MySQL

Enterprise applications

NetApp
January 02, 2026

Sommario

MySQL

Panoramica

MySQL e le sue varianti, tra cui MariaDB e Percona MySQL, è il database più diffuso al mondo.

Questa documentazione su ONTAP e il database MySQL sostituisce il database *TR-4722: MySQL pubblicato in precedenza sulle Best practice di ONTAP.*

ONTAP è una piattaforma ideale per database MySQL, perché ONTAP è letteralmente progettato per database. Sono state create numerose funzionalità come le ottimizzazioni della latenza io random per la qualità del servizio avanzata fino alle funzionalità FlexClone di base per rispondere specificamente alle esigenze dei carichi di lavoro dei database.

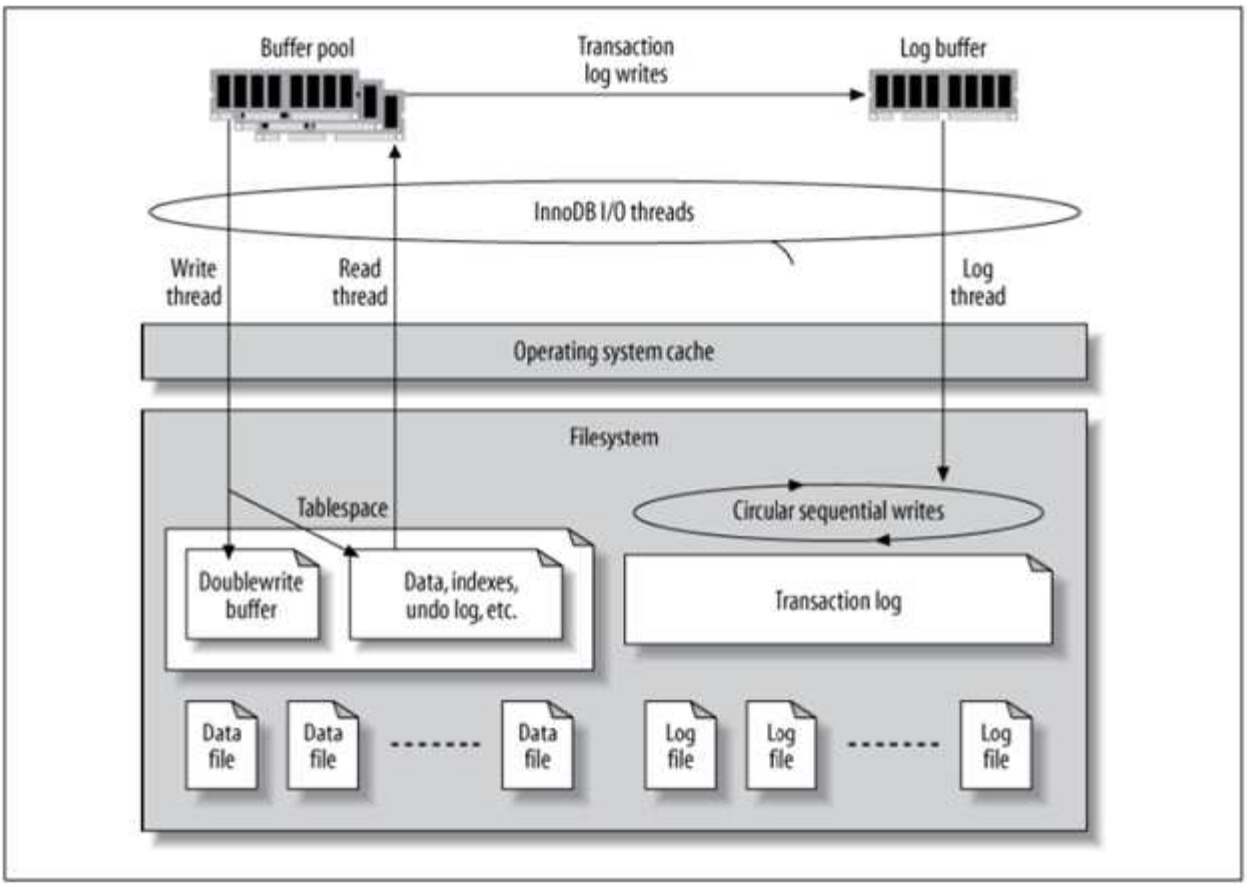
Funzioni aggiuntive come gli aggiornamenti senza interruzioni, (inclusa la sostituzione dello storage) garantiscono la disponibilità dei database critici. Puoi anche disporre di un disaster recovery istantaneo per ambienti di grandi dimensioni tramite MetroCluster o selezionare database tramite la sincronizzazione attiva di SnapMirror.

