Описание C# приложения "Krista Project".

Общее описание:

Написанный мною проект на C# является реализацией данного мне кейс-задания, которое я выполнял на протяжении недели. Приложение сочетает в себе несколько технологий (.NET Fraemwork, PostgreSQL, Docker) и разделов программирования на C# (сетевое программирование, параллельное программирование, парсинг json-данных и десериализация json to C# classes, взаимодействие с Postgres, работа с файловой системой в C# и с потоками ввода-вывода).

С некоторыми из использованных технологий и разделов C# я знакомился впервые, пришлось искать и изучать много информации в интернете. Сразу скажу, что успел реализовать не все что планировал: я не успел прогуглить про создание unit-тестов на C# и разработать их, а также рассчитывал переработать обычное консольное приложение в WPF-проект. Теперь обо всем по порядку:

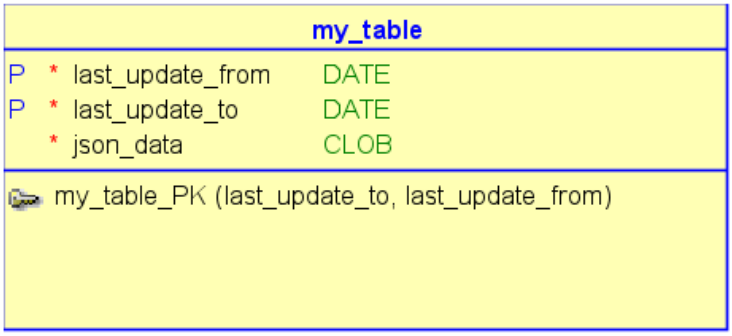
PostgreSQL.

Никогда не работал с этой СУБД, но не встретил ничего такого, чего бы я не делал в Microsoft SQL Server. Разве что не получилось интегрировать созданную базу данных в проект Visual Studio (в случае с MS SQL нужно было просто скопировать файл db_name.mdf и настроить подключение через среду разработки), поэтому для данной задачи мною было принято решение разворачивания базы в контейнере Docker и подключения в коде через соответствующую connection string.

Проектирование БД.

Разработал базу так, как сам понял требования – необходимо было разложить json-данные в плоскую (нереляционную) таблицу с двумя ключами. Поэтому была создана сущность my_table со следующими обязательными атрибутами: начальная и конечная даты (UID сущности), а также данные в формате json.

ER-диаграмма (схема БД):

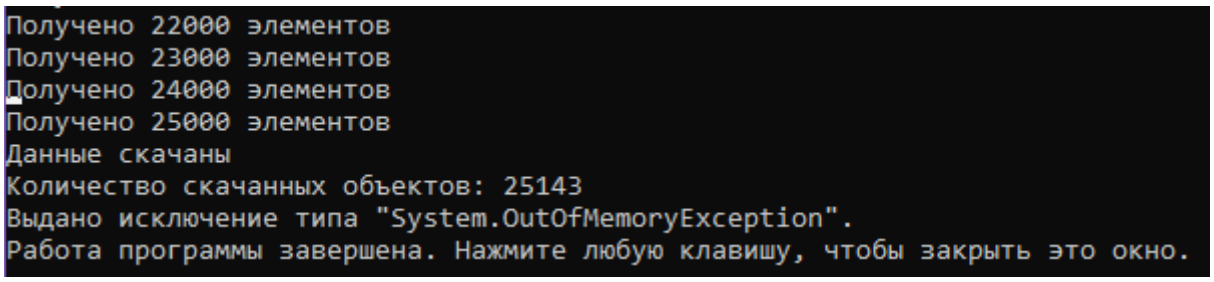


Docker.

Тема контейнеризации тоже была для меня новой, пришлось изучать с нуля. На определенном этапе я познакомился с инструментом Docker Compose, который я и использовал для создания единственного контейнера с базой Postgres. В соответствии с написанным docker-compose.yaml будет скачан образ postgres 15.3-alpine, создан volume где данные базы будут храниться локально на устройстве (даже в случае удаления Docker контейнера структура базы и данные в ней сохранятся) и в конце концов создан контейнер помещенной внутрь базой Postgres, к которой можно подключиться (подключаемся в C#). Единственное что не получилось сделать – проинициализировать базу (создать таблицу my_table) непосредственно средствами docker compose – через точку входа docker-entrypoint-initdb.d. Возможно это не работает только на Windows, но я перепробовал все найденные мною в интернете примеры выполнения инициализирующих базу sql-скриптов самим docker compose, и ничего из этого не сработало. В связи с этим задача инициализации базы переложена на C# и выполняется после подключения к базе непосредственно в коде.

.NET Fraemwork.

- Из раздела сетевого программирования: я не разобрался, как можно параллельно посылать несколько http-запросов через единственный объект HttpClient для оптимизации скорости работы приложения и одновременного получения нескольких страниц от сервера. Поэтому почитал про параллельное программирование и использовал асинхронный метод для создания асинхронных операций (собственно http-запросов), которые должны одновременно выполняться методом `Task.WhenAll`. Для этого в каждой асинхронной операции создается новый экземпляр класса HttpClient, от которого посылается 1 запрос. По умолчанию я выставил 10 параллельных запросов.
- Парсинг json, десериализация – все тривиально.
- Для того чтобы не перегружать вывод, решил не выводить информацию о скачанных объектах в консоль, а ограничиться только показом количества скачанных объектов – число, соответствующее `myObject.content.Length`.
- Сохранение в файл и чтение из него – есть нюанс: если указать временной промежуток с большим количеством данных, то может выброситься OutOfMemoryException. Связано с тем, что пытаюсь считать весь файл целиком в одну строку для формирования sql-команды INSERT или UPDATE с большим содержимым в поле json_data. Как исправить – пока не придумал.



На этом кажется всё, жду обратной связи.