ГОУ ВПО «Донецкий национальный университет»

Физико-технический факультет

Кафедра компьютерных технологий

**Отчет**

об учебной (НИР) практике

с «06» июня 2022 по «18» июня 2022

ВЫПОЛНИЛ:

студент II курса (I уск.) группы \_\_\_

заочной формы обучения

направление подготовки 09.03.01

«Информатика и вычислительная техника»

Горбань Антон Витальевич

(Ф. И. О. студента)

ПРОВЕРИЛА:

Ст. преподаватель\_\_\_\_\_\_\_\_ Бодряга В. Е.

Донецк 2021

содержание

[ВАРИАНТ ЗАДАНИЯ 3](#__RefHeading___Toc2419_1697418480)

[1. РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ В СУБД 5](#__RefHeading___Toc2421_1697418480)

[1.1 Модель базы данных (логическая, физическая) 5](#__RefHeading___Toc2423_1697418480)

[1.2 Реализация базы данных в СУБД MS Access 7](#__RefHeading___Toc2425_1697418480)

[1.3 Выполнение заданий по варианту, описание контрольных примеров 12](#__RefHeading___Toc2427_1697418480)

[2. РЕАЛИЗАЦИЯ ЗАДАНИЯ 16](#__RefHeading___Toc2429_1697418480)

[2.1 Выбор средств реализации 16](#__RefHeading___Toc2431_1697418480)

[2.2 Выполнение заданий по варианту, описание контрольных примеров 18](#__RefHeading___Toc2433_1697418480)

[ЗАКЛЮЧЕНИЕ 22](#__RefHeading___Toc2435_1697418480)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 23](#__RefHeading___Toc2437_1697418480)

# ВАРИАНТ ЗАДАНИЯ

Часть 1. Выполняют вариант задания с использованием базы данных и необходимых запросов, созданных в оболочке Microsoft Access.

Часть 2. При выполнении задания студенту необходимо составить на объектно-ориентированном языке оконное приложение, которое работает с файлами формата **CSV,** и отладить программу вычислений производственно-экономического характера.

При составлении программы необходимо разработать меню, содержащее для всех вариантов следующие режимы работы (все режимы оформить в виде отдельных методов класса):

1. Создание файла, содержащего сведения документа.
2. Печать исходного файла.
3. Добавление записей в файл.
4. Корректировка записей файла.
5. Удаление нескольких записей файла.
6. Упорядочение записей в файле по полю, указанному в задании.
7. Расчет итоговых показателей и необходимых полей записей.
8. Печать результирующего файла.
9. На оценку «отлично» предусмотреть вывод графической информации сводного характера.

В результате распределения был присвоен **ВАРИАНТ №2**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Регион** | **Страна** | **Сталь** | **Уголь** | **Нефть** |
| Северная Америка | США | 78,6 | 660,6 | 11,8 |
| Азия | Индия | 95,6 | 692,4 | 4,0 |
| Европа, Азия | Россия | 70,8 | 227,2 | 13,9 |
| Европа | Франция | 14,1 | 5,3 | 0,4 |
| Азия | Южная Корея | 68,6 | 1,15 | 1,2 |
| Азия | Турция | 33,2 | 70,6 | 2,5 |
| Азия | Япония | 104,8 | 5,8 | 1,22 |

Задание 1. Найти страну с наибольшим производством стали.

Задание 2. Найти страны, добыча нефти в которых превысила 1 млн. т.

Задание 3. Выдать информацию о странах, добыча угля в которых не

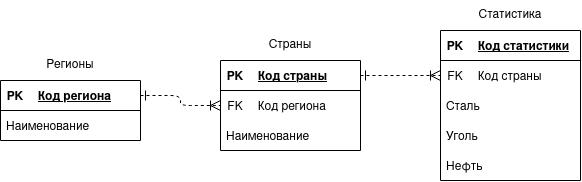
превысила 200 млн. т.

# РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ В СУБД

## Модель базы данных (логическая, физическая)

Логическая (даталогическая) модель представляет собой модель базы данных, которая не привязана к конкретной СУБД. В ней выделяют основные объекты БД и определяют связи между этими объектами. Иногда определятся типы данных отдельных объектов. Данная модель построена методом Сущность-связь (Entity Relationship).

