Restore Single Tenant

A SaaS-in-a-Box Tutorial

# Purpose of this tutorial

The Wingtip SaaS app is built using a single-tenant data model, with each venue (tenant) having their own database. One of the benefits of this model is that it is easy to restore a single tenant’s data in isolation without impacting other tenants. This tutorial showcases two data recovery patterns using PowerShell scripts – restoring data from a prior point into a parallel database (side-by-side), which can be used by the tenant for review and for compliance or auditing, and restoring data in-place, used to recover the tenant to a prior point if the tenant has accidentally corrupted their data in some way.

# Introduction to the SaaS tenant restore pattern

Planning for business continuity and disaster recovery is essential for a robust SaaS business and may be required for compliance reasons. A simple, repeatable, tenant data restore, and recovery process is a key element of such a plan. Databases can become corrupted for many reasons – a recovery process that leaves customers offline without access to their data for extended periods of time, or worse with no option to recover from corrupted data, could be disastrous. In this tutorial, you’ll explore two simple patterns for restoring tenant data.



In the first pattern, data is restored from a selectable point into a parallel database, and the tenant is then given access to the data alongside their production data. This pattern allows the tenant app to review the restored data and potentially use it to selectively overwrite current data values. It’s up to the app designer to determine just how sophisticated the data recovery options offered by the apps should be –simply being able to see the data as it was at some prior point alone may be valuable in some scenarios. If the database uses Active Geo-Replication, we recommend copying the required data from the restored copy into the original database. If you replace the original database with the restored database, you will need to reconfigure and resynchronize Active Geo-Replication (which can take a long time for a large database).

In the second pattern, which assumes that the tenant has suffered a serious, probably self-inflicted, loss or corruption of data, the tenant’s production database is restored to a prior point. In this pattern, the tenant is taken offline briefly, while the database is restored and then brought back online. The prior copy of the database is deleted but can be restored from again if the database needs to be restored again to an earlier point. A variation of this pattern could rename the database instead of deleting it, although renaming the database offers no additional advantage in terms of data security.

# 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.

## Getting Started

To demonstrate the recovery scenarios, the tutorial requires the creation and removal of tickets. In this getting started section, the scripts being executed will create ticket purchases and then remove the ticker purchases from a specific venue to simulate a ‘oops’ scenario being explored in the walk through exercises.

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.
2. Open the following scripts in **PowerShell ISE**
   1. ...\Learning Modules\Business Continuity and Disaster Recovery\RestoreTenant\Demo-RestoreTenant.ps1
   2. …\Learning Modules\Business Continuity and Disaster Recovery\RestoreTenant\Restore-TenantInParallel.ps1
   3. ...\Learning Modules\Business Continuity and Disaster Recovery\RestoreTenant\Restore-TenantInPlace.ps1
3. Navigate to **Demo-RestoreTenant.ps1**.
4. Run ticket generator script that will buy tickets for events for all registered tenant venues.
   1. **Modify $DemoScenario** to **1** to select the ‘Purchase tickets for events’ scenario.
   2. Execute using **F5**
      1. At this point ticket purchases will have been generated for all venues registered on the Wingtip platform. The ticket generator always leaves the last event with no tickets sold, which allows the event to be deleted. You can generate tickets again if you want to try deleting another event at the same venue.
5. Delete event with no ticket sales from the Contoso Concert Hall Venue. This deleted event will be restored later in the tutorial.
   1. **Modify $DemoScenario** to **2** to select the ‘delete event with no ticket sales’ scenario.
   2. Execute using **F5**
   3. This script opens the Contoso Concert Hall events page in your browser, and then separately deletes an event from Contoso Concert Hall. Additionally, it sets a recovery point (5 minutes before event deletion) that will be used in the restore scripts later in this tutorial.
   4. Verify that the event is deleted by refreshing the events page on your browser and looking for the deleted event that was output to the console.

# Walkthrough

## Exercise 1: Restore a tenant database in parallel with the production database

This exercise will restore the Contoso Concert Hall venue to a point in time before the event was  
deleted above. The *Restore-TenantInParallel.ps1* script creates a parallel tenant database, and a parallel catalog entry both named *ContosoConcertHall\_old*. This pattern of restore is best suited for recovering from a minor data loss or for compliance and auditing recovery scenarios. It is also the recommended approach if Active Geo-Replication is being used.

1. Complete the steps in the ‘Getting Started’ section before starting this exercise.
2. **Navigate to** **Demo-RestoreTenant.ps1** file in PowerShell ISE
3. **Modify** **$DemoScenario** – change this value to **3**. This selects the ‘restore tenant in parallel scenario’ in the demo script.
4. Execute using **F5**.
5. The script restores the tenant database in a parallel instance to 5 minutes before the event deletion that occurred in the 'Getting Started' section. It creates a new database at that point in time, removes any existing catalog metadata that exists in this database, and adds the database to the catalog under the *ContosoConcertHall\_old* entry.

You have now restored the deleted event under a new entry *ContosoConcertHall\_old.* The demo script will open the events page for the restored tenant in your browser. Inspect the event listing in the browser to confirm that the deleted event has been restored.

Note that exposing the restored tenant as a separately accessible tenant entry is unlikely to be how you would provide a tenant access to restored data, but serves to illustrate the restore pattern.

In reality, you would probably only retain this restored database for a defined period. You can delete the restored tenant entry once you are finished by calling the Remove-RestoredTenant.ps1 script.

1. **Modify** **$DemoScenario** to **4** to select the ‘remove restored tenant scenario’.
2. **Execute** **using** **F5**
3. The *ContosoConcertHall\_old* entry is now deleted from the catalog. You can close the events page for this tenant in your browser.

## Exercise 2: Restore a tenant in place, replacing the existing tenant database

This exercise will restore the Contoso Concert Hall tenant to a point in time before the event was deleted. The *Restore-TenantInPlace* script restores the current tenant database to a new database pointing to a previous point in time, and deletes the original database. This pattern of restore is best suited for recovering from serious data corruption as there may be significant data loss that the tenant would have to accommodate.

1. Complete the steps in the ‘Getting Started’ section before starting this exercise.
2. **Navigate to** **Demo-RestoreTenant.ps1** file in PowerShell ISE
3. **Modify** **$DemoScenario** to **5** to select the ‘restore tenant in place scenario’.
4. Execute using **F5**.

The script restores the tenant database to a point 5 minutes before the event deletion that occurred in the 'Getting started' section. It does this by first taking the tenant Contoso Concert Hall offline so there are no further updates to the data. Then, a parallel database is created by restoring from the restore point and named with a timestamp to ensure the database name does not conflict with the existing tenant database name. Next, the old tenant database is deleted, and the newly-restored database is renamed to the original database name. Finally, Contoso Concert Hall is brought online to allow the app access to the restored database.

At this point you have restored the database prior to the point the event was deleted. The demo script will open the events page for Contoso Concert Hall in your browser, where you can scroll the event listing to the end to confirm that the event has been restored.