Proposta de calendário - Comissão Permanente de Ensino

1º semestre	
> Início > Continuidade do 1º semestre, em reposição ao período de paralisação das atividades (17/06 a 21/08/2016) > Término > Reposição do período de aproveitamento	07/03/2016 a partir de 29/08 04/10/2016 05 a 08/10/16
> **Regime de Recuperação** (o aluno será automaticamente matriculado após a consolidação da disciplina normal, desde que atenda os requisitos exigidos).	**11 a 15/10/16**
> **Dias letivos: 100** > Dias para reposição: 02 (que podem ser aproveitados como dias letivos)	

Cronograma para encerramento da disciplina

Conteúdo	Data
Manipulações de BD com PDO - PHP Data Object	29-30/08
jQuery	05-06/09
Ajax	12-13/09
Apresentação do Projeto	19/09
Prova 2	20/09
Recuperação	27/09

unesp

Desenvolvimento de Aplicações para Internet

Aula 9

Celso Olivete Júnior

olivete@fct.unesp.br

Na aula de hoje

PDO

PHP Data Object

PDO

- PDO é uma extensão que fornece uma interface padronizada para trabalhar com bancos de dados, cuja finalidade é abstrair a conexão e interações com os bancos de dados
  - independente do BD que estiver sendo utilizado os métodos executados serão os mesmos
- Usar o PDO não significa que seu sistema será portável entre diversos BD

  → por mais que o uso do PDO facilite a portabilidade, esta interface significa apenas que você se comunicará com qualquer BD através de um determinado conjunto de métodos e classes.

PDO

- O PDO fornece uma camada de abstração de acesso a dados
  - independentemente do banco de dados, as mesmas funções para manipulação podem ser utilizadas.
    - insert, delete, update e select
  - Exemplo: se o banco de dados for alterado de MySQL para PostgreSQL não será necessário alterar todas as funções de consultas mysql_query() para pg_query() no seu código fonte PHP. Esta é uma das principais vantagens de usar PDO.

PDO

- Vantagens de utilizar **PDO**
  - Abstração de conexão e interação com banco de dados
  - Segurança
  - Suporte a diversos drivers

PDO

- Drivers suportados pelo PDO
  - CUBRID (PDO)
  - MS SQL Server (PDO)
  - Firebird (PDO)
  - IBM (PDO)
  - Informix (PDO)
  - MySQL (PDO)
  - MS SQL Server (PDO)
  - Oracle (PDO)
  - ODBC and DB2 (PDO)
  - PostgreSQL (PDO)
  - SQLite (PDO)
  - 4D (PDO)

PDO - MySQL

❑ Manipulando conexões

```
try
{
    //instancia o objeto PDO, conectando com o banco mysql
  $conexao = new PDO('mysql:host=localhost;dbname=nomeBD', 'usuario', 'senha');

}
catch (PDOException $i)
{
        //se houver exceção, exibe
        print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

PDO – interagindo com o banco de dados

- Métodos para interagir com o banco de dados

Método	Retorno	Objetivo
execute	int	Utilizado para insert, update e delete.
query	PDOStatement	Utilizado para resultados tabulares, comando select.
prepare	PDOStatement	Cria um prepared statement, utilizado para dados variáveis.

PDO - prepared statement

- Os ***prepared statements*** oferecem dois ótimos benefícios:
  - A query só precisa ser preparada uma vez, mas pode ser executada várias vezes.
  - Os parâmetros não precisam ser escapados, pois o driver cuida disso automaticamente.
- Esses benefícios significam duas coisas: agilidade e segurança. Exemplo ➔

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=nomeBD', 'usuario', 'senha');

   //prepara a instrução de insert - define dois parametros
   $sql = $conn->prepare('INSERT INTO categorias (cat_nome, cat_descricao) VALUES (?,?)');
   //setando os parâmetros
   $nome_enviado = 'nome da categoria';
   $desc_enviada = 'descrição da categoria';
   //atribui valor ao primeiro parametro
   $sql->bindValue(1,$nome_enviado);
   //atribui valor ao segundo parametro
   $sql->bindValue(2,$nome_enviado);
   $sql->execute();
   echo "Dados inseridos...";
//fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

Inserindo dados...

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=nomeBD', 'usuario', 'senha');

   //prepara a instrução de insert - define dois parametros
   $sql = $conn->prepare('INSERT INTO categorias (cat_nome, cat_descricao) VALUES (?,?)');
   //setando os parâmetros
   $nome_enviado = 'nome da categoria';
   $desc_enviada = 'descrição da categoria';
   //atribui valor ao primeiro parametro
   $sql->bindValue(1,$nome_enviado);
   //atribui valor ao segundo parametro
   $sql->bindValue(2,$nome_enviado);
   $sql->execute();
   echo "Dados inseridos...";
//fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

Cria o prepared statement com 2 parâmetros

Parâmetros são definidos com **?** No caso de mais de um parâmetro, separá-los com vírgula

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=nomeBD', 'usuario', 'senha');

   //prepara a instrução de insert - define dois parametros
   $sql = $conn->prepare('INSERT INTO categorias (cat_nome, cat_descricao) VALUES (?,?)');
   //setando os parâmetros
   $nome_enviado = 'nome da categoria';
   $desc_enviada = 'descrição da categoria';
   //atribui valor ao primeiro parâmetro
   $sql->bindValue(1,$nome_enviado);
   //atribui valor ao segundo parâmetro
   $sql->bindValue(2,$nome_enviado);
   $sql->execute();
   echo "Dados inseridos...";
//fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

Cria duas variáveis que serão passadas para o prepared statement

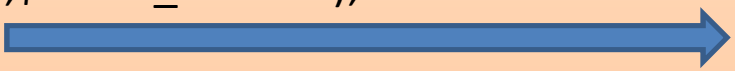
```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=nomeBD', 'usuario', 'senha');

