my.cnf: konfiguracja serwera

Poziom: podstawy **Dotyczy: MySQL 5.7–8.0, MariaDB 10.x–11.x** **Czas: ~25 min**

Plik `my.cnf` jest jedynym miejscem, w którym zapisane ustawienia przeżywają restart serwera — wszystko, co zmienimy tylko poleceniem `SET GLOBAL`, po restarcie wraca do wartości z pliku.

Skróty użyte na tej stronie (7)

Skrót	Co znaczy
B2B	business to business: sprzedaż i systemy skierowane do innych firm, nie do klientów indywidualnych
CLI	Command Line Interface: praca przez wpisywanie komend w terminalu
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
RAM	Random Access Memory: pamięć operacyjna
RHEL	Red Hat Enterprise Linux: komercyjna dystrybucja Linuksa
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
TCP	Transmission Control Protocol: protokół sieciowy z potwierdzaniem dostarczenia

Fiszka

TYLKO ODCZYT

Odczytanie wartości pojedynczego parametru: SHOW VARIABLES LIKE

```
SHOW VARIABLES LIKE 'max_connections';
```

Sprawdzamy bieżącą, obowiązującą wartość parametru bez przeglądania całego pliku konfiguracyjnego.

Przykład:

```
+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| max_connections | 151   |
+-----------------+-------+
1 row in set (0.00 sec)
```

show = FortiOS: wypisuje konfigurację, w podstawowej postaci tylko ustawienia różniące się od domyślnych, a z wariantem full-configuration także wartości domyślne

ZMIENIA SYSTEM

Zmiana parametru dla całej instancji w locie: SET GLOBAL

```
SET GLOBAL max_connections = 500;
```

Podnosimy limit połączeń natychmiast, bez restartu serwera — zmiana obowiązuje tylko do najbliższego restartu, jeśli nie zapiszemy jej też w `my.cnf`.

Przykład:

```
Query OK, 0 rows affected (0.00 sec)
```

SET = Switch Embedded Teaming: zespół kart sieciowych zbudowany w przełączniku wirtualnym Hyper-V; łączy dwie karty w jedno łącze bez ustawień po stronie przełącznika fizycznego, więc bez LACP

ZMIENIA SYSTEM

Trwała zmiana parametru bez edycji pliku (MySQL 8.0): SET PERSIST

```
SET PERSIST innodb_flush_log_at_trx_commit = 1;
```

Zmieniamy parametr w locie i jednocześnie zapisujemy go tak, żeby przetrwał restart serwera, bez ręcznej edycji `my.cnf`.

Przykład:

```
Query OK, 0 rows affected (0.01 sec)
```

SET = Switch Embedded Teaming: zespół kart sieciowych zbudowany w przełączniku wirtualnym Hyper-V; łączy dwie karty w jedno łącze bez ustawień po stronie przełącznika fizycznego, więc bez

LACP

TYLKO ODCZYT

Sprawdzenie, które parametry różnią się od wartości domyślnej: performance_schema.variables_info

```
SELECT VARIABLE_NAME, VARIABLE_SOURCE, VARIABLE_PATH
FROM performance_schema.variables_info
WHERE VARIABLE_SOURCE != 'COMPILED';
```

Sprawdzamy, które parametry zostały jawnie ustawione — w pliku konfiguracyjnym, wierszu poleceń albo poleceniem `SET PERSIST` — zamiast pozostać na wartości wkompilowanej w binarkę, przydatne przy audycie konfiguracji przejętego serwera.

