PHP基礎オンライン講座
4回目

2026/1/27

Kazuma SEKIGUCHI

前回のアジェンダ

- 配列の扱い
- セッションとは何か
- 画像のアップロードと表示

今回のアジェンダ

- ファイルへの保存、読み出し
- データベースの基本
- phpmyadminによるデータベースの操作
- クラスとは何か
- PDOを利用したデータベースの操作
- データベースへのデータの挿入
- SQLインジェクション対策
- データベースからのデータ取り出し、表示
- 条件を付けてデータ取り出し、並び替え

ファイル保存

- 書き込みが可能なフォルダーに対して、PHPからファイルの保存、読み取りが可能
  - WindowsやmacではあまりPermissionを意識しなくて問題無いが、レンタルサーバーなどではパーミッション（書き込みや読み取りの権限）が書き込み可能になっていることを確認すること
- どういう形式のファイルでも保存、読み取りは可能
  - そのままダウンロードさせる、なども可能
  - フォームでアップロードされてきた任意の形式のファイルを保存することも可能

テキストファイルの保存

- テキスト情報を保存する場合
  1. サニタイズ処理を行っておく
  2. 指定したファイルを開く
     - 開き方の指定次第で指定したファイルが無いとエラーになるパターンと自動的にファイルを作成してくれるパターンが存在する
     - fopen()関数で開く。第一引数で開くファイル名、第二引数で開き方を指定可能
     - “a”を指定するとファイルが無ければ新規に作成し、追記モード（そのまま後に書き足していくモード）で開かれる（一番楽）

テキストファイルの保存

- テキスト情報を保存する場合
  3. 書き込む内容を生成する
     - テキストの場合は、CSV書式（カンマ区切りで値を記述していく）がベターと思われる
     - 1つの書き込みに付き、1行を利用する
  4. 実際に書き込むには、fwrite()関数を利用
     - 第一引数にfopen()関数の返り値を指定、第二引数に書き込む内容を指定する

テキストファイルの読み出し

- 読み出す場合
  1. fopen()関数で、”r”を指定し、読み出しモードにする
  2. ループを利用して、!feof($fp)に到達するまでループを利用して中身を取り出す
  3. fgets()関数でファイルから1行取り出すことが可能。これを終わりまで繰り返せば、全て取得することが可能

- file_get_contents(ファイル名)で読み出すことも可能
  - こちらはファイルを一気に全て読み込む
  - 画像ファイルなどのバイナリファイルの読み出しに適している
  - 外部URLを指定することも可能

ファイルの保存

```
//ファイルへの書き込み
  $fp = fopen('data/data.txt','a');//data/data.txtというファイルを追記モード（ファイルが無ければ作成）で開く
  $content = $name.','.$email.','.$gender.','.$comment."¥n";//書き込む内容を作成する。カンマでそれぞれを区切っておく
  if(fwrite($fp,$content)){
    //fwrite()→指定したファイルに指定した内容を書き込む。成功するとtrueが返る
    print('登録しました');
  }
  else{
    //書き込みに失敗した場合はfwrite()でfalseが返る
    print('書き込みに失敗しました');
  }
```

ファイルの読み出し

```
//ファイルの読み出し
 $fp = fopen('data/data.txt','r');//data/data.txtというファイルを読み出しモードで開く
 while(!feof($fp)){//ファイルの最後まで読み込むと終了
  $line = fgets($fp);//1行分だけ読み出す（勝手に次の行を読んでくれる）
  print($line);//読み出した内容を表示する
 }
```

```
//ファイル全体を一気に読み込む
 $file = file_get_contents('data/imge.jpg'); //data/image.jpgというファイルを全部読み出して$fileに格納する
 header('Content-type: image/jpeg');//jpegファイルを出力することをブラウザーに伝える
 print($file);//ファイル内容を出力
```

サーバサイドとクライアントサイド

さまざまなデータを保存しておく

DBサーバ

MySQL、MariaDB、PostgreSQL、SQLite、Oracle、SQL Serverなど

Webサーバ

Apache、IIS、nginxなど

PHP , Python , Ruby
Perl , JavaScript
C , Java

要求

HTMLや画像など

JavaScript

データベース

- 動的に生成するためには、生成するための情報を保存しておく必要がある
  - 一般的には「ファイル」
  - サーバーサイドでもファイルとしてデータを保存しておくことは可能
- ファイルの場合、データが取り出しにくい
  - どこにファイルがあるのか？
  - ファイルの中にどのような形式で記述されているのか？
- データベースという仕組みを利用した方が楽
- MAMPではMySQLが同梱されている

