

Azure SQL Database

Hands-On Lab

Configuration Guide  
Version 2.1

07/22/2015

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](#_Toc424917738)

[Lab-Configuration Prerequisites 3](#_Toc424917739)

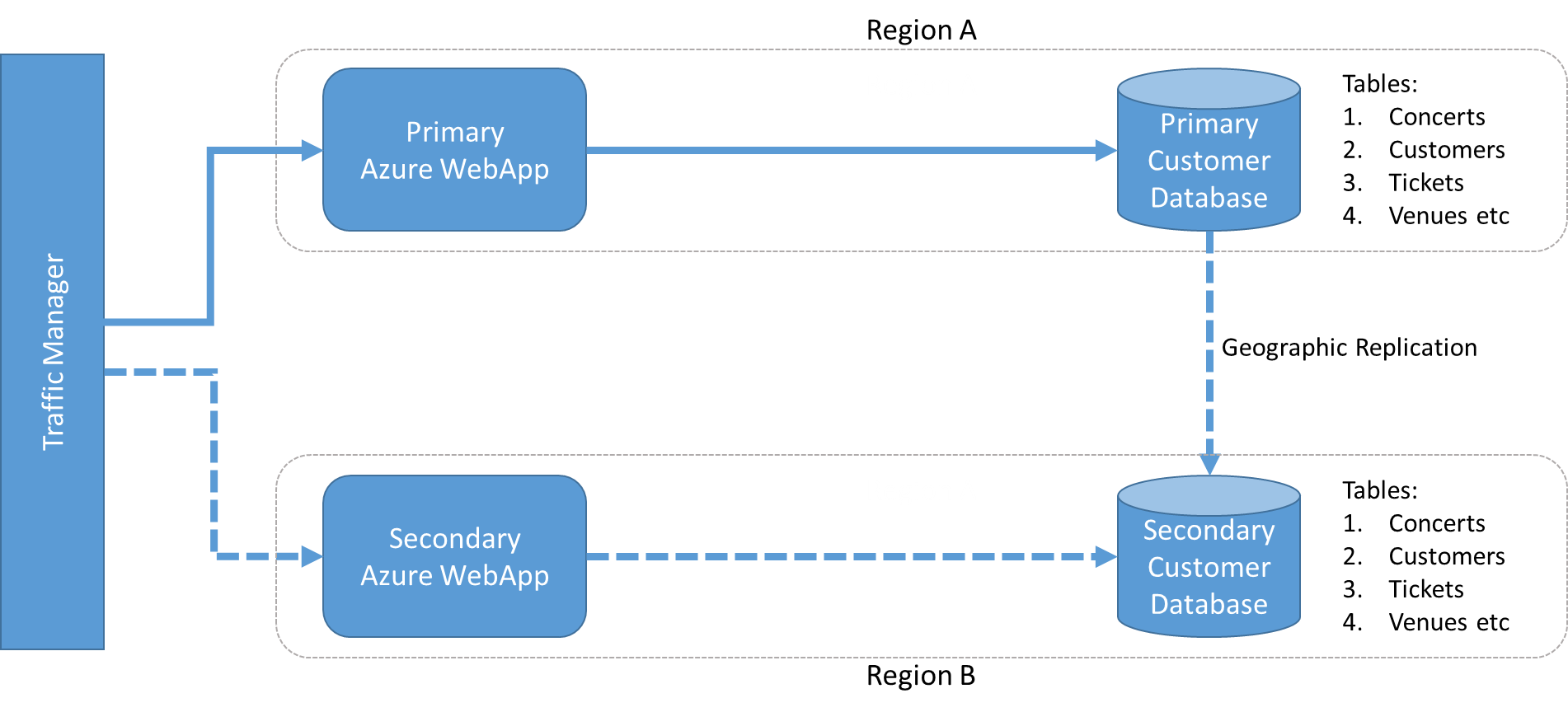
[Connecting Your Azure Account 5](#_Toc424917740)

[Deploying a New WingtipTickets Tenant Environment 7](#_Toc424917741)

[Appendix: Explanation of New-WTTEnvironment Params 12](#_Toc424917742)

