

Tarefa Orientada 8

Criação de tabelas

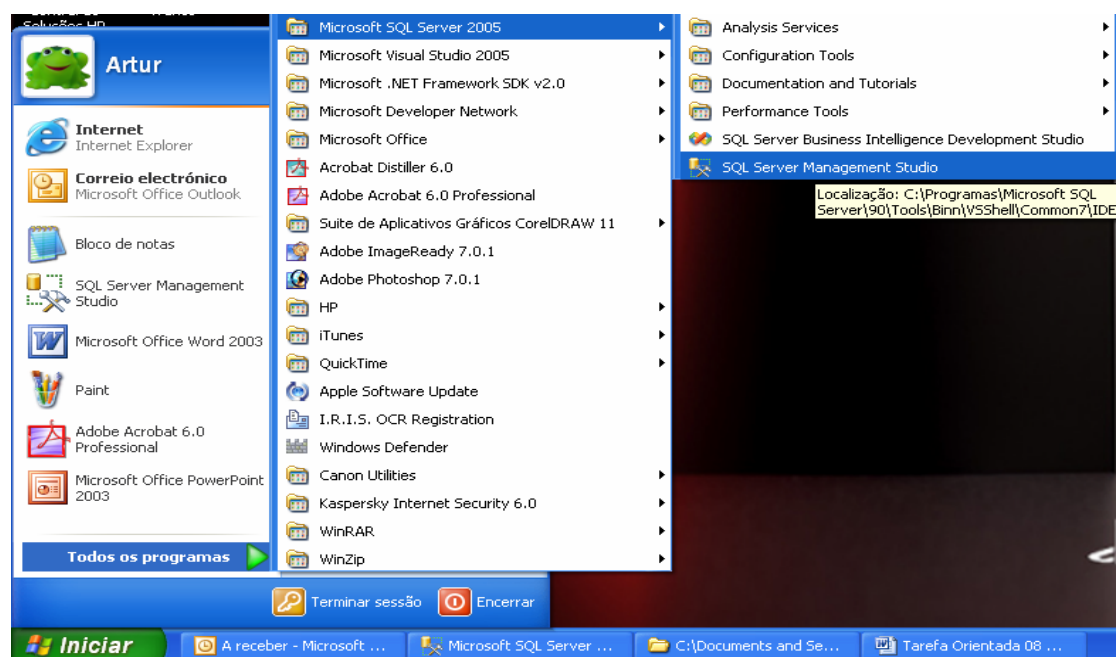
Objectivos:

- Iniciar o MS SQL Server Management Studio
- Criar tabelas
- Especificar tipos de dados
- Definir restrições
- Alterar as definições de uma tabela
- Eliminar uma tabela

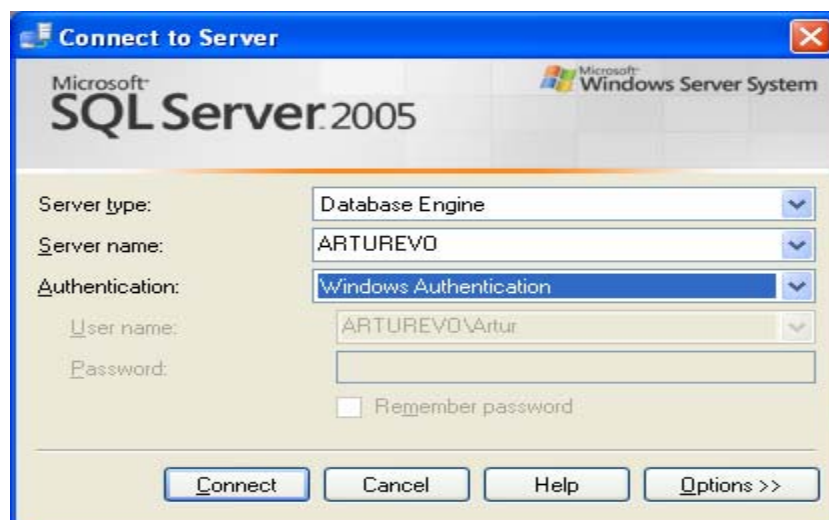
O SGDB *MS SQL Server 2005* consiste, basicamente, num servidor de bases de dados, que providencia os serviços para gerir bases de dados, e num conjunto de ferramentas, que providenciam a interface para trabalhar com as bases de dados. Deste conjunto, o *SQL Server Management Studio* é a principal ferramenta para trabalhar com as bases de dados.

