

Normalizacja — implementacja i przeniesienie danych

Czas: 2 × 45 min

praca indywidualna

Cel: Utworzenie w MySQL/MariaDB tabel w 3NF zaprojektowanych w zadaniu 6, przeniesienie do nich danych z tabeli płaskiej poleceniem INSERT ... SELECT i sprawdzenie, że żadne dane nie zginęły.

Po zajęciach potrafisz:

- tworzy tabele z kluczami głównymi i obcymi we właściwej kolejności
- przenosi dane poleceniem INSERT ... SELECT DISTINCT
- wykrywa niespójne dane dzięki ograniczeniu UNIQUE i poprawia je przed migracją
- weryfikuje kompletność migracji liczbą wierszy i sumą kontrolną
- sprawdza, że anomalie z zadania 6 nie występują w bazie znormalizowanej

Sprzęt i materiały

- komputer z XAMPP
- plik `r1_06_zamowienia_plaskie.sql` z folderu Pliki_do_zadan i schemat 3NF z zadania 6

Oprogramowanie

- **phpMyAdmin** lub **konsola mysql**
phpmyadmin.net
- **Visual Studio Code (opcjonalnie)** —
pisanie skryptu `zad07_nazwisko.sql`
code.visualstudio.com

Przygotowanie przed zajęciami

- Zaimportuj ponownie plik `r1_06_zamowienia_plaskie.sql`, aby zacząć od czystych danych (usunie to tabelę kopia z zadania 6).

Bezpieczeństwo

- Polecenie UPDATE w kroku 3 zmienia dane źródłowe — przed nim wykonaj SELECT z tym samym warunkiem i sprawdź, ile wierszy zostanie zmienionych.
- Gdy coś pójdzie nie tak, nie poprawiaj na ślepo: zaimportuj plik ponownie i uruchom swój skrypt od początku.

Przebieg zadania

1 Schemat docelowy i tworzenie tabel

Wszyscy implementują ten sam schemat (porównaj go ze swoim z zadania 6). Pisz polecenia w pliku `zad07_nazwisko.sql` i wykonuj je w bazie normalizacja.

```
Kategorie(id_kategorii PK, nazwa UNIQUE, stawka_vat)
Klienci(id_klienta PK, imie, nazwisko, email UNIQUE, miasto)
Produkty(kod_produkту PK, nazwa, id_kategorii FK→Kategorie)
Zamowienia(nr_zam PK, data_zam, id_klienta FK→Klienci)
Pozycje(nr_zam PK FK→Zamowienia, kod_produkту PK FK→Produkty, ilosc, cena_jedn)

-- Z1: wzór
CREATE TABLE kategorie (
  id_kategorii INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  nazwa          VARCHAR(50) NOT NULL UNIQUE,
  stawka_vat     TINYINT UNSIGNED NOT NULL
) ENGINE=InnoDB;
```

1. Z1 — wykonaj polecenie z ramki.
2. Z2–Z5 — utwórz tabele klienci, produkty, zamowienia i pozycje. Typy kolumn przepisz z tabeli `zamowienia_plaskie` (`DESCRIBE`); `id_klienta` zrób jak `id_kategorii`. Klucz obcy zapisz w `CREATE TABLE` np. tak: `FOREIGN KEY (id_kategorii) REFERENCES kategorie(id_kategorii)`.
3. Zwróć uwagę na kolejność: tabela, na którą wskazuje klucz obcy, musi istnieć wcześniej. Typ kolumny klucza obcego musi być taki sam jak typ kolumny, na którą wskazuje (np. `INT UNSIGNED`).
4. Sprawdź wynik w phpMyAdmin → normalizacja → Więcej → Projektant: czy widać 4 linie relacji?

2 Przeniesienie kategorii i klientów

`INSERT ... SELECT` wstawia do tabeli wynik zapytania. Kolumna `AUTO_INCREMENT` nada identyfikatory sama.

```
-- Z6: wzór
INSERT INTO kategorie (nazwa, stawka_vat)
SELECT DISTINCT kategoria, stawka_vat FROM zamowienia_plaskie;
```

1. Z6 — wykonaj polecenie z ramki i wyświetl tabelę `kategorie`. Ile wierszy wstawiono?
2. Z7 — napisz analogiczne polecenie przenoszące klientów (imię, nazwisko, e-mail, miasto bez powtórzeń). Zapisz komunikat, który otrzymasz, i wyjaśnij go — pomoże ci Z4 z zadania 6.

