МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М.Є. Жуковського

“Харківський авіаційний інститут”

Факультет СУЛА

Кафедра математичного моделювання та штучного інтелекту

**Лабораторна робота №9**

з Організація баз даних

на тему: «SQL + Entity Framework»

Виконав: студент 3 курсу групи №335a

напряму підготовки(спеціальності):

122 Комп’ютерні науки

(шифр і назва напряму підготовки(спеціальності))

Гринюк М.О.

(прізвище й ініціали студента)

Прийняв: ст. викл., к.т.н. Коробчинський К.П.

(посада, науковий ступінь, прізвище й ініціали)

Харків – 2020

**Цель работы:** Изучить принципы работы Entity Framework 6. Взаимодействие с разными подходами.(EF 6 Code-First, EF 6 Data Base First, EF 6 Model First)

**Ход работы:**

1. Пройти тест
2. Использовать в своём проекте класс логирования
3. Показать созданную базу данных на основе модели
4. Подключить проект C# проверить соединение с БД вывести отладочную информацию на консоль.
5. Создать используя EF и реализовать действия:
6. Выполнить по своему варианту запуск всех пользовательских функций;
7. Выполнить по своему варианту запуск всех хранимых процедур;
8. Вывод результатов сделать на форме;
9. Для каждого пункта показать программный код и пример работы (скрин)
10. Оформить отчёт и сохранить на STM
11. Знать ответы на контрольные вопросы.

**Содержание работы:**

1. Для своего варианта на основе построенной логической модели своего варианта создать БД.
2. Подключить класс для вывода логирования.
3. Выполнить соединение с правильными и не правильными данными
4. Создать форму для входа и после успешной авторизации запустить основную, если авторизация не успешная, то главная форма не должна создаваться. (повышенная сложность на максимальное количество баллов)
5. Создать EF и выполнить по ходу работы.

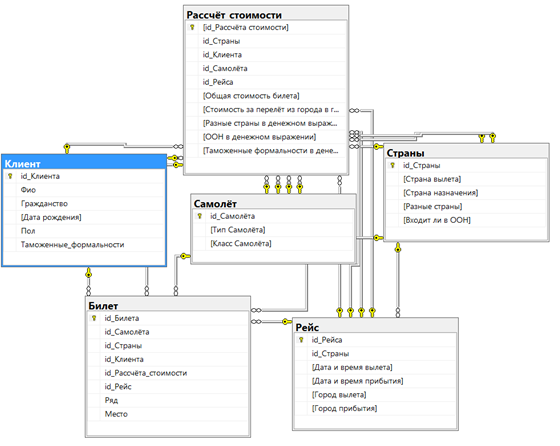
**Запросы:**

* Показать цены на Билеты в заданный город для каждого рейса в салонах каждого класса; Показать, в какие города нельзя долететь из заданного юрода вообще, а также, если сумма наличных денег на покупку Билета меньше заданной
* Показать надбавки на Билеты на заданный рейс (в денежном выражении);
* Показать города, в которые можно долететь из заданного города, имея определенную сумму денег;
* Найти самый дорогой и самый дешевый Билеты для перелета из одного заданною города в другой

**Выполнение работы**

1. Для своего варианта на основе построенной логической модели своего

варианта создать БД;

**Логическая модель БД**

**Подключить класс для вывода логирования**

class ConnectedLayer

{

public string connectionString = ConfigurationManager.ConnectionStrings["MyConnectionString"].ConnectionString;

Logger logger = LogManager.GetCurrentClassLogger();

SqlConnectionStringBuilder builder;

public ConnectedLayer()

{

builder = new SqlConnectionStringBuilder(connectionString);

//ds = new DataSet();

var oSqlCon = new SqlConnection(connectionString);

using (SqlConnection connection = new SqlConnection(connectionString))

{

logger.Trace(

"Connection to" + Environment.NewLine +

"Data Source: " + oSqlCon.DataSource + Environment.NewLine +

"Database: " + oSqlCon.Database + Environment.NewLine +

"State: " + oSqlCon.State +

"User: " + builder.UserID +

"Catalog: " + builder.InitialCatalog +

"Server: " + builder.DataSource

);

