

# Lekcja

**Temat:** Funkcje związane z czasem, datą, operatorami łańcuchowymi

**Funkcje daty i czasu**

**Link do dokumentacji MySQL:**

<https://dev.mysql.com/doc/refman/8.4/en/date-and-time-functions.html>

Metoda	Wyjaśnienie	Przykład SQL	Wynik
<b>ADDDATE()</b>	Dodaje interwał do daty	SELECT ADDDATE('2024-01-01', INTERVAL 5 DAY);	2024-01-06
<b>ADDTIME()</b>	Dodaje czas	SELECT ADDTIME('10:00:00','02:30:00');	12:30:00
<b>CONVERT_TZ()</b>	Konwersja strefy czasowej	SELECT CONVERT_TZ('2024-01-01 12:00','UTC','Europe/Warsaw');	2024-01-01 13:00
<b>CURDATE()</b>	Bieżąca data	SELECT CURDATE();	2025-11-10
<b>CURTIME()</b>	Bieżący czas	SELECT CURTIME();	np. 14:22:01
<b>DATE()</b>	Zwraca część datową	SELECT DATE('2024-01-01 10:00:00');	2024-01-01
<b>DATE_ADD()</b>	Dodaje interwał do daty	SELECT DATE_ADD('2024-01-01', INTERVAL 1 MONTH);	2024-02-01
<b>DATE_FORMAT()</b>	Formatuje datę	SELECT DATE_FORMAT('2024-01-15','%d-%m-%Y');	15-01-2024
<b>DATE_SUB()</b>	Odejmuje interwał	SELECT DATE_SUB('2024-01-10', INTERVAL 3 DAY);	2024-01-07
<b>DATEDIFF()</b>	Różnica między datami	SELECT DATEDIFF('2024-02-01','2024-01-01');	31

<b>DAY()</b>	Dzień miesiąca	<code>SELECT DAY('2024-01-15');</code>	15
<b>DAYNAME()</b>	Nazwa dnia	<code>SELECT DAYNAME('2024-01-15');</code>	Tuesday
<b>DAYOFMONTH()</b>	Dzień miesiąca	<code>SELECT DAYOFMONTH('2024-01-15');</code>	15
<b>DAYOFWEEK()</b>	Numer dnia tyg. (1=nd)	<code>SELECT DAYOFWEEK('2024-01-15');</code>	3
<b>DAYOFYEAR()</b>	Dzień roku	<code>SELECT DAYOFYEAR('2024-01-15');</code>	15
<b>EXTRACT()</b>	Wyodrębnia część daty	<code>SELECT EXTRACT(YEAR FROM '2024-01-15');</code>	2024
<b>FROM_DAYS()</b>	Dni → data	<code>SELECT FROM_DAYS(750000);</code>	2044-01-22
<b>FROM_UNIXTIME()</b>	UNIX → data	<code>SELECT FROM_UNIXTIME(1700000000);</code>	2023-11-14 22:13:20
<b>HOUR()</b>	Pobiera godzinę	<code>SELECT HOUR('12:45:00');</code>	12
<b>LAST_DAY()</b>	Ostatni dzień miesiąca	<code>SELECT LAST_DAY('2024-02-10');</code>	2024-02-29
<b>MAKEDATE()</b>	Tworzy datę z dnia roku	<code>SELECT MAKEDATE(2024,32);</code>	2024-02-01
<b>MAKETIME()</b>	Tworzy czas	<code>SELECT MAKETIME(10,20,30);</code>	10:20:30
<b>MICROSECOND()</b>	Mikrosekundy	<code>SELECT MICROSECOND('10:00:00.123456');</code>	123456

