# Curso de PHP

FATEC - Jundiaí




## PHP – Aula 11


### Funções de acesso ao Banco de Dados PostgreSQL

string pg_field_name(resource result, int field_number)

Retorna o nome do campo ocupando o campo de número igual a field_number no recurso (resource) de resultado result. A numeração de campo inicia-se em 0. Recebe um parâmetro relaciona a execução de uma query no banco, e o outro parâmetro é um inteiro referindo-se a coluna pretendida da pesquisa.

```
<?php
   $dbconn = pg_connect("dbname=editora");
   $res = pg_query($dbconn, "select * from autores where autor = 'Rossoni'");
   $i = pg_num_fields($res);
   for ($j = 0; $j < $i; $j++) {
      echo "coluna $j\n";
      $fieldname = pg_field_name($res, $j);
      echo "fieldname: $fieldname\n";
      echo "tamanho ocupado: " . pg_field_prtlen($res, $fieldname) . " caracteres\n";
      echo "tamanho armazenado: " . pg_field_size($res, $j) . " bytes\n";
      echo "field type: ".pg_field_type($res, $j)." \n\n";
   }
?>
```

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

int pg_field_num(resource result, string field_name)

Irá retornar o número da vaga (slot) da coluna (campo) que corresponde a field_name no recurso (resource) de resultado result. A numeração de campo inicia-se em 0. Esta função retornará -1 em caso de erro.

Vamos analisar uma consulta SQL da seguinte forma:
SELECT nome, cep, idade FROM cliente

Será retornado 0 se for passado a string "nome" como parâmetro, retornará 1 se "cep" for passado como parâmetro ou 2 se "idade" tenha sido passado.



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

int pg_field_prtlen(resource result, int row_number, string field_name)

Retorna o comprimento atual impresso (número de caracteres) de um valor especificado em um recurso (resource) de resultado result. A numeração de linha inicia-se em 0. Esta função retornará -1 em caso de erro.

int pg_field_size(resource result, int field_number)

Retorna o tamanho de armazenamento interno (em bytes) do número de campo do recurso (resource) de resultado result. A numeração de campo inicia-se em 0. Um campo de tamanho -1 indica um campo de tamanho variável. Esta função retornará FALSE em caso de erro.

string pg_field_type(resource result, int field_number)

Retorna uma string contendo o nome do tipo do campo de número field_number dado no recurso (resource) de resultado result. A numeração de campo inicia-se em 0.

Obs.: Vimos o exemplo destas duas funções em pg_field_name().

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

bool pg_free_result(resource result)

Precisa ser usada apenas se você está preocupado em usar muita memória enquanto seu script está rodando. Todos os resultados serão liberados da memória automaticamente assim que o script terminar de executar. Mas, se você tem certeza que não precisará mais dos dados do resultado em um script, você pode chamar pg_free_result() com o recurso (resource) de resultado result como argumento e a memória ocupada pelo resultado associado será liberada. Retorna TRUE em caso de sucesso ou FALSE em falhas.

Obs.: Vimos um exemplo desta função em pg_fetch_object().



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

array pg_get_notify(resource connection [, int result_type])

Retorna uma mensagem com as notificações emitidas pelo comando NOTIFY SQL. Para receber as mensagens, o comando LISTEN SQL deve ser emitido. Se existir uma mensagem, será retornada uma array contendo a mensagens o PID do processo. Se nenhuma mensagem for retornada, o comando retornará False.

```
<?php
   $conn = pg_pconnect("dbname=editora");
   // obtém mensagens de outros processos a 'autor_updated'
   pg_query($conn, 'LISTEN autor_updated;');
   $notify = pg_get_notify($conn);
   if (!$notify)
      print("Nenhuma mensagem\n");
   else
      print_r($notify);
?>
```

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

int pg_get_pid(resource connection)

Retorna o número do processo do usuário na base de dados.

```
<?php
   $conn = pg_pconnect("dbname=editora");
   if (!$conn) {
      echo "Um erro ocorreu";
      exit;
   }
   // Processo Backend PID. Use PID com pg_get_notify()
   $pid = pg_get_pid($conn);
?>
```



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

resource pg_get_result([resource connection])

