Contents

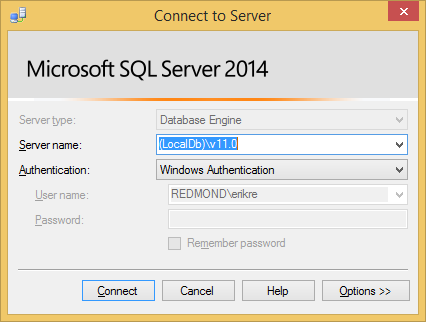
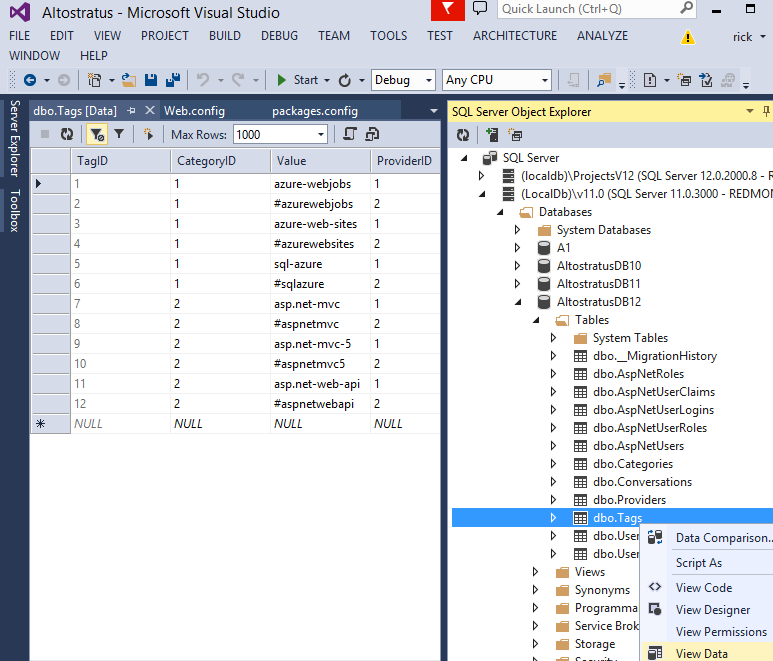
[Test locally. 1](#_Toc413063498)

[Test on Azure using VS to publish 2](#_Toc413063499)

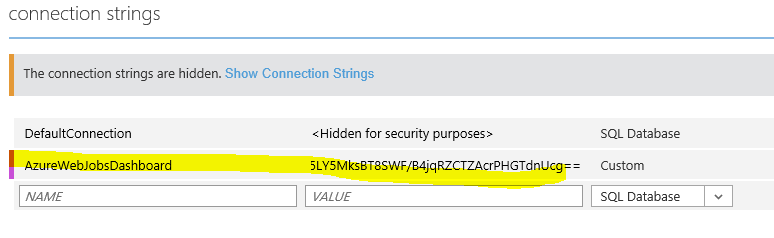
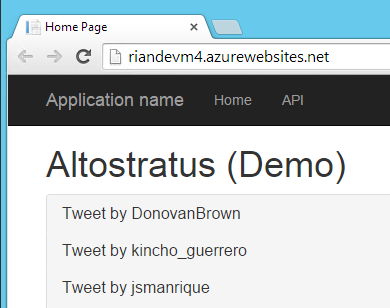
[Using PS scripts on Azure 3](#_Toc413063500)

[*<WebSiteName>.Publishsettings* file 4](#_Toc413063501)

# Test locally.

1. In PMC: run Update-Database which will seed the DB.
2. Set Altostratus.WebJob as startup project – then run it.
3. Open the **SQL Server Object Explorer** (SSOX) and add a new local database by typing the following for the **Server name**:  
   (LocalDb)\v11.0  
     
   
4. Verify the seed method ran and you have data by expanding the database tables to find the **Tags** table (dbo.Tags). Right-click the **Tags** to and select **View Data**.
5. Verify the seed method ran and you have data  
   
