목 차

Ⅰ. **UDS 개요** **2**
UDS 개요 3
UDS 애트리뷰트 5
환경 설정 9

Ⅱ. **데이터 질의문 인터페이스 사용 방법** **10**
기본 UDS 11
오즈 서버 Connection을 사용하는 UDS 20

Ⅲ. **DataAction 인터페이스 사용 방법** **25**

Ⅳ. **패러미터 넘겨주는 인터페이스 사용 방법** **42**

Ⅴ. **HttpRequest 접근을 위한 인터페이스 사용 방법** **48**

Ⅵ. **IDataReader 사용 방법** **54**

본 매뉴얼은 오즈 서버, 오즈 애플리케이션 디자이너, 오즈 리포트 디자이너/뷰어, 오즈 쿼리 디자이너 및 Visual Studio를 사용할 수 있는 중급 이상 개발자를 대상으로 작성된 매뉴얼입니다.

Ⅰ. UDS 개요

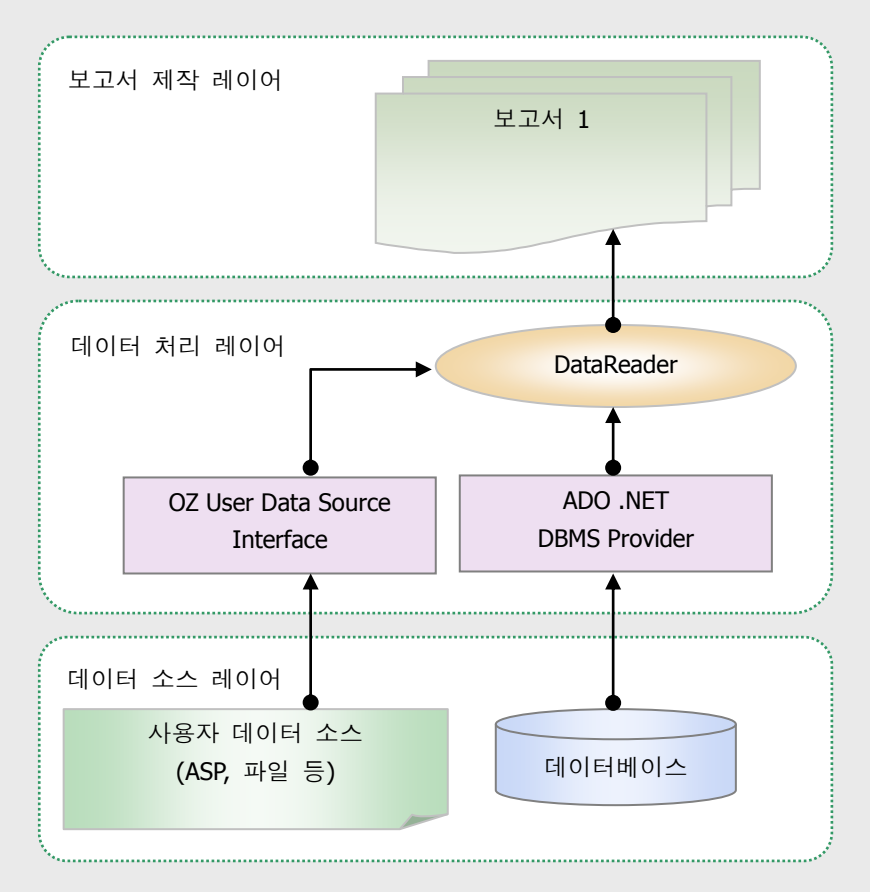
- UDS 개요
- UDS 애트리뷰트
- 환경 설정

UDS 개요

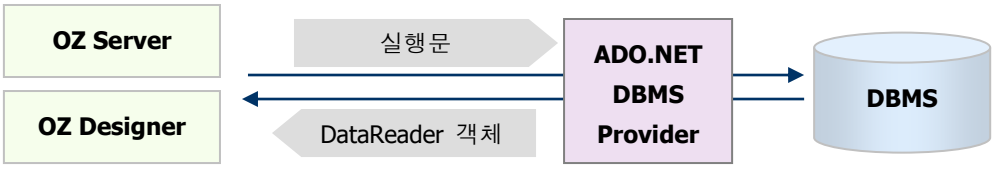
UDS(User Data Store)는 Non-DB 데이터를 연동하기 위한 ADO .NET Interface로 데이터베이스에 저장되어 있지 않은 일반 텍스트 파일, CSV, XML 등의 파일 데이터는 물론 다양한 형태의 외부 데이터 소스로부터 데이터 셋을 생성하여 보고서를 생성할 수 있도록하는 오즈만의 인터페이스입니다.

UDS 개발자는 다양한 형태의 사용자 데이터를 받아들이는 부분과 이 데이터를 오즈가 인식할 수 있는 DataReader 객체로 만들어서 오즈로 넘겨주는 부분을 구현해야 합니다. 이렇게 구현된 프로그램은 파일로 만들어 보고서와 연동할 수 있으며, 별도의 설정없이 일반 SQL 데이터 셋과 동일하게 사용할 수 있습니다.

오즈는 사용자 클래스로 패러미터 스트링을 전달할 수 있으며, 패러미터 값에 따라 유동적으로 변화하는 데이터를 전달받을 수 있습니다.

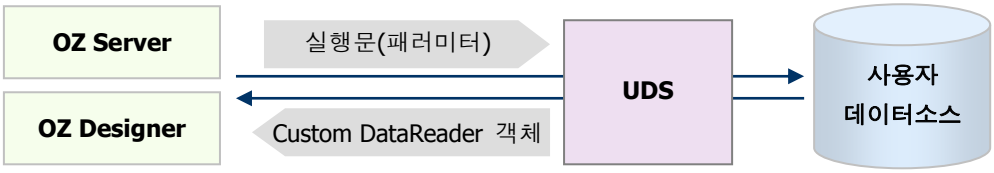


ADO.NET DBMS Provider를 사용하여 데이터를 처리할 경우 오즈 서버는 ADO.NET DBMS Provider를 통해 DBMS에 접속한 후 쿼리문을 전달하고 ADO.NET DBMS Provider는 DBMS로부터 전달받은 결과 데이터를 DataReader 객체로 오즈 디자이너로 전달합니다.



이때 DataReader 객체는 System.Data.IDataReader Interface를 구현한 것이어야 합니다. 또한 DBMS 벤더가 제공하는 ADO.NET DBMS Provider는 System.Data.IDataReader Interface를 구현한 Concrete DataReader 클래스를 함께 제공하여야 하며, System.Data.IDataReader Interface에 따라 해당 기능을 구현하여야 동작합니다.

UDS를 사용하면 UDS가 ADO.NET DBMS Provider 역할을 하므로, UDS를 통해 사용자 데이터 소스에 쉽게 접근하여 결과 데이터를 가져올 수 있습니다.



UDS 애트리뷰트

UDS 애트리뷰트를 사용하여 UDS를 구현할 경우 사용되는 함수의 패러미터는 인터페이스에 정의된 함수의 패러미터와 동일할 경우에만 동작합니다.

UDS 애트리뷰트를 설정할 때 postfix 형태의 애트리뷰트는 Visual Studio에서 자동 인식되므로 생략 가능합니다.

UDS 애트리뷰트

■ oz.uds.DataStoreAttribute

Definition	해당 클래스가 UDS임을 나타내는 속성입니다.
Target	Class
Argument	*dataSourceType* 제공하는 데이터 타입

Sample	[oz.uds.DataStore(typeof(IDataReader))]
	public class UDS{
	}

■ oz.uds.InitAttribute

Definition	UDS를 초기화하는 함수의 속성입니다
Target	Method

Sample	[oz.uds.Init]
	public void Initialize(){
	}

■ oz.uds.CloseAttribute

Definition	UDS를 종료하는 함수의 속성입니다.
Target	Method

Sample	[oz.uds.Close]
	public void Destruct(){
	}

■ oz.uds.GetDataAttribute

Definition	데이터를 가져오는 함수의 속성입니다.
Target	Method

Sample	[oz.uds.GetData]
	public IDataReader RetrieveData(string command){
	}

■ **oz.uds.ReleaseDataAttribute**

Definition	가져온 데이터의 리소스를 해제하는 함수의 속성으로 `DataStoreAttribute`의 `dataSourceType`에 설정한 타입의 데이터를 패러미터로 전달 받습니다.
Target	Method

Sample	[oz.uds.ReleaseData]
	public void FinalizeData(IDataReader reader){
	}

■ **oz.uds.InsertAttribute**

Definition	행 삽입 데이터 액션을 수행하는 함수의 속성입니다.
Target	Method

Sample	[oz.uds.Insert]
	public string InsertData(string command, OZDACData[] data, string extraArgs, Hashtable parameters){
	}

■ **oz.uds.DeleteAttribute**

Definition	행 삭제 데이터 액션을 수행하는 함수의 속성입니다.
Target	Method

Sample	[oz.uds.Delete]
	public string DeleteData(string command, OZDACData[] data, string extraArgs, Hashtable parameters){
	}

■ **oz.uds.UpdateAttribute**

Definition	행 변경 데이터 액션을 수행하는 함수의 속성입니다.
Target	Method

Sample	[oz.uds.Update]
	public string UpdateAttribute(string command, OZDACData[] data1, OZDACData[] data2, string extraArgs, Hashtable parameters){
	}

■ **oz.uds.CommitAttribute**

Definition	데이터 액션을 가져와서 queue에 넣고 queue를 서버로 전송하는 함수의 속성입니다.
Target	Method

Sample	[oz.uds.Commit]
	public string CommitChanges(){
	}

■ **oz.uds.RollbackAttribute**

Definition	데이터 액션을 취소하는 함수의 속성입니다.
Target	Method
Sample	public string DropChanges(){ }

■ **oz.uds.SetHttpContextAttribute**

Definition	HttpContext를 설정하는 함수의 속성입니다.
Target	Method, Property

Sample	[oz.uds.SetHttpContext]
	public void SetContext(HttpContext context){
	}
	또는
	[oz.uds.SetHttpContext]
	public HttpContextContext{set{_ctx = value;}}

■ **oz.uds.SetLoggerAttribute**

Definition	로그 내용을 기록하는 함수의 속성입니다.
Target	Method, Property

Sample	[oz.uds.SetLogger]
	public void SetLogger(oz.framework.log.OZLog log){
	}
	또는
	[oz.uds.SetLogger]
	public OZLog Logger{set{log = value;}}

■ **oz.uds.SetParameterAttribute**

Definition	사용자 지정 패러미터를 설정하는 함수의 속성입니다.
Target	Method, Property

Sample	[oz.uds.SetParameter]
	public void SetParameter(Hashtable parameters){
	}
	또는
	[oz.uds.SetParameter]
	public Hashtable{set{_parameters = value;}}

■ **oz.uds.GetAliasesAttribute**

Definition	오즈 서버의 DB 커넥션 풀 참조 시 DB 앨리어스 이름을 리스트 형태로 전달하는 함수의 속성입니다.
Target	Method, Property

Sample	[oz.uds.GetAliases]
	public string[] GetAliases(){
	}
	또는
	[oz.uds.GetAliases]
	public string[] Aliases{get{return new string[]{"alias", ...}}

■ **oz.uds.SetConnections**

Definition	GetAliasesAttribute 속성에 정의된 DB 앨리어스의 DB 커넥션 풀을 설정하는 함수의 속성입니다.
Target	Method, Property
Sample	[oz.uds.SetConnections] public void SetConnections(IDictionary connections){ 또는 [oz.uds.SetConnections] public IDictionary Connections{set{_connections = value;}}

환경 설정

본 매뉴얼에 있는 예제를 실행해보기 위해서는 다음과 같은 환경 설정이 필요합니다.

