



Introdução ao SQL Server 2016

- ◆ Banco de dados relacional;
- ◆ Design do banco de dados;
- ◆ Normalização de dados;
- ◆ Arquitetura cliente / servidor;
- ◆ As linguagens SQL e T-SQL;
- ◆ SQL Server;
- ◆ Ferramentas de gerenciamento;
- ◆ SQL Server Management Studio (SSMS).



Esta Leitura Complementar refere-se ao conteúdo das Aulas 1 a 5.

1.1. Banco de dados relacional

Um banco de dados é uma forma organizada de armazenar informações de modo a facilitar sua inserção, alteração, exclusão e recuperação.

Um banco de dados relacional é uma arquitetura na qual os dados são armazenados em tabelas retangulares, semelhantes a uma planilha. Na maioria das vezes, essas tabelas possuem uma informação chave que as relaciona.

1.2. Design do banco de dados

O design do banco de dados é fundamental para que o banco de dados possua um bom desempenho. Para isso, é importante entender os princípios que norteiam a boa construção de um banco de dados.

A construção do banco de dados passa por quatro etapas até a criação dos objetos dentro do banco de dados, a saber: modelos descritivo, conceitual, lógico e físico.

1.2.1. Modelo descritivo

O modelo descritivo de dados é um documento que indica a necessidade de construção de um banco de dados. Nesse modelo, o cenário é colocado de forma escrita, informando a necessidade de armazenamento de dados. Não existe uma forma padrão para escrever esse modelo, pois cada cenário é traduzido em um modelo diferente. A seguir, mostraremos um exemplo de modelo descritivo.

A fábrica de alimentos ImpactaNatural deseja elaborar um sistema de informação para controlar suas atividades. O objetivo é modelar uma solução que atenda às áreas mais importantes da empresa. Os requisitos estão listados a seguir:

- A empresa atua com clientes previamente cadastrados e só atende pessoas jurídicas. Todos os clientes da ImpactaNatural são varejistas de diversas regiões do país. Sobre os clientes, é fundamental armazenarmos um código, que será o identificador único, o CNPJ, a razão social, o endereço de cobrança, o endereço de correspondência e o endereço para entrega das mercadorias compradas, além dos telefones de contato, o contato principal, o ramo de atividade e a data do cadastramento da instituição;
- A empresa possui um elenco bastante variado de produtos, como pães, bolos, doces, entre outros. Sobre os produtos, devemos guardar o código, o nome, a cor, as dimensões, o peso, o preço, a validade, o tempo médio de fabricação e o desenho do produto;
- Para fabricar os alimentos, diversos componentes são essenciais: matéria-prima (farinha, sal, açúcar, fermento etc.), materiais diversos (embalagens, rótulos etc.) e máquinas (centrífuga, forno etc.);

- Cada componente pode ser utilizado em vários produtos e um produto pode utilizar diversos componentes. Sobre cada um dos componentes, devemos registrar: código, nome, quantidade em estoque, preço unitário e unidade de estoque. Para as máquinas, precisamos registrar o tempo médio de vida, a data da compra e a data de fim da garantia;
- No que diz respeito às matérias-primas e aos materiais diversos necessários na elaboração de um produto, precisamos controlar a quantidade necessária e a unidade de medida;
- Devemos controlar o tempo necessário de uso das máquinas e as ferramentas necessárias na elaboração de um produto;
- Precisamos controlar a quantidade de horas necessárias da mão de obra destinada à elaboração do produto;
- Sobre a mão de obra, devemos registrar matrícula, nome, endereço, telefones, cargo, salário, data de admissão e uma descrição com as qualificações profissionais. Toda mão de obra é empregada da empresa Madeira de Lei. Precisamos também registrar a hierarquia de subordinação entre os empregados, pois um empregado pode estar subordinado a somente um empregado e este, por sua vez, pode gerenciar vários outros empregados;
- A Madeira de Lei trabalha com o esquema de encomenda. Ela não mantém o estoque de produtos elaborados, ou seja, cada vez que um cliente solicita um produto, ela o elabora. Uma encomenda pode envolver vários produtos e pertencer a um único cliente. Sobre as encomendas, devemos registrar um número, a data da inclusão, o valor total da encomenda, o valor do desconto (caso exista), o valor líquido, um identificador para a forma de pagamento (se cheque, dinheiro ou cartão de crédito) e a quantidade de parcelas. Sobre os produtos solicitados em uma encomenda, precisamos registrar a quantidade e a data de necessidade do produto;
- Matéria-prima, materiais diversos, máquinas e ferramentas utilizadas para fabricar os alimentos possuem diversos fornecedores que devem ser controlados para que o item não falte e comprometa a fabricação e, conseqüentemente, a entrega de uma encomenda. Sobre os fornecedores, deve-se registrar CNPJ, razão social, endereço, telefones, pessoa de contato. Um determinado fornecedor pode fornecer diversos itens de cada um dos grupos citados;
- É necessário também realizar manutenções nas máquinas para que elas mantenham sua vida útil e trabalhem com eficiência. A manutenção é feita por empresas especializadas em máquinas industriais. Sobre essas empresas, registramos CNPJ, razão social, endereço, telefones e pessoa de contato. Sempre que uma máquina sofrer manutenção por uma empresa, deve-se registrar a data da manutenção e uma descrição das ações realizadas nela.

Notemos que esse modelo apenas apresenta o cenário de forma ampla. A próxima etapa é a construção do modelo conceitual.

1.2.2. Modelo conceitual

Nesse modelo, extraímos informações do modelo descritivo, usando a seguinte técnica:

- Substantivos (pessoas, coisas, papéis, objetos) são denominados entidades, as quais são elementos que possuem informações a serem tratadas;
- Propriedades ou características ligadas aos substantivos são chamadas de atributos, os quais são elementos que caracterizam uma entidade.

Nesse modelo, apenas qualificamos as entidades encontradas. Vejamos a seguir um exemplo do dicionário de dados de um modelo conceitual:

- CLIENTES = Código + CNPJ + razão social + ramo de atividade + data do cadastramento + {telefones} + {endereços} + pessoa de contato;
- EMPREGADOS = Matrícula + nome + {telefones} + cargo + salário + data de admissão + qualificações + endereço;
- EMPRESAS = CNPJ + razão social + {telefones} + pessoa de contato + endereço;
- FORNECEDORES = CNPJ + razão social + endereço + {telefones} + pessoa de contato;
- TIPO DE ENDEREÇO = Código + nome;
- ENDEREÇOS = Número + logradouro + complemento + CEP + bairro + cidade + Estado;
- ENCOMENDAS = Número + data da inclusão + valor total + valor do desconto + valor líquido + ID forma de pagamento + quantidade de parcelas;
- PRODUTOS = Código + nome + cor + dimensões + peso + preço + tempo de fabricação + desenho do produto + horas de mão de obra;
- TIPOS DE COMPONENTE = Código + nome;
- COMPONENTES = Código + nome + quantidade em estoque + preço unitário + unidade;
- MÁQUINAS = Tempo de vida + data da compra + data de fim da garantia;
- RE = Quantidade necessária + unidade + tempo de uso + horas da mão de obra;
- RM = Data + descrição;
- RS = Quantidade + data de necessidade.

A partir desse modelo, iniciamos a modelagem de dados no formato de diagrama, ou seja, utilizando representações gráficas para os elementos encontrados nos modelos descritivo e conceitual.

1.2.3. Modelo lógico

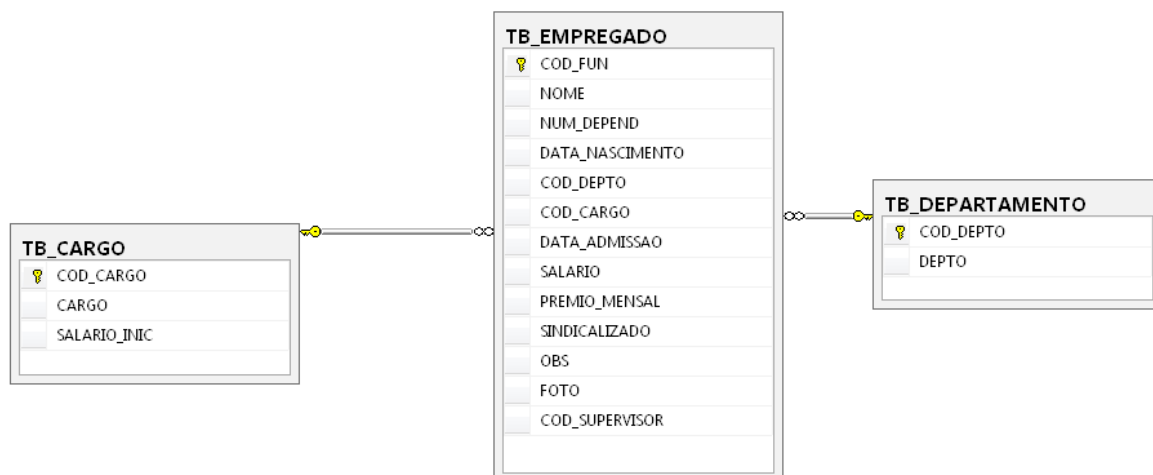
Esse modelo apresenta, em um formato de diagrama, as entidades e os atributos encontrados nos modelos anteriores. Nesse modelo, também conhecido como diagrama lógico de dados, ainda não é possível determinar qual banco de dados será utilizado.

As ferramentas de modelagem geralmente iniciam os modelos a partir do modelo lógico de dados. A partir dele é que será gerado o modelo para implementação no banco de dados. Veremos adiante um exemplo de um diagrama lógico de dados, mas antes vamos entender algumas regras importantes.

Toda entidade deve ter um identificador único, que representa um valor que não se repete para cada ocorrência da mesma entidade. Por exemplo, a entidade Fornecedor não tem nenhum atributo de valor único, dessa forma, criamos um atributo fictício chamado **ID_FORNECEDOR**. Essa regra é importante, pois esse identificador único permite o relacionamento entre as entidades.

Devemos definir quais atributos são obrigatórios e quais são opcionais, além de determinar os atributos que deverão ter valores específicos (por exemplo, sexo: M para masculino e F para feminino).

Nesse esquema, as caixas representam as entidades, as linhas representam os relacionamentos e os valores nas caixas representam os atributos.



