Activity Monitor

Según el Activity Monitor en SSMS las conexiones sin pool (tedious), se abren y se cierran tan rápido que ni siquiera aparecen en la interfaz, pero si se puede notar su influencia en el gráfico de batch. Para poder notarlo tuvimos que poner una función de sleep, para que aparezca la conexión antes de que se cierre.

De 5 a 10 Pools con 10 threads

Pool: Se logra ver que por la cantidad de threads decide hacer otro pool para la demanda.

No Pool: No se logran ver las conexiones ya que se ejecuta muy rápido.

A screenshot of a computer

Description automatically generated

De 5 a 10 Pools con 20 threads

Pool: Se logra ver que por la cantidad de threads decide usar 4 pools y el batch request escala el doble.

No Pool: No se logran ver las conexiones ya que se ejecuta muy rápido.

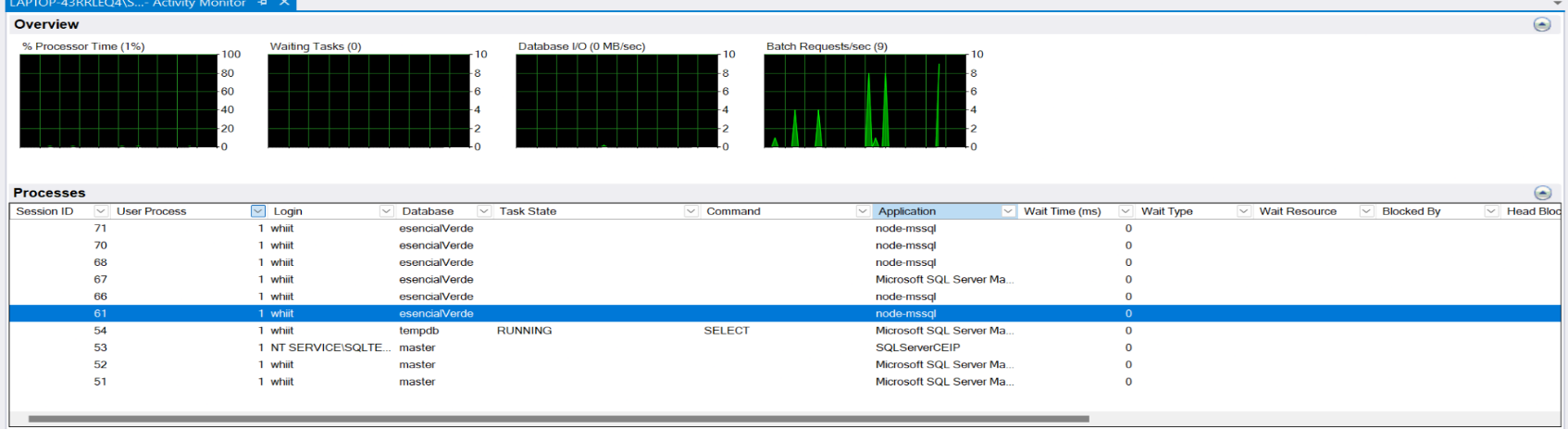
A screenshot of a computer

Description automatically generated

De 5 a 20 Pools con 20 threads

Pool: Se logra ver que decide abrir mas pools tomando un total de 5

No Pool: No se logran ver las conexiones ya que se ejecuta muy rápido

