http://adodb.org/dokuwiki/lib/exe/fetch.php?media=logo.png

**ADOdb MySQL Tutorial**

PHP está especialmente diseñado para la creación de sitios webs dinámicos. Para crear estos sitios normalmente se utiliza algún tipo de base de datos desde donde obtenemos la información que queremos mostrar, ya sean noticias, preguntas y respuestas de un foro u otro tipo de información dinámica. Si usamos php normalmente usaremos MySQL como base de datos para iniciar nuestro sitio, el problema aparece cuando nuestro proyecto crece tanto que necesita hacer uso de otro tipo de base de datos más robusta.

Desafortunadamente el acceso en PHP cada base de datos es muy diferente. Para conectarnos a mysql, debemos usar mysql\_connect(); cuando decida cambiar a Oracle o Microsoft SQL Server, debe usar ocilogon() o mssql\_connect() respectivamente. Lo peor es que también los parámetros de cada función son diferentes.

Una librería de abstracción de los datos como ADODB es lo que Ud. necesita si desea asegurar en gran medida la portabilidad de su aplicación. Provee una serie de funciónes comunes para comunicarse con las distintas bases de datos.

ADODB viene de " Active Data Objects DataBase" (no muy original pero efectivo). ADODB actualmente soporta MySQL, PostgreSQL, Oracle, Interbase, Microsoft SQL Server, Access, FoxPro, Sybase, ODBC y ADO. Puede obtener ADODB en <http://adodb.sourceforge.net/#download>.

**Ejemplo MySQL**

La base de datos usada con más frecuencia en PHP es MySQL, entonces es posible que le resulte familiar el siguiente código. Conectamos al server MySQL en localhost, a la base de datos mydb, y ejecutamos una sentencia SQL. El resultado es impreso en pantalla, una línea por cada fila obtenida.

$db = mysql\_connect("localhost", "root", "password");  
mysql\_select\_db("mydb",$db);  
$result = mysql\_query("SELECT \* FROM employees",$db);  
if ($result === false) die("failed");   
while ($fields = mysql\_fetch\_row($result)) {  
for ($i=0, $max=sizeof($fields); $i < $max; $i++) {  
print $fields[$i].' ';  
}  
print "<br>n";

}

El código superior está coloreado por sección. La primer seccion es la conexión a la base de datos. La segunda sección es la ejecución de la sentencia SQL, y la última es la impresión del resultado. El loop while recorre todo las filas hasta el final.

**Aquí su equivalente en ADODB**

include("adodb.inc.php");  
$db = NewADOConnection('mysql');  
$db->Connect("localhost", "root", "password", "mydb");  
$result = $db->Execute("SELECT \* FROM employees");  
if ($result === false) die("failed");   
while (!$result->EOF) {  
for ($i=0, $max=$result->FieldCount(); $i < $max; $i++)  
print $result->fields[$i].' ';  
$result->MoveNext();  
print "<br>n";  
}

Ahora portarlo a Oracle es tan simple como cambiar la segunda linea a NewADOConnection('oracle'). Vamos a mirar cada parte del código.

**Conectando a la base de datos**

include("adodb.inc.php");  
$db = NewADOConnection('mysql');  
$db->Connect("localhost", "root", "password", "mydb");

La conexión puede parecer algo más complicada que en MySQL pero se explica en que ADODB es totalmente orientado a objectos. Si no está familiarizado con la programación orientada a objetos no se preocupe porque todo queda oculto en la función NewADOConnection() .

Para ahorrar memoria solo se carga las funciónes específicas de la base de datos que vamos a utilizar. Cada driver tiene un archivo distinto que se carga mediante un include automáticamente.

Una vez cargado el driver que vamos a utilizar mediante la función NewADOConnection() nos conectamos a la base de datos usando $db->Connect().

**Ejecutando SQL**

$result = $db->Execute("SELECT \* FROM employees");  
if ($result === false) die("failed");

Para enviar una sentencia sql al motor de base de datos se utiliza la función Execute(). Esta función devueve un objeto "recordset" si la ejecución fue correcta o un "false" si huvo algun error.

Existen dos objetos principales en ADODB. El objeto connection y el objeto recordset. Cuando usar cada uno?

El objeto connection ($db) es responsable de la conexión a la base de datos, formateo de las SQL y las consultas al servidor. El objeto recordset ($result) es responsable de la obtención de los resultados y el formateo de los resultados.

**Obteniendo datos**

while (!$result->EOF) {  
for ($i=0, $max=$result->FieldCount(); $i < $max; $i++)  
print $result->fields[$i].' ';  
$result->MoveNext();  
print "<br>n";  
}

El proceso para obtener datos es como el proceso para leer desde un fichero. Para cada linea observamos si hemos llegado al final del fichero (EOF). Mientras no lleguemos al final del fichero seguimos leyendo y moviéndonos a la siguiente línea (movenext).

El array $result->fields[] es generaro por ADODB ante cada peticion de movenext(). Podemos obtener array asociativos o indexados por el número del campo.