   //prepara a instrução de insert - define dois parametros
   $sql = $conn->prepare('INSERT INTO categorias (cat_nome, cat_descricao) VALUES (?,?)');
   //setando os parâmetros
   $nome_enviado = 'nome da categoria';
   $desc_enviada = 'descrição da categoria';
   //atribui valor ao primeiro parâmetro
   $sql->bindValue(1,$nome_enviado);
   //atribui valor ao segundo parâmetro
   $sql->bindValue(2,$nome_enviado);
   $sql->execute();
   echo "Dados inseridos...";
//fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

bindValue → informa os valores para compor a query

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=nomeBD', 'usuario', 'senha');

   //prepara a instrução de insert - define dois parametros
   $sql = $conn->prepare('INSERT INTO categorias (cat_nome, cat_descricao) VALUES (?,?)');
   //setando os parâmetros
   $nome_enviado = 'nome da categoria';
   $desc_enviada = 'descrição da categoria';
   //atribui valor ao primeiro parâmetro
   $sql->bindValue(1,$nome_enviado);
   //atribui valor ao segundo parâmetro
   $sql->bindValue(2,$nome_enviado);
   $sql->execute();
   echo "Dados inseridos...";
//fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```



executa o prepared statement

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=nomeBD', 'usuario', 'senha');

   //prepara a instrução de insert - define dois parametros
   $sql = $conn->prepare('INSERT INTO categorias (cat_nome, cat_descricao) VALUES (?,?)');
   //setando os parâmetros
   $nome_enviado = 'nome da categoria';
   $desc_enviada = 'descrição da categoria';
   //atribui valor ao primeiro parâmetro
   $sql->bindValue(1,$nome_enviado);
   //atribui valor ao segundo parâmetro
   $sql->bindValue(2,$nome_enviado);
   $sql->execute();
   echo "Dados inseridos...";
//fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

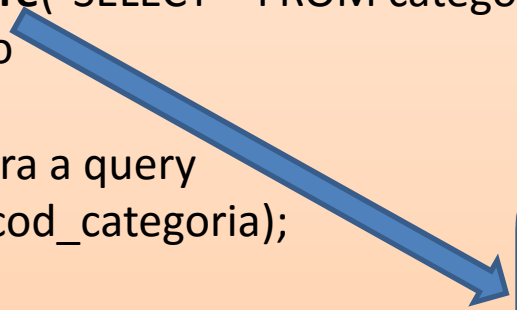
Fecha a conexão

PDO – comando select

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
    $conn = new PDO('mysql:host=localhost;dbname=oo', 'root', '') or die ("Erro na conexão");
    //cria a instrução de consulta com passagem de parâmetro
    $query = $conn->prepare("SELECT * FROM categorias where cat_id=?");
    //configura o parâmetro
    $cod_categoria = 4;
    //passa o parâmetro para a query
    $query->bindValue(1,$cod_categoria);
    $query->execute();
    while($row = $query->fetch()) {
        echo $row[0]." - " . $row[1]."<br>";
    }
    //fecha a conexão
    $conn = null;
}
catch (PDOException $i)
{
                //se houver exceção, exibe
                print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

Exibindo dados...