■ DLL 파일 등록

UDS 기능이 구현된 DLL 파일(OZUDSSample_Csharp.dll, OZUDSSample_VBNET.dll)을 클라이언트 프로그램이 설치된 경로의 lib 폴더에 복사하고, assembly.properties 파일을 편집기로 열어 DLL 파일의 경로를 추가합니다.

```
...

uds_sample_csharp.filepath=%OZ_HOME%/../lib/OZUDSSample_Csharp.dll
uds_sample_vbnet.filepath=%OZ_HOME%/../lib/OZUDSSample_VBNET.dll
```

■ ODBC 등록

본 매뉴얼의 예제에 사용될 AppExample.mdb를 ODBC 데이터 원본 관리자의 [시스템 DSN] 탭에 "AppExample" 이름으로 등록합니다.

■ 오즈 서버 설정

bin 폴더에 OZUDSSample_Csharp.dll, OZUDSSample_VBNET.dll 파일을 추가합니다.

오즈 서버의 conf 폴더에 있는 db.properties 파일에 ODBC 정보를 추가합니다.

```
AppExample.vendor=odbc
AppExample.dsn=AppExample
AppExample.user=
AppExample.password=
AppExample.maxconns=5
AppExample.initconns=2
AppExample.timeout=5

Member.vendor=odbc
Member.dsn=AppExample
Member.user=
Member.password=
Member.maxconns=5
Member.initconns=2
Member.timeout=5
```

Ⅱ. 데이터 질의문 인터페이스 사용 방법

- 기본 UDS
- 오즈 서버 Connection을 사용하는 UDS

기본 UDS

기본 UDS인 DefaultUserDataStore 클래스는 OZUserDataStore와 RawDataSource 인터페이스 상속과 함께 로그 내용을 표시하는 기능이 추가된 클래스입니다. UDS를 사용할 경우에는 DefaultUserDataStore 클래스를 상속받아 사용해야 합니다.

```
abstract class OZUserDataReaderStore
abstract class OZUserDataTableStore
```

■ oz.uds.OZUserDataReaderStore 클래스

```
namespace oz.uds
{
   public abstract class OZUserDataReaderStore
   {
      protected readonly oz.framework.log.OZLog log;
      abstract public void Init();
      abstract public IDataReader GetDataReader(string command);
      abstract public void FreeDataReader(IDataReader reader);
      abstract public void Close();
   }
}
```

- Init

Prototype	`public void Init()`
Definition	사용자 데이터 셋을 초기화합니다. UDS가 처음 생성된 후 한번 호출합니다.

- GetDataReader

Prototype	`public IDataReader GetDataReader(string command)`
Definition	오즈 서버나 디자이너로부터 실행문을 넘겨받아 해당 실행문에 맞는 사용자 데이터를 IDataReader 객체로 변환하여 반환합니다.
Argument	*command* 결과 셋 생성을 위한 실행문

- FreeDataReader

Prototype	`public void FreeDataReader(IDataReader reader)`
Definition	IDataReader를 종료합니다. 오즈 서버나 디자이너에서 GetDataReader()로 전송 받은 IDataReader을 모두 사용한 후에는 FreeDataReader() 함수를 호출하여 IDataReader을 종료하여야 합니다.
Argument	*reader* 종료할 IDataReader 객체

- Close

Prototype	`public void Close()`
Definition	사용자 데이터 셋을 종료합니다. UDS를 모두 사용한 후 마지막으로 호출하여야 합니다.

■ oz.uds.OZUserDataTableStore 클래스

```
namespace oz.uds
{
   public abstract class OZUserDataTableStore
   {
      protected readonly oz.framework.log.OZLog log;
      abstract public void Init();
      abstract public System.Data.DataTable GetDataTable(string command);
      abstract public void FreeDataTable(DataTable table);
      abstract public void Close();
   }
}
```

- Init

Prototype	`public void Init()`
Definition	사용자 데이터 셋을 초기화합니다. UDS가 처음 생성된 후 한번 호출합니다.

- GetDataReader

Prototype	`public System.Data.DataTable GetDataReader(string command)`
Definition	오즈 서버나 디자이너로부터 실행문을 넘겨받아 해당 실행문에 맞는 사용자 데이터를 DataTable 객체로 변환하여 반환합니다.
Argument	*command* DataTable 생성을 위한 실행문

- FreeDataReader

Prototype	`public void FreeDataTable(DataTable table)`
Definition	DataTable을 종료합니다. 오즈 서버나 디자이너에 GetDataTable()로 받은 DataTable을 모두 사용한 후에는 FreeDataTable() 메소드를 호출하여 DataTable을 종료하여야 합니다.
Argument	*table* 종료할 DataTable 객체

- Close

Prototype	`public void Close()`
Definition	사용자 데이터 셋을 종료합니다. UDS를 모두 사용한 후 마지막으로 호출하여야 합니다.

UserDataReaderStoreSample.cs

```
using System;
using System.Web;
using System.Data;
using System.Collections;

using oz.uds;
using oz.uds.dr;

namespace oz.uds.sample.csharp
{
   public    class    UserDataReaderStoreSample    :    OZUserDataReaderStore,
IHttpContextRef, IParameterRef
   {
      private HttpContext _ctx;
      private IDictionary _params;

      public HttpContext HttpContext{ set{ _ctx = value; } }

      public IDictionary Parameters{ set{ _params = value; } }

      public override void Init()
      {
      }

      public override IDataReader GetDataReader(string command)
      {
         int                              fieldCount                              =
```

```
oz.util.OZString.ParseInt(Convert.ToString(_params["field_count"]), 5);
          int                                rowCount                                =
oz.util.OZString.ParseInt(Convert.ToString(_params["row_count"]), 5);

          string[] fieldNames = new string[fieldCount];
          for(int i = 0; i < fieldCount; i++)
             fieldNames[i] = "field" + i;

          string[,] data = new string[rowCount, fieldCount];

          for(int rowIndex = 0; rowIndex < rowCount; rowIndex++)
          {
             for(int colIndex = 0; colIndex < fieldCount; colIndex++)
             {
                data[rowIndex,  colIndex]  =  "value  "  +  rowIndex  +  ",  "  +
colIndex;
             }
          }

          return new ArrayDataReader(fieldNames, data);
       }

       public override void FreeDataReader(IDataReader reader)
       {
          if(null != reader)
          {
             reader.Close();
             reader.Dispose();
          }
       }

       public override void Close()
       {
       }
    }
}
```

- UserDataReaderStoreSample.vb

```
Imports System.Web
Imports System.Collections
Imports oz.uds.dr

Public Class UserDataReaderStoreSample
```

```
   Inherits OZUserDataReaderStore
   Implements IParameterRef

   Private _params As IDictionary

   Public WriteOnly Property Parameters() As IDictionary Implements
IParameterRef.Parameters
      Set(ByVal value As IDictionary)
         Me._params = value
      End Set
   End Property

   Public Overrides Sub Init()
   End Sub

   Public Overrides Function GetDataReader(ByVal command As String) As
IDataReader

      Dim fieldCount As Integer =
oz.util.OZString.ParseInt(Convert.ToString(_params.Item("field_count")), 5)
      Dim rowCount As Integer =
oz.util.OZString.ParseInt(Convert.ToString(_params.Item("row_count")), 5)

      Dim fieldNames(fieldCount - 1) As String
      Dim i As Integer
      For i = 0 To fieldCount - 1
         fieldNames(i) = "field" & i
      Next

      Dim data(rowCount - 1, fieldCount - 1) As String

      Dim rowIndex, colIndex As Integer

      For rowIndex = 0 To rowCount - 1
         For colIndex = 0 To fieldCount - 1
            data(rowIndex, colIndex) = "value " & rowIndex & ", " & colIndex
         Next
      Next

      Return New ArrayDataReader(fieldNames, data)
   End Function

   Public Overrides Sub FreeDataReader(ByVal reader As IDataReader)
      If (Not reader Is Nothing) Then
```

```
            reader.Close()
            reader.Dispose()
        End If
    End Sub

    Public Overrides Sub Close()
    End Sub

End Class
```

- UserDataTableStoreSample.cs

```
using System;
using System.Web;
using System.Data;
using System.Collections;

using oz.uds;
using oz.uds.dr;

namespace oz.uds.sample.csharp
{
    public class UserDataTableStoreSample : OZUserDataTableStore, IParameterRef
    {
        private IDictionary _params;
        public IDictionary Parameters{ set{ _params = value; } }

        public override void Init()
        {
        }

        public override DataTable GetDataTable(string command)
        {
            int                             fieldCount                              =
oz.util.OZString.ParseInt(Convert.ToString(_params["field_count"]), 5);
            int                              rowCount                               =
oz.util.OZString.ParseInt(Convert.ToString(_params["row_count"]), 5);

            log.Debug("Start to create DataTable object. ");
            DataTable table = new DataTable();
            for(int i = 0; i < fieldCount; i++)
                table.Columns.Add("field" + i);

            for(int rowIndex = 0; rowIndex < rowCount; rowIndex++)
```

```
            {
                DataRow row = table.NewRow();
                for(int colIndex = 0; colIndex < fieldCount; colIndex++)
                {
                    row[colIndex] = "value " + rowIndex + ", " + colIndex;
                }

                table.Rows.Add(row);
            }

            return table;
        }

        public override void FreeDataTable(DataTable table)
        {
            if(null != table)
                table.Dispose();
        }

        public override void Close()
        {
        }
    }
}
```

- UserDataTableStoreSample.vb

```
Imports System.Web
Imports System.Collections
Imports oz.uds.dr

Public Class UserDataTableStoreSample
    Inherits OZUserDataTableStore
    Implements IParameterRef

    Private _params As IDictionary

    Public WriteOnly Property Parameters() As IDictionary Implements
IParameterRef.Parameters
        Set(ByVal value As IDictionary)
            Me._params = value
        End Set
    End Property
```

```
    Public Overrides Function GetDataTable(ByVal command As String) As DataTable

        Dim             fieldCount             As             Integer             =
oz.util.OZString.ParseInt(Convert.ToString(_params.Item("field_count")), 5)
        Dim             rowCount             As             Integer             =
oz.util.OZString.ParseInt(Convert.ToString(_params.Item("row_count")), 5)

        log.Debug("Start to create DataTable object. ")

        Dim table As New DataTable

        Dim i As Integer
        For i = 0 To fieldCount
            table.Columns.Add("field" & i)
        Next

        Dim rowIndex, colIndex As Integer

        For rowIndex = 0 To rowCount
            Dim row As DataRow = table.NewRow()
            For colIndex = 0 To fieldCount
                row.Item(colIndex) = "value " & rowIndex & ", " & colIndex
            Next
            table.Rows.Add(row)
        Next

        Return table

    End Function

    Public Overrides Sub FreeDataTable(ByVal table As DataTable)
        If (Not table Is Nothing) Then
            table.Dispose()
        End If
    End Sub

    Public Overrides Sub Init()
    End Sub

    Public Overrides Sub Close()
    End Sub

End Class
```

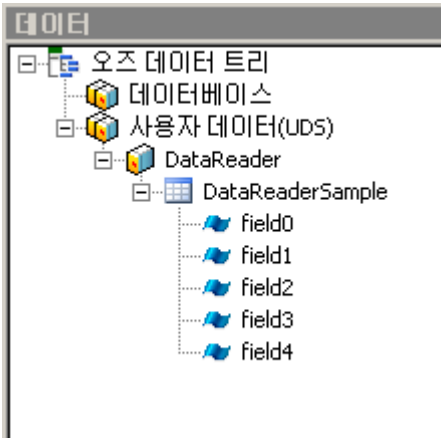
■ 적용 예

기본 UDS의 데이터를 보고서에 출력하는 예제를 살펴보겠습니다.