3 Poprawienie danych i dokończenie migracji

Klucz obcy w tabeli płaskiej nie istnieje — identyfikator kategorii lub klienta trzeba odnaleźć po nazwie lub e-mailu. Służą do tego złączenie JOIN (dokładnie poznasz je w zadaniu 21).

```
-- Z8: poprawka danych źródłowych (najpierw SELECT z tym samym WHERE!)
UPDATE zamowienia_plaskie SET klient_miasto = 'Kraków' WHERE klient_miasto = 'Krakow';

-- Z10: wzór z JOIN
INSERT INTO produkty (kod_produkту, nazwa, id_kategorii)
SELECT DISTINCT p.kod_produkту, p.nazwa_produkту, k.id_kategorii
FROM zamowienia_plaskie AS p
JOIN kategorie AS k ON k.nazwa = p.kategoria;
```

1. Z8 — wyświetl wiersze z miastem 'Krakow', a potem wykonaj UPDATE z ramki. Zapisz liczbę zmienionych wierszy.
2. Z9 — wykonaj ponownie polecenie z Z7. Wyświetl tabelę klienci. Od jakiego numeru zaczynają się identyfikatory i dlaczego nie od 1?
3. Z10 — wykonaj polecenie z ramki. Z11 — napisz na tej samej zasadzie przeniesienie zamówień (nr_zam, data_zam, id_klienta) — klienta odnajdź po e-mailu.
4. Z12 — przenieś pozycje zamówień. Tu DISTINCT ani JOIN nie są potrzebne — dlaczego?

4 Kontrola — czy nic nie zginęło

```
-- Z14: odtworzenie tabeli płaskiej z tabel 3NF
SELECT COUNT(*)
FROM pozycje AS po
JOIN zamowienia AS z ON z.nr_zam = po.nr_zam
JOIN klienci AS k ON k.id_klienta = z.id_klienta
JOIN produkty AS p ON p.kod_produkту = po.kod_produkту
JOIN kategorie AS kat ON kat.id_kategorii = p.id_kategorii;
```

1. Z13 — policz wiersze w każdej z pięciu nowych tabel (`SELECT COUNT(*) FROM ...`) i porównaj z liczbą różnych wartości w tabeli płaskiej (np. `SELECT COUNT(DISTINCT kod_produkту) FROM zamowienia_plaskie;`). Wyniki wpisz do tabeli kontrolnej.
2. Z14 — wykonaj zapytanie z ramki. Wynik musi równać się liczbie wierszy tabeli płaskiej.
3. Z15 — oblicz sumę kontrolną `SUM(ilosc * cena_jedn)` w tabeli zamowienia_plaskie i w tabeli pozycje. Wartości muszą być identyczne.

5 Anomalie — czy zniknęły?

Powtórz eksperymenty z zadania 6 na tabelach 3NF.

1. Z16 — dodaj produkt P09 „Linijka 30 cm” do kategorii Przybory do pisania (sprawdź jej id w tabeli kategorie). Czy potrzebne jest zamówienie?
2. Z17 — zmień e-mail Anny Nowak na anna.nowak2@example.pl. Ile wierszy zmieniono?
3. Z18 — spróbuj usunąć zamówienie 1006 z tabeli zamowienia. Zapisz błąd. Następnie usuń najpierw jego pozycje, potem zamówienie, i sprawdź, czy klient Marek Wrona i produkt P07 nadal są w bazie.

W sprawozdaniu umieść

- ☐ skrypt `zad07_nazwisko.sql` z poleceniami Z1–Z18
- ☐ tabela zapytań z wynikami i komunikatami błędów
- ☐ wyjaśnienie błędu z Z7 i luki w numeracji klientów
- ☐ tabela kontrolna migracji (liczby wierszy, Z14, sumy kontrolne)
- ☐ wyniki eksperymentów z anomaliami (Z16–Z18)

☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Dlaczego tabele trzeba tworzyć i wypełniać w określonej kolejności? Jaka jest poprawna kolejność w tym zadaniu?
2. Jak ograniczenie UNIQUE na kolumnie email pomogło wykryć błąd w danych?
3. Dlaczego po nieudanym INSERT identyfikatory AUTO_INCREMENT nie zaczynają się od 1 i czy to jest problem?
4. Dlaczego porównanie samych liczb wierszy nie wystarcza do sprawdzenia migracji i po co liczy się sumę kontrolną?

Kryteria oceny

Kryterium	Punkty
Poprawne tabele Z1–Z5 z kluczami głównymi i obcymi	5
Migracja danych Z6–Z12 z wyjaśnieniem błędu 1062	5
Kontrola kompletności Z13–Z15	3
Eksperymenty z anomaliami Z16–Z18	3
Odpowiedzi na pytania kontrolne	4
Razem	20

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