Configuration config = ConfigurationManager.OpenExeConfiguration(ConfigurationUserLevel.None);

config.ConnectionStrings.ConnectionStrings.Add(

new ConnectionStringSettings("ConnectionStr", "SomeConnectionString")

);

config.Save();

ConnectionStringsSection section = config.GetSection("connectionStrings") as ConnectionStringsSection;

if (section.SectionInformation.IsProtected)

{

section.SectionInformation.UnprotectSection();

}

else

{

section.SectionInformation.ProtectSection("DataProtectionConfigurationProvider");

}

config.Save();

}

}

public DataSet Read(string command)

{

Logger logger = LogManager.GetCurrentClassLogger();

var oSqlCon = new SqlConnection(connectionString);

builder = new SqlConnectionStringBuilder(connectionString);

oSqlCon.Open();

SqlDataAdapter oSqlDtAdptr = new SqlDataAdapter(command, oSqlCon);

DataSet ds = new DataSet();

oSqlDtAdptr.Fill(ds);

logger.Trace("UserId: " + builder.UserID);

logger.Trace("UserCommand: " + command);

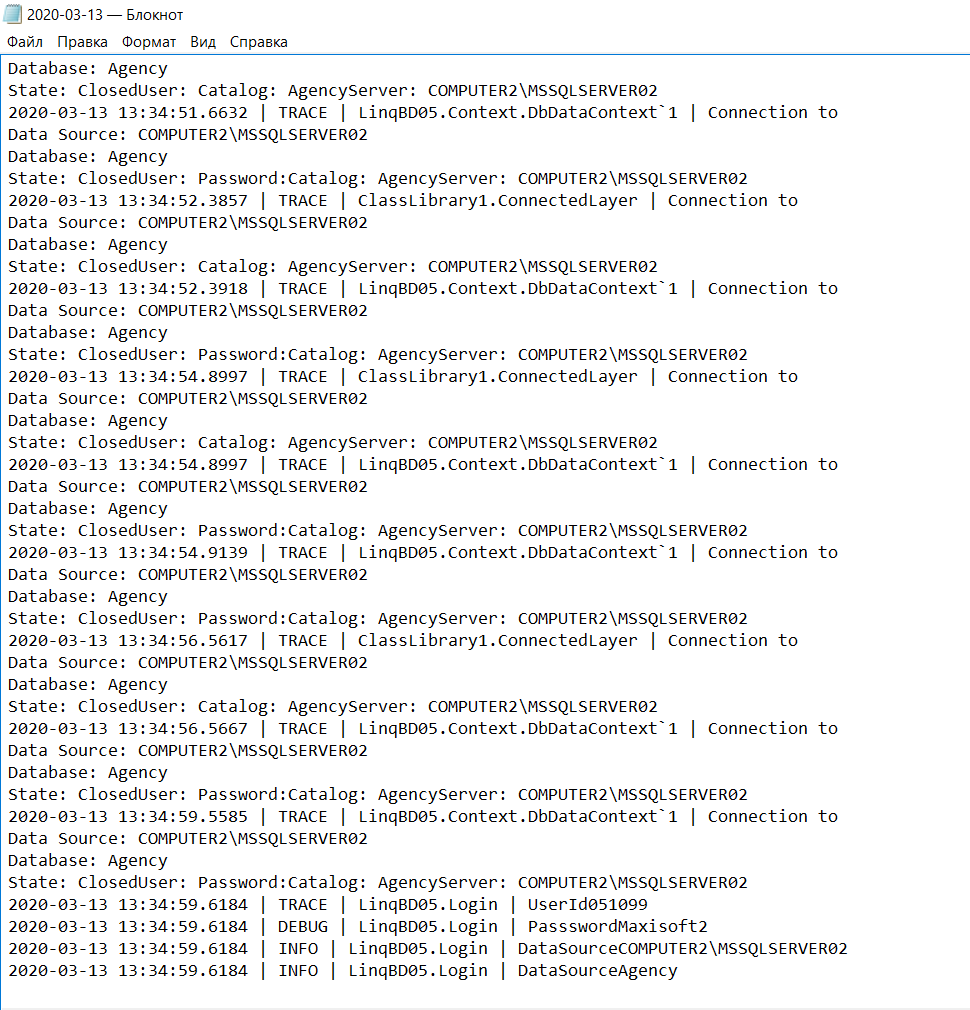
logger.Trace("User's DataSet: " + ds);

oSqlCon.Close();

return ds;

