

Funkcje składowane i pętle

Czas: 2 × 45 min

praca indywidualna lub w parach

Cel: Nauczenie się tworzenia własnych funkcji zwracających wartość, używania ich w zapytaniach oraz programowania pętli WHILE, REPEAT i LOOP.

Po zajęciach potrafisz:

- tworzy funkcję z RETURNS i RETURN oraz właściwą charakterystyką (DETERMINISTIC, NO SQL, READS SQL DATA)
- używa własnych funkcji w SELECT, WHERE, GROUP BY i ORDER BY
- programuje pętle WHILE, REPEAT ... UNTIL i LOOP z LEAVE i ITERATE
- odróżnia funkcję od procedury i rozpoznaje typowe błędy (przepętlenie, porównywanie napisów)

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 pisanie pliku .sql z zapytaniami
code.visualstudio.com

Przygotowanie przed zajęciami

- Powtórz z zadania 8 zapis DELIMITER i różnicę między parametrem a zmienną lokalną.

Bezpieczeństwo

- Każda pętla musi mieć warunek zakończenia, który na pewno zostanie spełniony. Nieskończona pętla w funkcji wywołanej dla każdego wiersza zablokuje serwer — w konsoli przerwiesz ją Ctrl+C, w phpMyAdmin trzeba zatrzymać zapytanie (KILL) albo zrestartować MySQL w panelu XAMPP.
- Zadanie nie zmienia danych — tworzysz tylko funkcje.

Przebieg zadania

1 Funkcje bez dostępu do tabel

Funkcja zwraca jedną wartość i używa się jej jak wbudowanej funkcji SQL. Charakterystyka po RETURNS: DETERMINISTIC (dla tych samych argumentów zawsze ten sam wynik), NO SQL (nie czyta tabel) albo READS SQL DATA (czyta tabele).

```
DELIMITER //
DROP FUNCTION IF EXISTS brutto //
CREATE FUNCTION brutto(p_netto DECIMAL(10,2)) RETURNS DECIMAL(10,2)
DETERMINISTIC NO SQL
RETURN ROUND(p_netto * 1.23, 2) //
DELIMITER ;
```

1. Zaimportuj `sklep.sql`. Utwórz plik `r2_09_nazwisko.sql`.
2. Z1. Utwórz funkcję `brutto` z ramki. Wypisz laptopy z ceną netto i brutto.
3. Z2. Utwórz funkcję `z1(p_kwota DECIMAL(12,2))` zwracającą tekst w formacie „3 299,00 zł” (spacja tysięcy, przecinek dziesiętny). Wskazówka: `FORMAT(x, 2)` daje „3,299.00” — zamień znaki funkcją `REPLACE` w odpowiedniej kolejności. Sprawdź dla 3299, 1234567.5 i 0.5 oraz w zapytaniu z Z1.
4. Z3. Utwórz funkcję `inicjaly(p_imie, p_nazwisko)` zwracającą np. „A.N.” (wielkie litery) i wypisz klientów w postaci „A.N. Nowak”.

2 Funkcje czytające dane

1. Z4. Utwórz funkcję `wartosc_zamowienia(p_zam INT) RETURNS DECIMAL(10,2) READS SQL DATA`, która zwraca sumę `ilosc × cena_jedn` pozycji zamówienia (0, gdy zamówienie nie istnieje). Wypisz 3 najdroższe zamówienia z wartością sformatowaną funkcją `z1`. Policz funkcją łączną wartość zamówień nieanulowanych (powinno być 31639.00).
2. Z5. Utwórz funkcję `segment_klienta(p_klient INT)` zwracającą VIP (od 5000 zł), stały (od 1000 zł), okazjonalny (powyżej 0) lub „bez zakupów” — według wartości zamówień nieanulowanych (porównaj z Z7 zadania 4). Policz klientów w każdym segmencie (`GROUP BY` po wyniku funkcji).
3. Z6. Wypisz nazwiska klientów VIP warunkiem `WHERE segment_klienta(id_klienta) = 'VIP'`. Jeśli pojawi się błąd 1267 (Illegal mix of collations), zapisz go i popraw warunek, dopisując po napisie `COLLATE utf8mb4_polish_ci`. Wyjaśnij przyczynę.

3 Pętle

WHILE warunek DO ... END WHILE (sprawdza przed krokiem), REPEAT ... UNTIL warunek END REPEAT (po kroku), etykieta: LOOP ... END LOOP z LEAVE (wyjście) i ITERATE (następny obrót).

1. Z7. Utwórz funkcję `silnia(p_n INT) RETURNS BIGINT` z pętlą WHILE. Sprawdź `silnia(0)`, `silnia(5)`, `silnia(10)`, `silnia(20)`, a potem osobno `silnia(21)`. Zapisz błąd i wyjaśnij.
2. Z8. Utwórz funkcję `suma_cyfr(p_liczba INT UNSIGNED)` z pętlą REPEAT (`MOD 10` i `DIV 10`). Sprawdź dla 2026, 0 i 99999. Dlaczego REPEAT, a nie WHILE, dobrze obsługuje liczbę 0?
3. Z9. Utwórz funkcję `parzyste(p_do INT) RETURNS VARCHAR(255)` z pętlą LOOP, która zwraca liczby parzyste od 2 do `p_do` oddzielone spacją (LEAVE po przekroczeniu `p_do`, ITERATE dla liczb nieparzystych). Sprawdź `parzyste(10)` i `parzyste(1)`.
4. Z10. Utwórz funkcję `dni_robocze(p_od DATE, p_do DATE)` liczącą dni od poniedziałku do piątku w przedziale (pętla po kolejnych dniach). Sprawdź wrzesień i październik 2026 (porównaj z Z6 zadania 5) oraz przedział z datą końcową wcześniejszą niż początkowa.

4 Funkcja czy procedura

1. Z11. Spróbuj wywołać funkcję jak procedurę: `CALL brutto(100);`. Zapisz błąd.
2. Z12. Wypisz z `information_schema.ROUTINES` nazwy, typ, `IS_DETERMINISTIC` i `SQL_DATA_ACCESS` swoich funkcji. Wyświetl `SHOW FUNCTION STATUS WHERE Db = 'sklep';` i usuń funkcję `parzyste` (`DROP FUNCTION`).

W sprawozdaniu umieść

- ☐ plik `.sql` z definicjami funkcji i zapytaniem Z1–Z12
- ☐ zrzuty ekranu wyników Z2, Z4, Z5 i Z10
- ☐ opis błędów z Z6, Z7 i Z11 z wyjaśnieniem
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Podaj trzy różnice między funkcją a procedurą składowaną.
2. Kiedy funkcję wolno oznaczyć jako `DETERMINISTIC`? Czy `wartosc_zamowienia` jest deterministyczna?
3. Dlaczego funkcja wywoływana w `WHERE` dla każdego wiersza może spowolnić zapytanie na dużej tabeli?
4. Jaką największą silnię zmieści `BIGINT`, a jaką `BIGINT UNSIGNED`? Jak można by policzyć większe?

Kryteria oceny

Kryterium	Punkty
Funkcje bez dostępu do tabel (Z1–Z3)	4
Funkcje czytające dane i ich użycie w zapytaniach (Z4–Z6)	5
Pętle <code>WHILE</code> , <code>REPEAT</code> i <code>LOOP</code> (Z7–Z10)	6
Funkcja a procedura, katalog funkcji (Z11–Z12)	2
Odpowiedzi na pytania kontrolne	3
Razem	20

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