Projekt Bazy Danych:
Baza danych automatycznej sprawdzarki zadań programistycznych

Paweł Goliszewski & Karol Chomoncik

Spis Treści

1. Podstawowe założenia projektu

Temat: Baza danych automatycznej sprawdzarki zadań programistycznych.
Autorzy: Paweł Goliszewski, Karol Chomoncik.
Cel: Stworzenie bazy danych zdolnej do wykorzystania jako baza danych automatycznej sprawdzarki zadań programistycznych. W szczególności zdolnej do obsługiwania automatycznej sprawdzarki BACA.
Główne założenia: Baza danych stworzona według norm i reguł przyjętych przez informatyczny świat, działająca w możliwie najbardziej wydajnościowy sposób.
Fazy projektu:
Faza 1: Ustalenie wstępnego wyglądu bazy danych i zasad jej działania.
Faza 2: Pierwsze próby utworzenia schematów i diagramów.
Faza 3: Ustalenie ostatecznego wyglądu i zasad działania bazy (drobne poprawki nadal mogą wystąpić podczas późniejszych faz).
Faza 3: Utworzenie diagramów i skryptu w oparciu o program Oracle SQL Developer Data Modeler w wersji 17.4.0.
Faza 4: Utworzenie Bazy Danych w oparciu o program Microsoft SQL Server Management Studio 17.4.
Faza 5a: Tworzenie oprogramowania do bazy danych oraz wprowadzenie przykładowych danych w celach testowych.
Faza 5b: Tworzenie programu klienckiego dla zwykłych użytkowników, a także dla administratorów.
Faza 6: Drobne prace kosmetyczne i naprawcze.
Faza 7: Oddanie projektu, a także jego obrona.
Faza 8: Dalszy rozwój projektu.
Możliwości i ograniczenia: Głównym ograniczeniem podjętym podczas projektu jest ograniczenie czasowe. Wymusza ono wykonanie faz: 1, 2, 3, 4, 5a, 6 oraz 7 w terminie nie przekraczającym końca semestru (koniec stycznia/początek lutego 2018 roku). Faza 5b jest dodatkowa i podjęta zostanie jedynie gdy fazy 1, 2, 3, 4 oraz 5a zostaną ukończone odpowiednio wcześnie. Faza 8 jest niezależna od wyżej wspomnianego terminu i będzie kontynuowana w kolejnych miesiącach.
Baza danych będzie unikała redundancji danych, jednak w niektórych przypadkach w celu osiągnięcia lepszej wydajności, część danych ulegnie denormalizacji. Będą to jednak jednostkowe przypadki i ogólnie całość będzie można określić mianem znormalizowanej bazy danych.

