



# Boas Vindas

**Apresentação**



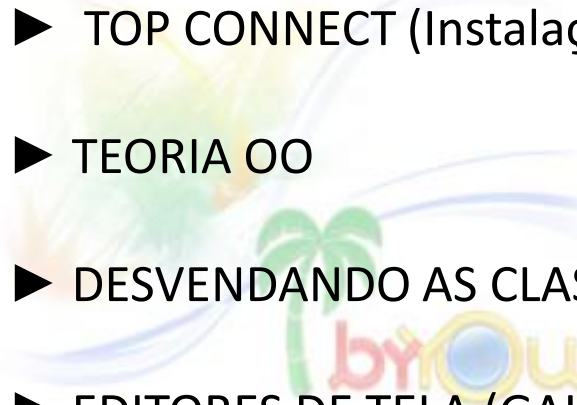
# **ADVPL II**

**Recursos Avançados**



# Conteúdo Programático

# Conteúdo Programático

- 
- ▶ TOP CONNECT (Instalação e configuração)
  - ▶ TEORIA OO
  - ▶ DESVENDANDO AS CLASSES (.CH)
  - ▶ EDITORES DE TELA (GAIA, DESIGNER, MKW)
  - ▶ MONTANDO JANELAS (ENCHOICEBAR, GETDADOS, FOLDER)
  - ▶ OBJETOS DE CONTROLE (RADIO, COMBO BOX, SAY, GET, CHECK)
  - ▶ RELATÓRIOS GRÁFICOS (TMSPRINTER)A

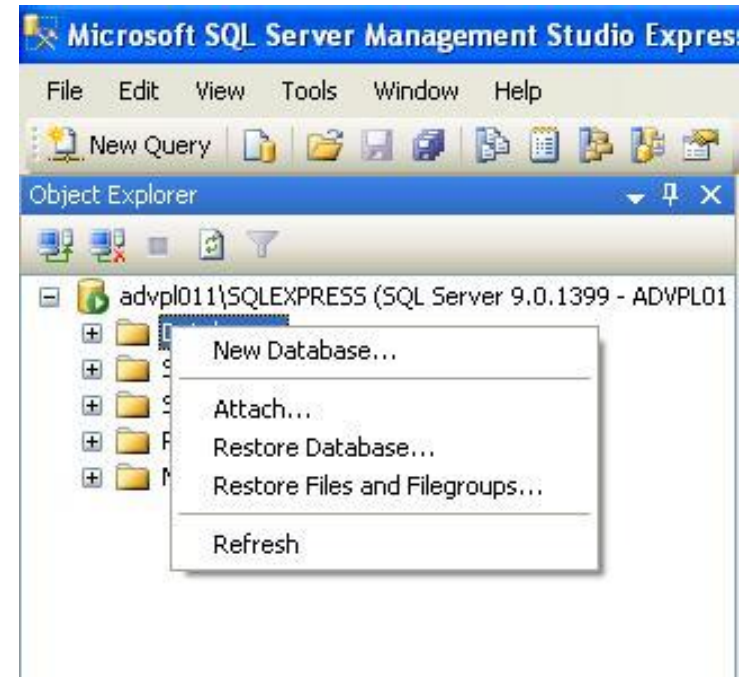


# **TRABALHANDO COM BANCO DE DADOS**

# Banco de Dados

## Microsoft SQL Server

Depois de instalado o Banco de Dados devemos inserir um novo banco de dados clicando com o botão direito em database e escolhendo “New database” que será o ambiente de relação com o top connect que veremos mais adiante.



# Banco de dados

## Microsoft SQL Server Express

### MSSQL

Se necessário crie um usuário qualquer , no caso do curso crie “sa” e coloque uma senha que será necessária na configuração do top connect. Clique em Security /Logins, botão direito “New login” . SQL Server e escolha uma senha.





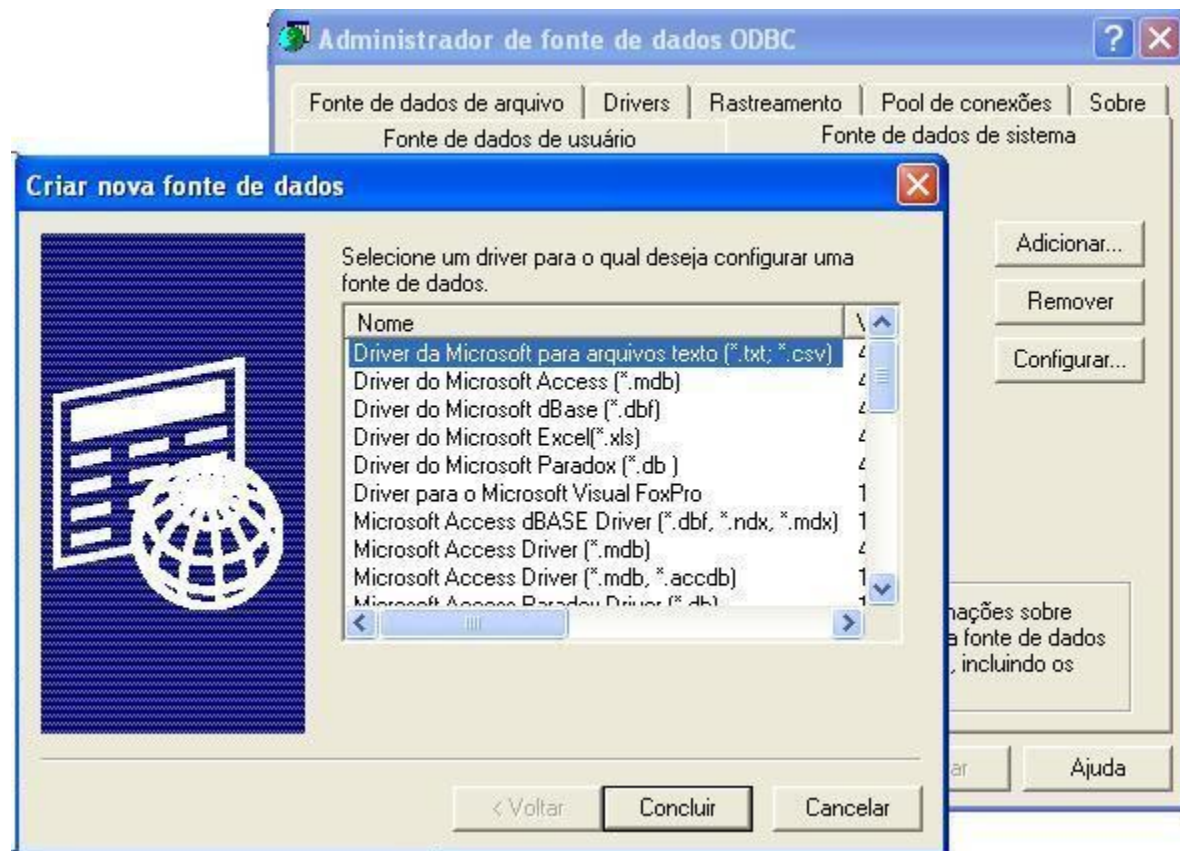
# **FONTE DE DADOS (ODBC)**



# Banco de Dados

Microsoft Windows

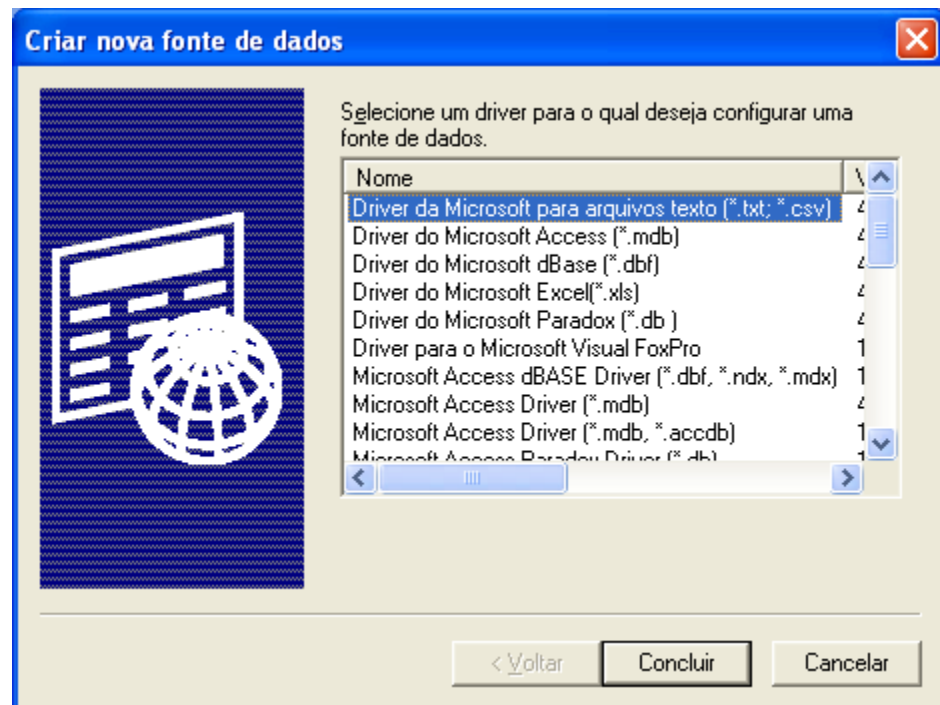
Em “Painel de Controle” abra a opção “Ferramentas administrativas”. Clique em “Fontes de Dados (ODBC)”. Na pasta “Fonte de dados de sistema” clique em “Adicionar”.



# Banco de Dados

Microsoft Windows

Selecione o driver “SQL Native Client”. Isto para este caso específico.

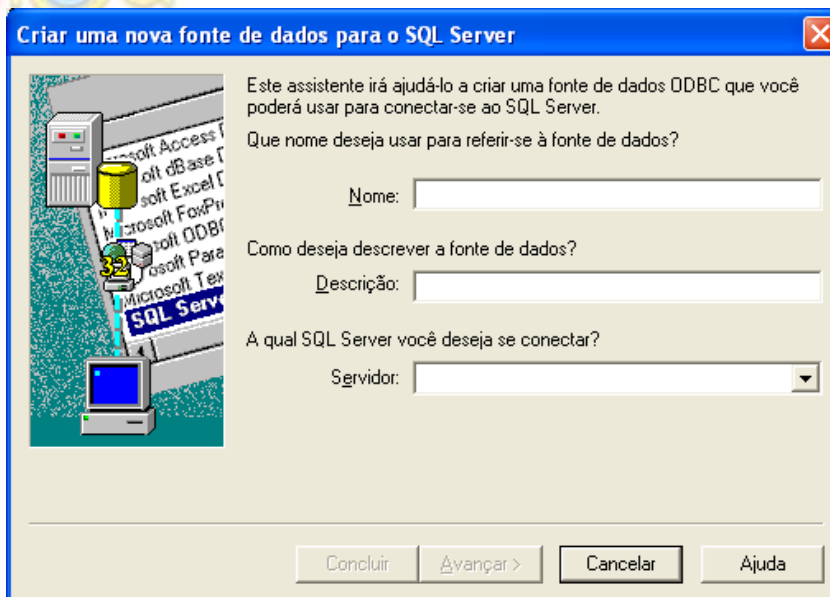


# Banco de Dados

## Microsoft Windows

**Criaremos o ODBC que fará conexão com o Sql. Neste caso definiremos o Servidor como a máquina que está instalado o Banco “aluno-PC\sqlexpress”**

**Nas na empresa será o servidor de banco de dados. Quanto ao nome e a Descrição fica a critério da política da empresa. Aqui usaremos “aula”.**



**Criar uma nova fonte de dados para o SQL Server**

Este assistente irá ajudá-lo a criar uma fonte de dados ODBC que você poderá usar para conectar-se ao SQL Server.

