“Año De La Recuperación Y
Consolidación De La Economía Peruana”

UNIVERSIDAD PERUANA LOS ANDES

“FACULTAD DE INGENIERÍA”

ESCUELA PROFESIONAL “SISTEMAS Y COMPUTACIÓN”

Semana 13: Monitoreo y rendimiento

CÁTEDRA: Base de Datos II

CATEDRÁTICO: Ing. Fernandez Bejarano Raul Enrique

ESTUDIANTE: Quispe Segama Franklin Noe

CICLO: V

SECCIÓN: A1

HUANCAYO PERÚ

2025

1. Análisis de rendimiento con SQL Profiler y Extended Events

✔ SQL Profiler

Es una herramienta gráfica que permite capturar eventos del servidor “en tiempo real”.
Se usa para:

- Identificar consultas lentas.
- Detectar bloqueos (locks).
- Ver uso intensivo de CPU o IO.
- Auditar actividad sospechosa.

Desventaja:
Consume muchos recursos → no recomendado en producción durante mucho tiempo.

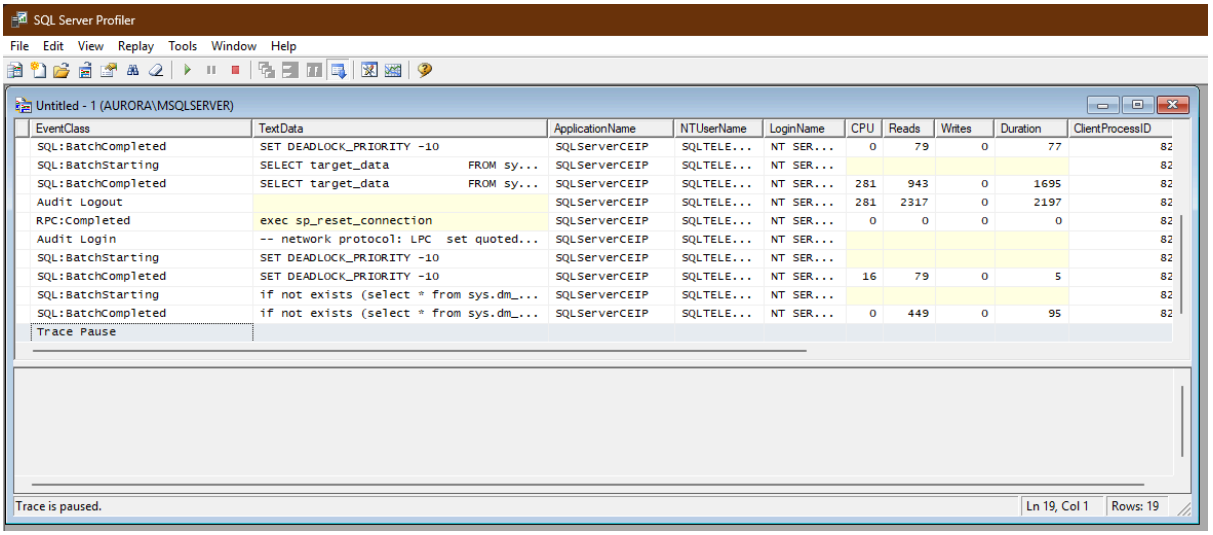
✔ Extended Events (XE)

Función moderna y ligera. Reemplaza al Profiler.

Beneficios:

- Bajo consumo de recursos.
- Mejor granularidad y filtros.
- Se puede almacenar en archivos para análisis posterior.

Uso típico:
Detectar deadlocks, capturar consultas que exceden cierto tiempo, monitorear el buffer pool.



2. Estadísticas e índices (creación, fragmentación, mantenimiento)

✔ Estadísticas

Las estadísticas describen la distribución de datos en una columna.
SQL Server las usa para generar **planes de ejecución óptimos**.

Problemas comunes:

- Estadísticas desactualizadas → consultas lentas.

Mantenimiento recomendado:

```sql
--Mantenimiento recomendado:
Use QhatuPERU;
UPDATE STATISTICS ARTICULO;
```

```
Messages
Commands completed successfully.

Completion time: 2025-11-26T16:36:31.0270081-05:00
```

O activar auto-update:

```sql
--O activar auto-update:
ALTER DATABASE QhatuPERU SET AUTO_UPDATE_STATISTICS ON;
```

```
Messages
Commands completed successfully.

Completion time: 2025-11-26T16:38:39.9889573-05:00
```

✔ Índices

Son estructuras que aceleran la búsqueda de datos.

