SQL w praktyce

Baza sklepu komputerowego | Materiał dla ucznia | INF.03

Poznasz polecenia SQL, wykonasz zapytania w phpMyAdmin i przećwiczysz operacje podobne do tych spotykanych w zadaniach egzaminacyjnych. Wszystkie dane w bazie są fikcyjne.

1. Przygotowanie i import bazy

1. Uruchom Apache i MySQL w XAMPP. Otwórz w przeglądarce localhost/phpmyadmin.

2. Wybierz Nowa, wpisz nazwę sklep_inf03 i utwórz pustą bazę z kodowaniem utf8mb4 (np. utf8mb4_general_ci).

3. Kliknij nazwę bazy, wybierz Import, wskaż plik Baza_sklep_INF03.sql i uruchom import. Plik tworzy tabele w aktualnie wybranej bazie; nie tworzy bazy samodzielnie.

4. Wybierz bazę i kartę SQL. Wpisz zapytanie i kliknij Wykonaj. Na egzaminie zapisuj treść i zrzuty ekranu zgodnie z poleceniem arkusza.

Ponowny import wykonuj do nowej pustej bazy albo po samodzielnym usunięciu wcześniej utworzonych tabel ćwiczeniowych. Nie importuj do bazy z ważnymi danymi.

2. Jak zbudowana jest baza?

kategorie - rodzaje produktów.

producenci - nazwy producentów i ich kraje.

produkty - nazwa, cena, stan i powiązanie z kategorią oraz producentem.

klienci - fikcyjne osoby i ich miasta.

pracownicy - osoby obsługujące zamówienia.

zamowienia - data, status, klient i pracownik.

pozycje_zamowien - produkty w zamówieniu, ich ilość oraz cena zakupu.

Klucz podstawowy (PRIMARY KEY) jednoznacznie identyfikuje wiersz. Klucz obcy (FOREIGN KEY) wskazuje powiązany wiersz w innej tabeli. Jedno zamówienie ma wiele pozycji; jeden klient może mieć wiele zamówień.

Relacje: kategorie → produkty ← producenci; klienci → zamowienia ← pracownicy; zamowienia → pozycje_zamowien ← produkty.

Cena w tabeli produkty to cena bieżąca, a cena_zakupu w pozycji to cena zapisana dla danego zamówienia. Obliczając wartość zamówienia, korzystaj z ceny zakupu.

3. Wyjaśnienie poleceń i przykładów

SELECT i FROM

SELECT wybiera kolumny, a FROM wskazuje tabelę. Gwiazdka oznacza wszystkie kolumny. DISTINCT usuwa powtórzenia w wyświetlanych kombinacjach wartości.

```
SELECT nazwa, cena FROM produkty;
SELECT DISTINCT miasto FROM klienci;
```

WHERE, operatory i zakres

WHERE ogranicza wiersze. Używaj =, <>, >, <, >= i <=. AND wymaga obu warunków, OR przynajmniej jednego. BETWEEN obejmuje granice zakresu; IN sprawdza listę.

```
SELECT nazwa, cena FROM produkty
WHERE cena BETWEEN 100 AND 500 AND stan > 0;
```

LIKE i wzorce

LIKE wyszukuje według wzorca: % oznacza dowolny ciąg znaków (także pusty), a znak podkreślenia _ oznacza dokładnie jeden znak. Napisy zapisuj w apostrofach.

```
SELECT imie, nazwisko FROM klienci
WHERE nazwisko LIKE 'K%';
```

ORDER BY i LIMIT

ORDER BY sortuje: ASC rosnąco, DESC malejąco. LIMIT ogranicza liczbę zwracanych wierszy. Najpierw sortuj, a potem ograniczaj wynik.

```
SELECT nazwa, cena FROM produkty
ORDER BY cena DESC, id_produktu ASC LIMIT 5;
```

