PHP + Oracle

```
<?
# ********************************************************
# database class ( 2009/09/14 )
# キャラクタセット変換を利用する場合は、
# 本体で以下を実行して下さい
# 1) mb_language( "ja" );
# 2) mb_internal_encoding("UTF-8");
# または
# 2') mb_internal_encoding("EUC-JP");
# ********************************************************
class Oracle {

var $Connect;
var $Result;

var $nField;
var $nRow;

var $Debug;
var $Error;☒ 未使用:ADOのエラー処理をする場合
var $Work;☒ 未使用

var $Cn;
var $Rs;
var $ConnectionString;

# ********************************************************
# Microsoft の仕様そのままで良いので、
# ドライバ指定も可能
# ********************************************************
function Oracle( $Server='default', $User='default', $Password='default' ) {

$Server = $Server == 'default' ? $GLOBALS['conf_db_host'] : $Server;
$User = $User == 'default' ? $GLOBALS['conf_db_user'] : $User;
$Password = $Password == 'default' ? $GLOBALS['conf_db_pass'] : $Password;

$this->Cn = new COM( "ADODB.Connection" );
$this->Cn->CursorLocation = 3;☒ クライアント側 カーソル
$this->Rs = new COM( "ADODB.Recordset" );

$ConnectionString = "Provider=MSDASQL;";
$ConnectionString .= "DSN=$Server;";
$ConnectionString .= "UID=$User;";
$ConnectionString .= "PWD=$Password;";

$this->Cn->Open( $ConnectionString );
if ( $GLOBALS['conf_db_connect_action'] != '' ) {
$this->Cn->Execute( $GLOBALS['conf_db_connect_action'] );
}

$this->Debug = FALSE;
}

# ********************************************************
# 接続解除
# ********************************************************
function Close( ) {

@$this->Cn->Close();

}
```

```
# ********************************************************
# select の実行
# ********************************************************
function Query( $SqlQuery ) {

// DB に引き渡す SQL のキャラクタセット変換
// ( ※ 通常は使用しません )
if ( $GLOBALS['conf_db_charset'] != '' ) {
$SqlQuery = mb_convert_encoding(
$SqlQuery,
$GLOBALS['conf_db_charset'],
$GLOBALS['conf_client_charset']
);
}

// レコードセットが開いたまの場合は閉じる
if ( $this->Rs->State >= 1 ) {
$this->Rs->Close();
}
// 接続オブジェクトでレコードセットを開く
$this->Rs->Open( $SqlQuery, $this->Cn );
// データが無い場合は false を返す
$ret = !$this->Rs->EOF;

return $ret;
}

# ********************************************************
# データの読み出し
# PHP で使う一般的な仕様に合わせて、連想配列化
# ********************************************************
function Fetch( $Result ) {

// 配列へ、インデックスと名前でデータをセット
$ret = array();
for( $i = 0; $i < $this->Rs->Fields->count; $i++ ) {
// 日付型( adDate )
if ( $this->Rs->Fields[$i]->type == 7 ) {
// PHP4 のみ
if ( substr(phpversion(),0,1) == '4' ) {
$test = date( "H:i:s", $this->Rs->Fields[$i]->value );
if ( $test == "00:00:00" ) {
// 時分秒が無い場合は文字列として省く
$ret[$i] = date(
"Y/m/d",
$this->Rs->Fields[$i]->value
);
$ret[$this->Rs->Fields[$i]->name] = date(
"Y/m/d",
$this->Rs->Fields[$i]->value
);
}
else {
$ret[$i] = date(
"Y/m/d H:i:s",
$this->Rs->Fields[$i]->value
);
$ret[$this->Rs->Fields[$i]->name] = date(
"Y/m/d H:i:s",
$this->Rs->Fields[$i]->value
);
}
}
```

```
// PHP5 以降
else {
$ret[$i] =
$this->Rs->Fields[$i]->value;
$ret[$this->Rs->Fields[$i]->name] =
$this->Rs->Fields[$i]->value;
}
}
else {
$ret[$i] =
$this->Rs->Fields[$i]->value;
