Tenant Analytics

A ‘SaaS in a Box’ Tutorial

# Purpose of this tutorial

This tutorial uses Elastic jobs to run analytics queries that are distributed across all the tenants in the Wingtip catalog and returns the results into a database named tenantanalytics on the catalog server. This database can then be queried to extract insights buried in the day-to-day operational data of all the tenants. As an output of the job, a table will be created from the results-returning queries inside the tenant analytics database created as part of getting started.

# Tenant Operational Analytics pattern

One of the great opportunities with SaaS applications is to leverage the rich tenant data that is stored in the cloud to gain insights into the operation and usage of your application, and your tenants. This can guide feature development, usability improvements and other investments in the app and platform. Accessing this data when it’s in a single multi-tenant database is easy, but not so easy when distributed at scale across potentially thousands of databases. One approach to accessing this data is to use Elastic jobs, which enables result-returning query results from job execution to be captured in an output database and table.



Elastic Jobs private preview

The ‘new’ Elastic Jobs is integrated as a feature in Azure SQL Database and requires no additional Azure services, compared to the Public Preview customer-hosted version of Elastic jobs. In scope for the Private Preview is PowerShell to create the job account, and T-SQL to create and manage jobs. Public Preview will broaden capabilities to Portal, PowerShell and REST APIs in addition to new features.

In this tutorial, a new job account database and job account will be created. The tutorial then walks you through creating jobs to deploy new reference data and manage indexes.

Prerequisites

1. Subscription must be registered to use the Elastic jobs Private Preview. Send email for [SaaSFeedback@microsoft.com](mailto:SaaSFeedback@microsoft.com) to request registration. Once your subscription is confirmed registered, you can also verify registration by calling the Get-AzureRmProviderFeature cmdlet with the following parameters:
   * Get-AzureRmProviderFeature -ProviderName Microsoft.Sql -FeatureName sqldb-JobAccounts
2. To update your Azure PowerShell cmdlets to include the new Elastic jobs APIs, you must download and install the ‘updated’ latest AzurePowerShell.msi located here: <https://github.com/jaredmoo/azure-powershell/releases>

# Setup

Download and extract **WTPLearningModules.zip** to a convenient folder.

Deploy the **WTP Application.** Ensurethe catalog is initialized using the Demo Assistant app. See the Introduction to the WTP SaaS Application tutorial for deployment instructions.

**SSMS** can be used to explore database schema and execute SQL queries directly.

**PowerShell ISE** is recommended to execute scripts and follow their execution in debug mode.

**PowerShell Tips**

* Open and configure demo- scripts in the PowerShell ISE.
* Use F5 to run the script (using F8 is not advised as the $PSScriptRoot variable is not evaluated when running snippets of a script).
* Use F9 to set a breakpoint to let you trace the script in debug mode to see how it works
* Use F10 to step through the script, F11 to step into a function, and Shift-F11 to step out.

# Setup

1. Update the user configuration file used by all tutorial scripts. Update again if you redeploy the app.
   1. Open ...\Learning Modules\UserConfig.psm1 in **PowerShell ISE**
   2. Modify **$userConfig.ResourceGroupName** to the resource group used for the deployed app.
   3. Modify **$userConfig.Name** to the User name used for the deployed app.

# Walkthrough

Getting Started - Deploy a database that will be used for tenant analytics results

This tutorial requires you to have a database deployed to capture the results from job execution of scripts which contain results-returning queries. Let’s create a database called tenantanalytics for this purpose.

1. Open …\Learning Modules\Provision and Catalog\Operational Analytics\Tenant Analytics\**Demo-TenantAnalyticsDB**.ps1 in PowerShell ISE.
2. Execute the script using **F5**.   
   This calls the Deploy-TenantAnalyticsDB.ps1 script which creates the tenant analytics database.

## Exercise 1: Create a scheduled job to retrieve tenant analytics about ticket purchases

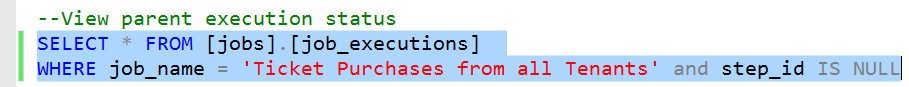
This script creates a job to retrieve ticket purchase information from all tenants. Once aggregated into a single table, you can gain rich insightful metrics about ticket purchasing patterns across the tenants.

1. Open SSMS and connect to the catalog-<USER>.database.windows.net server
2. Open the file …\Learning Modules\Provision and Catalog\Operational Analytics\Tenant Analytics\TicketPurchasesfromAllTenants.sql
3. Modify <WtpUser>, use the user name used when you deployed the WTP app
4. Right click, select **Connection**, and connect to the catalog-<WtpUser>.database.windows.net server, if not already connected
5. Ensure you are connected to the jobaccount database and press **F5** to run the script

* **sp\_add\_target\_group** creates the target group name TenantGroup, now we need to add target members.
* **sp\_add\_target\_group\_member** adds a *server* target member type which deems all databases within that server (note this is the customer1-<WtpUser> server containing the tenant databases) at time of job execution should be included in the job.
* **sp\_add\_job** creates a new weekly scheduled job called “Ticket Purchases from all Tenants”
* **sp\_add\_jobstep** creates the job step containing T-SQL command text to retrieve all the ticket purchase information from all tenants and copy the returning result set into a table called AllTicketsPurchasesfromAllTenants
* The remaining views in the script display the existence of the objects and monitor job execution.

1. Re-run the query view parent execution status until the job has completed

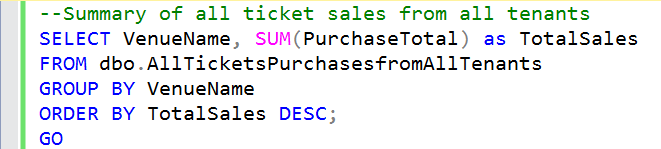
* select \* from [jobs].[job\_executions] where job\_name = ' Ticket Purchases from all Tenants' and step\_id IS NULL
* Review the status value from the lifecycle column to monitor the status. Once, Succeeded, the job has successfully finished on all tenant databases and the two additional databases containing the reference table.



## Exercise 2: View the summary of ticket purchases from all venues

This script creates a job to retrieve ticket purchase information from all tenants. Once aggregated into a single table, you can gain rich insightful metrics about ticket purchasing patterns across the tenants.

1. Open SSMS and connect to the catalog-<USER>.database.windows.net server
2. Right click, select Connection, and connect to the catalog-<WtpUser>.database.windows.net server, if not already connected
3. Open the file …\Learning Modules\Provision and Catalog\Operational Analytics\Tenant Analytics\Results-TicketPurchasesfromAllTenants.sql to view the summary ticket sells from all venues
4. Ensure you are connected to the tenantanalytics database and press **F5** to run the script



Congratulations, you have now created an scheduled job to run a sample distributed query across using Elastic jobs across all the tenant databases in the WTP app.

Additional resources

* To learn more about Elastic Jobs, see …\Learning Modules\Operational Analytics\Tenant Analytics\Elastic Jobs private preview (User Guide).docx