

Azure SQL Database

Hands-On Lab

Configuration Guide  
Version 2.2

1/05/2016

Prepared for: Guy Haycock

Azure SQL Database

Prepared by: Prowess Consulting

Table of Contents

[Architectural Overview 3](#_Toc425000559)

[Lab-Configuration Prerequisites 3](#_Toc425000560)

[Connecting Your Azure Account 5](#_Toc425000561)

[Deploying a New WingtipTickets Tenant Environment 7](#_Toc425000562)

[Appendix: Explanation of New-WTTEnvironment Parameters 12](#_Toc425000563)

[Appendix: Explanation of Web.config Properties 13](#_Toc425000564)

Table of Contents

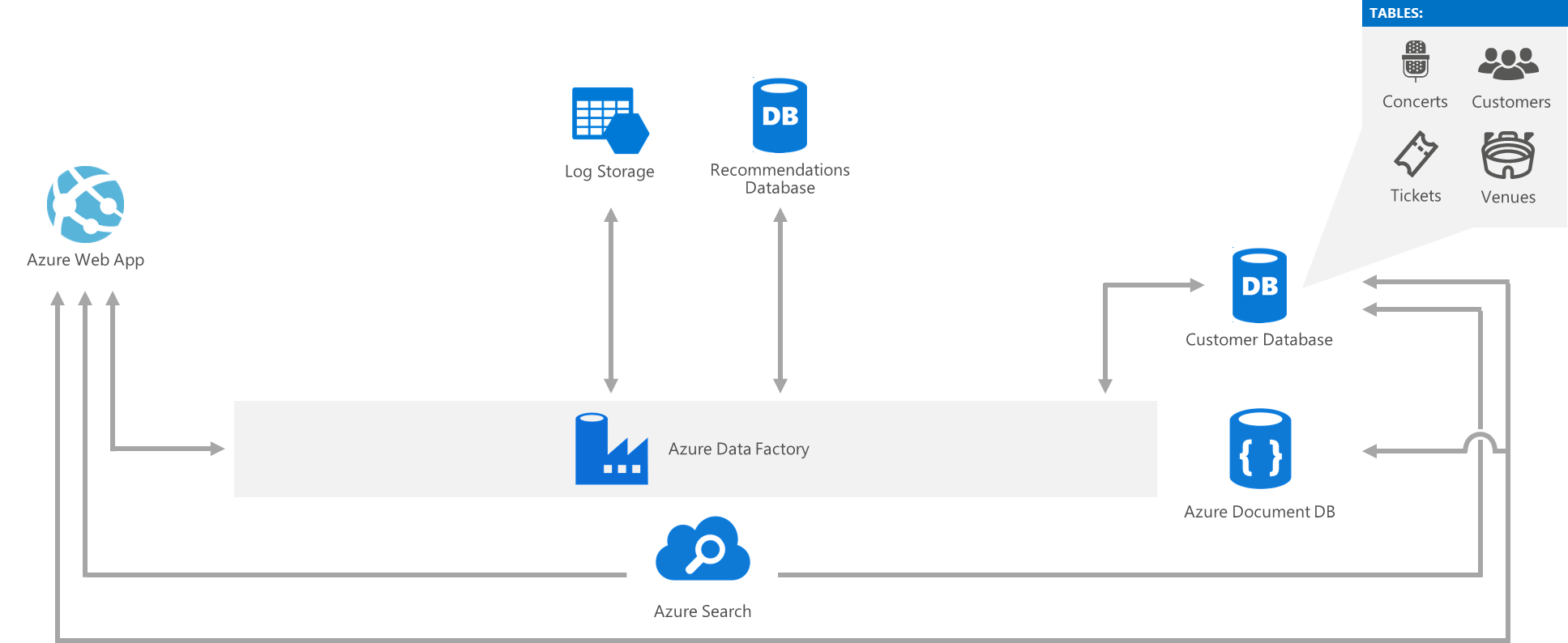
[Architectural Overview 3](#_Toc435182984)

[Lab-Configuration Prerequisites 3](#_Toc435182985)

[Connecting Your Azure Account 5](#_Toc435182986)

[Deploying a New WingTipTickets Tenant Environment 7](#_Toc435182987)