➢ ODI 만들기

쿼리 디자이너를 실행하여 "DataReader"로 사용자 데이터 스토어를 추가하고, "DataReaderSample"로 데이터 셋을 추가합니다.

쿼리문 입력 창에 실행문을 입력하고 쿼리문 실행 아이콘( )을 클릭하여 필드 리스트를 생성합니다.



ODI 파일을 "stores.odi"로 저장합니다.

➢ 보고서 디자인하기

리포트 디자이너를 실행하여 보고서를 새로 만든 후 "stores.odi" 파일을 추가합니다.

보고서에 데이터 밴드를 추가하고 데이터 밴드의 "ODI 이름" 속성을 "stores"로, "데이터셋" 속성을 "DataReaderSample"로 설정합니다.

데이터 트리 창에서 "DataReaderSample"의 "field1" 필드를 선택한 후 데이터 밴드로 드래그&드롭하여 추가합니다.

➢ 미리보기

툴바의 미리보기 아이콘을 클릭하여 보고서를 미리보기합니다.

오즈 서버 Connection을 사용하는 UDS

UDS 클래스에서 오즈 서버의 DB Connection Pool로부터 여러 개의 Connection을 공유해서 사용하기 위해서는 oz.uds.IConnectionRef 인터페이스를 구현해야 합니다.

■ oz.uds.IConnectionRef 인터페이스

```
public interface IConnectionRef
{
   string[] Aliases{get;}

   IDictionary Connections{set;}
}
```

- Aliases

Prototype	`string[] Aliases{get;}`
Definition	UDS가 사용할 DB Pool의 앨리어스 이름입니다.

- Connections

Prototype	`IDictionary Connections{set;}`
Definition	UDS가 사용할 Connection이 담겨 있는 Dictionary Connection입니다. Aliases를 통해 전달된 문자열을 Key로 Connection가지며, 해당 연결이 서버 Connection Pool에 존재하지 않는 경우 null(VB : Nothing)입니다.

※ 참고사항 : 서버에서 메소드 호출 순서

① Aliases{get;} (해당 Connection을 DB Pool에서 찾기)

② Connections{set;} (검색된 Connection을 UDS에 설정)

③ Init (초기화)

■ 제약 사항

- IConnectionRef 인터페이스를 구현한 클래스에서 사용하는 Connection은 서버의 Pool에서 관리되므로 Close() 메소드를 호출하여도 연결이 해제되지 않습니다.
- IConnectionRef 인터페이스를 구현한 클래스는 서버와 쿼리 디자이너에서 초기화하는 방법이 각각 다르므로 Init(), Close()를 주의해서 작성해야 합니다.

- 일반적으로 Init()에서 Connection을 생성하던 UDS와 달리 IConnectionRef 인터페이스를 구현한 UDS는 Connection을 Connections{set;}에 의해 서버로부터 받습니다.
- UDS가 서버에서 생성될 때에는 Connections{set;} → Init() 순서로 호출됩니다.

ConnectionReferenceSample.cs

```
using System;
using System.IO;
using System.Web;
using System.Data;
using System.Collections;

using oz.uds;
using oz.uds.dr;
using oz.framework.dac;

namespace oz.uds.sample.csharp
{
   public class ConnectionReferenceSample : OZUserDataReaderStore,
IConnectionRef
   {
      private IDbConnection _connection;
      private IDbCommand _command;

      public override void Init()
      {
         if(null == _connection)
            throw new OZUDSException("Cannot find proper connection from oz
connection pool.");

         _command = _connection.CreateCommand();
      }

      public override IDataReader GetDataReader(string command)
      {
         _command.CommandText = command;
         return _command.ExecuteReader();
      }

      public override void FreeDataReader(IDataReader reader)
      {
         if(null != reader)
```

```
         {
            reader.Close();
            reader.Dispose();
         }
      }

      public override void Close()
      {
      }

      public System.Collections.IDictionary Connections
      { set{ _connection = (IDbConnection)value["appexample"]; } }

      public string[] Aliases
      { get{ return new string[]{ "appexample"}; } }
   }
}
```

- **ConnectionReferenceSample.vb**

```
Imports System.IO
Imports System.Web
Imports System.Collections

Imports oz.uds.dr
Imports oz.framework.dac

Public Class ConnectionReferenceSample
   Inherits OZUserDataReaderStore
   Implements IConnectionRef

   Private _connection As IDbConnection
   Private _command As IDbCommand

   Public Overrides Sub Init()
      If (_connection Is Nothing) Then
         Throw New OZUDSException("Cannot find proper connection from oz
connection pool.")
      End If

      _command = _connection.CreateCommand()
   End Sub

   Public Overrides Function GetDataReader(ByVal command As String) As
```

```
IDataReader
       _command.CommandText = command
       Return _command.ExecuteReader()
   End Function

   Public Overrides Sub FreeDataReader(ByVal reader As IDataReader)
       If (Not reader Is Nothing) Then
          reader.Close()
          reader.Dispose()
       End If
   End Sub

   Public Overrides Sub Close()
   End Sub

   Public    ReadOnly    Property    Aliases()    As    String()    Implements
IConnectionRef.Aliases
       Get
          Return New String() {"appexample"}
       End Get
   End Property

   Public  WriteOnly  Property  Connections()  As  System.Collections.IDictionary
Implements IConnectionRef.Connections
       Set(ByVal Value As System.Collections.IDictionary)
          _connection = DirectCast(Value.Item("appexample"), IDbConnection)
       End Set
   End Property
End Class
```

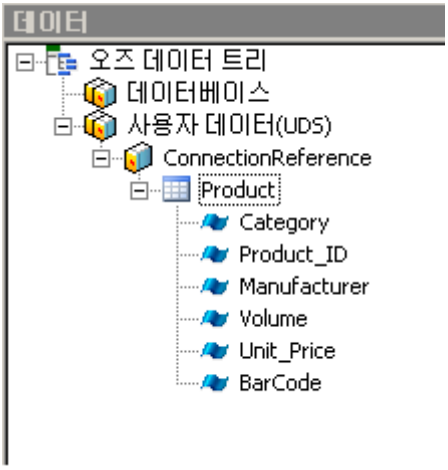
■ 적용 예

오즈 서버의 Connection을 여러 개 사용하는 UDS의 데이터를 보고서에 출력하는 예제를 살펴보겠습니다.

- ODI 만들기

  쿼리 디자이너를 실행하여 "ConnectionReference"로 사용자 데이터 스토어를 추가하고, "Product"로 데이터 셋을 추가합니다.

  쿼리문 입력 창에 실행문을 입력하고 쿼리문 실행 아이콘()을 클릭하여 필드 리스트를 생성합니다.



ODI 파일을 "connectionreference.odi"로 저장합니다.

- 보고서 디자인하기

  리포트 디자이너를 실행하여 보고서를 새로 만든 후 "connectionreference.odi" 파일을 추가합니다.

  보고서에 데이터 밴드를 추가하고 데이터 밴드의 "ODI 이름" 속성을 "connectionreference"로, "데이터셋" 속성을 "Product"로 설정합니다.

  데이터 밴드에 테이블을 추가하고 테이블에 표시할 필드를 추가합니다.

  디자인한 보고서를 "connectionreference.ozr"로 저장한 후 ODI와 OZR을 서버 리파지토리에 업로드합니다.

- 웹에서 보고서 미리보기

  보고서를 호출하는 웹 페이지를 작성한 후 웹 브라우저에서 웹 페이지를 실행하여 보고서를 바인딩합니다.

Ⅲ. DataAction 인터페이스 사용 방법

UDS에서 DataAction을 처리하기 위해서는 oz.uds.IDataAction 인터페이스를 구현하여야 합니다.

Init() 함수에서 트랜잭션 개체를 생성한 경우 해당 트랜잭션 개체는 반드시 반영되거나 삭제되어야 합니다.
IDbTransaction.Rollback(), IDbTransaction.Commit(), IDbConnection.BeginTransaction()을 통해 생성된 트랜잭션이 종료되지 않은 채로 UDS 처리가 끝나는 경우 DB 연결을 사용하는 다음 요청에서 트랜잭션 처리 오류가 발생하게 되므로 트랜잭션 처리 시 주의하시기 바랍니다.

