

# Visualize os dados do Access através do Excel

Mauro Possatto\*  
accxls.exe

Relatórios, por incrível que pareça muitas vezes este é o item mais complexo de um sistema. Isso acontece porque em muitos casos seu cliente acaba pedindo um daqueles relatórios mirabolantes onde não existe nenhum padrão para exibir os dados, isto é, você não sabe quantas linhas e nem quantas colunas seu relatório irá ter e, com isso ganha uma belíssima dor de cabeça no momento de desenvolver seu sistema.

Outro dia eu estava levantando as necessidades de um cliente e ele me apresentou um desses relatórios absurdos acompanhado da seguinte frase: “Este relatório é o mais importante do sistema, e deverá ser analisado por toda a diretoria da Empresa.” Em outras palavras ele quis dizer: “Se este relatório não sair, nada de pagamento no final do mês.”

Entendido o recado, o cliente lembrou de um detalhe quase sem importância: “Ahhh, ia me esquecendo, a maioria dos diretores quer receber este relatório via e-mail e visualizá-lo em forma de planilha.”

Controlado o frio na barriga e o desespero inicial respondi com aquela cara-de-pau padrão dos Analistas de Sistemas: “Sem problemas, sendo no Excel o senhor facilitou demais a vida dos meus programadores.”

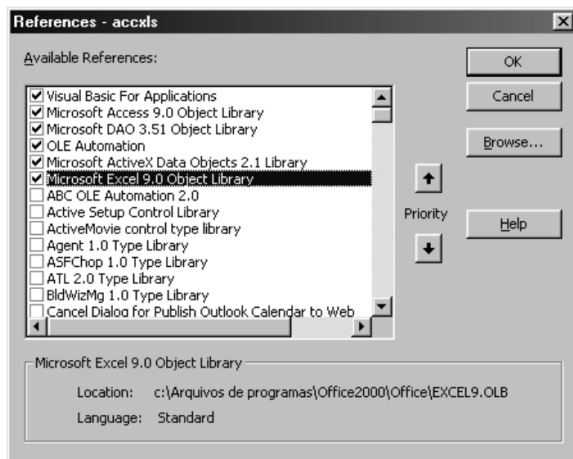
Resultado da reunião, um cliente cheio de esperanças e um Analista de Sistemas à beira do desespero. Ao chegar na Fórum tomei uma dose cavalares de café, sentei na minha mesa, respirei fundo, e comecei a analisar o relatório. Ele era totalmente dinâmico, não havia possibilidade de construí-lo no Access sem que fosse em tempo de execução, e isso iria deixar o sistema bastante pesado. De repente uma voz do além, cuja sonoridade era muito semelhante à voz do Bill Gates, me repetia “Construa direto no Excel....Construa direto no Excel...”. De imediato eu pensei: “Vou construir direto no Excel.” Pronto, problema resolvido, agora só falta fazer os códigos.

## Mãos a Obra

Como não posso demonstrar exatamente o relatório que criei para o Cliente, utilizei algumas tabelas importadas do Northwind para apresentar como é possível realizar a manipulação do Excel de dentro do Access. Se você quiser ver diretamente o resultado, execute o arquivo ACCXLS.MDB, abra o formulário frm\_Start e clique no botão Visualizar Relatório.

Agora, se você quiser acompanhar a matéria passo-a-passo crie um novo banco de dados e importe as tabelas, consultas e formulários existentes no arquivo ACCXLS.MDB.

O primeiro e mais importante passo para que tudo funcione corretamente, é estabelecer as referências à biblioteca de objetos Excel no seu novo MDB. Para isso, abra uma janela de código qualquer, vá ao menu Ferramentas e selecione Referências. Faça com que sua janela de referências fique idêntica à exibida na figura 1.



**Fig. 1- A Janela de referências para comunicação com o Excel**

É importante observar que estas referências são do Office 2000, portanto, se você estiver utilizando versões anteriores do Office, elas serão um pouco diferentes.

