www.coldfusionbrasil.com.br

Entendendo mais sobre CFOUTPUT

A tag CFOUTPUT é utilizada para mostrar resultados de pesquisas a banco de dados e o resultado de operações com variáveis.

No módulo 1 de nosso tutorial, colocamos uma pequena introdução do funcionamento da tag CFOUTPUT, que foi utilizada para exibir o resultado de uma pesquisa ao banco de dados no browser.

Nesta introdução, a tag CFOUTPUT foi seguida do atributo *QUERY*, que é opcional e indica o nome de qual pesquisa você pretende exibir os dados.

Mas a tag CFOUTPUT têm mais outros atributos. São eles:

GROUP	Opcional. Este atributo não exibe registros duplicados de acordo com o campo ordenado na query. Este parâmetro é caso sensitivo.
GROUPCASESENSITIVE	Opcional. Este campo indica se o campo da opção *GROUP* será ou não caso sensitivo. Seu padrão é YES.
STARTROW	Opcional. Especifica a linha que você quer iniciar a exibir os dados.
MAXROWS	Opcional. Especifica o número máximo de linhas que você quer exibir.

Tendo como base o datasource *agenda,* definido no módulo 1 de nosso tutorial, veremos agora como utilizar estes novos parâmetros da tag CFOUTPUT.

Uso do parâmetro GROUP

Insira mais dois registros em nosso banco de dados:

Nome	Tel
André Soares	11-234-9980
josé da silva	13-561-7898

**Note que josé da silva está em letras minúsculas e que o telefone de André Gomes é o mesmo de José da Silva, usado no módulo anterior.*

Agora iremos criar o acesso a base de dados com uma característica a mais, muito importante para a utilização do parâmetro *GROUP*:

```
<CFQUERY DATASOURCE=”agenda” NAME=”AgendaPessoal”>
  SELECT * FROM Pessoal
  ORDER BY tel
</CFQUERY>
```

Olhando acima, você irá notar a cláusula *ORDER BY* que faz com que seus registros sejam ordenados através do campo *tel*. Criado o acesso a sua base de dados, veremos agora como usar o parâmetro *GROUP*:

```
<CFOUTPUT QUERY=”AgendaPessoal” GROUP=”tel”>
#nome# - #telefone#
<HR>
</CFOUTPUT>
```

Notaremos que como resposta teremos apenas quatro registros, sendo que no banco de dados existem cinco. O registro não exibido foi o de José da Silva, que tem o mesmo número de telefone que André Soares, este que foi exibido por ser a última ocorrência encontrada no banco de dados.

Uso do parâmetro GROUPCASESENSITIVE

Continuando com a mesma base de dados, iremos ver agora o parâmetro GROUPCASESENSITIVE. Para isso precisaremos mudar na cláusula *ORDER BY*, o nome do campo para *nome*, ficando assim:

ORDER BY nome

* Note que estamos fazendo esta alteração, pois o parâmetro GROUPCASESENSITIVE só é empregado em campos string.

Agora coloque a tag CFOUTPUT da seguinte forma:

```
<CFOUTPUT QUERY=”AgendaPessoal” GROUP=”nome”
GROUPCASESENSITIVE=”yes”>
#nome# - #telefone#
<HR>
</CFOUTPUT>
```

Veremos que como resposta teremos os cinco registros do banco de dados.

Mude agora o parâmetro *GROUPCASESENSITIVE* para *NO*. Você terá como resposta apenas quatro registros, sendo o registro de José da Silva não sendo mostrado.

Uso do parâmetro STARTROW

Este parâmetro é usado para indicar em qual registro de nossa query desejamos começar a exibir. Para ilustrar melhor, façamos da seguinte forma:

```
<CFOUTPUT QUERY=”AgendaPessoal” STARTROW=”2”>
#nome# - #telefone#
<HR>
</CFOUTPUT>
```

Veremos que temos como resposta quatro registros, já que iniciamos no registro de número 2 de nossa query.

Uso do parâmetro MAXROWS

Usado para limitar o número máximo de registros da query a serem exibidos.

Vamos acrescentar ao exemplo do parâmetro *STARTROW* o parâmetro MAXROWS, e ver como este novo parâmetro funciona:

```
<CFOUTPUT QUERY=”AgendaPessoal” STARTROW=”2” MAXROWS=”3”>
#nome# - #telefone#
<HR>
```

</CFOUTPUT>

Neste exemplo teremos como resposta três registros. Note que o parâmetro MAXROWS nos limita ao número máximo de 3 registros e não que a exibição dos registros deva parar no registro de número 3.

