ARM Cache Rehydration

Last updated by | Peter Hewitt | Oct 7, 2022 at 7:11 AM PDT

Contents

How to re-hydrate ARM Cache.

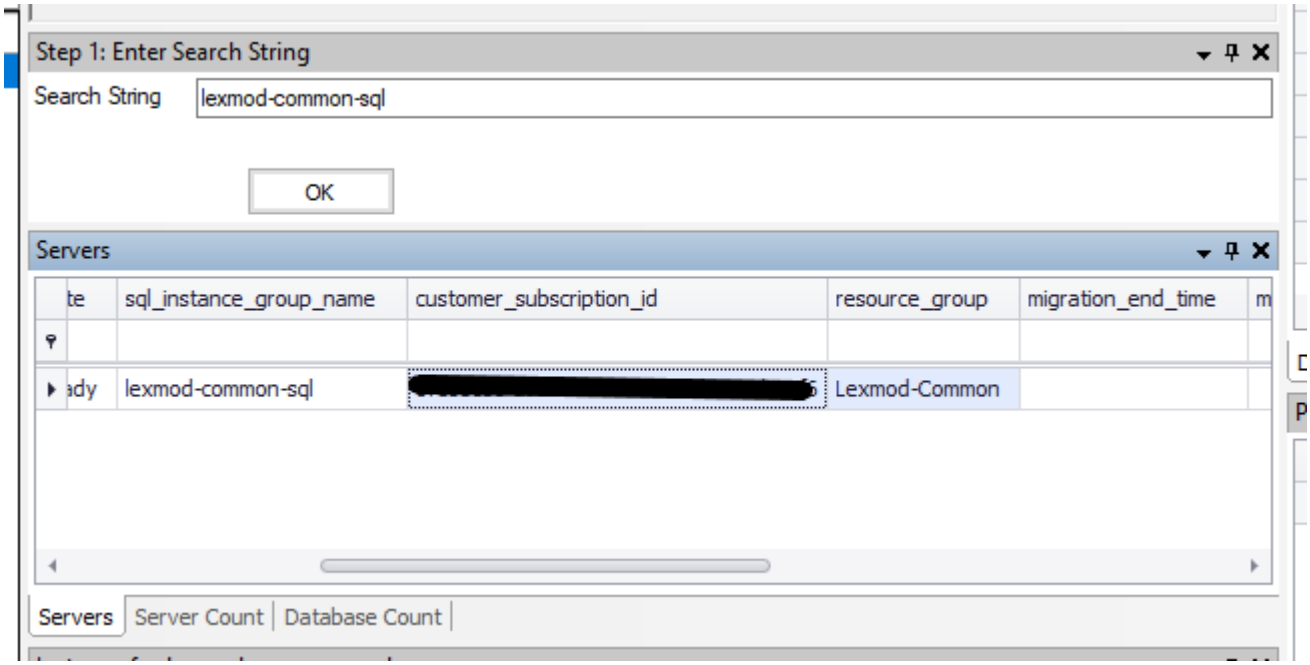
Analysis -	User is unable to see their resources (server, database, or elastic pool in the Ibiza portal (portal.azure.com). Or they see resource have wrong information presented in Portal, I.e a stretch database is shown as a SQL DB.
Azure Support Center -	https://msit.microsoftstream.com/video/1a20e639-27a9-46bb-89f5-2702f0bcfe0b?channelId=500dee5d-c51a-4738-b3d2-34f3ce9668b1
Other Tools -	<Kusto / XTS / Powershell / CLI Reference>
Permissions	<Review you have “Azure SQL CSS Std CAS” with ReadWrite permission level

Mitigation steps -

Keywords : ARM Cache, ARM Rehydration, ACIS, CSM, Sparta, CSM Rehydration, Resource Hydration

First check to see what exactly is the issue.

1. Check to make sure that the resources show up in XTS using "Sterling Servers and Databases.xts" view. Make note of **resource group** and **subscription id** in case the customer did not already tell you this info. Resource names are case sensitive in the old WCF API implementation, so make sure to preserve the correct casing for all resource names as presented in CMS for the next steps.



2. Open Geneva Actions portal: https://jarvis-west.dc.ad.msft.net/
   You will need to use a SAW or SAVM for this step SAVM can be requested here https://tdc.azure.net

3. Select appropriate environment ("Public" for regular production cloud) and click "Go to Geneva Actions" button.

NOTE: Make sure to select the proper environment endpoint at the top-left corner. Set "Public" for regions like West Europe or Central US; set e.g. "Fairfax or Mooncake " for the restricted data centers.

