1. Wstęp

PHP jest to skryptowy język programowania wykonywany po stronie serwera (server-side) służący do generowania stron internetowych. Skrypty napisane w PHP po uruchomieniu nie są kompilowane do postaci kodu maszynowego, tylko są wykonywane przez specjalną aplikację zwaną interpreterem PHP.
Język ten został stworzony głównie do tworzenia dynamicznych stron WWW. Obecnie można w nim pisać również zwyczajne aplikacje dla systemu operacyjnego.

Skrypty napisane w PHP są z reguły umieszczane w dokumentach tekstowych (najczęściej w formacie HTML lub XHTML). Można także wykonywać je z linii poleceń (podobnie jak w językach Perl czy Python).
PHP umożliwia współpracę z wieloma rodzajami źródeł danych, jak na przykład serwery relacyjnych baz danych, pliki tekstowe czy dokumenty XML.

Żeby skrypt został poprawnie rozpoznany przez interpreter języka, musi zostać umieszczony w pliku o odpowiednim rozszerzeniu (najczęściej .php, .php3, czy .phml).

Przykładowy skrypt umieszczony w dokumencie HTML wyświetlający na ekranie powitalny napis:

```
<html>
<body>
  <?php
        echo 'Hello world! :)';
  ?>
</body>
</html>
```

Aby mieć możliwość użycia języka PHP do obsługi bazy danych potrzebny jest:

Serwer WWW

Zainstalowany na lokalnym komputerze lub wykupiony u profesjonalnego dostawcy serwer HTTP. Najbardziej popularnym serwerem jest obecnie Apache HTTP Server Project. Jest on bezpłatny, bezpieczny i łatwy w konfiguracji. Odpowiednią wersję dla danego systemu operacyjnego możemy pobrać ze strony `http://httpd.apache.org`.

Interpreter PHP

Ze strony `http://www.php.net` można pobrać najnowszą wersję instalacyjną dla odpowiedniego systemu operacyjnego albo kod źródłowy.

Serwer systemu bazodanowego, który chcemy obsługiwać

MySQL – `http://www.mysql.com/` (instalacja php z parametrem *--with-mysql[=DIR])*
Oracle – `http://www.oracle.com/technology/tech/oci/instantclient/index.html`
PostgreSQL – `http://www.postgresql.org/`
MS SQL – `http://www.microsoft.com`, dla Unix/Linux: `http://www.freetds.org/`
IBM DB2 - `http://www-306.ibm.com/software/data/db2/udb/support/downloadv8.html`

Łatwo możemy sprawdzić czy nasz serwer obsługuje PHP i czy jest zainstalowany serwer bazy danych np. MySQL.

Tworzymy na serwerze plik tekstowy: **http://mojastrona.com/phpinfo.php**

```
<?php
  phpinfo();
?>
```

Po uruchomieniu instrukcja phpinfo() powinna wyświetlić tabelę z informacjami o zainstalowanej wersji PHP, serwera oraz innych zmiennych środowiskowych oraz dodatków (np. baz danych).

PHP Version 4.4.2

System	Linux geolog 2.4.31 #3 SMP Tue Jan 3 12:27:05 CET 2006 i686
Build Date	Feb 10 2006 15:52:02
Configure Command	'./configure' '--prefix=/usr' '--disable-static' '--with-apxs=/usr/sbin/apxs' '--sysconfdir=/etc' '--enable-discard-path' '--with-config-file-path=/etc/apache' '--enable-safe-mode' '--with-openssl' '--with-ldap' '--with-mhash' '--enable-bcmath' '--with-bz2' '--with-pic' '--enable-calendar' '--enable-ctype' '--with-gdbm' '--with-db4' '--with-imap-ssl=/usr/local/lib/c-client' '--with-imap=/usr/local/lib/c-client' '--enable-dbase' '--enable-ftp' '--with-iconv' '--with-dom' '--with-exif' '--enable-exif' '--with-gd' '--enable-gd-native-ttf' '--with-jpeg-dir=/usr' '--with-png' '--with-gmp' '--enable-mbstring' '--with-curl=/usr' '--with-pcre-regex=/usr' '--with-mysql=shared,/usr' '--with-gettext=shared,/usr' '--with-expat-dir=/usr' '--with-xml' '--enable-wddx' '--with-mm=/usr' '--enable-trans-sid' '--enable-shmop' '--enable-sockets' '--with-regex=php' '--enable-sysvsem' '--enable-sysvshm' '--enable-yp' '--enable-memory-limit' '--with-tsrm-pthreads' '--enable-shared' '--disable-debug' '--with-zlib=/usr'
Server API	Apache
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/apache/php.ini
PHP API	20020918
PHP Extension	20020429
Zend Extension	20050606
Debug Build	no
Zend Memory Manager	enabled
Thread Safety	disabled
Registered PHP Streams	php, http, ftp, https, ftps, compress.bzip2, compress.zlib

