МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК

ОТЧЕТ

О РЕЗУЛЬТАТАХ ГРУППОВОЙ

ПРАКТИКИ ПО ПОЛУЧЕНИЮ ПРОФЕССИОНАЛЬНЫХ УМЕНИЙ И ОПЫТА ПРОФЕССИОНАЛЬНОЙ ДЕЯТЕЛЬНОСТИ

ГАОУ ТО ДПО "Тюменский областной государственный институт развития регионального образования"

(наименование организации)

|  |  |  |
| --- | --- | --- |
|  | | |
| Выполнил  обучающийся 3 курса,  22МОиАИС184-1 группы |  | Ибраев Ерлан Иржанович |
| Выполнил  обучающийся 3 курса,  22МОиАИС184-1 группы | (подпись) | (ФИО)  Батт Артем Андреевич |
| Выполнил  обучающийся 3 курса,  22МОиАИС184-1 группы | (подпись) | (ФИО)  Зайцев Данил Евгеньевич |
|  | (подпись) | (ФИО) |
| Руководитель практики  от института / школы  Зав. каф, доцент, к. т. н. |  | Воробьева Марина Сергеевна |
|  | (подпись) | (ФИО) |
| Руководитель практики  от профильной организации  Инженер |  | Плотоненко Юрий Анатольевич |
|  | (подпись) | (ФИО) |

Защищен 05.08.2021

Результаты

экзамена / зачета \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись руководителя практики от института / школы)

**Содержание**

[**Разработка веб-сервиса для формирования отчетов по результатам итоговой  
аттестации школьников.**](#_mz788pmyr61m) **2**

[Введение](#_h1uzacahxvfp) 2

[Постановка задачи](#_mmunjtxs0adz) 2

[**Глава 1. Предметная область**](#_v7shjierpfp9) **3**

[1.1 Описание деятельности предприятия](#_okdvw7lturtv) 3

[1.2 Отдел технического обеспечения СПОДУ](#_v6jm933eei1) 4

[1.3 Постановка задачи практики](#_u6gnaa3ng1d1) 5

[**Глава 2. Описание используемых алгоритмов, методов, процессов**](#_8i7o44gvt1rm) **7**

[2.1 Работа с БД сайта](#_xhtandrjgg20) 7

[2.2 Разработка приложения для выгрузки данных из бд](#_tw90mmixbv08) 20

[2.2.1. Подключение к базам данных](#_a0prwcyveti0) 20

[2.2.2. Алгоритмы выгрузки данных](#_mnd1iuips27a) 21

[2.2.3. Интерфейс приложения](#_12agx8qkir6c) 22

[2.3. Сервис загрузки данных в бд сайта](#_3tp80cvqd05x) 25

[2.4 Авторизация и регистрация пользователей через базу данных](#_94db0b2iy9xt) 26

[2.4.1 Общие сведения](#_ayai69bcgq33) 26

[2.4.2 Регистрация](#_vl9tusmf2wnl) 26

[2.4.2 Авторизация](#_tao30d8et5gy) 27

[2.4.3 Выход](#_pyaqrx5rhqn0) 28

[2.5. Конструктор построения отчетов](#_kt601heeh4td) 28

[2.3.1. Алгоритмы выгрузки данных](#_qhw97b98fwpt) 28

[2.3.2. Интерфейс конструктора](#_i501uob5uox5) 30

[**Заключение**](#_x17nfjb02r7t) **31**

[**Список литературы**](#_fvkz5a66cnlu) **32**

[**Приложение**](#_n9vsruvrb8na) **33**

# Разработка веб-сервиса для формирования отчетов по результатам итоговой аттестации школьников.

## Введение

В сфере образования статистика по результатам итоговой аттестации школьников играет важную роль, например, для оценки ситуаций в отдельных школах и дальнейшего улучшения качества образования.

Данный проект решает несколько проблем:

* проблему ручного составления отчетов, которое занимает большое количество времени и ресурсов
* проблему отсутствия наглядного визуального представления, которое позволяет оценить текущее качество образования в образовательных учреждениях.

Проект призван автоматизировать формирование отчетов по данным итоговой аттестации школьников с возможностью графического представления статистики.

Визуализированная статистика результатов итоговой аттестации позволит наглядно оценить уровень образования в школах Тюменской области.

## Постановка задачи

**Цель**: Разработать веб-сайт с возможностью автоматического создания отчетов по

результатам итоговой аттестации школьников Тюменской области.

**Задачи:**

* Ознакомиться с реальными примерами отчетов, предоставляемых центром оценки качества образования. (Зайцев Данил, Батт Артем)
* Реализовать конструктор запросов к базе данных для выборки информации, с ее последующей визуализацией. (Зайцев Данил)
* Реализовать конструктор построения отчетов на основе выбранных данных. (Зайцев Данил)
* Реализовать авторизацию и регистрацию пользователей через базу данных.(Ибраев Ерлан)
* Реализовать пользовательский интерфейс с определенным уровнем доступа.(Ибраев Ерлан)
* Реализовать win32 приложение с выбором данных для выгрузки в файлы сайта из баз данных в закрытой сети.(Батт Артём)
* Реализовать загрузку данных из файлов сайта в базу данных сайта.(Батт Артем)
* Реализовать веб-сайт для просмотра автоматически созданного отчета. (Зайцев Данил, Батт Артем, Ибраев Ерлан)

**Функционал ролей пользователей:**

На веб-сайте должны присутствовать 2 вида пользователей:

1. Администратор - имеет возможность загружать данные из файлов сайта в бд, просматривать данные в пределах областей, а также создавать кастомные шаблоны отчетов.
2. Пользователь в рамках образовательного учреждения имеет возможность формировать отчеты по данным определенного образовательного учреждения.

# Глава 1. Предметная область

## 1.1 Описание деятельности предприятия

ТОГИРРО расшифровывается как «Тюменский областной государственный институт развития регионального образования».

Цель ТОГИРРО – определение стратегии развития образовательной системы области и ее научно-методическое обеспечение на основе мониторинговых исследований.

Деятельность института:

1. Информационно-аналитическая деятельность.
2. Проектно-прогностическая деятельность, обеспечение стратегического развития отрасли.
3. Подготовка и переподготовка педагогических и управленческих кадров, специалистов органов управления образованием, специалистов финансово-экономических служб образовательных учреждений и организаций,научно-педагогических работников.
4. Организационно-педагогическая деятельность.

## 1.2 Отдел технического обеспечения СПОДУ

Отдел технического обеспечения стандартизированных процедур создан для выполнения следующих функций:

* создание проектов на основе машиночитаемых форм;
* информационно-технологическое сопровождение проведения ГИА (государственной итоговой аттестации) 9 и 11 классов;
* создание и ведение региональных баз данных образовательных учреждений;
* создание и ведение баз данных участников ГИА;
* организация обработки данных с бланков ГИА;
* создание и ведение баз данных результатов ГИА по субъекту Федерации;
* обеспечение безопасности и комплексной защиты информации.

Одной из основных функций отдела является техническое сопровождение процедуры проведения ГИА, которая включает в себя следующие этапы:

1. формирование региональной базы участников ГИА в Тюменской области;
2. распределение участников ГИА по пунктам проведения экзамена;
3. обработка бланков ГИА (сканирование, верификация);
4. отправка результатов обработки бланков в Федеральный Центр тестирования;
5. получение и расшифровка результатов тестирования;
6. подведение итогов экзамена и формирование отчетной документации.

Информация, поступающая в отдел технического обеспечения СПОДУ:

* приказы, распоряжения и инструктивные материалы из Департамента образования и Федерального центра тестирования;
* списки учащихся, учителей, учебной литературы из муниципальных образовательных учреждений (МОУ СОШ);
* схемы итоговой аттестации учащихся из муниципальных органов управления образованием (МОУО);
* результаты ГИА из Федерального Центра тестирования.

Информация, выходящая из отдела технического обеспечения СПОДУ:

* электронные версии бланков для последующей обработки и проверки направляются в Федеральный Центр тестирования;
* результаты ГИА направляются в муниципальные органы управления образованием и в образовательные учреждения;
* статистическая отчетность по результатам итоговой аттестации направляется в Федеральное агентство по образованию, Департамент образования и науки Тюменской области и муниципальные органы управления образования;
* схемы итоговой аттестации учащихся направляются начальнику отдела и администраторам для формирования федеральной базы данных по проведению ГИА;
* списки учащихся, организаторов в ППЭ, сформированные перед экзаменом по представленной схеме проведения итоговой аттестации административной группой;
* электронные версии бланков после проведения экзамена направляются в Федеральный Центр тестирования (ФЦТ) для последующей обработки и проверки;
* результаты ГИА, полученные из ФЦТ, направляются в МОУО и МОУ;
* статистическая отчетность по результатам экзаменов направляется в Федеральное агентство по образованию, Департамент образования и науки Тюменской области, муниципальные органы управления образования.

## 1.3 Постановка задачи практики

Конечной целью прохождения производственной практики является создание веб-приложения, представляющего функционал для получения данных о результатах экзаменов и на основе этих данных формирования различных отчетов. Среди основных задач для достижения вышеуказанной цели выделяются следующие:

1. Реализация пользовательского интерфейса с определенным уровнем доступа.
2. Реализация авторизации и регистрации пользователей через базу данных.
3. Реализация функционала создания базовых и пользовательских шаблонов для отчетов.
4. Реализация построения диаграмм и графиков к отчетам.
5. Реализация интерфейса администратора для управления загрузкой данных.
6. Реализация приложения для выгрузки данных из бд в закрытой сети.

Проект основывается на данных, полученных на предприятии:

* Базы данных erbd\_ege\_reg\_21\_72 и erbd\_gia\_reg\_21\_72, содержащие полную информацию об аттестации школьников Тюменской области в 2021 году по ЕГЭ, ОГЭ и ГВЭ. Имена и фамилии школьников были заранее изменены. Каждая бд содержит больше 350 таблиц. В совокупности бд занимают 26.2 гигабайт дискового пространства.
* Прототип базы данных сайта, включающий 18 таблиц. Эта база подлежит переработке для использования в качестве основной бд сайта. Необходимо изменить, добавить и удалить таблицы для возможности аутентификации и авторизации пользователей, хранения информации о шаблонах отчетов, удовлетворения требований к данным, необходимым для формирования отчетов.
* pdf документ с описанием прототипа бд сайта.
* “Книга по ЕГЭ 2012” - архив документов в форматах .doc и .xls, которые содержат примеры боевых отчетов. Информация полезна для разработки шаблонов и конструктора автоматизированного формирования отчетов.

