Tablica informatyczna · egzamin INF.03

SQL

MariaDB / MySQL w phpMyAdmin — zapytania, funkcje, polecenia na tabelach i użytkownikach

Zestawienie poleceń SQL, które pojawiają się w części „Operacje na bazie danych” arkuszy INF.03. W każdym arkuszu są cztery lub pięć zapytań, zapisywanych w pliku kwerendy.txt i dokumentowanych zrzutami ekranu.

●●● ponad połowa arkuszy 2023–2026 ●●○ co najmniej 15% ●○○ rzadziej ○○○ poza arkuszami, wiedza uzupełniająca

Liczone: treść zadania lub tabela pomocnicza arkusza

Wybieranie danych — SELECT

SELECT … FROM ●●●

```
SELECT kol1, kol2 FROM tabela;
```

Wybiera wskazane kolumny ze wszystkich wierszy tabeli. Gwiazdka (*) oznacza wszystkie kolumny.

```
SELECT marka, model, cena FROM pojazdy;
```

Uwaga. Treść „wybierające jedynie pola…” oznacza wypisanie dokładnie tych kolumn, w tej kolejności.

DISTINCT ●○○

```
SELECT DISTINCT kol FROM tabela;
```

Usuwa powtarzające się wiersze z wyniku — każda wartość pojawia się raz.

```
SELECT DISTINCT miasto FROM adresy ORDER BY miasto;
```

AS (alias) ●●○

```
SELECT kol AS "Nazwa" FROM tabela;
```

Nadaje kolumnie lub wyrażeniu własną nazwę w wyniku. Tak realizuje się polecenie „nazwa kolumny (alias)”.

```
SELECT AVG(cena) AS "Średnia cena" FROM pojazdy;
```

Uwaga. Alias ze spacją albo polskimi znakami zapisz w cudzysłowie, dokładnie tak jak w treści zadania.

WHERE ●●●

```
… WHERE warunek;
```

Pozostawia tylko wiersze spełniające warunek. Operatory: =, <> (lub !=), <, >, <=, >=.

```
SELECT nazwa FROM szczyty WHERE wysokosc > 1500;
```

Uwaga. Tekst i daty zapisuj w apostrofach lub cudzysłowie: WHERE marka = 'BM'. Liczby bez cudzysłowu.

AND, OR, NOT ●●○

```
WHERE a AND b OR NOT c
```

Łączą warunki: AND — oba prawdziwe, OR — co najmniej jeden, NOT — negacja. AND wiąże silniej niż OR; nawiasy zmieniają kolejność.

```
SELECT nazwa FROM wycieczki WHERE poczatek = 'Muszyna' OR poczatek = 'Wieliczka';
```

BETWEEN … AND ●○○

```
WHERE kol BETWEEN a AND b
```

Wartość z przedziału domkniętego (łącznie z końcami). Odpowiada: kol >= a AND kol <= b.

```
SELECT nazwisko, pensja FROM pracownicy WHERE staz BETWEEN 10 AND 20;
```

IN ●○○

```
WHERE kol IN (w1, w2, …)
```

Wartość równa któremukolwiek z wymienionych. Krótszy zapis wielu warunków OR na jednej kolumnie.

```
SELECT imie FROM osoby WHERE miasto IN ('Lublin', 'Kielce');
```

LIKE ●●●

```
WHERE kol LIKE 'wzorzec'
```

Dopasowanie do wzorca: % — dowolny ciąg znaków (także pusty), _ — dokładnie jeden znak.

```
SELECT imie, nazwisko FROM osoby WHERE imie LIKE 'J%';
-- imiona na literę J
```

Uwaga. Początek: 'J%'. Koniec: '%a'. Zawieranie: '%mir%'. Wzorzec zawsze w apostrofach.