apache

APACHE_INCLUDE	*no value*
APACHE_TARGET	*no value*
Apache Version	Apache/1.3.34 (Unix) mod_ssl/2.8.25 OpenSSL/0.9.8a PHP/4.4.2
Apache Release	10334100
Apache API Version	19990320
Hostname:Port	www.geol.agh.edu.pl:80
User/Group	nobody(99)/98
Max Requests	Per Child: 0 - Keep Alive: off - Max Per Connection: 100
Timeouts	Connection: 300 - Keep-Alive: 15
Server Root	/usr
Loaded Modules	mod_ssl, mod_php4, mod_setenvif, mod_so, mod_unique_id, mod_usertrack, mod_headers, mod_expires, mod_cern_meta, mod_proxy, mod_digest, mod_auth_dbm, mod_auth_anon, mod_auth, mod_access, mod_rewrite, mod_alias, mod_userdir, mod_speling, mod_actions, mod_imap, mod_asis, mod_cgi, mod_dir, mod_autoindex, mod_include, mod_info, mod_status, mod_negotiation, mod_mime, mod_mime_magic, mod_log_config, mod_define, mod_env, mod_vhost_alias, http_core

Directive	Local Value	Master Value
child_terminate	0	0
engine	1	1
last_modified	0	0
xbithack	0	0

mysql

MySQL Support	enabled
Active Persistent Links	0
Active Links	0
Client API version	5.0.18
MYSQL_MODULE_TYPE	external
MYSQL_SOCKET	/var/run/mysql/mysql.sock
MYSQL_INCLUDE	-I/usr/include/mysql
MYSQL_LIBS	-L/usr/lib -lmysqlclient

Directive	Local Value	Master Value
mysql.allow_persistent	On	On
mysql.connect_timeout	60	60
mysql.default_host	*no value*	*no value*
mysql.default_password	*no value*	*no value*
mysql.default_port	*no value*	*no value*
mysql.default_socket	*no value*	*no value*
mysql.default_user	*no value*	*no value*
mysql.max_links	Unlimited	Unlimited
mysql.max_persistent	Unlimited	Unlimited
mysql.trace_mode	Off	Off

Apache Environment

Variable	Value
DOCUMENT_ROOT	/var/www/htdocs
HTTP_ACCEPT	text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
HTTP_ACCEPT_CHARSET	ISO-8859-2,utf-8;q=0.7,*;q=0.7
HTTP_ACCEPT_ENCODING	gzip,deflate
HTTP_ACCEPT_LANGUAGE	pl,en-us;q=0.7,en;q=0.3
HTTP_CONNECTION	keep-alive
HTTP_HOST	www.geol.agh.edu.pl
HTTP_KEEP_ALIVE	300
HTTP_USER_AGENT	Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.8.1.3) Gecko/20070309 Firefox/2.0.0.3
PATH	/sbin:/bin:/usr/sbin:/usr/bin
REMOTE_ADDR	83.4.196.200
REMOTE_PORT	4219
SCRIPT_FILENAME	/export/home1/s99832/www/phpinfo.php
SERVER_ADDR	149.156.104.1
SERVER_ADMIN	root@geol.agh.edu.pl
SERVER_NAME	www.geol.agh.edu.pl
SERVER_PORT	80
SERVER_SIGNATURE	<ADDRESS>Apache/1.3.34 Server at www.geol.agh.edu.pl Port 80</ADDRESS>
SERVER_SOFTWARE	Apache/1.3.34 (Unix) mod_ssl/2.8.25 OpenSSL/0.9.8a PHP/4.4.2
UNIQUE_ID	RksS238AAAEAAGYsmH0
GATEWAY_INTERFACE	CGI/1.1
SERVER_PROTOCOL	HTTP/1.1
REQUEST_METHOD	GET
QUERY_STRING	*no value*
REQUEST_URI	/~s99832/phpinfo.php
SCRIPT_NAME	/~s99832/phpinfo.php