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=oo', 'root', '') or die ("Erro na conexão");
   //cria a instrução de consulta com passagem de parâmetro
   $query = $conn->prepare("SELECT * FROM categorias where cat_id=?");
   //configura o parâmetro
   $cod_categoria = 4;
   //passa o parâmetro para a query
   $query->bindValue(1,$cod_categoria);
   $query->execute();
   while($row = $query->fetch()) {
       echo $row[0]." - " . $row[1]."<br>";
   }
   //fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```



Cria o prepared statement com 1 parâmetro

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=oo', 'root', '') or die ("Erro na conexão");
   //cria a instrução de consulta com passagem de parâmetro
   $query = $conn->prepare("SELECT * FROM categorias where cat_id=?");
   //configura o parâmetro
   $cod_categoria = 4;
   //passa o parâmetro para a query
   $query->bindValue(1,$cod_categoria);
   $query->execute();
   while($row = $query->fetch()) {
       echo $row[0]." - " . $row[1]."<br>";
   }
   //fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

Cria uma variável que será passada para o prepared statement

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=oo', 'root', '') or die ("Erro na conexão");
   //cria a instrução de consulta com passagem de parâmetro
   $query = $conn->prepare("SELECT * FROM categorias where cat_id=?");
   //configura o parâmetro
   $cod_categoria = 4;
   //passa o parâmetro para a query
   $query->bindValue(1,$cod_categoria);
   $query->execute();
   while($row = $query->fetch()) {
       echo $row[0]." - " . $row[1]."<br>";
   }
   //fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

bindValue → informa o valor para compor a query

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=oo', 'root', '') or die ("Erro na conexão");
   //cria a instrução de consulta com passagem de parâmetro
   $query = $conn->prepare("SELECT * FROM categorias where cat_id=?");
   //configura o parâmetro
   $cod_categoria = 4;
   //passa o parâmetro para a query
   $query->bindValue(1,$cod_categoria);
   $query->execute();
   while($row = $query->fetch()) {
       echo $row[0]." - " . $row[1]."<br>";
   }
   //fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

executa o prepared statement

```
try
{
   //instancia o objeto PDO, conectando com o banco mysql
   $conn = new PDO('mysql:host=localhost;dbname=oo', 'root', '') or die ("Erro na conexão");
   //cria a instrução de consulta com passagem de parâmetro
   $query = $conn->prepare("SELECT * FROM categorias where cat_id=?");
   //configura o parâmetro
   $cod_categoria = 4;
   //passa o parâmetro para a query
   $query->bindValue(1,$cod_categoria);
   $query->execute();
   while($row = $query->fetch()) {
       echo $row[0]." - " . $row[1]."<br>";
   }
   //fecha a conexão
   $conn = null;
}
catch (PDOException $i)
{
            //se houver exceção, exibe
            print "Erro: <code>" . $i->getMessage() . "</code>";
}
```

fetch() Percorre a linha da query e imprime os resultados *(while só é necessário no caso da consulta ter retornado mais de um valor*)

PDO – comando select

- Outras alternativas para resgatar resultados de um comando ***select***

Método	Objetivo
fetch()	Retorna a próxima linha do resultado.
fetchAll()	Retorna um array com todos os resultados.
fetchObject()	Retorna a próxima linha do resultado como objeto.
fetchColumn()	Retorna uma coluna da próxima linha do resultado.

Projeto Final

- Desenvolver um Sistema Web para Vendas (a definir)
  - Deverá conter:
    - HTML 5, CSS, jQuery, Ajax, MySql (PDO)
  - Funcionalidades:
    - Cadastros (pelo menos 3)
    - Consulta ao Estoque
    - Vendas
    - ...
  - Tipos de acesso: administrador e cliente
  - Em duplas
    - Entrega e apresentação em 19/09