ФОРМА 1

0 Общие дизайнерские элементы

Логотип

Надпись Рейтенговая Система МГКЭИТ

Меню

1 ВКЛАДКИ

Общий рейтинг колледже – показывает рейтиг учащихся между колледжей

Рейтинг колледжа 1

Рейтинг колледжа 2

Рейтинг колледжа 3

Кнопки

Кнопка “Поиск” При нажатие открывается форма для Поиска учащегося\*

Кнопка “Добавить ” При нажатие открывается форма для Добавление рейтенга учащегося\*

Кнопка “Редактировать” При нажатие открывается форма для Редактирования рейтинга учащегося \*

Кнопка “Список Изменений” При нажатие открывается форма Показывающая последние Добавленя и Редактирования Рейтинга

Кнопка Выйти

Кнопка поиска

Кнопка редактирования . Открывает форму редактирования

Сортировка Кнопки

Кнопка для Сортировки по алфавиту ФИО

Кнопка для Сортировки Групп по году (две цыфры в конце)

Кнопка для Сортировки По колледжам

Кнопка для Сортировки по колличеству Рейтинга от 0 до 100 и наоборот

Форма 2 Для поиска

Список для отоброжения результатов поиска

Столбик для отображения рейтинга с кнопкой сортировки по реитинга от самого большого до маленького и наоборот

Столбик для отображения ФИО с кнопкой сортировки по алфавиту А до Я И Я до А

Столбик для отображения Групп с кнопкой сортировки по году поступления (по двум последним цыфрам от 0 до 100 и наоборот)

Столбик для отображения Колледжа с кнопкой сортировки от 1 до 3

Столбик для отображения Даты рождения с кнопкой сортировки от 0 до 100

Блок для поиска

Строка для ввода ФИО с Всплываюшей подсказкой о ФИО

Строка для ввода Группы со списком для удобства

Строка для ввода Колледжа со списком для удобства

Строка для ввода Даты рождения с кнопкой для открытия каллендаря и выбора даты

Строка для ввода Рейтинга

Кнопка для поиска

Форма 3 для редактирования

3.1 Лого

3.2 Надпись : Форма редактирования студента (имя и фамилия студента редактирования)

3.3 Поле ID

3.4 Поле ФИО

3.5 Поле Группы

3.6 Поле Дата Рождения

3.7 Поле Колледжа

3.8 Поле Рейтинга

3.9 Поле ID которое можно редактировать

3.10 Поле ФИО которое можно редактировать

3.11 Поле группы которое можно редактировать

3.12 Поле даты рождения которое можно редактировать

3.13 Поле коледж в котором можно выбрать заготовленый вариант колледжа

3.14 Поле рейтинга которое можно редактировать

3.15 Поле с надписью Дополнительная информация

3.16 Поле в котором можно написать информацию о студенте

3.17 Кнопка для сохранения отредактированной информации

3.18 Список с наименованиями коледжей

3.19 Кнопка календаря для удобного выбора даты

Форма Список изменений

-- Таблица для студентов

CREATE

);

-- Таблица для оценок

CREATE TABLE Grades (

ID SERIAL PRIMARY KEY,

Student\_ID INT NOT NULL,

Предмет VARCHAR(100) NOT NULL,

Оценка FLOAT NOT NULL,

Дата DATE NOT NULL,

);

-- Таблица для изменений рейтинга

CREATE TABLE RaitingChanges (

ChangeID SERIAL PRIMARY KEY,

StudentID INT NOT NULL,

OldRating DECIMAL(5,2) NOT NULL,

NewRating DECIMAL(5,2) NOT NULL,

ChangeDate TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

Reason VARCHAR(255),

);

-- Таблица для пользователей

CREATE TABLE Users (

UserID SERIAL PRIMARY KEY,

Username VARCHAR(50) UNIQUE NOT NULL,

PasswordHash VARCHAR(255) NOT NULL,

FullName VARCHAR(100) NOT NULL,

Role VARCHAR(20) CHECK (Role IN ('admin', 'teacher', 'student')) NOT NULL,

CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE Changes (

id SERIAL PRIMARY KEY,

parameter VARCHAR(255) NOT NULL,

old\_value TEXT,

new\_value TEXT,

change\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

user\_id INT

);