<b>MINUTE()</b>	Minuta	SELECT MINUTE('12:45:30');	45
<b>MONTH()</b>	Numer miesiąca	SELECT MONTH('2024-05-10');	5
<b>MONTHNAME()</b>	Nazwa miesiąca	SELECT MONTHNAME('2024-05-10');	May
<b>NOW()</b>	Aktualny datetime	SELECT NOW();	2025-11-10 14:20:xx
<b>PERIOD_ADD()</b>	Dodaje miesiące do YYYYMM	SELECT PERIOD_ADD(202401,2);	202403
<b>PERIOD_DIFF()</b>	Ilość miesięcy między okresami	SELECT PERIOD_DIFF(202402,202401);	1
<b>QUARTER()</b>	Kwartał	SELECT QUARTER('2024-05-10');	2
<b>SEC_TO_TIME()</b>	Sekundy → czas	SELECT SEC_TO_TIME(3661);	01:01:01
<b>SECOND()</b>	Sekundy	SELECT SECOND('12:45:59');	59
<b>STR_TO_DATE()</b>	Tekst → data	SELECT STR_TO_DATE('31-01-2024','%d-%m-%Y');	2024-01-31
<b>SUBTIME()</b>	Odejmuje czas	SELECT SUBTIME('10:00:00','01:30:00');	08:30:00
<b>SYSDATE()</b>	Czas wykonania	SELECT SYSDATE();	2025-11-10...

<b>TIME()</b>	Czas z datetime	<code>SELECT TIME('2024-01-01 12:30:45');</code>	12:30:45
<b>TIME_FORMAT()</b>	Formatuje czas	<code>SELECT TIME_FORMAT('12:30:45','%H:%i');</code>	12:30
<b>TIME_TO_SEC()</b>	Czas → sekundy	<code>SELECT TIME_TO_SEC('01:00:00');</code>	3600
<b>TIMEDIFF()</b>	Różnica czasu	<code>SELECT TIMEDIFF('12:00:00','10:00:00');</code>	02:00:00
<b>TIMESTAMP()</b>	Tworzy datetime	<code>SELECT TIMESTAMP('2024-01-01');</code>	2024-01-01 00:00:00
<b>TIMESTAMPADD()</b>	Dodaje interwał	<code>SELECT TIMESTAMPADD(HOUR,2,'2024-01-01 10:00');</code>	2024-01-01 12:00
<b>TIMESTAMPDIFF()</b>	Różnica datetime	<code>SELECT TIMESTAMPDIFF(DAY,'2024-01-01','2024-01-10');</code>	9
<b>TO_DAYS()</b>	Data → dni od roku 0	<code>SELECT TO_DAYS('2024-01-01');</code>	739252
<b>TO_SECONDS()</b>	Data → sekundy od roku 0	<code>SELECT TO_SECONDS('2024-01-01');</code>	64092288000
<b>UNIX_TIMESTAMP()</b>	Aktualny UNIX time	<code>SELECT UNIX_TIMESTAMP();</code>	np. 1768060000
<b>UTC_DATE()</b>	Data UTC	<code>SELECT UTC_DATE();</code>	2025-11-10

<b>UTC_TIME()</b>	Czas UTC	<code>SELECT UTC_TIME();</code>	13:14:xx
<b>UTC_TIMESTAMP()</b>	Datetime UTC	<code>SELECT UTC_TIMESTAMP();</code>	2025-11-10 13:14:xx
<b>WEEK()</b>	Numer tygodnia	<code>SELECT WEEK('2024-01-10');</code>	1
<b>WEEKDAY()</b>	Dzień tyg. (0=pon)	<code>SELECT WEEKDAY('2024-01-10');</code>	3
<b>WEEKOFYEAR()</b>	Tydzien ISO	<code>SELECT WEEKOFYEAR('2024-01-10');</code>	2
<b>YEAR()</b>	Rok	<code>SELECT YEAR('2024-01-10');</code>	2024
<b>YEARWEEK()</b>	Rok + tydzień	<code>SELECT YEARWEEK('2024-01-10');</code>	202402

**UTC (Uniwersalny Czas Koordynowany) to światowy standard czasu atomowego, który służy jako podstawa do ustalania lokalnego czasu w różnych strefach czasowych.** Polska znajduje się w strefie czasowej UTC+1 (czas środkowoeuropejski, CET) zimą i UTC+2 (czas środkowoeuropejski letni, CEST) latem, a lokalny czas w Polsce jest o 1 lub 2 godziny późniejszy od czasu UTC.

- **Co to jest UTC:**
  - UTC to międzynarodowy standard czasu, który jest niezależny od ruchu obrotowego Ziemi i oparty na bardzo precyzyjnym czasie atomowym.

