



Instituto Superior de Economia e Gestão



Disciplina de Tecnologias de Informação

**CONCEPÇÃO E IMPLEMENTAÇÃO
DE BASES DE DADOS**

Artur Cunha e Mário Caldeira

Março de 2006

Índice

Concepção de Bases de Dados – utilização do modelo Entidade-Associação	2
Introdução	2
Etapas do método	3
Conceitos Básicos: Entidades, Atributos e Associações.....	6
Transformação do Modelo Entidade-Associação num Esquema Relacional.....	22
Normalização do esquema relacional.....	33
Implementação de uma Base de Dados em Microsoft Access	36
Introdução	36
O que é o Access	36
Como iniciar o Access	36
Criação de Tabelas	38
Exemplo: Caso Transimporta.....	40
Associações entre tabelas	44
Questionar a base de dados – a obtenção de listagens	46
Exemplo – Caso Gestão de contas de um Banco Comercial.....	55
Bibliografia	63

Concepção de Bases de Dados – utilização do modelo Entidade- Associação

Introdução

O Modelo Entidade-Associação permite uma representação gráfica das estruturas de dados necessárias ao funcionamento de um sistema de informação de uma organização. Tal como noutras técnicas de análise de sistemas, para desenhar o modelo é necessário ter um conhecimento profundo dos requisitos da organização. Esse conhecimento deverá ser obtido através da análise das necessidades de informação, documentação existente e diálogo com os utilizadores.

A utilização desta técnica tem algumas vantagens importantes, podendo levar o analista a questionar regras existentes da organização e propor algumas melhorias, eliminando eventuais incoerências ou redundâncias existentes.

A construção do modelo é faseada, com uma abordagem “Top-Down”. Esta abordagem destaca a necessidade de um envolvimento efectivo dos utilizadores do sistema (seja o gestor ou outros utilizadores, que devem definir “o que o sistema deve fazer”). Este envolvimento é fundamental na concepção dos conceitos corporativos que devem estar presentes no sistema. É neste momento que o analista de sistemas irá identificar os grandes grupos de informação (Entidades) e as relações existentes entre estes (Associações).

O objectivo é a obtenção de um modelo conceptual, que represente as necessidades de informação da organização, e que dará origem a um conjunto de tabelas passíveis de serem implementadas em qualquer Sistema de Gestão de Base de Dados (SGBD) relacional.

A base de dados assim obtida deverá ter as seguintes características:

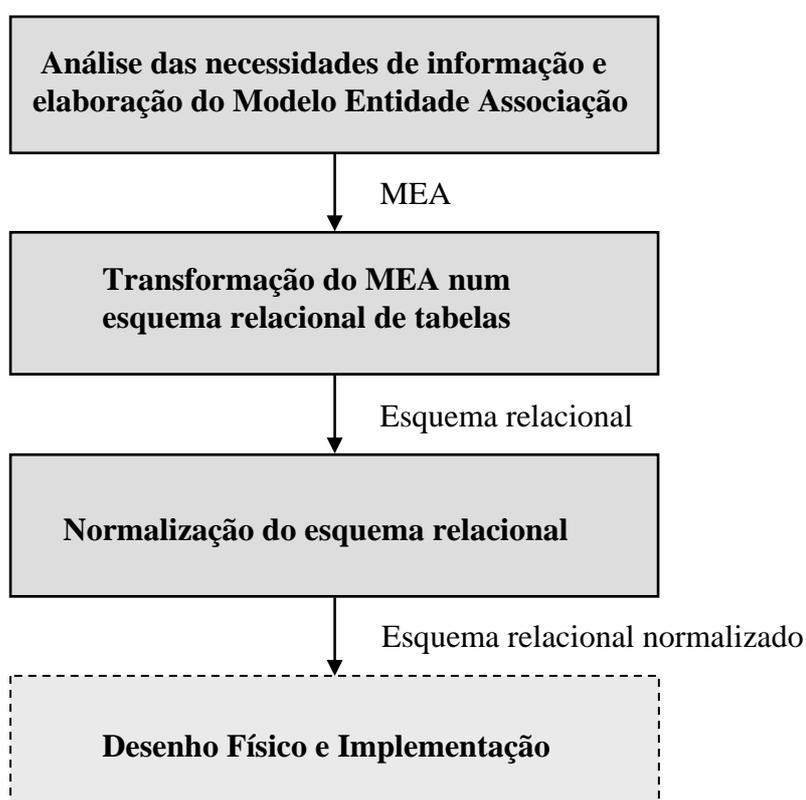
- **Ser completa** - conter toda a informação que a organização necessita;
- **Ser flexível** - fornecer, de um modo simples e em tempo útil, qualquer tipo de informação considerada necessária;
- **Evitar a redundância** - cada parcela de informação deverá ser introduzida na base de dados uma única vez e, a partir daí, utilizada em todos os processos para os quais seja necessária.

Se não forem definidas correctamente as estruturas globais de dados, independentemente das aplicações informáticas que os manipulam, tenderá a acontecer que os mesmos dados podem ser gerados e actualizados por várias aplicações, originando repetições e eventuais inconsistências.

As organizações empresariais modernas têm absoluta necessidade de SGBDs para estruturar e manipular o enorme volume de dados que normalmente resultam da sua actividade.

Etapas do método

O método de modelização de dados que vamos estudar inclui várias fases, que estão representadas na figura seguinte:



A primeira fase consta da elaboração de um modelo gráfico, a partir de uma análise das necessidades de informação do sistema em estudo. Esse modelo é, como veremos, constituído basicamente por um conjunto de elementos (Entidades) ligados entre si (Associações).

