

Troubleshoot replica management operation

Last updated by | Hamza Aqel | Mar 27, 2023 at 6:14 AM PDT

You can use the below Kusto query to check and debug any customer request related to replica management operation, these operations are:

- CreateReadReplicaManagementOperation
- DropReadReplicaManagementOperation
- PromoteReadReplicaManagementOperation

You need first to get:

- Start and end time of the customer operation (optional you can remove the time condition, but the query will be slower).
- Request ID, you can get using different ways, a quick one using orcasbreadth\orcasbreadth crud.xts

x testreplicaserveryu Find Clear								
	end_time	minutes	outcome	error_classification	operation_type	server_name	other_name	se
06:13:35	20/03/2023 06:23:35	10	Failed	User	CreateReadReplicaManagementOperation	testreplicaserveryu		P
06:40:57	20/03/2023 06:51:06	10.2	Failed	User	CreateReadReplicaManagementOperation	testreplicaserveryu		P

and enter these information in the below query, which is applicable for local and geo replica requests:

```

let AllRegionTable = (tableName:string)
{
cluster('sqlazureeus12').database('sqlazure1').table(tableName) // East US 1
| union cluster('sqlazureeus22').database('sqlazure1').table(tableName) // East US 2 and EUAP
| union cluster('sqlazurewus2').database('sqlazure1').table(tableName) // West US 2
| union cluster('sqlazureweu2').database('sqlazure1').table(tableName) // West Europe
| union cluster('sqlazureneu2').database('sqlazure1').table(tableName) // North Europe
| union cluster('sqlazureseas2').database('sqlazure1').table(tableName) // Southeast Asia
| union cluster('sqlazureuk2').database('sqlazure1').table(tableName) // UK
| union cluster('sqlazureau2').database('sqlazure1').table(tableName) // Australia
| union cluster('sqlazureca2').database('sqlazure1').table(tableName) // Canada
| union cluster('sqlazurencus3').database('sqlazure1').table(tableName) // North Central US
| union cluster('sqlazrwcus').database('sqlazure1').table(tableName) // West Central US
| union cluster('sqlazureja2').database('sqlazure1').table(tableName) // Japan
| union cluster('sqlazurekor').database('sqlazure1').table(tableName) // Korea
| union cluster('sqlazurecus2').database('sqlazure1').table(tableName) // Central US
| union cluster('sqlazureeas2').database('sqlazure1').table(tableName) // East Asia
| union cluster('sqlazureince2').database('sqlazure1').table(tableName) // Central India
| union cluster('sqlazurebr2').database('sqlazure1').table(tableName) // Brazil
| union cluster('sqlazureeas2').database('sqlazure1').table(tableName) // east asia
| union cluster('sqlazurefra').database('sqlazure1').table(tableName) // France
| union cluster('sqlazuregermany').database('sqlazure1').table(tableName) // Germany
| union cluster('sqlazureince2').database('sqlazure1').table(tableName) // India center
| union cluster('sqlazureinso2').database('sqlazure1').table(tableName) // India south
| union cluster('sqlazureinwe2').database('sqlazure1').table(tableName) // India west
| union cluster('sqlazurescus2').database('sqlazure1').table(tableName) // SouthCentralUS
| union cluster('sqlazureseas2').database('sqlazure1').table(tableName) // SouthEast Asia
| union cluster('sqlazuresouthafrica').database('sqlazure1').table(tableName) // South Africa
| union cluster('sqlazureswitzerland').database('sqlazure1').table(tableName) // Switzerland
| union cluster('sqlazureuae').database('sqlazure1').table(tableName) // UAE
| union cluster('sqlazurewus1').database('sqlazure1').table(tableName) // West US1
| union cluster('https://sqlazureusw3.westus.kusto.windows.net').database('sqlazure1').table(tableName) // Wes
| union cluster('sqlazurenorway').database('sqlazure1').table(tableName) // norway
//
};
let requestid = "D7F00404-C71D-492A-932E-C4B1816D4E6C"; // Request id of replica operation
let ['_startTime']=datetime('2023-03-19T02:03:34Z'); // Start time of the request
let ['_endTime']=datetime('2023-03-21T23:03:34Z'); // End time of the request, if it in progress use let ['_e
let replica_request_completed = toscalar(AllRegionTable('MonOrcasBreadthRp')
| where TIMESTAMP between (_startTime .. _endTime)
| where request_id =~ requestid
| where AppTypeName == "OrcasBreadthRp"
| where operation_type in ("CreateReadReplicaManagementOperation", "PromoteReadReplicaManagementOperat
| where event in ("management_operation_failure", "management_operation_success")
| extend replica_drop_operation_id = extract("\\<ReadReplicaDropOperationId\\>(.*?)\\</ReadReplicaDrop
| extend replica_promote_operation_id = extract("\\<ReadReplicaPromoteOperationId\\>(.*?)\\</ReadRepli
| extend replica_create_operation_id = extract("\\<ReadReplicaCreateOperationId\\>(.*?)\\</ReadReplica
| extend replica_request_id = case (
    isnotempty(replica_create_operation_id), replica_create_operation_id,
    isnotempty(replica_drop_operation_id), replica_drop_operation_id,
    isnotempty(replica_promote_operation_id), replica_promote_operation_id,
    requestid)
| extend replica_request_id = iff (replica_request_id =~ "00000000-0000-0000-0000-000000000000", reque
| project replica_request_id);
let replica_request_inprogress = toscalar(AllRegionTable('MonOrcasBreadthRp')
| where isempty(replica_request_completed)
| where TIMESTAMP between (_startTime .. _endTime)
| where request_id =~ requestid
| where AppTypeName == "OrcasBreadthRp"
| where state in ("PerformPreChecks")
| where event == "orcasbreadth_fsm_information"
| extend replica_request_id = extract("FSM RequestId: (.*?) FSM Id: (.*)", 2, message)
| project toupper(trim("", replica_request_id)));
let replica_source_request = toscalar(AllRegionTable('MonOrcasBreadthRp')
| where isnotempty(replica_request_inprogress) or isnotempty(replica_request_completed)
| where TIMESTAMP between (_startTime .. _endTime)
| where request_id =~ requestid
| where AppTypeName == "OrcasBreadthRp"

