

# Removing Watched Server Objects

Last Modified on 08 October 2021

## Stop Watching

Enable **Watching** and **Stop Watching** on the targets within your server environment.

**Note:** If you choose to **Stop Watching** a target, remember to reclaim the license for that target, allowing you to apply it elsewhere. For more information, see the [License Management](#) article.

**Note:** You can use PowerShell to stop watching targets. See the [PowerShell Module](#) article for more information.

**Note:** The **Inventory** node provides a read-only view of what you are watching. See the [Inventory View](#) article for details.

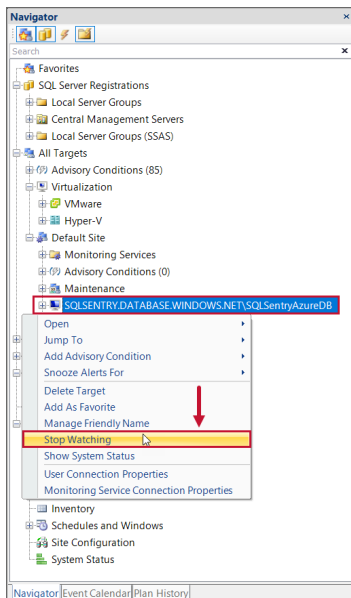
To **Stop Watching** a target in SentryOne, complete one of the following procedures:

### Using the Navigator pane

Stop watching a target through the **Navigator** pane:

**View > Navigator > *Desired Target* > Stop Watching**

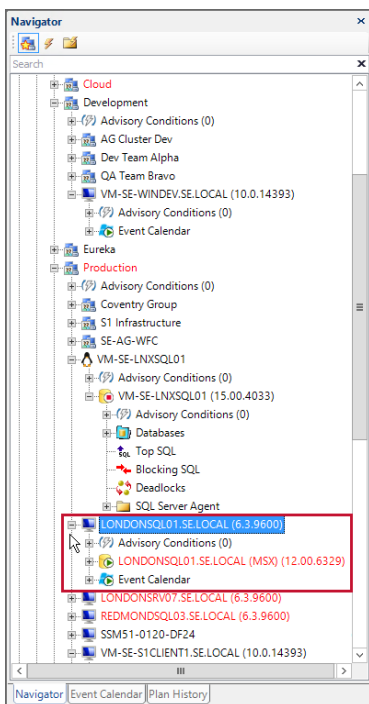
1. Open the **Navigator** pane (**View > Navigator**), and then select your desired target.
2. Right-click on the desired target, and then select **Stop Watching**.



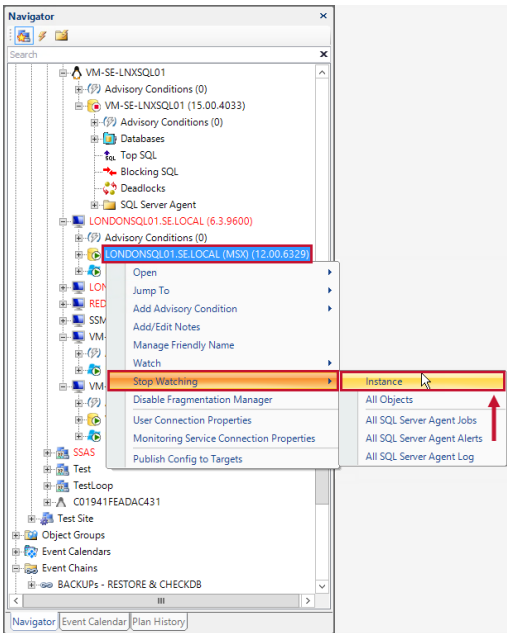
Stop watching a target hosting a SQL Server instance through the **Navigator** pane:

**View > Navigator > Desired Target > Desired Instance > Stop Watching Instance**

1. Open the Navigator pane (**View > Navigator**), and then expand your desired target.



2. Right click on the SQL Server Instance, and then select **Stop Watching Instance**.

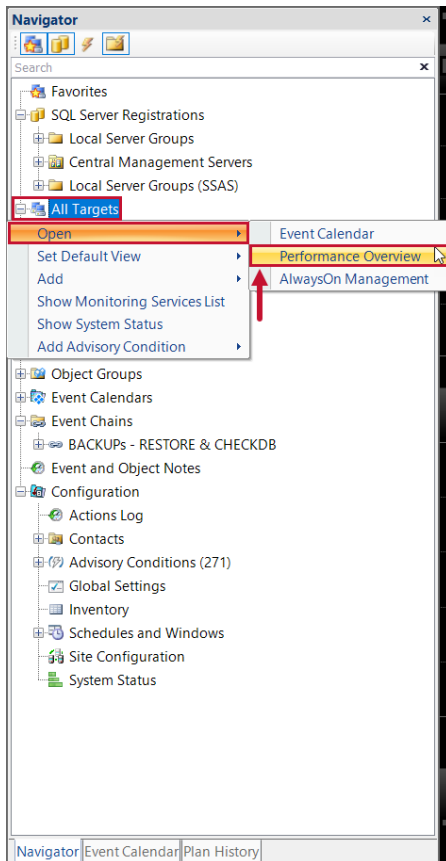


3. Right click on the target, and then select **Stop Watching**.

## Using Performance Overview

Stop watching through **Performance Overview**:

1. Right-click **All Targets (View > Navigator > All Targets)**, select **Open**, and then select **Performance Overview**. This opens the **All Targets** tab.



2. Right-click on the desired target, then select **Stop Watching**.

Target	Main Proc	Proc Time %	Kernel Time %	Context Switches	Queue Length	Total	Used	Free	Read	Write	Mbps	IOPS	msIO	Mbps	Stops	msIO	Total	Free	Adaptor Count	Config	Mbps	%	Packets	Errors	Mbps	%	Packets	Errors		
COMSERV10176.LOCAL	4	10	3	2,520	0	8,190	5,698	2,491	0	0.0	0	0	0.0	0	7.4	40	225.5	81.2	1,000	1	0.0	0.1	100	0	1.7	0.2	298	0		
LODWSRV176.LOCAL	4	6	1	2,065	0	8,190	1,768	6,422	0	0.0	0	0	0.0	1	0.0	1	237.5	194.2	1,000	1	1.3	0.1	602	0	2.6	0.2	646	0		
RDWMSRV176.LOCAL	4	0	0	387	0	8,190	2,659	5,531	0	0.0	0	0	0.0	1	3	175.5	128.9	1,000	1	0.1	0.0	33	0	0.1	0.0	19	0			
SSRSRV176.LOCAL	4	100	0	717	66	8,190	8,040	2,142	0	0.0	0	0	0.0	3	1	201.5	96.2	1,000	1	0.0	0.0	34	0	0.1	0.0	24	0			
SSRSRV176	16	9	1	2,625	0	136,869	124,990	5,979	0	0.0	0	0	0.0	0	7.4	130	4,149.3	96.7	4,000	4	15.5	0.4	2,947	0	6.9	0.2	2,206	0		
VM-10-102-10774	4	1	0	867	0	8,143	5,594	2,549	0	0.0	0	0	0.0	0	0.0	2	1	178.5	96.1	1,000	1	0.1	0.0	42	0	0.1	0.0	33	0	
VM-10-102-10774	4	15	1	5,121	0	8,091	5,220	2,870	0	0.0	0	0	0.0	3	1	191.5	129.1	1,000	1	0.1	0.0	42	0	0.2	0.0	37	0			
VM-10-101-10176-101	0	3	2	30,795	0	32,767	6,478	25,289	1	0.0	0	0	0.0	0	0.0	0	1	111.5	47.1	10,000	1	0.6	0.0	141	0	0.3	0.0	99	0	
VM-10-101-10176-101	4	0	0	549	0	8,190	5,152	2,937	0	0.0	0	0	0.0	1	4	896.5	801.7	1,000	1	0.1	0.0	42	0	0.3	0.0	23	0			
VM-10-102-10774	8	6	2	12,945	0	24,274	7,962	16,311	0	0.0	0	0	0.0	2	1	391.9	442.3	1,000	1	0.5	0.0	373	0	0.8	0.0	115	0			
VM-10-102-10774	2	23	5	4,093	0	8,190	2,227	5,963	1	0.0	0	0	0.0	2	1	431.9	32.9	1,000	1	14.2	1.4	1,892	0	2.6	0.3	1,167	0			
VM-10-102-10774	4	20	2	5,852	0	16,383	15,126	1,257	0	0.0	0	0	0.3	8	2	9.6	38	1	962.9	596.2	10,000	1	5.2	0.1	1,548	0	12.6	0.1	1,225	0
VM-10-102-10774	4	4	1	742	0	12,296	7,805	4,491	0	0.0	0	0	0.0	2	1	892.7	696.3	1,000	1	0.4	0.0	104	0	1.1	0.1	173	0			
VM-10-102-10774	4	0	0	1,994	0	16,383	15,779	604	0	0.1	6	2	6.4	13	1	831.5	590.1	10,000	1	0.7	0.0	120	0	0.2	0.0	98	0			
VM-10-102-10774	4	0	1	9,402	0	8,191	7,201	990	0	0.0	0	0	0.0	2	1	111.5	44.8	10,000	1	0.1	0.0	47	0	0.4	0.0	31	0			
VM-10-102-10774	2	15	3	7,278	10	8,191	2,990	5,192	21	0.0	0	0	0.1	18	1	183.5	46.9	10,000	1	17.8	0.2	3,209	0	0.9	0.1	2,381	0			
VM-10-102-10774	2	63	4	10,406	0	16,383	9,558	7,825	0	0.0	0	0	0.0	0	1	133.7	62.7	10,000	1	0.1	0.0	62	0	0.3	0.0	63	0			