IS NULL / IS NOT NULL ●○○

```
WHERE kol IS NULL
```

Sprawdza brak wartości. Porównanie = NULL nigdy nie jest prawdziwe, trzeba użyć IS.

```
DELETE FROM nagrody WHERE cena IS NULL OR cena = 0;
```

ORDER BY ●●○

```
… ORDER BY kol [ASC | DESC]
```

Sortuje wynik: ASC — rosnąco (domyślnie), DESC — malejąco. Można podać kilka kolumn po przecinku.

```
SELECT marka, model, cena FROM pojazdy ORDER BY cena DESC;
```

Uwaga. „Posortowane rosnąco według nazwy” = ORDER BY nazwa. „Malejąco” = DESC po nazwie kolumny.

LIMIT ●○○

```
… LIMIT n   albo   LIMIT przesunięcie, n
```

Ogranicza liczbę zwróconych wierszy do n (po sortowaniu).

```
SELECT marka, model, cena FROM pojazdy WHERE marka = 'BM' ORDER BY cena DESC LIMIT 15;
```

Uwaga. „Pierwsze 15 rekordów”, „najdroższy”, „dwa dowolne rekordy” — wszystko to LIMIT, zwykle z ORDER BY.

© Dawid Chaber · TablicaPro (tablicapro.pl) · materiał chroniony prawem autorskim — kopiowanie i udostępnianie bez zgody autora zabronione

Funkcje agregujące i grupowanie

COUNT ●●○

```
COUNT(*)   COUNT(kol)   COUNT(DISTINCT kol)
```

Liczba wierszy. COUNT(*) liczy wszystkie, COUNT(kol) pomija wartości NULL, z DISTINCT liczy różne wartości.

```
SELECT COUNT(*) FROM klienci;
```

SUM ●○○

```
SUM(kol)
```

Suma wartości kolumny (NULL pomijane).

```
SELECT SUM(koszt) AS "Całkowity koszt" FROM uczestnicy;
```

AVG ●●○

```
AVG(kol)
```

Średnia arytmetyczna wartości kolumny.

```
SELECT AVG(stanWody) FROM pomiary;
```

Uwaga. Wynik często owija się w ROUND(..., 2) albo FLOOR(...), gdy treść każe zaokrąglić.

MIN, MAX ●●○

```
MIN(kol)   MAX(kol)
```

Najmniejsza i największa wartość kolumny. „Najdroższy", „najwyższy", „najstarszy" to zwykle MAX albo ORDER BY ... DESC LIMIT 1.

```
SELECT MAX(wysokosc) FROM gory;
```

GROUP BY ●●○

```
SELECT kol, COUNT(*) FROM t GROUP BY kol;
```

Dzieli wiersze na grupy o tej samej wartości kolumny; funkcje agregujące liczą osobno dla każdej grupy.

```
SELECT wojewodztwa.nazwa, COUNT(miasta.nazwa) AS "Liczba
miast"
FROM wojewodztwa JOIN miasta ON wojewodztwa.id =
miasta.id_wojewodztwa
GROUP BY wojewodztwa.nazwa;
```

Uwaga. Wskazówki w treści: „dla każdego...", „zgrupowane", „w podziale na...". Kolumny w SELECT poza agregatami muszą być w GROUP BY.

HAVING ○○○

```
… GROUP BY kol HAVING warunek_na_agregacie
```

Filtruje grupy po obliczeniu agregatu (WHERE filtruje wiersze przed grupowaniem).

```
SELECT autor, COUNT(*) FROM ksiazki GROUP BY autor HAVING
COUNT(*) > 3;
```

Łączenie tabel i podzapytania

Polecenie „Należy posłużyć się relacją" oznacza złączenie tabel po kluczu obcym. Klucz obcy i główny widać na ilustracji bazy danych w arkuszu.