В ходе анализа предметной области была разработана логическая модель базы данных, представленная на рисунке 1.1

  
Рисунок 1.1 – Логическая модель базы данных

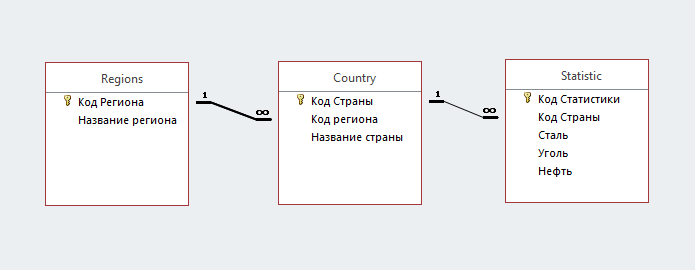
На основе данной логической модели была разработана физическая модель базы данных.

Физическая модель БД определяет способ размещения данных в среде хранения и способы доступа к этим данным, которые поддерживаются на физическом уровне. Исторически первыми системами хранения и доступа были файловые структуры и системы управления файлами (СУФ), которые фактически являлись частью операционных систем. СУБД создавала над этими файловыми моделями свою надстройку, которая позволяла организовать всю совокупность файлов таким образом, чтобы она выглядела как единое целое и получала централизованное управление от СУБД. Однако непосредственный доступ осуществлялся на уровне файловых команд, которые СУБД использовала при манипуляции всеми файлами, составляющие хранимые данные одной или нескольких баз данных.

Однако механизмы буферизации и управления файловыми структурами не приспособлены для решения задач собственно СУБД, эти механизмы разрабатывались просто для традиционной обработки файлов, и с ростом объемов хранимых данных они стали неэффективными для использования СУБД. Тогда постепенно произошёл переход от базовых файловых структур к непосредственному управлению размещением данных на внешних носителях самой СУБД.

В процессе проектирования при переходе от логической к физической модели наблюдается соответствие между основными категориями.

В ходе разработки базы данных в MS Access была спроектирована физическая модель, представленная на рисунке 1.2.

Рисунок 1.2 – Физическая модель базы данных

## Реализация базы данных в СУБД MS Access

Предметной областью этого проекта является «Анализ производства стали, угля и нефти в различных странах различных регионов в мире за 2016 год». Основной задачей продукта является упорядочивание, изменение, добавление и предоставление наглядных данных о производстве стали, угля и нефти в различных странах различных регионов за 2016 г.. Разработанная база данных ориентирована на самые различные сферы деятельности, она может быть использована как предпринимателями и учеными в сферах добычи, разработки и продажи стали, угля и нефти, так и частными и государственными компаниями, занимающимися сбором и обработкой статистики в представленных сферах.

В процессе анализа предметной области и построения логической модели базы данных было выделено три таблицы:

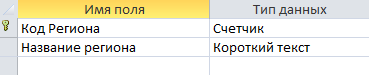
1. Regions;
2. Country;
3. Statistic.

Связи между таблицами создаются в полном соответствии с логической моделью базы данных:

1. Regions -> Country;
2. Country -> Statistic;

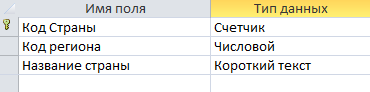
Выделим атрибуты таблицы «Regions» (рисунок 1.3):

1. Код Региона (Счетчик) – уникальный идентификатор для записи в таблицу, первичный ключ;
2. Название региона (Короткий текст) – содержит наименование региона.

Рисунок 1.3 – таблица «Regions» в режиме конструктора

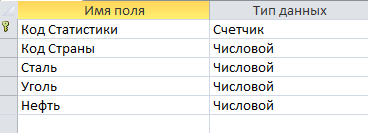
Выделим атрибуты таблицы «Country» (рисунок 1.4):

1. Код Страны (Счетчик) - уникальный идентификатор для записи в таблицу, первичный ключ;
2. Название страны (Короткий текст) – Наименование страны;
3. Код региона (Числовой) - идентификатор для связи с таблицей «Regions», внешний ключ;

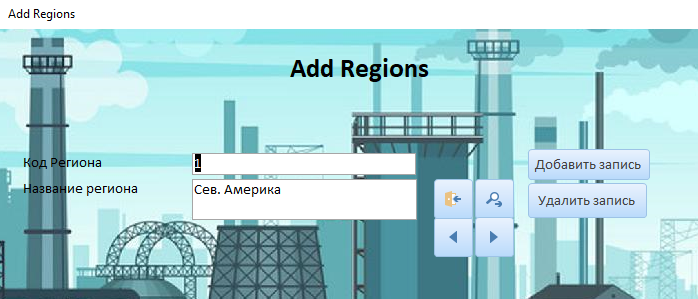
Рисунок 1.4 – таблица «Conutry» в режиме конструктора