Funkcje agregujące

COUNT liczy wiersze, SUM sumuje, AVG liczy średnią, MIN i MAX zwracają najmniejszą i największą wartość. AS nadaje wynikowi czytelny alias.

```
SELECT COUNT(*) AS liczba, AVG(cena) AS srednia
FROM produkty;
```

GROUP BY i HAVING

GROUP BY tworzy grupy, np. według miasta. WHERE filtruje pojedyncze wiersze przed grupowaniem, a HAVING filtruje grupy po obliczeniu agregatów.

```
SELECT miasto, COUNT(*) AS liczba FROM klienci
GROUP BY miasto HAVING COUNT(*) > 40;
```

JOIN i relacje

JOIN łączy wiersze na podstawie klucza obcego. Alias p oznacza produkty, k oznacza kategorie. INNER JOIN zwraca dopasowania; LEFT JOIN zachowuje także wiersze z lewej tabeli bez dopasowania.

```
SELECT p.nazwa AS produkt, k.nazwa AS kategoria
FROM produkty p JOIN kategorie k
ON p.id_kategorii = k.id_kategorii;
```

INSERT INTO

INSERT dodaje rekord. Podaj kolumny i odpowiadające im wartości w tej samej kolejności. Identyfikator AUTO_INCREMENT może zostać nadany automatycznie.

```
INSERT INTO kategorie (nazwa) VALUES ('Słuchawki');
```

UPDATE i DELETE

UPDATE zmienia dane, a DELETE usuwa całe wiersze. Warunek WHERE wybiera rekordy. Bez WHERE operacja dotyczy wszystkich wierszy! Przed zmianą sprawdź ten sam warunek za pomocą SELECT.

```
SELECT * FROM produkty WHERE id_produktu = 1;
UPDATE produkty SET stan = 25 WHERE id_produktu = 1;
```

CREATE TABLE i ALTER TABLE

CREATE TABLE tworzy tabelę. ALTER TABLE zmienia jej strukturę: ADD COLUMN dodaje, MODIFY COLUMN zmienia definicję kolumny w MySQL/MariaDB, a DROP COLUMN ją usuwa wraz z danymi.

```
ALTER TABLE produkty ADD COLUMN opis VARCHAR(255) NULL;
ALTER TABLE produkty MODIFY COLUMN opis TEXT NULL;
ALTER TABLE produkty DROP COLUMN opis;
```

NULL, LEFT JOIN i COUNT

NULL oznacza brak wartości, nie zero ani pusty tekst. Szukaj go przez IS NULL, nie przez = NULL. Przy LEFT JOIN COUNT(z.id_zamowienia) nie liczy niedopasowanych wartości NULL; COUNT(*) liczyłby także pusty wiersz połączenia.

```
SELECT k.id_klienta, k.imie FROM klienci k
LEFT JOIN zamowienia z ON k.id_klienta = z.id_klienta
WHERE z.id_zamowienia IS NULL;
```

Podzapytanie

Zapytanie w nawiasie może obliczyć wartość potrzebną zewnętrznemu zapytaniu. Poniżej wybieramy produkty o cenie większej od średniej dla wszystkich produktów.

```
SELECT nazwa, cena FROM produkty
WHERE cena > (SELECT AVG(cena) FROM produkty);
```

4. Zadania praktyczne

Wykonaj każde polecenie w phpMyAdmin. Zapisuj zapytania w pliku tekstowym. Zadania 1-27 i 39-40 odczytują dane, zadania 28-38 zmieniają dane lub strukturę. Wykonuj je w kolejności. Aby powtórzyć część modyfikującą, zaimportuj bazę do nowej pustej bazy. Zadania 34-36 oraz 37-38 są ze sobą powiązane.