■ **oz.uds.IDataAction 인터페이스**

```
namespace oz.uds
{
   public interface IDataAction
   {
      /// <summary>
      /// insert a row.
      ///
      /// it can be implemented as following SQL like logic.
      ///
      /// INSERT INTO 'table' (source[0].FieldName, source[1].FieldName, ...)
      /// VALUES(source[0].FieldData, source[1].FieldData, ...)
      ///
      /// extra arugment can be used to convert spacial types (TO_DATE etc) or
      /// modify data.
      /// </summary>
      /// <param name="command">command string</param>
      /// <param name="source">source field data</param>
      /// <param name="extra"> extra argument if needs.</param>
      /// <param name="parameters">field names and values of parameter and
master set</param>
      /// <returns>number of inserted rows. (always 1 if normal)</returns>
      string InsertRow(string command, OZDACData[] source,
         string extra, Hashtable parameters);

      /// <summary>
      /// delete row(s).
      ///
      /// it can be implemented as following SQL like logic.
      ///
      ///    DELETE    FROM    'table'    WHERE    condition[0].FieldName    =
condition[0].FieldData
```

```
        /// AND condition[1].FieldName = condition[1].FieldData
        /// AND ...
        /// AND extra condition
        /// </summary>
        /// <param name="command">command string</param>
        /// <param name="condition">condition field data</param>
        /// <param name="extra">extra condition if needs.</param>
        ///  <param name="parameters">field names and values of parameter and
master set</param>
        /// <returns>number of deleted rows.</returns>
        string DeleteRow(string command, OZDACData[] condition,
            string extra, Hashtable parameters);

        /// <summary>
        /// it can be implemented as following SQL like logic.
        ///
        /// UPDATE 'table' SET source[0].FieldName = source[0].FieldData,
        /// source[1].FieldName = source[1].FieldData,
        /// ...
        /// WHERE condition[0].FieldName = condition[0].FieldData
        /// AND condition[1].FieldName = condition[1].FieldData
        /// AND ...
        /// AND ext
        /// extra arugment can be used to convert spacial types (TO_DATE etc) or
        /// modify data.
        /// </summary>
        /// <param name="command">command string</param>
        /// <param name="condition">target  field data</param>
        /// <param name="source">source field data</param>
        /// <param name="extra">extra argument if needs.</param>
        ///  <param name="parameters">field names and values of parameter and
master set</param>
        /// <returns>number of updated rows.</returns>
        string UpdateRow(string command, OZDACData[] condition, OZDACData[]
source,
            string extra, Hashtable parameters);

        /// <summary>
        /// commit all data actions
        /// called after all data action transactions are successfully completed.
        /// </summary>
        /// <returns>commit result message</returns>
        string Commit();
```

```
        /// <summary>
        /// rollback all data actions
        /// called if exception occured while doing data actions.
        /// </summary>
        void Rollback();
    }
}
```

- InsertRow

Prototype	`string InsertRow(string command, OZDACData[] source, string extra, Hashtable parameters)`	
Definition	행을 삽입하고 데이터를 추가합니다.	
Argument	*command*	행 삽입할 쿼리문
	source	행 삽입할 필드 데이터
	extra	추가할 여분의 내용
	parameters	마스터 셋 패러미터의 필드 이름과 값(<string dataSetName, IDictionary parameters> 형태의 값이 들어감)

- DeleteRow

Prototype	`string DeleteRow(string command, OZDACData[] condition, string extra, Hashtable parameters)`	
Definition	행을 삭제합니다.	
Argument	*command*	행 삭제할 쿼리문
	condition	행 삭제할 필드 데이터
	extra	추가할 여분의 내용
	parameters	마스터 셋 패러미터의 필드 이름과 값(<string dataSetName, IDictionary parameters> 형태의 값이 들어감)

- UpdateRow

Prototype	`string UpdateRow(string command, OZDACData[] condition, OZDACData[] source, string extra, Hashtable parameters)`	
Definition	행을 변경합니다.	
Argument	*command*	행 변경할 쿼리문
	condition	행 변경할 대상 필드의 정보
	source	행 변경할 필드에 삽입할 데이터 정보

	extra	추가할 여분의 내용
	parameters	마스터 셋 패러미터의 필드 이름과 값(<string dataSetName, IDictionary parameters> 형태의 값이 들어감)

- Commit

Prototype	`string Commit()`
Definition	변경 내역을 반영합니다.

- Rollback

Prototype	`void Rollback()`
Definition	변경 내역을 폐기합니다.

DataActionSample.cs

```
using System;
using System.IO;
using System.Web;
using System.Data;
using System.Collections;

using oz.uds;
using oz.uds.dr;
using oz.framework.dac;

namespace oz.uds.sample.csharp
{
   public class DataActionSample : OZUserDataReaderStore, IConnectionRef,
IDataAction
   {
      private IDbConnection _connection;
      private IDbCommand _command;

      public override void Init()
      {
         if(null == _connection)
            throw new OZUDSException("Cannot find proper connection from oz
connection pool.");

         _command = _connection.CreateCommand();
         _command.Transaction = _connection.BeginTransaction();
```

```
    }

    public override IDataReader GetDataReader(string command)
    {
        _command.CommandText = command;
        return _command.ExecuteReader();
    }

    public override void FreeDataReader(IDataReader reader)
    {
        if(null != reader)
        {
            reader.Close();
            reader.Dispose();
        }
    }

    public override void Close()
    {
        if(null != _command.Transaction)
        {
            _command.Transaction.Rollback();
            _command.Transaction = null;
        }
    }

    public System.Collections.IDictionary Connections
    { set{ _connection = (IDbConnection)value["appexample"]; } }

    public string[] Aliases
    { get{ return new string[]{ "appexample"}; } }

    private void makeString(TextWriter writer, IDictionary parameters)
    {
        foreach(DictionaryEntry entry in parameters)
        {
            string masterSetName = (string) entry.Key;
            IDictionary masterFields = (IDictionary)entry.Value;

            writer.WriteLine("Master set; " + masterSetName);

            foreach(DictionaryEntry masterField in masterFields)
            {
                writer.Write("[" + masterField.Key + ":" + masterField.Value +
```

```
"]");
            }
         }
      }

      public string InsertRow(string command, OZDACData[] source, string
extraArgument, Hashtable parameters)
      {
         StringWriter writer = new StringWriter();
         writer.WriteLine("Testing data action - InsertRow ------------------
---------------");
         writer.WriteLine("Command : " + command);
         foreach(OZDACData field in source)
         {
            writer.WriteLine("Source field " + field.FieldName + " : " +
field.FieldData);
         }
         writer.WriteLine("Extra argument : " + extraArgument);
         makeString(writer, parameters);
         writer.WriteLine("-------------------------------------------------
--------");
         log.Debug(writer.ToString());

         _command.CommandText = command;
         return "Insert : " + _command.ExecuteNonQuery();
      }

      public string DeleteRow(string command, OZDACData[] destination, string
extraArgument, Hashtable parameters)
      {
         StringWriter writer = new StringWriter();
         writer.WriteLine("Testing data action - DeleteRow ------------------
---------------");
         writer.WriteLine("Command : " + command);
         foreach(OZDACData field in destination)
         {
            writer.WriteLine("Destination field " + field.FieldName + " : " +
field.FieldData);
         }
         writer.WriteLine("Extra argument : " + extraArgument);
         makeString(writer, parameters);
         writer.WriteLine("-------------------------------------------------
--------");
         log.Debug(writer.ToString());
```

```
            _command.CommandText = command;
            return "Delete : " + _command.ExecuteNonQuery();
        }

        public string UpdateRow(string command, OZDACData[] destination,
OZDACData[] source, string extraArgument, Hashtable parameters)
        {
            StringWriter writer = new StringWriter();
            writer.WriteLine("Testing data action - UpdateRow ----------
---------------");
            writer.WriteLine("Command : " + command);
            foreach(OZDACData field in destination)
            {
                writer.WriteLine("Destination field " + field.FieldName + " : " +
field.FieldData);
            }
            foreach(OZDACData field in source)
            {
                writer.WriteLine("Source field " + field.FieldName + " : " +
field.FieldData);
            }
            writer.WriteLine("Extra argument : " + extraArgument);
            makeString(writer, parameters);
            writer.WriteLine("----------------------------------------------------
--------");
            log.Debug(writer.ToString());

            _command.CommandText = command;
            return "Update : " + _command.ExecuteNonQuery();
        }

        public void Rollback()
        {
            if(null != _command.Transaction)
            {
                _command.Transaction.Rollback();
                _command.Transaction = null;
            }
        }

        public string Commit()
        {
            if(null != _command.Transaction)
```

```
            {
                _command.Transaction.Commit();
                _command.Transaction = null;
            }
            return "Commit";
        }
    }
}
```

- DataActionSample.vb

```
Imports System.IO
Imports System.Web
Imports System.Collections

Imports oz.uds.dr
Imports oz.framework.dac

Public Class DataActionSample
    Inherits OZUserDataReaderStore
    Implements IDataAction, IConnectionRef

    Private _connection As IDbConnection
    Private _command As IDbCommand

    Public Overrides Sub Init()
        If (_connection Is Nothing) Then
            Throw New OZUDSException("Cannot find proper connection from oz
connection pool.")
        End If

        _command = _connection.CreateCommand()
        _command.Transaction = _connection.BeginTransaction()
    End Sub

    Public Overrides Function GetDataReader(ByVal command As String) As
IDataReader
        _command.CommandText = command
        Return _command.ExecuteReader()
    End Function

    Public Overrides Sub FreeDataReader(ByVal reader As IDataReader)
        If (Not reader Is Nothing) Then
            reader.Close()
```

```
            reader.Dispose()
        End If
    End Sub

    Public Overrides Sub Close()
        If (Not _command.Transaction Is Nothing) Then
            _command.Transaction.Rollback()
            _command.Transaction = Nothing
        End If
    End Sub

    Public ReadOnly Property Aliases() As String() Implements
IConnectionRef.Aliases
        Get
            Return New String() {"appexample"}
        End Get
    End Property

    Public WriteOnly Property Connections() As System.Collections.IDictionary
Implements IConnectionRef.Connections
        Set(ByVal Value As System.Collections.IDictionary)
            _connection = DirectCast(Value.Item("appexample"), IDbConnection)
        End Set
    End Property

    Private Sub makeString(ByVal writer As TextWriter, ByVal parameters As
IDictionary)
        Dim entry As DictionaryEntry
        For Each entry In parameters
            Dim masterSetName As String = CStr(entry.Key)
            Dim masterFields As IDictionary = DirectCast(entry.Value,
IDictionary)
            writer.WriteLine(("Master set; " & masterSetName))
            Dim masterField As DictionaryEntry
            For Each masterField In masterFields
                writer.Write(String.Concat(New Object() {"[", masterField.Key, ":",
masterField.Value, "]"}))
            Next
        Next
    End Sub

    Public Function InsertRow(ByVal command As String, ByVal source() As
OZDACData, ByVal extraArgument As String, ByVal parameters As Hashtable) As
String Implements IDataAction.InsertRow
```

```
      Dim writer As New StringWriter
      writer.WriteLine("Testing data action - InsertRow ----------------------
-----------")
      writer.WriteLine(("Command : " & command))
      Dim field As OZDACData
      For Each field In source
         writer.WriteLine("Source  field  "  &  field.FieldName,  "  :  "  &
field.FieldData)
      Next

      writer.WriteLine(("Extra argument : " & extraArgument))
      makeString(writer, parameters)
      writer.WriteLine("------------------------------------------------------
----")
      log.Debug(writer.ToString)
      _command.CommandText = command
      Return ("Insert : " & _command.ExecuteNonQuery)

   End Function

   Public Function UpdateRow(ByVal command As String, ByVal destination() As
OZDACData, ByVal source() As OZDACData, ByVal extraArgument As String, ByVal
parameters As Hashtable) As String Implements IDataAction.UpdateRow

      Dim writer As New StringWriter
      writer.WriteLine("Testing data action - UpdateRow ----------------------
-----------")
      writer.WriteLine(("Command : " & command))

      Dim field As OZDACData
      For Each field In destination
         writer.WriteLine("Destination  field  " & field.FieldName & " : " &
field.FieldData)
      Next
      For Each field In source
         writer.WriteLine("Source  field  "  &  field.FieldName  &  "  :  "  &
field.FieldData)
      Next

      writer.WriteLine(("Extra argument : " & extraArgument))
      makeString(writer, parameters)
      writer.WriteLine("------------------------------------------------------
----")
```

```
        log.Debug(writer.ToString)
        _command.CommandText = command
        Return ("Update : " & _command.ExecuteNonQuery)

    End Function

    Public Function DeleteRow(ByVal command As String, ByVal destination() As
OZDACData, ByVal extraArgument As String, ByVal parameters As Hashtable) As
String Implements IDataAction.DeleteRow

        Dim writer As New StringWriter
        writer.WriteLine("Testing data action - DeleteRow ----------------------
-----------")
        writer.WriteLine(("Command : " & command))
        Dim field As OZDACData
        For Each field In destination
            writer.WriteLine("Destination field " & field.FieldName & " : " &
field.FieldData)
        Next

        writer.WriteLine(("Extra argument : " & extraArgument))
        makeString(writer, parameters)
        writer.WriteLine("------------------------------------------------------
----")
        log.Debug(writer.ToString)
        _command.CommandText = command
        Return ("Delete : " & _command.ExecuteNonQuery)

    End Function

    Public Sub Rollback() Implements IDataAction.Rollback

        If (Not _command.Transaction Is Nothing) Then
            _command.Transaction.Rollback()
            _command.Transaction = Nothing
        End If

    End Sub

    Public Function Commit() As String Implements IDataAction.Commit

        If (Not _command.Transaction Is Nothing) Then
            _command.Transaction.Commit()
            _command.Transaction = Nothing
```

```
      End If
      Return "Commit"

   End Function
End Class
```

■ 적용 예

오즈 애플리케이션의 테이블을 이용하여 데이터를 삽입, 삭제, 변경하는 예제를 살펴보겠습니다.