Retorna o recurso (resource) de resultado de uma consulta (query) executada por pg_send_query(). pg_send_query() pode enviar múltiplas consultas (queries) ao servidor PostgreSQL e pg_get_result() é usada para carregar os resultados das consultas, um por um.

```
<?php
   $dbconn = pg_connect("dbname=editora");
   if (!pg_connection_busy($dbconn)) {
      pg_send_query($dbconn,"select * from autores; select count(*) from autores;");
   }
   $res1 = pg_get_result($dbconn); echo "Primeira chamada a pg_get_result(): $res1\n";
   $rows1 = pg_num_rows($res1); echo "$res1 tem $rows1 registros\n\n";
   $res2 = pg_get_result($dbconn); echo "Segunda chamada a pg_get_result(): $res2\n";
   $rows2 = pg_num_rows($res2); echo "$res2 tem $rows2 registros\n\n";
?>
```

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

string pg_host(resource connection)

Retorna o nome da máquina com a qual o recurso (resource) de conexão connection está conectado.

```
<?php
   $conn = pg_pconnect("dbname=editora");
   if (!$conn) {
      echo "Um erro ocorreu.\n";
      exit;
   }

   echo "O servidor de banco de dados está em : " . pg_host($conn);
?>
```



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

bool pg_insert(resource connection, string table_name, array assoc_array [, int options])

Insere os valores de assoc_array na tabela especificada por table_name. table_name deve ter no mínimo tantas colunas quanto forem os elementos em assoc_array. Os nomes dos campos assim como os valores em table_name devem ser iguais aos índices e valores de assoc_array. Retorna TRUE em caso de sucesso ou FALSE em falhas. Se o parâmetro options for especificado, pg_insert() é aplicado em assoc_array com a opção especificada.

```
<?php
   $db = pg_connect ('dbname=meubanco');
   $res = pg_insert($db, 'post_log', $_POST);
   if ($res) { echo "Dados POST arquivados com sucesso\n";  }
   else { echo "O usuário deve ter inserido entradas inválidas\n";  }
?>
```

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

string pg_last_error(resource connection)

Retorna a última mensagem de erro para a conexão representada por connection. As mensagens de erro podem ser sobrescritas por chamadas internas ao PostgreSQL(libpq). Se múltiplos erros ocorrerem dentro de um módulo de função do PostgreSQL, pode não retornar a mensagem de erro correta.

string pg_last_notice(resource connection)

Retorna a última notificação do servidor PostgreSQL especificada por connection. O servidor PostgreSQL envia notificações em diversas situações, por exemplo, se as transações não puderem continuar. Com pg_last_notice() você pode evitar a chamada de consultas (queries) inúteis, checando quando a notificação está relacionada ou não com a transação.



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

int pg_last_oid(resource result)

É usado para recuperar o oid designado a uma linha (registro) se o recurso (resource) de resultado é usado a partir do último comando enviado através de pq_query() se este comando era um INSERT do SQL. Retorna um inteiro positivo se havia um oid válido. Retorna FALSE se um erro ocorrer ou o último comando enviado através de pg_query() não foi INSERT ou se o INSERT falhou. O campo OID tornou-se opcional a partir do PostgreSQL 7.2. Quando um campo OID não é definido em uma tabela, o programador deve usar pg_result_status() para checar se o registro foi inserido com sucesso ou não.

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

Para usar a interface de objetos grandes (lo), é necessário encapsulá-lo em um bloco de transação.

bool pg_lo_close(resource large_object)

Fecha um Objeto Grande (Large Object, daí vem o "lo" que integra o nome da função). large_object é um recurso (resource) para o objeto grande gerado a partir de pg_lo_open().

int pg_lo_create(resource connection)

Cria um Objeto Grande (Large Object) e retorna o seu oid. connection especifica uma conexão a um banco de dados válida aberta por pg_connect() ou pg_pconnect(). Os modos de acesso INV_READ, INV_WRITE e INV_ARCHIVE não são suportados, o objeto é criado sempre com acesso a leitura e escrita. INV_ARCHIVE foi removido do próprio PostgreSQL (a partir da 6.3). Retorna o oid do objeto. Retorna FALSE se um erro ocorrer.