[Appendix: Explanation of Web.config Properties 13](#_Toc424917743)

# Architectural Overview



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

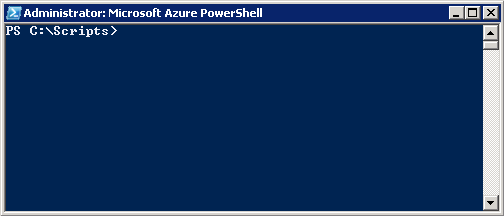
# Lab-Configuration Prerequisites

**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 20, 2015). Note: if the release is newer than 0.9.8.1, browse to <https://github.com/Azure/azure-powershell/releases> for the Azure PowerShell installation. |
| **Deployment scripts** | Unzip scripts.zip to a local folder (for example, c:\scripts). |
| **Microsoft Power Query for Excel (optional)** | 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)** | 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)** | 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>.  Unzip sourcecode.zip to a local folder (for example, c:\sourcecode). |
| **Microsoft Azure .NET 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 Visual Studio e.g. **VS 2013 Install**. |

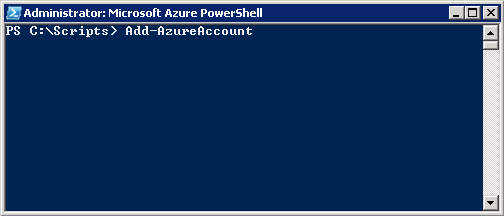
# Connecting Your Azure Account

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



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

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



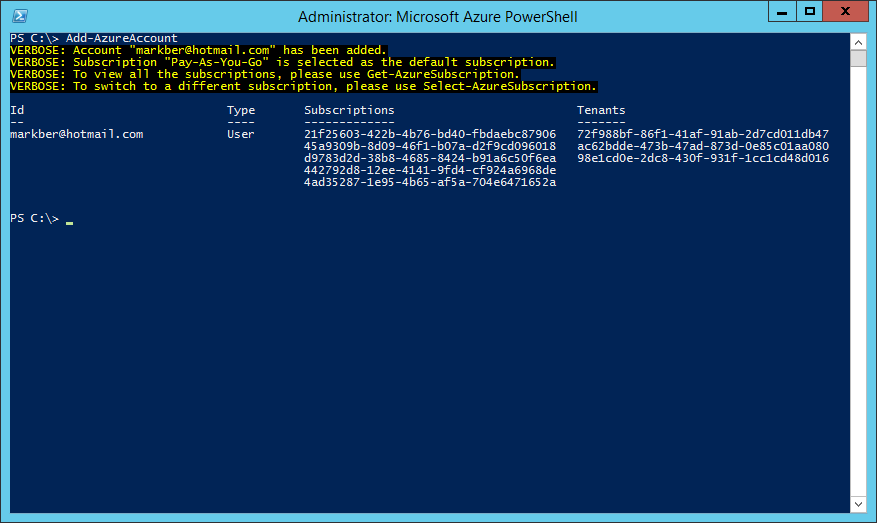
**Figure 3** Connect to your Microsoft Azure account through Microsoft Azure PowerShell

1. Type your Azure account credentials.



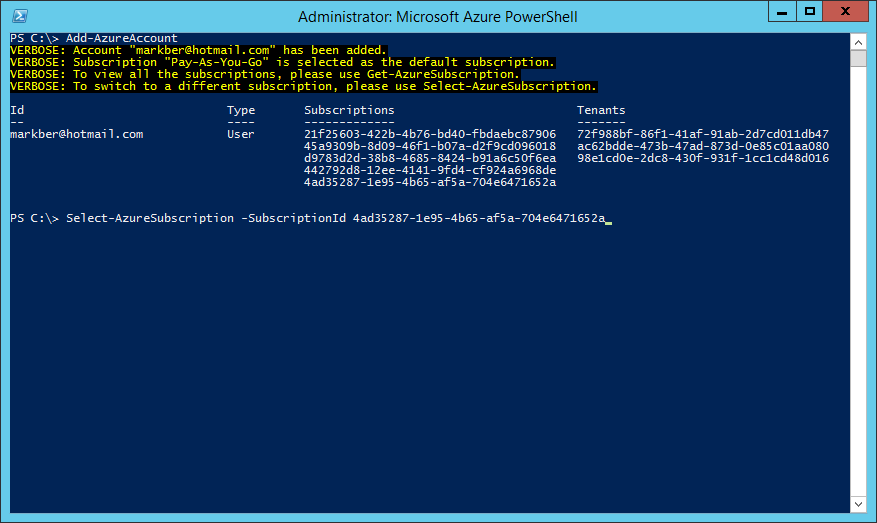
Your account should be link to your subscription(s) like below:

