Geekbrains

**Изучение основных этапов извлечения, обработки, хранения данных и формирование BI отчета на их основе.**

Программа: Разработчик

Специализация: Аналитик

Техническая специализация: Инженер данных

Окуневич Ирина Васильевна

Минск

2024

**Содержание**

Введение

1. Немного теории
2. Веб-скрепинг данных из сайта при помощи Python
3. Работа с базой данных
4. Настройка подключения БД к BI инструменту
5. Подключение к данным и создание отчета в Power BI Desktop
6. Публикация отчета в power BI service и подключение к нему через MS Excel

Заключение

Список используемой литературы

Приложения

**Введение**

В крупных организациях присутствуют различные отделы, такие как разработчики, специалисты по данным и аналитики. Каждый из этих отделов фокусируется на своей специфической области экспертизы. Вместе с тем далеко не каждая компания нуждается в огромных командах и дорогих инструментах обработки, но данные нужны всем. В своей работе я хочу пройти по всем базовым шагам потока данных в организации, объединить их в универсальный рабочий процесс, чтобы лучше понимать, как осуществляется обработка и трансформация данных. Решение, описанное в данной работе, включает в себя следующие  ключевые этапы: извлечение данных из внешнего источника, их загрузка в базу данных организации, подключение к этой базе данных с помощью Power BI Desktop, создание отчета и его публикация. В добавок к ним представлен еще один шаг - подключение к опубликованному отчету через MS Excel и извлечение данных в стандартную таблицу Excel.

Конечно, в условиях обработки объемных потоков данных данный вариант может казаться излишне упрощенным, важно не забывать о потребности в данных со стороны не только крупных, но и мелких предприятий. Представленное мною решение служит иллюстрацией того, как можно организовать процесс при помощи доступных инструментов и знаний одного человека, а не нескольких команд и дорогих платных инструментов. Т.к. в моей работе на примере и подробно рассмотрены необходимые шаги, решение может быть принято на вооружение, адаптировано и усложнено для решения конкретных прикладных проблем.

Итак, представим что мы сотрудничаем с небольшим агенством, которое предоставляет аналитические продукты в какой-то своей узкой области. У агенства есть наработанная база внутренних данных, которую они используют для формирования аналитических отчетов. Но для сравнительного анализа специалистам также нужны общие «глобальные» данные, которые можно найти в интернет, добавить в свою БД и использовать для нужд и целей нашей организации. В своей работе я буду использовать данные о населении стран Европы. Но в реальности это могут быть какие угодно необходимые данные (к примеру, использовать данные сети LinkedIn для изучения рынка труда и предоставление сравнительного анализа рынка и обобщенных данных по компании).

Наши данные будем извлекать из ресурса: <https://www.worldometers.info/population/countries-in-europe-by-population/>

**Задачи работы:**

1. Пройтись по основным шагам: извлечени, сохранение, использование сохраненных данных для создания отчета.
2. Изучить теоретическую часть, в частности использования MongoDB в процессе, а также создания подключения Power BI к MongoDB
3. Подтвердить функциональную работоспособность предложенного решения.

**Используемые инструменты:** MongoDB, MongoDB Compass, Python, Git, Visual Studio Code, Power BI Desktop, Power Bi Service, MS Excel и другие

**Состав команды:** Окуневич Ирина (аналитик, иеженер данных)

**Немного теории**

**Data Engineering (Инженерия данных/Инжиниринг данных/Дата инжиниринг)** — это практика проектирования и создания систем, которые собирают, управляют, преобразуют и сохраняют данные в пригодном для использования состоянии, а также предоставляют доступ к данным для различных специалистов по данным (data scientists, business analysts, BI analyst и т.д.). Специалисты по работе с данными смогут использовать их в своей работе. Т.е. другими словами Data Engineering помогает сделать данные более полезными и доступными для потребителей данных.

**Data Engineer (инженер данных)** — это специалист, который занимается подготовкой данных для их дальнейшего анализа.

Инженеры по обработке данных должны работать как со структурированными, так и с неструктурированными данными. Следовательно, им нужен опыт работы с базами данных SQL и NoSQL, а также с различными форматами данных и файлов (json, xml, csv и т.д.) и уметь работать с различными API. Data Engineers позволяют специалистам по обработке данных выполнять свои операции с данными.

