AdventureWorks SkiResort

Demonstration Deployment & Setup

Presenter Script  
May 2016

Draft — 05/24/2016

Contents

[PREREQUISITES 3](#_Toc455143792)

[CREATE DATA SERVICES 4](#_Toc455143793)

[CREATE AZURE RESOURCES 7](#_Toc455143794)

[SET UP SQL SERVER R SERVICES (In-Database) 13](#_Toc455143795)

[TEST SQL SERVER 2016 CONNECTION 16](#_Toc455143796)

[SET UP A DATA SCIENCE CLIENT 19](#_Toc455143797)

[TEST THE WEBAPPs 22](#_Toc455143798)

[TEST THE WEB MOBILE APP 24](#_Toc455143799)

[MODEL AND SAMPLE DATA GENERATION 27](#_Toc455143800)

# 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 2.   <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 |

# CREATE DATA SERVICES

| Screen | Click Steps |
| --- | --- |
| **DATA SERVICES** | |
|  | **Anomaly Detection API** runs different types of detectors on time series data and gives alerts when unusual patterns in the time series are detected. |
|  | 1. Go to <https://azure.microsoft.com/en-us/marketplace/partners/aml-labs/anomalydetection/> 2. Click on “Sign up”. |
|  | 1. Click on “Sign up”. |
|  | 1. Accept the agreement. 2. Click on “Sign up”. |
|  | Recommendations API by Azure Machine Learning helps your customer discover items in your catalog. Customer activity on your website is used to recommend items and to improve conversion in your digital or physical store. |
|  | 1. Go to https://azure.microsoft.com/en-us/marketplace/partners/amla/recommendations/ 2. Click on “Sign up”. 3. Accept the agreement. 4. Click on “Sign up”. |
|  | 1. Click on “My Account”. 2. Click on “Account Keys”. 3. Copy these values to use later. |

# CREATE AZURE RESOURCES

| Screen | Click Steps |
| --- | --- |
| **AZURE RESOURCES** | |
| Unblock button in Properties dialog | 1. Got to the AdventureWorks.SkiResort GitHub repository.   <https://github.com/Microsoft/AdventureWorksSkiApp> |
|  | 1. Clone the repository. |
|  | 1. Run Visual Studio 2015 **with admin rights.** |
|  | 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. Update the **DataMarketKey** property with your data market key. 2. Update the **DataMarketUser** property with your data market users. |
|  | 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. |
|  | 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. Create a new Resource Group in the location where you want to create the resources. 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**.   DO NOT open the URLs. |
|  | 1. Copy the first 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 |

# SET UP SQL SERVER R SERVICES (In-Database)

| Screen | Click Steps |
| --- | --- |
| **SET UP SQL SERVER R SERVICES (In-Database)** | |
|  | 1. Click on the resource group to view all of the existing resources. |
|  | 1. Click on the virtual machine. 2. Click on “Connect”. |
|  | 1. Click on “Connect”.    1. Username: skiresort    2. Password: P2ssw0rd@SkiResort |
|  | 1. Run SQL Server 2016 setup. (C:\SQLServer\_13.0\_Full) 2. On the **Installation** tab, click **New SQL Server stand-alone installation or add features to an existing installation** |
|  | 1. Click on “Add features to an existing installation” |
|  | 1. Click on R Services (In-Database)   This option configures the database services used by R jobs and installs the extensions that support external scripts and processes.   1. On the page, **Consent to Install Microsoft R Open**, click **Accept**. 2. When installation is complete, restart the computer. |

# TEST SQL SERVER 2016 CONNECTION

| Screen | Click Steps |
| --- | --- |
| **TEST SQL SERVER VIRTUAL MACHINE** | |
|  | 1. Click on the virtual machine. 2. Click on “Connect”. |
|  | 1. Click on “Connect”.    1. Username: skiresort    2. Password: P2ssw0rd@1 |
|  | 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 Configuration Manager. 2. Enable TCP/IP protocol. |
|  | 1. Restar the SQL Server service. |
|  | 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. Download the SQL Server 2016 Developer Edition setup in your local computer.   You will want to set up an R development environment that is capable of connecting to the server for remote execution and deployment. |
| Setup option for R Server Standalone | 1. Run SQL Server setup. 2. On the Installation tab, click New R Server (Standalone) installation. |
|  | 1. 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>" |

# 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. |
|  | 1. Check the two Web Apps and navigate to them. (Their names start with “web”. 2. Click on the webapp (in one of two existing webapps) |

# 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.    1. Without any additional steps, the settings file must be updated with your own deployment.    2. The mobile web app is configured to use the SQL Server hosted on Azure. |
|  | 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 |

# 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 AzureML 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 is configured and data market credentials 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”. |