2. Diagram ER

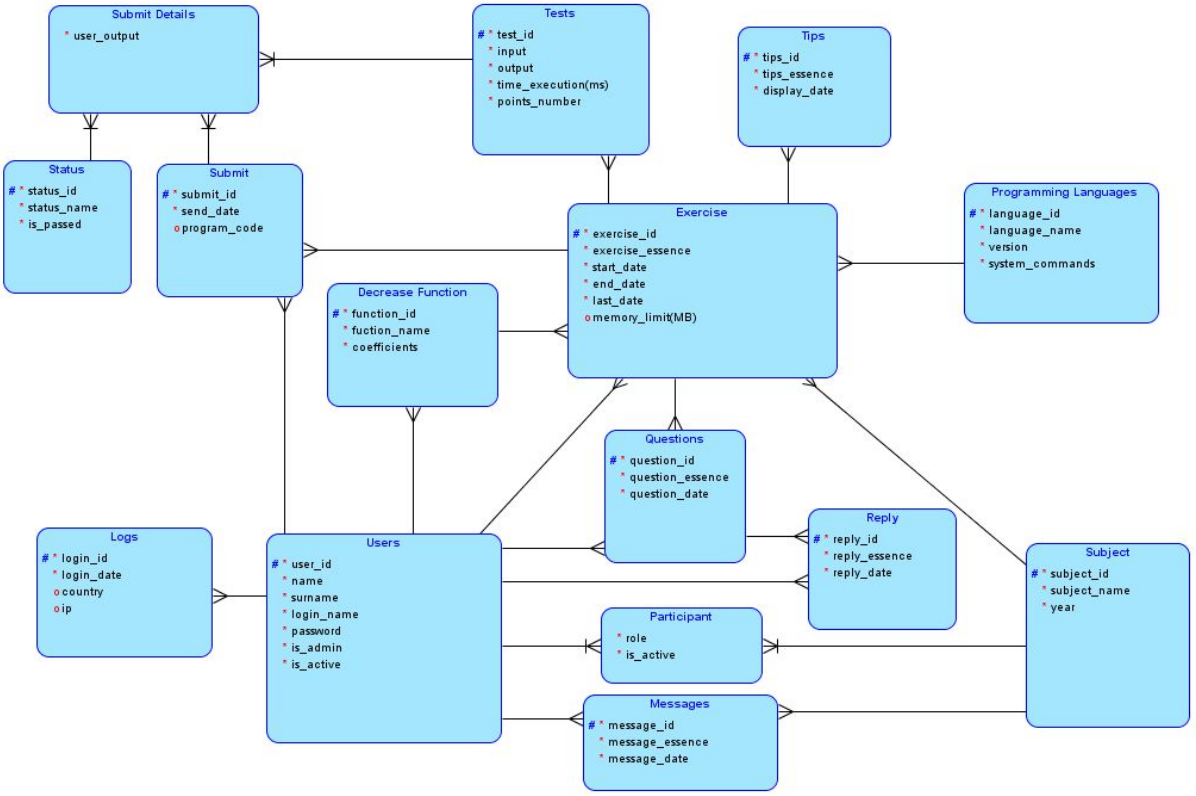

Diagram związków encji wygenerowany za pomocą programu Data Modeler w wersji 17.4.0.

