов / Роберт Уолтерс и др. - М.: Вильямс, 2017. - 768 с.

15) Хомоненко, А. Работа с базами данных / А. Хомоненко. - М.: Книга по Требованию, 2017. - 488 с.

16) Crystal Report. Официальный сайт разработчика. Электронный ресурс. Режим доступа: https://www.crystalreports.com/

17) Fast Reports. Официальный сайт разработчика. Электронный ресурс. Режим доступа: https://www.fast-report.com/en/

18) MS Access. Официальный сайт разработчика. Электронный ресурс. Режим доступа: https://www.microsoft.com/ru-ru/microsoft-365/access

19) MySQL. Электронный ресурс. Режим доступа: https://www.mysql.com/

20) SQL Server Соглашение по именованию и T-SQL стиль программирования. Электронный ресурс. Режим доступа: https://sqlserver-kit.org/ru/home/convention

Приложение А

Листинг метода AddTable

```
private void AddTable(Table table)
        {
            //проверяем, не добавлена ли уже таблица
            if              (panelQuery.Controls.Find(table.Name,
true).FirstOrDefault() != null)
                return;

            Panel tPanel = new Panel()
            {
                Padding = new Padding(10),
                AutoSize = true,
                BorderStyle = BorderStyle.FixedSingle,
                Tag = table,
                Name = table.Name
            };

            //шапка таблицы
            TableLayoutPanel tlpHeader = new TableLayoutPanel()
            {
                Height = 40,
                Dock = DockStyle.Top,
                RowCount = 1,
                ColumnCount = 2
            };

            tPanel.Controls.Add(tlpHeader);

            tlpHeader.ColumnStyles.Clear();
            //заговолок
            tlpHeader.ColumnStyles.Add(new
ColumnStyle(SizeType.Percent, 80));
            //кнопка Удалить
            tlpHeader.ColumnStyles.Add(new
ColumnStyle(SizeType.Percent, 20));

            tlpHeader.Controls.Add(new Label()
            {
                Text = table.Name,
                Dock = DockStyle.Fill,
                TextAlign = ContentAlignment.MiddleLeft
            });

            Button btnRemoveTable = new Button()
            {
                Text = "",
                Height = 20,
                Width = 20
            };
```

```
            btnRemoveTable.Click += BtnRemoveTable_Click;

            tlpHeader.Controls.Add(btnRemoveTable);

            //атрибуты таблицы
            foreach (var item in table.Fields)
            {
                Button btnField = new Button()
                {
                    Text = item.ToString(),
                    Dock = DockStyle.Top,
                    TextAlign = ContentAlignment.MiddleLeft,
                    Tag = item
                };
                btnField.Click += BtnField_Click;

                tPanel.Controls.Add(btnField);
            }

            //изменение размера таблицы в соответствии с
количеством и ширины строк
            int maxWidth = tPanel.Controls[0].Width;

            foreach (var item in table.Fields)
            {
                string s = item.ToString();
                Font f = tPanel.Font;

                Size controlSize = TextRenderer.MeasureText(s,
f);
                controlSize.Width += 30;

                if (controlSize.Width > maxWidth)
                    maxWidth = controlSize.Width;
            }

            tPanel.Size = new Size(maxWidth, tPanel.Size.Height);

            //сортируем атрибуты
            for (int i = 0; i < tPanel.Controls.Count; i++)
            {
                tPanel.Controls[i].BringToFront();
            }

            //отображаем таблицу в Конструкторе
            panelQuery.Controls.Add(tPanel);
            ReplaceControls(panelQuery);
        }
```

Приложение Б

Листинг метода AddField

```
private void AddField(Field field)
        {
            //если полей больше 10, то больше не добавляем
            if (panelFields.Controls.Count == 10)
            {
                MessageBox.Show($"Достигнуто максимальное число
полей: {maxFieldsCount}");
                return;
            }

            Panel fieldPanel = new Panel()
            {
                Size = new Size(170, 150),
                BorderStyle = BorderStyle.FixedSingle,
                Tag = field
            };

            Button btnRemoveField = new Button()
            {
                Text = "",
                Height = 22,
                Width = 20,
                Location = new Point(141, 5)
            };
            btnRemoveField.Click += BtnRemoveField_Click;
            fieldPanel.Controls.Add(btnRemoveField);

            fieldPanel.Controls.Add(new TextBox()
            {
                Size = new Size(130, 20),
                Location = new Point(8, 6),
                Text = field.Name,
                ReadOnly = true
            });
            fieldPanel.Controls.Add(new TextBox()
            {
                Size = new Size(152, 20),
                Location = new Point(8, 34),
                Text = field.Parent.Name,
                ReadOnly = true
            });
            ComboBox cbSorting = new ComboBox()
            {
                Size = new Size(152, 21),
                Location = new Point(8, 66),
                Tag = field
            };
```

```
            cbSorting.Items.AddRange(new    string[]    {    "по
возрастанию", "по убыванию", "(отсутствует)" });
            cbSorting.SelectedIndexChanged                      +=
CbSorting_SelectedIndexChanged;

            fieldPanel.Controls.Add(cbSorting);

            CheckBox checkOutputOnScreen = new CheckBox()
            {
                Size = new Size(21, 24),
                Location = new Point(75, 95),
                Checked = true,
                Tag = field
            };
            checkOutputOnScreen.CheckedChanged                  +=
CheckOutputOnScreen_CheckedChanged;
            fieldPanel.Controls.Add(checkOutputOnScreen);

            TextBox tbCondition = new TextBox()
            {
                Size = new Size(152, 20),
                Location = new Point(8, 122),
                Name = "tbCondition"
            };
            tbCondition.TextChanged += TbCondition_TextChanged;
            fieldPanel.Controls.Add(tbCondition);

            panelFields.Controls.Add(fieldPanel);

            ReplaceControls(panelFields);
        }
```