Podstawowe funkcje i metody dostępu do baz danych z poziomu skryptów PHP nieznacznie różnią się od siebie (przeważnie przedrostkiem nazwy) w zależności od systemu baz danych. Natomiast przedstawione algorytmy np. odczytu danych z bazy pozostają niezmienne.

2. PostgreSQL

Zaczynamy nasz skrypt php. Na początku deklarujemy zmienne potrzebne do połączenia:

```
<?
$host = "localhost";
$user = "postgres";
$pass = "postgres";
$db = "test";
```

2.1 Połączenie

pg_connect() — Otwiera tymczasowe (nietrwałe) połączenie do PostgreSQL-a określonego przez łańcuch połączenia (ang. connection string).

pg_connect("host=myHost port=myPort tty=myTTY options=myOptions dbname=myDB user=myUser password=myPassword ");

```
$connection = pg_connect("host=$host
                          dbname=$db
                          user=$user
                          password=$pass")
          or die('Nie można się połączyć' . pg_last_error());
echo "Poprawnie połączono";
```

pg_pconnect() zachowuje się prawie jak pg_connect() z dwoma zasadniczymi różnicami:
- Pierwsza, podczas łączenia funkcja najpierw spróbuje znaleźć połączenie (persistent = stałe) już otwarte dla tego samego hosta, użytkownika i hasła. Jeżeli je znajdzie, jego identyfikator zostanie zwrócony zamiast otwierania nowego połączenia.
- Druga, połączenie z serwerem SQL nie zostanie zamknięte po zakończeniu wykonywania skryptu. Zamiast tego połączenie pozostanie otwarte do późniejszego użycia (pg_close() nie zamyka połączeń nawiązanych za pomocą pg_pconnect()).

2.2 Odczyt danych

pg_query() — Wykonuje zapytanie do bazy w języku SQL
pg_query() zwraca identyfikator wyniku jeśli zapytnie zostało wykonane lub FALSE w razie niepowodzenia

```
$query = "SELECT * FROM osoby ORDER BY zarobki DESC";
$result = pg_query($connection, $query)
        or die('Problem z zapytaniem: ' . $query .' - '. pg_last_error());
```

pg_fetch_assoc() - jako parametr przyjmuje identyfikator wyniku zapytania, natomiast wartością zwracaną jest tablica asocjacyjna zawierająca dane z jednego wiersza tabeli.
pg_num_rows() – zwraca ilość pobranych wierszy.

W pętli while pobierane są kolejne wiersze tabeli zwróconej przez zapytanie wybierające i umieszczane w tablicy asocjacyjnej $myrow.

```
if (pg_num_rows($result) != 0) {
   while($myrow = pg_fetch_assoc($result)) {
        echo $myrow['id'];
        echo $myrow['imie'];
        echo $myrow['nazwisko'];
        echo $myrow['zarobki'];
        echo "<br /><hr />";
   }
```

```
} else {
   echo "Brak wyników wyszukiwania";
}
```

pg_fetch_result() - jako parametry przyjmuje identyfikator wyniku zapytania oraz współrzędne elementu tablicy (wyniku zapytania) i zwraca wartość znajdującą się w określonej współrzędnymi komórce tablicy

```
$result = pg_query($connection, "select version()");
$version = pg_fetch_result($result, 0, 0);
echo $version;
```

pg_fetch_array() - zwraca tablicę zawierającą dane z pobranego wiersza (numer) lub FALSE jeśli nie ma więcej wierszy

```
$array = pg_fetch_array ($result, 0, PGSQL_NUM);
echo $array[2];

$array = pg_fetch_array ($result, 1, PGSQL_ASSOC);
echo $array["nazwisko"];
```

