```php
<?php
//宣告時區為 亞洲/台北
date_default_timezone_set("Asia/Taipei");
//使用 session 來記錄用戶的資訊前，要先用 session_start() 告訴系統準備開始使用 session
session_start();
//session_start：啟用一個新的或開啟正在使用中的session。 session_destroy：清除正在使用中的session。
//session_name：取得正在使用中的名稱或將名稱更新為新的名稱。

//宣告 DB類別，定義其屬性(變數)與方法(函數)，用以連接資料庫，讀取特定資料表($table)內的資料
class DB
{
    // 設定 Data Source Name $dsn 連線參數，protected 宣告限定自己和子類別可以使用
    protected $dsn = "mysql:host=localhost;charset=utf8;dbname=db10";

    // 資料庫連接物件 - PDO：PHP Data Objects 在PHP裡連接資料庫的使用介面
    protected $pdo;     //宣告 class內部資料庫連接物件變數
    protected $table;   //宣告 class內部資料表屬性變數
    // 建立 類別的建構子方法(函數)：建構子是一個類別裡用於建立物件的特殊子程式。
    // 建構子能初始化一個新建的物件，並時常會接受參數用以設定實例(物件)變數
    public function __construct($table)
    {
        //將外部傳入的參數 $table 設定給 class內部屬性變數 $this->table
        $this->table = $table;
        //建立 PDO物件(連接資料庫的使用介面) 設定給 class內部物件變數 $this->pdo
        $this->pdo = new PDO($this->dsn, 'root', '');
    }

    // 建立 讀取資料表所有資料內容的方法並回傳執行得到之所有結果
    function all($where = '', $other = '')
    {
        //初始 SQL指令內容為： select * from TABLE
        $sql = "select * from `$this->table` ";
        //呼叫 sql_all方法，串接 $sql, $where, $other 成為新的 SQL指令
        $sql = $this->sql_all($sql, $where, $other);
        //從執行結果集回傳執行得到之所有結果，fetchAll：從結果集傳回所有列作為陣列或物件的陣列，
        //依預設，PDO 會以陣列形式傳回每一列，依直欄名稱及列中的 0 索引直欄位置進行索引。
        //FETCH_ASSOC：依照結果集中傳回的直欄名稱，傳回已編製索引的陣列。
        return  $this->pdo->query($sql)->fetchAll(PDO::FETCH_ASSOC);
    }

    //建立 查詢資料表符合篩選條件之資料筆數的方法
    function count($where = '', $other = '')
    {
        //初始 SQL指令內容為： select count(*) from TABLE
        $sql = "select count(*) from `$this->table` ";
        //呼叫 sql_all方法，串接 $sql, $where, $other 成為新的 SQL指令
        $sql = $this->sql_all($sql, $where, $other);
        // 從查詢結果集中的下一行回傳單獨的一列
        return  $this->pdo->query($sql)->fetchColumn();
```

```
    }
    //建立 聚合函數計算資料表特定欄位特定計算結果的方法
    private function math($math, $col, $array = '', $other = '')
    {
        //聚合函數初始 SQL指令內容為： select $math(`$col`) from TABLE
        $sql = "select $math(`$col`)  from `$this->table` ";
        //呼叫 sql_all方法，串接 $sql, $where, $other 成為新的 SQL指令
        $sql = $this->sql_all($sql, $array, $other);
        // 從計算結果集中的下一行回傳单獨的一列
        return $this->pdo->query($sql)->fetchColumn();
    }
    //建立 加總函數計算資料表特定欄位特定篩選條件加總結果的方法
    function sum($col = '', $where = '', $other = '')
    {
        //回傳 加總函數計算資料表特定欄位特定篩選條件加總結果
        return  $this->math('sum', $col, $where, $other);
    }
    //建立 最大值函數計算資料表特定欄位特定篩選條件取最大值結果的方法
    function max($col, $where = '', $other = '')
    {
        //回傳 最大值函數計算資料表特定欄位特定篩選條件取最大值結果
        return  $this->math('max', $col, $where, $other);
    }
    //建立 最小值函數計算資料表特定欄位特定篩選條件取最小值結果的方法
    function min($col, $where = '', $other = '')
    {
        //回傳 最小值函數計算資料表特定欄位特定篩選條件取最小值結果
        return  $this->math('min', $col, $where, $other);