- Jest to punkt odniesienia, taki sam na całym świecie, do którego dodaje się lub od którego odejmuje się czas, aby uzyskać lokalny czas dla danej strefy czasowej.
- **UTC w Polsce:**
  - Polska leży w strefie czasowej UTC+1 (czas zimowy) lub UTC+2 (czas letni).
  - Czas zimowy (CET): Obowiązuje od ostatniej niedzieli października do ostatniej niedzieli marca. Czas lokalny w Polsce jest o 1 godzinę późniejszy niż UTC. (np. jeśli UTC to 12:00, w Polsce jest 13:00).
  - Czas letni (CEST): Obowiązuje od ostatniej niedzieli marca do ostatniej niedzieli października. Czas lokalny w Polsce jest o 2 godziny późniejszy niż UTC. (np. jeśli UTC to 12:00, w Polsce jest 14:00).

#### Zastosowania:

- Programowanie - przechowywanie dat i czasu w bazach danych
- Lotnictwo - koordynacja lotów międzynarodowych
- Internet - synchronizacja serwerów
- Telekomunikacja - koordynacja transmisji
- Nauka - precyzyjne pomiary czasu

**W praktyce:** Gdy widzisz znacznik czasu typu [2025-11-11T14:30:00Z](#), litera "Z" na końcu oznacza właśnie UTC (od "Zulu time" - wojskowego określenia UTC).

#### Przykłady:

- Polska: UTC+1 (zimą) lub UTC+2 (latem)
- Nowy Jork: UTC-5 (zimą) lub UTC-4 (latem)
- Tokio: UTC+9
- Londyn: UTC+0 (zimą) lub UTC+1 (latem)

## Funkcje i operatory łańcuchowe

### Link do dokumentacji MySQL:

<https://dev.mysql.com/doc/refman/8.4/en/string-functions.html>

Metoda	Wyjaśnienie	Przykład	Wynik
<b>ASCII()</b>	Zwraca kod ASCII pierwszego znaku	<code>SELECT ASCII('A');</code>	65
<b>BIN()</b>	Zwraca liczbę w postaci binarnej	<code>SELECT BIN(10);</code>	1010
<b>BIT_LENGTH()</b>	Zwraca długość napisu w bitach	<code>SELECT BIT_LENGTH('ABC');</code>	24
<b>CHAR()</b>	Zwraca znak odpowiadający podanemu kodowi ASCII	<code>SELECT CHAR(65);</code>	'A'
<b>CHAR_LENGTH()</b>	Liczba znaków (nie bajtów)	<code>SELECT CHAR_LENGTH('Łódź');</code>	4
<b>CHARACTER_LENGTH()</b>	To samo co CHAR_LENGTH()	<code>SELECT CHARACTER_LENGTH('Test');</code>	4
<b>CONCAT()</b>	Łączy napisy	<code>SELECT CONCAT('A', 'B', 'C');</code>	'ABC'
<b>CONCAT_WS()</b>	Łączy napisy z separatorem	<code>SELECT CONCAT_WS('-', 'A','B','C');</code>	'A-B-C'

<b>ELT()</b>	Zwraca element listy na indeksie (1-based)	SELECT ELT(2,'jeden','dwa','trzy');	'dwa'
<b>EXPORT_SET()</b>	Zamienia liczby bitowe na tekst ON/OFF	SELECT EXPORT_SET(5, 'ON', 'OFF', '', 4);	ON,OFF,ON,OFF
<b>FIELD()</b>	Zwraca pozycję pierwszego argumentu w liście	SELECT FIELD('kot','pies','kot','mysz');	2
<b>FIND_IN_SET()</b>	Pozycja elementu w liście CSV	SELECT FIND_IN_SET('B', 'A,B,C');	2
<b>FORMAT()</b>	Formatuje liczbę z przecinkami	SELECT FORMAT(12345.678, 2);	'12,345.68'
<b>FROM_BASE64()</b>	Dekoduje Base64	SELECT FROM_BASE64('SGVsibG8=');	'Hello'
<b>HEX()</b>	Zamienia liczbę lub tekst na hex	SELECT HEX('ABC');	414243
<b>INSERT()</b>	Wstawia podciąg w podaną pozycję, zastępując określoną liczbę znaków	SELECT INSERT('abcdef', 3, 2, 'XYZ');	'abXYZef'
<b>INSTR()</b>	Pozycja pierwszego wystąpienia podciągu	SELECT INSTR('abcabc','ca');	3
<b>LCASE()</b>	To samo co LOWER() – zamienia na małe litery	SELECT LCASE('Test');	'test'