データベースを利用する理由

- ファイルに保存することで、データの保存は可能
  - ただし、任意のデータを取り出すには不向き
    - Ex:誕生日が1995年以降の人だけ取り出す
  - すべてのファイルを捜索して見つけ出す必要がある→時間が掛かりすぎる
- データベースであれば、内部で特殊形式によりデータを保持
  - 検索時に条件を使用してデータを取り出すことが可能
  - 高速にデータ取り出し、更新が可能
  - 多量のデータも規則性に従って保持可能

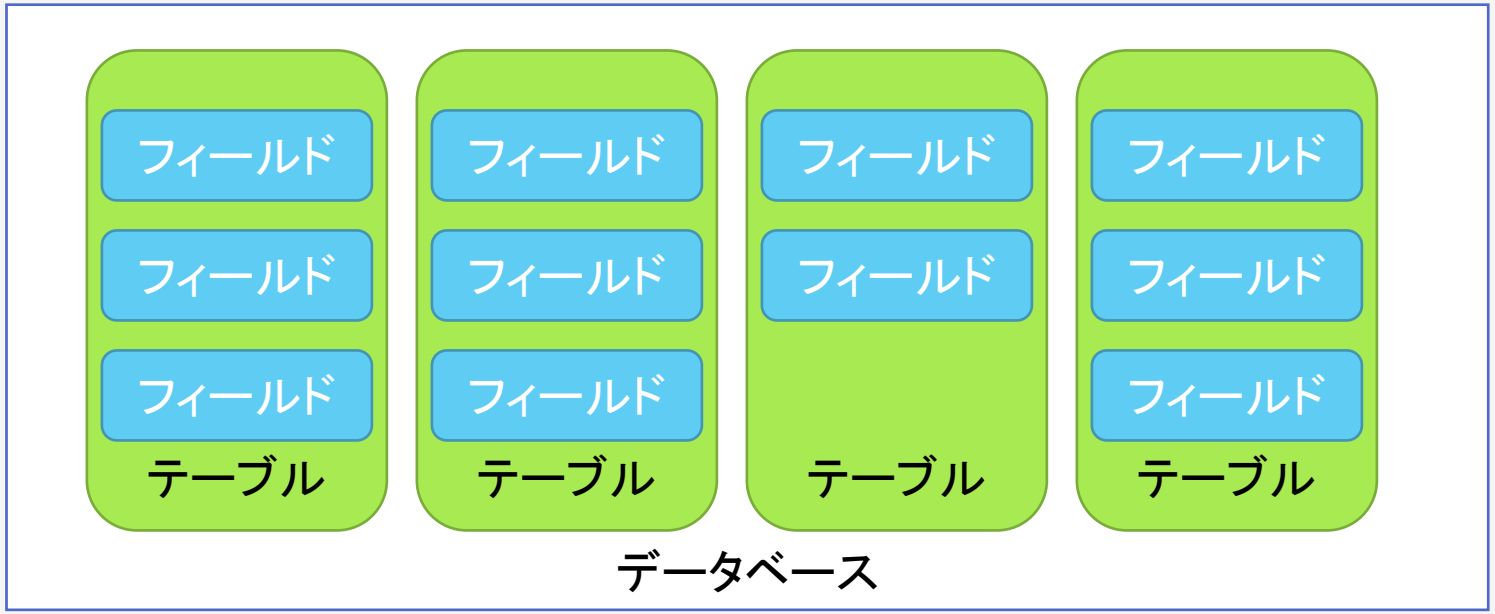
データベース（1）

- データをある規則に基づいて保存し、再利用する仕組み
- 大量のデータを保存、変更、削除、取り出すのに優れる

- 基本的に保存できるのは、数値か文字のデータなど
  - 画像なども保存可能だが、通常しない
  - 画像などはファイルとして保存し、データベースにはファイル名などを格納しておく

データベース（2）

- データベースは2つの意味がある
  - １．データベースソフト
  - ２．データベースソフト内で動くテーブルの集合体
- データベースは複数のテーブルから構成される
- テーブルには、いくつかのフィールドが存在する



