PHP y SQLite

Seleccionar los registros

Francisco Arce
www.pacoarce.com

Seleccionar los registros

Con la sentencia SELECT podemos extraer la información de una tabla.

Sintaxis:

```
SELECT column1, column2, columnN FROM nombre_tabla;
```

Para extraer todas las columnas, utilizamos en asterisco.

```
SELECT * FROM nombre_tabla;
```

Francisco Arce
www.pacoarce.com

Seleccionar los registros

SELECT [DISTINCT] *lista de columnas*
FROM *tablas*
WHERE *expresión*
GROUP BY *expresión_grupo*
HAVING *expresión_filtro*
ORDER BY *expresión_orden*
LIMIT *número*
OFFSET *número*

Francisco Arce
www.pacoarce.com

PHP y SQLite

La cláusula WHERE

Francisco Arce
www.pacoarce.com

La cláusula WHERE

Por medio de la cláusula WHERE podemos seleccionar por medio de una expresión. La cláusula WHERE es la misma para las sentencias SELECT, UPDATE y DELETE.

Usamos operadores dentro de esa expresión:

- Operadores matemáticos
- Operadores de comparación
- Operadores lógicos
- Operadores de desplazamientos de bits

PHP y SQLite

La cláusula DISTINCT

Francisco Arce
www.pacoarce.com

La cláusula DISTINCT

Con la cláusula DISTINCT eliminamos los registros duplicados de una selección.

Sintaxis:

```
SELECT DISTINCT column1, column2,.....columnN
FROM nombre_tabla
WHERE [condición]
```

Francisco Arce
www.pacoarce.com

PHP y SQLite

Operadores de comparación

Francisco Arce
www.pacoarce.com

Operadores de comparación

Suponga que la variable “a” tiene un valor de 5 y la variable “b”, un valor de 10.

Ejemplo:

== Verifica si dos valores son iguales o no. (a == b) Regresaría falso.
= Verifica si dos valores son iguales o no. (a = b) Regresaría falso.

Operadores de comparación

Suponga que la variable “a” tiene un valor de 5 y la variable “b”, un valor de 10.

Ejemplo:

!=	Verifica si dos valores son diferentes (true) o iguales (false). (a != b) is true.
<>	Verifica si dos valores son diferentes (true) o iguales (false). (a <> b) is true.

Operadores de comparación

Suponga que la variable “a” tiene un valor de 5 y la variable “b”, un valor de 10.

Ejemplo:

> Verifica si el operando de la izquierda es mayor al operando o constante de la derecha. (a > b) sería falso.
< Verifica si el operando de la izquierda es menor al operando o constante de la derecha. (a < b) regresaría verdadero.

Operadores de comparación

Suponga que la variable “a” tiene un valor de 5 y la variable “b”, un valor de 10.

Ejemplo:

>= Verifica si el operando de la izquierda es mayor o igual al operando o constante de la derecha. (a >= b) regresaría falso.
<= Verifica si el operando de la izquierda es menor o igual al operando o constante de la derecha. (a <= b) regresaría verdadero.

Operadores de comparación

Suponga que la variable “a” tiene un valor de 5 y la variable “b”, un valor de 10.

Ejemplo:

!< Verifica si el operando de la izquierda es “no menor” al operando o constante de la izquierda (verdadero). (a !< b) es falso.
!> Verifica si el operando de la izquierda es “no mayor” al operando o constante de la izquierda (verdadero). (a !> b) regresa verdadero.

PHP y SQLite

Operadores de matemáticos

Francisco Arce
www.pacoarce.com

Operadores de matemáticos

+	Suma valores de columnas o constantes
-	Resta valores entre columnas o constantes
*	Multiplicación entre columnas o constantes
/	División entre columnas o constantes
%	Módulo entre columnas o constantes

PHP y SQLite

Operadores de lógicos

Francisco Arce
www.pacoarce.com

Operadores de lógicos

AND	Es verdadera si todas las expresiones son verdaderas.
BETWEEN	Regresan los registros que se encuentran ENTRE dos valores.
EXISTS	Busca los renglones dentro de un subquery.
IN	Regresa los renglones si existe en una lista.

Operadores de lógicos

NOT IN No se encuentra en una lista.
LIKE Extrae registros similares utilizando comodines.
GLOB Es similar a LIKE pero es sensible a mayúsculas.
NOT Es el operador de negación. Es lo contrario a lo buscado, por ejemplo: NOT EXISTS, NOT BETWEEN, NOT IN, etc.

