

DDL — tworzenie bazy i tabel biblioteki

Czas: 2 × 45 min

praca indywidualna

Cel: Napisanie skryptu .sql, który tworzy bazę biblioteka z poprawnym kodowaniem i wszystkie jej tabele z kluczami, wartościami domyślnymi i ograniczeniami NOT NULL, oraz sprawdzenie go poleceniami SHOW.

Po zajęciach potrafisz:

- tworzy bazę z kodowaniem utf8mb4 i porównywaniem utf8mb4_polish_ci
- pisze polecenia CREATE TABLE z PRIMARY KEY, AUTO_INCREMENT, NOT NULL, DEFAULT, UNIQUE i FOREIGN KEY
- tworzy klucz złożony tabeli pośredniczącej
- zapisuje strukturę jako skrypt, który można uruchomić wielokrotnie
- sprawdza strukturę poleceniami SHOW CREATE DATABASE / TABLE

Sprzęt i materiały

- komputer z XAMPP
- schemat relacyjny biblioteki (zad. 5) i dobór typów (zad. 8)

Oprogramowanie

- **Visual Studio Code** — lub Notatnik++ —
pisanie skryptu `biblioteka.sql`
(kodowanie pliku UTF-8)
code.visualstudio.com
- **Konsola mysql lub phpMyAdmin** —
dokumentacja CREATE TABLE w MariaDB
mariadb.com/kb/en/create-table

Przygotowanie przed zajęciami

- Miej pod ręką swój schemat biblioteki i tabelę typów z zadania 8. Jeśli ich nie masz, poproś nauczyciela o schemat wzorcowy.

Bezpieczeństwo

- Polecenie DROP DATABASE IF EXISTS na początku skryptu usuwa bazę bez pytania — sprawdź, czy nazwa bazy w skrypcie to na pewno biblioteka, a nie np. sklep.
- Zapisuj skrypt w pliku na bieżąco — to on jest twoją kopią zapasową struktury.

Przebieg zadania

1 Baza danych

Piszesz skrypt w edytorze i uruchamiasz go w konsoli poleceniem SOURCE (albo wklejasz do zakładki SQL phpMyAdmin).

```
-- początek pliku biblioteka.sql
SET NAMES utf8mb4;
DROP DATABASE IF EXISTS biblioteka;
-- Z1: ...
USE biblioteka;
```

1. Z1 — dopisz polecenie tworzące bazę biblioteka z kodowaniem utf8mb4 i porównywaniem utf8mb4_polish_ci.
2. Z2 — po uruchomieniu wykonaj `SHOW CREATE DATABASE biblioteka;` i sprawdź kodowanie.
3. Z3 — porównaj działanie porównywań: `SELECT 'Kraków' = 'Krakow' COLLATE utf8mb4_polish_ci, 'Kraków' = 'Krakow' COLLATE utf8mb4_general_ci;`. Dlaczego dla polskiej bazy wybieramy polish_ci?

2 Tabele słownikowe i książki

Tabele twórz w kolejności zależności: najpierw te, na które wskazują klucze obce.

```
CREATE TABLE dzialy (
  id_dzialu INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  nazwa      VARCHAR(50) NOT NULL UNIQUE
) ENGINE=InnoDB;
```

1. Z4 — dopisz do skryptu tabelę dzialy z ramki oraz analogiczne tabele wydawnictwa i autorzy (imię i nazwisko osobno).
2. Z5 — utwórz tabelę ksiazki: identyfikator, ISBN (13 znaków, unikalny, może być pusty), tytuł, rok wydania, wydawnictwo (może być puste) i dział (wymagany). Dodaj dwa klucze obce.
3. Z6 — utwórz tabelę ksiazki_autorzy z kluczem złożonym `PRIMARY KEY (id_ksiazki, id_autora)`, kolumną kolejnosc z wartością domyślną 1 i dwoma kluczami obcymi.