<b>LEFT()</b>	Zwraca określoną liczbę znaków od lewej	SELECT LEFT('abcdef', 3);	'abc'
<b>LENGTH()</b>	Długość napisu w bajtach	SELECT LENGTH('ABC');	3
<b>LIKE</b>	Sprawdza dopasowanie wzorca	SELECT 'Ala' LIKE 'A%';	1
<b>LOAD_FILE()</b>	Wczytuje zawartość pliku (jeśli SQL ma dostęp)	SELECT LOAD_FILE('/path/file.txt');	<i>treść pliku</i>
<b>LOCATE()</b>	Pozycja podcięgu (jak INSTR, ale kolejność argumentów odwrotna)	SELECT LOCATE('b','abc');	2
<b>LOWER()</b>	Zamienia na małe litery	SELECT LOWER('TEST');	'test'
<b>LPAD()</b>	Uzupełnia z lewej do zadanej długości	SELECT LPAD('7', 3, '0');	'007'
<b>LTRIM()</b>	Usuwa spacje z lewej	SELECT LTRIM(' test');	'test'
<b>MAKE_SET()</b>	Zwraca listę elementów pasujących do bitów liczby	SELECT MAKE_SET(5,'A','B','C');	'A,C'
<b>MATCH() AGAINST()</b>	Pełnotekstowe wyszukiwanie	SELECT MATCH(text) AGAINST('kot');	<i>ocena dopasowania</i>

<b>MID()</b>	Alias SUBSTRING()	<code>SELECT MID('abcdef', 2, 3);</code>	'bcd'
<b>NOT LIKE</b>	Odwrotność LIKE	<code>SELECT 'Ala' NOT LIKE 'K%';</code>	1
<b>NOT REGEXP</b>	Odwrotność REGEXP	<code>SELECT 'abc' NOT REGEXP '^[0-9]+\$';</code>	1
<b>OCT()</b>	Zamienia liczbę na system ósemkowy	<code>SELECT OCT(15);</code>	'17'
<b>OCTET_LENGTH()</b>	Alias LENGTH()	<code>SELECT OCTET_LENGTH('ABC');</code>	3
<b>ORD()</b>	Kod ASCII pierwszego znaku	<code>SELECT ORD('A');</code>	65
<b>POSITION()</b>	Alias LOCATE()	<code>SELECT POSITION('a' IN 'banan');</code>	2
<b>QUOTE()</b>	Zwraca tekst w bezpiecznej formie (escape)	<code>SELECT QUOTE("Ala's cat");</code>	'Ala\'s cat'
<b>REGEXP</b>	Dopasowanie wyrażenia regularnego	<code>SELECT 'abc123' REGEXP '[0-9]+';</code>	1
<b>REGEXP_INSTR()</b>	Pozycja dopasowania regexu	<code>SELECT REGEXP_INSTR('abc123','[0-9]+');</code>	4
<b>REGEXP_LIKE()</b>	Czy pasuje regex	<code>SELECT REGEXP_LIKE('test123','[a-z]+');</code>	1