2.3 Dodawanie danych

Dodawanie i modyfikacja danych to zadania realizowane przez odpowiednie zapytania SQL. Stosujemy dokładnie te same metody komunikacji z bazą danych, jak w przypadku odczytu danych.

```
$query = "INSERT INTO osoby VALUES('$id','$imie','$nazwisko','$zarobki')";
$result = pg_query($connection, $query)
          or die('Wystąpił błąd: ' . pg_last_error());

printf ("Dodano nowy rekord: %s, %s, %s, %s", $id, $imie, $nazwisko,
$zarobki);
```

2.4 Modyfikacja danych

```
$query = "UPDATE osoby SET imie='$imie', nazwisko='$nazwisko',
zarobki='$zarobki' where id='$id'";
$result = pg_query($connection, $query)
          or die('Wystąpił błąd: ' . pg_last_error());

printf ("Dane zostały zmienione: %s %s %s %s", $id, $imie, $nazwisko,
$zarobki);
```

2.5 Usuwanie danych

pg_affected_rows() zwraca liczbę wierszy dodanych, zmodyfikowanych lub usuniętych przez zapytania INSERT, UPDATE i DELETE.

```
$query = "DELETE FROM osoby where id='$id'";
$result = pg_query($connection, $query)
          or die('Wystąpił błąd: ' . pg_last_error());

$count = pg_affected_rows ($result);
printf ("Usunięto %s wierszy.", $count);
```

2.6 Zamknięcie połączenia

Zamyka tymczasowe połączenie z serwerem związane z podanym parametrem połączenia.

```
pg_close($connection);
?>
```

3. MySQL

Deklarujemy zmienne potrzebne do połączenia – parametry połączenia:

```
<?php
$host = "localhost";
$user = "mysql";
$pass = "mysql";
$db = "test";
```

3.1 Połączenie

```
$connection = mysql_connect($host, $user, $pass)
              or die('Nie można się połączyć' . mysql_error());
echo "Poprawnie połączono";
```

lub

```
$connection = mysql_pconnect($host, $user, $pass)
              or die('Nie można się połączyć' . mysql_error());
echo "Poprawnie połączono";
```

Musimy także wybrać bazę danych na której będziemy operować:

```
mysql_select_db($db)
              or die ('Nie mozna wybrać bazy danych');
```

3.2 Odczyt danych

```
$query = "SELECT * FROM osoby ORDER BY zarobki DESC";
$result = mysql_query($query)
        or die('Problem z zapytaniem: ' . $query .' - '. mysql_error());


if (mysql_num_rows($result) != 0) {
   while($myrow = mysql_fetch_assoc($result)) {
        echo $myrow['id'];
        echo $myrow['imie'];
        echo $myrow['nazwisko'];
        echo $myrow['zarobki'];
        echo "<br /><hr />";
   }
} else {
   echo "Brak wyników wyszukiwania";
}

$array = mysql_fetch_array ($result, MYSQL_NUM);
echo $array[2];

$array = mysql_fetch_array ($result, MYSQL_ASSOC);
echo $array["nazwisko"];

$array = mysql_fetch_array ($result, MYSQL_BOTH);
echo $array[1].$array["nazwisko"];
```

3.3 Dodawanie danych

```
$query = "INSERT INTO osoby VALUES('$id','$imie','$nazwisko','$zarobki')";
$result = mysql_query($query)
        or die('Wystąpił błąd: ' . mysql_error());
printf ("Dodano nowy rekord: %s, %s, %s, %s", $id, $imie, $nazwisko,
$zarobki);
```

3.4 Modyfikacja danych

```
$query = "UPDATE osoby SET imie='$imie', nazwisko='$nazwisko',
zarobki='$zarobki' where id='$id'";
$result = mysql_query($query);
        or die('Wystąpił błąd: ' . mysql_error());
printf ("Dane zostały zmienione: %s %s %s %s", $id, $imie, $nazwisko,
$zarobki);
```

