Desarrollo De Un Sistema De Balanceo De Carga Utilizando MySQL Y MySQL Router Para Mejorar El Rendimiento De Aplicaciones Distribuidas

Angie Yulieth Feligrana 2190170

angie.feligrana@uao.edu.co

Jose Daniel Cruz Perdomo 2201855  
jose\_daniel.cruz@uao.Edu.co

Victor Manuel Gomez 2171253  
victor\_manuel.gomez@uao.edu.co

Juan Camilo Ceron de Jesus 2201856  
juan\_camilo.ceron@uao.edu.co

Abstract

In this project, load balancing was developed using MSQL and MQGLRouter. Through exhaustive tests performed with Sysbench on the master machine, the performance of the system was evaluated in different workload scenarios. The main objective was to achieve an equitable load distribution among the database nodes, ensuring optimal performance and high availability for end users. The project was notable for its focus on scalability, fault tolerance and performance optimization, ensuring a smooth and efficient experience for users of the database system.

Resumen

En este projecto se desarrolo el balance de cargas por medio de MSQL y MQGLRouter. Mediante pruebas exhaustivas realizadas con Sysbench en la máquina maestra, se evaluó el rendimiento del sistema en distintos escenarios de carga de trabajo. El objetivo principal era lograr una distribución equitativa de la carga entre los nodos de la base de datos, garantizando un rendimiento óptimo y una alta disponibilidad para los usuarios finales. El proyecto destacó por centrarse en la escalabilidad, la tolerancia a fallos y la optimización del rendimiento, garantizando así una experiencia fluida y eficiente a los usuarios del sistema de base de datos.

Contexto

El balanceo de carga es un mecanismo esencial para optimizar el rendimiento y la capacidad de los servicios web en entornos de alta demanda y tráfico intenso. En la era digital actual, donde las empresas dependen cada vez más de sus aplicaciones y servicios en línea, garantizar un rendimiento óptimo y una disponibilidad constante es fundamental para satisfacer las expectativas de los usuarios y asegurar la satisfacción del cliente.Este proceso de balanceo de carga se realiza mediante el uso de un balanceador, ya sea en forma de software o hardware especializado, que sirve como intermediario entre los clientes y los servidores que alojan los servicios web. Su función principal es distribuir de manera equitativa y eficiente la carga de trabajo entrante entre varios servidores, evitando así que un único dispositivo se sobrecargue o se convierta en un punto de congestión.El objetivo primordial del balanceo de carga es mejorar la capacidad y la escalabilidad de un servicio web, lo que permite a más usuarios acceder al servicio simultáneamente y reduce los tiempos de respuesta, contribuyendo a una experiencia más fluida para los usuarios.

Descripción del Problema

Conforme una empresa o plataforma experimenta un aumento notable en la cantidad de usuarios y transacciones, su infraestructura de bases de datos que opera en un único servidor enfrenta desafíos significativos. El aumento en la carga de trabajo resulta en tiempos de respuesta más prolongados y una experiencia insatisfactoria para los usuarios. Los recursos limitados del servidor no pueden manejar eficazmente el crecimiento en la demanda, lo que impacta adversamente la capacidad de la empresa para proporcionar un servicio rápido y fiable. Además, la carencia de redundancia y la falta de tolerancia a fallos pueden dar lugar a interrupciones y pérdida de datos en caso de una falla en el servidor único. Resulta evidente que se necesita una solución que equilibre la carga de manera justa, mejore el rendimiento y la disponibilidad, y permita una escalabilidad adecuada para satisfacer las cambiantes necesidades de la empresa.

Alternativas de solución

**Galera Cluster:** Galera Cluster es una solución de bases de datos en clúster que permite la replicación síncrona entre varios nodos, lo que significa que los datos se mantienen consistentes en todos los nodos en tiempo real. Uno de los aspectos destacados de Galera Cluster es su capacidad de realizar balanceo de carga a nivel de bases de datos sin necesidad de un balanceador de carga externo. Esto se logra mediante un componente llamado "arbitrator" o árbitro, que toma decisiones sobre cómo distribuir solicitudes de lectura entre los nodos del clúster. El árbitro utiliza un algoritmo interno para determinar qué nodo debe manejar cada solicitud de lectura.