3. Diagram relacji

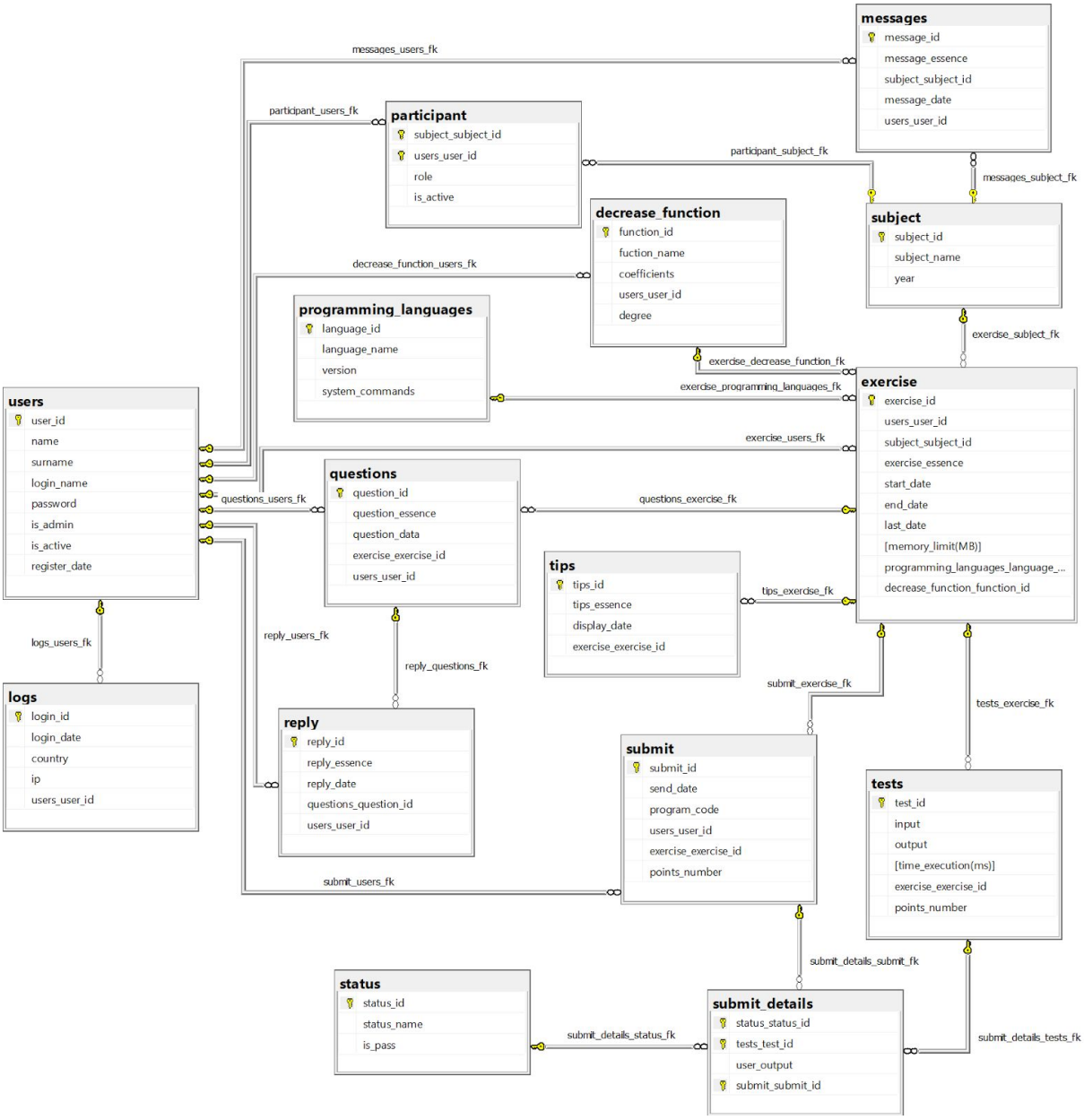

Diagram relacji wraz z zaznaczonymi kluczami obcymi wygenerowany za pomocą programu Microsoft SQL Server Management Studio w wersji 17.

4. Opis Tabel

Legenda:
Primary Key: Klucz główny,
Foreign Key: Klucz obcy.

Tabele:

1. users (user_id, name, surname, login_name, password, is_admin, is_active, register_date).

   Tabela users przechowuje dane na temat zarejestrowanych użytkowników. Jej kluczem głównym jest pole user_id, czyli unikalny numer nadawany kolejnym zarejestrowanym osobom, rozpoczynając od numeru 1. Ponadto jest kluczem obcym dla tabel participant, exercise, decrease_function, submit, messages, questions, reply i logs.

2. subject (subject_id, subject_name, year).

   Tabela subject przechowuje dane na temat dostępnych przedmiotów. W związku z tym, że poszczególne przedmioty mogą powtarzać się, posiadają one również informację o roku, w którym się rozpoczęły. Jako okres aktywności przedmiotu przyjmujemy 1 października wpisanego roku do 15 września roku następnego. Kluczem głównym jest pole subject_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Ponadto jest kluczem obcym dla tabel participant, messages oraz exercise.

3. participant (subject_id, user_id, role, is_active).

   Tabela participant przechowuje dane na temat osób biorących udział w danym przedmiocie i ich roli (moderator lub uczestnik). Posiada dwa klucze obce subject_subject_id oraz users_users_id, które razem tworzą klucz główny. Pole role określa czy uzytkownik jest zwykłym uczestnikiem, czy też adminem o zwiększonych uprawnieniach. Ostatnie pole is_active pozwala prowadzącemu kurs na dezaktywację studentów, którzy zrezygnowali w ciągu roku.

4. exercise (exercise_id, user_id, subject_id, exercise_essence, start_date, end_date, last_date, memory_limit(MB), language_id, function_id).

Tabela exercise jest najbardziej rozbudowaną tabelą w całym projekcie. Przechowuje ona dane na temat wszystkich zadań opublikowanych na Automatycznej Sprawdzarce bez podziału na konkretny przedmiot. Pole users_user_id to informacja o autorze zadania.Tabela posiada cztery klucze obce programming_languages_language_id, decrease_function_function_id, users_user_id, subject_subject_id. Ponadto jest kluczem obcym dla tabel questions, tips, submit oraz tests. Jej kluczem głównym jest pole exercise_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1.

5. programming_languages (language_id, language_name, version, system_commands).

   Tabela programming_languages przechowuje informacje na temat dostępnych języków, a także ich wersji oraz komendach wykorzystywanych przez przyszłą aplikację webową. Jej kluczem głównym jest pole language_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Ponadto jest kluczem obcym dla tabeli exercise.

6. decrease_function (function_id, function_name, user_id, degree, coefficients).

   Tabela decrease_function przechowuje informacje na temat dostępnych funkcji spadkowych, w jakich spada możliwa do zdobycia liczba punktów za dane zadanie. Jej kluczem głównym jest pole function_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Dostępne są funkcje wielomianowe określone przez pole degree oraz coefficients. Jej kluczem obcym jest pole users_user_id, czyli autor danej funkcji. Ponadto jest ona kluczem obcym dla tabeli exercise.

7. tips (tips_id, tips_essence, display_date, exercise_id).

   Tabela tips zawiera informacje o wskazówkach, które mogą pojawiać się dla danego zadania, w określonym przez pole display_date czasie. Jej kluczem głównym jest pole tips_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Posiada klucz obcy exercise_exercise_id.