Esta actividade de Análise envolve três factores: o analista, a realidade a analisar, e as ferramentas que são utilizadas para essa análise. Naturalmente, o produto final irá depender da conjugação deste conjunto de factores, pelo que, em muitos casos, a mesma realidade, analisada por diferentes pessoas, pode conduzir a modelos distintos. Ou seja, a actividade de análise não é uma ciência exacta: poderão surgir (e, em circunstâncias normais surgirão...) modelos distintos, para representar a mesma realidade, uns melhores, outros piores...

Vejamos a seguinte descrição sucinta e simplificada e uma das soluções possíveis:

A *Transimporta* é uma empresa de importação de mercadorias. A empresa efectua as suas compras através de contratos que são firmados com fornecedores estrangeiros.

Cada contrato diz respeito a várias mercadorias, as quais são identificadas através de um código. Do contrato consta também a data de assinatura, o prazo de validade, a moeda e o valor.

É fixado no contrato o preço unitário de compra de cada mercadoria, bem como a quantidade comprada, especificada numa unidade de medida. Esta unidade de medida é sempre a mesma para cada mercadoria, independentemente do contrato.

É necessário também guardar informação (nome, endereço, telefone e fax) sobre os vários fornecedores os quais são identificados por um código.

As mercadorias envolvidas num contrato são todas enviadas num único transporte (identificado por um número), relativamente ao qual é necessário conhecer o nome do navio, e as datas previstas de partida e chegada.

A primeira fase passa por definir as entidades e os seus atributos relevantes. **Ou seja, o analista deverá colocar-se as seguintes questões:**

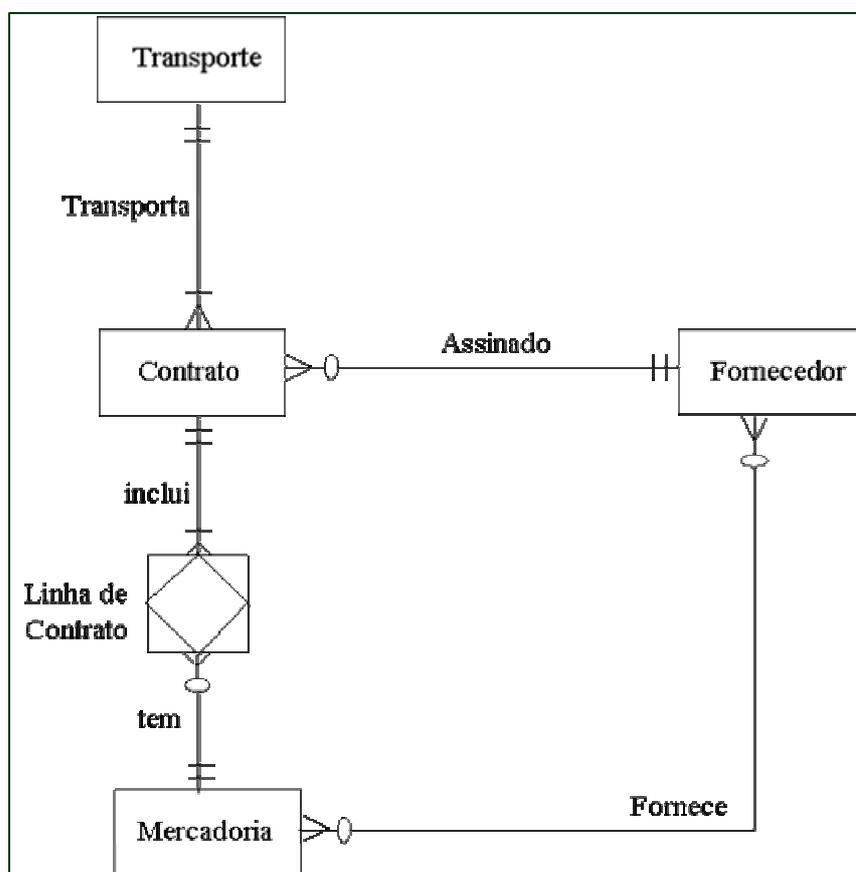
- **Para gerir esta unidade de negócio que informação necessito guardar?**
- **Como vou estruturar essa informação, tendo em atenção os objectivos do modelo (informação completa, flexível e não redundante)?**

Quanto mais cuidada for a análise nesta fase, mais fácil se tornará a elaboração do modelo.

Podem fazer-se, como apoio à construção do modelo, uma tabela do tipo:

Fornecedor	Contrato	Mercadoria	Transporte	<Linha de Contrato>
<u>Cod Fornecedor</u>	<u>Número Contrato</u>	<u>Cod Mercadoria</u>	<u>Cod Transporte</u>	<u>Número Contrato</u>
Nome	Data_Assinatura	Designação	Navio	<u>Cod_Mercadoria</u> ,
Morada	Prazo	Unidade_Medida	Data_Partida	Preço
Telefone	Moeda		Data_Chegada	Quantidade
Fax	Valor			

O próximo passo é a elaboração do modelo, que poderá ter o seguinte aspecto:



A partir do modelo, e em função das entidades definidas e respectivos atributos, bem como das associações e suas propriedades, é derivado um conjunto de um conjunto de tabelas de dados, com os atributos relevantes para o sistema de informação em estudo. Existem um conjunto de regras para efectuar esta derivação, sendo que a **cada modelo deverá corresponder um e um só esquema de tabelas**. Ou seja, a subjectividade inerente à actividade de análise está presente apenas na primeira fase (construção do MEA), sendo que, **se o modelo tiver deficiências, as tabelas dele resultantes irão sofrer dos mesmos problemas**.