JOIN … ON ●●●

```
FROM t1 JOIN t2 ON t1.id = t2.t1_id
```

Łączy wiersze dwóch tabel o pasujących kluczach (INNER JOIN — wiersze bez pary znikają).

```
SELECT nazwisko, imie, zamowienia.id
FROM klienci JOIN zamowienia ON klienci.id =
zamowienia.id_klienta;
```

Uwaga. Gdy nazwa kolumny występuje w obu tabelach, poprzedź ją nazwą tabeli: zamowienia.id. Brak ON daje iloczyn kartezjański.

JOIN … USING

```
FROM t1 JOIN t2 USING(kol)
```

Skrót, gdy kolumna łącząca ma w obu tabelach tę samą nazwę.

```
SELECT model, nazwa, cena FROM buty JOIN produkt
USING(model);
```

Występuje w rozwiązaniach wzorcowych CKE

Złączenie przez WHERE

```
FROM t1, t2 WHERE t1.id = t2.t1_id
```

Starszy zapis złączenia — tabele po przecinku, warunek łączący w WHERE. Daje ten sam wynik co JOIN ... ON.

```
SELECT tytul, tresc FROM ogloszenie, uzytkownik
WHERE ogloszenie.uzytkownik_id = uzytkownik.id AND
kategoria = 1;
```

Występuje w rozwiązaniach wzorcowych CKE

LEFT JOIN ○○○

```
FROM t1 LEFT JOIN t2 ON …
```

Zachowuje wszystkie wiersze lewej tabeli, także bez pary (kolumny prawej tabeli mają wtedy NULL).

```
SELECT imie, COUNT(zamowienia.id)
FROM klienci LEFT JOIN zamowienia ON klienci.id =
id_klienta
GROUP BY klienci.id;
```

Uwaga. Użyteczny, gdy trzeba pokazać także klientów (autorów, kategorie) bez żadnych powiązanych rekordów.

Relacja n:m ●○○

```
t1 JOIN laczaca ON … JOIN t2 ON …
```

Trzy tabele: dwie encje i tabela łącząca z dwoma kluczami obcymi. Złączenie przechodzi przez tabelę łączącą.

```
SELECT choroby.nazwa, COUNT(objawy.id)
FROM choroby JOIN choroby_objawy ON choroby.id =
id_choroby
  JOIN objawy ON objawy.id = id_objawu
GROUP BY choroby.id;
```

NOT IN — brak powiązanych rekordów ●○○

```
WHERE id NOT IN (SELECT klucz_obcy FROM t2)
```

Wybiera rekordy, których identyfikator nie występuje w drugiej tabeli (np. osoby, które nie mają wpisu w rejestrze). Alternatywa: LEFT JOIN ... WHERE t2.id IS NULL.

```
SELECT id, nazwisko FROM personel
WHERE id NOT IN (SELECT id_personelu FROM rejestr);
```

© Dawid Chaber · TablicaPro (tablicapro.pl) · materiał chroniony prawem autorskim — kopiowanie i udostępnianie bez zgody autora zabronione

Podzapytanie ●○○

```
WHERE kol = (SELECT …)
```

Zapytanie wewnątrz zapytania; wewnętrzne wylicza wartość (np. maksimum), zewnętrzne z niej korzysta.

```
SELECT imie, nazwisko, pensja FROM pracownicy
WHERE pensja = (SELECT MAX(pensja) FROM pracownicy);
```

Uwaga. Wewnętrzne zapytanie zwracające jedną wartość porównuj operatorem =, kilka wartości — IN.

Wyrażenia i funkcje liczbowe

Kolumna wyliczana ●●○

```
SELECT kol1 * kol2 AS wynik …
```

W SELECT można używać działań + – * / na kolumnach; wynik nazywa się aliasem.

```
SELECT liczba_odbitek * cena AS "Do zapłaty" FROM
zamowienia;
```

ROUND ●○○

```
ROUND(x, miejsca)
```

Zaokrągla do podanej liczby miejsc po przecinku (do najbliższej wartości).

```
SELECT ROUND(AVG(Cena), 2) AS "Średnia_cena" FROM wyroby
GROUP BY Rodzaj;
```

FLOOR, CEIL ●○○

```
FLOOR(x)   CEIL(x)
```

FLOOR — największa liczba całkowita nie większa niż x (w dół). CEIL — najmniejsza nie mniejsza niż x (w górę).

```
SELECT FLOOR(AVG(rokUr)) AS "Średni rok urodzenia" FROM
uczestnicy;
```

Uwaga. „Zaokrąglony do najbliższej mniejszej liczby całkowitej” = FLOOR.

