

PORADNIK UCZNIA

Bazy danych i SQL na egzaminie INF.03

Jak zamienić opis zapytania na poprawny kod SQL i nie stracić punktów przy imporcie bazy

Czego się dowiesz z tego poradnika:

- Jak tłumaczyć słowa klucze z treści zadania na klauzule SQL (ściągawka)
- Jak rozpoznać i zapisać relację 1:n oraz n:m (JOIN, podwójny JOIN)
- Jak krok po kroku budować zapytanie SELECT, żeby niczego nie pominąć
- Jak wykonać poprawnie import bazy i zrzuty ekranu w phpMyAdmin
- Listę najczęstszych błędów w zapytaniach SQL na egzaminie
- Checklistę do sprawdzenia kwerend przed zapisaniem pliku

1. Tłumacz „polski → SQL” — najważniejsza ściągawka

Treść zadania nigdy nie pisze wprost „użyj GROUP BY” — opisuje słowami, czego oczekuje. Twoja praca to rozpoznać w opisie zapytania konkretne słowa-klucze i zamienić je na odpowiednią klauzulę SQL. Poniższa tabela to najważniejsze narzędzie w tym rozdziale — naucz się jej najlepiej na pamięć.

Fraza w treści zadania	Klauzula SQL	Uwaga
„wybierające jedynie X, Y, Z”	SELECT X, Y, Z	nie SELECT * — wypisz dokładnie te kolumny, które wymieniono
„wszystkie pola / wszystkie dane”	SELECT *	jedyny przypadek, gdy gwiazdka jest poprawna
„dla których... / o wartości...”	WHERE warunek	filtruje wiersze
„posortowane rosnąco według X”	ORDER BY X ASC	ASC można pominąć — to wartość domyślna
„posortowane malejąco według X”	ORDER BY X DESC	DESC trzeba napisać zawsze wprost
„pierwsze N rekordów”	LIMIT N	razem z ORDER BY daje np. „5 najdroższych”
„bez powtórzeń / unikalne”	SELECT DISTINCT	usuwa duplikaty z wyniku
„zaokrąglone do N miejsc po przecinku”	ROUND(kolumna, N)	N to liczba miejsc dziesiętnych
„nadaj nazwę / alias”	AS nazwa	np. COUNT(id) AS liczba
„zaczyna się od X”	WHERE kolumna LIKE 'X%'	znak % zastępuje dowolny ciąg znaków
„kończy się na X”	WHERE kolumna LIKE '%X'	% przed wartością
„zawiera X”	WHERE kolumna LIKE '%X%'	% z obu stron
„zgrupowane dla każdego X / w podziale na X”	GROUP BY X	zawsze razem z funkcją agregującą
„liczba / ile rekordów”	COUNT(kolumna)	liczy wiersze (nie licz gwiazdką bez potrzeby)
„suma”	SUM(kolumna)	suma wartości liczbowych
„średnia”	AVG(kolumna)	często razem z ROUND()
„najmniejsza / najstarsza wartość”	MIN(kolumna)	
„największa / najnowsza wartość”	MAX(kolumna)	
„losowe N rekordów”	ORDER BY RAND() LIMIT N	losuje inne wiersze przy każdym uruchomieniu
„której X nie występuje w Y”	WHERE X NOT IN (SELECT ...)	podzapytanie w nawiasie

Operacje modyfikujące bazę — też mają swoje słowa klucze

Fraza w treści zadania	Klauzula SQL
„dodaj nowy rekord / wstaw dane”	INSERT INTO tabela (...) VALUES (...)
„zmień / zaktualizuj wartość”	UPDATE tabela SET kolumna = wartość WHERE ...
„usuń rekord(y)”	DELETE FROM tabela WHERE ...

Fraza w treści zadania	Klauzula SQL
„utwórz nową tabelę”	CREATE TABLE nazwa (...)
„dodaj nową kolumnę do tabeli”	ALTER TABLE tabela ADD kolumna TYP
„utwórz nowego użytkownika bazy”	CREATE USER 'login'@'localhost' IDENTIFIED BY 'hasło'

2. Relacje między tabelami — 1:n i n:m

Prawie każde zadanie ma bazę złożoną z kilku powiązanych tabel. Zanim napiszesz zapytanie z JOIN-em, ustal, z jakim typem relacji masz do czynienia — od tego zależy, ile tabel musisz połączyć.