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

bool pg_lo_export(int oid, string pathname [, resource connection])

O argumento oid especifica o oid do objeto grande (large object) a exportar e o argumento pathname especifica o caminho até o arquivo. Retorna FALSE se um erro ocorrer, caso contrário retorna TRUE.

int pg_lo_import([resource connection, string pathname])

O argumento pathname especifica o caminho do arquivo a ser importado como um objeto grande (large object). Retorna FALSE se um erro ocorrer, caso contrário, retorna o oid do objeto recém criado.

resource pg_lo_open(resource connection, int oid, string mode)

Abre um Objeto Grande (Large Object em inglês, daí o "lo"). O recurso (resource) encapsula informações sobre a conexão. oid especifica um oid de objeto grande válido e o parâmetro mode pode ser "r", "w" ou "rw". A função retorna FALSE se houver algum erro.

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

int pg_lo_read_all(resource large_object)

Lê um objeto grande (large object) e passa-o diretamente para o navegador depois de enviar todos os cabeçalhos pendentes. A intenção principal é enviar dados binários como imagens ou som. Retorna o número de bytes lidos ou FALSE se ocorrer algum erro.

string pg_lo_read(resource large_object, int len)

Lê o número de bytes equivalente ao valor de len de um objeto grande (large object) e retorna-o como uma string. large_object especifica um recurso (resource) válido de objeto e len especifica o tamanho máximo permitido do segmento do objeto grande. Retorna FALSE se algum erro acontecer.



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

bool pg_lo_seek(resource large_object, int offset [, int whence])

Procura uma posição em um recurso (resource) de um objeto grande (large objet). whence pode ser PGSQL_SEEK_SET, PGSQL_SEEK_CUR ou PGSQL_SEEK_END. Lembrando que a linguagem PHP traz grande semelhança com o padrão C ANSI, as constantes definidas na linha de cima seguem também o mesmo padrão.

PGSQL_SEEK_SET - moverá o cursor no valor de offset a partir do início do objeto.

PGSQL_SEEK_CUR - moverá o cursor no valor de offset a partir da posição atual cursor no objeto.

PGSQL_SEEK_END - moverá o cursor no valor de offset a partir do fim do objeto.

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

int pg_lo_tell(resource large_object)

Retorna a posição atual (deslocamento a partir do início do objeto grande).

bool pg_lo_unlink(resource connection, int oid)

Remove um objeto grande (large object) com um determinado oid. Retorna TRUE em caso de sucesso ou FALSE em falhas.

int pg_lo_write(resource large_object, string data)

Escreve em um objeto grande (large object) a partir de uma variável data e retorna o número de bytes escritos, ou FALSE em caso de erro. large_object é um recurso (resource) de objeto grande criado a partir de pg_lo_open().



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

array pg_metadata(resource connection, string table_name)

Retorna a definição da tabela com o nome igual ao valor de table_name como um array. Se houver algum erro, retorna FALSE.

```
<?php
   $dbconn = pg_connect("dbname=editora");
   $meta = pg_meta_data($dbconn,'autores');
   if (is_array ($meta)) {
      echo '<pre>';
      var_dump ($meta);
      echo '</pre>';
   }
?>
```

Resultado ->

```
array(3) {
  ["autor"]=>
  array(5) {
    ["num"]=>
    int(1)
    ["type"]=>
    string(7) "varchar"
    ["len"]=>
    int(-1)
    ["not null"]=>
    bool(false)
    ["has default"]=>
    bool(false)
  }
  ["ano"]=>
  array(5) {
    ["num"]=>
    int(2)
    ["type"]=>
    string(4) "int2"
    ["len"]=>
    int(2)
    ["not null"]=>
    bool(false)
    ["has default"]=>
    bool(false)
  }
  ["titulo"]=>
  array(5) {
    ["num"]=>
    int(3)
    ["type"]=>
    string(7) "varchar"
    ["len"]=>
    int(-1)
    ["not null"]=>
    bool(false)
    ["has default"]=>
    bool(false)
  }
}
```

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

int pg_num_fields(resource result)

Retorna o número de campos (colunas) do recurso (resource) de resultado result. O argumento é um recurso (resource) de resultado criado a partir de pg_query(). Esta função irá retornar -1 em caso de erro.
Podemos ver em uma consulta SQL da seguinte forma:
SELECT nome, cep, idade FROM cliente
será retornado o valor 3, pois há três campos na consulta.

int pg_num_rows(resource result)

Irá retornar o número de linhas do recurso de resultado result. result é um recurso (resource) de resultado de consulta (query) feito por pg_query(). Esta função retornará -1 em caso de erro. O único parâmetro utilizado nesta função é vindo do retorno de uma função pg_query().



