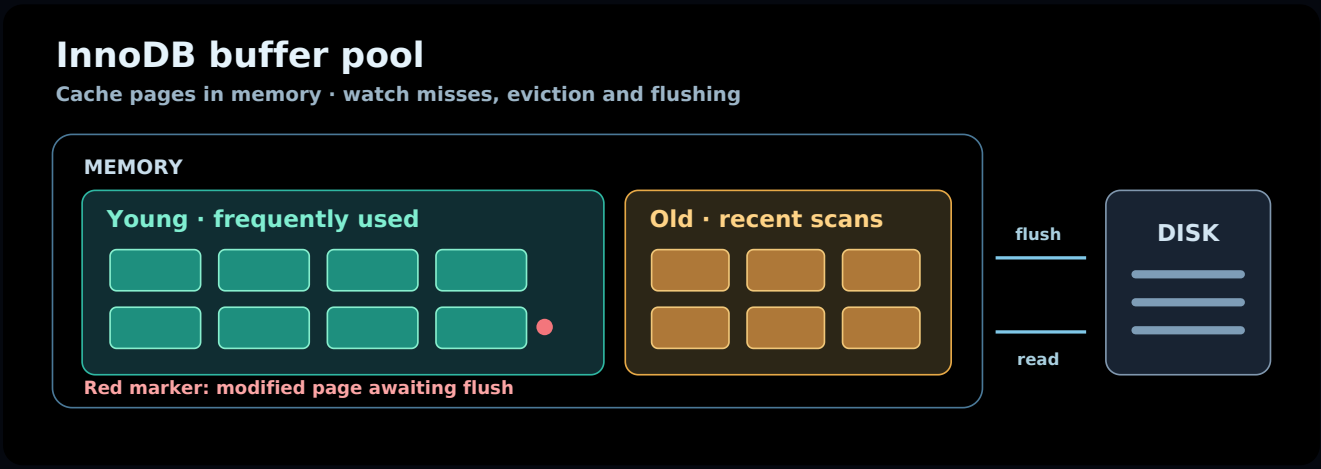
Bufor InnoDB: jak działa i jak stroić jego wydajność

Duthaho · September 28, 2026

MYSQL INNODB BUFFER-POOL PERFORMANCE



InnoDB korzysta z pamięci, aby nie odczytywać każdej strony z dysku. Bufor przechowuje niedawno używane strony danych i indeksów, a operacje zapisu modyfikują je w pamięci przed zrzuceniem na dysk. Wiedza o tym, które strony trafiają do bufora, które go opuszczają i jak szybko się on rozgrzewa, pomaga wyjaśnić zarówno wydajność w normalnej pracy, jak i spowolnienia po restarcie.

Co przechowuje bufor

`innodb_buffer_pool_size` określa wielkość tego wspólnego obszaru pamięci. Strony InnoDB mają zwykle 16 KiB, choć `innodb_page_size` można ustawić inaczej podczas tworzenia instancji. Strony danych i indeksów dzielą miejsce ze strukturami wewnętrznymi. Strona obecna w pamięci nie wymaga odczytu pliku danych; brakującą trzeba pobrać z nośnika.

Na serwerze przeznaczonym wyłącznie na bazę duży bufor może pomóc, lecz żaden stały procent pamięci RAM nie jest właściwy wszędzie. Zostaw pamięć systemowi operacyjnemu, połączeniom, operacjom tymczasowym, innym silnikom i usługom działającym na tym samym hoście. Bufor 48 GiB na serwerze z 64 GiB RAM to **przykład do przetestowania**, a nie uniwersalne zalecenie.

Mierz odczyty w tym samym przedziale czasu, zamiast polegać na procentach liczonych od uruchomienia serwera:

```
SHOW GLOBAL STATUS WHERE Variable_name IN (
  'Innodb_buffer_pool_read_requests',
  'Innodb_buffer_pool_reads',
  'Innodb_buffer_pool_pages_dirty',
  'Innodb_buffer_pool_wait_free'
);
```

Wskaźnik nietrafień w danym przedziale to `ΔInnodb_buffer_pool_reads / ΔInnodb_buffer_pool_read_requests`. Jeśli rośnie wraz z liczbą odczytów z nośnika i opóźnieniem zapytań, aktywny zbiór danych może nie mieścić się dobrze w pamięci. Wysoki współczynnik trafień sam w sobie nie dowodzi, że zapytanie jest szybkie. Liczba zmodyfikowanych stron i `Innodb_buffer_pool_wait_free` pomagają ocenić presję zrzucania danych.

Dlaczego duży skan nie musi usuwać wszystkich często używanych stron

InnoDB używa zmodyfikowanej listy LRU. Nowo wczytane strony trafiają do części **old**, a często używane mogą awansować do części **young**. Wstawianie w środku listy ogranicza wypieranie aktywnego zbioru danych przez jednorazowy skan tabeli.