Soprattutto, ONTAP offre prestazioni senza pari con la possibilità di dimensionare la soluzione in base alle proprie esigenze specifiche. I nostri sistemi high-end possono offrire oltre 1M IOPS con latenze misurate in microsecondi, ma se ti servono solo 100K IOPS, puoi dimensionare correttamente la tua soluzione storage con un controller più piccolo che esegue ancora lo stesso sistema operativo per lo storage.

Configurazione del database

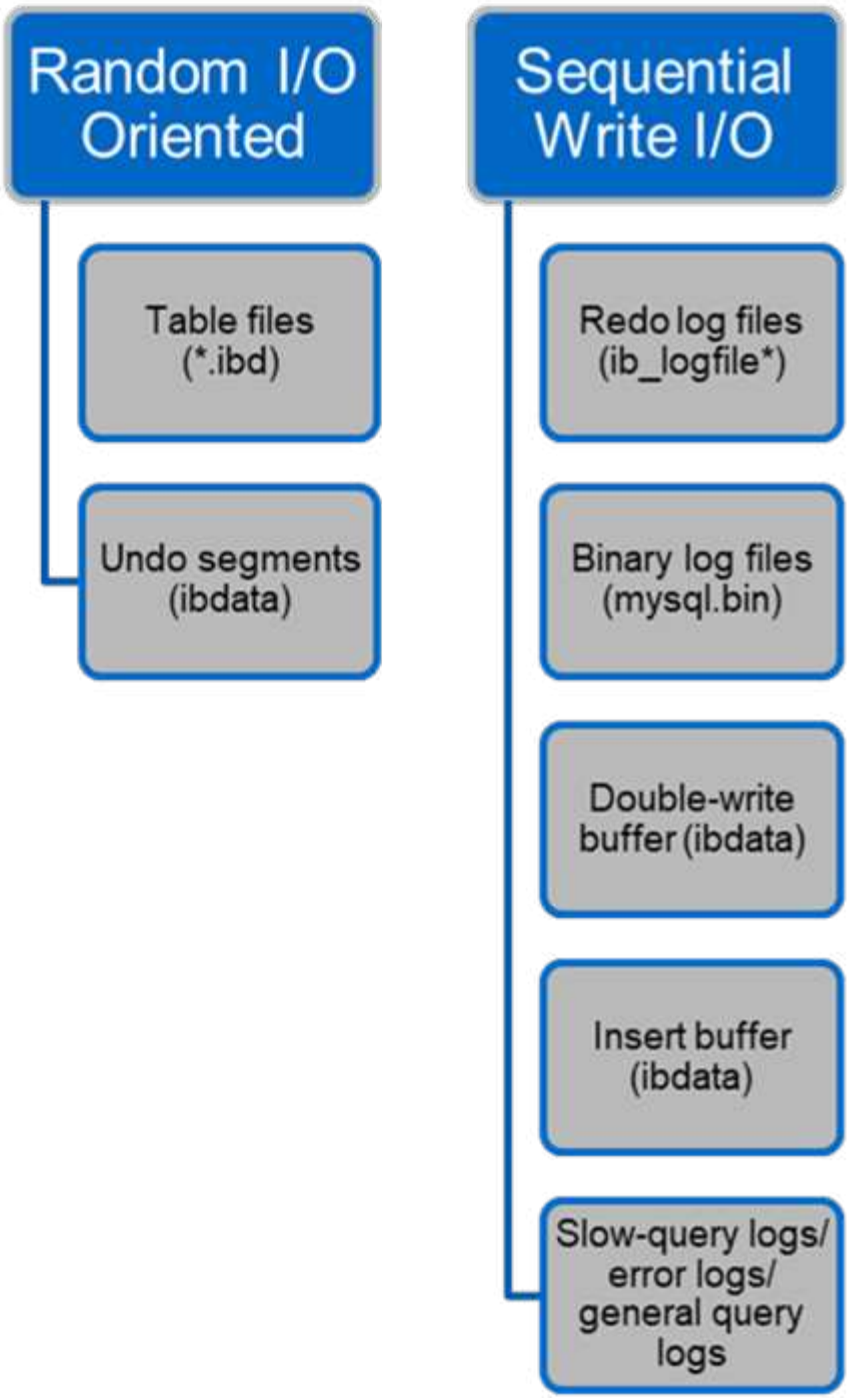
Struttura dei file

InnoDB funge da livello intermedio tra lo storage e il server MySQL, e memorizza i dati nelle unità.



I/o MySQL è suddiviso in due tipi:

- I/o di file casuali
- I/o di file sequenziale



I file di dati vengono letti e sovrascritti in modo casuale, con conseguente aumento degli IOPS. Pertanto, si consiglia di utilizzare l'unità SSD.

