

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”  
НАУКОВО-НАВЧАЛЬНИЙ КОМПЛЕКС  
“ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ”

Курсова робота

з дисципліни “Бази даних”  
на тему: ”Театр”

Виконала:  
Акінфієва Є.О.  
група КА-32

Прийняла:  
Яковлєва Т.В.

Київ-2016

## Зміст

Вступ.....	5
1.Постановка задачі.....	6
2. Архітектура та інформаційне забезпечення АСОД.....	6
2.1.Аналіз функціонування та організаційні засади автоматизованої частини підприємства.....	6
2.2. Проектування інформаційного забезпечення системи Комплексу.....	8
3. Програмна реалізація АСОД.....	10
4. Випробування підсистеми АСОД.....	18
Висновки.....	20
Список використаної літератури.....	21
Додатки.....	22

## АННОТАЦИЯ

В данной работе разработана и описана информационная система “Театр”

Цель создания этого проекта – оптимизировать и автоматизировать работу сотрудников театра.

Программное обеспечение (ПО) имеет интуитивно понятный и простой интерфейс и предназначено для персонала, имеющего навыки работы с компьютером, в частности в операционной системе (ОС) Windows, а также свободно ориентирующегося в области театральной организации.

## АНОТАЦІЯ

В цій роботі розібрано та описано інформаційну систему «Театр». Мета створення цього проекту – оптимізувати та автоматизувати роботу працівників театру.

Програмне забезпечення (ПЗ) має інтуїтивно зрозумілий та простий інтерфейс та призначений для персоналу, що має навички роботи з комп'ютером, а саме з операційною системою (ОС) Windows, а також що вільно орієнтується в сфері театральної організації.

## ANNOTATION

In this work developed and described the information system "Theatre"

The purpose of this project - to optimize and automate the work of employees of the theater.

Software (SW) has an intuitive and simple interface and is designed for staff with computer skills, in particular in the operating system (OS), the Windows, also well as freely orient themselves in the field of theatrical organization.

## ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

АРМ – автоматизоване робоче місце

МТ – менеджер театру

НР – hr менеджер

БД – база даних

АСОД – автоматизована система обробки даних

ПЗ – програмне забезпечення

ОС – операційна система

ОЗУ – оперативний запам'ятовуючий пристрій

Мб – мега байт

ЕОМ – електронно-обчислювальна машина

## ВСТУП

Автоматизована система обробки даних була спроектована для досить широкої області, що охоплює сферу мистецтва. Дана система може бути застосована не тільки театрами, а й іншими представникам відповідних сфер, наприклад кінотеатр. Передбачається, що відвідувачі театру купують квитки без реєстрації. Всі працівники театру знаходяться в базі даних, при цьому кожен може працювати тільки на одній посаді, щоб максимально задовольнити потреби відвідувачів.

Розроблена система враховує всі особливості даної предметної області та є ідеальним інструментом для обробки даних.

## 1. ПОСТАНОВКА ЗАДАЧІ

Дана автоматизована система обробки даних призначена для використання її в театрі. Її призначення являє собою полегшення роботи менеджерів театру, а також візуалізація всіх даних систем, з можливістю маніпулювати ними.

## 2. АРХІТЕКТУРА ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ АСОД

2.1 Аналіз функціонування та організаційні засади автоматизованої частини підприємства

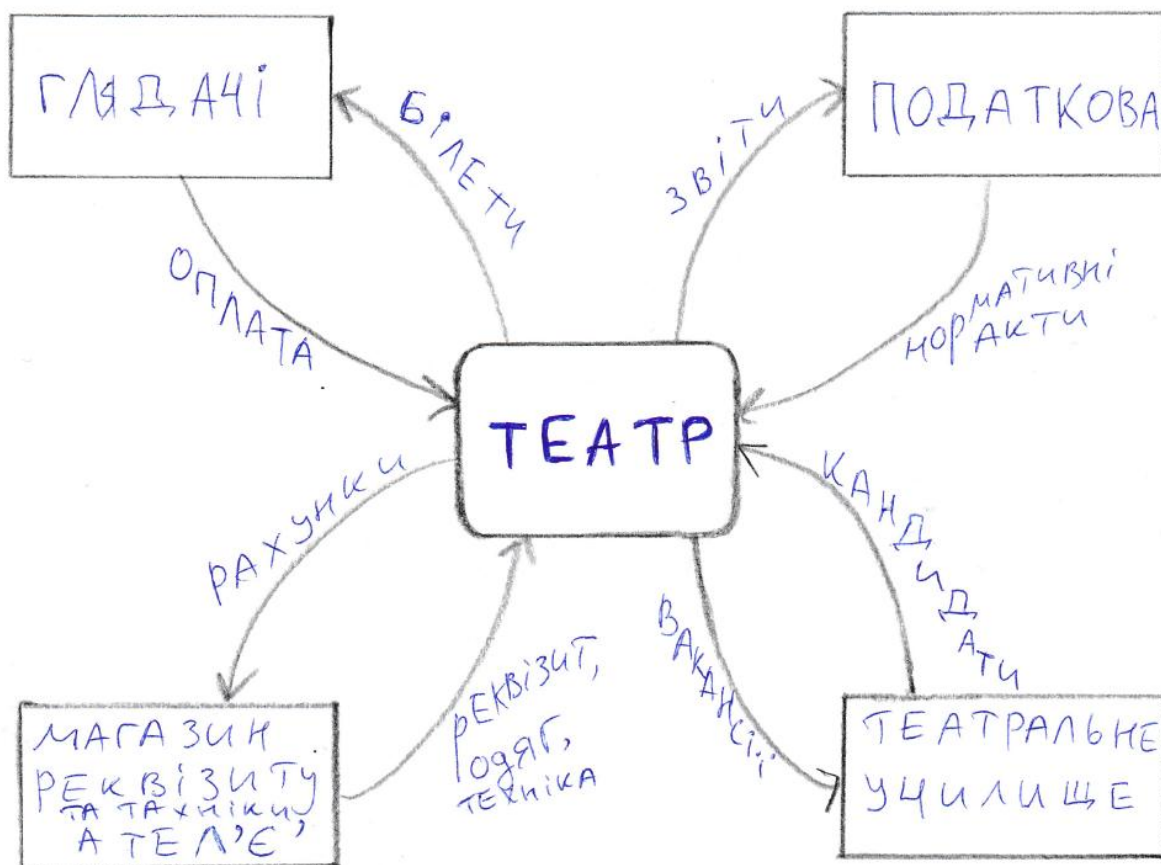
Модельований комплекс має досить широко орієнтовану діяльність, спрямовану на споживачів різних категорій з метою надання можливості отримання високоякісного обслуговування в даній сфері. Можливості досягнення поставлених цілей, базуються на вивченні ринку попиту, а також бажань споживача, з підтримкою стратегії гнучкого управління.

В залежності від рівня користувача, можна сформулювати різні інформаційні потреби в типі даних.

Функції менеджерів:

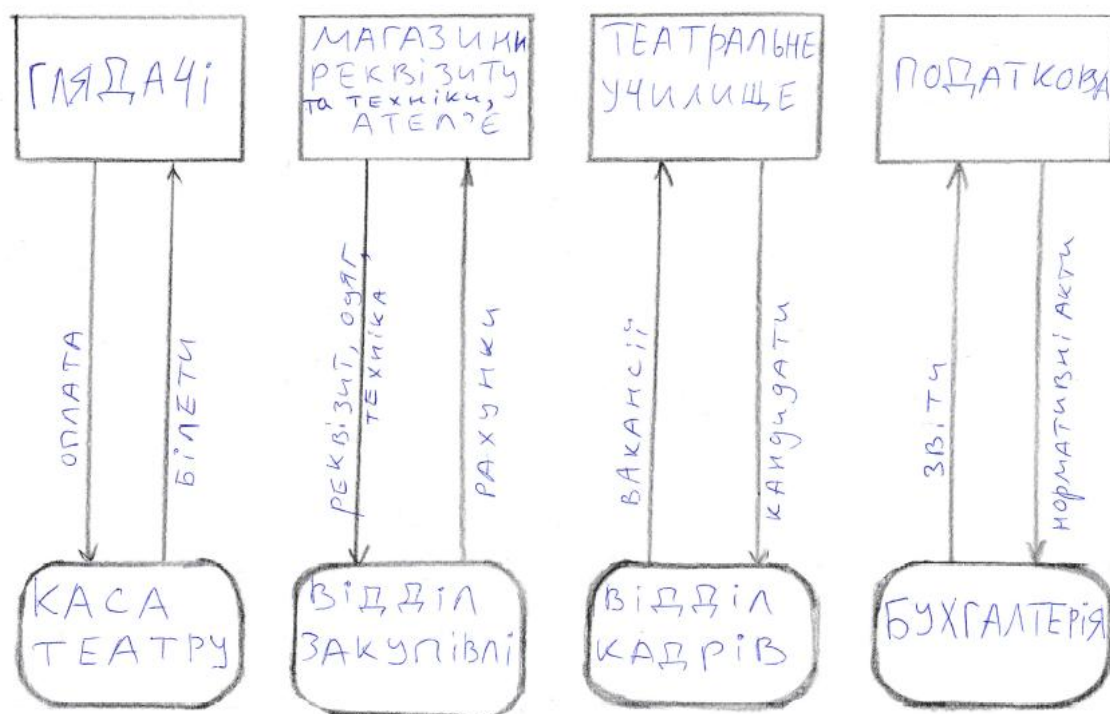
- здійснення різних операцій з інвентарем та продуктами;
- оформлення контрактів при прийомі на роботу персоналу;
- облік продуктів та інвентарю;
- прийом на роботу, звільнення персоналу;
- зміна посади персоналу;

Діаграма потоків даних нульового рівня (ДФД\_0) (рис.2.1).

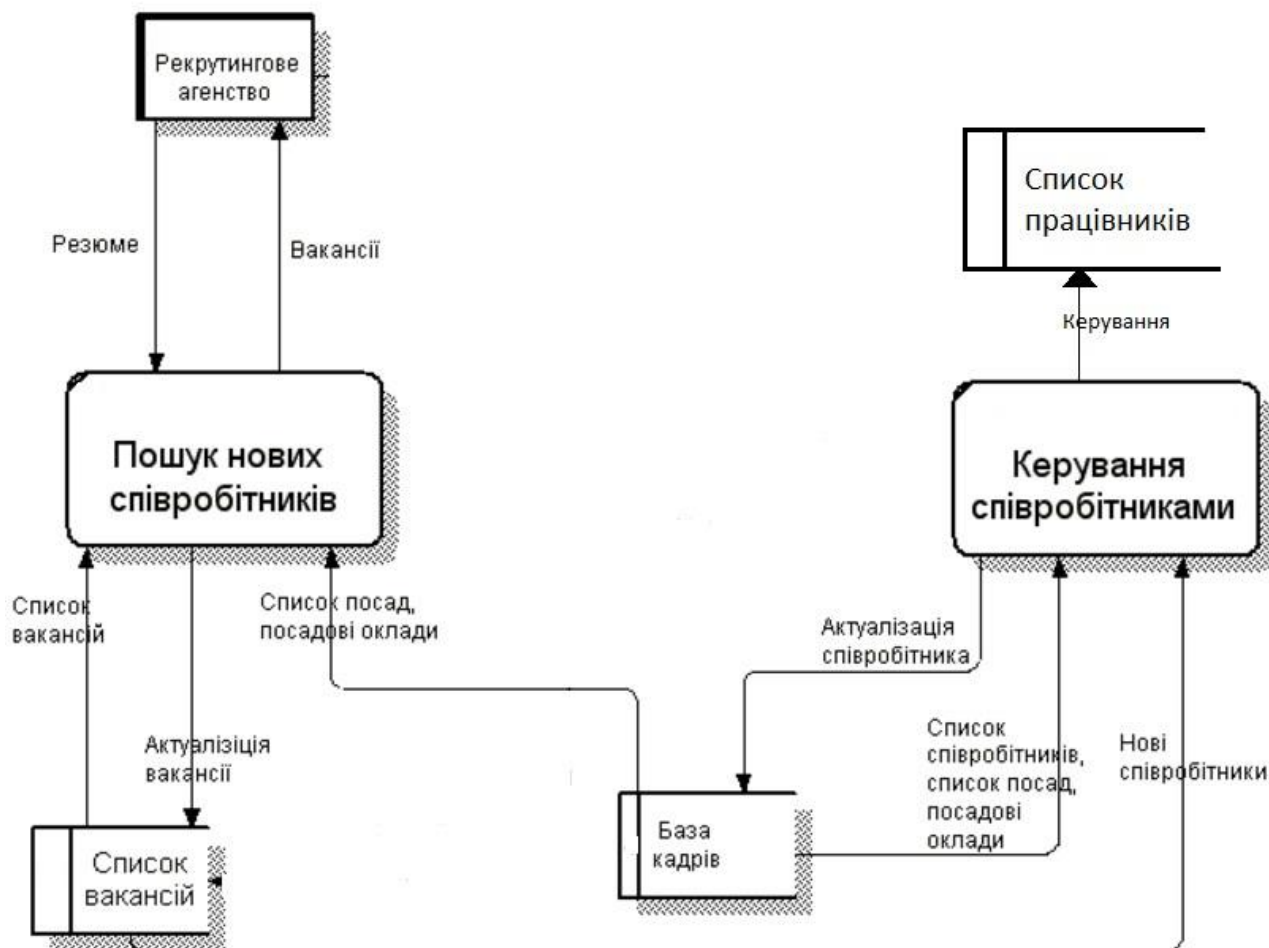


ДФД1:

Діаграма потоків даних першого рівня (ДФД1) для театру.



ДФД\_2 для АРМ менеджера театру:



У таблиці наведено опис подій, характерних для АРМ менеджера театру:

Опис події	Реакція системи
Людина хоче влаштуватися на роботу	Знайти в базі всі вакансії та вивести цей список на екрані
Людина влаштовується на роботу	Вивести форму для заповнення, внести в базу нового робітника, записати в базу, що дана вакансія зайнята
Робітник переводиться на іншу посаду	Знайти нову вакансію, звільнити стару, внести зміни до бази персоналу.
Робітник звільнюється	Звільнити вакансію, внести зміну до бази персоналу