3.5 Usuwanie danych

```
$query = "DELETE FROM osoby where id='$id'";
$result = mysql_query($query);
        or die('Wystąpił błąd: ' . mysql_error());
$count = mysql_affected_rows ($result);
printf ("Usunięto %s wierszy.", $count);
```

3.6 Zamknięcie połączenia

Dodatkowo przed zamknięciem połączenia możemy zwolnić pamięć zajmowaną przez wynik przypisany do danego wskaźnika $result. Funkcji tej używa się tylko w wypadkach obawy zajęcia zbyt dużej ilości pamięci przez zapytania zwracające duże ilości danych. Pamięć jest automatycznie zwalniana po zakończeniu działania skryptu.

```
mysql_free_result($result);

mysql_close($connection);
?>
```

3.7 Tworzenie i usuwanie bazy

Służą do tego dwa polecenia: mysql_create_db() i mysql_drop_db().
Stosowanie tych funkcji nie jest jednak niezalecane. Sugerowane jest wydawanie poleceń CREATE DATABASE i DROP DATABASE przy użyciu funkcji mysql_query().

```
if (mysql_create_db($db)) {
        print ("Utworzono bazę\n");
    } else {
        printf ("Błąd podczas tworzenia bazy: %s\n", mysql_error());
    }

if (mysql_drop_db($db)) {
        print ("Utworzono bazę\n");
    } else {
        printf ("Błąd podczas tworzenia bazy: %s\n", mysql_error());
    }
```

4. MSSQL

Zestaw zmiennych potrzebnych do połączenia podobny:

```
<?php
$host = "localhost";
$user = "mssql";
$pass = "mssql";
$db = "test";
```

4.1 Połączenie

```
$connection = mssql_connect($host, $user, $pass)
              or die('Nie można się połączyć');
echo "Poprawnie połączono";
```

lub

```
$connection = mssql_pconnect($host, $user, $pass')
              or die('Nie można się połączyć');
echo "Poprawnie połączono";
```

oraz wybór bazy danych:

```
mssql_select_db($db, $connection)
              or die ('Nie mozna wybrać bazy danych');
```

4.2 Odczyt danych

```
$query = "SELECT * FROM osoby ORDER BY zarobki DESC";
$result = mssql_query($query)
        or die('Problem z zapytaniem: ' . $query );


if (mssql_num_rows($result) != 0) {
   while($myrow = mssql_fetch_assoc($result)) {
        echo $myrow['id'];
        echo $myrow['imie'];
        echo $myrow['nazwisko'];
        echo $myrow['zarobki'];
        echo "<br /><hr />";
   }
} else {
   echo "Brak wyników wyszukiwania";
}

$array = mssql_fetch_array ($result, MSSQL_NUM);
echo $array[2];

$array = mssql_fetch_array ($result, MSSQL_ASSOC);
echo $array["nazwisko"];

$array = mssql_fetch_array ($result, MSSQL_BOTH);
echo $array[1].$array["nazwisko"];
```

4.3 Dodawanie danych

```
$query = "INSERT INTO osoby VALUES('$id','$imie','$nazwisko','$zarobki')";
$result = mssql_query($query)
        or die('Wystąpił błąd');
printf ("Dodano nowy rekord: %s, %s, %s, %s", $id, $imie, $nazwisko,
$zarobki);
```

4.4 Modyfikacja danych

```
$query = "UPDATE osoby SET imie='$imie', nazwisko='$nazwisko',
zarobki='$zarobki' where id='$id'";
$result = mssql_query($query);
        or die('Wystąpił błąd');
printf ("Dane zostały zmienione: %s %s %s %s", $id, $imie, $nazwisko,
$zarobki);
```

4.5 Usuwanie danych

```
$query = "DELETE FROM osoby where id='$id'";
$result = mssql_query($query);
        or die('Wystąpił błąd');
printf ("Usunięto wiersze");
```

4.6 Zamknięcie połączenia

Opcjonalne zwolnienie pamięci i zamknięcie połączenia:

```
mssql_free_result($result);

mssql_close($connection);
?>
```

5. IBM DB2

Parametry połączenia:

```
<?php
$host = "localhost";
$user = "db2";
$pass = "db2";
$db = "test";
$port = 50000;
```

