AdventureWorks SkiResort

Demonstration Deployment & Setup

Contents

[PREREQUISITES 3](#_Toc474745776)

[CREATE AZURE RESOURCES 4](#_Toc474745777)

[TEST SQL SERVER 2016 CONNECTION 11](#_Toc474745778)

[SET UP A DATA SCIENCE CLIENT 14](#_Toc474745779)

[GET THE MACHINE LEARNING WEB SERVICE KEYS 17](#_Toc474745780)

[TEST THE WEBAPPs 19](#_Toc474745781)

[TEST THE WEB MOBILE APP 22](#_Toc474745782)

[ENABLE EXTERNAL SCRIPTS 26](#_Toc474745783)

[MODEL AND SAMPLE DATA GENERATION 28](#_Toc474745784)

[XAMARIN 33](#_Toc474745785)

[POWERBI 35](#_Toc474745786)

# PREREQUISITES

| Screen | Click Steps |
| --- | --- |
| **PREREQUISITES** | |
| *Disclaimer:*  *To deploy this demo and use it, we expect the user to have an intermediate knowledge of Microsoft Azure.* | 1. Visual Studio 2015 with Update 3.   <https://www.visualstudio.com/products/vs-2015-product-editions>   1. Lastest Microsoft Azure Sdk.   <https://azure.microsoft.com/en-us/downloads/>   1. .NET Core SDK Visual Studio Tools   <https://go.microsoft.com/fwlink/?LinkId=798481>   1. Microsoft Azure Subscription.   <https://azure.microsoft.com/en-us/free/>   1. gulp and bower installed as global.  * npm install bower -g * npm install gulp -g  1. Download [SQL Server 2016 Developer Edition](https://aka.ms/skiapp) 2. Power BI command line tool  * npm install powerbi-cli -g  1. Visual Studio updated with the latest Xamarin version. 2. Java Jdk 1.8   <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html> |

# 

# CREATE AZURE RESOURCES

| Screen | Click Steps |
| --- | --- |
| **AZURE RESOURCES** | |
|  | 1. Open Visual Studio 2015 with admin rights. |
|  | 1. Open the AdventureWorks.SkiResort solution. |
|  | 1. Open AdventureWorks.SkiResort solution. 2. Expand the deployment “deployment” solution folder. 3. Expand the SkiResort.Deploy project. 4. Click on the **Skiresort.parameters.json** file. |
|  | 1. By default, Stream Analytics uses a free pricing tier (Shared cluster). You can only create one Search service at the free pricing tier. **Change to “basic” or “standard” if you already have a free search service in your azure subscription**. |
|  | 1. The Azure ML Web Services provide multiple tiers that you can use to provision your billing plan. The dev/test tier is a tier that provides limited included quantities that allow you to test your experiment as new web service without incurring costs. **You can only create one Azure ML Web Services at the dev/test tier.** Change to “S1” if you already have a dev/test service in your azure subscription |
|  | The project contains an ARM template (skiresort.json) to create all the Azure Resources needed in the demo environments. |
|  | 1. Click on the **SkiResort.Deploy**. 2. Click on deploy. |
|  | 1. Add your Microsoft Azure account or choose the credentials that you want to use if you are already logged in. |
|  | 1. Choose your Azure Subscription. 2. Select the Resource group that you created before. 3. Choose the **skiresort.json** files as a deployment template. 4. Choose the **skiresort.parametes.json** files as template parameters. |
|  | 1. Click on Deploy. 2. Check the output window to follow the process. The process will take several minutes (20-30 minutes). |
|  | 1. After finishing the process**, the output window will show the URI of the webapps deployed and the connection string to the SQL Server VM that is created**. |
|  | 1. Copy the URL (Website basic). 2. Open a browser and paste the URL (The database and sample data is created the first time you navigate. The creation process may take a few minutes) 3. By default, the credentials are:    1. Username: skiresort    2. Password: P2ssw0rd@1 4. Copy the second URL (Website advanced). 5. Open a browser and paste the URL. 6. By default, the credentials are:    1. Username: skiresort    2. Password: P2ssw0rd@1 |
|  | 1. In your Azure subscription you will have something similar to the picture. |

