# PHP×DB（MySQL）

入門

# アジェンダ & 目標

# アジェンダ&目標

■授業内制作物

シンプルブログ作成（記事登録・表示）

■アジェンダ

・データベースとは？データベースの構造を知る。

・データベースを作成してみる。

・phpMyAdmin を使ってデータベース操作を行う。

（登録・表示・更新・削除）

・PHP からデータベース操作を行う。

・データベースから抽出したデータをブラウザ表示する。

# データベースとは？

# データベースって？？

いきなりですが、みなさんにご質問です。
みなさんが「**海女ぞん**」という EC サイトを作成することになりました。

「海女ぞん」の EC サイトを作る上で、必要なデータベースって、どんなものが考えられるでしょうか？

# データベースって？？

- **会員情報を管理するデータベース**
- **商品情報を管理するデータベース**

大概この２つを思い浮かべられる方が多いのではないでしょうか！？

# データベースって？？ - MySQL -


**■データベース**

データの集まりであり、データが一定のルールに乗っ取って整理整頓されたもの。また、必要に応じてデータを取り出せるもの。
（特定のプログラミング言語に依存するものではありません）

**■ MySQL**

データベースの中でもっとも世界で利用されていると言われるもの。
エクセルのような、**表型**で構成されている。

# データベースって？？ - MySQL -


## ■ SQL

データベース言語（プログラム）の一種。

現在一番使用されているデータベース言語は **SQL**。

## ■ phpMyAdmin

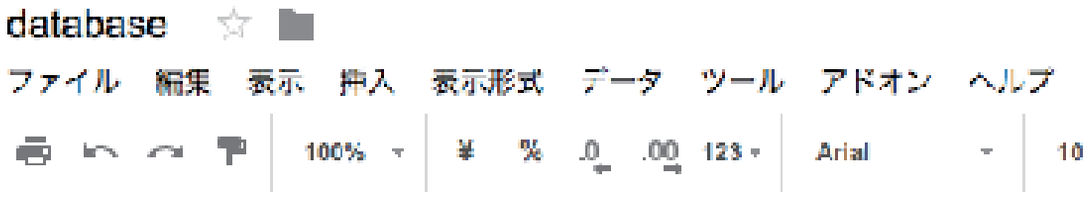
MySQL をわかりやすくブラウザ上で扱えるシステム。

いわゆる「黒い画面」を使わなくても色々出来る。

本来は下記のようにターミナルを使ってログインコマンドを打ちログインするのが通常。

```
kosugatatsuya-no-MacBook-Air:nodejs TatsuyaKosuge$ cd
kosugatatsuya-no-MacBook-Air:~ TatsuyaKosuge$ cd desktop
kosugatatsuya-no-MacBook-Air:desktop TatsuyaKosuge$ mysql -u root -D localhost -
p
```

# データベースって？？




エクセル

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | id | className | studentNumber | limitedNumber | tableNumber |
| 2 | 1 | Dev9期 | 50 | 50 | 9 |
| 3 | 2 | Lab4期 | 28 | 28 | 6 |
| 4 | | | | | |

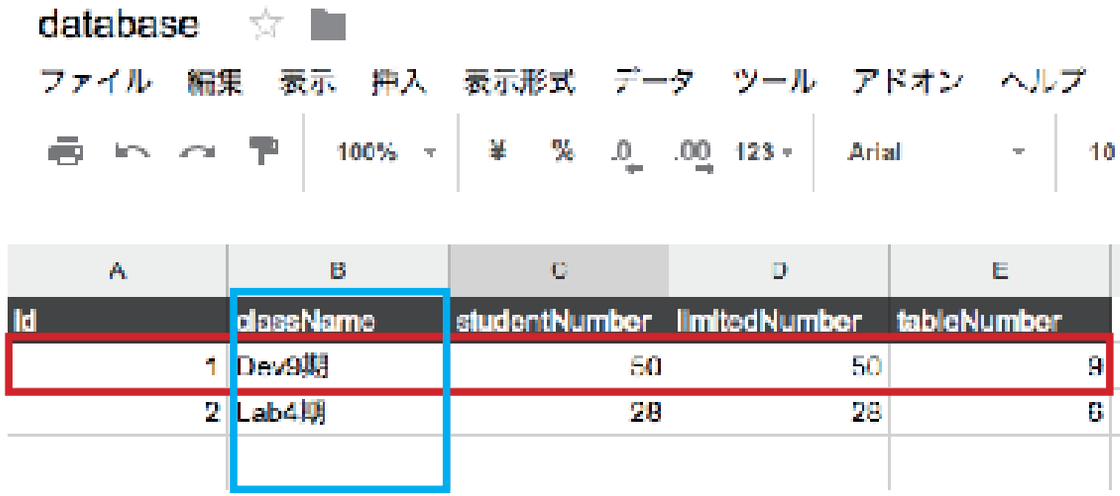
| ←T→ | id | classname | studentnumber | limitednumber | tablenumber |
|---|---|---|---|---|---|
| ☐ ✎ ✕ | 1 | Dev9期 | 50 | 50 | 9 |
| ☐ ✎ ✕ | 2 | Lab4期 | 28 | 28 | 6 |