4. Confirm that the entry is missing from ARM cache (to prove that this is the correct root cause):

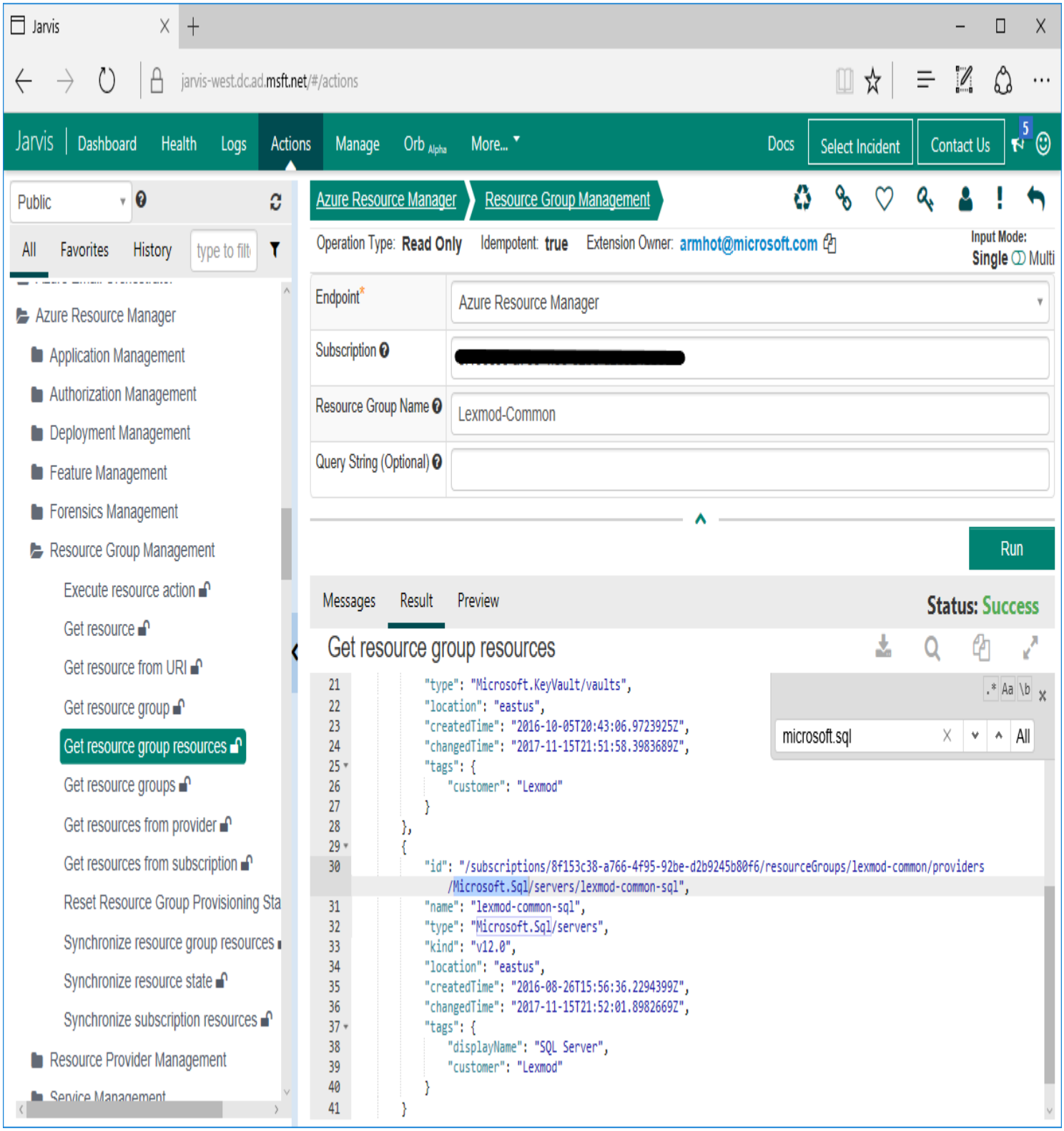
   Use action "Azure Resource Manager" -> "Resource Group Management" -> "Get resource group resources".

   Fill in the Subscription & Resource Group name and press Run.

   Search through the output (you can click the Result area and use Ctrl-F to bring up search menu) and see if the resource in question is in the result.

   If the resource is missing, continue to **Missing resources** below.