# Architectural Overview



**Figure 1** Overall architecture of the various lab components

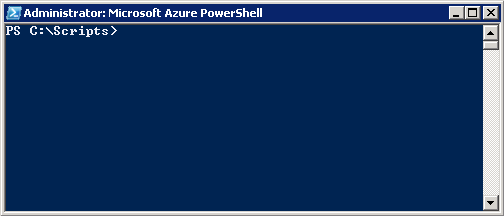
# Lab-Configuration Prerequisites

**Table 1** Lab-configuration prerequisites and setup instructions

| **Prerequisite** | **Setup instructions** |
| --- | --- |
| **Microsoft account** | 1. If you don’t already have a Microsoft account, browse to [http://account.live.com](http://account.live.com/). 2. Click **Sign up now**. |
| **Microsoft Azure account** | 1. If you don’t already have an Azure account, browse to <http://azure.microsoft.com/en-us/pricing/free-trial/>. 2. Click **Try it now**. |
| **Microsoft Azure PowerShell** | 1. Browse to <http://azure.microsoft.com/en-us/downloads/>. 2. Under **Command-line tools**, **Windows PowerShell**, click **Install**. 3. Verify that the version being installed is at least 0.9.8.1 (released October 13, 2015). |
| **Deployment scripts** | 1. Unzip **scripts.zip** to a local folder (for example, C:\scripts). |
| **Microsoft Power Query for Excel 2013 (optional)** | 1. If Power Query for Excel is not installed, and you’d like to complete the Auditing section in the hands-on lab (HOL) manual, download and install Power Query from <https://www.microsoft.com/en-us/download/details.aspx?id=39379>. |
| **Microsoft SQL Server Management Studio (optional)** | 1. If SQL Server Management Studio is not installed, and you’d like to run queries against the databases, download and install SQL Server Management Studio from <https://www.microsoft.com/en-us/download/details.aspx?id=42299>. The package is listed as **SQLManagementStudio\_<X86/X64>\_ENU.exe**. |
| **Microsoft Visual Studio (optional)** | 1. If Visual Studio is not installed, and you’d like to explore any of the source code, download and install Visual Studio from <http://go.microsoft.com/?linkid=9832446&clcid=0x409>. 2. Unzip **sourcecode.zip** to a local folder (for example, C:\sourcecode). |
| **Microsoft Azure .NET software-development kit (SDK) (if installing Visual Studio)** | 1. Browse to <http://azure.microsoft.com/en-us/downloads/>. 2. Under **SDKs**, **.NET**, select the installer for your version of Visual Studio (for example, **VS 2013 Install**). |

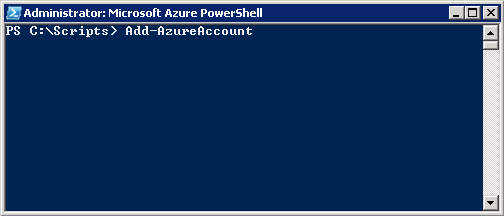
# Connecting Your Azure Account

1. Launch a Microsoft Azure PowerShell session as an administrator (click **Run as administrator**), and then browse to the folder where you’ve saved the scripts.



**Figure 2** A Microsoft Azure PowerShell command-line interface session

1. Connect to your Azure account by typing **Add-AzureAccount**.



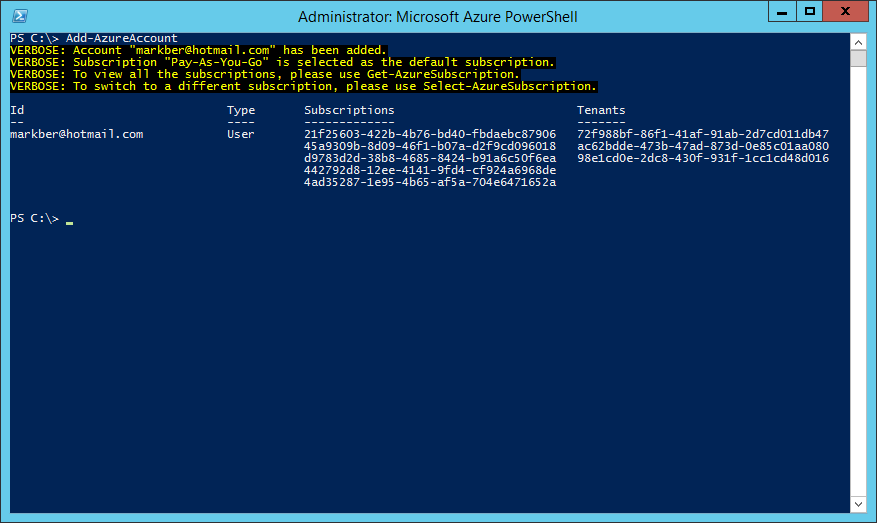
**Figure 3** Connect to your Azure account through Azure PowerShell