<b>REGEXP_REPLACE()</b>	Zamienia dopasowane fragmenty	<code>SELECT REGEXP_REPLACE('a1b2c3','[0-9]','X');</code>	'aXbXcX'
<b>REGEXP_SUBSTR()</b>	Zwraca fragment pasujący do regexu	<code>SELECT REGEXP_SUBSTR('abc123','[0-9]+');</code>	'123'
<b>REPEAT()</b>	Powtarza tekst	<code>SELECT REPEAT('A',3);</code>	'AAA'
<b>REPLACE()</b>	Podmienia tekst	<code>SELECT REPLACE('ala ma kota','a','X');</code>	'XIX mX kotX'
<b>REVERSE()</b>	Odwraça napis	<code>SELECT REVERSE('kota');</code>	'atok'
<b>RIGHT()</b>	Znaki od prawej	<code>SELECT RIGHT('abcdef', 2);</code>	'ef'
<b>RLIKE</b>	Alias REGEXP	<code>SELECT 'abc' RLIKE '[a-z]+';</code>	1
<b>RPAD()</b>	Uzupełnia napis z prawej	<code>SELECT RPAD('A', 4, '.');</code>	'A...'
<b>RTRIM()</b>	Usuwa spacje z prawej	<code>SELECT RTRIM('test ');</code>	'test'
<b>SOUNDEX()</b>	Kod fonetyczny słów	<code>SELECT SOUNDEX('Robert');</code>	'R163'
<b>SOUNDS LIKE</b>	Porównanie brzmienia	<code>SELECT 'Robert' SOUNDS LIKE 'Rupert';</code>	1
<b>SPACE()</b>	Generuje spacje	<code>SELECT SPACE(5);</code>	' '
<b>STRCMP()</b>	Porównuje napisy	<code>SELECT STRCMP('abc','abd');</code>	-1

<b>SUBSTR()</b>	Podciąg (alias SUBSTRING)	<code>SELECT SUBSTR('abcdef',2,3);</code>	'bcd'
<b>SUBSTRING()</b>	Podciąg	<code>SELECT SUBSTRING('abcdef',3);</code>	'cdef'
<b>SUBSTRING_INDEX()</b>	Podciąg do N-tego separatora	<code>SELECT SUBSTRING_INDEX('a,b,c','',2);</code>	'a,b'
<b>TO_BASE64()</b>	Kodowanie Base64	<code>SELECT TO_BASE64('Hello');</code>	'SGVsbG8='
<b>TRIM()</b>	Usuwa spacje z obu stron	<code>SELECT TRIM(' test ');</code>	'test'
<b>UCASE()</b>	Alias UPPER()	<code>SELECT UCASE('abc');</code>	'ABC'
<b>UNHEX()</b>	Hex → tekst	<code>SELECT UNHEX('414243');</code>	'ABC'
<b>UPPER()</b>	Zamienia na wielkie litery	<code>SELECT UPPER('kot');</code>	'KOT'
<b>WEIGHT_STRING()</b>	Zwraca wewnętrzną wagę znaków (techniczne)	<code>SELECT WEIGHT_STRING('A');</code>	(hex bajty)

**Na następnej lekcji kartkówka z powyższych tabel. Wymagam znajomości metod napisanych czerwonym kolorem**

# Lekcja

**Temat:** ERD (Diagram związków encji ang. Entity Relationship Diagram). Właściwości kolumn (pół) w MySQL: NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, DEFAULT, CHECK, AUTO\_INCREMENT, ENUM, COMMENT. Polecenie DELETE i DROP

## ERD — diagram związków encji

To graficzny sposób przedstawienia struktury bazy danych:

- jakie **tabele (encje)** istnieją,
- jakie mają **atrybuty (kolumny)**,
- jakie występują **relacje** między tabelami:
  - **1:1**
  - **1:N**
  - **N:M**

ERD jest tworzony zanim powstanie baza danych, aby zaplanować jej strukturę.

**Encja (Entity)** = obiekt, który ma znaczenie w systemie i który chcesz zapisać w bazie.

Inaczej mówiąc:

- 👉 **Encja** = tabela w bazie danych
- 👉 **Atrybut** = kolumna w tabeli

Przykłady encji:

- **User (użytkownik)**
- **Product (produkt)**
- **Order (zamówienie)**
- **Invoice (faktura)**
- **Department (dział firmy)**

Każda encja ma klucz główny (Primary Key, PK) – unikalny identyfikator, np. id.

## Tworzenie krok po kroku diagramu związków encji

**Krok 1: Zidentyfikuj encje (tabele)**

**Krok 2: Określ atrybuty**

Dla każdej encji określasz pola.

Przykład:

Customer

- id
- first\_name
- last\_name

- email

### Krok 3: Ustal klucze główne

Każda encja ma PK:

### Krok 4: Określ relacje między encjami

1) Relacja 1:1 (One to One)

Jeden rekord odpowiada dokładnie jednemu rekordowi w drugiej tabeli.

