

DML — INSERT, UPDATE i DELETE

Czas: 2 × 45 min

praca indywidualna

Cel: Dodawanie, zmienianie i usuwanie danych poleceniami INSERT, UPDATE i DELETE oraz nawyk bezpiecznej pracy: najpierw SELECT z tym samym warunkiem, a przy ryzykownych zmianach — transakcja z możliwością ROLLBACK.

Po zajęciach potrafisz:

- wstawia wiele wierszy jednym poleceniem INSERT i kopiuje dane poleceniem INSERT ... SELECT
- zmienia dane poleceniem UPDATE z warunkiem, także z wyrażeniem (podwyżka o 10%)
- usuwa wybrane wiersze poleceniem DELETE i przewiduje wpływ kluczy obcych
- sprawdza zakres zmian zapytaniem SELECT przed UPDATE i DELETE
- używa transakcji (START TRANSACTION, COMMIT, ROLLBACK) jako zabezpieczenia

Sprzęt i materiały

- komputer z XAMPP
- plik `sklep.sql` z folderu Pliki_do_zadan

Oprogramowanie

- **Konsola mysql (klient MariaDB)** — transakcje wykonuj w konsoli — w phpMyAdmin każde „Wykonaj” to osobne połączenie
mariadb.com/kb/en/mariadb-command-line-client
- **phpMyAdmin** — podgląd danych
phpmyadmin.net

Przygotowanie przed zajęciami

- Zaimportuj ponownie `sklep.sql`, aby zacząć od danych wzorcowych — wyniki zależą od kolejności poleceń.

Bezpieczeństwo

- Zasada na całe życie: przed każdym UPDATE i DELETE wykonaj SELECT z dokładnie tym samym WHERE i sprawdź liczbę wierszy.
- UPDATE lub DELETE bez WHERE zmienia całą tabelę. Na serwerze produkcyjnym przed zmianą danych wykonaj kopię zapasową albo pracuj w transakcji.
- Po zajęciach zaimportuj `sklep.sql` ponownie, żeby kolejne zadania zaczynały się od tych samych danych.

Przebieg zadania

1 INSERT — nowe dane

Zaloguj się do konsoli (`mysql -u root --default-character-set=utf8mb4`), wybierz bazę sklep. Polecenia zapisuj w pliku `zad12_nazwisko.sql`.

1. Z1 — jednym poleceniem INSERT dodaj trzech klientów: Oskar Bielecki (oskar.b@example.pl, Lublin), Hanna Czarnecka (hanna.cz@example.pl, Kraków), Filip Duda (filip.duda@example.pl, bez miasta); data rejestracji 2026-10-20. Wymień kolumny jawnie, bez id_klienta. Jakie identyfikatory dostali?
2. Z2 — dodaj kategorię Sieci i sprawdź jej identyfikator (`SELECT LAST_INSERT_ID()`). Jednym poleceniem dodaj do niej dwa produkty: Router Wi-Fi 7 (899.00 zł, 6 szt.) i Switch 8-portowy (159.00 zł, 15 szt.), data dodania 2026-10-20.
3. Z3 — utwórz tabelę archiwum_zamowien o strukturze tabeli zamowienia (`CREATE TABLE ... LIKE`) i skopiuj do niej poleceniem `INSERT ... SELECT` wszystkie zamówienia o statusie wyslane. Ile wierszy skopiowano?

2 UPDATE — zmiany z warunkiem

Przy każdym poleceniu: najpierw SELECT z tym samym WHERE (zapisz liczbę wierszy), potem UPDATE, potem sprawdzenie.

1. Z4 — podnieś o 10% ceny wszystkich produktów z kategorii Akcesoria (id 3), zaokrąglając do grosza: `SET cena = ROUND(cena * 1.10, 2)`. Zapisz ceny przed i po zmianie.
2. Z5 — zamówienia o statusie nowe złożone przed 1 lipca 2026 r. zmień na anulowane (dwa warunki połączone AND).
3. Z6 — klientom bez podanego miasta wpisz miasto Lublin (pamiętaj: porównanie z NULL wymaga `IS NULL`). Ilu klientów to objęło i dlaczego tyłu?
4. Z7 — produktom o stanie 0 zwiększ stan o 10 sztuk (`SET stan = stan + 10`).

