Національний УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

факультет ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

кафедра комп’ютерних наук

**курсовИй Проект**

з дисципліни «Організація баз даних»

на тему:

**Автоматизоване робоче місце вчителя**

Виконав: \_\_\_\_\_\_\_\_\_\_**/**Прокіпчук Олександр Валентинович, ІПЗ-18005Б

 підпис ПІБ, група

**Робота захищена:**

« \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 р.

Оцінка за національною шкалою:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка ECTS: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 Члени комісії: \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

 (підпис) (прізвище та ініціали)

\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

**КИЇВ – 2020**

# ЗМІСТ

[ЗМІСТ 2](#_Toc41911610)

[ВСТУП 3](#_Toc41911611)

[РОЗДІЛ 1: ПОСТАНОВКА ЗАДАЧІ 4](#_Toc41911612)

[РОЗДІЛ 2: ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ 5](#_Toc41911613)

[2.1. Логічна модель даних у вигляді ER-діаграми 5](#_Toc41911614)

[2.2.1. Створення таблиць 6](#_Toc41911615)

[2.2.2. Створення уявлень 8](#_Toc41911616)

[2.2.3. Створення збережених процедур, тригерів 9](#_Toc41911617)

[2.2.4. Створення користувачів 11](#_Toc41911618)

[РОЗДІЛ 3: ПРИКЛАДНЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ 12](#_Toc41911619)

[3.1. Вибір інструментарію для створення прикладного програмного забезпечення 12](#_Toc41911620)

[3.2. Алгоритмізація та програмування програмних модулів 13](#_Toc41911621)

[3.2.1 Реалізація інтерфейсу користувача 13](#_Toc41911622)

[4.1. Тестування системи 15](#_Toc41911623)

[4.2Вимоги до апаратного та програмного забезпечення 23](#_Toc41911624)

[4.2.1. Вимоги до апаратного забезпечення 23](#_Toc41911625)

[4.2.2Вимоги до програмного забезпечення 23](#_Toc41911626)

[ВИСНОВКИ 24](#_Toc41911627)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 25](#_Toc41911628)

[ДОДАТОК A 26](#_Toc41911629)

[ДОДАТОК Б 28](#_Toc41911630)

ВСТУП

В наш час технології розвиваються дуже швидко, а отже зростає потреба в зберіганні та обробці великих об’ємів даних. Тому бази даних відіграють важливу роль в сучасному світі. Для керування ними використовують системи управління базами даних (СУБД). Це робить процес управління базою даних набагато ефективнішим та простішим. Більшість СУБД використовують структуровану мову запитів SQL для управління БД.

Я буду використовувати SQL та C# для створення свого проекту.

# РОЗДІЛ 1: ПОСТАНОВКА ЗАДАЧІ

Програмний продукт складається з інформаційної та програмної частини.

Ціллю даної роботи є полегшення рутинної роботи вчителям, зменшення роботи з паперами, покращення продуктивності праці.

Спершу потрібно створити базу даних, яка буде містити всю необхідну інформацію. Після цього потрібно розробити графічний інтерфейс, що дозволяє зручно керувати даними.

**Завдання які має виконувати система:**

1. Проведення уроків та виставлення оцінок
2. Забезпечення навчального процесу
3. Контроль над усіма учасниками навчального процесу

# РОЗДІЛ 2: ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

## 2.1. Логічна модель даних у вигляді ER-діаграми

Логічну модель даних у вигляді ER-діаграми було розроблено в середовищі розробки Visual Studio.



**ERдіаграма**

1. Кожна таблиця відповідає **першому правилу Кодда,** так як дані в комірках є атомарними, тобто кожен атрибут має лише одне значення, а не множину значень.
2. Кожна таблиця відповідає **1-й НФ** тому, що в таблицях відсутні повторення груп та правильно визначені неключові атрибути.
3. Кожна таблиця відповідає **2-й НФ** тому, що всі неключові поля залежать від ключового в межах однієї абстракції; визначено 8 абстракцій, які використовують первинний ключ; всі дані, які повторно з’являються в декількох рядках, винесені в окремі таблиці.
4. Кожна таблиця відповідає **3-й НФ** тому, що в моделі даних відсутні зв’язки багато до багатьох. В даній моделі будь-яке поле, що залежить від основного ключа та від будь-якого іншого поля, має виноситись в окрему таблицю; кожна сутність має лише один ключовий атрибут. 2.2. Створення бази даних

Для розробки БД було обрано систему управління базами даних Microsoft SQL Server та мову SQL.

