

データベース利用実習

PHPとMySQLの連携

株式会社ジードライブ

この講義で学ぶこと

- データベースのユーザを追加する方法
- PHPからMySQLを利用する方法
- データベースに対する攻撃と対策
- 例外処理

データベースのユーザについて

- データベースをルートユーザ（root）で利用するのは危険である
- 通常は、操作権限を制限したユーザアカウントを作り、そのユーザアカウントでデータベースを操作する

データベースへのユーザ追加方法

1. mysqlデータベース内に直接INSERTでユーザ追加する方法
 - 新規にユーザを追加する際にのみ使える
2. GRANT構文を利用する方法
 - データベースのユーザに操作権限を与える際に使用する命令
 - 指定したユーザが存在しない場合は新規のユーザを追加する

GRANT構文を利用する方法を学びます

GRANTを使ったユーザ追加

- 書式

```
GRANT 操作の種類 ON データベース名.テーブル名  
TO ユーザ名@ホスト名 IDENTIFIED BY パスワード
```

指定したユーザに操作権限を与えるユーザが存在しない場合はユーザの作成も行う

- 例

```
GRANT SELECT,INSERT,UPDATE,DELETE ON mydb.*  
TO sysuser@localhost IDENTIFIED BY 'secret';
```

secret というパスワードを持つ sysuser というユーザが、localhostから接続した場合、データベース mydb の全てのテーブルに対して SELECT, INSERT, UPDATE, DELETE の操作を行うことを許可する

練習

- データベースユーザの作成

PHPでのMySQL操作

1. mysql関数を使う方法
 - PHPで標準で用意されているMySQL操作関数群
2. PDO(PHP Data Objects)を使う方法
 - PDO: PHP5.1から追加されたデータベース操作クラス

PDOを使う方法を学びます

PDOを使ったMySQL利用手順

1. MySQLサーバへ接続する
2. 各種クエリーを発行する
 - SELECTの場合は結果を取得する
3. 接続を終了する

MySQLサーバへの接続

- 書式

```
変数 = new PDO(データソース名, ユーザ名, パスワード);
```

- データソース名は、利用するデータベースの情報を指定するもの
- 接続に成功するとPDOオブジェクトを返す失敗すると「例外」が発生する

- 例

```
$pdo = new PDO("mysql:host=localhost;dbname=mydb",  
               "sysuser", "secret");
```

文字コードの設定

- 「SET NAMES」 クエリを発行する
- 例

```
$pdo->exec("SET NAMES utf8");
```

クエリの送信（１）

- \$PDOオブジェクト->exec(SQL文);
 - DBからデータを取得しない場合に使用する
- 例

```
$pdo->exec("INSERT INTO members VALUES ('太郎')");
```

クエリの送信（２）

- \$PDOオブジェクト->query(SQL文);
 - DBからデータを取得する場合に使用する

- 例

```
$stmt = $pdo->query("SELECT * FROM members");
```

- 戻り値は、PDOStatement という種類のオブジェクト

結果データを取り出す

- \$pdo->query(...)の戻り値である
PDOStatement オブジェクトに対して：
 - fetch()メソッドを使用する（1件取得）
 - データが無い場合はFALSEを返す

```
$data = $stmt->fetch();
```

- fetchAll()メソッドを使用する（全部取得）
 - データが無い場合は空の配列を返す

```
$data = $stmt->fetchAll();
```

MySQLサーバへの接続の終了

- PDOオブジェクトにnullを代入する

```
$pdo = null;
```

- PHPプログラム終了時に自動的に接続が終了するため、通常は呼び出す必要はない

練習

- member.php
 - List 30-1-1
 - List 30-1-2

データベースに対する攻撃

- SQLインジェクション
 - ユーザからフォームなどを通じて受け取ったデータをSQL文内で利用する場面で、悪意のあるデータを送信することにより、そのサイトが備える通常の認証や権限のチェックを回避したり、管理者用のパスワードを書き換えたりする攻撃


SQLインジェクション対策

- エスケープ処理を行う
 - 入力値に含まれる、SQL文で特殊な文字（シングルクォート等）を無効な形式に変換する
 - PHPでは、PDOのプリペアドステートメントを利用することでエスケープ処理を施すことができる

プリペアドステートメント

- クエリの実行準備を済ませたSQL文
 - 同じクエリを複数回処理する際の効率が上がる
 - 入力値のエスケープ処理も行ってくれる
- 例

```
$stmt = $pdo->prepare("SELECT * FROM members  
WHERE age >= ? AND age <= ?");  
  
$stmt->execute(array($minAge, $maxAge));
```



例外処理

- プログラムの実行中に例外（エラー）が発生した際に行う処理
 - 一連の処理の中で発生するエラー処理を一括で行える

```
try {  
    // 例外が発生する可能性のある処理  
}  
catch (Exception $e) {  
    // 例外が発生した場合の処理  
}
```

例外処理 (続き)

- PDOを利用する際の例外処理の記述例

```
try {  
    $pdo = new PDO(...);  
    ...  
}  
catch (PDOException $e) {  
    echo $e->getMessage();  
    exit;  
}
```

エラーモードの設定

- PDO処理で発生するエラーを全て例外にする設定：

```
$pdo->setAttribute(PDO::ATTR_ERRMODE,  
                    PDO::ERRMODE_EXCEPTION);
```

- PDOのインスタンスを生成した直後に設定する

練習

- member.php
 - List 30-1-3
- addmember.php
 - List 30-2-1
- addmember.php
 - List 30-2-2

実習課題

- 実習課題03 を行う