6. In **Solutions Explorer**, right-click **Altostratus.Website** and select **Set as StartUp Project**.
7. IMPORTANT – Verify the "DefaultConnection" string is the same in the Website project as in the WebJobs project (Altostratus.WebJob\App.config and \Altostratus.Website\Web.config). Right now they are different. This isn’t a problem on Azure, but it is a problem when the connection string is spread over 3 projects. Right now, the easiest way to consolidate migrations is to just change the connection string and start over.
8. If you want to test user preferences, run **Altostratus.MvcWeb** and make sure it’s got the correct connection string.

# Test on Azure using VS to publish

1. Build the **Altostratus.WebJob** project before deployment. Right-click **Altostratus.WebJob** and select **Publish as Azure Web Job**. Check the **Web Publish Activity** output window, it can fail for many reasons.
2. From the portal, click on the website, select the config tab, then under conn strings add the AzureWebJobsDashboard and key from the app.conf file, Key starts with DefaultEndpointsProtoco  
   <add name="AzureWebJobsDashboard" connectionString="DefaultEndpointsProtocol=https;AccountName=tdykstrastorage1;AccountKey=zbvE4WFmkHUroAXf9Lbe6YaLZILp9JpxET2tTMWQtqbaNaLJyPE5LY5MksBT8SWF/B4jqRZCTZAcrPHGTdnUcg==" />  
     
     
     
   Type is optional, but custom makes sense. BE SURE to click Save at the bottom.
3. Still under WebSites, click on the WEBJOBS tab and at the bottom of the page, click **Run Once**. That will seed/populate the DB. You can copy the connection string and look at the data from SSOX – or just deploy the web site and look at the data.
4. In solution explorer, right click Altostratus.Website > Publish. Make sure you publish to the same web site as you published the WJ in the step above. In a few seconds you’ll see something like:  
   

# Using PowerShell scripts on Azure

Note: Per Vijay Ramakrishnan [vramak@microsoft.com](mailto:vramak@microsoft.com) to prevent the following warning from VSO build summary

Summary

**Debug | Any CPU**

0 error(s), 1 warning(s)

$/CodeFirst-Altostratus/Altostratus/Altostratus.sln - 0 error(s), 1 warning(s), View Log File

cid:image001.png@01D06E05.C96FE5F0d:\a\src\CodeFirst-Altostratus\Altostratus\packages\Microsoft.Web.WebJobs.Publish.1.0.2\tools\webjobs.console.targets (149): WebJob schedule for AltoStratWJ will not be created. WebJob schedules can only be created when the publish destination is an Azure Website

$/CodeFirst-Altostratus/Altostratus/Altostratus.sln compiled

1 test run completed - 100% pass rate

No Code Coverage Results

1. You need to - You can change the runmode in webjob-publish-settings.json

  "runMode": "Scheduled" to  "runMode": "Continuous"

but don’t use continuous – or it will run, then restart continuously. This is just to get a clean screen shot.

1. One time task. Start **Windows PowerShell ISE** and run the following command: (Note: On new VM’s, launch PS ISE in administrator mode by right-clicking and selecting “Run as administrator”.  
   Set-ExecutionPolicy -ExecutionPolicy Unrestricted --
2. Then, exit PowerShell and restart **PowerShell ISE** normally (not as admin). You only need to do this once.
3. To run the following scripts before unattended security is configured, you need to run   
    Add-AzureAccount # see http://azure.microsoft.com/en-us/documentation/articles/powershell-install-configure/
4. Next, run the following from PowerShell:  
   Get-AzurePublishSettingsFile   
     
   This command will prompt you to login and download your Azure publish settings file. Save your publish settings file locally.
5. Run the following PowerShell command using your publish settings file:  
   Import-AzurePublishSettingsFile "C:\Users\YourName\Downloads\Prototype5-1-2015-credentials.publishsettings"  
     