I file di log di ripristino e i file di log binari sono registri transazionali. Vengono scritti in sequenza, così potrai ottenere buone performance sul disco HDD con cache in scrittura. Al momento del ripristino si verifica una lettura sequenziale, che raramente causa problemi di prestazioni, poiché le dimensioni dei file di registro sono in genere inferiori ai file di dati e le letture sequenziali sono più veloci delle letture casuali (che si verificano sui file di dati).

Il buffer double-write è una caratteristica speciale di InnoDB. InnoDB prima scrive le pagine svuotate nel buffer di doppia scrittura e poi scrive le pagine nelle posizioni corrette sui file di dati. Questo processo impedisce il

danneggiamento della pagina. Senza il buffer di scrittura doppia, la pagina potrebbe danneggiarsi se si verifica un'interruzione dell'alimentazione durante il processo di scrittura su unità. La scrittura nel buffer double-write è sequenziale, pertanto è altamente ottimizzato per gli HDD. Al momento del ripristino vengono eseguite letture sequenziali.

Poiché la NVRAM ONTAP fornisce già la protezione in scrittura, non è necessario il doppio buffer in scrittura. MySQL ha un parametro, `skip_innodb_doublewrite`, per disattivare il buffer di doppia scrittura. Questa funzione può migliorare notevolmente le prestazioni.

Il buffer insert è anche una caratteristica speciale di InnoDB. Se i blocchi di indice secondari non univoci non sono in memoria, InnoDB inserisce le voci nel buffer di inserimento per evitare operazioni di i/o casuali. Periodicamente, il buffer di inserimento viene Unito agli alberi di indice secondari nel database. Il buffer di inserimento riduce il numero di operazioni di i/o unendo le richieste di i/o allo stesso blocco; le operazioni di i/o casuali possono essere sequenziali. Anche il buffer di inserimento è altamente ottimizzato per gli HDD. Durante le normali operazioni, vengono eseguite operazioni di scrittura e lettura sequenziali.

I segmenti di annullamento sono orientati all'i/o casuale. Per garantire la concorrenza multi-versione (MVCC), InnoDB deve registrare le vecchie immagini nei segmenti di annullamento. La lettura delle immagini precedenti dai segmenti di annullamento richiede letture casuali. Se si esegue una transazione lunga con letture ripetibili (come mysqldump, una singola transazione) o si esegue una query lunga, è possibile che si verifichino letture casuali. Pertanto, in questo caso è preferibile memorizzare i segmenti di annullamento negli SSD. Se si eseguono solo transazioni o query brevi, le letture casuali non costituiscono un problema.

NetApp consiglia il seguente layout di progettazione dello storage a causa delle caratteristiche i/o di InnoDB.

- Un unico volume per memorizzare i file di MySQL orientati ai/o casuali e sequenziali
- Un altro volume per memorizzare i file di MySQL orientati a i/o puramente sequenziali

Questo layout aiuta inoltre a progettare politiche e strategie di protezione dei dati.

Parametri di configurazione

NetApp consiglia alcuni importanti parametri di configurazione di MySQL per ottenere prestazioni ottimali.

Parametri	Valori
innodb_log_file_size	256M
innodb_flush_log_at_trx_commit	2
innodb_doublewrite	0
innodb_flush_method	fsync
innodb_buffer_pool_size	11G
innodb_io_capacity	8192
innodb_buffer_pool_instances	8
innodb_lru_scan_depth	8192
open_file_limit	65535

Per impostare i parametri descritti in questa sezione, è necessario modificarli nel file di configurazione MySQL (my.cnf). Le Best practice di NetApp sono il risultato di test eseguiti internamente.

innodb_log_file_size

La scelta della dimensione corretta per il file di log InnoDB è importante per le operazioni di scrittura e per avere un tempo di ripristino decente dopo un arresto anomalo del server.

Poiché molte transazioni sono registrate nel file, la dimensione del file di registro è importante per le operazioni di scrittura. Quando i record vengono modificati, la modifica non viene immediatamente riscritta nello spazio di tabella. La modifica viene invece registrata alla fine del file di registro e la pagina viene contrassegnata come sporca. InnoDB utilizza il proprio registro per convertire l'i/o casuale in i/o sequenziale

Quando il log è pieno, la pagina sporca viene scritta nello spazio di tabella in sequenza per liberare spazio nel file di log. Ad esempio, si supponga che un server si blocchi nel corso di una transazione e che le operazioni di scrittura vengano registrate solo nel file di registro. Prima che il server possa tornare attivo, deve passare attraverso una fase di recupero in cui vengono riprodotte le modifiche registrate nel file di registro. Maggiore è il numero di voci presenti nel file di registro, maggiore sarà il tempo necessario al server per il ripristino.