Antes de começar a construir o código vamos entender o objetivo do relatório demonstrado neste artigo. Execute a consulta `qct_OrdersTotals`. Nela você verá uma consulta de referência cruzada que apresenta os valores de produtos vendidos para o ano de 1997. As consultas de referência cruzada são excelentes exemplos para este tipo de relatório, pois se os parâmetros variam, obviamente irão variar também a quantidade de linhas e colunas que devem ser apresentadas no relatório.

No exemplo deste artigo, o número de colunas está fixo, porém você verá, através da construção do código, que este número poderia perfeitamente ser variável.

É importante que você saiba que não é preciso conhecer perfeitamente toda a estrutura dos objetos do Excel, pois o grande macete para realizar esta tarefa é utilizar como referência o código gerado pelas macros do Excel, isto é, sempre que você desejar realizar alguma operação automática no excel via código, crie uma macro e depois analise o código gerado por ela, você terá então a estrutura que seu código precisará ter para fazer a operação desejada.

Agora vamos começar a criar os códigos para o nosso relatório no Excel. Primeiro crie uma variável global que será utilizada para manipular o Excel. Em qualquer seção de declarações de código escreva a linha abaixo:

```
Global Nexcapp As New Excel.Application
```

A utilização desta variável irá forçar a criação de uma nova instância do Microsoft Excel e será tratada como uma variável do tipo aplicativo.

A seguir crie as funções abaixo:

```
Function BordaDir(strange As String)
    Nexcapp.Range(strange).Select
    With Nexcapp.Selection.Borders(xlEdgeRight)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
End Function
Function BordaESQ(strange As String)
    Nexcapp.Range(strange).Select
    With Nexcapp.Selection.Borders(xlEdgeLeft)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
End Function
Function BordaSup(strange As String)
    Nexcapp.Range(strange).Select
    With Nexcapp.Selection.Borders(xlEdgeTop)
        .LineStyle = xlContinuous
```

```

        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
End Function
Function BordaInf(strrange As String)
    Nexccapp.Range(strrange).Select
    With Nexccapp.Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlMedium
        .ColorIndex = xlAutomatic
    End With
End Function

```

A finalidade destas funções é permitir que você possa definir o estilo de cada uma das bordas da(s) célula(s) fornecidas através da variável "strrange". Obviamente, a função BordaDir cria a borda da direita, a BordaEsq cria a da esquerda, a Borda Sup cria a borda superior e a BordaInf cria a borda inferior. Estas funções foram criadas em separado pois imagine o que aconteceria se, para cada vez que você precisasse configurar uma das bordas, você escrevesse 5 linhas de código. A função viraria um verdadeiro monstro e, mais a frente você verá que ela não é tão pequena assim.

Agora chegou a hora de escrever a função propriamente dita. Esta função irá abrir o Excel, configurar o cabeçalho do relatório e realizar uma leitura em toda a consulta que vimos anteriormente para ir alimentando o Excel linha a linha.

Dê uma olhada no código que se encontra dentro da função PlanXLS, no módulo bas\_Excel do nosso mdb(accxls).

Acho que deu para ficar um pouco assustado não é verdade? Não se preocupe, a função é grande mas não é complicada de ser entendida, vamos dar uma passada por alguns pontos importantes dela para que você possa entender perfeitamente a lógica que eu utilizei para desenvolvê-la.

No início foram declaradas as variáveis que serão utilizadas pela função. No código contido no arquivo ACCXLS.MDB estas variáveis estão comentadas, facilitando a compreensão da sua finalidade dentro do código.

Logo em seguida, as variáveis de banco de dados e de recordset são inicializadas com o banco de dados atual e com a consulta de referência cruzada citada anteriormente. Depois a aplicação Excel é criada, sua visualização fica disponível e o nome da planilha é criado. Logo em seguida o cabeçalho do relatório é formatado. Eu determinei que o cabeçalho deveria ocupar uma célula mesclada e que fosse fixado da coluna A até a coluna N.

Como neste exemplo a quantidade de colunas é fixa eu aproveitei para formatar a largura de cada uma das colunas em 3 intervalos. Na coluna A ficará o nome de cada um dos produtos, das colunas B até M os valores mensais e na coluna N um total geral por produto.