## 2.2. Проектування інформаційного забезпечення системи комплексу

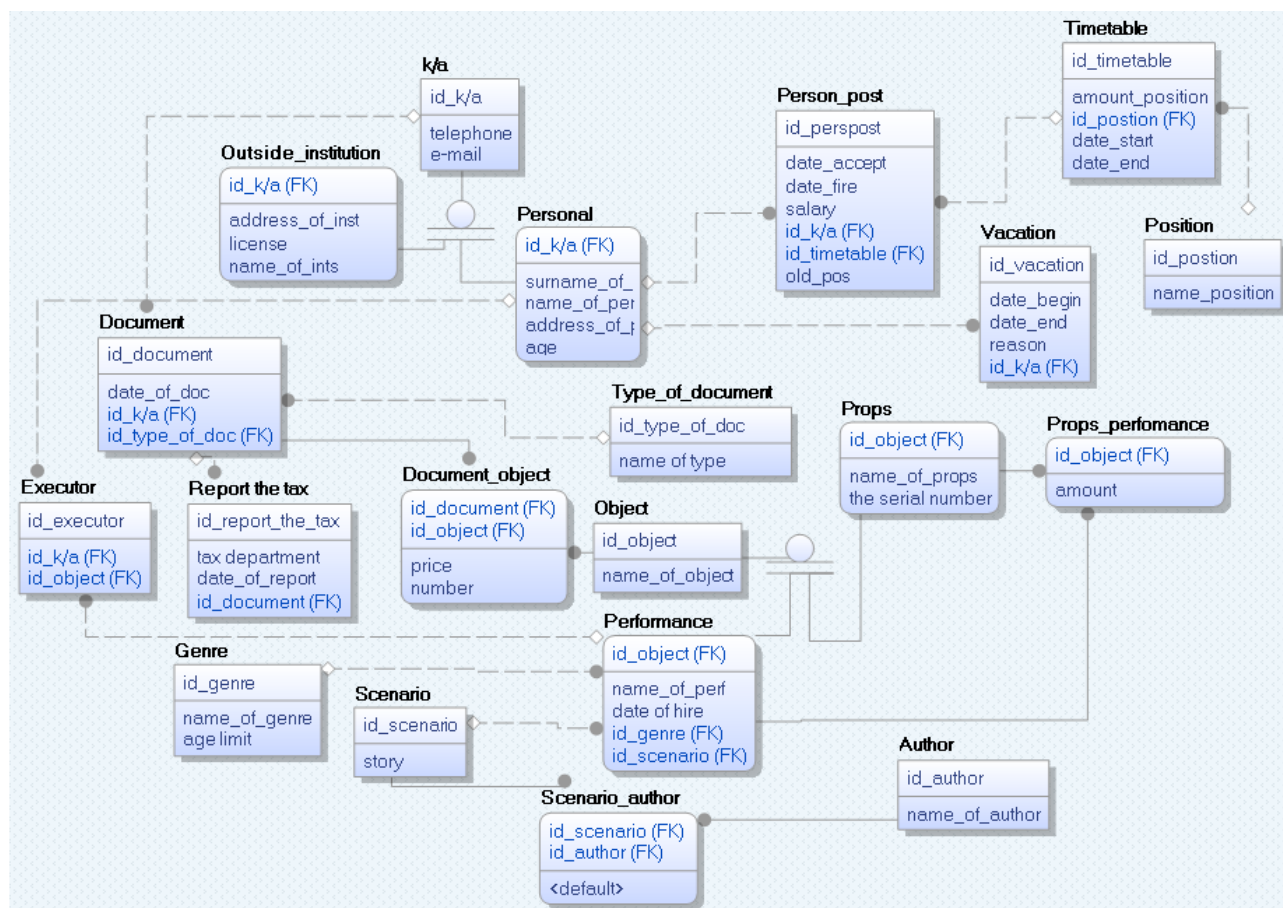
### 2.2.1. Інфологічне проектування (локальна модель)

Інфологічною моделлю є інформаційно-логічна модель комплексу. Для даної системи розроблена локальна інфологічна модель.

Були виділені такі основні сутності:

1. Персонал
2. Стороння установа
3. Вистава
4. Реквізит

Зв'язки між сутностями продемонстровані на інфологічній моделі системи "Театр" представленої у вигляді логічної ER - діаграми виконаної засобами case-засоби Erwin ERX 3.5.2 (рис. 2.2).



### 2.2.2. . Даталогічне проектування (локальна модель)

Даталогічне проектування являє собою фізичну модель орієнтовану на конкретну СУБД.

Даталогічне модель представлена на рис.2.3. У даталогічній моделі для кожної сутності вказується назва атрибутів, чи є атрибут первинним ключем, альтернативним або зовнішнім ключем, чи може він приймати значення NULL, значення атрибутів за замовчуванням, як такі значень можуть виступати «0» або «1», задається тип даних .

Зв'язки між сутностями представлені у вигляді фізичної ER - діаграми підсистеми виконаної за допомогою case-засоби Erwin 3.5.2 (рис. 2.3.)

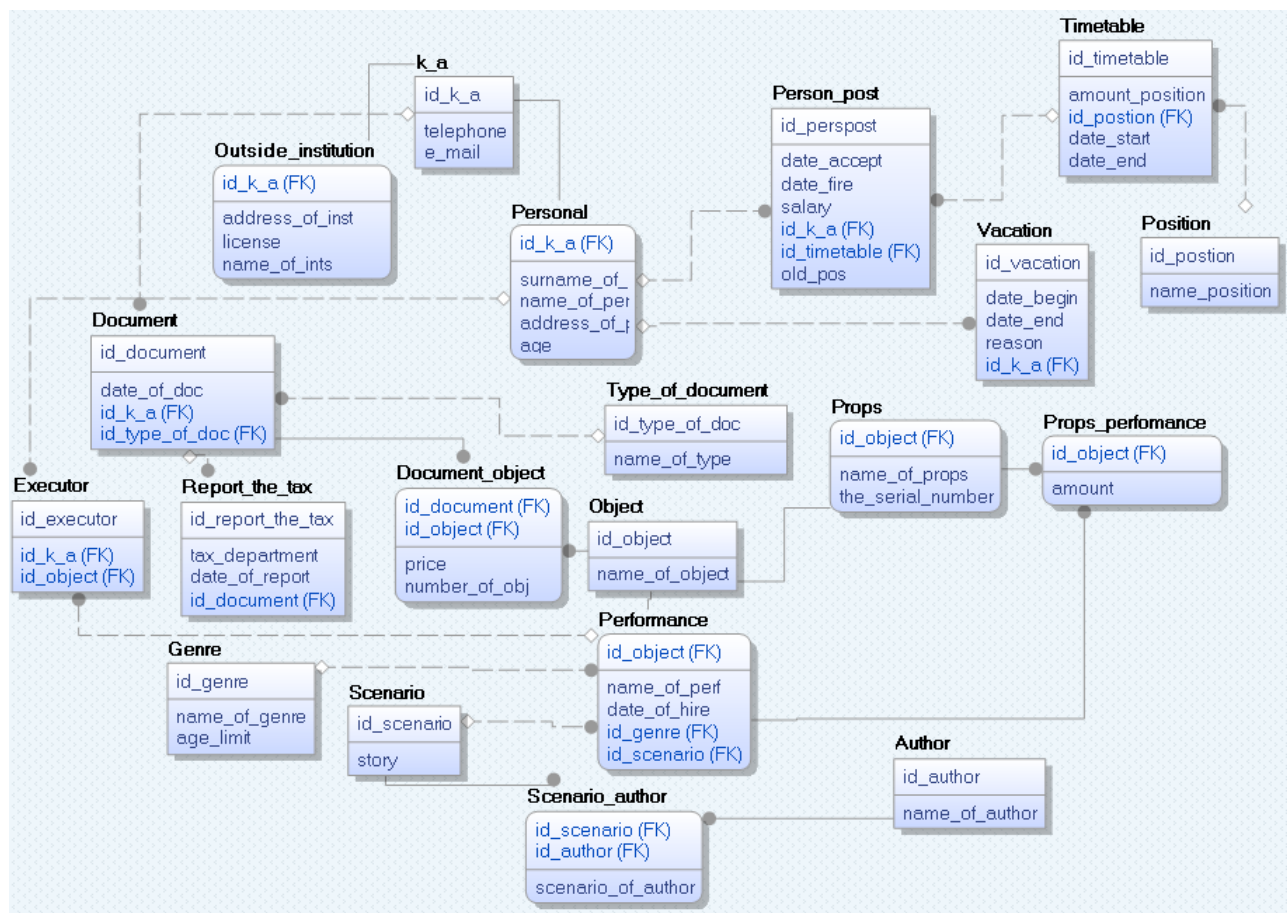


Рис.2.3. Фізична ER-діаграма системи “Театр”

Бізнес правила відділу кадрів:

1. На роботу можна приймати працівників без судимостей.

2. Працівник може займати одночасно лише одну посаду.
3. Змінювати посаду працівника можливо лише за існуючої вакансії .
4. В постанові можуть приймати участь діти за згодою батьків.

### 2.2.3. Проектування інтерфейсу користувача

Інтерфейс розроблений в стандарт і за допомогою засобів MS Windows.

Основні принципи організації інтерфейсу:

1. Розробка багатовіконного додатка.
2. Використання меню, контекстного меню і кнопок для управління ходом роботи.
3. Відображення, редагування і додавання інформації БД проводиться з використанням таблиць та інших форм відображення даних.
4. Використання необхідних підписів до елементів інтерфейсу (полях введення, таблицями і т. Д.).
5. Повідомлення користувачеві про хід роботи видаються в окремому вікні.

Однією з цілей проектування інтерфейсу була розробка найбільш зручного і інтуїтивно зрозумілого для користувача інтерфейсу.

Дані вимоги до призначеного для користувача інтерфейсу були виконані розробником.