Используемые технологии:

* СУБД Microsoft SQL Server
  + причина использования: изначально базы данных были предоставлены в форматах, используемых в Microsoft SQL Server
  + цель использования: работа с базами данных: изучение, изменение, удаление и добавление таблиц.
* HTML, CSS
  + причина использования: стандартизированные инструменты для верстки сайтов
  + цель использования: верстка страниц сайта, каркас и стили для веб-приложения
* JavaScript
  + причина использования: современный и активно развивающийся язык программирования, заточенный под работу с вебом
  + цель использования: разработка динамических элементов сайта: построение отчетов, работа с Excel
* PHP
  + причина использования: наличие всеобъемлющей документации
  + цель использования: написание бэкенда веб-приложения (Аутентификация и авторизация пользователей)
* C#, WPF
  + причина использования: наиболее полная поддержка функционала взаимодействия с базами данных в SQL Server, т.к. технологии разрабатываются и поддерживаются самой Microsoft, как и СУБД, с которой необходимо взаимодействовать
  + цель использования: разработка приложения для выгрузки данных из бд с использованием фреймворка ASP.NET, ORM Entity Framework Core 5, Json.NET
* Python, Flask
  + причина использования: актуальность технологий, а также наличие опыта использования инструментов в проекте
  + цель использования: разработка загрузчика файлов формата json в бд сайта

# Глава 2. Описание используемых алгоритмов, методов, процессов

## 2.1 Работа с БД сайта

**Таблицы БД сайта**

|  |  |  |  |
| --- | --- | --- | --- |
| № | БД сайта | | Соответствующие таблицы в базах егэ и гиа |
| Название таблицы | Число строк | Название таблиц |
| 1 | answers | 457157 | res\_Marks |
| 2 | competencies | 140 | админ заполняет вручную |
| 3 | difficulties | 3 | админ заполняет вручную |
| 4 | districts | 26 | rbd\_Areas |
| 5 | schools | 365 | rbd\_Schools |
| 6 | exams | 24064 | res\_HumanTests, res\_Marks |
| 7 | examtypes | 3 | админ заполняет вручную |
| 8 | governments | 27 | Таблица удалена |
| 9 | parts | 2 | админ заполняет вручную |
| 10 | requirements | 25 | админ заполняет вручную |
| 11 | schoolkinds | 54 | rbdc\_SchoolKinds |
| 12 | schoolproperties | 14 | rbdc\_SchoolProperties |
| 13 | schooltypes | 7 | rbdc\_SchoolTypes |
| 14 | specifications | 19 | админ заполняет вручную |
| 15 | specscompets | 116 | админ заполняет вручную |
| 16 | specsrequirements | 80 | админ заполняет вручную |
| 17 | subjects | 33 | админ заполняет вручную |
| 18 | testschema | 69 | админ заполняет вручную |
| 19 | towntypes |  | Таблица добавлена |
| 20 | participantcategories |  | Таблица добавлена |

**Диаграмма изначальной БД сайта**

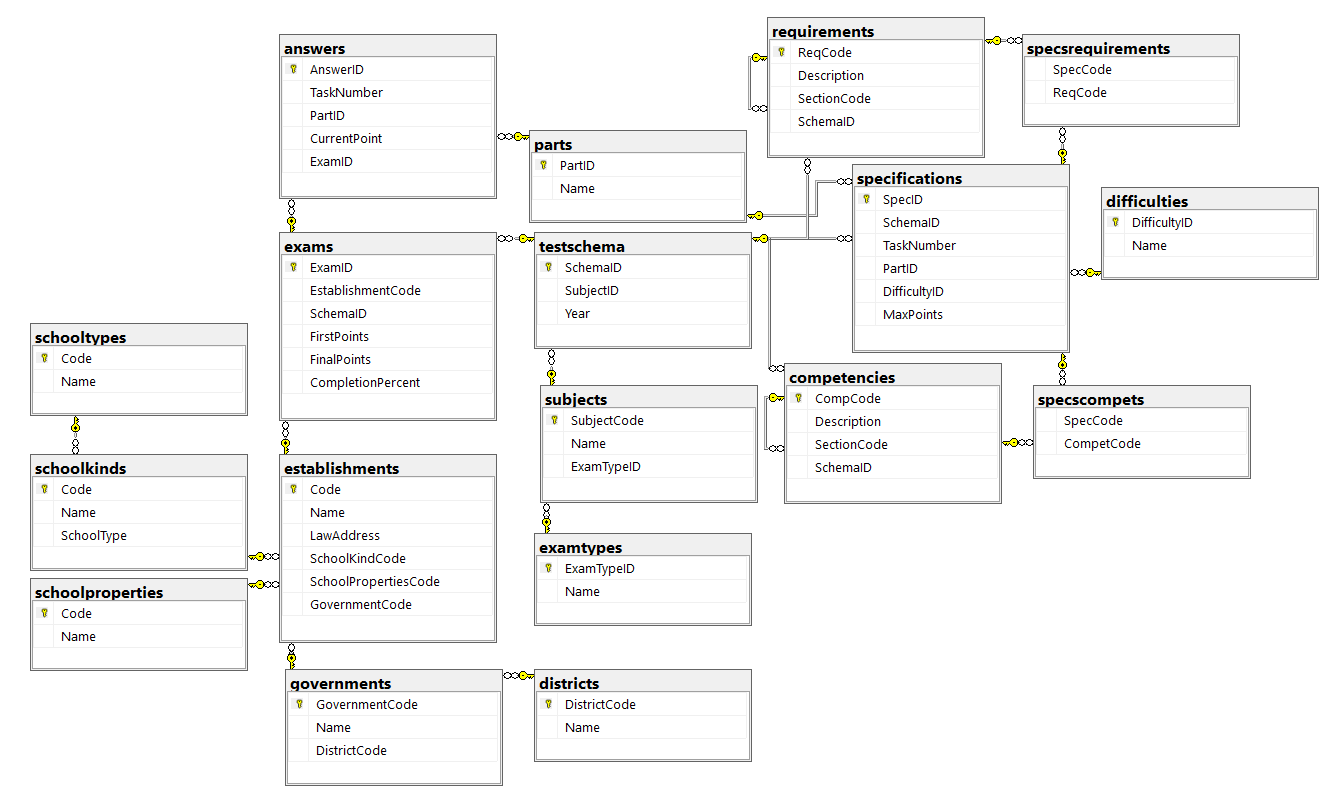
****

Рис. 1. Диаграмма изначальной БД сайта

**Описание работы с каждой таблицей прототипа БД сайта**

1. **Таблица answers** (одинаковая структура таблиц в БД егэ и гиа, разные данные об ответах на задания)

* БД сайта

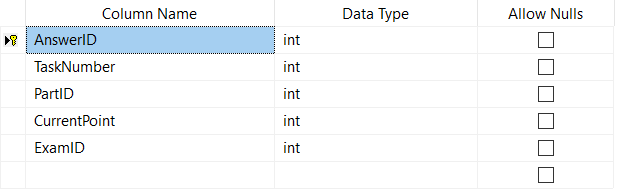


Рис. 2. Таблица answers в БД сайта

Содержит данные таблицы res\_Marks

ВЫГРУЗКА ДАННЫХ В ТАБЛИЦУ ANSWERS:

1. Удалить данные таблицы
2. Выгрузить данные из бд егэ **и** гиа из таблицы: dbo.res\_Marks; из столбцов: TestResultB, TestResultC, TestResultD
3. Для выгрузки в столбец CurrentPoint нужно парсить TestResultB(плюс = 1, минус = 0, число = число); в TestResultC и TestResultD убрать скобки и их содержимое
4. Если задание из TestResultB => PartID = 1, TestResultC => PartID = 2, TestResultD => PartID = 3
5. AnswerID формируется при выгрузке

**2.** **Таблица competencies**

* БД сайта

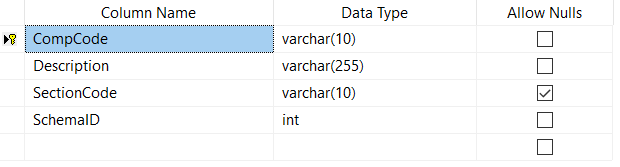


Рис. 3. Таблица competencies в БД сайта

Администратор заполняет вручную по кодификаторам ФИПИ

**3.** **Таблица difficulties**

* БД сайта

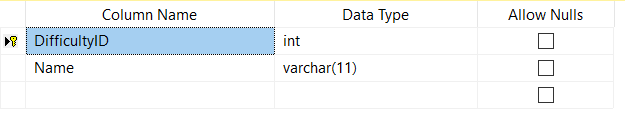


Рис. 4. Таблица difficulties в БД сайта

Администратор заполняет вручную согласно кодификаторам ФИПИ

**4.** **Таблица areas**(одинаковая структура таблиц в БД егэ и гиа, одинаковые данные во всех БД)

* БД сайта

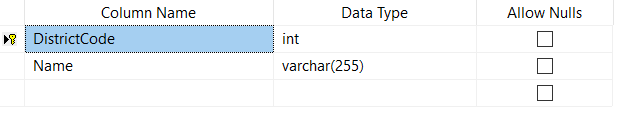


Рис. 5. Таблица areas в БД сайта

* Измененная таблица

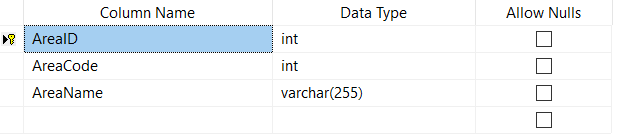


Рис. 6. Измененная таблица areas в БД сайта

ВЫГРУЗКА ДАННЫХ В ТАБЛИЦУ AREAS:

1. Удалить данные этой таблицы
2. Добавить столбцы: AreaID
3. Выгрузить данные из бд егэ **или** гиа из таблицы dbo.rbd\_Areas из столбцов: AreaCode, AreaName, AreaID формируется в коде

**5.** **Таблица schools**(одинаковая структура таблиц в БД егэ и гиа, разные данные о школах)

* БД сайта

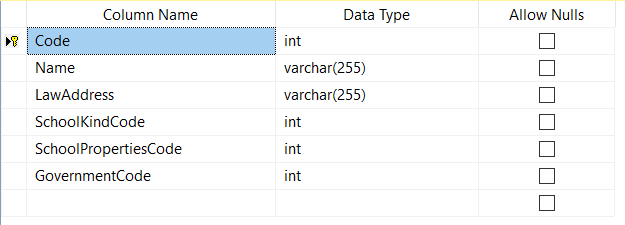


Рис. 7. Таблица schools в БД сайта

* Измененная таблица

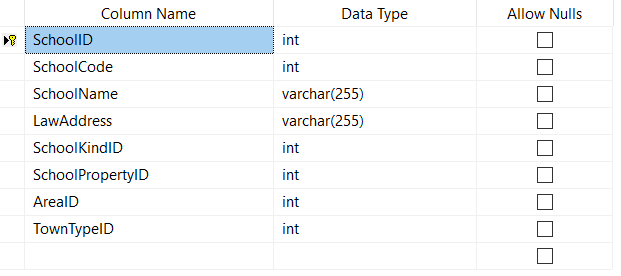


Рис. 8. Измененная таблица schools в БД сайта

Содержит данные таблицы rbd\_Schools

ВЫГРУЗКА ДАННЫХ В ТАБЛИЦУ SCHOOLS:

1. Удалить данные из таблицы
2. Добавить столбцы: SchoolID, AreaID, TownTypeID
3. Выгружать школы, у которых DeleteType = 0
4. Выгрузить данные из бд егэ **и** гиа из таблицы dbo.rbd\_Schools из столбцов: SchoolCode, ShortName, LawAddress, SchoolKindFK, SchoolPropertyFk, TownTypeFk