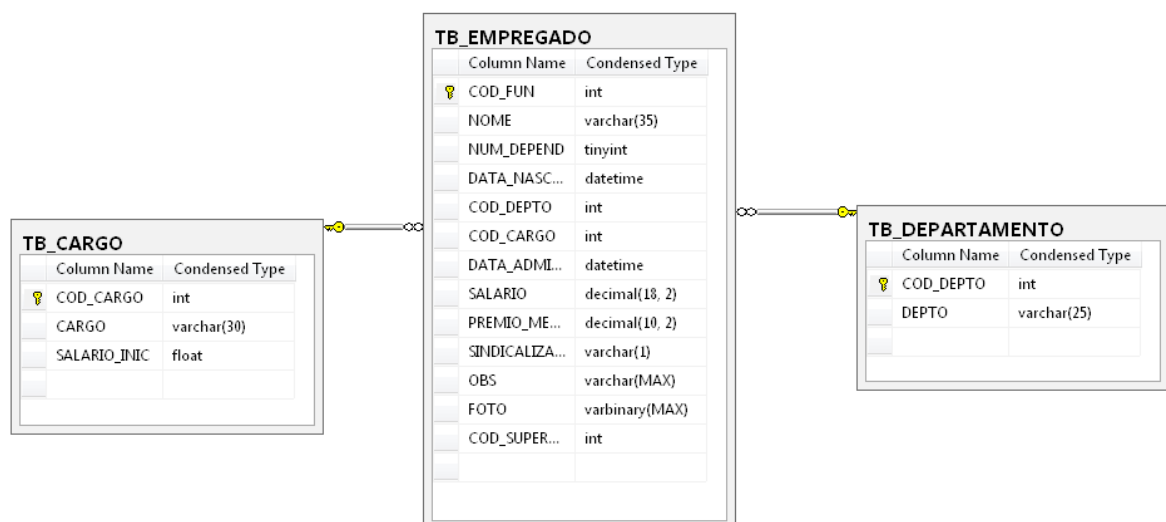
1.2.4. Modelo físico

O modelo físico de dados pode ser obtido por meio do diagrama lógico de dados e está associado ao software de gerenciamento de banco de dados, neste caso, o SQL Server 2016.

Vejamos as definições:

- **Tabelas (entidades):** Local de armazenamento das informações;
- **Campos (atributos):** Características da tabela;
- **Chave primária:** Campo único que define a exclusividade da linha. Toda chave primária possui as seguintes características:
 - Valores únicos;
 - Não permite valores nulos;
 - A tabela será ordenada pela chave primária (índice clusterizado).
- **Relacionamento:** Relação entre tabelas através de um ou mais campos. As relações podem ser:
 - 1 para 1;
 - 1 para N;
 - N para N.

Vejamos, a seguir, um exemplo de modelo físico de dados:



Esse modelo apresenta a modelagem das informações referentes aos empregados, departamentos e cargos.

1.2.5. Dicionário de dados

O dicionário de dados complementa o diagrama físico descrevendo as características da tabela.

Para entendermos a importância da descrição formal dos campos, imaginemos que existe um campo de status em uma tabela qualquer. Observe, adiante, alguns valores:

- 1 – Cancelado
- 2 – Em espera
- 3 – Encerrado
- 4 – Finalizado

Caso não seja documentada esta informação, será muito difícil a construção das consultas.

A seguir, um exemplo de dicionário de dados da tabela **TB_EMPREGADO**:

Sequência	Campo	Descrição	Tipo de Dados	NOT NULL	Identity	Bytes	PK	FK	Regras	Default
1	COD_FUN	Código do empregado	int		Sim	4	Sim			
2	NOME	Nome do empregado	varchar			35				
3	NUM_DEPEND	Nº de dependentes	tinyint	Sim		1				
4	DATA_NASCIMENTO	Data de nascimento	datetime			8				
5	COD_DEPTO	Código do departamento	int			4		Sim		
6	COD_CARGO	Código do cargo	int			4		Sim		
7	DATA_ADMISSAO	Data de admissão	datetime			8				GETDATE()
8	SALARIO	Salário	decimal			9			Não permite valores negativos	
9	PREMIO_MENSAL	Prêmio mensal	decimal	Sim		9			Valores: S e N	
10	SINDICALIZADO	Campo para verificar se o empregado é sindicalizado	varchar			1				
10	OBS	Observações	varchar	Sim						
10	FOTO	Foto	varbinary	Sim						
10	COD_SUPERVISOR	Código do supervisor	int	Sim		4				

1.3. Normalização de dados

O processo de organizar dados e eliminar informações redundantes de um banco de dados é denominado normalização.

A normalização envolve a tarefa de criar as tabelas e definir os seus relacionamentos. O relacionamento entre as tabelas é criado de acordo com regras que visam à proteção dos dados e à eliminação de dados repetidos. Essas regras são denominadas **normal forms** ou formas normais.

A normalização apresenta grandes vantagens:

- Elimina dados repetidos, o que torna o banco mais compacto;
- Garante o armazenamento dos dados de forma lógica;
- Oferece maior velocidade dos processos de classificar e indexar, já que as tabelas possuem uma quantidade menor de colunas;
- Permite o agrupamento de índices conforme a quantidade de tabelas aumenta. Além disso, reduz o número de índices por tabela. Dessa forma, permite melhor performance de atualização do banco de dados.

Entretanto, o processo de normalização pode aumentar a quantidade de tabelas e, conseqüentemente, a complexidade das associações exigidas entre elas para que os dados desejados sejam obtidos. Isso pode acabar prejudicando o desempenho da aplicação.

Outro aspecto negativo da normalização é que as tabelas, em vez de dados reais, podem conter códigos. Nesse caso, será necessário recorrer à tabela de pesquisa para obter os valores necessários. A normalização também pode dificultar a consulta ao modelo de dados.


1.3.1. Regras de normalização

A normalização inclui três regras principais: **first normal form** (1NF), **second normal form** (2NF) e **third normal form** (3NF).

Consideramos que um banco de dados está no **first normal form** quando a primeira regra (1NF) é cumprida. Se as três regras forem cumpridas, o banco de dados estará no **third normal form**.

É possível atingir outros níveis de normalização (4NF e 5NF), entretanto, o 3NF é considerado o nível mais alto requerido pela maior parte das aplicações.

Verifique a tabela **TB_ALUNO**, que atende as necessidades da área acadêmica de uma instituição de ensino:

TB_ALUNO	
	COD_ALUNO
	NOME
	DATA_NASCIMENTO
	IDADE
	E_MAIL
	FONE_RES
	FONE_COML
	FONE_CELULAR
	FONE_RECADO
	PROFISSAO
	EMPRESA

Ao executarmos o comando de criação, o SQL cria a tabela e é possível a inserção, alteração e exclusão de dados.

Vejamos, a seguir, quais as regras que devem ser cumpridas para atingir cada nível de normalização:

- **First Normal Form (1NF)**

Para que um banco de dados esteja nesse nível de normalização, cada coluna deve conter um único valor e cada linha deve abranger as mesmas colunas. A fim de atendermos a esses aspectos, os conjuntos que se repetem nas tabelas individuais devem ser eliminados. Além disso, devemos criar uma tabela separada para cada conjunto de dados relacionados e identificar cada um deles com uma chave primária.

Uma tabela sempre terá este formato:

CAMPO 1	CAMPO 2	CAMPO 3	CAMPO 4

E nunca poderá ter este formato:

CAMPO 1	CAMPO 2	CAMPO 3	CAMPO 4

Considere os seguintes exemplos:

- Uma pessoa tem apenas um nome, um RG, um CPF, mas pode ter estudado em **N** escolas diferentes e pode ter feito **N** cursos extracurriculares;
- Um treinamento da Impacta tem um único nome, uma única carga horária, mas pode haver **N** instrutores que ministram esse treinamento;
- Um aluno da Impacta tem apenas um nome, um RG, um CPF, mas pode ter **N** telefones.

Percebemos aqui que a tabela **TB_ALUNO**, que criamos anteriormente, precisa ser reestruturada para que respeite a primeira forma normal.

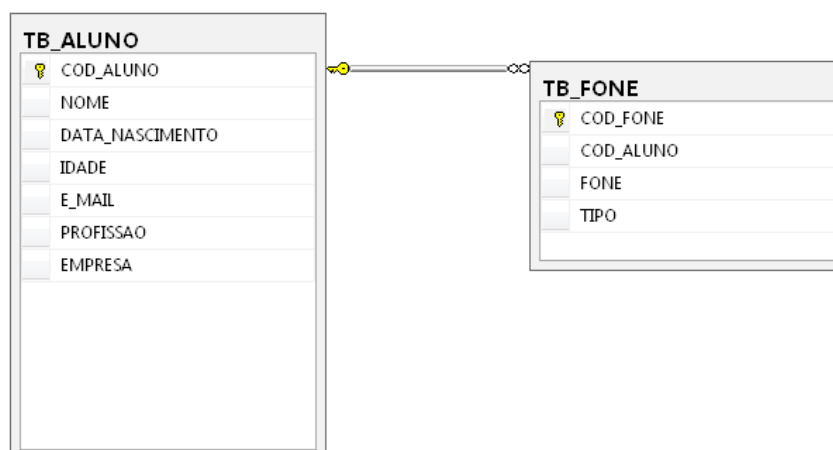
Verifique que existe um conjunto de dados repetidos: **FONE_RES**, **FONE_COML**, **FONE_CELULAR** e **FONE_RECADO**.

Numa necessidade de inclusão de um novo telefone, serão necessárias as seguintes alterações:

- Tabela;
- View;
- Procedure;
- Integrações entre banco e sistema;
- Alteração da aplicação.

Sempre que uma linha de uma tabela tiver **N** informações relacionadas a ela, precisaremos criar outra tabela para armazenar essas **N** informações.

Observe, a seguir, o modelo após a execução da primeira forma normal:



Opcionalmente, podemos eliminar o campo **COD_FONE** e utilizar os campos **NUM_ALUNO** e **FONE** como chave da tabela, impedindo que se cadastre o mesmo telefone mais de uma vez para o mesmo aluno.

- **Second Normal Form (2NF)**

No segundo nível de normalização, devemos criar tabelas separadas para conjuntos de valores que se aplicam a vários registros, ou seja, que se repetem.

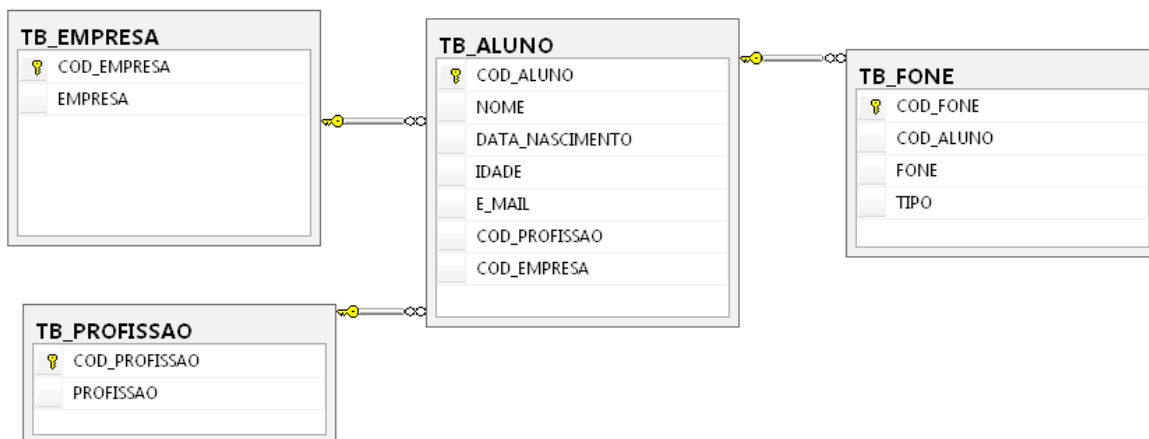
Com a finalidade de criar relacionamentos, devemos relacionar essas novas tabelas com uma chave estrangeira e identificar cada grupo de dados relacionados com uma chave primária.

