

Politechnika Śląska

Wydział Automatyki, Elektroniki i Informatyki

Kierunek Informatyka

Praca dyplomowa magisterska

Projekt i realizacja dydaktycznego systemu

wspomagającego projektowanie zapytań SQL

zawierających elementy rozmyte

Autor: Krzysztof Miemiec

Kierujący pracą: Bożena Małysiak-Mrozek

Gliwice, wrzesień 2018

# Wprowadzenie

Celem niniejszej pracy dyplomowej jest realizacja nowego narzędzia dydaktycznego umożliwiającego tworzenie zapytań SQL zawierających elementy rozmyte z wykorzystaniem nowych technologii i zbadanie wydajności takiego rozwiązania. Nowe narzędzie ma bazować na operatorach rozmytych opracowanych przez dr inż. Bożenę Małysiak-Mrozek w rozprawie doktorskiej „Metody aproksymacyjnego wyszukiwania obiektów w bazach danych” oraz narzędzia dydaktycznego FuzzyQ, stworzonego w 2005r. przez mgr Bartosza Dziedzica w jego pracy dyplomowej. 13-letnia aplikacja, wykorzystująca połączenie technologii Flash oraz C#, skompilowana jako samodzielna aplikacja działająca w środowisku Windows, jest na dzień dzisiejszy przestarzała technologicznie. Nowe narzędzie ma wprowadzić funkcjonalność tworzenia zapytań z użyciem podpowiedzi kontekstowych, a także umożliwić automatyczne dodawanie operatorów do bazy. Ponadto, dzięki zastosowaniu wieloplatformowego środowiska Electron, ma być możliwe uruchomienie aplikacji na systemach Linux, Windows oraz macOS.

## Plan pracy

Praca składa się z dziesięciu rozdziałów. W rozdziale pierwszym opisany jest cel, motywacja i plan pracy dyplomowej. W rozdziale drugim są określone wymagania funkcjonalne systemu i usprawnienia względem FuzzyQ. Rozdział trzeci opisuje język SQL, na którym opiera się system dydaktyczny. W rozdziale czwartym omawiane jest zagadnienie logiki rozmytej i podstawowych operatorów w analogii do logiki boolowskiej. Rozdział piąty przybliża technologie wykorzystane do zbudowania systemu, uzasadniając ich wybór. W rozdziale szóstym opisana jest architektura całego systemu, składającego się z oddzielnie przygotowanych operatorów rozmytych i aplikacji do zarządzania bazami danych ze wsparciem zapytań rozmytych, działającej w środowisku Electron. Rozdział ósmy dotyczy rozwiązań technicznych, takich jak budowa komponentów aplikacji, komunikacja między wątkami, czy dodawanie operatorów do dowolnej bazy relacyjnej. W rozdziale dziewiątym jest opisany interfejs użytkownika. W rozdziale dziesiątym zaprezentowane są badania porównujące szybkość prezentacji danych w środowisku przeglądarkowym jakim jest Electron, a także różnice w wykorzystaniu zasobów między FuzzyQ, a nowym systemem. Rozdział jedenasty zawiera podsumowanie prac nad systemem i możliwości jego rozwoju.

# Analiza tematu

Obecną wersja systemu dydaktycznego FuzzyQ wyróżnia prosty interfejs, możliwość tworzenia zapytań do bazy i graficzny kreator zapytań. Aplikacja jest jednak przestarzała pod względem technologicznym, gdyż wykorzystuje porzucone środowisko Flash i nie zadziała bezpośrednio na systemie Ubuntu, wykorzystywanym w większości stanowisk laboratoryjnych. FuzzyQ ma także braki pod względem interfejsu, nie spełniając wysokich standardów dzisiejszych narzędzi programistycznych i trendów z zakresu *user experience* wyznaczanych przez firmy JetBrains, Microsoft, czy Google. Za przykład może posłużyć brak kolorowania składni, wsparcia dla wcięć i podpowiedzi kontekstowych w polu edycji zapytania. Nowy system dydaktyczny nie będzie zatem prostym przepisaniem istniejącego narzędzia na inny język programowania, lecz nową implementacją pomysłu. System, z nowymi funkcjonalnościami, zostanie zrealizowany w sposób otwarty na rozszerzanie i zgodny ze współczesnymi standardami projektowania interfejsów.