1. Type your Azure account credentials.



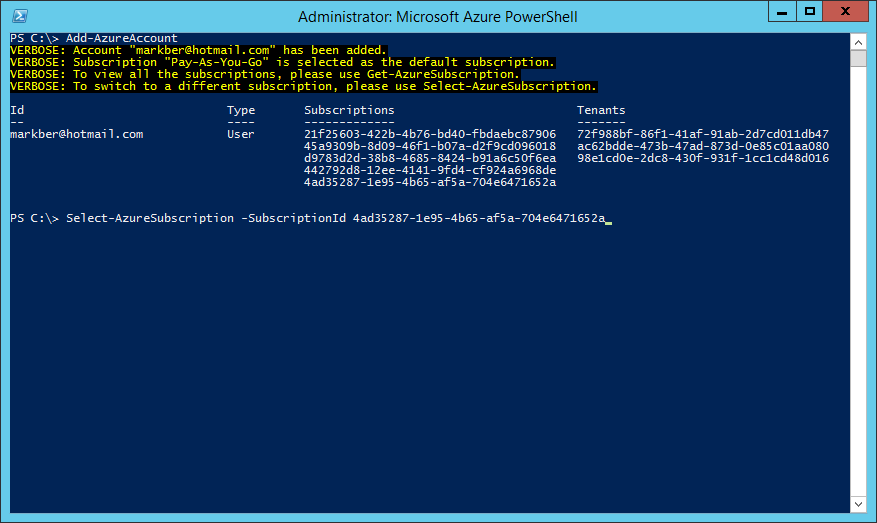
**Figure 4** Type your Azure account credentials

Your account should be linked to your subscription(s), as shown in Figure 5.



**Figure 5** Subscription confirmation in Azure PowerShell

1. Type **Select-AzureSubscription –SubscriptionId** *your subscription id*, as shown in Figure 6.



**Figure 6** You can also change or supply your Azure credentials through the Select-AzureSubscription command in Azure PowerShell

# Deploying a New WingTipTickets Tenant Environment

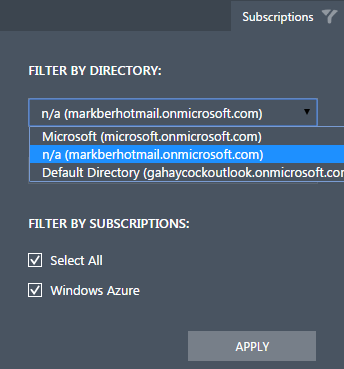
1. Run the following commands to load the Azure PowerShell script:
   1. **PS C:\Scripts>Set-ExecutionPolicy -Scope LocalMachine -ExecutionPolicy Unrestricted -Force**
   2. **PS C:\Scripts>Unblock-file .\New-WTTEnvironment.ps1**
   3. **PS C:\Scripts>. .\New-WTTEnvironment.ps1**



**Figure 7** Deploy a new tenant environment using the New-WTTEnvironment Azure PowerShell script

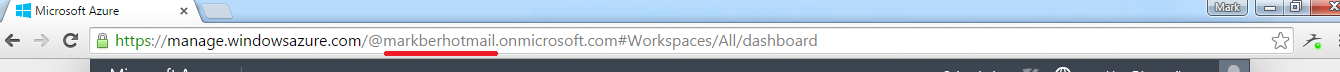
**Note:** To load the Azure PowerShell script, you must type a period and then a space before the path to the script: .\New-WTTEnvironment.ps1

1. To verify the script is loaded, type **New-WTTEnvironment -W**, and then press Tab, which should autocomplete to WTTEnvironmentApplicationName.
2. Verify whether you’re using a custom Azure Active Directory domain by browsing to <https://manage.windowsazure.com>.
3. If you have more than one subscription, select the correct subscription from the **Subscription** drop-down menu at the top right, and then click **Apply**. Note, this option won’t appear if you only have one subscription.

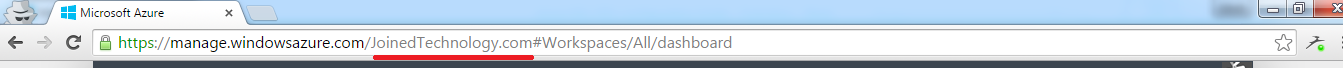


**Figure 8** Selecting a subscription on the Azure portal (https://manage.windowsazure.com)

1. Once the correct subscription is loaded, note the value of the directory that is being used in the URL after manage.windowszure.com/. If it looks like your email address, for example [dataplatformuser1@hotmail.com](mailto:dataplatformuser1@hotmail.com), with the @ character and the end of the domain name removed (in this example, “.com”), you’re using the default Azure Active Directory domain for your subscription.