Microsoft SQL Server – система управління реляційними базами даних (СУБД), розроблена корпорацією Microsoft. Це програмний продукт з основною функцією зберігання та обробки даних, як цього вимагають інші програмні додатки – з можливістю працювати на інших комп’ютерних системах. Основна мова для створення запитів - Transact-SQL.

База даних створюється за допомогою запиту на мові програмування SQL:



## 2.2.1. Створення таблиць

Для прикладу розглянемо створення 3-х основних таблиць (код створення дивитись в додатку **А**), таблиці Shedule, Student та Success:







Конструкція Drop table if exists використовується для перевірки перед створенням таблиці чи існує таблиця з такою назвою, якщо так то видаляє.

FOREIGN KEY, REFERENCES команди для реалізації зв’язків між таблицями.

## 2.2.2. Створення уявлень

Уявлення – це віртуальна таблиця, що може розглядатися як засіб отримання заздалегідь запланованого відображення і обробки даних, збережених в таблицях бази даних.

Створення уявлення Teacher\_lessons:

use ARM\_teacher

GO

Drop view if exists dbo.Teacher\_lessons

GO

CREATE VIEW dbo.Teacher\_lessons

AS

SELECT dbo.Lessons.Date\_of\_lesson, dbo.Teacher.Teacher\_name, dbo.Shedule.Cls\_ID, dbo.Subjct.Subject\_name

FROM dbo.Shedule

INNER JOIN dbo.Lessons ON dbo.Lessons.Shed\_ID = dbo.Shedule.Shedule\_ID

INNER JOIN dbo.Teacher ON dbo.Teacher.Teacher\_ID = dbo.Shedule.Tch\_ID

INNER JOIN dbo.Subjct ON dbo.Subjct.Subject\_ID = dbo.Teacher.Sub\_ID

## 2.2.3. Створення збережених процедур, тригерів

***Тригер*** — це спеціальний тип збережених процедур, які запускаються автоматично при виконанні певних дій з таблицями бази даних. Кожний тригер прив’язується до конкретної таблиці. Коли користувач БД робить спробу, наприклад, змінити якусь таблицю, сервер автоматично запускає тригер, та, якщо він завершився успішно, дозволяється виконати зміни

Створимо тригер для видалення записів з дочірньої таблиці перед видаленням записів з батьківської таблиці

go

create trigger Del\_Student on Student

instead of delete

as

begin

delete from Success where Success.Stdnt\_ID in (select deleted.Student\_ID from deleted)

delete from Student where Student.Student\_ID in (select deleted.Student\_ID from deleted)

end

***Збережені процедури*** - це набір операторів мови T-SQL. Збережена процедура зберігається у базі даних і є об’єктом цієї бази, так само, як таблиці, уявлення, користувачі. Вона виконується на сервері, що значно пришвидшує роботу. Перевагою збережених процедурою можна назвати значне полегшення роботи АРМ, оскільки замість безлічі звернень, створені збережені процедури виконуються тільки один раз (при звернені відповідно до неї).

Процедура для додавання запису в таблицю Success:

CREATE PROC Add\_success

(

 @Date\_of\_mark datetime,

 @Mark int,

 @Stdnt\_ID char(10),

 @Wrk\_ID char(10),

 @Ls\_ID char(10)

)

AS

BEGIN

 SET @Date\_of\_mark = LTRIM(RTRIM(@Date\_of\_mark));

 SET @Mark = LTRIM(RTRIM(@Mark));

 SET @Stdnt\_ID = LTRIM(RTRIM(@Stdnt\_ID));

 SET @Wrk\_ID = LTRIM(RTRIM(@Wrk\_ID));

 SET @Ls\_ID = LTRIM(RTRIM(@Ls\_ID));

 INSERT INTO Success(Date\_of\_mark, Mark, Stdnt\_ID, Wrk\_ID, Ls\_ID)

 VALUES (@Date\_of\_mark, @Mark, @Stdnt\_ID, @Wrk\_ID, @Ls\_ID)

END;

## 2.2.4. Створення користувачів

В моїй базі даних присутні 2 ролі: вчитель та завуч.

Завуч має право на управління всією базою даних.