Przykład:

```
+--------------------------+---------------+-----------------------+
| VARIABLE_NAME            | VARIABLE_SOURCE | VARIABLE_PATH         |
+--------------------------+---------------+-----------------------+
| innodb_buffer_pool_size  | CONFIG          | /etc/mysql/my.cnf     |
| max_connections          | GLOBAL          |                       |
+--------------------------+---------------+-----------------------+
2 rows in set (0.00 sec)
```

SET = Switch Embedded Teaming: zespół kart sieciowych zbudowany w przełączniku wirtualnym Hyper-V; łączy dwie karty w jedno łącze bez ustawień po stronie przełącznika fizycznego, więc bez LACP

Z życia wzięte

Z ŻYCIA WZIĘTE

Sklep B2B na my01 zwalnia zauważalnie w listopadzie, kiedy ruch zamówień hurtowych rośnie przed świętami. Sprawdzamy `innodb_buffer_pool_size` i widzimy wartość ustawioną jeszcze przy pierwszej instalacji, dużo mniejszą niż dzisiejszy rozmiar bazy `sklep_b2b`. Podnosimy bufor do siedemdziesięciu procent dostępnego RAM-u serwera i zapisujemy zmianę przez `SET PERSIST`, żeby przetrwała najbliższy restart.

Gdzie leży plik i w jakiej kolejności jest czytany

> **Pojęcie: my.cnf.** Plik tekstowy w formacie INI, czytany przez `mysqld` przy starcie — na Linuksie zwykle `my.ini` na Windows. Serwer nie czyta jednego pliku, tylko sekwencję lokalizacji po kolei, a każda kolejna nadpisuje wartości ustawione we wcześniejszej. Kolejność (typowa dla dystrybucji opartych na RHEL/Debian, sprawdzana poleceniem `mysqld --verbose --help` na końcowej liście `Default options`):

```
/etc/my.cnf
/etc/mysql/my.cnf
/etc/mysql/mysql.conf.d/*.cnf   (albo mariadb.conf.d na MariaDB)
~/.my.cnf
```

Sprawdzamy dokładną, obowiązującą na danym hoście kolejność poleceniem:

```
mysqld --verbose --help | grep -A 5 "Default options"
```

Rozbicie na `/etc/mysql/mysql.conf.d/*.cnf` zamiast jednego pliku jest konwencją pakietów dystrybucyjnych — pozwala trzymać ustawienia sieciowe w jednym pliku, a limity zasobów w drugim, bez edytowania wspólnego monolitu.

Sekcje pliku

Plik dzieli się na sekcje w nawiasach kwadratowych, z których każda dotyczy innego programu czytającego ten sam plik:

```
[mysqld]
bind-address = 0.0.0.0
port = 3306

[client]
port = 3306

[mysqldump]
quick
```

```
max_allowed_packet = 256M
```

Sekcja `[mysqld]` konfiguruje sam serwer i jest tą, którą edytujemy najczęściej. Sekcja `[client]` obowiązuje klienta `mysql` i pozostałe narzędzia CLI, opisana też na stronie Klient mysql i połączenie. Osobne sekcje (`[mysqldump]`, `[mysqld_safe]`) pozwalają nadpisać domyślne parametry konkretnego narzędzia bez wpływu na resztę.

Najważniejsze parametry

bind-address — adres IP, na którym `mysqld` nasłuchuje połączeń TCP. Wartość `127.0.0.1` (domyślna w wielu dystrybucjach ze względów bezpieczeństwa) ogranicza dostęp do połączeń lokalnych, `0.0.0.0` otwiera nasłuch na wszystkich interfejsach sieciowych hosta.

port — numer portu TCP, domyślnie `3306`.

datadir — katalog danych opisany szerzej na stronie Architektura MySQL i MariaDB: silniki składowania, pliki, procesy, domyślnie `/var/lib/mysql`.

max_connections — maksymalna liczba równoczesnych połączeń klienckich. Przekroczenie limitu kończy się błędem `Too many connections` dla nowych połączeń, podczas gdy istniejące działają dalej — jedno dodatkowe połączenie zawsze zostaje zarezerwowane dla konta z uprawnieniem `SUPER`, żeby dało się zalogować i zdiagnozować problem nawet przy wyczerpanym limicie.

