**Федеральное государственное автономное образовательное учреждение высшего**

**образования**

**«Национальный исследовательский университет «Высшая школа экономики» Факультет компьютерных наук**

**О Т Ч Е Т**  **по дисциплине «Теория Баз Данных»**

**Выполнили студенты**

**Карлов Владимир БПМИ 193, Кононова Ирина БПМИ 194, Петров Олег БПМИ 193**

**(ФИО)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

***(подписи)***

**Проверил:**

*(должность, ФИО руководителя практики)*

*00.00.2021*

*(дата)*

**2021 год**

# Содержание

[Содержание 2](#_Toc35069)

[1. Описание предметной области 3](#_Toc35070)

[*1.1.* Цель 3](#_Toc35071)

[*1.2.* Внешние данные 3](#_Toc35072)

[*1.3.* Основные сценарии использования 3](#_Toc35073)

[2. Концептуальная модель 4](#_Toc35074)

[*2.1.* Диаграмма «Сущность-связь» 4](#_Toc35075)

[*2.2.* Описание сущностей и связей 4](#_Toc35076)

[3. Инфологическая модель 6](#_Toc35077)

[*3.1.* Диаграмма «Таблица-связь» 6](#_Toc35078)

[*3.2.* Словарь данных 6](#_Toc35079)

[4. Даталогическая модель 12](#_Toc35080)

[*4.1.* Используемая СУБД и диалект *SQL* 12](#_Toc35081)

[*4.2.* Хранимые процедуры и триггеры 12](#_Toc35082)

[*4.3.* Описание механизмов обеспечения целостности данных 14](#_Toc35083)

[*4.4.* *DDL*-скрипты 15](#_Toc35084)

[5. Клиентское приложения 18](#_Toc35085)

[*5.1.* Архитектура 18](#_Toc35086)

[*5.2.* Сценарии использования 18](#_Toc35087)

[*5.3.* Организация доступа к данным 18](#_Toc35088)

[*5.4.* Интерфейс с пользователем 18](#_Toc35089)

[*5.5.* Отчёты 19](#_Toc35090)

[6. Заключение 26](#_Toc35091)

[*6.1.* Объёмные характеристики разработки 26](#_Toc35092)

[*6.2.* Авторский вклад и комментарии по выполнению проекта 26](#_Toc35093)

[*6.3.* Репозиторий 26](#_Toc35094)

[7. Источники 27](#_Toc35095)

# Описание предметной области

В наши дни огромное количество людей использует автомобили. На данный момент топовые производители топлива ‒ производители нефти и газа. В своем проекте мы хотим создать продукт, с помощью которого можно будет осуществлять контроль над поставками некоторых видов топлива.

## Цель

Цель: создание приложения (Telegram-бот) для просмотра и учета информации о сетях АЗС,

определенных точках и поставках. На основе данных выдавать определенные статистики.

В перспективе приложение должно уметь отслеживать цепочку поставок от места добычи газа/нефти до распределения топлива от перерабатывающих компаний по определенным сетям АЗС и отдельным точкам. Безусловно, для этого потребуется внести изменения в имеющуюся структуру проекта, начиная с небольших изменений в ER-диаграмме и заканчивая самим клиентом.

## Внешние данные

В роли внешних данных выступали отчетности нефтегазовых компаний и следующий сайт: [https://energybase.ru.](https://energybase.ruс/) Также некоторые данные (такие как данные по таблице заказов) приходилось генерировать самостоятельно. Это связано с неимением доступа к подобного рода информации.

## Основные сценарии использования

Важной частью является сбор и хранение данных о поставках/продажах. Это необходимо для возможности ведения учета об операциях компаний. При этом в нашем продукте сбор данных обязательно должен проходить через создателей данного проекта. Вывод информации реализуется с помощью чат-бота. Основной сценарий работы приложения происходит от лица внешнего пользователя. Он просматривает собранную в базе данных (БД) информацию через реализованные функции. Первым делом он запускает чат-бот, а далее совершает действия через интуитивно понятный интерфейс взаимодействия (кнопки).

# Концептуальная модель

## Диаграмма «Сущность-связь»

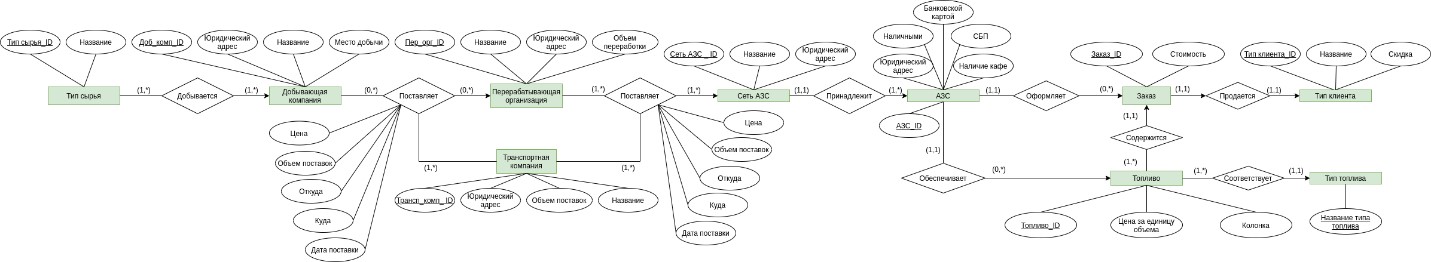


Рисунок 2.1. ER-диаграмма в нотации Чена

Диаграмму в лучшем качестве можно посмотреть по ссылке:

<https://github.com/ArtIrina/TDB_project/blob/main/ERD-ERD.drawio%20(3).png>

## Описание сущностей и связей

ER-диаграмма создавалась на основе выделенных нами сущностей.

Пусть существует сеть АЗС. Для нормального функционирования она должна наладить поставки продаваемого на ней товара (топлива). Исходя из этого, понимаем, что существует связь с поставщиками. Получив в свои хранилища товар, они должны найти, куда его сбыть. Так появляется некоторый клиент в данной области.

Разобравшись чуть детальнее, мы выделили 10 сущностей. У девяти из них имеется уникальный идентификатор ‒ name\_ID. Уникальность десятой (тип топлива) формируется через единственный атрибут “Название типа топлива”.