As tabelas são sempre constituídas pelos atributos que seleccionámos previamente para cada entidade e, eventualmente, se o grau da associação o justificar, por chaves estrangeiras (chaves de outras tabelas com as quais se encontra associada). Existem associações que podem dar origem a tabelas (as de grau M:N), como veremos mais à frente.

O esquema de tabelas correspondente ao modelo anterior é o seguinte:

Transporte (<u>Cod_Transporte</u> , Navio, Data_Partida, Data_Chegada)
Contrato (<u>Número_Contrato</u> , Data_Assinatura, Prazo, Moeda, Valor, Cod_Transporte Cod_Fornecedor)
Mercadoria (<u>Cod_Mercadoria</u> , Designação, Unidade_Medida)
Linha de Contrato (<u>Número_Contrato</u> , <u>Cod_Mercadoria</u> , Preço, Quantidade)
Fornecedor (<u>Cod_Fornecedor</u> , Nome, Morada, Telefone, Fax)
Fornece (<u>Cod_Fornecedor</u> , <u>Codigo_Mercadoria</u>)

Note que a tabela **Contrato** tem, para além dos seus atributos próprios (Data_Assinatura, Prazo, Moeda, Valor), as chaves estrangeiras das tabelas **Fornecedor** e **Transporte**.

Posteriormente, a cada uma das tabelas resultantes, aplica-se a **teoria da normalização**, constituída por um **conjunto de regras às quais as tabelas resultantes devem obedecer, para cumprirem os objectivos de integridade e não redundância pretendidos para a base de dados**. A aplicação das três formas normais, que veremos mais à frente, pode resolver alguns dos problemas que as tabelas possam apresentar, resultantes de erros ou incorrecções na primeira fase. **Se a análise e o respectivo modelo estiverem bem feitas, o conjunto de tabelas resultantes já deverão estar normalizadas**.

Finalmente, as tabelas já normalizadas serão implementadas (construídas no sistema informático) através do recurso a um Sistema de Gestão de Bases de Dados relacional.

Conceitos Básicos: Entidades, Atributos e Associações.

Entidade

Um dos conceitos fundamentais do modelo entidade-associação é o conceito de **Entidade**:

Uma Entidade é qualquer objecto ou conceito, com interesse para a organização ou para o sistema em estudo, sobre o qual é importante guardar informação, a qual deve ser identificada de uma forma inequívoca.

No modelo entidade-associação, **uma entidade é representada através de um rectângulo, com um substantivo designando o respectivo nome no meio**:

Cliente

Contrato

Produto

Cada entidade é, assim, um conjunto coerente de informação, com características comuns. **Essas propriedades ou características são os chamados Atributos**. A entidade Cliente, por exemplo, poderá ser constituída pelos seguintes atributos: Id_Cliente, Nome, Número de Contribuinte, Morada, etc.

Atributo

Um atributo é uma propriedade de uma entidade, relevante para o sistema em análise. Deverá ser um elemento atómico, ou seja, indivisível em função das necessidades de tratamento da informação que se perspectivem.

Atente, por exemplo, ao atributo Morada, da entidade Cliente. Ao introduzir a informação do cliente Xpto, Lda, “Rua A, nº12- 2º Esq. 4567-123 Lisboa”, esta fica associada a este cliente, por forma a que, sempre que necessitar de a consultar, basta aceder ao respectivo registo. No entanto, se precisar, por exemplo, de listar os clientes de Lisboa, ou de ordenar a base de dados por códigos postais, o facto desta informação se encontrar toda em conjunto, no atributo “Morada”, pode dificultar ou mesmo impossibilitar este tipo de operações. Ou seja, neste caso, a “morada” não é um elemento atómico, pelo que será aconselhável considerar a sua divisão em, pelo menos, três atributos: “endereço”, “código postal” e “localidade”.

Os atributos poderão ser de dois tipos:

- **Identificadores** – é o atributo (ou conjunto de atributos) que identificam, de forma inequívoca, uma ocorrência específica dessa entidade, distinguindo-a das restantes. Para um atributo ser identificador, não podem existir, em caso algum, duas ocorrências distintas dessa entidade em que esse atributo assuma o mesmo valor. **Todas as entidades tem que ter, obrigatoriamente, pelo menos um atributo identificador.** Há situações, nomeadamente em associações que originam tabelas ou entidades associativas, em que, como seguidamente veremos, o identificador pode ser obtido através da junção das chaves de outras tabelas.
- **Descritores** - são os restantes atributos, que apenas descrevem ou caracterizam as várias ocorrências de uma entidade.

Por exemplo, na tabela seguinte, tanto o número de empregado como o BI podem ser identificadores, ou seja, ambos são chaves candidatas da tabela. Neste caso, deveremos escolher uma delas. No entanto, poderão existir funcionários com o mesmo nome, categoria ou data de admissão, pelo que estes não são chaves candidatas.