## 3. ПРОГРАМНА РЕАЛІЗАЦІЯ АСОД

### 3.1. Структура програмного забезпечення

Головний модуль управляє запуском всіх модулів програми. Головний модуль містить компонент ADO, за допомогою якого за допомогою технології ADO та інтерфейсу OLE DB, реалізується з'єднання з сервером баз даних. Решта модулів використовують цей компонент, таким чином система в своїй роботі використовує тільки одне з'єднання, що економить цінні ресурси сервера. Кожен модуль працює окремо, тому що реалізований у вигляді класу, а також може обмінюватися інформацією з іншими модулями. Кожен модуль має

свій компонент для взаємодії з сервером баз даних, але вони використовують одне єдине з'єднання даних.

Клієнтська програму реалізує бізнес-логіку і рівень представлення. Серверний додаток, реалізований набором незв'язаних підпрограм (процедур, тригерів і функцій), реалізує логіку бази даних.

Кожен програмний модуль реалізує деяку частину спільного завдання за допомогою генеруються на своєму боці запитів на діалекті мови Transact-SQL від корпорації Microsoft, або, в більш складному випадку, коли завдання виконується дуже складно шляхом генерації SQL-запиту безпосередньо на стороні клієнта, зверненням до підпрограми на стороні сервера (збереженій процедури).

### 3.2. Опис програми

Програмне забезпечення розроблене в середовищі Visual Studio 2012, C# , а також використовувалася программа баз даних Oracle SQLDeveloper.

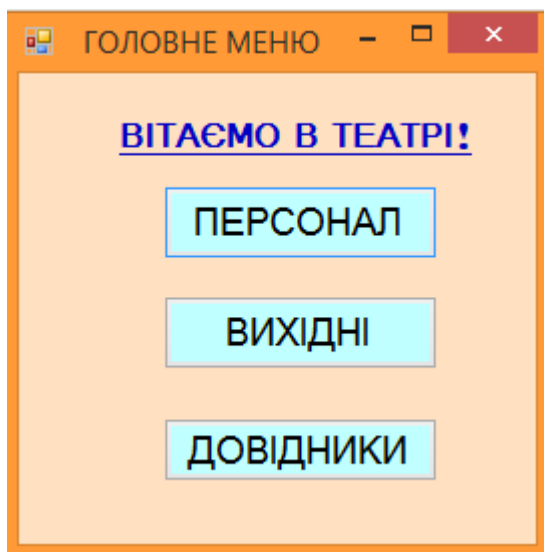
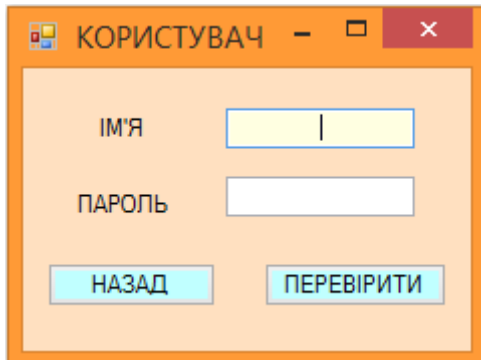
Нижче описані програмні модулі клієнтської частини системи і їх інтерфейси, модуль працює одночасно і як візуальна форма (тобто реалізує рівень представлення), і реалізує частину бізнес логіки шляхом додавання в клас додаткових методів.

Кожен програмний модуль реалізує деяку частину спільного завдання за допомогою генеруються на своєму боці запитів на діалекті мови Transact-SQL від корпорації Microsoft, або, в більш складному випадку, коли завдання виконується дуже складно шляхом генерації SQL-запиту безпосередньо на стороні клієнта, зверненням до підпрограми на стороні сервера.

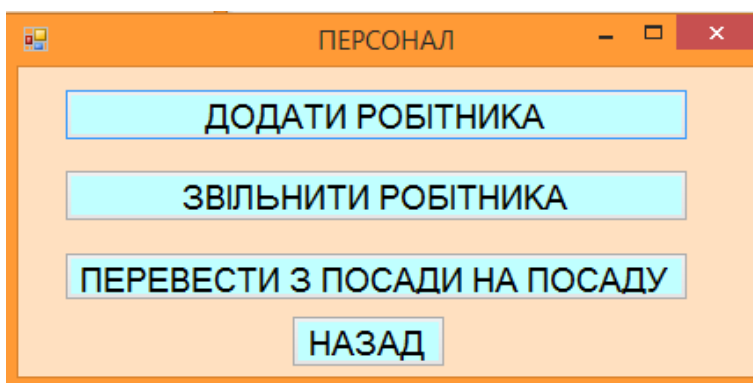
В системі використовуються тригери і процедури.

### 3.3. Інструкція користувача

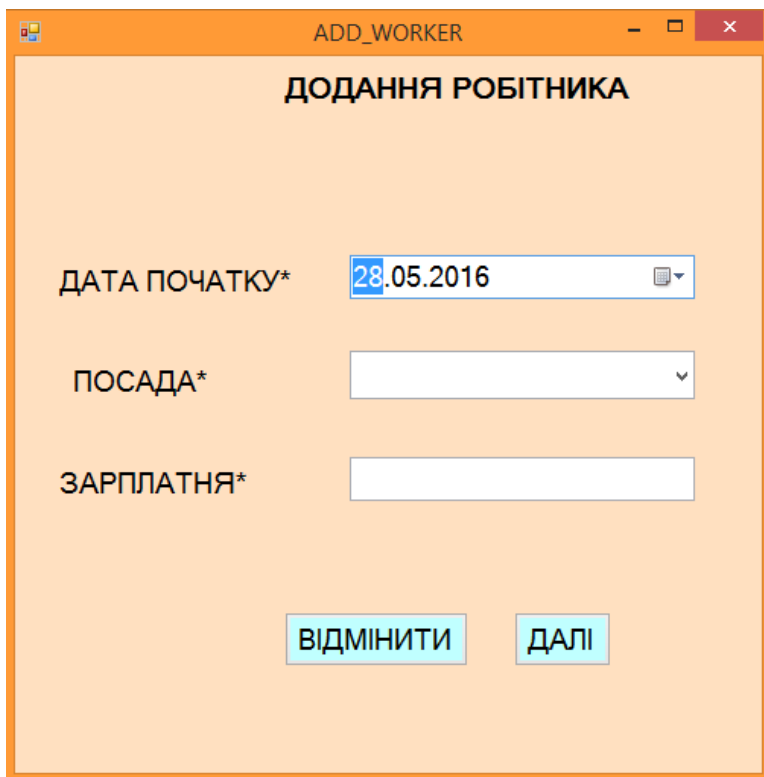
При запуску програми з'являється вікно користувача, де необхідно ввести ім'я та пароль користувача. За успішного з'єднання з'являється головного меню, де можна обрати або виконання дій з даними працівників, або перегляд цих даних.



При виборі «ПЕРСОНАЛ» з'являється нове вікно, де можна додати нового робітника, звільнити існуючого робітника, або перевести робітника з посади на посаду.



При виборі «ДОДАТИ РОБІТНИКА» відкривається вікно, де необхідно ввести дату прийняття на роботу, посаду та зарплатню, після чого натиснути «ДАЛІ», тоді відкриється вікно, де необхідно ввести основні дані робітника, такі як: ім'я, прізвище, вік, телефон, адреса та пошта.