**6.** **Таблица exams**(одинаковая структура таблиц в БД егэ и гиа, разные данные об экзаменах)

* БД сайта

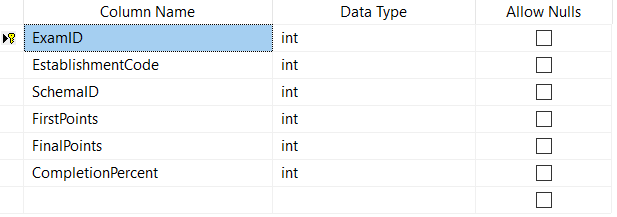


Рис. 9. Таблица exams в БД сайта

* Измененная таблица

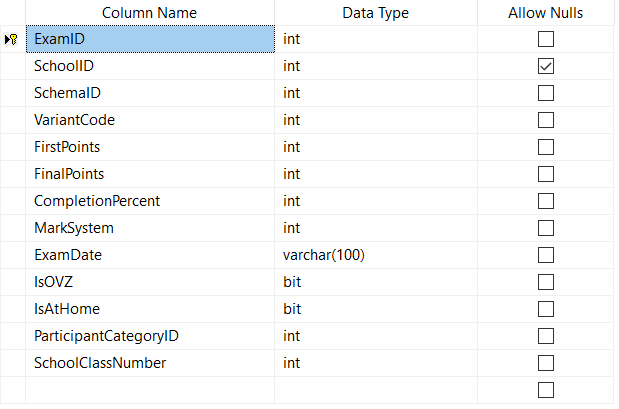


Рис. 10. Измененная таблица exams в БД сайта

Содержит данные таблицы res\_HumanTests, rbd\_Schools, res\_Marks, rbd\_ParticipantProperties, rbd\_ParticipantCategories

ВЫГРУЗКА ДАННЫХ В ТАБЛИЦУ EXAMS:

1. Удалить данные из таблицы
2. Добавить столбцы: ExamDate, ОВЗ(Property = 6), на дому(StationCode > 899), ParticipantCategoryID, VariantCode, MarkSystem, SchoolClassNumber
3. TestTypeCode in (ege: 4 or 5; gia: 6 or 7), SubjectCode < 50 ЕГЭ, > 50 ОГЭ
4. Выгрузить данные из бд егэ **и** гиа из таблицы dbo.res\_HumanTests из столбцов: HumanTestID, SchoolFK, SchemaID, VariantCode, PrimaryMark, (если ЕГЭ: Mark100, остальное: Mark5), PercentMark, (5, 100), ExamDate, (if property = 6 then 1 else 0), (if StationCode > 899 then 1 else 0)
5. Выгружать данные, где поле ProcessCondition = 6
6. SchemaID добавляется при загрузке данных, если данных в таблице testschema нет - выводится сообщение о необходимости заполнения таблицы администратором

**7.** **Таблица examtypes**(одинаковая структура таблиц в БД егэ и гиа, разные данные о типах экзаменов)

* БД сайта

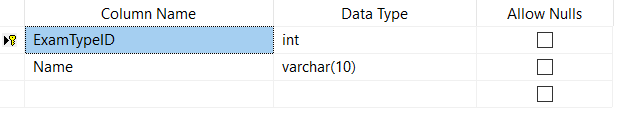


Рис. 11. Таблица examtypes в БД сайта

* Измененная таблица

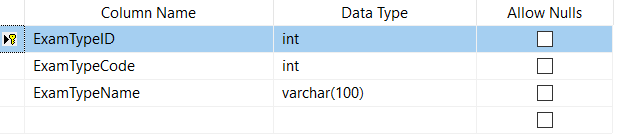


Рис. 12. Измененная таблица examtypes в БД сайта

Администратор заполняет вручную согласно кодификаторам ФИПИ

**8.** **Таблица governments**(одинаковая структура таблиц в БД егэ и гиа, одинаковые данные)

Таблица определена как излишняя и подлежит удалению

**9.** **Таблица parts**

* БД сайта

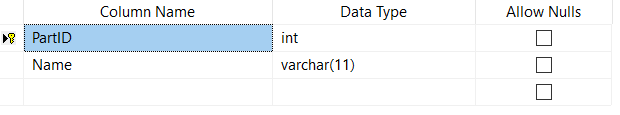


Рис. 12. Таблица parts в БД сайта

* Измененные данные

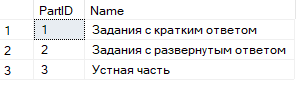


Рис. 13. Данные таблицы part в БД сайта

Администратор заполняет вручную

**10.** **Таблица requirements**

* БД сайта

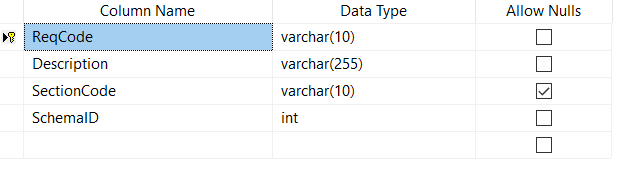


Рис. 14. Таблица requirements в БД сайта

Администратор заполняет вручную по кодификаторам ФИПИ

**11.** **Таблица schoolkinds**(одинаковая структура таблиц в БД егэ и гиа, разные данные о видах школ(егэ 59 записей, гиа 60 записей))

* БД сайта

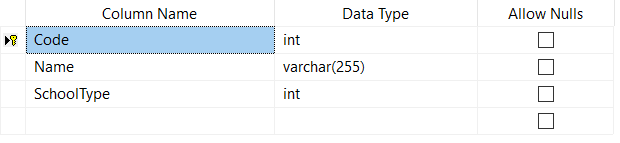


Рис. 15. Таблица schoolkinds в БД сайта

* Измененная таблица

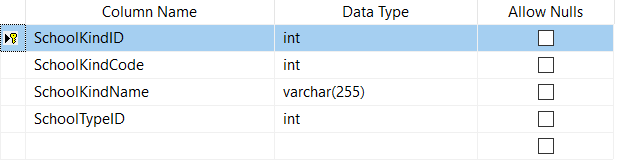


Рис. 16. Измененная таблица schoolkinds в БД сайта

Различие данных в базах(один и тот же вид школы ссылается на различный тип школы):

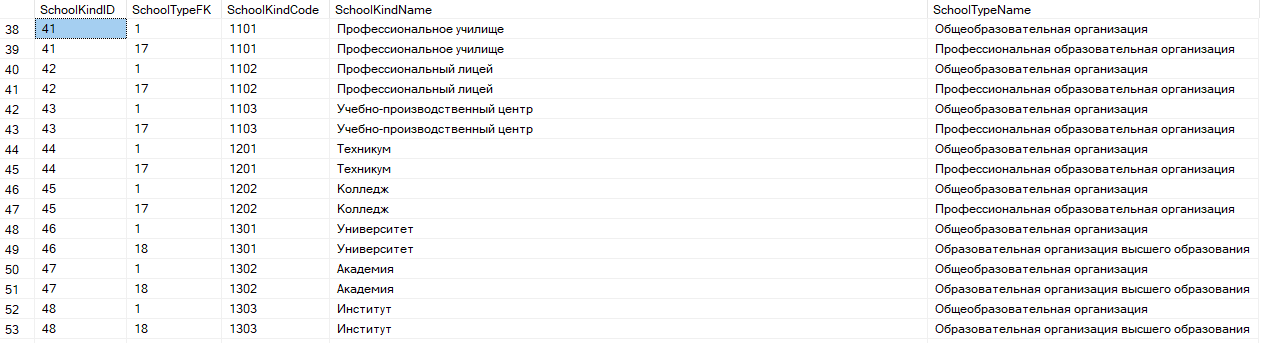


Рис. 17. Таблица schoolkinds в БД сайта

Содержит данные таблицы rbdc\_SchoolKinds

ВЫГРУЗКА ДАННЫХ В ТАБЛИЦУ SCHOOLKINDS:

1. Удалить данные из таблицы
2. Добавить столбцы: SchoolKindID
3. Выгрузить данные из бд егэ **и** гиа из таблицы dbo.rbdc\_SchoolKinds из столбцов: SchoolKindID, SchoolKindCode, SchoolKindName, SchoolTypeFK

**12.** **Таблица schoolproperties**(одинаковая структура таблиц в БД егэ и гиа, одинаковые данные)

* БД сайта

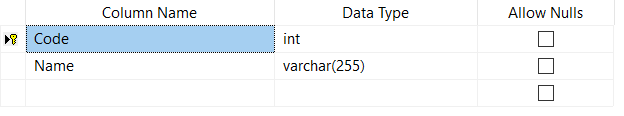


Рис. 18. Таблица schoolproperties в БД сайта

* Измененная таблица

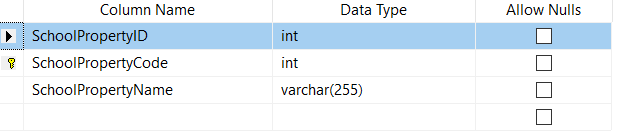


Рис. 19. Измененная таблица schoolproperties в БД сайта

Содержит данные таблицы rbdc\_SchoolProperties

ВЫГРУЗКА ДАННЫХ В ТАБЛИЦУ SCHOOLPROPERTIES:

1. Удалить данные из таблицы
2. Добавить столбцы: SchoolPropertyID
3. Выгрузить данные из бд егэ **или** гиа из таблицы dbo.rbdc\_SchoolProperties из столбцов: SchoolPropertyID, SchoolPropertyCode, SchoolPropertyName

**13.** **Таблица schooltypes**(одинаковая структура таблиц в БД егэ и гиа, одинаковые данные)

* БД сайта

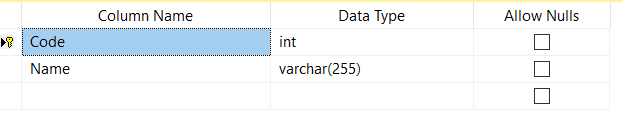


Рис. 20. Таблица schooltypes в БД сайта

* Измененная таблица

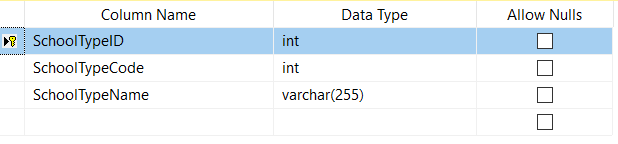


Рис. 21. Измененная таблица schooltypes в БД сайта

Содержит данные таблицы rbdc\_SchoolTypes

ВЫГРУЗКА ДАННЫХ В ТАБЛИЦУ SCHOOLTYPES:

1. Удалить данные из таблицы
2. Добавить столбцы: SchoolTypeID
3. Выгрузить данные из бд егэ **или** гиа из таблицы dbo.rbdc\_SchoolTypes из столбцов: SchoolTypeID, SchoolTypeCode, SchoolTypeName

**14.** **Таблица specifications**

* БД сайта

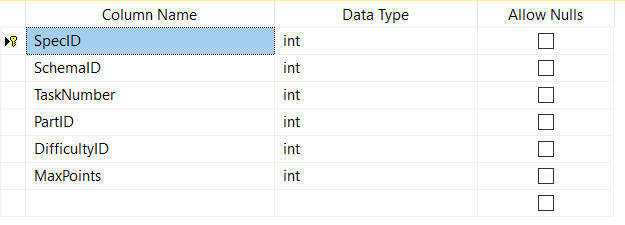


Рис. 22. Таблица specifications в БД сайта

Администратор заполняет вручную согласно кодификаторам ФИПИ

**15.** **Таблица specscompets**(ассоциативная таблица, соединяет specifications и competencies связью “многие ко многим”)