## Removing Watched Server Objects

**Note:** When you watch a SQL Server with **Event Manager**, SQL Sentry places a few objects in MSDB that facilitate its lightweight polling architecture. No agents are placed on the server. This enables SQL Sentry to monitor the server with a performance overhead that's typically less than SQL Agent. For more information about watched server objects, see the [Watched Server Objects](#) topic.

If you've **stopped watching** a SQL Server instance or Azure SQL Database target with SentryOne, and don't have plans to **Watch** it again, scripts are provided to automate the process of removing the objects SentryOne places on a watched target.

**Important:** **Stop Watching** the server through the SentryOne client before attempting to remove the object.

Select the appropriate tab below to view the script.

### SQL Server 2000 Instances

```
--SQL Server 2000
USE msdb
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[sp_sentry_mail]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[sp_sentry_mail_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[SQLSentryEmails_20]')
and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryEmails_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spGetBlockInfo_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetBlockInfo_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spGetBlockInfo_Pre8sp3]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetBlockInfo_Pre8sp3]
GO
BEGIN TRANSACTION
DECLARE @jobid BINARY(16)
```

```

DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
-- Delete the job with the same name (if it exists)
SELECT @JobID = job_id
FROM msdb.dbo.sysjobs
WHERE (name = N'SQL Sentry 2.0 Queue Monitor')
IF (@JobID IS NOT NULL)
BEGIN
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobsservers
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job "SQL Sentry Queue Monitor" since there is already a multi-server job with this name.'
, 16, 1)
GOTO QuitWithRollback
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Queue Monitor'
SELECT @JobID = NULL
END
COMMIT TRANSACTION
GOTO EndSave QuitWithRollback:
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spGetJobInfo_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetJobInfo_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spGetDTSLog_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetDTSLog_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spQueueHeartbeat_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueHeartbeat_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spQueueJob_Start_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueJob_Start_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spQueueJob_End_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueJob_End_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spQueueMonitor_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueMonitor_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spReadLogFile_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spReadLogFile_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[SQLSentryQueueLog_20]')
and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryQueueLog_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[SQLSentryLogCache_20]')
and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogCache_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[SQLSentryLogCachedTS_20]')
and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogCachedTS_20]
GO
BEGIN TRANSACTION
DECLARE @JobID BINARY(16)
DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
-- Delete the job with the same name (if it exists)

```

```

SELECT @JobID = job_id
FROM msdb.dbo.sysjobs
WHERE (name = N'SQL Sentry 2.0 Alert Trap')
IF (@JobID IS NOT NULL)
BEGIN
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobobservers
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job "SQL Sentry Alert Trap" since there is already a multi-server job with this name.', 16,
1)
GOTO QuitWithRollback
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Alert Trap'
SELECT @JobID = NULL
END
COMMIT TRANSACTION
GOTO EndSave QuitWithRollback:
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spTrapAlert_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spTrapAlert_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[spSetupAlertsTrap_20]')
and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spSetupAlertsTrap_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[SQLSentryAlertLog_20]')
and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryAlertLog_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[SQLSentryLogData_20]')
and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogData_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[SQLSentryObjectVersion_20]')
and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryObjectVersion_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[fnGetSQL_20]')
and OBJECTPROPERTY(id, N'IsScalarFunction') = 1)
drop function [dbo].[fnGetSQL_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[fnGetWaittypeDesc_20]')
and OBJECTPROPERTY(id, N'IsScalarFunction') = 1)
drop function [dbo].[fnGetWaittypeDesc_20]

```

## SQL Server 2005+ Instances

```

--SQL Server 2005+
USE msdb
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[sp_sentry_mail]') and OBJECTPROPERTY(object_id,
N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[sp_sentry_mail_20]') and OBJECTPROPERTY(object
_id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[sp_sentry_dbmail_20]') and OBJECTPROPERTY(obj
ect_id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_dbmail_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[SQLSentryEmails_20]') and OBJECTPROPERTY(obj