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

string pg_options(resource result)

Retornará uma string contendo as opções especificadas no recurso (resource) de conexão PostgreSQL connection.

```
<?php
   $dbconn = pg_connect("dbname=editora");
   echo pg_options($dbconn); // Escreverá "dbname=editora"
?>
```

int pg_pconnect(string connection_string)

Abre uma conexão persistente no banco de dados PostgreSQL. Retorna um recurso (resource) de conexão que é necessário para outras funções PostgreSQL. Utilizada da mesma forma que pg_connect(). Para habilitar uma conexão persistente, a diretiva pgsql.allow_persistent do php.ini deve ser definida como "On" (que é o padrão). O número máximo de conexões persistentes pode ser definida com a diretiva pgsql.max_persistent do php.ini (o padrão é -1 para sem limite).

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

bool pg_ping(resource connection)

Faz um ping na conexão com o banco de dados, tenta reconectar se a conexão foi quebrada. Retorna TRUE se a conexão está ativa, do contrário, retorna FALSE.

```
<?php
   $conn = pg_pconnect ("dbname=publicacao");
   if (!$conn) {
      echo "Ocorreu um erro.\n";
      exit;
   }

   if (!pg_ping($conn))
      die("Conexão quebrada\n");
?>
```



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

int pg_port(resource connection)

Retorna o número da porta a qual o recurso de conexão PostgreSQL connection está conectado.

```
<?php
   $dbconn = pg_connect("dbname=editora");
   echo pg_port($dbconn);
   // Escreverá "5432", pois a porta padrão do PostgreSQL é a 5432.
?>
```

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

bool pg_put_line([resource connection, string data])

Envia uma string terminada em NULL para o servidor backend PostgreSQL. Isso é útil, por exemplo, para a inserção de dados em uma tabela em alta velocidade, iniciada através de uma operação de cópia PostgreSQL. O caractere NULL final é adicionado automaticamente. Retorna TRUE em caso de sucesso ou FALSE em falhas. A aplicação deve enviar os dois caracteres "\." explicitamente na última linha para indicar ao backend que ela terminou de enviar seus dados.

```
<?php
    $conn = pg_pconnect("nomebd=meubanco");
    pg_query($conn, "create table bar (a int4, b char(16), d float8)");
    pg_query($conn, "copy bar from stdin");
    pg_put_line($conn, "3\tola mundo\t4.5\n");
    pg_put_line($conn, "4\ttchau mundo\t7.11\n");
    pg_put_line($conn, "\\.\n");
    pg_end_copy($conn);
?>
```



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

resource pg_query(resource connection, string query)

Retorna um recurso (resource) de resultado da consulta (query) se a consulta pôde ser executada. Retorna FALSE em caso de falha ou se a conexão não é uma conexão válida. Detalhes sobre os erros podem ser recuperados usando a função pg_last_error() se a conexão é válida.

pg_query() envia uma declaração SQL para o banco de dados PostgreSQL especificado pelo recurso de conexão connection. connection deve ser uma conexão válida que foi criado por pg_connect().

O valor de retorno dessa função é um recurso (resource) de resultado de consulta (query) para ser usado para acessar os resultados de outras funções PostgreSQL como pg_fetch_array().

Prof. Cláudio Farias Rossoni

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

string pg_result_error(resource result)

Retorna a mensagem de erro associada ao recurso (resource) de resultado result. Deste modo, o usuário tem melhores chances de ter uma mensagem de erro melhor que a retornada por pg_last_error().

array pg_result_seek(resource result, int offset)

Altera a posição do ponteiro interno de um recurso (resource) de resultado. Retorna FALSE em caso de erro.

int pg_result_status(resource result)

Retorna o status do recurso (resource) de resultado. Os valores de retorno possíveis são: PGSQL_EMPTY_QUERY, PGSQL_COMMAND_OK, PGSQL_TUPLES_OK, PGSQL_COPY_TO, PGSQL_COPY_FROM, PGSQL_BAD_RESPONSE, PGSQL_NONFATAL_ERROR e PGSQL_FATAL_ERROR.