Que nome deseja usar para referir-se à fonte de dados?

Nome:

Como deseja descrever a fonte de dados?

Descrição:

A qual SQL Server você deseja se conectar?

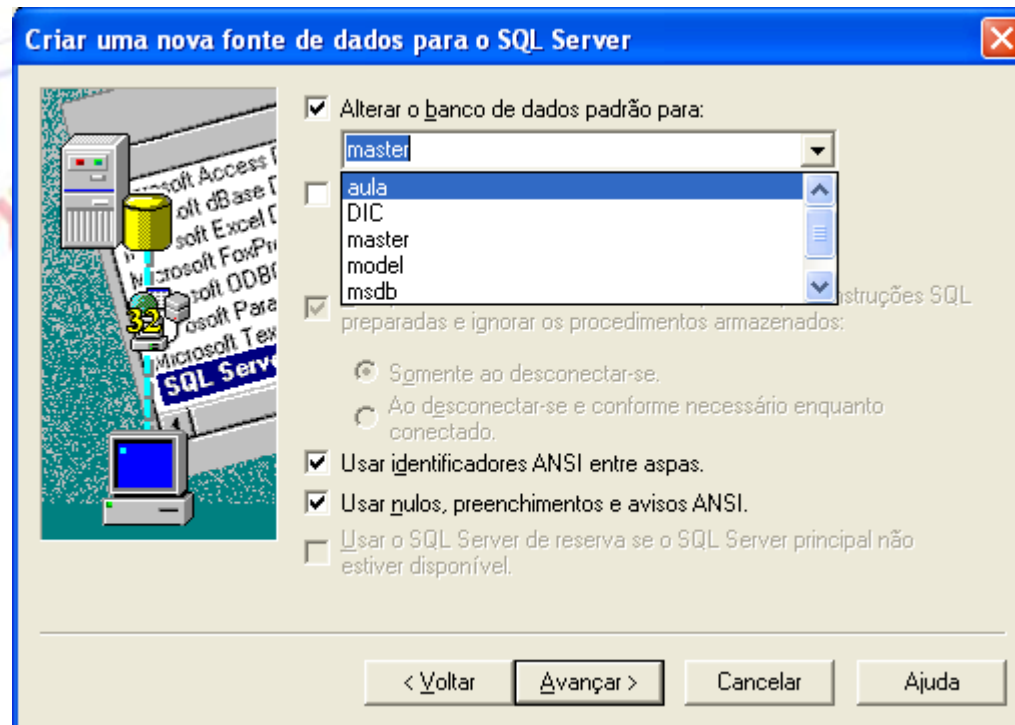
Servidor:

Concluir Avançar > Cancelar Ajuda

# Banco de Dados

Microsoft Windows

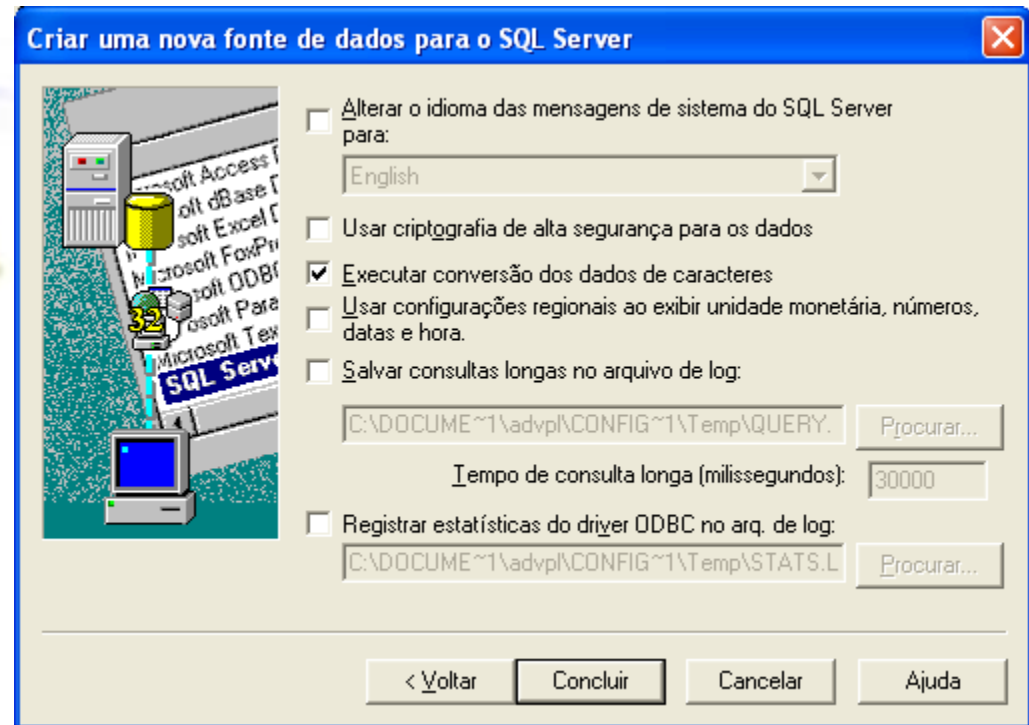
Neste passo escolheremos o banco criado “aula”



# Banco de Dados

Microsoft Windows

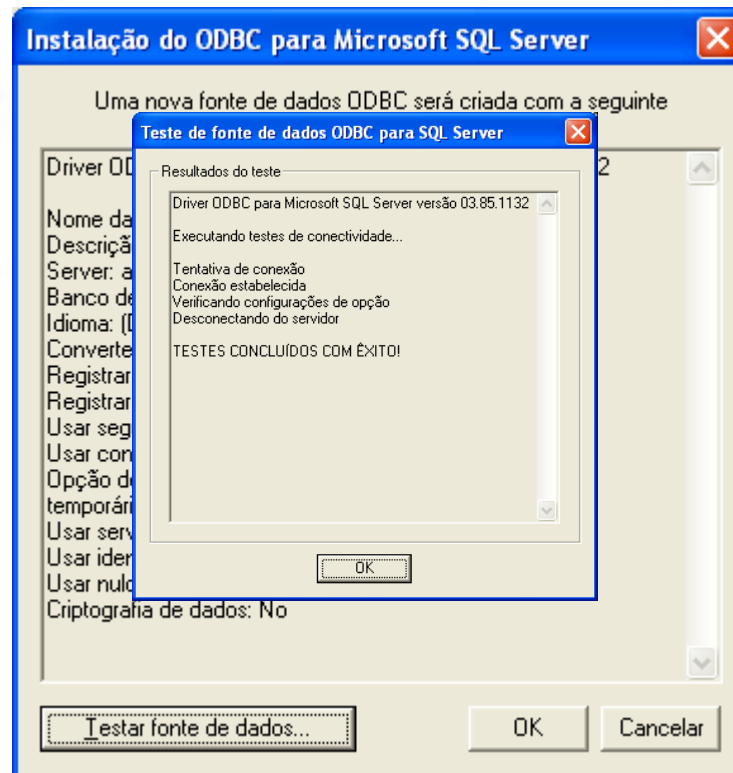
Neste último passo manteremos os parâmetros default. Clique em  
“Concluir”



# Banco de Dados

Microsoft Windows

Clique em testar fonte de dados.





# Top Connect

# Banco de Dados

TopConnect

**Agora acesse o TopConnect (TotvsDBAcces Monitor V. 10)**



**Neste o servidor é o próprio computador, portanto, definido como Localhost, na empresa deve ser o nome do servidor. A porta e sugestão do Próprio sistema , porém na empresa deve ser a porta utilizada para acesso Ao servidor.**



# Banco de Dados

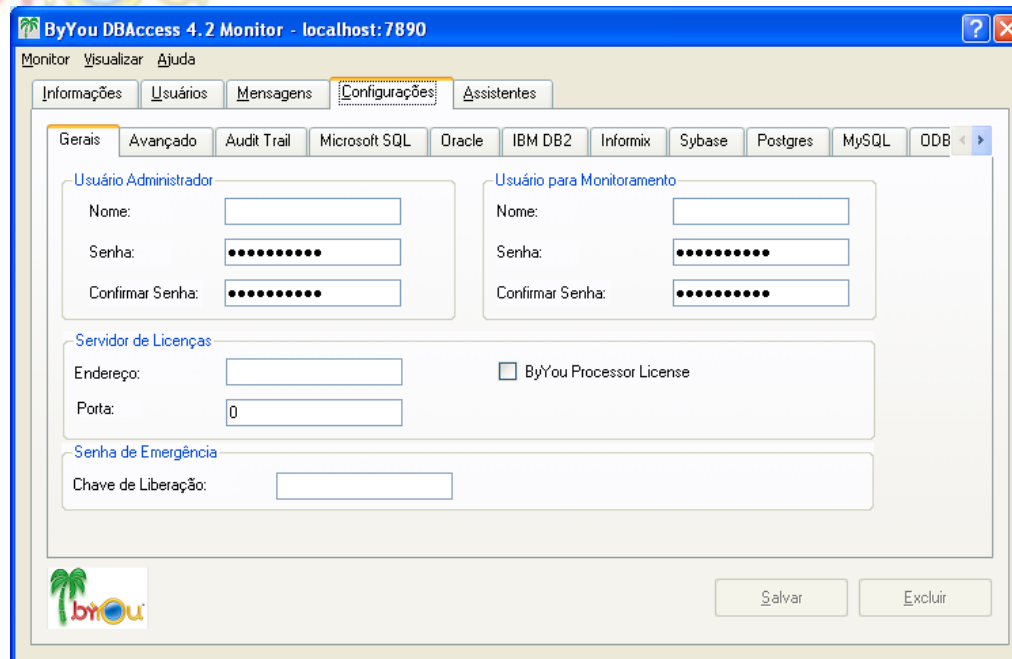
## TopConnect

Selecionando a pasta configurações teremos as opções “Gerais” é onde

Faremos a confirmação da senha.

Nas outras pastas definiremos qual o banco de dados utilizado e

Configuraremos o ambientes.