➢ ODI 만들기

쿼리 디자이너를 실행하여 "DataAction"으로 사용자 데이터 스토어를 추가하고, "MASTER"로 데이터 셋을 추가합니다.

쿼리문 입력 창에 아래와 같이 입력하고

```
select distinct country
from ECustomer
```

쿼리문 실행 아이콘( )을 클릭하여 필드 리스트를 생성합니다.



"DataAction"에 "DETAIL"로 데이터 셋을 추가한 후 "마스터셋" 속성을 "MASTER"로 설정합니다. "DETAIL" 데이터 셋의 쿼리문 입력 창에 아래와 같이 입력하고

```
select customer_id, fname, lname, member_card, gender, phone1, city
from ECustomer
where ECustomer.country = '#MASTER.country#'
```

쿼리문 실행 아이콘( )을 클릭하여 필드 리스트를 생성합니다.



"DETAIL" 데이터 셋에 아래와 같이 행 삽입, 삭제, 변경 쿼리문을 입력합니다.

- 행 삽입 쿼리

```
INSERT INTO ECustomer (
  #@ARG_SF1#,  #@ARG_SF2#,
  #@ARG_SF3#,  #@ARG_SF4#,
  #@ARG_SF5#,  #@ARG_SF6#,
  #@ARG_SF7#,  country
)
VALUES (
  #@ARG_SV1#,  '#@ARG_SV2#',
  '#@ARG_SV3#',  '#@ARG_SV4#',
  '#@ARG_SV5#',  '#@ARG_SV6#',
  '#@ARG_SV7#',  '#MASTER.country#'
)
```

- 행 삭제 쿼리

```
DELETE FROM ECustomer
WHERE #@ARG_DF1# = #@ARG_DV1# and country = '#MASTER.country#'
```

- 행 변경 쿼리

```
UPDATE ECustomer SET
  #@ARG_SF2# = '#@ARG_SV2#',  #@ARG_SF3# = '#@ARG_SV3#',
  #@ARG_SF4# = '#@ARG_SV4#',  #@ARG_SF5# = '#@ARG_SV5#',
  #@ARG_SF6# = '#@ARG_SV6#',  #@ARG_SF7# = '#@ARG_SV7#'
WHERE
  #@ARG_DF1# = #@ARG_DV1# and  country = '#MASTER.country#'
```

ODI 파일을 "dataaction.odi" 파일로 저장합니다.

- 폼 디자인하기

  애플리케이션 디자이너를 실행하여 프로젝트를 새로 만든 후 "dataaction.odi" 파일을 추가합니다.

  보드에 콤보박스를 추가하고 콤보박스의 "ODIKey" 속성을 "dataaction"으로, "DataSet" 속성을 "MASTER"로, "Field" 속성을 "country"로, "FireRowCursorChange" 속성을 "True"로 설정합니다.

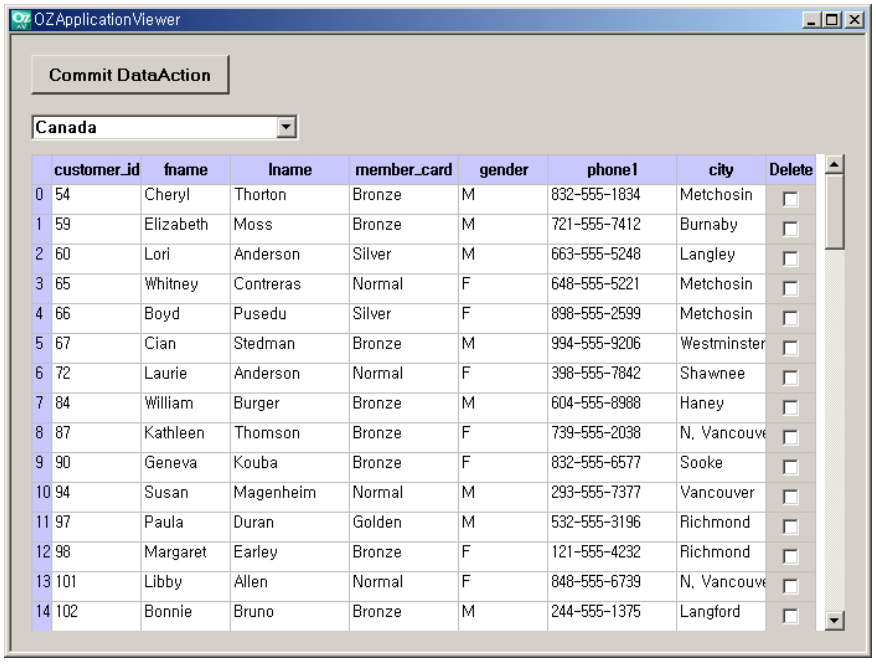
  보드에 테이블을 추가하고 테이블의 "ODIKey" 속성을 "dataaction"으로, "DataSet" 속성을 "DETAIL"로, "AllowInsert", "AllowDelete", "AllowUpdate" 속성을 "True"로, "CellSelectionMode" 속성을 "Single"로 설정합니다.
  테이블 마법사를 실행하여 "DETAIL" 데이터 셋의 필드와 "#OZDeleteFlag#"를 추가하고 추가된 모든 항목을 선택하여 "ColumnEditable" 속성을 "True"로 설정합니다.

  보드에 버튼을 추가하고 버튼의 OnClick 이벤트에 아래와 같이 스크립트를 입력합니다.

```
var Result = Table1.CommitQueuedActions();
_MessageBox(Result);
Table1.GetDataSet().RefreshDataSet();
```

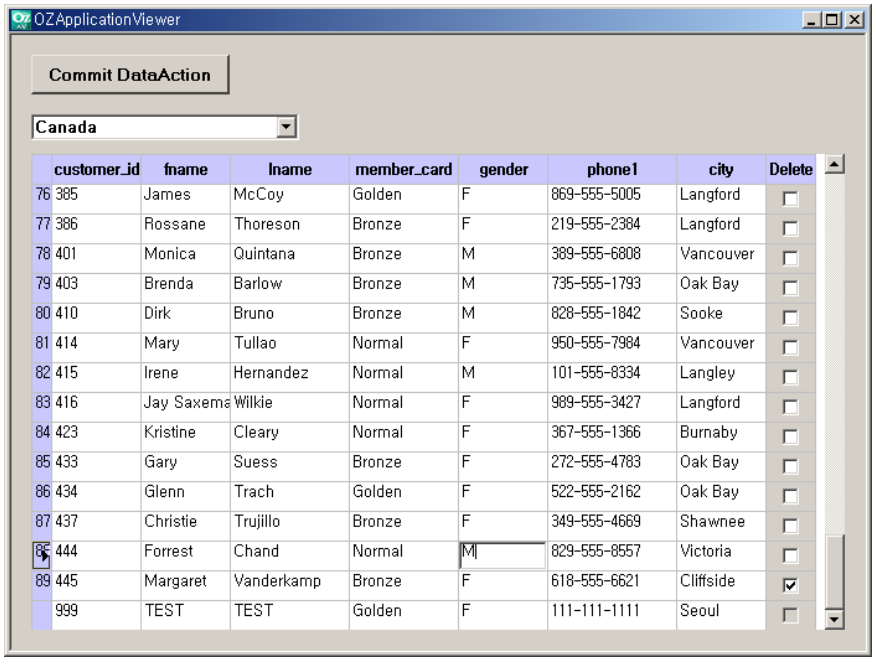
- DataAction 실행하기

  툴바의 미리보기 아이콘을 클릭하여 폼을 미리보기합니다.



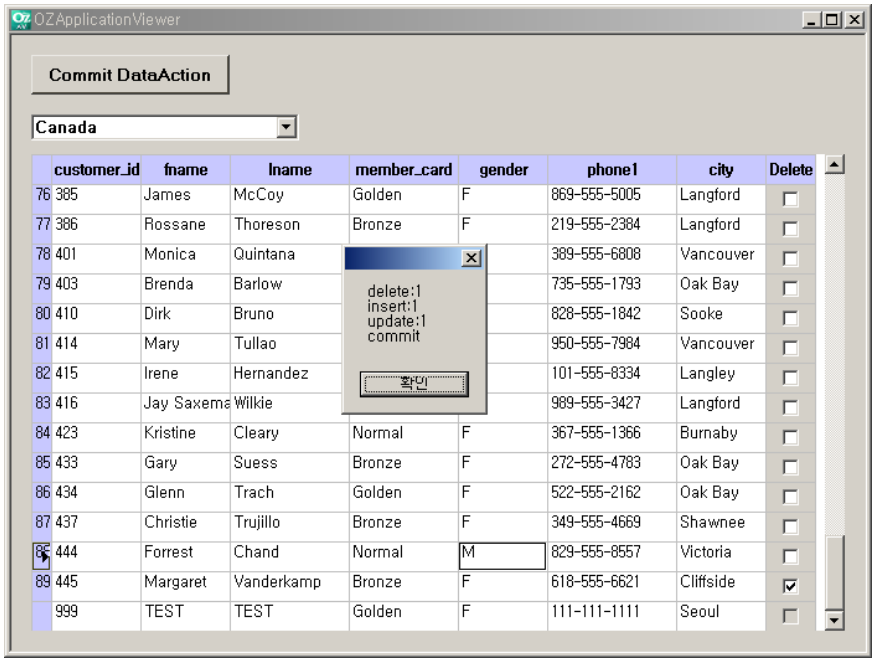
	customer_id	fname	lname	member_card	gender	phone1	city	Delete
0	54	Cheryl	Thorton	Bronze	M	832-555-1834	Metchosin	
1	59	Elizabeth	Moss	Bronze	M	721-555-7412	Burnaby	
2	60	Lori	Anderson	Silver	M	663-555-5248	Langley	
3	65	Whitney	Contreras	Normal	F	648-555-5221	Metchosin	
4	66	Boyd	Pusedu	Silver	F	898-555-2599	Metchosin	
5	67	Cian	Stedman	Bronze	M	994-555-9206	Westminster	
6	72	Laurie	Anderson	Normal	F	398-555-7842	Shawnee	
7	84	William	Burger	Bronze	M	604-555-8988	Haney	
8	87	Kathleen	Thomson	Bronze	F	739-555-2038	N. Vancouve	
9	90	Geneva	Kouba	Bronze	F	832-555-6577	Sooke	
10	94	Susan	Magenheim	Normal	M	293-555-7377	Vancouver	
11	97	Paula	Duran	Golden	M	532-555-3196	Richmond	
12	98	Margaret	Earley	Bronze	F	121-555-4232	Richmond	
13	101	Libby	Allen	Normal	F	848-555-6739	N. Vancouve	
14	102	Bonnie	Bruno	Bronze	M	244-555-1375	Langford	