5.1 Połączenie

Połączenie z bazą danych:

```
$connection = db2_connect($db, $user, $pass);
              or die('Nie można się połączyć');
echo "Poprawnie połączono";
```

lub

```
$conn_string = "DRIVER={IBM DB2 ODBC DRIVER};DATABASE=$db;" .
          "HOSTNAME=$host;PORT=$port;PROTOCOL=TCPIP;UID=$user;PWD=$pass;";
$connection = db2_connect($conn_string, '', '');
```

Persistent connection:

```
$connection = db2_pconnect($db, $user, $pass);
```

Istnieje możliwość dodanie do opcji połączenia tablicy opcji dodatkowych z parametrami: autocommit, DB2_ATTR_CASE czy CURSOR.

5.2 Odczyt danych

W IBM DB2 istnieje funkcja db2_prepare() przygotowująca zapytanie SQL do wykonania przez funkcję db2_execute(). Korzyści takiego rozwiązania:
- wydajność – serwer bazodanowy przygotowuje optymalny plan wykonania zapytania
- bezpieczeństwo – podczas przygotowania zapytania możemy zdefiniować jego parametry, które później są weryfikowane pod względem poprawności przez serwer
- funkcjonalność – możemy zdefiniować parametry wejściowe i wyjściowe dla procedur przy pomocy funkcji db2_bind_param()

Przykład prosty:

```
$query = "SELECT * FROM osoby ORDER BY zarobki DESC";
$result = db2_prepare($connection, $query);
db2_execute($result);

while (db2_fetch_row($result)) {
    $id = db2_result($result, 0);
    $imie = db2_result($result, 'imie');
    $nazwisko = db2_result($result, 'nazwisko');
    $zarobki = db2_result($result, 'zarobki');
    print "$id $imie $nazwisko $zarobki";
}
```

lub

```
if (db2_num_rows($result) != 0) {
   while($myrow = db2_fetch_assoc($result)) {
        echo $myrow['id'];
        echo $myrow['imie'];
        echo $myrow['nazwisko'];
        echo $myrow['zarobki'];
   }
} else {
   echo "Brak wyników wyszukiwania";
}
```

Przykład użycia funkcji db2_fetch_object() - pobranie wyników zapytania jako obiekt.

```
$sql = "SELECT imie, nazwisko, zarobki FROM osoby WHERE id = ?";

$stmt = db2_prepare($connection, $sql);
db2_execute($stmt);

while ($osoba = db2_fetch_object($stmt)) {
        echo "Pracownik {$osoba->imie} {$osoba->nazwisko}
             zarabia {$osoba->zarobki}zł";
    }
```

Definiowanie parametrów przy pomocy funkcji db2_bind_param():

```
$sql = 'SELECT imie, nazwisko, zarobki FROM osoby
    WHERE zarobki > ? AND zarobki < ?';

$stmt = db2_prepare($connection, $sql);

// Możemy zadeklarować zmienne przed wykonaniem db2_bind_param()
$lower_limit = 1000.0;

db2_bind_param($stmt, 1, "lower_limit", DB2_PARAM_IN);
db2_bind_param($stmt, 2, "upper_limit", DB2_PARAM_IN);

// ...lub po
$upper_limit = 3000.0;

if (db2_execute($stmt)) {
    while ($row = db2_fetch_array($stmt)) {
        print "{$row[0]}, {$row[1]}, {$row[2]}\n";
    }
}
```

5.3 Operacje na danych

Przykład tworzenia tabeli z wykorzystaniem funkcji db2_exec() – bezpośrednie wykonanie polecenia SQL.

```
$create = 'CREATE TABLE osoby (id INTEGER, imie VARCHAR(20),
    nazwisko CHAR(20), zarobki DECIMAL(5,2))';
$result = db2_exec($conn, $create);
if ($result) {
    print "Successfully created the table.\n";
}
```