Приложение В

Листинг метода BuildSqlQuery

```
private void BuildSqlQuery()
        {
            List<Table> selectedTables = new List<Table>();
            List<Field> selectedFields = new List<Field>();

            foreach (Panel tablePanel in panelQuery.Controls)
            {
                Table table = tablePanel.Tag as Table;

                if    (selectedTables.Where(t    =>    t.Name    ==
table.Name).FirstOrDefault() == null)
                    selectedTables.Add(table);
            }

            foreach (Panel fieldPanel in panelFields.Controls)
            {
                Field field = fieldPanel.Tag as Field;

                if (field.IsOutputOnScreen)
                    selectedFields.Add(field);
            }

            string sql = "SELECT ";

            if (selectedFields.Count == 0)
                sql += "*";
            else
            {
                foreach (Field field in selectedFields)
                {
                    sql  +=  $"{field.Parent.Name}.{field.Name}  as
{field.Parent.Name}_{field.Name},";
                }

                sql = sql.Remove(sql.Length - 1, 1);
            }

            sql += " FROM ";

            foreach (Table table in selectedTables)
            {
                sql += $"{table.Name},";
            }

            sql = sql.Remove(sql.Length - 1, 1);

            List<string> relations = new List<string>();
            List<string> conditions = new List<string>();
```

```
            //смотрим, есть ли связи между выбранными таблицами
            for (int i = 0; i < selectedTables.Count; i++)
            {
                Table childTable = selectedTables[i];

                if (childTable.Parents.Count == 0)
                    continue;

                for (int j = 0; j < selectedTables.Count; j++)
                {
                    Table parentTable = selectedTables[j];

                    if (parentTable == childTable)
                        continue;

                    if (childTable.Parents.Where(p => p.Name ==
parentTable.Name).FirstOrDefault() != parentTable)
                        continue;

                    //нашли связь
                    if (childTable.Parents.Where(p => p.Name ==
parentTable.Name).FirstOrDefault() != null)
                    {
                        Field          referencedField          =
parentTable.Fields.Where(f => f.IsReferenced).FirstOrDefault();
                        Field         referencingField          =
childTable.Fields.Where(f => f.IsReferencing).FirstOrDefault();


relations.Add($"{parentTable.Name}.{referencedField.Name}       =
{childTable.Name}.{referencingField.Name}");
                    }
                }
            }

            //смотрим, есть ли прочие условия, выставленные
пользователем
            foreach (Panel fieldPanel in panelFields.Controls)
            {
                TextBox               tbCondition               =
fieldPanel.Controls.Find("tbCondition", true).FirstOrDefault() as
TextBox;

                //если условие задано
                if (tbCondition.TextLength > 0)
                {
                    Field field = fieldPanel.Tag as Field;

conditions.Add($"{field.Parent.Name}.{field.Name}               =
'{tbCondition.Text}'");
```

```
                }
            }

            sql += " WHERE ";

            if (relations.Count > 0)
            {
                foreach (string relation in relations)
                {
                    sql += $"{relation} AND ";
                }

                sql = sql.Remove(sql.Length - 5, 5);
            }
            if (conditions.Count > 0)
            {
                if (relations.Count > 0)
                    sql += " AND ";
                foreach (string condition in conditions)
                {
                    sql += $"{condition} AND ";
                }
                sql = sql.Remove(sql.Length - 5, 5);
            }
            if (relations.Count == 0 && conditions.Count == 0)
                sql = sql.Replace(" WHERE ", "");
            //определяем сортировку полей, если имеется
            if  (selectedFields.Where(f  =>  f.SortingOrder  !=
SortingOrder.None).Any())
            {
                sql += " ORDER BY ";
                foreach (Field field in selectedFields)
                {
                    if (field.SortingOrder == SortingOrder.Asc)
                        sql += $"{field.Parent.Name}.{field.Name}
ASC, ";
                    else     if     (field.SortingOrder     ==
SortingOrder.Desc)
                        sql += $"{field.Parent.Name}.{field.Name}
DESC, ";
                }
                sql = sql.Remove(sql.Length - 2, 2);
            }

            rtbSql.Text = sql;
        }
```