How to Run SQL Profiler Trace to Capture Information

03/27/2025 10:52 am EDT

Issue:

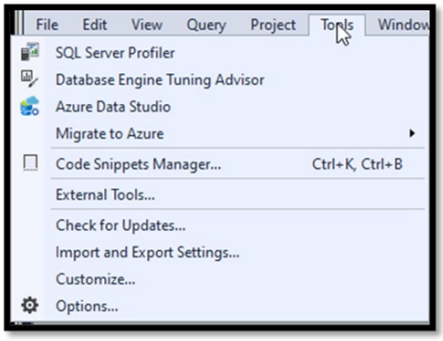
Customer is receiving errors in their SedonaOffice company. Is there a way to get more information on why the errors occur?

Resolution:

SQL Server Profiler can capture information from processes accessing the SQL server databases. Using SQL Server Profiler, you can create a trace that records and displays the commands sent to and the responses from the SQL server.

Open SQL Server Management Studio

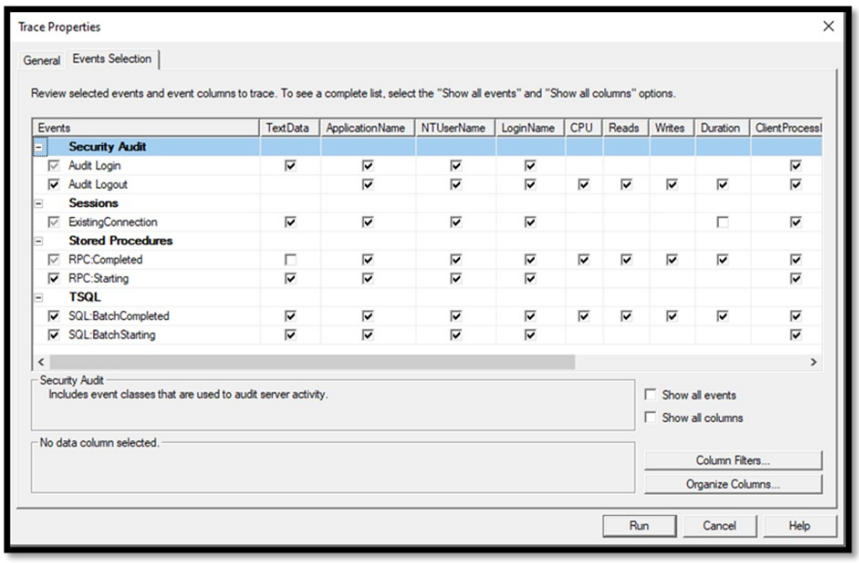
Connect to the correct SQL server w/ a sysadmin user and select Tools then SQL Server Profiler.



It can also be launched from the Windows Start Menu

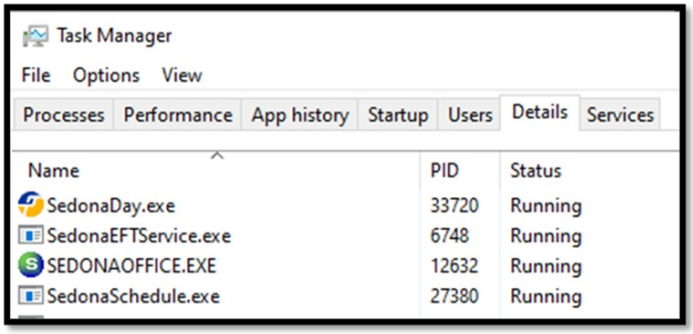
In the Events Selection Tab, you'll see the events it will monitor.

Be sure the options selected are the same as in the screenshot below

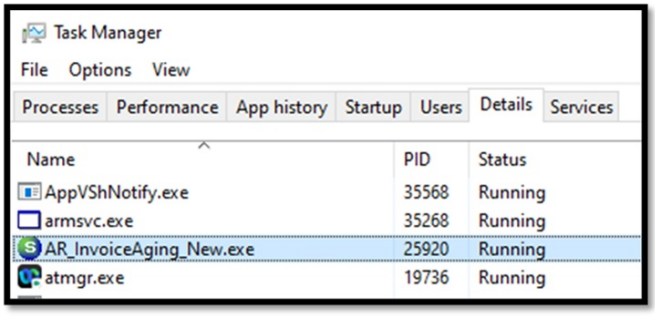


To limit the results to the machine having the issue we use the ProcessID[PID] from the application.
You can get this from the Windows Task Manager on the Details tab.

