

# MySQL Базовий

Підготовка запитів

# MySQL Базовий

## Introduction



Хаджийський Ян  
middle PHP developer

 /yanchuga

 /@hackmymozg



# MySQL Базовий

## Тема уроку

### Підготовка запитів

# MySQL Базовий

## План уроку

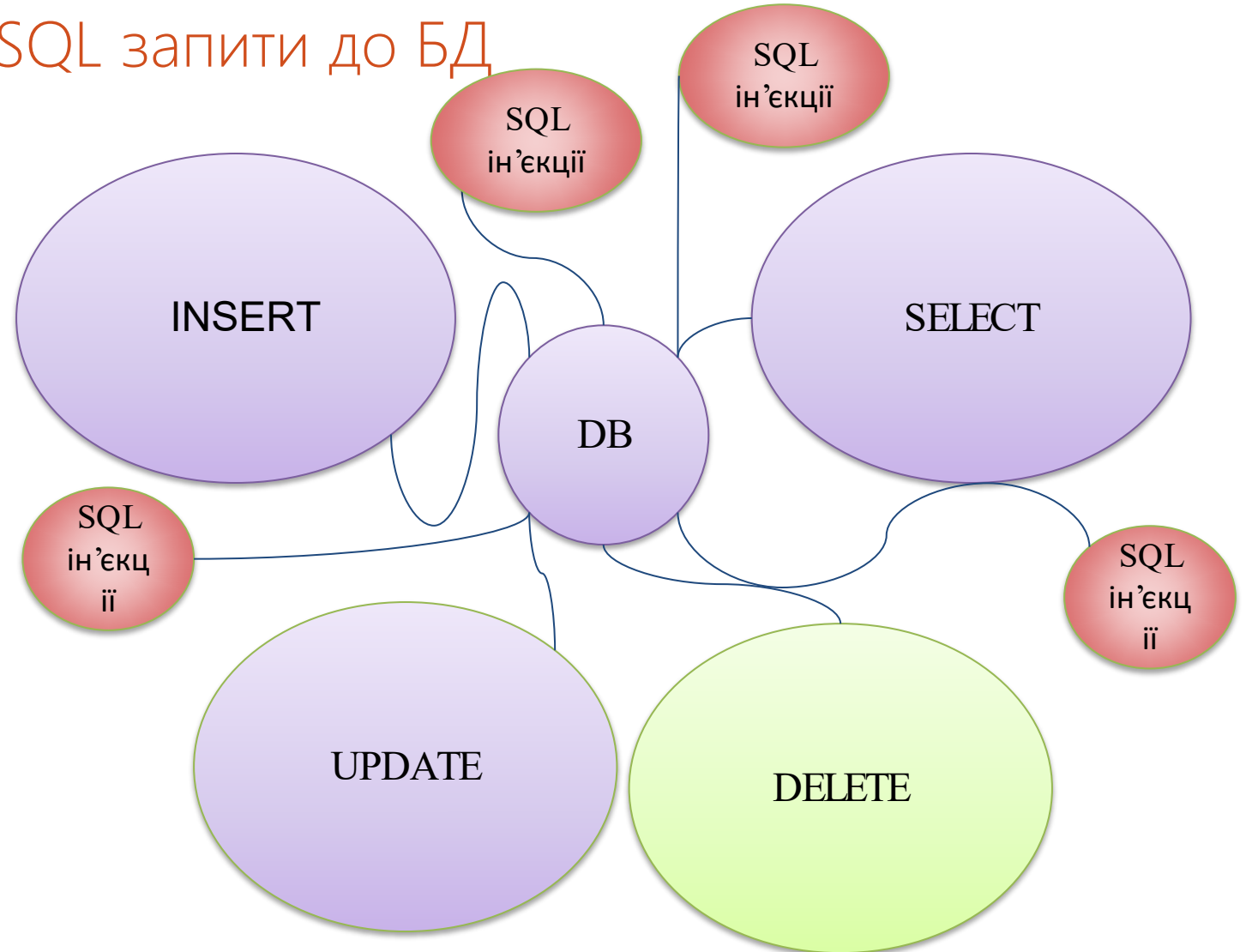
1. Робимо наші запити безпечними. Захищаємося від SQL injections
2. Практикуємо Prepare STMT, Execute
3. Робимо вибірку всіх людей, які мають більше 3 дітей та живуть у містах
4. Використовуємо @ для таблиць та query

# MySQL Базовий

## SQL запити до БД

◆ **Базові методи**  
80% всіх запитів припадає на INSERT, SELECT та UPDATE

◆ **Непередбачувані Запити**  
Краще захистити змінні і не давати шанс хакерам



# MySQL Базовий

## Практичне застосування

Довідка про Prepared: [https://www.tutorialspoint.com/mysql/mysql\\_prepare\\_statement.htm](https://www.tutorialspoint.com/mysql/mysql_prepare_statement.htm)

### Застосування функцій

Підготовлені запити повністю повторюють звичайні

Головна перевага - це безпека та прив'язка змінних до SQL queries

- PREPARE stmt — створення виразу
- EXECUTE stmt — виконання запиту
- DEALLOCATE PREPARE stmt — видалення запиту з пам'яті сервера

# MySQL Базовий

## Основні команди та функції

Змінні  
передаються  
прямо

Змінні  
передаються по  
масці

```
PREPARE people_child FROM 'SELECT * FROM `people_tbl` WHERE  
has_child > ?';  
SET @child = 3;  
EXECUTE people_child USING @child;
```

# MySQL Базовий

## Підстановка змінних

Змінні  
передаються в  
стрічці через ?

```
PREPARE people_child FROM 'SELECT * FROM `people_tbl` WHERE  
has_child > ?';  
SET @child = 3;  
EXECUTE people_child USING @child;
```

Стрічка  
складається через  
CONCAT

```
SET @statement_join = CONCAT('SELECT ', @table_from, '.OrderID', ' ',  
@table_on, '.PersonID FROM ',
```



# MySQL Базовий

## Запит INNER JOIN для 2 таблиць

Не використовуємо змінні та запит не можна використати повторно

```
SELECT Orders.OrderID, Persons.PersonID  
FROM Orders  
INNER JOIN Persons ON Persons.PersonID = Orders.PersonID;
```

# MySQL Базовий

## Використання spread operator

Можемо не обмежуватися кількістю елементів.

