Technical Note: Database Monitoring

**Description:** This document describes troubleshooting steps and Monitoring for various frequent issues experienced with EnterWorks product at database level.

**EnterWorks Product Applicability:** All versions of Enable.

**Third Party Product:** Microsoft SQL Server

**Customer Scope:** Applicable on all customers.

**Note:** Scope this Document is for Monitoring and General health Checkup of Database. Its Highly Recommended **DO Not Kill any Transaction**, it may go into Rollback state.

# Overview

This document is intended to be used by DevOps/Support Team to troubleshoot customer issues on database level and serve as a guide for quickly identifying issues and solving them.

## 1.Connect to the SQL Server using SSMS

To connect to the SQL Server using the Microsoft SQL Server Management Studio, you use these steps:

First, launch the Microsoft SQL Server Management Studio from the **Start** menu:

Graphical user interface, text, application

Description automatically generated

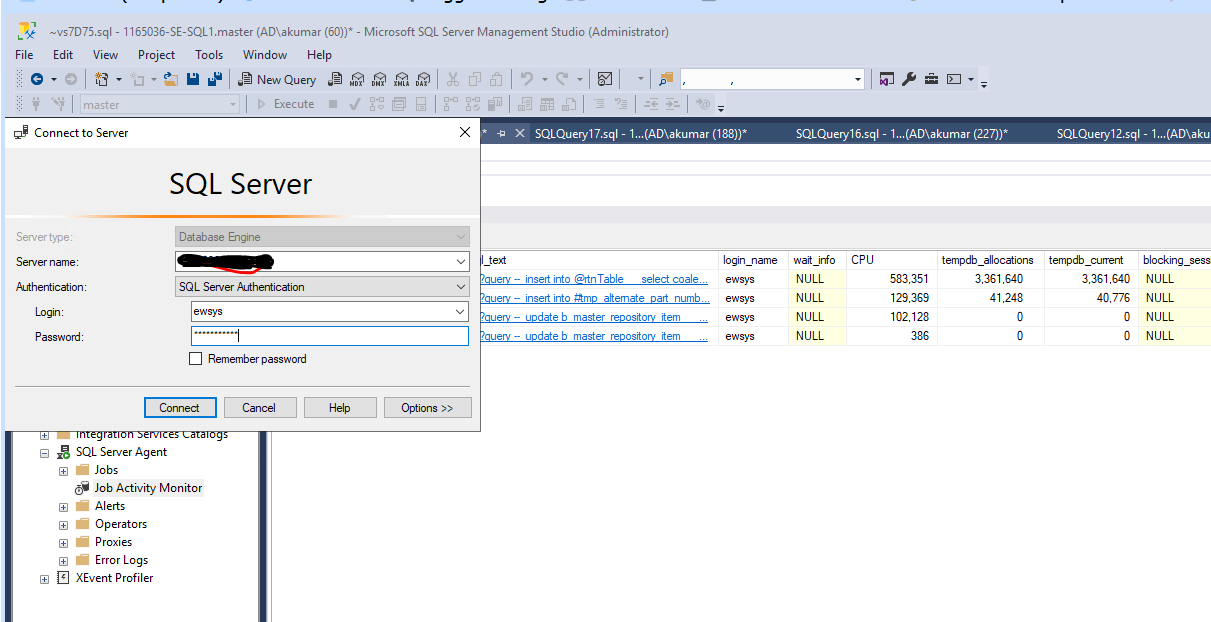
Step2. Select Connect menu under the Object Explorer, choose the Database Engine.