Iniciar o MS SQL Server Management Studio

1. Inicie o *SQL Server Management Studio* através do menu *INICIAR*.

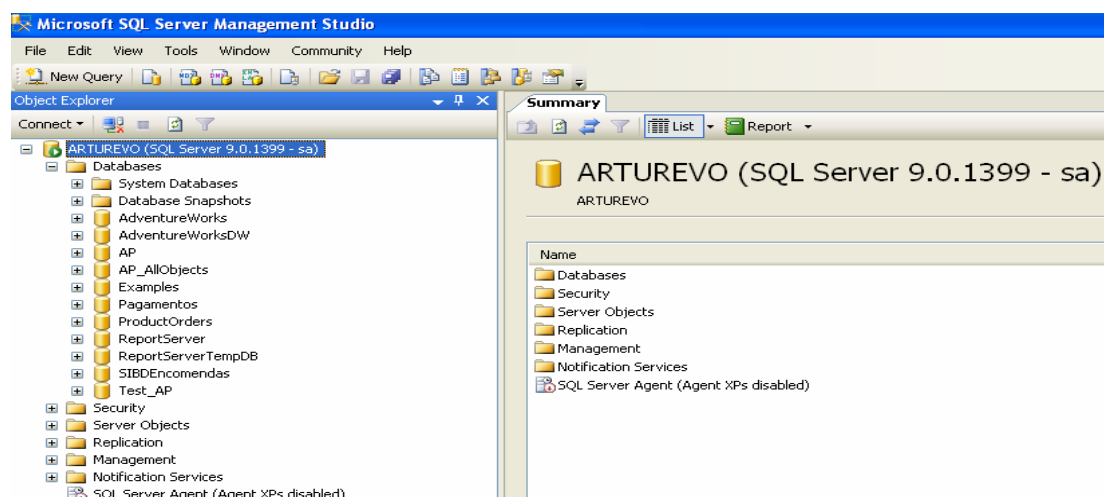


2. Ao abrir o *SQL Server Management Studio* aparece a seguinte caixa de diálogo.



3. Em *Server name* coloque o valor *diserver2*.
4. Em *Authentication*, seleccione a opção *Windows Authentication*.
5. Pressione o botão *Connect*.

Vai aparecer a janela com o ambiente de trabalho do *SQL Server Management Studio*.