**MySQL**（phpMyAdmin で表示）

# データベースの構造

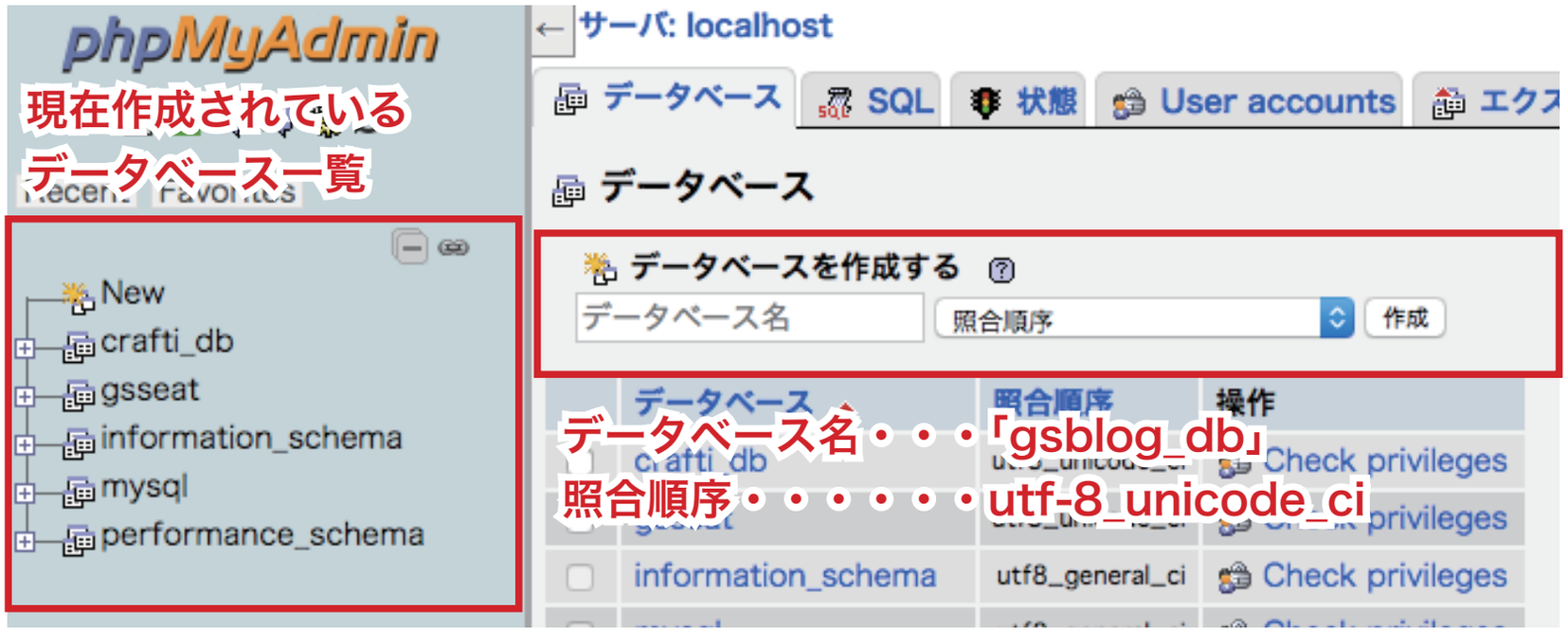
データベースの構造は、エクセルと似ています。

・**データベース**・・・エクセルでいう**ファイル（◎◎ .xls）**に相当。
・**テーブル（表）**・・・エクセルでいう **Sheet（タブのところ）**に相当。
・**レコード（行）**・・・エクセルでいう**横の概念（下記参照）**。
・**フィールド（列）**・・・エクセルでいう**縦の概念（下記参照）**。