* БД сайта

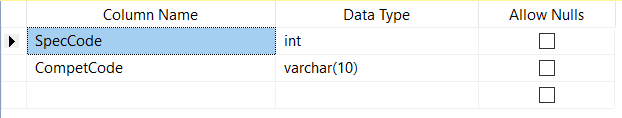


Рис. 23. Таблица specscompets в БД сайта

Администратор заполняет вручную согласно кодификаторам ФИПИ

**16.** **Таблица specsrequirements**(ассоциативная таблица, соединяет specifications и requirements связью “многие ко многим”)

* БД сайта

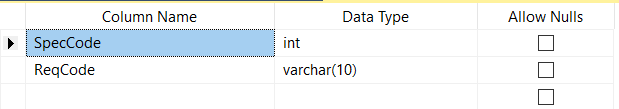


Рис. 24. Таблица specsrequirements в БД сайта

Администратор заполняет вручную согласно кодификаторам ФИПИ

**17.** **Таблица subjects**(одинаковая структура таблиц в БД егэ и гиа, разные данные о предметах(егэ 40 записей, гиа 39 записей))

* БД сайта

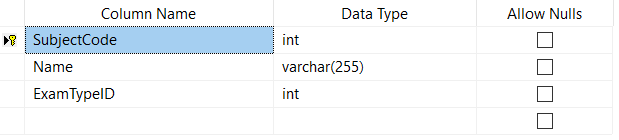


Рис. 25. Таблица subjects в БД сайта

* Измененная таблица

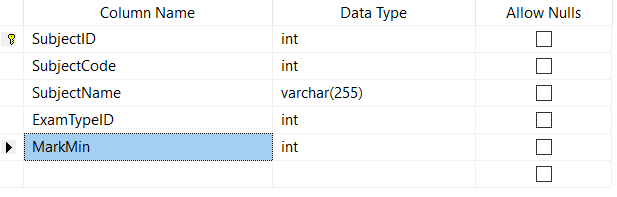


Рис. 26. Измененная таблица subjects в БД сайта

Содержит данные таблицы dat\_Subjects

**18.** **Таблица testschema**

* БД сайта

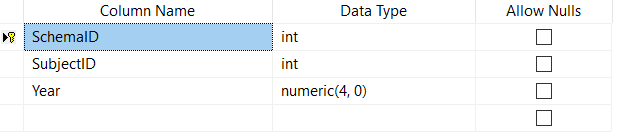


Рис. 27. Таблица testschema в БД сайта

Администратор заполняет вручную

**19.** **Таблица towntypes**(одинаковая структура таблиц в БД егэ и гиа, одинаковые данные)

Таблица добавлена в бд сайта

* БД сайта

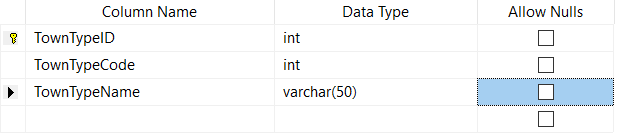


Рис. 28. Таблица towntyopes в БД сайта

ВЫГРУЗКА ДАННЫХ В ТАБЛИЦУ TOWNTYPES:

1. Добавить таблицу
2. Выгрузить данные из бд егэ **или** гиа из таблицы dbo.rbdc\_TownTypes из столбцов: TownTypeID, TownTypeCode, TownTypeName

**20.** **Таблица participantcategories**(одинаковая структура таблиц в БД егэ и гиа, разные данные(две категории в кодом 8))

Таблица добавлена в бд сайта

* БД сайта

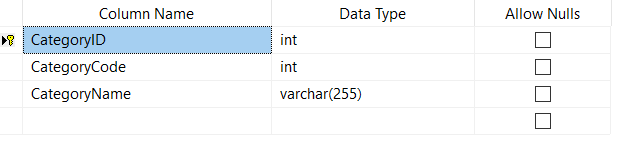


Рис. 28. Таблица towntypes в БД сайта

ВЫГРУЗКА ДАННЫХ В ТАБЛИЦУ PARTICIPANTCATEGORIES:

1. Добавить таблицу
2. Выгрузить данные из бд егэ **и** гиа из таблицы dbo.rbdc\_ParticipantCategories из столбцов: CategoryCode, CategoryName, CategoryID формируется при выгрузке

**Диаграмма итоговой БД сайта, подготовленной для выгрузки данных**

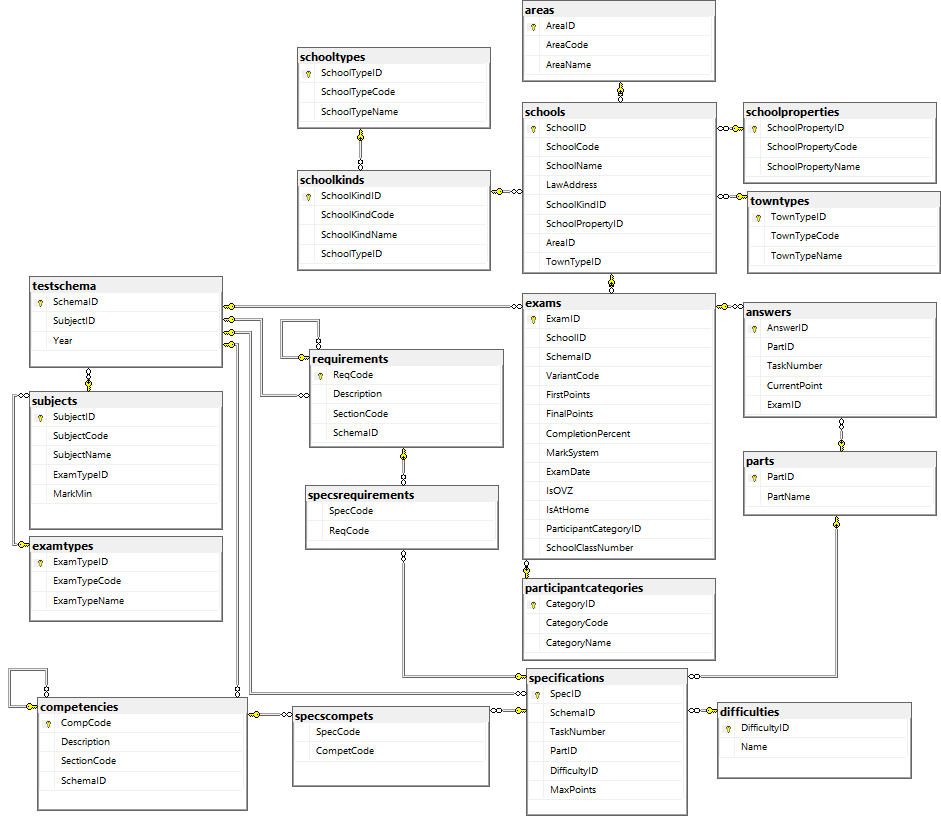


Рис. 29. Диаграмма итоговой БД сайта

## 2.2 Разработка приложения для выгрузки данных из бд

### 2.2.1. Подключение к базам данных

Подключение к базам данных Microsoft SQL Server осуществлялось на языке C# с помощью классов-контекстов для каждой базы. Необходимые классы-контексты и классы таблиц были построены с помощью утилиты Scaffold-DbContext из консоли пакетного менеджера Nuget.

Построение контекста и классов таблиц для базы егэ:

Scaffold-DbContext -Connection "Data Source=DESKTOP-TH1IT1Q;Initial Catalog=erbd\_ege\_reg\_21\_72;Integrated Security=True;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=ReadWrite;Trusted\_Connection=True;" Microsoft.EntityFrameworkCore.SqlServer -Tables dbo.res\_Answers, dbo.res\_Marks, dbo.rbd\_Areas, dbo.rbd\_Schools, dbo.res\_HumanTests, dbo.rbd\_ParticipantProperties, dbo.rbd\_Participants, dbo.rbdc\_ParticipantCategories, dbo.rbdc\_SchoolKinds, dbo.rbdc\_SchoolProperties, dbo.rbdc\_SchoolTypes, dbo.dat\_Subjects, dbo.rbdc\_TownTypes, dbo.rbd\_SchoolClasses -OutputDir "Models\erbd\_ege\_reg\_21\_72" -ContextDir Context -Context EgeContext2172

Построение контекста для базы гиа:

Scaffold-DbContext -Connection "Data Source=DESKTOP-TH1IT1Q;Initial Catalog=erbd\_gia\_reg\_21\_72;Integrated Security=True;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=ReadWrite;Trusted\_Connection=True;" Microsoft.EntityFrameworkCore.SqlServer -ContextDir Context -Context GiaContext2172

### 2.2.2. Алгоритмы выгрузки данных

Всего было разработано 8 алгоритмов для выгрузки данных для 9 таблиц. Данные для выгрузки в одну таблицу могут содержаться в нескольких таблицах и в нескольких базах данных поэтому в запросах используются объединение и соединение таблиц. Каждый алгоритм состоит из трёх частей:

1. Запрос к базам данных
2. Запись данных в формате Json в объект StringBuilder
3. Запись полученной строки в файл

Пример запроса для выгрузки в таблицу школ:

var schools = egeContext.RbdSchools

.Where(s => s.DeleteType == 0).Join(

egeContext.RbdAreas,

u => u.AreaFk,

c => c.AreaId,

(u, c) => new

{

SchoolId = u.SchoolId,

SchoolCode = u.SchoolCode,

ShortName = u.ShortName,

LawAddress = u.LawAddress,

SchoolKindFk = u.SchoolKindFk,

SchoolPropertyFk = u.SchoolPropertyFk,

TownTypeFk = u.TownTypeFk,

AreaID = c.AreaId,

AreaCode = c.AreaCode

}).ToList();

Пример запроса для выгрузки в таблицы экзаменов и ответов:

var egeprops = egeContext.RbdParticipantProperties.Where(p => p.Property == 6);

var query = (from tests in egeContext.ResHumanTests

join marks in egeContext.ResMarks

on tests.HumanTestId equals marks.HumanTestId

join participants in egeContext.RbdParticipants

on tests.ParticipantFk equals participants.ParticipantId

join subjects in egeContext.DatSubjects

on tests.SubjectCode equals subjects.SubjectCode

join categories in egeContext.RbdcParticipantCategories

on participants.ParticipantCategoryFk equals categories.CategoryId

join classes in egeContext.RbdSchoolClasses

on participants.SchoolClassId equals classes.SchoolClassId into gj1

from classes in gj1.DefaultIfEmpty()

join schools in egeContext.RbdSchools

on classes.SchoolId equals schools.SchoolId into gj2

from schools in gj2.DefaultIfEmpty()

join props in egeprops

on participants.ParticipantId equals props.ParticipantId into gj3

from props in gj3.DefaultIfEmpty()

select new

{

HumanTestID = tests.HumanTestId,

ParticipantID = tests.ParticipantFk,

TestTypeCode = tests.TestTypeCode,

SubjectCode = tests.SubjectCode,

ExamDate = tests.ExamDate,

StationCode = tests.StationCode,

VariantCode = tests.VariantCode,

ProcessCondition = tests.ProcessCondition,

PrimaryMark = marks.PrimaryMark,

PercentMark = marks.PercentMark,

Mark100 = marks.Mark100,

Mark5 = marks.Mark5,

PrimaryMarkB = marks.PrimaryMarkB,

TestResultB = marks.TestResultB,

PrimaryMarkC = marks.PrimaryMarkC,

TestResultC = marks.TestResultC,

PrimaryMarkD = marks.PrimaryMarkD,

TestResultD = marks.TestResultD,

ParticipantCode = participants.ParticipantCode,

ParticipantCategoryID = participants.ParticipantCategoryFk,

SchoolCode = schools == null ? -1 : schools.SchoolCode,

SchoolDeleteType = schools == null ? -1 : schools.DeleteType,

CategoryCode = categories.CategoryCode,

CategoryName = categories.CategoryName,

Property = props == null ? -1 : props.Property,

SubjectName = subjects.SubjectName,

}).Where(q => q.ProcessCondition == 6 && (q.TestTypeCode == 4 || q.TestTypeCode == 5) && (q.SchoolDeleteType == 0 || q.SchoolDeleteType == -1) && selectedSubjects.Any(s => s == q.SubjectName)).ToList();

Для ускорения записи данных используются низкоуровневые функции библиотеки NewtonsoftJson объекта JsonWriter. Ответы парсятся из строки с помощью регулярного выражения.