Перечислим сущности и их атрибуты:

* Тип сырья – то, какие типы сырья могут встречаться в нашей БД.  
  Атрибуты: Типы\_сырья\_ID, Название.
* Добывающая компания – сущность, описывающая добывающие топливо организации. Атрибуты: Доб\_комп\_ID, Юридический адрес, Название, Место добычи.
* Перерабатывающая компания – сущность, описывающая перерабатывающие топливо организации.  
  Атрибуты: Пер\_орг\_ID, Название, Юридический адрес, Объем переработки (млн. тонн в год).
* Транспортная компания – сущность, описывающая организацию, специализирующуюся на транспортировке топлива или сырья от одной организации другой.  
  Атрибуты: Трансп\_комп\_ID, Юридический адрес, Объем поставок, Название.
* Сеть АЗС – компания, владеющая сразу несколькими АЗС. Например, Лукойл.  
  Атрибуты: Сеть\_АЗС\_ID, Название, Юридический адрес.
* Точка АЗС (далее – АЗС) – сущность, описывающая каждую отдельную автозаправочную станцию.   
  Атрибуты: АЗС\_ID, Юридический адрес, Возможно ли оплата наличными, Банковской картой, СБП, Наличие кафе.
* Заказ – сущность, описывающая заказ, проведенный на АЗС.  
  Атрибуты: Заказ\_ID, Стоимость.
* Топливо – сущность, описывающая проданное топливо, регламентированное в заказе на АЗС.  
  Атрибуты: Топливо\_ID, Цена за единицу объема, Колонка.
* Тип клиента – сущность, описывающая, какой тип клиента совершал покупку на АЗС. Атрибуты: Тип\_клиента\_ID, Название, Скидка.
* Тип топлива – сущность-справочник. Содержит названия каждого доступного вида топлива.  
  Атрибут: Название типа топлива.

Наши сущности объединены друг с другом связями “один к одному”, “один ко многим”, “многие ко многим”. Имеются тернарные связи, связанные с транспортной компанией.

Ниже покажем, как связаны сущности между собой:

* “Добывается” ‒ сырье добывается многими добывающими компаниями, и компания может добывать несколько типов сырья. Связь “многие ко многим”.
* “Поставляет” ‒ тернарная связь. Представлена в двух типах. Первый соединяет добывающую компанию и перерабатывающую компании, между которыми исполняет свои услуги транспортная компания. Второй тип описывается связь между перерабатывающей организацией и сетью АЗС. Как раз ее можно будет расширить для улучшения проекта. Тернарная связь имеет свои атрибуты: цену перевозки, объем поставки, начало пути конкретной перевозки, конец пути конкретной перевозки, а также дату, в которую осуществляется поставка.
* “Принадлежит” ‒ соединяет сеть АЗС и точку АЗС (конкретную станцию). Между ними стоит связь “один ко многим”. Одной сети может принадлежать много АЗС. В свою очередь, одна АЗС принадлежит одной определенной сети АЗС.
* “Оформляет” ‒ связь “один ко многим” между АЗС и Заказом. На АЗС может быть от 0 до N заказов. При этом заказ был оформлен на одной конкретной АЗС.
* “Обеспечивает” ‒ такая же связь, как “Оформляет”, только между АЗС и топливом.
* “Содержится” ‒ связь “один ко многим” между заказом и топливом. В одном заказе может быть несколько позиций топлива, однако одна конкретная строка в БД по топливу имеется в одном конкретном заказе.
* “Соответствует” ‒ связь “один ко многим” между типом топлива и топливом. Топливо соответствует конкретному типу. Тип соответствует множеству топлива.
* “Продается” ‒ связь “один к одному”. Заказ продается одному клиенту. Один тип клиента участвует в заказе.

# Инфологическая модель

## Диаграмма «Таблица-связь»

В данной диаграмме были введены дополнительные таблицы для тернарных связей, для связей “многие ко многим”. Это таблицы “Доб\_Тран\_АЗС” и “Пер\_тран\_АЗС”, “Типы сырья \_ Доб комп”. Проставлены ключи (внешние и первичные). Применялись приемы нормализации. В качестве части нормализации в таблицах имеется идентифицирующий скалярный атрибут (name\_ID).

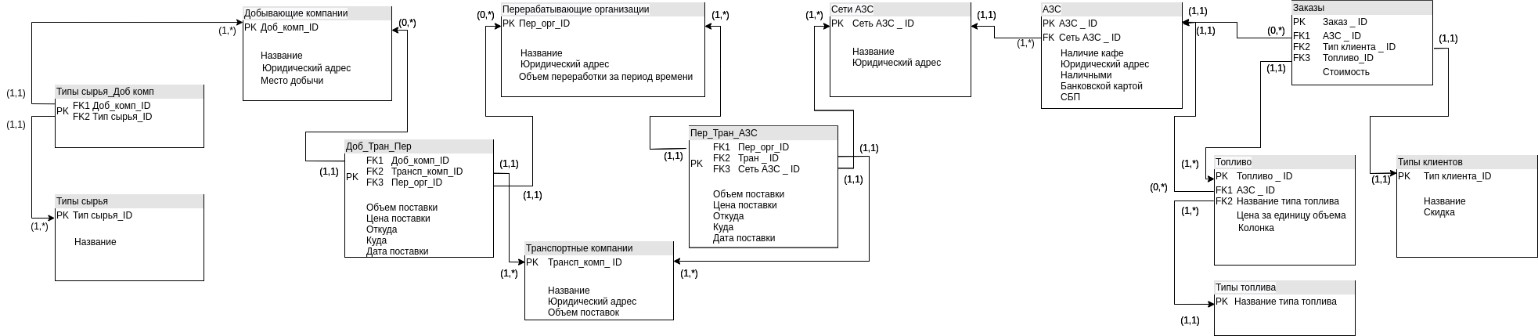


Рисунок 3.2. TR-диаграмма в нотации Чена

Диаграмму в лучшем качестве можно посмотреть по ссылке:

<https://github.com/ArtIrina/TDB_project/blob/main/ERD-TR.drawio%20(2).png>

## Словарь данных

По составленному словарю данных очень удобно составлять TR-диаграмму. Мы определяем какие атрибуты сущностей будут иметь статус внешнего ключа, то же самое для первичного ключа, а также на этом этапе мы определяем типы данных.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Сущность** | **Ключ** | **Атрибут** | **Тип** | **Связь** |
| **Типы сырья**  **FeedstockType** | PK | Типы сырья\_ID  FeedstockType\_ID | int |  |
|  |  | Название  title | varchar(200) |  |
| **Типы сырья\_Доб комп**  **FeedstockType\_MiningOrg** | PK,  FK1 | Доб\_комп\_ID mining\_ID | int | Добывающие компании (Доб\_комп\_ID)  MiningOrg  (mining\_ID) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | PK,  FK2 | Тип сырья\_ID  FeedstockType\_ID | int | Типы сырья (Тип сырья\_ID) FeedstockType  (FeedstockType\_ID) |
| **Добывающие компании**  **MiningOrg** | PK | Доб\_комп\_ID mining\_ID | int |  |
|  |  | Название  title | varchar(200) |  |
|  |  | Юридический ардрес address | varchar(200) |  |
|  |  | Место добычи  location | varchar(200) |  |
| **Доб\_Тран\_Пер**  **Transit\_Refiner** | PK,  FK1 | Доб\_комп\_ID mining\_ID | int | Добывающие компании (Доб\_комп\_ID)  MiningOrg  (mining\_ID) |
|  | PK,  FK2 | Трансп\_комп\_ID trans\_ID | int | Транспортные компании (Трансп\_ комп\_ID) TransporterOrg  (trans\_ID) |
|  | PK,  FK3 | Пер\_орг\_ID refiner\_ID | int | Перерабатывающие организации  (Пер\_орг\_ID)  Refiner (refiner\_ID) |
|  |  | Объем поставки shipment\_vol | float |  |
|  |  | Цена поставки shipment\_cost | money |  |
|  |  | Откуда  \_from\_ | varchar(200) |  |
|  |  | Куда  \_to\_ | varchar(200) |  |
|  |  | Дата поставки shipment\_date | date |  |
| **Перерабатывающие организации**  **Refiner** | PK | Пер\_орг\_ID refiner\_ID | int |  |
|  |  | Название  title | varchar(200) |  |
|  |  | Юридический  адрес  address | varchar(200) |  |
|  |  | Объем переработки за период времени processing\_volume | float |  |
| **Транспортные компании**  **TransporterOrg** | PK | Трансп\_комп\_ID trans\_ID | int |  |
|  |  | Название  title | varchar(200) |  |
|  |  | Юридический  адрес  address | varchar(200) |  |
|  |  | Объем поставок supply\_vol | float |  |
| **Пер\_Тран\_АЗС**  **Transit\_Net** | PK,  FK1 | Пер\_орг\_ID refiner\_ID | int | Перерабатывающие организации  (Пер\_орг\_ID)  Refiner (refiner\_ID) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | PK,  FK2 | Тран\_ID  trans\_ID | int | Транспортные компании  (Трансп\_комп\_ID)  TransporterOrg  (trans\_ID) |
|  | PK,  FK3 | Сеть\_АЗС\_ID net\_ID | int | Сети АЗС (Сеть АЗС\_ID)  GAS\_Station\_Net  (net\_ID) |
|  |  | Объем поставки shipment\_vol | float |  |
|  |  | Цена поставки shipment\_cost | money |  |
|  |  | Откуда  \_from\_ | varchar(200) |  |
|  |  | Куда  \_to\_ | varchar(200) |  |
|  |  | Дата поставки shipment\_date | date |  |
| **Сети АЗС**  **GAS\_Station\_Net** | PK | Сеть АЗС\_ID net\_ID | int |  |
|  |  | Название  brand | varchar(200) |  |
|  |  | Юридический  адрес  address | varchar(200) |  |
| **АЗС**  **GAS\_Station** | PK | АЗС\_ID  station\_ID | int |  |
|  | FK | Сеть АЗС\_ID net\_ID | int | Сети АЗС (Сеть АЗС\_ID)  GAS\_Station\_Net  (net\_ID) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Наличие кафе has\_cafe | bool |  |
|  |  | Юридический  адрес address | varchar(200) |  |
|  |  | Наличными  cash | bool |  |
|  |  | Банковской картой by\_card | bool |  |
|  |  | СБП  fps | bool |  |
| **Заказы**  **Orders** | PK | Заказ\_ID order\_ID | int |  |
|  | FK1 | АЗС\_ID  station\_ID | int | АЗС (АЗС\_ID)  GAS\_Station  (station\_ID) |
|  | FK2 | Тип клиента\_ID customer\_type | int | Типы клиентов  (Тип клиента\_ID)  Customer  (customer\_type) |
|  | FK3 | Топливо\_ID fuel\_ID | int | Топливо  (Топливо\_ID)  Fuel (fuel\_ID) |
|  |  | Стоимость total\_sum | money |  |
| **Топливо**  **Fuel** | PK | Топливо\_ID fuel\_ID | int |  |
|  | FK1 | АЗС\_ID  station\_ID | int | АЗС (АЗС\_ID)  GAS\_Station  (station\_ID) |
|  | FK2 | Название типа  топлива type | varchar(200) | Типы топлива (Название типа  топлива) FuelType (type) |
|  |  | Цена за единицу объема  price | money |  |
|  |  | Колонка pump\_num | int |  |
| **Типы топлива**  **FuelType** | PK | Название типа  топлива type | varchar(200) |  |
| **Типы клиентов**  **Customer** | PK | Тип клиента\_ID customer\_type | int |  |
|  |  | Название  title | varchar(200) |  |
|  |  | Скидка  sale | float |  |

Зеленым цветом выделены соответствующие названия на английском. Так они будут называться в DDL-скрипте, представленном далее.

# Даталогическая модель

## Используемая СУБД и диалект *SQL*

Использовались СУБД SQLite Studio 3.3.3 и диалект SQLite.

