**Примеры кода ADO.NET**

В листингах кода на этой странице показано, как получить данные из базы данных с помощью следующих технологий ADO.NET:

***Поставщики данных ADO.NET:***

* SqlClient ( System.Data.SqlClient )
* OLEDB ( System.Data.OleDb )
* ODBC ( System.Data.Odbc )
* OracleClient ( System.Data.OracleClient )

***Платформа ADO.NET Entity Framework:***

* LINQ to Entities
* Типизированный запрос ObjectQuery
* EntityClient ( System.Data.EntityClient )

***LINQ to SQL***

**Примеры поставщика данных ADO.NET**

В приведенных ниже листингах кода демонстрируется извлечение данных из базы данных с помощью поставщиков данных ADO.NET. Данные возвращаются в классе DataReader.

***SqlClient***

Код в этом примере предполагает, что вы можете подключиться к Northwind образцу базы данных на Microsoft SQL Server. Код создает команду SqlCommand для выборки строк из таблицы Products, к которой добавляется параметр SqlParameter, ограничивающий результат строками, для которых значение UnitPrice превышает указанное значение параметра, в данном случае 5. Объект SqlConnection открывается внутри using блока, который гарантирует, что ресурсы закрываются и удаляются при выходе из кода. Команда выполняется с помощью объекта SqlDataReader, а результаты выводятся в окно консоли. Если вы используете, следует рассмотреть возможность обновления до в Microsoft.Data.SqlClient соответствии с System.Data.SqlClient будущими вложениями и новыми разработками функций.

using System;

using System.Data;

using System.Data.SqlClient;

class Program

{

static void Main()

{

string connectionString =

"Data Source=(local);Initial Catalog=Northwind;"

+ "Integrated Security=true";

// Provide the query string with a parameter placeholder.

string queryString =

"SELECT ProductID, UnitPrice, ProductName from dbo.products "

+ "WHERE UnitPrice > @pricePoint "

+ "ORDER BY UnitPrice DESC;";

// Specify the parameter value.

int paramValue = 5;

// Create and open the connection in a using block. This

// ensures that all resources will be closed and disposed

// when the code exits.

using (SqlConnection connection =

new SqlConnection(connectionString))

{

// Create the Command and Parameter objects.

SqlCommand command = new SqlCommand(queryString, connection);

command.Parameters.AddWithValue("@pricePoint", paramValue);

// Open the connection in a try/catch block.

// Create and execute the DataReader, writing the result

// set to the console window.

try

{

connection.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

Console.WriteLine("\t{0}\t{1}\t{2}",

reader[0], reader[1], reader[2]);

}

reader.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Console.ReadLine();

}

}

***OleDb***

Данный образец кода предполагает возможность подключения к образцу базы данных Northwind из Microsoft Access. Код создает команду OleDbCommand для выборки строк из таблицы Products, к которой добавляется параметр OleDbParameter, ограничивающий результат строками, для которых значение UnitPrice превышает указанное значение параметра, в данном случае 5. Соединение OleDbConnection открывается в блоке using, что гарантирует закрытие и освобождение ресурсов после завершения работы кода. Команда выполняется с помощью объекта OleDbDataReader, а результаты выводятся в окно консоли.

using System;

using System.Data;

using System.Data.OleDb;

class Program

{

static void Main()

{

// The connection string assumes that the Access

// Northwind.mdb is located in the c:\Data folder.

string connectionString =

"Provider=Microsoft.Jet.OLEDB.4.0;Data Source="

+ "c:\\Data\\Northwind.mdb;User Id=admin;Password=;";

// Provide the query string with a parameter placeholder.

string queryString =

"SELECT ProductID, UnitPrice, ProductName from products "

+ "WHERE UnitPrice > ? "

+ "ORDER BY UnitPrice DESC;";

// Specify the parameter value.

int paramValue = 5;

// Create and open the connection in a using block. This

// ensures that all resources will be closed and disposed

// when the code exits.

using (OleDbConnection connection =

new OleDbConnection(connectionString))

{

// Create the Command and Parameter objects.

OleDbCommand command = new OleDbCommand(queryString, connection);

command.Parameters.AddWithValue("@pricePoint", paramValue);

// Open the connection in a try/catch block.

// Create and execute the DataReader, writing the result

// set to the console window.

try

{

connection.Open();

OleDbDataReader reader = command.ExecuteReader();

while (reader.Read())

{

Console.WriteLine("\t{0}\t{1}\t{2}",

reader[0], reader[1], reader[2]);

}

reader.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Console.ReadLine();

}

}

***ODBC***

Данный образец кода предполагает возможность подключения к образцу базы данных Northwind из Microsoft Access. Код создает команду OdbcCommand для выборки строк из таблицы Products, к которой добавляется параметр OdbcParameter, ограничивающий результат строками, для которых значение UnitPrice превышает указанное значение параметра, в данном случае 5. Объект OdbcConnection открывается внутри using блока, который гарантирует, что ресурсы закрываются и удаляются при выходе из кода. Команда выполняется с помощью объекта OdbcDataReader, а результаты выводятся в окно консоли.

using System;

using System.Data;

using System.Data.Odbc;

class Program

{

static void Main()

{

// The connection string assumes that the Access

// Northwind.mdb is located in the c:\Data folder.

string connectionString =

"Driver={Microsoft Access Driver (\*.mdb)};"

+ "Dbq=c:\\Data\\Northwind.mdb;Uid=Admin;Pwd=;";

// Provide the query string with a parameter placeholder.

string queryString =

"SELECT ProductID, UnitPrice, ProductName from products "

+ "WHERE UnitPrice > ? "

+ "ORDER BY UnitPrice DESC;";

// Specify the parameter value.

int paramValue = 5;

// Create and open the connection in a using block. This

// ensures that all resources will be closed and disposed

// when the code exits.

using (OdbcConnection connection =

new OdbcConnection(connectionString))

{

// Create the Command and Parameter objects.

