Q29

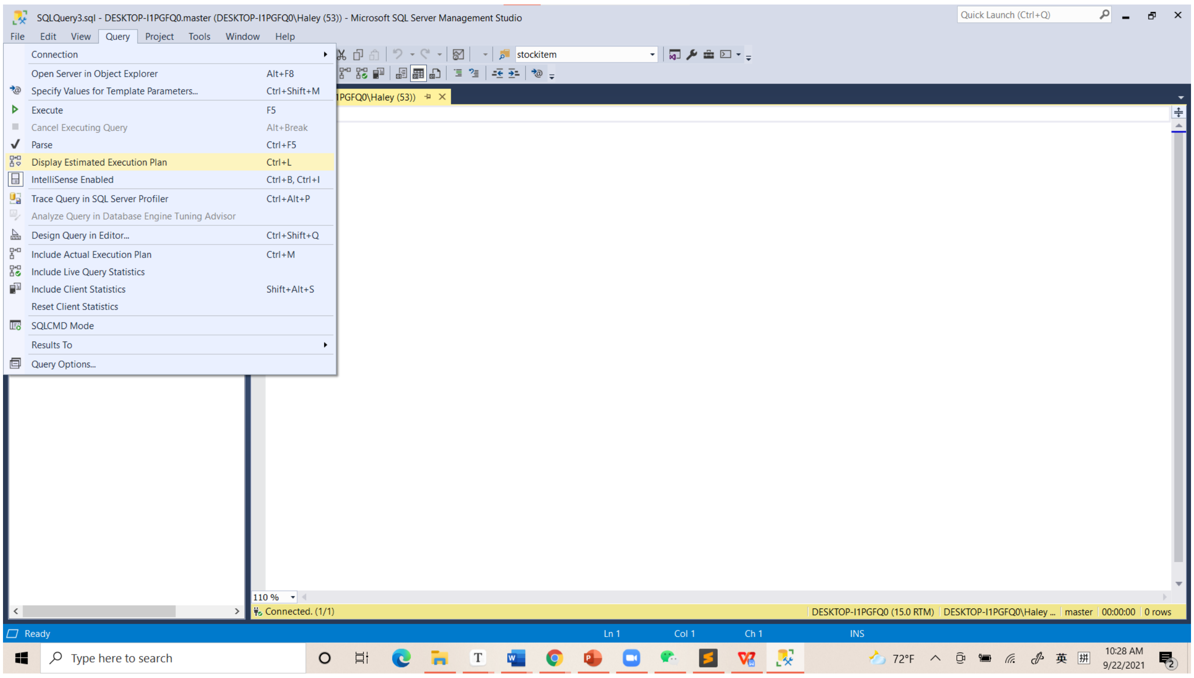
In a nutshell, SQL performance tuning consists of making queries of a relation database run as fast as possible.

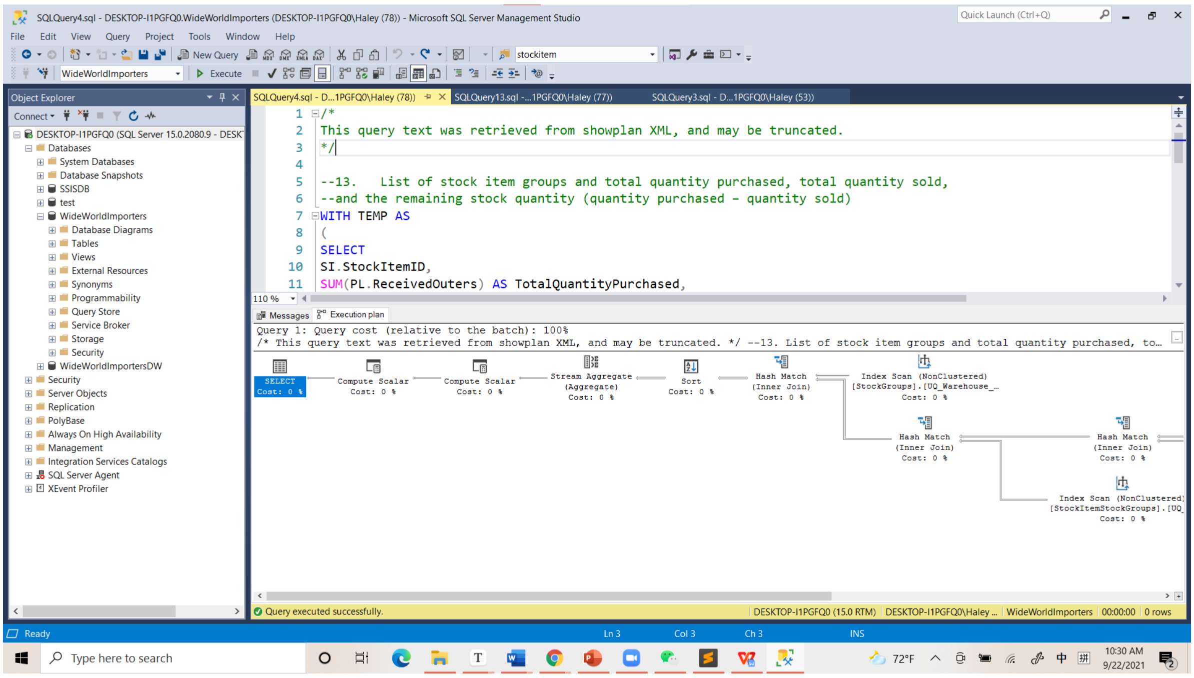
Execution Plan

In order to diagnose slow queries, it's crucial to be able to generate graphical execution plans, which you can do by using SQL Server Management Studio. Actual execution plans are generated after the queries run.

Begin by clicking on "Database Engine Query", on the SQL Server Management Studio toolbar.

You do that by clicking on the "Execute" toolbar button. Then, SQL Server Management Studio will display the execution plan in the results pane, under the "Execution Pane" tab.


SQLQuery3.sql - DESKTOP-I1PGFQ0.master (DESKTOP-I1PGFQ0\Haley (53)) - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Tools Window Help
Connection
Open Server in Object Explorer Alt+F8
Specify Values for Template Parameters... Ctrl+Shift+M
Execute F5
Cancel Executing Query Alt+Break
Parse Ctrl+F5
Display Estimated Execution Plan Ctrl+L
IntelliSense Enabled Ctrl+B, Ctrl+I
Trace Query in SQL Server Profiler Ctrl+Alt+P
Analyze Query in Database Engine Tuning Advisor
Design Query in Editor... Ctrl+Shift+Q
Include Actual Execution Plan Ctrl+M
Include Live Query Statistics
Include Client Statistics Shift+Alt+S
Reset Client Statistics
SQLCMD Mode
Results To
Query Options...
stockitem
1PGFQ0\Haley (53))
110 %
Connected. (1/1)
DESKTOP-I1PGFQ0 (15.0 RTM) DESKTOP-I1PGFQ0\Haley ... master 00:00:00 0 rows
Ready Ln 1 Col 1 Ch 1 INS
Type here to search
72°F
10:28 AM
9/22/2021



```
/*
This query text was retrieved from showplan XML, and may be truncated.
*/

--13.   List of stock item groups and total quantity purchased, total quantity sold,
--and the remaining stock quantity (quantity purchased - quantity sold)
WITH TEMP AS
(
SELECT
SI.StockItemID,
SUM(PL.ReceivedOuters) AS TotalQuantityPurchased,
```