TRUNCATE ○○○

```
TRUNCATE(x, miejsca)
```

Obcina część ułamkową do podanej liczby miejsc bez zaokrąglania.

```
SELECT TRUNCATE(2.789, 1);   -- 2.7
```

RAND() ●○○

```
ORDER BY RAND() LIMIT n
```

Liczba losowa z przedziału [0, 1). W ORDER BY miesza kolejność wierszy — z LIMIT daje losowe rekordy.

```
SELECT nazwa, opis, cena FROM nagrody ORDER BY RAND()
LIMIT 5;
```

Uwaga. „Losujące dwa dowolne rekordy” = ORDER BY RAND() LIMIT 2. FLOOR(1 + RAND() * 10) daje liczbę całkowitą 1–10.

MOD, ABS, POWER ○○○

```
MOD(a, b)   ABS(x)   POWER(x, y)
```

Reszta z dzielenia, wartość bezwzględna, potęga.

```
SELECT id FROM zamowienia WHERE MOD(id, 2) = 0;
```

Funkcje tekstowe

CONCAT ○○○

```
CONCAT(a, b, …)
```

Skleja napisy w jeden.

```
SELECT CONCAT(imie, ' ', nazwisko) AS osoba FROM klienci;
```

LENGTH, CHAR_LENGTH ●○○

```
CHAR_LENGTH(txt)
```

Liczba znaków napisu (LENGTH liczy bajty — dla polskich liter wynik bywa większy).

```
SELECT nazwa FROM miasta WHERE CHAR_LENGTH(nazwa) > 10;
```

UPPER, LOWER ●○○

```
UPPER(txt)   LOWER(txt)
```

Zamiana na wielkie i małe litery (synonimy: UCASE, LCASE).

```
SELECT LOWER(nazwa) FROM wojewodztwa;
```

LEFT, RIGHT ○○○

```
LEFT(txt, n)   RIGHT(txt, n)
```

Pierwsze lub ostatnie n znaków napisu.

```
SELECT LEFT(imie, 2), RIGHT(nazwisko, 3) FROM czytelnicy;
```

SUBSTRING ○○○

```
SUBSTRING(txt, od, ile)
```

Fragment napisu od pozycji „od” (liczonej od 1) o długości „ile”.

```
SELECT SUBSTRING(pesel, 1, 2) FROM osoby;
```

REPLACE, TRIM ○○○

```
REPLACE(txt, co, naco)   TRIM(txt)
```

Zamiana ciągu znaków na inny; usunięcie spacji z początku i końca napisu.

```
SELECT REPLACE(telefon, '-', '') FROM klienci;
```

Daty i czas

CURDATE, NOW ●○○

```
CURDATE()   CURRENT_DATE   NOW()   CURTIME()
```

Bieżąca data; data i czas; bieżący czas. CURRENT_DATE, CURRENT_TIME, CURRENT_TIMESTAMP — synonimy.

```
UPDATE wagi SET dzien = CURRENT_DATE, czas = CURRENT_TIME
WHERE lokalizacje_id BETWEEN 2 AND 4;
```

Uwaga. W tabelach pomocniczych arkuszy pojawiają się też ADDDATE, ADDTIME i CONVERT_TZ.

YEAR, MONTH, DAY ●○○

```
YEAR(data)   MONTH(data)   DAY(data)
```

Wydobywają rok, miesiąc i dzień z daty. Przydają się w WHERE i GROUP BY.

```
SELECT imie FROM swinki WHERE MONTH(data_ur) = 7;
```

Uwaga. Alternatywa dla LIKE: data LIKE '2021-08-%' wybiera sierpień 2021.