```

4/5/23, 4:46 PM

Troubleshoot replica management operation - Overview

```
| where state in ("NotifyCreateSourceServerLink", "NotifyDropSourceServerLinkCommunication")
| where event == "orcasbreadth_fsm_information"
| extend source_request_id = extract("The source server link create FSM Id (.+?) , Region:(.+?)", 1, m
| extend source_request_id = iff (isempty(trim(" ", source_request_id)), extract("The source server li
| project toupper(trim(" ", source_request_id)));
let replica_request_id = case (
    isempty(replica_request_completed), replica_request_completed,
    isempty(replica_request_inprogress), replica_request_inprogress,
    requestid);
let replica_source_request_id = iff(isempty(replica_source_request) or replica_request_id =~ requestid, reques
let requestids = set_union(pack_array(toupper(requestid),toupper(replica_request_id), toupper(replica_source_r
AllRegionTable('MonOrcasBreadthRp'))
| where TIMESTAMP between (_startTime .. _endTime)
| where request_id in (requestids)
| where AppTypeName == "OrcasBreadthRp"
| project originalEventTimestamp,PreciseTimeStamp, request_id, state_machine_type, ['state'], old_state, new_s
| order by originalEventTimestamp asc
```

◀

Check the results, especially for message, stack_trace and error_message columns.

▶

ate	old_state	new_state	message
WaitingForOpera...	WaitingForOpera...	WaitingForOpera...	
WaitingForOpera...	WaitingForOpera...	WaitingForOpera...	
VerifyTcpConnecti...			The TCP connection between your replica server and primary server failed. If replica and primary server is in different virtual networks make sure both virtual networks peered. If you create Network Security Groups...
			The TCP connection between your replica server and primary server failed. If replica and primary server is in different virtual networks make sure both virtual networks peered. If you create Network Security Groups...
	VerifyTcpConnecti...	DropReplicaServer	
	VerifyTcpConnecti...	DropReplicaServer	
	VerifyTcpConnecti...	DropReplicaServer	
	VerifyTcpConnecti...	DropReplicaServer	
ropReplicaServer			
ropReplicaServer			

for example, in the above error message:

Failure error message,

The TCP connection between your replica server and primary server failed. If replica and primary server is in different virtual networks make sure both virtual networks peered. If you create Network Security Groups (NSG) to deny traffic flow to or from your Flexible Server within the subnet where it's deployed, please make sure to allow both inbound and outbound traffic to destination port 5432. Refer to <https://docs.microsoft.com/en-us/azure/postgresql/flexible-server/concepts-networking#virtual-network-concepts> for more details.