Prepared statement

0513 이유경

사전 지식

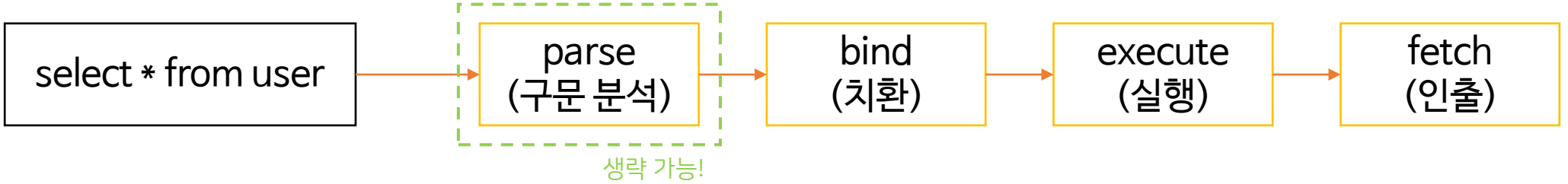
- DBMS: 데이터베이스 관리 시스템
  - 사용자나 다른 프로그램의 요구를 처리하고 적절히 응답하여 데이터를 사용할 수 있게 함

Statement *VS* Prepared Statement

Statement	*Prepared Statement*
쿼리문을 실행할 때 생성	쿼리문을 미리 생성
반복 실행 시 효율 하락	반복 실행 시 효율 상승
SQL Injection 공격에 취약	SQL Injection 공격이 발생하지 않음

Prepared Statement 작동 원리

- parse, bind, execute, fetch



SQL Injection 공격에 안전한 이유

- 바인딩 데이터는 SQL 문법이 아닌 내부의 인터프리터나 컴파일 언어로 처리하기 때문에 문법적인 의미를 가질 수 없음
- 바인딩 변수에 SQL 공격 쿼리를 입력할지라도 의미 있는 쿼리로 동작하지 않음

- 즉, Prepared Statement에서 바인딩 변수를 사용했을 때, 쿼리의 문법 처리 과정이 미리 선수행되기 때문에 바인딩 데이터는 SQL 문법적인 의미를 가질 수 없음

사용 예제

- Statement

```
Connection conn = DriverManager.getConnection(url,id,pass);
String sql = "SELECT * FROM TABLE WHERE = '" + "FIND_VALUE" + "'";
Statement stmt =  conn.createStatement();
ResultSet rs =   stmt.executeQuery(sql);
```

- Prepared Statement

```
Connection conn = DriverManager.getConnection(url,id,pass);
String sql = "SELECT * FROM TABLE WHERE =?";
PreparedStatement pstmt =  conn.prepareStatement(sql);
pstmt.setString(1, "FIND_VALUE"); // Parameter 할당
ResultSet rs =   pstmt.executeQuery();
```

예시

- 기존 php

```php
$id=$_GET['id'];
$pw=$_GET['pw'];

$query = "insert into user (id, password) values ('$id', '$pw')";
```

- Prepared Statement

```php
<?php
    $db = mysqli_connect('127.0.0.1', 'root', '', '20210318');
    $query ="select * from board";
    $result = $db->query($query);
    session_start();

// prepare and bind
$stmt = $conn->prepare("INSERT INTO  user (id, password) VALUES (?, ?)");
$stmt->bind_param("is", $id, $password);

// set parameters and execute
$id = "jungwoo@nct.com";
$password = "kimjungwoo";
$stmt->execute();


$id = "yukyeong@nctzen.com";
$password = "leeyukyeong";
$stmt->execute();


$stmt->close();
$conn->close();

?>
```