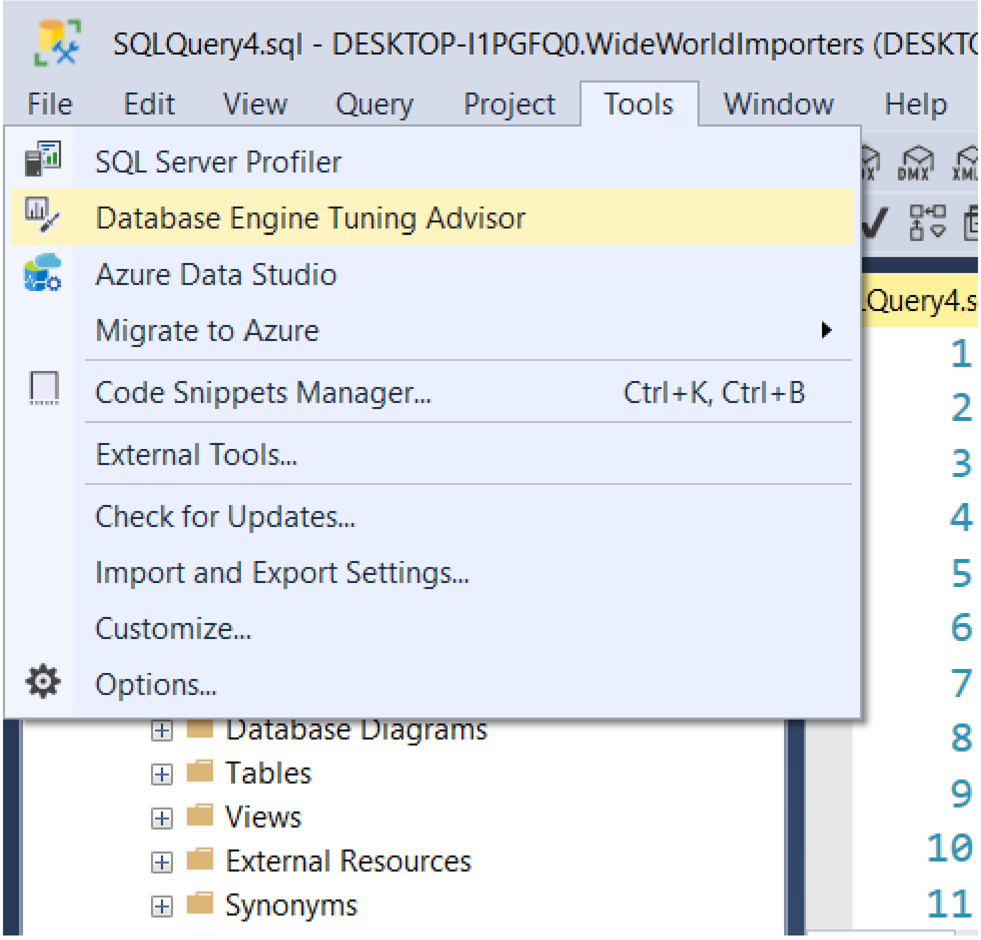
```
Query 1: Query cost (relative to the batch): 100%
/* This query text was retrieved from showplan XML, and may be truncated. */ --13. List of stock item groups and total quantity purchased, to...
```

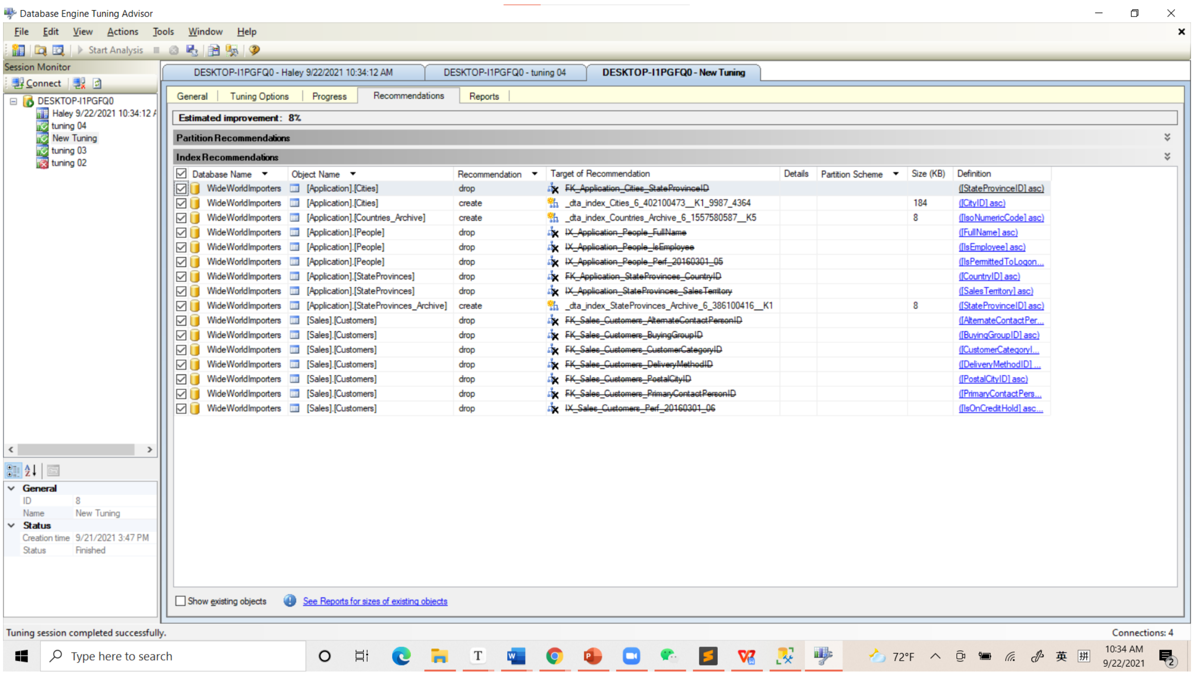
Database Engine Tuning Advisor

Another important technique for SQL performance tuning is to analyze the performance of Transact-SQL statements that are run against the database you intend to tune.

You can use the Database Engine Tuning Advisor to analyze the performance implications.

But the tool goes beyond that: it also recommends actions you should take based on its analysis. For instance, it might advise you to create or remove indexes.





General

ID	8
Name	New Tuning

Status

Creation time	9/21/2021 3:47 PM
Status	Finished

Estimated improvement: 8%

☑	Database Name	Object Name	Recommendation	Target of Recommendation	Details	Partition Scheme	Size (KB)	Definition
☑	WideWorldImporters	[Application].[Cities]	drop	~~FK_Application_Cities_StateProvinceID~~				([StateProvinceID] asc)
☑	WideWorldImporters	[Application].[Cities]	create	_dta_index_Cities_6_402100473__K1_9987_4364			184	([CityID] asc)
☑	WideWorldImporters	[Application].[Countries_Archive]	create	_dta_index_Countries_Archive_6_1557580587__K5			8	([IsoNumericCode] asc)
☑	WideWorldImporters	[Application].[People]	drop	~~IX_Application_People_FullName~~				([FullName] asc)
☑	WideWorldImporters	[Application].[People]	drop	~~IX_Application_People_IsEmployee~~				([IsEmployee] asc)
☑	WideWorldImporters	[Application].[People]	drop	~~IX_Application_People_Perf_20160301_05~~				([IsPermittedToLogon...
☑	WideWorldImporters	[Application].[StateProvinces]	drop	~~FK_Application_StateProvinces_CountryID~~				([CountryID] asc)
☑	WideWorldImporters	[Application].[StateProvinces]	drop	~~IX_Application_StateProvinces_SalesTerritory~~				([SalesTerritory] asc)
☑	WideWorldImporters	[Application].[StateProvinces_Archive]	create	_dta_index_StateProvinces_Archive_6_386100416__K1			8	([StateProvinceID] asc)
☑	WideWorldImporters	[Sales].[Customers]	drop	~~FK_Sales_Customers_AlternateContactPersonID~~				([AlternateContactPer...
☑	WideWorldImporters	[Sales].[Customers]	drop	~~FK_Sales_Customers_BuyingGroupID~~				([BuyingGroupID] asc)
☑	WideWorldImporters	[Sales].[Customers]	drop	~~FK_Sales_Customers_CustomerCategoryID~~				([CustomerCategoryI...
☑	WideWorldImporters	[Sales].[Customers]	drop	~~FK_Sales_Customers_DeliveryMethodID~~				([DeliveryMethodID] ...
☑	WideWorldImporters	[Sales].[Customers]	drop	~~FK_Sales_Customers_PostalCityID~~				([PostalCityID] asc)
☑	WideWorldImporters	[Sales].[Customers]	drop	~~FK_Sales_Customers_PrimaryContactPersonID~~				([PrimaryContactPers...
☑	WideWorldImporters	[Sales].[Customers]	drop	~~IX_Sales_Customers_Perf_20160301_06~~				([IsOnCreditHold] asc...