마지막에 새로 추가된 행에 데이터를 입력하여 추가합니다. Delete 체크박스를 체크하여 삭제할 행을 선택합니다. 데이터를 변경할 셀을 더블 클릭하여 값을 변경합니다.



	customer_id	fname	lname	member_card	gender	phone1	city	Delete
76	385	James	McCoy	Golden	F	869-555-5005	Langford	
77	386	Rossane	Thoreson	Bronze	F	219-555-2384	Langford	
78	401	Monica	Quintana	Bronze	M	389-555-6808	Vancouver	
79	403	Brenda	Barlow	Bronze	M	735-555-1793	Oak Bay	
80	410	Dirk	Bruno	Bronze	M	828-555-1842	Sooke	
81	414	Mary	Tullao	Normal	F	950-555-7984	Vancouver	
82	415	Irene	Hernandez	Normal	M	101-555-8334	Langley	
83	416	Jay Saxema	Wilkie	Normal	F	989-555-3427	Langford	
84	423	Kristine	Cleary	Normal	F	367-555-1366	Burnaby	
85	433	Gary	Suess	Bronze	F	272-555-4783	Oak Bay	
86	434	Glenn	Trach	Golden	F	522-555-2162	Oak Bay	
87	437	Christie	Trujillo	Bronze	F	349-555-4669	Shawnee	
88	444	Forrest	Chand	Normal	M	829-555-8557	Victoria	
89	445	Margaret	Vanderkamp	Bronze	F	618-555-6621	Cliffside	☑
	999	TEST	TEST	Golden	F	111-111-1111	Seoul	

[Commit DataAction] 버튼을 클릭하면 DataAction이 실행된 후 실행된 DataAction 개수를 나타내는 메시지 박스가 표시됩니다.


OZApplicationViewer
Commit DataAction
Canada
customer_id fname lname member_card gender phone1 city Delete
76 385 James McCoy Golden F 869-555-5005 Langford
77 386 Rossane Thoreson Bronze F 219-555-2384 Langford
78 401 Monica Quintana 389-555-6808 Vancouver
79 403 Brenda Barlow 735-555-1793 Oak Bay
80 410 Dirk Bruno 828-555-1842 Sooke
81 414 Mary Tullao 950-555-7984 Vancouver
82 415 Irene Hernandez 101-555-8334 Langley
83 416 Jay Saxema Wilkie 989-555-3427 Langford
84 423 Kristine Cleary Normal F 367-555-1366 Burnaby
85 433 Gary Suess Bronze F 272-555-4783 Oak Bay
86 434 Glenn Trach Golden F 522-555-2162 Oak Bay
87 437 Christie Trujillo Bronze F 349-555-4669 Shawnee
88 444 Forrest Chand Normal M 829-555-8557 Victoria
89 445 Margaret Vanderkamp Bronze F 618-555-6621 Cliffside
999 TEST TEST Golden F 111-111-1111 Seoul
delete:1
insert:1
update:1
commit
확인

Ⅳ. 패러미터 넘겨주는 인터페이스 사용 방법

UDS에서 데이터 모듈 요청시에 패러미터를 필요로 할 경우가 있습니다. 이때 패러미터를 처리하기 위해서는 oz.uds.IParameterRef 인터페이스를 구현해야 합니다.

oz.uds.IParameterRef 인터페이스의 Parameters{set;} 프로퍼티를 사용할 수 있으며, 해당 패러미터 타입은 System.Collection.IDictionary로 주고 받습니다.

■ oz.uds.IParameterRef 인터페이스

```
using System.Collection;

namespace oz.uds
{
   public interface IParameterRef
   {
      IDictionary Parameters{set;}
   }
}
```

- Parameters{set;}

Prototype	System.Collection.IDictionary Parameters{set;}
Definition	IDictionary 형태의 사용자 패러미터 컬렉션
Value	*value* 설정할 패러미터 값

■ ParameterReferenceSample.cs

```
using System;
using System.Web;
using System.Data;
using System.Collections;

using oz.uds;
using oz.uds.dr;

namespace oz.uds.sample.csharp
{
   public class ParameterReferenceSample : OZUserDataReaderStore,
IHttpContextRef, IParameterRef
   {
      private HttpContext _ctx;
      private IDictionary _params;
```

```
        public HttpContext HttpContext{ set{ _ctx = value; } }

        public IDictionary Parameters{ set{ _params = value; } }

        public override void Init()
        {
        }

        public override IDataReader GetDataReader(string command)
        {
            ArrayList fieldNames = new ArrayList();
            ArrayList fieldValues = new ArrayList();

            fieldNames.Add("KEY");
            fieldNames.Add("VALUE");

            foreach(DictionaryEntry entry in _params)
            {
               fieldValues.Add(ArrayList.Adapter(new          object[]{entry.Key,
entry.Value}));
            }

            fieldValues.Add(ArrayList.Adapter(new object[]{"command", command}));

            return new ArrayListDataReader(fieldNames, fieldValues);
        }

        public override void FreeDataReader(IDataReader idr)
        {
        }

        public override void Close()
        {
        }
    }
}
```

- ParameterReferenceSample.vb

```
Imports System.Web
Imports System.Collections
Imports oz.uds.dr

Public Class ParameterReferenceSample
```

```
   Inherits OZUserDataReaderStore
   Implements IHttpContextRef, IParameterRef

   Private _ctx As System.Web.HttpContext
   Private _params As IDictionary

   Public WriteOnly Property HttpContext() As HttpContext Implements
IHttpContextRef.HttpContext
      Set(ByVal value As HttpContext)
         Me._ctx = value
      End Set
   End Property

   Public WriteOnly Property Parameters() As IDictionary Implements
IParameterRef.Parameters
      Set(ByVal value As IDictionary)
         Me._params = value
      End Set
   End Property

   Public Overrides Sub FreeDataReader(ByVal idr As IDataReader)
   End Sub

   Public Overrides Function GetDataReader(ByVal command As String) As
IDataReader

      Dim fieldNames As New ArrayList
      Dim fieldValues As New ArrayList

      fieldNames.Add("KEY")
      fieldNames.Add("VALUE")

      Dim entry As DictionaryEntry
      For Each entry In _params
         fieldValues.Add(ArrayList.Adapter(New Object() {entry.Key,
entry.Value}))
      Next

      fieldValues.Add(ArrayList.Adapter(New Object() {"command", command}))

      Return New ArrayListDataReader(fieldNames, fieldValues)

   End Function
```

```
    Public Overrides Sub Init()
    End Sub

    Public Overrides Sub Close()
    End Sub

End Class
```

■ 적용 예

패러미터 값을 보고서에 출력하는 예제를 살펴보겠습니다.

➢ ODI 만들기

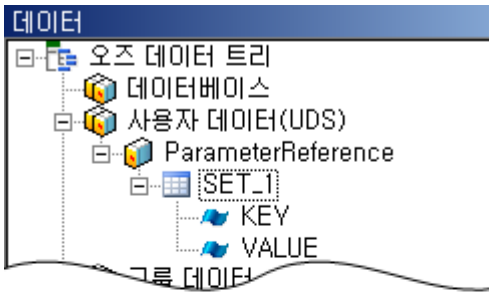
쿼리 디자이너를 실행하여 "City"와 "Gender"라는 이름으로 두 개의 패러미터를 추가하고, 패러미터의 값을 각각 "Canada"와 "F"로 설정합니다.

"ParameterReference"로 사용자 데이터 스토어를 추가하고, "SET_1"로 데이터 셋을 추가합니다.

데이터 셋의 쿼리문 입력 창에 아래와 같이 입력하고

```
select * from ECustomer where city = "#OZParam.City#" and gender = "#OZParam.Gender#"
```

쿼리문 실행 아이콘( )을 클릭하여 필드 리스트를 생성합니다.





ODI 파일을 "parameterreference.odi" 파일로 저장합니다.

- 보고서 디자인하기

  리포트 디자이너를 실행하여 보고서를 새로 만든 후 "parameterreference.odi" 파일을 추가합니다.

  보고서에 데이터 밴드를 추가하고 데이터 밴드의 "ODI 이름" 속성을 "parameterreference"로, "데이터셋" 속성을 "SET_1"로 설정합니다.

  데이터 트리 창에서 "SET_1"의 "KEY"와 "VALUE" 필드를 선택한 후 데이터 밴드로 드래그&드롭하여 추가합니다.

- 미리보기

  툴바의 미리보기 아이콘을 클릭하여 보고서를 미리보기합니다.

Ⅴ. HttpRequest 접근을 위한 인터페이스 사용 방법

IIS의 HttpContext 객체를 통해 UDS를 사용할 경우에는 HttpContext 객체를 처리하기 위해서 oz.uds.IHttpContextRef 인터페이스를 구현해야 합니다.

oz.uds.IHttpContextRef 인터페이스는 oz.uds.IUserDataStore 클래스의 Init() 메소드를 호출하기 전에 setServlet() 함수와 setHttpRequest() 함수를 전달하는 역할을 합니다.