If it looks like a custom domain name is being used, such as joinedtechnology.com in this example, then make a note of it, as you’ll need this value during the setup to ensure authentication against the correct directory.



If you’re using a microsoft.com email address, then you’ll likely see Microsoft as the directory name and no additional action is required.

Depending on your network connection, setup should take no more than 15 minutes.

**Note**: -WTTEnvironmentApplicationName is the most important value because it’s used to prefix Azure resources—for example, storage accounts, web apps, and database servers.

Because this HOL uses the Azure public cloud, it is crucial that you choose a name that is unique in order for the lab to work. For example: your initials followed by julieandtheplantes (xx*julieandtheplantes*).

1. If you are not using a custom Azure Active Directory domain (as in step 5), run the cmdlet using syntax similar to this (remember to substitute the *xx* in *xx*julieandtheplantesfor your initials): **New-WTTEnvironment -WTTEnvironmentApplicationName *xx*julieandtheplantes**

If you are using a custom Azure Active Directory domain (as in step 5), then you’ll need to add a parameter for your custom domain similar to the following: **New-WTTEnvironment –WTTEnvironmentApplicationName *xx*julieandtheplantes –AzureActiveDirectoryTenantName JoinedTechnology.com**

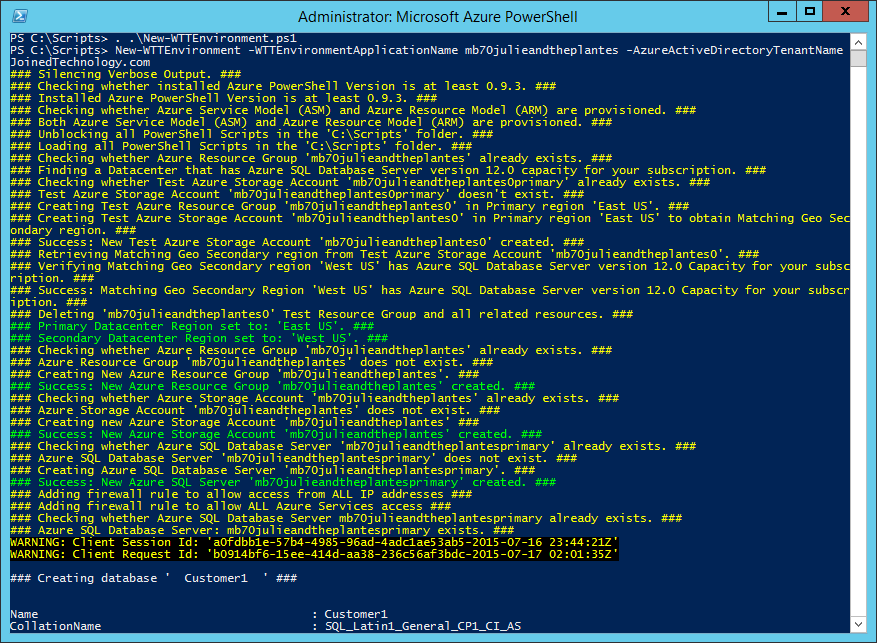


**Figure 9** Example of creating a new deployment in Azure PowerShell that uses the default Azure Active Directory domain for the subscription

**Note:** During setup, Azure PowerShell will output a number of success messages in green and informational messages in yellow.



**Figure 10** Azure PowerShell success and informational messages that you might encounter during setup (this example shows creating a new deployment that uses the default Azure Active Directory domain for the subscription)



**Figure 11** Azure PowerShell success and informational messages that you might encounter during setup (this example shows creating a new deployment that uses a custom Azure Active Directory domain for the subscription)

**Note:** If you have issues setting up the environment, you’ll need to review the errors that are returned to determine which Azure PowerShell function is causing an issue and, subsequently, what is causing the error.

The most common issues are due to name conflicts, webapp-package upload failures, and subscription quota limits.

For name conflicts, you can remove a deployment by running: **PS C:\Scripts> Remove-WTTEnvironment -WTTEnvironmentApplicationName *xx*julieandtheplantes**. This will delete any resources that were created by the New-WTTEnvironment cmdlet. Then try to re-run the New-WTTEnvironment cmdlet again as in step 6, using a different   
-WTTEnvironmentApplicationName (for example, *xx1*julieandtheplantes).

For package upload failures, re-running the same command that you previously ran in step 6 will try the upload again.