### 2.2.3. Интерфейс приложения

Сперва приложение предлагает выбрать из каких баз данных будут выгружаться сведения. Выбор осуществляется на вкладке “Базы данных”.

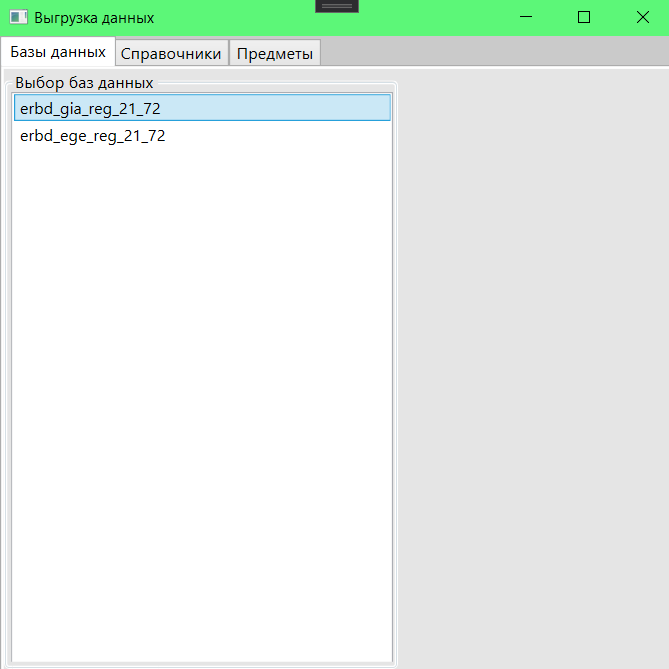


Рис. 30. Вкладка “Базы данных” приложения для выгрузки

Затем на вкладке “Справочники” можно выбрать и выгрузить необходимые справочники.

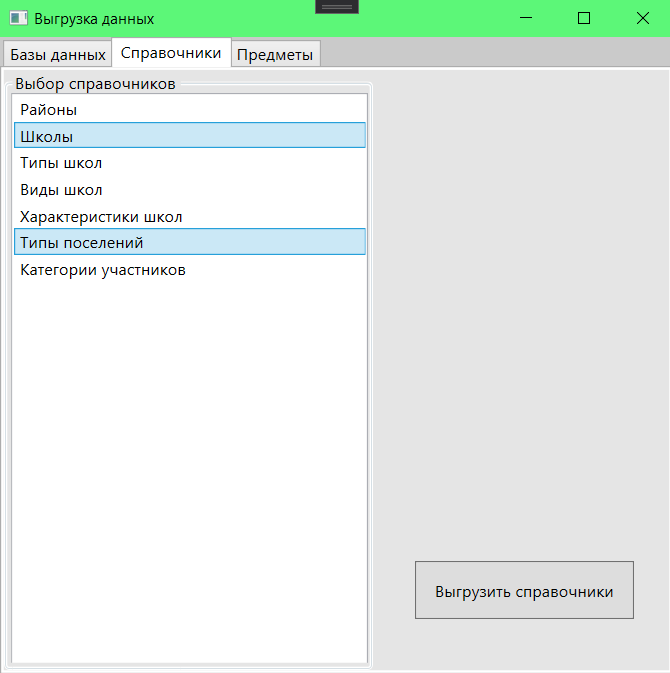


Рис. 31. Вкладка “Справочники” приложения для выгрузки

На вкладке “Предметы” можно выбрать предметы и выгрузить результаты экзаменов по ним. Список предметов отличается в зависимости от выбранных баз данных.

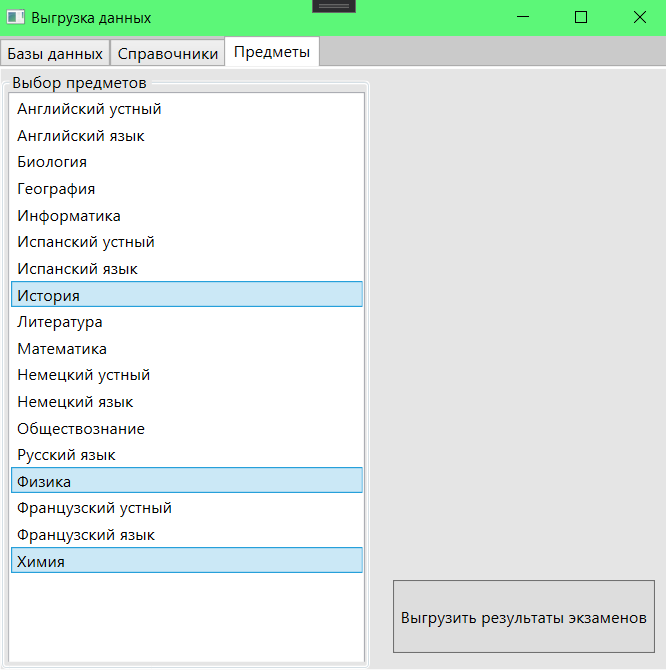


Рис. 32. Вкладка “Предметы” приложения для выгрузки

## 2.3. Сервис загрузки данных в бд сайта

Сервис был реализован с использованием языка Python и микрофреймворка Flask.

Подключенные библиотеки:

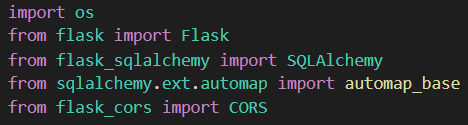


Рис. 33. Используемые библиотеки

С помощью функции automap\_base создаются типы данных для всех таблиц бд сайта.

Данные загружаются из файлов в формате json и помещаются в объекты созданных типов данных.

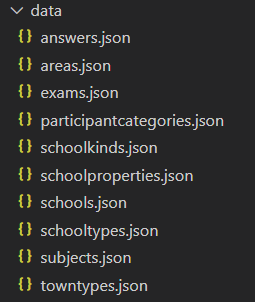


Рис. 34. Данные для загрузки

Пример алгоритма загрузки ответов:

def upload\_answers():

Answers = Base.classes.answers

need\_to\_commit = False

db.session.query(Answers).delete()

db.session.commit()

with open('data\\answers.json', 'r', encoding='utf-8') as json\_file:

answers = json.load(json\_file)

for s in answers:

obj = Answers()

obj.AnswerID = s['AnswerID']

obj.PartID = s['PartID']

obj.TaskNumber = s['TaskNumber']

obj.CurrentPoint = s['CurrentPoint']

obj.ExamID = s['ExamID']

db.session.add(obj)

if not need\_to\_commit:

need\_to\_commit = True

if need\_to\_commit:

db.session.commit()

## 2.4 Авторизация и регистрация пользователей через базу данных

### 2.4.1 Общие сведения

Авторизация была реализована с помощью программы MAMP для разработки и запуска динамических веб-сайтов. В нем существует система управление базы данными, путем настройки локальной среды разработки на портативных или настольных компьютерах без необходимости в отдельном веб-сервере. В MAMP использует веб-приложение PhpMyAdmin представляющее собой веб-интерфейс для администрирования СУБД MySQL. PhpMyAdmin позволяет через браузер и не только осуществлять администрирование сервера MySQL, запускать команды SQL и просматривать содержимое таблиц и баз данных. Пример алгоритма регистрации пользователя через базу данных.

### 2.4.2 Регистрация

Регистрировать пользователей может только администратор сайта.

Получаем все значения из полей регистрации и убираем все ненужные теги и пробелы.

<?php

$login = filter\_var(trim($\_POST['login']), FILTER\_SANITIZE\_STRING);

$name = filter\_var(trim($\_POST['name']), FILTER\_SANITIZE\_STRING);

$pass = filter\_var(trim($\_POST['pass']), FILTER\_SANITIZE\_STRING);

Проверяем логин, пароль и имя на длину. Если длина меньше 2, мы выводим сообщение об ошибке.

if(mb\_strlen($login) < 2 || mb\_strlen($login) > 90) {

echo "Недопустимая длина логина";

exit();

} else if(mb\_strlen($name) < 3 || mb\_strlen($name) > 50) {

echo "Недопустимая длина имени";

exit();

} else if(mb\_strlen($pass) < 2 || mb\_strlen($pass) > 6) {

echo "Недопустимая длина пароля (от 2 до 6 символов)";

exit();

}

Хешируем пароль, с помощью md5(), и добавляем свои какие-то рандомные символы, для усложнения пароля.

$pass = md5($pass."asdfafd2312");

Подключаемся к базе данных, первый параметр это хост, второй — имя пользователя, третье — пароль и четвертое это название БД.

$mysql = new mysqli('localhost', 'root', 'root', 'register-bd');

Проверяем логин на совпадение, если в базе данных система находит такой же логин — сообщаем об ошибке.

$mysql->query("INSERT INTO `users` (`login`, `pass`, `name`)

VALUES('$login', '$pass', '$name')");

$mysql->close();

header('Location: /');

### 2.4.3 Авторизация

В первой строке идет язык SQL, мы говорим базе данных отбирать из таблицы users, login, который совпадает с логином, который ввел пользователь, дальше создаем массив и если массив не пустой, значит система что-то нашла, из этого выходит что такой логин уже зарегистрирован — уведомляем пользователя об этом. Если все в порядке — регистрируется в Базе Данных пользователя и закрываем соединение:

$mysql->query("INSERT INTO `users` (`login`, `pass`, `name`)

VALUES('$login', '$pass', '$name')");

$mysql->close();

?>

В авторизации, получаем данные из полей.

