Engineer

Escalabilidad

📈 Hasta dónde basta con añadir máquinas para asumir más carga

Qué es

La escalabilidad describe cómo crece la capacidad de un sistema al añadirle recursos. Un sistema escala bien cuando el doble de máquinas soporta aproximadamente el doble de carga con los mismos tiempos de respuesta. Escala mal cuando el hardware adicional pasa la mayor parte del tiempo esperando a que se resuelva un cuello de botella que ese hardware no puede aliviar. La palabra describe una relación entre carga y coste, no una característica que un sistema tenga o no tenga.

Se puede crecer en dos direcciones. El escalado vertical consiste en una máquina más grande: más núcleos, más memoria y discos más rápidos. No exige cambiar el software y se detiene en la máquina más grande que merezca la pena comprar. El escalado horizontal consiste en más máquinas que se reparten el trabajo. No tiene ese techo, pero solo beneficia al trabajo que se puede dividir.

HTTP se diseñó pensando en el escalado horizontal. RFC 9110 señala que muchas implementaciones se apoyan en que HTTP no mantiene estado para reutilizar conexiones a través de proxies o repartir dinámicamente las peticiones entre varios servidores. Una capa web que no conserva nada entre peticiones puede crecer añadiendo copias detrás de un balanceador de carga, y cualquiera de ellas puede responder a cualquier petición.

La carga tiene más de una dimensión, y un sistema puede escalar en una y estancarse en otra: peticiones por segundo, volumen de datos almacenados, número de organizaciones cliente y número de escritores que compiten por el mismo registro. La última es la más difícil, porque una sola fila puede ser modificada por una transacción a la vez, por muchos servidores que la rodeen.

La aritmética que explica esto es la ley de Amdahl. Cuando parte del trabajo debe ejecutarse en secuencia, el hardware paralelo solo acelera el resto, por lo que la parte secuencial fija el límite: con un 5 % del trabajo en serie, el conjunto nunca puede ejecutarse más de 20 veces más rápido, ni siquiera en mil máquinas. En una base de datos, la parte secuencial suele ser un bloqueo. En una aplicación, suele ser un contador, una cola o un almacén de sesiones compartidos.

Utilice el concepto en una decisión

Aplique «¿Qué es la escalabilidad?» a una decisión real. Podemos ayudarle a relacionar el concepto con el sistema que está construyendo o comprando. Monitorización y alertas que avisan cuando un fallo aún es pequeño, y cifras que muestran con qué frecuencia se mantiene así.

Explore nuestros servicios Póngase en contacto →

Compruebe los supuestos detrás del término

Utilice la explicación y los trabajos enlazados para formular preguntas sobre su proyecto. Podemos examinar los supuestos con usted antes de elegir una herramienta. Primer panel de observabilidad de la empresa — la primera visión de la empresa sobre sus propios sistemas

Primer panel de observabilidad de la empresa Póngase en contacto →

Por qué importa

La escalabilidad determina cuánto cuesta crecer. Si un sistema escala bien, duplicar el tráfico es una línea en una factura. Si escala mal, la factura crece más deprisa que la carga, o llega un punto en el que añadir hardware ya no ayuda. Entonces, la solución pasa a ser un rediseño mientras sigue entrando tráfico.

El hardware puede aliviar cuellos de botella del procesador, la memoria o los discos, pero el trabajo ineficiente y la ejecución en serie pueden persistir. Una consulta que recorre sin necesidad toda una tabla puede seguir malgastando trabajo en un servidor mayor. Una máquina más grande no elimina la ejecución en serie de las actualizaciones de una fila compartida, y más servidores de aplicaciones pueden añadir escritores a la cola. La mayoría de los sistemas que parecen necesitar más máquinas necesitan primero que alguien los examine.

En PostgreSQL, el punto de partida es `pg_stat_statements`, una extensión que registra cada sentencia normalizada con su número de llamadas y su tiempo de ejecución. Al ordenar sus registros por tiempo total, no por la media, muestra a qué dedica realmente su tiempo el servidor: una consulta que tarda 2 ms y se ejecuta un millón de veces por hora cuesta 2.000 segundos de esa hora, mientras que un informe de dos segundos ejecutado dos veces cuesta cuatro.

```
SELECT calls, round(total_exec_time) AS total_ms,
       left(query, 60) AS query
  FROM pg_stat_statements
 ORDER BY total_exec_time DESC
 LIMIT 10;
```

Cada paso hacia más máquinas añade sus propios costes operativos, por lo que importa el orden. Una sentencia más eficiente o un índice mejor puede aumentar el rendimiento sin añadir otro servidor. Una réplica de lectura requiere configuración. Particionar cambia una tabla. El sharding, que reparte los datos entre servidores de bases de datos distintos, cambia las operaciones asociadas: las migraciones, las copias de seguridad, las restauraciones y los informes deben tener en cuenta que los datos están repartidos entre shards; las uniones entre shards pueden trasladarse al código de la aplicación, y una consulta que no conoce su clave de shard puede tener que consultarlos todos.