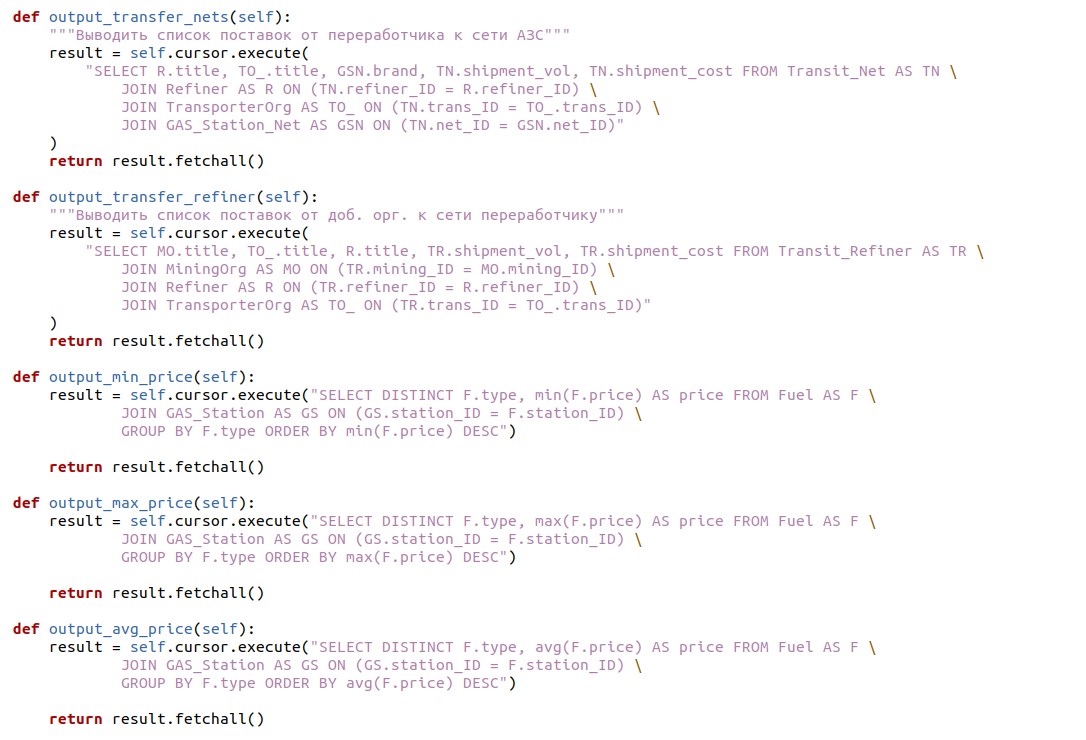
## Хранимые процедуры и триггеры

Процедуры:

* output\_stations - выводить список точек АЗС (адрес, к какой сети АЗС принадлежит точка, есть ли кафе, возможен ли безналичный расчет, есть ли система быстрых платежей);
* output\_minings - выводить называние и адрес добывающей компании;
* output\_refiners - выводить называние и адрес перерабатывающей компании;
* output\_feedstocks\_oil - выводить месторождение нефти, локацию, добывающую компанию;
* output\_feedstocks\_gas - выводить месторождение нефти, локацию, добывающую компанию;
* output\_stations - выводить список сетей АЗС (название сети, адрес, число точек);
* output\_nets - выводить список сетей АЗС (название сети, адрес, число точек);
* output\_transfer\_nets - выводить список поставок от переработчика к сети АЗС;
* output\_transfer\_refiner - выводить список поставок от доб. орг. к сети переработчику;
* output\_min\_prices - выводить минимальную цену по каждому типу топлива для всех точек АЗС;
* output\_max\_prices - выводить максимальную цену по каждому типу топлива для всех точек АЗС;
* output\_avg\_prices - выводить среднюю цену по каждому типу топлива для всех точек АЗС.

Соответствующий код:





## Описание механизмов обеспечения целостности данных

Для минимизации аномалий при работе с реляционной моделью применяли приемы нормализации.

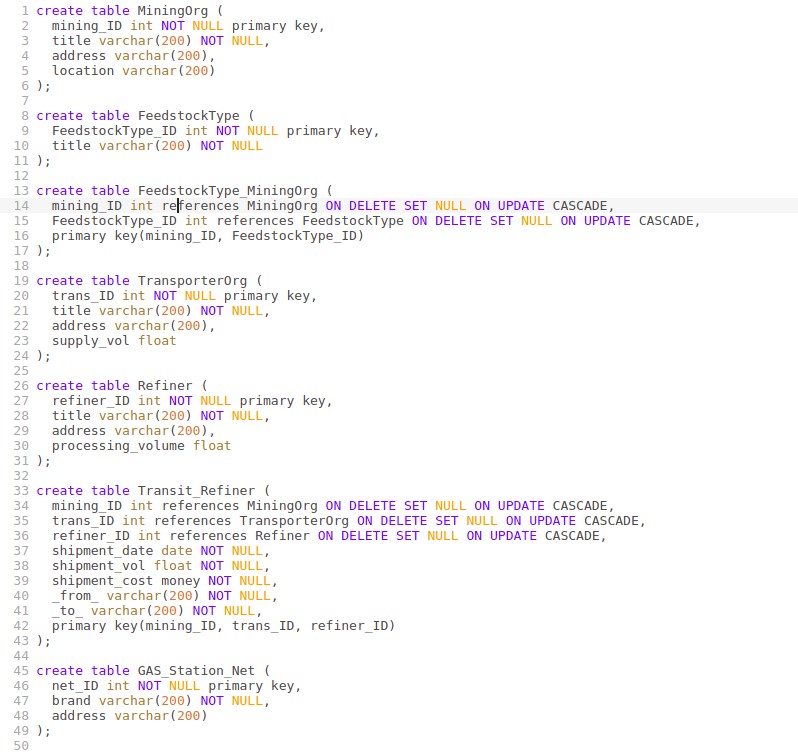
Создали уникальный идентификатор для каждой таблицы, прописали условия на связи: как действовать, если удаляем запись в таблице, как действовать, если обновляем данные таблицы. Для атрибутов, которые мы считаем важными, поставили “NOT NULL”. То есть хотим, чтобы некоторые данные вводились в обязательном порядке. В полях с типом varchar указываем