Em outras palavras, a segunda forma normal pede que evitemos campos descritivos (alfanuméricos) que se repitam várias vezes na mesma tabela. Além de ocupar mais espaço, a mesma informação pode ser escrita de formas diferentes. Veja o caso da tabela **ALUNOS**, em que existe um campo chamado **PROFISSAO** (descritivo) onde é possível grafarmos a mesma profissão de várias formas diferentes:

ANALISTA DE SISTEMAS
ANALISTA SISTEMAS
AN. SISTEMAS
AN. DE SISTEMAS
ANALISTA DE SIST.

Isso torna impossível que se gere um relatório filtrando os **ALUNOS** por **PROFISSAO**. A solução, neste caso, é criar uma tabela de profissões em que cada profissão tenha um código. Para isso, na tabela **ALUNOS**, substituiremos o campo **PROFISSAO** por **COD_PROFISSAO**.

A seguir, vejamos o modelo após a execução da segunda forma normal:



- **Third Normal Form (3NF)**

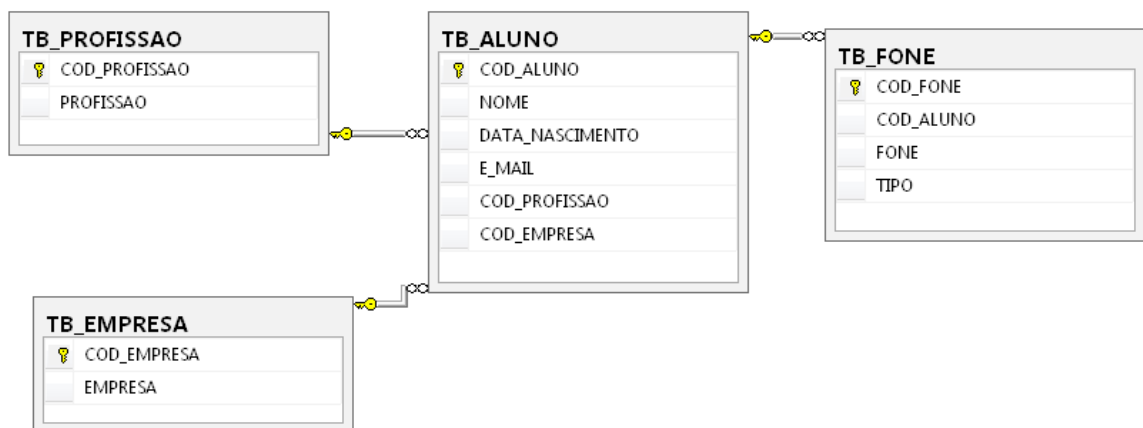
No terceiro nível de normalização, após ter concluído todas as tarefas do **1NF** e **2NF**, devemos eliminar os campos que não dependem de chaves primárias.

Cumpridas essas três regras, atingimos o nível de normalização requerido pela maioria dos programas.

Considere os seguintes exemplos:

- Em uma tabela de **PRODUTOS** que tenha os campos **PRECO_COMPRA** e **PRECO_VENDA**, não devemos ter um campo **LUCRO**, pois ele não depende do código do produto (chave primária), mas sim dos preços de compra e de venda. O lucro será facilmente gerado através da expressão **PRECO_VENDA - PRECO_CUSTO**;
- Na tabela **TB_ALUNO**, não devemos ter o campo **IDADE**, pois ele não depende do número do aluno (chave primária), mas sim do campo **DATA_NASCIMENTO**.

A modelagem final após a aplicação da terceira forma normal é a seguinte:

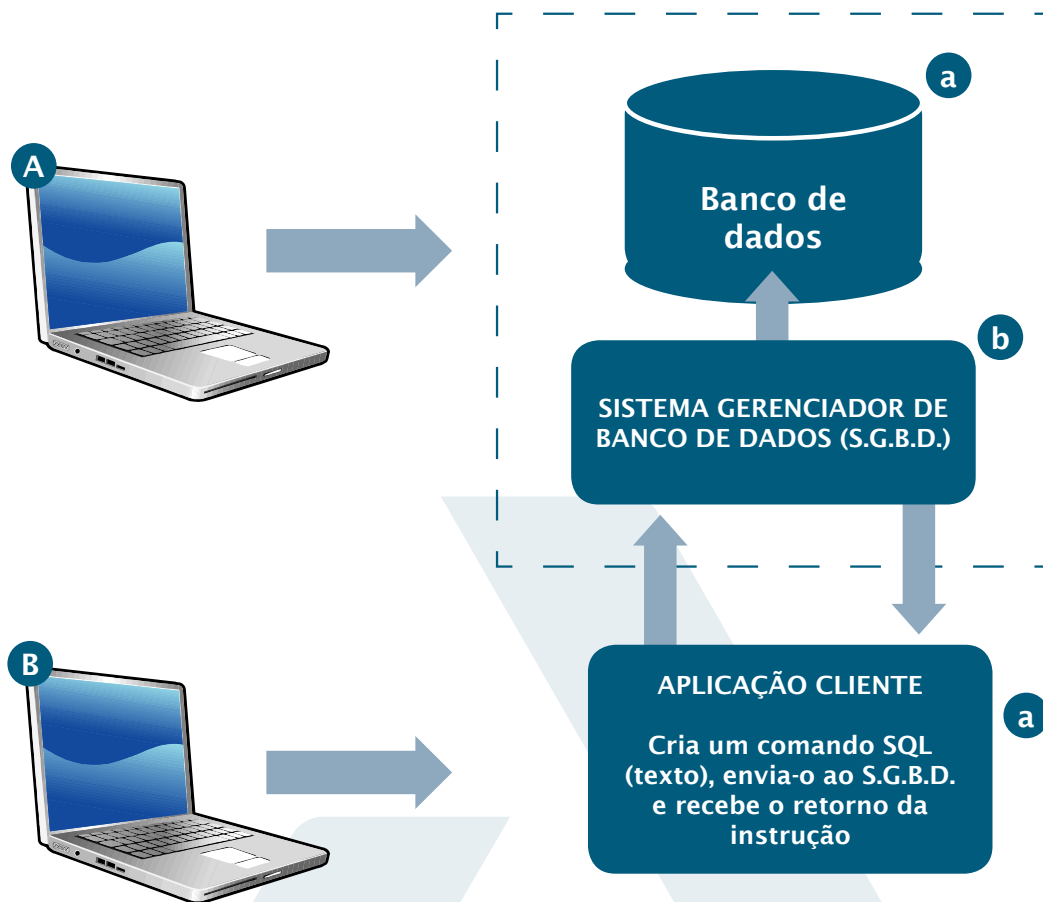


1.4.Arquitetura cliente / servidor

Essa arquitetura funciona da seguinte forma:

- Usuários acessam o servidor por meio de um aplicativo instalado no próprio servidor ou de outro computador;
- O computador cliente executa as tarefas do aplicativo, ou seja, fornece a interface do usuário (tela, processamento de entrada e saída) e manda uma solicitação ao servidor;
- O servidor de banco de dados processa a solicitação, que pode ser uma consulta, alteração, exclusão, inclusão etc.

A imagem a seguir ilustra a arquitetura cliente / servidor:



- **A - Servidor**
 - **a** - Software servidor de banco de dados. É ele que gerencia todo o acesso ao banco de dados. Ele recebe os comandos SQL, verifica sua sintaxe e os executa, enviando o retorno para a aplicação que enviou o comando;
 - **b** - Banco de dados, incluindo as tabelas que serão manipuladas.
- **B - Cliente**
 - **a** - Aplicação contendo a interface visual que envia os comandos SQL ao servidor.

1.5.As linguagens SQL e T-SQL

Toda a manipulação de um banco de dados é feita por meio de uma linguagem específica, com uma exigência sintática rígida, chamada SQL (Structured Query Language). Os fundamentos dessa linguagem estão baseados no conceito de banco de dados relacional.

Essa linguagem foi desenvolvida pela IBM no início da década de 1970 e, posteriormente, foi adotada como linguagem padrão pela ANSI (American National Standards Institute) e pela ISO (International Organization for Standardization), em 1986 e 1987, respectivamente.

A T-SQL (Transact-SQL) é uma implementação da Microsoft para a SQL padrão ANSI. Ela cria opções adicionais para os comandos e também cria novos comandos que permitem o recurso de programação, como os de controle de fluxo, variáveis de memória etc.

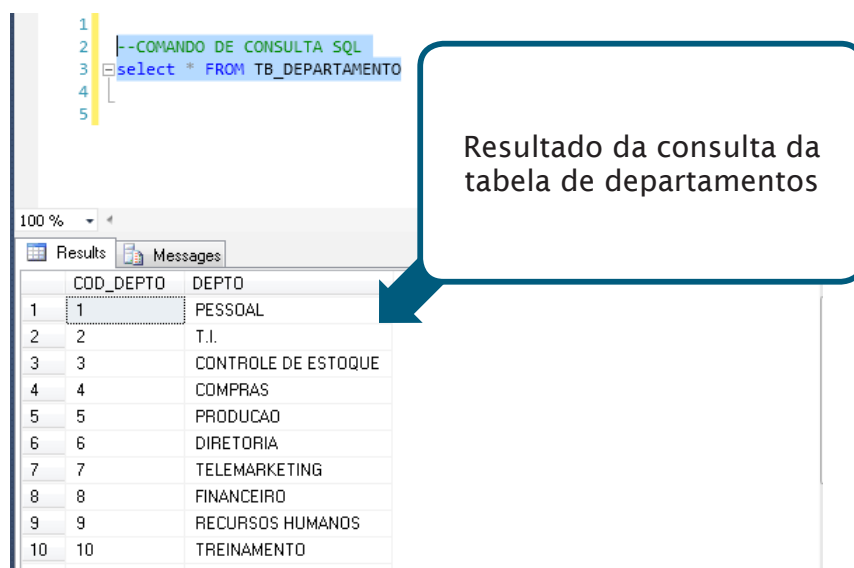
Além da T-SQL, há outras implementações da SQL, como Oracle PL/SQL (Procedural Language/SQL) e IBM's SQL Procedural Language.

