Module 4

Database Access

- Overview
- Access SQL Server
- Access Excel Sheet
- Use XML Documents

1

Overview

Overview of SQL DB queries

- **Several objects**
  - SQLClient
  - OLEDBClient
  - ODBC
- **SQL Statements**
  - Data Manipulation Language (DML)
  - Other object(s) for
    - Data Definition Language (DDL)
    - Data Control Language (DCL)

2

Overview

Overview of SQL DB queries

- **Important SQL Statements**
  - Select … From … Where
    - Getting data from SQL Server
  - Delete From … Where
    - Delete existing datarow/record from a table
  - Insert into … (…) values (…)
    - Add new datarow/record to a table
  - Update … Set … Where
    - Change existing datarow/record in a table

```
Select * From Customers
Select Lastname,CustID From Customers
Select * From Customers Where CustID=1234
```

```
Delete From Customers Where CustID=1234
Delete From Orders
```

```
Insert Into Customers Values 1324, John, Doe
Insert Into Customers (CustID, Firstname, Lastname) Values 1324, John, Doe
```

```
Update Customers Set Lastname=Mustermann Where CustID=1234
```

Get SQL Data via PowerShell

Objects used to get and change data from a database

- **Components**
  - Connection & Connectionstring
  - Command & Connection
  - Objects for requested Data
    - DataReader
    - DataAdapter & DataSet, DataTable

Access a SQL Server

Objects used to get data from SQL Server database and change them.

- **Used Objects**
  - System.Data.SqlClient
  - *.SqlClient.Connection

  - *.SqlCommand
  - *.SqlDataReader

  - *.SqlDataAdapter
  - System.Data.DataSet or
  - Sysetm.Data.DataTable
  - *.SqlCommandBuilder

5

Connectionstring

What is a connectionstring and how to get it

- **A connectionstring contains all connection and authentication data.**
- **Depends on data source, driver/provider and options**
  - Use: http://www.connectionstrings.com

- **For SQL Server and .Net Data Provider for SQL Server**
  - “Server=SQLServer;Database=DBName;Integrated Security=True“
  - “Server=SQLServer\Instance;Database=DBName;Integrated Security=True”
  - “Server=SQLServer;Database=DBName;User ID=UName;Password=UPassword”

6

Connection

How to create, open and close a connection to a SQL Server

- **Requirements**
  - SqlConnection
  - Connectionstring
  - Methods Open() and Close()

- **Hint**
  - Use try … catch … finally to close a connection even in case of failure

```
Try
{
   $myConn = New-Object System.Data.SqlClient.SqlConnection
   $myConn.ConnectionString = "this is the connectionstring"
   $myConn.Open()

   $myConn.State
}
Catch { … }
Finally
{
   $myConn.Close()
}
```

DIST IT SERVICES & TRAINING

Page ▪ 7

7

Command and DataReader

How to create and use a SELECT command with a DataReader

- **Requirements**
  - Select-Statement (querystring)
  - SqlCommand
  - SqlDataReader

- **Steps**
  - Build the command
  - Execute the command
  - Save the result in a DataReader

```
Try
{
   …
   $myComm = New-Object System.Data.SqlClient.SqlCommand
   $myComm.CommandText = "this is the select-statement."
   $myComm.Connection = $myConn

   $myDataReader = $myComm.ExecuteReader()
}
Catch { … }
Finally
{
   $myComm.Close()
}
```

DIST IT SERVICES & TRAINING

Page ▪ 8

8

DataReader

How to get the data from a reader

- **What is a DataReader**
- **Members**
  - Read()
  - Item(ordinalNumber) or Item("field")
  - FieldCount
  - GetName(ordinalNumber) to get fieldname

```
Try
{
   …
   while ($myDataReader.Read())
   {
     $myDateReader.Item(0) *** $myDataReader.Item("Lastname")
   }
}
Catch { … }
Finally
{
   $myConn.Close()
}
```