© Dawid Chaber · TablicaPro (tablicapro.pl) · materiał chroniony prawem autorskim — kopiowanie i udostępnianie bez zgody autora zabronione

DATEDIFF ●○○

```
DATEDIFF(data1, data2)
```

Liczba dni między dwiema datami (data1 – data2).

```
SELECT pseudonim, DATEDIFF(CURRENT_DATE(), data_zdobycia)
AS dni FROM zawodnicy;
```

DATE_ADD, ADDDATE ○○○

```
ADDDATE(data, INTERVAL n DAY)
```

Dodaje do daty podany odstęp (DAY, MONTH, YEAR).

```
SELECT ADDDATE(data_zamowienia, INTERVAL 14 DAY) FROM
zamowienia;
```

DATE_FORMAT ○○○

```
DATE_FORMAT(data, '%d.%m.%Y')
```

Formatuje datę według szablonu (%Y rok 4-cyfrowy, %m miesiąc, %d dzień, %H godzina).

```
SELECT DATE_FORMAT(data, '%d.%m.%Y') FROM wagi;
```

Zmiana danych — INSERT, UPDATE, DELETE

INSERT INTO ●●○

```
INSERT INTO t (kol1, kol2) VALUES (w1, w2);   INSERT
INTO t VALUES (NULL, w1, w2);
```

Dodaje wiersz. Przy zapisie bez listy kolumn podaj wartości dla wszystkich kolumn w kolejności z tabeli; NULL w miejscu AUTO_INCREMENT nada kolejny numer.

```
INSERT INTO logowanie (id, nick, haslo) VALUES (3,
'Jeremi', 'Jer123');
```

Uwaga. Wstawianą wartość tekstową otocz apostrofami, liczbę zapisz bez.

UPDATE … SET ●●○

```
UPDATE t SET kol = wartość WHERE warunek;
```

Zmienia wartości w wierszach spełniających warunek. Można użyć wyrażenia na samej kolumnie (staz = staz + 1).

```
UPDATE zawodnicy SET klasa = '5A' WHERE klasa = '4A';
```

Uwaga. Bez WHERE zmieni WSZYSTKIE wiersze tabeli.

DELETE FROM ●○○

```
DELETE FROM t WHERE warunek;
```

Usuwa wiersze spełniające warunek.

```
DELETE FROM wagi WHERE id > 22;
```

Uwaga. Bez WHERE usunie wszystkie wiersze (struktura tabeli zostaje).

Struktura tabel i baz — DDL

ALTER TABLE … ADD ●○○

```
ALTER TABLE t ADD COLUMN kol typ [AFTER inna];
```

Dodaje kolumnę do istniejącej tabeli. AFTER wskazuje, po której kolumnie ją umieścić.

```
ALTER TABLE wycieczki ADD COLUMN ocena INT;
```

ALTER TABLE … DROP ●○○

```
ALTER TABLE t DROP COLUMN kol;
```

Usuwa kolumnę wraz z jej danymi.

```
ALTER TABLE uczestnicy DROP COLUMN haslo;
```

ALTER TABLE … CHANGE / MODIFY ●○○

```
ALTER TABLE t CHANGE COLUMN stara nowa typ;
ALTER TABLE t MODIFY COLUMN kol nowy_typ;
```

CHANGE zmienia nazwę (i typ) kolumny; MODIFY zmienia tylko typ lub właściwości.

```
ALTER TABLE zapisy CHANGE COLUMN Id_klienta Id_sluchacza
INT;
```