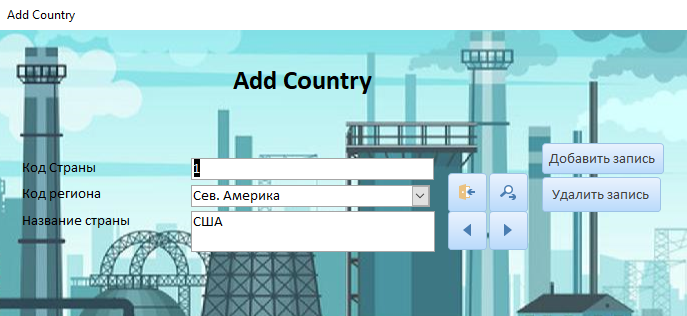
Выделим атрибуты таблицы «Statistic» (рисунок 1.5):

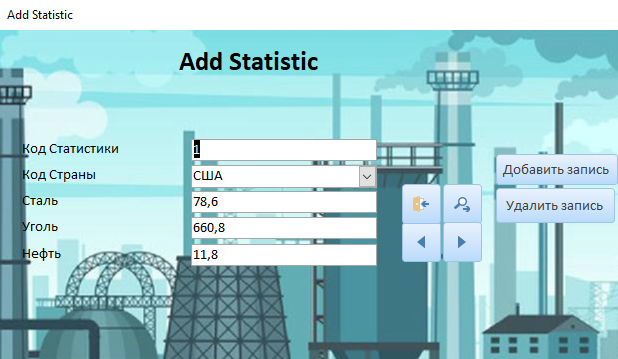
1. Код Статистики (Счетчик) - уникальный идентификатор для записи в таблицу, первичный ключ;
2. Код Страны (Числовой) - идентификатор для связи с таблицей «Country», внешний ключ;
3. Сталь (Числовой) — производство стали в млн. т. в 2016 г.
4. Уголь (Числовой) — производство угля в млн. т. в 2016 г.
5. Нефть (Числовой) — производство нефти в млн. т. в 2016 г.

Рисунок 1.5 – таблица «Statistic» в режиме конструктора

Для предоставления пользователю возможности взаимодействия с функционалом базы данных были разработаны формы, соответствующие таблицам «Regions», «Country» и «Statistic». Данные формы представлены на рисунках 1.6 – 1.8:

Рисунок 1.6 – форма «Regions»

Рисунок 1.7 – форма «Country»

Рисунок 1.8 – форма «Statistic»

Одним из требований к проекту является разработка некоторых запросов, соответствующих предметной области. Листинг SQL запросов представлен ниже:

Листинг запроса, выполняющего задание 1: Найти страну с наибольшим производством стали.

select

Statistic.[Код Статистики],

Regions.[Название региона],

Country.[Название страны],

Statistic.Сталь

from

Regions

inner join (Country

inner join Statistic on

Country.[Код Страны] = Statistic.[Код Страны]) on

Regions.[Код Региона] = Country.[Код региона]

where

(((Statistic.Сталь)=(

select

Max(Statistic.Сталь)

from

Statistic;

)));

Листинг запроса, выполняющего задание 2: Найти страны, добыча нефти в которых превысила 1 млн. т.

select

Statistic.[Код Статистики],

Regions.[Название региона],

Country.[Название страны],

Statistic.Нефть

from

Regions

inner join (Country

inner join Statistic on

Country.[Код Страны] = Statistic.[Код Страны]) on

Regions.[Код Региона] = Country.[Код региона]

where

(((Statistic.Нефть)>1));

Листинг запроса, выполняющего задание 3: Выдать информацию о странах, добыча угля в которых не превысила 200 млн. т.

select

Statistic.[Код Статистики],

Regions.[Название региона],

Country.[Название страны],

Statistic.Уголь

from

Regions

inner join (Country

inner join Statistic on

Country.[Код Страны] = Statistic.[Код Страны]) on

Regions.[Код Региона] = Country.[Код региона]

