Zaawansowane technologie baz danych

Marcin Rajs, Michał Soboń

Wstęp

Celem projektu było przygotowanie aplikacji do badania popularności piosenek i sprawdzającej, jakie parametry wpływają na liczbę wyświetleń. Wykorzystano dane z istniejącej bazy piosenek dostępnej na kaggle.com jako zbiór Genius Song Lyrics:

https://www.kaggle.com/datasets/carlosgdcj/genius-song-lyrics-with-language-information

Do realizacji zadania wykorzystano 2 bazy nierelacyjne (MongoDB i Elasticsearch) oraz bazę relacyjną (PostgreSQL).

MongoDB

MongoDB została opracowana w języku C++ i opiera się na modelu dokumentowym. Zamiast korzystać z tabel i wierszy, MongoDB przechowuje dane w postaci dokumentów w formacie BSON (Binary JSON), który jest rozszerzeniem JSON. Jedną z głównych cech MongoDB jest jej zdolność do obsługi danych o zmiennym schemacie. Oznacza to, że każdy dokument w kolekcji może mieć inny zestaw pól, co pozwala na elastyczne zarządzanie danymi. To odróżnia MongoDB od tradycyjnych baz danych relacyjnych, w których schemat musi być zdefiniowany z góry.

Elasticsearch

Głównym celem Elasticsearch jest zapewnienie szybkiego i łatwego dostępu do dużych zbiorów danych. Może ono przechowywać, indeksować i wyszukiwać różnego rodzaju dane, takie jak tekst, liczby, geolokalizacje czy struktury złożone. Język zapytań w Elasticsearch umożliwia zaawansowane wyszukiwanie i filtrowanie danych. Można wykonywać zapytania pełnotekstowe, uwzględniać kryteria geograficzne, przeprowadzać agregacje i wiele innych operacji. Elasticsearch obsługuje również indeksowanie w czasie rzeczywistym, co umożliwia natychmiastową dostępność do najnowszych danych.

PostgreSQL

PostgreSQL to zaawansowany system zarządzania relacyjnymi bazami danych (RDBMS). Jest znany ze swojej niezawodności, trwałości i zgodności ze standardami. Ma zaawansowany mechanizm transakcji, który zapewnia spójność danych i możliwość przywracania bazy danych do poprzedniego stanu w przypadku awarii. PostgreSQL obsługuje również mechanizmy zabezpieczeń, takie jak uwierzytelnianie, uprawnienia użytkowników i szyfrowanie danych. Może on obsługiwać duże ilości danych i zapewniać wydajne operacje, takie jak złączenia, sortowanie i grupowanie. PostgreSQL oferuje również zaawansowane optymalizacje zapytań.

Opis projektu i danych

Cały zbiór po rozpakowaniu ma 10 GB i jest zapisany w postaci pliku CSV. Znajduje się tam 7,8 mln utworów oraz ich teksty. Dane składają się z następujących atrybutów:

- id
- title
- genre – w przypadku bazy relacyjnej, dane trafiają do osobnej tabeli
- artist – w przypadku bazy relacyjnej, dane trafiają do osobnej tabeli
- year
- views
- lyrics
- lang

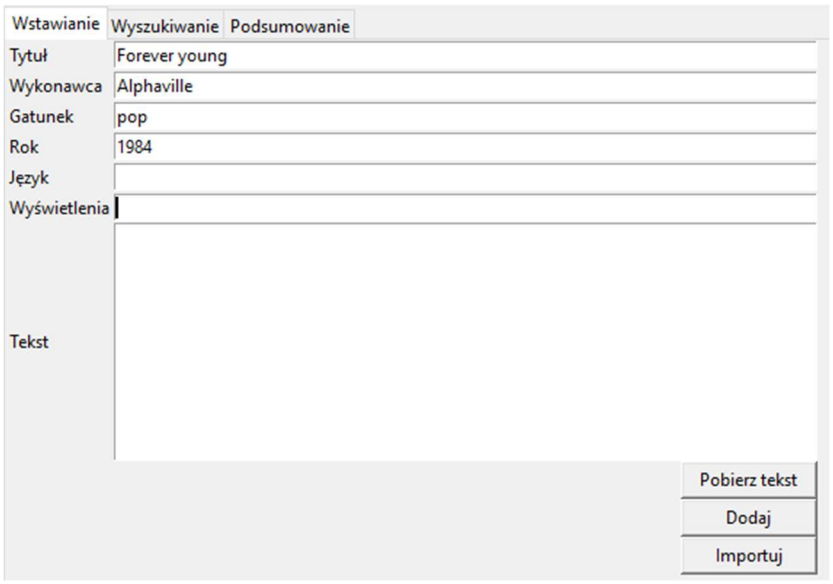
Program napisano w języku Python, korzystając z następujących bibliotek:

- elasticsearch
- psycopg2-binary
- pymongo
- contourpy
- cycler
- fonttools
- importlib-resources
- kiwisolver
- matplotlib
- numpy
- requests
- lxml
- scipy