Tipos:

- **Clustered:** ordenan físicamente los datos.

- **Nonclustered:** mejoran búsquedas específicas.
- **Columnstore:** para análisis masivo (OLAP).
- **Full-text:** búsquedas de texto.

✔ Fragmentación

Cuando las páginas de un índice quedan desordenadas, el servidor necesita más IO → baja el rendimiento.

Soluciones:

- Menor a 30% → reorganizar:

```
--Menor a 30% → reorganizar:
ALTER INDEX PK__PROVEEDO__BFBE6B07AEFC9CA5 ON PROVEEDOR REORGANIZE;
ALTER INDEX PK GUIA DETALLE ON GUIA DETALLE REORGANIZE;
```

- Mayor a 30% → reconstruir:

```
--Mayor a 30% → reconstruir:
ALTER INDEX PK__PROVEEDO__BFBE6B07AEFC9CA5 ON PROVEEDOR REBUILD;
ALTER INDEX PK_GUIA_DETALLE ON GUIA_DETALLE REBUILD;
```

3. Administración de transacciones y bloqueos (locking)

SQL Server controla la concurrencia usando **bloqueos** (locks) para garantizar integridad.

Tipos de locks:

- **Shared (S):** lectura.
- **Exclusive (X):** escritura.
- **Update (U):** evita deadlocks.
- **Intent Locks:** a nivel de tabla/página.

Problemas comunes:

1. **Bloqueos largos (blocking):**
   Una transacción mantiene un lock por mucho tiempo → otras consultas se quedan esperando.

2. **Deadlocks:**
   Dos sesiones esperan recursos entre sí → SQL Server elige una víctima.

Soluciones:

- Asegurar transacciones cortas.
- Usar índices para reducir lecturas innecesarias.
- Usar hints como `READ COMMITTED SNAPSHOT`.
- Revisar queries que escanean tablas completas.

4. Análisis de planes de ejecución

El plan de ejecución muestra cómo SQL Server procesará una consulta.

Qué revisar:

- **Scans vs. Seeks**
  Seek = rápido
  Scan = lento
- **Operadores costosos:**
  - Sort
  - Hash Match
  - Key Lookup
- **Estimación vs. ejecución real**
  - Grandes diferencias → estadísticas malas.

¿Cómo obtenerlo?

```
--¿Cómo obtenerlo?
SET SHOWPLAN_ALL ON;
```

o desde SSMS → *Include Actual Execution Plan*.

5. Optimización de consultas T-SQL

Principios fundamentales:

✔ Evitar SELECT *

Carga innecesaria de columnas → más IO.

✔ Filtrar con WHERE adecuadamente

Más selectividad → mejores planes de ejecución.

✔ Evitar funciones sobre columnas indexadas

Mal ejemplo:

```
--Mal ejemplo:
WHERE YEAR(Fecha) = 2024
```

```
%
Messages
Msg 156, Level 15, State 1, Line 15
Incorrect syntax near the keyword 'WHERE'.
```

Esto rompe el índice.

Mejor:

```
--Mejor:
WHERE Fecha >= '2024-01-01' AND Fecha < '2025-01-01'
```

✔ Usar JOINs correctamente

Preferir joins explícitos:

```
INNER JOIN, LEFT JOIN
```

✔ Crear índices según la consulta

Especialmente sobre columnas:

- en WHERE
- en JOIN

- en ORDER BY

6. Control de recursos con Resource Governor

Permite controlar y limitar:

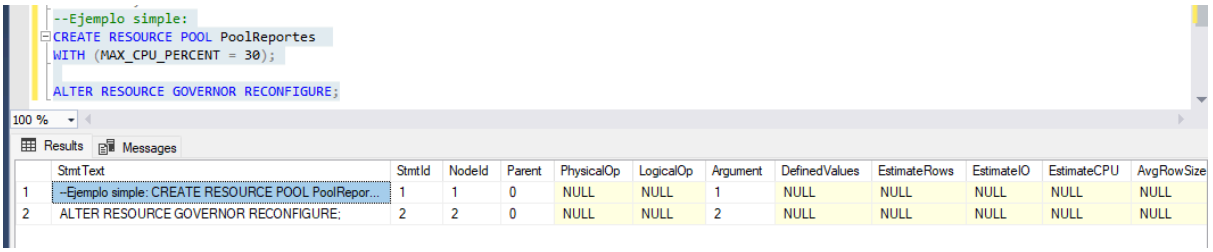
- CPU
- Memoria
- Threads

Entre diferentes grupos de carga.

