

Wyzwalacze walidujące i zdarzenia (EVENT)

Czas: 2 × 45 min

praca indywidualna lub w parach

Cel: Nauczenie się pilnowania reguł biznesowych wyzwalaczami (stan magazynu, dozwolone zmiany statusu, limit obniżki) oraz automatyzowania zadań okresowych zdarzeniami serwera.

Po zajęciach potrafisz:

- pisze wyzwalacze BEFORE, które blokują niedozwoloną zmianę poleceniem SIGNAL
- utrzymuje stan magazynu wyzwalaczami INSERT, UPDATE i DELETE na pozycjach zamówień
- włącza harmonogram zdarzeń i tworzy zdarzenia jednorazowe i cykliczne
- projektuje archiwizację starych danych w procedurze uruchamianej przez zdarzenie

Sprzęt i materiały

- komputer w pracowni z XAMPP (MariaDB 10.4) lub MySQL 8.4
- baza sklep zaimportowana z pliku `sklep.sql` (folder Pliki_do_zadan)

Oprogramowanie

- **XAMPP (Apache + MariaDB + phpMyAdmin)** — panel sterowania XAMPP, konsola: przycisk Shell albo `C:\xampp\mysql\bin\mysql.exe -u root`
apachefriends.org/download.html
- **phpMyAdmin** — zakładka SQL do uruchamiania zapytań; wchodzi w skład XAMPP
localhost/phpmyadmin
- **VS Code lub Notepad++** — do pisania pliku `.sql` z zapytaniem
code.visualstudio.com

Bezpieczeństwo

- Zaimportuj świeżą bazę sklep przed zadaniem i po nim — wyzwalacze zmieniają stany magazynowe, a archiwizacja usuwa zamówienia.
- Zdarzenie usuwające dane najpierw sprawdź ręcznie (CALL procedury) na kopii bazy. Po zajęciach usuń zdarzenia testowe i wyłącz harmonogram.
- `SET GLOBAL` zmienia ustawienie całego serwera — wykonuj je tylko na własnym serwerze XAMPP, nie na serwerze szkolnym.

Przebieg zadania

1 Stan magazynu pilnowany przez wyzwalacze

Wyzwalacz BEFORE INSERT na pozycje_zamowienia sprawdza stan i zmniejsza go. Błąd SIGNAL w wyzwalaczu przerywa całe polecenie.

```
DELIMITER //
DROP TRIGGER IF EXISTS trg_pozycje_stan //
CREATE TRIGGER trg_pozycje_stan
BEFORE INSERT ON pozycje_zamowienia
FOR EACH ROW
BEGIN
    DECLARE v_stan INT;
    SELECT stan INTO v_stan FROM produkty WHERE id_produkту = NEW.id_produkту;
    IF v_stan < NEW.ilosc THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Brak towaru na stanie';
    END IF;
    UPDATE produkty SET stan = stan - NEW.ilosc WHERE id_produkту = NEW.id_produkту;
END //
DELIMITER ;
```

1. Zaimportuj `sklep.sql`. Utwórz plik `r2_12_nazwisko.sql`.
2. Z1. Utwórz wyzwalacz `trg_pozycje_stan` (kod powyżej). Dodaj do zamówienia 9 pozycję: produkt 7, 5 sztuk, cena 129.00 i sprawdź stan produktu 7. Potem spróbuj dodać do zamówienia 11 produkt 3 (stan 0).
3. Z2. Jednym poleceniem INSERT spróbuj dodać do zamówienia 11 dwie pozycje: (produkt 7, 1 szt.) i (produkt 3, 1 szt.). Sprawdź stan produktu 7 i liczbę pozycji zamówienia 11. Czy pierwsza pozycja została dodana?
4. Z3. Utwórz wyzwalacz `trg_pozycje_zwrot` AFTER DELETE, który oddaje na stan ilość z usuniętej pozycji. Usuń pozycję z Z1 i sprawdź stan produktu 7.
5. Z4. Utwórz wyzwalacz `trg_pozycje_zmiana` BEFORE UPDATE: zabrania zmiany produktu w pozycji (SIGNAL), a przy zmianie ilości koryguje stan o różnicę i blokuje zwiększenie ponad stan. Uwaga: kolumna ilosc jest UNSIGNED — różnicę liczb w zmiennej INT: `CAST(NEW.ilosc AS SIGNED) - CAST(OLD.ilosc AS SIGNED)`. Testy: zamówienie 12, produkt 7 → ilość 3; zamówienie 8, produkt 9 → ilość 10, potem 1; zmiana produktu 9 na 1 w zamówieniu 8.

2 Reguły dla statusu i ceny

1. Z5. Utwórz wyzwalacz `trg_zamowienia_status` BEFORE UPDATE na zamowienia, który nie pozwala zmienić statusu zamówienia wysłanego lub anulowanego. Sprawdź zmianę zamówienia 1 na 'nowe' i zamówienia 9 na 'opłacone'.
2. Z6. Utwórz wyzwalacz `trg_produkty_obnizka` BEFORE UPDATE na produkty, który nie odrzuca zmiany, ale ogranicza obniżkę ceny do 50% (ustawia NEW.cena). Zmień cenę produktu 2 na 100.00, a potem na 3000.00. Jakie ceny zapisano?
3. Z7. Zapisz stan produktu 11 i usuń zamówienie 12 (pozycje usuwa kaskada). Czy wyzwalacz `trg_pozycje_zwrot` oddał towar na stan? Porównaj z Z10 zadania 11.