   If the resource exist, but the casing for the resource name is not the same as presented in CMS, continue to **Mismatch in resource name casing** below.



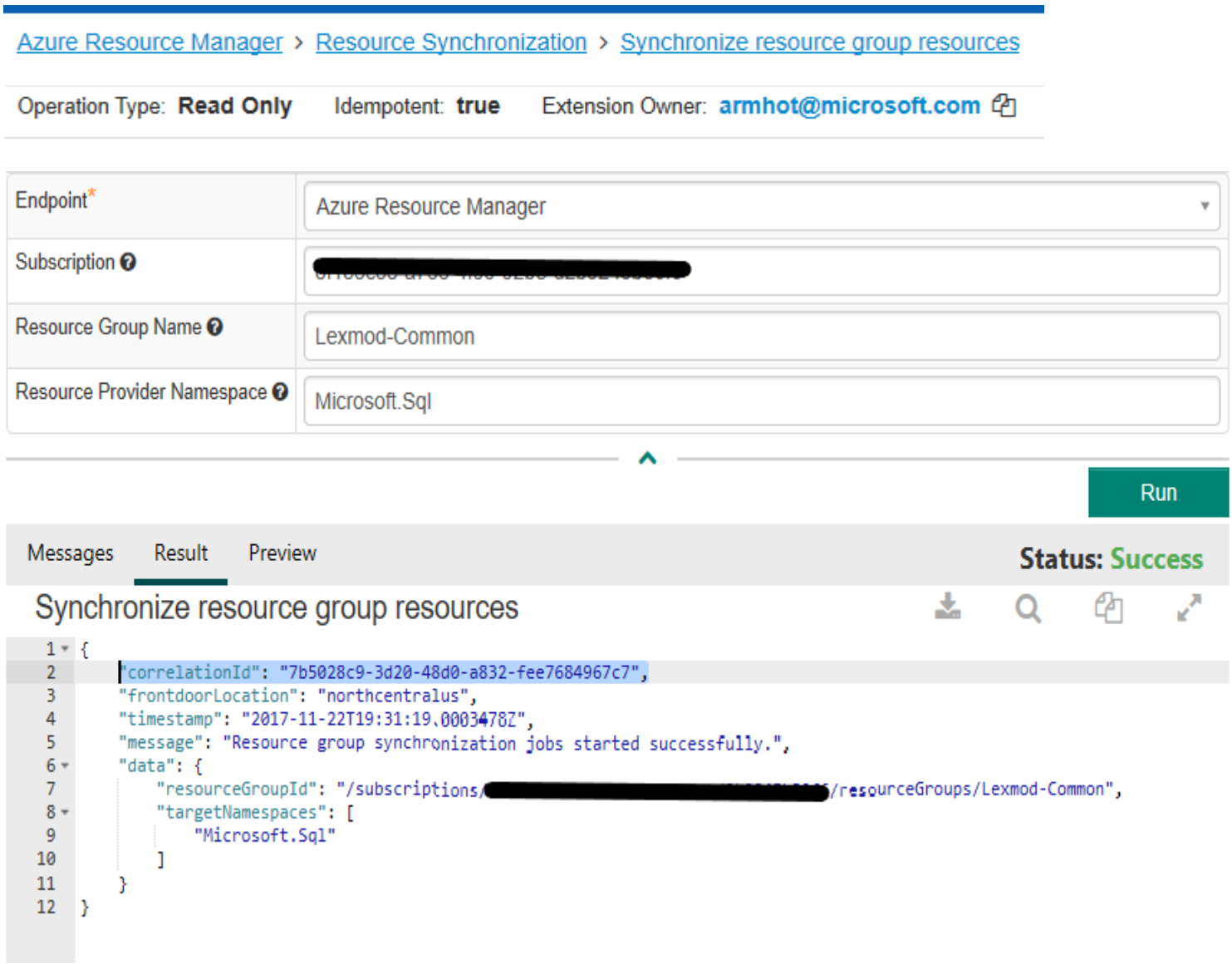
Missing resources

1. Synchronize ARM cache:

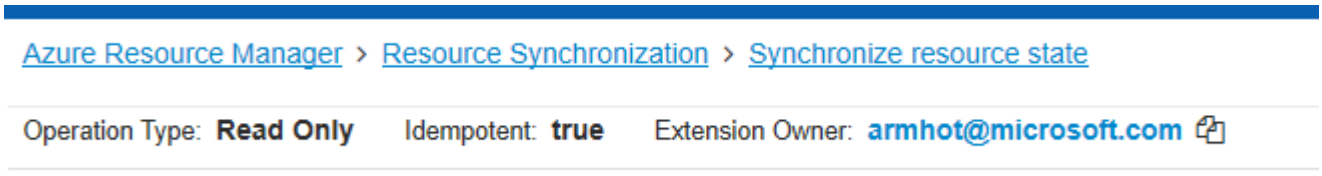
   Use action "Azure Resource Manager" -> "Resource Synchronization" -> "Synchronize resource group resources".