Tabela resultante da Entidade Empregado

Nº Empregado	Nome	BI	Categoria	Data de Admissão
2635	Carlos Dias	34534543	A	01-03-2000
2232	João Anjos	89789789	B	12-03-2000
1231	Ana Silva	63453453	A	12-03-2000
3432	Luís Meireles	78678678	C	01-03-2000
1234	Ana Silva	75678678	C	01-03-2000

Os atributos podem assumir diferentes valores dentro de um domínio. Por exemplo, poderá ser definido que o número de telefone tem de ser um número inteiro com 9 dígitos, ou que a nota de um aluno só poderá assumir valores de 0 a 20. No que concerne aos identificadores, é conveniente que estes sejam curtos, simples e inequívocos, para facilitar posterior implementação e exploração da base de dados.

Associação

No entanto, a informação isolada sobre cada entidade não é, na maior parte das vezes, suficiente. É necessário obter informação cruzada entre as várias entidades, como, por exemplo “Quais os contratos assinados com um dado cliente?” ou “quais os produtos que os clientes já compraram”, entre muitos outros. Chegamos assim ao conceito de **Associação**:

Uma associação é uma interligação relevante entre as entidades do sistema.

Na notação que adoptamos, **as associações são representadas por uma linha e tem um verbo que as identifica.**



As associações não são direccionais, podendo ser interpretadas do ponto de vista de cada uma das entidades que a constituem. No caso da associação acima indicada, esta irá informar, relativamente aos clientes, quais os contratos que lhes pertencem, e relativamente aos contratos, com que clientes foram assinados.

Os nomes dos componentes do modelo (Entidades e Associações) devem ser claros e representativos, para que não haja lugar a interpretações dúbias. A identificação das entidades, nomeadamente das que são fundamentais para o funcionamento do sistema não oferece, normalmente, dificuldades de maior. Já no que concerne às associações, a sua identificação e validação requer alguma perspicácia e, também, alguma prática. Num sistema de informação, é muito frequente termos a ideia de que todas as entidades estão, de alguma forma, relacionadas entre si: a identificação daquelas associações que são fundamentais e intrínsecas ao sistema não é, muitas vezes, tarefa fácil.

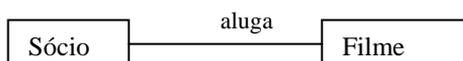
Exemplo

Vejamus um pequeno exemplo, muito simples, para melhor ilustrar este conceitos. Suponha que pretende informatizar um o sistema de aluguer de filmes de um clube de vídeo. O primeiro aspecto importante passa por identificar que “objectos de informação” (“entidades”) são necessários para o funcionamento do clube. Será natural que o clube de vídeo necessite de guardar vários tipos de dados (“atributos”) que caracterizem os seus sócios (como o nome, morada, telefone, ...), os filmes (título, género, actores, ano de realização,...), etc.

No Modelo Entidade - Associação inerente ao funcionamento do sistema de informação do clube de vídeo, o conjunto dos seus sócios é representado pela entidade “Sócio”, enquanto que o conjunto dos filmes é representado pela entidade “Filme”. Estes são objectos de informação fundamentais para o seu funcionamento, pois não é razoável construir um sistema de informação para o clube de

vídeo sem ter dados disponíveis sobre os seus sócios ou sobre os filmes que estão disponíveis para aluguer.

No entanto, não basta ter informação disponível sobre os sócios ou sobre os filmes. é fundamental, por exemplo, poder saber quais os filmes que um sócio alugou. Sendo assim, podemos e devemos estabelecer associação entre as entidades “sócio” e “filme”.



Numa análise mais pormenorizada, naturalmente surgirão outras entidades e associações relevantes, associadas ao sistema de informação de gestão do clube de vídeo. Um modelo entidade-associação de um sub-sistema organizacional poderá envolver dezenas ou centenas de entidades com várias associações entre si, por forma a garantir a flexibilidade, segurança e não redundância da base de dados. A representação apresentada é apenas um exemplo elementar.

Tipos de Associação

Como vimos, uma associação estabelece uma relação entre uma ou várias entidades.

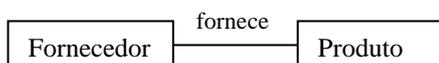
Uma associação pode ser de vários tipos:

- **Binária** - associação entre duas entidades;
- **Complexa** - associação entre várias entidades;
- **Unária** - associação de uma entidade consigo própria.

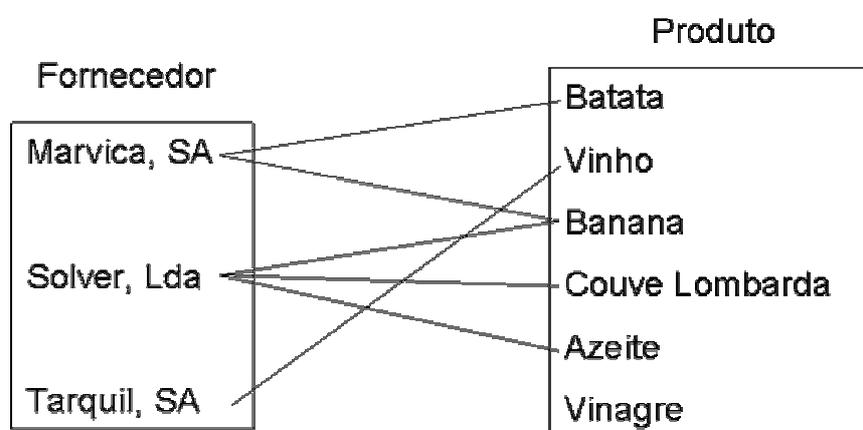
Associações Binárias

As associações mais frequentes são as associações binárias. A grande maioria das associações encontradas no mundo real podem ser representadas através de associações binárias, isto é associações entre duas entidades. As associações unárias e complexas são menos frequentes.