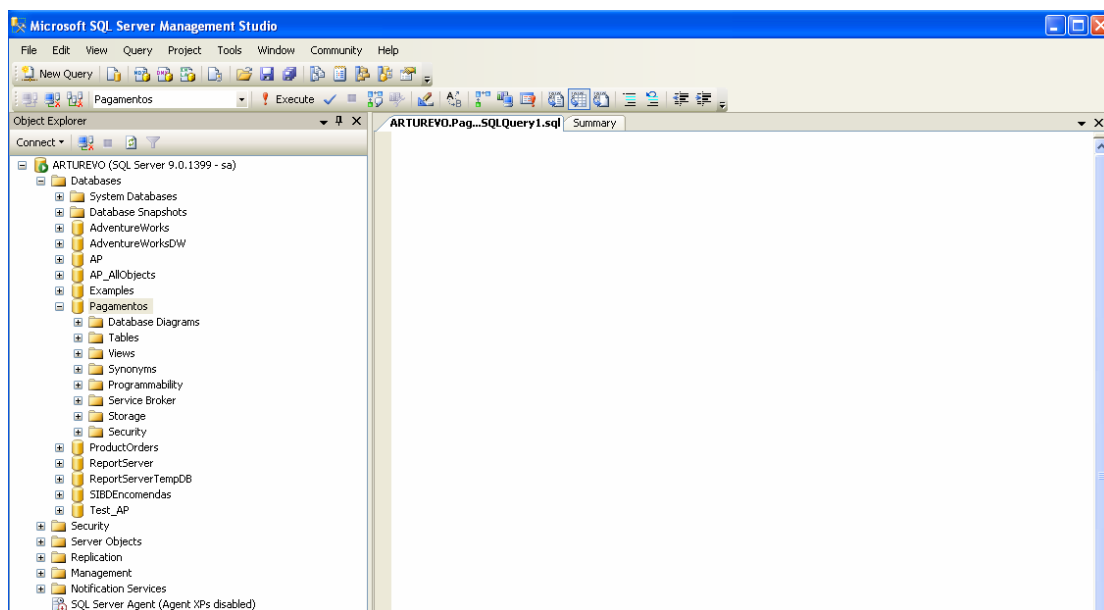


6. Expanda a pasta *Databases* e seleccione a base de dados que lhe está destinada.

7. Na barra de ferramentas, pressione o botão *New Query*.





No lado direito do *SQL Server Management Studio* vai aparecer o editor de consultas.



ATENÇÃO!

Sempre que quiser formular uma consulta, verifique se, na caixa de combinação da barra de ferramentas , está seleccionada a base de dados que pretende.

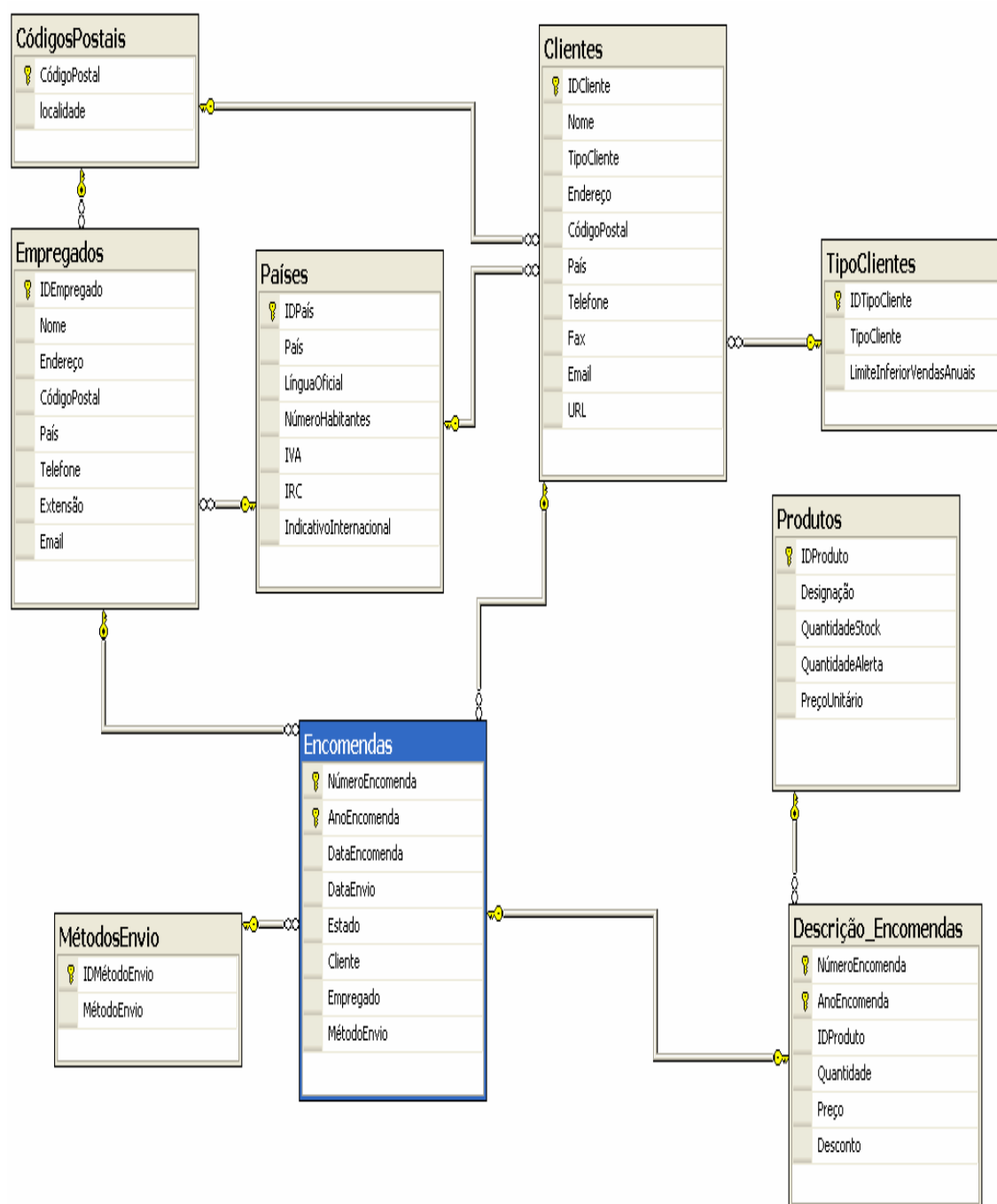
Para executar um comando SQL, pressione o botão  .