**Percona XtraDB Cluster:** Percona XtraDB Cluster es una solución de clustering de bases de datos que se basa en el motor de almacenamiento Percona XtraDB, que es una variante mejorada del motor InnoDB de MySQL. Percona XtraDB Cluster proporciona replicación síncrona multi-master, lo que significa que varios nodos pueden aceptar escrituras al mismo tiempo. También ofrece capacidades de balanceo de carga al distribuir consultas entre los nodos del clúster. Esto mejora la disponibilidad y el rendimiento al permitir que las consultas se distribuyan de manera eficiente.

**MaxScale:** MaxScale es una plataforma de enrutamiento y balanceo de carga desarrollada por MariaDB Corporation. Es compatible tanto con MariaDB como con MySQL y ofrece una variedad de funciones para el balanceo de carga. Estas funciones incluyen enrutamiento basado en reglas, caché de consultas, filtrado de consultas y gestión de alta disponibilidad y conmutación por error. MaxScale se utiliza para distribuir y administrar las consultas entrantes a los servidores de bases de datos de manera eficiente, asegurando un rendimiento óptimo y una alta disponibilidad.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Característica | Galera Cluster | MaxScale |  | Percona XtraDB Cluster |
| Tipo de software | Solución de clustering de bases de datos | Plataforma de enrutamiento y balanceo de carga |  | Solución de clustering de bases de datos |
| Motor de almacenamiento | InnoDB (MySQL) | InnoDB (MySQL o MariaDB) |  | Percona XtraDB (MySQL o MariaDB) |
| Tipo de replicación | Síncrona | Síncrona o asíncrona |  | Síncrona multi-master |
| Balanceo de carga | Sí, a nivel de bases de datos | Sí, a nivel de consultas |  | Sí, a nivel de bases de datos |
| Alta disponibilidad | Sí | Sí |  | Sí |
| Seguridad | Buena | Buena |  | Buena |
| Compatibilidad | MySQL | MariaDB o MySQL |  | MariaDB o MySQL |
| Dificultad de configuración | Media | Fácil |  | Media |
| Coste | Gratuito | Gratuito |  | Gratuito |

¿Cuál será nuestra solución? 

Una de las posibles soluciones era el “Group Replication”, una característica que proporciona el mismo MySQL con el nombre de MySQL InnoDB Cluster (versiones 5.7.17. en adelante). Este consta de componentes esenciales para su funcionamiento 

**1.MySQL Shell:**

**Función**: MySQL Shell es la interfaz de línea de comandos que te permite administrar, configurar y monitorear el clúster de bases de datos.

**Tareas típicas:** Puedes utilizar MySQL Shell para configurar el clúster, agregar o quitar nodos, realizar copias de seguridad, configurar políticas de alta disponibilidad y más.

**Interfaz de programación (API)**: MySQL Shell también ofrece una interfaz de programación para la automatización de tareas de administración.

**2.MySQL Router:**

**Función:** El Router es un componente esencial para la distribución del tráfico. Se asegura de que las solicitudes de los clientes se enruten al nodo adecuado en el clúster.

**Balanceo de carga:** Distribuye de manera equitativa las solicitudes de lectura y escritura a las diferentes máquinas del clúster, mejorando el rendimiento.

**Alta disponibilidad:** En caso de que uno de los nodos falle, el Router redirigirá automáticamente las solicitudes a nodos operativos, minimizando el tiempo de inactividad.

