

PHP proceduralnie — schematy pracy z bazą danych (INF.03)

Jak korzystać z tej instrukcji

Każdy schemat to gotowy szablon: kopiujesz go, zmieniasz nazwę tabeli, kolumn i znaczniki HTML zgodnie z poleceniem w zadaniu. Przykłady korzystają z bazy biblioteka i tabeli książki (kolumny: id, tytuł, autor, rok, kategoria).

Każdy skrypt zaczyna się tak samo — od połączenia z bazą — i kończy zamknięciem połączenia:

```
<?php
// 1. połączenie: serwer, użytkownik, hasło, nazwa bazy
$con = mysqli_connect("localhost", "root", "", "biblioteka");

// ... tutaj zapytanie i wyświetlanie (schematy poniżej) ...

// ostatni krok: zamknięcie połączenia
mysqli_close($con);
?>
```

Kod PHP wstawiasz dokładnie w tym miejscu strony HTML, w którym ma pojawić się wynik (np. wewnątrz <section> albo <div> wskazanego w zadaniu).

Schemat 1. Wyświetlenie jednej wartości (MIN, MAX, COUNT, AVG, SUM)

Gdy zapytanie zwraca jeden wynik (np. najmniejszą lub największą wartość), nie potrzebujesz pętli — pobierasz jeden wiersz i wypisujesz go.

1. Napisz zapytanie z funkcją agregującą i nadaj wynikowi nazwę przez AS.
2. Wykonaj zapytanie funkcją `mysqli_query()`.
3. Pobierz jeden wiersz funkcją `mysqli_fetch_array()`.
4. Wypisz wartość po nazwie z AS.

```
<?php
$con = mysqli_connect("localhost", "root", "", "biblioteka");

// AS nadaje nazwę wynikom - po tej nazwie je odczytamy
$sql = "SELECT MIN(rok) AS najstarsza, MAX(rok) AS najnowsza FROM książki";
$wynik = mysqli_query($con, $sql);

// jeden wiersz wyniku - bez pętli
$w = mysqli_fetch_array($wynik);

echo "<p>Najstarsza książka: " . $w['najstarsza'] . "</p>";
echo "<p>Najnowsza książka: " . $w['najnowsza'] . "</p>";

mysqli_close($con);
?>
```

Tak samo działają inne funkcje: `COUNT(*) AS ile`, `AVG(rok) AS srednia`, `SUM(cena) AS suma`. Zmieniasz tylko zapytanie i nazwę w `$w['...']`.

Bez AS też się da: `mysqli_fetch_array()` zwraca kolumny także po numerach, więc odczytasz wartości po kolejności — `$w[0]`, `$w[1]`.

Schemat 2. Wyświetlanie wielu rekordów w pętli

Gdy zapytanie zwraca wiele wierszy, pętla `while` pobiera je po jednym, aż się skończy.

1. Wykonaj zapytanie `SELECT` funkcją `mysqli_query()`.
2. W warunku pętli `while` pobieraj kolejny wiersz funkcją `mysqli_fetch_array()`.
3. W środku pętli wypisz kolumny po nazwach: `$w['nazwa_kolumny']`.

```
<?php
$con = mysqli_connect("localhost", "root", "", "biblioteka");

$sql = "SELECT tytuł, autor, rok FROM ksiazki ORDER BY rok";
$wynik = mysqli_query($con, $sql);

// pętla wykonuje się tyle razy, ile jest wierszy
while ($w = mysqli_fetch_array($wynik)) {
    echo "<p>" . $w['tytuł'] . ", " . $w['autor'] . " (" . $w['rok'] . ")</p>";
}

mysqli_close($con);
?>
```

Wersja z tabelą HTML — nagłówek wypisujesz przed pętlą, a w pętli tylko wiersze `<tr>`:

```
<?php
echo "<table>";
echo "<tr><th>Tytuł</th><th>Autor</th><th>Rok</th></tr>";
while ($w = mysqli_fetch_array($wynik)) {
    echo "<tr><td>" . $w['tytuł'] . "</td><td>" . $w['autor'] . "</td><td>" . $w['rok'] .
    "</td></tr>";
}
echo "</table>";
?>
```

Zasada: to, co ma się pojawić raz (np. `<table>`, `<ul>`, `<ol>`), piszesz przed i po pętli; to, co powtarza się dla każdego rekordu (`<tr>`, `<li>`, `<p>`), piszesz wewnątrz pętli.