No caso de ter vários comandos SQL formulados no editor e só desejar executar um determinado comando, seleccione esse comando e pressione o botão  .

Criar as tabelas

Vamos agora criar as tabelas que fazem parte da base de dados Encomendas.

8. Utilize o editor “aberto” no passo 7 para criar as seguintes tabelas, de acordo com as notas que se seguem.



9. Durante a criação das tabelas defina ainda as seguintes restrições de integridade.

Tabela TipoClientes

Campo *tipoCliente* - não pode receber valores nulos

Tabela CódigosPostais

Campo *localidade* - não pode receber valores nulos

Tabela Países

Campos *País*, *IVA*, *IRC* - não podem receber valores nulos

Tabela Clientes

Campos *Nome*, *TipoCliente*, *Endereço*, *CódigoPostal* e *País* - não podem receber valores nulos

Tabela Empregados

Campos *Nome*, *Endereço*, *CódigoPostal* e *País* - não podem receber valores nulos

Tabela MétodosEnvio

Campo *MétodoEnvio* - não pode receber valores nulos

Tabela Encomendas

Campo *AnoEncomenda* - deve apresentar como valor predefinido a data do sistema. Para obter a data do sistema pode usar a função *GETDATE()*.

Campo *DataEncomenda* - não pode receber valores nulos e deverá apresentar como valor predefinido a data do sistema. Para obter o ano da de uma data utilize a função *YEAR()*. Especifique ainda que este campo não pode receber valores anteriores à data do sistema.

Campo *estado* - não pode receber valores nulos e deverá apresentar como valor predefinido o valor 'P'. Especifique ainda que este campo apenas pode receber os valores 'P' ou 'D'.

Campo *Envio* - não pode receber valores nulos e não pode receber valores anteriores à data de encomenda.

Campos *Cliente*, *empregado* e *MétodoEnvio* - não podem receber valores nulos.

Tabela Produtos

Campos *Designação*, *QuantidadeStock*, *QuantidadeAlerta* e *PreçoUnitário* - não podem receber valores nulos.

Tabela Descrição_Encomendas

Campos *IDProduto*, *Quantidade*, *Preço* e *Desconto* - não podem receber valores nulos.

Campo *Desconto* - deverá apresentar como valor predefinido o valor 0.

Para criar uma tabela, utilize a seguinte sintaxe.

```
CREATE TABLE nome_tabela  
( nome_coluna1 tipo_dados [restrições_coluna],  
 nome_coluna2 tipo_dados [restrições_coluna],  
 ... ,  
[restrições_tabela] )
```

Durante a criação de uma tabela pode utilizar as seguintes restrições.

PRIMARY KEY - para especificar que, em cada registo, a(s) coluna(s) tenha(m) um valor único e não nulo.

NOT NULL - para impedir que sejam armazenados valores nulos (NULL) na coluna.

UNIQUE - para especificar que, em cada registo, a coluna deve ter valores distintos.

DEFAULT - para definir, em cada novo registo, um valor predefinido para a coluna (que pode ser obviamente alterado).