Page ▪ 9

DataAdapeter and DataTable

How to use a DataAdapter and a DataSet to manipulate a datasource

- **What is a DataAdapter**
  - Connection between DataSource and DataTable with the ability to change data.
- **Requirements**
  - *.SqlDataAdapter which
  - fills a DataTable (or a DataSet)
- **Steps**
  - Create the DataAdapter
  - Fill the DataTable
  - Read from the DataTable

```
$myDataAdapter = New-Object System.Data.SqlClient.SqlDataAdapter("queryString",$myConn)
$myDataTable = New-Object System.Data.DataTable
$rowcount = $myDataAdapter.Fill($myDataTable)

$myDataTable.Rows[3].Item(4)
```

Page ▪ 10

Change Data via DataAdapter and DataTable

How to change data on a SQL Server via DataAdapter and DataTable

- **Insert a new record**
  - Create a SqlCommandBuilder for the DataAdapter
  - Create a new datarow with the fields of existing table
  - Set the value of each field
  - Update the DataAdapter

```
$myDataAdapter = New-Object System.Data.SqlClient.SqlDataAdapter($strCommandText,$myConn)
$myCommandBuilder = New-Object System.Data.SqlClient.SqlCommandBuilder($myDataAdapter)

# Create DataTable
$myDataTable = New-Object -TypeName System.Data.DataTable
$rowcount = $myDataAdapter.Fill($myDataTable)

# Create a DataRow and add it to the DataTable
$myDataRow = $myDataTables.NewRow()
$myDataRow.CategoryName = "Uneatable"
$myDataRow.Description = "Do not eat it ..."
$myDataTable.Rows.Add($myDataRow)

# Update the source
$myDataAdapter.Update($myDataTable")
```

DIST IT SERVICES & TRAINING

Change Data via DataAdapter and DataTable

How to change data on a SQL Server via DataAdapter and DataTable

- **Update a record**
  - Create a SqlCommandBuilder for the DataAdapter
  - Navigate to the respective datarow
  - Set the value of each field
  - Update the DataAdapter

```
$strCommandText = "Select * From dbo.Categories"
$myDataAdapter = New-Object System.Data.SqlClient.SqlDataAdapter($strCommandText,$myConn)
$myCommandBuilder = New-Object System.Data.SqlClient.SqlCommandBuilder($myDataAdapter)

# Create DataSet
$myDataTable = New-Object -TypeName System.Data.DataTable
$rowcount = $myDataAdapter.Fill($myDataTable)

# Select a specific DataRow and change it
$myDataTables.Rows[10].Item(1) = "Flowers"

# Update the source
$myDataAdapter.Update($myDataTable)
```

DIST IT SERVICES & TRAINING

Change Data via DataAdapter and DataTable

How to change data on a SQL Server via DataAdapter and DataTable

- **Delete a record**
  - Create a SqlCommandBuilder for the DataAdapter
  - Navigate to the respective datarow
  - Use the Delete()-Method of the row
  - Update the DataAdapter

```
$strCommandText = "Select * From dbo.Categories"
$myDataAdapter = New-Object System.Data.SqlClient.SqlDataAdapter($strCommandText,$myConn)
$myCommandBuilder = New-Object System.Data.SqlClient.SqlCommandBuilder($myDataAdapter)

# Create DataSet
$myDataTable = New-Object -TypeName System.Data.DataTable
$rowcount = $myDataAdapter.Fill($myDataTable)

# Select a specific DataRow and change it
$myDataTables.Rows[8].Delete()

# Update the source
$myDataAdapter.Update($myDataTable)
```

DIST IT SERVICES & TRAINING

Change Data via Commands

How to insert a new dataset with a SqlCommand

- **Insert a new record**
  - Create a SqlCommand and the 'INSERT INTO …' as constructor-parameter
  - Add the connection to the command
  - Execute the 'ExecuteNonQuery()' method

```
# simple

$strCommandText = "INSERT INTO dbo.Categories ([CategoryName],[Description]) " `
                   VALUES ('Flowers','Only the best.')"

try
{
   …
   $myConn.Open()

   $myComm = New-Object -TypeName System.Data.SqlClient.SqlCommand($strCommandText)
   $myComm.Connection = $myConn

   $affectedRows = $myComm.ExecuteNonQuery()
}
```