Operadores de lógicos

OR	Regresa verdadero si una de las condiciones .
IS NULL	Regresa si la columna es nula.
IS	Funciona como el operador igual a (=)
IS NOT	Funciona como el operador diferente a (!=)
||	Concatena dos cadenas..
UNIQUE	Regresa los valores que no estén duplicados.

PHP y SQLite

El operador GLOB

Francisco Arce
www.pacoarce.com

El operador GLOB

El operador GLOB es muy similar a LIKE, pero se apega más a UNIX o LINUX.

Es sensible a mayúsculas y minúsculas, utiliza como comodines “*” para una serie de carcateres o cero y “?” para un sólo caracter.

PHP y SQLite

La cláusula ORDER BY

Francisco Arce
www.pacoarce.com

La cláusula ORDER BY

Por medio de la cláusula ORDER BY nos permite ordenar la extracción de datos en forma ascendente o descendente.

Sintaxis:

```
SELECT lista_columnas
FROM nombre_tablas
[WHERE condición]
[ORDER BY columna1, columna2, .. columnaN] [ASC | DESC];
```

Francisco Arce
www.pacoarce.com

PHP y SQLite

La cláusula LIMIT

Francisco Arce
www.pacoarce.com

La cláusula LIMIT

La clásula LIMIT nos permite limitar el número de registros extraídos de una tabla.

Sintaxis:

```
SELECT columna1, columna2, columnaN
FROM tablas
LIMIT [num_renglones]
```

La cláusula LIMIT

La cláusula OFFSET nos indica a partir de cuál renglón haremos la extracción de la base de datos:

```
SELECT columna1, columna2, columnaN
FROM tablas
LIMIT [num_renglones] OFFSET [a_partir_del_renglón]
```

Francisco Arce
www.pacoarce.com

PHP y SQLite

Paginar una consulta

Francisco Arce
www.pacoarce.com

Paginar una consulta

```
//Limitar la busqueda
$TAMANO_PAGINA = 5;
$PAGINAS_MAXIMAS = 5;
```

Paginar una consulta

```
//examinar la página a mostrar
if(isset($_GET["p"])){
    $pagina = $_GET["p"];
} else {
    $pagina = 1;
}
```

Paginar una consulta

```
//Determinamos el número de registros
$inicio = ($pagina - 1) * $TAMANO_PAGINA;
$rows = $db->query("SELECT COUNT(*) as libros FROM libros");
$num = $rows->fetchArray();
$numLibros = $num['libros'];
```

Paginar una consulta

```
//Cálculo el total de páginas
$total_paginas = ceil($n / $TAMANO_PAGINA);
//
$sql = "SELECT * FROM libros LIMIT ".$TAMANO_PAGINA."
OFFSET ".$inicio;
$r = $db->exec($sql);
```

PHP y SQLite

La cláusula GROUP BY

Francisco Arce
www.pacoarce.com

La cláusula GROUP BY

La cláusula GROUP BY, dentro de SELECT, nos sirve para agrupar los registros idénticos.

La cláusula GROUP BY sigue a la cláusula WHERE y precede a ORDER BY (si se utiliza).

Puede utilizar más de una columna en GROUP BY.

Todas las columnas utilizadas en el GROUP BY deben estar incluidas en la lista de columnas del SELECT.

Francisco Arce
www.pacoarce.com

La cláusula GROUP BY

Sintaxis:

```
SELECT column-list
FROM table_name
WHERE [ conditions ]
GROUP BY column1, column2....columnN
ORDER BY column1, column2....columnN
```

PHP y SQLite

La cláusula HAVING

Francisco Arce
www.pacoarce.com

La cláusula HAVING

La cláusula HAVING permite filtrar los resultados obtenidos por medio de GROUP BY .

La cláusula HAVING afecta a las columnas que son usuadas en GROUP BY, no afecta a las columnas utilizadas en el SELECT.

La cláusula HAVING debe seguir la cláusula GROUP BY en una consulta y también preceder a la cláusula ORDER BY (si se utiliza).

Francisco Arce
www.pacoarce.com

La cláusula HAVING

Sintaxis:

```
SELECT columna1, columna2
FROM tabla1, tabla2
WHERE [ condiciones ]
GROUP BY columna1, columna2
HAVING [ condiciones ]
ORDER BY columna1, columna2
```