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

array pg_select(resource connection, string table_name, array assoc_array [, int options])

Seleciona registros especificados por assoc_array, que contém pares do tipo campo=>valor. Para uma consulta (query) válida, retorna um array que contém todos os registros e campos que combinam com a condição especificada por assoc_array. Se options for especificado, pg_convert() será aplicada à assoc_array com as opções especificadas.

```
<?php
$db = pg_connect ('nomebd=meubanco');
$rec = pg_select($db, 'post_log', $_POST);
if ($rec) {
   echo "Registros selecionados\n";
   var_dump($rec);
}
else { echo "O usuário deve ter inserido dados errados\n"; }
?>
```

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

bool pg_send_query(resource connection, string query)

Envia uma consulta (query) assíncrona para connection. Diferente de pg_query(), ela pode enviar consultas múltiplas para o PostgreSQL e carregar os resultados, um por um, usando pg_get_result(). A execução do script não é bloqueada enquanto as consultas estão sendo executadas. Use pg_connection_busy() para checar se a conexão está ocupada. (por exemplo, se uma consulta está sendo executada). A consulta pode ser cancelada chamando pg_cancel_query(). Apesar de ser possível enviar múltiplas consultas de uma vez, você não pode enviar múltiplas consultas para uma conexão ocupada. Se a consulta é enviada enquanto a conexão está ocupada ela espera até que a última consulta seja finalizada e descarta todos os resultados.

Obs.: vimos o exemplo na função pg_get_result().



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

int pg_set_client_encoding([resource connection, string encoding])

Define a codificação do cliente e retorna 0 caso haja sucesso e -1 se houver erro.
encoding é a codificação do cliente e pode ter os valores: SQL_ASCII, EUC_JP, EUC_CN, EUC_KR, EUC_TW, UNICODE, MULE_INTERNAL, LATINX (X=1...9), KOI8, WIN, ALT, SJIS, BIG5, WIN1250. As codificações disponíveis dependem da versão do PostgreSQL e libpq. Vide o manual PostgreSQL para saber das codificações disponíveis para o seu PostgreSQL.

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

bool pg_trace(string pathname [, string mode [, resource connection]])

Habilita o rastreamento da comunicação frontend/backend do PostgreSQL para um arquivo de depuração especificado pelo parâmetro pathname. Para entender completamente estes resultados, você deve estar familiarizado com o protocolo de comunicação interno do PostgreSQL. Para aqueles que não estão, isso ainda pode ser útil para rastrear erros em consultas (queries) enviadas ao servidor, você poderia fazer por exemplo grep '^Para backend' rastro.log.

*pathname* e *mode* são os mesmos que na função **fopen()** (o *mode* padrão é 'w'), *connection* especifica a conexão a ser rastreada e seu padrão é a última conexão aberta.

Retorna **TRUE** se o *pathname* pode ser aberto para escrita, **FALSE** caso contrário.



# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

string pg_tty(resource connection)

Retorna o nome da tty que a saída do lado do servidor é enviada no recurso (resource) de conexão connection. Está função já é considerada depreciada.

string pg_unescape_bytea(string data)

Faz uma versão binária da string do tipo bytea. Retorna a string em binário(binary).

bool pg_untrace([resource connection])

Pára o rastreamento iniciado por pg_trace(). connection especifica a conexão que está sendo rastreada e seu padrão é a última conexão aberta.

# PHP – Aula 11

## Funções de acesso ao Banco de Dados PostgreSQL

bool pg_update(resource connection, string table_name, array data, array condition [, int options])

Atualiza registros que combinam com a condição especificada pelo argumento condition com os dados do parâmetro data. Se options for especificado, pg_convert() será aplicada a data com as opções especificadas.

```
<?php
   $bd = pg_connect('dbname=meubanco');
   $dados = array('campo1'=>'AA', 'campo2'=>'BB');
   $res = pg_update($bd, 'post_log', $_POST, $dados);
   if ($res) {
      echo "Dados atualizados: $res\n";
   }
   else { echo "Usuário deve ter enviado entradas inválidas\n"; }
?>
```



# PHP – Aula 11

## Configurações durante a execução

O comportamento dessas funções podem ser modificado pelas configurações do php.ini.

| | Nome | Padrão | Alterável | Old |
|---|---|---|---|---|
| 1- | pgsql.allow_persistent | "On" | PHP_INI_SYSTEM | 1 |
| 2- | pgsql.max_persistent | "Unlimited" | PHP_INI_SYSTEM | -1 |
| 3- | pgsql.max_links | "Unlimited" | PHP_INI_SYSTEM | -1 |
| 4- | pgsql.auto_reset_persistent | "Off" | PHP_INI_SYSTEM | 0 |
| 5- | pgsql.ignore_notice | "Off" | PHP_INI_ALL | 0 |
| 6- | pgsql.log_notice | "Off" | PHP_INI_ALL | 0 |

