SQL6

SQLインジェクション対策

- 人間が入力した値は、信頼性に欠ける（Dirtyという）ため、そのままSQL文に埋め込んではいけない。
- 不正な文字列を渡すことで、予想しない操作され、DB上のデータの取得・削除などが起こりうる。

危険な例

```
  ⋮
$age  = $_POST['age'];

  ⋮
$sql = "SELECT * FROM table WHERE age > $age";
$result = $pdo->query($sql);
  ⋮
```

回避する方法1

```
  ⋮
$age  = $_POST['age'];

  ⋮
$sql = "SELECT * FROM table WHERE age > ?";
$result = $pdo->prepare($sql);
$result->execute(array($age));
  ⋮
```

- PDO::prepare() で、あらかじめ（テンプレート化した）SQL文を渡す。
- パラメータに疑問符パラメータを用いる
- その後、PDOStatement::execute() で、疑問符パラメータに値を渡し（？の順に配列に格納して）実行する。

回避する方法2

```
  ⋮
$age  = $_POST['age'];

  ⋮
$sql = "SELECT * FROM table WHERE age = :age";
$result = $pdo->prepare($sql);
$result->execute(array(':age' => $age));
  ⋮
```

- パラメータに名前付きパラメータを用いる
- 順序に関係なく、値を渡せる。

その他

- 名前付きパラメータに値を設定する場合に、個別に設定する方法もある。
    - PDOStatement::bindParam()
    - PDOStatement::bindValue()

課題

過去に作った提出課題に対し、SQLインジェクション対策を講じなさい。