**3.MySQL Servers:**

Función: Estos servidores son donde se almacenan los datos. En un clúster de MySQL InnoDB, hay dos tipos de nodos:

**Servidor Maestro:** Tiene el poder de lectura y escritura. Aquí se realizan todas las operaciones de escritura. En caso de un fallo en el maestro, uno de los servidores secundarios puede ser promovido automáticamente para asumir ese papel.

**Instancias Secundarias:** Estas instancias solo tienen el poder de lectura. Sirven solicitudes de consulta y ayudan a distribuir la carga de trabajo.

**Replicación Sincrónica:** Los nodos secundarios mantienen una réplica sincrónica de los datos del nodo maestro, lo que garantiza que los datos estén siempre actualizados.

**Detección de Fallos y Tolerancia a Fallos:** El sistema es capaz de detectar fallos en los nodos y reorganizar automáticamente el clúster para mantener la disponibilidad de datos.

**Alta Disponibilidad:**

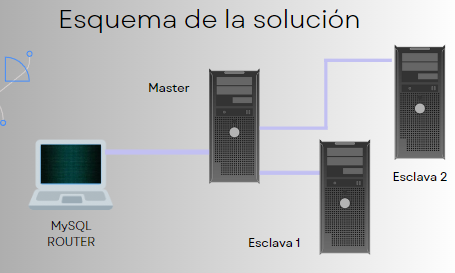
El uso de MySQL InnoDB Cluster con Group Replication permite lograr alta disponibilidad, ya que los datos están distribuidos entre varios nodos, lo que reduce el riesgo de pérdida de datos debido a fallos de hardware o software.

**Escalabilidad y Rendimiento:**

La distribución de carga de trabajo entre nodos secundarios mejora el rendimiento de las consultas, ya que el servidor maestro no se sobrecarga con solicitudes de lectura.

**Mantenimiento y Administración Simples:**

MySQL Shell facilita la administración y el monitoreo del clúster, lo que simplifica tareas como la copia de seguridad y la escalabilidad.

Esquema de solución:  


1. Maestro:Desempeña un papel central en la administración y coordinación de una red, asegurando un funcionamiento eficiente y seguro de los dispositivos conectados y los servicios ofrecidos a través de esa red.

2. MySQL Router: Es el encargado de enrutar las peticiones y de distribuirlas a lo largo del clúster de datos.

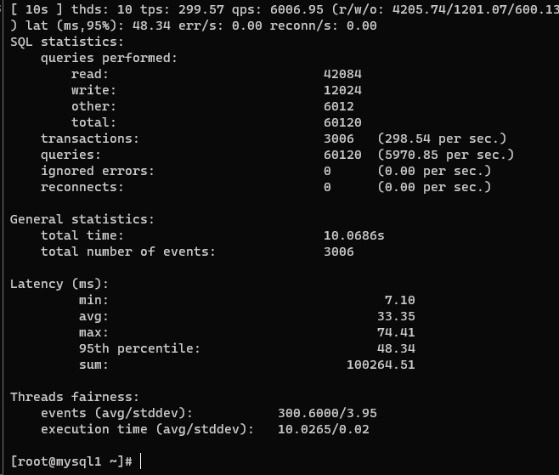
3. Maquinas esclavos:Son componentes subordinados o periféricos controlados por un sistema maestro o controlador en un sistema telemático o automatizado.

Pruebas

La primera prueba se hace en la maquina donde está el Docker, para observar la tabla estudiantes de la base de datos “UAO”, para observarla se digita un ciclo while, siempre se mostrará la tabla hasta que se interrumpan las 3 máquinas al mismo tiempo(Mysqlrouter, esclavo1 y esclavo 2), en caso tal de que la maquina “master” se caiga por diferentes motivos, una de las dos máquinas quedaría como la master (primaria), esto con el fin de descentralizar el proceso, para que no suceda que si se cae la “master” se caigan las demás máquinas esclavas. Siempre va a estar mostrando la consulta a pesar de que ocurra un error repentino con la “master". Este es el comando el cual se utilizó: while [ 1 ] do sleep 1 docker exec -it mysql1 mysql -h 192.168.50.2 -P 6446 -uinnodbcluster -pcontraseña\_fuerte1! -e "SELECT \* FROM UAO.estudiantes;" done