Veja um exemplo de comando SQL:

Para realizar uma consulta na tabela de departamento (TB_DEPARTAMENTO) é utilizado o comando **SELECT**. Esse comando permite a visualização das informações contidas na referida tabela.

```
SELECT * FROM TB_DEPARTAMENTO
```

Ao executarmos o comando no SQL Server 2016, veremos o resultado conforme a ilustração adiante:



Resultado da consulta da tabela de departamentos

	COD_DEPTO	DEPTO
1	1	PESSOAL
2	2	T.I.
3	3	CONTROLE DE ESTOQUE
4	4	COMPRAS
5	5	PRODUCAO
6	6	DIRETORIA
7	7	TELEMARKETING
8	8	FINANCEIRO
9	9	RECURSOS HUMANOS
10	10	TREINAMENTO
11	11	PRODUTIVIDADE

1.6. SQL Server

O SQL Server é uma plataforma de banco de dados utilizada para armazenar dados e processá-los, tanto em um formato relacional quanto em documentos XML. Também é utilizada em aplicações de comércio eletrônico e atua como uma plataforma inteligente de negócios para integração e análise de dados, bem como de soluções.

Para essas tarefas, o SQL Server faz uso da linguagem T-SQL para gerenciar bancos de dados relacionais, que contém, além da SQL, comandos de linguagem procedural.

1.6.1. Componentes

O SQL Server oferece diversos componentes opcionais e ferramentas relacionadas que auxiliam e facilitam a manipulação de seus sistemas. Por padrão, nenhum dos componentes será instalado.

A seguir, descreveremos as funcionalidades oferecidas pelos principais componentes do SQL Server:

- **SQL Server Database Engine:** É o principal componente do MS-SQL. Recebe as instruções SQL, executa-as e devolve o resultado para a aplicação solicitante. Funciona como um serviço no Windows e, normalmente, é iniciado juntamente com a inicialização do sistema operacional;
- **Analysis Services:** Usado para consultas avançadas, que envolvem muitas tabelas simultaneamente, e para geração de estruturas OLAP (On-Line Analytical Processing);
- **Reporting Services:** Ferramenta para geração de relatórios;
- **Integration Services:** Facilita o processo de transferência de dados entre bancos de dados.

1.6.2. Objetos de banco de dados

Os objetos que fazem parte de um sistema de banco de dados do SQL Server são criados dentro do objeto **DATABASE**, que é uma estrutura lógica formada por dois tipos de arquivo: um arquivo responsável por armazenar os dados e outro por armazenar as transações realizadas. Veja a seguir alguns objetos de banco de dados do SQL Server.

1.6.2.1. Tabelas

Os dados são armazenados em objetos de duas dimensões denominados tabelas (**tables**), formadas por linhas e colunas. As tabelas contêm todos os dados de um banco de dados e são a principal forma para coleção de dados.

1.6.2.2. Índices

Quando realizamos uma consulta de dados, o SQL Server 2016 faz uso dos índices (**index**) para buscar, de forma fácil e rápida, informações específicas em uma tabela ou VIEW indexada.

1.6.2.3. CONSTRAINT

São objetos cuja finalidade é estabelecer regras de integridade e consistência nas colunas das tabelas de um banco de dados. São cinco os tipos de CONSTRAINT oferecidos pelo SQL Server: **PRIMARY KEY**, **FOREIGN KEY**, **UNIQUE**, **CHECK** e **DEFAULT**.

1.6.2.4. VIEW (Visualização)

Definimos uma VIEW (visualização) como uma tabela virtual composta por linhas e colunas de dados, os quais são provenientes de tabelas referenciadas em uma consulta que define essa tabela.

Esse objeto oferece uma visualização lógica dos dados de uma tabela, de modo que diversas aplicações possam compartilhá-la.

Essas linhas e colunas são geradas de forma dinâmica no momento em que é feita uma referência a uma VIEW.

1.6.2.5. PROCEDURE (Procedimento armazenado)

Nesse objeto, encontramos um bloco de comandos T-SQL, responsável por uma determinada tarefa. Sua lógica pode ser compartilhada por diversas aplicações. A execução de uma procedure é realizada no servidor de dados. Por isso, seu processamento ocorre de forma rápida, visto que seu código tende a ficar compilado na memória.

1.6.2.6. FUNCTION (Função)

Nesse objeto, encontramos um bloco de comandos T-SQL responsável por uma determinada tarefa, isto é, a função (**FUNCTION**) executa um procedimento e retorna um valor. Sua lógica pode ser compartilhada por diversas aplicações.

1.6.2.7. TRIGGER (Gatilho)

Esse objeto também possui um bloco de comandos T-SQL. O TRIGGER é criado sobre uma tabela e ativado automaticamente no momento da execução dos comandos **UPDATE**, **INSERT** ou **DELETE**.

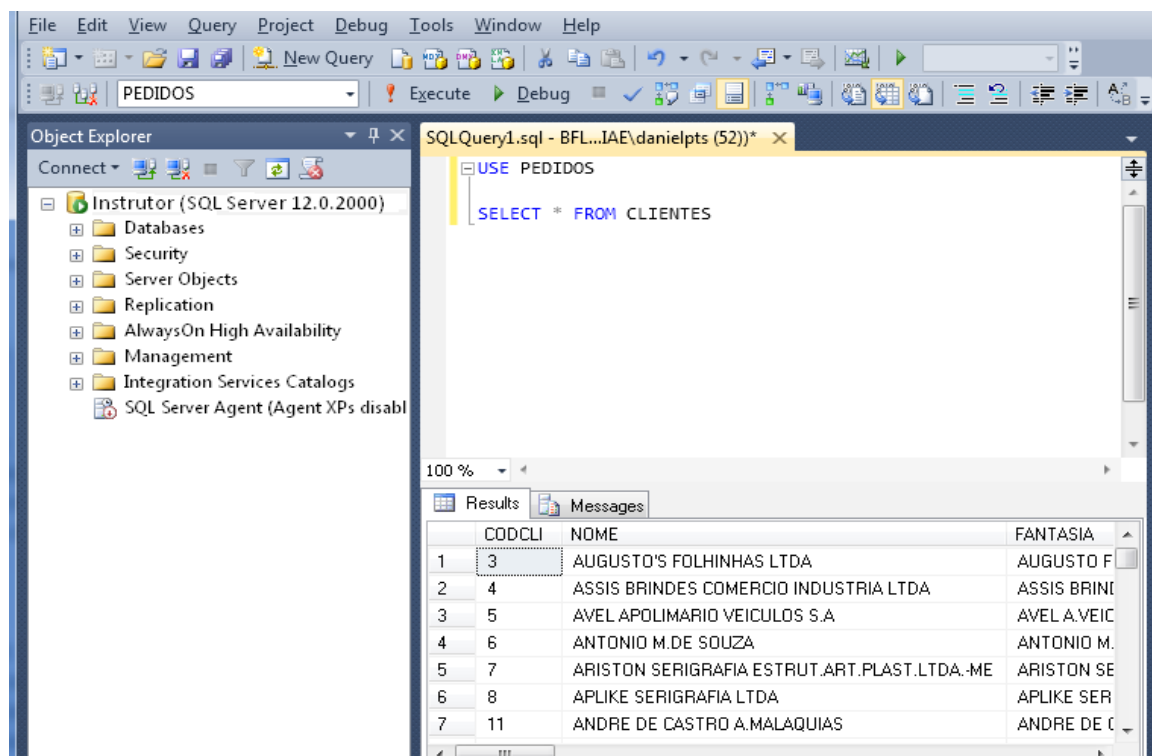
Quando atualizamos, inserimos ou excluimos dados em uma tabela, o TRIGGER automaticamente grava em uma tabela temporária os dados do registro atualizado, inserido ou excluído.

1.7. Ferramentas de gerenciamento

A seguir, descreveremos as funcionalidades oferecidas pelas ferramentas de gerenciamento disponíveis no SQL Server e que trabalham associadas aos componentes descritos anteriormente:

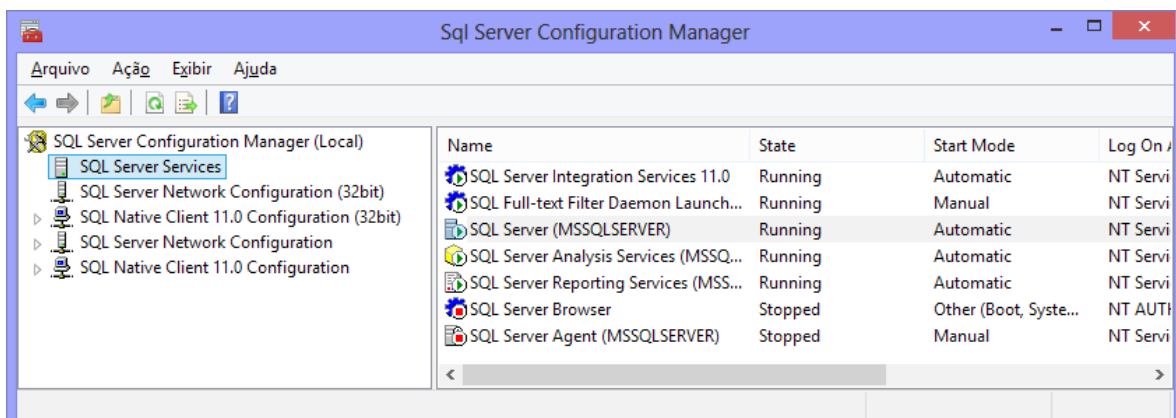
- **SQL Server Management Studio (SSMS)**

É um aplicativo usado para gerenciar bancos de dados e que permite criar, alterar e excluir objetos no banco de dados:



- **SQL Server Configuration Manager**

Permite visualizar, alterar e configurar os serviços dos componentes do SQL Server:



- **Microsoft SQL Server Profiler**

Essa ferramenta permite capturar e salvar dados de cada evento em um arquivo ou tabela para análise posterior.

- **Database Engine Tuning Advisor**

Analisa o desempenho das operações e sugere opções para sua melhora.

- **SQL Server Data Tools**

Possui uma interface que integra os componentes **Business Intelligence, Analysis Services, Reporting Services e Integration Services**.

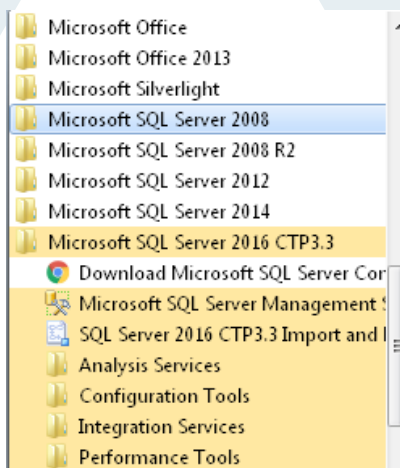
1.8. SQL Server Management Studio (SSMS)

Essa é a principal ferramenta para gerenciamento de bancos de dados, por isso é fundamental conhecer o seu funcionamento. Os próximos tópicos apresentam como inicializar o SSMS, sua interface, como executar comandos e como salvar scripts.