Graphical user interface, application, Word

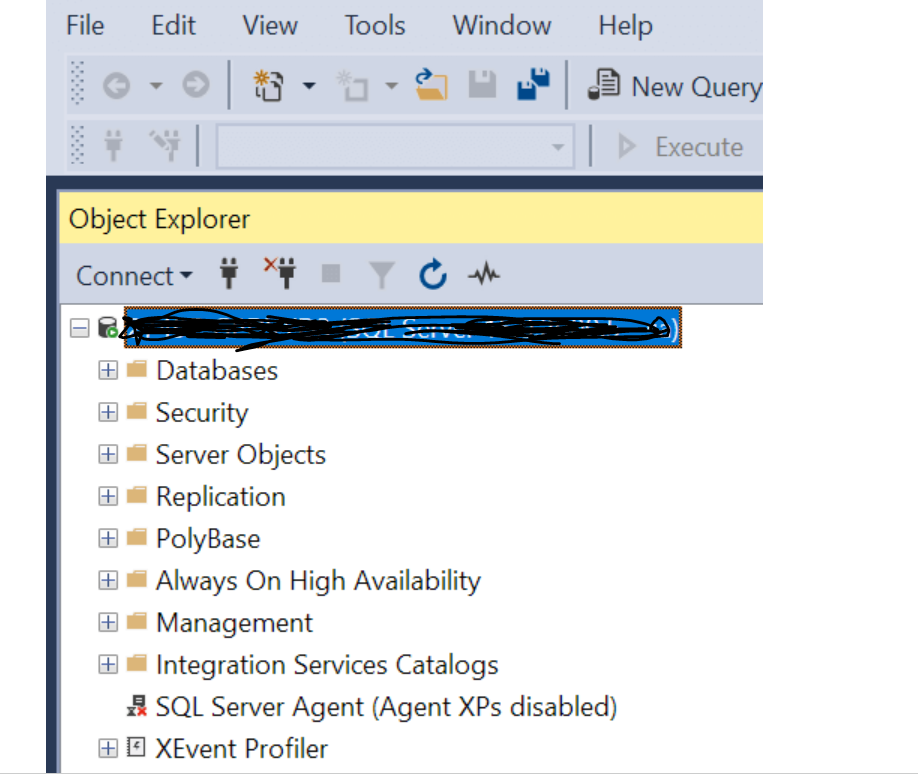
Description automatically generated

# Step3.Connect to MSSQL Server Management Studio

Then, enter the information for the Server name (Instance Name/Endpoint), Authentication (SQL Server Authentication), and password for the ewsys user and click the **Connect** button to connect to the SQL Server.



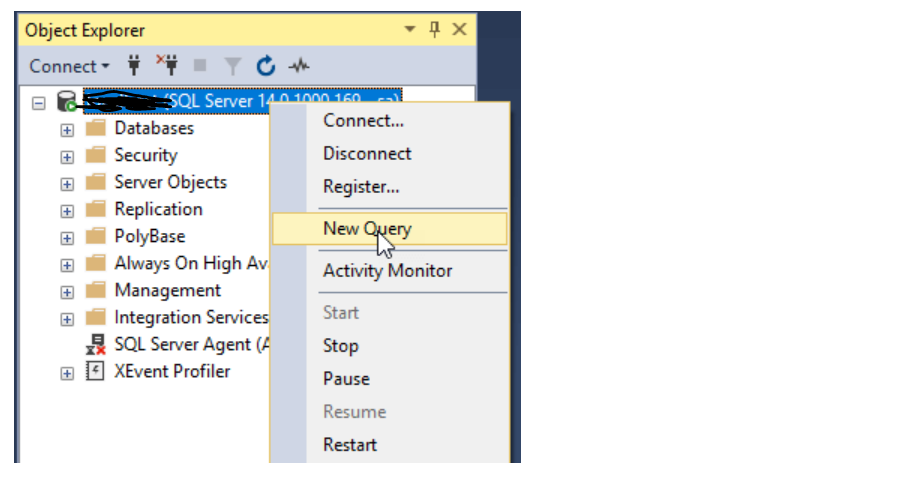
5. Once connection is established successfully, then you will see the following Object Explorer panel:



## **6.Execute a query**

To execute a query, you follow these steps:

First, right-click on the DB SERVER **(Instance Name/Endpoint)** node and choose the **New Query** menu item:

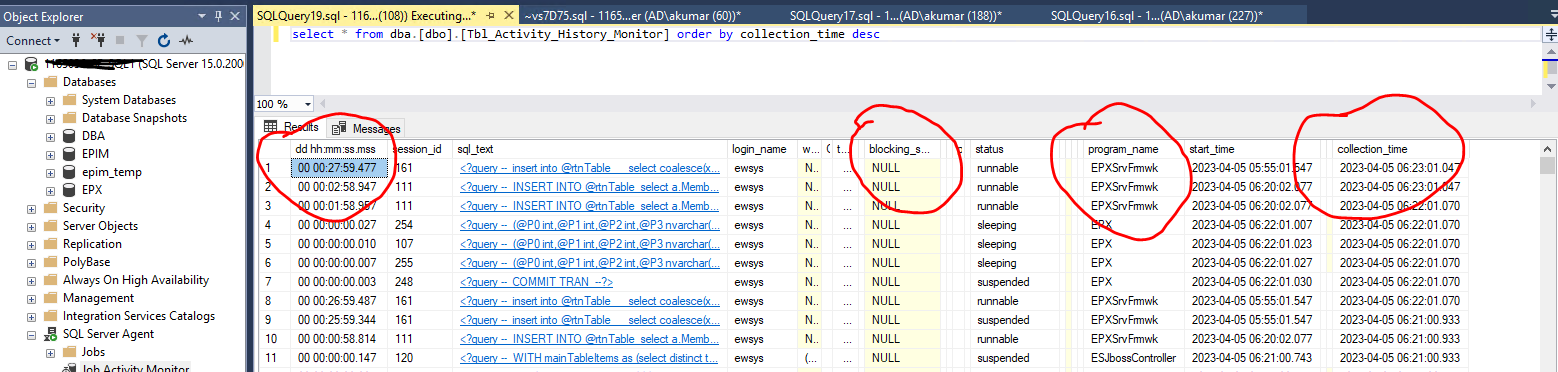


## **7.Find blocking sessions.**

Open SQL Server management studio -> new query window and execute the following SQL statement.