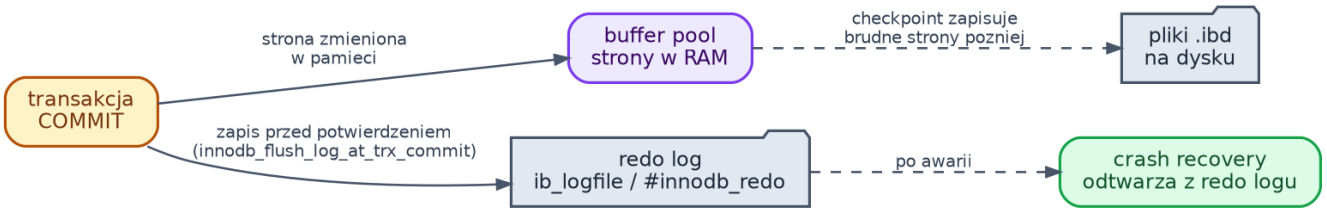
innodb_buffer_pool_size — rozmiar bufora w pamięci RAM, w którym InnoDB trzyma strony danych i indeksów odczytane z dysku. Najważniejszy pojedynczy parametr wydajnościowy całej instalacji — na serwerze dedykowanym pod bazę danych zaleca się 60–75% dostępnego RAM-u. Zbyt mały bufor oznacza, że InnoDB ciągle czyta te same strony z dysku zamiast z pamięci.

innodb_log_file_size (do 8.0) / innodb_redo_log_capacity (od 8.0.30) — rozmiar redo logu opisanego na stronie Architektura MySQL i MariaDB: silniki składowania, pliki, procesy. Większy redo log pozwala InnoDB opóźniać zapis brudnych stron na dysk i zbierać więcej zmian między checkpointami, kosztem dłuższego czasu odtwarzania po awarii.

innodb_flush_log_at_trx_commit — jak agresywnie InnoDB zrzuca redo log na dysk przy każdym `COMMIT`. Trzy wartości, kompromis między trwałością a wydajnością:

Wartość	Zachowanie	Ryzyko przy awarii
1 (domyślna)	zapis i `fsync` na dysk przy każdym commicie	brak utraty potwierdzonych transakcji
2	zapis do cache systemu operacyjnego przy każdym commicie, `fsync` raz na sekundę	utrata do 1 sekundy transakcji przy awarii systemu operacyjnego, nie samego MySQL

Wartość	Zachowanie	Ryzyko przy awarii
0	zapis i `fsync` raz na sekundę	utrata do 1 sekundy transakcji przy awarii samego procesu MySQL



COMMIT zapisuje strone w buffer poolu i w redo logu, checkpoint zrzuca brudne strony do plikow .ibd pozniej, redo log sluzy do crash recovery (kliknięcie otwiera pełny rozmiar)

log_bin — włącza binary log opisany szerzej na stronie Logi binarne: archiwizacja i czyszczenie, niezbędny do replikacji i do odtwarzania do punktu w czasie.

expire_logs_days (do 8.0) / binlog_expire_logs_seconds (od 8.0) — po ilu dniach (sekundach) serwer sam usuwa stare pliki binloga. Parametr `expire_logs_days` jest przestarzały od MySQL 8.0 i usunięty w nowszych wersjach na rzecz jednostki w sekundach.

slow_query_log — włącza zapisywanie zapytań przekraczających próg `long_query_time` do osobnego logu, opisane na stronie Sesje, długie zapytania i blokady.

sql_mode — zestaw flag zmieniających ścisłość walidacji SQL po stronie serwera. Dwie flagi mają praktyczne znaczenie przy migracji ze starszych wersji:

> **Pojęcie: ONLY_FULL_GROUP_BY.** Wymusza, żeby każda kolumna w `SELECT` przy użyciu `GROUP BY` była albo w klauzuli `GROUP BY`, albo wewnątrz funkcji agregującej — bez tej flagi MySQL do wersji 5.7 (i nadal, jeśli flagę świadomie wyłączymy) pozwalał zwrócić dowolną, niezdefiniowaną wartość kolumny spoza `GROUP BY`, co dawało niedeterministyczne wyniki. Domyślnie włączona od MySQL 5.7.5.