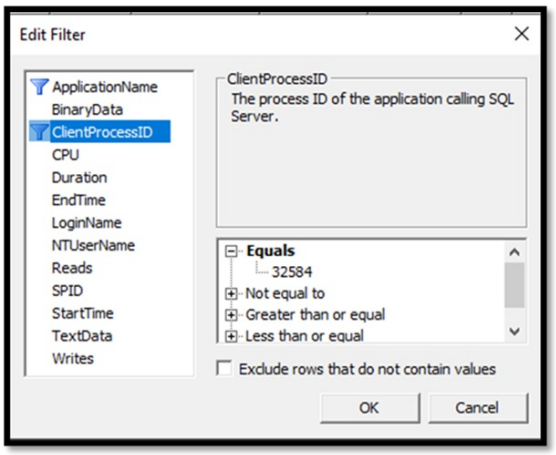
If the problem is with a report or Service Ticket, you will need to look for the exe for the particular report or the SedonaScheduler



If the problem is any other application that opens outside of the SedonaOffice window, look for a separate application in the Task Manager. You may need to use that [PID] instead of the SedonaOffice.exe



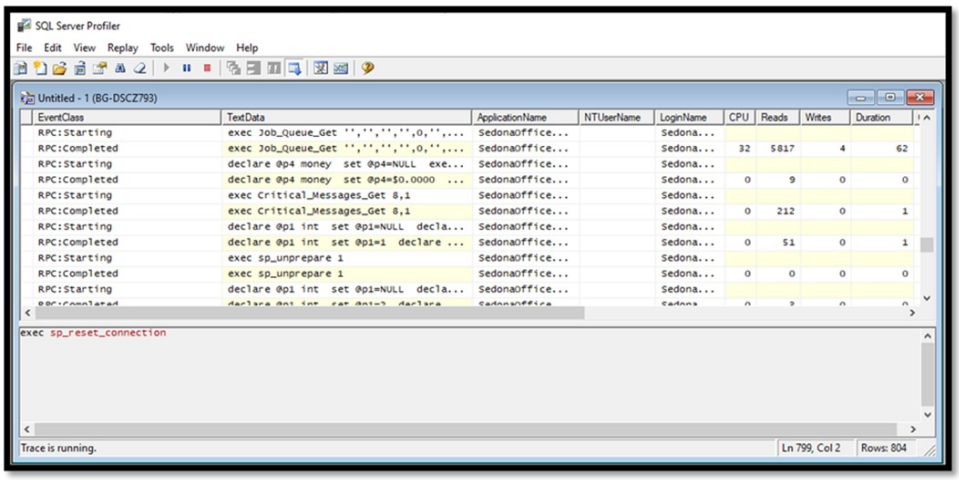
Enter the PID from the application in the Column Filter for ClientProcessID and Click OK



Once the events and filters are correct, click Run on the Trace Properties.
NOTE: Start the trace **immediately before** performing the action

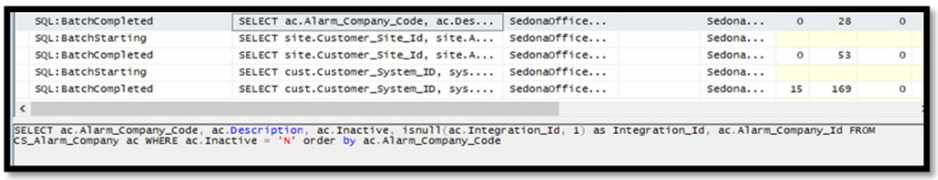
Once running, it will capture the activity between the application and the server.
Perform the action in the application that is causing the error. Once the error is received, stop the Trace and save the file.

The trace will show every transaction between the application and server and can be used to see what event caused the errors. The saved trace file can be attached to the support case or emailed to Support/Development for analysis.