8. tests (test_id, exercise_id, input, output, points_number, time_execution(ms)).

   Tabela tests zawiera informacje o kolejnych testach do zadań. Jej kluczem głównym jest pole test_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Posiada klucz obcy exercise_exercise_id.

9. submit (submit_id, send_date, program_code, user_id, exercise_id, points_number).

   Tabela submit zawiera informacje o wszystkich nadesłanych przez użytkowników submitach. Jest to tabela, która będzie zawierała w sobie najwięcej danych. Jej kluczem głównym jest pole submit_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Posiada dwa klucze obce users_user_id oraz exercise_exercis_id.

10. submit_details (status_id, test_id, submit_id, user_output).

    Jest to tabela, która pozwala na sprawdzenie poprawności nadesłanego submitu. Przechowuje wyjścia programu, które uzyskano w poszczególnych testach. Tabela posiada trzy klucze obce status_status_id, tests_test_id oraz submit_submit_id, które ponadto tworzą razem klucz główny.

11. status (status_id, status_name, is_pass).

    Tabela status przechowuje informacje na temat statusów, które mogą osiągnąć testy danych submitów. Pole is_pass informuje o tym czy status oznacza, że program przeszedł dany test. Jej kluczem głównym jest pole status_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Jest kluczem obcym dla tabeli submit_details.

12. messages (message_id, message_essence, message_date, subject_id, user_id).

    Tabela messages zawiera informacje, które pojawiać się będą na głównej stronie konkretnego przedmiotu, przede wszystkim komunikaty od prowadzącego. Jej kluczem głównym jest pole message_id, czyli unikalny

numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Posiada dwa klucze obce subject_subject_id oraz users_users_id.

13. questions (question_id, question_essence, question_date, exercise_id, user_id).

    Tabela questions dotyczy konkretnego zadania tylko podczas jego trwania. Przechowuje pytania zadane przez użytkowników autorowi. Jej kluczem głównym jest pole question_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Jej klucze obce to users_user_id oraz exercis_exercise_id. Ponadto jest kluczem obcym dla tabeli reply.

14. reply (reply_id, reply_essence, reply_date, question_id, user_id).

    Tabela reply przechowuje informacje na temat odpowiedzi do zadanego pytania, czyli tworzy wątek danego pytania. Jej kluczem głównym jest pole reply_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Jej klucze obce to questions_question_id oraz users_user_id.

15. logs (login_id, login_date, country, ip, user_id).

    Tabela logs zawiera informacje na temat logowań poszczególnych użytkowników. Jej kluczem głównym jest pole login_id, czyli unikalny numer nadawany kolejnym dodanym przedmiotom, rozpoczynając od numeru 1. Jedynym kluczem obcym jest pole users_user_id.

5. Skrypt tworzący bazę danych

Skrypt znajduje się w pliku BACA.sql.

6. Przykładowe dane

Przykładowe dane znajdują się w pliku przykladoweDane.sql.

7. Opis stworzonych widoków

1. Admins:
   Wyświetla listę wszystkich adminów platformy.

2. ActiveSubjects:
   Wyświetla listę wszystkich aktywnych przedmiotów. Przedmiot aktywny jest od 1 października podanego roku, do 10 września kolejnego roku.

3. Exercises:
   Wyświetla listę wszystkich zadań wraz z liczbą punktów możliwych do zdobycia, za wykonanie danego zadania. Liczba punktów, to suma punktów ze wszystkich testów danego zadania.

4. Exercises_Point_Average:
   Wyświetla listę zadań wraz ze średnią liczbą punktów. Średnia liczona jest z najlepszego wyniku każdego studenta.

5. LastLogs:
   Wyświetla datę ostatniego logowania dla każdego uczestnika.

Kody widoków znajdują się w pliku: BACAviews.sql

8. Opis stworzonych procedur

1. countPoints @submitId INT:
   Procedura wylicza ilość punktów za submit na podstawie funkcji spadkowej jeśli został przekroczony czas i zmienia pole w tabeli submit, jeśli zmieniła się liczba punktów.

2. extendExerciseDate @exerciseId INT, @timeH INT:
   Procedura wydłuża czas zadania o podaną liczbę godzin, zmieniając pola end_date oraz last_date.

3. changePoints @submitID INT, @newPoints INT:
   Procedura zmienia liczbę punktów uzyskanych podany submit na liczbę podaną przez prowadzącego. Możliwe na przykład po nieudanej obronie programu.

