

Projekt końcowy cz. 1 — baza danych z logiką

Czas: 2 × 45 min

praca w zespołach 2–3-osobowych

Cel: Zaprojektowanie i wykonanie w zespole bazy danych dla wybranej dziedziny wraz z logiką po stronie serwera (widoki, procedura z transakcją, wyzwalacz audytu), indeksami i kontem aplikacji z minimalnymi uprawnieniami — jako podstawy aplikacji PHP w części 2.

Po zajęciach potrafisz:

- analizuje wymagania dziedziny i projektuje schemat ERD w 3NF z kluczami i ograniczeniami
- pisze skrypt tworzący bazę z danymi, który wykonuje się bez błędów na czystym serwerze
- tworzy widoki, procedurę składowaną z transakcją i walidacją oraz wyzwalacz audytu
- dobiera indeksy na podstawie zapytań i dokumentuje ich działanie w EXPLAIN
- tworzy konto aplikacji z minimalnymi uprawnieniami i sprawdza odmowę dostępu

Sprzęt i materiały

- komputer z XAMPP (MariaDB 10.4) lub MySQL 8.4 dla każdego członka zespołu
- wspólny folder zespołu (dysk sieciowy, pendrive lub repozytorium Git)

Oprogramowanie

- **XAMPP (MariaDB, phpMyAdmin)** — konsola mysql do testów kont i procedur apachefriends.org
- **draw.io (diagrams.net)** — diagram ERD (biblioteka Entity Relation) app.diagrams.net
- **VS Code** — plik projekt.sql code.visualstudio.com
- **Git (opcjonalnie)** — wspólna praca nad plikami zespołu git-scm.com

Przygotowanie przed zajęciami

- Zespół wybiera dziedzinę z listy w kroku 1 (lub własną po akceptacji nauczyciela) i zapisuje 5 typowych czynności użytkownika systemu (np. „recepcja przyjmuje zlecenie”, „mechanik dodaje części”).
- Przejrzyj swoje rozwiązania z zadań 2.02 (widoki), 2.08–2.12 (procedury, wyzwalacze, transakcje, indeksy) i 1.16 (uprawnienia).

Bezpieczeństwo

- Baza zawiera wyłącznie dane fikcyjne — nie wpisujcie prawdziwych danych osobowych kolegów ani klientów.
- Hasło konta aplikacji wymyślcie na potrzeby projektu; nie używajcie haseł z innych serwisów. W dokumentacji zapiszcie je tylko w pliku konfiguracyjnym, nie w sprawozdaniu.
- Skrypt projekt.sql zaczyna się od DROP DATABASE IF EXISTS tylko dla bazy projektu — sprawdźcie nazwę, żeby nie usunąć innej bazy.
- Przed testem „na czystym serwerze” upewnijcie się, że macie aktualną kopię pliku projekt.sql we wspólnym folderze.

Przebieg zadania

1 Dziedzina i wymagania

Przykładowe dziedziny: warsztat samochodowy, przychodnia weterynaryjna, siłownia z karnetami, szkoła językowa (kursy i zapisy), hotel (pokoje i rezerwacje), biblioteka gier planszowych, firma cateringowa (menu i zamówienia).

1. Zapiszcie w sprawozdaniu nazwę dziedziny, podział ról w zespole i 5 czynności użytkowników.
2. Do każdej czynności dopiszcie, jakie dane są potrzebne i które z nich wymagają kontroli (np. brak towaru, termin zajęty, limit miejsc).

2 Projekt ERD i schemat

Wymagania minimalne: co najmniej 5 tabel (w tym jedna łącząca relację wiele-do-wielu) i tabela dziennika zmian; 3NF; klucze główne i obce z przemyślanym ON DELETE; NOT NULL, UNIQUE i co najmniej jedno ograniczenie CHECK; kodowanie utf8mb4_polish_ci.

1. Narysujcie diagram ERD w draw.io (encje, atrybuty, klucze, krotności) i zapiszcie jako `erd.png`.
2. Napiszcie w pliku `projekt.sql` polecenia CREATE DATABASE i CREATE TABLE. Przy każdym kluczu obcym dopiszcie komentarz, dlaczego wybraliście RESTRICT, CASCADE albo SET NULL.
3. Dodajcie dane testowe: co najmniej 8–10 wierszy w głównych tabelach, tak aby dało się przetestować wszystkie zapytania i błędy.

