DMV database_service_objective slow or timing out

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:28 AM PST

Contents

Issue

The customer is running a query on DMV sys.database_service_objective to maintain an overview on the existing databases, their edition and service objective, and their membership in Elastic Pools. The server is hosting more than 2000 databases grouped into several Elastic Pools, and some stand-alone databases.

When they run a join query like the following, the execution takes about 15 minutes and is roughly proportional to the number of databases at the server:

```
-- in master
SELECT
    [server_name] = @@serverName,
    [database_name] = db.name,
    db.create_date,
    dso.edition,
    dso.service_objective,
    dso.elastic_pool_name
FROM sys.databases db
INNER JOIN sys.database_service_objectives dso on db.database_id = dso.database_id
ORDER BY db.name
```

Investigation / Analysis

When running individual queries on `sys.databases` and `sys.database_service_objectives` without the join, both return in less than a second. This indicates that the issue is related to the execution plan of the join between both DMVs.

Querying `sys.database_service_objectives` is a very expensive operation in itself, as it reaches out to the Azure control plane to retrieve the current service objective of a database. The execution plan showed a Nested Loop join between both DMVs. The join was fetching the list of databases from `sys.databases` first, then looping over `sys.database_service_objectives' for each database to retrieve the details. This involved a roundtrip from the` master` database to the control plane for each of the databases that is hosted on the server.

Mitigation

There are two options to mitigate this issue:

Mitigation 1: Force a MERGE JOIN or HASH JOIN to avoid the LOOP JOIN

A `MERGE JOIN` or `HASH JOIN` will read the complete `sys.database_service_objectives` content in one step before joining it to `sys.databases` . It is still an expensive operation, but at least causing only one roundtrip to the control ring. This can be achieved with a join hint, like this:

```
-- in master
SELECT
    [server_name] = @@serverName,
    [database_name] = db.name,
    db.create_date,
    dso.edition,
    dso.service_objective,
    dso.elastic_pool_name
FROM sys.databases db
INNER MERGE JOIN sys.database_service_objectives dso on db.database_id = dso.database_id
-- INNER HASH JOIN sys.database_service_objectives dso on db.database_id = dso.database_id
ORDER BY db.name
```

Mitigation 2: Use PowerShell instead of Transact-SQL

The Get-AzSqlServerServiceObjective PowerShell cmdlet returns the available service objectives for the databases hosten on an Azure SQL Database server. See Example 1 for a sample command and its results.

More Information

Other mitigation options are not readily available. You could try using the workaround of querying the DATABASEPROPERTYEX function. But it doesn't work on Azure SQL Database, because it will return NULL for all databases that are not the current database. You would have to run this inside each database individually and collect the results:

```
-- does not return the required results on Azure SQL Database
SELECT
        db_name(),
        db.name,
        DATABASEPROPERTYEX(db.name, 'ServiceObjective'),
        DATABASEPROPERTYEX(db.name, 'Edition')
FROM sys.databases db
```

Public Doc References

sys.database_service_objective

Get-AzSqlServerServiceObjective

How good have you found this content?

🙂 🙁