In questo esempio, la dimensione del file di registro influisce sia sul tempo di ripristino che sulle prestazioni di scrittura. Quando si sceglie il numero giusto per la dimensione del file di registro, bilanciare il tempo di ripristino rispetto alle prestazioni di scrittura. In genere, qualsiasi valore compreso tra 128M e 512M è un buon valore.

innodb_flush_log_at_trx_commit

In caso di modifica dei dati, la modifica non viene immediatamente scritta nell'archivio.

I dati vengono invece registrati in un buffer di registro, che è una porzione di memoria allocata da InnoDB alle modifiche del buffer registrate nel file di registro. InnoDB svuota il buffer nel file di registro quando viene eseguito il commit di una transazione, quando il buffer diventa pieno o una volta al secondo, a seconda dell'evento che si verifica per primo. La variabile di configurazione che controlla questo processo è innodb_Flush_log_at_trx_commit. Le opzioni valore includono:

- Quando si imposta `innodb_flush_log_trx_at_commit=0`, InnoDB scrive i dati modificati (nel pool di buffer InnoDB) nel file di log (ib_logfile) e scarica il file di log (write to storage) ogni secondo. Tuttavia, non fa nulla quando la transazione è impegnata. Se si verifica un'interruzione dell'alimentazione o un arresto anomalo del sistema, nessuno dei dati non scaricati è recuperabile perché non vengono scritti né nel file di registro né nelle unità.
- Quando si imposta `innodb_flush_log_trx_commit=1`, InnoDB scrive il buffer di log nel log delle transazioni e lo svuota nello storage durevole per ogni transazione. Ad esempio, per tutti i commit delle transazioni, InnoDB scrive nel registro e quindi nello storage. La lentezza dello storage influisce negativamente sulle performance, ad esempio riducendo il numero di transazioni InnoDB al secondo.
- Quando si imposta `innodb_flush_log_trx_commit=2`, InnoDB scrive il buffer di log nel file di log ad ogni commit; tuttavia, non scrive dati nell'archivio. InnoDB scarica i dati una volta al secondo. Anche in caso di interruzione dell'alimentazione o arresto anomalo del sistema, i dati dell'opzione 2 sono disponibili nel file di registro ed è recuperabile.

Se l'obiettivo principale è la prestazione, impostare il valore su 2. Poiché InnoDB scrive sui dischi una volta al secondo, non per ogni commit delle transazioni, le performance migliorano in modo significativo. Se si verifica un'interruzione dell'alimentazione o un arresto anomalo, i dati possono essere recuperati dal registro delle transazioni.

Se l'obiettivo principale è la sicurezza dei dati, impostare il valore su 1 in modo che per ogni commit di transazione, InnoDB si scarichi sulle unità. Tuttavia, le prestazioni potrebbero risentirne.

NetApp recommended impostare il valore innodb_Flush_log_trx_commit su 2 per ottenere prestazioni migliori.

innodb_doublewrite

Quando `innodb_doublewrite` È attivato (impostazione predefinita), InnoDB memorizza tutti i dati due volte: Prima nel buffer di doppia scrittura e poi nei file di dati effettivi.

È possibile disattivare questo parametro con `--skip-innodb_doublewrite` per i benchmark o quando siete più preoccupati per le prestazioni superiori che l'integrità dei dati o possibili guasti. InnoDB utilizza una tecnica di scaricamento file chiamata double-write. Prima di scrivere le pagine nei file di dati, InnoDB le scrive in un'area contigua denominata buffer double-write. Una volta completata la scrittura e lo scarico nel buffer di doppia scrittura, InnoDB scrive le pagine nelle posizioni corrette nel file di dati. Se il sistema operativo o un processo mysqld si blocca durante la scrittura di una pagina, InnoDB può in seguito trovare una buona copia della pagina dal buffer di doppia scrittura durante il recupero del crash.

NetApp recommended disabilitare il buffer double-write. La NVRAM ONTAP svolge la stessa funzione. Il doppio buffering danneggia inutilmente le prestazioni.

innodb_buffer_pool_size

Il pool di buffer InnoDB è la parte più importante di qualsiasi attività di ottimizzazione.

InnoDB si affida in gran parte al pool di buffer per la memorizzazione nella cache degli indici e il reming dei dati, all'indice hash adattivo, al buffer insert e a molte altre strutture di dati utilizzate internamente. Il pool di buffer memorizza inoltre le modifiche ai dati in modo che le operazioni di scrittura non debbano essere eseguite immediatamente nello storage, migliorando così le prestazioni. Il pool di buffer è parte integrante di InnoDB e le sue dimensioni devono essere regolate di conseguenza. Per impostare le dimensioni del pool di buffer, tenere conto dei seguenti fattori:

- Per una macchina dedicata solo InnoDB, impostare la dimensione del pool di buffer su 80% o più della RAM disponibile.
- Se non si tratta di un server dedicato MySQL, impostare la dimensione al 50% della RAM.

innodb_flush_method

Il parametro innodb_Flush_Method specifica come InnoDB apre e svuota i file di log e di dati.