   PowerShell will display the Id, Name, Environment, Account and Properties associated with your Azure publishing settings.
6. In **Solution Explorer**, right-click the solution and open it in **File Explorer**. Change to the PowerShell\Notes directory and copy the path.
7. In the PowerShell ISE window, change to the PowerShell\Notes directory:  
   cd “C:\Users\user\Documents\Visual Studio 2013\Altostratus\PowerShell\Notes”
8. Create all the credentials files to store secrets  
   .\CreateFBCredential.ps1 -Password "11a227e1258cf825a623afadb162ece5"  
   .\CreateGoogCredential.ps1 -Password "v9xaOV71J6LPtoTVGpqD9ryT"   
   .\CreateTwitterCredential.ps1 -Password "HOAuQSdxd3Px66oRMZ826AFvaqrXQ3JMJXOp0KQvUy5Qx8HFw3"
9. Run the CreateSQLpwCredential.ps1 command file: (note SQL pw is only for rick’s gmail account)  
   .\CreateSQLpwCredential.ps1 -SQLsrvPW 'Pa$$w0rd' -user 'rickand'
10. Supply the value for the SQL Server password (SQLsrvPW) and to create the credentials file (CreateSQLpwCredential.ps1.credential) needed in the next step.   
      
    If you want to test this step, you can run CreateSQLdb.ps1. It will use the credential file. Put a break point on the remove-db (last step) if you want to see the DB in azure.
11. Copy this credentials file (CreateSQLpwCredential.ps1.credential) from the Altostratus\PowerShell\Notes directory to the Altostratus\PowerShell directory. If you like, open and look at the xml file.
12. In PowerShell ISE change to the Altostratus\PowerShell.
13. Run the PowerShell script New-AzureWebsitewithDB\_WebJob.ps1:  
    .\New-AzureWebsitewithDB\_WebJob.ps1  
      
    This will create the SQL database, the storage account, the website and the website publish script. Right now you can run it with no arguments and it will ***not*** create a DB server but it will create a DB using my DB server. That’s OK for now. The script also creates a *website-environment.xml* file which contains the storage account and keys, the DB connection string and password. The *website-environment.xml* file is need to create the .pubxml file. The *<WebSiteName>.Publishsettings* file has all the publish settings needed. See typical *<WebSiteName>.Publishsettings* file at the end of this document.
14. Build the entire solution (this will pull down the NuGet files).
15. Run Publish-AzureWebsite.ps1 passing in the path of the webjob .csproj file  
    ./Publish-AzureWebsite.ps1 -ProjectFile "..\Altostratus.WebJob\Altostratus.WebJob.csproj"
16. Run *Publish-AzureWebsite.ps1* passing in the path of the website .csproj file. That will correctly deploy to Azure. PS C:\TfsOnline\Altostratus\PowerShell>   
    .\Publish-AzureWebsite.ps1 ..\Altostratus.Website\Altostratus.Website.csproj

## *<WebSiteName>.Publishsettings* file

<publishData>

<publishProfile profileName="riande462149633web - Web Deploy" publishMethod="MSDeploy" publishUrl="riande462149633web.scm.azurewebsites.net:443" msdeploySite="riande462149633web" userName="$riande462149633web" userPWD="J2DwLBTv9JuMiqmdWB4NHwaM2GBkCJouWC2swxsi9ble6FZa74okl4jrZieD" destinationAppUrl="http://riande462149633web.azurewebsites.net" SQLServerDBConnectionString="Server=tcp:pow0bmj01i.database.windows.net,1433;Database=riande462149633web\_db;User ID=user1@pow0bmj01i;Password=Pa$$w0rd;Trusted\_Connection=False;Encrypt=True;Connection Timeout=30;" mySQLDBConnectionString="" hostingProviderForumLink="" controlPanelLink="http://windows.azure.com" webSystem="WebSites" targetDatabaseEngineType="sqlazuredatabase" targetServerVersion="Version100">