# データベースを作ってみる！

# データベースを作ってみる




まずは上記のようにして、データベースを作成してみましょう！

**■手順（XAMPP の場合）**

・http://localhost/xampp/ にアクセス
・phpMyAdmin ロゴ→データベースタブの順にクリック
・上記画面になるので、指示の通りに設定し、作成ボタンを押す！

# ちなみに・・・・照合順序って？？


例えば google で、「PHPMYADMIN」と打っても「phpmyadmin」と打っても同じ検索結果が表示されます。

これは、Google 検索が大文字と小文字を区別していないことが理由なのですが、このように目的によってデータベースの照合順序の設定は変わってくることになります。

探したい内容（文字）とデータベースの内容を比較した時に

**・文字コードや言語は同じか？**
**・大文字、小文字を区別するか？全角、半角を区別するか？**

のルールを決めるためのものが照合順序だと思ってください。

# ちなみに・・・・照合順序って？？

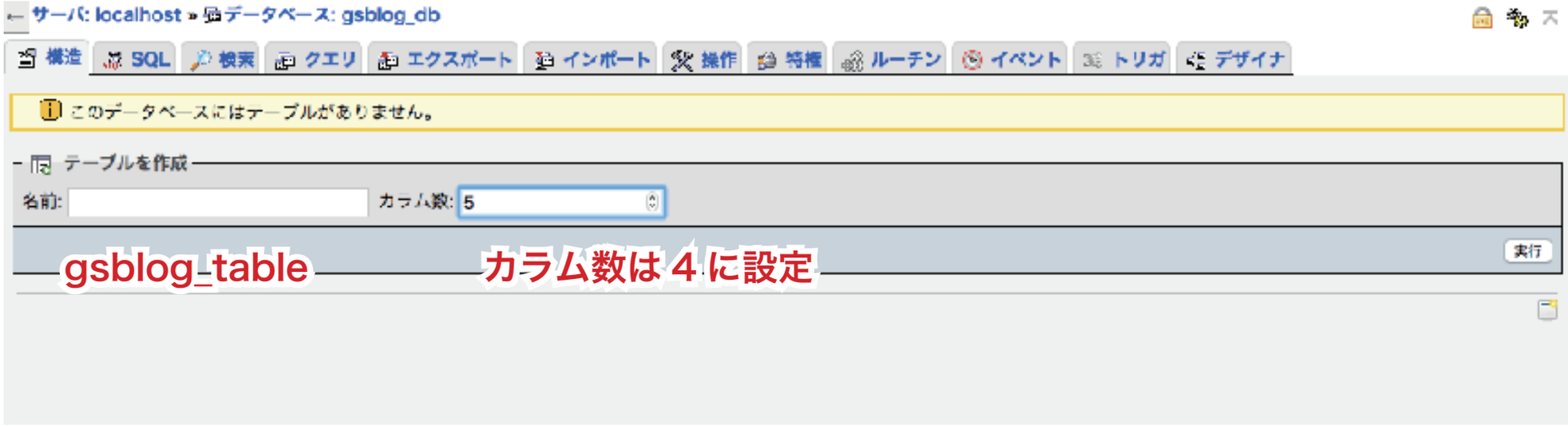

## utf8_unicode_ci

**文字コード**　　**言語名**　　**比較法**

文字コード・・・文字コードが入る。
言語名・・・・・そのままですが言語名が入る。unicode はマルチリンガル。
比較法・・・・・前ページにもある「大文字、小文字を区別するか？全角、半角を区別するか？」の部分を決める。ちなみに ci は「大文字と小文字を区別しない」という意味。