ADD\_WORKER

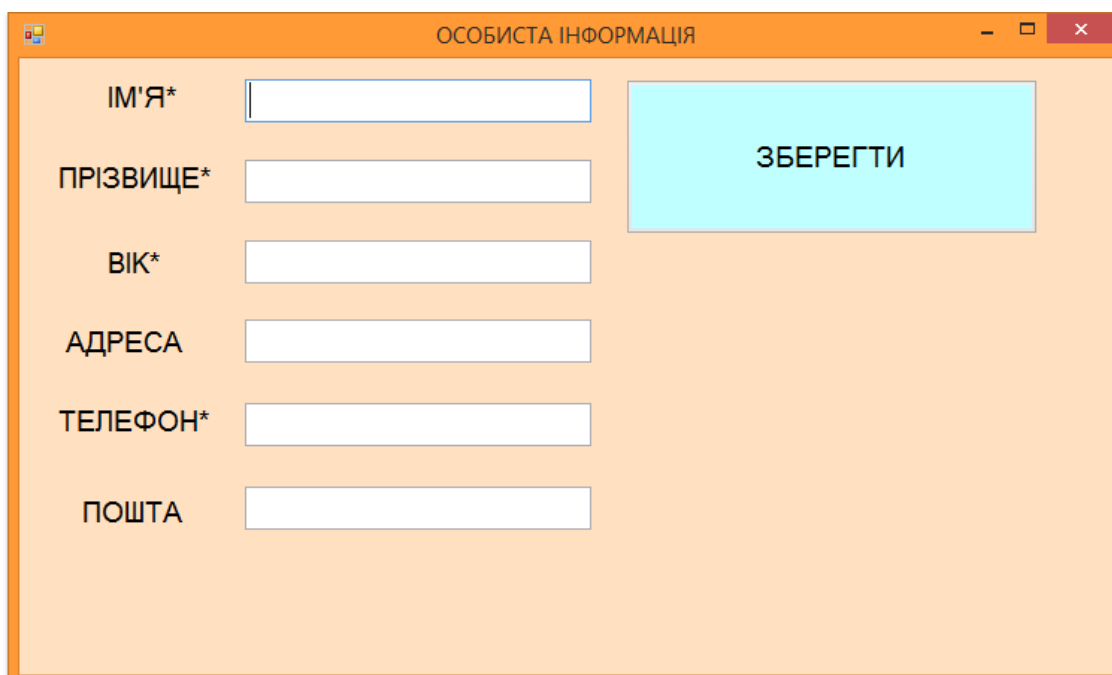
**ДОДАННЯ РОБІТНИКА**

ДАТА ПОЧАТКУ\* 28.05.2016

ПОСАДА\*

ЗАРПЛАТНЯ\*

ВІДМІНИТИ ДАЛІ



ОСОБИСТА ІНФОРМАЦІЯ

ІМ'Я\*

ПРІЗВИЩЕ\*

ВІК\*

АДРЕСА

ТЕЛЕФОН\*

ПОШТА

ЗБЕРЕГТИ

При виборі «ЗВІЛЬНИТИ РОБІТНИКА» з'являється вікно, де необхідно зі списку прізвищ обрати працівника, якого треб звільнити, інформація про

нього з'являється в полях вікна, треба ввести дату звільнення та натиснути «ЗВІЛЬНИТИ».

The screenshot shows a window titled "FIRE" with a light gray background. On the left, there is a form with the following fields: "ІМ'Я" (Name) with value "qqq", "ПРИЗВИЩЕ" (Surname) with value "qqq", "ВІК" (Age) with value "111", "посада" (Position) with a dropdown menu showing "актор", and "ДАТА ЗВІЛЬНЕННЯ" (Release Date) with a date picker showing "28 травня 2016 р.". Below these fields are two buttons: "ВІДМІНИТИ" (Cancel) and "ЗВІЛЬНИТИ" (Release). On the right side of the window, there is a vertical list box containing the values "111" and "qqq", with "qqq" currently selected and highlighted in blue.

При виборі «ВИХІДНІ» з'являється вікно , де є можливість додати відпустку або лікарняний, декретну відпустку та інші можливі вихідні дні, які були у робітника протягом певного періоду часу. Необхідно вибрати працівника з таблиці, додати дату початку вихідного, дату закінчення та вказати причину, після чого зберегти введені дані.

The screenshot shows a window titled "ВИХІДНІ" (Holidays). On the left, there is a table with the following columns: "NAME\_OF\_PER", "SURNAME\_OF", and "NAME\_POSITIO". The table contains two rows: the first row has values "11", "111", and "візажист" (Hairdresser), and the second row has values "qqq", "qqq", and "актор" (Actor). Below the table is a button labeled "ПЕРЕГЛЯНУТИ ВИХІДНІ ПЕРСОНАЛУ" (View Staff Holidays). On the right side of the window, there is a form titled "ДОДАТИ ВИХІДНИЙ" (Add Holiday) with the following fields: "ІМ'Я" (Name), "ПРИЗВИЩЕ" (Surname), "ПОСАДА" (Position), "ПОЧАТОК" (Start Date) with a date picker showing "28 травня 2016 р.", "ЗАКІНЧЕННЯ" (End Date) with a date picker showing "28 травня 2016 р.", and "ПРИЧИНА\*" (Reason) with a text input field. Below these fields is a button labeled "ДОДАТИ" (Add). At the bottom center of the window is a button labeled "МЕНЮ" (Menu).

При виборі «ДОВІДНИКИ» з'являється вікно з таблицею даних про робітників, які зараз працюють в театрі. В цьому вікні також є можливість переглянути які робітники працювали в театрі за певний період часу, а також змінити дані таблиці, клікнувши два рази на інформацію, яку треба змінити. Для цього відкриється нове вікно з даними про цього робітника та можна буде легко редагувати необхідні дані.

**ДОВІДНИКИ**

	ІМ'Я	ПРІЗВИЩЕ	ВІК	АДРЕСА	ТЕЛЕФОН	E_MAIL	ЗАРПЛАТНЯ	ПОСАДА	ДАТА ПРИЙНЯТТЯ
▶	qqq	qqq	111	qqqq	1111	aaaa	2222	актор	18.02.2015
	11	111	111	111	111	11	6000	візажист	18.02.2015
	gjh	gigf	88	88	88	88	888	бухгалтер	18.02.2015

	НАЗВА ПОСАДИ	ВСЬОГО МІСЦЬ	К-СТЬ ЗАЙНЯТИХ	КІЛЬКІСТЬ ВІПЬНИХ
▶	охорона	4	0	4
	актор	5	1	4
	прибиральник	2	0	2
	звукорежисер	3	0	3
	сценарист	2	0	2
	візажист	4	1	3

ПЕРЕГЛЯНУТИ КОЛИШНІХ ПРАЦІВНИКІВ

З 28 травня 2016 р.

ПО 28 травня 2016 р.

ПЕРЕГЛЯНУТИ ПРАЦІВНИКІВ НА ПОСАДІ

ПЕРЕГЛЯНУТИ ВІДПУСТКИ

## 4. ВИПРОБУВАННЯ ПІДСИСТЕМИ АСОД

### 4.1. Опис тестової бази даних

#### 1. Додання робітника:

- Спочатку необхідно заповнити дату прийняття на роботу, посаду та зарплату нового робітника, потім його особисті дані.
- Обов'язковими є поля дата, посада, ім'я, прізвище, вік та телефон.
- Кнопка збереження даних недоступна при незаповнених обов'язкових полях або некоректно введених даних.

#### 2. Звільнення робітника

- Звільнення робітника відбувається шляхом додавання дати звільнення та звільнення посади. При цьому в базі даних зберігається на якій посаді працював робітник.
- Звільнивши робітника, його дані не будуть доступні для переведення з посади на посаду, в довідниках у таблиці «працюючі робітники».
- Щоб звільнити робітника, необхідно зі списку обрати відповідне прізвище, його основні дані з'являться в полях, щоб було легше визначити необхідного робітника. Необхідно вибрати дату звільнення.
- Натиснувши кнопку «звільнити», в дані робітника додається дата звільнення.



## 2. Переведення з посади на посаду

- Обравши пункт меню «перевести з посади на посаду» відкривається вікно, в якому зі списку необхідно обрати потрібного робітника та змінити його посаду.
- Дана операція означає, що робітник вважається звільненим з минулої посади в дату, коли він був переведений на нову посаду, відповідно прийнятий на роботу знову.
- В даному вікні можливо змінити зарплатню на новій посаді.

### 4.2. Результати тестування на ЕВМ

В результаті випробувань роботи програми з базою даних були випробувані такі функції АРМ:

1. Додавання робітника.
2. Модифікація даних робітника.
3. Звільнення робітника.
4. Створення звіту.
5. Перегляд даних за датою.
6. Перегляд даних за посадою.
7. Перегляд вакансій.
8. Перегляд інформації про посади.
9. Додання вихідного робітникам.

Всі перераховані приклади підтверджують надійність і працездатність розробленої АСОД.

## ВИСНОВКИ

Отже, в складі АСОД «Театр» реалізовано АРМ «Менеджер театру», який виконує такі функції:

1. Додає працівника.
2. Звільняє працівника.
3. Додає вихідні.
4. Переводить з посади на посаду.
5. Робить звіт в бухгалтерію.