3 DELETE — usuwanie

1. Z8 — usuń klientów dodanych w Z1 (warunek na id_klienta). Najpierw SELECT!
2. Z9 — spróbuj usunąć klienta nr 2. Zapisz błąd i wyjaśnij, dlaczego klienta z zamówieniami nie da się usunąć (zadanie 11).
3. Z10 — usuń tabelę archiwum_zamowien (to DDL, nie DML — czym różni się od DELETE FROM archiwum_zamowien?).

4 Transakcje jako zabezpieczenie

Domyślnie każde polecenie jest od razu zatwierdzane (autocommit = 1). Po START TRANSACTION zmiany są widoczne tylko w twojej sesji, dopóki nie wykonasz COMMIT; ROLLBACK je cofa.

```
-- Z11: „pomyłka” pod kontrolą
START TRANSACTION;
DELETE FROM pozycje_zamowienia;          -- zapomniany WHERE!
SELECT COUNT(*) FROM pozycje_zamowienia;
ROLLBACK;
SELECT COUNT(*) FROM pozycje_zamowienia;
```

1. Sprawdź `SELECT @@autocommit;`. Wykonaj Z11 z ramki i zapisz obie liczby.
2. Z12 — w jednej transakcji zapisz sprzedaż: nowe zamówienie klientki nr 3 (data 2026-10-20 12:00:00, status nowe), pozycja: produkt 10, 2 sztuki po 699.00 (id zamówienia weź z `LAST_INSERT_ID()`), zmniejszenie stanu produktu 10 o 2. Zatwierdź COMMIT i sprawdź wszystkie trzy zmiany.
3. Z13 — rozpocznij transakcję, zwiększ stan produktu 1 o 5, a potem spróbuj zmniejszyć stan produktu 2 o 5 (na stanie są 4 sztuki). Zapisz błąd. Czy pierwsza zmiana nadal obowiązuje w transakcji? Wykonaj ROLLBACK i sprawdź stan obu produktów.
4. Z14 — wyloguj się i zaloguj ponownie z opcją `--safe-updates`. Spróbuj wykonać `UPDATE produkty SET cena = cena * 2;`. Zapisz komunikat — ten tryb chroni przed UPDATE i DELETE bez warunku na kluczu.

W sprawozdaniu umieść

- ☐ plik `zad12_nazwisko.sql` z poleceniami Z1–Z14 (z zapytaniami kontrolnymi SELECT)
- ☐ tabela poleceń z liczbą wierszy w SELECT kontrolnym i w poleceniu zmieniającym
- ☐ ceny akcesoriów przed i po podwyżce (Z4)
- ☐ wyniki eksperymentów z transakcjami (Z11–Z13) i komunikat trybu safe-updates (Z14)
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Dlaczego w poleceniu INSERT warto wymieniać nazwy kolumn, nawet gdy podajemy wartości dla wszystkich?
2. Co się stanie po wykonaniu `UPDATE produkty SET cena = cena * 1.10;` bez WHERE? Jak można to odwrócić, a jak — zabezpieczyć się przed tym?
3. Dlaczego warunek `WHERE miasto = NULL` nie znajduje żadnego wiersza?
4. Dlaczego sprzedaż (zamówienie, pozycja, zmiana stanu) powinna być jedną transakcją?

Kryteria oceny

Kryterium	Punkty
INSERT wielu wierszy i INSERT ... SELECT (Z1–Z3)	4
UPDATE z warunkami, poprzedzone SELECT (Z4–Z7)	5
DELETE i rozumienie kluczy obcych (Z8–Z10)	3
Transakcje i safe-updates (Z11–Z14) z wnioskami	5
Odpowiedzi na pytania kontrolne	3
Razem	20

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).