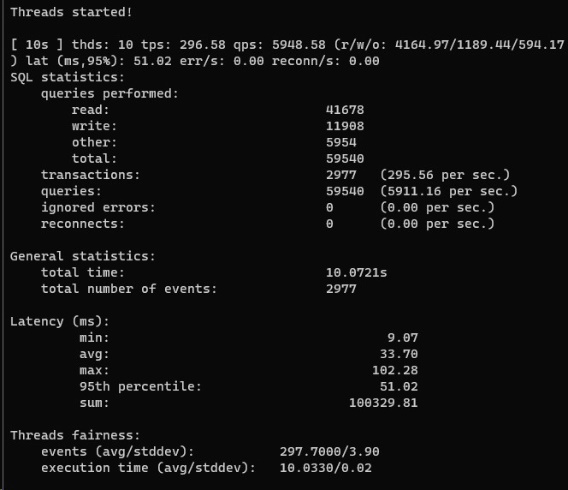
Lectura queries maquina 1.

Podemos evidenciar que el tiempo de ejecución es uno de los mas bajos ya que solo se esta utilizando una máquina, pero al mismo tiempo el número de eventos que puede realizar es bajo.



Lectura queries con dos maquinas

Podemos evidenciar que el tiempo de ejecución es uno de los mas bajos ya que solo se esta utilizando una máquina, y su numero de eventos de eventos es el mas bajo ya que es el que tiene menos capacidad de generar eventos.



Lectura queries con 3 maquinas

Texto

Descripción generada automáticamente

Una vez hecho las pruebas de disponibilidad de las maquinas, se realizaron las respectivas pruebas de estrés para percatarse de un posible error por n número de inserciones al mismo tiempo. Se utilizó sysbench con 10 hilos esto se refiere en palabras cotidianas que 10 personas están ejecutando sentencias sql de inserción al mismo tiempo.

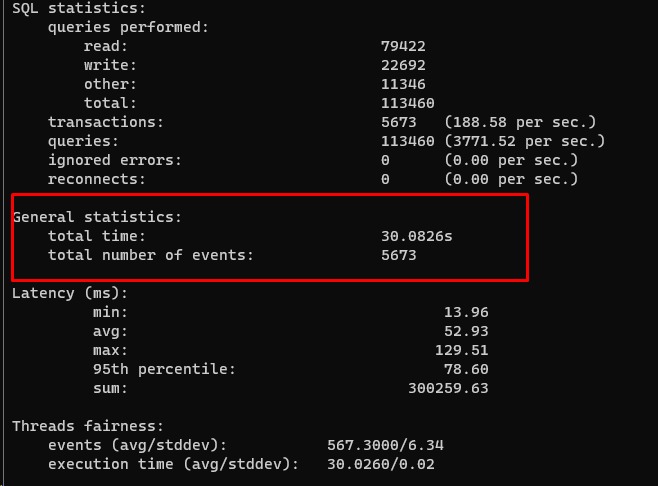
sysbench oltp\_write\_only --tables=10 --table-size=100 --db-driver=mysql --mysql-host=localhost --mysql-db=UAO --mysql-user=root --mysql-port=6446 --time=120 --threads=10 prepare.