    }

    //建立 查詢資料表符合特定 id條件之資料的方法
    function find($id)
    {
        //初始 SQL指令內容為： select * from TABLE
        $sql = "select * from `$this->table` ";

        //如果傳入的 id是陣列，則呼叫 a2s陣列元素轉字串的方法
        if (is_array($id)) {
            //呼叫 a2s方法，將 id陣列轉換為包含列名和對應值的字串之 $tmp陣列
            $tmp = $this->a2s($id);
            //使用串接運算，串接 $sql, $where, $other 成為新的 SQL指令
            $sql .= " where " . join(" && ", $tmp);
            //如果傳入的 id是數值，則直接串接 $sql, $where, $other 成為新的 SQL
指令
        } else if (is_numeric($id)) {
            //使用串接運算，串接 $sql, $where及 $id 成為新的 SQL指令
            $sql .= " where `id`='$id'";
        } else {
            //如果傳入的 id不是陣列，也不是數值，則顯示錯誤訊息
            echo "錯誤:參數的資料型態比須是數字或陣列";
        }
        //echo 'find=>'.$sql;   //若執行結果錯誤，除錯時顯示 SQL指令

        //從查詢結果集中的下一行回傳单獨的一列，存入 $row變數(物件)
        $row = $this->pdo->query($sql)->fetch(PDO::FETCH_ASSOC);
```

```
        //回傳 $row變數(物件)
        return $row;
    }

    //建立 將陣列元素儲存到 DB的資料表中的方法
    function save($array)   // 傳入 $array陣列
    {
        //如果 $array陣列中有'id'的欄位，則更新資料表中的該筆資料
        if (isset($array['id'])) {
            //初始 SQL指令內容為： update TABLE set
            $sql = "update `$this->table` set ";
            //若 $array陣列不是空的(有值)
            if (!empty($array)) {
                //則將陣列元素轉換成字串(array to string)->"`$col`='$value'"
                $tmp = $this->a2s($array);
            } else {
                echo "錯誤:缺少要編輯的欄位陣列";
            }
            //串接 SQL指令內容成為： update TABLE set
`$col`='$value',`$col`='$value'...
            $sql .= join(",", $tmp);
            //串接 SQL指令內容成為： update TABLE set
`$col`='$value',`$col`='$value'...
            // where `id`='{$array['id']}'，以符合 UPDATE 的 SQL 語法
            $sql .= " where `id`='{$array['id']}'";
            //如果 $array陣列中沒有'id'的欄位，則新增資料到資料表中
        } else {
            //初始 SQL指令內容為： insert into TABLE
            $sql = "insert into `$this->table` ";
            //串接 $cols = (`col1`,`col2`,`col3`,...)
            $cols = "(`" . join("`,`", array_keys($array)) . "`)";
            //串接 $vals = ('value1','value2','value3',...)
            $vals = "('" . join("','", $array) . "')";
            //串接 SQL指令內容成為： INSERT INTO
`TABLE`(`col1`,`col2`,`col3`,...)
            // VALUES('value1','value2','value3',...);以符合 INSERT INTO 的 SQL
語法
            $sql = $sql . $cols . " values " . $vals;
        }

        //回傳 exec($sql) 執行結果
        return $this->pdo->exec($sql);
    }

    //建立 依資料表主鍵或特定條件篩選，到 DB的資料表中刪除特定資料的方法
    function del($id)
    {
        //初始 SQL指令內容為： delete from TABLE where
        $sql = "delete from `$this->table` where ";
        //如果 $array陣列中有'id'的欄位，則刪除多筆資料
        if (is_array($id)) {
            //則將陣列元素轉換成字串(array to string)->"`$col`='$value'"
            $tmp = $this->a2s($id);
            //串接 SQL指令內容成為： delete from TABLE where `$col`='$value' &&
`$col`='$value'...