<databases>

<add name="riande462149633web\_db" connectionString="Server=tcp:pow0bmj01i.database.windows.net,1433;Database=riande462149633web\_db;User ID=user1@pow0bmj01i;Password=Pa$$w0rd;Trusted\_Connection=False;Encrypt=True;Connection Timeout=30;" providerName="System.Data.SqlClient" type="Sql" targetDatabaseEngineType="sqlazuredatabase" targetServerVersion="Version100" />

</databases>

</publishProfile>

<publishProfile profileName="riande462149633web - FTP" publishMethod="FTP" publishUrl="ftp://waws-prod-bay-003.ftp.azurewebsites.windows.net/site/wwwroot" ftpPassiveMode="True" userName="riande462149633web\$riande462149633web" userPWD="J2DwLBTv9JuMiqmdWB4NHwaM2GBkCJouWC2swxsi9ble6FZa74okl4jrZieD" destinationAppUrl="http://riande462149633web.azurewebsites.net" SQLServerDBConnectionString="Server=tcp:pow0bmj01i.database.windows.net,1433;Database=riande462149633web\_db;User ID=user1@pow0bmj01i;Password=Pa$$w0rd;Trusted\_Connection=False;Encrypt=True;Connection Timeout=30;" mySQLDBConnectionString="" hostingProviderForumLink="" controlPanelLink="http://windows.azure.com" webSystem="WebSites" targetDatabaseEngineType="sqlazuredatabase" targetServerVersion="Version100">

<databases>

<add name="riande462149633web\_db" connectionString="Server=tcp:pow0bmj01i.database.windows.net,1433;Database=riande462149633web\_db;User ID=user1@pow0bmj01i;Password=Pa$$w0rd;Trusted\_Connection=False;Encrypt=True;Connection Timeout=30;" providerName="System.Data.SqlClient" type="Sql" targetDatabaseEngineType="sqlazuredatabase" targetServerVersion="Version100" />

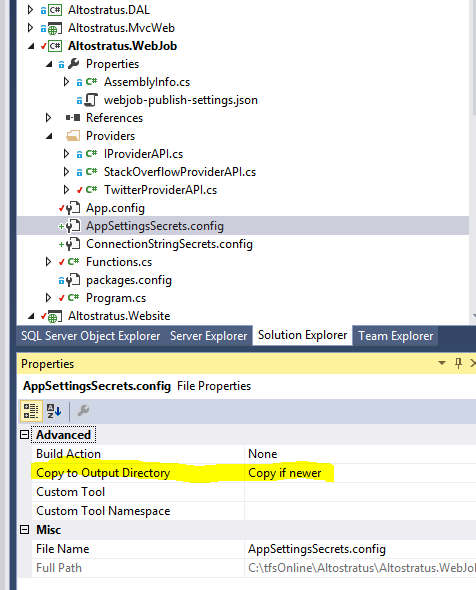
</databases>

</publishProfile>

</publishData>

# Secrets Notes

For RTM, we must empty out the AppSettingsSecrets.config file.

On the AppSettingsSecrets.config file, you must change the property from Copy to Output Dir: Do not copy to copy if newer.  
  


# Transform notes

From Web Publish Activity:  
Transformed Web.config using C:\TfsOnline\Altostratus\Altostratus.Website\Web.Release.config into obj\Release\TransformWebConfig\transformed\Web.config.

Auto ConnectionString Transformed Views\Web.config into obj\Release\CSAutoParameterize\transformed\Views\Web.config.

Auto ConnectionString Transformed obj\Release\TransformWebConfig\transformed\Web.config into obj\Release\CSAutoParameterize\transformed\Web.config.

The above web.config contains:  
<connectionStrings>

<add name="DefaultConnection" connectionString="$(ReplacableToken\_DefaultConnection-Web.config Connection String\_0)" providerName="System.Data.SqlClient" />

</connectionStrings>