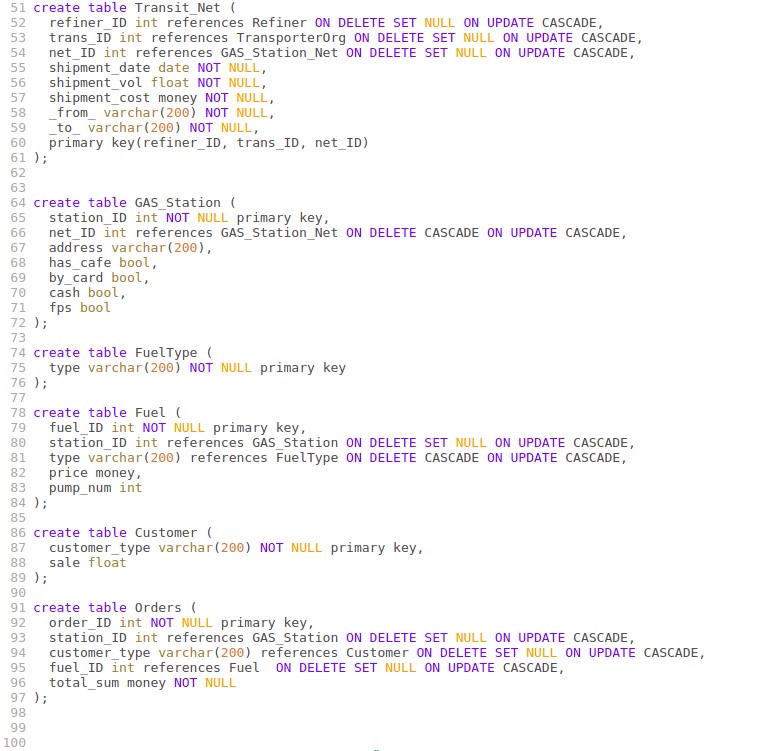
достаточный, на наш взгляд, размер 200.

Мы предоставляем доступ к БД без регистрации. Соответственно, ни к чему добавлять всевозможные проверки на корректную работу пользователя. Если пользователь не использует интерфейс взаимодействия – кнопки в Telergram-боте, значит он не работает, не обращается к нашему приложению. Соответственно, бот не реагирует.

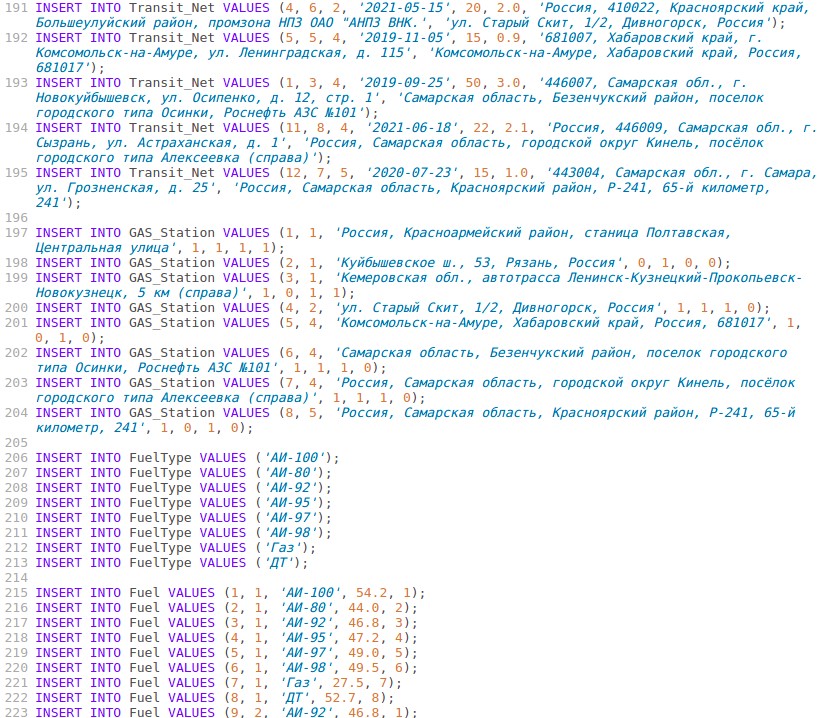
Проход по меню приложения мы ограничиваем, и выдаем сообщение, если пользователь решает пойти дальше разработанного меню. Если на первом меню нажать “назад”, то он скажет, что мы уже на первой странице, и раньше ничего нет. Аналогично для последнего. Только предупреждение уже о том, что дальше ничего нет.

## *DDL*-скрипты





Примеры инсертов (DML код):



# Клиентское приложения

## Архитектура

Наше клиентское приложение ‒ это чат-бот в мессенджере Telegram, реализованный на языке программирования Python.

Библиотеки, использованные в работе: sqlite3, aiogram.

Реализован класс FuelControlDB, через который обеспечивается доступ к базе данных. Проводимые им методы (функции) описаны в пункте 4.2.

## Сценарии использования

Для кого полезно наше приложение? Приложение может принести пользу тем, кто интересуется нефтегазовой логистикой, кому нужна информация по АЗС. Это могут быть сотрудники аналитического или логистического отдела сети АЗС, менеджеры нефтегазовых компаний, акционеры, сотрудники дочерних компаний, налоговая служба, или простые автомобилисты.

Данный бот позволяет своим пользователям получать информацию из базы данных по запросам, используя функции, перечисленные в предыдущем пункте. Вывод информации осуществляется очень быстро. Это важный критерий, который ценят пользователи.

Стандартный сценарий использования выглядит так:

1. открываем чат с ботом в Telegram;
2. нажимаем «старт» или же вводим команду /start;
3. выбираем меню с интересующим нас запросом (кнопки /назад, /далее);
4. нажимаем на запрос и просматриваем появившийся вывод;
5. либо завершаем пользование, либо повторяем пункты 3, 4.

## Организация доступа к данным

Пользователь имеет доступ к данным, обозначенным в кнопках, только на просмотр. Все изменения в таблицы вносят исключительно создатели проекта, так как основной целью бота является обеспечение read-only доступа к хранимым данным. Отмечается, что главной идеей было создать справочник.