Дана програма повністю відповідає поставленим функціям та вимогам системи:

- Результати виводяться в зрозумілому вигляді.
- Коректна обробка виключень та помилок.
- Зручний, інтуїтивно-зрозумілий інтерфейс.

## СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ:

1. Конноллі Томас, Бегг Каролін. Бази даних: проектування, реалізація і супровід. Теорія і практика, 3-е вид.: Пер. з англ. - М.: Видавничий дім «Вільямс», 2003. - 1440 с.: Іл. - Хрон. тит. англ.;
2. Нейгел Крістіан, Івєн Білл, Глин Джей і ін С # 2005 для професіоналів. : Пер з англ. - М.: ТОВ «І.Д. Вільямс », 2006. - 1376 с.
3. Миколаєва Н.А. Мова структурованих запитів. Лабораторні роботи: навчальний посібник / Н.А. Миколаєва, Т.Ю. Калініна. - Ухта: УГТУ, 2006. - 124 с. мул.

## ДОДАТКИ:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace BD1
{
    static class dateStart
    {
        //add personal add date
        public static DateTime Value { get; set; }
    }
    static class changeId
    {
        //change position
        public static int Value { get; set; }
    }
    static class delId
    {
        //to fire person
        public static int Value { get; set; }
    }
    static class salaryVal
    {
        public static int Value { get; set; }
    }
    static class conn_string
    {
        public static string Value { get; set; }
    }
    static class idPo
    {
        public static int Value { get; set; }
    }
    static class indexPosition
    {
        public static int Value { get; set; }
    }
    static class Program1
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();

            Application.SetCompatibleTextRenderingDefault(false);
            //Application.Run(new FIRE());
            // Application.Run(new dovid());
            // Application.Run(new ADD_WORKER());
            // Application.Run(new Reporte());
            Application.Run(new USER());
            // Application.Run(new changep());
        }
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Oracle.DataAccess.Client;
using Oracle.DataAccess.Types;
namespace BD1
{
    public partial class USER : Form
    {
        public USER()
        {
            InitializeComponent();
        }
        // private OracleConnection conn = new
        OracleConnection();
        private void button1_Click(object sender,
        EventArgs e)
        {
            bool connected = true;
            string userid = textBox1.Text;
            string password = textBox2.Text;
```

```
        //OracleConnection conn = new
        OracleConnection("Data Source = localhost; User ID =
        elizabeth ;Password=12345");
        conn_string.Value = "Data Source = localhost;"
        + "User ID = " + userid + ";Password=" + password;
        OracleConnection conn = new
        OracleConnection(conn_string.Value);
        try
        {
            conn.Open();
        }
        catch(OracleException ex)
        {
            connected = false;
            switch(ex.Number)
            {
                case 1:
                    MessageBox.Show("Error attempting
                    to insert duplicate data.");
                    break;
                case 12560:
                    MessageBox.Show("The database is
                    unavailable.");
                    break;
                default:
                    MessageBox.Show("Database error: "
                    + ex.Message.ToString());
                    break;
            }
        }
        catch(Exception ex)
        {
            connected = false;
            MessageBox.Show(ex.Message.ToString());
        }
        if (connected)
        {
            this.Hide();
            main_menu m = new main_menu();
            m.Show();
        }
    }

    private void textBox1_TextChanged(object sender,
    EventArgs e)
    {
    }

    private void textBox2_TextChanged(object sender,
    EventArgs e)
    {
    }

    private void button2_Click(object sender,
    EventArgs e)
    {
        OracleConnection conn = new
        OracleConnection(conn_string.Value);
        conn.Open();
        this.Hide();
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Oracle.DataAccess.Client;
using Oracle.DataAccess.Types;
//using System.Data.OracleClient;

namespace BD1
{
    public partial class main_menu : Form
    {
        public main_menu()
        {
            InitializeComponent();
```

```

        if (conn_string.Value == ("Data Source =
localhost;" + "User ID = " + "human_res" + ";Password="
+"12345"))
        {
            button1.Enabled = true;
            button4.Enabled = true;
        }
    }
    private void label1_Click(object sender, EventArgs
e)
    {
    }

    private void button1_Click(object sender,
EventArgs e)
    {
        PERSONAL p = new PERSONAL();
        this.Hide();
        p.Show();
    }

    private void button4_Click(object sender,
EventArgs e)
    {
        dovid d = new dovid();
        this.Hide();
        d.Show();
    }

    private void button2_Click(object sender,
EventArgs e)
    {
    }

    private void button3_Click(object sender,
EventArgs e)
    {
    }

    private void main_menu_Load(object sender,
EventArgs e)
    {
    }

    private void button5_Click(object sender,
EventArgs e)
    {
        vac v = new vac();
        this.Hide();
        v.Show();
    }
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Oracle.DataAccess.Client;
using Oracle.DataAccess.Types;
//using System.Data.OracleClient;

namespace BD1
{
    public partial class PERSONAL : Form
    {
        public PERSONAL()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender,
EventArgs e)
        {
            OracleConnection conn = new
OracleConnection(conn_string.Value);
            conn.Open();
            ADD_WORKER a = new ADD_WORKER();
            a.Show();
            name_surname p = new name_surname();
            // p.Show();

```

```

                this.Hide();
            }

            private void button4_Click(object sender,
EventArgs e)
            {
                OracleConnection conn = new
OracleConnection(conn_string.Value);
                conn.Open();
                main_menu m = new main_menu();
                this.Hide();
                m.Show();
            }

            private void button3_Click(object sender,
EventArgs e)
            {
                this.Hide();
                changep n = new changep();
                n.button1.Enabled = true;
                n.Show();
                // n.button1.Enabled = false;
            }

            private void button2_Click(object sender,
EventArgs e)
            {
                FIRE f = new FIRE();
                this.Hide();
                f.Show();
            }

            private void PERSONAL_Load(object sender,
EventArgs e)
            {
            }
        }
    }
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
//using Oracle.DataAccess.Client;
//using Oracle.DataAccess.Types;
using System.Data.OracleClient;

namespace BD1
{
    public partial class ADD_WORKER : Form
    {
        public ADD_WORKER()
        {
            InitializeComponent();
            FillCombo();
        }

        public void FillCombo()
        {
            string sq = "select po.name_position,
va.amount_position, va.kilkist_viln_posad " +
                        "from elizabeth.position po
                        "inner join
elizabeth.vakansii va " +
                        "on po.id_position =
va.id_position " +
                        "order by po.name_position";
            string query = "SELECT * FROM
elizabeth.POSITION";
            OracleConnection conn = new
OracleConnection(conn_string.Value);
            OracleCommand cmd = new OracleCommand(sq,
conn);

            OracleDataReader myReader;
            try
            {
                conn.Open();
                myReader = cmd.ExecuteReader();
                while (myReader.Read())
                {
                    if
(myReader.GetInt32(myReader.GetOrdinal("kilkist_viln_posad
")) != 0)

```

```

        {
            string sname =
myReader.GetString(myReader.GetOrdinal("name_position"));
            comboBox1.Items.Add(sname);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}

public void dateTimePicker1_ValueChanged_1(object
sender, EventArgs e)
{
    dateStart.Value = dateTimePicker1.Value.Date;
}
public void comboBox1_SelectedIndexChanged(object
sender, EventArgs e)
{
    string query = "SELECT * FROM
elizabeth.position where name_position='" + comboBox1.Text
+ "'";
    OracleConnection conn = new
OracleConnection(conn_string.Value);
    OracleCommand cmd = new OracleCommand(query,
conn);
    OracleDataReader myReader;
    try
    {
        conn.Open();
        myReader = cmd.ExecuteReader();
        while (myReader.Read())
        {
            string sid =
myReader.GetInt32(myReader.GetOrdinal("id_position")).ToSt
ring();
            indexPosition.Value =
int.Parse(sid);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}
public void button2_Click(object sender, EventArgs
e){}
public void button1_Click(object sender, EventArgs
e)
{
    this.Hide();
    main_menu m = new main_menu();
    m.Show();
}
public void ADD_WORKER_Load(object sender,
EventArgs e) { }
public void button2_Click_1(object sender,
EventArgs e)
{
    if (comboBox1.Text == "" || salaryBox.Text ==
"")
    {
        MessageBox.Show("ЗАПОВНИТЬ ОБОВ'ЯЗКОВІ
ПОЛЯ");
    }
    else
    {
        OracleConnection conn = new
OracleConnection(conn_string.Value);
        conn.Open();
        name_surname n = new name_surname();
        n.Show();
        this.Hide();
    }
}
public void textBox1_TextChanged_1(object sender,
EventArgs e)
{
    try
    {
        salaryVal.Value =
Int32.Parse(salaryBox.Text);
    }
    catch (Exception ex)
    {
    }
}
}

```