**Figure 4** Type your Microsoft Azure account credentials



**Figure 5** Subscription confirmation in Microsoft Azure PowerShell

1. Select-AzureSubscription –SubscriptionId ‘YOUR SUBSCRIPTON ID’ (like below)



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

# Deploying a New WingtipTickets Tenant Environment

1. Run the following commands to load the Microsoft Azure PowerShell script:  
   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



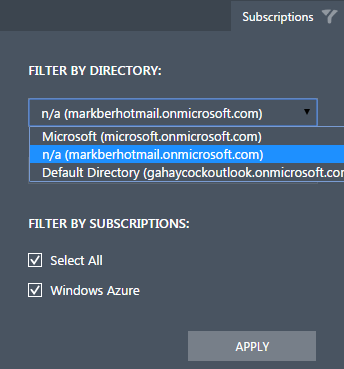
**Figure 7** Deploy a new tenant environment using the New-WTTEnvironment Microsoft Azure PowerShell script

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

To verify the script is loaded, type **New-WTTEnvironment -W** and then press Tab, which should autocomplete WTTEnvironmentApplicationName.

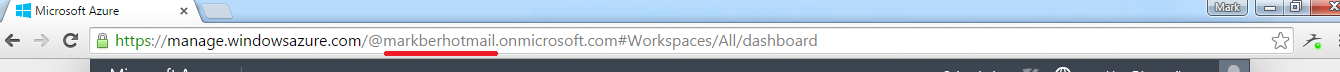
1. Verify whether you’re using a custom Azure Active Directory Domain by browsing to <https://manage.windowsazure.com>

If you have more than one subscription, select the correct subscription from the Subscription dropdown on the top right, and click Apply. Note, this option won’t appear if you only have one subscription:

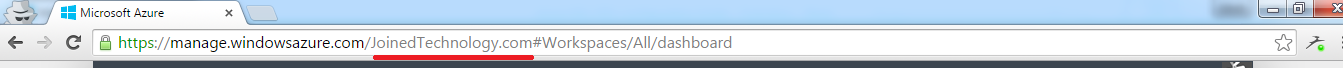


**Figure 8** Selecting a subscription on the Azure Portal (https://manage.windowsazure.com)

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 e.g. [markber@hotmail.com](mailto:markber@hotmail.com) with the “@” sign removed as well as the end of the domain name removed, in this example “.com”, then you’re using the default Azure AD 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 value most important because it’s used to prefix Azure resources e.g. storage account, webapps, and database servers.

Because this hands-on lab uses the Azure public cloud, it is crucial that you choose a name that is unique in order for the lab work.

* For example: your initials followed by *julieandtheplantes* e.g. mb*julieandtheplantes*

1. If you’re not using a custom Azure Active Directory Domain (per step 5), run the cmdlet using syntax similar to this (remember to substitute the *mb* in *mbjulieandtheplantes for your initials*):  
   New-WTTEnvironment -WTTEnvironmentApplicationName *mbjulieandtheplantes*

If you’re using a custom Azure AD Domain (per step 5), then you’ll need to add a parameter for your custom Domain similar to this:

New-WTTEnvironment –WTTEnvironmentApplicationName mbjulieandtheplantes –AzureActiveDirectoryTenantName JoinedTechnology.com