CREATE TABLE ●○○

```
CREATE TABLE t (id INT PRIMARY KEY AUTO_INCREMENT, nazwa
VARCHAR(50) NOT NULL);
```

Tworzy tabelę. PRIMARY KEY — klucz główny, AUTO_INCREMENT — automatyczna numeracja, NOT NULL — wartość obowiązkowa.

```
CREATE TABLE kategorie (
  id INT PRIMARY KEY AUTO_INCREMENT,
  nazwa VARCHAR(40) NOT NULL
);
```

Typy danych ●○○

```
INT  VARCHAR(n)  TEXT  DATE  TIME  DATETIME
DECIMAL(p,s)  TINYINT(1)
```

INT — liczba całkowita; VARCHAR(n) — tekst do n znaków; TEXT — długi tekst; DATE — data RRRR-MM-DD; DECIMAL(p,s) — liczba dziesiętna; TINYINT(1) — wartość logiczna 0/1.

```
ALTER TABLE imieniny ADD COLUMN zyczenia VARCHAR(500);
```

DROP TABLE / DROP DATABASE ○○○

```
DROP TABLE t;   DROP DATABASE baza;
```

Usuwa tabelę lub całą bazę razem z danymi — bezpowrotnie.

```
DROP TABLE IF EXISTS tymczasowa;
```

CREATE DATABASE ●●●

```
CREATE DATABASE nazwa CHARACTER SET utf8 COLLATE
utf8_polish_ci;
```

Tworzy bazę z zestawem znaków obsługującym polskie litery. W phpMyAdmin odpowiada przyciskowi „Nowa” — wpisujesz nazwę i wybierasz porównywanie znaków (collation) utf8_polish_ci albo utf8_unicode_ci.

```
CREATE DATABASE samochody CHARACTER SET utf8 COLLATE
utf8_polish_ci;
```

Import i zrzut po imporcie ●●●

```
phpMyAdmin → baza → Import → Wybierz plik → Wykonaj
```

Plik .sql z archiwum importujesz do utworzonej bazy. Zrzut ekranu „import” ma obejmować cały ekran z paskiem zadań, na którym widać poprawny import (lista tabel lub komunikat).

```
-- plik baza.sql zawiera polecenia CREATE TABLE i INSERT
INTO
```

© Dawid Chaber · TablicaPro (tablicapro.pl) · materiał chroniony prawem autorskim — kopiowanie i udostępnianie bez zgody autora zabronione

Użytkownicy i uprawnienia

Zapytania 4 i 5 w wielu arkuszach to para: utworzenie konta i nadanie mu praw.

CREATE USER ●○○

```
CREATE USER 'nazwa'@'localhost' IDENTIFIED BY 'hasło';
```

Tworzy konto, z którego można łączyć się z serwera lokalnego. Hasło podajesz dokładnie tak, jak w treści.

```
CREATE USER 'Marek'@'localhost' IDENTIFIED BY 'M@reK';
```

Uwaga. Konto nie ma żadnych praw, dopóki nie nadasz ich poleceniem GRANT.

GRANT ●○○

```
GRANT prawa ON baza.tabela TO 'nazwa'@'localhost';
```

Nadaje uprawnienia: SELECT (odczyt), INSERT (dodawanie), UPDATE (zmiana), DELETE (usuwanie), ALL PRIVILEGES (wszystkie). baza.* oznacza wszystkie tabele bazy.

```
GRANT SELECT, UPDATE ON obuwie.produkt TO
'Marek'@'localhost';
```

Uwaga. „Wszystkie prawa do manipulowania danymi” = SELECT, INSERT, UPDATE, DELETE (bez praw do zmiany struktury).

