Nazwa projektu: **SQL\_Projekt**  
Nazwisko i imię studenta: **Kuchnia Jan**

Opis projektu:

Mój projekt o nazwie SQL\_Projekt jest bazą danych, która przechowuje istotne informacje dotyczące uczniów, nauczycieli, przedmiotów, ocen i klas w szkole podstawowej. Dzięki tym tabelom możliwe jest zarządzanie danymi szkolnymi, umożliwiając rejestrowanie uczniów, przypisywanie ich do odpowiednich klas, śledzenie ocen z różnych przedmiotów oraz monitorowanie nauczycieli zatrudnionych w szkole. Moja baza danych w teorii (ponieważ zdaje sobie sprawę z ograniczeń tego projektu) pozwala na efektywne gromadzenie i analizowanie danych, co może przyczynić się do lepszej organizacji i funkcjonowania szkoły.

CREATE DATABASE SQL\_Projekt

USE SQL\_Projekt

--Tabela z podstawowymi danymi uczniów

CREATE TABLE Uczniowie (

ID INT IDENTITY(1,1) PRIMARY KEY,

Imię NVARCHAR(50) NOT NULL,

Nazwisko NVARCHAR(50) NOT NULL,

Płeć CHAR(1) CHECK (Płeć IN ('k', 'm')),

Data\_Urodzenia DATE NOT NULL,

Miejsce\_Urodzenia NVARCHAR(100) NOT NULL,

Obywatelstwo NVARCHAR(50) CHECK (Obywatelstwo IN ('polskie', 'inne')),

PESEL NVARCHAR(11),

Adres NVARCHAR(255) NOT NULL,

CHECK (Data\_Urodzenia < GETDATE())

)

--Tabela z podstawowymi danymi nauczycieli

CREATE TABLE Nauczyciele (

ID INT IDENTITY(1,1) PRIMARY KEY,

Imię NVARCHAR(50) NOT NULL,

Nazwisko NVARCHAR(50) NOT NULL,

Płeć CHAR(1) CHECK (Płeć IN ('k', 'm')),

Data\_Zatrudnienia DATE NOT NULL,

Data\_Urodzenia DATE NOT NULL,

Miejsce\_Urodzenia NVARCHAR(100) NOT NULL,

Obywatelstwo NVARCHAR(50) CHECK (Obywatelstwo IN ('polskie', 'inne')),

PESEL NVARCHAR(11) UNIQUE,

Adres NVARCHAR(255) NOT NULL,

CHECK (Data\_Zatrudnienia < GETDATE()),

CHECK (Data\_Urodzenia < GETDATE())

)

--Tabela z nazwami przedmiotów i ich poziomem

CREATE TABLE Przedmioty (

ID INT IDENTITY(1,1) PRIMARY KEY,

Nazwa\_Przedmiotu NVARCHAR(100) NOT NULL,

Poziom\_Klasy INT NOT NULL CHECK (Poziom\_Klasy BETWEEN 1 AND 8)

)

--Tabela z ocenami uczniów

CREATE TABLE Oceny (

ID INT IDENTITY(1,1) PRIMARY KEY,

Przedmiot\_ID INT NOT NULL,

Uczen\_ID INT NOT NULL,

Nauczyciel\_ID INT NOT NULL,

Ocena INT NOT NULL CHECK (Ocena BETWEEN 1 AND 6),

Data\_Oceny DATE NOT NULL,

FOREIGN KEY (Przedmiot\_ID) REFERENCES Przedmioty(ID),

FOREIGN KEY (Uczen\_ID) REFERENCES Uczniowie(ID),

FOREIGN KEY (Nauczyciel\_ID) REFERENCES Nauczyciele(ID)

)

--Tabela przechowująca dane o klasach

CREATE TABLE Klasy (

ID INT IDENTITY(1,1) PRIMARY KEY,

Klasa INT NOT NULL CHECK (Klasa BETWEEN 1 AND 8),

Rok\_Szkolny INT NOT NULL,

CHECK (Rok\_Szkolny <= YEAR(GETDATE()))

)  
  