DIST IT SERVICES & TRAINING

Change Data via Commands

How to insert a new dataset with a SqlCommand

- **Insert a new record**

```
# more detailed

$strCommandText = "INSERT INTO dbo.Categories ([CategoryName],[Description])” `
                   VALUES (@CatName,@Desc)"
try
{
   …
   $myConn.Open()

   $myComm = New-Object -TypeName System.Data.SqlClient.SqlCommand($strCommandText)
   $myComm.Connection = $myConn

   $paraCategoryName = New-Object -TypeName System.Data.SqlClient.SqlParameter
   $paraCategoryName.ParameterName = "@catName"
   $paraCategoryNAme.Value = "Uneatable"
   …
…$myComm.Parameters.Add($paraCategoryName)

   $affectedRows = $myComm.ExecuteNonQuery()
}
```

Change Data via Commands

How to change data on a SQL Server via Sqlcommand

- **Equivalent to ‘INSERT INTO …’**
  - Update
  - Delete

```
$strCommandText = "UPDATE dbo.Categories SET CategoryName = @CatName” `
                     WHERE CategoryID = @CatID"


$strCommandText = "DELETE FROM dbo.Categories WHERE CategoryID = @CatID‟
```

Access a Excel Sheet

How to access data in an Excel Sheet

- **Access Database Engine**
  - http://www.microsoft.com/en-ie/download/details.aspx?id=13255
- **System.Data.OleDB.***
  - *.OleDBConnection
  - *.OleDBCommand
  - *.OleDBDataReader
  - *.OleDBDataAdapter
  - *.OleDBParameter
- **Connectionsstring**
  - http://www.connectionstrings.com
  - “Provider=Microsoft.ACE.OLEDB.12.0; `
    Data Source = Path-to.xlsx;Extended Properties='Excel 12.0 Xml; HDR = yes';”

Access a Excel Sheet

- **Hint**
  - It’s impossible to delete a datarow in a sheet as a whole.
  - Workaround: Set all fields to $null (update) and any further select-statements must have the option NOT NULL
  - http://support.microsoft.com/kb/257819/en-us

Lab

- **Create a script to change the LastName of employees**
  - Employees are saved in NorthWind database in the table 'Employees'
  - Define two parameters
    - [parameter(Mandatory=$true)][String]$Oldname
    - [parameter(Mandatory=$true)][String]$Newname
  - Find the user via Oldname and replace the LastName with Newname

XML

How to use a XML content as datasource

- **XML wording**
  - Document
  - Tag
  - Attribute
  - Node
- **Getting xml-document**
  - Open a xml-file
  - Convert a collection

Using XML as Object

How to get XML data by using as object

- **Know the structure**
- **Get data into variable**

```
[xml]$inventory = Get-Content .\Inventory.xml
```

- **Navigate through “xml-tree” like through nested arrays**
- **Examples**

Command	Result
$myXML.Sites.Site	From all sites all attributes, tags and tags with children
$myXML.Sites.Site[0]	From first site all attributes, tags and tags with children
$myXML.Sites.Site.Servers	From all sites all servers (summarized by <Name> or repeating tagname)
$myXML.Sites.Site.Servers.Server	From all sites all servers all attributes, tags and tags with children
$myXML.Sites.Site.Servers.Server[1]	From the second server in the document all attributes, tags and tags with children

DIST IT SERVICES & TRAINING

Change Data in XML Document

How to change data in xml documents

- **Add a node**

(1) Clone a node
(2) Change attributes and tags
(3) Add clone to document
(4) Save document (Attention)

```
[xml]$inventory = Get-Content .\Inventory.xml

$newServer = $inventory.Sites.Site.servers.server[0].Clone()

$newServer.model = "Dell"
$newServer.Name = "Mars"
$newServer.IP = "2.3.4.8"
$newServer.OS = "Windows Server 2012 R2"

$inventory.Sites.Site[1].servers.AppendChild($newServer)

$inventory.Save(path)
```

IT SERVICES & TRAINING

Change Data in XML Document

How to change data in XML documents

- **Change a tag/attribute**

(1) Set new content to existing tag

(2) Save document

```
[xml]$inventory = Get-Content .\Inventory.xml

$inventory.Sites.Site.servers.server[0].OS = “Windows 10”

$inventory.Save(path)
```

- **Remove a node**

(1) Use method RemoveAll() to delete node

(2) Save document

```
[xml]$inventory = Get-Content .\Inventory.xml

$inventory.Sites.Site.servers.server[0].RemoveAll()

$inventory.Save(path)
```

Do You Have Any Questions?