Ottimizzazioni

Nell'ottimizzazione InnoDB, l'impostazione di questo parametro modifica le prestazioni del database, se applicabile.

Le seguenti opzioni consentono di svuotare i file tramite InnoDB:

- `fsync`. InnoDB utilizza `fsync()` chiamata di sistema per cancellare sia i file di dati che i file di registro.

Questa opzione è l'impostazione predefinita.

- `O_DSYNC`. InnoDB utilizza `O_DSYNC` possibilità di aprire e svuotare i file di log e fsync() per svuotare i file di dati. InnoDB non utilizza `O_DSYNC` Direttamente, perché ci sono stati problemi con esso su molte varietà di UNIX.
- `O_DIRECT`. InnoDB utilizza `O_DIRECT` (oppure `directio()` Su Solaris) per aprire i file di dati e gli usi `fsync()` per cancellare sia i file di dati che i file di registro. Questa opzione è disponibile su alcune versioni di GNU/Linux, FreeBSD e Solaris.
- `O_DIRECT_NO_FSYNC`. InnoDB utilizza `O_DIRECT` Durante lo spurgo dell'i/o, tuttavia, salta `fsync()` chiamata di sistema successiva. Questa opzione non è adatta per alcuni tipi di file system (ad esempio, XFS). Se non si è certi che il file system richieda un `fsync()` chiamata di sistema, ad esempio per conservare tutti i metadati dei file, utilizzare `O_DIRECT` invece.

Osservazione

Nei test di laboratorio di NetApp, il `fsync` L'opzione predefinita è stata utilizzata su NFS e SAN ed è stata un'improvvisazione per le prestazioni eccezionale rispetto a `O_DIRECT`. Mentre si utilizza il metodo di lavaggio come `O_DIRECT` Con ONTAP, abbiamo osservato che il client scrive molte scritture a byte singolo al margine del blocco 4096 in modo seriale. Queste operazioni di scrittura hanno aumentato la latenza sulla rete e degradato le performance.

innodb_io_capacity

Nel plug-in InnoDB è stato aggiunto un nuovo parametro chiamato innodb_io_Capacity da MySQL 5,7.

Controlla il numero massimo di IOPS eseguiti da InnoDB (che include la velocità di scaricamento delle pagine sporche e la dimensione batch del buffer di inserimento [ibuf]). Il parametro innodb_io_Capacity imposta un limite massimo per le IOPS da parte delle attività in background di InnoDB, come il lavaggio delle pagine dal pool di buffer e l'Unione dei dati dal buffer di modifica.

Impostare il parametro innodb_io_Capacity sul numero approssimativo di operazioni di i/o che il sistema può eseguire al secondo. Idealmente, mantenere l'impostazione più bassa possibile, ma non così bassa che le attività in background rallentano. Se l'impostazione è troppo alta, i dati vengono rimossi dal pool di buffer e il buffer di inserimento troppo rapidamente per la memorizzazione nella cache, per fornire un vantaggio significativo.

NetApp consiglia che, se si utilizza questa impostazione su NFS, analizzi il risultato del test di IOPS (SysBench/FiO) e imposti il parametro di conseguenza. Utilizzare il valore più piccolo possibile per lo spurgo e lo spurgo per continuare a meno che non vengano visualizzate pagine modificate o sporche di quanto si desidera nel pool di buffer InnoDB.

Non utilizzare valori estremi come 20.000 o più a meno che non si sia dimostrato che valori inferiori non sono sufficienti per il carico di lavoro.

Il parametro InnoDB_io_Capacity regola le velocità di lavaggio e i/o correlati

È possibile danneggiare seriamente le prestazioni impostando questo parametro o il parametro innodb_io_Capacity_max troppo alto e wastin

innodb_lru_scan_depth

Il `innodb_lru_scan_depth` Parametro influenza gli algoritmi e le euristiche dell'operazione di scaricamento per il pool di buffer InnoDB.

Questo parametro è principalmente di interesse per gli esperti di performance che ottimizzano i carichi di lavoro i/o-intensive. Per ogni istanza del pool di buffer, questo parametro specifica fino a che punto nell'elenco di pagine LRU (Last Recently Used) il thread di pulitura della pagina deve continuare la scansione, cercando le pagine sporche da eliminare. Questa operazione in background viene eseguita una volta al secondo.