データベース（3）

- 各テーブルは表形式になっている
  - Excelみたいなものを想定すればOK
- 1つのデータを1行に収める
- 各フィールドに何を収めるかは別に規定をする（数値？文字？日付？）
  - 空（何もデータが無い）を認めるかどうかも規定する（NULL値）

列がフィールド

1つのデータ →

1	みかん	和歌山	100	2022/11/20
2	りんご	青森	150	2022/11/22
3	桃	山梨	320	2022/08/20
4	いちご	三重	80	2022/05/10

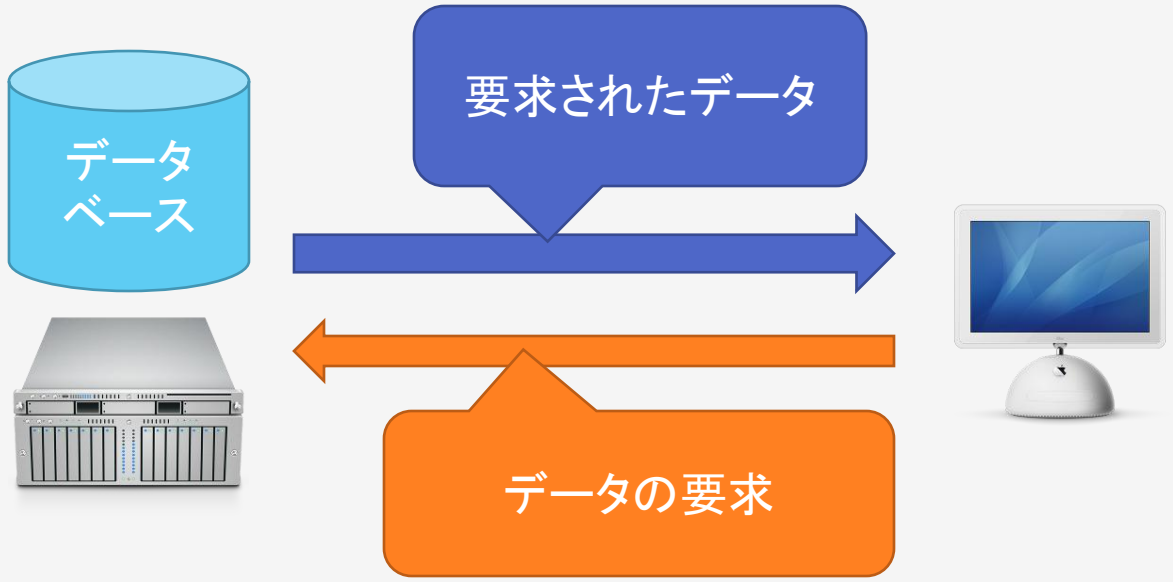
データベース（4）

- 各フィールドには型及び長さがある
  - 文字列、数値、日時など
  - 「長さ」はデータの最大の長さを指定する
    - 郵便番号なら「7」など
  - 入れる予定のデータに合わせて形式および長さを選択する必要がある

- 各データを一意に決めるデータが必須
  - 他のデータと必ず違う値を取るフィールドが必要
  - 名前などでも一意になるとは限らない
    - 「ID」というフィールドを作り、順番に数字を付けていくことが多い（データベースに順番に数字を振る機能がある）

データベース（5）

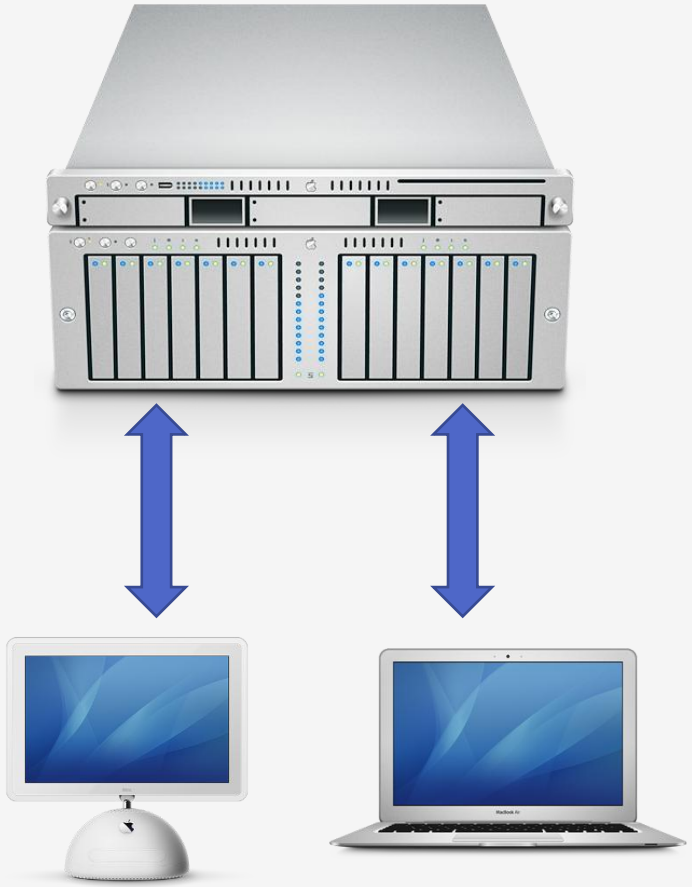
- ほとんどのデータベースがクライアントサーバ方式をとる
  - データはデータベースサーバにあり、必要とするPCからデータベースサーバに接続して取得する