Por exemplo, se num determinado sistema de informação de uma empresa comercial pretendermos saber quais são os fornecedores dos produtos que comercializamos, representamos esse facto através de uma associação entre as entidades produto e fornecedor, da seguinte forma:



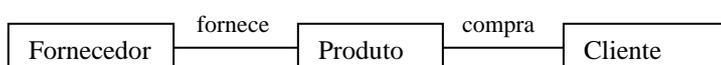
Um possível conteúdo das tabelas e as ligações entre os seus elementos está expresso na figura seguinte:



Associações Complexas

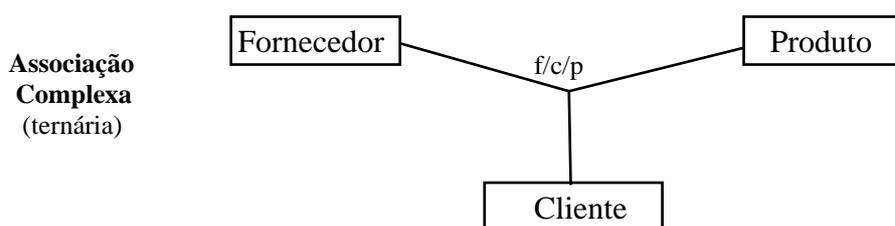
As associações complexas representam associações de várias entidades entre si. São muito menos frequentes do que as associações binárias, sendo paralelamente mais complexa a sua análise. Normalmente, a informação que estas associações dão pode ser obtida através de várias associações binárias. Há, no entanto, casos em que tal não é possível.

Por exemplo, numa empresa comercial que comercializa (compra e vende) produtos alimentares indiferenciados (como, por exemplo, fruta) provenientes de diversos fornecedores, pode haver interesse em saber o que um determinado cliente comprou (tendo para tal que estabelecer uma associação cliente-produto), ou quais os produtos fornecidos por um fornecedor (associação fornecedor-produto). Tal pode ser resolvido com duas associações binárias:



No entanto, se quisermos informação mais detalhada como, por exemplo, saber quem foi o fornecedor que nos forneceu um determinado produto específico que foi comprado por um

cliente específico, sabendo nós que o mesmo tipo de produto pode ser fornecido por vários fornecedores, as duas associações binárias anteriormente referidas não permitem responder a esta questão. Permitem identificar quem são os fornecedores que nos fornecem um determinado tipo de produto e quem são os clientes que compraram esse mesmo tipo de produto, mas não seria possível saber que a fruta que o cliente X comprou foi fornecida pelo fornecedor Z, e não por outro qualquer. Ora esta situação só poderia ser respondida por uma associação complexa entre as entidades cliente, produto e fornecedor, da forma representada na figura seguinte:



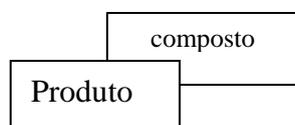
Na prática, este tipo de associações acabam normalmente por originar Entidades Associativas, pois normalmente existe sempre um atributo próprio que seja relevante juntar à associação. No caso anterior, poderia ser relevante saber, por exemplo, a **data da transacção**, o que transformaria a associação **f/c/p** numa entidade associativa, conceito que veremos mais tarde.

Associações Unárias

O tipo de associações que vimos até aqui associam elementos de uma entidade com elementos de outra ou outras entidades. Há, no entanto, casos em que pode ser conveniente obter informação cruzada entre elementos da mesma entidade. As associações unárias associam uma entidade com ela própria.

Vejamos o exemplo dos computadores pessoais numa loja de informática:

Um computador pode ser vendido directamente ao público como um produto final, mas cada um dos seus componentes (placa de som, monitor, disco, memória, etc) também poderão ser comercializados directamente ao cliente. Para o sistema de informação da loja, poderá ser importante registar quais os componentes que estão incorporados em cada um dos diferentes modelos de computador, para efeitos de actualização dos dados ou definição dos preços (em função do preço dos componentes, que na indústria da informática estão a ser constantemente alterados). Na medida em que um produto - neste caso um componente, poderá estar associado a outro produto - o computador - esta situação deverá ser representada através de uma associação unária.



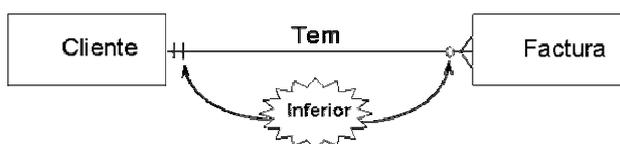
Graus de Associações

Para a elaboração do modelo e consequente derivação para o esquema relacional de tabelas, não é suficiente a identificação das associações. Paralelamente, dever-se-á indicar no modelo qual o número mínimo e máximo de ocorrências de uma entidade que poderão estar associadas com as outras ocorrências, ou seja, definir o grau da associação ou cardinalidade. A representação é feita incluindo em cada um dos lados da associação os símbolos seguintes:

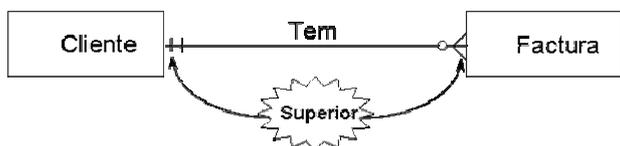
○	Zero
	Um
≤	N – Vários

A cada extremo de uma associação correspondem dois graus:

- **Inferior** - Número mínimo possível de ocorrências ("0" ou "1")



- **Superior** - Número máximo possível de ocorrências ("1" ou "N")



O número mínimo de ocorrências é colocado na parte interior e está relacionado com a obrigatoriedade, isto é com a dependência entre as ocorrências das entidades e acarreta, para além da análise efectiva da realidade, opções sobre a forma de implementação.