<?php

$login = filter\_var(trim($\_POST['login']), FILTER\_SANITIZE\_STRING);

$pass = filter\_var(trim($\_POST['pass']), FILTER\_SANITIZE\_STRING);

Шифруем пароль, подключаемся к базе данных, создаем массив и проверяем в БД на совпадение пароля и логина — если совпадения находит, то одобряем пользователю вход, если нет — выводим сообщение об ошибке:

if(count($user) == 0){

echo "Такой пользователь не найден.";

exit();

}

else if(count($user) == 1){

echo "Логин или пароль введены неверно";

exit();

}

setcookie('user', $user['name'], time() + 3600, "/");

$mysql->close();

Означает что пользователя, если авторизация прошла успешно, будет перебрасывать на страницу page.html. Поэтому создаем страницу page.html и пишем просто ссылку:

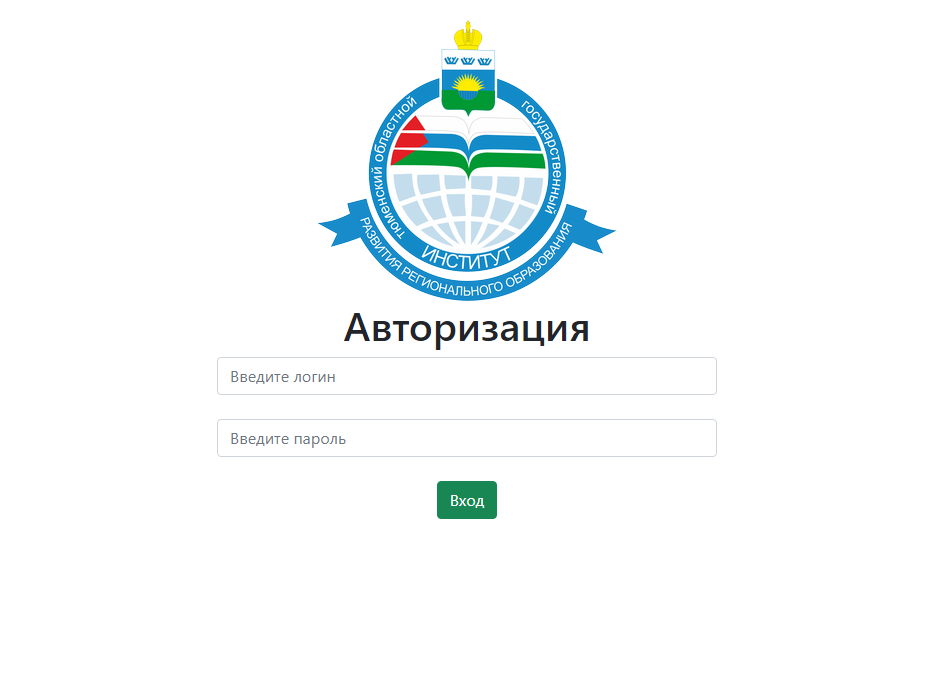
header('Location: page.html');

### 2.4.4 Выход

Для того чтобы пользователь вышел — нам всего лишь нужно очистить куки, создаем файл exit.php

setcookie('user', $user['name'], time() - 3600, "/");

header('Location: /');

  
Рис.35. Форма аутентификации.

## 

## 2.5. Конструктор построения отчетов

### 2.5.1. Алгоритмы обработки данных

Конструктор построения отчетов реализован как веб-форма, открывающаяся в модальном окне. На форме находятся различные переключатели, списки с множественным выбором, а также элементы направленные на определенные задачи (выбор разбалловки, выбор периода дат).

При разработке конструктора были описаны алгоритмы получения выбранных данных в форме, алгоритмы, определяющие взаимосвязь компонентов конструктора и др. Среди наиболее важных алгоритмов, выделяются алгоритм генерирующий критерии разбалловки для ЕГЭ, ГВЭ11 и алгоритм, отвечающий за построение графиков.

Работа алгоритма, генерирующего критерии оценивания ЕГЭ и ГВЭ11 начинается с нажатия кнопки “Добавить”. Начинается проверка что в поле input введено число меньше 100, далее начинается сборка вспомогательного элемента input, ему присваиваются соотвествующий класс и описание. Данный процесс запускается с каждым нажатием на кнопку “Добавить” и может выполняться 5 раз, так как большее количество критериев разбалловки не является нужным.

Алгоритм, отвечающий за построение графиков, разработан не до конца и в данный момент времени его функционал ограничивается построением различных “Шапок” для Excel отчета. Работа алгоритма заключается в проверке выбранных значений формы (для этого описаны вспомогательные функции) и далее в кропотливой сборке строк и столбцов таблицы, которая впоследствии отображается на HTML странице и может быть экспортирована как .xlsx документ.

### 2.5.2. Интерфейс конструктора

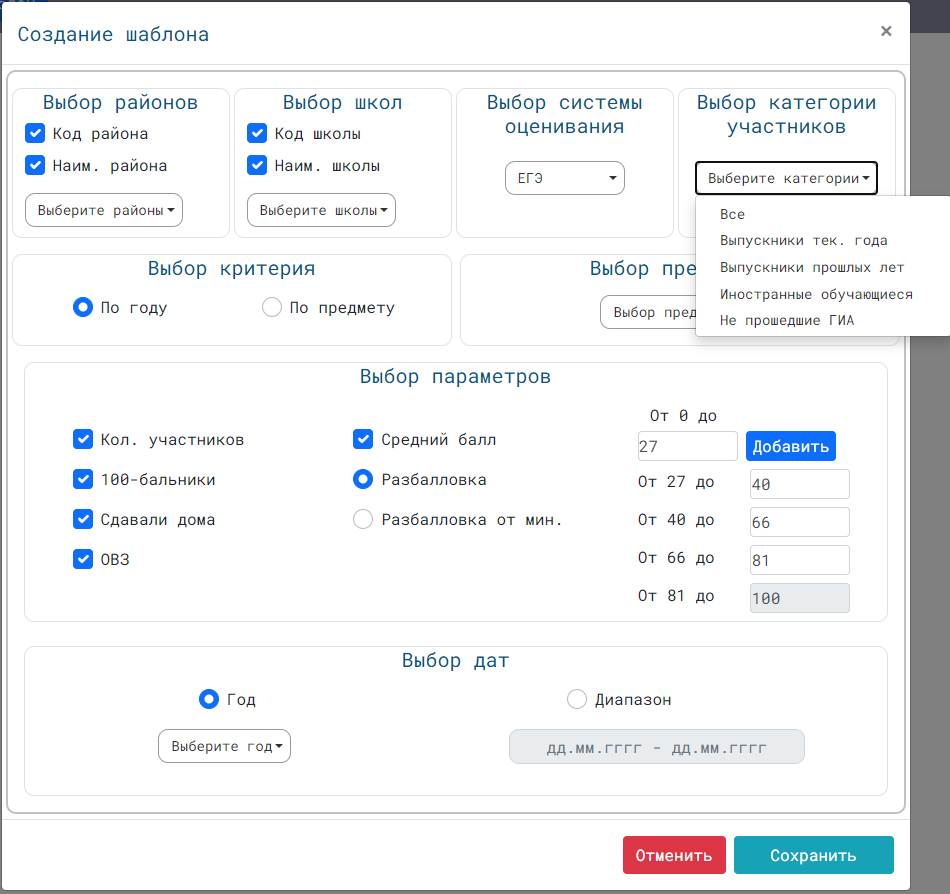
****

Рис. 36. Форма конструктора

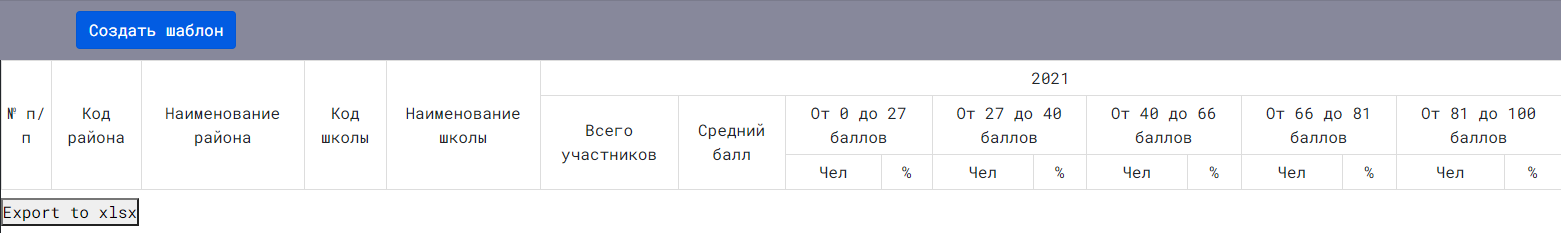
****

Рис. 37. Шапка отчета

# Заключение

В результате прохождения практики были выполнены следующие задачи:

* Ознакомились с реальными примерами отчетов, предоставляемых центром оценки качества образования.
* Реализован конструктор запросов к базе данных для выборки информации, с ее последующей визуализацией.
* Был реализован конструктор построения отчетов на основе выбранных данных.
* Реализована авторизация и регистрация пользователей через базу данных.
* Реализован пользовательский интерфейс с определенным уровнем доступа.
* Реализовано win32 приложение с выбором данных для выгрузки в файлы сайта из баз данных в закрытой сети.
* Реализована загрузку данных из файлов сайта в базу данных сайта.

В результате прохождения практики, командой были получены ценные практические знания и умения. Каждым членом команды были выполнены поставленные задачи. Были получены знания о ранее не изученных технологиях, а также закреплены существующие знания в области IT.

В перспективе дальнейшей работы над проектом, планируется:

1. Доработать интерфейс загрузчика данных в базу сайта
2. Доработать конструкторы отчетов для графической визуализации статистики с помощью диаграмм и графиков

# Список литературы

1. Веллинг, Томсон. Разработка веб-приложений с помощью PHP и MySQL (PHP 7 и MySQL 5.7!) 5-издание 2018.
2. Марк Лутц. Изучаем Python, том 1, 5-е изд.: Пер. с англ. — СПб.: ООО “Диалектика”, 2019. — 832 с.: ил. — Парал. тит. англ.
3. Документация по использованию веб-фреймворка Flask: [Электронный ресурс]. URL:<https://flask.palletsprojects.com/en/2.0.x/>. (Дата обращения: 26.05.2021)
4. Документация по использованию библиотеки SQLAlchemy: [Электронный ресурс]. URL:<https://docs.sqlalchemy.org/en/14/changelog/migration_14.html>. (Дата обращения: 26.05.2021).
5. Информация по языку программирования Python: [Электронный ресурс]. URL:<https://pythonworld.ru/samouchitel-python>. (Дата обращения: 20.05.2021).
6. Баррет Д. JavaScript. Web - профессионалам. Изд-во: Киев: БХВ - Киев, 2001.
7. Вадим Дунаев. HTML, скрипты и стили. Изд-во: БХВ - Петербург, 2008 г. 1024 с.
8. Гольцман В. MySQL 5.0. Библиотека программиста: Питер; Санкт-Петербург; 2010
9. Дунаев Вадим JavaScript. Самоучитель; Питер - М., 2015. - 400 c.
10. Дэвид Хейнмейер Ханссон. Гибкая разработка веб-приложений. - СПб.: Питер, 2014. – 448 c.
11. Инькова Н. А., Зайцева Е. А., Кузьмина Н. В., Толстых С. Г. Создание Web-сайтов: Учебно-методическое пособие. Ч. 5. Тамбов: Изд-во Тамб. гос. техн. ун-та, 2005. – 56 с.
12. Линн Бейли и Майкл Моррисон. Изучаем PHP и MySQL. 2010. 2-издание.