È possibile regolare il valore verso l'alto o verso il basso per ridurre al minimo il numero di pagine libere. Non impostare un valore molto superiore al necessario, poiché le scansioni possono avere un costo significativo in termini di prestazioni. Inoltre, è consigliabile regolare questo parametro quando si modifica il numero di istanze del pool di buffer, perché `innodb_lru_scan_depth * innodb_buffer_pool_instances` definisce la quantità di lavoro eseguito dal filo del pulitore di pagina ogni secondo.

Un'impostazione più piccola di quella predefinita è adatta per la maggior parte dei carichi di lavoro. Considerare l'aumento del valore solo se si dispone di capacità i/o di riserva con un workload tipico. Per contro, se un carico di lavoro con un numero elevato di operazioni di scrittura satura la capacità i/o, diminuirne il valore, soprattutto se si dispone di un pool di buffer di grandi dimensioni.

open_file_limits

Il `open_file_limits` parametro determina il numero di file che il sistema operativo consente a mysqld di aprire.

Il valore di questo parametro in fase di esecuzione è il valore reale consentito dal sistema e potrebbe essere diverso dal valore specificato all'avvio del server. Il valore è 0 sui sistemi in cui MySQL non può modificare il numero di file aperti. L'efficace `open_files_limit` il valore si basa sul valore specificato all'avvio del sistema (se presente) e sui valori di `max_connections` e. `table_open_cache` utilizzando queste formule:

- 10 + `max_connections` + (`table_open_cache` x 2)
- `max_connections` x 5
- Limite del sistema operativo se positivo
- Se il limite del sistema operativo è infinito: `open_files_limit` il valore viene specificato all'avvio; 5.000 se nessuno

Il server tenta di ottenere il numero di descrittori di file utilizzando il massimo di questi quattro valori. Se non è possibile ottenere molti descrittori, il server tenta di ottenere il numero di descrittori consentito dal sistema.

Configurazione dell'host

Containerizzazione

La containerizzazione dei database MySQL sta diventando sempre più diffusa.

La gestione di container a basso livello viene quasi sempre eseguita con Docker. Le piattaforme di gestione dei container come OpenShift e Kubernetes semplificano ulteriormente la gestione di ambienti container di grandi dimensioni. I vantaggi della containerizzazione includono una riduzione dei costi, poiché non è necessario acquistare una licenza per un hypervisor. Inoltre, i container consentono l'esecuzione di più

database isolati l'uno dall'altro, condividendo lo stesso kernel e sistema operativo sottostanti. È possibile eseguire il provisioning dei container in microsecondi.

NetApp offre Astra Trident per fornire funzionalità di gestione avanzate dello storage. Ad esempio, Astra Trident consente a un container creato in Kubernetes di eseguire il provisioning automatico del proprio storage nel Tier appropriato, applicare policy di esportazione, impostare policy di snapshot e persino clonare un container in un altro. Per ulteriori informazioni, vedere "Documentazione di Astra Trident".

NFSv3 tavoli con slot

NFSv3 le prestazioni di Linux dipendono da un parametro chiamato `tcp_max_slot_table_entries`.

Le tabelle degli slot TCP sono l'equivalente di NFSv3 della profondità della coda degli HBA (host Bus Adapter). Queste tabelle controllano il numero di operazioni NFS che possono essere in sospeso in qualsiasi momento. Il valore predefinito è di solito 16, che è troppo basso per ottenere prestazioni ottimali. Il problema opposto si verifica sui kernel Linux più recenti, che possono aumentare automaticamente il limite della tabella degli slot TCP a un livello che satura il server NFS con le richieste.

Per prestazioni ottimali e per evitare problemi di prestazioni, regolare i parametri del kernel che controllano le tabelle degli slot TCP.

Eseguire `sysctl -a | grep tcp.*.slot_table` e osservare i seguenti parametri:

```
# sysctl -a | grep tcp.*.slot_table
sunrpc.tcp_max_slot_table_entries = 128
sunrpc.tcp_slot_table_entries = 128
```

Tutti i sistemi Linux dovrebbero includere `sunrpc.tcp_slot_table_entries`, ma solo alcuni includono `sunrpc.tcp_max_slot_table_entries`. Entrambi devono essere impostati su 128.

> La mancata impostazione di questi parametri può avere effetti significativi sulle prestazioni. In alcuni casi, le prestazioni sono limitate poiché il sistema operativo linux non fornisce i/o sufficienti In altri casi, le latenze i/o aumentano quando il sistema operativo linux tenta di emettere più i/o di quanto possa essere gestito.

Scheduler i/O.

Il kernel Linux permette un controllo di basso livello sul modo in cui l'i/o blocca i dispositivi è programmato.

Le impostazioni predefinite su varie distribuzioni di Linux variano notevolmente. MySQL consiglia di utilizzare `NOOP` oppure un `deadline` Scheduler i/o con i/o asincrono nativo (AIO) su Linux. In generale, i clienti NetApp e i test interni mostrano risultati migliori con NoOps.