$ret[$this->Rs->Fields[$i]->name] =
$this->Rs->Fields[$i]->value;
}
}

// 変数にセットされたキャラクタセット変換
// ( ※ 通常は使用しません )
if ( $GLOBALS['conf_db_charset'] != '' ) {

if ( $ret ) {
$ret2 = array();
while (list($Key, $Value) = @each($ret)) {
$ret2[$Key] = mb_convert_encoding(
$Value,
$GLOBALS['conf_client_charset'],
$GLOBALS['conf_db_charset']
);
$Key2 = mb_convert_encoding(
$Key,
$GLOBALS['conf_client_charset'],
$GLOBALS['conf_db_charset']
);
$ret2[$Key2] = mb_convert_encoding(
$Value,
$GLOBALS['conf_client_charset'],
$GLOBALS['conf_db_charset']
);
}
}

}
else {
$ret2 = $ret;
}

return $ret2;
}

# *********************************************************
# フィールド名を取得
# *********************************************************
function FieldName( $idx ) {
$ret = $this->Rs->Fields[$idx]->name;
if ( $GLOBALS['conf_db_charset'] != '' ) {
$ret = mb_convert_encoding(
$ret,
$GLOBALS['conf_client_charset'],
$GLOBALS['conf_db_charset']
);
}
return $ret;
}
```

```
# ********************************************************
# 外部から使うループを想定した SQL 実行
# ********************************************************
function QueryEx( $SqlQuery='' ) {

// 初回 ( SQL を実行 )
if ( $SqlQuery != '' ) {
if ( $this->Debug ) {
print "<TABLE border=0 cellpadding=5>";
print "<TH align=left bgcolor=skyblue><pre>" .
$this->Arrange($SqlQuery) . "</pre></TD>";
print "</TABLE>";
}

// フィールド一覧を取得するこのクラスのコマンド
if ( substr( $SqlQuery, 0, 7 ) == "COLUMNS" ) {
$table_name = Explode( " ", $SqlQuery );
$this->Rs = $this->Cn->OpenSchema( 4 );
if ( $GLOBALS['conf_db_charset'] != '' ) {
$tbl = mb_convert_encoding(
$table_name[1],
$GLOBALS['conf_db_charset'],
$GLOBALS['conf_client_charset']
);
}
else {
$tbl = $table_name[1];
}
$this->Rs->Filter = "TABLE_NAME = '" . $tbl . "'";
$this->Rs->Sort = "ORDINAL_POSITION";
}
// 通常り処理
else {
$this->Result = $this->Query( $SqlQuery );
if ( !$this->Result ) {
return FALSE;
}
}

// フィールド数と行数をプロパティにセット
$this->nField = $this->Rs->Fields->count;
$this->nRow = $this->Rs->RecordCount;

return $this->Fetch ( $this->Result );
}
//  ２回目以降
else {
$this->Rs->MoveNext();
if ( $this->Rs->EOF ) {
return FALSE;
}
return $this->Fetch ( $this->Result );
}

}

# ********************************************************
# レコードセットを返さない SQL の実行
# ********************************************************
function Execute( $SqlExec ) {

if ( $this->Debug ) {
print "<TABLE border=0 cellpadding=5>";
print "<TH align=left bgcolor=skyblue><pre>" .
```

```
$this->Arrange($SqlExec) . "</pre></TD>";
print "</TABLE>";
}

// 変数にセットされたキャラクタセット変換
// ( ※ 通常は使用しません )
if ( $GLOBALS['conf_db_charset'] != '' ) {
$SqlExec = mb_convert_encoding(
$SqlExec,
$GLOBALS['conf_db_charset'],
$GLOBALS['conf_client_charset']
);
}

$this->Cn->Execute( $SqlExec );

// 必ず true を返す
$ret = TRUE;

return $ret;
}


# ********************************************************
# 内部関数
# ********************************************************
function Arrange( $target ) {
$ret = str_replace( ',', "\n\t,", $target );
$ret = str_replace( 'set ', "set\n\t", $ret );
$ret = str_replace( 'where ', "\nwhere ", $ret );
return "\n$ret";
}
}
?>
```