OdbcCommand command = new OdbcCommand(queryString, connection);

command.Parameters.AddWithValue("@pricePoint", paramValue);

// Open the connection in a try/catch block.

// Create and execute the DataReader, writing the result

// set to the console window.

try

{

connection.Open();

OdbcDataReader reader = command.ExecuteReader();

while (reader.Read())

{

Console.WriteLine("\t{0}\t{1}\t{2}",

reader[0], reader[1], reader[2]);

}

reader.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Console.ReadLine();

}

}

}

***OracleClient***

Данный пример кода предполагает наличие соединения с базой данных DEMO.CUSTOMER на сервере Oracle. Кроме того, необходимо добавить ссылку на файл System.Data.OracleClient.dll. Этот код возвращает данные в объекте OracleDataReader.

using System;

using System.Data;

using System.Data.OracleClient;

class Program

{

static void Main()

{

string connectionString =

"Data Source=ThisOracleServer;Integrated Security=yes;";

string queryString =

"SELECT CUSTOMER\_ID, NAME FROM DEMO.CUSTOMER";

using (OracleConnection connection =

new OracleConnection(connectionString))

{

OracleCommand command = connection.CreateCommand();

command.CommandText = queryString;

try

{

connection.Open();

OracleDataReader reader = command.ExecuteReader();

while (reader.Read())

{

Console.WriteLine("\t{0}\t{1}",

reader[0], reader[1]);

}

reader.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

}

}

**Примеры Entity Framework**

В приведенных ниже листингах кода демонстрируется извлечение данных из источника данных путем выполнения запросов к сущностям модели EDM. В этих примерах используется модель, основанная на образце базы данных Northwind.

***LINQ to Entities***

В данном примере кода запрос LINQ используется для возврата данных в виде объектов Categories, которые проецируются в анонимный тип, содержащий только свойства CategoryID и CategoryName.

using System;

using System.Linq;

using System.Data.Objects;

using NorthwindModel;

class LinqSample

{

public static void ExecuteQuery()

{

using (NorthwindEntities context = new NorthwindEntities())

{

try

{

var query = from category in context.Categories

select new

{

categoryID = category.CategoryID,

categoryName = category.CategoryName

};

foreach (var categoryInfo in query)

{

Console.WriteLine("\t{0}\t{1}",

categoryInfo.categoryID, categoryInfo.categoryName);

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

}

}

***Типизированный запрос ObjectQuery***

В данном примере кода для возврата данных в виде объектов Categories используется запрос ObjectQuery<T>.

using System;

using System.Data.Objects;

using NorthwindModel;

class ObjectQuerySample

{

public static void ExecuteQuery()

{

using (NorthwindEntities context = new NorthwindEntities())

{

ObjectQuery<Categories> categoryQuery = context.Categories;

foreach (Categories category in

categoryQuery.Execute(MergeOption.AppendOnly))

{

Console.WriteLine("\t{0}\t{1}",

category.CategoryID, category.CategoryName);

}

}

}

}

***EntityClient***

В данном примере кода для выполнения запроса Entity SQL используется команда EntityCommand. Этот запрос возвращает список записей, представляющих экземпляры типа сущности Categories. Для доступа к записям данных в результирующем наборе используется объект EntityDataReader.

using System;

using System.Data;

using System.Data.Common;

using System.Data.EntityClient;

using NorthwindModel;

class EntityClientSample

{

public static void ExecuteQuery()

{

string queryString =

@"SELECT c.CategoryID, c.CategoryName

FROM NorthwindEntities.Categories AS c";

using (EntityConnection conn =

new EntityConnection("name=NorthwindEntities"))

{

try

{

conn.Open();

using (EntityCommand query = new EntityCommand(queryString, conn))

{

using (DbDataReader rdr =

query.ExecuteReader(CommandBehavior.SequentialAccess))

{

while (rdr.Read())

{

Console.WriteLine("\t{0}\t{1}", rdr[0], rdr[1]);

}

}

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

}

}

***LINQ to SQL***

В данном примере кода запрос LINQ используется для возврата данных в виде объектов Categories, которые проецируются в анонимный тип, содержащий только свойства CategoryID и CategoryName. Этот пример основан на контексте данных Northwind.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using Northwind;

class LinqSqlSample

{

public static void ExecuteQuery()

{

using (NorthwindDataContext db = new NorthwindDataContext())

{

try

{

var query = from category in db.Categories

select new

{

categoryID = category.CategoryID,

categoryName = category.CategoryName

};

foreach (var categoryInfo in query)

{

Console.WriteLine("vbTab {0} vbTab {1}",

categoryInfo.categoryID, categoryInfo.categoryName);

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

}

}