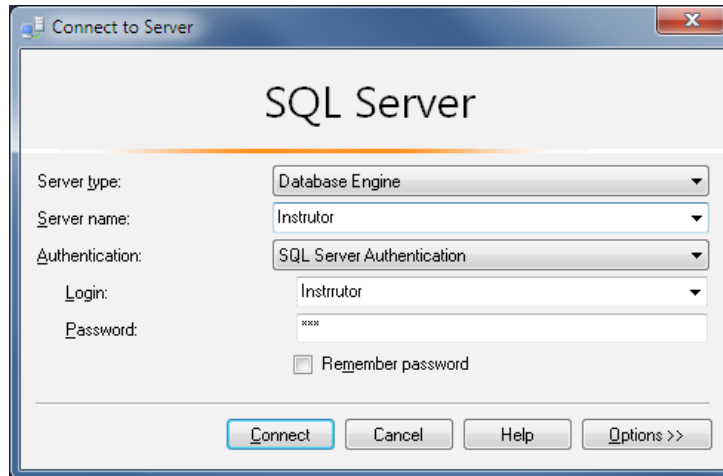
1.8.1. Inicializando o SSMS

Para abrir o SQL Server Management Studio, siga os passos adiante:

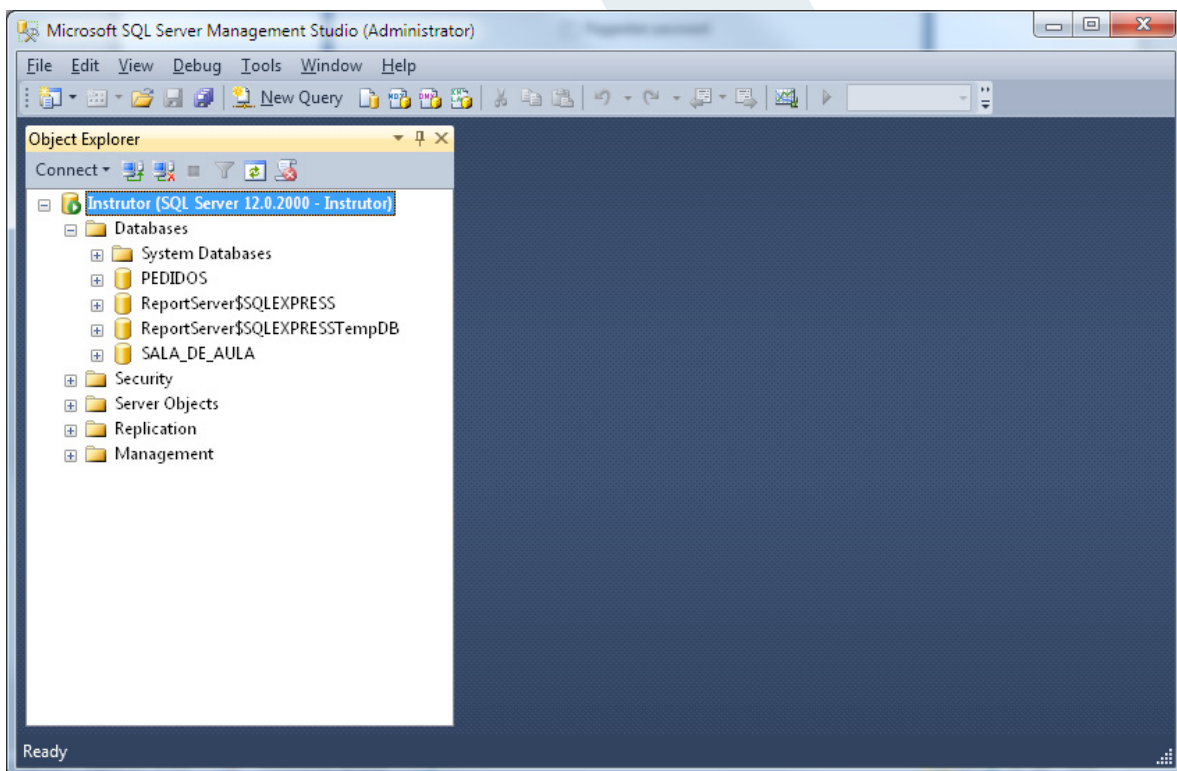
1. Clique no botão **Iniciar** e depois na opção **Todos os Programas**;
2. Clique em **Microsoft SQL Server 2016** e, em seguida, em **Microsoft SQL Server Management Studio**:



3. Na tela **Connect to Server**, escolha a opção **SQL Server Authentication** para o campo **Authentication** e, no campo **Server Name**, especifique o nome do servidor com o qual será feita a conexão:



4. Clique no botão **Connect**. A interface do SQL Server Management Studio será aberta, conforme mostra a imagem a seguir:

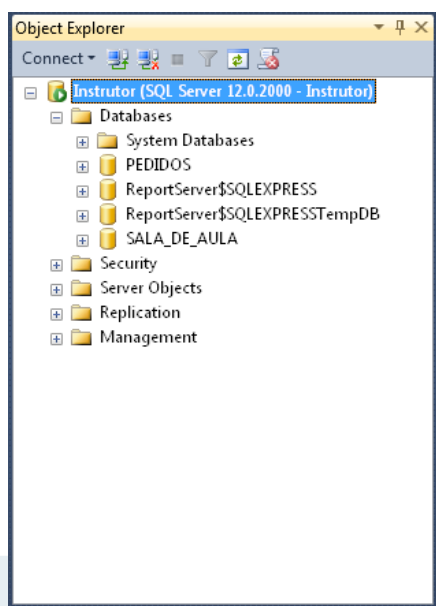


1.8.2. Interface

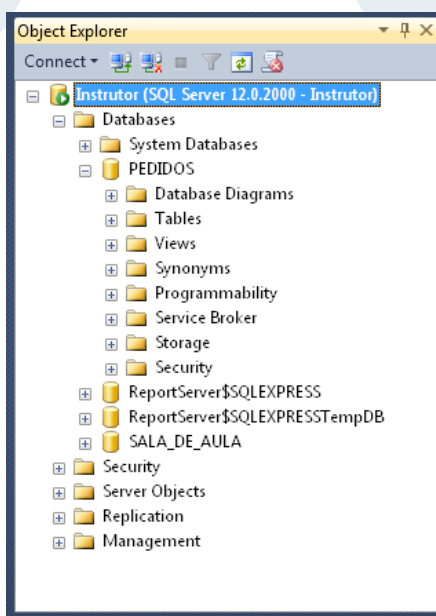
A interface do SSMS é composta pelo **Object Explorer** e pelo **Code Editor**, explicados a seguir:

- **Object Explorer**

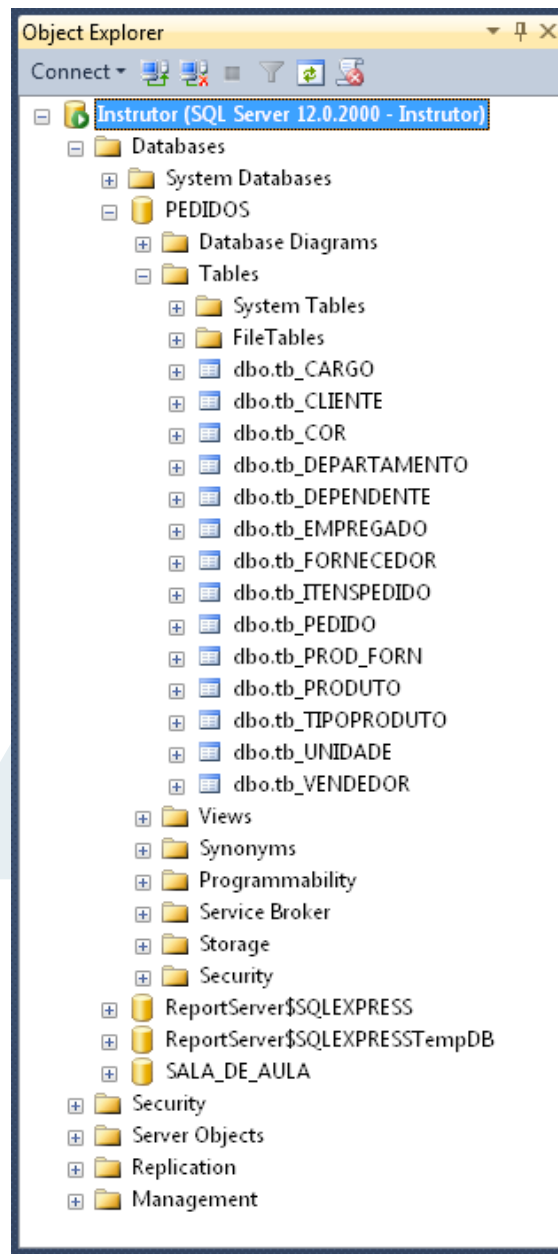
É uma janela que contém todos os elementos existentes dentro do seu servidor MS-SQL Server no formato de árvore:



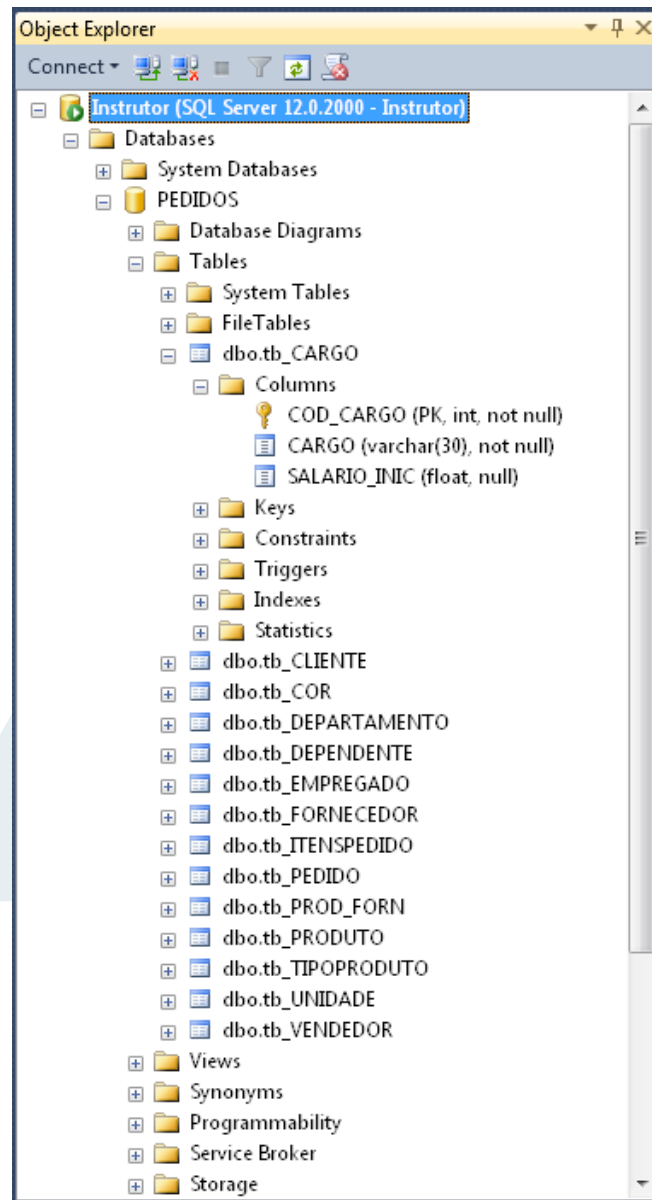
Observe que a pasta **Databases** mostra todos os bancos de dados existentes no servidor. No caso da imagem a seguir, **PEDIDOS** é um banco de dados:



Expandindo o item **PEDIDOS** e depois o item **Tables**, veremos os nomes das tabelas existentes no banco de dados:

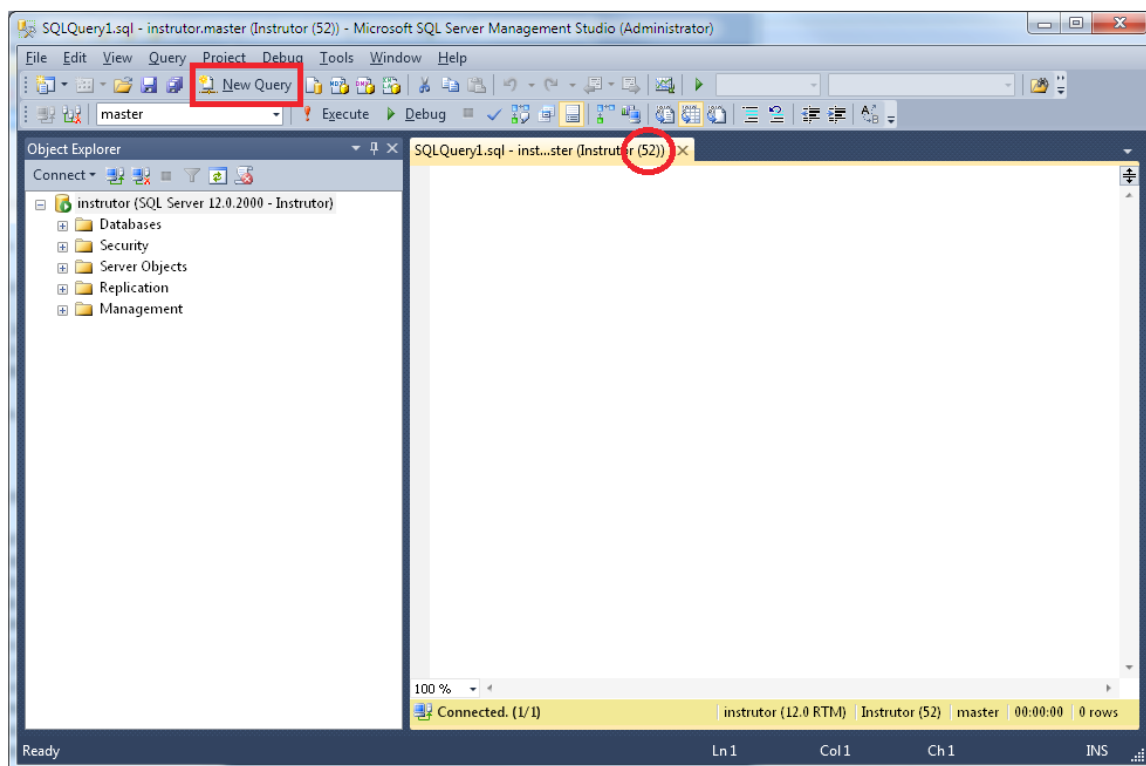


Expandindo uma das tabelas, veremos características da sua estrutura, bem como as suas colunas:



- **Code Editor**

O **Code Editor** (Editor de Código) do SQL Server Management Studio permite escrever comandos T-SQL, MDX, DMX, XML/A e XML. Clicando no botão **New Query** (Nova Consulta), será aberta uma janela vazia para edição dos comandos. Cada vez que clicarmos em **New Query**, uma nova aba vazia será criada:

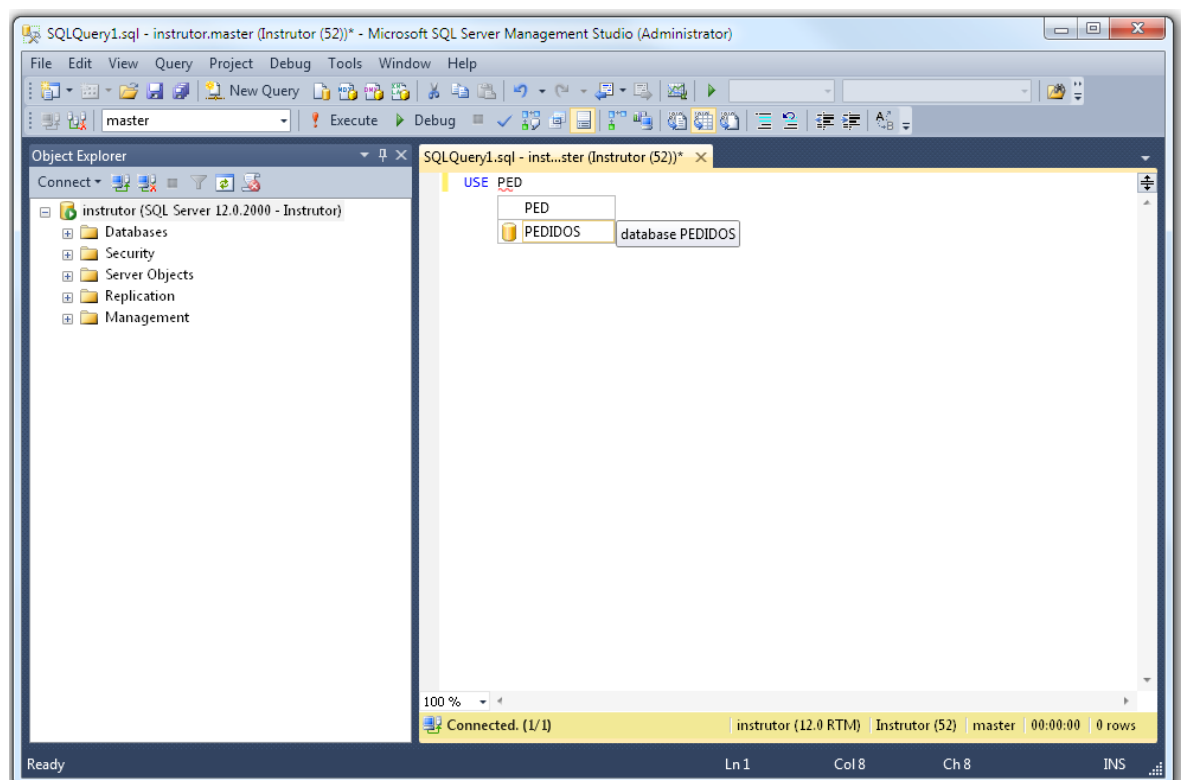


Cada uma dessas abas representa uma conexão com o banco de dados e recebe um **número de sessão**. Cada conexão tem um número de sessão único, mesmo que tenha sido aberta pelo mesmo usuário com o mesmo login e senha. Quando outra aplicação criada em outra linguagem, como Delphi, VB ou C#, abrir uma conexão, ela também receberá um número único de sessão.

1.8.3.Executando um comando

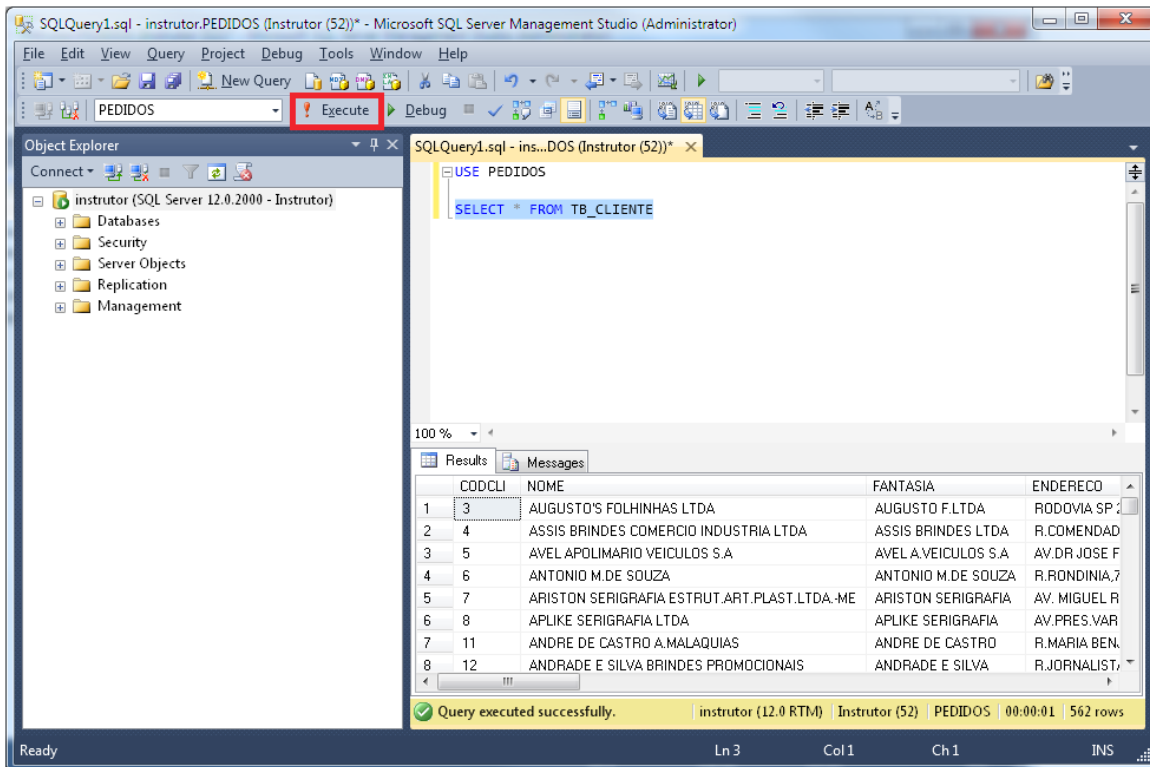
Para executar um comando a partir do SQL Server Management Studio, adote o seguinte procedimento:

1. Escreva o comando desejado no **Code Editor**. Enquanto um comando é digitado no **Code Editor**, o SQL Server oferece um recurso denominado **IntelliSense**, que destaca erros de sintaxe e digitação e fornece ajuda para a utilização de parâmetros no código. Ele está ativado por padrão, mas pode ser desativado. Para forçar a exibição do **IntelliSense**, utilize CTRL + Barra de espaço:



2. Selecione o comando escrito. A seleção é necessária apenas quando comandos específicos devem ser executados, dentre vários;

3. Na barra de ferramentas do **Code Editor**, clique sobre o botão **Execute** ou pressione a tecla **F5** (ou **CTRL + E**) para que o comando seja executado. O resultado do comando será exibido na parte inferior da interface, conforme a imagem a seguir:



Com relação ao procedimento anterior, é importante considerar as seguintes informações:

- É possível ocultar o resultado do comando por meio do atalho **CTRL + R**;
- **MASTER** é um banco de dados de sistema e que já vem instalado no MS-SQL Server;
- Caso não selecione um comando específico, todos os comandos escritos no texto serão executados e, nesse caso, eles precisarão estar organizados em uma sequência lógica perfeita, senão ocorrerão erros;
- Quando salvamos o arquivo contido no editor, ele recebe a extensão **.sql**, por padrão. É um arquivo de texto também conhecido como **SCRIPT SQL**.