□ Show existing objects

Tuning session completed successfully.

Connections: 4

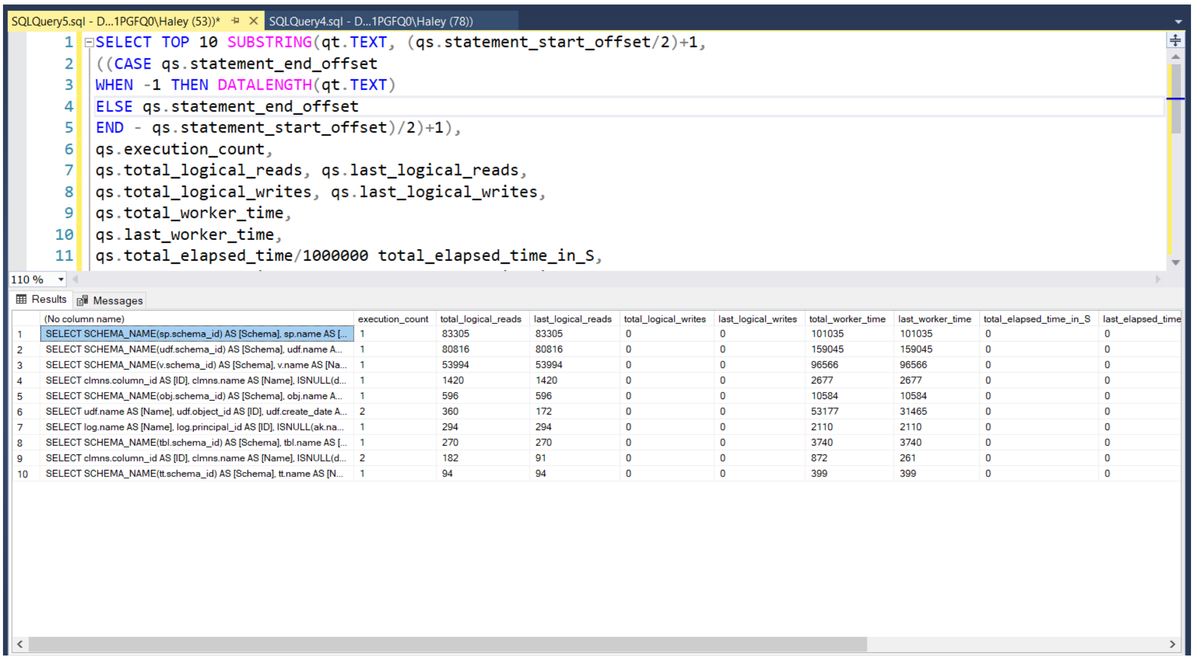
DMV

One of the great features of SQL Server is all of the dynamic management views (DMVs) that are built into it. There are dozens of them and they can provide a wealth of information about a wide range of topics.

There are several DMVs that provide data about query stats, execution plans, recent queries and much more. These can be used together to provide some amazing insights.

For example, the query below can be used to find the queries that use the most reads, writes, worker time (CPU), etc.

```
SELECT TOP 10 SUBSTRING(qt.TEXT, (qs.statement_start_offset/2)+1,
((CASE qs.statement_end_offset
WHEN -1 THEN DATALENGTH(qt.TEXT)
ELSE qs.statement_end_offset
END - qs.statement_start_offset)/2)+1),
qs.execution_count,
qs.total_logical_reads, qs.last_logical_reads,
qs.total_logical_writes, qs.last_logical_writes,
qs.total_worker_time,
qs.last_worker_time,
qs.total_elapsed_time/1000000 total_elapsed_time_in_S,
qs.last_elapsed_time/1000000 last_elapsed_time_in_S,
qs.last_execution_time,
qp.query_plan
FROM sys.dm_exec_query_stats qs
CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) qt
CROSS APPLY sys.dm_exec_query_plan(qs.plan_handle) qp
ORDER BY qs.total_logical_reads DESC -- logical reads
-- ORDER BY qs.total_logical_writes DESC -- logical writes
-- ORDER BY qs.total_worker_time DESC -- CPU time
```


SQLQuery5.sql - D...1PGFQ0\Haley (53))*
SQLQuery4.sql - D...1PGFQ0\Haley (78))
110 %
Results
Messages

```
SELECT TOP 10 SUBSTRING(qt.TEXT, (qs.statement_start_offset/2)+1,
((CASE qs.statement_end_offset
WHEN -1 THEN DATALENGTH(qt.TEXT)
ELSE qs.statement_end_offset
END - qs.statement_start_offset)/2)+1),
qs.execution_count,
qs.total_logical_reads, qs.last_logical_reads,
qs.total_logical_writes, qs.last_logical_writes,
qs.total_worker_time,
qs.last_worker_time,
qs.total_elapsed_time/1000000 total_elapsed_time_in_S,
```

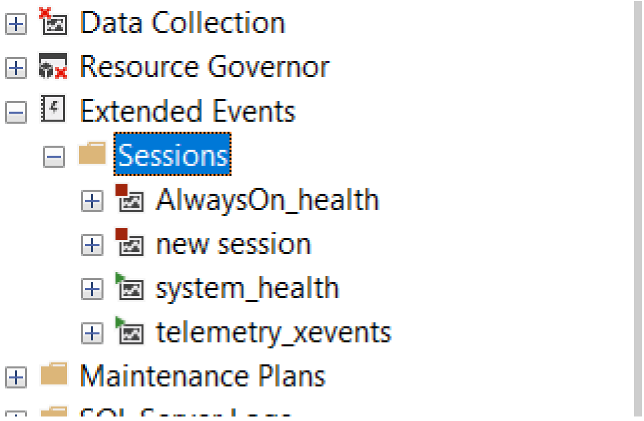
	(No column name)	execution_count	total_logical_reads	last_logical_reads	total_logical_writes	last_logical_writes	total_worker_time	last_worker_time	total_elapsed_time_in_S	last_elapsed_time
1	SELECT SCHEMA_NAME(sp.schema_id) AS [Schema], sp.name AS [...	1	83305	83305	0	0	101035	101035	0	0
2	SELECT SCHEMA_NAME(udf.schema_id) AS [Schema], udf.name A...	1	80816	80816	0	0	159045	159045	0	0
3	SELECT SCHEMA_NAME(v.schema_id) AS [Schema], v.name AS [Na...	1	53994	53994	0	0	96566	96566	0	0
4	SELECT clmns.column_id AS [ID], clmns.name AS [Name], ISNULL(d...	1	1420	1420	0	0	2677	2677	0	0
5	SELECT SCHEMA_NAME(obj.schema_id) AS [Schema], obj.name A...	1	596	596	0	0	10584	10584	0	0
6	SELECT udf.name AS [Name], udf.object_id AS [ID], udf.create_date A...	2	360	172	0	0	53177	31465	0	0
7	SELECT log.name AS [Name], log.principal_id AS [ID], ISNULL(ak.na...	1	294	294	0	0	2110	2110	0	0
8	SELECT SCHEMA_NAME(tbl.schema_id) AS [Schema], tbl.name AS [...	1	270	270	0	0	3740	3740	0	0
9	SELECT clmns.column_id AS [ID], clmns.name AS [Name], ISNULL(d...	2	182	91	0	0	872	261	0	0
10	SELECT SCHEMA_NAME(tt.schema_id) AS [Schema], tt.name AS [N...	1	94	94	0	0	399	399	0	0

