Retrieving data with SqliteDataReader

The `SqliteDataReader` is a class used to retrieve data from the database. It is used with the `SqliteCommand` class to execute an SQL `SELECT` statement and then access the returned rows. It provides fast, forward-only, read-only access to query results. It is the most efficient way to retrieve data from tables.

Rather than using a constructor, we create an instance of the `SqliteDataReader` by calling the `ExecuteReader()` method of the `SqliteCommand` object. While the `SqlDataReader` is being used, the associated `SqlConnection` serves the `SqlDataReader`. No other operations can be performed on the `SqlConnection` other than closing it.

```
Option Strict On

Imports Mono.Data.Sqlite

Module Example

    Sub Main()

        Dim cs As String = "URI=file:test.db"

        Using con As New SqliteConnection(cs)
            con.Open()

            Using cmd As New SqliteCommand(con)

                cmd.CommandText = "SELECT * FROM Cars LIMIT 5"

                Dim rdr As SqliteDataReader = cmd.ExecuteReader()

                Using rdr
                    While (rdr.Read())
                        Console.WriteLine(rdr.GetInt32(0) & " " _
                            & rdr.GetString(1) & " " & rdr.GetInt32(2))
                    End While
                End Using
            End Using

            con.Close()
        End Using

    End Sub

End Module
```

We get 5 cars from the `Cars` table and print them to the console.

```
Dim rdr As SqliteDataReader = cmd.ExecuteReader()
```

To create an `SQLiteDataReader` object, we must call the `ExecuteReader()` method of the `SqliteCommand` object.

```
While (rdr.Read())
    Console.WriteLine(rdr.GetInt32(0) & " " _
        & rdr.GetString(1) & " " & rdr.GetInt32(2))
End While
```

The `Read()` method advances the data reader to the next record. It returns true if there are more rows; otherwise false. We can retrieve the value using the array index notation, or use a specific method to access column values in their native data types. The latter is more efficient.

```
$ mono retrieve.exe
1 Audi 52642
2 Mercedes 57127
```

```
3 Skoda 9000
4 Volvo 29000
5 Bentley 350000
```

The first five rows of the `Cars` table.

We can retrieve the fields by their column names.

```
Option Strict On

Imports Mono.Data.Sqlite

Module Example

    Sub Main()

        Dim cs As String = "URI=file:test.db"

        Using con As New SqliteConnection(cs)
            con.Open()

            Using cmd As New SqliteCommand(con)
                cmd.CommandText = "SELECT * FROM Cars LIMIT 5"

                Dim rdr As SqliteDataReader = cmd.ExecuteReader()

                Using rdr
                    While (rdr.Read())
                        Console.Write("{0} ", rdr("Id"))
                        Console.Write("{0} ", rdr("Name"))
                        Console.WriteLine("{0} ", rdr("Price"))
                    End While
                End Using

            End Using
            con.Close()
        End Using

    End Sub

End Module
```

The example prints 5 rows from the `Cars` table. This time we use the column names to get the table fields.

```
While (rdr.Read())
    Console.Write("{0} ", rdr("Id"))
    Console.Write("{0} ", rdr("Name"))
    Console.WriteLine("{0} ", rdr("Price"))
End While
```

The database table fields are referenced by their column names.

Multiple statements

The ADO.NET specification allows to execute multiple statements in a single string. In case of queries, the `SqliteDataReader` returns multiple result sets. It has the `NextResult()` method to navigate through the result sets.

```
Option Strict On

Imports Mono.Data.Sqlite

Module Example

    Sub Main()

        Dim cs As String = "URI=file:test.db"

        Using con As New SqliteConnection(cs)
            con.Open()

            Using cmd As New SqliteCommand(con)
                cmd.CommandText = "SELECT 25; SELECT 44; SELECT 33"

                Dim rdr As SqliteDataReader = cmd.ExecuteReader()

                Using rdr
```

```
                    Do
                        rdr.Read()
                        Console.WriteLine("{0}", rdr.GetInt32(0))
                    Loop While rdr.NextResult()
                End Using
            End Using

            con.Close()
        End Using

    End Sub

End Module
```

We have three queries in one SQL string. There will be three result sets.

```
cmd.CommandText = "SELECT 25; SELECT 44; SELECT 33"
```

There are three `SELECT` statements. They are separated by the semicolon character. Each of them will return a single value.

```
Do
    rdr.Read()
    Console.WriteLine("{0}", rdr.GetInt32(0))

Loop While rdr.NextResult()
```

The `Read()` method advances the `SqliteDataReader` to the next record. The `GetInt32()` method retrieves the value as a 32-bit signed integer. The `NextResult()` advances the data reader to the next result.

```
$ mono multiple.exe
25
44
33
```

Running the example.

We have finished reading data with the `SqliteDataReader`.

Home Contents Top of Page Previous Next