--Tabela z przypisanymi uczniami do klas

CREATE TABLE Klasa\_Uczen (

Klasa\_ID INT NOT NULL,

Uczen\_ID INT NOT NULL,

PRIMARY KEY (Klasa\_ID, Uczen\_ID),

FOREIGN KEY (Klasa\_ID) REFERENCES Klasy(ID),

FOREIGN KEY (Uczen\_ID) REFERENCES Uczniowie(ID)

)

--Tabela zawierająca informację, jaki nauczyciel uczy danego przedmiotu

CREATE TABLE Nauczyciel\_Przedmiot (

Nauczyciel\_ID INT NOT NULL,

Przedmiot NVARCHAR(50) NOT NULL,

PRIMARY KEY (Nauczyciel\_ID, Przedmiot),

FOREIGN KEY (Nauczyciel\_ID) REFERENCES Nauczyciele(ID)

)

A screenshot of a computer

Description automatically generated with medium confidence

--Widoki  
  
--Widok TopStudents zwraca podstawowe dane 3 najlepszych uczniów

CREATE VIEW TopStudents AS

SELECT TOP 3

Oceny.Uczen\_ID,

Uczniowie.Imię,

Uczniowie.Nazwisko,

Klasy.Klasa,

AVG(Ocena) AS Średnia\_Ocen

FROM Oceny

JOIN Uczniowie ON Uczniowie.ID = Oceny.Uczen\_ID

JOIN Klasa\_Uczen ON Klasa\_Uczen.Uczen\_ID = Uczniowie.ID

JOIN Klasy ON Klasy.ID = Klasa\_Uczen.Klasa\_ID

GROUP BY Oceny.Uczen\_ID, Uczniowie.Imię, Uczniowie.Nazwisko, Klasy.Klasa

ORDER BY Średnia\_Ocen DESC

--Widok TopStubjects zwraca dane przedmiotów innych niż WF i muzyka ze średnią wszystkich ocen powyżej 4.5

CREATE VIEW TopSubjects AS

SELECT

Przedmioty.Nazwa\_Przedmiotu,

AVG(Oceny.Ocena) AS Średnia\_Ocen

FROM Oceny

JOIN Przedmioty ON Przedmioty.ID = Oceny.Przedmiot\_ID

GROUP BY Przedmioty.Nazwa\_Przedmiotu

HAVING AVG(Oceny.Ocena) > 4.5 AND Przedmioty.Nazwa\_Przedmiotu NOT IN ('WF', 'Muzyka')

--Wyzwalacze

--Wyzwalacz tr\_check\_pesel wyświetla błąd jeżeli nie podano PESELu, gdy obywatelstwo zostało zadeklarowane jako polskie i wycofuje transakcję

CREATE TRIGGER tr\_check\_pesel ON Uczniowie

AFTER INSERT

AS

BEGIN

DECLARE @obywatelstwo NVARCHAR(50);

DECLARE @pesel NVARCHAR(11);

SELECT @obywatelstwo = Obywatelstwo, @pesel = PESEL FROM inserted;

IF @obywatelstwo = 'polskie' AND @pesel IS NULL

BEGIN

RAISERROR('Obywatele polscy muszą podać swój PESEL', 16, 1);

ROLLBACK TRANSACTION;

RETURN

END

END

GO

--Wyzwalacz tr\_check\_if\_pesel\_unique wyświetla błąd, jeżeli osoba o danym PESELu jest już w tabeli i wycofuje transakcję

CREATE OR ALTER TRIGGER tr\_check\_if\_pesel\_unique ON Uczniowie

AFTER INSERT

AS

BEGIN

DECLARE @pesel NVARCHAR(11);

DECLARE @count INT;

DECLARE @errorMessage NVARCHAR(255);

SELECT @pesel = PESEL FROM inserted;

SELECT @count = COUNT(\*) FROM Uczniowie WHERE PESEL = @pesel;

IF @count > 1

BEGIN

SET @errorMessage = 'Osoba o numerze PESEL ' + @pesel + ' jest już w systemie';

RAISERROR(@errorMessage, 16, 1);

ROLLBACK TRANSACTION;

RETURN

END

END

GO

--Procedury

--Procedura P\_add\_a\_teacher wpisuje nowy rekord do tabeli Nauczyciele jednocześnie sprawdzając, czy PESEL nowego rekordu jest już w tabeli i czy nowy nauczyciel jest pełnoletni.