¿Para qué sirve?

- Evitar que un usuario o proceso consuma toda la CPU.
- Proteger consultas críticas.
- Mantener estable el servidor en horas pico.

Ejemplo simple:



También se definen:

- Workload groups
- Clasificador (función que asigna cada sesión a un pool)

RESUMEN GENERAL

Tema	Qué permite mejorar
SQL Profiler / XE	Detectar problemas en tiempo real
Estadísticas	Planes de ejecución más eficientes
Índices	Velocidad de búsqueda
Manejo de transacciones	Concurrencia y evitar bloqueos
Planes de ejecución	Identificar consultas lentas
Optimización T-SQL	Reducir consumo de CPU e IO
Resource Governor	Controlar quién consume recursos

PRÁCTICA 1: Identificar consultas lentas y reescribirlas

1.1 Identificar consultas lentas con Extended Events

Extended Events es ideal porque consume pocos recursos.

Crear una sesión para capturar consultas lentas (> 1 segundo):

```
--PRACTICA 1
CREATE EVENT SESSION SlowQueries
ON SERVER
ADD EVENT sqlserver.rpc_completed(
    WHERE duration > 1000
),
ADD EVENT sqlserver.sql_batch_completed(
    WHERE duration > 1000
)
ADD TARGET package0.event_file(SET filename = 'C:\Temp\SlowQueries.xel');
GO

ALTER EVENT SESSION SlowQueries ON SERVER STATE = START;
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRowSiz
1	--PRACTICA 1 CREATE EVENT SESSION SlowQueries O...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRowS
1	ALTER EVENT SESSION SlowQueries ON SERVER STAT...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL

Luego se analiza con:

```
--Luego se analiza con:
SELECT * FROM sys.fn_xe_file_target_read_file('C:\Temp\SlowQueries*.xel', NULL, NULL, NULL);
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU
1	--Luego se analiza con: SELECT * FROM sys.fn_xe...	1	1	0	NULL	NULL	1	NULL	1000	NULL	NULL
2	|--Table-valued function	1	2	1	Table-valued function	Table-valued function	NULL	NULL	1000	0	0.00100015

1.2 Ejemplo de consulta lenta

Consulta original (ineficiente):

```
--Consulta original (ineficiente):
SELECT *
FROM ORDEN_COMPRA
WHERE YEAR(FechaOrden) = 2023;
```

100 %

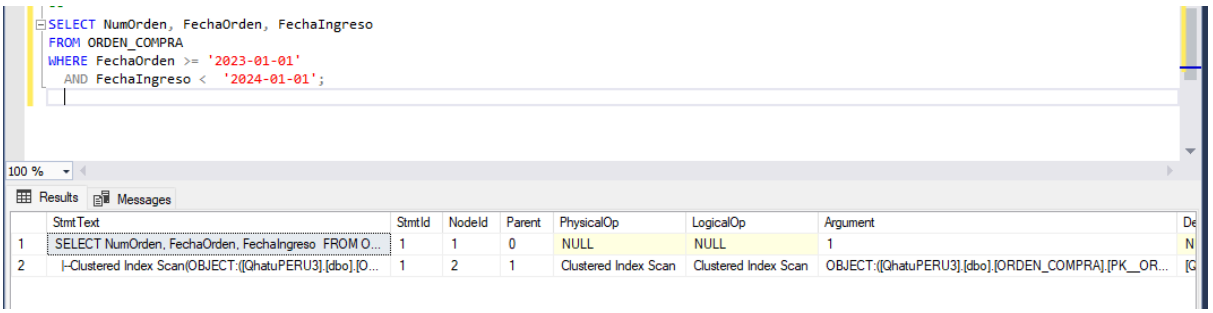
Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument
1	SELECT * FROM ORDEN_COMPRA WHERE YEAR(FechaOr...	1	1	0	NULL	NULL	1
2	|--Clustered Index Scan(OBJECT:([QhatuPERU3].[dbo].[ORDEN...	1	2	1	Clustered Index Scan	Clustered Index Scan	OBJECT:([QhatuPERU3].[dbo].[ORDEN_COMPRA].[PK__OR

Problemas:

- Usa `YEAR()` → rompe el índice.
- No especifica rango → obliga a *scan* completo.
- Selecciona todas las columnas.