**utf8_general_ci** はアルファベットの大文字小文字は区別しない。他は全て区別。
**utf8_unicode_ci** は大文字小文字 / 全角半角を区別しない。

# テーブルの作成




作成すると上記のような画面になると思います。ここでテーブル名とカラム数を決定し、実行ボタンを押します。

※カラム数・・・フィールド（列）の数のことです。

# テーブルのフィールド名の設定


■長さ / 値
int・・・桁数
varchar・・・文字数

■インデックス
id は自動的に連番で生成させるために A_I をチェック、インデックスは PRIMARY 設定。

■データ型
int ・・・・・・・数値
varchar・・・・最大サイズ 65535 文字（可変）の文字
text ・・・・・最大サイズ 65535 文字（固定）の文字
date（datetime）・・・・・日付型 + 時刻型のデータ
※それ以外は、さらに細かいデータ型のことを指していますが今回は割愛します。

# テーブルのフィールド名の設定


id・・・int(12) **PRIMALY KEY：AUTO INCREMENT**（A_I にチェック）
title・・・varchar（64）
detail・・・varchar（500）
posttime・・・datetime

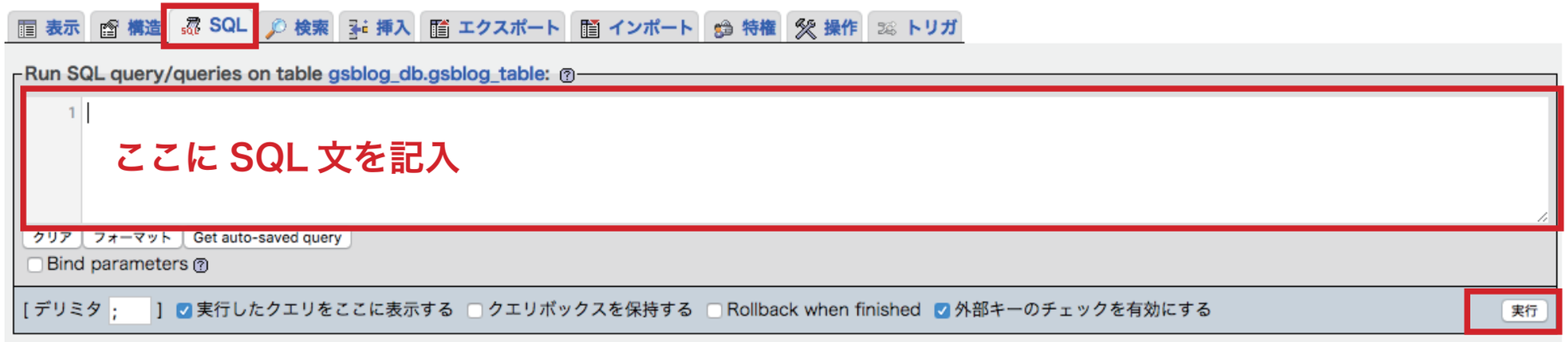
# データベース操作

# データベースの操作 -INSERT-

phpMyAdmin を使って、SQL 言語を書いてみましょう！
文字列はシングルクオーテーションで囲むようにしましょう！
まずはデータ登録をしてみましょう。

**■構文**
INSERT INTO テーブル名 ( カラム１ , カラム２,...) VALUES( 値１ , 値２,...)

※例
INSERT INTO gsblog_table (id, title, detail, time) VALUES (NULL,' はじめまして ',' はじめまして、初投稿です。',sysdate())



# sysdate() って？？

**sysdate()** を実行すると、データベース操作そのものを実行した際の時間を取得することができます。
（一応、現在の日時、タイムスタンプを取得できるものとして紹介されているケースが多い）

他に類似したものとして、**now()** や **CURTIME()** というものがあり、こういったものは MySQL 関数と呼ばれ、SQL 文の中で実行できる関数です。

# データベースの操作 -SELECT-

phpMyAdmin を使って、SQL 言語を書いてみましょう！
文字列はシングルクオーテーションで囲むようにしましょう！
次に、検索してデータ表示を行ってみましょう。