Вчитель може додавати, видаляти, редагувати дані лише в певних таблицях.

Для прикладу покажу створення користувача вчителя та надання йому привілеїв:

EXEC sp\_addlogin @loginame = 'MathTeacher',

 @passwd = 'MathTeacher'

EXEC sp\_adduser 'MathTeacher'

GRANT SELECT,INSERT,UPDATE,DELETE ON Work TO MathTeacher

GRANT SELECT,INSERT,UPDATE,DELETE ON Success TO MathTeacher

GRANT SELECT,INSERT,UPDATE,DELETE ON Shedule TO MathTeacher

GRANT SELECT,INSERT,UPDATE,DELETE ON Lessons TO MathTeacher

GRANT EXECUTE to MathTeacher

# РОЗДІЛ 3: ПРИКЛАДНЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

## 3.1. Вибір інструментарію для створення прикладного програмного забезпечення

Для розробки прикладного програмного забезпечення було обрано мову С# та середовище Microsoft Visual Studio

**Microsoft Visual Studio** — інтегроване середовище розробки (IDE) від Microsoft. Використовується для розробки комп’ютерних програм, а також веб-сайтів, веб-додатків, веб-служб та мобільних додатків. Може створювати як власний код, так і керований код.

**Windows Forms** — це безкоштовна графічна бібліотека (GUI) з відкритим кодом, що входить до складу Microsoft .NET Framework або Mono Framework, що забезпечує платформу для запису багатьох клієнтських додатків для настільних, ноутбуків та планшетних ПК.

**C#** — [об'єктно-орієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%E2%80%99%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) з безпечною [системою типізації](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%97) для платформи [NET](https://uk.wikipedia.org/wiki/.NET). [Синтаксис](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81) C# близький до [С++](https://uk.wikipedia.org/wiki/%D0%A1%2B%2B) і [Java](https://uk.wikipedia.org/wiki/Java). Мова багатогранної парадигми загального призначення, яка охоплює сильні типізаційні, лексично-розширені, імперативні, декларативні, функціональні, загальні, об'єктно-орієнтовані (на основі класу) та компонентні дисципліни програмування. Вона була розроблена близько 2000 року в рамках свого проекту .NET

Для створення звітів використано програмний компонент, який вбудований в Microsoft Visual Studio – Microsoft RDLC (Report Definition Language). Microsoft Definition Language (RDL) – стандарт, запропонований Microsoft для опису звітів. RDL – це XML-додаток, який найчастіше використовується в Microsoft SQL Server Reporting Services. RDL зазвичай роблять за допомогою Visual Studio.

## 3.2. Алгоритмізація та програмування програмних модулів

### 3.2.1 Реалізація інтерфейсу користувача

Під час створення інтерфейсу користувача було використано такі компоненти:

Lable – надпис

TextBox – поле для введення певної інформації

Button – кнопка

ToolStripMenu – меню, яке має підпункти

DataGridView – засіб перегляду даних з бази даних

CheckBox – поле з випадаючим списком

**3.2.2. Забезпечення інтерфейсу з базою даних**

Базу даних до проекту було під’єднано за допомогою власно розробленого модулю DataModule та додатку Wisard.exe. Далі за допомогою SQLConnection, SQLDataAdapter виконувалися дії по відкриттю та закриттю підключень та вибірок з бази даних до інтерфейсу.

**DataModule:**

public class DataModule

 {

 static string StringFromWizard = null;

 static bool GetConnectionString()

 {

 string startupPath = Application.StartupPath;

 string FileName = startupPath + "\\DBConnectionString";

 try

 {

 using (StreamReader sr = new StreamReader(FileName))

 {

 StringFromWizard = sr.ReadToEnd();

 }

 return true;

 }

 catch (Exception e)

 {

 return false;

 }

 }

 static string ComposeConnectionString(string UserID, string Pass)

 {

 string Provider, Persist\_Security\_Info, User\_ID, Password, Initial\_Catalog, Data\_Source;

 GetConnectionString();

 string[] splitted = StringFromWizard.Split(';');

 string ConnStr;

 Provider = splitted[0];

 Persist\_Security\_Info = splitted[1];

 User\_ID = "User ID=" + UserID;

 Password = "Password=" + Pass;

 Initial\_Catalog = splitted[3];

 Data\_Source = splitted[4];