For pay-as-you-go subscription quota-limit-related issues, check the following:

* Typically there is a limit of six Azure SQL Database servers. Make sure there are no more than four Azure SQL Database servers in your subscription before running the New-WTTEnvironment cmdlet.
* Typically there is a limit of one free Azure Search service. Make sure there are no Azure Search services in your subscription before running the New-WTTEnvironment cmdlet.
* For other issues, please refer to [FAQ section](#FAQ) of this document.

1. Once the deployment is completed, open a browser and browse to [http://*xx*julieandtheplanets.trafficmanager.net](http://xxjulieandtheplanets.trafficmanager.net) (remember to replace *xx* with your initials).



**Figure 12** Web site running correctly on the primary web app

## 

## Appendix: Explanation of New-WTTEnvironment Parameters

**Table 2** New-WTTEnvironment parameters

| **Parameter** | **Purpose** | **Default Value** |
| --- | --- | --- |
| -WTTEnvironmentApplicationName | Name that will differentiate your WingTipTickets tenant environment from others running in the Azure public cloud (in this example, *xx*julieandtheplantes) |  |
| -AzureSqlDatabaseServerAdministratorUserName | Database server-administrator user name | developer |
| -AzureSqlDatabaseServerAdministratorPassword | Database server-administrator password | P@ssword1 |
| -AzureSqlDatabaseServerVersion | Azure SQL Database server version | 12.0 |
| -AzureSqlDatabaseName | Name of the tenant database | Customer1 |
| -AzureWebSiteWebDeployPackagePath | Path to the Azure Web App Web Deploy packages | <unzip location>\Scripts\Packages |
| -AzureWebSitePrimaryWebDeployPackageName | Primary Web App Web Deploy package name | primarypackage.zip |
| -AzureWebSiteSecondaryWebDeployPackageName | Secondary Web App Web Deploy package name | secondarypackage.zip |
| -WTTEnvironmentPrimaryServerLocation | Azure datacenter region | <auto configured based on capacity> |
| -AzureActiveDirectoryTenantName | Custom Azure Active Directory domain name |  |

## Appendix: Explanation of Web.config Properties

**Table 3** Web.config property names and descriptions

|  |  |  |
| --- | --- | --- |
| **Property name** | **Description** | **Value** |
| TenantEventTypeGenre | Tenant event type used to dynamically theme the site; valid values include: pop, rock, and classical | Pop |
| TenantEventName | Tenant (application) name, used to dynamically theme the site | Set from  –WTTEnvironmentApplicationName <value> |
| PrimaryDatabaseServer | Name for the primary Azure SQL Database server where concerts, customers, venues, and tickets tables exist | Set from  –WTTEnvironmentApplicationName <value> + primary |
| SecondaryDatabaseServer | Name of the Azure SQL Database server that is configured as the target server (also known as the secondary) for geo-replication | Set from  –WTTEnvironmentApplicationName <value> + secondary |
| DatabaseUserName | User name to be used for all application-related connections to the Azure SQL Database servers | Set from  -AzureSqlDatabaseServerAdministratorUserName |
| DatabaseUserPassword | Password to be used for all application-related connections to the Azure SQL Database servers | Set from  -AzureSqlDatabaseServerAdministratorPassword |
| TenantDbName | Name for the Azure SQL Database tenant database, where concerts, customers, venues, and tickets tables exist | Set from  -AzureSqlDatabaseName |
| SearchServiceName | Name of the Azure Search service that indexes a view for the tables in the TenantDbName database | Set from  –WTTEnvironmentApplicationName <value> |
| SearchServiceKey | Azure Search service key | Dynamically retrieved during setup |

## Frequently Asked Questions

1. **Q**: I am getting ”account expired” or “400 bad request” errors in the deployment after the Add-AzureAccount step, or an account is listed that doesn’t belong to me when calling Get-AzureSubscription in Azure PowerShell.

**A**: You need to type **Get-AzureAccount | Remove-AzureAccount** in Azure PowerShell. Close the Azure PowerShell console, open a new Azure PowerShell, and then try the **Add-AzureAccount** command again.

1. **Q**: I am getting a “Key Not Found in Dictionary” error when running the Add-AzureAccount command.

**A**: Try the following steps:

* **Remove-AzureAccount**
* **Clear-AzureProfile**
* **Add-AzureAccount**

1. **Q**: I have an active Microsoft Developer Network (MSDN) subscription, and it seems I have enough resources, but my provisioning failed somehow, and I am not sure why.

**A**: Please check that you have Azure Search in your subscription by going to [http://portal.a**zure.com**](http://portal.azure.com). Log on with your account, and then click **Browse All**. If Azure Search is not shown as available, you need to switch to a subscription that does include Azure Search service.