■ oz.uds.IHttpContextRef 인터페이스

```
namespace oz.uds
{
   public interface IHttpContextRef
   {
      HttpContext HttpContext{set;}
   }
}
```

- HttpContext

Prototype	HttpContext HttpContext{set;}
Definition	UDS에서 사용할 HttpContext 값
Argument	*value* 현재 처리중인 Http 요청의 컨텍스트

■ HttpContextReferenceSample.cs

```
using System;
using System.Web;
using System.Data;
using System.Collections;
using System.Collections.Specialized;

using oz.uds;
using oz.uds.dr;

namespace oz.uds.sample.csharp
{
   public class HttpContextReferenceSample : OZUserDataReaderStore,
IHttpContextRef, IParameterRef
   {
      private HttpContext _ctx;
      private IDictionary _params;

      public HttpContext HttpContext{ set{ _ctx = value; } }
```

```
        public IDictionary Parameters{ set{ _params = value; } }

        public override void Init()
        {
        }

        public override IDataReader GetDataReader(string command)
        {
            ArrayList fieldNames = new ArrayList();
            ArrayList fieldValues = new ArrayList();
            fieldNames.Add("KEY");
            fieldNames.Add("VALUE");

            if(null != _ctx)
            {
                NameValueCollection @params = _ctx.Request.Params;
                foreach(string key in @params.AllKeys)
                {
                    log.Debug(key);
                    fieldValues.Add(ArrayList.Adapter(new             object[]{key,
@params[key]}));
                }
            }

            if(0 == fieldNames.Count)
                return new NullDataReader();
            else
                return new ArrayListDataReader(fieldNames, fieldValues);

        }

        public override void FreeDataReader(IDataReader idr)
        {
        }

        public override void Close()
        {
        }
    }
}
```

- HttpContextReferenceSample.vb

```
Imports System.Web
```

```
Imports System.Collections
Imports oz.uds.dr

Public Class HttpContextReferenceSample
   Inherits OZUserDataReaderStore
   Implements IHttpContextRef, IParameterRef

   Private _ctx As System.Web.HttpContext
   Private _params As IDictionary

   Public WriteOnly Property HttpContext() As HttpContext Implements
IHttpContextRef.HttpContext
      Set(ByVal value As HttpContext)
         Me._ctx = value
      End Set
   End Property

   Public WriteOnly Property Parameters() As IDictionary Implements
IParameterRef.Parameters
      Set(ByVal value As IDictionary)
         Me._params = value
      End Set
   End Property

   Public Overrides Sub FreeDataReader(ByVal idr As IDataReader)
   End Sub

   Public Overrides Function GetDataReader(ByVal command As String) As
IDataReader

      Dim fieldNames As New ArrayList
      Dim fieldValues As New ArrayList
      fieldNames.Add("KEY")
      fieldNames.Add("VALUE")

      If (Not _ctx Is Nothing) Then

         Dim params As System.Collections.Specialized.NameValueCollection =
_ctx.Request.Params
         Dim key As String
         For Each key In params.AllKeys
            log.Debug(key)
            fieldValues.Add(ArrayList.Adapter(New Object() {key,
params.Item(key)}))
```

```
            Next
        End If

        If (fieldNames.Count = 0) Then
            Return New NullDataReader
        Else
            Return New ArrayListDataReader(fieldNames, fieldValues)
        End If

    End Function

    Public Overrides Sub Init()
    End Sub

    Public Overrides Sub Close()
    End Sub

End Class
```

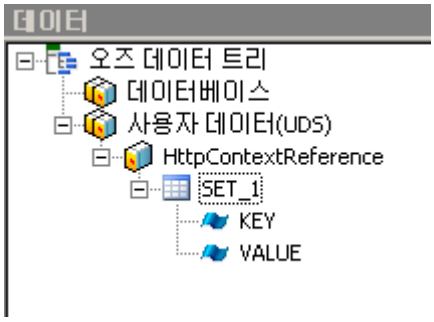
■ 적용 예

IIS의 HttpContext 객체를 사용하는 UDS의 데이터를 보고서에 출력하는 예제를 살펴보겠습니다.

➢ ODI 만들기

쿼리 디자이너를 실행하여 "HttpContextReference"로 사용자 데이터 스토어를 추가하고, "SET_1"로 데이터 셋을 추가합니다.

쿼리문 입력 창에 실행문을 입력하고 쿼리문 실행 아이콘( )을 클릭하여 필드 리스트를 생성합니다.



ODI 파일을 "httpcontextreference.odi"로 저장합니다.

➢ 보고서 디자인하기

리포트 디자이너를 실행하여 보고서를 새로 만든 후 "httpcontextreference.odi" 파일을 추가합니다.

보고서에 데이터 밴드를 추가하고 데이터 밴드의 "ODI 이름" 속성을 "httpcontextreference"로, "데이터셋" 속성을 "SET_1"로 설정합니다.

데이터 트리 창에서 "SET_1"의 "KEY"와 "VALUE" 필드를 선택한 후 데이터 밴드로 드래그&드롭하여 추가합니다.

디자인한 보고서를 "httpcontextreference.ozr"로 저장한 후 ODI와 OZR을 서버 리파지토리에 업로드합니다.

➢ 웹에서 보고서 미리보기

작성된 보고서를 호출하는 asp.net 페이지를 작성합니다.

HTTP Request의 Key와 Value는 보고서를 호출하는 웹페이지의 뷰어 패러미터 중 "connection.servlet" 패러미터에 "?key1=value1&key2=value2&..."이런 식으로 입력하여야 합니다.

예를 들어 보고서를 호출하는 asp.net 페이지를 아래처럼 작성한 후

```
...
<param name="connection.servlet"
value="http://127.0.0.1/OZASPNET/Server.aspx?k1=a1&k2=a2&k3=a3">
...
```

웹 브라우저에서 asp.net 페이지를 실행하여 보고서를 바인딩합니다.

Ⅵ. IDataReader 사용 방법

oz.uds.dr.DefaultDataReader

기본적인 값을 제공하는 추상 클래스입니다.

DefaultDataReader를 사용하는 경우 다음의 추상 멤버를 반드시 구현해야 합니다.

```
abstract public bool Read()
abstract public DataTable GetSchemaTable()
abstract public object this[string name]{get;}
abstract public object this[int zeroBasedIndex]{get;}
```

oz.uds.dr.ArrayDataReader

배열 형태의 데이터를 처리하는 클래스입니다.

■ Constructor Summary

- **public ArrayDataReader(string[] fieldNames, string[][] data)**
- **public ArrayDataReader(string[] fieldNames, string[,] data)**
- **public ArrayDataReader(string[] fieldNames, Type [] types, string[][] data)**
- **public ArrayDataReader(string[] fieldNames, Type[] types, string[,] data)**

■ Constructor Detail

Prototype	//데이터 타입이 문자형인 경우
	public ArrayDataReader(string[] fieldNames, string[][]data)
	public ArrayDataReader(string[] fieldNames, string[,]data)
	
	//데이터 타입을 지정하는 경우
	public ArrayDataReader(string[] fieldNames, Type[] types, string[][] data)
	public ArrayDataReader(string[] fieldNames, Type[] types, string[,] data)	

Argument	*fieldNames*	필드 이름
	types	데이터 타입
	data	2차 배열 형태의 데이터(사각형 또는 가변형)

ArrayDataReaderSample.cs

```
using System;
using System.Web;
using System.Data;
using System.Collections;

using oz.uds;
using oz.uds.dr;

namespace oz.uds.sample.csharp
{
   public class ArrayDataReaderSample : OZUserDataReaderStore, IHttpContextRef,
IParameterRef
   {
      private HttpContext _ctx;
      private IDictionary _params;

      public HttpContext HttpContext{ set{ _ctx = value; } }

      public IDictionary Parameters{ set{ _params = value; } }

      public override void Init()
      {
      }

      public override IDataReader GetDataReader(string command)
      {
         int fieldCount =
oz.util.OZString.ParseInt(Convert.ToString(_params["field_count"]), 5);
         int rowCount =
oz.util.OZString.ParseInt(Convert.ToString(_params["row_count"]), 5);

         bool useJaggedArray =
oz.util.OZString.ParseBool(Convert.ToString(_params["use_jagged_array"]),
false);

         string[] fieldNames = new string[fieldCount];
         string[][] jagged = null;
         string[,] rectangular = null;
         if(useJaggedArray)
            jagged = new string[rowCount][];
         else
            rectangular = new string[rowCount, fieldCount];
```

```
            for(int col = 0; col < fieldCount; col++)
                fieldNames[col] = "Field" + col;

            for(int row = 0; row < rowCount; row++)
            {
                if(useJaggedArray)
                    jagged[row] = new string[fieldCount];

                for(int col = 0; col < fieldCount; col++)
                {
                    string value = "Value - " + row + "," + col;
                    if(useJaggedArray)
                        jagged[row][col] = value;
                    else
                        rectangular[row,col] = value;
                }
            }

            if(useJaggedArray)
                return new ArrayDataReader(fieldNames, jagged);
            else
                return new ArrayDataReader(fieldNames, rectangular);
        }

        public override void FreeDataReader(IDataReader reader)
        {
        }

        public override void Close()
        {
        }
    }
}
```

- **ArrayDataReaderSample.vb**

```
Imports System.Web
Imports System.Collections
Imports oz.uds.dr

Public Class ArrayDataReaderSample
    Inherits OZUserDataReaderStore
    Implements IHttpContextRef, IParameterRef
```

```
   Private _ctx As System.Web.HttpContext
   Private _params As IDictionary

   Public WriteOnly Property HttpContext() As HttpContext Implements
IHttpContextRef.HttpContext
      Set(ByVal value As HttpContext)
         Me._ctx = value
      End Set
   End Property

   Public WriteOnly Property Parameters() As IDictionary Implements
IParameterRef.Parameters
      Set(ByVal value As IDictionary)
         Me._params = value
      End Set
   End Property

   Public Overrides Sub FreeDataReader(ByVal reader As IDataReader)
   End Sub

   Public Overrides Function GetDataReader(ByVal command As String) As
IDataReader

      Dim fieldCount As Integer =
oz.util.OZString.ParseInt(Convert.ToString(_params.Item("field_count")), 5)
      Dim rowCount As Integer =
oz.util.OZString.ParseInt(Convert.ToString(_params.Item("row_count")), 5)

      Dim useJaggedArray As Boolean =
oz.util.OZString.ParseBool(Convert.ToString(_params.Item("use_jagged_array")),
False)

      log.Debug("Field count : " & fieldCount)

      Dim fieldNames(fieldCount - 1) As String
      Dim jagged()() As String
      Dim rectangular(,) As String

      If (useJaggedArray) Then
         jagged = New String(rowCount - 1)() {}
      Else
         rectangular = New String(rowCount - 1, fieldCount - 1) {}
      End If
```

```
        Dim col As Integer
        For col = 0 To fieldCount - 1
            fieldNames(col) = "Field" & col
        Next col

        Dim row As Integer
        For row = 0 To rowCount - 1

            If (useJaggedArray) Then
                jagged(row) = New String(fieldCount - 1) {}
            End If

            For col = 0 To fieldCount - 1
                Dim value = "Value - " & row & "," & col
                If (useJaggedArray) Then
                    jagged(row)(col) = value
                Else
                    rectangular(row, col) = value
                End If
            Next col
        Next row

        If (useJaggedArray) Then
            Return New ArrayDataReader(fieldNames, jagged)
        Else
            Return New ArrayDataReader(fieldNames, rectangular)
        End If

    End Function

    Public Overrides Sub Init()
    End Sub

    Public Overrides Sub Close()
    End Sub

End Class
```

oz.uds.dr.DynamicDataReader

동적인 형태의 데이터를 처리하는 클래스입니다.