REVOKE / DROP USER ○○○

```
REVOKE prawa ON baza.* FROM 'u'@'localhost';   DROP USER
'u'@'localhost';
```

Odebranie wybranych praw; usunięcie konta.

```
REVOKE DELETE ON obuwie.* FROM 'Marek'@'localhost';
```

SHOW GRANTS ●○○

```
SHOW GRANTS FOR 'u'@'localhost';
```

Wyświetla prawa konta — można użyć do udokumentowania zrzutem, że uprawnienia zostały nadane.

```
SHOW GRANTS FOR 'Marek'@'localhost';
```

Z arkusza do kodu: typowe sformułowania

Polecenia w arkuszach powtarzają się. Po lewej zwrot z treści zadania, po prawej jego zapis.

wybierające jedynie pola A, B, C z tabeli T

```
SELECT A, B, C FROM T;
```

dla wiersza o id równym 2 / dla rekordu o numerze 5

```
… WHERE id = 2;
```

posortowane malejąco według ceny

```
… ORDER BY cena DESC;
```

pierwsze 15 rekordów

```
… LIMIT 15;
```

losujące dwa dowolne rekordy

```
… ORDER BY RAND() LIMIT 2;
```

liczba wszystkich rekordów w tabeli

```
SELECT COUNT(*) FROM T;
```

średnia cena oraz najdroższy pojazd, nazwy kolumn (aliasy) „Średnia cena”, „Maksymalna cena”

```
SELECT AVG(cena) AS "Średnia cena", MAX(cena) AS
"Maksymalna cena" FROM pojazdy;
```

należy posłużyć się relacją

```
… FROM T1 JOIN T2 ON T1.id = T2.id_T1 …
```

zaokrąglony do najbliższej mniejszej liczby całkowitej

```
FLOOR( … )
```

zgrupowane dla każdej choroby / dla każdego autora

```
… GROUP BY choroby.id;
```

imiona rozpoczynające się literą J

```
… WHERE imie LIKE 'J%';
```

w latach 1970–1979

```
… WHERE YEAR(data_ur) BETWEEN 1970 AND 1979;
```

tworzące konto użytkownika Anna na localhost z hasłem …

```
CREATE USER 'Anna'@'localhost' IDENTIFIED BY '…';
```

nadające użytkownikowi prawa do wybierania i aktualizacji danych jedynie dla tabeli T

```
GRANT SELECT, UPDATE ON baza.T TO 'Anna'@'localhost';
```

nadające użytkownikowi wszystkie prawa do manipulowania danymi we wszystkich tabelach bazy

```
GRANT SELECT, INSERT, UPDATE, DELETE ON baza.* TO
'u'@'localhost';
```

dodające kolumnę / usuwające kolumnę

```
ALTER TABLE T ADD COLUMN ocena INT;   ALTER TABLE T DROP
COLUMN ocena;
```

nowa kolumna typu TINYINT, długość 1, wartość domyślna 0

```
ALTER TABLE ksiazka ADD COLUMN rezerwacja TINYINT(1)
DEFAULT 0;
```

nowa kolumna typu napisowego o maksymalnej długości 100 znaków

```
ALTER TABLE parada ADD COLUMN lokalizacja VARCHAR(100);
```

tworzące tabelę klienci z kolumnami … klucz główny generowany automatycznie

```
CREATE TABLE klienci (
  id_klienta INT PRIMARY KEY AUTO_INCREMENT,
  imie VARCHAR(20) NOT NULL
);
```

klucz główny nadawany automatycznie (wstawianie rekordu)

```
INSERT INTO kategorie VALUES (NULL, 'Sandały');   -- NULL
w miejscu klucza
```

automatycznie wpisywana aktualna data

```
INSERT INTO rejestr (id_personelu, id_pojazdu, data)
VALUES (1, 14, CURDATE());
```

© Dawid Chaber · TablicaPro (tablicapro.pl) · materiał chroniony prawem autorskim — kopiowanie i udostępnianie bez zgody autora zabronione

pierwsze 100 znaków opisu

```
SELECT nazwa, LEFT(opis, 100) FROM gry WHERE id = 1;
```

cena wzrasta o 10% / obniżając cenę o 25%

```
UPDATE tatuaze SET cena = cena * 1.1 WHERE kolor =
'czerwony';
UPDATE opony SET cena = cena * 0.75 WHERE sezon =
'letnie';
```