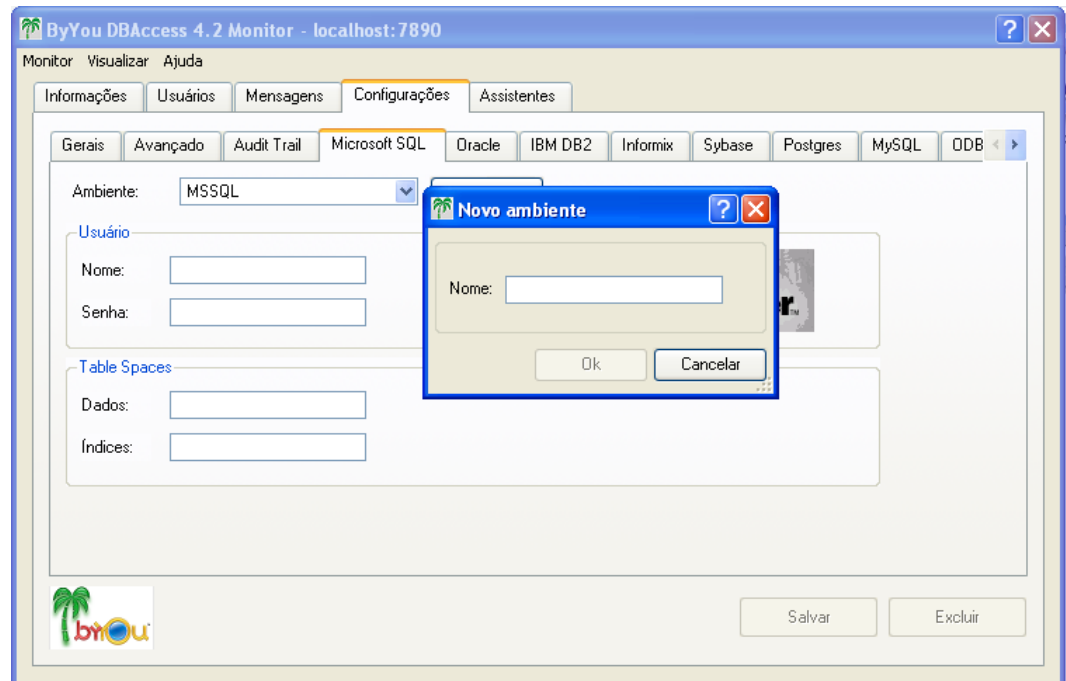


The screenshot shows the 'ByYou DBAccess 4.2 Monitor - localhost: 7890' application window. The 'Monitor' menu is open, showing options like 'Monitor', 'Visualizar', and 'Ajuda'. The 'Configurações' (Settings) tab is selected, and within it, the 'Gerais' (General) sub-tab is active. The 'Gerais' sub-tab contains several sections: 'Usuário Administrador' (Administrator User) with fields for 'Nome' (Name), 'Senha' (Password), and 'Confirmar Senha' (Confirm Password); 'Usuário para Monitoramento' (User for Monitoring) with similar fields; 'Servidor de Licenças' (License Server) with fields for 'Endereço' (Address) and 'Porta' (Port), and a checkbox for 'ByYou Processor License'; and 'Senha de Emergência' (Emergency Password) with a 'Chave de Liberação' (Release Key) field. The 'ByYou' logo is visible in the bottom left corner, and 'Salvar' (Save) and 'Excluir' (Delete) buttons are in the bottom right corner.

# Banco de Dados

TopConnect

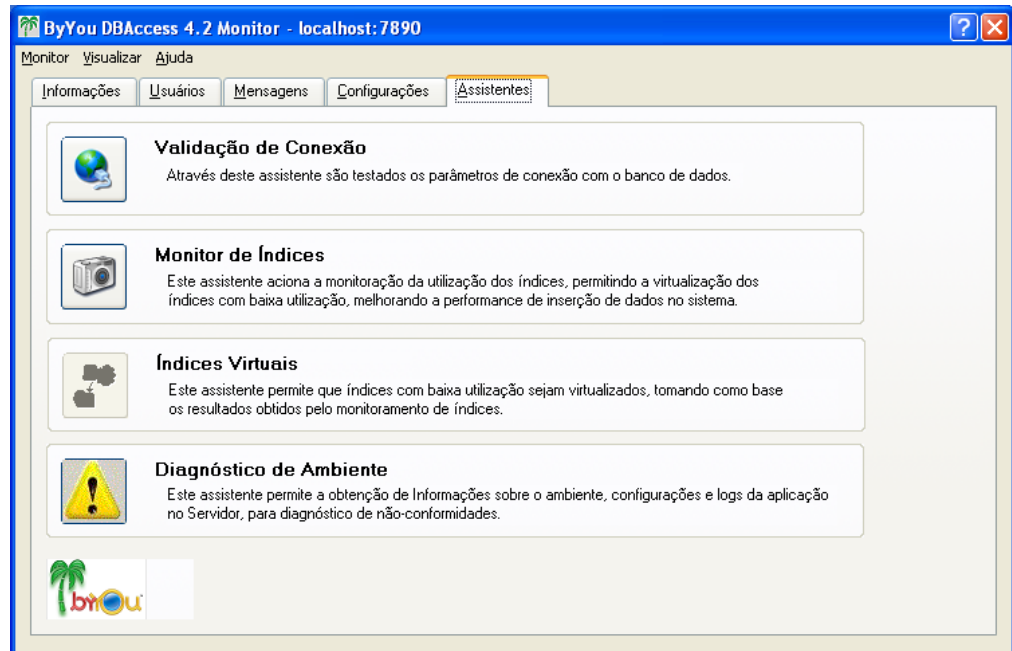
Neste caso o ambiente ficou definido como siga, porém, na empresa  
Pode-se criar o ambiente de acordo com a politica definida. Clicando em  
**Novo.**



# Banco de Dados

TopConnect

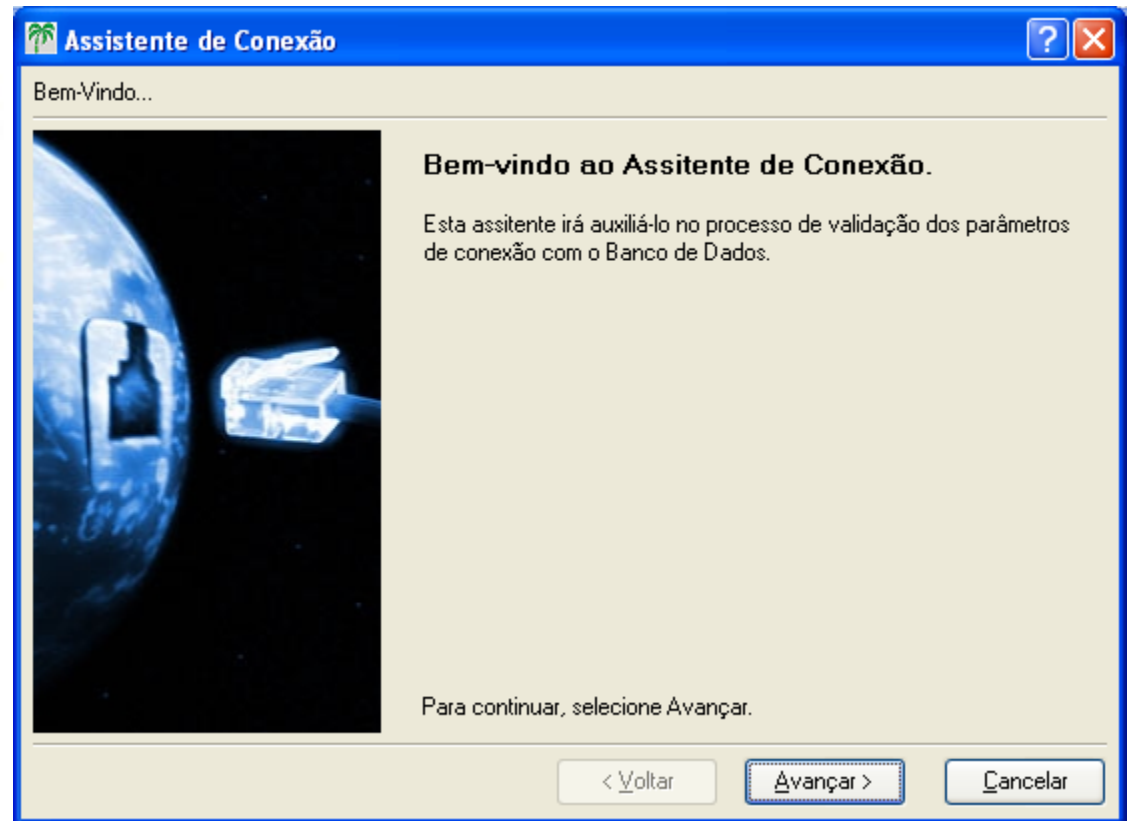
Clique na pasta “Assistentes” e na opção “Validação de Conexão”



# Banco de Dados

TopConnect

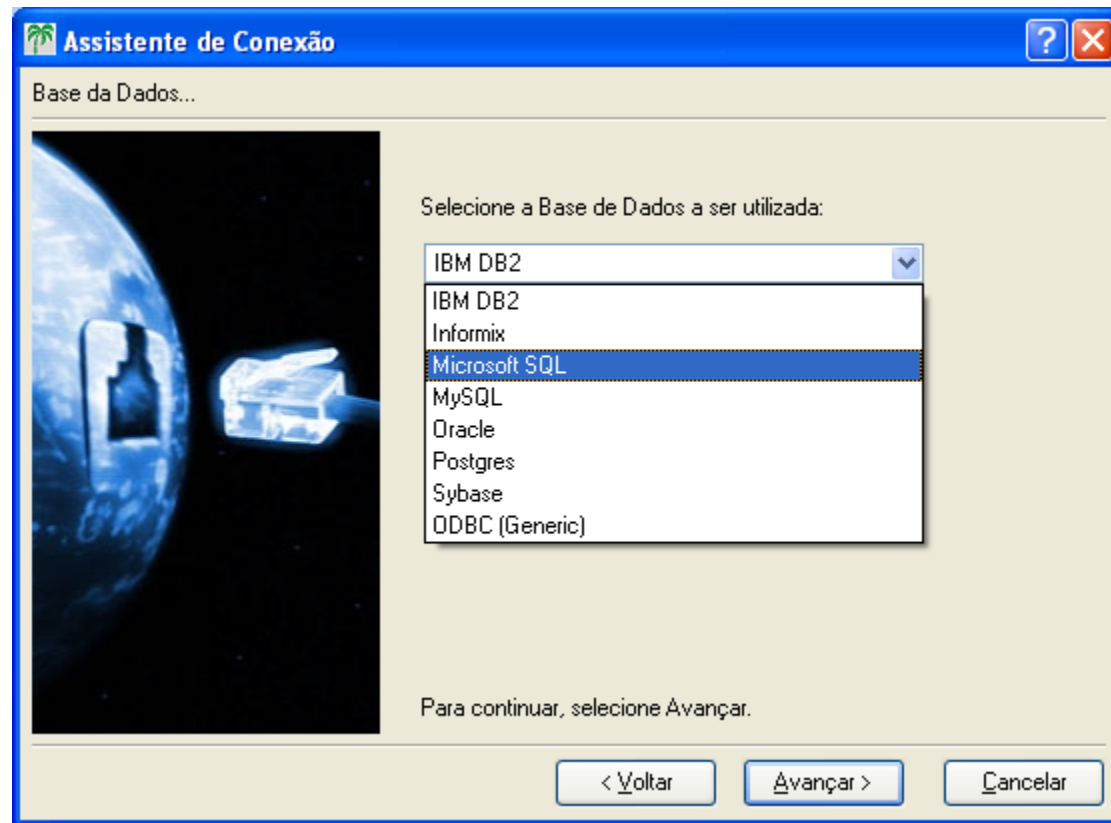
No assistente de conexão validaremos a conexão do Protheus com o banco de dados.



