

Module 6: Storing Tabular Data in Azure

Demo: Implementing Azure Storage Tables

1. On the Start screen, locate and click the **Visual Studio 2017** tile.

Note: You might have to use the down arrow to locate the Visual Studio 2017 tile on your Start screen.

1. On the **File** menu, point to **New**, and then click **Project**.
2. In the **New Project** dialog box, perform the following steps:
3. Expand **Templates**, **Visual C#**, **Windows** and then click **Classic Desktop**.
4. Click the **Console Application** template.
5. In the **Name** box, enter the value **Contoso.Storage.Table**.
6. In the **Location** box, specify the value **AllFiles (F):\Mod06\Labfiles\Starter**
7. Click **OK**.
8. On the **View** menu, point to **Other Windows**, and then click **Package Manager Console**.
9. In the console, enter the following command:

```
Install-Package Microsoft.Data.Services.Client -Version 5.6.0;
```

10. Press Enter.
11. After execution of the first command is completed, enter the following command:

```
Install-Package WindowsAzure.Storage -Version 3.1.0.1;
```

12. Press Enter.
13. In the **Solution Explorer** pane, expand the **Contoso.Storage.Table** project.
14. Double-click the **Program.cs** file.
15. Add the following **using** statement at the top of the code file:

```
using Microsoft.WindowsAzure.Storage;
```

16. Add the following **using** statement at the top of the code file:

```
using Microsoft.WindowsAzure.Storage.Table;
```

17. At the end of the **Main** method and before the closing parenthesis, add the following code:

```
CloudTableClient tableClient =  
    CloudStorageAccount.DevelopmentStorageAccount.CreateCloudTableClient();
```

18. At the end of the **Main** method and before the closing parenthesis, add the following code:

```
CloudTable table = tableClient.GetTableReference("roster");
```

19. At the end of the **Main** method and before the closing parenthesis, add the following code:

```
table.CreateIfNotExists();
```

20. In the **Solution Explorer** pane, right-click the **Contoso.Storage.Table** project, point to **Add**, and then click **New Item**.

21. In the **Add New Item** dialog box, perform the following steps:

- a. Expand **Installed**, expand **Visual C# Items**, and then click **Code**.
- b. Click the **Class** template.
- c. In the **Name** box, type **Employee.cs**.
- d. Click **Add**.

22. In the **Employee** class, add the **public** accessor at the left side of the class definition:

```
class Employee
```

23. Verify that the updated class definition reads as follows:

```
public class Employee
```

24. Add the following **using** statement at the top of the code file:

```
using Microsoft.WindowsAzure.Storage.Table;
```

25. In the **Employee** class, add the **inheritance** statement : **TableEntity** at the right side of the class definition:

```
public class Employee
```

26. Verify that the updated class definition reads as follows:

```
public class Employee : TableEntity
```

27. Within the **Employee** class, add the following line of code:

```
public int YearsAtCompany { get; set; }
```

28. Within the **Employee** class, add the following method:

```
``` public override string ToString() {  
}
}```
```

1. Within the **ToString** method, add the following line of code:

```
return RowKey + "\t\t[" + YearsAtCompany + "];
```

2. In the **Solution Explorer** pane, expand the **Contoso.Storage.Table** project.

3. Double-click the **Program.cs** file.

4. At the end of the **Main** method and before the closing parenthesis, add the first Employee with a partition key of **IT** as shown below:

```
Employee first = new Employee { PartitionKey = "IT", RowKey = "ibahena", YearsAtCompany = 7 };
```

5. Add a second employee with a partition key of **HR**, as shown below:

```
Employee second = new Employee { PartitionKey = "HR", RowKey = "rreeves", YearsAtCompany = 12 };
```

6. Add a third employee with a partition key of **HR**, as shown below:

```
Employee third = new Employee { PartitionKey = "HR", RowKey = "rromani", YearsAtCompany = 3 };
```

7. At the end of the **Main** method and before the closing parenthesis, create a new **TableOperation** that inserts the first Employee as shown below:

```
TableOperation insertOperation = TableOperation.InsertOrReplace(first);
```

8. On the next line, use the **Execute** method on the table variable to execute the **TableOperation**, as shown below:

```
table.Execute(insertOperation);
```

9. At the end of the **Main** method and before the closing parenthesis, create a new **TableBatchOperation** with the following code:

```
TableBatchOperation batchOperation = new TableBatchOperation();
```

10. On the next line, add an **InsertOrReplace** operation to the batch for the second Employee, as shown below:

```
batchOperation.InsertOrReplace(second);
```

11. On the next line, add an **InsertOrReplace** operation to the batch for the third Employee, as shown below:

```
batchOperation.InsertOrReplace(third);
```

12. On the next line, use the **ExecuteBatch** method on the table variable to execute the **TableBatchOperation**, as shown below:

```
table.ExecuteBatch(batchOperation);
```

13. At the end of the **Main** method and before the closing parenthesis, create a string filter to retrieve only entities with a partition key of **HR** by using the **TableQuery.GenerateFilterCondition** static method:

```
string queryFilter = TableQuery.GenerateFilterCondition("PartitionKey",
QueryComparisons.Equal, "HR");
```

14. On the next line, create a new **TableQuery** and invoke the **Where** method by using the string filter, as shown below:

```
TableQuery<Employee> query = new TableQuery<Employee>().Where(queryFilter);
```

15. On the next line, write a header to the Console window, as shown below:

```
Console.WriteLine("HR Employees\n");
```

16. On the next line, use a **foreach** loop to iterate over the results of the query, as shown below:

```
``` foreach (Employee hrEmployee in table.ExecuteQuery(query)) {  
}  
```
```

1. Within the loop, write the **Employee** object to the Console window, as shown below:

```
Console.WriteLine(hrEmployee);
```

2. At the end of the main method and before the closing parenthesis, write a header to the Console window:

```
Console.WriteLine("\n\n\nIT Employee\n");
```

3. On the next line, create a new **TableOperation** to retrieve the single entity with a partition key of **IT** and row key of **ibahena**:

```
TableOperation retrieveOperation = TableOperation.Retrieve<Employee>("IT", "ibahena");
```

4. On the next line, execute the **TableOperation** by using the **Execute** method of the table variable and store the result in a *TableResult* variable, as shown below:

```
TableResult result = table.Execute(retrieveOperation);
```

5. On the next line, cast the **Result** property of the *TableResult* variable to an **Employee** object, as shown below:

```
Employee itEmployee = (Employee)result.Result;
```

6. On the next line, write the **Employee** object to the Console window, as shown below:

```
Console.WriteLine(itEmployee);
```

7. On the Start screen, click the **Internet Explorer** tile.

8. In the *Address bar* navigate to the following address:

<https://go.microsoft.com/fwlink/?LinkId=717179&clcid=0x409>

9. In the **Internet Explorer** download dialog box, click **Save**.

**Note:** The download of the *Azure Storage Emulator* executable typically takes around five minutes.

17. Click the **Windows File Explorer** icon in your Taskbar.

18. On the left navigation bar, expand the **This PC** node and click the **Downloads** node:

19. Double-click the **MicrosoftAzureStorageEmulator.msi** file to start the emulator.
20. In the **Microsoft Azure Storage Emulator Setup** wizard, select the checkbox next to the "**I accept the terms in the License Agreement**" statement.
21. Click the **Install** button to install the emulator.
22. Wait for the installer to complete.

**Note:** The installer can take between two to five minutes.

23. Click the **Finish** button to close the installer wizard.
24. On the Start screen, type **Azure Storage Emulator**.
25. Click the **Microsoft Azure Storage Emulator** tile.
26. After the command-line application is finished, close the open console window.
27. Switch to the Contoso.Storage.Table – Microsoft Visual Studio window.
28. On the **Debug** menu, click **Start Without Debugging**.
29. View the output in the console window.
30. Press any key to close the console window.
31. Close the **Contoso.Storage.Table – Microsoft Visual Studio** application.