}}

**Выполнить соединение с правильными и не правильными данными**



**Создать форму для входа и после успешной авторизации запустить основную, если авторизация не успешная, то главная форма не должна создаваться.**

**App.config:**

<connectionStrings>

<add name="DefaultConnection" connectionString="Data Source=.\SQLEXPRESS;Initial Catalog=Players;Integrated Security=True; User ID = Maxisoft; Password = 051099;"

providerName="System.Data.SqlClient"/>

</connectionStrings>

**Login.cs:**

private void button1\_Click(object sender, EventArgs e)

{

ConnectedLayer conn = new ConnectedLayer();

SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder(conn.connectionString);

MainForm main = new MainForm();

if (loginBox.Text == builder.UserID && passwordBox.Text == builder.Password)

{

this.Hide();

main.Show();

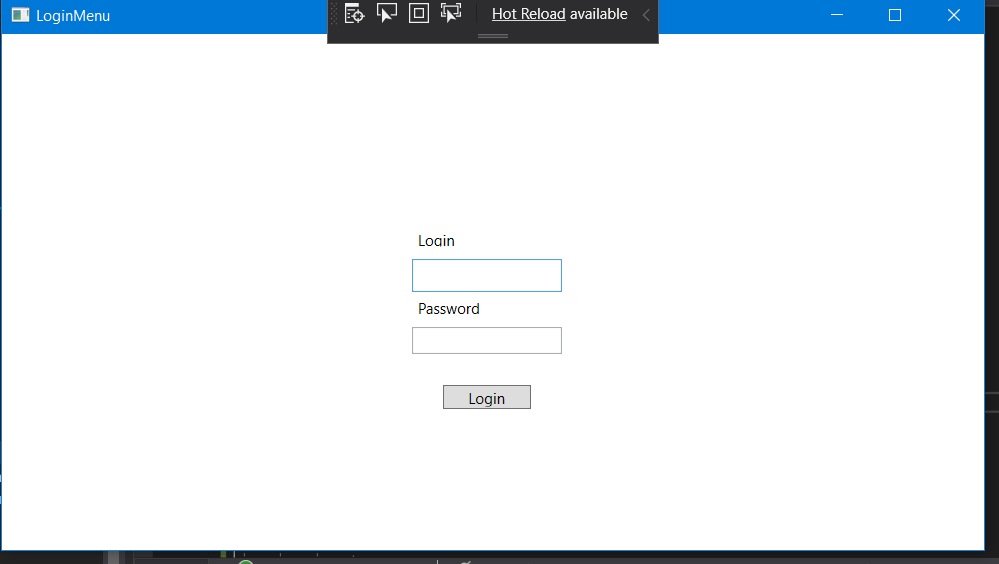
}

else

{

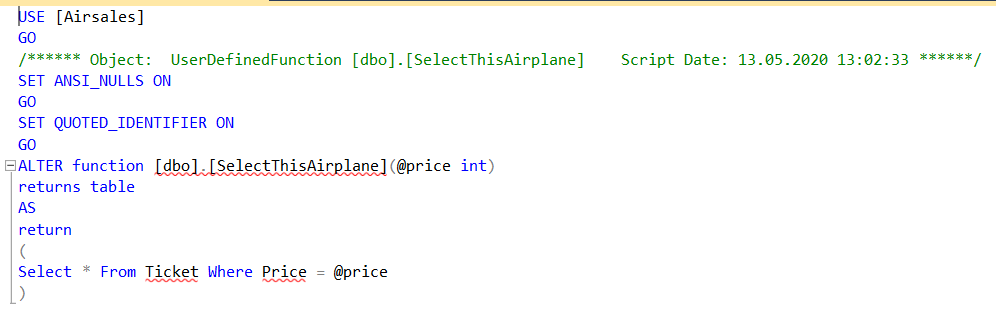
MessageBox.Show("Enter another login");