# Bando de Dados

TopConnect

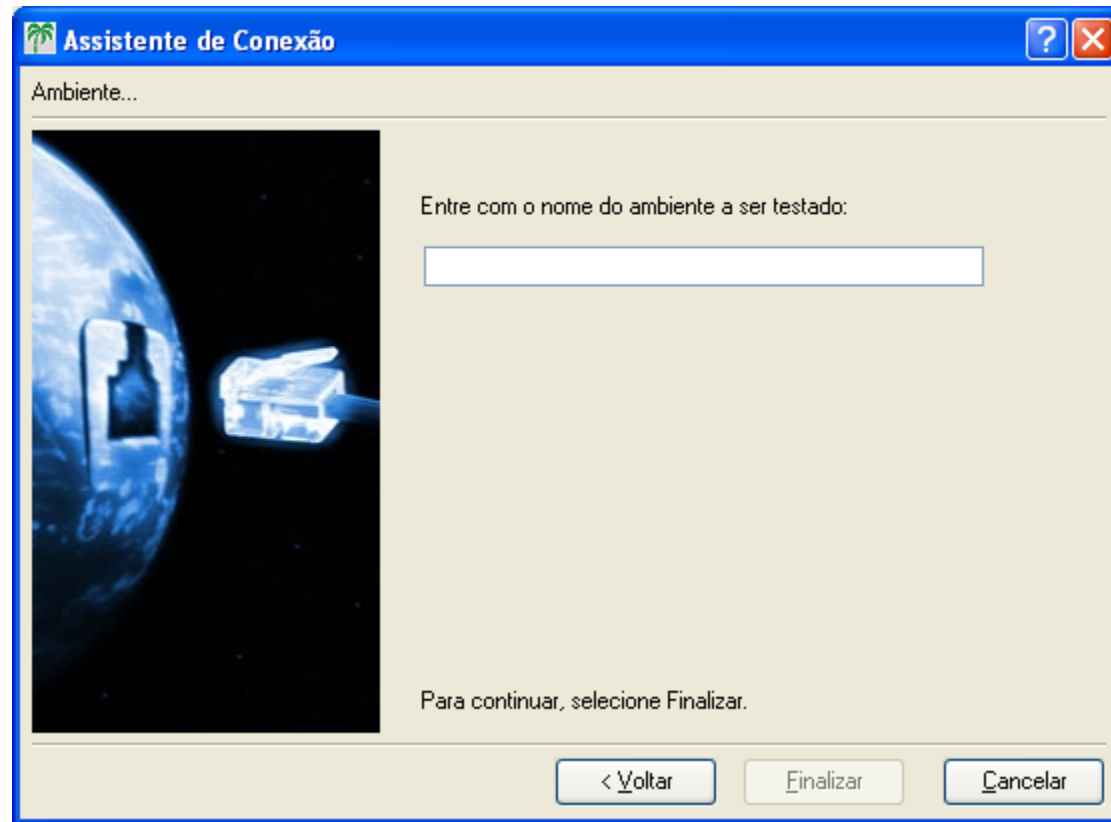
Aqui definiremos qual banco usaremos. E neste caso Microsoft SQL.



# Banco de Dados

TopConnect

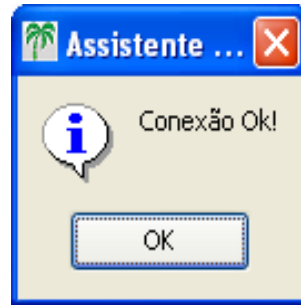
**Este ambiente é o mesmo ambiente que foi criado na instalação e  
Configuração do MSSQL.**



# Banco de Dados

TopConnect

**Aparecendo “Conexão OK !” Significa que sua conexão foi realizada com Sucesso e seu banco de dados está ligado ao Protheus.**





**APSDU**  
**Ou MPSDU**



# Banco de Dados

Protheus - APSDU

**APSDU**

**Acesse a APSDU ou MPSDU através do remote**



The image shows a screenshot of a software dialog box titled "ByYou - Parâmetros Iniciais". The dialog box has a blue header bar with the title. Below the header, there is a decorative banner with a colorful abstract design and the "byYou" logo. The main area of the dialog contains three sections, each with a label and a dropdown menu:

- Programa inicial**: The dropdown menu shows "APSDU".
- Comunicação no cliente**: The dropdown menu shows "tcp".
- Ambiente no servidor**: The dropdown menu shows "environment".

At the bottom of the dialog, there are two buttons: "Ok" and "Cancela".

# Bando de Dados

Protheus - APSDU

## APSDU

Defina o usuário e clique OK.



A screenshot of a Windows-style login dialog box titled "Login" with a small tree icon. The dialog has a blue title bar and a red close button. It contains two input fields: "Usuário" (User) and "Senha" (Password). The "Usuário" field has a blue selection bar. To the right of the fields is the TOTVS logo, which consists of a blue circle with a white 'T' and the word "TOTVS" below it. At the bottom are two buttons: "Ok" and "Cancelar".

Usuário

Senha

Ok Cancelar

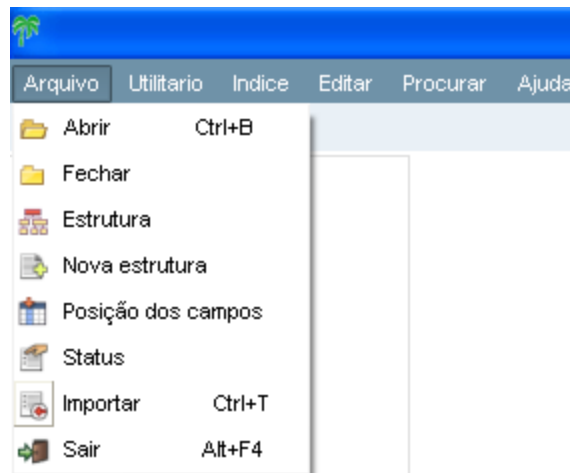
TOTVS

# Bando de Dados

Protheus - APSDU

## APSDU

**Selecione a opção arquivo e depois ou simplesmente CTRL+T**



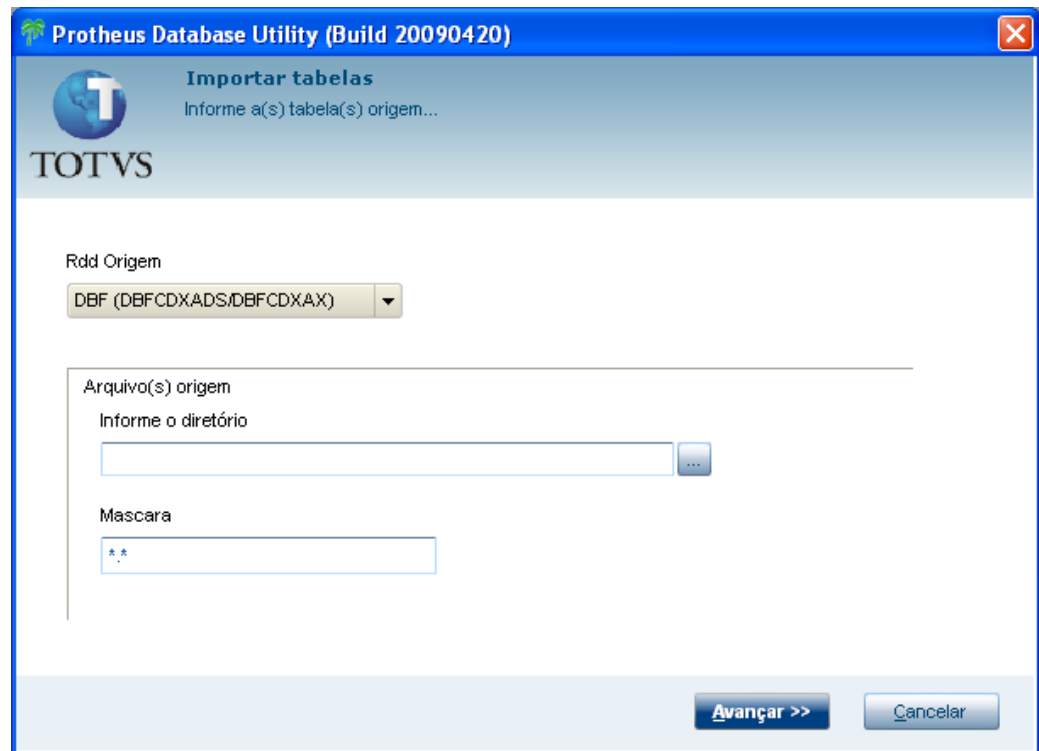
# Banco de Dados

Protheus - APSDU

## APSDU

Aqui selecionaremos o tipo de banco de dados que importaremos.

O arquivos estar gravado na raiz do root path ou em um diretório dele.



Protheus Database Utility (Build 20090420)

**Importar tabelas**  
Informe a(s) tabela(s) origem...

TOTS

Rdd Origem  
DBF (DBFCDXADS/DBFCDXAX)

Arquivo(s) origem  
Informe o diretório

Mascara  
\*\*

Avançar >> Cancelar

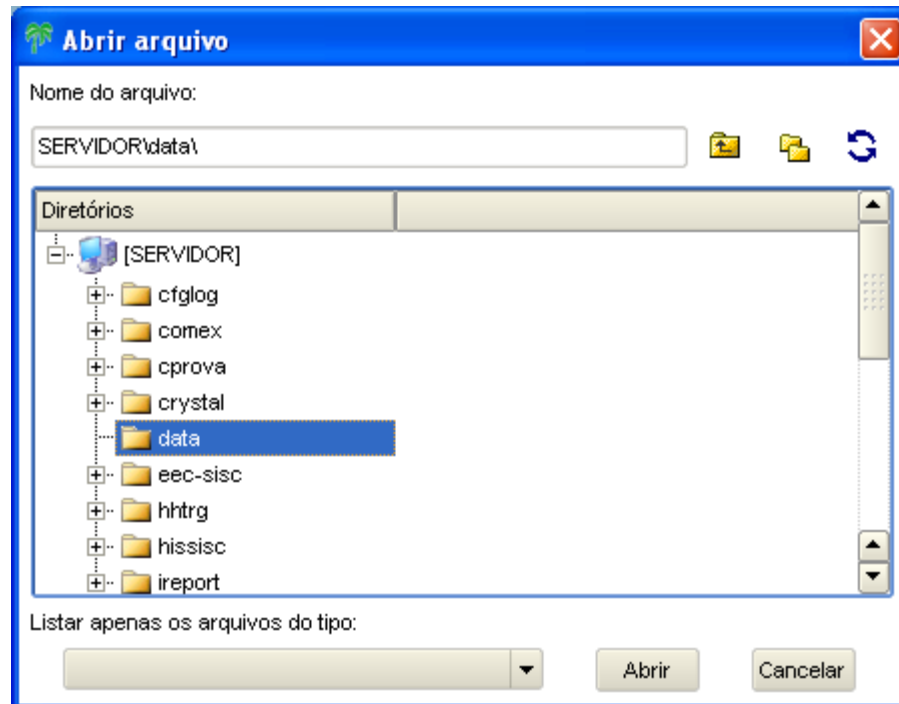
# Bando de Dados

Protheus - APSDU

## APSDU

Selecionaremos o arquivo que queremos importar. E clicaremos em “Abrir”