# Приложение

*Приложение 1. Алгоритм получения данных формы*

function getParameters() {

let flag = "";

let select = getValues("selectType");

if (select == 'ЕГЭ' || select == 'ГВЭ11') {

let checkPeopleE = $('.row-e input[id="epeople"]').prop('checked');

let check100E = $('.row-e input[id="hundred"]').prop('checked');

let checkHomeE = $('.row-e input[id="ephome"]').prop('checked');

let checkOVZE = $('.row-e input[id="eovz"]').prop('checked');

let checkScoreE = $('.row-e input[id="escore"]').prop('checked');

let radioBall = $('input[name="radioBall"]:checked').val();

if (checkPeopleE || check100E || checkHomeE

|| checkOVZE || checkScoreE) {

switch (parseInt(radioBall)) {

case 1:

flag = "e0";

var valuesArr = [];

$(".inp-range").each(function (index, el) {

var v = $(el).val();

if (v == 0) v = 100;

valuesArr.push(v);

});

return {

flag: flag, people: checkPeopleE, hundred: check100E,

home: checkHomeE, ovz: checkOVZE, score: checkScoreE,

rangeVal: valuesArr

};

case 2://Ращбаловка от мин.

flag = "emin";

var valuesMinArr = [];

$(".inp-range").each(function (index, el) {

var v = $(el).val();

if (v == 0) v = 100;

valuesMinArr.push(v);

});

let checkMin = $('.row-e input[id="ballMin"]').prop('checked');

return {

flag: flag, people: checkPeopleE, hundred: check100E,

home: checkHomeE, ovz: checkOVZE, score: checkScoreE,

rangeVal: valuesMinArr, checkMin: checkMin

};

default:

flag = "edef"

return {

flag: flag, people: checkPeopleE, hundred: check100E,

home: checkHomeE, ovz: checkOVZE, score: checkScoreE

};

}

}

return { flag: flag = "none" };

}

else if ((select != 'ЕГЭ' || select != 'ГВЭ11') && select != '') {

flag = "o";

let checkPeopleO = $('.row-o input[id="opeople"]').prop('checked');

let checkHomeO = $('.row-o input[id="ophome"]').prop('checked');

let checkOVZO = $('.row-o input[id="oovz"]').prop('checked');

let checkScoreO = $('.row-o input[id="oscore"]').prop('checked');

let checkBallO = $('.row-o input[id="oball"]').prop('checked');

if (checkPeopleO || checkHomeO ||

checkOVZO || checkScoreO) {

return {

flag: flag, people: checkPeopleO, home: checkHomeO,

ovz: checkOVZO, score: checkScoreO, ball: checkBallO

};

}

else {

return { flag: flag = "none" };

}

}

else return { flag: flag = "none" };

}

*Приложение 2. Алгоритм создания критериев разбалловки*

let count = 0;

$("#btn-range").on('click', function () {

if ($(".inp-range-first").val() != 100) {

let first = "<div class='row row-range ml-1 mt-1 mb-2'><label class='col-5 label-range' style='padding: 0!important;'>От ";

let middle = " до</label><input class='col-6 inp-range form-control ";

let valid = "validity.valid||(value='');"

let end = " ml-1' type='number' min=0 max=100 oninput=" + valid + " value=100></div>";

let number = 0;

let \_class = "";

switch (count) {

case 0:

number = $(".inp-range-first").val();

\_class = "inp-range-1";

$(".ball-range-area").append(first + Number(number) + middle + \_class + end);

count++;

break;

case 1:

if (parseInt($(".inp-range-1").val()) != 100 &&

parseInt($(".inp-range-1").val()) > parseInt($(".inp-range-first").val())) {

number = $(".inp-range-1").val();

\_class = "inp-range-2";

$(".ball-range-area").append(first + Number(number) + middle + \_class + end);

count++;

}

break;

case 2:

if (parseInt($(".inp-range-2").val()) != 100 &&

parseInt($(".inp-range-2").val()) > parseInt($(".inp-range-1").val())) {

number = $(".inp-range-2").val();

\_class = "inp-range-3";

$(".ball-range-area").append(first + Number(number) + middle + \_class + end);

count++;

}

break;

case 3:

if (parseInt($(".inp-range-3").val()) != 100 &&

parseInt($(".inp-range-3").val()) > parseInt($(".inp-range-2").val())) {

number = $(".inp-range-3").val();

\_class = "inp-range-4";

$(".ball-range-area").append(first + Number(number) + middle + \_class + end);

$(".inp-range-4").prop("disabled", true);

count++;

}

break;

default: return;

}

}

});

*Приложение 2. Алгоритм выгрузки экзаменов и ответов к ним*

void UnloadExamsAndAnswers()

{

using (var egeContext = new EgeContext2172())

using (var giaContext = new GiaContext2172())

{

string examsPath = Path.Combine(directoryPath, @"exams.json");

string answersPath = Path.Combine(directoryPath, @"answers.json");

if (!File.Exists(Path.Combine(directoryPath, "schools.json")) || !File.Exists(Path.Combine(directoryPath, "participantcategories.json"))) MessageBox.Show("Не удалось выгрузить результаты экзаменов, т.к. не были выгружены школы или категории участников");

else

{

#region BIGQUERY

List<string> selectedSubjects = new List<string>();

foreach (var item in lbSubjects.SelectedItems)

{

selectedSubjects.Add(item.ToString());

}

var egeprops = egeContext.RbdParticipantProperties.Where(p => p.Property == 6);

var query = (from tests in egeContext.ResHumanTests

join marks in egeContext.ResMarks

on tests.HumanTestId equals marks.HumanTestId

join participants in egeContext.RbdParticipants

on tests.ParticipantFk equals participants.ParticipantId

join subjects in egeContext.DatSubjects

on tests.SubjectCode equals subjects.SubjectCode

join categories in egeContext.RbdcParticipantCategories

on participants.ParticipantCategoryFk equals categories.CategoryId

join classes in egeContext.RbdSchoolClasses

on participants.SchoolClassId equals classes.SchoolClassId into gj1

from classes in gj1.DefaultIfEmpty()

join schools in egeContext.RbdSchools

on classes.SchoolId equals schools.SchoolId into gj2

from schools in gj2.DefaultIfEmpty()

join props in egeprops

on participants.ParticipantId equals props.ParticipantId into gj3

from props in gj3.DefaultIfEmpty()

select new

{

HumanTestID = tests.HumanTestId,

ParticipantID = tests.ParticipantFk,

TestTypeCode = tests.TestTypeCode,

SubjectCode = tests.SubjectCode,

ExamDate = tests.ExamDate,

StationCode = tests.StationCode,

VariantCode = tests.VariantCode,

ProcessCondition = tests.ProcessCondition,

PrimaryMark = marks.PrimaryMark,

PercentMark = marks.PercentMark,

Mark100 = marks.Mark100,

Mark5 = marks.Mark5,

PrimaryMarkB = marks.PrimaryMarkB,

TestResultB = marks.TestResultB,

PrimaryMarkC = marks.PrimaryMarkC,

TestResultC = marks.TestResultC,

PrimaryMarkD = marks.PrimaryMarkD,

TestResultD = marks.TestResultD,

ParticipantCode = participants.ParticipantCode,

ParticipantCategoryID = participants.ParticipantCategoryFk,

SchoolCode = schools == null ? -1 : schools.SchoolCode,

SchoolDeleteType = schools == null ? -1 : schools.DeleteType,

CategoryCode = categories.CategoryCode,

CategoryName = categories.CategoryName,

Property = props == null ? -1 : props.Property,

SubjectName = subjects.SubjectName,

}).Where(q => q.ProcessCondition == 6 && (q.TestTypeCode == 4 || q.TestTypeCode == 5) &&

(q.SchoolDeleteType == 0 || q.SchoolDeleteType == -1) && selectedSubjects.Any(s => s == q.SubjectName)).ToList();

if (lbDatabases.SelectedItems.Count == 2)

{

var giaprops = giaContext.RbdParticipantProperties.Where(p => p.Property == 6);

query = query.Union((from tests in giaContext.ResHumanTests

join marks in giaContext.ResMarks

on tests.HumanTestId equals marks.HumanTestId

join participants in giaContext.RbdParticipants

on tests.ParticipantFk equals participants.ParticipantId

join subjects in giaContext.DatSubjects

on tests.SubjectCode equals subjects.SubjectCode

join categories in giaContext.RbdcParticipantCategories

on participants.ParticipantCategoryFk equals categories.CategoryId

join classes in giaContext.RbdSchoolClasses

on participants.SchoolClassId equals classes.SchoolClassId into gj1

from classes in gj1.DefaultIfEmpty()

join schools in giaContext.RbdSchools

on classes.SchoolId equals schools.SchoolId into gj2

from schools in gj2.DefaultIfEmpty()

join props in giaprops

on participants.ParticipantId equals props.ParticipantId into gj3

from props in gj3.DefaultIfEmpty()

select new

{

HumanTestID = tests.HumanTestId,

ParticipantID = tests.ParticipantFk,

TestTypeCode = tests.TestTypeCode,

SubjectCode = tests.SubjectCode,

ExamDate = tests.ExamDate,

StationCode = tests.StationCode,

VariantCode = tests.VariantCode,

ProcessCondition = tests.ProcessCondition,

PrimaryMark = marks.PrimaryMark,

PercentMark = marks.PercentMark,

Mark100 = marks.Mark100,

Mark5 = marks.Mark5,

PrimaryMarkB = marks.PrimaryMarkB,

TestResultB = marks.TestResultB,

PrimaryMarkC = marks.PrimaryMarkC,

TestResultC = marks.TestResultC,

PrimaryMarkD = marks.PrimaryMarkD,

TestResultD = marks.TestResultD,

ParticipantCode = participants.ParticipantCode,

ParticipantCategoryID = participants.ParticipantCategoryFk,

SchoolCode = schools == null ? -1 : schools.SchoolCode,

SchoolDeleteType = schools == null ? -1 : schools.DeleteType,

CategoryCode = categories.CategoryCode,

CategoryName = categories.CategoryName,

Property = props == null ? -1 : props.Property,

SubjectName = subjects.SubjectName

}).Where(q => q.ProcessCondition == 6 && (q.TestTypeCode == 6 || q.TestTypeCode == 7) &&

(q.SchoolDeleteType == 0 || q.SchoolDeleteType == -1) && selectedSubjects.Any(s => s == q.SubjectName))).ToList();

}

else if (lbDatabases.SelectedItems.Count == 1)

{

if (lbDatabases.SelectedItems[0].ToString() == "erbd\_gia\_reg\_21\_72")

{

var giaprops = giaContext.RbdParticipantProperties.Where(p => p.Property == 6);

query = (from tests in giaContext.ResHumanTests

join marks in giaContext.ResMarks

on tests.HumanTestId equals marks.HumanTestId

join participants in giaContext.RbdParticipants

on tests.ParticipantFk equals participants.ParticipantId

join subjects in giaContext.DatSubjects

on tests.SubjectCode equals subjects.SubjectCode

join categories in giaContext.RbdcParticipantCategories

on participants.ParticipantCategoryFk equals categories.CategoryId

join classes in giaContext.RbdSchoolClasses

on participants.SchoolClassId equals classes.SchoolClassId into gj1

from classes in gj1.DefaultIfEmpty()

join schools in giaContext.RbdSchools

on classes.SchoolId equals schools.SchoolId into gj2

from schools in gj2.DefaultIfEmpty()

join props in giaprops

on participants.ParticipantId equals props.ParticipantId into gj3

from props in gj3.DefaultIfEmpty()

select new

{

HumanTestID = tests.HumanTestId,

ParticipantID = tests.ParticipantFk,

TestTypeCode = tests.TestTypeCode,

SubjectCode = tests.SubjectCode,

ExamDate = tests.ExamDate,

StationCode = tests.StationCode,

VariantCode = tests.VariantCode,

ProcessCondition = tests.ProcessCondition,

PrimaryMark = marks.PrimaryMark,

PercentMark = marks.PercentMark,

Mark100 = marks.Mark100,

Mark5 = marks.Mark5,

PrimaryMarkB = marks.PrimaryMarkB,

TestResultB = marks.TestResultB,

PrimaryMarkC = marks.PrimaryMarkC,

TestResultC = marks.TestResultC,

PrimaryMarkD = marks.PrimaryMarkD,

TestResultD = marks.TestResultD,

ParticipantCode = participants.ParticipantCode,

ParticipantCategoryID = participants.ParticipantCategoryFk,

SchoolCode = schools == null ? -1 : schools.SchoolCode,

SchoolDeleteType = schools == null ? -1 : schools.DeleteType,

CategoryCode = categories.CategoryCode,

CategoryName = categories.CategoryName,

Property = props == null ? -1 : props.Property,

SubjectName = subjects.SubjectName

}).Where(q => q.ProcessCondition == 6 && (q.TestTypeCode == 6 || q.TestTypeCode == 7) &&