## Istniejące rozwiązania

Poza aplikacją FuzzyQ stworzoną na Politechnice Śląskiej, na Politechnice Poznańskiej w 2006 roku powstało narzędzie SQLf\_j oparte o bazę danych MySQL, dokonujące translacji z języka SQLf (SQL z rozszerzeniami rozmytymi) na SQL przy użyciu tabeli pomocniczych. Tabela „possibility” zawiera definicje funkcji trapezowych dla danych liczbowych, a tabela „similarity” opisuje podobieństwa między parami ciągów znaków. Obsługiwany jest również minimalny próg przynależności wyników i ograniczenie ich ilości. Jest to podejście, które pozwala na dodawanie definicji i zależności w prosty sposób, lecz całość wymaga korzystania z ograniczonego interfejsu konsolowego, który służy do tłumaczenia poleceń, pozwalającego jedynie na wykonanie instrukcji SELECT.

W sieci obecna jest również strona opisująca język FSQL (Fuzzy SQL) wraz z serwerem FSQL dla nieprecyzyjnej relacyjnej bazy danych (FSQL Server for a Fuzzy Relational Database) bazująca na serwerze Oracle. Rozwiązanie pozwala na definiowanie etykiet dla zakresów danych i definicję progu wyników oraz posiada standardowy zestaw komparatorów rozmytych.

Poza wymienionymi narzędziami nadającymi się wyłącznie do celów akademickich, nie ma rozpowszechnionych i nadających się do wykorzystania w praktyce systemów wyszukiwania z numerycznymi elementami rozmytymi. Istnieją natomiast przykłady pokazujące jak rozszerzyć bazę danych o odpowiednie funkcje i operatory, które można zastosować w medycynie czy ekonomii. Popularnością cieszą się natomiast funkcje do wyszukiwania rozmytego w ciągach znaków. PostgreSQL zawiera implementację metod ułatwiających użytkownikowi wyszukanie tekstu nawet, jeśli został on wpisany z błędem: obliczającą odległość Levenshteina między wyszukiwaną frazą a przeszukiwanymi danymi, *metaphone* – określającą przybliżony zapis fonetyczny. Za przykład może posłużyć również pełnotekstowy silnik wyszukiwania Elasticsearch, mogący przeszukiwać tekst licząc odległość Levenshteina.

## Wymagania

Podstawowym wymaganiem jest współczesny interfejs umożliwiający tworzenie zapytań SQL ze wsparciem dla elementów rozmytych. Przyjmując, że użytkownik systemu zna język SQL na poziomie podstawowym, w odróżnieniu od FuzzyQ, tworzony system nie będzie miał graficznej reprezentacji zapytań. Zamiast tego edytor zapytań SQL ma być wzbogacony o kolorowanie składni oraz podpowiedzi kontekstowe, takie jakie występują we wszystkich popularnych środowiskach programistycznych po naciśnięciu skrótu klawiszowego (np. IntelliJ, VS Code). Podpowiedzi kontekstowe mają zawierać zarówno standardowe elementy języka SQL, jak i rozszerzenie w postaci elementów rozmytych. System ma wspierać tworzenie połączeń z bazami danych i przechowywanie ich. Interfejs aplikacji ma umożliwiać wyświetlanie tabel bazy i swobodne przeglądanie wyników zapytań. System powinien działać na wszystkich popularnych systemach operacyjnych: Windows, Linux (Ubuntu) oraz macOS.

System ma posiadać zintegrowany zestaw operatorów, które będzie można dodać do dowolnej bazy. Operatory w systemie będą bazować na operatorach będących częścią rozprawy doktorskiej dr inż. Bożeny Małysiak-Mrozek, zostaną jednak zrefaktoryzowane i przystosowane do systemu tak, aby można było rozszerzyć o nie dowolną bazę danych. Atutem byłaby możliwość samodzielnego tworzenia nowych operatorów.

# Język SQL

Język SQL został zaproponowany przez Donalda Chamberlina i Raymonda Boyce’a dwa lata po tym, jak Edgar „Ted” Codd zaprezentował koncepcję relacyjnego modelu danych na sympozjum w 1972 roku. Twórcy SQLa zauważyli, że matematyczną notację w językach Codda – algebrze relacji (*relational algebra*) i rachunku relacji (*relational calculus*) – ciężko pojąć osobie bez wykształcenia matematycznego, a także jej zapis na klawiaturze będzie problematyczny. Tak powstał SEQUEL (*A Structured English Query Language*) - język pozwalający na tworzenie złożonych zapytań do relacyjnych baz danych w czytelny i prosty sposób, a także obejmujący operacje modyfikacji danych i administracji bazą. Po rozpowszechnieniu języka SQL wśród twórców oprogramowania konieczna była standaryzacji. Na przestrzeni lat powstało 9 kolejnych wersji standardu, dodające nowe typy danych, rodzaje dozwolonych operacji i zabezpieczenia. Najnowsza wersja standardu, SQL:2016, została wydana w grudniu 2016 roku, m.in. dodając wsparcie dla dokumentów w notacji JSON, formatowanie i parsowanie daty i czasu oraz rozpoznawanie wzorców w wierszach. Niestety, żadna z istniejących implementacji nie jest zgodna ze standardami opracowanymi 2 lata temu. Nie zawsze oznacza to brakujące funkcjonalności, lecz ich realizację przy użyciu składni niezgodnej ze standardem.