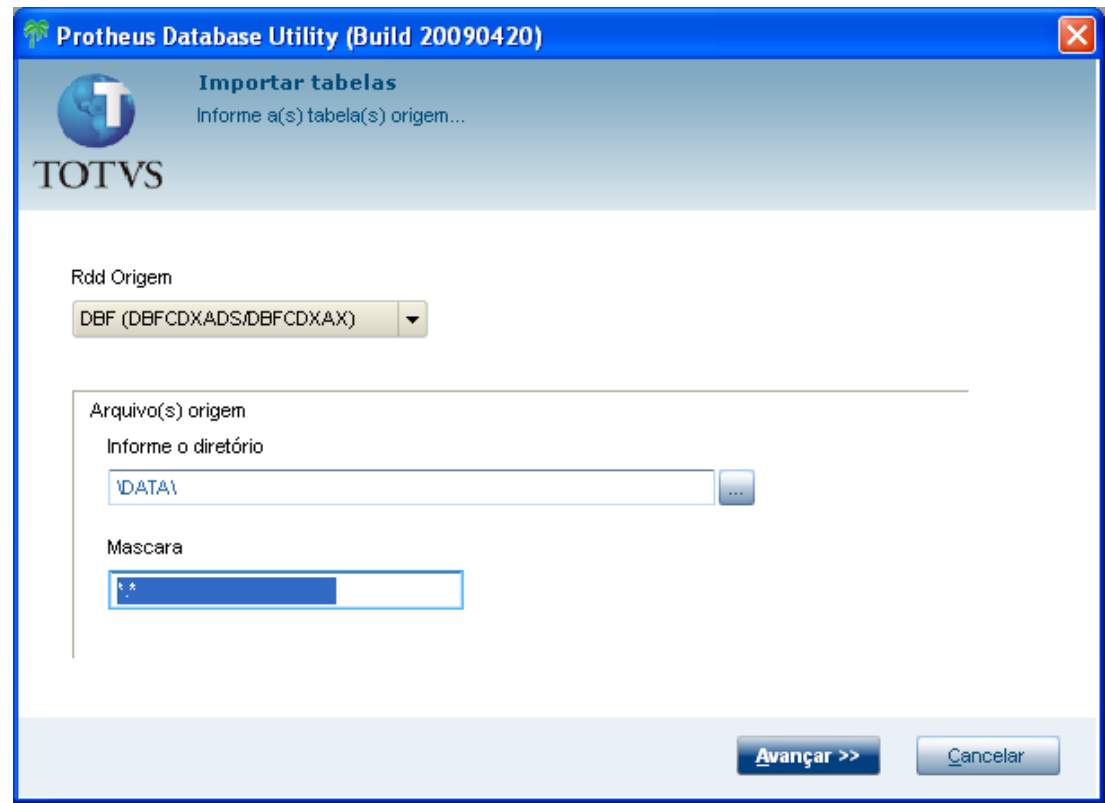
Usaremos “Data”



# Banco de Dados

Protheus - APSDU

Em “Máscara” pode-se definir a extensão que se pretende salvar.



Protheus Database Utility (Build 20090420)

**Importar tabelas**  
Informe a(s) tabela(s) origem...

TOTVS

Rdd Origem  
DBF (DBFCDXADS/DBFCDXAX)

Arquivo(s) origem  
Informe o diretório  
DATA\

Mascara  
\*

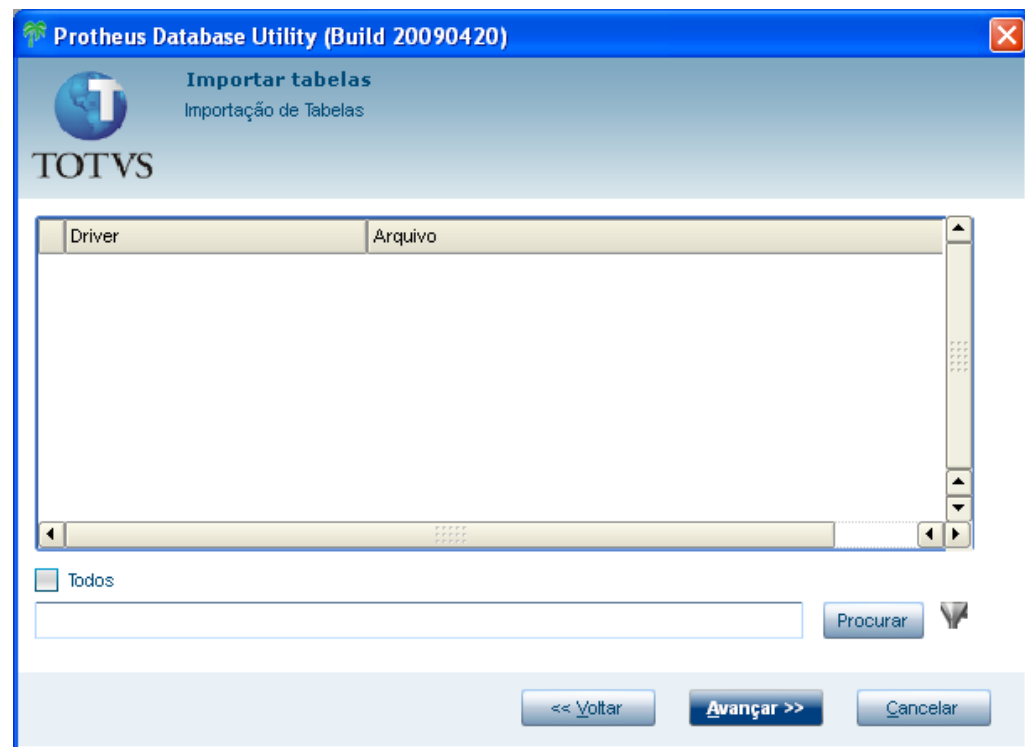
Avançar >> Cancelar

# Banco de Dados

Protheus - APSDU

## APSDU

Aqui selecionaremos os arquivos que queremos importar.

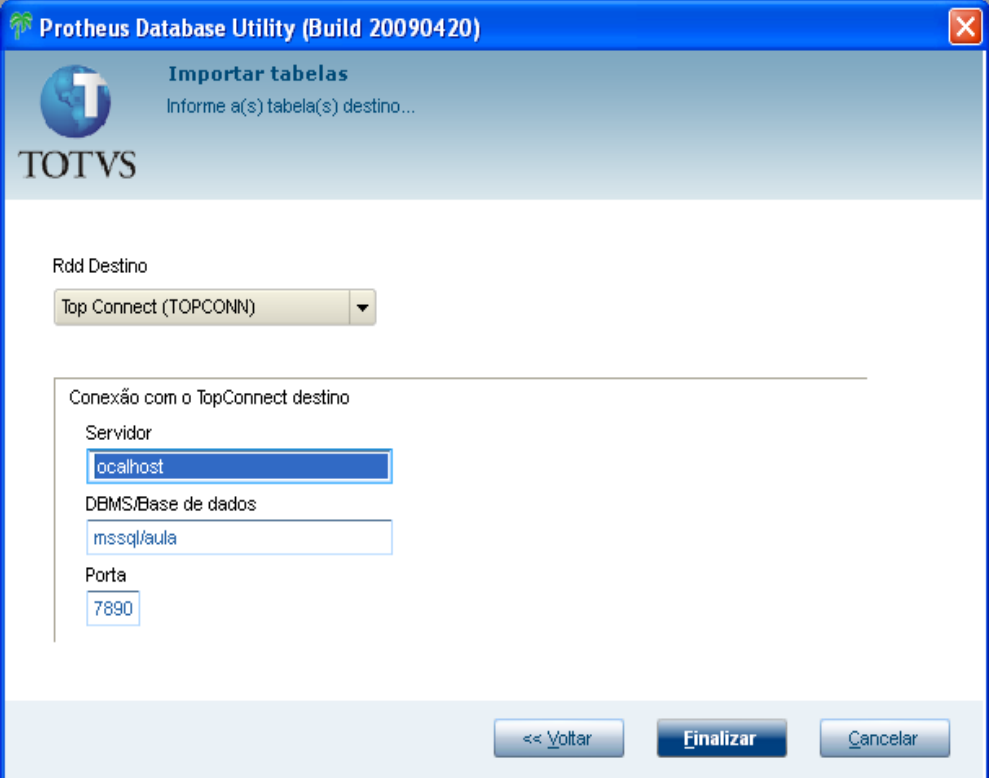


# Banco de Dados

Protheus - APSDU

## APSDU

Escolha a opção TOPCONN, selecione o Servidor, base de Dados e Porta do Ambiente.



Protheus Database Utility (Build 20090420)

**Importar tabelas**  
Informe a(s) tabela(s) destino...

TOTVS

Rdd Destino  
Top Connect (TOPCONN)

Conexão com o TopConnect destino

Servidor  
localhost

DBMS/Base de dados  
mssql/aula

Porta  
7890

<< Voltar Finalizar Cancelar





# **Programando com Banco de Dados**

# Banco de Dados



Diferenças DBF e SQL

Comandos interpretados

Velocidade

Integridade

# Banco de Dados

**TCDELFILE – apaga um arquivo de um banco de dados.**

**Sintaxe**

**TCDELFILE (cTabela)**

**Exemplo:**

**If TCDeFile (“SA1020”)**

**MSGINFO (“Tabela excluída com sucesso”)**

**Else**

**MSGINFO (“Não foi possível excluir tabela”)**

# Banco de Dados

**TCGENQURY - Define a execução de uma Query , a próxima chamada a DBUseArea será a abertura de uma Query e não de tabela**

**Sintaxe:**

**TCGENQRY ([XPar1 , XPar2], cQuery)**

**Exemplo:**

**cQuery := “SELECT X2\_CHAVE, R\_E\_C\_N\_O\_ From SX@990**

# Banco de Dados

**TCSETFIELD – Define formata de campos numéricos e data de acordo com o dicionário.]**

**Sintaxe**

**Tcsetfield (cAlias, cCampo, cTipo, nTam, nDec)**

# Banco de Dados

**RETSQLNAME** - Retorna o nome da tabela de acordo com a empresa  
Posicionada.

Sintaxe

**Retsqlname (cAlias)**

**Exemplo:**

**cQry:= “ Select \* from “+RETSQLNAME(“SZ@”)+ where D\_E\_L\_E\_T\_ = “”\_**

# Banco de Dados



**SQLORDER – Retorna a instrução do índice corrente.**

**Sinntaxe**

**SQLORDER (cChave)**

# Banco de Dados

**TCSQLEXEC – Executa um comando em SQL**

**Sintaxe:**

**TSCQLEXEC (cComando)**

**TCSQLEXEC (“DELETE”+RETSQlname (“SZ2”)+”WHERE D\_E\_L\_E\_T\_ =”“)**



# Banco de Dados

**TCREFRESH** - Faz refresh em uma tabela, através de uma leitura forçada

Da tabela no banco de dados, utilizada após o DELETE e o INSERT.

Sintaxe:

**TCREFRESH** (cTabela)

**EXEMPLO:**

cTabela := "SA1990"

cComando := "Delete"+ cTabela + "WHERE R\_E\_C\_N\_O\_ > 5000"

# Banco de Dados

**TSSPEXIST - Verifica se uma Stored Procedure existe.**

**Sintaxe**

**TSSPEXIST(cStroredProc)**

**Exemplo:**

**IF TCSPEXIST("SP00001")**

**cStr := "Drop Procedure"+"SP00001"**

**TCSqlExec (cStr)**

**Endif**

# Banco de Dados

**TCSPEXEC – Executa uma Store Procedure, no banco de dados, com número variável de parâmetros. Retorna um array contendo os**

**Valores de retorno da SP**

**Sintaxe:**

**TCSPEXEC (cSPProc[xParam1,...xParamN])->[array]**

# Banco de Dados

**Exercícios:**

**Fazer um programa utilizando funções de banco de Dados que imprima os seguintes campos:**

**Campos**

**Nº Pedido, Nome do clientes, Código do Produto, Descrição do produto,  
Data de Emissão do pedido**

**Tabelas de apoio: SC5, SC6, SA1 e SB1**

# Banco de Dados

## **Exercícios:**

**Fazer um programa utilizando funções de bancos de dados.**

## **Parâmetros:**

**Cliente Inicial (SA1)**

**Cliente Final (SA1)**

**Ordena por: Cliente / Produto**

**Analítico/ Sintético**

## **Campos:**

**Nome do cliente, Nº da nota, Data de emissão, Código do produto,**

**Descrição do produto, Quantidade, Valor unitário, total.**

# Banco de Dados

## EMBEDDED SQL

**Begin SQL Alias (Inicia o Embedded SQL )**

**EndSQL (Finaliza a instrução)**

**%XXX% (Instrução literal a ser substituída)**

**Column (Especifica o tipo de conversão de campos)**

**%exp.% (Utilizado para especificar variáveis, expressões e funções)**

**%NoParcer% ( Indica que a consulta não deve passar pela função)**

**%table . <cAlias>% (Indica a tabela a ser utilizada.)**

# Banco de Dados

## Embedded SQL

**%Order . <alias>% (Define a ordem a utilizar no ORDERBY)**

**%Order. <cAlias>, <nIndide>% (Define ordem a utilizar)**

**%Order,<cAlias>, <nNick>% (Define a ordem a utilizar no Order by,**

**Substitui a função )**

**SQLORDER (<Alias>-> (DBNickIndex(cNick)))**

# Banco de Dados

## Exercícios:

Fazer um programa utilizando o Embedded SQL que imprima os seguintes

Campos:

Campos:

Nº Pedido, Nome do Cliente, Código do Produto, Descrição do Produto

Data da Emissão do Pedido.