select \* from dba.[dbo].[Tbl\_Activity\_History\_Monitor] order by collection\_time desc

* To check the blocking session pls. check “blocking\_session\_id” Column in O/P window. If any SPID Shows Blocking for > 30 Minutes. It’s a Active Blocking,need to check the Blocked query in “Sql\_Text” Column name in Query O/P Window,and try to find out he cause of the Blocking.

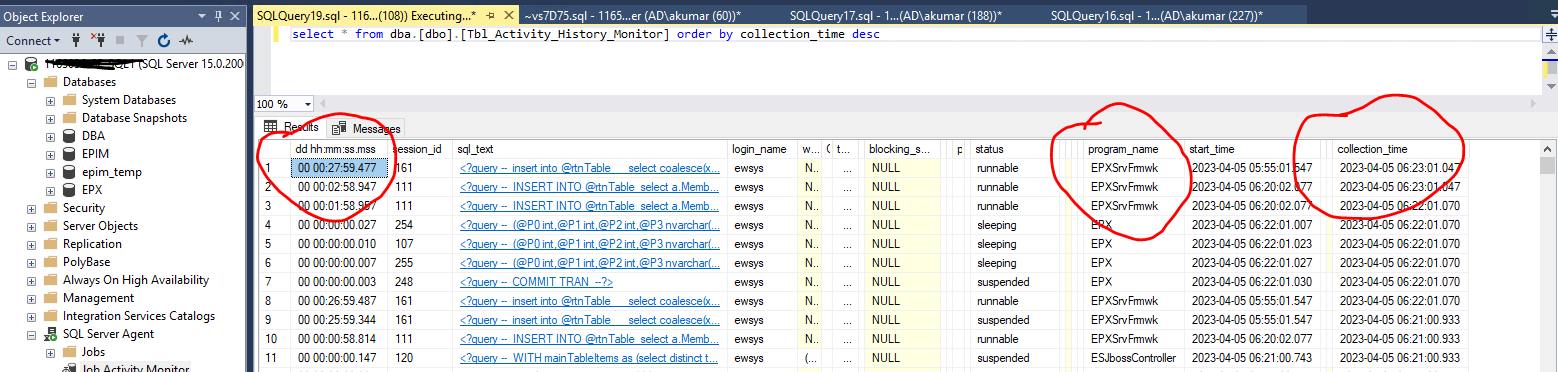


## **8.Find Long Running sessions.**

Open SQL Server management studio -> new query window and execute the following SQL statement.

select \* from dba.[dbo].[Tbl\_Activity\_History\_Monitor] order by collection\_time desc

To check the blocking session pls. check “blocking\_session\_id” Column in O/P window. If any SPID Shows Blocking for > 30 Minutes. It’s a Active Blocking,need to check the Blocked query in “Sql\_Text” Column name in Query O/P Window,and try to find out he cause of the Blocking.



**For Further Depth Level Query Analysis, use the Table below to get the description of each column in table.**

|  |  |
| --- | --- |
| Column Name | Description |
| dd hh:mm:ss.mss | the timestamp value that indicates how much time a query has been running |
| session\_id | the session id from where the query is being executed |
| Sql\_Text | the actual TSQL statement being executed |
| login\_name | the name of the login that triggered the query |
| wait\_info | the information on what the query is waiting on (if it's not waiting for anything this value is NULL) |
| CPU | the CPU time, in milliseconds, that the query has been consuming so far |
| tempdb\_allocations | NIL |
| tempdb\_current | NIL |
| blocking\_session\_id | iif there is any blocking, this field will state the session id of the query that is still running and that must finish for the actual blocked query to run |
| reads | the number of logical reads (in 8KB pages) performed by the query |
| writes | the number of writes performed by the query |
| physical\_reads | the number of physical reads (in 8KB pages) performed by the query |
| used\_memory | the amount of memory the query has been using so far |
| status | the current status of the query (sleeping, dormant, runnable, suspended, etc.) |
| open\_tran\_count | NIL |
| percent\_complete | NIL |
| host\_name | the hostname from where the query was triggered |
| database\_name | the target database where the query will be executed |
| program\_name | the name of the application that triggered the query |
| start\_time | the timestamp value when the query was triggered |
| login\_time | the timestamp value when the query user login to database |
| request\_id | nil |
| collection\_time | the timestamp value when the record was fetched from the instance |