Al realizar el Select de las tablas, nos generó 10 tablas:

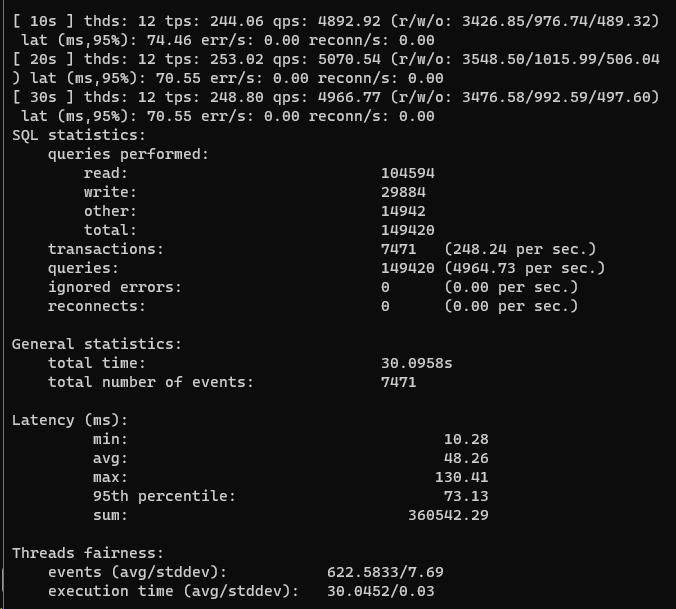
Texto

Descripción generada automáticamente

Prueba de rendimiento con una sola maquina, al apagar una de las maquinas podemos evidenciar que el proceso de lectura se vuelva mas lento y el tiempo de ejecución aumenta



Por otro lado se realizaron pruebas de lectura ejecutando las 3 maquinas la cual se evidencia el aumento de numero de eventos y un tiempo casi igual al manejar dos maquinas, la diferencia es que puede abarcar mas eventos, optimizando el rendimiento siendo eficiente.



Conclusión

Esta implementación permite distribuir la carga de trabajo de manera equitativa entre múltiples servidores de bases de datos, lo que mejora el rendimiento general del sistema al evitar la sobrecarga en un solo servidor. Al implementar esta solución, se logra una mayor disponibilidad y tolerancia a fallos gracias a la configuración de replicación y la redundancia de los datos. Además, al distribuir la carga entre múltiples servidores, se mejora la escalabilidad del sistema, ya que se puede agregar capacidad adicional según sea necesario para manejar el crecimiento futuro de la empresa. Este proyecto de servicios telemáticos nos ha brindado una valiosa experiencia práctica en la implementación de soluciones de balanceo de carga utilizando MySQL y MySQL Router. Hemos fortalecido nuestras habilidades en comunicación, coordinación y gestión de proyectos, mientras adquirimos conocimientos técnicos sobre configuración de servidores de base de datos y distribución equitativa de solicitudes.

Referencias

[1] Byte. "Balanceo de carga". Redes Teleco. [En línea]. Disponible en: https://redesteleco.com/balanceo\_de\_Carga Accedido el 13 de mayo de 2023.

[2] MySQL. "InnoDB Cluster". MySQL. [En línea]. Disponible en: https://dev.mysql.com/doc/refman/8.0/en/mysql-innodb-cluster-introduction.html Accedido el 13 de mayo de 2023.

[3] MySQL. "MySQL router 8.0". MySQL. [En línea]. Disponible en: https://dev.mysql.com/doc/mysql-router/8.0/en/ Accedido el 13 de mayo de 2023.

[4] Tilsor S.A. "Tilsor S.A.". [En línea]. Disponible en: https://www.tilsor.com.uy/index.php/blog/72-mysql Accedido el 13 de mayo de 2023.

[5] Cluster de balanceo de carga. (s.f.). [En línea]. Disponible en: https://enciclopedia\_universal.es-academic.com/43711/Cluster\_de\_balanceo\_de\_carga Accedido el 17 de mayo de 2023.

[6] Percona. (2019, 12 de septiembre). How to Build Percona XtraDB Cluster from Sources. Percona. [En línea]. Disponible en: https://www.percona.com/blog/how-to-build-percona-xtradb-cluster-from-sources/ Accedido el 17 de mayo de 2023.

[7] MariaDB MaxScale - MariaDB Knowledge Base. (n.d.). [En línea]. Disponible en: https://mariadb.com/kb/en/maxscale/ Accedido el 17 de mayo de 2023.