1.3 Consulta reescrita y optimizada

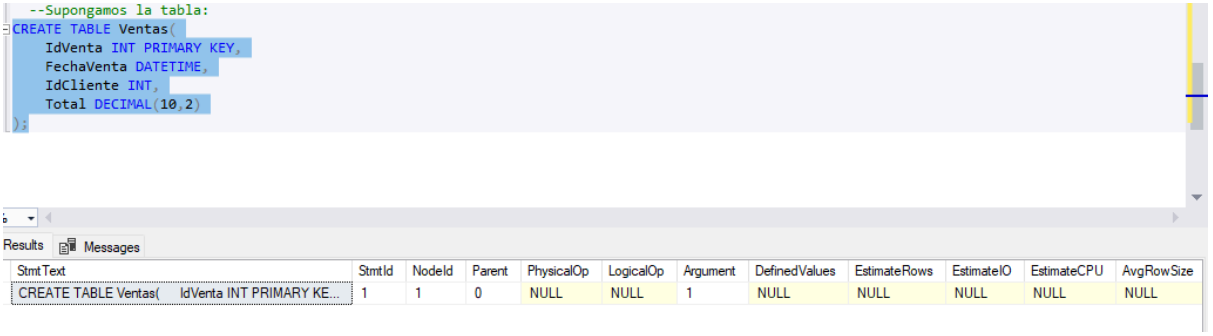


✔ Mejoras:

- Busca por rango → usa índices.
- Evita funciones sobre columnas.
- Reduce columnas → menos I/O.

PRÁCTICA 2: Crear y mantener índices para una tabla de ventas

Supongamos la tabla:



2.1 Crear índices recomendados

1) Índice por Fecha

Para reportes y filtros por rango:

```sql
--Para reportes y filtros por rango:
CREATE INDEX IX_Ventas_FechaVenta
ON Ventas(FechaVenta);
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRowSize	Total
1	--Para reportes y filtros por rango: CREATE INDE...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL	NUL

2) Índice por Cliente

Para análisis por cliente:

```sql
--Para análisis por cliente:
CREATE INDEX IX_Ventas_IdCliente
ON Ventas(IdCliente);
```

%

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRowSize	Tota
	--Para análisis por cliente: CREATE INDEX IX_Ve...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL	NU

2.2 Revisar la fragmentación de índices

```sql
--
SELECT
    DB_NAME() AS BD,
    OBJECT_NAME(object_id) AS Tabla,
    index_id,
    avg_fragmentation_in_percent
FROM sys.dm_db_index_physical_stats(DB_ID(), OBJECT_ID('Ventas'), NULL, NULL, 'DETAILED');
```

00 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValue
1	SELECT DB_NAME() AS BD, OBJECT_NAME(o...	1	1	0	NULL	NULL	1	NULL
2	|--Compute Scalar(DEFINE:([Expr1000]=db_name(), [E...	1	2	1	Compute Scalar	Compute Scalar	DEFINE:([Expr1000]=db_name(), [Expr1001]=object_...	[Expr1000]=d
3	|--Table-valued function	1	3	2	Table-valued function	Table-valued function	NULL	NULL

2.3 Mantenimiento de índices

Si la fragmentación es menor de 30% → REORGANIZE

```sql
--Si la fragmentación es menor de 30% → REORGANIZE
ALTER INDEX IX_Ventas_FechaVenta ON Ventas REORGANIZE;
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRowS
1	ALTER INDEX IX_Ventas_FechaVenta ON Ventas REORGA...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL

Si es mayor de 30% → REBUILD

```
--Si es mayor de 30% → REBUILD
ALTER INDEX IX_Ventas_FechaVenta ON Ventas REBUILD;
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRowSi
1	ALTER INDEX IX_Ventas_FechaVenta ON Ventas REBUIL...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL

Actualizar estadísticas (muy importante)

```
--Actualizar estadísticas (muy importante)
UPDATE STATISTICS Ventas;
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRowSize	TotalSubtreeCost	Outp
1	UPDATE STATISTICS Ventas;	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL	NULL	NU

PRÁCTICA 3: Configurar alertas de bloqueo y saturación de CPU

3.1 Alerta: Bloqueos prolongados (Blocking)

Crear alerta cuando una consulta queda bloqueada por más de 20 segundos:

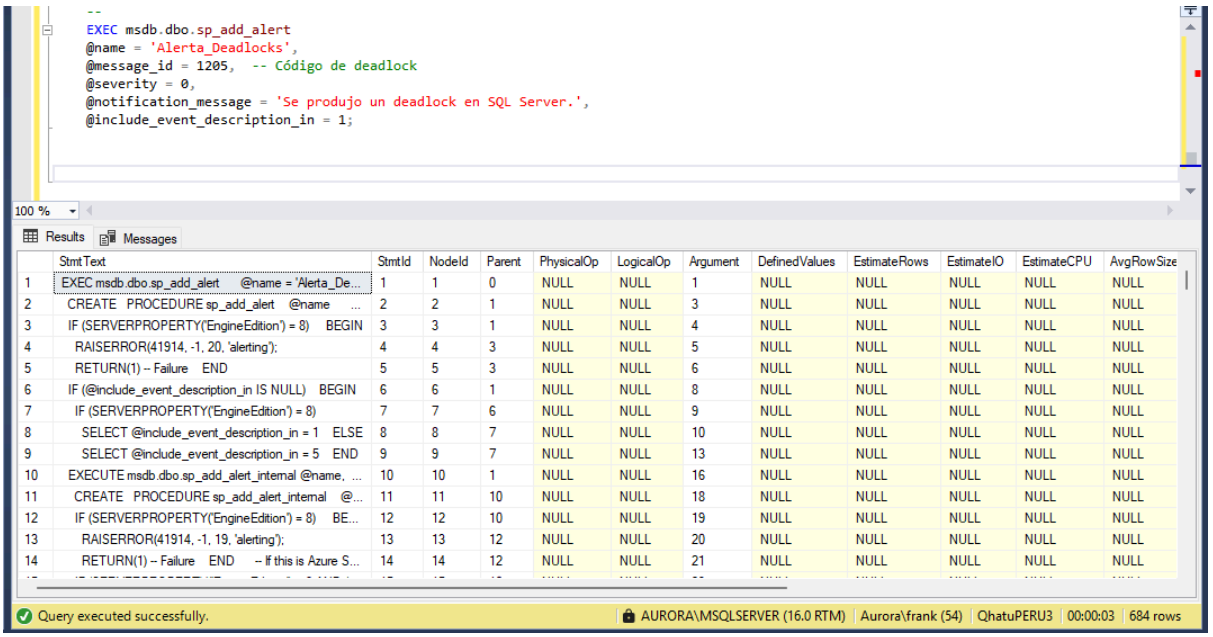
```
EXEC msdb.dbo.sp_add_alert
    @name = 'Alerta_Bloqueos',
    @message_id = 0,
    @severity = 0,
    @notification_message = 'Se detectó un bloqueo prolongado',
    @job_name = NULL,
    @performance_condition = 'Latches > 0',
    @delay_between_responses = 60,
    @include_event_description_in = 1;
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgR
1	EXEC msdb.dbo.sp_add_alert @name = 'Alerta_B...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL
2	CREATE PROCEDURE sp_add_alert @name ...	2	2	1	NULL	NULL	3	NULL	NULL	NULL	NULL	NULL
3	IF (SERVERPROPERTY('EngineEdition') = 8) BE...	3	3	1	NULL	NULL	4	NULL	NULL	NULL	NULL	NULL
4	RAISERROR(41914, -1, 20, 'alerting');	4	4	3	NULL	NULL	5	NULL	NULL	NULL	NULL	NULL
5	RETURN(1) -- Failure END	5	5	3	NULL	NULL	6	NULL	NULL	NULL	NULL	NULL
6	IF (@include_event_description_in IS NULL) BE...	6	6	1	NULL	NULL	8	NULL	NULL	NULL	NULL	NULL
7	IF (SERVERPROPERTY('EngineEdition') = 8)	7	7	6	NULL	NULL	9	NULL	NULL	NULL	NULL	NULL
8	SELECT @include_event_description_in = 1 E...	8	8	7	NULL	NULL	10	NULL	NULL	NULL	NULL	NULL
9	SELECT @include_event_description_in = 5 E...	9	9	7	NULL	NULL	13	NULL	NULL	NULL	NULL	NULL
10	EXECUTE msdb.dbo.sp_add_alert_internal @name...	10	10	1	NULL	NULL	16	NULL	NULL	NULL	NULL	NULL
11	CREATE PROCEDURE sp_add_alert_internal ...	11	11	10	NULL	NULL	18	NULL	NULL	NULL	NULL	NULL
12	IF (SERVERPROPERTY('EngineEdition') = 8) B...	12	12	10	NULL	NULL	19	NULL	NULL	NULL	NULL	NULL
13	RAISERROR(41914, -1, 19, 'alerting');	13	13	12	NULL	NULL	20	NULL	NULL	NULL	NULL	NULL

3.2 Crear alerta cuando SQL Server detecte DEADLOCKS