クライアントサーバ方式

- データベースはサーバソフトと
  クライアントソフトから構成されるのが一般的
- メリット
  - 複数台から接続し、データを取得してもらえる
  - データベースサーバの数を減らし、データの
    管理がしやすい

- 中小規模ならば、クライアントとサーバを同じ
  マシンで兼ねることが可能
  - 自分から自分のマシンにあるデータベースサーバ
    に接続する
  - MAMP環境ではクライアントとサーバーが同一マ
    シンということになる



データベースの操作

- データベースの操作は本来CUIを利用してコマンドを利用し操作を行う
  - CUI＝CharacterUserInterface
- マウスなどで操作できないし、一覧性にも乏しい
  - 慣れるとCUIでも操作はできるが、慣れていないうちは難しい
- データベースを画面上で見ながら操作するためのツールが作られている
  - MySQLの場合PHPで作成されたphpmyadminが有名
  - MariaDBでも利用可能

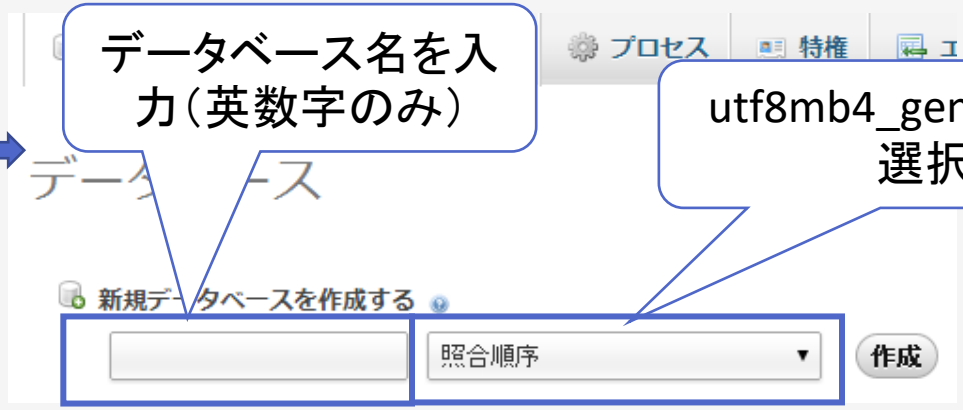
phpmyadmin

現在MySQL内にある
データベースの一覧

データベースをクリックすると
データベース内にある
テーブルの一覧が出てくる

表示をクリックすると
テーブル内に格納されて
いるデータが表示される

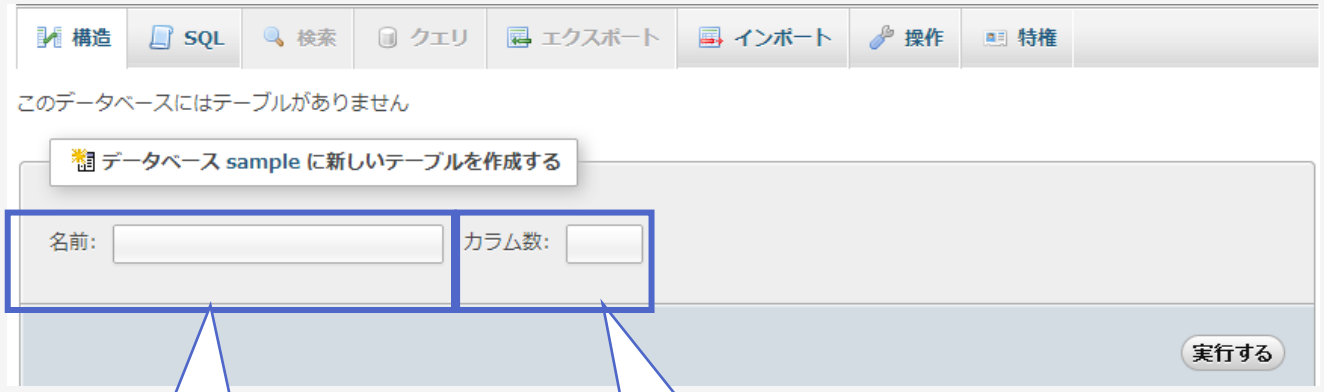