`innodb_old_blocks_pct` określa udział części old (domyślnie 37 w MySQL 8.4). `innodb_old_blocks_time` ustala opóźnienie przed awansem nowej strony (domyślnie 1000 ms w MySQL 8.4). Dłuższy czas może chronić strony OLTP podczas dużych raportów, ale także opóźniać awans naprawdę przydatnych stron. Ustawienie zera usuwa to opóźnienie; nie jest ogólnym lekarstwem na obciążenie odczytami. Zanim zmienisz któryś parametr, zmierz rzeczywisty ruch.

Rozgrzewanie bufora po restarcie lub przełączeniu

Po restarcie pamięć podręczna jest zimna. Pierwsze żądania mogą intensywnie czytać pliki danych, mimo że SQL i indeksy się nie zmieniły. InnoDB może zapisać **identyfikatory** stron przy zamykaniu i ponownie je wczytać przy starcie za pomocą `innodb_buffer_pool_dump_at_shutdown` oraz `innodb_buffer_pool_load_at_startup`. `innodb_buffer_pool_dump_pct` określa, jaka część najczęściej używanych stron zostanie zapamiętana. Zrzut nie zawiera samych stron; uruchomienie nadal wymaga odczytów dyskowych.

Można również pozwolić normalnemu ruchowi rozgrzać bufor, uruchomić ograniczone i reprezentatywne zapytania albo przygotować replikę przed skierowaniem na nią ruchu. Wstępne ładowanie może konkurować z produkcją o I/O, więc ogranicz jego tempo i obserwuj opóźnienia. Bufor zapełniony w 80% niekoniecznie jest ciepły dla ważnych zapytań. Gotowość oceniaj na podstawie opóźnień i odczytów fizycznych reprezentatywnych żądań oraz, jeśli używasz ładowania, jego statusu.

Zapisy, redo i inne struktury

Gdy transakcja zmienia stronę, jej kopia w buforze staje się **brudna**. InnoDB zapisuje rekordy redo, aby po awarii móc odtworzyć zatwierdzone zmiany, a procesy w tle zapisują później zmodyfikowane strony do plików danych. To przesuwa punkt kontrolny. Dziennik redo i bufor pełnią różne role: zbyt agresywny cel dla brudnych stron zwiększa I/O, a zbyt wolny zapis może później spowodować presję. Dobierz trwałość do wymagań; `innodb_flush_log_at_trx_commit=1` jest zwykłym wyborem, gdy każda zatwierdzona transakcja ma przetrwać awarię serwera.

Change buffer może odroczyć zmiany stron nieunikalnych indeksów wtórnych, których nie ma w pamięci. W niektórych obciążeniach zapisami w MySQL ogranicza to losowe I/O, ale powoduje późniejsze scalanie i zużywa pamięć. **Adaptive hash index** może przyspieszyć powtarzane wyszukiwania, lecz również zużywa pamięć i może zwiększać rywalizację między wątkami. Żaden z tych mechanizmów nie pomaga automatycznie: sprawdź dostępność w używanej wersji, obserwuj liczniki i przetestuj reprezentatywny ruch przed zmianą ustawień. W szczególności MariaDB od wersji 10.5 używa jednej instancji bufora, więc rada, by ustawić `innodb_buffer_pool_instances=8`, jej nie dotyczy.

Praktyczna kolejność strojenia

W obciążonym systemie zamówień zacznij od rzeczywistego aktywnego zbioru danych i budżetu pamięci. Zapisz opóźnienia zapytań, odczyty fizyczne, liczbę brudnych stron i I/O przy typowym oraz szczytowym ruchu. Jeśli przy pełnym buforze rosną odczyty z dysku, najpierw popraw indeksy i sposób dostępu do danych, a potem przetestuj większy bufor, obserwując całkowite zużycie pamięci hosta. Jeśli skan analityczny wypiera aktywne strony, przetestuj ustawienia części old. Jeśli opóźnienia rosną tylko po restarcie lub awansowaniu repliki, sprawdź dump/load i rozgrzej przyszły serwer główny przed skierowaniem na niego ruchu produkcyjnego.

W MySQL 8.4 kilka instancji bufora może ograniczyć rywalizację przy buforach wielogigabajtowych, lecz ich liczba powinna wynikać z pomiarów i pozostawiać około 1 GiB lub więcej na instancję. Ogólny cel trafień, taki jak 99%, jest wskazówką diagnostyczną, a nie samodzielnym celem jakości usługi. Porównuj zmiany przy tym samym obciążeniu i uwzględniaj zarówno zyski szybkości, jak i regresje pamięci oraz I/O.

Podsumowanie

Bufor InnoDB to więcej niż duża pamięć podręczna. Zasady przyjmowania stron chronią często używane dane, rozgrzewanie wpływa na opóźnienia po restarcie, a zrzucanie łączy zapisy w pamięci z trwałym przechowywaniem. Strojenie opieraj na pomiarach własnego obciążenia i zachowaj zapas pamięci, zamiast stosować uniwersalny przepis.

Opracowano i przetłumaczono za zgodą autora na podstawie artykułu Duthaho na Medium, opublikowanego pierwotnie 22 sierpnia 2025 r. Szczegóły zależne od wersji sprawdzono w dokumentacji bufora MySQL 8.4 i dokumentacji MariaDB.