Il motore di storage InnoDB di MySQL utilizza il sottosistema i/o asincrono (AIO nativo) su Linux per eseguire richieste di lettura e scrittura per le pagine dei file di dati. Questo comportamento è controllato da `innodb_use_native_aio` opzione di configurazione, attivata per impostazione predefinita. Con un sistema AIO nativo, il tipo di pianificatore i/o influisce maggiormente sulle prestazioni di i/O. Esegui benchmark per determinare quale scheduler i/o offrirà i risultati migliori per il tuo carico di lavoro e l'ambiente.

Per istruzioni sulla configurazione dello scheduler i/o, consultare la documentazione relativa a Linux e MySQL.

Descrittori di file

Per l'esecuzione, il server MySQL ha bisogno di descrittori di file, e i valori predefiniti non sono sufficienti.

Le utilizza per aprire nuove connessioni, archiviare tabelle nella cache, creare tabelle temporanee per risolvere query complesse e accedere a quelle persistenti. Se mysqld non è in grado di aprire nuovi file quando necessario, può smettere di funzionare correttamente. Un sintomo comune di questo problema è l'errore 24, "troppi file aperti". Il numero di descrittori di file che mysqld può aprire simultaneamente è definito dal `open_files_limit` opzione impostata nel file di configurazione (`/etc/my.cnf`). Ma `open_files_limit` dipende anche dai limiti del sistema operativo. Questa dipendenza rende l'impostazione della variabile più complicata.

MySQL non può impostare ITS `open_files_limit` opzione superiore a quanto specificato in `ulimit 'open files'`. Pertanto, è necessario impostare esplicitamente questi limiti a livello del sistema operativo per consentire a MySQL di aprire i file in base alle necessità. Ci sono due modi per controllare il limite dei file in Linux:

- Il `ulimit` command fornisce rapidamente una descrizione dettagliata dei parametri consentiti o bloccati. Le modifiche apportate eseguendo questo comando non sono permanenti e si cancellano dopo un riavvio del sistema.
- Modifiche al `/etc/security/limit.conf` i file sono permanenti e non sono interessati dal riavvio del sistema.

Assicurarsi di modificare sia i limiti hard che soft per l'utente mysql. I seguenti estratti provengono dalla configurazione:

```
mysql hard nofile 65535
mysql soft nofile 65353
```

In parallelo, aggiornare la stessa configurazione in `my.cnf` per utilizzare completamente i limiti dei file aperti.

Configurazione dello storage

NFS

La documentazione MySQL consiglia di utilizzare NFSv4 per le implementazioni NAS.

Dimensioni del trasferimento di NFS ONTAP

Per impostazione predefinita, ONTAP limiterà le dimensioni di i/o NFS a 64K. I/o casuali con un database MySQL utilizzano blocchi di dimensioni molto inferiori, che sono ben al di sotto del massimo di 64K KB. L'io a blocchi di grandi dimensioni è solitamente parallelizzato, quindi anche il massimo di 64K KB non costituisce un limite.

Ci sono alcuni carichi di lavoro in cui il massimo di 64K crea un limite. In particolare, le operazioni single-threaded, come le operazioni di backup della scansione completa del piano d'esame, verranno eseguite in modo più rapido ed efficiente se il database è in grado di eseguire un numero di io inferiore ma maggiore. La

dimensione ottimale di gestione io per ONTAP con carichi di lavoro del database è 256K. Le opzioni di montaggio NFS elencate per i sistemi operativi specifici elencati di seguito sono state aggiornate da 64K a 256K di conseguenza.

Le dimensioni massime di trasferimento per una SVM ONTAP possono essere modificate come segue:

```
Cluster01::> set advanced

Warning: These advanced commands are potentially dangerous; use them only
when directed to do so by NetApp personnel.

Do you want to continue? {y|n}: y

Cluster01::*> nfs server modify -vserver vserver1 -tcp-max-xfer-size
262144
```

Non diminuire mai la dimensione di trasferimento massima consentita su ONTAP al di sotto del valore rsize/wsize dei filesystem NFS attualmente montati. In alcuni sistemi operativi, ciò può causare blocchi o addirittura danni ai dati. Ad esempio, se i client NFS sono attualmente impostati su un valore rsize/wsize di 65536, la dimensione massima di trasferimento ONTAP potrebbe essere regolata tra 65536 e 1048576 senza alcun effetto perché i client stessi sono limitati. La riduzione della dimensione massima di trasferimento inferiore a 65536 GB può danneggiare la disponibilità o i dati.