Extended Events

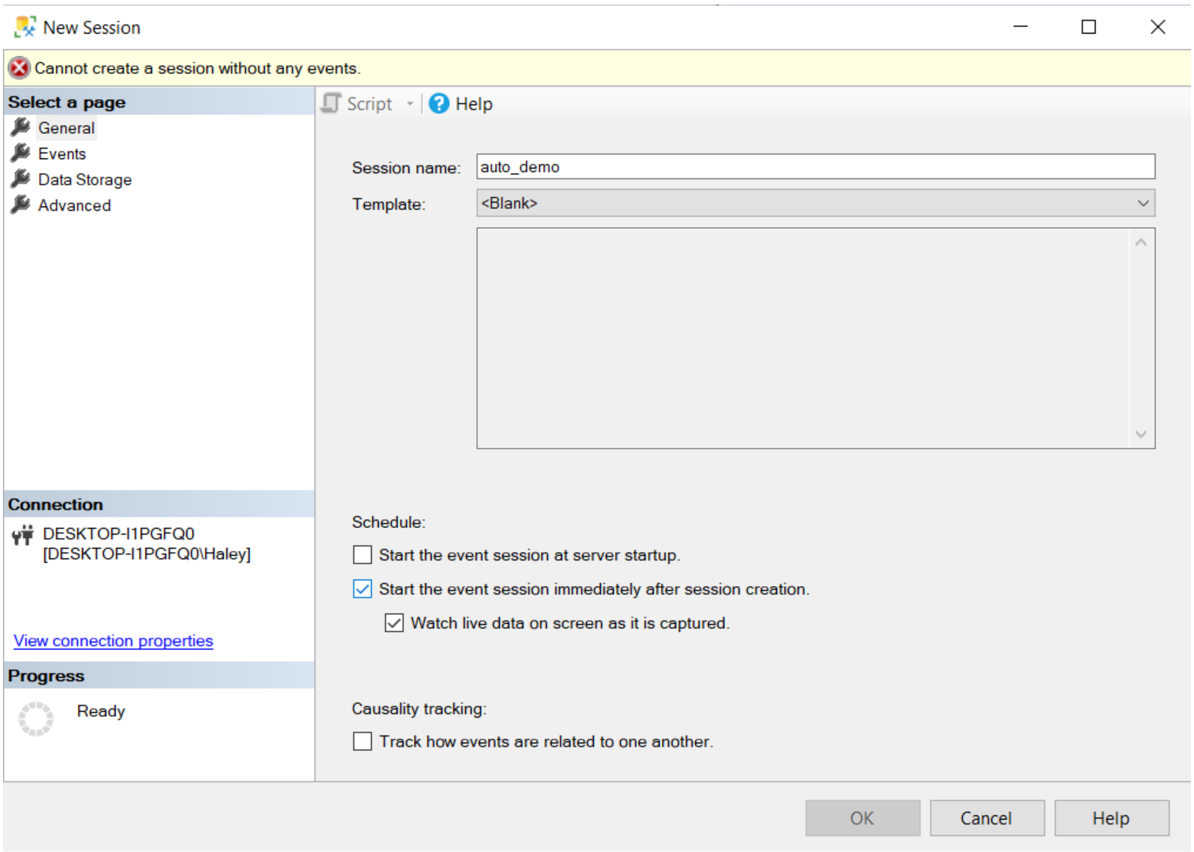
Extended Events is a light weight performance monitoring system that uses very few performance resources. Extended Events provides three graphical user interfaces (New Session Wizard, New Session and the XE Profiler) to create, modify, display, and analyze your session data.

SQL Server Extended Events (Extended Events) is a general event-handling system for server systems. The Extended Events infrastructure supports the correlation of data from SQL Server, and under certain conditions, the correlation of data from the operating system and database applications. In the latter case, Extended Events output must be directed to Event Tracing for Windows (ETW) in order to correlate the event data with operating system or application event data.

SQL Server Extended Events can be used also for SQL Server auditing purposes. For example, you can create a SQL Server Extended Events session that audits both the succeeded and failed login processes. To do that, expand the Extended Events option under the Management node, right-click on the Sessions option and choose New Session..., as below:



On the displayed New Session window, provide a meaningful name for the new session, which is Audit_Demo in our example, and set the appropriate scheduling settings from the available options, as shown below:



Cannot create a session without any events.

Session name: auto_demo

Template: <Blank>

DESKTOP-I1PGFQ0
[DESKTOP-I1PGFQ0\Haley]

Ready

Schedule:

□ Start the event session at server startup.

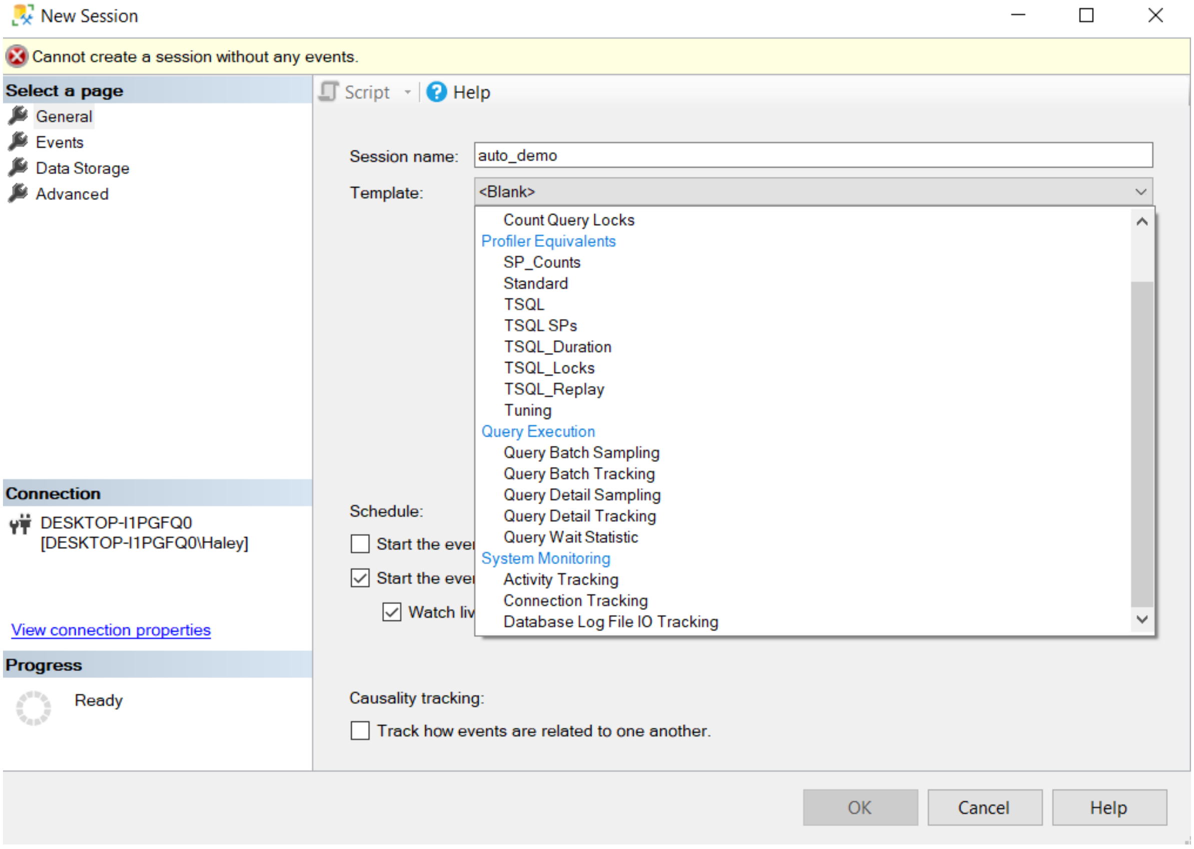
☑ Start the event session immediately after session creation.

☑ Watch live data on screen as it is captured.

Causality tracking:

□ Track how events are related to one another.

The New Extended Events Session wizards allows you to choose from the available default events templates, similar to the SQL Server Profiler templates, as shown below:



New Session

Cannot create a session without any events.

