Bazy Danych

2. Zadanie domowe

Opracował: Maciej Penar

Spis treści

[1. Zadanie domowe 3](#_Toc6308329)

[Zależności funkcyjne – top-down 3](#_Toc6308330)

[ORM 4](#_Toc6308331)

[Proxy cache’ujące 5](#_Toc6308332)

[2. … i wykonać 5](#_Toc6308333)

[3. Na kiedy 5](#_Toc6308334)

# 1. Zadanie domowe

Wybrać dowolne zadanie i zrealizować:

## Zależności funkcyjne – top-down

Mała pomoc: [Link](http://mst.mimuw.edu.pl/lecture.php?lecture=bad&part=Ch6)

Zależność funkcyjna oznacza że dla pewnych wartości atrybutów (kolumn) tabeli, pewne inne atrybuty **zawsze** mają określoną wartość.

Zależność funkcyjną (ZF) zapisujemy jako: np. co oznacza że każdy Pesel jednoznacznie wyznacza imię.

Ogólnie rzecz biorąc ZF przyjmują postać:

Co oznacza że każda kombinacja kolumn wyznacza dokładnie wartości kolumn

ZF mają kilka cech:

1) Prawą stronę ZF możemy dekomponować:

2) **Lewej strony ZF nie wolno dekomponować**

3) Są przechodnie, tj.:

Jeśli wiemy, że i , to też

4) Są zwrotne: oraz

Jeśli w schemacie atrybut X (lub zbiór atrybutów) wyznacza pozostałe kolumny to ZF: , to jest nazywany **kluczem** (więzy UNIQUE).

Zazwyczaj w tabeli o schemacie wybierany jest **klucz główny** – tylko jeden.

Jedna z metodologii projektowania Baz Danych polega na utworzeniu A) schematu uniwersalnego czyli bazy danych z jedną tabelą oraz B) sformułowania zbioru ZF dzięki którym nastąpi dekompozycja. Na przykład:

A) **Baza Danych(PESEL, Imie, Nazwisko, Miasto, Ulica, Numer Domu, Kod-Pocztowy, Numer Zamówienia, Adres Wysyłki, Pozycja Paragonu, Nazwa Produku, Ilość na Paragonie, …. itp.)**

B) , , ,

Dekompozycja jest wykonywana w oparciu o pewne reguł – zbiory tych reguł nazywamy postaciami normalnymi. Jeśli wszystkie tabele spełniają postać normalną, to mówimy że baza danych jest w postaci normalnej.

Przykładowe postacie normalne:

1. 1PN – gdy wartości kolumn są atomowe (nie są kolekcjami) oraz istnieje klucz główny
2. 2PN – gdy kolumny niekluczowe są w pełni zależne funkcyjnie, czyli dla ZF: i kolumn kluczowych , nie istnieje ZF: , , ,
3. 3PN – gdy kolumny niekluczowe nie posiadają zależności przechodnich, czyli jeśli A jest kolumną klucza i ZF: , to albo:
   1. nie istnieje
   2. nie istnieje
   3. istnieje i (wszystkie kolumny są kluczami - tricky part)
4. BCNF – gdy kolumny niekluczowe nie posiadają zależności przechodnich, czyli jeśli A jest kolumną klucza **głównego** i ZF: , to nie istnieje i nie istnieje

Napisać program który na wejściu przyjmuje:

* zbiór atrybutów
* zbiór zależności funkcyjnych
* poziom normalizacji (1PN/2PN/3PN/BCNF)

I dzieli atrybuty na podzbiory (tabele) spełniające kryteria wg. wybranego poziomu normalizacji.

Jakieś przypadki testowe:

Dekompozycja do:

Dekompozycja do:

Dekompozycja do:

## ORM

Napisać prosty ORM – Code First - w Javie lub C# wykorzystujący:

* Atrybuty (C#)
* Adnotacje (Java)

ORM powinien umożliwiać:

* CREATE TABLE ze wsparciem dla kluczy obcych i ograniczeniami ON DELETE/ON UPDATE
* DROP TABLE
* INSERT
* UPDATE
* DELETE
* Wykonanie dowolnego SQL’a (RawSQL)

Jeśli chcecie zaimponować to rozwiązać którykolwiek z tych problemów:

* opracować sposób zmiany schematu: np. dodawania kolumn, usuwania kolumn
* dodać wsparcie do SELECT WHERE

## Proxy cache’ujące

Spiąć się z dowolnym ORM’em oraz:

* Backend
* Redis
* Bazę danych - dowolną

I napisać kawałek kodu w sposób **możliwie ogólny** dzięki któremu można wykonywać operacje odczytu na bazie danych które w pierwszej kolejności pobierane są z Redisa.

W momencie w którym użytkownik formułuje zapytanie np. **SELECT \* FROM tableX;** Wykonywany jest następujący algorytm:

1. Dla zapytania wejściowego biblioteka ustala klucz cache’owania oraz dopuszczalny termin przedawnienia
2. Bibliotek idzie z < do **Redisa** 
   1. Jeśli Redis posiada dane odpowiadające kluczowi spełniające to następuje zwrot danych
   2. Jeśli Redis nie posiada danych:
      1. Biblioteka wykonuje zapytanie na bazie danych
      2. Biblioteka umieszcza wynikowe w Redisie pod kluczem
      3. Biblioteka zwraca użytkownikowi dane

Wyobrażam sobie że taki framework byłby parametryzowany np.:

* Dopuszczalnym czasem nieświeżości danych
* Adresem Redisa
* Adresem BD

# 2. … i wykonać

Przygotować sprawozdanie zawierające:

1. Napisać kod, spakować do zip
2. Napisać dokumentację omawiającą problem wraz z omówieniem przypadków testowych

# 3. Na kiedy

**Wysłać na maila mpenar[at]kia.prz.edu.pl do 25 maja 2019.**