3 Egzemplarze, czytelnicy, wypożyczenia

1. Z7 — utwórz tabelę egzemplarze: numer inwentarzowy jako klucz naturalny (bez AUTO_INCREMENT), książka, stan jako ENUM('dobry','zniszczony','wycofany') z wartością domyślną 'dobry'.
2. Z8 — utwórz tabelę czytelnicy: numer karty (AUTO_INCREMENT), imię, nazwisko, typ ENUM('uczen','nauczyciel') z wartością domyślną, klasa (może być pusta), e-mail (unikalny, może być pusty).
3. Z9 — utwórz tabelę wypozyczenia: identyfikator, egzemplarz, czytelnik, data wypożyczenia, termin zwrotu, data zwrotu (pusta = książka u czytelnika), liczba przedłużeń z wartością domyślną 0.
4. Dla chętnych: dodaj tabelę rezerwacje (tytuł, czytelnik, data rezerwacji, status).

4 Sprawdzenie struktury i skryptu

```
-- Z12: testy (wykonaj po kolei, zapisz komunikaty)
INSERT INTO dzialy (nazwa) VALUES ('Lektury');
INSERT INTO ksiazki (tytul, id_dzialu) VALUES ('Lalka', 1), ('Pan Tadeusz', 1);
INSERT INTO egzemplarze (nr_inwentarzowy, id_ksiazki) VALUES (4501, 1);
SELECT * FROM egzemplarze;
INSERT INTO egzemplarze (nr_inwentarzowy, id_ksiazki) VALUES (4502, 99);
INSERT INTO egzemplarze (nr_inwentarzowy, id_ksiazki, stan) VALUES (4503, 1, 'zgubiony');
```

1. Z10 — `SHOW TABLES;` — ile tabel ma baza? Z11 — `SHOW CREATE TABLE ksiazki;` — jakie nazwy nadał serwer kluczom obcym, których nie nazwałeś?
2. Z12 — wykonaj testy z ramki. Wyjaśnij: jaką wartość dostała kolumna stan, dlaczego dwie książki bez ISBN nie naruszyły UNIQUE, co oznaczają dwa ostatnie błędy.
3. Z13 — uruchom cały skrypt jeszcze raz (`SOURCE C:/.../biblioteka.sql`). Musi wykonać się bez żadnego błędu, a baza po nim jest pusta i gotowa do pracy.
4. Zrób zrzut widoku Projektant bazy biblioteka i porównaj z diagramem ERD z zadań 3–4. Zachowaj skrypt — w zadaniu 10 będziesz zmieniać strukturę.

W sprawozdaniu umieść

- ☐ pełny skrypt `biblioteka.sql` (wydruk lub plik)
- ☐ tabela poleceń Z1–Z13 z wynikami i komunikatami
- ☐ zrzut widoku Projektant bazy biblioteka
- ☐ wyjaśnienie wyników testów z Z12
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Czym różni się CHARACTER SET od COLLATE? Co by się stało, gdyby baza miała kodowanie latin1?
2. Dlaczego kolejność poleceń CREATE TABLE w skrypcie ma znaczenie?
3. Po co w skrypcie DROP DATABASE IF EXISTS i dlaczego jest to zarazem niebezpieczne?
4. Dlaczego kolumna z ograniczeniem UNIQUE może zawierać wiele wartości NULL?

Kryteria oceny

Kryterium	Punkty
Baza z poprawnym kodowaniem i porównywaniem (Z1–Z3)	2
Tabele z poprawnymi typami, PK, NOT NULL i DEFAULT (Z4–Z9)	6
Klucze obce i klucz złożony tabeli pośredniczącej	4
Sprawdzenie struktury i testy Z10–Z12 z wyjaśnieniem	3
Skrypt uruchamiający się wielokrotnie bez błędów (Z13)	2
Odpowiedzi na pytania kontrolne	3
Razem	20

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