Tabelas de Apoio: SC5, SC6, SA1 e SB1





# **Orientação a Objetos**

# Orientação a Objetos



**Conjunto de classe pré- definidas ou definidas pelo usuário que possuem Atributos e métodos, e que são instancias em objetos, durante a execução Do programa.**

# Orientação a Objetos

**Classe** representa um conjunto de objetos com diversas Características.

Uma classe define o comportamento dos objetos, através de métodos, quais estados ele é capaz de manter, através de atributos

Exemplo de classe : Empresa.

# Orientação a Objetos



**Objeto é uma Instância de uma classe.**

**Um objeto é capaz de armazenar estados através de seus atributos**

**E reage a mensagens enviadas a ele , assim como se relacionar e enviar mensagens a outros objetos.**

# Orientação a Objetos

Atributos são os dados ou informações do objeto, basicamente a estrutura de dados que vai representar a classe.

Exemplo:

Funcionário: nome , endereço e telefone

Cursos: nome , tempo e preço

# Orientação a Objetos

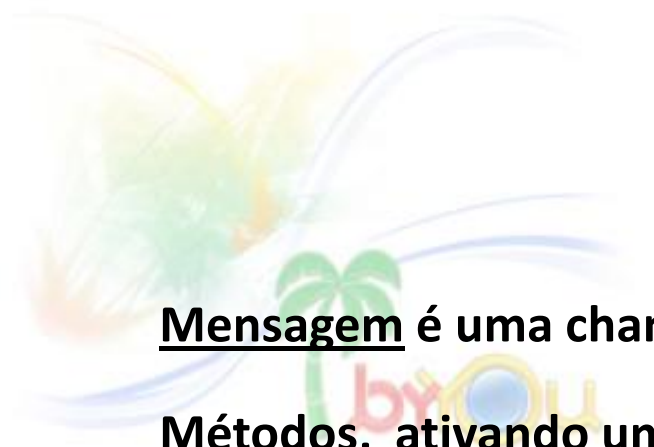
**Métodos definem as habilidades de cada projeto.**

**Normalmente , uma classe possui diversos métodos, que no caso da classe**

**Empresa poderiam ser trina() , atendetelefone().**



# Orientação a Objetos



**Mensagem** é uma chamada de um objeto para invocar um de seus **Métodos**, ativando um comportamento descrito por sua classe.

**Também** pode ser direcionada diretamente a uma classe.

# Orientação a Objetos

**Herança** aproveitamento de métodos e atributos de uma classe superior.

Um exemplo de herança: Filial é super- classe de empresa.

Ou seja, uma filial é uma empresa.



# Orientação a Objetos

**Encapsulamento** consiste na separação de aspectos internos e externos de um objeto,

Este mecanismo é utilizado para impedir o acesso direto aos atributos

De um objeto, disponibilizando externamente apenas os métodos

Que alteram estes estados.

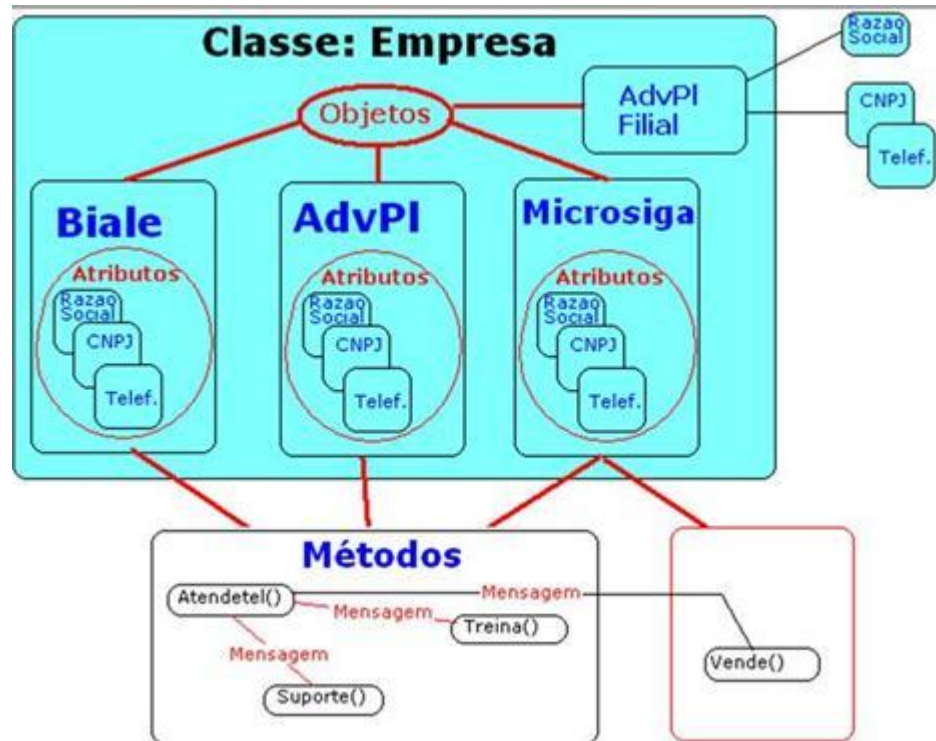
Exemplo: Telefone e suas teclas.

# Orientação a Objetos

**Polimorfismo é o princípio pelo qual duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que têm a mesma assinatura.**

**A decisão sobre qual o método que se deve ser selecionada, de acordo com o tipo da classe derivada, é tomada em tempo de execução.**

# Orientação a Objetos



# Orientação a Objetos

**Exercícios:**

**Crie uma Classe, defina:**

**3 Objetos**

**Propriedades e conteúdos**

**Métodos.**

**Apresente a Classe criada para os outros alunos da sala.**

# Desvendando as Classes



# Desvendando as Classes

## Twindow

**New** - Método construtor da janela.

### Sintaxe

```
New( [anTop], [anLeft],[anBottom], [anRight], [acTitle], [nPar6], [oPar7],;  
      [oPar8],[oPar9], [aoParent], [IPar11], [IPar12], [anClrFore],;  
      [anClrBack], [oPar15], [cPar16], [IPar17], [IPar18], [IPar19],;  
      [IPar20],[alPixel] );
```

**Activate** - Ativa (exibe) a janela. Chamar esse método apenas uma vez.

### Sintaxe

```
Activate([acShow], [bPar2], [bPar3], [bPar4], [bPar5], [bPar6], [ abInit ],;  
        [bPar8], [bPar9], [bPar10], [bPar11], [bPar12] ,[bPar13], ;  
        [bPar14], [bPar15], [abValid], [bPar17], [bPar18] ).
```

**End** – Solicita fechamento da janela.

**Sintaxe:** End()

**Retorno** - Lógico. .T. se encerrou a janela e .F. se não.

**Center** – Centraliza a janela.

**Sintaxe:** Center()

# Desvendando as Classes

## Exemplo

```
#INCLUDE "PROTHEUS.CH"
USER FUNCTION Teste()
Local oWindow
Local abInit:= {||conout("ativando!")}
Local abValid:= {||conout("encerrando!"),.T.}
    oWindow:= tWindow():New( 10, 10, 200, 200, "Meu programa",;
                           ,,,,,, CLR_WHITE,CLR_BLACK,;
                           ,,,,,.T. )
    oWindow:Activate("MAXIMIZED",,,,,,abInit,,,,,,,abValid,,)
// os comandos abaixo proporcionam o mesmo resultado

DEFINE WINDOW oWindow FROM 10, 10 TO 200,200 PIXEL ;
    TITLE "Meu programa" COLOR
    CLR_WHITE,CLR_BLACK
    ACTIVATE WINDOW oWindow MAXIMIZED ON INIT abInit VALID
    abValid
Return NIL
```

# Desvendando as classes

## Exemplo

```
#INCLUDE "PROTHEUS.CH"
USER FUNCTION Teste()
Local oWindow
Local abInit:= {||conout("ativando!")}
Local abValid:= {||conout("encerrando!"),.T.}
    oWindow:= tWindow():New( 10, 10, 200, 200, "Meu programa",;
                           ,,,,,, CLR_WHITE,CLR_BLACK,;
                           ,,,,,.T. )
    oWindow:Activate("MAXIMIZED",,,,,,abInit,,,,,,abValid,,)
// os comandos abaixo proporcionam o mesmo resultado

DEFINE WINDOW oWindow FROM 10, 10 TO 200,200 PIXEL ;
                        TITLE "Meu programa" COLOR
                        CLR_WHITE,CLR_BLACK
                        ACTIVATE WINDOW oWindow MAXIMIZED ON INIT abInit VALID
                        abValid
Return NIL
```



# Desvendando as classes

## Includes

```
#xcommand DEFINE MSDIALOG <oDlg> ;  
    [ <resource: NAME, RESNAME, RESOURCE> <cResName> ] ;  
    [ TITLE <cTitle> ] ;  
    [ FROM <nTop>, <nLeft> TO <nBottom>, <nRight> ] ;  
    [ <lib: LIBRARY, DLL> <hResources> ] ;  
    [ <vbx: VBX> ] ;  
    [ STYLE <nStyle> ] ;  
    [ <color: COLOR, COLORS> <nClrText> [,<nClrBack> ] ] ;  
    [ BRUSH <oBrush> ] ;  
    [ <of: WINDOW, DIALOG, OF> <oWnd> ] ;  
    [ <pixel: PIXEL> ] ;  
    [ ICON <olco> ] ;  
    [ FONT <oFont> ] ;  
    [ <status: STATUS> ] ;  
=> ; //Equivalencia  
<oDlg> = MsDialog():New( <nTop>, <nLeft>, <nBottom>, <nRight>;  
    <cTitle>, <cResName>, <hResources>, <.vbx.>, <nStyle>;  
    <nClrText>, <nClrBack>, <oBrush>, <oWnd>, <.pixel.>;  
    <olco>, <oFont> , <.status.> )
```

# Desvendando as classes

## MSDIALOG

### New

**Descrição** Método construtor da classe.

**Sintaxe** New([anTop], [anLeft], [anBottom], [anRight], [acCaption], [cPar6], ;  
[nPar7], [lPar8], [nPar9], [anClrText], [anClrBack], [oPar12], ;  
[aoWnd], [alPixel], [oPar15], [oPar16], [lPar17])

### Exemplo

```
#INCLUDE "protheus.ch"  
User Function Teste()
```

```
// cria diálogo  
Local oDlg:=MSDialog():New(10,10,300,300,"Meu dialogo",,,,,;  
                           CLR_BLACK,CLR_WHITE,,,.T.)  
    // ativa diálogo centralizado  
    oDlg:Activate(,,,.T.,{||msgstop("validou!"),.T.},,,;  
                  {||msgstop("iniciando...") }  
Return
```

# Desvendando as classes

Private oDlg

Define MSDialog oDlg Title OemToAnsi("Titulo da janela") From 0,0 To 160,380 Pixel

@05,10 To 50,180 Pixel

@15,20 Say "Colocar aqui a mensagem que quiser" Pixel Of oDlg

@25,20 MSGet oVar Var nVar Picture "@E 999,999.99" Size 50,10 Pixel Of oDlg

@70,20 Button oBtnOk Prompt "&Ok" Size 30,15 Pixel ;  
Action (msginfo("Cliquei no OK"), Close(oDlg)) Of oDlg

@70,80 Button oBtnCancel Prompt "&Cancelar" Size 30,15 Pixel ;  
Action (msginfo("Cliquei no Cancelar"), oDlg:End())  
Cancel Of oDlg

Activate MSDialog oDlg Centered

Static Function Close

oDlg:End()

Return

# Desvendando as classes

TFONT

Classe que encapsula fonte de edição.