Przykład wykonania polecenia INSERT z możliwością podania parametrów:

```
$osoba = array(0, 'Jan', 'Nowak', 1500.00);
$insert = 'INSERT INTO osoby (id, imie, nazwisko, zarobki)
    VALUES (?, ?, ?, ?)';
$stmt = db2_prepare($connection, $insert);
if ($stmt) {
    $result = db2_execute($stmt, $osoba);
    if ($result) {
        print "Dodano nową osobę";
    }
}
```

5.4 Zamknięcie połączenia

```
db2_close($connection);
?>
```

6. Oracle

Operacje na bazie Oracle 10, Oracle 9, Oracle 8 oraz Oracle 7 wykonuje się za pośrednictwem klienta Oracle Call Interface (OCI).
W systemie Windows potrzebna biblioteka *php_oci8.dll.*

6.1 Połączenie

Połączenie z bazą danych możemy nawiązać na trzy sposoby. Ponieważ łączenie się z bazą Oracle jest kosztowną operacją należy wybrać odpowiedni sposób połączenia w zależności od naszych potrzeb.
oci_pconnect() – otwiera stałe połączenie, które może być wielokrotnie użyte podczas wykonywania wielu skryptów, parametry oci8.max_persistent i oci8.persistent_timeout dodatkowo pomagają w optymalnym ustawieniu ilości i czasu połączeń, zwłaszcza dla wielu użytkowników
oci_connect() – tworzy nowe połączenie, połączenia tworzone przez oci_pconnect() i oci_connect() mogą używać wspólnych istniejących uchwytów połączenia (problemy z izolowanymi transakcajami)
oci_new_connect() – zawsze tworzy nowe połączenie z bazą danych Oracle z nowym uchwytem, niezależnie od istniejących połączeń

Dane do połączenia:

```
<?php
$host = "localhost";
$user = "oracle";
$pass = "oracle";
$db = "test";
```

Proste połączenie:

```
$connection = oci_connect(&user, &passwd, &db);
if (!$connection) {
  $e = oci_error();
  echo 'Nie można się połączyć' . $e['message'];
};
```

Szczegółowe parametry:

```
$dbcon = oci_new_connect('XXXXX', 'XXXXXX',
'
(DESCRIPTION =
            (ADDRESS =
                  (PROTOCOL = TCP)
                  (HOST = XXX.XXX.XXX.XXX)
                  (PORT = 1521)
                  (HASH = XXXXX)
             )
            (CONNECT_DATA =(SID = XXX))
 )
');
```

6.2 Odczyt danych

Podobnie jak w IBM DB2 baza Oracle wstępnie analizuje zapytania SQL i przygotowuje optymalny plan ich wykonania. Służy do tego funkcja oci_parse().

```
$query = "SELECT * FROM oracle.osoby ORDER BY zarobki DESC";
$stmt = oci_parse ($connection, $query);
oci_execute ($statement);

while ($row = oci_fetch_array ($stmt, OCI_ASSOC)) {
    echo $row['id'];
    echo $row['imie'];
    echo $row['nazwisko'];
    echo $row['zarobki'];
}
oci_free_statement($stmt);
```

Zwolnienie zasobów powiązanych z zapytaniem.

6.3 Dodawanie danych

Tworzenie tabeli:

```
$stmt = oci_parse($conn, "create table oracle.osoby (id number,
        imie varchar2(20), nazwisko varchar2(20), zarobki number(5,2))");
oci_execute($stmt);
echo "created table";
```

Usuwanie tabeli:

```
$stmt = oci_parse($conn, "drop table oracle.osoby");
oci_execute($stmt);
echo "dropped table";
oci_free_statement($stmt);
```

6.4 Modyfikacja danych

```
$stmt = oci_parse($conn, "insert into oracle.osoby
            values(1, Jan, Nowak, 1500.00))");
oci_execute($stmt, OCI_DEFAULT);
echo oci_num_rows($stmt) . " rows inserted";
oci_free_statement($stmt);
```

6.5 Usuwanie danych

```
$stmt = oci_parse($conn, "delete from oracle.osoby where id=1");
oci_execute($stmt, OCI_DEFAULT);
echo oci_num_rows($stmt) . " deleted \n";
oci_free_statement($stmt);
```

6.6 Zamknięcie połączenia

```
oci_close($connection);
?>
```