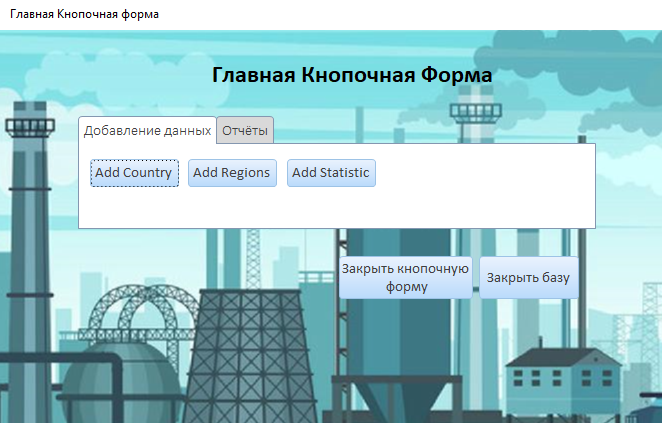
where

(((Statistic.Уголь)<200));

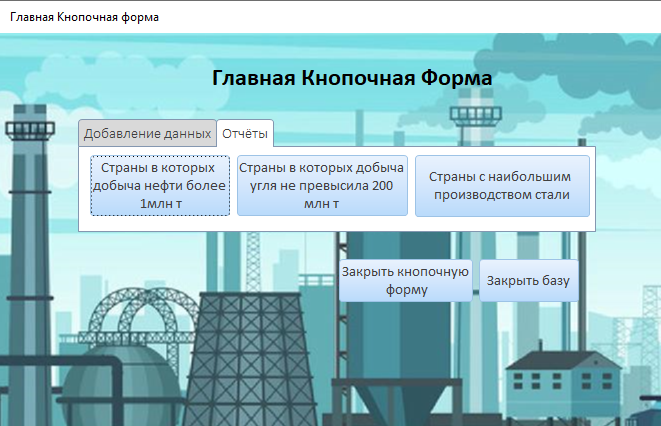
## Выполнение заданий по варианту, описание контрольных примеров

Для обеспечения безопасности базы данных пользователю не должно быть предоставлено доступа к внутренним объектам Access, таким как таблицы и запросы. Для обеспечения пользователю взаимодействия с этими данными были разработаны формы и отчеты.

При открытии базы данных пользователя встречает «Главная кнопочная форма», представленная на рисунке 1.9, осуществляющая функцию главного меню приложения.

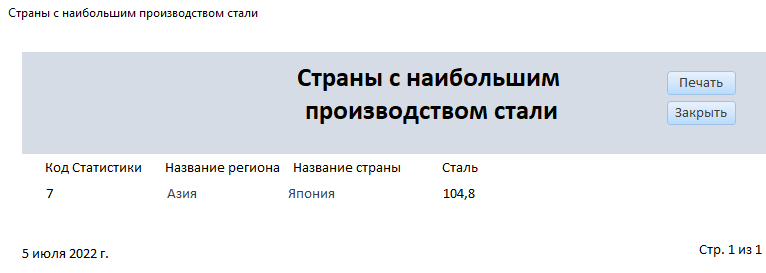
Рисунок 1.9 - «Главная кнопочная форма» на вкладке «Добавление данных»

При помощи «Главной кнопочной формы» есть возможность переходить между формами и отчетами, исполняющих роль наглядно отформатированных результатов выполнения запросов. Для перехода между формами и отчетами требуется переключиться между вкладками «Добавление данных» и «Отчеты» (рисунок 1.10).

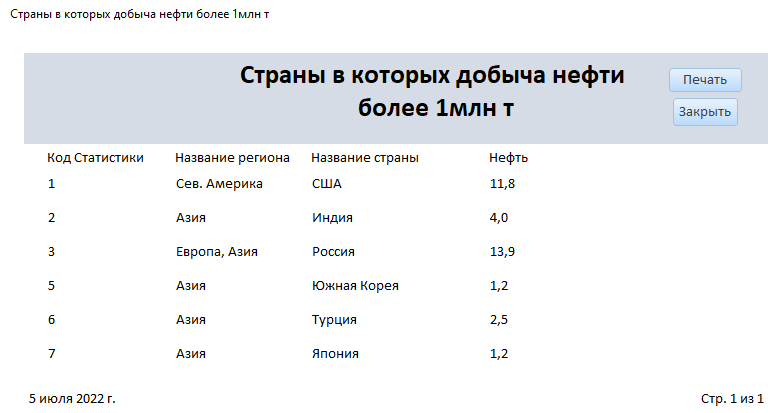
Рисунок 1.10 - «Главная кнопочная форма» на вкладке «Отчеты»

Просмотр результата выполнения запроса осуществляется через три основные кнопки с описанием запроса. При нажатии на определенную кнопку открывается отчет, соответствующий её запросу.