**■構文**
SELECT 表示するカラム FROM テーブル名 WHERE name = ' 内容 '

※例
**SELECT * FROM gsblog_table WHERE** title = ' はじめまして '

※ FROM の後にテーブル名、WHERE をかいてその後に具体的な抽出条件を記載すると、より詳細にデータを取得することが出来ます。
SELECT の次の「*」は、全カラムという意味です。

# データベースの操作 -SELECT-

**■構文**

SELECT * FROM gsblog_table・・・・全カラム指定

SELECT name FROM gsblog_table・・・・単体指定。使用するカラムのデータのみ取得

SELECT name,email FROM gsblog_table・・・・複数指定。上記の複数版

SELECT * FROM gsblog_table WHERE id = 1・・・・特定の条件に合致するデータ取得

**※ WHERE の使い方例**

SELECT name,age FROM gsuser_table WHERE age >= 30

→ age（年齢）30 歳以上のユーザーデータを取得する、といった場合の使い方。

# データベースの操作 -SELECT-

**■ AND, OR で検索条件を複数指定する**

SELECT * FROM テーブル名 WHERE id >= 1 AND id<=3

**■ あいまい検索をする**

SELECT * FROM テーブル名 WHERE indate LIKE '2015-06%'

SELECT * FROM テーブル名 WHERE email LIKE '%@gmail.com'

**■表示をソートする**

SELECT * FROM テーブル名 ORDER BY email, name DESC

→ DESC は降順で SORT、ASC は昇順で SORT

**■表示件数を制限する**

SELECT * FROM テーブル名 LIMIT 5

SELECT * FROM テーブル名 LIMIT 3, 5

# データベースの操作 -UPDATE-

phpMyAdmin を使って、SQL 言語を書いてみましょう！
文字列はシングルクオーテーションで囲むようにしましょう！
次に、更新処理を実行してみましょう。

**■構文**

```
UPDATE gsblog_table SET name= ' 小菅 ' WHERE id = 1;
```

※ WHERE を使用して特定のデータを更新します。これを忘れると全てのデータが
更新されるので知っておきましょう。
上記の例の場合、WHERE の部分がないと、デテーブルの中に入っている全てのデータの
name が小菅になってしまいます。

# データベースの操作 -DELETE-

phpMyAdmin を使って、SQL 言語を書いてみましょう！
文字列はシングルクオーテーションで囲むようにしましょう！
次に、データの削除を行ってみましょう。

**■構文**
DELETE FROM テーブル名 WHERE 削除するデータの条件

※例
**DELETE FROM gsblog_table WHERE** id=1

→id=1 のデータを削除する事例です。一度 DELETE するとデータは二度と復旧できないので、
注意しましょう。ちなみに WHERE がなければテーブル内の全データが削除されます。

