

Fulltext Investigation Tips

Last updated by | Radhika Shah | May 30, 2022 at 4:55 PM PDT

Contents

- [Related Kusto Queries](#)
 - [SQL Server error log: WASD2ProdMonSQLSystemHealth](#)
 - [Fulltext log: MonFullTextInfo](#)
 - [Resource governance: MonRgManager](#)
 - [Resource usage: MonRgLoad](#)
 - [New Telemetry added: MonFulltextActiveCatalogs, MonFull...](#)
- [PG can help investigate and add additional log on backend...](#)
- [DMV queries which can collect from customer's side](#)

Related Kusto Queries

SQL Server error log: WASD2ProdMonSQLSystemHealth

Example query:

```
MonSQLSystemHealth
| where TIMESTAMP >= datetime(2016-11-01 10:30:00Z)
| where TIMESTAMP <= datetime(2016-11-23 18:50:00Z)
| where LogicalServerName == "fjmsql"
| where message contains "FDHost"
| project AppName, PreciseTimeStamp, TIMESTAMP, LogicalServerName, message
```

Fulltext log: MonFullTextInfo

Example query:

```
MonFullTextInfo
| summarize ErrorCount = countif(error_number > 0) by ClusterName, LogicalServerName, database_name
| where ErrorCount > 50
| top 50 by ErrorCount desc
```

If fdhost has exception and dump, you will not see new entry in MonFullTextInfo until fdhost is killed by kill-process. Also, there will not be new fdhost related in error log. You can check the error using

```
MonDmOsExceptionStats
| where LogicalServerName == "informat" and exception_id == 30046 | where AppName == "a4a474450972" | where TI
| order by TIMESTAMP desc
```

Resource governance: MonRgManager

Example query:

```
ProdMonRgManager
| where originalEventTimestamp > ago(1h)
| summarize failedInstanceCount = countif(event == "failed_instance" and code_package_name == 'FullTextSearchSe
| where failedInstanceCount > 50
| top 100 by failedInstanceCount desc
```

This looks for full text that failed frequently in last hour.

Resource usage: MonRgLoad

Example query:

1. CPU

```
MonRgLoad | where code_package_name == "FullTextSearchService"
and event == "instance_load" and application_name == "fabric:/Worker.ISO.Premium/f978ec7fa418"
| project TIMESTAMP, cpu_percent = (cpu_load * 1.0 / cpu_load_cap ) * 100
| render timechart
```

2. Memory

```
MonRgLoad | where code_package_name == "FullTextSearchService"
and event == "instance_load" and application_name == "fabric:/Worker.ISO.Premium/f978ec7fa418"
| project TIMESTAMP, mem_percent = (memory_load * 1.0 / memory_load_cap ) * 100
| render timechart
```

3. IOPS

```
MonRgLoad | where code_package_name == "FullTextSearchService"
and event == "instance_load" and application_name == "fabric:/Worker.ISO.Premium/f978ec7fa418"
| project TIMESTAMP, , iops_percent = (iops_load * 1.0 / iops_load_cap) * 100
| render timechart
```

4. All in one

```
MonRgLoad | where code_package_name == "FullTextSearchService"
and event == "instance_load" and application_name == "fabric:/Worker.ISO.Premium/f978ec7fa418"
| order by TIMESTAMP asc
```

Note if the date went back too far and Kusto doesn't have it, you can consider Cosmos (which keeps 30 days of data).

New Telemetry added: MonFulltextActiveCatalogs, MonFulltextIndexFragment

Example query:

```

MonFulltextActiveCatalogs
| where TIMESTAMP > datetime(2020-06-11 02:40:27)
| where TIMESTAMP < datetime(2020-06-11 03:40:27)
| where LogicalServerName == "ams-brso-1-sqs-stream-1-1"
| where has_crawl_completed == 0
| where catalog_id == 5
| where is_enabled == 1
| project TIMESTAMP, AppName, database_id, catalog_id, active_fts_index_count, auto_population_count, manual_p
has_crawl_completed, crawl_start_date, crawl_end_date, docs_processed, fail_count, item_count, pending_changes