Как писалось выше, на своей работе я покажу основные этапы процесса, который можно представить этой схемой:

Извлекаем данные из внешнего ресурса

Подключа-емся к отчету из MS Excel

Создаем отчет, публикуем его

Подключаем базу данных к BI инструменту

Загружаем в базу данных MongoDB

Преобразуем, сохраняем в формате json

**С**

**Веб-скрепинг данных из сайта при помощи Python**

В своей работе я буду веб-скрепингом, который позволит припомощи python кода собирать данные с интернет-страницы для последующего анализа. Веб-скрепинг используется не только в исследовательском анализе для комплексных исследований, где требуется объединение различных источников данных, но и при мониторинге цен, сентимент-анализе (анализ отзывы клиентов ), сбора данных для машинного обучения, в работе бизнес-интеллектуальных приложений , которые помогают принимать обоснованные решения.

Необходимая нам информация находится по данному URL адресу: <https://www.worldometers.info/population/countries-in-europe-by-population/>

Для скрепинга необходимой информации с сайта напишем код на языке программирования python. Для этого нам нужно зайти в одну из CDE, т.е. сред разработки. Я в своем проекте буду писать код в Visual Studio Code, в который буду заходить из Anaconda.Navigator. Анаконда — это дистрибутив Python, который включает множество полезных питоновских библиотек. Для работы с "виртуальными окружениями" в Анаконде можно использовать инструмент, который называется conda и который входит в состав дистрибутива по умолчанию. Перед началом работы создаю и активирую виртуальное окружение.

Остановимся немного подробнее на коде. Для его выполнения нам нужно импортировать следующие библиотеки:

import requests

from lxml import html

import json

from pymongo import MongoClient

* **requests** - для отправки HTTP-запросов, в нашем случае чтобы запрашивать веб-страницу и получить ее содержимое
* **from lxml import html** - это конкретный импорт, который подгружает модуль html из библиотеки lxml. Этот модуль содержит функции для разбора HTML. Перед импортом необходимо было усановить библиотеку lxml командой pip install lxml.
* **json** – для того чтобы сохранитьданные в формате .json и последующей загрузки содержимого этого файла в базу данных.
* **from pymongo import MongoClient** - MongoClient это класс из библиотеки pymongo, который предоставляет интерфейс для соединения с MongoDB сервером. PyMongo - это популярная библиотека Python для работы с MongoDB. Ее предварительно необходимо установить командой pip install pymongo.

Функции и их описание:

* функция для веб-скрейпинга табличных данных сайта:

def scrape\_page\_data(url, user\_agent):

    response = requests.get(url, headers = {"User-Agent": user\_agent})

    tree = html.fromstring(response.content)

    table\_rows = tree.xpath('//table/tbody/tr')

    data = []

    for row in table\_rows:

        name = ''.join(row.xpath(".//td/a/text()"))

        columns = row.xpath(".//td/text()")

        data.append({

            "country": name,

            "population": int(columns[1].replace(",","")),

            "net\_change": int(columns[3].replace(",","")),

            "land\_area\_sq\_km": int(columns[5].replace(",","")),

            "dencity": int(columns[4].strip().replace(",","")),

            "med\_age": columns[8].strip().replace(",",""),

        })

    return data

Функция отправляет GET-запрос на указанный URL, получает оттуда HTML-страницу и преобразует ее в объект, с которым удобно работать. Затем она ищет все строки таблицы на этой странице и из каждой строки извлекает данные, которые затем добавляются в список словарей. Каждый словарь содержит определённую информацию о стране, включая население, изменение численности населения, площадь территории и т.д. В конце работы функция возвращает этот список словарей.

Также на этом этапе мы преобразуем тип данных некоторых из текстового в integer, чтобы с цифрами впоследствии было удобно работать и они не содержали лишние запятые, которые по умолчанию тянутся к нам из таблицы сайта.