1) Se quer ou não permitir conexões persistentes com o PostgreSQL.
2) O número máximo de conexões persistentes com PostgreSQL por processo.
3) O número máximo de conexões PostgreSQL por processo, incluindo as persistentes.
4) Detecta e automaticamente tenta reiniciar uma conexão persistente que foi quebrada, porém, gera uma pequena sobrecarga no desempenho.
5) Ignorar ou não os avisos do PostgreSQL.
6) Quando realizar ou não o log de avisos do PostgreSQL. A diretiva pgsql.ignore_notice deve estar em off para poder fazer o log dos avisos.

# PHP – Aula 11

## Novos nomes de Funções

Os nomes das funções PostgreSQL serão alterados na versão 4.2.0 para confirmar os padrões de programação atuais.

A maioria dos novos nomes terão sublinhados (underscore) adicionais, por exemplo, pg_lo_open(). Algumas funções foram renomeadas para uma maior consistência, por exemplo, pg_exec() mudou para pg_query().

Os nomes antigos podem ser usados na 4.2.0 e em algumas poucas outras versões após esta, mas eles serão removidos futuramente.

Obs.: próximo slide apresento uma tabela contendo as funções que foram alteradas.



# PHP – Aula 11

## Tabela – Alteração - Nomes das Funções

| Anterior | Nova |
|---|---|
| pg_exec() | pg_query() |
| pg_getlastoid() | pg_last_oid() |
| pg_cmdtuples() | pg_affected_rows() |
| pg_numrows() | pg_num_rows() |
| pg_numfields() | pg_num_fields() |
| pg_fieldname() | pg_field_name() |
| pg_fieldsize() | pg_field_size() |
| pg_fieldnum() | pg_field_num() |
| pg_fieldprtlen() | pg_field_prtlen() |
| pg_fieldisnull() | pg_field_is_null() |

| Anterior | Nova |
|---|---|
| pg_freeresult() | pg_free_result() |
| pg_result() | pg_fetch_result() |
| pg_loreadall() | pg_lo_read_all() |
| pg_locreate() | pg_lo_create() |
| pg_lounlink() | pg_lo_unlink() |
| pg_loopen() | pg_lo_unlink() |
| pg_loclose() | pg_lo_close() |
| pg_loread() | pg_lo_read() |
| pg_lowrite() | pg_lo_write() |
| pg_loexport() | pg_lo_export() |
| pg_loimport() | pg_lo_import() |

# PHP – Aula 11

## Novos nomes e atributos das Funções

Exemplo: pg_connect

Uma conexão com o servidor PostgreSQL pode ser estabelecida com os seguintes pares de valores definidos na string de comando: **$conn = pg_connect("host=seuHost port=suaPorta tty=seuTTY options=suasOpcoes dbname=seuDB user=seuUsuario password=suaSenha");**

A sintaxe anterior: **$conn = pg_connect ("host", "porta", "opcoes", "tty", "nomebd")** ficará obsoleta.



# PHP – Aula 11

## Large Objects (Objetos Grandes)

A partir do PostgreSQL 7.1.0, você pode armazenar até 1GB dentro de um campo do tipo texto. Em versões mais antigas, isto era limitado ao tamanho do bloco (o padrão era 8KB e o máximo era 32KB, definido em tempo de compilação).

Para usar a interface de objetos grandes (large objects), é exigido que se encapsule as funções de objetos grandes em um bloco de transação. Um bloco de transação inicia-se com a declaração SQL **BEGIN** e, se a transação for válida, termina com **COMMIT** ou **END**. Se a transação falhar, ela deve ser fechada com **ROLLBACK** ou **ABORT**.

```
<?php
  $database = pg_connect ("dbname=editora");
  pg_query ($database, "begin");
  $oid = pg_lo_create ($database);
  $handle = pg_lo_open ($database, $oid, "w");
  pg_lo_write ($handle, "large object data");
  pg_lo_close ($handle);
  pg_query ($database, "commit");
?>
```

**Obs.: Você não deve fechar a conexão com o servidor PostgreSQL antes de fechar o objeto grande.**

# PHP – Aula 11