CREATE TABLE Competitions (

competition\_id SERIAL PRIMARY KEY,

competition\_name VARCHAR(255) NOT NULL,

competition\_date DATE NOT NULL

);

CREATE TABLE CompetitionParticipants (

participant\_id SERIAL PRIMARY KEY,

competition\_id INT REFERENCES Competitions(competition\_id),

student\_id INT REFERENCES Student(student\_id),

placement INT,

score DECIMAL(5, 2)

);мне нужно чтобы после нажатия кнопки в datagridview отображались данные из таблицы

CREATE OR REPLACE FUNCTION UpdateStudentRatings()

RETURNS VOID AS $$

DECLARE

student\_record RECORD;

total\_rating FLOAT;

average\_grade FLOAT;

BEGIN

FOR student\_record IN SELECT Student\_ID FROM Grades GROUP BY Student\_ID LOOP

-- Вычисление среднего значения оценок

SELECT AVG(Оценка) INTO average\_grade FROM Grades WHERE Student\_ID = student\_record.Student\_ID;

-- Определение рейтинга на основе среднего значения оценок

IF average\_grade IS NOT NULL THEN

total\_rating :=

CASE

WHEN average\_grade = 2 THEN -10

WHEN average\_grade = 3 THEN 0

WHEN average\_grade = 4 THEN 5

WHEN average\_grade = 5 THEN 10

ELSE 0

END;

-- Обновление таблицы Ratings

UPDATE Ratings

SET AverageGrade = average\_grade, Rating = total\_rating, CalculationDate = CURRENT\_TIMESTAMP

WHERE Student\_ID = student\_record.Student\_ID;

END IF;

END LOOP;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION update\_rating\_on\_new\_grade()

RETURNS TRIGGER AS $$

BEGIN

-- Обновление рейтинга на основе новой оценки

UPDATE Ratings

SET Rating = Rating + CASE

WHEN NEW.Оценка = 2 THEN -10

WHEN NEW.Оценка = 3 THEN 0

WHEN NEW.Оценка = 4 THEN 5

WHEN NEW.Оценка = 5 THEN 10

ELSE 0 -- Для других оценок рейтинг не изменяется

END

WHERE Ratings.Student\_ID = NEW.Student\_ID;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER after\_grade\_insert

AFTER INSERT ON Grades -- Предполагается, что у вас есть таблица Grades

FOR EACH ROW

EXECUTE FUNCTION update\_rating\_on\_new\_grade();

-- Создание функции триггера

CREATE OR REPLACE FUNCTION create\_rating\_trigger()

RETURNS TRIGGER AS $$

BEGIN

INSERT INTO ratings (student\_id, averagegrade, rating, calculationdate)

VALUES (NEW.id, NEW.averagegrade, NEW.rating, NOW());

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Создание триггера

CREATE TRIGGER student\_rating\_trigger

AFTER INSERT ON students

FOR EACH ROW

EXECUTE FUNCTION create\_rating\_trigger();

Создание функции триггера:

sql

RunCopy code

CREATE OR REPLACE FUNCTION update\_student\_rating()

RETURNS TRIGGER AS $$

BEGIN

UPDATE students

SET rating = rating + NEW.score

WHERE id = NEW.student\_id;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

Создание триггера:

sql

RunCopy code

CREATE TRIGGER competition\_participant\_insert

AFTER INSERT ON competitionparticipants

FOR EACH ROW

EXECUTE FUNCTION update\_student\_rating();

Объяснение

Функция update\_student\_rating:

Это функция, которая будет выполняться при срабатывании триггера.

Она обновляет таблицу students, прибавляя NEW.score к полю rating у записи, где id соответствует NEW.student\_id.

Триггер competition\_participant\_insert:

Срабатывает после вставки (AFTER INSERT) новой строки в таблицу competitionparticipants.

Для каждой новой строки (FOR EACH ROW)