* сохранения данных в json файл;

def save\_to\_json(data):

    with open('data.json', 'w') as f:

        json.dump(data, f)

* функция сохранения данных из json файла в базу данных

def upload\_to\_db (my\_file):

    client = MongoClient('mongodb://localhost:27017/')

    db = client['diploma'] # выбор базы данных

    collection = db['external\_data']

    with open(my\_file) as f:

        db\_data = json.load(f)

    collection.insert\_many(db\_data)

Здесь мы: подключаемся к MongoDB, запущенному на локальной машине, на стандартном порту 27017 🡪 выбираем базу данных с именем "diploma" 🡪 выбираем (или создаем) коллекцию под названием "external\_data" в рамках выбранной базы данных 🡪 открываем файл, путь к которому был передан в функцию в качестве аргумента my\_file 🡪 и загружаем данные из этого файла в базу данных в коллекцию "external\_data". На подготовке БД мы остановимся чуть позже.

* функция main, в которой собираются предыдущие функции и исполняется задуманное. Здесь мы также указываем адрес страницы, из которой мы будем забирать данные, а также user agent.

def main():

    base\_url = "https://www.worldometers.info/population/countries-in-europe-by-population/"

    user\_agent = "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36"

    data = scrape\_page\_data(base\_url, user\_agent)

    save\_to\_json(data)

    upload\_to\_db('data.json')

    print("Data loaded")

После запуска кода получаем сообщение о том, что данные загружены в нашу БД:

A black screen with yellow and green text

Description automatically generated

**Работа с базой данных**

Оговорюсь сразу, я прекрасно понимаю, что для табличной структуры данных, с которой мы работаем, лучше подойдет реляционная база данных – например, MySQL или MS SQL Server. Но так как я достаточно хорошо знаю, как работать с SQL и реляционными базами данных, из академических целей я остановилась на MongoDB – чтобы больше узнать о ней и о NoSQL базах данных в целом, а также ***изучить вопрос подключения Power BI к MongoDB и попробовать это подключение на практике.***

Итак, что из себя представляет MongoDB?

MongoDB — система управления базами данных, которая работает с документоориентированной моделью данных. В отличие от реляционных СУБД, MongoDB не требуются таблицы, схемы (что повышает производительность всей системы) или отдельный язык запросов. Информация хранится в виде документов либо коллекций.

СУБД используется для хранения JSON-документов в коллекциях и осуществления запросов по нужным полям. В отличии от традиционных СУБД, в коллекциях можно сохранять любые виды данных. Особенность в том, что у MongoDB нет связей между документами и “коллекциями”. Т.е. есть набор данных, который никак не связан с другой информацией в базе, и не существует никакого способа объединить данные из различных документов (в SQL-системах это было бы элементарной задачей).

Однако эта СУБД отлично масштабируется, и по сравнению с традиционными SQL-системами, гораздо быстрее осуществляет чтение и запись. Т.е. MongoDB подходит для приложений, в которых практически не используются данные с зависимостями и необходима масштабируемость базы данных.

Для нашего упражнения отсутствие возможности построить связи не является помехой. В Power BI можно подключаться к разным базам данных (и не только), загружать данные и устанавливать необходимые связи внутри самого отчета.

И для того чтобы добавить, просматривать, или при необходимости удалить базы данных и коллекции в MongoDB я буду использовать графичечкий пользовательский интерфейс Compass.

Для начала создадим базу, с коллекцией, которую будем наполнять данными.

A screenshot of a computer

Description automatically generated

Задаем имя базы и коллекции. На самом деле этот шаг можно пропустить, даже если мы не создадим коллекцию и базу в Compass, выполняя код они создадутся автоматически.

A screenshot of a computer

Description automatically generated

База данных создана и она пока не содежит никаких данных:

A screenshot of a computer

Description automatically generated

Но после запуска кода из раздела выше и нажатия кнопки обновения  в Compass, мы видим, что наши данные успешно загружены:

A screenshot of a computer

Description automatically generated

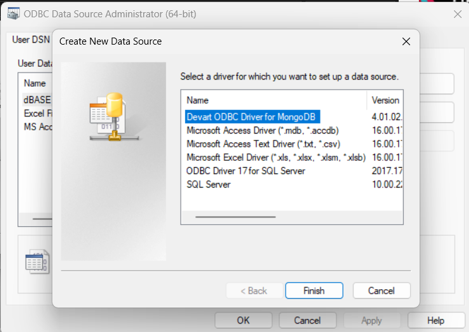
Таким образом, мы извлекли данные из таблицы и сохранили этот документ в базу данных MongoDB. Теперь нужно подключить нашу базу к BI инструменту, чтобы на основе этих данных создать отчет для пользователей.