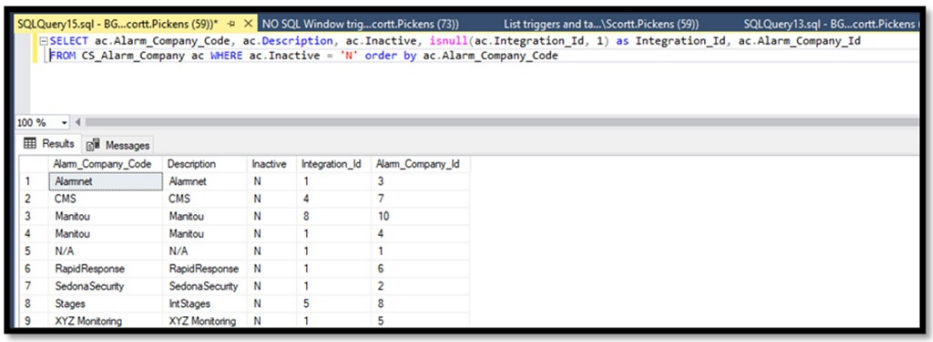


If the trace is stopped as soon as the error occurs, you should be able to see the last transactions that caused the error.

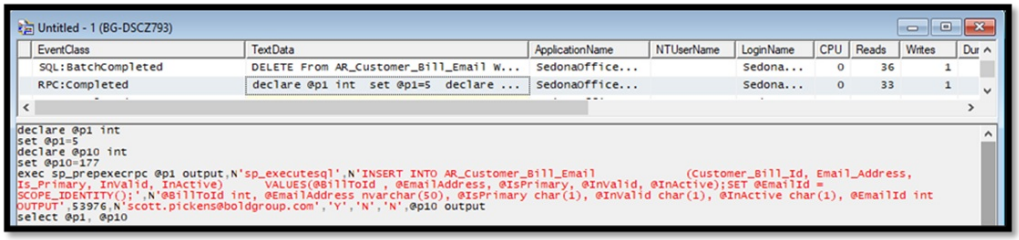
You can select the line in the trace to see what transaction was attempted/occurring.



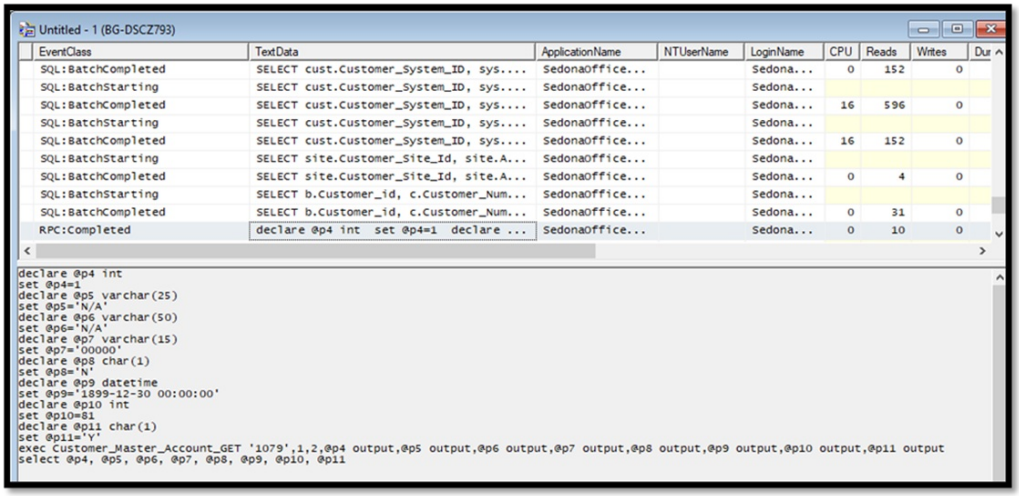
You can copy and paste this in an SSMS Query window to attempt executing the transaction to see if there is a problem.
NOTE: Be careful of procedures that are (or include) an update, add, Delete, or an Insert. If executed, these may add or change records that should not be changed. Always ensure you have backed up the database or table before executing changes -or- the customer has a refreshed sandbox you can execute the changes against to replicate the results



If it is a stored procedure, you can copy and paste the statement also.
If you need to test something like this, it is always recommended to use the Sandbox company.



If it is a Select or a stored procedure to GET information, it is OK to run in the live company. This is OK to run since it is a stored procedure to get the Master Account information and will not be making changes.
NOTE: If the GET is pulling too much information or from a lot of different fields this could result in performance issues for end users. STOP the query if it runs for longer than a couple of minutes and advise the customer that this should be tested during non-peak or outside of business hours. Queries being executed for extended periods of time



The results will help narrow down where the problem is.