# PHP からデータベース操作

# PHP でデータベース操作


先ほどは phpMyAdmin を直接開いて、SQL を直接入力して SQL 言語を実行することを行いました。

でも、実際に開発をするときには、データベース操作をするたびにいちいち phpMyAdmin を開くわけにもいかないですよね。

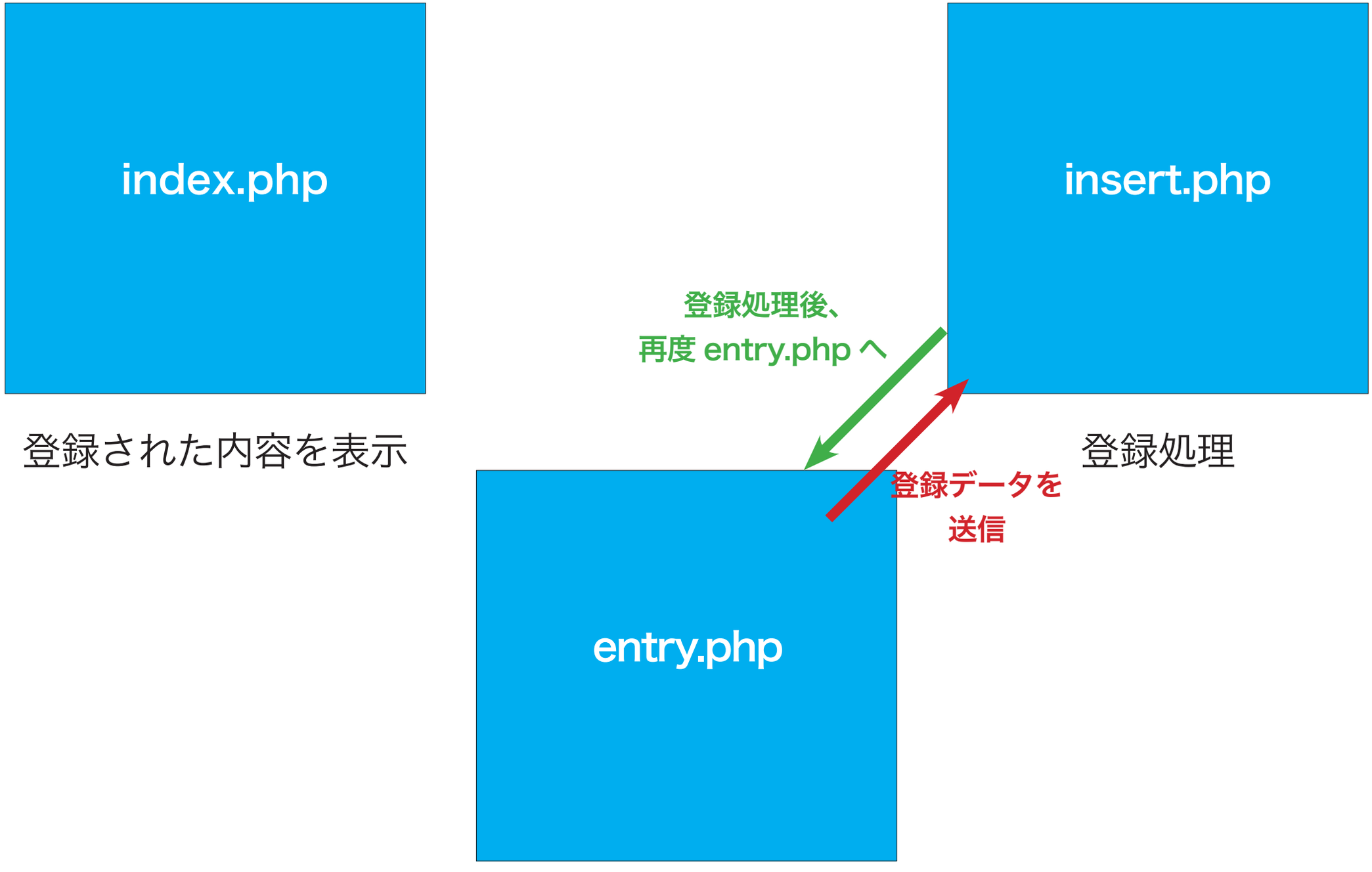
ここからは、**PHP プログラムからデータベース操作をして、SQL を叩き、データをとってきて内容をブラウザに表示する**、ということをやっていきたいと思います！

# PHP でデータベース操作するための「PDO」


PDO は **PHP Data Objects** の略で、PDO を使うことで **MySQL・SQLite・PostgreSQL** などのデータベースシステムを利用する場合でも、同じ関数で使うことができます。
（ PDO は PHP5.1 以降に標準で装備されました。）

先ほど手動でデータベース操作を行った時に必要だったのは
**１、データベースへの接続（アドレス &ID&PW が必要）**
→ phpMyAdmin を開く。
**２、データ操作をするためのテーブル名**
**３、実際に実行したい SQL の内容（SQL 文）**

の３つでしたよね！
このことを頭に置いておきながら、PDO でのデータベース操作を学んでいきましょう！




登録された内容を表示

登録処理

新規データ登録フォーム

# 0、entry.php の内容を確認

まずは、**entry.php** の中身を確認しましょう。
新しい記事を登録するための、フォームが構成されているかと思います。

フォームのデータ送信先が insert.php になっているかを確認し、それぞれの form のパーツの name 属性がきちんと振られているかも確認しましょう。