## **Métodos**

**New** – Construtor do objeto

**Sintaxe**      New([acName], [nPar2], [anHeight],;  
[lPar4], [alBold], [nPar6], [lPar7], [nPar8],; [alItalic], [alUnderline])

# Desvendando as classes

## TFont

### Exemplo usando a classe TFont

```
#include "protheus.ch"
User Function Teste()
Local oDlg, oSay
Local oFont:= TFont():New("Courier New",-14,,.T.)
DEFINE MSDIALOG oDlg FROM 0,0 TO 200,200 ;
    TITLE "Minha tela Courier New" PIXEL
// apresenta o tSay com a fonte Courier New //
oSay:= tSay():New(10,10,{"|"para exibir"}, oDlg,,;
    oFont,,,,.T.,CLR_WHITE,,100,20)

ACTIVATE MSDIALOG oDlg CENTERED
Return
```

# Desvendando as classes

## TFont

### Exemplo usando equivalencia à Classe TFont

```
#include "protheus.ch"
User Function Telafont()
Local oDlg, oSay, oFont
DEFINE FONT oFont Name "Courier New" SIZE 0,-14 BOLD
DEFINE MSDIALOG oDlg FROM 0,0 TO 200,200 ;
    TITLE "Minha tela Courier New" PIXEL

@ 010,010 SAY "para exibir" SIZE 100,20 FONT oFont;
    COLOR CLR_RED PIXEL of ODlg

ACTIVATE MSDIALOG oDlg CENTERED
Return
```

# Desvendando as classes

TFONT



**Protheus.ch**

**Fonts.ch**

**Colors.ch**

# Desvendando as classes

TFONT

Usando Norton Guide

Funcoes.ng -> Clipper em Portugues

Fw16.ng -> Fivewin em Inglês

C53G01C.ng -> Clipper em Inglês



# Classes

Objetos de Controle

Dialog\_objetos.prw

Say

Folder

MSGet

Button / SButton

Box (Label)

ComboBox

CheckBox

Radio

# Busca de diretórios

## Função cGetFile

**cGetFile ( cMask, cTit, nMask , cDirIni, lBut, nBits)**

**cMask -> Mascara para filtro (Ex: "Arquivos Texto|\*.\*R|Tabelas Dbase|\*.DBF")**

**cTit-> Titulo da Janela**

**nMask -> No. Mascara Default-> ( Ex: 1 p/ \*.\*R, 2 p/ \*.\*DBF )**

**cDirIni-> Diretorio inicial se necessario (Ex: C:\ADVPL)**

**lBut-> .T. para mostrar botao como "Salvar" e .F. para botao "Abrir"**

**nBits-> Mascara de bits para escolher as opcoes de visualizacao do Objeto.**

<b>GETF_OVERWRITEPROMPT</b>	<b>1</b>	<b>Retorna o nome do arquivo</b>
<b>GETF_LOCALFLOPPY</b>	<b>8</b>	<b>Mostra as unidades flexiveis</b>
<b>GETF_LOCALHARD</b>	<b>16</b>	<b>Mostra as Unidades Locais</b>
<b>(c:)</b>		
<b>GETF_NETWORKDRIVE</b>	<b>32</b>	<b>Mostra as unidade de Rede (F:</b>
<b>G: Etc)</b>		
<b>GETF_RETDIRECTORY</b>	<b>128</b>	<b>Retorna o diretório escolhido</b>

# Busca de diretórios

## Diretório

```
cTitle:= "Escolha o Arquivo"  
nBits := GETF_RETDIRECTORY  
cDir:=AllTrim(cGetFile(, cTitle,,,.T.,nBits))
```

## Arquivo

```
cExt := "Arquivos Texto | *.*##R | Tabelas Dbase | *.DBF"  
cTitle:= "Escolha o Arquivo"  
nBits := GETF_OVERWRITEPROMPT  
cArq:=AllTrim(cGetFile(cExt,cTitle,,,.T.,nBits))
```

# Classes

## Usando o Gaia

Fazer o Exercício da página 21

cUsername - Nome do usuário

Funções de apoio:

AADD(aMatriz, cConteúdo)

dbSetOrder(nOrd)

dbSeek(cChave, lAcha)

EOF()

While cCondição

RecLock(cAlias, lNovo)

(cAlias)->nome\_campo := "0015"

MSUnlock()

# Montando as Telas



# Montando Telas

## LISTBOX

- **Lista simples:** lista de apenas uma coluna no formato de um vetor, a qual não necessita da especificação de um cabeçalho.

```
#INCLUDE "PROTHEUS.CH"
User Function listbox1()
Local OLISTBOX
Private nLista := 2
Private aLista := {"Linha 1", "Linha 2"}
DEFINE MSDIALOG _oDlg TITLE "ListBox" FROM (210),(210) ;
    TO (468),(520) PIXEL
    @ 05,05 LISTBOX oList VAR nLista ITEMS aLista;
    PIXEL SIZE 100,100 OF _oDlg;
    ON CHANGE MSGINFO(nLista)
ACTIVATE MSDIALOG _oDlg CENTERED
Return()
```

# Telas

## Padrão Microsiga



# Telas Padrão Microsiga



- mBrowse(nTop, nLeft, nBot, nRig, cAlias)
- Modelo2()
- Modelo3()



# Telas Padrão Microsiga

## MBROWSE

- Premissas para o Browse
  - criar variavel private cCadastro do tipo Character
  - criar variavel private aRotina do tipo Array
  - a chamada da funcao mbrowse envia para a funcao
  - 3 argumentos: -
    - alias()
    - recno()
    - Opcao do browse
  - A funcao deve receber esses argumentos:
    - EX: funcao(cAlias, nReg, nOpc)

# Telas Padrão Microsiga

## Exercício

- Fazer um browse pelo assistente de código e deixar as opções:
  - Pesquisa (axpesqui)
  - Visualiza(axvisual)
  - GAIA (colocar a user function do exercício da página 16)



# Telas Padrão Microsiga

Modelos 2 e 3

- Diferença:
  - Modelo2 – Monta tela com capa e item, onde o Alias para capa e item é o mesmo
  - Modelo3 – Monta tela com capa e item, onde o Alias para capa é um e para item é outro.

# Telas Padrão Microsiga

## Modelos 2 e 3

- Passos para construção

1. Criar variavel private aHeader do tipo Array

2. Criar variavel private aCols do tipo Array

3. Preparar variaveis de memoria dos campos de acordo com o dicionario

4. Montar aHeader de acordo com o dicionario

5. Montar aCols de acordo com o dicionario

6. Chamar a funcao modelo3

# Telas Padrão Microsiga

## Monta aHeader

```
SX3->(dbSetorder(1))
SX3->(dbSeek("SC6" ))
While SX3-> (!EOF() .AND. SX3->X3_ARQUIVO == "SC6")
    If X3USO(SX3->X3_USADO) .AND. cNIVEL >= SX3->X3_NIVEL
        AAdd (aHeader, {Trim(SX3->X3_TITULO),;
            SX3->X3_CAMPO      ;;
            SX3->X3_PICTURE   ;;
            SX3->X3_TAMANHO    ;;
            SX3->X3_DECIMAL    ;;
            SX3->X3_VALID      ;;
            SX3->X3_USADO      ;;
            SX3->X3_TIPO       ;;
            SX3->X3_ARQUIVO    ;;
            SX3->X3_CONTEXTO})
    ENDIF
    SX3->(DBSKIP())
ENDDOs
```

# Telas Padrão Microsiga

Modelos 2 e 3

## Monta aCols

```
Aadd (aCols, Array(Len(aHeader)+1))  
For nAdi:= 1 to Len(aHeader)  
  aCols[1] [nADI] := Criavar(aHeader[nADI][2])  
Next  
ACols[1][len(aHeader)+1] := .F.
```

# Telas Padrão Microsiga

## Chama a função modelo3

lRet := Modelo3(cCadastro, "SC5", "SC6", , ".T.", ".T." , nOpc, nOpc)



# Telas Padrão Microsiga

- Sintaxe

Modelo3(cCadastro, cAlias1, cAlias2, [aCpoEnchoice], cLinOK, ;  
cTudoOK, nOpcE, nOpcG, [cFieldOK,] [IVirtual], ;  
[nLinhas], [aAltEnch], [nFreeze])

- cCadastro – Titulo da Janela
- cAlias1 – alias da enchoice
- cAlias2 – Alias da getdados
- aCpoEnchoice – Campos da enchoice, Se nulo obedece dicionario
- cLinOk - Validacao da linha na Getdados
- cTudook – Validacao no Ok
- nOpcE - Opcao da Enchoice
- nOpcG – Opcao da Getdados
- cFieldOk – Validacao dos campos da enchoice, Se nulo obedece dicionario
- IVirtual – Aceita campos Virtuais , Se nulo .T.
- nLinhas – Qtde de Linhas da Acols, , Se nulo, qtde de linhas ilimitadas
- aAltEnch - Campos que podem ser alterados na enchoice, Se nulo obedece dicionario
- nFreeze – Campos que devem ser congelados



# Telas Padrão Microsiga

- Exercício

Criar uma tela com 2 Gets com consulta padrão para o SX2

- 1- Caracter 3 posicoes para o alias da enchoice

- 2- Caracter 3 posicoes para o alias da getdados

2 botoes:

- monta Arrays (chama a função de browse criada no exercicio anterior
- cancela.

Criar uma rotina no browse para incluir dados utilizando a enchoice e getdados informada pelo usuario, utilizando a função modelo3.

# Telas Padrão Microsiga

- Exercício

Inserir 3 itens no arotina do browse feito no exercício anterior para:

- Alterar ,visualizar , excluir os dados

- Preencher aheader e acols na getdados informada pelo usuario,

- Utilizar a função modelo3,

- Trazer o acols já preenchido conforme registro posicionado pelo usuario.