Отчёт запроса, выполняющего задание 1: «Найти страну с наибольшим производством стали», представлен на рисунке 1.11:

Рисунок 1.11 - Отчёт запроса, выполняющего задание 1

Отчет запроса, выполняющего задание 2: «Найти страны, добыча нефти в которых превысила 1 млн. т.», представлен на рисунке 1.12:

Рисунок 1.12 - Отчёт запроса, выполняющего задание 2

Отчет запроса, выполняющего задание 3: «Выдать информацию о странах, добыча угля в которых не превысила 200 млн. т.», представлен на рисунке 1.13:

Рисунок 1.13 - Отчёт запроса, выполняющего задание 3

# РЕАЛИЗАЦИЯ ЗАДАНИЯ

## Выбор средств реализации

При выполнении практического задания будет разработано оконное приложение. В процессе анализа различных платформ выполнения и языков программирования была выбрана платформа Node.js и язык программирования JavaScript, а именно его модификация от Microsoft TypeScript.

Ключевыми преимуществами в пользу выбора платформы:

1. Кроссплатформенность - способность программного обеспечения работать с несколькими аппаратными платформами или операционными системами;
2. Большое количество бесплатных инструментов, фреймворков и библиотек, а также удобное управление ими;
3. Оптимальная скорость работы программ при низком потреблении ресурсов вашего компьютера.

Ключевыми преимуществами в пользу выбора языка программирования:

1. Строгая типизация – обеспечивает большую защищенность кода;
2. Удобоваримость - благодаря типам код становится понятнее. Его легче читать, а значит и подключение новых разработчиков происходит быстрее;
3. Стандартизация рабочего процесса - подобно подходу TDD, при котором разработчик сначала создает тесты для проверки системы и только после этого реализует функционал, вам необходимо думать об интерфейсах кода, прежде чем переходить к его внутренней реализации;
4. Помощь в предотвращении багов - если код компилируется, высок процент вероятности, что он работает;
5. Упрощенный рефакторинг – разработчик может без опаски вносить изменения в кодовую базу — среда разработки поможет найти все варианты использования реорганизованных частей кода, укажет на измененные классы, функции и объекты, предупредит об ошибке компиляции, в случае несоответствия типов после рефакторинга;
6. Мультипарадигменный язык программирования – поддерживает объектно-ориентированную и функциональную парадигмы.

Приложение должно предоставлять данные и возможность взаимодействия с ними при помощи наглядного GUI (Graphical User Interface). Пользовательский интерфейс в приложении будет разработан с помощью объектно-ориентированного фреймворка Angular.

Angular — открытая и свободная платформа для разработки веб-приложений, написанная на языке TypeScript, разрабатываемая командой из компании Google, а также сообществом разработчиков из различных компаний.

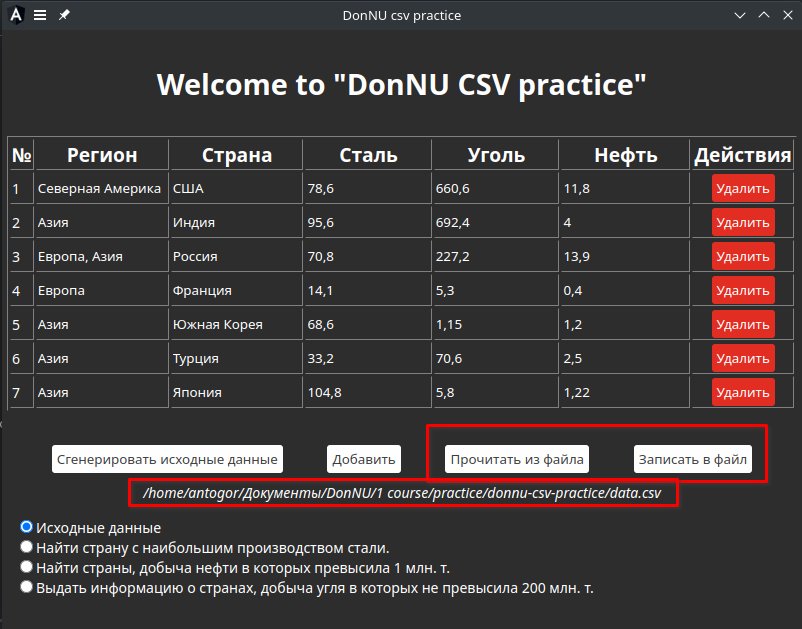
Разрабатываемое приложение должно быть настольным - это такая программа, которая обрабатывается на стороне клиента и запускается в виде обыкновенного исполняемого файла на устройстве пользователя. В качестве такого устройства может быть компьютер, коммуникатор или смартфон. Этого мы достигнем при помощи фреймворка Electron, разработанного GitHub, который позволяет разрабатывать нативные графические приложения для операционных систем с помощью веб-технологий. Фреймворк включает в себя Node.js для работы с back-end и библиотеку веб-рендеринга из Chromium.