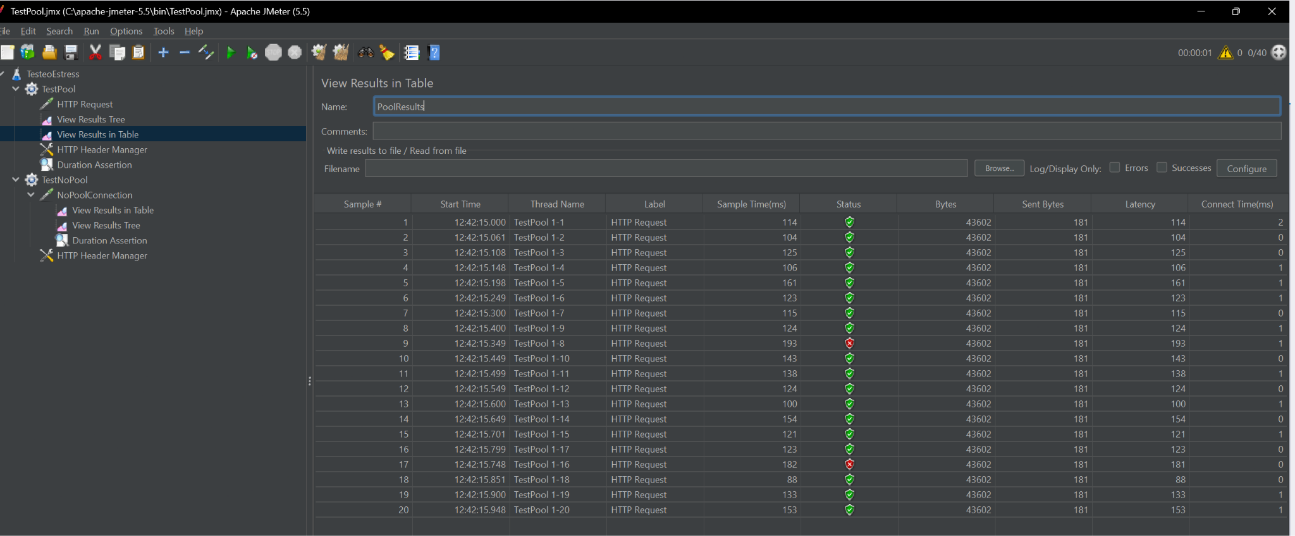


Prueba de sample time

Sample time restriction = 180

Threads = 20

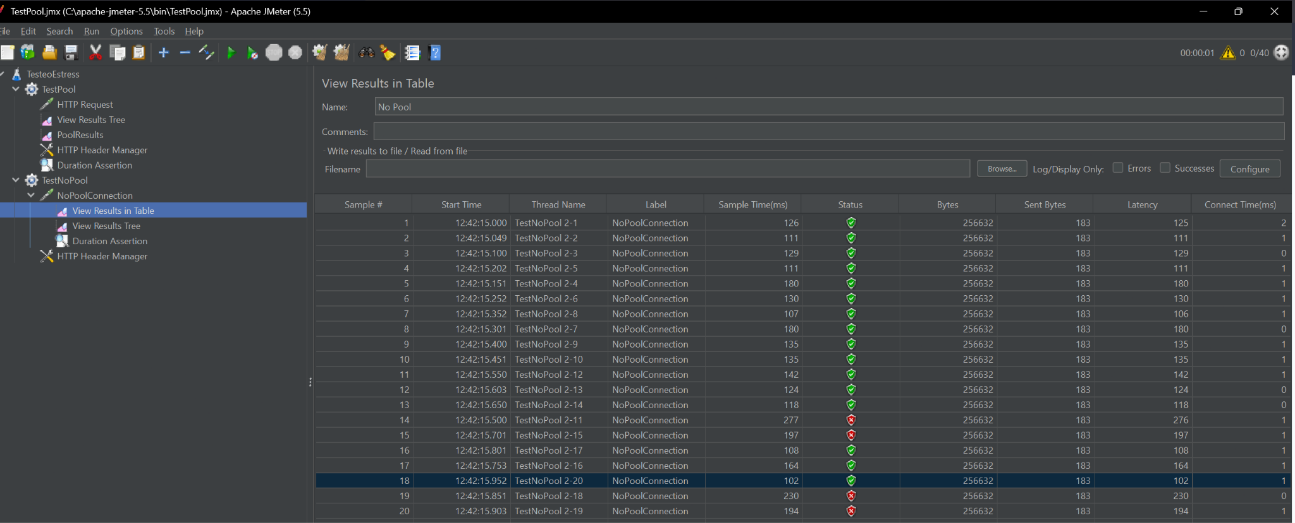
Pool: No se pudieron retirar 2



Sample time restriction = 180

Threads = 20

NoPool: No se pudieron retirar 4



Prueba de sample time

Probamos hacer 20 treads y 10 loops para solo una conexión con pool donde su average sample time es 290.

A screenshot of a computer

Description automatically generated with medium confidence

Probamos hacer 20 threads y 10 loops para solo una conexión sin pool donde su average sample time es 335.

A screenshot of a computer

Description automatically generated with medium confidence

Prueba de alta concurrencia

Probamos hacer 20 treads y 100 loops para solo una conexión con pool donde su average sample time es 348 y 36 segundos para los 2000 registros.

**A screenshot of a computer

Description automatically generated with medium confidence**

Probamos hacer 20 treads y 100 loops para solo una conexión sin pool donde su average sample time es 499 y 52 segundos para los 2000 registros.