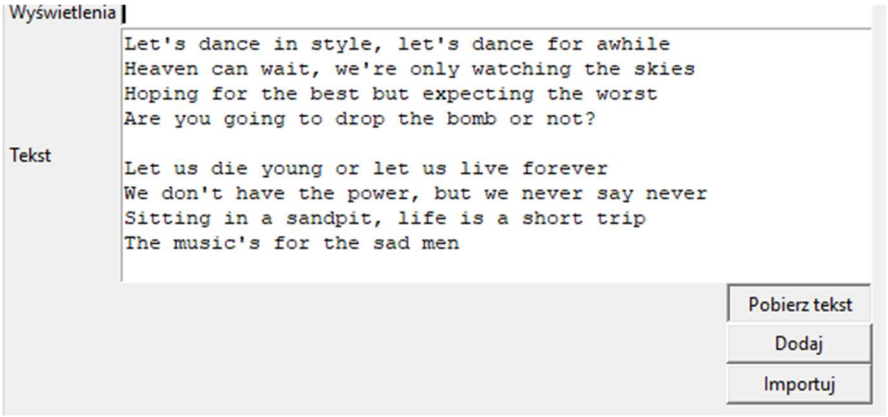
Obliczenia statystyczne realizowane są przez moduł scipy i numpy, a za ich prezentację odpowiada maptlotlib. GUI opracowano z wykorzystaniem modułu tkinter. Ponadto, program umożliwia pobieranie tekstów piosenek dedykowanym scraperem stron internetowych, który działa w oparciu o requests i lxml.

Działanie aplikacji

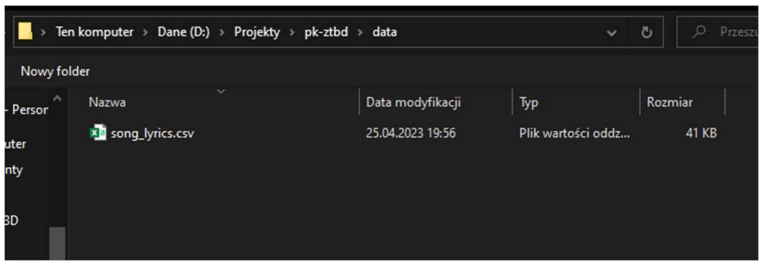
Pierwszy z dostępnych paneli umożliwia wprowadzanie danych, zarówno ręczne jak i importowanie pliku:



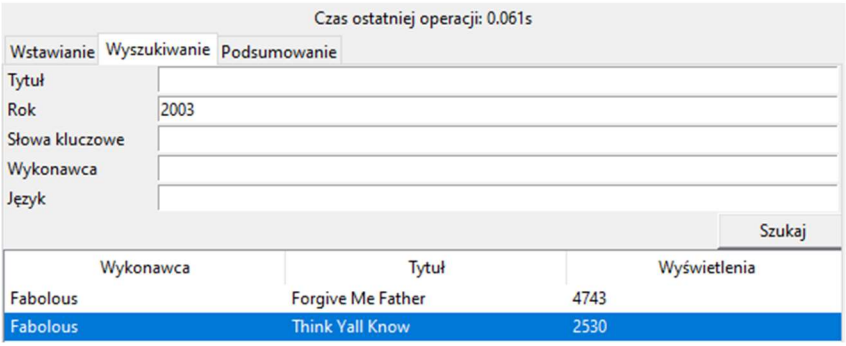
Możliwe jest wypełnienie tekstu piosenki po podaniu tytułu oraz wykonawcy. Dane pobierane są ze strony https://www.songlyrics.com, z odpowiednimi parametrami wyszukiwania:



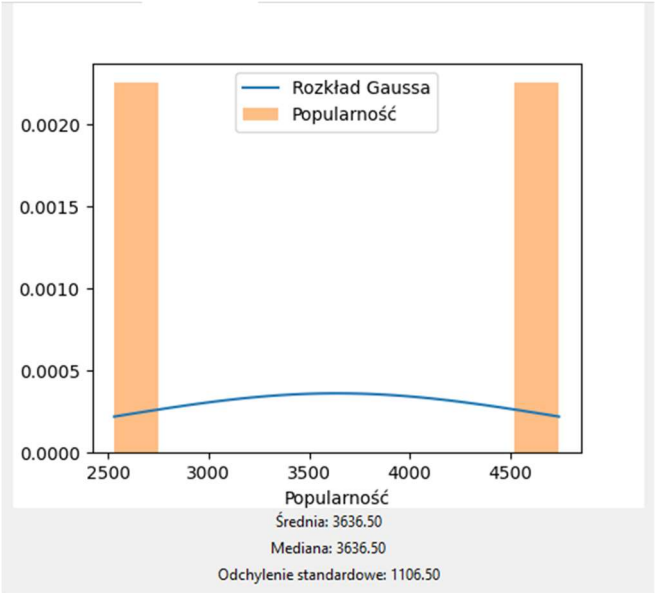
Po wybraniu opcji do importowania danych, wyświetla się okno z możliwością wybrania pliku CSV:



Kolejne okno pozwala na wyszukiwanie informacji na podstawie podanych parametrów:



Ostatnia zakładka wyświetla podstawowe statystyki utworów pasujących do kryteriów wyszukiwania:



Model danych

W przypadku bazy PostgreSQL, wszystkie informacje składowane są w 3 tabelach powiązanych odpowiednimi kluczami (numerycznymi):

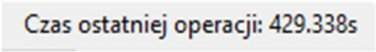
- songs
- genres
- artists

W przypadku baz nierelacyjnych, wszystkie dane importowane są w postaci słownika i nie są rozwiązywane na relacje. Program nie korzysta z ORM, a wszystkie operacje wykonywane są na prostym słowniku:

```
yield {
    "id": song_id,
    "title": title,
    "genre": {
        "id": genre_id,
        "name": genre
    },
    "artist": {
        "id": artist_id,
        "name": artist
    },
    "year": int(year),
    "views": int(views),
    "lyrics": lyrics,
    "lang_cld3": lang_cld3,
    "lang_ft": lang_ft,
    "language": lang
}
```

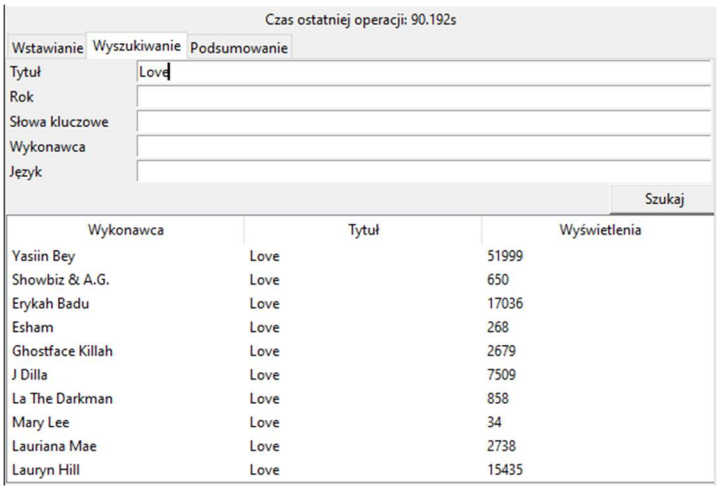
Eksperymenty

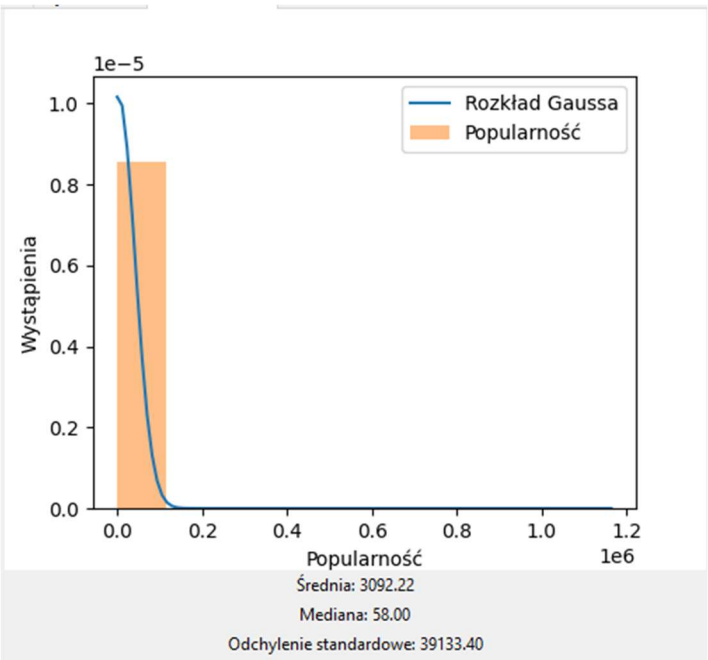
Z badań wynika, że najszybciej działa baza danych MongoDB. Import prawie 8 mln piosenek trwał ok. 7 minut:



W przypadku PostgreSQL, czas ten wyniósł aż 2 godziny, a Elasticsearch poradził sobie w 31 minut.

W przypadku elasticsearch, czas dodawania danych wzrasta w miarę upływu czasu. Najgorzej sprawdza się postgres. W przypadku wyszukiwania piosenek o tytule „Love”, również najszybciej działa MongoDB:





Wygenerowany wykres świadczy o tym, że występuje bardzo dużo piosenek o małej popularności.

Po skończonym imporcie, wszystkie z przedstawionych baz danych radzą sobie z dodawaniem nowych obiektów. Trwa to ok. 60 ms w przypadku MongoDB, 100 ms w przypadku PostgreSQL i 75 ms w przypadku Elasticsearch.