 ConnStr = Data\_Source + ";" + Persist\_Security\_Info + ";" + Initial\_Catalog + ";" + User\_ID + ";" + Password;

 return ConnStr;

 }

 public static bool StartConnection(string loginUser, string passUser)

 {

 string cs;

 if (!GetConnectionString())

 {

 MessageBox.Show("Не знайдений файл з параметрами зв'язку", "Помилка авторизації!");

 Application.Exit();

 return false;

 }

 else

 {

 cs = ComposeConnectionString(loginUser, passUser);

 try

 {

 Connection.con.ConnectionString = cs;

 Connection.con.Open();

 return true;

 }

 catch (Exception)

 {

 MessageBox.Show("Неправильний логін або пароль", "Помилка авторизації");

 return false;

 }

 }

 }

 }

Методи DataModule отримують рядок, що створив Wisard.exe та розпочинають сеанс роботи з базою даних.

РЕКОМЕНДАЦІЇ ЩОДО ВПРОВАДЖЕННЯ ТА ЕКСПЛУАТАЦІЇ СИСТЕМИ

## 4.1. Тестування системи

Після запуску програми користувач потрапить на вікно авторизації, де потрібно ввести логін та пароль, після чого натиснути кнопку ‘Увійти’ або кнопку ‘Вийти’ для завершення роботи.****

Після авторизації користувач потрапляє на головне вікно на якому розташовані: логотип школи, кнопка ‘Закрити’, яка завершить роботу та відкриє вікно авторизації, календар для планування навчального процесу, у верхній частині вікна є меню з вкладками ‘Навчальна робота’, ‘Довідник’ та ‘Про програму’, в кутку є зображення, яке показує інформацію про користувача. 

Вкладка ‘Навчальна робота’ для адміністратора містить 2 підвкладки: 

Натиснувши на ‘Змінити розклад’ адміністратор зможе редагувати розклад



Натиснувши на ‘Змінити учасників навчального процесу’ адміністратор зможе редагувати списки учнів та вчителів 



Вкладка ‘Навчальна робота’ для вчителя містить 1 підвкладку:



Натиснувши на ‘Розклад’ користувач (вчитель) побачить розклад занять (для кожного вчителя свій): 

На вкладці ‘Довідник’ містяться 3 підвкладки: ‘Учні’, ‘Вчителі’, ‘Предмети’.



Обравши певну вкладку буде показано інформацію про учнів, вчителів та предмети відповідно.



Натиснувши на вкладку ‘Про програму’ відкриється вікно: 

Натиснувши на  відкриється інформація про користувача. 

Натиснувши на вкладку ‘Звіт’відкриється вікно в якому можна згенерувати звіт з інформацією про проведені уроки:****

## 4.2Вимоги до апаратного та програмного забезпечення

### 4.2.1. Вимоги до апаратного забезпечення

**Мінімальні технічні вимоги до апаратного забезпечення користувача для роботи з програмою**

* Процесор класу Intel Core i3 1GHz;
* Об’єм оперативної пам’яті не менше 1GB;
* Мінімальна пропускна здатність каналу зв’язку – 128 Kbit/s;

### 4.2.2Вимоги до програмного забезпечення

**4.2.2.1. Вимоги до системного програмного забезпечення**

Операційна система Windows XP та вища;

Наявність Microsoft SQL Server.

## ВИСНОВКИ

Було створено систему для взаємодії людини з базою даних в зручних умовах через графічний інтерфейс. Цей програмний продукт може бути використаний в реальних умовах, так як фактично виконує основні задачі які перед ним стоять, а саме:

* Видалення даних з таблиць.
* Додавання даних до таблиць.
* Редагування.
* Пошук.
* Зручний перегляд.

Звісно, можна покращувати програму, додаючи до неї нові компоненти, збільшувати функціонал.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Самоучитель Visual Studio .NET 2003 — Андрей Гарнаев
2. М. Грубер «Понимание SQL»
3. Полное руководство по языку программирования С# 8.0 и платформе .NET Core 3: <https://metanit.com/sharp/tutorial/>