```

MonFulltextIndexFragment

PG can help investigate and add additional log on backend node (after they JIT into the node)

1. Look for SQLFT0000x0000? Files under log directory. That's full text's work log. Connect to the server using name pipe, do

```
Dbcc traceon(7601, 7603, 7604, 7605, 7605, 7606, 7607, 7609, 7684, 7689, -1)
```

Those traceflags can add more details in fulltext log and errorlog

2. You can also turn on verbose tracing

```

exec sp_fulltext_service 'fdhost_tracing', 1
exec sp_fulltext_service 'fdhost_tracing_tag', '5Exceptions:5FTEAdmin:5FTEFiles:5FDProtocolHandlerInternal:5FDP

```

Information about these tags are in retailtags.hxx and this generates xml files under dump directory

Don't forget to turn the dbcc traces off when you are done collecting data. To turn off tracing do


```
exec sp_fulltext_service 'fdhost_tracing', 0
```

To remove the tracing tags do

```
exec sp_fulltext_service 'fdhost_tracing_tag', ''
```

3. Tools to use: isqlw, procexp, procmon, windbg
4. Kill fdhost command Get-FabricNode -NodeName DB.64 -NodeClusterName [tr4.westcentralus1-a.worker.database.windows.net](https://a.worker.database.windows.net) | Kill-Process -ProcessName fdhost.exe -ApplicationNameUri fabric:/Worker.ISO.Premium/ef526cc33421
5. Use this internal table to see status of rows (retry, failures):


sys.fulltext_index_docidstatus_<objid>

Good reads: <https://thesqldude.com/2012/07/06/how-to-identify-the-row-or-document-that-failed-to-index-during-a-fulltext-index-population/> 

DMV queries which can collect from customer's side

1. Get fulltext fragments that don't have clustered index:

```
SELECT *
FROM sys.fulltext_index_fragments
WHERE fragment_object_id NOT IN
(
SELECT frag.fragment_object_id
FROM sys.fulltext_index_fragments frag
JOIN sys.indexes ind
ON frag.fragment_object_id = ind.object_id
AND ind.type = 1
)
```

2. Check number of items already indexed or if fulltext is doing any actual work. Replacing [cat.name](#)  with actual catalog name.

```
SELECT
DATEADD(ss, FULLTEXTCATALOGPROPERTY(cat.name, 'PopulateCompletionAge'), '1/1/1990') AS LastPopulated,
FULLTEXTCATALOGPROPERTY( cat.name, 'ItemCount') AS [ItemCount],
(CASE FULLTEXTCATALOGPROPERTY(cat.name, 'PopulateStatus')
WHEN 0 THEN 'Idle'
WHEN 1 THEN 'Full Population In Progress'
WHEN 2 THEN 'Paused'
WHEN 3 THEN 'Throttled'
WHEN 4 THEN 'Recovering'
WHEN 5 THEN 'Shutdown'
WHEN 6 THEN 'Incremental Population In Progress'
WHEN 7 THEN 'Building Index'
WHEN 8 THEN 'Disk Full. Paused'
WHEN 9 THEN 'Change Tracking' END) AS PopulateStatus,
CASE FULLTEXTCATALOGPROPERTY( cat.name, 'MergeStatus') WHEN 0 THEN N'master merge not in progress' ELSE N'maste
CASE FULLTEXTCATALOGPROPERTY( cat.name, 'ImportStatus') WHEN 0 THEN N'full-text catalog not being imported' ELS
FROM sys.fulltext_catalogs AS cat
```

If crawl is going on, the itemcount number should be increasing.

3. Figure out the catalog and index name

```
Select * from sys.fulltext_catalogs
Select * from sys.fulltext_indexes
```

4. Figure out if there is a fulltext crawl or population. If the crawl is going on, you should see outstanding batch

```
Select * from sys.dm_fts_outstanding_batches
Select * from sys.dm_fts_index_population
```

More information can refer to [public documentation regarding fulltext dmvs](#) 

5. About whether the table have rows failed to index:

```
select objectproperty(<object id>, 'TableFulltextFailCount')
```

Note You can fetch the object id of fulltext index from sys.fulltext_indexes.

How good have you found this content?