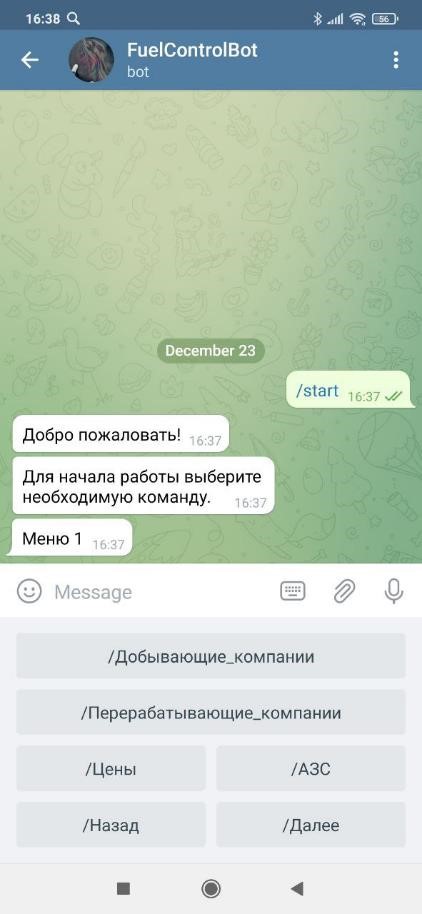
## Интерфейс пользователя

Пользование приложением (чат-ботом) осуществляется через Telegram. Все взаимодействие осуществляется через кнопки (см. пункт 1.3 и 5.2).

## Отчёты

Вот как выглядит работа в нашем приложении.

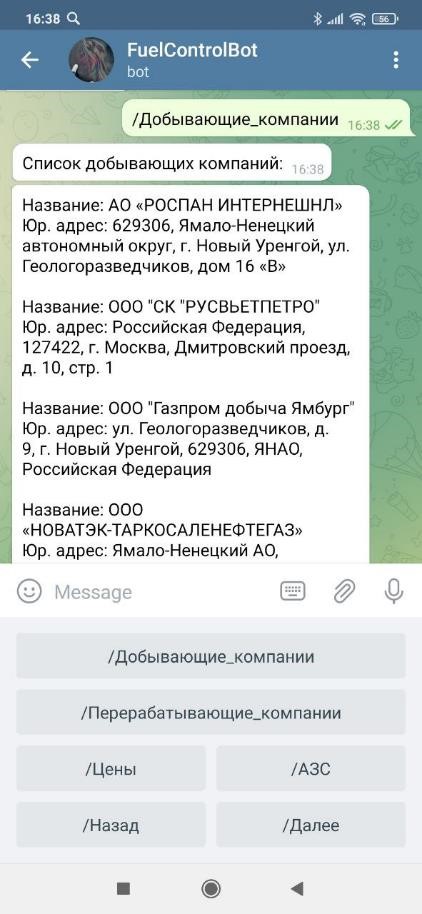
Начало работы с ботом:

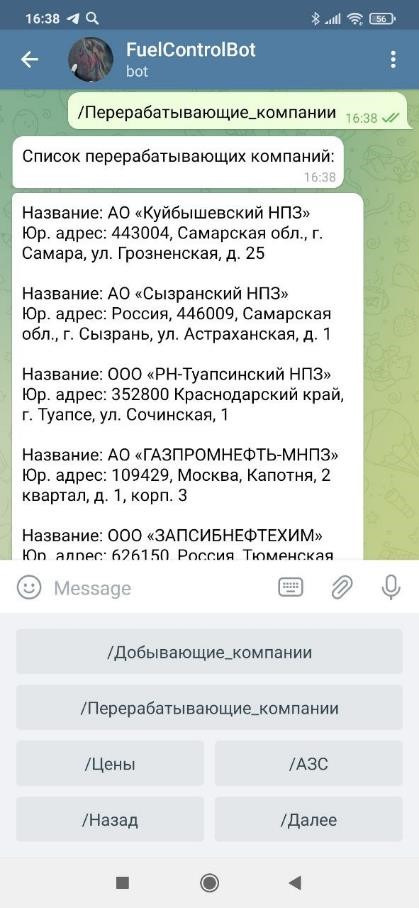


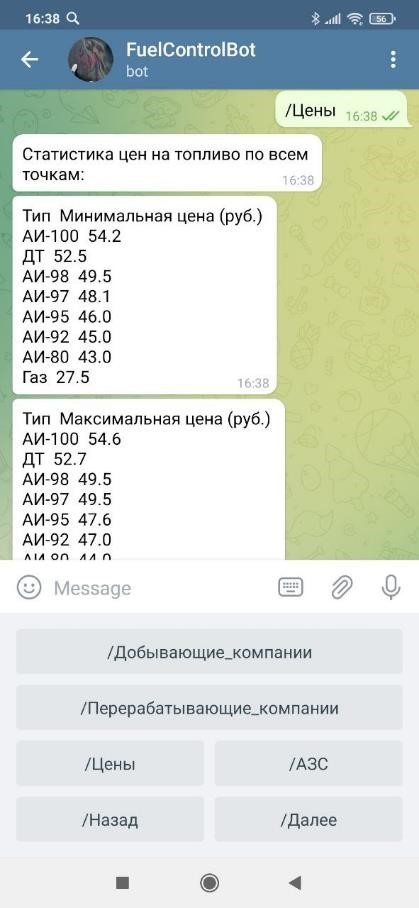
(Нажимаем на старт)

Мы оказались на уровне первого меню, где уже представлена часть команд.

Запросы:

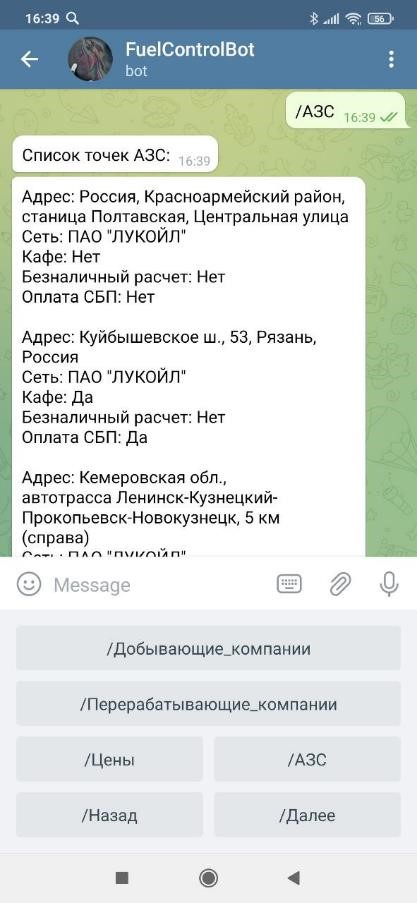
Список добывающих организаций

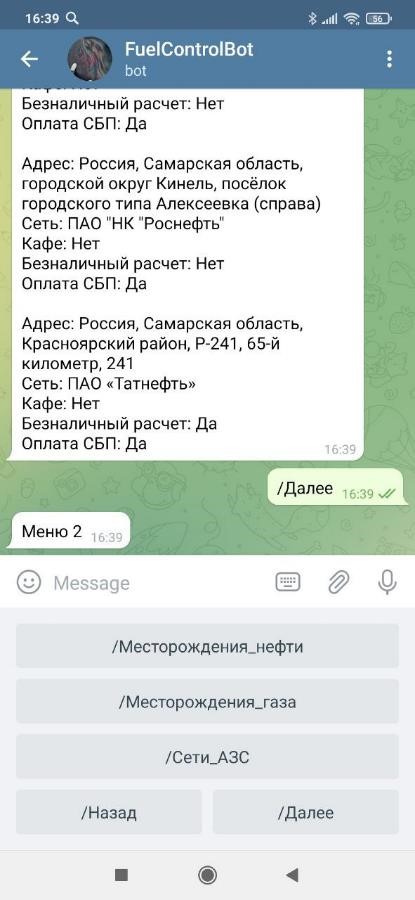
Список перерабатывающих организаций

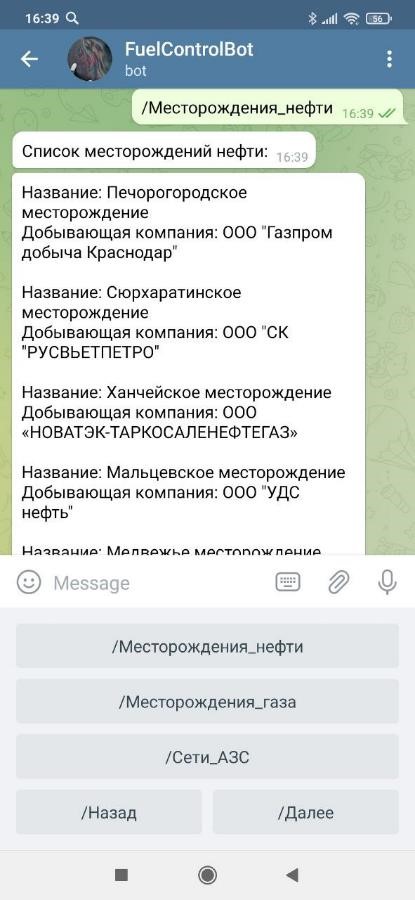
Цены

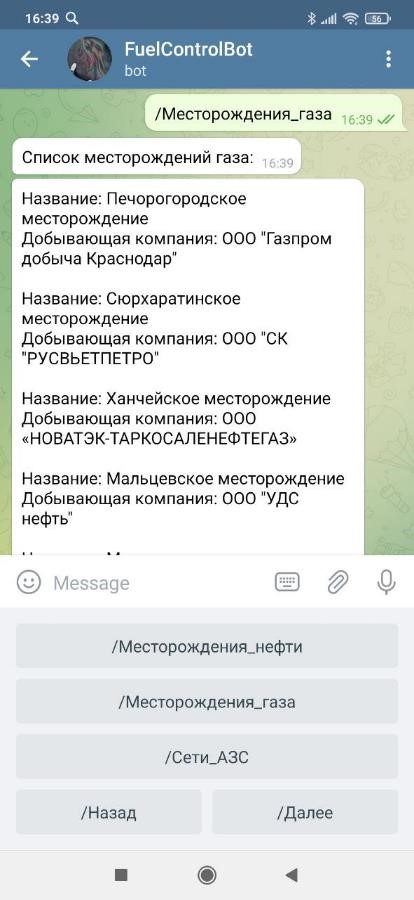
Вид статистического запроса.

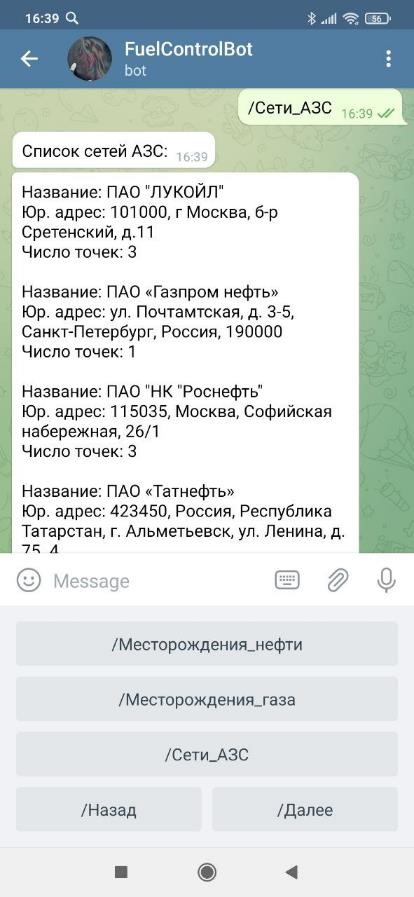
Идет подсчет минимальной / максимальной / средней цены по всем АЗС.

Список АЗС

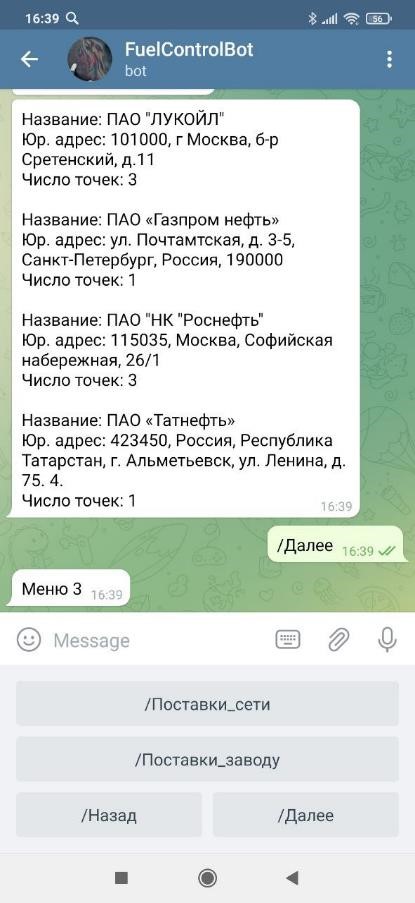
Функции в первом меню закончились. Чтобы посмотреть остальные виды выводимой нами информации, перейдем на меню 2.