```
--
EXEC msdb.dbo.sp_add_alert
@name = 'Alerta_Deadlocks',
@message_id = 1205,  -- Código de deadlock
@severity = 0,
@notification_message = 'Se produjo un deadlock en SQL Server.',
@include_event_description_in = 1;
```

100 %

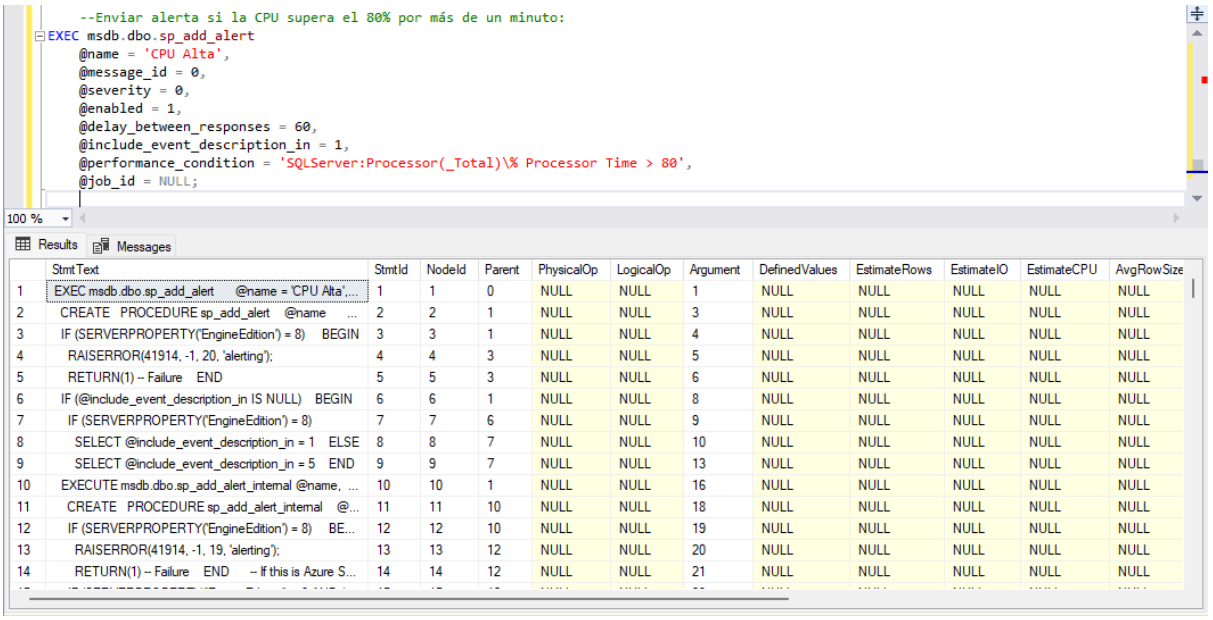
Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRowSize
1	EXEC msdb.dbo.sp_add_alert @name = 'Alerta_De...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL
2	CREATE PROCEDURE sp_add_alert @name ...	2	2	1	NULL	NULL	3	NULL	NULL	NULL	NULL	NULL
3	IF (SERVERPROPERTY('EngineEdition') = 8) BEGIN	3	3	1	NULL	NULL	4	NULL	NULL	NULL	NULL	NULL
4	RAISERROR(41914, -1, 20, 'alerting');	4	4	3	NULL	NULL	5	NULL	NULL	NULL	NULL	NULL
5	RETURN(1) -- Failure END	5	5	3	NULL	NULL	6	NULL	NULL	NULL	NULL	NULL
6	IF (@include_event_description_in IS NULL) BEGIN	6	6	1	NULL	NULL	8	NULL	NULL	NULL	NULL	NULL
7	IF (SERVERPROPERTY('EngineEdition') = 8)	7	7	6	NULL	NULL	9	NULL	NULL	NULL	NULL	NULL
8	SELECT @include_event_description_in = 1 ELSE	8	8	7	NULL	NULL	10	NULL	NULL	NULL	NULL	NULL
9	SELECT @include_event_description_in = 5 END	9	9	7	NULL	NULL	13	NULL	NULL	NULL	NULL	NULL
10	EXECUTE msdb.dbo.sp_add_alert_internal @name, ...	10	10	1	NULL	NULL	16	NULL	NULL	NULL	NULL	NULL
11	CREATE PROCEDURE sp_add_alert_internal @...	11	11	10	NULL	NULL	18	NULL	NULL	NULL	NULL	NULL
12	IF (SERVERPROPERTY('EngineEdition') = 8) BE...	12	12	10	NULL	NULL	19	NULL	NULL	NULL	NULL	NULL
13	RAISERROR(41914, -1, 19, 'alerting');	13	13	12	NULL	NULL	20	NULL	NULL	NULL	NULL	NULL
14	RETURN(1) -- Failure END -- If this is Azure S...	14	14	12	NULL	NULL	21	NULL	NULL	NULL	NULL	NULL

Query executed successfully. | AURORA\MSQLSERVER (16.0 RTM) | Aurora\frank (54) | QhatuPERU3 | 00:00:03 | 684 rows

3.3 Alerta: Saturación de CPU

Enviar alerta si la CPU supera el 80% por más de un minuto:



```
--Enviar alerta si la CPU supera el 80% por más de un minuto:
EXEC msdb.dbo.sp_add_alert
    @name = 'CPU Alta',
    @message_id = 0,
    @severity = 0,
    @enabled = 1,
    @delay_between_responses = 60,
    @include_event_description_in = 1,
    @performance_condition = 'SQLServer:Processor(_Total)\% Processor Time > 80',
    @job_id = NULL;
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRowSize
1	EXEC msdb.dbo.sp_add_alert @name = 'CPU Alta',...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL
2	CREATE PROCEDURE sp_add_alert @name ...	2	2	1	NULL	NULL	3	NULL	NULL	NULL	NULL	NULL
3	IF (SERVERPROPERTY('EngineEdition') = 8) BEGIN	3	3	1	NULL	NULL	4	NULL	NULL	NULL	NULL	NULL
4	RAISERROR(41914, -1, 20, 'alerting');	4	4	3	NULL	NULL	5	NULL	NULL	NULL	NULL	NULL
5	RETURN(1) -- Failure END	5	5	3	NULL	NULL	6	NULL	NULL	NULL	NULL	NULL
6	IF (@include_event_description_in IS NULL) BEGIN	6	6	1	NULL	NULL	8	NULL	NULL	NULL	NULL	NULL
7	IF (SERVERPROPERTY('EngineEdition') = 8)	7	7	6	NULL	NULL	9	NULL	NULL	NULL	NULL	NULL
8	SELECT @include_event_description_in = 1 ELSE	8	8	7	NULL	NULL	10	NULL	NULL	NULL	NULL	NULL
9	SELECT @include_event_description_in = 5 END	9	9	7	NULL	NULL	13	NULL	NULL	NULL	NULL	NULL
10	EXECUTE msdb.dbo.sp_add_alert_internal @name, ...	10	10	1	NULL	NULL	16	NULL	NULL	NULL	NULL	NULL
11	CREATE PROCEDURE sp_add_alert_internal @...	11	11	10	NULL	NULL	18	NULL	NULL	NULL	NULL	NULL
12	IF (SERVERPROPERTY('EngineEdition') = 8) BE...	12	12	10	NULL	NULL	19	NULL	NULL	NULL	NULL	NULL
13	RAISERROR(41914, -1, 19, 'alerting');	13	13	12	NULL	NULL	20	NULL	NULL	NULL	NULL	NULL
14	RETURN(1) -- Failure END -- If this is Azure S...	14	14	12	NULL	NULL	21	NULL	NULL	NULL	NULL	NULL