tytuł niepusty (dowolny, niepusty)

```
… WHERE tytul <> '';
```

w nazwie zawiera się słowo „mebli” / cząstka „mir”

```
… WHERE zadanie LIKE '%mebli%';
```

w opisie znajdują się słowa „zdrowie” albo „opieką”

```
… WHERE opis LIKE '%zdrowie%' OR opis LIKE '%opieką%';
```

dodane w sierpniu 2021 roku

```
… WHERE data_dodania LIKE '2021-08-%';   albo
MONTH(d)=8 AND YEAR(d)=2021
```

mieści się w przedziale <100, 1000>

```
… WHERE cena BETWEEN 100 AND 1000;
```

id osób, które nie pojawiają się w rejestrze

```
SELECT id, nazwisko FROM personel
WHERE id NOT IN (SELECT id_personelu FROM rejestr);
```

zestawienie: wszystkie kursy i liczba zapisanych (także zero)

```
SELECT nazwa, COUNT(uczestnicy.id) AS "Zapisanych"
FROM kursy LEFT JOIN uczestnicy ON kursy.id = id_kursu
GROUP BY kursy.id;
```

wybierany jest tylko jeden losowy rekord (z relacją)

```
SELECT … FROM zamowienia JOIN opony ON … ORDER BY RAND()
LIMIT 1;
```

średnia zaokrąglona do liczby całkowitej, tylko dla lat po 2005

```
SELECT rok, ROUND(AVG(dlugosc)) AS "Średnia długość" FROM
parada JOIN smok ON …
WHERE rok > 2005 GROUP BY rok;
```

wyeksportuj dane z tabeli do pliku CSV

```
phpMyAdmin → tabela → Eksport → format CSV → Wykonaj
(nazwa pliku zgodna z treścią)
```

aktualizujące dane w rekordzie o id równym 5: pole podkategoria przyjmuje wartość 1

```
UPDATE ogloszenia SET podkategoria = 1 WHERE id = 5;
```

Typowe błędy

UPDATE albo DELETE bez WHERE

Zmienia lub usuwa wszystkie wiersze tabeli. Zawsze sprawdź warunek przed wykonaniem.

JOIN bez warunku ON

Powstaje iloczyn kartezjański — każdy wiersz z każdym. Dodaj ON t1.id = t2.t1_id.

Kolumna „id” jest niejednoznaczna

Obie tabele mają kolumnę o tej samej nazwie. Zapisz tabela.kolumna.

Alias w pojedynczych apostrofach w zrzucie

Aliasy ze spacjami zapisuj w cudzysłowie (lub w znakach `…`), a nazwę kolumny skopiuj z treści zadania.

WHERE kol = NULL

Użyj IS NULL / IS NOT NULL.

Agregat w WHERE

Warunek na wyniku COUNT/SUM/AVG należy do HAVING albo do podzapytania.

Zrzut ekranu wykadrowany

Zrzut ma obejmować cały ekran z paskiem zadań; zawiera zapytanie i wynik. Wykadrowane zrzuty nie są oceniane.

Zapytanie tylko „działa na tych danych”

Zapytanie ma być uniwersalne dla każdego zestawu danych — nie wpisuj wyniku na sztywno (np. WHERE id = 3, gdy treść pyta o najwyższy).

Opracowanie TablicaPro na podstawie analizy arkuszy części praktycznej egzaminu zawodowego (E.14, EE.09, INF.03) z lat 2016–2026 opublikowanych przez CKE i OKE: 123 arkuszy z treścią, w tym 70 z lat 2023–2026. Znaczniki częstości są szacunkiem na podstawie automatycznego wyszukiwania w treści zadań i tabelach pomocniczych — mają orientacyjny charakter. Arkusze są własnością CKE; tablica zawiera własne opisy i przykłady.

© Dawid Chaber · TablicaPro (tablicapro.pl) · materiał chroniony prawem autorskim — kopiowanie i udostępnianie bez zgody autora zabronione