```

```

ect_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryEmails_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[SQLSentryDBEmails_Attachments_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryDBEmails_Attachments_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[SQLSentryDBEmails_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryDBEmails_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spGetBlockInfo_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetBlockInfo_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spGetBlockInfo_Pre8sp3]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetBlockInfo_Pre8sp3]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spGetQueryStatsData]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetQueryStatsData]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spGetProcedureStatsData]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetProcedureStatsData]
GO
BEGIN TRANSACTION
DECLARE @JobID BINARY(16)
DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
-- Delete the job with the same name (if it exists)
SELECT @JobID = job_id
FROM msdb.dbo.sysjobs
WHERE (name = N'SQL Sentry 2.0 Queue Monitor')
IF (@JobID IS NOT NULL)
BEGIN
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobserver
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job "SQL Sentry Queue Monitor" since there is already a multi-server job with this name.', 16, 1)
GOTO QuitWithRollback
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Queue Monitor'
SELECT @JobID = NULL
END
COMMIT TRANSACTION
GOTO EndSave QuitWithRollback:
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spGetJobInfo_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetJobInfo_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spGetDTSLog_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetDTSLog_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spQueueHeartbeat_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueHeartbeat_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spQueueJob_Start_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueJob_Start_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spQueueJob_End_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueJob_End_20]

```

```

ct_id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueJob_End_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spQueueMonitor_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueMonitor_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spReadLogFile_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spReadLogFile_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[SQLSentryQueueLog_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryQueueLog_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[SQLSentryLogCache_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogCache_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[SQLSentryLogCacheDTS_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogCacheDTS_20]
GO
BEGIN TRANSACTION
DECLARE @JobID BINARY(16)
DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
-- Delete the job with the same name (if it exists)
SELECT @JobID = job_id
FROM msdb.dbo.sysjobs
WHERE (name = N'SQL Sentry 2.0 Alert Trap')
IF (@JobID IS NOT NULL)
BEGIN
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobserver
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job "SQL Sentry Alert Trap" since there is already a multi-server job with this name.', 16, 1)
GOTO QuitWithRollback
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Alert Trap'
SELECT @JobID = NULL
END
COMMIT TRANSACTION
GOTO EndSave QuitWithRollback:
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spTrapAlert_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spTrapAlert_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[spSetupAlertsTrap_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spSetupAlertsTrap_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[SQLSentryAlertLog_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryAlertLog_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[SQLSentryLogData_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogData_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[SQLSentryObjectVersion_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryObjectVersion_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[fnGetSQL_20]') and OBJECTPROPERTY(object_id, N'IsFunction') = 1)
drop function [dbo].[fnGetSQL_20]

```



```

if exists (select * from sys.objects where object_id = object_id(N'[dbo].[fnGetSQL_20]') and OBJECTPROPERTY(object_id, N'IsScalarFunction') = 1)
drop function [dbo].[fnGetSQL_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].[fnGetWaitypeDesc_20]') and OBJECTPROPERTY(object_id, N'IsScalarFunction') = 1)
drop function [dbo].[fnGetWaitypeDesc_20]
if exists (select * from sys.objects where object_id = object_id(N'[SQLSentry].[SQLSentryObjectVersion_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [SQLSentry].[SQLSentryObjectVersion_20]

```

## Azure SQL Database

```

--Azure
if exists (select * from sys.objects where object_id = object_id(N'[SQLSentry].[spGetProcedureStatsData]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [SQLSentry].[spGetProcedureStatsData]
if exists (select * from sys.objects where object_id = object_id(N'[SQLSentry].[spGetQueryStatsData]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [SQLSentry].[spGetQueryStatsData]
if exists (select * from sys.objects where object_id = object_id(N'[SQLSentry].[SQLSentryObjectVersion_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [SQLSentry].[SQLSentryObjectVersion_20]
DECLARE @listOfSqlSentryTables VARCHAR(MAX)
DECLARE @sqlStatement VARCHAR(MAX)
select @listOfSqlSentryTables = COALESCE(@listOfSqlSentryTables+',','') + 'SQLSentry.' + name from sys.objects where name like N'ProcedureStats%' and OBJECTPROPERTY(object_id, N'IsUserTable') = 1 and schema_id = schema_id('SQLSentry')
set @sqlStatement = 'drop table ' + @listOfSqlSentryTables
exec(@sqlStatement)
SET @listOfSqlSentryTables = NULL;
select @listOfSqlSentryTables = COALESCE(@listOfSqlSentryTables+',','') + 'SQLSentry.' + name from sys.objects where name like N'QueryStats%' and OBJECTPROPERTY(object_id, N'IsUserTable') = 1 and schema_id = schema_id('SQLSentry')
set @sqlStatement = 'drop table ' + @listOfSqlSentryTables
exec(@sqlStatement)
if schema_id('SQLSentry') is not null
drop schema SQLSentry

```