# ДОДАТОК A

Створення таблиць:

use ARM\_teacher

IF EXISTS (SELECT name FROM sys.objects

 WHERE name = 'Success' AND type\_desc = 'USER\_TABLE')

drop table Success

IF EXISTS (SELECT name FROM sys.objects

 WHERE name = 'Lessons' AND type\_desc = 'USER\_TABLE')

drop table Lessons

 IF EXISTS (SELECT name FROM sys.objects

 WHERE name = 'Shedule' AND type\_desc = 'USER\_TABLE')

drop table Shedule

IF EXISTS (SELECT name FROM sys.objects

 WHERE name = 'Student' AND type\_desc = 'USER\_TABLE')

drop table Student

IF EXISTS (SELECT name FROM sys.objects

 WHERE name = 'Class' AND type\_desc = 'USER\_TABLE')

drop table Class

IF EXISTS (SELECT name FROM sys.objects

 WHERE name = 'Teacher' AND type\_desc = 'USER\_TABLE')

drop table Teacher

IF EXISTS (SELECT name FROM sys.objects

 WHERE name = 'Subjct' AND type\_desc = 'USER\_TABLE')

drop table Subjct

IF EXISTS (SELECT name FROM sys.objects

 WHERE name = 'Work' AND type\_desc = 'USER\_TABLE')

drop table Work

-------------------------------------------------------------------------

create table Subjct

( Subject\_ID nchar(10) NOT NULL primary key,

 Subject\_name nchar(50));

create table Teacher

( Teacher\_ID char(50) NOT NULL primary key,

 Teacher\_name char(50),

 Sub\_ID nchar(10) NOT NULL,

 CONSTRAINT Sub\_ID FOREIGN KEY (Sub\_ID) REFERENCES Subjct (Subject\_ID));

create table Class

( Class\_ID char(10) NOT NULL primary key);

create table Student

( Student\_ID char(10) NOT NULL primary key,

 Student\_name char(50),

 Student\_class char(10) NOT NULL,

 CONSTRAINT Student\_class FOREIGN KEY (Student\_class) REFERENCES Class (Class\_ID));

 create table Shedule

( Shedule\_ID char(10) NOT NULL primary key,

 Day\_of\_lesson char(10) NOT NULL,

 Number\_of\_lesson int NOT NULL,

 Room\_number int NOT NULL,

 Cls\_ID char(10) NOT NULL,

 Tch\_ID char(50) NOT NULL,

 CONSTRAINT Cls\_ID FOREIGN KEY (Cls\_ID) REFERENCES Class (Class\_ID),

 CONSTRAINT Tch\_ID FOREIGN KEY (Tch\_ID) REFERENCES Teacher (Teacher\_ID) );

create table Lessons

( Lesson\_ID char(10) primary key,

 Date\_of\_lesson date,

 Shed\_ID char(10) NOT NULL,

 CONSTRAINT Shed\_ID FOREIGN KEY (Shed\_ID) REFERENCES Shedule (Shedule\_ID));

create table Work

( Work\_ID char(10) NOT NULL primary key,

 Form\_of\_work char(30));

create table Success

( Date\_of\_mark datetime primary key,

 Mark int,

 Stdnt\_ID char(10) NOT NULL,

 Wrk\_ID char(10) NOT NULL,

 Ls\_ID char(10) NOT NULL,

 CONSTRAINT Stdnt\_ID FOREIGN KEY (Stdnt\_ID) REFERENCES Student (Student\_ID),

 CONSTRAINT Wrk\_ID FOREIGN KEY (Wrk\_ID) REFERENCES Work (Work\_ID),

 CONSTRAINT Ls\_ID FOREIGN KEY (Ls\_ID) REFERENCES Lessons (Lesson\_ID));

# ДОДАТОК Б

Програмний код для під’єднання користувача до додатку:

public partial class LoginForm : Form

 {

 public LoginForm()

 {

 InitializeComponent();

 }

 private void buttonBack\_Click(object sender, EventArgs e)

 {

 this.Close();

 Application.Exit();

 }

 private void buttonEnter\_Click(object sender, EventArgs e)

 {

 progressBar1.Visible = true;

 for (progressBar1.Value = progressBar1.Minimum; progressBar1.Value < progressBar1.Maximum; progressBar1.Value++)

 {

 Thread.Sleep(10);

 }

 if (!DataModule.StartConnection(loginField.Text.ToString(), passField.Text.ToString()))

 {

 this.Hide();

 LoginForm fr = new LoginForm();

 fr.Show();

 }

 else

 {

 Connection.user = loginField.Text.ToString();

 MainForm fr = new MainForm();

 fr.Show();

 Hide();

 }

 }

 }