**Настройка подключения БД к BI инструменту**

Традиционные инструменты бизнес-аналитики работают с плоскими табличными данными. Эти инструменты недостаточно сложны для понимания трехмерных данных, хранящихся в базах данных MongoDB. Можно скачать [BI Connector](https://www.mongodb.com/docs/bi-connector/current/tutorial/install-bi-connector-windows/) и пойти этим путём, но я буду соединять BI и базу данных при помощи ODBC (Open Database Connectivity).

ODBC — это стандартный интерфейс между базой данных и приложением, взаимодействующим с ней. Наличие подобного стандарта позволяет любому приложению на клиентском компьютере получать доступ к любой базе данных на сервере. Единственное требование заключается в том, чтобы клиентская и серверная части поддерживали стандарт ODBC. Приложение получает доступ к конкретной базе данных, используя специально разработанный для нее драйвер (в данном случае драйвер ODBC). Клиентская часть драйвера, работающая напрямую с приложением, должна строго соответствовать стандарту ODBC. Приложению все равно, какая СУБД установлена на сервере. Серверная часть драйвера адаптирована к конкретной базе данных. Благодаря такой архитектуре не только не нужно настраивать приложения на определенную СУБД — им даже не обязательно знать, какая именно СУБД используется. Драйвер скрывает различия между серверами баз данных.

Для того чтобы подконнектиться я скачиваю [ODBC Driver for MongoDB](https://www.devart.com/odbc/mongodb/). И добавляю его к ODBC Data Sources в Control Panel\System and Security\Windows Tool, заполнив данные для подключения:



Задаем Data Source Name (dPowerBI):

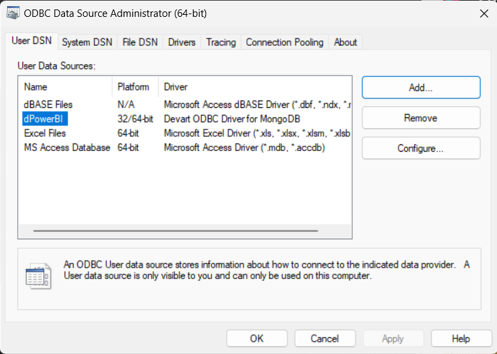
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Предыдущие шаги были успешно выполнены и теперь мы видим наш dPowerBI в списке User Data Sources. Это даст нам возможность подключиться к базе данных при создании отчета в Power BI Desktop.



**Подключение к данным и создание отчета в Power BI Desktop**

Для того чтобы подключиться к нашей базе создаем новый .pbix файл, открыв Power BI Desktop. Выбираем “Get Data”, в строке поиска набираем ODBC и “connect”