$ADODB\_FETCH\_MODE = ADODB\_FETCH\_NUM;  
$rs1 = $db->Execute('select \* from table');  
$ADODB\_FETCH\_MODE = ADODB\_FETCH\_ASSOC;  
$rs2 = $db->Execute('select \* from table');  
print\_r($rs1->fields); // muestra array([0]=>'v0',[1] =>'v1')  
print\_r($rs2->fields); // muestra array(['col1']=>'v0',['col2'] =>'v1')

Como se ve en el ejemplo, los dos recorsets usan diferentes métodos para almacenar los datos recuperados de la consulta usando $ADODB\_FETCH\_MODE (el cambio de $ADODB\_FETCH\_MODE debe ser previo a Execute()).

**ADOConnection**

El objeto que realiza la conexión a la base de datos, ejecuta las sentencias SQL y tiene otro set de funciónes para estandarizar el formato de sentencias SQL como la concatenación de cadenas o formatos de fechas.

**Otras funciónes especiales**

$recordset->Move($pos) mueve el recordset a una fila en particular. ADODB soporta avance de filas en todas las bases de datos. Algunas bases de datos no soportan retroceso en el recordset. Puede de todas maneras simular este comportamiento usando un caché.

$recordset->RecordCount() retorna el total de filas obtenidas en el recordset. Algunas bases de datos no soportan esto y se devuelven un -1.

$recordset->GetArray() coloca el resultado del recordset en un array.

rs2html($recordset) esta función genera una tabla HTML con el resultado del recordset.

include('adodb.inc.php');   
include('tohtml.inc.php'); /\* includes the rs2html function \*/  
$conn = &ADONewConnection('mysql');   
$conn->PConnect('localhost','userid','password','database');  
$rs = $conn->Execute('select \* from table');  
rs2html($rs); /\* recordset to html table \*/

Más funciónes útiles pueden ser encontradas en http://adodb.sourceforge.net/#docs.

**Temas avanzados  
Inserts y Updates**

Imagine que tenemos los siguientes datos para guardar en una base de datos.

ID = 3  
TheDate=mktime(0,0,0,8,31,2001) /\* 31st August 2001 \*/  
Note= sugar why don't we call it off

Cuando cambie de base de dato este formato de datos puede no funciónar.

El primer problema es que cada base de datos usa las fechas en un formato distinto. . MySQL espera YYYY-MM-DD como formato, otras bases de datos requierien otros formatos. ADODB tiene una función que puede ayudarnos DBDate() que convierte una fecha en el formato correcto para realizar updates o inserts.

La comilla simple del campo Note también es un problema. La función qstr() nos puede ayudar en esto.

Entonces como usamos estas funciónes ?

$sql = "INSERT INTO table (id, thedate,note) values ("   
. $ID . ','  
. $db->DBDate($TheDate) .','  
. $db->qstr($Note).")";  
$db->Execute($sql);

ADODB tiene tambien dos funciónes que ayudan a reconocer si un insert update o delete han sido correctos.   
$connection->Affected\_Rows() (devuelve el total de filas afectadas por un insert o un delete ) y $recordset->Insert\_ID() (devuelve el último valor generado por un campo autoincrement en un insert). No todas las bases de datos soportan estas funciónes (vea la documentación).

**MetaTypes**

Existen funciónes para recuperar información de cada campo, el método FetchField($fieldoffset). Esto devuelve un objeto con tres propiedades name, type y max\_length. Si max\_length es desconocido o no se puede obtener es = a -1.

Por Ejemplo.

$recordset = $conn->Execute("select adate from table");  
$f0 = $recordset->FetchField(0);

$f0->name contiene 'adata', $f0->type contiene 'date'.

Cada base de datos define sus columnas de una manera particular, ADODB tiene una función especial para estandarizar los tipos de datos, MetaType($type, $max\_length).

C: character and varchar types  
X: text or long character (eg. more than 255 bytes wide).  
B: blob or binary image  
D: date  
T: timestamp  
L: logical (boolean)  
I: integer  
N: numeric (float, double, money)

En el ejemplo de arriba.

$recordset = $conn->Execute("select adate from table");  
$f0 = $recordset->FetchField(0);  
$type = $recordset->MetaType($f0->type, $f0->max\_length);  
print $type; /\* debe imprimir 'D' por fecha\*/

**Limites de selección en una SQL**

ADODB tiene una función que asegura la portabilidad de una sql en el tema de los límites de un recordset, $connection->SelectLimit($sql,$nrows,$offset), en el caso de los productos Microsoft se usará SELECT TOP, en mysql, postgresql se usará LIMIT y en otras bases de datos que no lo soporten se emula.

**Cacheo de consultas**

ADODB permite cachear los resultados de las consultas SQL en función de un tiempo definido. Las consultas deben entonces realizarse con connection->CacheExecute($secs2cache,$sql) y $connection->CacheSelectLimit($secs2cache,$sql,$nrows,$offset). n.trad. Los resultados se almacenan en un directorio que debe tener los permisos apropiados.

**Soporte de sesiones PHP4**

ADODB soporta sesiones PHP. Puede guardar variables de sesiones en una base de datos para asegurar escalabilidad. Para más información visite [docs-session-es.htms](http://www.lacorona.com.mx/fortiz/adodb/docs-session-es.htm)

Si está pensando en escribir software comercial en PHP y desea distribuirlo, debe considerar ADODB. Está siendo usado con éxito, es LGPL, que significa que puede ser distribuido con aplicaciones comerciales, dejando a su código propietario.

**Conclusión**

ADODB es lo suficientemente robusto y completo para casi cualquier aplicación. Está siendo actualizado permanentemente y tiene muchos seguidores por lo que es una muy buena opción.