При проведении операций ввода/вывода над файловой системой, валидации и в случае возникновения ошибок приложение должно выводить определенные диалоговые окна, которое обеспечивается вызовом нативных методов Electron.

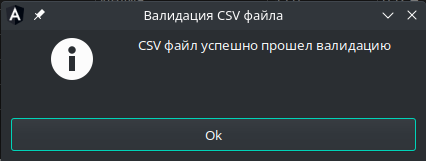
## Выполнение заданий по варианту, описание контрольных примеров

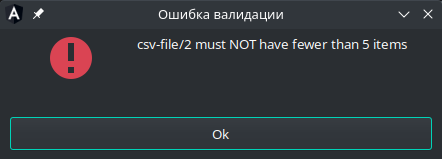
Оконное приложение работает с локально-расположенным файлом, формата CSV (Comma-Separated Values представляет собой файл текстового формата, который предназначен для отображения табличных данных).

CSV файл используется оконным приложением, как постоянное хранилище данных, считывая оттуда данные при начале сессии и перезаписывая их при изменении данных в приложении. Данные операции вызываются путём нажатия соответствующих кнопок, представленных на рисунке 2.1.

Рисунок 2.1 – Функционал взаимодействия с CSV файлом

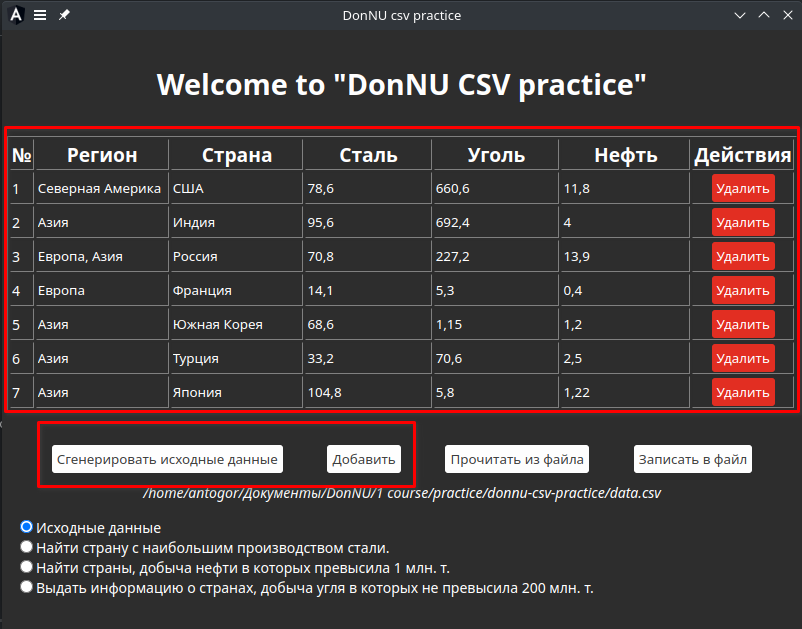
При чтении данных из локального CSV файла приложение проводит обязательную валидацию (проверка значений, указанных пользователем, и отображение найденных ошибок) этих данных. По итогам валидации приложение показывает диалоговое окно с результатами проверки (рисунки 2.2 и 2.3).