```

private void button3_Click(object sender,
EventArgs e)
{
    button2.Enabled = true;
}

private void button4_Click(object sender,
EventArgs e)
{
    changep c = new changep();
    c.Show();
    c.button3.Enabled = true;
    this.Hide();
}

private void comboBox1_Validating(object sender,
CancelEventArgs e)
{
}

private void dateTimePicker1_Validating(object
sender, CancelEventArgs e) {
}

private void checkBox1_CheckedChanged(object
sender, EventArgs e)
{
}

private void textBox1_TextChanged(object sender,
EventArgs e)
{
}

private void salaryBox_Validating(object sender,
CancelEventArgs e)
{
    int n;
    bool isNumeric = int.TryParse(salaryBox.Text,
out n);
    if (isNumeric!=true)
        MessageBox.Show("некоректна зарплатня!");
}
}
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
//using Oracle.DataAccess.Client;
using Oracle.DataAccess.Types;
using System.Data.SqlTypes;
using System.Data.OracleClient;

namespace BD1
{
    public partial class name_surname : Form
    {
        public name_surname()
        {
            InitializeComponent();
        }

        private void name_surname_Load(object sender,
EventArgs e){}

        private void textBox4_TextChanged(object sender,
EventArgs e){}

        private void textBox3_TextChanged(object sender,
EventArgs e){}

        public void button1_Click(object sender, EventArgs
e)
        {
            //string c = "Data Source = localhost;" +
            "User ID = " + "elizabeth" + ";Password=" + "12345";
            // OracleConnection conn = new
            OracleConnection(c);
            OracleConnection conn = new
            OracleConnection(conn_string.Value);
            try

```

```

{
    conn.Open();
}
catch (Exception ex)
{
    MessageBox.Show(ex.ToString());
}
}
OracleCommand cmd = new
OracleCommand("elizabeth.ADD_PERSONAL", conn);
cmd.CommandType = CommandType.StoredProcedure;

if (textBox3.Text == "" || textBox4.Text == ""
|| textBox6.Text == "" || textBox2.Text=="")
{
    MessageBox.Show("ЗАПОЛНИТЬ ОБОВ'ЯЗКОВІ
ПОЛЯ!");
}
else
{
    OracleParameter NAME_OF_PERS1 = new
OracleParameter("NAME_OF_PERS1", OracleType.VarChar, 150);
// OracleParameter NAME_OF_PERS1 = new
OracleParameter("NAME_OF_PERS1", OracleDbType.Varchar2,
150);
NAME_OF_PERS1.Direction =
ParameterDirection.Input;
NAME_OF_PERS1.Value = this.textBox3.Text;
cmd.Parameters.Add(NAME_OF_PERS1);

    OracleParameter SURNAME_OF_PERS1 = new
OracleParameter("SURNAME_OF_PERS1", OracleType.VarChar,
150);
//OracleParameter NAME_OF_PERS1 = new
OracleParameter("SURNAME_OF_PERS1", OracleDbType.Varchar2,
150);
SURNAME_OF_PERS1.Direction =
ParameterDirection.Input;
SURNAME_OF_PERS1.Value =
this.textBox4.Text;
cmd.Parameters.Add(SURNAME_OF_PERS1);

    OracleParameter ADDRESS_OF_PERS1 = new
OracleParameter("ADDRESS_OF_PERS1", OracleType.VarChar,
150);
//OracleParameter ADDRESS_OF_PERS1 = new
OracleParameter("ADDRESS_OF_PERS1", OracleDbType.Varchar2,
150);
ADDRESS_OF_PERS1.Direction =
ParameterDirection.Input;
ADDRESS_OF_PERS1.Value =
this.textBox1.Text;
cmd.Parameters.Add(ADDRESS_OF_PERS1);

    OracleParameter TELEPHONE1 = new
OracleParameter("TELEPHONE1", OracleType.Int32, 150);
// OracleParameter telephon = new
OracleParameter("TELEPHONE1", OracleDbType.Int32,150);
TELEPHONE1.Direction =
ParameterDirection.Input;
try
{
    TELEPHONE1.Value = this.textBox2.Text;
}
catch (Exception ex) { }
cmd.Parameters.Add(TELEPHONE1);

    OracleParameter E_MAIL1 = new
OracleParameter("E_MAIL1", OracleType.VarChar, 150);
//OracleParameter email = new
OracleParameter("email", OracleDbType.Varchar2,150);
E_MAIL1.Direction =
ParameterDirection.Input;
E_MAIL1.Value = this.textBox5.Text;
cmd.Parameters.Add(E_MAIL1);

    ADD_WORKER a = new ADD_WORKER();

    OracleParameter DATE_ACCEPT1 = new
OracleParameter("DATE_ACCEPT1", OracleType.DateTime);
//OracleParameter dateOfStart = new
OracleParameter("DATE_ACCEPT1", OracleDbType.Date);
DATE_ACCEPT1.Direction =
ParameterDirection.Input;
DATE_ACCEPT1.Value = dateStart.Value;
cmd.Parameters.Add(DATE_ACCEPT1);

    OracleParameter id_timetable1 = new
OracleParameter("id_timetable1", OracleType.Int32, 150);
//OracleParameter id_timetable1 = new
OracleParameter("id_timetable1", OracleDbType.Int32, 150);

```

```

id_timetable1.Direction =
ParameterDirection.Input;
id_timetable1.Value = indexPosition.Value;
cmd.Parameters.Add(id_timetable1);

    OracleParameter salary1 = new
OracleParameter("salary1", OracleType.Int32, 150);
//OracleParameter salary1 = new
OracleParameter("salary1", OracleDbType.Int32, 150);
salary1.Direction =
ParameterDirection.Input;
salary1.Value = salaryVal.Value;
cmd.Parameters.Add(salary1);

    OracleParameter age1 = new
OracleParameter("age1", OracleType.Int32, 150);
//OracleParameter age1 = new
OracleParameter("age1", OracleDbType.Int32, 150);
age1.Direction = ParameterDirection.Input;
try
{
    age1.Value = textBox6.Text;
}
catch (Exception ex) { }
cmd.Parameters.Add(age1);

    try
    {
        cmd.ExecuteNonQuery();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message,
"Retrieving Oracle Sequence Values", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        return;
    }
}

main_menu n = new main_menu();
this.Hide();
n.Show();
}

private void textBox1_TextChanged(object sender,
EventArgs e){}
private void textBox2_TextChanged(object sender,
EventArgs e){}
private void textBox6_TextChanged(object sender,
EventArgs e){}
private void textBox5_TextChanged(object sender,
EventArgs e){}

private void textBox2_Validating(object sender,
CancelEventArgs e)
{
    int n;
    bool isNumeric = int.TryParse(textBox2.Text,
out n);
    if (isNumeric != true)
        MessageBox.Show("НЕКОРЕКТНИЙ ТЕЛЕФОН!");
}

private void textBox6_Validating(object sender,
CancelEventArgs e)
{
    int n;
    bool isNumeric = int.TryParse(textBox6.Text,
out n);
    if (isNumeric != true)
        MessageBox.Show("НЕКОРЕКТНИЙ ВІК!");
}

private void textBox4_Validating(object sender,
CancelEventArgs e)
{
}

private void textBox3_Validating(object sender,
CancelEventArgs e)
{
}
}

using System;
using System.Collections.Generic;

```

```

using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
//using Oracle.DataAccess.Client;
using System.Data.OracleClient;

namespace BD1
{
    public partial class FIRE : Form
    {
        public FIRE()
        {
            InitializeComponent();
            FillList();
        }

        void FillList()
        {
            string query = "SELECT * FROM elizabeth.VIEW1";
            //string query = "SELECT * FROM
            elizabeth.VIEW1";
            OracleConnection conn = new
            OracleConnection(conn_string.Value);
            //OracleConnection conn = new
            OracleConnection("Data Source = localhost; User ID =
            elizabeth ;Password=12345");
            conn.Open();
            OracleCommand cmd = new
            OracleCommand(query, conn);
            OracleDataReader myReader = null;
            try
            {
                myReader = cmd.ExecuteReader();
                while (myReader.Read())
                {
                    string sname =
                    myReader.GetString(myReader.GetOrdinal("surname_of_pers"));
                }
                listBox1.Items.Add(sname);