> **Pojęcie: STRICT_TRANS_TABLES.** Zamienia ciche obcinanie i konwersję nieprawidłowych wartości (na przykład wstawienie tekstu do kolumny liczbowej) na twardy błąd zamiast ostrzeżenia. Bez tej flagi `INSERT` z wartością zbyt długą dla kolumny `VARCHAR` po cichu ją obcina; z flagą kończy się błędem `Data too long for column`. Domyślnie włączona od MySQL 5.7.

Odczyt i zmiana parametru w locie

> **Pojęcie: SHOW VARIABLES.** Polecenie zwracające aktualną wartość parametrów działającego serwera — niekoniecznie tę zapisaną w pliku `my.cnf`, tylko tę, która faktycznie obowiązuje w tej chwili, po uwzględnieniu ewentualnych zmian wykonanych `SET GLOBAL` od czasu ostatniego startu.

```
SHOW VARIABLES LIKE 'NazwaParametru';
```

> **Pojęcie: SET GLOBAL.** Zmienia wartość parametru dla całej instancji, dla wszystkich przyszłych połączeń, ale wyłącznie w pamięci działającego procesu — restart serwera przywraca wartość z pliku `my.cnf`, jeśli tam ją zapisano, albo wartość domyślną wkompilowaną w binarkę, jeśli nie.

```
SET GLOBAL NazwaParametru = WartoscParametru;
```

> **Pojęcie: SET PERSIST.** Nowość MySQL 8.0 (brak w MariaDB) — działa jak `SET GLOBAL`, ale dodatkowo zapisuje wartość do pliku `mysqld-auto.cnf` wewnątrz `datadir`, skąd serwer odczytuje ją automatycznie przy każdym kolejnym starcie, bez ręcznej edycji `my.cnf`.

```
SET PERSIST NazwaParametru = WartoscParametru;
```

Nie każdy parametr da się zmienić w locie — kolumna `Dynamic` w tabeli `information_schema.SYSTEM_VARIABLES` (albo `performance_schema.variables_info` na nowszych wersjach) mówi, czy dany parametr jest `dynamic` czy tylko do odczytu w trakcie działania. Parametry wymagające restartu to typowo te wpływające na strukturę pamięci alokowaną raz przy starcie: `innodb_buffer_pool_size` w starszych wersjach (od 5.7.5 można go zmieniać w locie, ale ze skutkami ubocznymi na wydajność w trakcie zmiany), `innodb_log_file_size`, `port`, `datadir` — te ostatnie dwa nie zmienią się nigdy bez restartu, bo dotyczą samego procesu nasłuchującego i otwartego katalogu danych.

Częste błędy

Zmiana SET GLOBAL bez zapisania jej też w pliku. Parametr zmieniony wyłącznie w locie wraca do poprzedniej wartości po najbliższym restarcie serwera — który przy aktualizacji pakietu albo awarii zasilania przychodzi bez zapowiedzi. Trwałe zmiany na MySQL 8.0 wygodniej robić od razu przez `SET PERSIST`.

Ustawienie innodb_buffer_pool_size większego niż dostępny RAM. Serwer wystartuje, ale system zacznie swapować, gdy bufor rzeczywiście zapełni się danymi — objaw to gwałtowny spadek wydajności bez widocznego błędu w logu MySQL, bo problem leży w systemie operacyjnym, nie w samym silniku.

Powiązane: Architektura MySQL i MariaDB: silniki składowania, pliki, procesy · Klient mysql i połączenie · Logi binarne: archiwizacja i czyszczenie

Źródła: dokumentacja MySQL 8.0, rozdział „Server System Variables" i „The Server Configuration File" · dokumentacja MariaDB, rozdział „Server System Variables"

Revision #42
Created 2026-09-25 00:00:31 UTC by Tomasz Konieczny
Updated 2026-10-26 14:08:18 UTC by Tomasz Konieczny