Poziom 1: wybieranie i filtrowanie

1. Wyświetl wszystkie kolumny tabeli kategorie.

2. Wyświetl tylko nazwę, cenę i stan produktów.

3. Wyświetl różne miasta klientów bez powtórzeń.

4. Wyświetl nazwy i ceny produktów droższych niż 3000 zł.

5. Wyświetl klientów z Lublina, podając imię, nazwisko i adres e-mail.

6. Wyświetl nazwy produktów o cenie od 100 do 500 zł włącznie.

7. Wyświetl produkty o stanie magazynowym mniejszym niż 10 i cenie poniżej 1000 zł.

8. Wyświetl klientów z Niemiec lub Lubartowa.

9. Wyświetl klientów, których nazwisko zaczyna się od litery K.

10. Wyświetl nazwy produktów zawierające tekst SSD.

11. Wyświetl produkty od najdroższego do najtańszego. W razie równej ceny sortuj po id rosnąco.

12. Wyświetl 5 najtańszych produktów; przy tej samej cenie wybierz produkt o mniejszym id.

Poziom 2: obliczenia i grupowanie

13. Policz wszystkich klientów. Nadaj wynikowi nazwę liczba_klientow.

14. Oblicz średnią, najmniejszą i największą cenę produktów.

15. Policz, ilu klientów pochodzi z każdego miasta.

16. Wyświetl miasta, w których jest więcej niż 40 klientów.

Poziom 3: relacje i łączenie tabel

17. Wyświetl nazwę produktu i nazwę jego kategorii.

18. Wyświetl nazwę, cenę i nazwę producenta każdego produktu.

19. Wyświetl identyfikator zamówienia, datę oraz imię i nazwisko klienta.

20. Wyświetl identyfikatory zamówień klientów z Lublina oraz ich nazwiska.

21. Wyświetl nazwy produktów z kategorii Myszki wraz z ich cenami.

22. Policz produkty w każdej kategorii. Podaj nazwę kategorii i liczbę produktów.

23. Wyświetl wszystkie zamówienia ze statusem wyslane złożone w 2024 roku.

24. Wyświetl numer każdej pozycji, ilość, cenę zakupu i wartość pozycji (ilość × cena).

25. Oblicz wartość każdego zamówienia na podstawie pozycji. Wyświetl id i wartość.

26. Wyświetl imię, nazwisko i liczbę zamówień każdego klienta, także bez zamówień.

27. Znajdź klientów bez zamówienia. Wyświetl id, imię i nazwisko.

Poziom 4: modyfikacje danych i struktury

28. Dodaj kategorię o nazwie Słuchawki. Identyfikator ma nadać baza.

29. Dodaj klienta: Adam Testowy, adres adam.testowy@example.test, miasto Niemce, data 2025-01-15.

30. Zmień stan produktu o id 1 na 25.

31. Zwiększ cenę produktu o id 2 o 10%.

32. Zmień status zamówienia o id 1 na wyslane.

33. Usuń dodanego w zadaniu 29 klienta na podstawie jego unikalnego adresu e-mail.

34. Dodaj do tabeli produkty kolumnę opis typu VARCHAR(255), która może przyjmować NULL.

35. Zmień typ dodanej kolumny opis na TEXT, dopuszczając NULL. Wymaga wykonania zadania 34.

36. Usuń kolumnę opis z tabeli produkty. Wymaga wykonania zadania 34.

37. Utwórz tabelę opinie z kolumnami: id_opinii INT jako klucz podstawowy AUTO_INCREMENT, id_produktu INT NOT NULL oraz tresc VARCHAR(255) NOT NULL. Dodaj klucz obcy do produkty.

38. Dodaj opinię „Działa dobrze” do produktu o id 1. Wymaga wykonania zadania 37.

Poziom 5: dodatkowe wyzwania

39. Wyświetl nazwy produktów droższych od średniej ceny wszystkich produktów.

40. Wyświetl identyfikator i wartość trzech najdroższych zamówień. Przy równej wartości sortuj po id rosnąco.

5. Krótka ściąga

```
SELECT kolumny FROM tabela
WHERE warunek
GROUP BY kolumna
HAVING warunek_grupy
ORDER BY kolumna DESC
LIMIT 5;
```