}}



**Выполнить по своему варианту запуск всех пользовательских функций**

* 1. Показать цены на Билеты в заданный город для каждого рейса в салонах каждого класса; Показать, в какие города нельзя долететь из заданного юрода вообще, а также, если сумма наличных денег на покупку Билета меньше заданной



private void button\_Click(object sender, RoutedEventArgs e)

{

using (TicketContext db = new TicketContext())

{

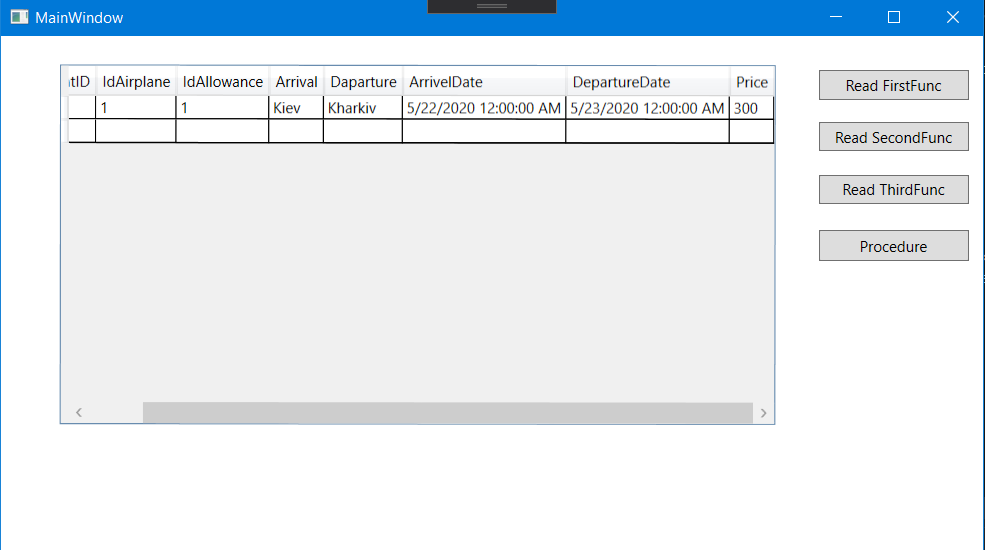
var input = ("Kiev", 200);

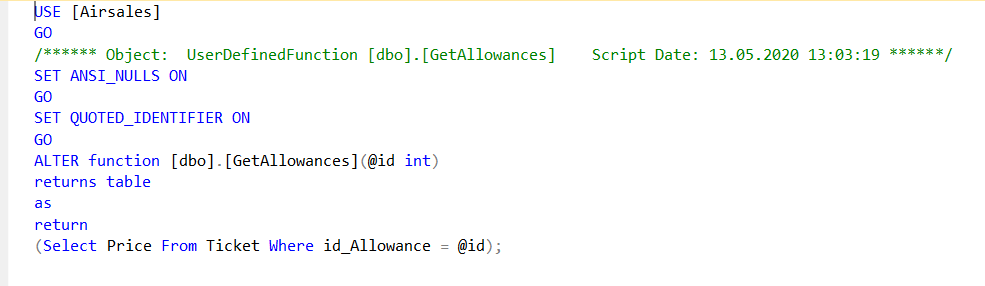
System.Data.SqlClient.SqlParameter param = new System.Data.SqlClient.SqlParameter("@arrival,@price", input);

var phones = db.Database.SqlQuery<Ticket>("SELECT \* FROM FirstFunc (@arrival,@price)", param);

}

}



* 2. Показать надбавки на Билеты на заданный рейс (в денежном выражении);
* 

using (TicketContext db = new TicketContext())