# TEST SQL SERVER 2016 CONNECTION

| Screen | Click Steps |
| --- | --- |
| **TEST SQL SERVER VIRTUAL MACHINE** | |
|  | 1. Click on the virtual machine. 2. Click on “Connect”.   **NOTE:** You may receive an error telling that it is not possible to connect because the remote machine may be shut down. Wait a few moments and try again. This is because it is still restarting due to the SQL Server installation. |
|  | 1. Click on “Connect”.    1. Username: skiresort    2. Password: P2ssw0rd@SkiResort |
|  | 1. In the Virtual Machine, open SQL Server Management Studio. 2. Connect using SQL Server Authentication.    1. Username: skiresort    2. Password: P2ssw0rd@1   **NOTE**: The first time the Server will NOT contain any database. The first time that you run the web app in your local environment the database will be created. |
|  | 1. Open SQL Server 2016 Configuration Manager. 2. Click on Protocolos for MSSQLSERVER 3. Open SQL Server Configuration Manager. 4. Enable TCP/IP protocol. |
|  | 1. Click on SQL Server Services 2. Restart the SQL Server service. 3. Start the “LaunchPad” service if it´s stopped. |
|  | 1. Open SQL Server Management Studio in your **local computer**. 2. Try to connect to the remote SQL Server that is hosted on Microsoft Azure. (Your corporate firewall should allow SQL remote connections, port 1433) 3. Connect to the server using SQL Server Authentication.    1. Servername: <VM public DNS name>    2. Username: skiresort    3. Password: P2ssw0rd@1   **NOTE**: If there is any issue with these steps, connect to the server using RDP and enable remote connections in SQL Server 2016 |

# SET UP A DATA SCIENCE CLIENT

| Screen | Click Steps |
| --- | --- |
| **SET UP A DATA SCIENCE CLIENT** | |
|  | 1. If have not already done so, download the SQL Server 2016 Developer Edition from [here](https://aka.ms/skiapp).   You will want to set up an R development environment that is capable of connecting to the server for remote execution and deployment.   1. Run SQL Server setup. |
| Setup option for R Server Standalone | 1. On the Installation tab, click New R Server (Standalone) installation. |
|  | 1. Click Next until Feature Selection. 2. On the Feature Selection page, the following option should already be selected:    1. R Server (Standalone)   This option installs shared features, including open source R tools and base packages, and the enhanced R packages and connectivity tools provided by Microsoft R Services. |
|  | 1. Accept the license terms for downloading and installing Microsoft R Open. 2. Click on Next and install it. |
|  | 1. Open <https://www.visualstudio.com/en-us/features/rtvs-vs.aspx> 2. Click on “Download R Tools for VS”.   Note that if you install R Tools for Visual Studio, in some cases, the add-in will not recognize that the R runtime is installed and will offer to download and install another version of R, such as the 64 bit editions of R 3.2.1, 3.2.2 and 3.2.3 (CRAN R distributions), or the Microsoft R distributions (Microsoft R Open, Revolution R Open).  It is important that you do NOT use the provided links to install a different version of R. Instead, use one of the following options to register the currently installed version of the R runtime:   * Run the R utility, RSetReg, and locate the locally installed R runtime version. * Manually create the following registry key:   [HKEY\_LOCAL\_MACHINE\SOFTWARE\R-core\R\3.2.2.802 Microsoft R Server] "installPath"="<INSTALLATION\_FOLDER>" |

# GET THE MACHINE LEARNING WEB SERVICE KEYS

| Screen | Click Steps |
| --- | --- |
| **RESOURCE GROUP** | |
|  | 1. Go to your Microsoft Azure subscription. <https://portal.azure.com> 2. Click on “Resource groups”. 3. Search for the resource group that you created before. |
|  | 1. Open the azure portal and go to the resource group that you created before. 2. Click on the Machine Learning Web Service. |
|  | 1. Click on Machine Learning Web Service Management. |
|  | 1. Click on consume. 2. Copy the primary key and Request-Response URI to use later. |

# TEST THE WEBAPPs

| Screen | Click Steps |
| --- | --- |
| **RESOURCE GROUP** | |
|  | 1. Go to your Microsoft Azure subscription. <https://portal.azure.com> 2. Click on “Resource groups”. 3. Search for the resource group that you created before. |
|  | 1. Click on the resource group to view all of the existing resources.    1. SQL Server 2016 Virtual Machine.    2. WebApps.    3. Virtual Network.    4. Search Service.    5. Stream Analytics.    6. Event Hub.    7. Application Insights.    8. Cognetive Services.    9. AzureML    10. PowerBI    11. …. |
|  | 1. Click on the second website, the one that has the “adv” key in the its name. |
|  | 1. Click on Application Settings. 2. Update the anomaly service settings with the “primarykey” and “Request-Response” URI of the Machine Learning Web Service. |
|  | 1. Click overview and click on the URL to open the application. (The AdventureWorks.SkiResort database will be created in the SQL Server VM) |