# 1、PDO でデータベースに接続する


**new PDO（データベース情報 , ユーザー名 , パスワード）**

**※例（この時点でデータベース接続開始！）**

```
$pdo = new PDO("mysql:host=localhost;dbname=gsblog_db;charset=utf8",'root','root');
```

これにより PDO オブジェクトというものが生成され、$pdo 変数に格納されます。

これで、手動でデータベース操作を行ったときでいう、
**「1、データベースへの接続（アドレス &ID&PW が必要。phpMyAdmin を開く）」**
状態が完了したことになります。

# 2、prepare メソッド ( 関数)


**$stmt = $pdo->prepare(' 実行したい SQL 文 ')**

※ SQL 例文

**INSERT INTO gsblog_table** (id, title, detail, time) **VALUES**
(NULL, ' はじめまして ',' はじめまして、初投稿です。',sysdate())

PDO オブジェクトを利用することで、**prepare メソッド（関数）**というものが利用できるようになります。
この関数の引数に実行したい SQL 文を入れることで、
**PDOStatement オブジェクト**を取得することができます。

これで、手動でデータベース操作を行ったときでいう、
**「2、データ操作をするためのテーブル名」**
**「3、実際に実行したい SQL の内容（SQL 文)」**
の用意ができた状態になります。

# 注意：SQL インジェクション


ここで注意をしなければならないポイントとして、SQL 文に使用したい値を直接含めてしまうと、SQL インジェクションに引っかかり、データベース内全ての情報が盗まれる・内容が書き換えられるなどが行われてしまう可能性があります。

**※例**

$age = '1;DELETE FROM gsuser_table';

SELECT * FROM gsuser_table WHERE age >= $age

→ SELECT * FROM gsuser_table WHERE age >= 1;DELETE FROM gsuser_table';

→テーブル内のデータが消えてしまう！！

# 3、bindValue() で実際の値を SQL 文にセット


■ SQL文修正

INSERT INTO gsblog_table (id, title, detail, time) VALUES
(NULL, :name, :detail ,sysdate())

■ bindValue() でパラメーター

**$stmt->bindValue(':title', ' はじめまして ', PDO:: PARAM_STR);**

**$stmt->bindValue(':detail', ' はじめまして初投稿です。', PDO::**
**PARAM_STR);**

bindValue() や bindParam() で指定することによって、値は適切にエスケープされるので、先ほどのような危険性を回避できます。

# 3、bindValue() で実際の値を SQL 文にセット

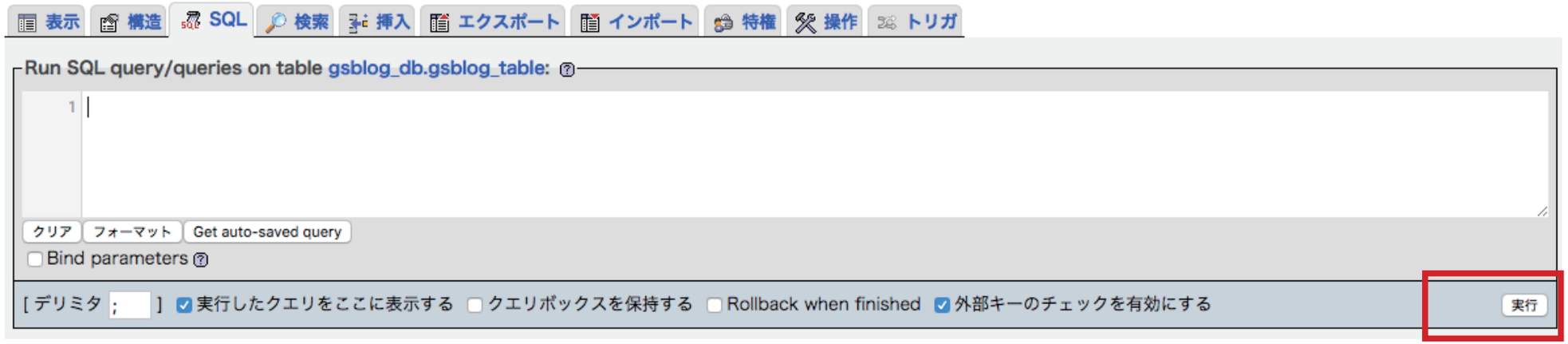

ここまでで改めて、手動でデータベース操作を行ったときでいう、

**「2、データ操作をするためのテーブル名」**

**「3、実際に実行したい SQL の内容（SQL 文）」**

が用意できた状態になります。

# 4、execute() でセットした SQL 文を実行




## $flag = $stmt->execute();

最後は、**execute()** を使って、SQL 文を実行します。
手動でデータベース操作をした時でいうと、SQL タブの実行ボタン（上記赤枠）を押したのと同じ状態を指します。