A screenshot of a computer

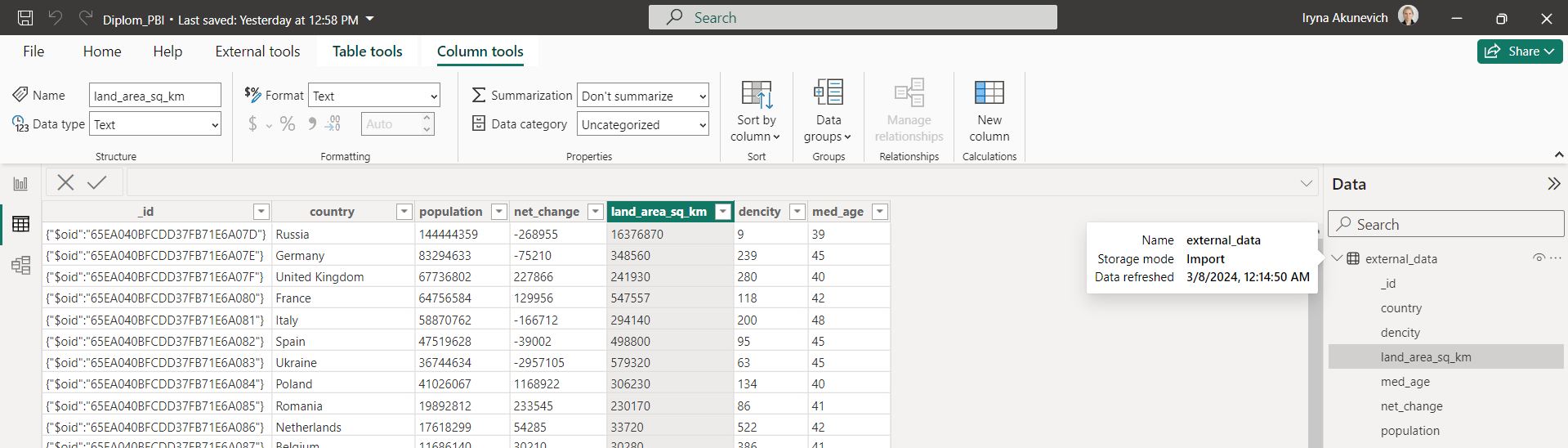
Description automatically generated

Дальше выбираем наше подключене dPowerBI:

A white computer screen with a white background

Description automatically generated

Таким образом мы загрузили наши данные в отчет и можем начинать строить на них визуальные эелементы.



Для того чтобы продемонстрировать, что отсутствие связи в БД не повлияет на построение нашего отчета, загрузим еще одну таблицу – в данном случае просто из файла .csv через графический интерфайс Compass. В той же базе данных создадим еще одну коллекцию company\_data и зальем в нее наш файл, который бедут состоять из трех колонок: страна, количество сотрудников, выручка за 2023 год.

A screenshot of a computer

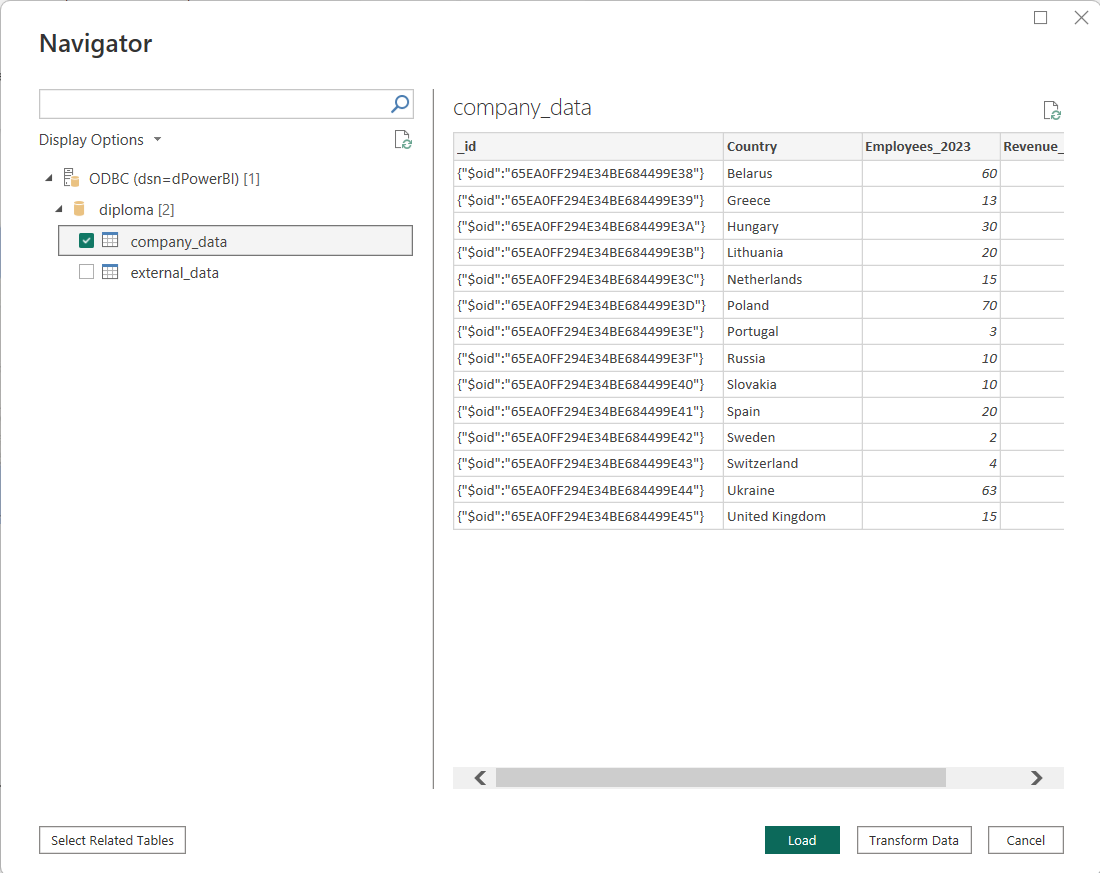
Description automatically generated

Данные загрузились в коллекцию:

A screenshot of a computer

Description automatically generated

Теперь тем же способом через get data подключим новую таблицу к нашему отчету:

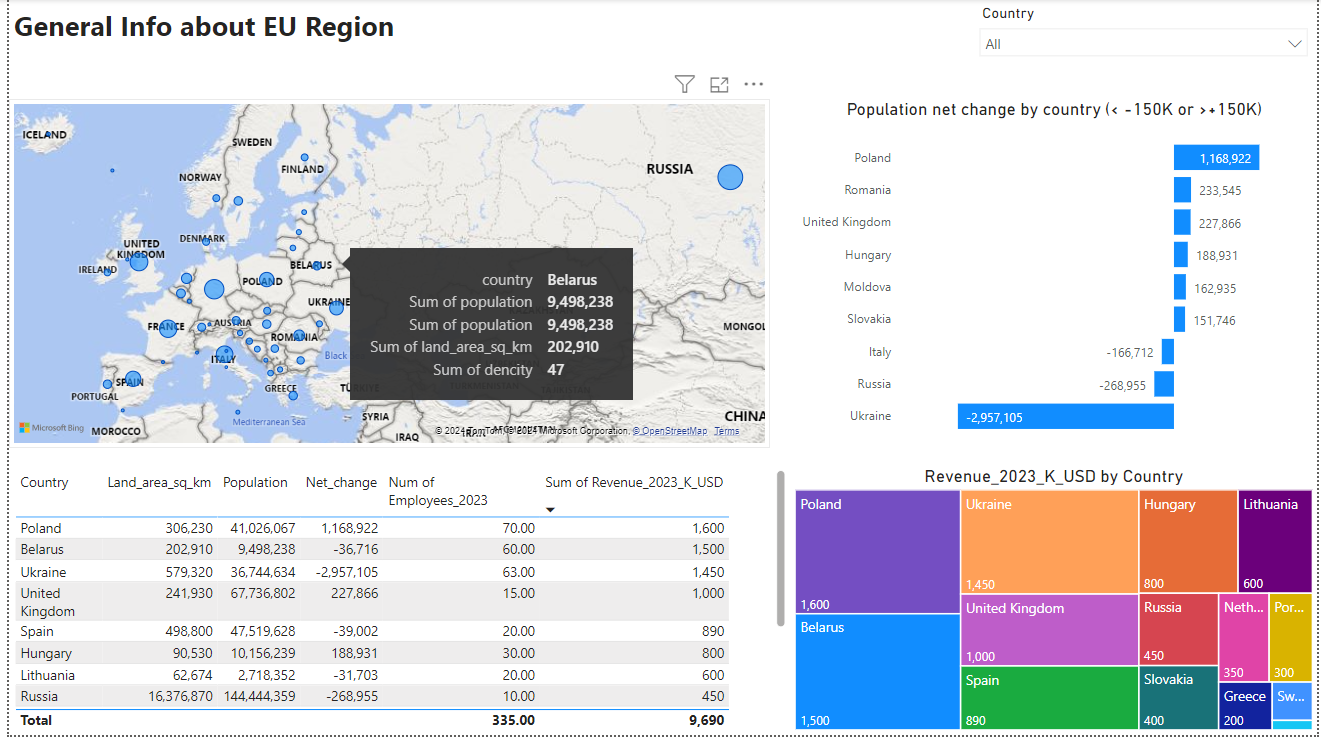


У нас в отчете две таблицы, которые мы можем связать между собой и установить двунаправленную связь, таким образом, чтобы обе таблицы были взаимосвязаны, фильровали одна другую и для расчетов мы могли использовать данные из обеих таблиц. Связь устанавливаем по колонке «Country», которая присутствует в обеих таблицах.

