МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

ОТЧЕТ

по практической работе №5

«Работа с автономной частью клиента»

по дисциплине

Базы данных

РУКОВОДИТЕЛЬ:

_____ _профессор каф. ВСТ
(подпись) _____Мисевич П. В.__
(фамилия, и.,о.)

СТУДЕНТ:

_____ __Сухоруков В.А.___
(подпись) (фамилия, и.,о.)

_____19-В-2_____
(шифр группы)

Работа защищена «___»_____

С оценкой _____

Нижний Новгород 2021

Цель

Изучить способы работы с автономной частью клиента

Ход выполнения

Создание колонок в таблице

```
DataTable table = new DataTable("MyFirstTable");

DataColumn firstColumn = new DataColumn("First Column", typeof(int));
DataColumn secondColumn = new DataColumn("Second column", typeof(string));

DataColumnCollection columnCollection = table.Columns;
columnCollection.AddRange(new DataColumn[]{firstColumn, secondColumn});

foreach (DataColumn column in table.Columns)
    Console.WriteLine("{0}: {1};", column.ColumnName, column.DataType);
Console.ReadKey();
```

```
First Column: System.Int32;
Second column: System.String;
```

Добавление в колонки строк

```
DataTable table = new DataTable();

table.Columns.Add(new DataColumn("Column1", typeof(int)));
table.Columns.Add(new DataColumn("Column2"));

DataRow newRow = table.NewRow();

newRow["Column1"] = 1; // индексатор объекта DataRow в качестве строкового индекса принимает имя поля в строке к которому нужно обратиться
//newRow[0] = 1;

newRow["COlumn2"] = "One";
//newRow[1] = "One";          // индексатор объекта DataRow в качестве целочисленного индекса принимает индекс поля в строке к которому нужно обратиться

Console.WriteLine("table.Rows.Count: " + table.Rows.Count); // выведется 0

table.Rows.Add(newRow); // строка становится строкой таблицы при добавлении её в коллекцию Rows таблицы

Console.WriteLine("table.Rows.Count: " + table.Rows.Count); // выведется 1

Console.WriteLine();

foreach (DataRow row in table.Rows)
{
    foreach (DataColumn column in table.Columns)
        Console.WriteLine("{0}: {1}", column.ColumnName, row[column]);
    Console.ReadKey();
}
```

```
table.Rows.Count: 0
table.Rows.Count: 1

Column1: 1
Column2: One
```

Создание методов, позволяющих создавать новую таблицу на основе данных, предоставляемых объектом DataReader

```
// This method creates new DataTable with schema same to SqlDataReader
private static DataTable CreateSchemaFromReader(SqlDataReader reader, string tableName)
{
    DataTable table = new DataTable(tableName);

    for (int i = 0; i < reader.FieldCount; i++)
        table.Columns.Add(new DataColumn(reader.GetName(i), reader.GetFieldType(i)));

    return table;
}

// This method write data to DataTable whith same schema as DataReader
private static void WriteDataFromReader(DataTable table, SqlDataReader reader)
{
    while (reader.Read())
    {
        DataRow row = table.NewRow();

        for (int i = 0; i < reader.FieldCount; i++)
            row[i] = reader[i];

        table.Rows.Add(row);
    }
}

static void Main(string[] args)
{
    string conStr = @"Data Source=.\SQLEXPRESS; Initial Catalog=ShopDB; Integrated Security=True;"; // создание строки подключения

    SqlConnection connection = new SqlConnection(conStr);
    connection.Open();

    SqlCommand cmd = new SqlCommand("SELECT * FROM Customers", connection);

    SqlDataReader reader = cmd.ExecuteReader();
    DataTable table = CreateSchemaFromReader(reader, "Customers"); // создание новой таблицы на основе схемы, предоставляемой DataReader

    foreach (DataColumn column in table.Columns)
        Console.WriteLine("{0}: {1}", column.ColumnName, column.DataType);

    WriteDataFromReader(table, reader); // запись данных  в таблицу с помощью DataReader
    Console.WriteLine();

foreach (DataRow row in table.Rows)
{
    foreach (DataColumn column in table.Columns)
        Console.WriteLine("{0}: {1}", column.ColumnName, row[column]);

    Console.WriteLine();
    Console.ReadKey();
}

reader.Close();
connection.Close();
```

```
CustomerNo: System.Int32
FName: System.String
LName: System.String
MName: System.String
Address1: System.String
Address2: System.String
City: System.String
Phone: System.String
DateInSystem: System.DateTime

CustomerNo: 1
FName: Иван*******
LName: Круковский
MName: Петрович
Address1: Лужная 15
Address2:
City: Харьков
Phone: (052)1245789
DateInSystem: 20.11.2009 0:00:00
```

Использование метода GetSchemaTable для получения информации о схеме таблицы к которой обращается объект DataReader

```
string conStr = @"Data Source=.\SQLEXPRESS; Initial Catalog=ShopDB; Integrated Security=True;"; // создание строки подключения

SqlConnection connection = new SqlConnection(conStr);
connection.Open();

SqlCommand cmd = new SqlCommand("SELECT * FROM Customers", connection);

SqlDataReader reader = cmd.ExecuteReader();

DataTable schemaTable = reader.GetSchemaTable(); //получение информации о схеме таблицы Customers

foreach (DataRow row in schemaTable.Rows) // вывод на экран информации, предоставляемой методом GetSchemaTable
{
    foreach (DataColumn column in schemaTable.Columns)
        Console.WriteLine("{0}: {1}", column.ColumnName, row[column]);

    Console.WriteLine();
    Console.ReadKey();
}

DataTable customers = new DataTable("Customers");

foreach (DataRow row in schemaTable.Rows)
{
    var dataColumnToInsert = new DataColumn((string)row["ColumnName"], (Type)row["DataType"]);
    customers.Columns.Add(dataColumnToInsert); // добавление столбцов в таблицу customers
}

Console.WriteLine(new string('-', 20));
foreach (DataColumn customersColumn in customers.Columns)
    Console.WriteLine("{0}: {1}", customersColumn.ColumnName, customersColumn.DataType);  // вывод имен и типов данных столбцов таблицы Customer

reader.Close();
connection.Close();
```

```
ColumnName: CustomerNo
ColumnOrdinal: 0
ColumnSize: 4
NumericPrecision: 10
NumericScale: 255
IsUnique: False
IsKey:
BaseServerName:
BaseCatalogName:
BaseColumnName: CustomerNo
BaseSchemaName:
BaseTableName:
DataType: System.Int32
AllowDBNull: False
ProviderType: 8
IsAliased:
IsExpression:
IsIdentity: True
IsAutoIncrement: True
IsRowVersion: False
IsHidden:
IsLong: False
IsReadOnly: True
ProviderSpecificDataType: System.Data.SqlTypes.SqlInt32
DataTypeName: int
XmlSchemaCollectionDatabase:
XmlSchemaCollectionOwningSchema:
XmlSchemaCollectionName:
UdtAssemblyQualifiedName:
NonVersionedProviderType: 8
IsColumnSet: False
```

```
CustomerNo: System.Int32
FName: System.String
LName: System.String
MName: System.String
Address1: System.String
Address2: System.String
City: System.String
Phone: System.String
DateInSystem: System.DateTime
```

Вывод

В результате данной практической работы были изучены способы работы с автономной частью клиента. Были рассмотрены способы создания колонок и добавления в них данных, создания таблицы на основе данных, предоставляемых объектом DataReader.