2.1. Relacja 1:n (jeden do wielu)

Przykład: jeden rodzaj (np. „rodzaj” potrawy) może mieć wiele potraw, ale każda potrawa ma tylko jeden rodzaj. W tabeli „po stronie wielu” (potrawy) znajduje się klucz obcy wskazujący na tabelę „jeden” (rodzaje). Wystarczy jeden JOIN:

```
SELECT potrawy.nazwa, rodzaje.rodzaj
FROM potrawy
JOIN rodzaje ON potrawy.idRodzaje = rodzaje.idRodzaje;
```

2.2. Relacja n:m (wiele do wielu)

Przykład: jeden film może mieć wielu aktorów, a jeden aktor może grać w wielu filmach. Taką relację NIGDY nie zapisuje się kluczem obcym wprost w jednej z tabel — potrzebna jest trzecia, „pośrednicząca” tabela (często o nazwie łączącej obie, np. filmy_aktorzy), która przechowuje same pary identyfikatorów. Zapytanie wymaga wtedy PODWÓJNEGO JOIN-a:

```
SELECT filmy.tytul, aktorzy.imie, aktorzy.nazwisko
FROM filmy
JOIN filmy_aktorzy ON filmy.id_filmu = filmy_aktorzy.id_filmu
JOIN aktorzy ON filmy_aktorzy.id_aktora = aktorzy.id_aktora;
```

Jak rozpoznać relację n:m po strukturze bazy?

Jeśli w zrzucie struktury bazy (ilustracja z podglądem tabel) widzisz osobną, małą tabelę, która ma tylko dwie kolumny z kluczami obcymi (np. id_filmu + id_aktora, albo id_potrawy + id_alergenu) i żadnych własnych danych opisowych — to jest właśnie tabela pośrednicząca relacji n:m. Szukaj w jej nazwie połączenia nazw obu tabel głównych.

3. Jak budować zapytanie SELECT krok po kroku

Zamiast pisać całe zapytanie od razu, rozbij je na cztery pytania, na które odpowiadasz PO KOLEI — w takiej właśnie kolejności powstaje później kod:

Krok	Pytanie do siebie	Co z tego wynika w kodzie
1	Z jakiej tabeli / tabel biorę dane?	FROM ... (+ JOIN, jeśli dane są w więcej niż jednej tabeli)
2	Jakie dokładnie kolumny mam wypisać?	SELECT kolumna1, kolumna2 (nigdy "na wszelki wypadek" więcej niż wymieniono)
3	Czy jest jakiś warunek / filtr?	WHERE warunek (uważaj na dokładną wartość — wielkość liter, cudzysłowy dla tekstu)
4	Czy trzeba posortować, zgrupować albo ograniczyć liczbę wierszy?	ORDER BY / GROUP BY / LIMIT — w tej kolejności również w samym zapytaniu

Przykład zastosowania tej metody dla opisu: „wybierające jedynie nazwę i cenę potrawy o identyfikatorze 7, posortowane rosnąco według ceny”:

```
SELECT nazwa, cena
```

```
FROM potrawy  
WHERE idPotrawy = 7  
ORDER BY cena ASC;
```

4. Import bazy i zrzuty ekranu w phpMyAdmin

Ta część zadania nie wymaga umiejętności programistycznych, ale bardzo łatwo stracić na niej punkty przez pominięcie formalnego wymogu.

Tworzenie bazy i import

1. W phpMyAdmin kliknij „Nowa baza danych”, wpisz DOKŁADNIE nazwę wskazaną w treści zadania (wielkość liter ma znaczenie!) i wybierz zestaw znaków wskazany w treści (najczęściej coś z „unicode” lub „polish” w nazwie — pozwala to poprawnie wyświetlać polskie znaki).
2. Wejdź w utworzoną bazę, otwórz zakładkę „Importuj”, wskaż plik .sql z rozpakowanego archiwum i zatwierdź.
3. Po zakończeniu importu sprawdź, czy po lewej stronie pojawiła się poprawna liczba tabel — to Twoje pierwsze potwierdzenie, że import się udał.

Zrzut ekranu po imporcie — częste powody utraty punktów

Zrzut musi obejmować CAŁY ekran monitora, z widocznym paskiem zadań (nie tylko okno przeglądarki) i NIE może być kadrowany. Musi być zapisany dokładnie w formacie wskazanym w treści (raz jest to PNG, raz JPEG — zawsze sprawdź to osobno dla każdego zadania!) pod dokładnie wskazaną nazwą pliku. Na zrzucie musi być widoczny efekt importu — np. lista tabel po lewej stronie phpMyAdmin.