NetApp consiglia

Impostazione della seguente impostazione NFSv4 fstab (/etc/fstab):

```
nfs4 rw,
hard,nointr,bg,vers=4,proto=tcp,noatime,rsize=262144,wsize=262144
```

Un problema comune con NFSv3 è stato il blocco dei file di registro InnoDB dopo un'interruzione dell'alimentazione. Questo problema è stato risolto utilizzando i file di registro Time o Switching. Tuttavia, NFSv4 ha operazioni di blocco e tiene traccia dei file aperti e delle delegazioni.

SAN

È possibile collocare database di dimensioni inferiori su una coppia di LUN standard, a condizione che le richieste di i/o e capacità rientrino nei limiti di un singolo file system LUN. Ad esempio, un database che richiede circa 2K IOPS casuali può essere ospitato su un singolo file system su un singolo LUN. Analogamente, un database di sole 100GB GB di dimensioni dovrebbe adattarsi a un singolo LUN, senza creare problemi di gestione.

Database di dimensioni maggiori richiedono LUN multiple. Ad esempio, un database che richiede 100K IOPS avrà probabilmente bisogno di almeno otto LUN. Un singolo LUN sarebbe diventato un collo di bottiglia a causa del numero inadeguato di canali SCSI per le unità. Analogamente, sarebbe difficile gestire un database da 10TB TB su un singolo LUN da 10TB GB. I gestori di volumi logici sono progettati per unire le funzionalità di performance e capacità di più LUN per migliorare le prestazioni e la gestibilità.

In entrambi i casi, dovrebbe essere sufficiente una coppia di ONTAP Volumes. Con una configurazione semplice, la LUN dei file di dati viene posizionata in un volume dedicato, come farebbe la LUN di log. Con una configurazione di volume manager logica, tutte le LUN del gruppo di volumi dei file di dati si troverebbero in un volume dedicato e le LUN del gruppo di volumi di log si troverebbero in un secondo volume dedicato.

NetApp consiglia di utilizzare due file system per le distribuzioni MySQL su SAN:

- Il primo file system memorizza tutti i dati MySQL inclusi tablespace, dati e indice.
- Il secondo file system archivia tutti i log (log binari, log lenti e log delle transazioni).

Esistono diverse ragioni per separare i dati in questo modo, tra cui:

- I modelli di i/o dei file di dati e di registro sono diversi. La loro separazione permetterebbe più opzioni con i controlli QoS.
- L'uso ottimale della tecnologia Snapshot richiede la capacità di ripristinare in maniera indipendente i file di dati. L'associazione di file di dati con file di registro interferisce con il ripristino dei file di dati.
- La tecnologia NetApp SnapMirror può essere utilizzata per fornire una semplice funzionalità di disaster recovery con RPO ridotto per un database; tuttavia, richiede diverse pianificazioni della replica per i file di dati e log.

Utilizzare questo layout di base a due volumi per rendere la soluzione a prova di futuro, in modo che tutte le funzioni di ONTAP possano essere utilizzate se necessario.

NetApp consiglia la formattazione dell'unità con il file system ext4, grazie alle seguenti funzioni:

- Approccio esteso alle funzioni di gestione dei blocchi utilizzate nel file system di journaling (JFS) e nelle funzioni di allocazione differita del file system esteso (XFS).
- ext4 permette file system fino a 1 exbibyte (2^60 byte) e file fino a 16 tebibyte (16 * 2^40 byte). Al contrario, il file system ext3 supporta solo file system di dimensioni massime pari a 16TB MB e file di dimensioni massime pari a 2TB MB.
- Nei file system ext4, l'allocazione di più blocchi (mballoc) alloca più blocchi per un file in un'unica operazione, invece di assegnarli uno alla volta, come in ext3. Questa configurazione riduce l'overhead di chiamata dell'allocatore di blocchi diverse volte e ottimizza l'allocazione di memoria.
- Anche se XFS è il default per molte distribuzioni Linux, gestisce i metadati in modo diverso e non è adatto per alcune configurazioni MySQL.

NetApp consiglia di utilizzare le opzioni di dimensione del blocco 4K con l'utilità mkfs per allinearsi alle dimensioni del LUN del blocco esistenti.

```
mkfs.ext4 -b 4096
```

Le LUN NetApp memorizzano dati in blocchi fisici da 4KB KB, ottenendo otto blocchi logici da 512 byte.

Se non si impostano le stesse dimensioni del blocco, l'i/o non verrà allineato correttamente con i blocchi fisici e potrebbe scrivere in due unità diverse in un gruppo RAID, con conseguente latenza.

È importante allineare l'i/o per semplificare le operazioni di lettura/scrittura. Tuttavia, quando l'i/o inizia ad un blocco logico che non si trova all'inizio di un blocco fisico, l'i/o è disallineato. Le operazioni di i/o sono allineate solo quando iniziano presso un blocco logico, il primo blocco logico in un blocco fisico.