CREATE OR ALTER PROC P\_add\_a\_teacher

@Imię NVARCHAR(50),

@Nazwisko NVARCHAR(50),

@Płeć CHAR(1),

@Data\_Zatrudnienia DATE,

@Data\_Urodzenia DATE,

@Miejsce\_Urodzenia NVARCHAR(100),

@Obywatelstwo NVARCHAR(50),

@PESEL NVARCHAR(11),

@Adres NVARCHAR(255)

AS

BEGIN

--Sprawdzanie, czy nauczyciel o podanym numerze PESEL jest już w systemie

IF EXISTS (SELECT 1 FROM Nauczyciele WHERE PESEL = @PESEL)

BEGIN

RAISERROR('Nauczyciel o numerze PESEL %s jest już w systemie.', 16, 1, @PESEL);

RETURN;

END

--Sprawdzanie, czy nauczyciel ma co najmniej 18 lat

IF DATEDIFF(YEAR, @Data\_Urodzenia, GETDATE()) < 18

BEGIN

RAISERROR('Nauczyciel musi mieć co najmniej 18 lat.', 16, 1);

RETURN;

END

--Dodanie nauczyciela do tabeli

INSERT INTO Nauczyciele

(

Imię,

Nazwisko,

Płeć,

Data\_Zatrudnienia,

Data\_Urodzenia,

Miejsce\_Urodzenia,

Obywatelstwo,

PESEL,

Adres

)

VALUES

(

@Imię,

@Nazwisko,

@Płeć,

@Data\_Zatrudnienia,

@Data\_Urodzenia,

@Miejsce\_Urodzenia,

@Obywatelstwo,

@PESEL,

@Adres

);

END;

GO

--Procedura AvgGradesPerTeacher wyświetla tabelę z danymi nauczycieli, nazwą przedmiotu, którego i średnią wystawionych przez nich ocen z zakresu określonego parametrami

CREATE OR ALTER PROCEDURE P\_AvgGradesPerTeacherRange @MinAvgGrade float, @MaxAvgGrade float

AS

BEGIN

SELECT n.Imię, n.Nazwisko, np.Przedmiot, AVG(o.Ocena) as [Średnia Wystawionych Ocen]

FROM Nauczyciele AS n

INNER JOIN Oceny AS o

ON n.ID = o.Nauczyciel\_ID

INNER JOIN Nauczyciel\_Przedmiot AS np

ON n.ID = np.Nauczyciel\_ID

GROUP BY n.Imię, n.Nazwisko, np.Przedmiot

HAVING AVG(o.Ocena) BETWEEN @MinAvgGrade AND @MaxAvgGrade

ORDER BY [Średnia Wystawionych Ocen] DESC

END

--Funkcje

--Funkcja ForeignStudents wyświetla liczbę uczniów płci męskiej i żeńskiej o obywatelstwie innym niż polskie z roku określonego parametrem

CREATE FUNCTION dbo.ForeignStudents\_Y(@Rok INT)

RETURNS TABLE

AS

RETURN

(

SELECT

U.Płeć,

COUNT(\*) as [Liczba Obcokrajowców]

FROM

Uczniowie U

INNER JOIN

Klasa\_Uczen KU ON U.ID = KU.Uczen\_ID

INNER JOIN

Klasy K ON KU.Klasa\_ID = K.ID

WHERE

U.Obywatelstwo <> N'polskie'

AND K.Rok\_Szkolny = @Rok

GROUP BY

U.Płeć

)

--Funkcja YearsOfExperience oblicza i wyświetla staż pracy nauczycieli.

CREATE FUNCTION dbo.YearsOfExperience()

RETURNS TABLE

AS

RETURN

(

SELECT

ID,

Imię,

Nazwisko,

DATEDIFF(YEAR, Data\_Zatrudnienia, GETDATE()) AS [Staż pracy w latach]

FROM

Nauczyciele

)