Criando Formulários HTML

Para que possamos continuar com nosso tutorial, é necessário que saibamos construir e entender como funciona um formulário HTML.

O formulário é representado pelas tags <FORM></FORM>, que indicam o início e o fim do formulário respectivamente. Estas tags tem seus atributos e suas sub-tags.

Quando o usuário clica no botão de um formulário, os dados que ele preencheu são enviados para um arquivo (aplicação), onde estes dados são tratados e é apresentada uma resposta ao usuário.

Vamos ver agora alguns deles.

Atributos

Action – Indica para qual arquivo será enviado os dados do formulário.
Method – Indica o método usado pelo formulário. Vamos utilizar aqui o método POST.
Name – Opcional. Indica o nome do formulário.

Sub-Tags

As sub-tags Text, Password, Hidden, Radio Button e CheckBox são representadas pela tag <INPUT>, diferenciando-se apenas em seu tipo (TYPE) e em alguns atributos.

- Text
Representa um campo texto, que tem como atributos NAME, SIZE, MAXLENGTH.
Sintaxe:
<INPUT TYPE=”Text” NAME=”nome” SIZE=”30” MAXLENGTH=”50”>

- Password
Representa um campo senha, que tem os mesmos atributos da sub-tag TEXT.
Sintaxe:
<INPUT TYPE=”Password” NAME=”senha” SIZE=”8” MAXLENGTH=”8”>

- Hidden
Representa um campo escondido, que não é mostrado ao usuário. Seus atributos são NAME e VALUE.
Sintaxe:
<INPUT TYPE=”Hidden” NAME=”secreto” VALUE=”segredo”>

- Radio Button
Representa um campo de única escolha. Seus atributos são NAME, VALUE e CHECKED, este último opcional, indicado que o campo está selecionado.
Sintaxe:
<INPUT TYPE=”Radio” NAME=”escolha” VALUE=”1” CHECKED>

- CheckBox
Representa um campo de múltiplas escolhas. Seus atributos são os mesmos da sub-tag RADIO BUTTON.
Sintaxe:
<INPUT TYPE=”Checkbox” NAME=”escolha” VALUE=”1” CHECKED>

- Text Area
Representada pelas tags <TEXTAREA></TEXTAREA>, esta sub-tag é uma caixa de texto. Seus atributos são NAME, COLS, ROWS. E seu valor padrão deve estar entre as duas tags.
Sintaxe:
<TEXTAREA NAME=”Texto” COLS=”50” ROWS=”10”>Texto inicial</TEXTAREA>

- Select
Representada pelas tags <SELECT></SELECT>, esta sub-tag é uma caixa de seleção. Seu atributo é NAME. Tem ainda uma tag própria para suas opções de seleção, <OPTION></OPTION>. Seus atributos são VALUE e SELECTED, que indica se a opção está selecionada.
Sintaxe:
```
<SELECT NAME="seleção">
  <OPTION VALUE="1" SELECTED>1</OPTION>
  <OPTION VALUE="2">2</OPTION>
</SELECT>
```

- Submit
Representa o botão de envio do formulário. Representado também pela tag <INPUT>. Seus atributos são VALUE e NAME, este opcional.
Sintaxe:
<INPUT TYPE=”Submit” VALUE=”Enviar” NAME=”envia”>

- Reset
Representa o botão para limpar todos os dados do formulário. Representado também pela tag <INPUT>. Seus atributos são VALUE e NAME, este opcional.
Sintaxe:
<INPUT TYPE=”Reset” VALUE=”Limpar” NAME=”limpa”>

Inserindo dados em um banco de dados

Agora que já entendemos como funciona um formulário e sabemos como criar-lo, vamos começar a aprender como inserir dados em nosso banco de dados.

Antes de iniciar insira mais um campo chamado **código** (número inteiro), como chave primária, na tabela *Pessoal* e coloque códigos nos registros já existentes.

Seguindo ainda nossa primeira aplicação, crie um arquivo HTML contendo um formulário com a opção method = post, a action = http://seu_servidor/scripts/Tutorial_1/insere.cfm, três campos texto, de nomes: *codigo, nome* e *tel*, que representam os nomes dos campos da tabela *Pessoal* em nosso banco de dados. Crie também o botão de envio, *submit*.