# ここまでのまとめ

```
$title = htmlspecialchars( $_POST["title"], ENT_QUOTES);
$detail = htmlspecialchars( $_POST["detail"], ENT_QUOTES);
try {
  $pdo = new PDO("mysql:host=localhost;dbname=gsblog_db;charset=utf8",'root','root');
} catch (PDOException $e) {
  exit( 'DbConnectError:' . $e->getMessage() );
}

$sql = INSERT INTO gsblog_table (id, title, detail, time) VALUES (NULL, :name,
:detail ,sysdate());
$stmt = $pdo->prepare($sql);
$stmt->bindValue(':title', ' はじめまして ', PDO:: PARAM_STR);
$stmt->bindValue(':detail', ' はじめまして初投稿です。', PDO:: PARAM_STR);
$flag = $stmt->execute();
```

# データ抽出 & 表示

# 登録したデータを表示してみよう


先ほどまでの工程で、フォームに入力したデータを登録するというところまではできました。
次に、登録したデータをページに表示する、というものをやってみましょう。

データベースに入っているデータを抽出する際は PDOStatement オブジェクトに対して fetch( ) あるいは fetchAll( ) 関数を実行することで、抽出が可能になります。

# 登録したデータを表示してみよう -fetch()-


```
while($result = $stmt->fetch(PDO::FETCH_ASSOC)){

    /* データ表示のためのプログラム内容をここに書く */

}
```

**fetch()** は、実行結果から１行データ取り出して内容を出力していきます。

while 文と合わせることで、
**「１行データ抽出して表示→まだデータが残っていたら次の１行のデータ抽出して表示」**
この繰り返しを実施することができます。

# 登録したデータを表示してみよう -fetchAll()-


```
foreach($result = $stmt->fetchAll(PDO::FETCH_ASSOC)){

    /* データ表示のためのプログラム内容をここに書く */

}
```

**fetchAll()** は、実行結果からデータを全てまるごと配列として抽出する動きをします。

foreach 文と合わせることで、

**「まるごとデータ抽出して 1 つずつデータ表示」**

を実施することができます。

# データ表示部分のまとめ例

```
//DB 接続部分は割愛

$sql = 'SELECT * FROM gsblog_table ORDER BY time DESC LIMIT 3';

$stmt = $pdo->prepare($sql);
$flag = $stmt->execute();

if($flag==false){
    $error = $stmt->errorInfo();
    exit("ErrorQuery:".$error[2]);
}else{
    while($result = $stmt->fetch(PDO::FETCH_ASSOC)){
            /* ここに表示部分。HTML と併記する時は書き方に注意 */
            echo $result["title"];
    }
}
```

# 課題

# 次回までの課題


**本をブックマークする DB 及びアプリを作りましょう！**
**もちろんカスタマイズや先取り大歓迎！**

DB 名 : **gs_db**
Table 名 : **gs_an_table**

項目名 :
1．ユニーク値（int 12 , PRIMARY, AutoIncrement）
2．書籍名 (varChar 64)
3．書籍 URL (text)
4．書籍コメント (text)
5．登録日時 (datetime)

**※ LAB4 期の例（ログイン機能などは後から追加で基本は同じ）**
https://www.linoup.jp/
http://kanomata.sakura.ne.jp/20171205_kadai/login.php

# PHP のデバッグ方法


**１、var_dump( デバックしたい内容 )**
地道に出力してデバッグするのに便利。

**２、Visual Studio Code に PHP デバッグ機能を追加**
**https://arrown-blog.com/vscode-phpdebugger/**

※かなり根気が必要で、一発でうまくいかないこともたくさんあるので、取り組みたい方は相当時間のある時にやりましょう！