Zrzuty z wykonania kwerend

Dla każdego zapytania z pliku kwerendy.txt wklej jego treść w zakładce SQL, wykonaj i zrób zrzut widocznego wyniku — również cały ekran, z paskiem zadań, w formacie i pod nazwą wskazaną w treści (typowo kw1, kw2, kw3... w kolejności zapytań).

Uwaga: dla zapytań **INSERT**, **UPDATE**, **DELETE** czy **CREATE TABLE** phpMyAdmin nie pokaże tabeli z danymi, tylko komunikat o liczbie zmienionych/dodanych wierszy — to jest poprawny, oczekiwany efekt takiego zrzutu, nie błąd.

5. Najczęstsze błędy w zapytaniach SQL

Błąd	Dlaczego to boli
SELECT * zamiast konkretnych kolumn	Zadanie wprost wymienia kolumny — dopisanie dodatkowych to niezgodność z treścią.
Zapomniany WHERE albo zły warunek	Zapytanie zwraca wszystkie wiersze zamiast przefiltrowanych — łatwo to przeoczyć, patrząc tylko na wynik z pierwszymi wierszami zgadzającymi się przypadkowo.
Zła kolejność ASC/DESC	„Malejąco” i „rosnąco” to przeciwne kierunki — czytaj to słowo bardzo uważnie, łatwo pomylić.
JOIN bez warunku ON (albo z błędnym warunkiem)	Powoduje pomnożenie liczby wierszy (tzw. iloczyn kartezjański) — wynik wygląda na "jakiś" wynik, ale jest błędny.
Brak GROUP BY przy użyciu funkcji agregującej z inną kolumną	MySQL bywa pobłażliwy i "coś" zwróci, ale wynik będzie nielogiczny (przypadkowa wartość z grupy).
Literówka w nazwie kolumny/tabeli	Sprawdź dokładnie zrzut struktury bazy — czasem nazwa różni się od tej użytej w treści opisowej zadania (np. idPotrawy a nie id_potrawy).
Cudzysłowy wokół liczb albo ich brak wokół tekstu	Wartości tekstowe zawsze w cudzysłowie ('alfa'), liczby bez cudzysłowu (WHERE id = 7).
Zły alias kolumny (AS)	Jeśli treść zadania podaje dokładną nazwę aliasu (np. AS liczba_filmow), przepisuj ją dosłownie — bywa sprawdzana.
Pomylenie relacji n:m z 1:n	Brak drugiego JOIN-a przez tabelę pośredniczącą, gdy dane są w trzech tabelach.
Zapomniane ROUND() przy wymaganym zaokrągleniu	Wynik z wieloma miejscami po przecinku nie jest zgodny z treścią, mimo że logika zapytania jest poprawna.

6. Checklista przed zapisaniem kwerend

Przejdź przez tę listę dla KAŻDEGO zapytania osobno, zanim przejdziesz do pisania skryptów PHP.

- ☐ Wypisałem dokładnie te kolumny, które wymieniono w treści (nie SELECT *, chyba że wymagano „wszystkich pól”).
- ☐ Sprawdziłem nazwy kolumn i tabel na rzucie struktury bazy — zgadzają się co do joty (wielkość liter, podkreślenia).
- ☐ Warunek WHERE odpowiada dokładnie opisanej wartości (uwzględam na ASC/DESC, na dokładny próg liczbowy).
- ☐ Jeśli dane są w dwóch tabelach — dodałem JOIN z poprawnym warunkiem ON.
- ☐ Jeśli dane są w trzech tabelach (relacja n:m) — dodałem PODWÓJNY JOIN przez tabelę pośredniczącą.
- ☐ Alias kolumny (AS) — jeśli podano w treści dokładną nazwę, przepisałem ją dosłownie.
- ☐ Zaokrągliłem wynik przez ROUND(), jeśli zadanie tego wymagało.
- ☐ Zapisałem wszystkie zapytania do pliku kwerendy.txt w dokładnie takiej kolejności, w jakiej występują w treści.
- ☐ Zrobiłem zrzuty ekranu w odpowiednim formacie, z widocznym paskiem zadań, bez kadrowania.

Ostatnia rada

Zanim uznasz zapytanie za gotowe, przeczytaj jeszcze raz zdanie z treści zadania i porównaj je słowo po słowie ze swoim kodem SQL — każde słowo w opisie zapytania (jedynie, rosnąco, pierwsze, bez powtórzeń, zaokrąglone) odpowiada konkretnemu fragmentowi kodu. Jeśli nie potrafisz wskazać, któremu słowu odpowiada dana klauzula w Twoim zapytaniu — prawdopodobnie coś pominąłeś.