4. rejectSubmit @submitID INT
   Procedura zmienia status submitu o podanym ID na “ręcznie odrzucony”, a także zmienia liczbę punktów uzyskaną za dany submit na 0.

Kody stworzonych procedur znajdują się w pliku: BACAprocedures.sql

9. Opis stworzonych funkcji

1. userActiveSubjects (@userID INT)
   Wyświetla aktywne przedmioty użytkownika o podanym userID.

2. userPoints (@userID INT, @subjectID INT)
   Wyświetla wszystkie zadania z przedmiotu subjectID użytkownika o podanym userID wraz z uzyskaną przez niego liczbą punktów za każde zadanie.

3. subjectRanking (@subjectID INT)
   Wyświetla listę użytkowników danego przedmiotu (tych, którzy zrobili jakieś zadanie) wraz z liczbą punktów uzyskanych przez nich.

4. getPolynomialCoeffs (@id int)
   Funkcja, która zwraca tabelę współczynników wielomianu (coeff) wraz z ich stopniem (oznaczonym jako id).

5. countPolynomialVal (@coeffs tableCoeffs READONLY, @x FLOAT)
   Funkcja, która zwraca wartość wielomianu dla danego argumentu @x o podanych współczynnikach @coeffs.

6. countPolynomialValById (@id INT, @x FLOAT)
   Funkcja, która zwraca wartość wielomianu od id @id dla danego argumentu @x.

7. userLogin (@userId INT, @password VARCHAR(20))
   Funkcja, która zwraca 1, gdy użytkownik o podanym @id może się zalogować podanym hasłem. W przeciwnym wypadku zwraca 0.

8. showTopic (@questionID INT)
   Funkcja, która wyświetla pytanie o id @questionID wraz z odpowiedziami. Zawiera dodatkową kolumnę type, która przyjmuje wartość q dla pytania, a r dla odpowiedzi.

9. userEndSubjects (@userID INT)
   Wyświetla zakończone przedmioty użytkownika o podanym id @userID.

10. subjectExercises (@subjectID INT)
    Wyświetla wszystkie zadania przedmiotu o podanym @subjectID.

11. submitNumber (@exerciseID INT, @userID INT)
    Funkcja, która zwraca ilość wysłanych rozwiązań przez użytkownika o podanym @userID do zadania o danym @exerciseID.

Kody widoków znajdują się w pliku: BACAfunctons.sql

10. Opis stworzonych wyzwalaczy

1. Tr1 na tabeli tips - zapewnia, że data wyświetlenia wskazówki jest późniejsza niż data początku obowiązywania zadania, ale wcześniejsza niż data końca zadania.

2. Tr2 na tabeli submits - zapewnia, że data wysłania kodu przez użytkownika jest późniejsza niż data początku obowiązywania zadania, ale wcześniejsza niż data końca zadania.

3. Tr3 na tabeli messages - zapewnia, że komunikaty mogą być dodawane tylko w czasie aktywności przedmiotu (od 1 października roku aktywności przedmiotu do 15 września roku następnego).

4. Tr4 na tabeli questions - zapewnia, że pytania mogą być dodawane tylko w czasie aktywności zadania.

5. Tr5 na tabeli reply - zapewnia, że daty odpowiedzi do pytań mogą być z zakresu od daty zadania pytania do końca aktywności zadania.

6. Tr6 oraz Tr6a na tabeli users - zapewniają, że zawsze istnieje co najmniej jeden administrator (nie można usunąć, dezaktywować, ani odebrać uprawnień ostatniemu administratorowi).

7. Tr7 oraz Tr7a na tabeli participants - zapewniają, że nie można dezaktywować (w zakresie przedmiotu), odebrać uprawnień, ani usunąć uczestnictwo jedynemu moderatorowi przedmiotu.

8. Tr8 na tabeli users - zapewnia szyfrowanie hasła przy rejestracji użytkowników.

9. Tr9 na tabeli decrease_function - zapewnia, że wielomian spadku jest poprawny, tj. dla argumentu 0 daje wartość 1, a dla argumentu 1 wartość 0.

Kody widoków znajdują się w pliku: BACAtriggers.sql

11. Strategia wykonywania kopii zapasowych

Ze względu na ilość potencjalnych użytkowników i zależność wrzucanych danych od oceny użytkowników szacujemy częstość wykonywania kopii zapasowych na 1 dziennie.