(q.SchoolDeleteType == 0 || q.SchoolDeleteType == -1) && selectedSubjects.Any(s => s == q.SubjectName)).ToList();

}

}

else

{

throw new Exception("База данных не выбрана");

}

#endregion

#region READING\_SCHOOLS

//Reading schools from json file

var schoolsList = new List<(int SchoolID, int SchoolCode)>();

string jsonSchools = File.ReadAllText(Path.Combine(directoryPath, "schools.json"));

JsonTextReader schoolReader = new JsonTextReader(new StringReader(jsonSchools));

while (schoolReader.Read())

{

if (schoolReader.Value != null)

{

(int SchoolID, int SchoolCode) school = new();

if (schoolReader.Value.ToString() == "SchoolID")

{

schoolReader.Read();

school.SchoolID = int.Parse(schoolReader.Value.ToString());

schoolReader.Read();

}

if (schoolReader.Value.ToString() == "SchoolCode")

{

schoolReader.Read();

school.SchoolCode = int.Parse(schoolReader.Value.ToString());

schoolsList.Add(school);

}

}

}

#endregion

#region READING\_CATEGORIES

var categoriesList = new List<(int CategoryID, string CategoryName)>();

string jsonCategories = File.ReadAllText(Path.Combine(directoryPath, "participantcategories.json"));

JsonTextReader categoryReader = new JsonTextReader(new StringReader(jsonCategories));

while (categoryReader.Read())

{

if (categoryReader.Value != null)

{

(int CategoryID, string CategoryName) category = new();

if (categoryReader.Value.ToString() == "CategoryID")

{

categoryReader.Read();

category.CategoryID = int.Parse(categoryReader.Value.ToString());

categoryReader.Read();

categoryReader.Read();

categoryReader.Read();

}

if (categoryReader.Value.ToString() == "CategoryName")

{

categoryReader.Read();

category.CategoryName = categoryReader.Value.ToString();

categoriesList.Add(category);

}

}

}

#endregion

#region WRITING\_EXAMS\_AND\_ANSWERS

StringBuilder examsJsonString = new StringBuilder();

StringBuilder answersJsonString = new StringBuilder();

StringWriter examsStringWriter = new StringWriter(examsJsonString);

StringWriter answersStringWriter = new StringWriter(answersJsonString);

int examId = 0;

int answerId = 0;

using (JsonWriter examsWriter = new JsonTextWriter(examsStringWriter))

using (JsonWriter answersWriter = new JsonTextWriter(answersStringWriter))

{

examsWriter.WriteStartArray();

answersWriter.WriteStartArray();

foreach (var exam in query)

{

#region EXAM

examId++;

examsWriter.WriteStartObject();

examsWriter.WritePropertyName("ExamID");

examsWriter.WriteValue(examId);

examsWriter.WritePropertyName("SchoolID");

if (exam.SchoolCode == -1) examsWriter.WriteValue("NULL");

else

{

int schoolId = GetSchoolId(exam.SchoolCode, schoolsList);

if (schoolId != -1) examsWriter.WriteValue(schoolId);

else throw new KeyNotFoundException();

}

examsWriter.WritePropertyName("SchemaID");

examsWriter.WriteValue("NULL");

examsWriter.WritePropertyName("VariantCode");

examsWriter.WriteValue(exam.VariantCode);

examsWriter.WritePropertyName("FirstPoints");

examsWriter.WriteValue(exam.PrimaryMark);

examsWriter.WritePropertyName("FinalPoints");

if (exam.TestTypeCode == 4) examsWriter.WriteValue(exam.Mark100);

else examsWriter.WriteValue(exam.Mark5);

examsWriter.WritePropertyName("CompletionPercent");

examsWriter.WriteValue(exam.PercentMark);

examsWriter.WritePropertyName("MarkSystem");

if (exam.TestTypeCode == 4) examsWriter.WriteValue(100);

else examsWriter.WriteValue(5);

examsWriter.WritePropertyName("ExamDate");

examsWriter.WriteValue(exam.ExamDate);

examsWriter.WritePropertyName("IsOVZ");

if (exam.Property == 6) examsWriter.WriteValue(1);

else examsWriter.WriteValue(0);

examsWriter.WritePropertyName("IsAtHome");

if (exam.StationCode > 899) examsWriter.WriteValue(1);

else examsWriter.WriteValue(0);

examsWriter.WritePropertyName("ParticipantCategoryID");

int categoryId = GetParticipantCategoryId(exam.CategoryName, categoriesList);

if (categoryId != -1) examsWriter.WriteValue(categoryId);

else throw new KeyNotFoundException();

examsWriter.WritePropertyName("SchoolClassNumber");

if (exam.TestTypeCode == 4 || exam.TestTypeCode == 5) examsWriter.WriteValue(9);

else examsWriter.WriteValue(11);

examsWriter.WriteEndObject();

#endregion

#region ANSWERS

int taskNum = 0;

//1) Задания с кратким ответом

if (exam.TestResultB != string.Empty)

{

foreach (char answer in exam.TestResultB)

{

answersWriter.WriteStartObject();

answersWriter.WritePropertyName("AnswerID");

answerId++;

answersWriter.WriteValue(answerId);

answersWriter.WritePropertyName("PartID");

answersWriter.WriteValue(1);

answersWriter.WritePropertyName("TaskNumber");

taskNum++;

answersWriter.WriteValue(taskNum);

answersWriter.WritePropertyName("CurrentPoint");

switch (answer)

{

case '-': answersWriter.WriteValue(0); break;

case '+': answersWriter.WriteValue(1); break;

default: answersWriter.WriteValue(answer); break;

}

answersWriter.WritePropertyName("ExamID");

answersWriter.WriteValue(examId);

answersWriter.WriteEndObject();

}

}

//2) Задания с развернутым ответом

if (exam.TestResultC != string.Empty)

{

Regex regex = new Regex(@"\(\d\)");

string testResultC = regex.Replace(exam.TestResultC, "");

foreach (char answer in testResultC)

{

answersWriter.WriteStartObject();

answersWriter.WritePropertyName("AnswerID");

answerId++;

answersWriter.WriteValue(answerId);

answersWriter.WritePropertyName("PartID");

answersWriter.WriteValue(1);

answersWriter.WritePropertyName("TaskNumber");

taskNum++;

answersWriter.WriteValue(taskNum);

answersWriter.WritePropertyName("CurrentPoint");

answersWriter.WriteValue(answer);

answersWriter.WritePropertyName("ExamID");

answersWriter.WriteValue(examId);

answersWriter.WriteEndObject();

}

}

//3) Задания устной части

if (exam.TestResultD != string.Empty)

{

Regex regex = new Regex(@"\(\d\)");

string testResultD = regex.Replace(exam.TestResultD, "");

foreach (char answer in testResultD)

{

answersWriter.WriteStartObject();

answersWriter.WritePropertyName("AnswerID");

answerId++;

answersWriter.WriteValue(answerId);

answersWriter.WritePropertyName("PartID");

answersWriter.WriteValue(1);

answersWriter.WritePropertyName("TaskNumber");

taskNum++;

answersWriter.WriteValue(taskNum);

answersWriter.WritePropertyName("CurrentPoint");

answersWriter.WriteValue(answer);

answersWriter.WritePropertyName("ExamID");

answersWriter.WriteValue(examId);

answersWriter.WriteEndObject();

}

}

#endregion

}

examsWriter.WriteEnd();

answersWriter.WriteEnd();

}

examsStringWriter.Close();

answersStringWriter.Close();

if (!File.Exists(examsPath)) File.Create(examsPath).Close();

File.WriteAllText(examsPath, examsJsonString.ToString());

if (!File.Exists(answersPath)) File.Create(answersPath).Close();

File.WriteAllText(answersPath, answersJsonString.ToString());

#endregion

MessageBox.Show($"Экзамены выгружены ({examId} экзаменов / {answerId} ответов)");

}

}

}

*Приложение 3 Алгоритм регистрации и авторизации через базу данных.*

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="x-ua-compatible" content="ie=edge">

<title>Авторизация</title>

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">

<link rel="stylesheet" href="css/style.css">

</head>

<body>

<div class="container mt-4">

<?php

if($\_COOKIE['user'] == ''):

?>

<center>

<a href="https://togirro.ru/"><img src="png/logo.png"></a>

<h1>Авторизация</h1>

<form action="validation-form/auth.php" method="post">

<input type="text" class="form-control" name="login" id="login" placeholder="Введите логин"><br>

<input type="text" class="form-control" name="pass" id="pass" placeholder="Введите пароль"><br>

<button class="btn btn-success" type="submit" >Вход</button>

</form>

</center>

</div>

<?php else: ?>

<p>Привет <?=$\_COOKIE['user']?>. Чтобы выйти нажмите <a href="/exit.php">здесь</a>.</p>

<?php endif;?>

</body>

</html>

<?php

$login = filter\_var(trim($\_POST['login']),

FILTER\_SANITIZE\_STRING);

$pass = filter\_var(trim($\_POST['pass']),

FILTER\_SANITIZE\_STRING);

$pass = md5($pass."asdfafd2312");

$mysql = new mysqli('localhost', 'root', 'root', 'register-bd');

$result = $mysql->query("SELECT \* FROM `users` WHERE `login` =

'$login' AND `pass` = '$pass'");

$user = $result->fetch\_assoc();

if(count($user) == 0) {

echo "Такой пользователь не найден";

exit();

}

setcookie('user', $user['name'], time() + 3600, "/");

header('Location: /');

?>

<?php

$login = filter\_var(trim($\_POST['login']),

FILTER\_SANITIZE\_STRING);

$name = filter\_var(trim($\_POST['name']),

FILTER\_SANITIZE\_STRING);

$pass = filter\_var(trim($\_POST['pass']),

FILTER\_SANITIZE\_STRING);

if(mb\_strlen($login) < 2 || mb\_strlen($login) > 90) {

echo "Недопустимая длина логина";

exit();

} else if(mb\_strlen($name) < 3 || mb\_strlen($name) > 50) {

echo "Недопустимая длина имени";

exit();

} else if(mb\_strlen($pass) < 2 || mb\_strlen($pass) > 6) {

echo "Недопустимая длина пароля (от 2 до 6 символов)";

exit();

}

$pass = md5($pass."asdfafd2312");

$mysql = new mysqli('localhost', 'root', 'root', 'register-bd');

$mysql->query("INSERT INTO `users` (`login`, `pass`, `name`)

VALUES('$login', '$pass', '$name')");

$mysql->close();

header('Location: /');

?>