**A screenshot of a computer

Description automatically generated with medium confidence**

Diferencias y Ventajas

Conexión a una DB con pool:

En este enfoque, se crea un "pool" o grupo de conexiones de base de datos preestablecidas y se administran de manera centralizada. Cuando una aplicación necesita acceder a la base de datos, solicita una conexión del pool. Una vez que se ha utilizado la conexión, se devuelve al pool en lugar de cerrarse. Al tener conexiones preestablecidas listas para usar, se evita el costo de establecer una nueva conexión cada vez que se necesita acceder a la base de datos.

**Ventajas del pool de conexiones:**

* Rendimiento mejorado: Al tener conexiones preestablecidas en el pool, se evita la sobrecarga de establecer y cerrar conexiones repetidamente. Esto puede mejorar significativamente el rendimiento de la aplicación, especialmente en escenarios con muchas solicitudes a la base de datos.
* Gestión eficiente de conexiones: El pool administra automáticamente las conexiones y las distribuye entre las solicitudes de la aplicación. Esto garantiza un uso óptimo de los recursos de la base de datos y evita la sobrecarga de conexiones simultáneas. Por ejemplo analiza las solicitudes y decide cuantos pools pueden ser necesarios para realizar las conexiones.
* Control de concurrencia: El pool de conexiones puede incluir mecanismos para controlar la concurrencia y garantizar que varias solicitudes se ejecuten de manera segura en paralelo.

**Desventajas del pool de conexiones:**

* Uso de memoria: El pool de conexiones requiere almacenar conexiones abiertas en memoria, lo que puede consumir recursos significativos, especialmente si el tamaño del pool es grande. Esto puede afectar el rendimiento general del sistema.
* Configuración y mantenimiento: Configurar y ajustar correctamente el pool de conexiones puede ser un desafío. Se deben considerar factores como el tamaño óptimo del pool, el tiempo de espera de las conexiones inactivas y la cantidad máxima de conexiones permitidas.

**Conexión a una DB sin pool:**

En este enfoque, cada vez que una aplicación necesita acceder a la base de datos, se establece una nueva conexión y se cierra después de su uso. No se utiliza ningún pool para gestionar las conexiones.

**Ventajas de la conexión sin pool:**

* Menor uso de memoria: No se requiere mantener un pool de conexiones en memoria, lo que puede ser beneficioso en escenarios donde los recursos son limitados.
* Simplicidad: No se necesita configurar ni mantener un pool de conexiones, lo que puede hacer que este enfoque sea más fácil de mantener.

**Desventajas de la conexión sin pool:**

* Rendimiento inferior: Establecer y cerrar conexiones repetidamente puede tener un impacto negativo en el rendimiento de la aplicación, especialmente si hay muchas solicitudes a la base de datos.
* Mayor latencia: El tiempo necesario para establecer una nueva conexión puede aumentar la latencia general de la aplicación, ya que se agrega una sobrecarga adicional en cada solicitud de la base de datos.

Conclusión:

En resumen, el uso de un pool de conexiones puede mejorar el rendimiento y la eficiencia en escenarios donde se requiere un acceso frecuente a la base de datos. Sin embargo, si el acceso a la base de datos es ocasional o no hay restricciones de recursos, utilizar conexiones sin pool puede simplificar la implementación y el mantenimiento del sistema

La elección entre usar una conexión a una base de datos con pool o sin pool depende de las necesidades y restricciones específicas de tu aplicación. Algunos factores a considerar incluyen el volumen de solicitudes a la base de datos, la carga de trabajo esperada, los recursos disponibles y los requisitos de rendimiento.

En general, si tu aplicación maneja una gran cantidad de solicitudes concurrentes a la base de datos y el rendimiento es una prioridad, un pool de conexiones puede ser beneficioso. Proporciona una gestión eficiente de conexiones y puede mejorar el tiempo de respuesta y la escalabilidad de la aplicación. Sin embargo, debes tener en cuenta el consumo de memoria y los aspectos de configuración y mantenimiento asociados.

Por otro lado, si tu aplicación tiene una carga de trabajo más baja o recursos limitados, el uso de conexiones sin pool puede ser una opción más simple. Aunque puede haber un impacto en el rendimiento debido a la sobrecarga de establecer y cerrar conexiones repetidamente, puede ser adecuado en escenarios donde el acceso a la base de datos es ocasional y la latencia no es crítica.

En última instancia, la elección entre una conexión a una base de datos con pool o sin pool debe basarse en un análisis de las necesidades específicas de tu aplicación, considerando factores como el rendimiento, la eficiencia, los recursos disponibles y la simplicidad de implementación y mantenimiento.