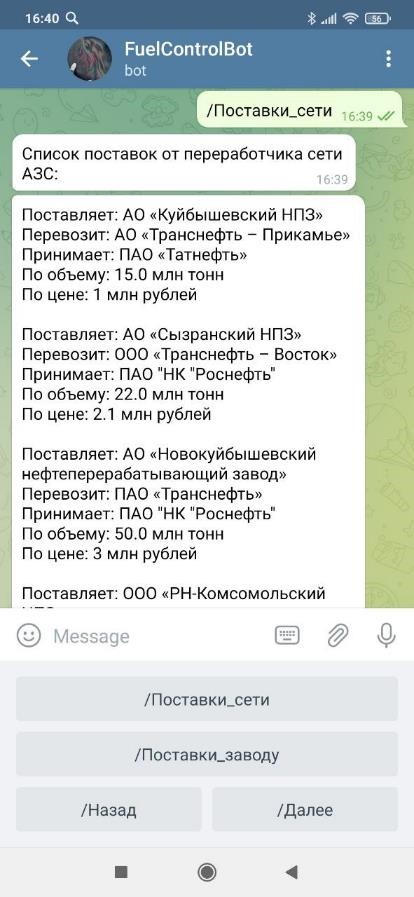
Список месторождений нефти

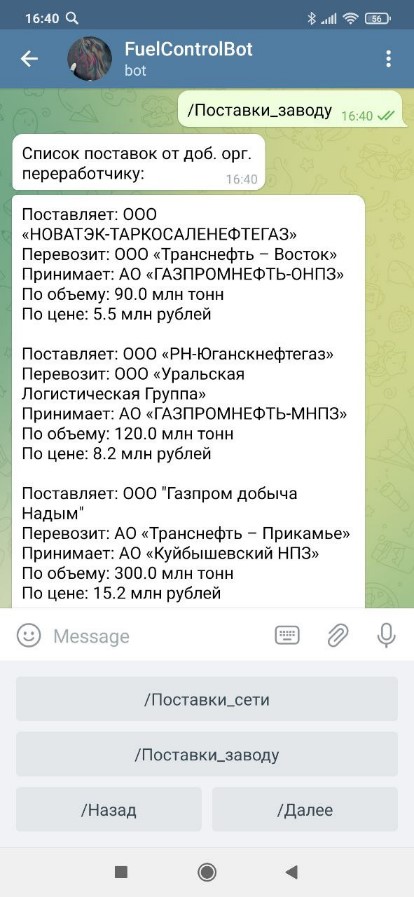
Список месторождений газа



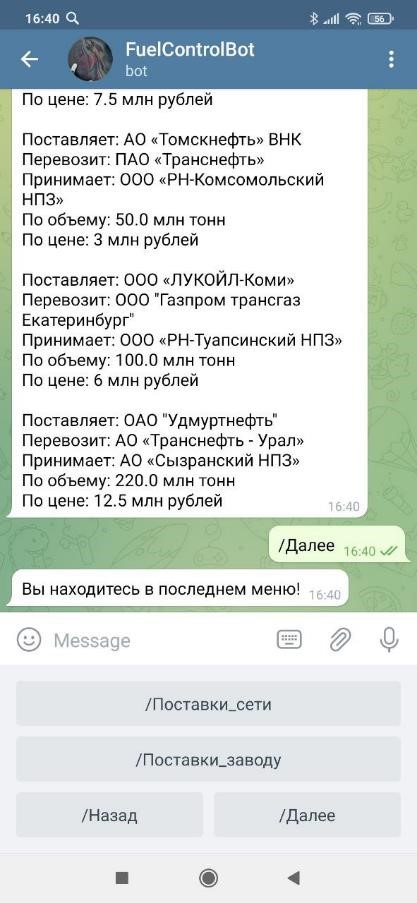
Список сетей АЗС

Переход к меню 3. Аналогично переходу к меню 2, рассмотренному выше

Список поставок от перерабатывающей организации к сети

Список поставок от добывающей организации к перерабатывающей

Как выглядит защита от неправильных действий клиента



# Заключение

## Объёмные характеристики разработки

В процессе работы над проектом было выделено 10 сущностей, каждая из которых имеет от 1 до 6 атрибутов. Была создана база данных. Она состоит из 13 отдельных табличек, соединенных различными видами связи.

Бот реализуется с помощью 12 написанных на python файлов. Умеет выводит общую информацию об объектах и статистику по ним.

Количество реализованных функций в приложении: 9.

## Авторский вклад и комментарии по выполнению проекта

При выборе идеи мы не были знакомы с аналогами нашего проекта, считали его уникальным. Однако в процессе работы, столкнулись с наличием чат-ботов по смежным темам. Проект оказался достаточно объемным, и будь у нас больше времени, возможного на реализацию, были бы реализованы более сложные запросы обращения к БД и дополнительный функционал, например, вывод информации на карте.

За время выполнения данной работы наша команда научилась работать с БД и продумывать детали ее реализации.

## Репозиторий

Ссылка на репозиторий: <https://github.com/ArtIrina/TDB_project>

# Источники

1. energybase.ru [Электронный ресурс] / Режим доступа: [https://energybase.ru](https://energybase.ru/) свободный.

(дата обращения: 18.12.2021)

1. Ternary Relationship in ER Modeling / Trevor H. Jones & Il-Yeol Song. – College of Information Studies Drexel University, Philadelphia, P.A., 19104, 1993. – 30с.
2. Лукойл [Электронный ресурс] / Режим доступа: <https://lukoil.ru/>свободный. (дата обращения: 18.12.2021)
3. Роснефть [Электронный ресурс] / Режим доступа: <https://www.rosneft.ru/> [своб](https://energybase.ru/)одный. (дата обращения: 18.12.2021)
4. Газпром нефть [Электронный ресурс] / Режим доступа: <https://www.gazprom-neft.ru/>свободный. (дата обращения: 18.12.2021)
5. TATNEFT [Электронный ресурс] / Режим доступа: <https://www.tatneft.ru/>свободный. (дата обращения: 18.12.2021)
6. Нефть Магистраль [Электронный ресурс] / Режим доступа: <https://neftm.ru/>свободный. (дата обращения: 18.12.2021)