A screenshot of a computer

Description automatically generated

Я не буду подробно останавливаться на том, как строить отчет в Power BI, т.к. это не цель данной работы. В результате у нас получилась вот такая страница отчета, которую мы обубликуем в power BI service, на отдельном рабочем пространстве (workspace). Здесь приведу скриншот страницы отчета и обращу внимание, что в таблице присутствуют дынне из двух импортированных в отчет коллекций.



Сам .pbix файл будет приложен к данной работе и его можно изучить непосредственно в Power BI Desktop.

**Публикация отчета в power BI service и подключение к нему через MS Excel**

После создания отчета публикуем его в рабочем пространстве и выдаем доступ всем сотрудникам организации, кто может его видеть.

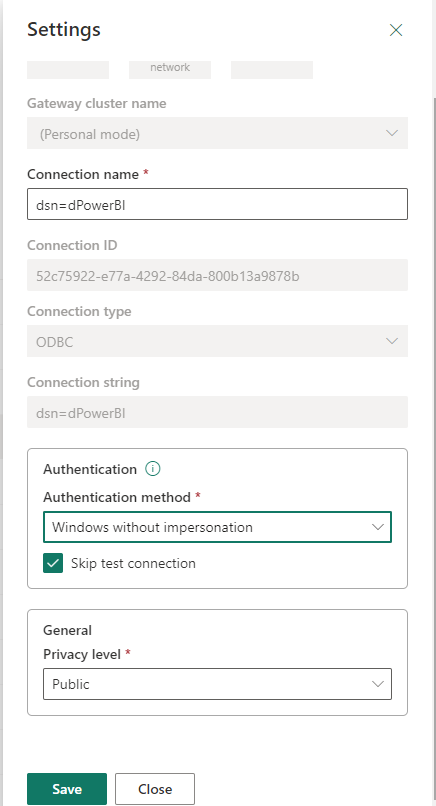
A screenshot of a computer

Description automatically generated

После публикации в Power BI Service в настройках к модели также нужно добавить коннекшен к нашему подключению, предварительно настроив его. А также установить график обновления отчета – та частота, с которой модель будет обращаться к нашей базе и обновдять данные.

A screenshot of a computer

Description automatically generated



Отчетом можно пользоваться: изучать его, выгружать из него данные или сохранять визуальные элементы в pdf формате.

Но дополнительно в своей работе я хочу также продемонстрировать как в можно подключиться из MS Excel к отчету. Это становится возможным после публикации его в рабочем пространстве Power BI Service.

Для чего это может понадобиться? В самом деле, вся необходимая информация есть в отчете, к которому у пользователя есть доступ. Среди визуалов при необходимости могут присутствовать таблицы или матрицы, которые как правило содержат не обобщенную информацию, а операционные данные – во всех необходимых деталях и эти данные можно выгрузить непосредственно из PBI, наведя на три точки над таблицей и выбрав «Export Data».