Outro ponto de extrema importância para deixar a quantidade de colunas variável é utilizar a função CHR para manipular a coluna desejada. Por exemplo, se você deseja manipular a coluna A, você deverá passar esse valor através da função CHR, que transforma valores numéricos em caracteres alfanuméricos segundo a tabela ASC, como a letra A equivale ao número 65, basta utilizar CHR(65) para dizer ao Excel que você está manipulando a coluna A.

Voltando à explicação da função, você verá que as variáveis que determinam a linha e a coluna iniciais foram configuradas para linha 3 e coluna B, a coluna B foi utilizada pois eu sei que na coluna A sempre estará o nome do meu produto.

Em seguida, é criado o cabeçalho do relatório. Para definir os títulos das colunas B em diante eu utilizei alguns recursos especiais, em primeiro lugar criei uma estrutura de controle FOR que será executada de acordo com a quantidade de colunas existentes na minha consulta excluindo a primeira coluna que contém o nome do produto, por isso o -1 após a propriedade rsOrders.Fields.Count. Nesta estrutura o código irá coluna-a-coluna no Excel passando o nome do campo e aproveitará para formatar a célula onde o valor será passado. Depois da estrutura FOR, mais uma coluna é adicionada para exibir os totais por produto.

Em seguida, será iniciado o preenchimento das células de acordo com o conteúdo do recordset. Observe que a variável intLinhaIni é utilizada para saber em que linha foi iniciado o preenchimento do conteúdo. Esta informação será útil mais adiante.

Duas estruturas de controles aninhadas são utilizadas, uma LOOP que irá percorrer o recordset do início ao fim e, novamente, uma FOR que servirá para preencher cada uma das colunas da planilha.

Neste momento a coluna inicial é configurada com o código 65, que representa a coluna A e, então, cada uma das colunas e linhas são preenchidas. Observe que existe um IF dentro da estrutura FOR que tem como objetivo formatar como moeda as colunas que não apresentam o nome do produto.

Depois da estrutura FOR um pequeno código cria a fórmula exibida na coluna de total do produto.

```

Nexccapp.Selection.Value = "=SUM(B" & intlinha & ":" & Chr(66 + (rsOrders.Fields.Count - 2)) & intlinha & ")"

```

Repare que a fórmula é montada em tempo de execução e ela irá realizar a soma de todos os valores contidos entre a coluna B e a quantidade de colunas do recordset para a linha atual.

Depois que todo o conteúdo do recordset é transferido para a planilha, é criada uma última linha que irá apresentar o Total Geral.

Nexccapp.Selection.Value = "=SUM(B" & intlinhaIni & ":" & Chr(66 + (rsOrders.Fields.Count - 2)) & intlinha & ")"

Repare que para determinar a área de seleção da fórmula, a variável IntLinhaIni foi utilizada. Em seguida a planilha é gravada de acordo com o parâmetro strfilename e a aplicação Excel fechada. Agora basta você abrir a planilha no Excel para ver o resultado final.

### **Fechando o assunto**

Infelizmente, devido ao contrato de confidencialidade de informações e dos direitos autorais do sistema desenvolvido, eu não pude apresentar neste artigo o relatório que foi criado para o Cliente. Este relatório utiliza uma infinidade de outros recursos do Excel e apresenta uma complexidade muito superior à apresentada aqui. Para que você possa ter uma idéia desta complexidade, em determinado momento, quando solicitei a compilação do código, recebi uma mensagem do Access que dizia o seguinte texto "Código muito grande!", em outras palavras, estourei o limite de linhas por função do Access e fui obrigado a dividir minha função em várias outras subs, entre elas, as de configuração das bordas.

Apesar de estar utilizando um exemplo bem simples neste artigo, acredito que as informações passadas são suficientes para que a partir de agora você possa explorar um pouco mais dos recursos de compatibilidade oferecidos no ambiente Office e, principalmente, que você possa desenvolver aqueles famigerados relatórios com muito mais facilidade, permitindo que seu Cliente tenha em mãos um resultado acima do esperado como foi o que aconteceu comigo.

Por hoje é tudo, fico por aqui com a esperança de ter contribuído para o aumento dos seus conhecimentos. Um grande abraço e até a próxima.

**\*Mauro Possatto** (maurop@forumaccess.com) é gerente de sistemas do Fórum Access.