SQL Server™ 2005: Web Services

Table of Contents

SQL Server 2005: Web Services

Objectives

After completing this lab, you will be able to:

- Create HTTP Endpoints
- Consume HTTP Endpoints

NOTE: This lab focuses on the concepts in this module and as a result may not comply with Microsoft® security recommendations.

NOTE: The SQL Server 2005 labs are based on beta builds of the product. The intent of these labs is to provide you with a general feel of some of the planned features for the next release of SQL Server. As with all software development projects, the final version may differ from beta builds in both features and user interface. For the latest details on SQL Server 2005, please visit http://www.microsoft.com/sql/2005/.

Scenario

This lab will show you how to create HTTP endpoints, view the WSDL exposed by the endpoints, and consume the HTTP endpoints.

Prerequisites

- Experience with T-SQL
- Basic understanding of Microsoft Windows® networking concepts
- Experience with SQL Server 7.0 or SQL Server 2000
- An overall understanding of the .NET Framework

Estimated Time to Complete This Lab

30 Minutes

Lab Setup

Tasks	Detailed Steps
1. Log in.	**1.** Log in using the **Administrator** user account. The password is **Pass@word1**.

Exercise 1
Create HTTP Endpoints

Scenario

In this exercise, you will learn how to create HTTP endpoints in SQL Server 2005 and how to view the WSDL exposed by these endpoints. The ability to easily expose your data from SQL Server 2005 gives you great flexibility in letting a variety of clients access it. By leveraging Web service standards such as HTTP, WSDL, SOAP, and XML, clients or servers that run on other platforms can access data stored in SQL Server 2005. Also, the ability to do this using a standards-based approach, as opposed to previous approaches such as linked servers, requires nothing more from the consuming client or server than following established standards in consuming WSDL and sending and receiving SOAP and XML data.

Tasks	Detailed Steps
1. Create a Management Studio Project. NOTE: All of the code for this exercise is contained in the script file HTTP.sql in the C:\SQL Labs\Lab Projects\Web Services Lab\SQLHTTP directory. You can copy and paste from this file to complete any portion of the exercise.	**1.** Click **Start | All Programs | Microsoft SQL Server 2005 | SQL Server Management Studio**. **2.** When asked to connect, make sure the server name is **localhost**, and that **Windows Authentication** is selected in the Authentication dropdown, and then click **Connect**. **3.** Select the **File | New | Project** menu item. **4.** In the **New Project** dialog box, click **SQL Server Scripts**. **5.** Change the name of the project to **SQLHTTP.** **6.** Change the Location to **C:\SQL Labs\User Projects**. **7.** Click **OK** to create and save the new project.
2. Add a new script to the project.	**1.** In the Solution Explorer, right-click **Queries** and click **New Query**. **2.** When prompted for connection information, type "localhost" for the **Server name**, ensure that **Windows Authentication** is selected, and click **Connect**. **3.** Right-click the new script file and click **Rename**. **4.** Rename the script to **HTTP.sql**.

3. Create a Stored Procedure.	**1.** In the HTTP.sql script, type the following statements, which create a new function in the AdventureWorks database.
	
	`USE AdventureWorks`
	`GO`
	`CREATE FUNCTION EmployeeOrderCount(@EmpID INT)`
	`RETURNS INT`
	`AS`
	`BEGIN`
	`RETURN`
	`(`
	`SELECT COUNT(*) AS 'Employee Order Count'`
	`FROM Purchasing.PurchaseOrderHeader`
	`WHERE EmployeeID = @EmpID`
	`GROUP BY EmployeeID`
	`)`
	`END`
	
	2. Select the statement you just typed.

Tasks	Detailed Steps
	3. Press **F5** to execute the script and create the function. This function will be called when a user makes a method call on the HTTP Endpoint that you will create in the following task.