## Rodzaje zapytań

Z języka SQL można wydzielić kilka podzbiorów zajmujących się osobnymi rodzajami zadań.

### DQL – *Data Query Language* (z ang. język zapytań do danych); zawiera polecenie SELECT zwracające dane z podanej tabeli.

### DML – *Data Manipulation Language* (z ang. język manipulacji danymi); zawiera polecenia INSERT, UPDATE i DELETE, które odpowiednio wstawiają, modyfikują i usuwają dane z tabeli.

* DDL – *Data Definition Language* (z ang. język definiowania danych); składa się z poleceń CREATE, ALTER i DROP, służące do tworzenia, modyfikacji i usuwania tabel.
* DCL – *Data Control Language* (z ang. język kontroli nad danymi); obejmuje polecenia GRANT, DENY i REVOKE, pozwalające na nadanie uprawnień, zabronienie wykonania operacji lub odebranie uprawnień.
* TCL – *Transactional Control Language* (z ang. język kontroli nad transakcjami); składa się z poleceń COMMIT i ROLLBACK służących do zatwierdzania i wycofywania zmian z transakcji.

## Składnia

# Logika rozmyta

# Stos technologiczny

Do zbudowania systemu dydaktycznego postanowiono wykorzystać jedno z najpopularniejszych w ostatnich latach środowisk do tworzenia aplikacji wieloplatformowych – Electron. Aplikacja, oparta o frameworki React i Redux, zostanie napisana w języku TypeScript, który jest następnie tłumaczony do języka JavaScript. Całość jest pakowana do skompresowanej i zoptymalizowanej paczki z kodem przy użyciu narzędzia Webpack, a następnie zamknięta w pliku natywnej aplikacji (np. \*.exe dla Windows, \*.app dla macOS) poprzez wspomniane środowisko Electron. Uzasadnieniem powyższych wyborów jest uniwersalność, prostota działania, a także znajomość technologii przez autora pracy.

Poza samą aplikacją, system dydaktyczny ma zawierać również zestaw funkcji rozmytych stanowiących rozszerzenie dla baz danych. Podobnie jak w poprzednim rozwiązaniu, zostanie użyta baza Postgres, a funkcje pozostaną w języku C, jednak zostaną poddane refaktoryzacji.

## PostgreSQL

## JavaScript/ECMAScript

## TypeScript

## Webpack

## Electron

## React

## Redux

# Implementacja operatorów rozmytych

# Architektura systemu

# Specyfikacja wewnętrzna

# Specyfikacja zewnętrzna

# Badania wydajności

# Podsumowanie

otwarcie źródła, rozszerzalność, klient sql

# Źródła

SQLf - implementacja w Javie

http://calypso.cs.put.poznan.pl/projects/sqlf\_j/

http://calypso.cs.put.poznan.pl/projects/sqlf\_j2006/tutorial/

FSQL - Oracle (model GEFRED):

http://www.lcc.uma.es/~ppgg/FSQL.html

Grupowanie:

http://bdas.polsl.pl/BDAS%6017%20-%20Realizacja%20grupowania%20i%20agregacji%20danych%20w%20module%20translacji%20zapytań%20rozmytych%20na%20zapytania%20klasyczne.pdf?Id=575&val=3

więcej o GEFRED:

http://sci-hub.tw/https://doi.org/10.1016/0020-0255(94)90069-8

fuzzy (trapezoid):

http://sci-hub.tw/https://doi.org/10.1016/j.proeng.2011.11.2653

dokumentacja postgresa dot. fuzzy text search:

https://www.postgresql.org/docs/9.1/static/fuzzystrmatch.html

fuzzy queries - artykuł polskiego autorstwa:

http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.baztech-90ea0e0a-66bc-4424-8711-f35c1cb4859f

wyszukiwanie fonetyczne tekstu - metaphone:

http://www.informit.com/articles/article.aspx?p=1848528

fuzzy query w elasticu:

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-fuzzy-query.html>

historia sqla:

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6359709

sql 2016

https://modern-sql.com/blog/2017-06/whats-new-in-sql-2016