phpmyadminによるデータベース作成



データベース名を入力（英数字のみ）

utf8mb4_general_ciを選択

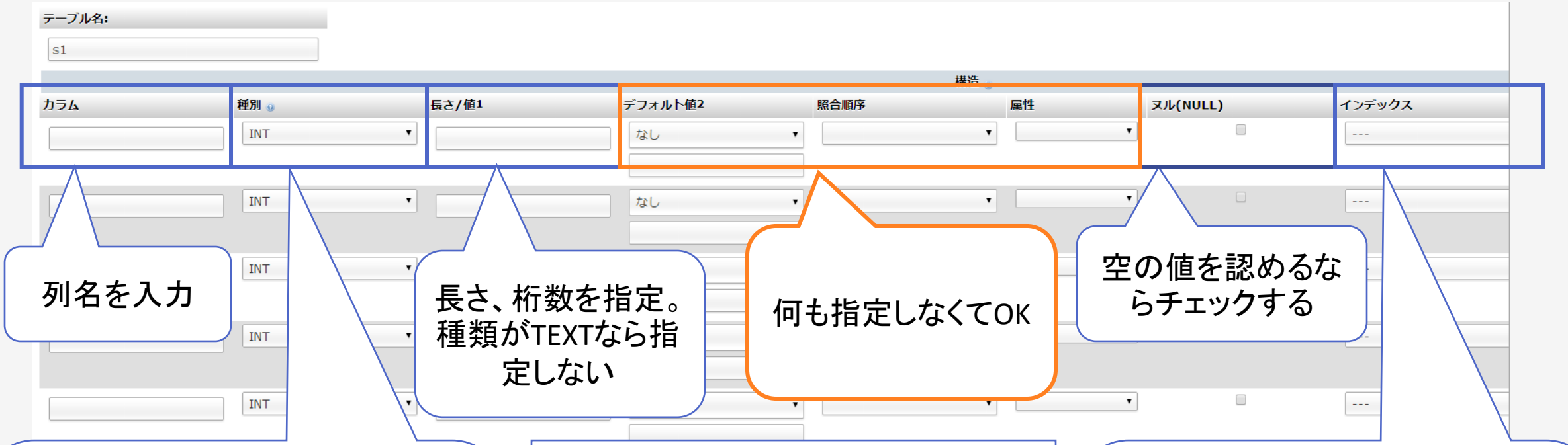
utf8_general_ciと異なり絵文字が格納可能



テーブル名を入力

必要な列数を入力

phpmyadminによるデータベース作成



列名を入力

長さ、桁数を指定。
種類がTEXTなら指定しない

何も指定しなくてOK

空の値を認めるならチェックする

列にどういうデータが入るかを指定
INT：整数数字
VARCHAR：可変長文字列
TEXT：文字列
DATETIME：日付

IDのように一意にするのであれば、「A_I」にチェックを入れておく
（データが入る度に1ずつ足していって一意のデータを作成してくれる）

ほかと被らない一意であれば
PRIMARYかUNIQEを指定
数字などではINDEXを指定
文字列には何も指定しないのが一般的
PRIMARYは必ず1つは必要

SQL（1）

- データをどのように取得するか？
  - データベースを操作する言語がある
  - SQL（エスキューエル）
    - データベースを操作し、データを挿入、取得、更新、削除を行うための言語
    - 標準化されており、どのデータベースでもほとんど利用可能（多少の差異はある）
  - SQL自体は小文字でもOKだが、PHPで記述するときは大文字での記述が多い