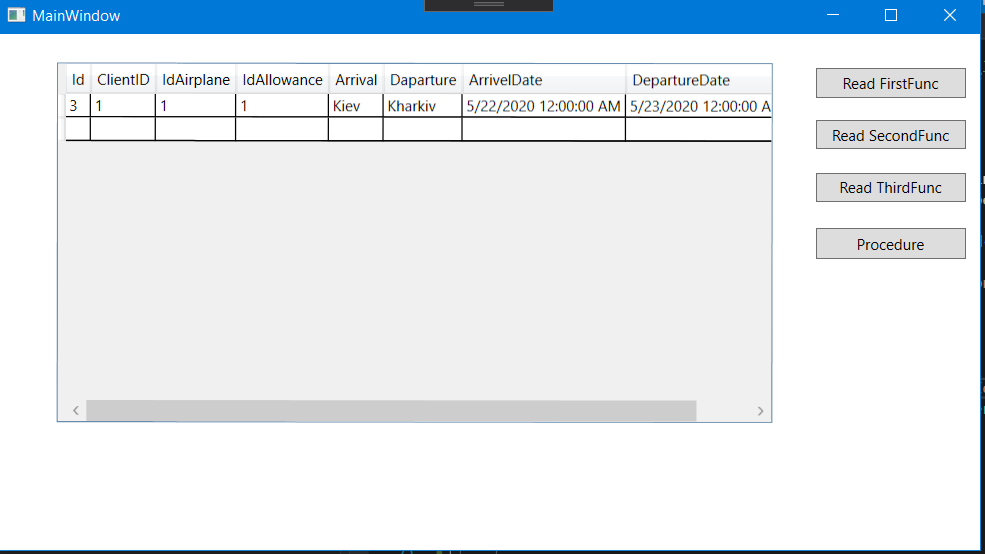
{

System.Data.SqlClient.SqlParameter param = new System.Data.SqlClient.SqlParameter("@id", 1);

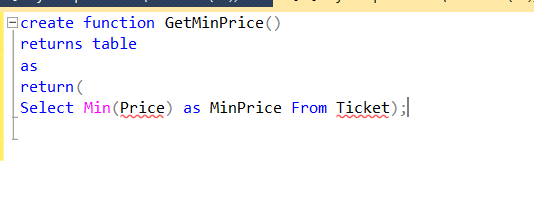
var tickets = db.Database.SqlQuery<Ticket>("SELECT \* FROM GetAllowances (@id)", param);

dataGrid.ItemsSource = tickets;

}



1. Найти самый дорогой и самый дешевый Билеты для перелета из одного заданною города в другой



private void button3\_Click(object sender, RoutedEventArgs e)

{

using (TicketContext db = new TicketContext())

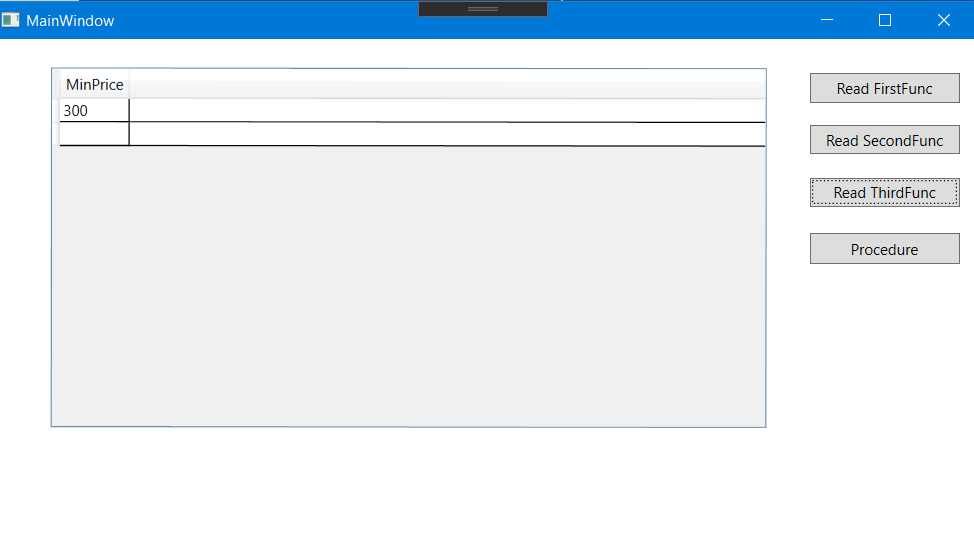
{

var phones = db.Database.SqlQuery<Ticket>(“GetMinPrice

", param);

}

}



**Выполнить по своему варианту запуск всех хранимых процедур**

private void procedure\_Click(object sender, RoutedEventArgs e)