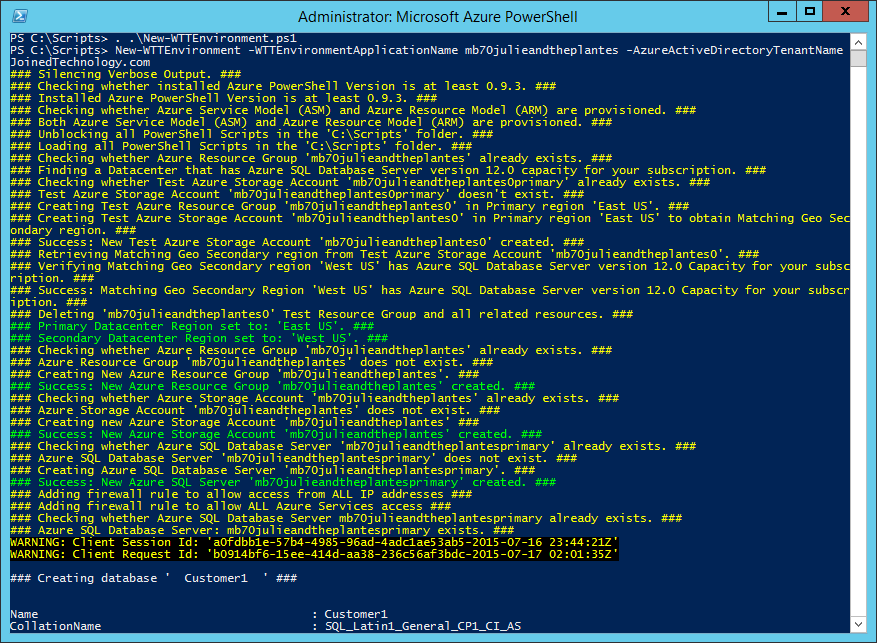


**Figure 9** Example creating a new deployment in Microsoft Azure PowerShell that uses the default Azure AD Domain for the subscription

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



**Figure 10** Microsoft Azure PowerShell success and informational messages that you might encounter while running setup. Example for creating a new deployment that uses the default Azure AD Domain for the subscription



**Figure 11** Microsoft Azure PowerShell success and informational messages that you might encounter while running setup. Example for creating a new deployment that uses a Custom Azure AD 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 PowerShell function is causing an issue and subsequently determine the cause of 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 mbjulieandtheplantes

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 e.g. mb1julieandtheplantes

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 6 Azure SQL Database Servers. Make sure there are no more than 4 Azure SQL Database Servers in your subscription before running New-WTTEnvironment
* Typically there is a limit of 1 Free Azure Search Service. Make sure there are no Search Services in your subscription before running New-WTTEnvironment
* For other issues, please refer to [FAQ section](#FAQ) of the doc.

1. Once the deployment is completed, open a browser and browse to <http://mbjulieandtheplanets.trafficmanager.net> (remember to replace mb with your initials).



**Figure 12** Web site running correctly on the primary WebApp

# 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 Microsoft Azure public cloud (in this example, *mbjulieandtheplantes*) |  |
| -AzureSqlDatabaseServerAdministratorUserName | Database server administrator username | 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 WebApp WebDeploy Packages | <unzip location>\Scripts\Packages |
| -AzureWebSitePrimaryWebDeployPackageName | Primary WebApp WebDeploy package name | primarypackage.zip |
| -AzureWebSiteSecondaryWebDeployPackageName | Secondary WebApp WebDeploy 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 4** 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 Microsoft 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 (aka 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 which Indexes a View for the tables in the TenantDbName database | Set from  –WTTEnvironmentApplicationName <value> |
| SearchServiceKey | Azure Search service key | Dynamically retrieved during setup |

**FAQ:**

(1) Q: I am getting--”… account expired” or “400 bad request” error in the deployment after Add-AzureAccount step or I have account that doesn’t belong to me when calling Get-AzureSubscription in PowerShell.

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

(2) Q: I am getting “Key Not found in Dictionary” error when Add-AzureAccount

A: Please try the following steps:

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

(3) Q: I have my active MSDN subscription and it seems have enough resources, but my provision failed somehow and I am not sure why.

A: Please check that you have Azure Search in your subscription by going to <http://Portal.Azure.com> and log in with your account and select “Browse All”. If Azure Search is not shown up as available Azure Search Service in the item, then you need to switch to a subscription that does have Azure Search Service