- SQL文で文字を扱うときは「’ 」（シングルクォート）で括る
- 条件式で列名を指定するときは「`」（バッククォート）で括る（こともある）

SQL（1）

- SQL自体は標準化されているため、データベースソフトでは基本的にサポートされている
  - 他のデータベースソフトに変えても同じSQLが利用できる
  - 現実的には、サポートしていない構文があったり、書き方が少し違ったりするためある程度考慮が必要
- PHPと組み合わせて使う場合はさほど深い知識は要らないが、SQL文を上手く使えると効率的にデータを加工することが可能

SQL（2）

- データの取得（１）

SELECT「取得するフィールド名」FROM「テーブル名」

全部取得
SELECT * FROM「テーブル名」

条件にあったものだけ取得
SELECT * FROM「テーブル名」WHERE `id` = 1

指定した数だけ取得
SELECT * FROM「テーブル名」LIMIT 10

whereで条件を付けることができる
条件は不等号を使うことが可能

SQL（2）

- データの取得（2）

並び替えて取得

IDが大きいもの順に取得
SELECT * FROM「テーブル名」ORDER BY id DESC

IDが小さいもの順に取得
SELECT * FROM「テーブル名」ORDER BY id ASC

- 複数の条件を満たしたものだけ取得するには「AND」使用
- 条件のいずれかを満たしたものだけ取得するには「OR」を使用

SQL（2）

- データの取得（３）

全部使う

```
SELECT id , name FROM「テーブル名」WHERE
`id` = 1 OR `id` = 10 LIMIT 10 ORDER BY id DESC
```

SQL（3）

- データの更新

UPDATE「テーブル名」SET「フィールド名」=「更新する値」

全部更新
UPDATE「テーブル名」SET `name` = ‘Kazuma’
＊テーブル内の全てのnameが変わることに注意！

条件を指定して更新
UPDATE「テーブル名」SET `name` = ‘Kazuma’ WHERE `id` = 1
＊ほとんどこの書式を使う

SQL（4）

- データの挿入（１）

```
INSERT INTO「テーブル名」VALUES (「挿入する値」)
```

- 挿入する値は「,」で区切ることで複数挿入可能
- テーブルで指定されているフィールドの左端の値から指定する
- 値を入力したくないフィールドは「”」で飛ばす

SQL（4）

- データの挿入（２）

```
INSERT INTO「テーブル名」VALUES
(1,’kazuma’,’19800807’,’kazuma@example.co.jp’,’’)
```

SQL（5）

- データの削除

DELETE FROM「テーブル名」
＊テーブル内のデータが全部消えることに注意！

条件を指定して削除
DELETE FROM「テーブル名」WHERE `id` = 1
＊ほとんどこの書式を使う

ダイナミックページの作成

- あらかじめDBなどに保存してあるデータを取得してきて表示する
  - 表示する際に条件に応じてデータを取り出してくる

- DBに保存したり更新したりする仕組みが必要
  - 挿入、抽出（取り出し）、更新、削除が必要

すべてはDBへの操作

- PHPなどのプログラムはデータをどうやって取り出すか、加工するかということに主眼を置いて作成する
- データを保存する、ということはDBに任せる

- 条件をきちんと精査してデータを取り出す、更新する必要がある
- 取り出す条件を変更する場合は、ブラウザーのリフレッシュ（リロード）が必要

PDOで接続

- 現在のPHPではDBへの接続にはPDOという仕組みを利用する
- PDOを利用することでDB自体のソフトが変更されても運用が可能である、という建前
  - 実際にはSQLが少し違うこともあり、難しい
- PDOは便利な仕組みなところもあるが、取っつきづらいかも
  - PDO自体PHPのクラスとして作成されている

クラス

- オブジェクト指向言語で出てくる概念＝クラス
- クラス＝データと関数の集まり
- ある機能を実現するために関数群の集まりから作成するよりも1つのクラスという設計図に収めた方が使い回しが効くことから非常に良く使われる
  - 実体化することで実際に使用できる（クラス自体は設計図であるため）
  - 実体化＝インスタンス化
- 現在のプログラミング言語はほとんどがクラスを基本にしてプログラムを構築していく

PDOでDBに接続

```
try {
$pdo = new PDO('mysql:host=ホスト名;dbname=DB名;charset=utf8','ユーザー名','パスワード',array(PDO::ATTR_EMULATE_PREPARES => false));
} catch (PDOException $e) {
 exit('データベース接続失敗。'.$e->getMessage());
}
```