CHECK - para limitar os valores admissíveis para a coluna.

[FOREIGN KEY] REFERENCES - para impor a integridade referencial entre a(s) coluna(s) na nova tabela e a(s) coluna(s) de uma tabela relacionada.

De modo a perceber mais facilmente as mensagens de erro durante a execução de instruções SQL ou a poder identificar as restrições numa instrução *ALTER TABLE*, pode definir nomes para as restrições de integridade que utiliza na criação das tabelas, através da inclusão da palavra-chave **CONSTRAINT** seguida do nome da restrição.

Também pode utilizar a palavra-chave **IDENTITY** na definição de uma coluna, de modo a que seja o *SQL SERVER* a gerar os valores para a coluna. Note, contudo, que apenas pode existir uma coluna *IDENTITY* por tabela.

A seguir, apresentam-se alguns dos tipos de dados que poderá utilizar durante a criação de uma tabela.

Tipos de dados inteiros

bigint - para valores inteiros entre -9.223.372.036.854.775.808 e 9.223.372.036.854.775.807

int - para valores inteiros entre -2.147.483.648 e 2.147.483.647

smallint - para valores inteiros entre -32.768 e 32.767

tinyint - para valores inteiros positivos entre 0 e 255

bit - para os valores inteiros 0 ou 1

Tipos de dados decimais

decimal[(p[,s])] - para valores decimais com precisão fixa (p) e escala (s) desde $-10^{38} + 1$ até $10^{38} - 1$. A precisão pode ser qualquer número entre 1 e 38; a precisão predefinida é 18. A escala pode ser qualquer número entre 0 e a precisão. A precisão predefinida é 0.

Numeric[(p[,s])] - sinónimo de *decimal*.

money - para valores monetários, com quatro casas decimais, desde -922.337.203.685.477,5808 até 922.337.203.685.477,5807. Sinónimo de *decimal(19,4)*.

Smallmoney - para valores monetários, com quatro casas decimais, desde -214.748,3648 até 214.748,3647. Sinónimo de *decimal(10,4)*.

Tipos de dados reais

float[(n)] - para valores com virgula flutuante com precisão dupla desde -1.79×10^{308} até 1.79×10^{308} . o valor *n* representa o número de *bits* utilizados para armazenar a parte decimal do número (a mantissa); o seu valor predefinido é 53. *n*= 24 representa precisão simples. *n*=53 representa precisão dupla.

Real - para valores com virgula flutuante com precisão simples desde -3.4×10^{38} até 3.4×10^{38} . Sinónimo de *float(24)*.

Tipos de dados alfanuméricos standard

char[(n)] - para valores alfanuméricos de tamanho fixo até um limite de *n*=8000 caracteres. O valor predefinido é 1.

Varchar[(n)] - para valores alfanuméricos de tamanho variável até um limite de *n*=8000 caracteres. O valor predefinido é 1. Note que o número de bytes utilizado para armazenar a cadeia de caracteres depende do seu tamanho.

Tipos de dados alfanuméricos Unicode

nchar(n) - para valores alfanuméricos *unicode* de tamanho fixo até um limite de *n*=4000 caracteres. O valor predefinido é 1.

Nvarchar(n) - para valores alfanuméricos *unicode* de tamanho variável até um limite de *n*=4000 caracteres. O valor predefinido é 1. Note que o número de bytes utilizado para armazenar a cadeia de caracteres depende do seu tamanho e que são necessários 2 bytes para armazenar cada carácter *unicode*.

Tipos de dados para datas e horas

datetime - para valores com datas e horas desde 1 de Janeiro de 1753 até 31 de Dezembro de 9999, com uma precisão de 3,33 milissegundos.

Smalldatetime - para valores com datas e horas desde 1 de Janeiro de 1900 até 6 de Junho de 2079, com uma precisão de 1 minuto.

Tipos de dados para grande capacidade de armazenamento

varchar(ax) - similar ao tipo de dados *varchar* anteriormente apresentado. Contudo, devido à utilização de especificador *ax* permite o armazenamento de dados até 2.147.483.648 bytes (aprox 2GB).