# Relatórios Graficos

# Relatórios Graficos

## Classe TMSPrinter

//Cria um novo relatorio  
oPrn := TMSPrinter():New(cTitulo)

//Seleciona modo paisagem  
oPrn:SetLandscape()

//Inicializa uma página nova  
oPrn:StartPage()

//Finaliza a pagina  
oPrn:EndPage()

//Encerra o objeto  
oPrn:End()

# Relatórios Graficos

## Classe TMSPrinter

//Cria uma linha na Horizontal  
oPrn:Line(0150,1050,0150,3330)

//Cria uma linha na Vertical  
oPrn:Line(0150,2570,0821,2570)

//Cria um texto fixo a ser impresso  
DEFINE FONT oFnt NAME "Arial" SIZE 0,16 OF oPrn BOLD  
oPrn:Say(0080,0080,"Mensagem a ser impressa",oFnt)

//Cria um bitmap a ser impresso  
oPrn:SayBitmap(0405,0800,"AUTENTIC.BMP",30,310)

# Relatórios Graficos

## Classe TMSPrinter

//Cria um Box

oPrn:Box(0080,1050,1101,3330)

//Mostra uma previa da impressão na tela

oPrn:Preview()

//Mostra tela de Configuração de impressão

oPrn:Setup()

//Imprime o que já estiver instanciado

oPrn:Print()

# Relatórios Graficos

## Exercício

Incluir um relatorio do Cadastro de Produtos com quebra por tipo, a cada quebra deverá ter uma separação com uma linha (usem o método `line()` )

- a. Logo a esquerda
- b. Data
- c. Hora
- d. Pagina
- e. Cabecalho em negrito e separado por linhas(`line()`)
- f. Detalhamento da quebra (vermelho)
- g. Coloquem esse programa tambem no browse.

# Usando um ponto de Entrada

**Utilizado como recurso para customização em aberturas do sistema.**

Exemplo:

Após a Inclusão do Produto: MT010INC

user function MT010INC()

msginfo("Voce acabou de incluir um produto")

Return



# Usando um ponto de Entrada

Antes da exclusão do Produto: MTA010OK

Retorno .T. – Permite excluir

Retorno .F. – Não Permite Excluir

User function MTA010OK()

Local lRet := .T.

    If dDataBase < ctod("01/01/2008")

        msginfo("Voce não pode excluir esse produto")

        lRet := .F.

    Endif

Return lRet



# Funções de Servidor

# Funções de Servidor

GetRemoteIniName()-

Retorna o nome do arquivo de configuracao remote.

GetEnvServer() -

Retorna o ambiente atual

GetsrvProfString("ROOTPATH", "C:\")

Retorna o conteudo do parametro da seção

Exemplo:

```
cTemProc := "Menu: " + CARQMNU + CRLF //Nome do menu
```

```
cTemProc += "Modulo: " + cModulo+ CRLF // Modulo atual
```

```
cTemProc += "Ambiente: " + Getenvserver() + CRLF //Ambiente atual
```

```
cTemProc += "Usuario: " + cUserName + CRLF //Nome Usuario
```

```
cTemProc += "Programa: " + funname() + CRLF //Funcao chamada Menu
```

# Funções de Servidor

**Funname(n)**- Retorna a função da pilha, onde n é a posição na pilha.

**Conout(cMens)** – Imprime cMens no console do Server

**GetFuncArr("A")** – Retorna as funções do repositório.

**GetFuncPrm(cFun)** – Retorna parâmetros da função especificada.

**GetTheme()** – Retorna o tema escolhido.

**GetDBExtension()** – Retorna a extensão da Base (DBF, DTC, etc)

**OrdBagExt()** – Retorna a extensão do Índice(CDX, IDX, NTX, etc)

# Funções de Servidor

**CPYS2T - Cópia arquivos do servidor para a máquina do cliente.**

`CPYS2T(cArqOri, cArqDes, lComp)`

**cArqOri** - Arquivo de Origem, visível a partir do rootpath do ambiente corrente

**cArqDest** - Arquivo de Destino, visível na máquina do cliente

**lComp** - Define se compacta antes de enviar o arquivo.

**CPYT2S - Cópia arquivos da máquina do cliente para o servidor.**

`CPYT2S(cArqOri, cArqDes, lComp)`

**cArqOri** - Arquivo de Destino, visível na máquina do cliente

**cArqDest** - Arquivo de Origem, visível a partir do rootpath do ambiente corrente

**lComp** - Define se compacta antes de enviar o arquivo.

# Funções de Servidor

- **Exemplo:**

```
PREPARE ENVIRONMENT EMPRESA '99' FILIAL '01' MODULO 'FAT';  
TABLES "SA1", "SA2","SA3"
```

Ou

```
RpcClearEnv() //LIMPA O AMBIENTE
```

```
RpcSetType( 3 ) //Nao utiliza licenca
```

```
RpcSetEnv("99","01",cUser,cSenha,"FAT",,{"SA1", "SA2","SA3"})
```

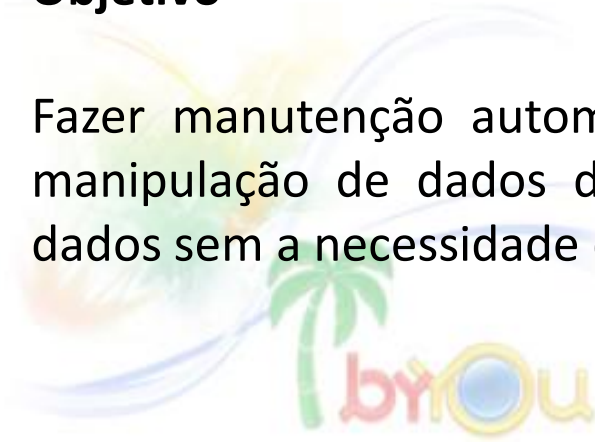


# **Rotinas Automáticas**

# MSEXECAUTO

## Objetivo

Fazer manutenção automática (inclusão, alteração e exclusão) das rotinas de manipulação de dados do sistema, automatizando o processo de entrada de dados sem a necessidade de desenvolver rotinas específicas.





# MSEXECAUTO

## Sintaxe:

MSExecAuto({|x,y|fpgmpad0(x,y)},aCampos,nOpc)

- fPgmPad = Nome da Função Padrão
- aCampos = Array com os campos a gravar
- nOpc = Opção que será utilizada.

# MSEXECAUTO

## Exemplo:

```
User Function fCadForn()
Local aVetor := {}
lMsErroAuto := .F.
aVetor := { {"A2_COD"      , "999999"      , nil}, ;
            {"A2_LOJA"     , "01"         , nil}, ;
            {"A2_NOME"     , "Fornecedor Teste" , nil}, ;
            {"A2_NREDUZ"   , "Teste F"         , nil}, ;
            {"A2_END"      , "Rua teste"        , nil}, ;
            {"A2_MUN"      , "teste"            , nil}, ;
            {"A2_EST"      , "SP"               , nil}}
MSExecAuto({|x,y| Mata020(x,y)},aVetor,3) //Inclusao
If lMsErroAuto
    Alert("Erro")
    mostraerro()
Else
    Alert("Ok")
Endif
Return
```

# MSEXECAUTO

## EXERCÍCIOS

Criar uma rotina que inclua automaticamente pelo MSEXECAUTO um pedido de vendas com 2 itens.

Coloque essa rotina no Browse do Exercício anterior.

Apoio: ROT\_AUTO.PRW

Funcao de inclusao:mata410()

Tabelas: SC5 e SC6



# Integração Excel

# Integração Excel

## **TSTEXCEL.PRW**

(insira o pedido 000019 para conseguir simular esse programa)

## **DLGTOEXCEL( aArray)**

aArray := Array multidimensional que define o que se quer inserir no excel, onde o 1º. Elemento é o tipo de inserção

- CABEÇALHO – Insere a parte superior no Excel
- GETDADOS – Insere diversos itens
- ARRAY – Insere um array qualquer

-Ex:

DlgToExcel({ {"ARRAY", "Exporta Excel", aCabec, aDados} })

# Integração Excel

## Usando APOLEClient()

//Verifica se O Excel está instalado e se Conecta

```
If !ApOleClient("MSEExcel")
```

```
MsgAlert("Microsoft Excel não instalado!")
```

```
Return
```

```
EndIf
```

//Cria o Objeto de Integração com Excel

```
oExcel := MSEExcel():New()
```

```
oExcel:WorkBooks:Open("C:\ARQUIVO.CSV")
```

```
oExcel:SetVisible(.T.)
```

```
oExcel:Destroy()
```

# Integração Excel

## Exercício

Fazer uma consulta usando instrução em SQL e com o resultado da consulta gerar integração com o Excel.

Campos: Num. Pedido , Nome Cliente, Estado, Condição pagamento

Tabelas de Apoio: SC5 e SA1

Perguntar ao usuario se ele prefere resultado em DBF ou CSV.



# Integração Word



# Integração Word

//Conecta ao word

hWord := OLE\_CreateLink()

OLE\_NewFile(hWord, "c:\arquivo.dot" )

//Montagem das variaveis que serão lidas no Modelo DOT

OLE\_SetDocumentVar(hWord, 'nVar1', 15)

OLE\_SetDocumentVar(hWord, 'dData1', date())

//Executa Macro no word

OLE\_ExecuteMacro(hWord,"macro1")

# Integração Word

// Atualizando as variaveis do documento do Word  
OLE\_UpdateFields(hWord)

//Imprime no word  
OLE\_PrintFile(hWord,"ALL",,,1)

//Fecha o arquivo no word  
OLE\_CloseFile( hWord )

//Fecha o link no word  
OLE\_CloseLink( hWord )



# Envio de E-MAIL

# Envio de E-MAIL

## Parâmetros que contem configuração de conexão

```
Private cSrv := GetMV("MV_RELSERV") // smtp.ig.com.br
Private cMail := GetMV("MV_RELACNT") //
meuemail@ig.com.br
Private cPass := GetMV("MV_RELPSW") // 123abc
Private lAuth := GetMv("MV_RELAUTH") //Requer
Autenticação?
Private cPswA := GetMV("MV_RELAPSW") // 123abc
```

Ou

```
Private cSrv := GetMV("MV_WFSMTP") // smtp.ig.com.br
Private cMail:= GetMV("MV_WFACC") // meuemail@ig.com.br
Private cPass:= GetMV("MV_WFPASSW") // 123abc
Private lAuth := GetMv("MV_RELAUTH") //Requer
Autenticação?
Private cPswA := GetMV("MV_RELAPSW") // 123abc
```

# Envio de E-MAIL


```
//Conecta ao Servidor SMTP
CONNECT SMTP SERVER cSrv;           //Nome do servidor SMTP
                                ACCOUNT cMail;       //Conta de Email
                                PASSWORD cPass;      //Senha de conexão
                                RESULT lResul         //Resultado da Conexão

//Retorna se conseguiu fazer autenticação
lok := MailAuth(cEmail,cPass)

// Abre janela perguntando o usuario e senha para fazer
autenticacao
lok := QAGetMail()

//Atribui retorno de envio de email na variável cError
GET MAIL ERROR cError
```

# Envio de E-MAIL



```
//Envio de email  
SEND MAIL FROM cDe ;  
TO cPara;  
SUBJECT cAssunto;  
BODY cMsg;  
CC cCC;  
CCO cCCO;  
ATTACHMENT cAnexo;  
RESULT lSend
```

```
//Desconecta do servidor  
DISCONNECT SMTP SERVER
```

Veja programa EMAIL.PRW



## PROJETO

Entregar o projeto em 30 dias.

Nota Minima para certificado: 7,50

Esclarecer Dúvidas por email com o instrutor.

**BOA SORTE!!!**