Session name: auto_demo

Template: <Blank>

Count Query Locks
Profiler Equivalents
SP_Counts
Standard
TSQL
TSQL SPs
TSQL_Duration
TSQL_Locks
TSQL_Replay
Tuning
Query Execution
Query Batch Sampling
Query Batch Tracking
Query Detail Sampling
Query Detail Tracking
Query Wait Statistic
System Monitoring
Activity Tracking
Connection Tracking
Database Log File IO Tracking

Schedule:

☐ Start the eve

☑ Start the eve

☑ Watch liv

Causality tracking:

☐ Track how events are related to one another.

Connection

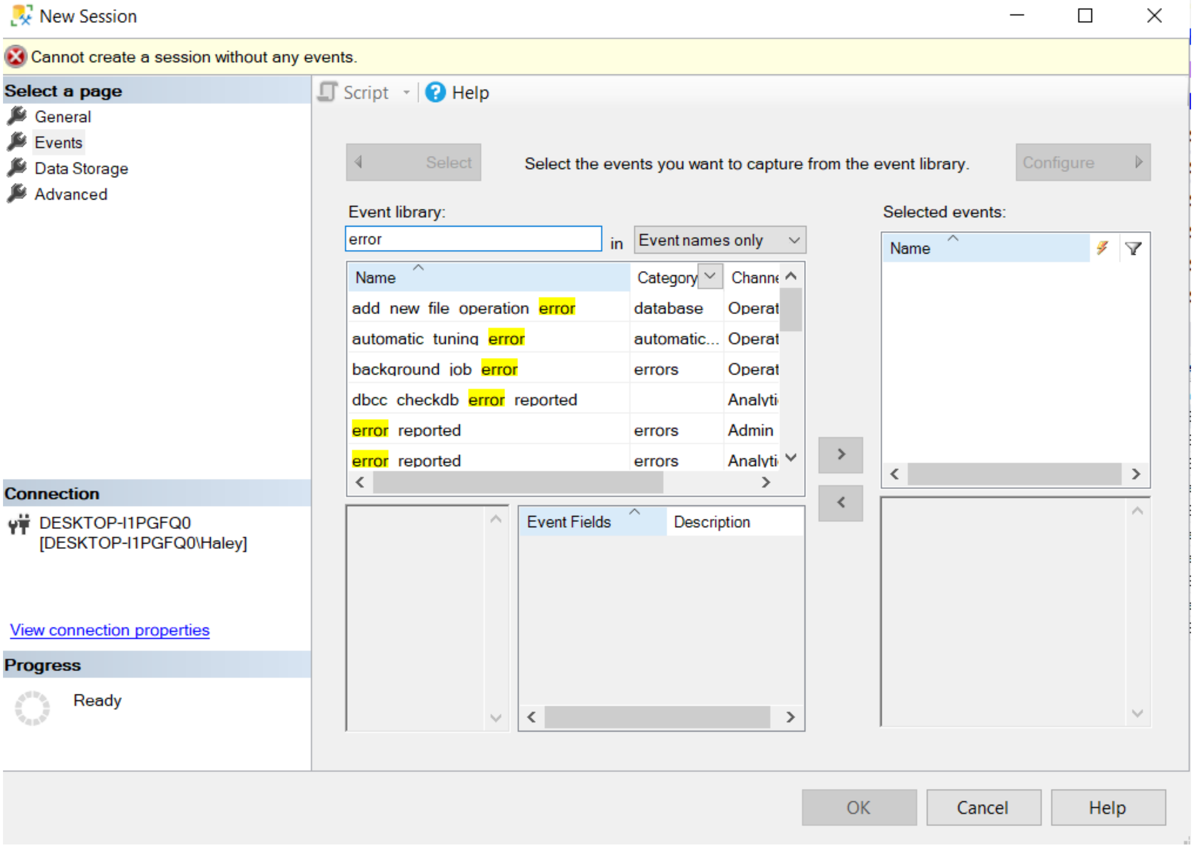
DESKTOP-I1PGFQ0
[DESKTOP-I1PGFQ0\Haley]

View connection properties

Progress

Ready

Or click on the Events tab, to customize your own session and choose the events that you are managed to monitor. In our example here, we will choose the Login event to track the successful login processes and the Error_Reported event to collect the failed logins as follows:



New Session

Cannot create a session without any events.

Select the events you want to capture from the event library.

Event library:

error in Event names only

Name	Category	Channe
add new file operation error	database	Operat
automatic tuning error	automatic...	Operat
background job error	errors	Operat
dbcc checkdb error reported		Analyti
error reported	errors	Admin
error reported	errors	Analyti

Selected events:

Connection

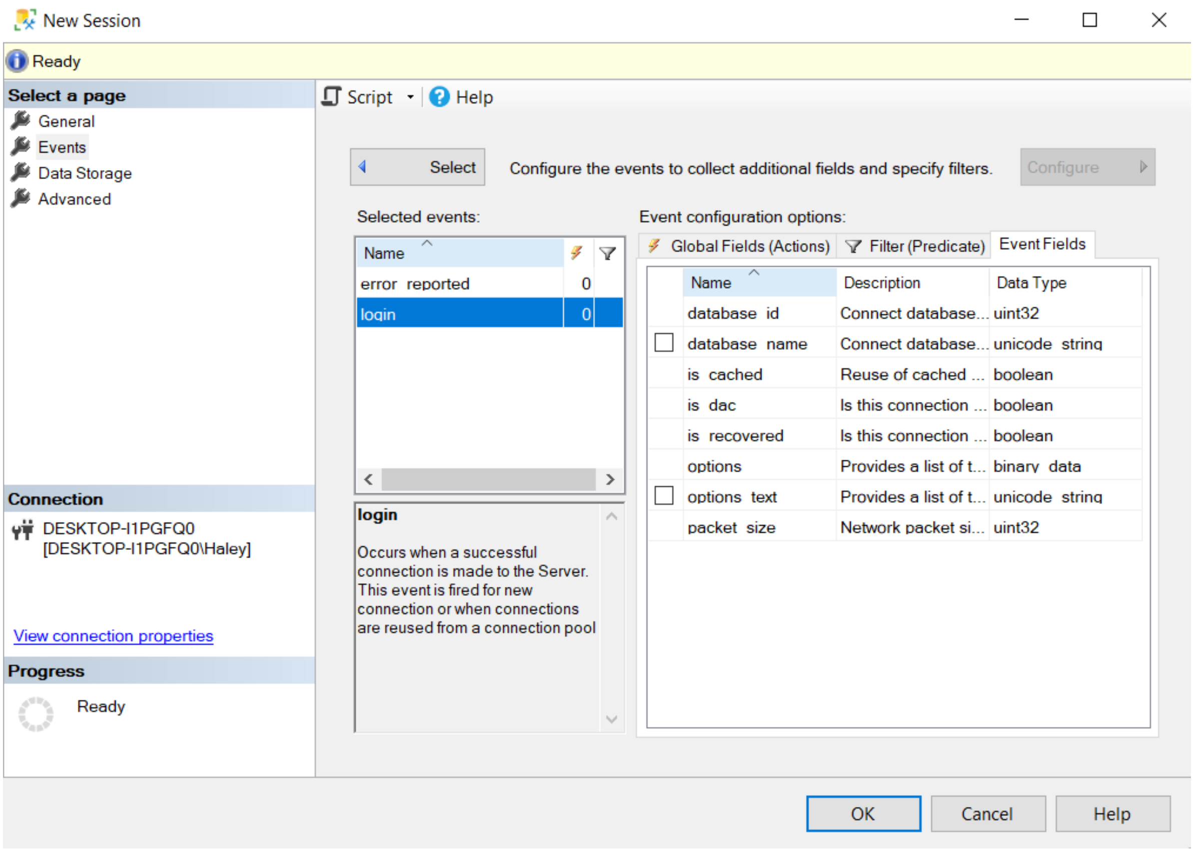
DESKTOP-I1PGFQ0
[DESKTOP-I1PGFQ0\Haley]

View connection properties

Progress

Ready

Double-clicking on the selected event will move you to a new window, on which you can customize the columns that will be recorded and received for that event. For example, we are interested in retrieving specific global information about the successful login process, as shown below:



Configure the events to collect additional fields and specify filters.