   Fill in the Subscription & Resource Group name, and Resource Provider Namespace "Microsoft.Sql" and press Run.

The sync will take a few minutes. Please make note of the correlationId and write it in the ICM incident - this will be useful to query ARMProd Kusto in case the sync does not work.



If some of the resources do not show up in the output, you can also try synchronize them specifically by using the "Synchronize Resource State" operation:



Use these exact input, Id is **case sensitive**:

Subscription	Subscription of resource
Resource Group Name	Exact resource group name from "Get resource group resources" above
Resource Provider Namespace	Microsoft.Sql
Id	servers/<servername>/databases/<exact db name from **CMS**>
Resource Location	Correct resource location from "Get resource group resources" above
Api Version	2017-10-01-preview
Provisioning State	Create

2. Wait a few minutes for the sync job to run. It is possible to view its progress in ARMProd kusto. **The correlationId is the same as the correlationId that Geneva showed above.**

   Sync job logs:

   Execute: [Web] [Desktop] [Web (Lens)] [Desktop (SAW)]
   https://armprod.kusto.windows.net:443/ARMProd

```
JobTraces
| where TIMESTAMP > ago(3h)
| where correlationId == "3c1048f2-c0e6-4884-9a99-4b825bc1066e"
| where jobId contains "ResourceConsistencyJob"
//| where message contains "Resource creation operation completed. Resource is provisioned in CSM."
| project PreciseTimeStamp , message
| sort by PreciseTimeStamp desc
```

   HTTP requests sent to resource providers:

   https://armprod.kusto.windows.net:443/ARMProd [Run in Kusto.Explorer] [Run in Kusto.Explorer on SAW] [Run in Kusto.WebExplorer]

```
HttpOutgoingRequests
| where correlationId == "7b5028c9-3d20-48d0-a832-fee7684967c7"
```

3. Use "Get resource group resources" as in previous step to confirm that

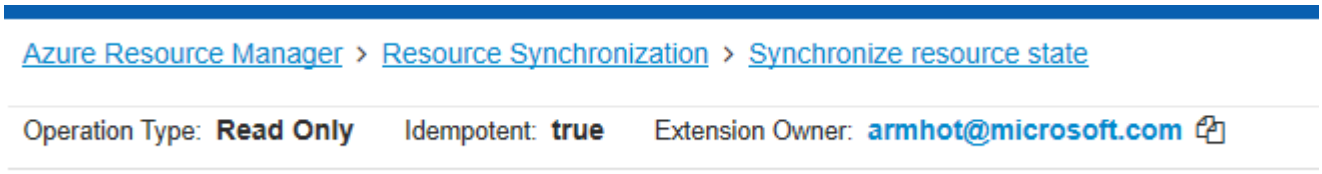
Mismatch in resource name casing

If the refresh does not work you will want to check one more scenario that could be causing the issue and that is the CaSe of the DB name not matching.

Preferred:

Drop then re-create the ARM cache such that the resource name matches those stored in CMS

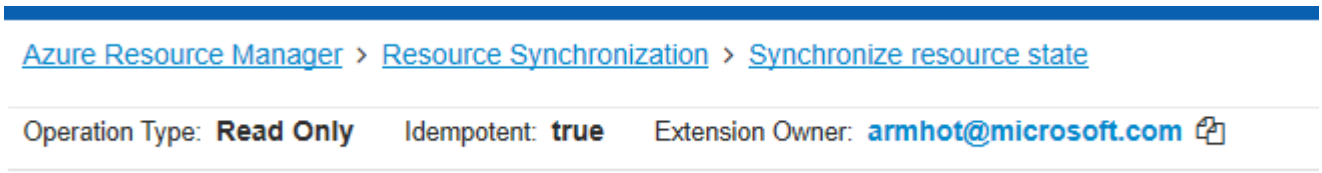
1. Remove the existing resource from ARM using Synchronize resource state. **Ensure that the Api Version is set to 2014-04-01.** The property will not be deleted if the database names differ or if a newer API version is used