# TEST THE WEB MOBILE APP

| Screen | Click Steps |
| --- | --- |
| **RUN THE WEB MOBILE APP IN YOUR LOCAL COMPUTER** | |
|  | 1. Open the AdventureWorks.SkiResort solution.   **NOTE**: The first time you open the Visual Studio solution will restore the web project dependencies. |
|  | 1. Expand the SkiResort.Web project. 2. Click on the “appsettings.json” file to update the anomaly detectation keys. |
|  | 1. Update your settings with the “primarykey” and “Request-Response” URI of the Machine Learning Web Service. |
|  | 1. Set the web project as StartUp project. 2. Click on F5 or in the “IIS Express” button. |
|  | 1. By default, the credentials are:    1. Username: skiresort    2. Password: P2ssw0rd@1 |

# ENABLE EXTERNAL SCRIPTS

| Screen | Click Steps |
| --- | --- |
| **ENABLE EXTERNAL SCRIPTS** | |
|  | 1. Click on the virtual machine. 2. Click on “Connect”. |
|  | 1. Click on “Connect”.    1. Username: skiresort    2. Password: P2ssw0rd@SkiResort |
|  | 1. In the Virtual Machine, open SQL Server Management Studio. 2. Connect using SQL Server Authentication.    1. Username: skiresort    2. Password: P2ssw0rd@1 |
|  | 1. Enable external scripts running this command:   sp\_configure 'external scripts enabled', 1;  RECONFIGURE;   1. Restart the SQL Server Service. |

# MODEL AND SAMPLE DATA GENERATION

| Screen | Click Steps |
| --- | --- |
| **DATA GENERATION** | |
|  | 1. Open the **SkiResort.DataGeneration.sln** solution. (src\SkiResort.DataGeneration directory) |
|  | **gen-skirentals** generates ski rentals records for the last few years into an SQL table.   1. Set “gen-skiretals” as StartUp project. 2. Click on App.config file. The ConnectionString uses the SQL Server hosted on Azure. |
|  | 1. Click on F5. 2. Wait to finish. |
|  | **gen-skilocations** simulates skiers movements around the mountain by continuously moving them from the bottom of chairlifts to the top and back. The location events are pushed to an event hub that is then consumed by Stream Analytics.   1. Set “gen-skilocations” as the StartUp project. 2. Click on App.config file to check that the EventHub connectionstring is configured. |
|  | 1. Click on “Ctrl + F5” to run the application without debugging.   The application will continuously send the events without stopping. |
|  | 1. Stream Analytics consumes the events to process and store them in three different outputs. |
|  | **geo-recomodel**: giving 2 inputs (restaurant list, user restaurant attendance) creates a recommendations model in the pre-built Cognitive Services recommendations API. It uses the recommendations data service created in the first steps.   1. Set “gen-recomodel” as StartUp project. 2. Click on App.config file to check that the connectionstring and recommendations key are configured. |
|  | 1. Click on “Ctrl + F5” to run the application without debugging. 2. Copy the **modelId**. You will need it later. |
|  | 1. After creating the model the application is able to read the RestaurantId and search for the service recommendations. 2. Click “1”. 3. Click “2”. 4. …. |
|  | **geo-restaurantsearch** runs all restaurants by the recommendations API, retrieves a list of recommended restaurants for each one, and pushes the list of recommended/related restaurants to the Azure Search index.   1. Set “gen-restaurantsearch” as the StartUp project. 2. Click on App.config file. 3. Update the RecoModelId with the value copied in the previous step. 4. Click on “F5”. |
|  | 1. Open SQL Server Management Studio. 2. Connect your database using SQL Server Authentication.    1. Username: skiresort    2. Password: P2ssw0rd@1 |
|  | 1. Run the store procedure “TrainRentalModel”. |

# XAMARIN

| Screen | Click Steps |
| --- | --- |
| **Configure your enviroment** | |
|  | 1. Open Visual Studio. 2. Go to Options>Xamarin>Other. 3. Check for updates. |
| enter image description here | 1. Click on Android Settings. 2. Be sure that the Java jdk 1.8 is used. |
|  | 1. Open Android Sdk Manager. |
|  | 1. Install the latest version of the Android Sdk tool, Sdk platform tools and Sdk build tools. |

# POWERBI

| Screen | Click Steps |
| --- | --- |
| **Create embed token** | |
|  | 1. Go to the “reports” directory. |
|  | 1. Open a new terminal window.   This directory contains a the .powerbirc file that has the config values to connect to your powerbi embedded collection. |
|  | 1. Open the Azure Portal, go to the SkiResort Resource Group and get the workspaceId. |
| powerbi get-reports -w <workspaceId> | 1. List of all reports within a workspace: |
|  | 1. Copy the reportId. 2. Copy the token to update the Xamarin app configuration. |
| powerbi create-embed-token -w <workspaceId> -r <reportId> | 1. Create a Power BI embed token. 2. Copy the token to update the Xamarin app configuration. |
|  | 1. Open the AdventureWorks.SkiResort solution. 2. Click on the Config.cs file. (SkiResort.XamarinApp project) 3. Update the reportId and token values. |