

Azure SQL Database – Auditing

Level-300 Demonstration

Script

Version 1.0

9/3/2015

Prepared for: Darmadi Komo

Prepared by: Prowess Consulting

|  |  |
| --- | --- |
| **Form Factor** | Demo script (Microsoft Word) |
| **Target Audience** | TDMs and database administrators |
| **Goals/Objectives** | Increase confidence among Microsoft SQL Server administrators that Azure SQL Database is easy to configure and can still meet the needs of their production databases. |

Overview

This demonstration explores auditing in Microsoft Azure SQL Database in the context of a software-as-a-service (SaaS) provider, Wingtip Tickets, which provides ticketing software to artists and groups. This demonstration centers on the tenant Julie and the Plantes (a fictitious pop-music tenant).

Other Tenants that will be discussed in future labs, will include the following:

* The Archie Boyle Band (a fictitious rock-music tenant)
* Walla Walla Symphony (a fictitious classical-music tenant)

## Demo Architecture

## 

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

## Dependencies

This demonstration requires running the deployment and configuration PowerShell scripts from the Level-200 demonstration in order to work. The demo also requires the Azure Search code, which comes in a compressed archive file distinct from this script or the PowerShell scripts.

|  |  |  |
| --- | --- | --- |
|  | Task 0: Prerequisite step for Auditing - Connecting Excel template to Azure SQL DB |  |
|  | 1. To explore the auditing scenario, please complete the following steps.   Make sure your Microsoft Excel template 01-Azure SQL DB Audit Logs Report Template is connected to your storage account.   1. Open **01-Azure SQL DB Audit Logs Report Template**. 2. Click the **PowerQuery** tab. |  |
|  | 1. Make sure **Show Pane** is enabled. 2. On the right, under Workbook Queries, right-click on **AuditLogs**. 3. Click **Edit**. |  |
|  | 1. Click the gear next to Source. 2. Insert your Azure Table Storage account that was created during setup. (In this example *mbjulieandtheplantes*) |  |
|  | 1. In case you’re prompted to provide the Azure Storage Key, browse to **portal.azure.com**, click **Browse** > **Storage Account** and select the Storage Account that was previously created. |  |
|  | 1. Click the key and then copy the **Primary Access Key**. |  |
|  | 1. Click **OK**. 2. At the top left, click **Close & Load**.   **Note:** If you’ve previously used the demo, use the AuditLogUtility to purge the *Azure Storage Account* audit log table, which will improve load time during the auditing section.  AuditLogUtility.exe <your storage account name> <your storage account primary key> |  |

|  |  |  |
| --- | --- | --- |
|  | **Section 1: Auditing** |  |
|  | Auditing can be used to keep track of infrastructure changes made to the environment, in addition to database-related issues such as failed logon attempts. In this demonstration we will:   * Enable and set up auditing for a specific database in the Azure portal. * Simulate a task to highlight the benefits of auditing:   + Delete one of the concert events.   + See how this effectively deletes all events associated with that concert. * Use the auditing dashboard in an Excel template to drill down into the auditing events and examine who deleted the event and at what time.   In addition to being able to provide excellent scale-up and scale-out capabilities, Azure SQL Database also helps database administrators gain insights into infrastructure-related issues as and when they happen through events that can be used to drill down further to determine root cause.  In this demonstration we’re going to delete an event to simulate an “oops” scenario—for example, where a developer or database administrator (DBA) accidentally delete some records without realizing what they are deleting. | |
|  | 1. Switch back to the Tenant website. 2. Click the **My Events** tab. 3. Notice the one (or more) event(s) that you previously purchased. 4. Click **Sign Off**. | While setting up the demo, I purchased tickets for two separate events. Let’s now delete one of those events. |
|  | 1. Click **Sign In**. | First, we’ll sign out of the website as a customer … |
|  | 1. Sign in using the administrator credentials:    * **admin@admin.com**    * **P@ssword1** | … and then we’ll sign in as an administrator. |
|  | * 1. Click the **Admin** tab. | From the Admin menu … |
|  | 1. In the **City**, **Venue**, **Event**, and **Artist** drop-down menus, select the values that match those for the event to which you bought tickets in Task 1. 2. Verify that this event is the same event for which you purchased tickets in the Task 1. 3. Click **Delete Event**. | … we’ll delete the Julie and the Plantes event. |
|  | 1. At this point you should receive a notification that the event has been deleted. Make a note of the event ID, in this example #1. Click **OK**. | We get confirmation that we successfully deleted the event—Concert ID 1. We’ll need to remember that for later. |
|  | 1. Click **Sign Off**. 2. Click **Sign In**. | And now we sign out again. |
|  | 1. Sign in using the account you previously created. | So, now we’ll go back into the website as a customer. |
|  | 1. Click the **My Events** tab. | Sure enough, the Julie and the Plantes event that we previously purchased is now missing. |
|  | 1. Switch back to the Azure portal. 2. Browse to the Customer1 SQL Database. 3. Click the **Customer1 Database** on the Primary Database *S*erver (in this example, *mbjulieandtheplantesprimary.database.windows.net*). 4. In the **Summary** pane, scroll down to the Operations tile. | In the real world, the DBAs would want to figure out what went wrong as quickly as possible when customer calls start coming in. |
|  | 1. Click **Auditing**. 2. Click **Configure** at the top of the Auditing pane. | This is where database auditing comes in handy.  Auditing can be enabled by specifying which events to capture and which Azure Storage Table to store those.  Auditing can be enabled by specifying which events to capture and to which Azure Storage Table to store them.  Because it takes some time for events to start being collected, I enabled this ahead of time for the demo.  In the pane with the Auditing events graph, notice the Open in Excel button.  [**Note:** For the purposes of the lab, please use the Excel template that was included with this HOL, as it should already be connected to the Azure Table in Task 0, Steps 4-14.] |
|  | 1. **Open** the 01-Azure SQL DB Audit Logs Report Template.   Make sure the Workbook Queries pane is displaying on the right. If not, from the **Power Query** menu, click **Show Pane**.   1. Right-click **AuditLogs**. 2. Click **Refresh**. | In the pane with the Auditing Events donut, notice the Open in Excel button. An Excel Power Query spreadsheet is provided in Azure to connect to the audit log that is being saved to the Azure Table Storage.  The Excel template should be connecting to the Azure Storage account. (In this example: *mbjulieandtheplantes*)  The query should show how many rows have loaded once the refresh is completed. |
|  | 1. Make sure the **Drilldown** worksheet is selected. 2. To the right of **DataChanges**, double-click the count (in this example, **4**).   A new worksheet will load, populated with all rows related to Data Changes.   1. Notice the delete event that corresponds to the event that was deleted from the Admin page. In this example, this is ConcertId 1. | Once the refresh of the spreadsheet is complete, we will drill down into Data Changes.  This loads a new worksheet, which is populated with all rows related to Data Changes. Notice the delete event for Concert ID 1. |
|  | 1. Notice also the time that this took place.   Audit analysis shows that a concert event was deleted. In the next task, you will use point-in-time restore to restore the missing concert event. | And notice the time the deletion took place: precisely when we deleted the event. Now we need to restore the event. |
|  | **Section 2: Conclusion** |  |
|  | Auditing in Azure SQL Database can help you keep track of changes made to your environment. It can also help you gain insights into infrastructure-related issues and drill down further to determine root cause. But what about resolving issures you discover? To see the Azure tool designed for resolving localized, one-off erros, ask your Microsoft sales representative about seeing the Azure SQL Database point-in-time restore demonstration. | |