Create or Manage Resources

Last updated by | Charlene Wang | Jan 3, 2023 at 12:30 AM PST

Contents

```
Created On: Dec 4, 2022
Authored by: luyang1
Reviewed By: zhizhwan
```

Create or Managed Resources

Customers may create and manage SQL Database servers, single databases, elastic pool using the Azure portal, PowerShell, Azure CLI, REST API, and Transact-SQL as per their business needs.

Create a single database - Azure SQL Database

Create and manage elastic pool - Azure SQL Database

Check operation info from telemetry

If customer submit the operation request via Azure portal/PowerShell/Azure CLI/REST API, then the request will go to ARM layer first, then ARM pass the request to Resource Provider layer. Please refer to ARM overview.

Check the operation info on **ARM layer**

Refer to ARM info

```
// Check ARM layer log/error
let server = '{serverName}';
let db = '{dbName}';
let subID = '{SubscriptionId}';
let crlID = '{correlationId}'  //if the correlationId of the operation is unknown then remove this filter and
let startTime = ago(1d);
let endTime = now();
union withsource=SourceTable kind=outer HttpIncomingRequests,HttpOutgoingRequests,EventServiceEntries
| where TIMESTAMP >= startTime and TIMESTAMP <= endTime
| where subscriptionId =~ subID
| where properties contains server //and properties contains db
| where correlationId =~ crlID  //if the correlationId of the operation is unknown then remove this filter and
| where isempty(server) or resourceUri endswith strcat('/',server) or resourceUri has strcat('/',server,'/')
| extend statusMessage = parse_json(tostring(parse_json(properties).statusMessage))
| extend statusMessageStatus = statusMessage.status
| extend error = statusMessage.error
| extend errorMessage = statusMessage.error.message
| extend errorMessageCode = tostring(statusMessage.error.details[0].code)
| extend errorMessageDetails = tostring(statusMessage.error.details[0].message)
| where errorMessageDetails != 'The operation timed out and automatically rolled back. Please retry the operat
| parse resourceUri with '/subscriptions/' subId  '/resourcegroups/' RGname 'providers/Microsoft.Sql/' Resourc
| parse resourceUri with '/subscriptions/' subId2  '/resourceGroups/' RGname2 'providers/Microsoft.Sql/' Resou
| extend Message = iif(errorMessageDetails!='', errorMessageDetails, errorMessage)
| extend Resource = iif(isnotempty(Resource), Resource, Resource2)
| project SourceTable, PreciseTimeStamp, resourceProvider, operationName, Resource
, status, subStatus, errorCode, errorMessage, Message, statusMessageStatus, errorMessageCode
, eventTimestamp, correlationId, operationId, resourceUri, targetResourceProvider, targetUri
, properties, authorization, claims, Deployment, principalOid, principalPuid

// authorization column contains: scope, action, role info, etc
// claims column contains: appid, ipaddr, user name(operation committer), user email account info(sometimes te


// get ARM layer correlationID from the above query, and use it in below queries
Traces
| where TIMESTAMP > {startTime}
| where TIMESTAMP < {endTime}
| where subscriptionId =~ "{subscriptionId}"
| where correlationId =~ "{correlationId}"
| project PreciseTimeStamp, TaskName, correlationId, operationName, message, exception, SourceNamespace, addit
, ActivityId, subscriptionId, tenantId

Errors
| where TIMESTAMP > {startTime}
| where TIMESTAMP < {endTime}
| where correlationId =~ "{correlationId}"
| project PreciseTimeStamp, correlationId, operationName, message, exception, additionalProperties, tenantId,

ProviderTraces
| where TIMESTAMP > {startTime}
| where TIMESTAMP < {endTime}
| where subscriptionId =~ "{subscriptionId}"
| where correlationId =~ "{correlationId}"
| project PreciseTimeStamp, correlationId, operationName, message, exception, ActivityId, principalOid, provid
```

Subscription level operations from ARM

Can't find an operation customer claims they have done?
This query checks all (top 100) the create/update/delete operations done from ARM perspective for the subscription.

- **Tracking ID** may match **serviceRequestId**
- **serviceRequestId** will be the same as **request_id** in MonManagement

```
//Connect to https://armprod.kusto.windows.net/ARMProd
let starttime = ago(1d);
let endtime = now();
let subId = tolower('{SubscriptionId}');
HttpOutgoingRequests
| where TIMESTAMP between (starttime .. endtime)
| where subscriptionId == subId
| where TaskName == 'HttpOutgoingRequestStart'
| where httpMethod != 'GET'
| where targetResourceProvider == 'MICROSOFT.SQL'
| where targetResourceType startswith 'SERVERS'
| take 100
| parse hostName with 'management.' Region '.control.database.windows.net'
| project TIMESTAMP, ActivityId
| join kind=leftouter (
HttpOutgoingRequests
| where TIMESTAMP between (starttime .. endtime)
| where subscriptionId == subId
| where TaskName != 'HttpOutgoingRequestStart'
| where httpMethod != 'GET'
| where targetResourceProvider == 'MICROSOFT.SQL'
| where targetResourceType startswith 'SERVERS'
| parse hostName with 'management.' Region '.control.database.windows.net'
| project TIMESTAMP, TaskName, correlationId, serviceRequestId, httpMethod
, Region, operationName, targetUri, ActivityId
) on ActivityId
| parse TaskName with 'HttpOutgoingRequestEndWith' Result
| project Start=TIMESTAMP, End=TIMESTAMP1, Result, correlationId, serviceRequestId
, httpMethod, Region, operationName, targetUri
```

Check **resource provider** layer telemetry

```
// -- run below Kusto query in corresponding region
// check firewall rule change operations for a specific server
MonManagementResourceProvider
| where http_verb == "PUT"
| where TIMESTAMP > {startTime}
| where TIMESTAMP < {endTime}
| where request_url contains '{serverName}'
| where operation_name contains "MICROSOFT.SQL"
| where operation_name contains "FIREWALLRULES" or operation_name contains "VIRTUALNETWORKRULES"
| project originalEventTimestamp, logical_server_name, request_id, controller_name, action_name, http_verb, me
exception_http_code, request_url, code_package_version, operation_type, api_version,
response_code, body_size, headers, azure_async_operation_header, client_request_id, exception_type, error_desc
extra_parameters, client_routing_id, caller_address

// use the request id got from the above query
MonManagement
| where request_id =~ "{requestId}"
| project originalEventTimestamp, request_id, event, state, old_state, new_state, action, request_name, operat
, message, fsm_error_message, error_message, is_user_error, last_exception, exception_message
```

Check database and server creation/drop status on XTS:

XTS(view):

```
sterling servers and databases.xts (sterling)
```

XTS(CMS query):

```
-- Check management operations related to the server
select * from management_operations
where logical_server_name = '{serverName}' and subscription_id = '{subscriptionId}'
order by start_time

-- Check database state (pay attention to state, parent state and tombstone_reason,etc)
select * from all_logical_databases
where logical_server_name = '{serverName}' and logical_database_name = '{databaseName}'

select * from logical_databases
where logical_server_name = '{serverName}' and logical_database_name = '{databaseName}'

select * from logical_servers where name = '{serverName}'

select * from upsert_logical_server_requests
where requested_logical_server_name = '{serverName}'

-- Check elastic pool info
select * from all_elastic_pools where logical_server_name = '{serverName}'
```

How good have you found this content?