DESKTOP-I1PGFQ0
[DESKTOP-I1PGFQ0\Haley]

Selected events:

Name		
error reported	0	
login	0	

login

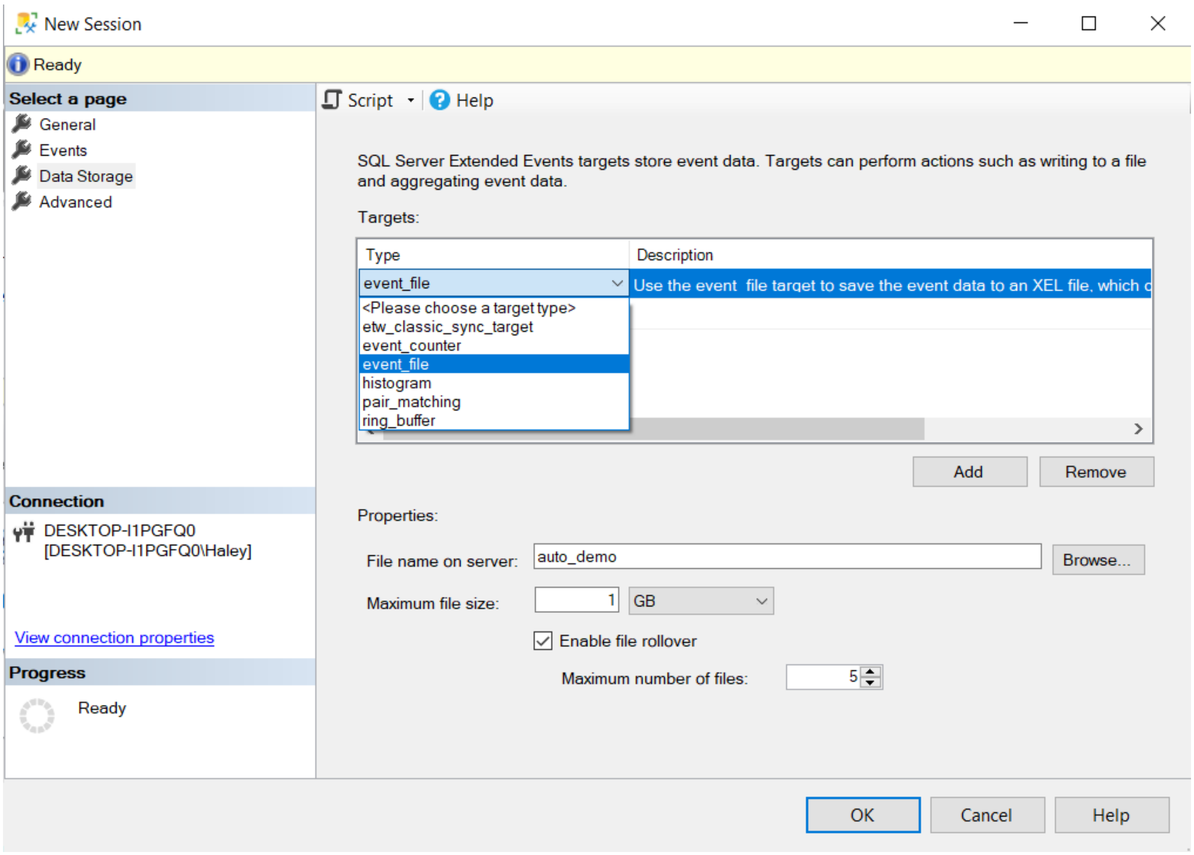
Occurs when a successful connection is made to the Server. This event is fired for new connection or when connections are reused from a connection pool

Event configuration options:

Global Fields (Actions) | Filter (Predicate) | Event Fields

	Name	Description	Data Type
	database id	Connect database...	uint32
□	database name	Connect database...	unicode string
	is cached	Reuse of cached ...	boolean
	is dac	Is this connection ...	boolean
	is recovered	Is this connection ...	boolean
	options	Provides a list of t...	binary data
□	options text	Provides a list of t...	unicode string
	packet size	Network packet si...	uint32

The location where the SQL Server auditing session result will be written can be specified from the Data Storage tab, by choosing the type of output target and configure its settings as follows:



DESKTOP-I1PGFQ0
[DESKTOP-I1PGFQ0\Haley]

SQL Server Extended Events targets store event data. Targets can perform actions such as writing to a file and aggregating event data.

Targets:

Type	Description
event_file	Use the event file target to save the event data to an XEL file, which c

<Please choose a target type>
etw_classic_sync_target
event_counter
event_file
histogram
pair_matching
ring_buffer

Properties:

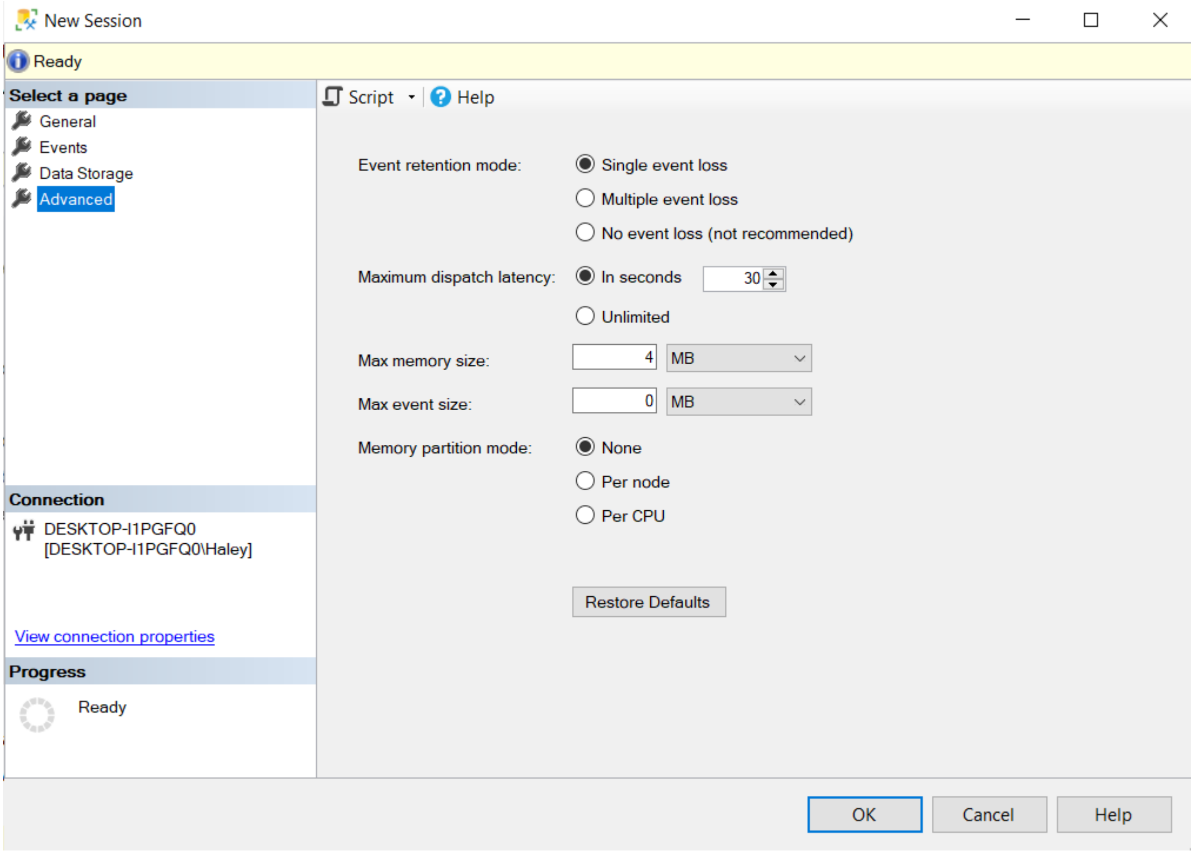
File name on server: auto_demo

Maximum file size: 1 GB

☑ Enable file rollover

Maximum number of files: 5

In the Advanced tab, you can configure the retention and resources settings for the SQL Server Extended Events session. In our example, we will keep the default values, as shown below:



DESKTOP-I1PGFQ0
[DESKTOP-I1PGFQ0\Haley]

Event retention mode:
● Single event loss
○ Multiple event loss
○ No event loss (not recommended)

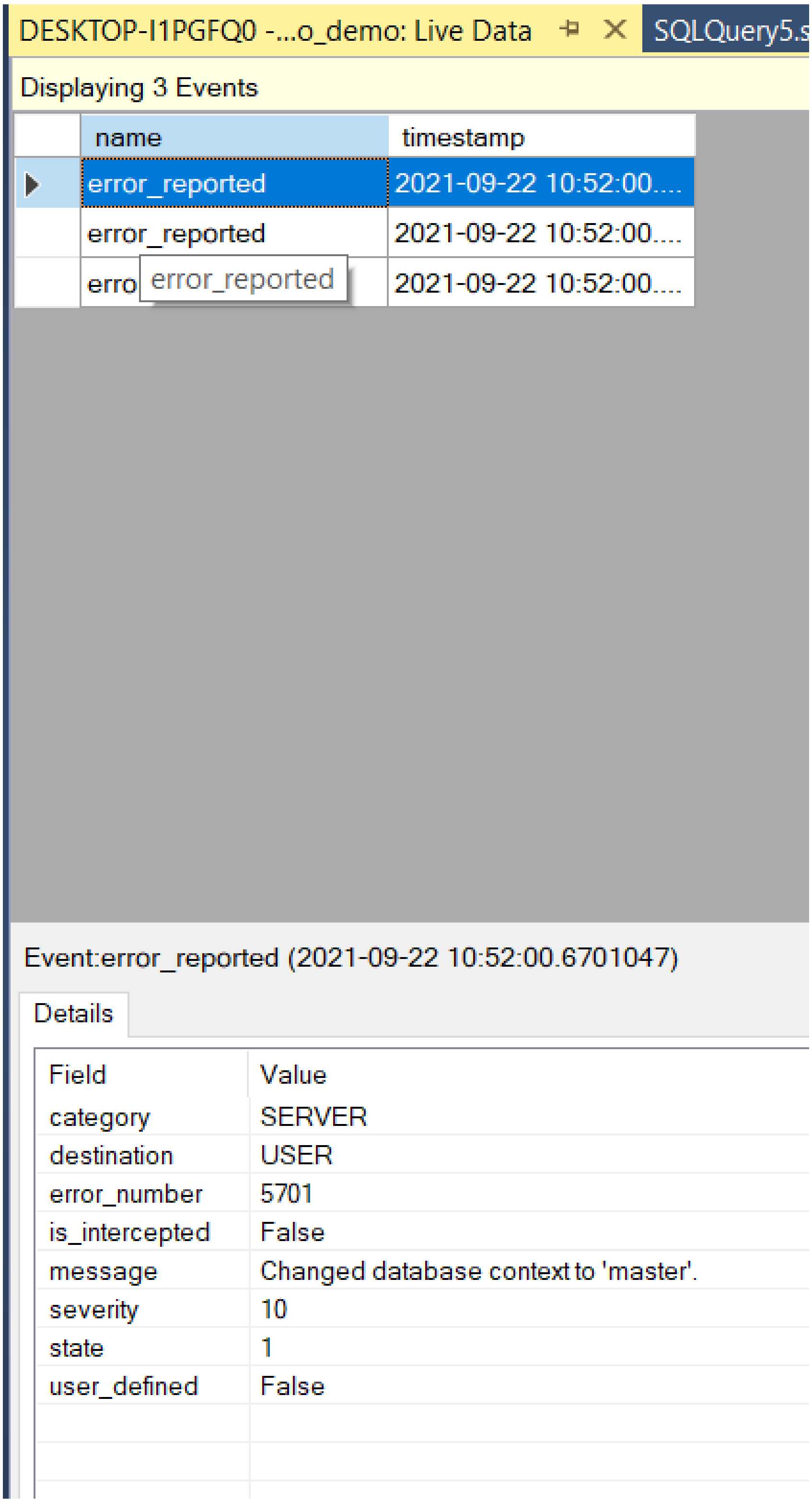
Maximum dispatch latency:
● In seconds 30
○ Unlimited

Max memory size: 4 MB

Max event size: 0 MB

Memory partition mode:
● None
○ Per node
○ Per CPU

Once the SQL Server Extended Events session created, a new empty window will be displayed in the SQL Server Management Studio, in which the caught events will be displayed, as follows:

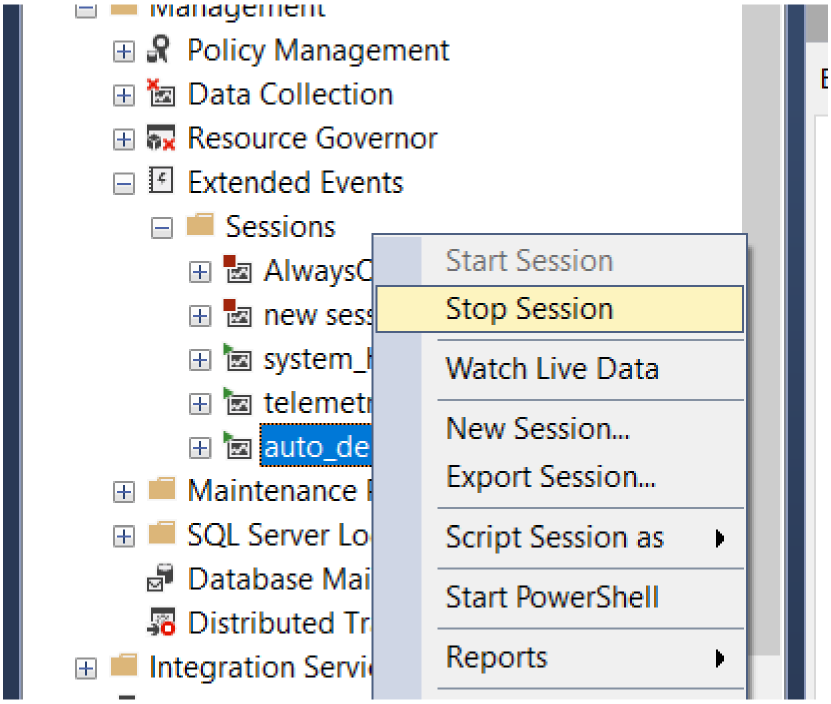


name	timestamp
error_reported	2021-09-22 10:52:00....
error_reported	2021-09-22 10:52:00....
erro error_reported	2021-09-22 10:52:00....

Event:error_reported (2021-09-22 10:52:00.6701047)

Field	Value
category	SERVER
destination	USER
error_number	5701
is_intercepted	False
message	Changed database context to 'master'.
severity	10
state	1
user_defined	False

After performing successful and failed login processes, the events will be collected and displayed by the SQL Server Extended Events session. For example, the successful login process properties, including the user name, the host name, the application used for the login and other useful information will be displayed as shown clearly below:



Logs

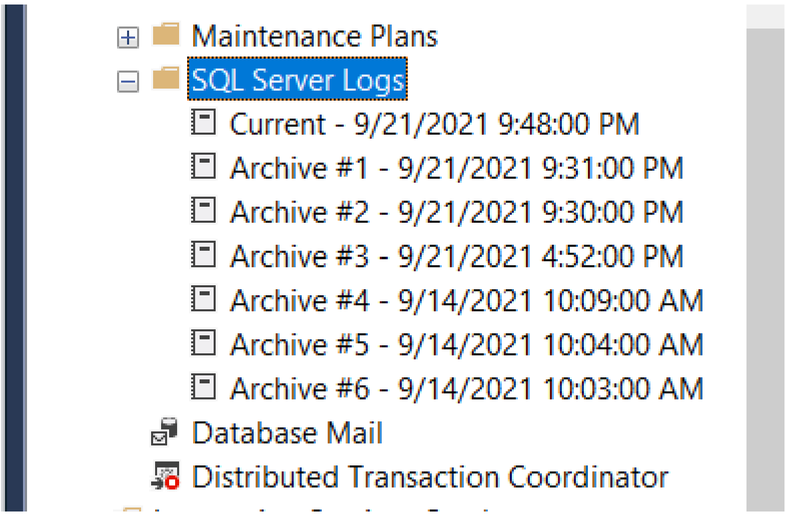
The Windows application event log provides an overall picture of events occurring on the Windows Server and Windows operating systems as a whole, as well as events in SQL Server, SQL Server Agent, and full-text search. It contains information about events in SQL Server that is not available elsewhere. You can use the information in the error log to troubleshoot SQL Server-related problems.