Constructor Summary

- public DynamicDataReader(IDataReader reader)

Constructor Detail

Prototype	public DynamicDataReader(IDataReader reader)
Argument	*reader* 데이터가 동적으로 변경되는 ResultSet을 패러미터로 설정

DynamicDataReaderSample.cs

```
using System;
using System.Web;
using System.Data;
using System.Collections;

using oz.uds;
using oz.uds.dr;

namespace oz.uds.sample.csharp
{
   public class DynamicDataReaderSample : OZUserDataReaderStore,
IHttpContextRef, IParameterRef
   {
      private HttpContext _ctx;
      private IDictionary _params;

      public HttpContext HttpContext{ set{ _ctx = value; } }

      public IDictionary Parameters{ set{ _params = value; } }

      public override void Init()
      {
      }

      public override IDataReader GetDataReader(string command)
      {
         int fieldCount =
oz.util.OZString.ParseInt(Convert.ToString(_params["field_count"]), 5);
         int rowCount =
oz.util.OZString.ParseInt(Convert.ToString(_params["row_count"]), 5);

         if(fieldCount > 100)
```

```
            throw new ArgumentOutOfRangeException("field_count", "Too big
field count. ");

         ArrayList fieldNames = new ArrayList();
         ArrayList data = new ArrayList();

         for(int col = 0; col < fieldCount; col++)
            fieldNames.Add("Field" + col);

         for(int row = 0; row < rowCount; row++)
         {
            ArrayList rowValues = new ArrayList();

            for(int col = 0; col < fieldCount; col++)
            {
               rowValues.Add("Value - " + row + "," + col);
            }

            data.Add(rowValues);
         }

         return new DynamicDataReader(new ArrayListDataReader(fieldNames,
data));
      }

      public override void FreeDataReader(IDataReader reader)
      {
      }

      public override void Close()
      {
      }
   }
}
```

- DynamicDataReaderSample.vb

```
Imports System.Web
Imports System.Collections

Public Class DynamicDataReaderSample
   Inherits OZUserDataReaderStore
   Implements IHttpContextRef, IParameterRef
```

```
    Private _ctx As System.Web.HttpContext
    Private _params As IDictionary

    Public WriteOnly Property HttpContext() As HttpContext Implements
IHttpContextRef.HttpContext
        Set(ByVal value As HttpContext)
            Me._ctx = value
        End Set
    End Property

    Public WriteOnly Property Parameters() As IDictionary Implements
IParameterRef.Parameters
        Set(ByVal value As IDictionary)
            Me._params = value
        End Set
    End Property

    Public Overrides Sub FreeDataReader(ByVal reader As IDataReader)
    End Sub

    Public Overrides Function GetDataReader(ByVal command As String) As
IDataReader

        Dim fieldCount As Integer =
oz.util.OZString.ParseInt(Convert.ToString(_params.Item("field_count")), 5)
        Dim rowCount As Integer =
oz.util.OZString.ParseInt(Convert.ToString(_params.Item("row_count")), 5)
        If (fieldCount > 100) Then
            Throw New ArgumentOutOfRangeException("field_count", "Too big field
count. ")
        End If

        Dim fieldNames As New ArrayList
        Dim data As New ArrayList

        Dim col As Integer
        For col = 0 To fieldCount - 1
            fieldNames.Add(("Field" & col))
        Next col

        Dim row As Integer
        For row = 0 To rowCount - 1
            Dim rowValues As New ArrayList
            For col = 0 To fieldCount - 1
```

```
            rowValues.Add("Value - " & row & "," & col)
         Next col
         data.Add(rowValues)
      Next row

      Return New oz.uds.dr.DynamicDataReader(New
oz.uds.dr.ArrayListDataReader(fieldNames, data))

   End Function

   Public Overrides Sub Init()
   End Sub

   Public Overrides Sub Close()
   End Sub

End Class
```

oz.uds.dr.ArrayListDataReader

ArrayList를 형태의 데이터를 처리하는 클래스입니다.

■ Constructor Summary

- public ArrayListDataReader(ArrayList fieldNames, ArrayList data)
- public ArrayListDataReader(ArrayList fieldNames, ArrayList[] data)
- public ArrayListDataReader(ArrayList fieldNames, ArrayList dataTypes, ArrayList data)
- public ArrayListDataReader(ArrayList fieldNames, ArrayList dataTypes, ArrayList[] data)

■ Constructor Detail

Prototype

```
//데이터 타입이 VARCHAR인 경우
public ArrayListDataReader(ArrayList fieldNames, ArrayList
data)
public ArrayListDataReader(ArrayList fieldNames,
ArrayList[] data)

//데이터 타입이 VARCHAR가 아닌 경우
```

	public ArrayListDataReader(ArrayList fieldNames, ArrayList dataTypes, ArrayList data) public ArrayListDataReader(ArrayList fieldNames, ArrayList dataTypes, ArrayList[] data)	
Definition	ArrayList 형태로 데이터를 설정합니다.	
Argument	*fieldNames*	필드 이름
	fieldTypes	데이터 타입(컬렉션의 객체의 타입은 System.Type 형태)
	data	ArrayList 또는 ArrayList[] 형태의 데이터

■ ArrayListDataReaderSample.cs

```
using System;
using System.Web;
using System.Data;
using System.Collections;

using oz.uds;
using oz.uds.dr;

namespace oz.uds.sample.csharp
{
   public class ArrayListDataReaderSample : OZUserDataReaderStore,
IHttpContextRef, IParameterRef
   {
      private HttpContext _ctx;
      private IDictionary _params;

      public HttpContext HttpContext{ set{ _ctx = value; } }
      public IDictionary Parameters{ set{ _params = value; } }

      public ArrayListDataReaderSample()
      {}

      public override void Init()
      {
      }

      public override IDataReader GetDataReader(string command)
      {
         ArrayList fieldNames = new ArrayList();
         fieldNames.Add("field1");
         fieldNames.Add("field2");
```

```
          ArrayList fieldTypes = new ArrayList();
          fieldTypes.Add(typeof(string));
          fieldTypes.Add(typeof(int));

          int rowCount =
oz.util.OZString.ParseInt(Convert.ToString(_params["row_count"]), 5);

          if("ArrayList".Equals(_params["return_type"]))
          {
              ArrayList data = new ArrayList();

              for(int i = 0; i < rowCount; i++)
              {
                  ArrayList row = new ArrayList();
                  row.Add("value" + i);
                  row.Add(i);
                  data.Add(row);
              }

              return new oz.uds.dr.ArrayListDataReader(fieldNames, fieldTypes,
data);
          }
          else if("ArrayList[]".Equals(_params["return_type"]))
          {
              ArrayList[] data = new ArrayList[rowCount];

              for(int i = 0; i < rowCount; i++)
              {
                  data[i] = new ArrayList();
                  data[i].Add("value" + i);
                  data[i].Add(i);
              }

              return new oz.uds.dr.ArrayListDataReader(fieldNames, fieldTypes,
data);
          }
          else
          {
              log.Error("Unknown return type. ");
              return new oz.uds.dr.NullDataReader();
          }
      }

      public override void FreeDataReader(IDataReader reader)
```

```
        {
        }

        public override void Close()
        {
        }
    }
}
```

- ArrayListDataReaderSample.vb

```
Imports System.Web
Imports System.Collections
Imports oz.uds.dr

Public Class ArrayListDataReaderSample
    Inherits OZUserDataReaderStore
    Implements IHttpContextRef, IParameterRef

    Private _ctx As System.Web.HttpContext
    Private _params As IDictionary

    Public WriteOnly Property HttpContext() As HttpContext Implements
IHttpContextRef.HttpContext
        Set(ByVal value As HttpContext)
            Me._ctx = value
        End Set
    End Property

    Public WriteOnly Property Parameters() As IDictionary Implements
IParameterRef.Parameters
        Set(ByVal value As IDictionary)
            Me._params = value
        End Set
    End Property

    Public Overrides Sub FreeDataReader(ByVal reader As IDataReader)
    End Sub

    Public Overrides Function GetDataReader(ByVal command As String) As
IDataReader
        Dim fieldNames As New ArrayList
        fieldNames.Add("field1")
        fieldNames.Add("field2")
```

```
        Dim fieldTypes As New ArrayList
        fieldTypes.Add(GetType(String))
        fieldTypes.Add(GetType(Int32))

        Dim rowCount As Int32 =
oz.util.OZString.ParseInt(Convert.ToString(_params.Item("row_count")), 5)

        If ("ArrayList".Equals(_params.Item("return_type"))) Then

            Dim data As New ArrayList

            Dim i As Integer
            For i = 0 To rowCount - 1
                Dim row As New ArrayList
                row.Add(("value" & i))
                row.Add(i)
                data.Add(row)
            Next i

            Return New ArrayListDataReader(fieldNames, fieldTypes, data)
        ElseIf ("ArrayList[]".Equals(_params.Item("return_type"))) Then
            Dim data(rowCount) As ArrayList

            Dim i As Integer
            For i = 0 To rowCount - 1
                data(i) = New ArrayList
                data(i).Add(("value" & i))
                data(i).Add(i)
            Next

            Return New ArrayListDataReader(fieldNames, fieldTypes, data)
        Else
            log.Error("Unknown return type. ")
            Return New oz.uds.dr.NullDataReader
        End If

    End Function

    Public Overrides Sub Init()
    End Sub

    Public Overrides Sub Close()
    End Sub
```

```
End Class
```

oz.uds.dr.NullDataReader

null 값을 리턴해주는 클래스입니다.

- **NullDataReaderSample.cs**

```
using System;
using System.Web;
using System.Data;
using System.Collections;

using oz.uds;
using oz.uds.dr;

namespace oz.uds.sample.csharp
{
   public class NullDataReaderSample : OZUserDataReaderStore, IHttpContextRef,
IParameterRef
   {
      private HttpContext _ctx;
      private IDictionary _params;

      public HttpContext HttpContext{ set{ _ctx = value; } }

      public IDictionary Parameters{ set{ _params = value; } }

      public override void Init()
      {
      }

      public override IDataReader GetDataReader(string command)
      {
         return new NullDataReader();
      }

      public override void FreeDataReader(IDataReader reader)
      {
      }
```

```
        public override void Close()
        {
        }
    }
}
```

- **NullDataReaderSample.vb**

```
Imports System.Web
Imports System.Collections
Imports oz.uds.dr

Public Class NullDataReaderSample
    Inherits OZUserDataReaderStore
    Implements IHttpContextRef, IParameterRef

    Private _ctx As System.Web.HttpContext
    Private _params As IDictionary

    Public WriteOnly Property HttpContext() As HttpContext Implements
IHttpContextRef.HttpContext
        Set(ByVal value As HttpContext)
            Me._ctx = value
        End Set
    End Property

    Public WriteOnly Property Parameters() As IDictionary Implements
IParameterRef.Parameters
        Set(ByVal value As IDictionary)
            Me._params = value
        End Set
    End Property

    Public Overrides Sub FreeDataReader(ByVal reader As IDataReader)
    End Sub

    Public Overrides Function GetDataReader(ByVal command As String) As
IDataReader
        Return New NullDataReader
    End Function

    Public Overrides Sub Init()
    End Sub
```

```
    Public Overrides Sub Close()
    End Sub

End Class
```