Schemat 3. Sprawdzenie, czy są rekordy: lista albo komunikat

Funkcja `mysqli_num_rows()` zwraca liczbę znalezionych wierszy. Jeśli jest większa od zera — generujesz listę, w przeciwnym razie wypisujesz komunikat.

1. Wykonaj zapytanie `SELECT`.
2. Sprawdź warunek `mysqli_num_rows($wynik) > 0`.
3. Jeśli prawda: otwórz `<ul>`, w pętli wypisz `<li>`, zamknij `</ul>`.
4. Jeśli fałsz (else): wypisz komunikat o braku rekordów.

```
<?php
$con = mysqli_connect("localhost", "root", "", "biblioteka");

$sql = "SELECT tytuł FROM ksiazki WHERE kategoria = 'Fantastyka'";
```

```

$wynik = mysqli_query($con, $sql);

if (mysqli_num_rows($wynik) > 0) {
    // są rekordy - generujemy listę
    echo "<ul>";
    while ($w = mysqli_fetch_array($wynik)) {
        echo "<li>" . $w['tytul'] . "</li>";
    }
    echo "</ul>";
} else {
    // brak rekordów - komunikat
    echo "<p>Brak rekordów w bazie.</p>";
}

mysqli_close($con);
?>

```

Lista numerowana: zamień na . Tabela: zamień na <table> z nagłówkiem, a na <tr><td>...</td></tr> (jak w schemacie 2).

Schemat 4. Wstawienie rekordu po kliknięciu przycisku

Formularz i skrypt są w jednym pliku. Kod zapisu wykonuje się tylko wtedy, gdy przycisk został kliknięty — sprawdza to `isset($_POST['nazwa_przycisku'])`. Poniżej dwie wersje zapisu: najpierw prostsza (zmienne wklejone w zapytanie), potem bezpieczna (zapytanie przygotowane).

Formularz (wspólny dla obu wersji)

Przycisk musi mieć atrybut `name` (tu `name="zapisz"`), a formularz `method="post"`. `action=""` oznacza, że dane trafiają do tego samego pliku.

```

<form action="" method="post">
    Tytuł: <input type="text" name="tytul"><br>
    Autor: <input type="text" name="autor"><br>
    Rok: <input type="number" name="rok"><br>
    <button type="submit" name="zapisz">Zapisz</button>
</form>

```

Wersja A. Prosta — zmienne wklejone w zapytanie (niebezpieczna)

1. Sprawdź `if (isset($_POST['zapisz']))` — bez tego skrypt próbowałby zapisywać przy każdym otwarciu strony.
2. Odbierz dane z `$_POST`.
3. Połącz się z bazą.
4. Wpisz zmienne bezpośrednio w treść `INSERT` i wykonaj je funkcją `mysqli_query()`.
5. Wypisz komunikat i zamknij połączenie.

```

<?php
// czy przycisk "zapisz" został kliknięty?
if (isset($_POST['zapisz'])) {

    // odbieramy dane z formularza
    $tytul = $_POST['tytul'];
    $autor = $_POST['autor'];

```

```

$rok    = $_POST['rok'];

$con = mysqli_connect("localhost", "root", "", "biblioteka");

// zmienne wklejone w zapytanie: tekst w apostrofach, liczba bez
$sql = "INSERT INTO ksiazki (tytul, autor, rok) VALUES ('$tytul', '$autor', $rok)";

// mysqli_query zwraca true, gdy zapis się udał
if (mysqli_query($con, $sql)) {
    echo "<p>Dodano rekord do bazy.</p>";
} else {
    echo "<p>Błąd zapisu.</p>";
}

mysqli_close($con);
}
?>

```

Uwaga: ta wersja jest podatna na SQL Injection — użytkownik może wpisać w pole fragment kodu SQL, a nawet zwykły apostrof w tytule (np. Rock 'n' roll) psuje zapytanie. Działa, ale na egzaminie za bezpieczeństwo lepiej użyć wersji B.

Wersja B. Bezpieczna — zapytanie przygotowane

Kroki 1–3 i 5 są takie same jak w wersji A. Zmienia się tylko krok 4: zamiast zmiennych w zapytaniu wpisujesz znaki ?, a dane podpinasz osobno.