- Poderá ser 0 – pode haver pelo menos uma ocorrência que não está (ainda que temporariamente) relacionada com a outra.
- ou 1 – em que se **obriga** a que exista a relação, ou seja, uma ocorrência não pode existir sem a outra.

O número máximo define o grau de associação e assume particular importância na forma como o modelo se irá reflectir na estrutura das tabelas resultantes. Tendo em conta esse número máximo, as associações podem apresentar os seguintes graus de associação:

- um-para-um (1:1)
- um-para-vários (1:N)
- vários-para-vários (M:N)

No caso anterior, quando se indica que um cliente tem no mínimo 0 (zero) facturas associadas, tal implica que se pode registar alguém como cliente, sem ter obrigatoriamente que registar uma factura. Na prática, podemos ter, na nossa tabela de clientes, para além dos clientes efectivos (os que já compraram alguma coisa, tendo pelo menos uma factura associada), os potenciais clientes, que ainda não compraram nada (não têm facturas associadas). O número máximo é, naturalmente N, ou seja um cliente pode ter mais do que uma factura associada (pode fazer mais do que uma compra).

Por outro lado, a cada factura está associado um e um só cliente, o que, na verdade, poderá fazer sentido na maior parte dos casos. Se tivesse colocado como mínimo 0, permitiria a inserção de facturas sem se saber qual é o cliente a que pertencem. Se, no número máximo se optasse por N, a base de dados resultante permitiria que a mesma factura pertencesse a vários clientes.

Associações de Um para Um (1:1)

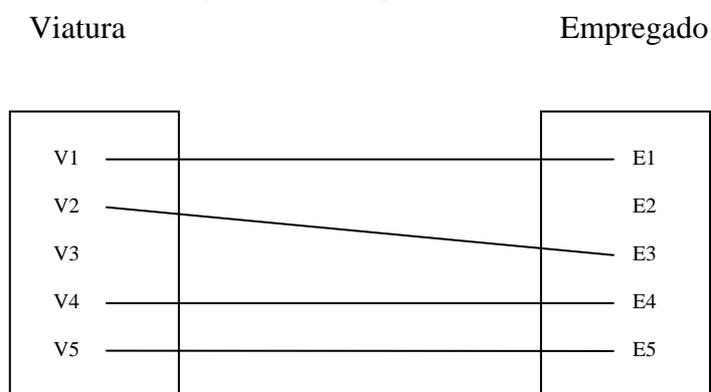
O grau de associação entre duas entidades diz-se de “um para um” quando a cada ocorrência da Entidade A está associada no máximo uma ocorrência da entidade B e a cada ocorrência da entidade B está associada no máximo uma ocorrência da entidade A.

Exemplo:

Numa determinada empresa, cada viatura está atribuída a um e um só empregado.



As ocorrências e as associações são as seguintes:



Repare que, neste caso, há viaturas que não estão (ainda...) atribuídas a empregados, caso da V3, e vice-versa, caso de E2, o que justifica o “zero” como mínimo. Repare que, neste caso, se por exemplo, uma viatura mudasse de um empregado para outro, a base de dados resultante apenas poderia guardar os dados referentes à última atribuição, sendo por isso,

perdida a informação histórica. Esta opção é, por isso, limitativa, devendo ser aplicada com o devido cuidado.

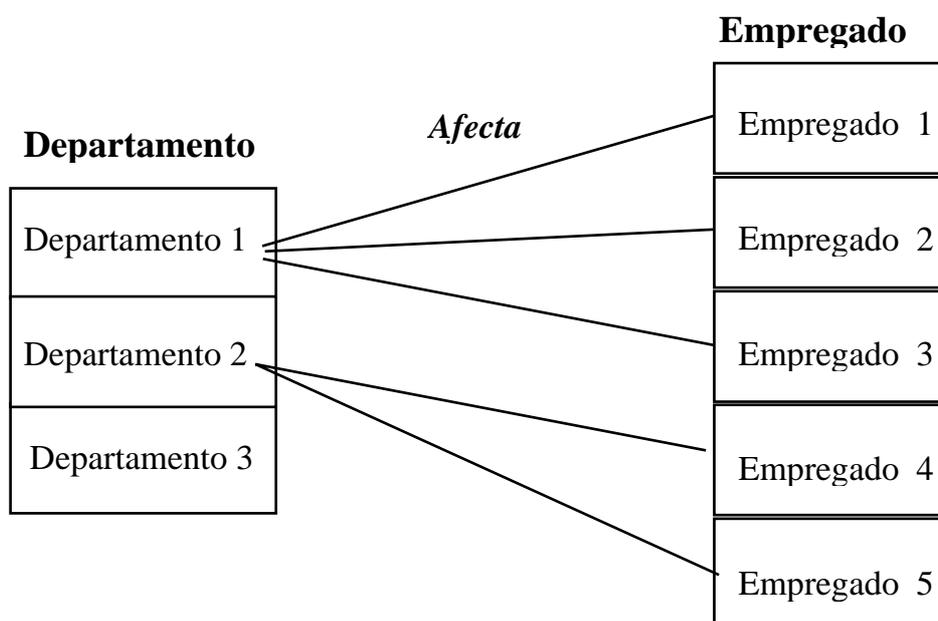
Associações de Um para Vários(1:N)

O grau de uma associação binária diz-se de “um para vários” quando a cada ocorrência da Entidade A estão associadas várias (uma ou nenhuma) ocorrência da entidade B, mas a cada ocorrência da entidade B está associada no máximo uma ocorrência da entidade A.