Nvarchar(ax) - similar ao tipo de dados *nvarchar* anteriormente apresentado. Contudo, devido à utilização de especificador *ax* permite o armazenamento de dados até 2.147.483.648 bytes (aprox 2GB).

Varbinary(ax) - para armazenar dados binários de tamanho variável até 2.147.483.648 bytes (aprox 2GB). O número de bytes usado para armazenar os dados depende do tamanho dos dados.

Alterar a definição de uma tabela

Pode utilizar a instrução *ALTER TABLE* para modificar a definição de uma tabela, nomeadamente adicionar colunas ou restrições, apagar colunas ou restrições, modificar a definição de uma coluna, incluindo, por exemplo, a alteração do tipo de dados.

Todavia, o *SQL SERVER*, antes de permitir as alterações, faz uma verificação, de modo a garantir que não se perdem nenhuns dados. Se o resultado da verificação for negativo, o *SQL SERVER* não autoriza a alteração. Por exemplo, não é permitido eliminar colunas que façam parte da chave primária de uma tabela, que sejam usadas em restrições *CHECK* ou *FOREIGN KEY*, ou que tenham um índice baseado nelas.

Pode modificar a definição de uma coluna, de modo a que passe a permitir o armazenamento de valores nulos, desde que nenhuns dos registos já armazenados contenham valores nulos para essa coluna.

A Cláusula *ALTER COLUMN* apenas permite alterar o tipo de dados ou os atributos *NULL* e *NOT NULL* de uma coluna.

Quando adiciona uma nova restrição, o *SQL SERVER* vai verificar se os dados já existentes estão de acordo com a nova restrição. Todavia, se não desejar que isso aconteça, inclua as palavras-chave *WITH NOCHECK* na instrução *ALTER TABLE*.

Para alterar a definição de uma tabela, utilize a seguinte sintaxe.

```
ALTER TABLE nome_tabela [WITH CHECK | WITH NOCHECK]
    {ADD nome_nova_coluna tipo_dados [restrições_coluna] |
    DROP COLUMN nome_coluna|
    ALTER COLUMN nome_coluna novo_tipo_dados [NULL | NOT NULL] |
    ADD [CONSTRAINT] nova_definição_de_restrição |
    DROP [CONSTRAINT] nome_restrição }
```

Exemplos de instruções *ALTER TABLE*

Instrução que adiciona uma coluna

```
ALTER TABLE Fornecedores  
ADD e-mail VARCHAR(30) NULL
```

Instrução que elimina uma coluna

```
ALTER TABLE Fornecedores  
DROP COLUMN e-mail
```

Instrução que adiciona uma nova restrição *CHECK*

```
ALTER TABLE Facturas WITH NOCHECK  
ADD CHECK (TotalFactura >= 1)
```

Instrução que adiciona uma nova restrição *FOREIGN KEY*

```
ALTER TABLE ItensFactura WITH CHECK  
ADD FOREIGN KEY (NumContaBal) REFERENCES ContasBalanço(NumConta)
```

Instrução que modifica o tipo de dados de uma coluna

```
ALTER TABLE ItensFactura  
ALTER COLUMN DescriçãoItem VARCHAR(200)
```

Eliminar uma tabela

Pode eliminar uma, ou mais, tabelas através da utilização da instrução *DROP TABLE*.

Para eliminar tabelas, utilize a seguinte sintaxe.

```
DROP TABLE nome_tabela1 [, nome_tabela2] ...
```

Note que se existirem outras tabelas que dependam da tabela a eliminar, o *SQL SERVER* não vai permitir a remoção da tabela. Por exemplo, não pode eliminar uma tabela se existir uma outra tabela que contenha uma restrição de chave forasteira (*FOREIGN KEY*) que referencie a tabela a remover.

Quando elimina uma tabela, muitos dos objectos que lhe estão relacionados são também removidos, nomeadamente índices, *triggers* ou restrições. Em contraste, as vistas ou os procedimentos armazenados que estejam associados à tabela a eliminar não são removidos. Tem que os remover explicitamente.