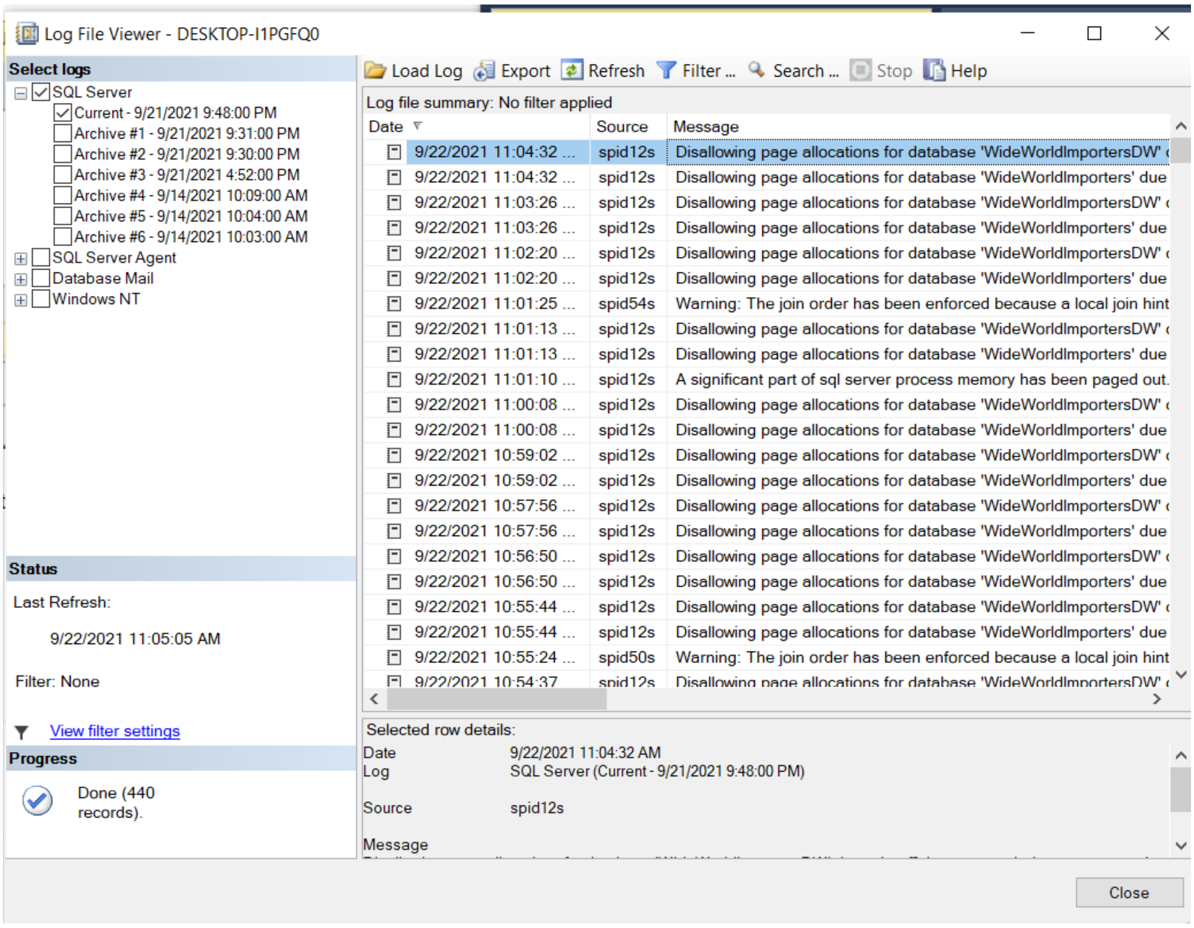
In SQL Server Management Studio, select Object Explorer. To open Object Explorer, select F8. Or on the top menu, select View, and then select Object Explorer:

In Object Explorer, connect to an instance of SQL Server, and then expand that instance.

Find and expand the Management section (assuming you have permissions to see it).

Right-click SQL Server Logs, select View, and then choose SQL Server Log.


Maintenance Plans
SQL Server Logs
Current - 9/21/2021 9:48:00 PM
Archive #1 - 9/21/2021 9:31:00 PM
Archive #2 - 9/21/2021 9:30:00 PM
Archive #3 - 9/21/2021 4:52:00 PM
Archive #4 - 9/14/2021 10:09:00 AM
Archive #5 - 9/14/2021 10:04:00 AM
Archive #6 - 9/14/2021 10:03:00 AM
Database Mail
Distributed Transaction Coordinator



Select logs

- ☑ SQL Server
  - ☑ Current - 9/21/2021 9:48:00 PM
  - □ Archive #1 - 9/21/2021 9:31:00 PM
  - □ Archive #2 - 9/21/2021 9:30:00 PM
  - □ Archive #3 - 9/21/2021 4:52:00 PM
  - □ Archive #4 - 9/14/2021 10:09:00 AM
  - □ Archive #5 - 9/14/2021 10:04:00 AM
  - □ Archive #6 - 9/14/2021 10:03:00 AM
- □ SQL Server Agent
- □ Database Mail
- □ Windows NT

Status

Last Refresh:

9/22/2021 11:05:05 AM

Filter: None

View filter settings

Progress

Done (440 records).

Log file summary: No filter applied

Date	Source	Message
9/22/2021 11:04:32 ...	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'
9/22/2021 11:04:32 ...	spid12s	Disallowing page allocations for database 'WideWorldImporters' due
9/22/2021 11:03:26 ...	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'
9/22/2021 11:03:26 ...	spid12s	Disallowing page allocations for database 'WideWorldImporters' due
9/22/2021 11:02:20 ...	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'
9/22/2021 11:02:20 ...	spid12s	Disallowing page allocations for database 'WideWorldImporters' due
9/22/2021 11:01:25 ...	spid54s	Warning: The join order has been enforced because a local join hint
9/22/2021 11:01:13 ...	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'
9/22/2021 11:01:13 ...	spid12s	Disallowing page allocations for database 'WideWorldImporters' due
9/22/2021 11:01:10 ...	spid12s	A significant part of sql server process memory has been paged out.
9/22/2021 11:00:08 ...	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'
9/22/2021 11:00:08 ...	spid12s	Disallowing page allocations for database 'WideWorldImporters' due
9/22/2021 10:59:02 ...	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'
9/22/2021 10:59:02 ...	spid12s	Disallowing page allocations for database 'WideWorldImporters' due
9/22/2021 10:57:56 ...	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'
9/22/2021 10:57:56 ...	spid12s	Disallowing page allocations for database 'WideWorldImporters' due
9/22/2021 10:56:50 ...	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'
9/22/2021 10:56:50 ...	spid12s	Disallowing page allocations for database 'WideWorldImporters' due
9/22/2021 10:55:44 ...	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'
9/22/2021 10:55:44 ...	spid12s	Disallowing page allocations for database 'WideWorldImporters' due
9/22/2021 10:55:24 ...	spid50s	Warning: The join order has been enforced because a local join hint
9/22/2021 10:54:37	spid12s	Disallowing page allocations for database 'WideWorldImportersDW'

Selected row details:

Date	9/22/2021 11:04:32 AM
Log	SQL Server (Current - 9/21/2021 9:48:00 PM)
Source	spid12s
Message	