Рисунок 2.2 – Диалоговое окно: валидация прошла успешно

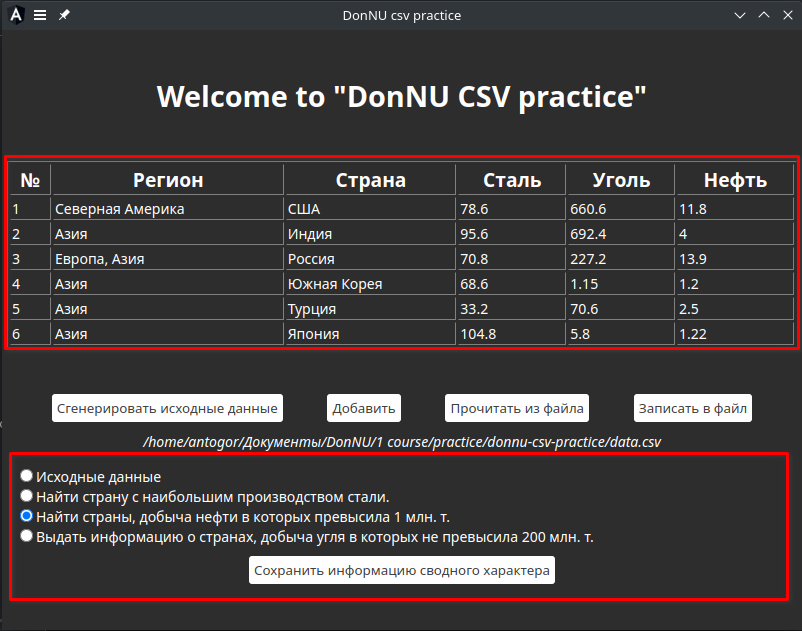
Рисунок 2.3 – Диалоговое окно: ошибка валидации

Приложение предоставляет пользователю доступ к данным, а также обеспечение CRUD операций (рисунок 2.4):

* Create – создание / запись;
* Read – чтение / отображение;
* Update – обновление / корректировка;
* Delete – удаление записи.

Рисунок 2.4 – Функциональное обеспечение CRUD операций

Также приложение предоставляет возможность просмотра данных сводного характера в соответствии с вариантом и сохранения итоговых показателей в отдельный CSV файл. Данный функционал представлен на рисунке 2.5.

Рисунок 2.5 – Информация сводного характера

# ЗАКЛЮЧЕНИЕ

В ходе практики были реализованы база данных и оконное приложение. Было выполнено практическое задание с использованием СУБД MS Access для базы данных, а также TypeScript, платформы Node.js, фреймворков Angular и Electron для оконного приложения.

В ходе работы над заданием использовались знания, полученные по таким дисциплинам: «программирование», «базы данных»

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Philipp Kief : статья «How to create Angular Desktop Apps» : [сайт] – URL: <https://pkief.medium.com/angular-desktop-apps-a9ce9e3574e8> (дата обращения: 20.06.2022).
2. Ahmed Bouchefra : статья «Creating a Desktop App with Electron and Angular» : [сайт] – URL: <https://buddy.works/tutorials/building-a-desktop-app-with-electron-and-angular> (дата обращения: 19.06.2022).
3. Библиотека «ngx-electron» : обсуждение проблемы «Namespace 'Electron.CrossProcessExports' has no exported member 'Remote'. #71» : [сайт] – URL: <https://github.com/ThorstenHans/ngx-electron/issues/71> (дата обращения: 19.06.2022).
4. Angular : официальная документация фреймворка «Angular» : [сайт] – URL: <https://angular.io/docs> (дата обращения: 19.06.2022).
5. Electron : официальная документация фреймворка «Electron» : [сайт] – URL: <https://www.electronjs.org/docs/latest> (дата обращения: 19.06.2022).
6. Node.js : официальная документация библиотеки «fs» : [сайт] – URL: <https://nodejs.org/api/fs.html> (дата обращения: 19.06.2022).
7. Node package manager (npm) : официальная документация библиотеки «electron-packager» : [сайт] – URL: <https://www.npmjs.com/package/electron-packager> (дата обращения: 22.06.2022).
8. Манс Microsoft Access 2,0 за пять минут / Манс, Визе. - М.: Бином, 2017.
9. Бекаревич, Ю. Б. Самоучитель MS Office Access 2016 / Ю. Б. Бекаревич, Н. В. Пушкина. — СПб.: БХВ-Петербург, 2017. — 480 с.: ил. — (Самоучитель)