2) Relacja 1:N (One to Many)

Jeden klient może mieć wiele zamówień.

3) Relacja N:M (Many to Many)

Tworzy się tabelę pośredniczącą.

#### 1. Relacja 1 : 1 (Osoba — Adres)

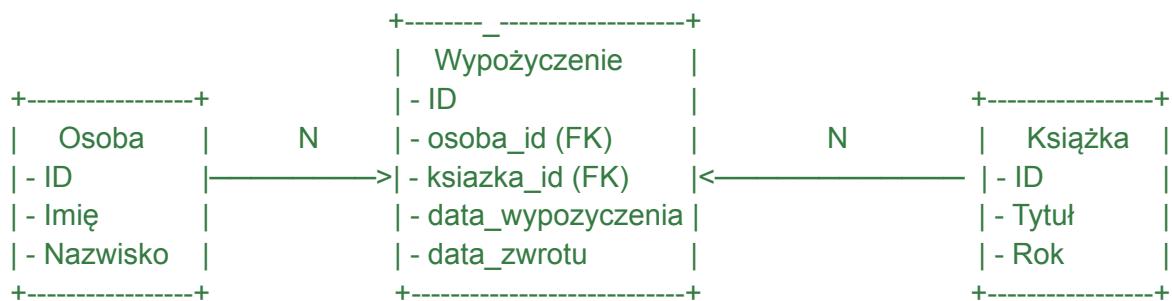


#### 2. Relacja 1 : N (Autor — Książka)



#### 3. Relacja N : N (Osoba — Książka) przez tabelę Wypożyczenie

W MySQL/SQL relacja N:N **zawsze wymaga tabeli pośredniej**.



N : N  
(wiele osób wypożycza wiele książek)

Opis reprezentacji graficznej stopnia związku został pokazany na rysunku

**Rysunek**

Graficzna reprezentacja  
związków zachodzą-  
cych między encjami



Opis reprezentacji graficznej opcjonalności związku został pokazany na rysunku

**Rysunek**

Graficzna reprezentacja  
opcjonalności związku



Diagramy ERD możemy tworzyć za pomocą różnych notacji. Najpopularniejsze są diagramy w zapisie według Martina i Chena.

**Właściwości kolumn (pól) w MySQL:** NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, DEFAULT, CHECK, AUTO\_INCREMENT, ENUM, COMMENT

W MySQL możesz nałożyć **wiele rodzajów właściwości (constraints)** na pojedynczą kolumnę albo na kilka kolumn naraz, żeby wymusić reguły zachowania danych.

**✓ 1. NOT NULL**

Kolumna **nie może przyjmować wartości NULL**.  
Wymusza, że musisz zawsze podać wartość.

**Przykład:**

```
CREATE TABLE osoby (
    id INT NOT NULL,
    imie VARCHAR(100) NOT NULL
);
```

### Wyjaśnienie:

- imię i id **musi** być podane.

## ✓ 2. UNIQUE

Wymusza **unikalne wartości** w kolumnie — nie mogą się powtarzać.

### Przykład:

```
CREATE TABLE klienci (
    id INT PRIMARY KEY,
    email VARCHAR(255) UNIQUE
);
```

### Wyjaśnienie:

Dwa takie same maile → **X** błąd.

Można też ustawić UNIQUE na **kilka kolumn naraz**:

```
UNIQUE (uczen_id, kurs_id)
```

### Dodanie UNIQUE do istniejącej tabeli

```
ALTER TABLE klienci
```

```
ADD UNIQUE (email);
```

## Różnica: PRIMARY KEY vs UNIQUE

Cecha	PRIMARY KEY	UNIQUE
Musi być unikalne	✓ Tak	✓ Tak
Może być NULL	✗ Nie	✓ Tak
Można mieć więcej niż jeden?	✗ Nie (tylko jeden PK na tabelę)	✓ Tak (wiele UNIQUE)
Tworzy indeks	✓ Tak	✓ Tak

## ✓ 3. PRIMARY KEY

- jednoznacznie identyfikuje każdy wiersz (unikalny),
- automatycznie ma **UNIQUE + NOT NULL**.