4. Create an HTTP Endpoint.	**1.** In the HTTP.sql script, add the following statement.
	`CREATE ENDPOINT AdvEnd`
	`STATE = STARTED`
	`AS HTTP`
	`(`
	`SITE = 'localhost',`
	`PATH = '/AdvEnd',`
	`AUTHENTICATION = ( INTEGRATED ),`
	`PORTS = ( CLEAR )`
	`)`
	`FOR SOAP`
	`(`
	`WEBMETHOD 'GetEmployeeOrderCount'`
	`(`
	`NAME = 'AdventureWorks.dbo.EmployeeOrderCount',`
	`SCHEMA = STANDARD`
	`),`
	`WSDL = DEFAULT,`
	`BATCHES = ENABLED,`
	`DATABASE = 'AdventureWorks'`
	`)`
	2. Select the statement you just typed.
	3. Press **F5** to create the HTTP Endpoint.
	Notice some key elements in the CREATE ENDPOINT statement. The location of the endpoint is specified by the PATH attribute. The STATE is set to STARTED. If this STATE parameter were omitted, the CREATE ENDPOINT statement would execute without error, but the endpoint would not serve requests. The authentication model is set to INTEGRATED. When the method is defined, the database object to invoke when this method is called is specified in the NAME attribute. The WSDL attribute allows the Web Service to expose its WSDL to consumers.

5. Access Endpoint WSDL. NOTE: The WSDL may take some time to appear. If it does not appear, or if you get a message that the schema is being generated, wait 30 seconds or so, close Internet Explorer, and try again.	**1.** Open Internet Explorer. **2.** Type the URL **http://localhost/AdvEnd?WSDL** and press **Enter**. Internet Explorer displays the Web Services Description Language (WSDL) XML document for the HTTP endpoint. WSDL describes the interface of a Web service, including elements such as method names, parameters, and return values. A WSDL document can be consumed by tools to generate a proxy that correctly initiates calls against the Web method. The **Add Web Reference** mechanism in Microsoft Visual Studio® .NET can consume WSDL, as you will see in the next exercise. **3.** Use Internet Explorer's **Edit | Find (on This Page)** dialog box to search for "GetEmployeeOrderCount". This takes you to the following XML node. `<xsd:element name="GetEmployeeOrderCount">` **4.** Scroll farther down the WSDL document to find the following XML.

Tasks	Detailed Steps
	<wsdl:operation name="GetEmployeeOrderCount"> Inspect these sections to see the WSDL that is specific to this endpoint. **5.** Close Internet Explorer.

Exercise 2
Consume HTTP Endpoints

Scenario

In this exercise, you will use Visual Studio 2005 to consume the HTTP endpoint that you created in the previous exercise.

Tasks	Detailed Steps
1. Create a Windows Application. NOTE: All of the code for this exercise is contained in the Visual Studio solution located at C:\SQL Labs\Lab Projects\Web Services Lab\ConsumeHTTP\ConsumeHTTP.sln. You can copy and paste from these files to complete any portion of the exercise.	**1.** From the **Start** menu, load **Microsoft Visual Studio 2005.** **2.** Click **File | New | Project**. **3.** In the **New Project** dialog box create a new Microsoft Visual C#® project, using the Windows Application template. **4.** For the project **Name**, enter **ConsumeHTTP**. **5.** Change the Location to **C:\SQL Labs\User Projects**. **6.** Click **OK** to create and open the new project.
2. Add a Web Reference to an HTTP Endpoint.	**1.** In the Solution Explorer, right-click the ConsumeHTTP project name, and then click **Add Web Reference.** **2.** In the **Add Web Reference** dialog box, enter the URL **http://localhost/AdvEnd?WSDL** and click **Go**. **3.** In the **Web reference name** text box, enter **SQLHTTP.** **4.** Click **Add Reference**. Adding a Web Reference to the project creates a proxy class automatically. This proxy class contains the types defined in the WSDL and also contains information on how to call methods exposed by the Web service. This eliminates the need to create and handle programmatically the SOAP and XML messages that are sent or received from the Web service.

3. Write code to call the Web service.	**1.** In the Solution Explorer, right-click **Form1.cs** and click **View Designer**.
	2. Double-click the form in the Designer to create an event handler for the form's **Load** event.
	3. Type the following code into the form's Load event handler.
	`SQLHTTP.AdvEnd ws = new SQLHTTP.AdvEnd();`
	`ws.Credentials =`
	`  new System.Net.NetworkCredential("Administrator", "Pass@word1");`
	`MessageBox.Show("Total Orders Placed by Employee 231: " +`
	`  ws.GetEmployeeOrderCount(231));`
	This code creates an instance of the proxy class and a network credential. The code then calls the GetEmployeeOrderCount() method, passing in a valid employee ID, and displays the results in a message box. The proxy class handles all processing of the SOAP and XML message and data serialization. Your code can call the Web service as it would call any standard class method.
	4. Press **F5** to build and run the project.

Tasks	Detailed Steps
	A message box displays the total number of orders taken by the specified employee, 360. **5.** Close the form when you're done. **6.** Close Visual Studio.