3.4 Asignar destinatarios de alertas (correo o operador)

Definir un operador:

```
--
EXEC msdb.dbo.sp_add_operator
@name = 'AdminDB',
@email_address = 'admin@empresa.com';
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRow
1	EXEC msdb.dbo.sp_add_operator @name = 'AdminDB...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL
2	CREATE PROCEDURE sp_add_operator @name ...	2	2	1	NULL	NULL	3	NULL	NULL	NULL	NULL	NULL
3	SELECT @name = LTRIM(RTRIM(@name))	3	3	1	NULL	NULL	4	NULL	NULL	NULL	NULL	NULL
4	SELECT @email_address = LTRIM(RTRIM(@email_addr...	4	4	1	NULL	NULL	5	NULL	NULL	NULL	NULL	NULL
5	SELECT @pager_address = LTRIM(RTRIM(@pager_ad...	5	5	1	NULL	NULL	6	NULL	NULL	NULL	NULL	NULL
6	SELECT @netsend_address = LTRIM(RTRIM(@netsend...	6	6	1	NULL	NULL	7	NULL	NULL	NULL	NULL	NULL
7	SELECT @category_name = LTRIM(RTRIM(@category...	7	7	1	NULL	NULL	8	NULL	NULL	NULL	NULL	NULL
8	IF (@netsend_address <> N'' AND SERVERPROPERTY('...	8	8	1	NULL	NULL	9	NULL	NULL	NULL	NULL	NULL
9	RAISERROR(41914, -1, 12, 'NetSend')	9	9	8	NULL	NULL	10	NULL	NULL	NULL	NULL	NULL
10	RETURN(1) -- Failure END -- Turn [nullable] empty s...	10	10	8	NULL	NULL	11	NULL	NULL	NULL	NULL	NULL
11	IF (@email_address = N'')	11	11	1	NULL	NULL	13	NULL	NULL	NULL	NULL	NULL
12	SELECT @email_address = NULL	12	12	11	NULL	NULL	14	NULL	NULL	NULL	NULL	NULL
13	IF (@pager_address = N'')	13	13	1	NULL	NULL	16	NULL	NULL	NULL	NULL	NULL
14	SELECT @pager_address = NULL	14	14	13	NULL	NULL	17	NULL	NULL	NULL	NULL	NULL

Asociarlo con una alerta:

```
EXEC msdb.dbo.sp_add_notification
@alert_name = 'CPU Alta',
@operator_name = 'AdminDB',
@notification_method = 1;  -- Email
```

100 %

Results | Messages

	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp	Argument	DefinedValues	EstimateRows	EstimateIO	EstimateCPU	AvgRow
1	EXEC msdb.dbo.sp_add_notification @alert_name = '...	1	1	0	NULL	NULL	1	NULL	NULL	NULL	NULL	NULL
2	CREATE PROCEDURE sp_add_notification @alert_n...	2	2	1	NULL	NULL	3	NULL	NULL	NULL	NULL	NULL
3	IF ((@notification_method & 4 = 4) AND SERVERPROPE...	3	3	1	NULL	NULL	4	NULL	NULL	NULL	NULL	NULL
4	RAISERROR(41914, -1, 15, 'NetSend')	4	4	3	NULL	NULL	5	NULL	NULL	NULL	NULL	NULL
5	RETURN(1) -- Failure END	5	5	3	NULL	NULL	6	NULL	NULL	NULL	NULL	NULL
6	SELECT @res_notification = FORMATMESSAGE(14210)...	6	6	1	NULL	NULL	8	NULL	NULL	NULL	NULL	NULL
7	SELECT @alert_name = LTRIM(RTRIM(@alert_name))	7	7	1	NULL	NULL	9	NULL	NULL	NULL	NULL	NULL
8	SELECT @operator_name = LTRIM(RTRIM(@operator_...	8	8	1	NULL	NULL	10	NULL	NULL	NULL	NULL	NULL
9	IF ((ISNULL(IS_SRVROLEMEMBER(N'sysadmin'), 0) <> 1...	9	9	1	NULL	NULL	11	NULL	NULL	NULL	NULL	NULL
10	RAISERROR(15003, 16, 1, N'sysadmin')	10	10	9	NULL	NULL	12	NULL	NULL	NULL	NULL	NULL
11	RETURN(1) -- Failure END -- Check if the Notificati...	11	11	9	NULL	NULL	13	NULL	NULL	NULL	NULL	NULL
12	EXECUTE @retval = msdb.dbo.sp_verify_notification @al...	12	12	1	NULL	NULL	15	NULL	NULL	NULL	NULL	NULL
13	CREATE PROCEDURE sp_verify_notification @alert...	13	13	12	NULL	NULL	33	NULL	NULL	NULL	NULL	NULL
14	IF ((@notification_method & 4 = 4) AND SERVERPROP...	14	14	12	NULL	NULL	34	NULL	NULL	NULL	NULL	NULL