* *A opção Action pode conter apenas o nome do arquivo destino, caso o arquivo html seja gravado no mesmo diretório do arquivo de aplicação. Ex.: action = insere.cfm*

Após criado o arquivo HTML, iremos agora criar o arquivo aplicação para inserir dados em nosso banco de dados.

Insere.cfm

```
<CFINSERT DATASOURCE=”Agenda” TABLENAME=”Pessoal”>
```

Você deve estar estranhando a aplicação ter apenas uma linha. Mas é isso mesmo, esta é a tag responsável pela inserção de dados em nosso banco de dados.

Iremos agora entender um pouco mais sobre atributos mais utilizados desta tag:

DATASOURCE	Requerido. Nome do datasource que contém o banco de dados
TABLENAME	Requerido. Nome da tabela onde você irá inserir os dados.
USERNAME	Opcional. Se especificado, cancela o username especificado no setup do ODBC.
PASSWORD	Opcional. Se especificado, cancela o password especificado no setup do ODBC.
FORMFIELDS	Opcional. Indicam quais os campos que devem ser inseridos. Os campos devem estar separados por vírgula.

Para que não haja problemas na hora de inserir os dados, é necessário que o nome dos campos no formulário sejam idênticos aos usados no banco de dados. Veremos mais adiante como quebrar esta regra.

Você já deve ter criado o exemplo como mostrado a seguir, mas sentiu falta de alguma coisa, como uma confirmação de que os dados foram realmente cadastrados. Pois bem, como já falamos, o CFML é muito parecido com o HTML e por isso também aceita suas tags. Vejamos a seguir:

Insere.cfm (Completo)

```
<CFINSERT DATASOURCE=”Agenda” TABLENAME=”Pessoal”>

<HTML>
<HEAD>
        <TITLE>Inserção em Banco de Dados</TITLE>
</HEAD>

<BODY>
<H2>Dados Cadastrados com Sucesso!</H2>
</BODY>

</HTML>
```

Atualizando dados em um banco de dados

Já vimos como consultar e inserir dados em um banco de dados, mas se notamos que existe algum campo com informações erradas, como podemos mudar-lo? É nesta parte do módulo que você vai aprender como fazer isto.

Selecionando o registro a ser alterado

Primeiro precisamos criar uma pesquisa em nosso banco de dados e mostrar o resultado desta pesquisa para que possamos escolher qual registro vamos alterar. Quando for mostrar o resultado desta pesquisa, será necessário criar um pequeno formulário com um campo escondido (hidden) para poder indicar à aplicação que este é o registro que se deseja alterar.

Veja como:

Altera.cfm

```
<CFQUERY DATASOURCE=”agenda” NAME=”AgendaPessoal”>
  SELECT * FROM Pessoal
</CFQUERY>

<CFOUTPUT QUERY=”AgendaPessoal”>
  #nome# - #tel# -
  <FORM METHOD=”POST” ACTION=”formaltera.cfm”>
    <INPUT TYPE=”HIDDEN” NAME=”codigo” VALUE=”#codigo#”>
    <INPUT TYPE=”SUBMIT” VALUE=”Alterar”>
  </FORM>
</CFOUTPUT>
```

Criando um formulário de alteração

O próximo passo é criar um formulário com os dados do registro escolhido para serem alterados.

Criaremos então uma nova consulta em nosso banco de dados onde o número do campo *tel* seja igual ao do registro escolhido na aplicação *Altera.cfm*. Feito isto colocaremos seus valores em um formulário bem semelhante ao que usamos para inserir novos dados, visto anteriormente.

Veja como:

FormAltera.cfm

```
<CFQUERY DATASOURCE=”agenda” NAME=”AgendaPessoal”>
  SELECT * FROM Pessoal
  WHERE codigo = #form.codigo#
</CFQUERY>

<CFOUTPUT QUERY=”AgendaPessoal”>
  <FORM METHOD=”POST” ACTION=”AlteraFim.cfm”>
    <INPUT TYPE=”TEXT” NAME=”nome” VALUE=”#nome#”>
    <INPUT TYPE=”TEXT” NAME=”tel” VALUE=”#tel#”>
    <INPUT TYPE=”HIDDEN” NAME=”codigo” VALUE=”#codigo#”>
    <INPUT TYPE=”SUBMIT” VALUE=”Alterar”>
  </FORM>
</CFOUTPUT>
```

Finalizando a alteração

O último passo é criar o arquivo que registrará as alterações no banco de dados.