                //comboBox1.Items.Add(myReader.GetString(myReader.GetOrdin
                al("NAME_OF_POSITION")));
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString());
            }
        }

        private void button2_Click(object sender,
        EventArgs e)
        {
            PERSONAL p = new PERSONAL();
            this.Hide();
            p.Show();
        }

        private void listBox1_SelectedIndexChanged(object
        sender, EventArgs e)
        {
            string query = "SELECT * FROM elizabeth.VIEW1
            where surname_of_pers='" + listBox1.Text + "'";
            OracleConnection conn = new
            OracleConnection("Data Source = localhost; User ID =
            elizabeth ;Password=12345");
            OracleCommand cmd = new OracleCommand(query,
            conn);
            OracleDataReader myReader;
            try
            {
                conn.Open();
                myReader = cmd.ExecuteReader();
                while (myReader.Read())
                {
                    delId.Value =
                    myReader.GetInt32(myReader.GetOrdinal("ID_K_A"));

                    string sname =
                    myReader.GetString(myReader.GetOrdinal("NAME_OF_PERS"));
                    string ssurname =
                    myReader.GetString(myReader.GetOrdinal("SURNAME_OF_PERS"))
                    ;

```

```

                    string age =
                    Convert.ToString(myReader.GetInt32(myReader.GetOrdinal("ag
                    e")));

                    string name_position1 =
                    myReader.GetString(myReader.GetOrdinal("NAME_POSITION"));
                    textBox1.Text = sname;
                    textBox2.Text = ssurname;
                    textBox3.Text = age;
                    textBox4.Text = name_position1;
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString());
            }
        }

        private void button1_Click(object sender,
        EventArgs e)
        {
            OracleConnection conn = new
            OracleConnection("Data Source = localhost; User ID =
            Elizabeth; Password= 12345");
            conn.Open();
            OracleCommand cmd = new
            OracleCommand("elizabeth.DEL_PERSONAL", conn);
            cmd.CommandType = CommandType.StoredProcedure;

            OracleParameter id_k = new
            OracleParameter("id_k", OracleType.Number, 150);
            //OracleParameter id_k = new
            OracleParameter("id_k", OracleDbType.Int32, 150);
            id_k.Direction =
            ParameterDirection.Input; //OracleType
            id_k.Value = delId.Value;
            cmd.Parameters.Add(id_k);

            //OracleParameter date_release = new
            OracleParameter("date_release", OracleDbType.Date);
            OracleParameter date_release = new
            OracleParameter("date_release", OracleType.DateTime);
            date_release.Direction =
            ParameterDirection.Input;
            date_release.Value =
            dateTimePicker1.Value.Date;
            cmd.Parameters.Add(date_release);

            OracleParameter name_position1 = new
            OracleParameter("name_position1", OracleType.VarChar,
            150);
            //OracleParameter name_position1 = new
            OracleParameter("name_position1", OracleDbType.Int32,
            150);
            name_position1.Direction =
            ParameterDirection.Input; //OracleType
            name_position1.Value = textBox4.Text;
            cmd.Parameters.Add(name_position1);

            try
            {
                cmd.ExecuteNonQuery();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Retrieving
                Oracle Sequence Values", MessageBoxButtons.OK,
                MessageBoxIcon.Error);
                return;
            }
            MessageBox.Show("Звільнений.");
            this.Close();
            PERSONAL personal = new PERSONAL();
            personal.Show();
        }
    }
}

```

Процедури:

1. Додавання робітника

```

create or replace PROCEDURE ADD_PERSONAL (name1 IN
VARCHAR2,surname1 IN VARCHAR2,
address IN VARCHAR2 ,telephon in number,email in
varchar2,dateOfStart in DATE,
id_position1 in number,salary1 in number,age1 in number)
IS

```



```

id_k NUMBER;
BEGIN
INSERT INTO elizabeth.k_a (telephone,e_mail)
VALUES (telephon,email)
returning id_k_a into id_k;
INSERT INTO
elizabeth.personal(id_k_a,surname_of_pers,name_of_pers,address_of_pers,age
)
VALUES (id_k,surname1,name1,address,age1);
INSERT INTO
elizabeth.AGREEMENT_ON_JOB_PLACEMENT(id_position,id_k_a,salary,
date_of_begin)
VALUES (id_position1,id_k,salary1,dateOfStart);
COMMIT;
END ADD_PERSONAL;
2. Звільнення робітника
create or replace PROCEDURE DEL_PERSONAL(id_k in
number,date_release in date)

IS
--id_k NUMBER;
BEGIN

update agreement_on_job_placement
set date_of_release=date_release
where agreement_on_job_placement.ID_K_A=id_k;
COMMIT;
END DEL_PERSONAL;
3. Переміщення з посади на посаду
create or replace PROCEDURE CHANGEPOS (id_k in number,
dateOfRelease in DATE,name1 IN VARCHAR2,surname1 IN
VARCHAR2,
address IN VARCHAR2 ,telephon in number,email in
varchar2,dateOfStart in DATE,
id_position1 in number,salary1 in number,age1 in number)
IS
id_ka NUMBER;
BEGIN
update agreement_on_job_placement
set date_of_release=dateOfRelease
where agreement_on_job_placement.ID_K_A=id_k;
INSERT INTO elizabeth.k_a (telephone,e_mail)
VALUES (telephon,email)
returning id_k_a into id_ka;

INSERT INTO
elizabeth.personal(id_k_a,surname_of_pers,name_of_pers,address_of_pers,age
)
VALUES (id_ka,surname1,name1,address,age1);
INSERT INTO
elizabeth.AGREEMENT_ON_JOB_PLACEMENT(id_position,id_k_a,salary,
date_of_begin)
VALUES (id_position1,id_ka,salary1,dateOfStart);

```

```

COMMIT;
END CHANGEPOS;
4.Оновлення даних:
create or replace PROCEDURE UPDATE_DATA
(name_of_pers1 IN VARCHAR2,
surname_of_pers1 IN VARCHAR2,
address_of_pers1 IN VARCHAR2 ,
telephone1 in number,
e_mail1 in varchar2,
date_of_begin1 in DATE,
id_position1 in number,
salary1 in number,
AGE1 in number,
ID_KA IN NUMBER)
IS
BEGIN
UPDATE ELIZABETH.K_A
SET telephone=telephone1,
e_mail=e_mail1
WHERE ID_K_A=ID_KA;
UPDATE ELIZABETH.PERSONAL
SET surname_of_pers=surname_of_pers1,
AGE=AGE1,
name_of_pers=name_of_pers1,
address_of_pers=address_of_pers1
WHERE ID_K_A=ID_KA;
UPDATE
ELIZABETH.AGREEMENT_ON_JOB_PLACEMENT
SET id_position=id_position1,
salary=salary1,
date_of_begin=date_of_begin1
WHERE ID_K_A=ID_KA;
COMMIT;
END UPDATE_DATA;

```

```

Триггера:
create or replace TRIGGER TRIGGER1
BEFORE INSERT ON agreement_on_job_placement
for each row
BEGIN
select agree_seq.Nextval
into :new.id_agreement_on_job_plac
from dual;
END;
create or replace TRIGGER TRIGGER2
BEFORE INSERT ON k_a
for each row
BEGIN
select k_a_seq.Nextval
into :new.id_k_a
from dual;
END;

```