Починаючи з PHP 7.4 маємо spread operator для додавання від 1 і до бескінечності даних.

```
$values = [1, 2, 3, 4];  
  
$stmt = $mysqli->prepare("INSERT INTO test(id) VALUES (?), (?), (?), (?");  
$stmt->bind_param('iiii', ...$values);  
$stmt->execute();
```

# MySQL Базовий

## Деякі особливості роботи з stmt

Не забуваємо, що PHP буферизує запит на сервері і використовує всі потужності. RAM має свої ліміти.

Тому об'єкт запиту буде видалений мусорщиком, коли буде обхід всіх змінних або тільки після завершення роботи скрипта.

Одже, після виконання підготовленого запита закривайте його.

```
if (preg_match_all('/%(\w)/sim', $sql, $types)) {  
    $sql = preg_replace('/%(\w)/sim', '?', $sql);  
    if ($stmt = $db->prepare($sql)) {  
        $stmt->bind_param(implode('', $types[1]), ...$params);  
        $stmt->execute();  
        $query = $stmt->get_result();  
        $stmt->close();  
    }  
}
```

# MySQL Базовий

Ви можете прив'язати запит до змінних

```
/* prepare statement */
$stmt = $mysqli->prepare("SELECT Code, Name FROM Country ORDER BY Name LIMIT 5");
$stmt->execute();

/* bind variables to prepared statement */
$stmt->bind_result($col1, $col2);

/* fetch values */
while ($stmt->fetch()) {
    printf("%s %s\n", $col1, $col2);
}
```

# MySQL Базовий

## Буферизація даних

Об'єкти запитів за замовчуванням повертають небуферизовані результуючі набори. Ці таблиці ніяким чином не переносяться на клієнта.

Можна також буферизувати дані результуючих таблиць підготовленого запиту за допомогою функції `mysqli_stmt::store_result()`.

Можна циклом пройтись за результатом буферизації через `get_result`

Також можна буферизувати набір для подальшого використання

```
$query = "SELECT Name, CountryCode FROM City ORDER BY Name LIMIT 20";  
$stmt = $mysqli->prepare($query);  
$stmt->execute();  
  
/* сохранение результата во внутреннем буфере */  
$stmt->store_result();
```

Довідка про Store: <https://www.php.net/manual/en/mysqli-stmt.store-result.php>

# MySQL Базовий

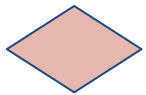
## Різниця між звичайним та stmt

Comparison of prepared and non-prepared statements		
	Prepared Statement	Non-prepared statement
Client-server round trips, SELECT, single execution	2	1
Statement string transferred from client to server	1	1
Client-server round trips, SELECT, repeated (n) execution	1 + n	n
Statement string transferred from client to server	1 template, n times bound parameter, if any	n times and parsed every time
Input parameter binding API	Yes	No, manual input escaping
Output variable binding API	Yes	No
Supports use of mysqli_result API	Yes, use <a href="#">mysqli_stmt::get_result()</a>	Yes
Buffered result sets	Yes, use <a href="#">mysqli_stmt::get_result()</a> or binding with <a href="#">mysqli_stmt::store_result()</a>	Yes, default of <a href="#">mysqli::query()</a>
Unbuffered result sets	Yes, use output binding API	Yes, use <a href="#">mysqli::real_query()</a> with <a href="#">mysqli::use_result()</a>

<https://www.php.net/manual/en/mysqli.quickstart.prepared-statements.php>

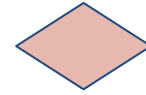
# MySQL Базовий

## Висновки



Інколи можна обійтися звичайним

Коли ви не дістаєте `secure` дані і не треба перевіряти на тип



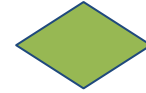
Непідготовлені запити краще?

Теоретично він проходить швидше через те, що не потрібно збирати змінні та буферизувати запит



Але краще не ризикувати

Якщо в змінну попаде щось інше, SQL ін'єкція неможлива



Для PHP API робимо `stmt`

Тут вже не до жартів, тому одразу робимо Prepared `stmt`

# Інформаційний відеосервіс для розробників програмного забезпечення





# Перевірка знань

TestProvider.com



Перевірте, як ви засвоїли даний матеріал на [TestProvider.com](https://testprovider.com)

TestProvider – це online-сервіс перевірки знань з інформаційних технологій. За його допомогою ви можете оцінити свій рівень та виявити слабкі місця. Він буде корисним як у процесі вивчення технології, так і для загальної оцінки знань IT-спеціаліста.

Успішне проходження фінального тестування дозволить вам отримати відповідний Сертифікат.

# MySQL Базовий

Дякую за увагу! До нової зустрічі!



Хаджийський Ян  
middle PHP developer