```

```
            //以符合 delete 的 SQL 語法
            $sql .= join(" && ", $tmp);
            //如果傳入的 id是數值，則刪除指定 id的該筆資料
        } else if (is_numeric($id)) {
            //串接 SQL指令內容成為： delete from TABLE where `id`='$id'
            $sql .= " `id`='$id'";
        } else {
            //參數的資料若非陣列或數值，則顯示錯誤訊息
            echo "錯誤:參數的資料型態比須是數字或陣列";
        }
        //echo $sql;    //若執行結果錯誤，除錯時顯示 SQL指令
        //回傳 exec($sql) 執行結果
        return $this->pdo->exec($sql);
    }

    /**
     * 可輸入各式SQL語法字串並直接執行
     */
    function q($sql)
    {
        //從執行結果集回傳執行得到之所有結果，fetchAll：從結果集傳回所有列作為陣列或物件的陣列，
        //依預設，PDO 會以陣列形式傳回每一列，依直欄名稱及列中的 0 索引直欄位置進行索引。
        //FETCH_ASSOC：依照結果集中傳回的直欄名稱，傳回已編製索引的陣列。
        return $this->pdo->query($sql)->fetchAll(PDO::FETCH_ASSOC);
    }

    //建立 將陣列元素轉換成字串(array to string)的方法
    private function a2s($array)    // 傳入 $array陣列
    {
        // 遍歷$array陣列，將每個元素轉換為一個包含列名和對應值的字串
        foreach ($array as $col => $value) {
            // 將"列名=對應值"的字串放入 $tmp陣列中
            $tmp[] = "`$col`='$value'";
        }
        // 回傳包含"列名=對應值"的字串之 $tmp陣列
        return $tmp;
    }
    // 建立 串接 SQL指令並回傳完整 SQL指令結果的方法
    private function sql_all($sql, $array, $other)
    {
        //如果資料表屬性有被定義且不為空值，則執行後續動作
        if (isset($this->table) && !empty($this->table)) {
            //如果傳入的 $array參數為陣列，則執行後續動作
            if (is_array($array)) {
                //如果傳入的 $array參數不為空值，則串接sql指令
                if (!empty($array)) {
                    //呼叫 a2s方法，將陣列元素轉換成字串(array to string)
                    $tmp = $this->a2s($array);
                    //串接 SQL指令與 where 條件式
                    $sql .= " where " . join(" && ", $tmp);
                }
            } else {
                //如果傳入的 $array參數不是陣列，則串接 SQL指令與 $array參數
```

```
                $sql .= " $array";
            }
            //串接 SQL指令與 $other 參數
            $sql .= $other;
            // echo 'all=>'.$sql;   //若執行結果錯誤，除錯時顯示 SQL指令
            // $rows = $this->pdo->query($sql)->fetchColumn();

            // 回傳傳完整 SQL指令結果
            return $sql;
        } else {
            echo "錯誤:沒有指定的資料表名稱";
        }
    }
}
//顯示傳入的 $array陣列參數之內容
function dd($array)
{
    echo "<pre>";
    print_r($array);    // 顯示 $array陣列內容
    echo "</pre>";
}
//重新導向程式執行節點
function to($url)
{
    header("location:$url");    //重新導向程式到 $url的連結位置
}

//產生不同資料表($table)的連線物件
$Title = new DB('titles');
$Total = new DB('total');
$Bottom = new DB('bottom');
$Ad = new DB('ad');
$Mvim = new DB('mvim');
$Image = new DB('image');
$News = new DB('news');
$Admin = new DB('admin');
$Menu = new DB('menu');

//第一版 未排除['do']值非資料表變數
// if (isset($_GET['do'])) {
//     $DB = ${ucfirst($_GET['do'])};   //利用首字大寫函數轉換 do值為 資料表物件
變數
// } else {
//     $DB = $Title;
// }

//第二版 排除['do']值非資料表變數，判斷 do值是合裡值後才轉換為 資料表物件變數
// $tables=array_keys(get_defined_vars());
// /* dd($tables); */
// if(isset($_GET['do'])){
//     $key=ucfirst($_GET['do']);
//     if(in_array($key,$tables)){
//         $DB=$$key;
//     }
// }else{
```

```
//      $DB=$Title;
// }

//第三版 排除['do']值非資料表變數，若首字大寫後的 do值有定義，則指派資料表物件變數為 $DB
if (isset($_GET['do'])) {
    if (isset(${ucfirst($_GET['do'])})) {
        $DB = ${ucfirst($_GET['do'])};
    }
} else {
    $DB = $Title;
}

// 若先前未定義 $_SESSION['visited']，代表新一次進站，則將 進站人數 + 1
if(!isset($_SESSION['visited'])){
    $Total->q("update `total` set `total`=`total`+1 where `id`=1");
    $_SESSION['visited']=1;     //若將 $_SESSION['visited']=0，isset()判斷結果會變成 null，造成誤判

}

?>
```