Por exemplo, um departamento pode ter vários empregados mas cada empregado está afecto apenas a um departamento. Esta situação seria representada por uma associação de 1 para N, da seguinte forma:



Vejamos melhor o conceito de ocorrência, através do seguinte exemplo:



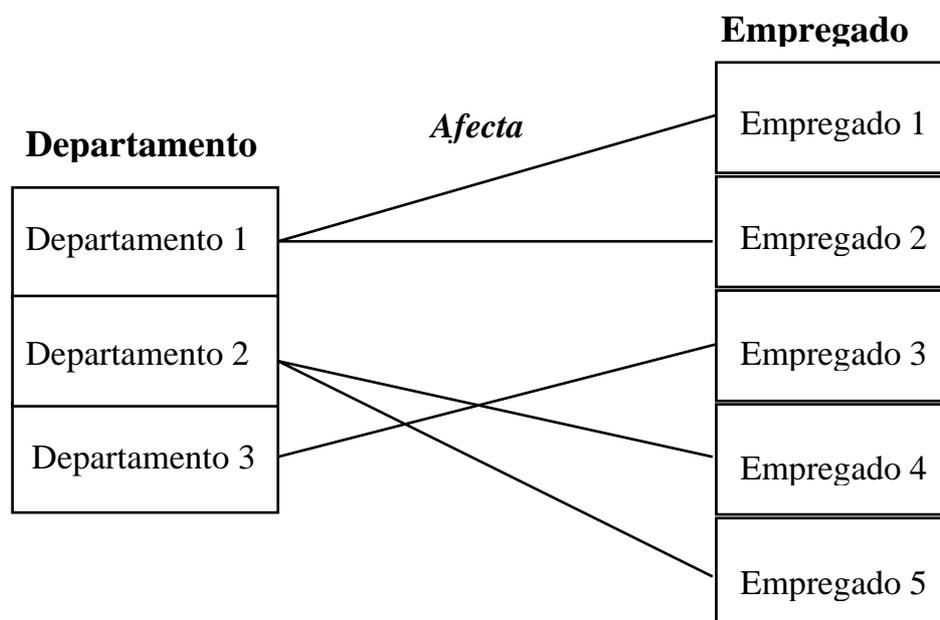
Neste exemplo:

- A entidade DEPARTAMENTO apresenta 3 ocorrências;
- A entidade EMPREGADO apresenta 5 ocorrências;

Neste caso, todos os empregados têm que estar afectos a departamentos, ou seja, não é possível introduzir um novo empregado na tabela sem indicar imediatamente qual o departamento a que pertence. Por outro lado, o Dep. 3 não tem empregados associados. Esta é a situação actual de ocorrências na base de dados. Não significa que o D3 fique eternamente sem empregados, pois em qualquer altura poderemos afectá-los. Significa

apenas que é possível criar um novo departamento, e só mais tarde proceder à afectação dos respectivos empregados.

Ainda relativamente ao exemplo anterior, pensemos no que acontece se um empregado mudar de departamento. No caso, o Empregado 3 passou para o Departamento 3.

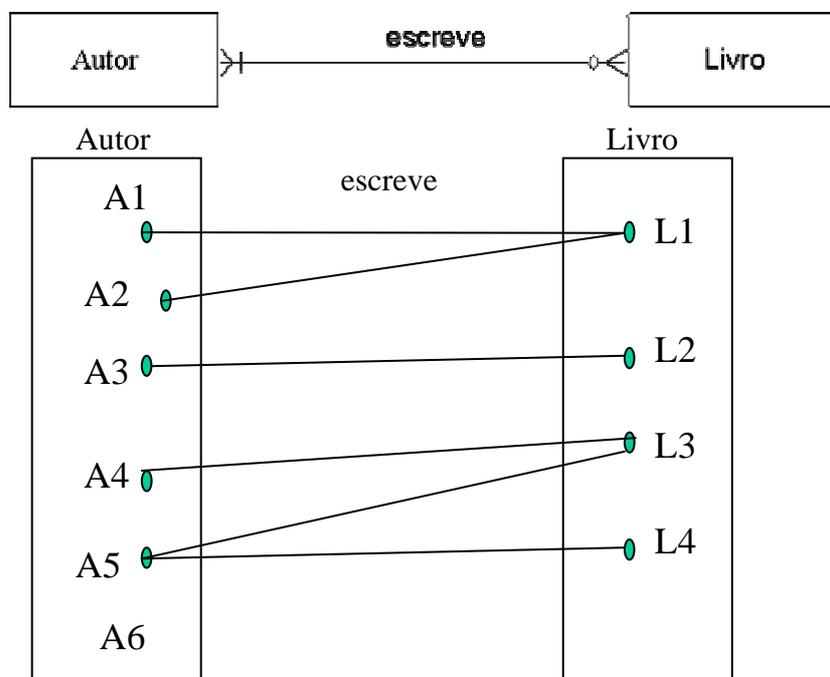


Como a cada empregado só pode corresponder um departamento, o que irá ficar registado na base de dados é o departamento ao qual esse empregado passou a pertencer. Como resultado, perde-se a informação do(s) departamento(s) onde este esteve anteriormente. Se essa informação fosse considerada relevante, a base de dados, estruturada desta forma, não permitiria guardá-la. Neste caso, a resposta seria considerar que o grau da associação, do lado do Departamento, seria também N. Ou seja, um empregado poderá estar afecto a vários departamentos, embora em diferentes períodos.