A screenshot of a computer

Description automatically generated

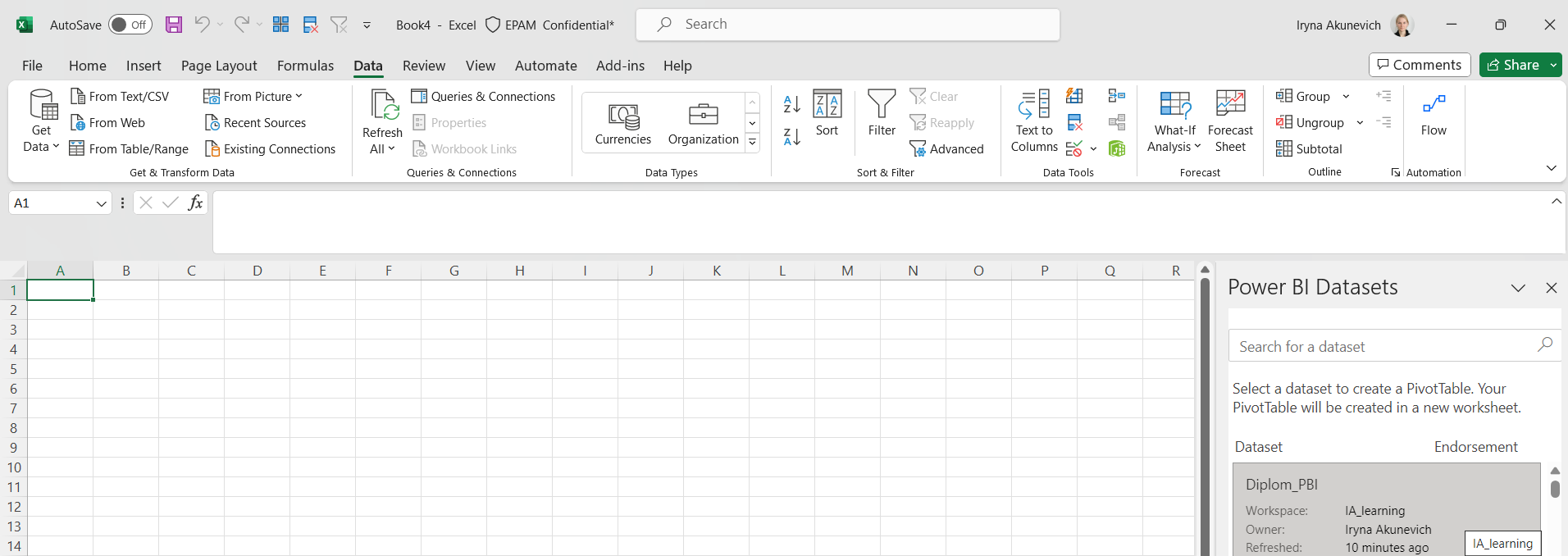
Но у PBI есть ограничение – он не позволяет выгружать в файл более 30 тысяч строк. И если для нашего примера это не критично, т.к. максимальное количество строк у нас равно 47, то на реальных прод данных это ограничение может принести дополнительные трудности. Чтобы получить полный набор данных придется применять фильтры, разбивать нашу выгрузку на несколько частей и после «склеивать». А иногда и этот подход может не помочь в случае когда данных очень много. Решить эту проблему можно при помощи подключения к отчету через Excel. Второй момент – Excel остается очень удобным и важнм инструментом для «быстрой» аналитики и не только. В моей практике этот прием помог быстро сверить данные из двух отчетов и я считаю этот «лайфхак» очень полезным в работе.

Итак, подключаемся к отчету через Excel. И идем во вкладку Data / Get data / From Power Platform / From Power BI

A screenshot of a computer

Description automatically generated

После этого Excel выдаст нам список наборов данных, из которого мы выбираем на датасет и нажатием на него подключаемся.



Видим, что у нас есть доступ к нашим двум таблицам и мы можем работать с подключением как со сводной таблицей, выгружая необходимые нам данные в Excel.

A screenshot of a computer

Description automatically generated

**Заключение**

Таким образом мы успешно прошли самые базовае этапы работы с данными: извлечение из внешнего ресурса, добавление в базу данных, подключение БД к BI инструменту, формирование и публикация отчета для пользователей, и дополнительно – подключение к отчету через Excel для решения рактических задач.

Мы убедились, что табличные данные можно сохранить json файл и залить его в докуменоориентированную базу данных, которую после подключить к BI инструменту и успешно построить отчет на этих данных. В ходе работы я также ближе познакомилась с Mongo DB, ее предназначением, отличием от реляционных баз данных, плючах и минусах.

С практической точки зрения каждый из пунктов данной работы может быть применен на практике: полностью или частично, с адаптацией под конкретные нужды.

**Список используемой литературы**

<https://www.mongodb.com/docs/bi-connector/current/connect/powerbi/#prerequisites>

<https://learn.microsoft.com/ru-ru/sql/odbc/reference/what-is-odbc?view=sql-server-ver16>

<https://ivan-shamaev.ru/data-engineering-etl-pipeline-data-warehouse-datalake/#_data_engineering>

<https://skillbox.ru/media/code/mongodb-chto-eto-za-subd-plyusy-minusy-podvodnye-kamni/>

<https://www.dialektika.com/PDF/978-5-907144-81-1/part.pdf>

<https://itglobal.com/ru-ru/company/glossary/mongodb/>

<https://habr.com/ru/companies/latera/articles/280196/>

<https://www.youtube.com/watch?v=mGYB2u3okDk>

**Приложения**

* Dipl.py
* Diplom\_PBI.pbix
* data.json
* Company\_data.csv
* Ссылка на гит: https://github.com/IrynaAk/diploma.git