1.8.4. Comentários

Todo script deve possuir comentários que ajudarão a entender e documentar as instruções. A cor do comentário é o verde e o SQL ignora o conteúdo do texto.

Para comentar uma linha utilize dois traços (--). Por exemplo:

```
-- Comentário
-- Primeira aula de SQL Server

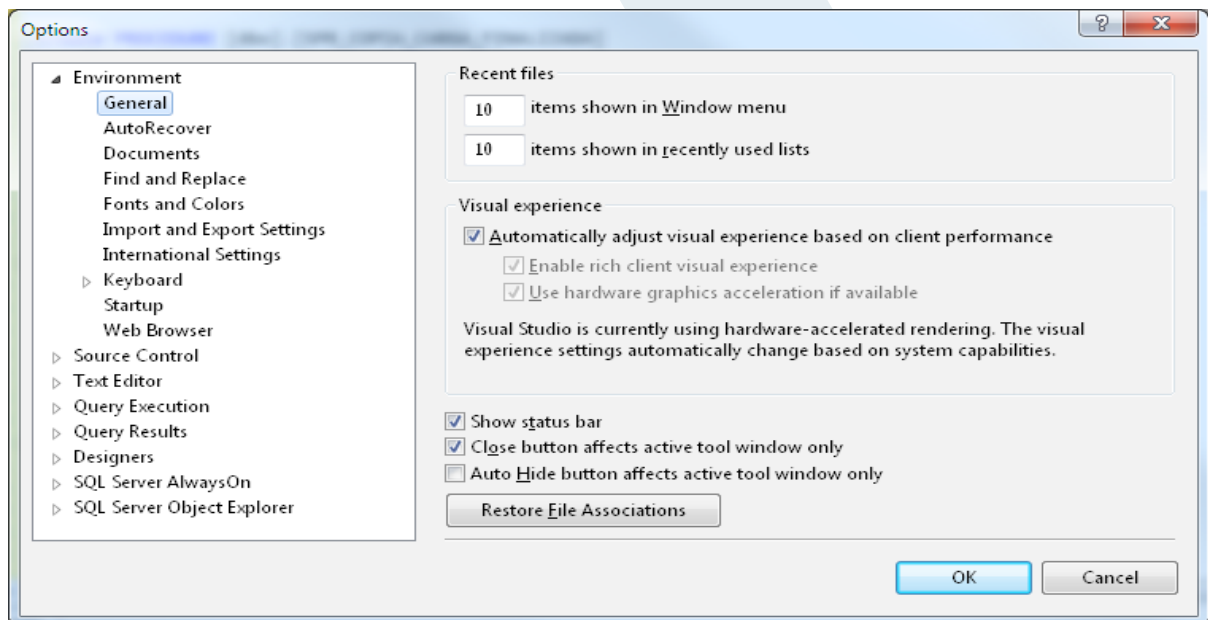
-- O comando a seguir executa uma consulta que retorna as informações da tabela
de departamentos
SELECT ...
```

Também é possível a criação de um bloco de comentários, utilizando /* e */:

```
/*
Bloco de texto comentado que não é executado pelo SQL
*/
```

1.8.5. Opções

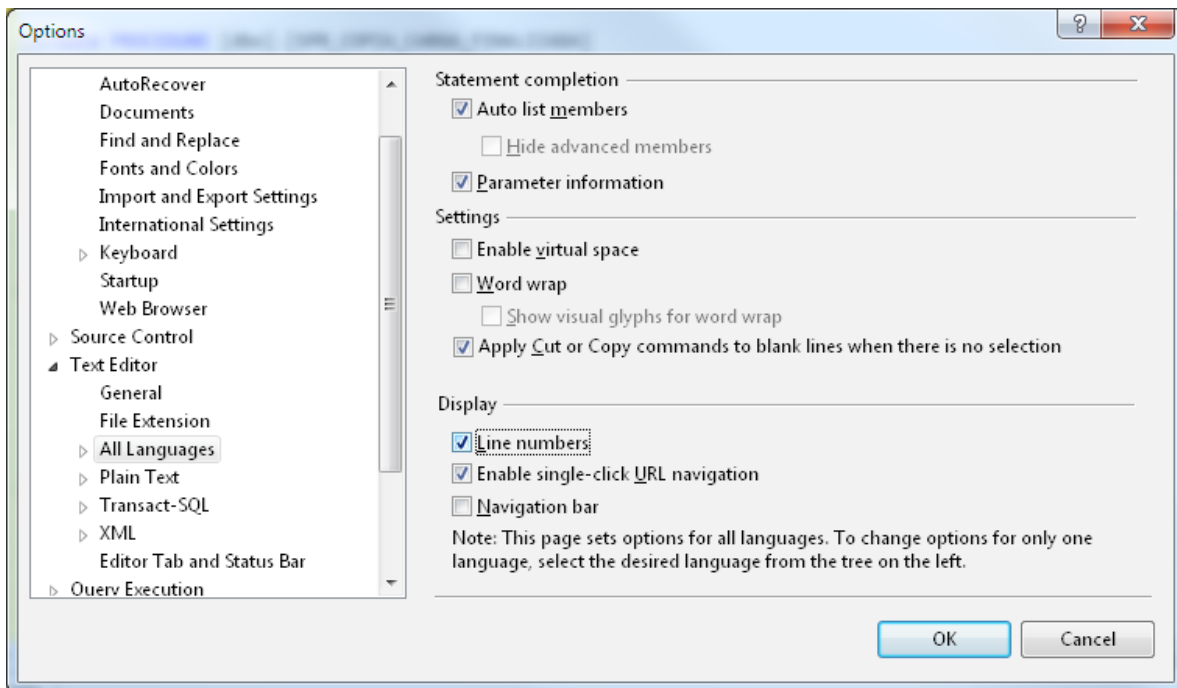
Várias opções de customização do SSMS podem ser ajustadas conforme a preferência do usuário. Para isso, no menu **Tools**, clique em **Options**, abrindo a seguinte janela:



Vejamos, a seguir, alguns exemplos de customização:

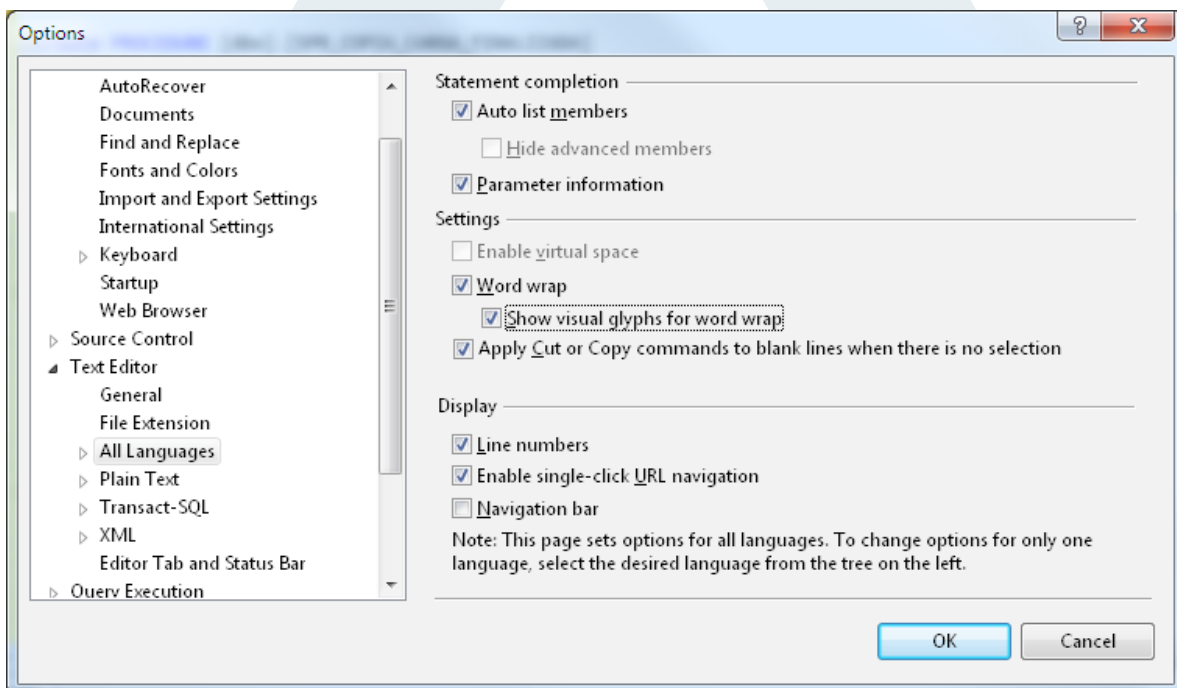
- **Numeração automática da linha**

Na janela **Options**, clique em **Text Editor**. Na guia **All Languages**, selecione **Line numbers**:



- **Quebra de linha de automática**

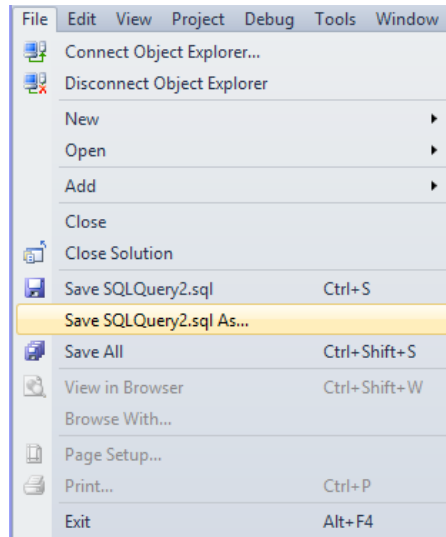
Na mesma guia, selecione **Word wrap** e **Show visual glyphs for word wrap**:



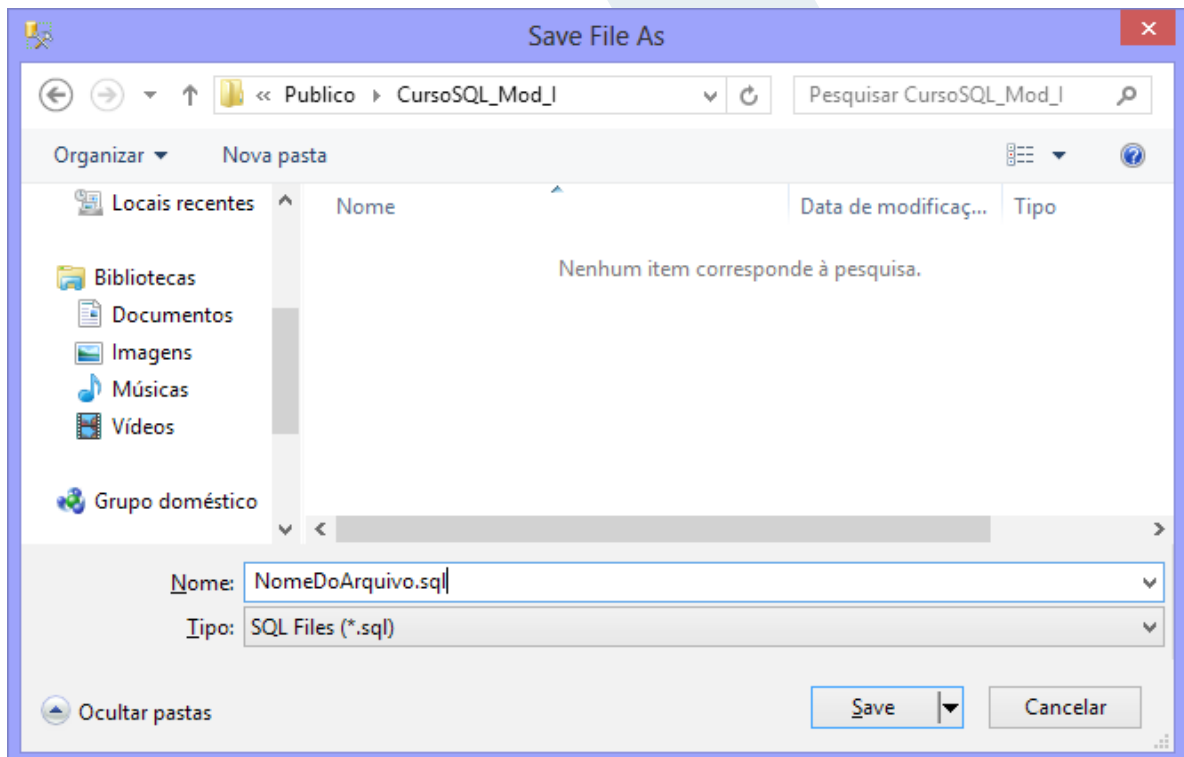
1.8.6. Salvando scripts

Para salvar os scripts utilizados, siga os passos adiante:

1. Acesse o menu **File** e clique em **Save As**:



2. Digite o nome escolhido para o script:



3. Clique em **Save**.

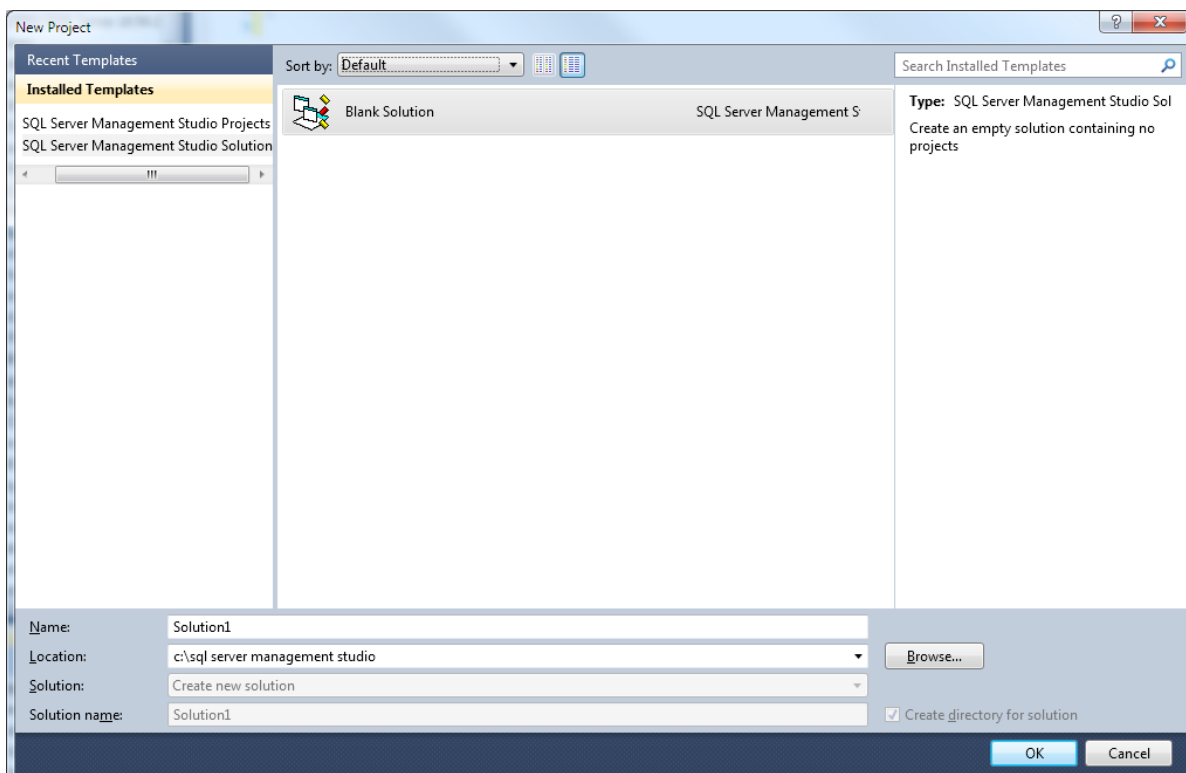
1.8.7. Soluções e Projetos

Uma solução é um conjunto de projetos, enquanto que um projeto é a coleção dos scripts. A organização é a grande vantagem deste tipo de recurso.

Vejamos, a seguir, como criar soluções e projetos:

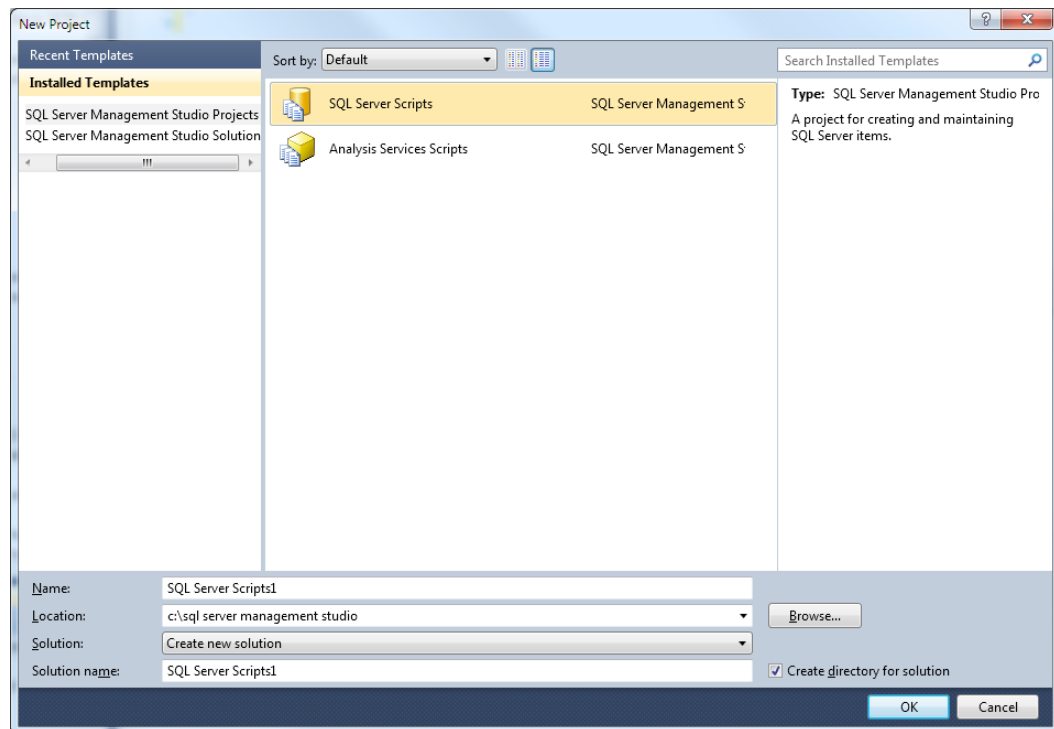
- **Criando uma solução**

No SSMS, clique no menu **File / New / Project** e, em seguida, selecione **SQL Server Management Studio Solution**:



- **Criando projetos**

No SSMS, clique no menu **File / New / Project** e, em seguida, **selecione SQL Server Management Studio Projects**:



Pontos principais

Atente para os tópicos a seguir. Eles devem ser estudados com muita atenção, pois representam os pontos mais importantes da leitura.

- Um banco de dados armazena informações e seu principal objeto são as tabelas;
- É fundamental o design de um banco de dados para que possua um bom desempenho;
- Os modelos de design de um banco de dados são: modelo descritivo, modelo conceitual, modelo lógico e modelo físico;
- Uma tabela precisa ter uma coluna que identifica de forma única cada uma de suas linhas. Essa coluna é chamada de chave primária;
- Normalização é o processo de organizar dados e eliminar informações redundantes de um banco de dados. Envolve a tarefa de criar as tabelas, bem como definir relacionamentos. O relacionamento entre as tabelas é criado de acordo com regras que visam à proteção dos dados e à eliminação de dados repetidos. Essas regras são denominadas **normal forms**, ou formas normais;
- A linguagem T-SQL (Transact-SQL) é baseada na linguagem SQL ANSI, desenvolvida pela IBM na década de 1970;
- Os principais objetos de um banco de dados são: tabelas, índices, CONSTRAINT, VIEW, PROCEDURE, FUNCTION e TRIGGER;
- O SQL Server Management Studio (SSMS) é a principal ferramenta para gerenciamento de bancos de dados.