Cómo se aplica en la práctica

Una base de datos PostgreSQL que daba servicio a una aplicación de tamaño medio contenía cuentas de usuario, organizaciones cliente, datos geográficos y consumo eléctrico diario. Tenía poco tráfico, pero se ralentizaba a medida que aumentaba el volumen de datos. El encargo era hacerla diez veces más rápida sin rehacer la arquitectura, con margen para que los datos siguieran creciendo como parte del objetivo. El trabajo incluyó índices GiST para los datos espaciales, índices compuestos por organización cliente y rango de fechas, particionado temporal de la tabla de electricidad, particionado por hash para los datos geográficos, consultas reescritas comprobadas con `EXPLAIN ANALYZE` y un pool de conexiones. Los tiempos de respuesta se redujeron a la décima parte, el uso de recursos de la base de datos cayó un 40 % y el sistema admitió volúmenes de datos crecientes sin ralentizarse, con menos necesidad de ampliar el hardware.

En una empresa de tamaño medio con diez servidores de bases de datos PostgreSQL y SQL Server, la infraestructura se amplió para soportar un aumento del 40 % en el tráfico de usuarios sin degradar el servicio. Además, los tiempos de respuesta de las consultas mejoraron un 30 % mediante ajustes de configuración y optimización de índices.

El orden del trabajo se expone en una nota sobre optimizar antes de aplicar sharding: observar, ajustar, sacar las lecturas del primario, particionar las tablas alrededor del punto de mayor contención y aplicar sharding solo cuando una

máquina esté ocupada por completo con trabajo eficiente. Nombra tres casos en los que el sharding es razonable: el primario está saturado por sentencias ya optimizadas, los datos han superado la capacidad de mantenimiento o el aislamiento es, por sí mismo, un requisito.

Dónde surgen los problemas

Más servidores web delante de una sola base de datos. Cada servidor de aplicaciones abre sus propias conexiones. Cuando hay muchas veces más sesiones activas que núcleos en la base de datos, el servidor dedica el tiempo a alternar entre ellas. Un pool de conexiones resuelve eso antes que cualquier hardware, y otro servidor web lo empeora.

Una fila de alta contención. Cuando cada petición escribe en el mismo registro, como un contador de existencias o un saldo, la capacidad de procesamiento queda limitada aproximadamente por las escrituras que admite un bloqueo divididas por el tiempo que se mantiene, con cualquier número de máquinas. Cualquier tarea lenta dentro de ese bloqueo reduce el techo. Mantenga la transacción breve y las llamadas externas fuera de ella. Cuando un artículo supera la capacidad de su máquina, reparta sus existencias en partes más pequeñas, cada una con exactamente un propietario.

Estado que reside en un solo servidor. Las sesiones guardadas en la memoria de un proceso, los archivos subidos a su disco local o una tarea programada que supone ser la única copia impiden que una segunda instancia sea equivalente. Mantenga ese estado en un almacén compartido para que cualquier instancia pueda responder a cualquier petición.

Réplicas tratadas como copias del primario. Una réplica en streaming lleva un pequeño retraso. Un usuario que acaba de escribir algo y lo lee desde ella puede no verlo: lea del primario las escrituras recientes. Un informe largo en un servidor en espera también puede cancelarse si retrasa los cambios que este está reproduciendo. PostgreSQL los documenta como conflictos de consultas, junto con los ajustes que permiten equilibrar esos conflictos y el retraso de la réplica.

Particiones cuyas claves no aparecen en las consultas. El particionado divide una tabla por una clave, como un mes o una organización cliente. Una consulta que indica la clave solo lee las particiones que necesita. Una que la omite lee todas y no gana nada con la división. Particione según los campos por los que ya

filtran las consultas y espere beneficios solo con tablas grandes: la regla orientativa del manual de PostgreSQL es una tabla mayor que la memoria física del servidor de bases de datos.

Lecturas adicionales

- RFC 9110: semántica de HTTP, sección 3.3 — Conexiones, clientes y servidores, y el diseño sin estado que permite distribuir peticiones entre varios servidores.
- PostgreSQL: particionado de tablas — Qué es el particionado, cuándo compensa y cómo la poda de particiones omite las que una consulta no puede necesitar.
- PostgreSQL: pg_stat_statements — La extensión que conserva el número de llamadas y los tiempos de cada sentencia normalizada.
- PostgreSQL: Hot Standby — Consultas de solo lectura en un servidor en espera y conflictos que pueden cancelarlas.
- RFC 9111: caché HTTP — Cómo las cachés HTTP guardan y reutilizan respuestas para que las peticiones repetidas no tengan que llegar al servidor de origen.

Glosario

Elija según sus restricciones reales

Describa la ruta de despliegue, los riesgos operativos y quiénes se encargan de las guardias. Podemos definir el alcance de un cambio con comprobaciones, una vía de recuperación y responsabilidades claras.

Vea cuánto cuesta Póngase en contacto →

Haga práctica la terminología

Pregunte cómo se aplica «¿Qué es la escalabilidad?» a su organización. Basta un ejemplo concreto de su sistema actual para iniciar la conversación.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://engineer.company/es/glossary/scalability/