### Przykład:

```
CREATE TABLE produkty (
    produkt_id INT PRIMARY KEY,
    nazwa VARCHAR(100)
);
```

Możesz też zrobić klucz **złożony z kilku kolumn**:

```
PRIMARY KEY (zamowienie_id, produkt_id)
```

## 4. FOREIGN KEY

Łączy tabele — kolumna musi wskazywać na wartość z innej tabeli.

**Przykład:**

```
CREATE TABLE zamowienia (
```

```
    id INT PRIMARY KEY
```

```
);
```

```
CREATE TABLE produkty_w_zamowieniu (
```

```
    zamowienie_id INT,
```

```
    produkt_id INT,
```

```
    FOREIGN KEY (zamowienie_id) REFERENCES zamowienia(id)
```

```
);
```

Nie można dodać produktu do zamówienia, które nie istnieje.

## 5. DEFAULT

Ustawia **wartość domyślną**, jeśli użytkownik nie poda swojej.

**Przykład:**

```
CREATE TABLE artykuły (
```

```
    id INT PRIMARY KEY,
```

```
    status VARCHAR(20) DEFAULT 'aktywny'
```

```
);
```

Jeśli nie podasz statusu → automatycznie będzie „aktywny”.

## 6. CHECK

Wymusza spełnienie **logicznego warunku**.

**Przykład:**

```
CREATE TABLE pracownicy (
```

```
    id INT PRIMARY KEY,
```

```
    wiek INT CHECK (wiek >= 18 AND wiek <= 65)
```

```
);
```

Próba dodania `wiek = 10` →  błąd.

## 7. AUTO\_INCREMENT

Automatycznie zwiększa wartość w kolumnie liczbowej przy każdym INSERT.

**Przykład:**

```
CREATE TABLE logi (
```

```
    id INT PRIMARY KEY AUTO_INCREMENT,
```

```
    opis VARCHAR(255)
```

```
);
```

Dodajesz 5 logów → id będą: 1, 2, 3, 4, 5.

## ✓ 8. ENUM

Ogranicza wartości w kolumnie do **zamkniętej listy dopuszczalnych opcji**.

**Przykład:**

```
CREATE TABLE użytkownicy (
    id INT PRIMARY KEY,
    plec ENUM('M', 'K', 'INNE') DEFAULT 'INNE'
);
```

Próba zapisania plec = 'ABC' → ✗ błąd.

## ✓ 9. COMMENT

Pozwala dopisać **komentarz** do kolumny — bardzo przydatne przy dokumentowaniu schematu.

**Przykład:**

```
CREATE TABLE produkty (
    id INT PRIMARY KEY,
    cena DECIMAL(10,2) COMMENT 'Cena brutto w zł'
);
```

W narzędziach typu phpMyAdmin, DBeaver zobaczysz komentarz przy kolumnie.

## Polecenie **DELETE i DROP**

### ◆ **DELETE FROM**

**Polecenie:**

```
DELETE FROM nazwa_tabeli;
```

**Usuwa rekordy (wiersze) z tabeli**, ale:

- **nie usuwa struktury tabeli**, kolumn ani jej definicji,
- **nie resetuje auto\_increment** (chyba że użyjesz TRUNCATE),
- może usuwać pojedyncze wiersze lub wszystkie — zależnie od warunku WHERE.

**Przykłady:**

Usuń wszystkie rekordy:

```
DELETE FROM users;
```

Usuń tylko wybrane:

```
DELETE FROM users WHERE id = 5;
```

## ◆ **DROP**

**Polecenie:**

**DROP TABLE nazwa\_tabeli;**

**Usuwa całą tabelę z bazy danych,** czyli:

- usuwa wszystkie dane,
- usuwa strukturę tabeli (kolumny, indeksy, klucze),
- usuwa definicję tabeli z katalogu bazy.

Po wykonaniu DROP tabela **przestaje istnieć.**

**Przykłady:**

Usuń tabelę:

**DROP TABLE users;**

Usuń całą bazę danych:

**DROP DATABASE sklep;**

**Na tej lekcji będzie kartkówka**

- z funkcji związanych z czasem, datą, operatorami łańcuchowymi.
- **POPRAWA SPRAWDZIANU**

**Na najbliższej lekcji kartkówka diagramów ERD, właściwości kolumn, polecenie drop i delete**