Semelhante a tag <CFINSERT>, utilizaremos a tag <CFUPDATE> para alterar as informações do registro selecionado.

AlteraFim.cfm

```
<CFUPDATE DATASOURCE=”Agenda” TABLENAME=”Pessoal”>

<HTML>
<HEAD>
        <TITLE>Alteração em Banco de Dados</TITLE>
</HEAD>

<BODY>
<H2>Dados Alterados com Sucesso!</H2>
</BODY>

</HTML>
```

Alguns atributos utilizados nesta tag:

DATASOURCE	Requerido. Nome do datasource que contém o banco de dados
TABLENAME	Requerido. Nome da tabela onde você irá inserir os dados.
USERNAME	Opcional. Se especificado, cancela o username especificado no setup do ODBC.
PASSWORD	Opcional. Se especificado, cancela o password especificado no setup do ODBC.
FORMFIELDS	Opcional. Indicam quais os campos que devem ser inseridos. Os campos devem estar separados por vírgula.

Inserindo, Atualizando e Excluindo registros com CFQUERY

Vimos anteriormente como inserir e atualizar registros com <CFINSERT> e <CFUPDATE> respectivamente. Mas caso necessitemos de inserções e atualizações mais complexas podemos utilizar instruções SQL para executar-los.

Iremos utilizar aqui os formulários exemplos utilizados nos tópicos anteriores.

Inserindo

Para inserir um registro basta substituir a tag <CFINSERT> pelo seguinte:

Insere.cfm

```
<CFQUERY DATASOURCE=”Agenda” NAME=”Insere”>
  INSERT INTO Pessoal (codigo, nome, tel)
  VALUES (#form.codigo#, ‘#form.nome#’, ‘#form.tel#’)
</CFQUERY>

<HTML>
<HEAD>
```

```
        <TITLE>Inserção em Banco de Dados</TITLE>
</HEAD>

<BODY>
<H2>Dados Cadastrados com Sucesso!</H2>
</BODY>

</HTML>
```

Alterando

Para alterar um registro basta substituir a tag <CFUPDATE> pelo seguinte:

AlteraFim.cfm

```
<CFQUERY DATASOURCE=”Agenda” NAME=”Altera”>
  UPDATE Pessoal
  SET nome = ‘#form.nome#’,
  tel = ‘#form.tel#’
  WHERE codigo = #form.codigo#
</CFQUERY>

<HTML>
<HEAD>
        <TITLE>Alteração em Banco de Dados</TITLE>
</HEAD>

<BODY>
<H2>Dados Alterados com Sucesso!</H2>
</BODY>

</HTML>
```

Excluindo registros

Para excluir registros não a outro modo a não ser utilizando instruções SQL através da tag <CFQUERY>. Para ilustramos como funciona, vamos começar selecionando qual registro desejamos excluir de nosso banco de dados.

Exclui.cfm

```
<CFQUERY DATASOURCE=”agenda” NAME=”AgendaPessoal”>
  SELECT * FROM Pessoal
</CFQUERY>

<CFOUTPUT QUERY=”AgendaPessoal”>
  #nome# - #tel# -
  <FORM METHOD=”POST” ACTION=”excluifim.cfm”>
    <INPUT TYPE=”HIDDEN” NAME=”codigo” VALUE=”#codigo#”>
    <INPUT TYPE=”SUBMIT” VALUE=”Excluir”>
  </FORM>
</CFOUTPUT>
```

** Note que o exemplo acima é igual ao utilizado para selecionar o registro a ser alterado, exceto pelo nome da aplicação da action.*

Agora vamos ver como fica a aplicação que exclui em definitivo o registro:

ExcluiFim.cfm

```
<CFQUERY DATASOURCE=”Agenda” NAME=”Exclui”>
  DELETE FROM Pessoal
  WHERE codigo = #form.codigo#
</CFQUERY>

<HTML>
<HEAD>
        <TITLE>Exclusão em Banco de Dados</TITLE>
</HEAD>

<BODY>
<H2>Dados Excluídos com Sucesso!</H2>
</BODY>

</HTML>
```