{

using (TicketContext db = new TicketContext())

{

System.Data.SqlClient.SqlParameter param = new System.Data.SqlClient.SqlParameter("@Arrival", "Kiev");

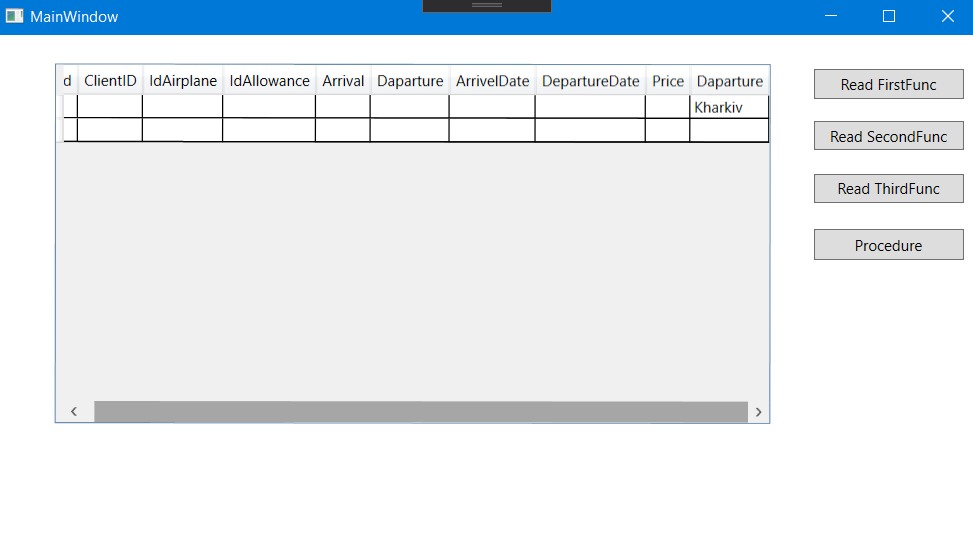
var tickets = db.Database.SqlQuery<Ticket>(" exec SelectDaparture (@Arrival)", param);

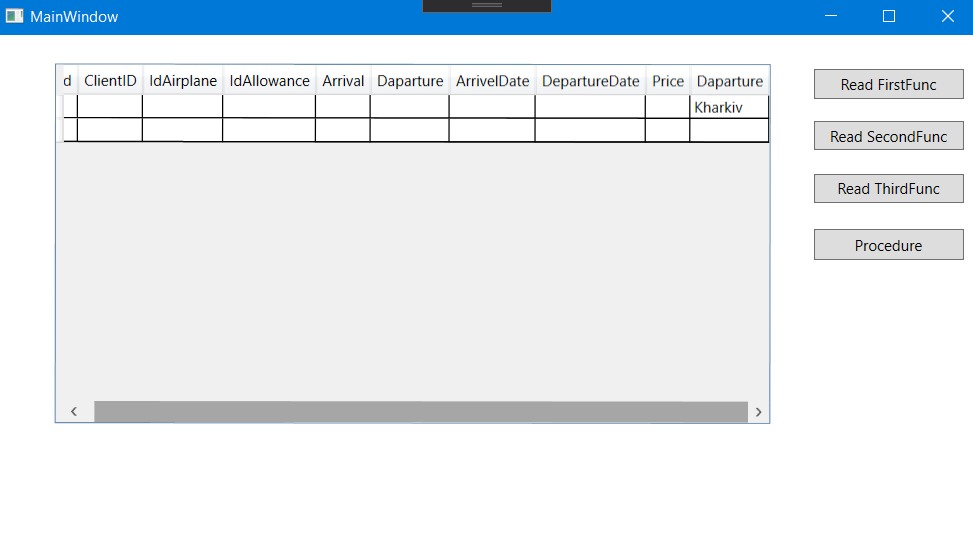
dataGrid.ItemsSource = tickets.ToList();

logger.Info(tickets);

logger.Info(dataGrid.ItemsSource);

}



 }

**Вывод:** Изучили принципы работы Entity Framework 6. Взаимодействие с разными подходами(EF 6 Code-First, EF 6 Data Base First, EF 6 Model First).