3 Zdarzenia (EVENT)

Zdarzenia uruchamia wątek harmonogramu. W XAMPP (MariaDB) jest on domyślnie wyłączony, w MySQL 8.4 — włączony. Na stałe włącza go wpis `event_scheduler=ON` w sekcji [mysqld] pliku my.ini.

1. Z8. Sprawdź `SHOW VARIABLES LIKE 'event_scheduler'`. Utwórz tabelę `log_zdarzen` (id, kiedy DATETIME, opis VARCHAR(100)) i zdarzenie `ev_test` wykonywane co 10 sekund, które dopisuje wiersz do `log_zdarzen`. Wyświetl `SHOW EVENTS`. Włącz harmonogram `SET GLOBAL event_scheduler = ON;`, odczekaj ok. 30 sekund i policz wiersze.
2. Z9. Wyłącz zdarzenie (`ALTER EVENT ev_test DISABLE;`), odczekaj 10 sekund i sprawdź, że wierszy nie przybywa.
3. Z10. Utwórz tabele archiwum i procedurę `archiwizuj_anulowane(IN p_dni, OUT p_file)` (kod w ramce poniżej). Najpierw sprawdź ją ręcznie: `CALL` z `p_dni = 30`, potem zawartość obu tabel archiwum i liczbę zamówień. Wywołaj ją drugi raz — ile zamówień przeniesiono?
4. Z11. Utwórz zdarzenie `ev_archiwizacja` wykonywane codziennie o 2:00, zaczynając od jutra: `ON SCHEDULE EVERY 1 DAY STARTS CURRENT_DATE + INTERVAL 1 DAY + INTERVAL 2 HOUR DO CALL archiwizuj_anulowane(30, @ile);`. Utwórz też zdarzenie jednorazowe `ev_sprzatanie` (`AT CURRENT_TIMESTAMP + INTERVAL 5 SECOND`), które usuwa wiersze z `log_zdarzen`. Wypisz zdarzenia z `information_schema.EVENTS` (typ, harmonogram, status). Po 10 sekundach sprawdź `log_zdarzen` i listę zdarzeń.
5. Z12. Usuń zdarzenia `ev_test` i `ev_archiwizacja` (`DROP EVENT`), wyłącz harmonogram (`SET GLOBAL event_scheduler = OFF;`) i zaimportuj ponownie `sklep.sql`.

4 Procedura archiwizacji do Z10

```
CREATE TABLE zamowienia_archiwum LIKE zamowienia;
CREATE TABLE pozycje_archiwum LIKE pozycje_zamowienia;
DELIMITER //
DROP PROCEDURE IF EXISTS archiwizuj_anulowane //
CREATE PROCEDURE archiwizuj_anulowane(IN p_dni INT, OUT p_file INT)
BEGIN
    DECLARE EXIT HANDLER FOR SQLEXCEPTION BEGIN ROLLBACK; RESIGNAL; END;
    START TRANSACTION;
    INSERT INTO zamowienia_archiwum
        SELECT * FROM zamowienia
        WHERE status = 'anulowane' AND data_zamowienia < NOW() - INTERVAL p_dni DAY;
    SET p_file = ROW_COUNT();
    INSERT INTO pozycje_archiwum
        SELECT pz.* FROM pozycje_zamowienia pz
        JOIN zamowienia_archiwum za ON za.id_zamowienia = pz.id_zamowienia;
    DELETE FROM zamowienia
    WHERE id_zamowienia IN (SELECT id_zamowienia FROM zamowienia_archiwum);
    COMMIT;
END //
DELIMITER ;
```

W sprawozdaniu umieść

- ☐ plik `.sql` z wyzwalaczami, procedurą, zdarzeniami i testami Z1–Z12
- ☐ tabela testów wyzwalaczy (polecenie, oczekiwany wynik, wynik, stan magazynu)
- ☐ zrzuty ekranu `SHOW EVENTS` / `information_schema.EVENTS` i wyniku archiwizacji
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Dlaczego w Z2 nie została dodana także pierwsza, poprawna pozycja?

2. Czym różni się wyzwalacz, który blokuje zmianę (Z5), od wyzwalacza, który ją poprawia (Z6)?
Który sposób jest bardziej przejrzysty dla użytkownika aplikacji?
3. Co się stało w Z7 i jak można to naprawić?
4. Co stanie się ze zdarzeniem jednorazowym po wykonaniu (ON COMPLETION NOT PRESERVE), a co, gdy w chwili planowanego uruchomienia harmonogram jest wyłączony?

Kryteria oceny

Kryterium	Punkty
Wyzwalacze stanu magazynu z testami (Z1–Z4)	6
Wyzwalacze reguł statusu i ceny, analiza kaskady (Z5–Z7)	4
Zdarzenia cykliczne i jednorazowe, archiwizacja (Z8–Z12)	7
Odpowiedzi na pytania kontrolne	3
Razem	20

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