Use these exact input, Id is **case sensitive**:

Subscription	Subscription of resource
Resource Group Name	Exact resource group name from "Get resource group resources" above
Resource Provider Namespace	Microsoft.Sql
Id	servers/<exact servername from **CMS**>/databases/<db name from **CMS**, change the casing of one or more characters, ie. db-> DB >
Resource Location	Correct resource location from "Get resource group resources" above
Api Version	2014-04-01
Provisioning State	Delete

1. Re-add the resource in ARM using Synchronize resource state



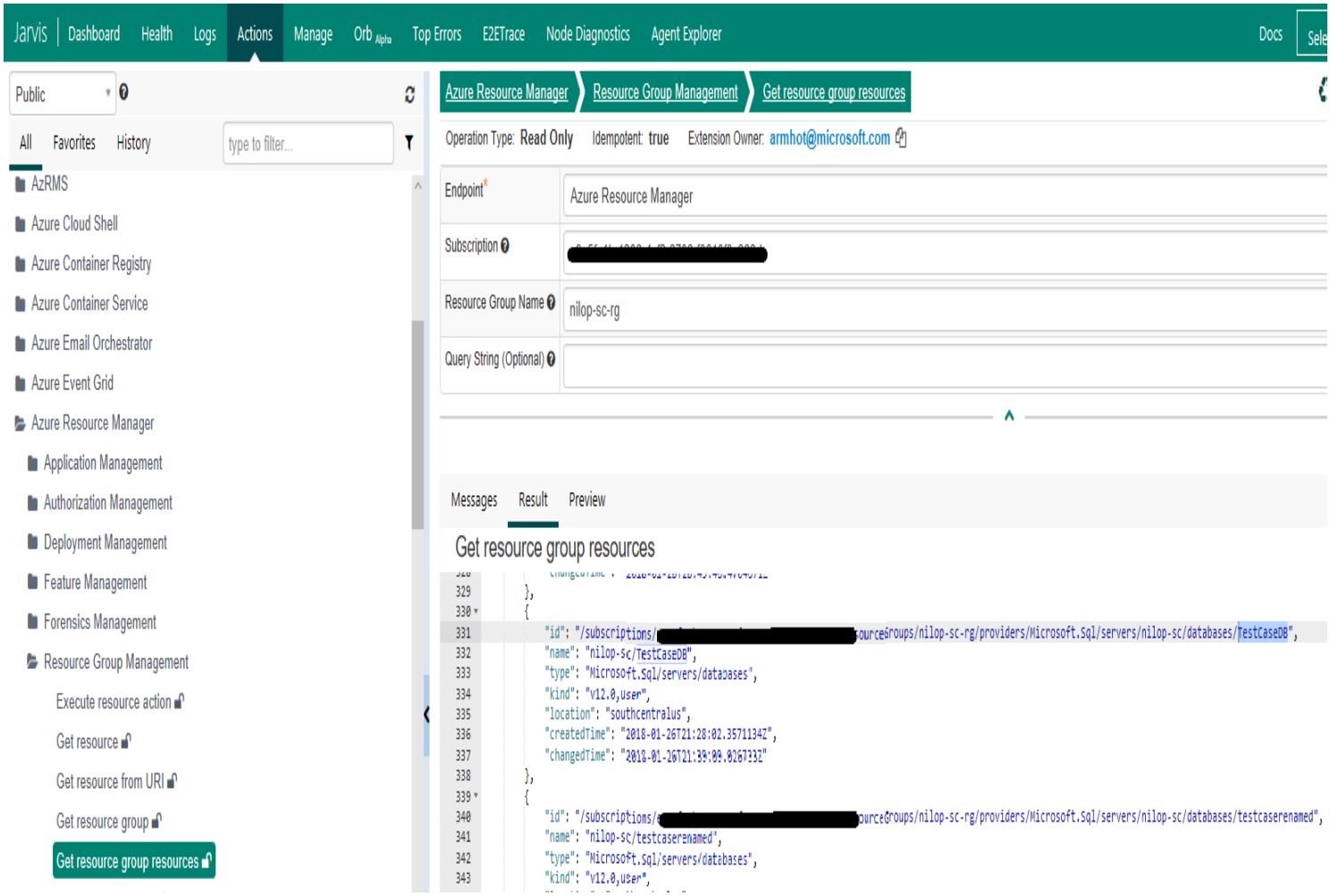
Use these exact input, Id is **case sensitive**:

Subscription	Subscription of resource	
Resource Group Name	Exact resource group name from "Get resource group resources" above	
Resource Provider Namespace	Microsoft.Sql	
Id	servers/<exact servername from **CMS**>/databases/<exact db name from **CMS**>	
Resource Location	Correct resource location from "Get resource group resources" above	
Api Version	2017-10-01-preview	
Provisioning State	Create	

Alternative:

Rename the database to match case as stored in ARM

1. Like above, check the ARM Cache and confirm the database name and particularly case of the name.



2. However checking CMS, we see that it is in fact testcasedb and not the same as TestCaseDB in the ARM Cache.

select logical_database_name, logical_server_name, State, service_level_objective, last_update_time from logical_databases where logical_server_name='nilop-sc'

	logical_database_name	logical_server_name	State	service_level_objective	last_update_time
▶	testcasedb	nilop-sc	Ready	B1M50	1/26/2018 21:59:08

This can be addressed one of two ways. Rename the database to a different name, wait for cache to reflect this and then rename back to the same name but desired case OR rename the database back to the case that matches the ARM cache. The first will have a timeframe of the database not being the same name so applications may not access it depending on their configuration. The second will be minimum impact as the rename takes seconds but does not allow the desired case to be established.

Option 1

a. Rename the existing database to a different name, even just a single character change.

--Run against master

ALTER DATABASE [testcasedb] MODIFY NAME = [testcasedb_]

b. Wait a few minutes and confirm the portal shows the database as testcasedb_

c. Once confirmed, rename the database to the original name but with the case intended.

--Run against master

ALTER DATABASE [testcasedb_] MODIFY NAME = [testcasedb]

It should take around 10 minutes to reflect the changes in the portal but can be quicker

Option 2

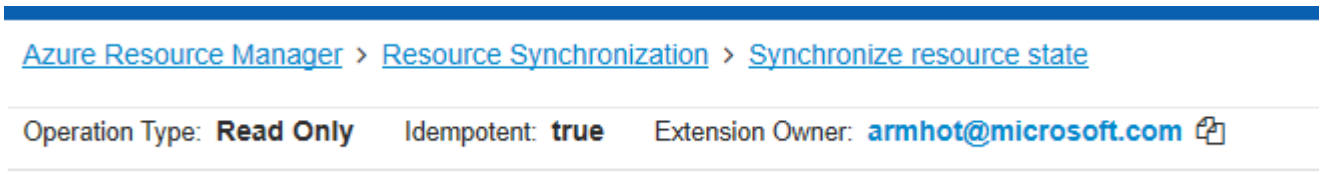
--Run against master

ALTER DATABASE [testcasedb] MODIFY NAME = [testcasedb_]

--Run against master

ALTER DATABASE [testcasedb_] MODIFY NAME = [TestCaseDB]

3. Try Synchronizing the resource directly using Synchronize resource state:



Use these exact input, Id is **case sensitive**:

Subscription	Subscription of resource
Resource Group Name	Exact resource group name from "Get resource group resources" above
Resource Provider Namespace	Microsoft.Sql
Id	servers/<servername>/databases/<exact db name from **CMS**>
Resource Location	Correct resource location from "Get resource group resources" above
Api Version	2017-10-01-preview
Provisioning State	Create

Provisioning state means:

- Create will add resource to ARM cache
- Delete will drop resource from ARM cache

Public facing RCA

All resources in Azure are cached in a caching system called the Azure Resource Manager (ARM) cache. So when you go to the portal it goes through the ARM cache to check for Azure resources. This is a performance optimization feature to ensure the portal and other read operations from customers can be handled at extremely large scale across all regions.Due to the nature of SQL Azure we allow customers to modify resources using both ARM APIs as well as T-SQL back-end APIs (such as ALTER DATABASE) for backwards compatibility with legacy applications. When you modify a resource in T-SQL, the SQL Azure resource provider has to update the ARM cache to keep it in sync. In some cases these synchronization attempts fail and the ARM cache gets out of sync and this requires we manually refresh the ARM cache. We have been working with the ARM team to improve this ARM cache synchronization over time to reduce these cases but they still can happen.

We are committed to keep pushing scale latency down and improving platform reliability and experience. We do appreciate you as a customer and we understand that you've tied success of your product to success of SQL Azure platform. This is not something we take lightly. Please continue scaling your database to manage cost and meet traffic demands.

Public Doc Reference :

Internal MSFT Reference:

221755943

Classification

Root cause path -

Azure Portal/ Service issue/UX malfunction

How good have you found this content?