1. Napisz zapytanie z ? w miejscu każdej wartości.
2. Przygotuj je funkcją `mysqli_prepare()`.
3. Podepnij zmienne funkcją `mysqli_stmt_bind_param()` — litery typów: s tekst, i liczba całkowita.
4. Wykonaj funkcją `mysqli_stmt_execute()`.

```

<?php
if (isset($_POST['zapisz'])) {

    $tytul = $_POST['tytul'];
    $autor = $_POST['autor'];
    $rok    = $_POST['rok'];

    $con = mysqli_connect("localhost", "root", "", "biblioteka");

    // zapytanie z parametrami ? - ochrona przed SQL Injection
    $sql = "INSERT INTO ksiazki (tytul, autor, rok) VALUES (?, ?, ?)";
    $st = mysqli_prepare($con, $sql);

    // "ssi" = string, string, integer - po jednej literze na każdy ?
    mysqli_stmt_bind_param($st, "ssi", $tytul, $autor, $rok);

    if (mysqli_stmt_execute($st)) {
        echo "<p>Dodano rekord do bazy.</p>";
    } else {
        echo "<p>Błąd zapisu.</p>";
    }
}

```

```

    mysqli_close($con);
}
?>

```

Jeśli zadanie każe sprawdzić, czy pola nie są puste, dopisz warunek: `if (!empty($_POST['tytul']) && !empty($_POST['autor']))`, a w `else` wypisz komunikat o brakujących danych.

Przykład: sprawdzenie, czy pola nie są puste

Warunek `empty()` jest w środku `isset()`: najpierw sprawdzasz, czy kliknięto przycisk, potem — czy pola są wypełnione. Zapis wykonuje się tylko wtedy, gdy oba warunki są spełnione.

1. `isset($_POST['zapisz'])` — czy przycisk został kliknięty.
2. `!empty($_POST['pole'])` dla każdego wymaganego pola, połączone `&&` (i).
3. Jeśli wszystko wypełnione — zapis do bazy i komunikat o dodaniu.
4. `else` — komunikat o brakujących danych.

```

<?php
if (isset($_POST['zapisz'])) {

    // czy wszystkie pola są wypełnione?
    if (!empty($_POST['tytul']) && !empty($_POST['autor']) && !empty($_POST['rok'])) {

        $tytul = $_POST['tytul'];
        $autor = $_POST['autor'];
        $rok    = $_POST['rok'];

        $con = mysqli_connect("localhost", "root", "", "biblioteka");

        $sql = "INSERT INTO ksiazki (tytul, autor, rok) VALUES (?, ?, ?)";
        $st = mysqli_prepare($con, $sql);
        mysqli_stmt_bind_param($st, "ssi", $tytul, $autor, $rok);
        mysqli_stmt_execute($st);

        echo "<p>Dodano rekord do bazy.</p>";

        mysqli_close($con);

    } else {
        // co najmniej jedno pole jest puste
        echo "<p>Uzupełnij wszystkie pola formularza.</p>";
    }
}
?>

```

W środku możesz użyć zapisu z wersji A albo B — warunek `empty()` wygląda tak samo. `!` oznacza zaprzeczenie: `!empty()` czytaj jako „nie jest puste”.

Ściąga: funkcje i częste błędy

Funkcja	Do czego służy
<code>mysqli_connect()</code>	Połączenie z bazą (serwer, użytkownik, hasło, baza)
<code>mysqli_query()</code>	Wykonanie zapytania bez danych od użytkownika
<code>mysqli_fetch_array()</code>	Pobranie jednego wiersza — kolumny po nazwie (<code>\$w['tytuł']</code>) lub po numerze (<code>\$w[0]</code>)
<code>mysqli_fetch_row()</code>	Pobranie jednego wiersza jako tablicy z numerami (0, 1, 2...)
<code>mysqli_num_rows()</code>	Liczba wierszy w wyniku
<code>mysqli_prepare()</code> + <code>mysqli_stmt_bind_param()</code> + <code>mysqli_stmt_execute()</code>	Bezpieczne zapytanie z danymi z formularza
<code>isset(\$_POST['...'])</code>	Sprawdzenie, czy przycisk został kliknięty
<code>mysqli_close()</code>	Zamknięcie połączenia

Najczęstsze błędy:

- Brak AS przy MIN/MAX, a potem odwołanie `$w['najstarsza']` do nieistniejącej nazwy.
- Pętla przy jednej wartości lub jej brak przy wielu rekordach — wypisuje się tylko pierwszy wiersz.
- `<ul>` lub `<table>` wewnątrz pętli zamiast przed nią — każdy rekord trafia do osobnej listy.
- Przycisk bez atrybutu name — `isset()` nigdy nie zwróci prawdy.
- Nazwa w `$_POST['...']` inna niż atrybut name w formularzu.
- Sklejanie zmiennych z formularza w treści zapytania zamiast parametrów ?.