- try文は指定された処理を実行してエラーが生じたらcatchに指定された部分を実行する
- newでインスタンスを作成
  - インスタンスを作成し、$pdoに格納することでDBを利用できる

PDOで取り出し

```
$stmt = $pdo->query("SELECT * FROM テーブル名 ORDER BY no ASC");
while($row = $stmt -> fetch(PDO::FETCH_ASSOC)) {
 $title = $row["title"];
 print($title);
}
```

取り出したデータが格納される
取り出すデータが無くなるとfalse
になる

テーブルの列名を指定することで
その値を取り出すことが可能

- PDOのqueryメソッドを利用してSQL文を発行する
- 結果は$stmtに入ってくる
  - これをfetchメソッドを使って中身を取り出す
  - DBの結果がいくつ入っているか分からないため、ループで処理を行う

PDOでデータを挿入

```
$name = 'one‘;
$value = 12;
$sql = "INSERT INTO テーブル名 (name, value) VALUES (?,?)";
$stmt = $pdo -> prepare($sql);
$stmt->execute(array($name,$value));
```

prepareを使ってSQL文を一度格納することでSQLインジェクションを防ぐ

?で指定した順番に格納されていく

- ユーザーから受け取ったデータはそのままDBに入れると危険
  - SQLインジェクション脆弱性
- prepareメソッドを利用し、一時的に別の文字（ここでは「？」）を指定しておく
  - 最後にexecuteメソッドを実行するときに変数を与えることで、問題の無いSQL文にすることが可能

PDOその他

```
$stmt = $pdo -> query("SELECT * FROM テーブル名");
$count = $stmt -> rowCount();
```

- テーブル内にあるデータ数を数える
  - データが無かったら、、という処理をするときに利用

別ファイルにPHPプログラムを記述

- PHPではプログラムの文中で他のプログラムファイルを読み込むことが可能
  - 機能ごとにファイルを分ける
  - 再利用しやすいようにしておく
- includeすると読み込むファイルがそのまま展開される
  - 指定した順番に読み込まれる

include ‘読み込むファイル名’;//読み込めないと警告出すがそのまま継続して処理される
require ‘読み込むファイル名’;//読み込めないとエラーで終了
include_once ‘読み込むファイル名’;//1度だけ読み込む。同じファイルが2度読み込まれたら読み込まずに警告
require_once ‘読み込むファイル名’; //1度だけ読み込む。同じファイルが2度読み込まれたらエラーで終了

定数

- 変数と同じようなものであるが、置換えが不可能な変数
  - 設定値の格納などに利用される（後から変更はできるが、変数ほどは変更しないようなものに使う）
  - define()として引数を2つ与えて記述する
  - 通常定数の値は大文字で記述する

```
define("DBNAME","data");
```

定数の名前

定数の値

- 利用する時は、定数の名前をそのまま記述する

```
$dsn = 'mysql:dbname='.DBNAME.';host='.DBHOST;
```

利用する定数

利用する定数

この後の学習

- セキュリティを気にしつつ、PHPで掲示板を作成してみる
  - データの挿入、読み出し、更新、削除を実装してみる
  - 画像ファイルなどもアップロードできるようにしてみる
  - 正規表現によるデータチェックも忘れずに
- PHPにおけるオブジェクト指向の書き方を学ぶ
  - https://qiita.com/mpyw/items/41230bec5c02142ae691 とかを参照
  - そもそもオブジェクト指向とはなんぞや？という場合は https://qiita.com/tip1t/items/b2f8e39d7cc23ad505f9
- 少しオブジェクト指向になれたらフレームワークに触れてみる
  - 今ならLaravel一択

この後の学習

- PHPの記述方法は比較的少ない
  - がやり方がいくつも存在するため、ある程度記法を覚える必要はある
  - セキュリティが甘くなりやすい点には注意が必要
  - Webサーバーなどの知識も同時に入れていくと、今後の学習がしやすくなる
    - URL構造の理解やmod_rewriteなどによるURL変換など
- 現在のPHPはオブジェクト指向的に記述することが多い
  - OOPが理解できていれば、フレームワークも触りやすい
  - 完全に理解せずにフレームワークを触ってから戻ってくる手もある

ありがとうございました。

良きPHPライフを