3 Logika w bazie

Każdy obiekt opatrzcie komentarzem: do czego służy i która czynność z kroku 1 go używa.

1. Utwórzcie co najmniej dwa widoki: raport z JOIN i GROUP BY (np. koszt zleceń, obłożenie zajęć) oraz widok dla aplikacji bez kolumn wrażliwych.
2. Napiszcie procedurę wykonującą główną operację biznesową (np. dodanie części do zlecenia, rezerwacja) z parametrami IN, transakcją, blokadą wierszy (SELECT ... FOR UPDATE) i walidacją przez SIGNAL SQLSTATE '45000' z czytelnym komunikatem; w razie błędu ROLLBACK.
3. Utwórzcie wyzwalacz audytu zapisujący do tabeli dziennika zmianę ważnej kolumny (OLD i NEW, użytkownik — funkcja USER(), czas).
4. Przetestujcie procedurę: jedno poprawne wywołanie i co najmniej dwa wywołania kończące się waszymi komunikatami błędów. Zapiszcie wyniki.

4 Wydajność i bezpieczeństwo

1. Wybierzcie dwa zapytania, które aplikacja będzie wykonywać najczęściej. Pokażcie EXPLAIN przed i po dodaniu indeksów (co najmniej jeden indeks złożony) i uzasadnijcie kolejność kolumn.
2. Utwórzcie konto aplikacji `<baza>_app @ localhost` z minimalnymi uprawnieniami: SELECT na widokach, tylko potrzebne SELECT/INSERT/UPDATE na tabelach, EXECUTE na procedurze; bez DELETE i bez dostępu do tabeli dziennika. Sprawdźcie, że zalogowany tym kontem użytkownik może wywołać procedurę, a DELETE kończy się błędem 1142.
3. Sprawdźcie, czy wyzwalacz zapisuje w dzienniku nazwę konta aplikacji, gdy zmiana pochodzi z tego konta.

5 Test na czystym serwerze i oddanie

1. Wykonajcie cały plik `projekt.sql` na komputerze innego członka zespołu (albo po usunięciu bazy) poleceniem `mysql -u root < projekt.sql` i poprawcie wszystkie błędy, aż plik wykona się w całości.
2. Dopiszcie na końcu pliku zakomentowane polecenia testowe (CALL, UPDATE wyzwalający audyt, EXPLAIN), żeby nauczyciel mógł je uruchomić.
3. Oddajcie folder zespołu: projekt.sql, erd.png, plik z poleceniami tworzącymi konto (bez hasła w treści sprawozdania) i sprawozdanie.

W sprawozdaniu umieść

- ☐ opis dziedziny, podział ról i 5 czynności użytkowników z kontrolami
- ☐ diagram ERD (erd.png) i uzasadnienie wyboru ON DELETE
- ☐ wyniki testów procedury (poprawne wywołanie i dwa błędy) oraz zawartość tabeli dziennika
- ☐ plany EXPLAIN przed i po indeksach z uzasadnieniem
- ☐ wynik SHOW GRANTS konta aplikacji i zrzut odmowy DELETE (błąd 1142)

Pytania kontrolne

1. Dlaczego główną operację biznesową umieściliście w procedurze z transakcją, a nie jako kilka osobnych zapytań w aplikacji?
2. Po co w procedurze SELECT ... FOR UPDATE? Co mogłoby się stać bez blokady przy dwóch równoczesnych użytkownikach?
3. Dlaczego konto aplikacji nie ma uprawnień DELETE ani dostępu do tabeli dziennika, a mimo to wyzwalacz może do niej pisać?
4. Czym różni się w wyzwalaczu wynik funkcji USER() od CURRENT_USER()?

Kryteria oceny

Kryterium	Punkty
Wymagania i ERD: kompletność, 3NF, poprawne krotności	3
Schemat i dane: klucze, ograniczenia, uzasadnione ON DELETE, plik wykonuje się na czystym serwerze	4
Widoki (2) i procedura z transakcją, blokadą i walidacją SIGNAL, przetestowana	5
Wyzwalacz audytu działający także dla konta aplikacji	2
Indeksy z EXPLAIN przed/po i uzasadnieniem	2
Konto aplikacji z minimalnymi uprawnieniami i test odmowy	2
Odpowiedzi na pytania i udział każdego członka zespołu (ocena nauczyciela)	2
Razem	20

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