Associações de Vários para Vários (N:M)

A cada ocorrência da Entidade A estão associadas várias (uma ou nenhuma) ocorrências da entidade B, e a cada ocorrência da entidade B estão associadas várias (uma ou nenhuma) ocorrências da entidade A.

Por exemplo, um livro pode ser escrito por vários autores, assim como o mesmo autor pode escrever vários livros.



Neste caso, o autor A6 já consta da base de dados, mas ainda não está registado nenhum livro da sua autoria (daí o zero como mínimo).

No sentido de clarificar melhor a problemática dos graus de associação vejamos agora o seguinte exemplo:



1ª hipótese:

- Um professor lecciona no máximo um curso;
- Um curso é leccionado no máximo por um professor;

Neste caso o grau de associação será 1 para 1.



2ª hipótese:

- Um professor pode leccionar vários cursos;
- Um curso é leccionado no máximo por um professor;



Neste caso o grau de associação será 1 para N.

3ª hipótese:

- Um professor pode leccionar vários cursos;
- Um curso pode ser leccionado por vários professores;

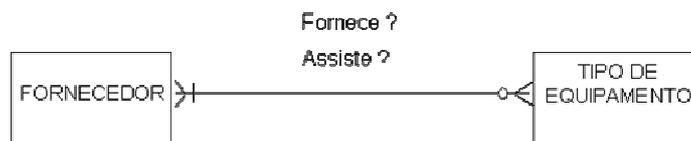
Neste caso o grau de associação será M para N.



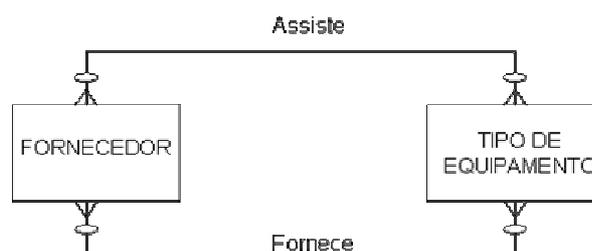
Podem existir várias associações entre as mesmas entidades. Daí a importância de se definir um nome adequado para a associação, por forma a podermos saber exactamente qual a informação que estas nos dão.

Por exemplo, numa empresa, um determinado equipamento é fornecido e assistido por vários fornecedores. No entanto, o fornecedor que assiste esse equipamento pode não ser o mesmo que o forneceu.

Representar este caso com uma associação única entre as duas entidades não resolve o problema:



Sendo assim, a situação terá obrigatoriamente que de ser representada por duas associações, como podemos observar na figura seguinte:



Entidade-Associativa

Uma associação pode limitar-se a relacionar entidades entre si, como, por exemplo, uma associação que relaciona “Empregado” com “Departamento”:

- os empregados afectos a um departamento;
- e a que departamento está afecto um empregado.

Esta associação é representada, como vimos, da seguinte forma:



Suponha, no entanto, que um empregado muda de departamento. Neste caso, o modelo já não responde a esta situação, pois só permite saber qual o último departamento a que o empregado está afecto. Na verdade, um empregado está afecto a um só departamento, de cada vez, mas ao longo do tempo, pode pertencer a vários departamentos. Nesse caso, o modelo tem o seguinte aspecto:

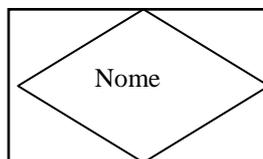


Desta forma, passamos a saber **quais os departamentos pelo qual o empregado foi passando, mas não sabemos quando...** Ou seja, será provavelmente relevante guardar a data de início e a data de fim da actividade do empregado no departamento. A questão é, agora, saber onde colocar estes dois atributos. **Ora, os atributos deverão estar na entidade que caracterizam, e da qual dependem.** As datas de início e fim dependem do binómio departamento- empregado: O empregado E1 iniciou em 01/01/2002 a actividade no departamento D1, tendo passado em 01/01/2003 para o Departamento D2... Assim, concluímos que as datas dependem da associação “Afecta”, que adquire assim o estatuto de **associação com atributos ou Entidade Associativa**, passando o modelo a ter o seguinte aspecto:



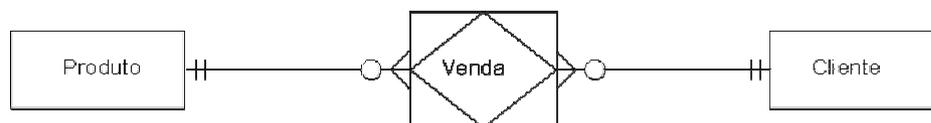
Uma Entidade Associativa (ou associação atributiva) é uma associação com atributos próprios e surge quando um ou vários atributos dependem não de uma entidade, mas de duas ou mais.

É representada pela seguinte figura:



Vejamos um outro exemplo:

Uma associação que relaciona cliente com produto, para além de fornecer informação sobre os produtos vendidos ao cliente, poderá igualmente indicar o preço que o cliente pagou pelo produto e a quantidade comprada.



Neste caso, estamos a assumir que o preço do produto pode ser diferente de venda para venda.

Uma entidade-associativa é, por definição, uma entidade, pois tem atributos próprios. Pode, por isso, ser também representada como tal.

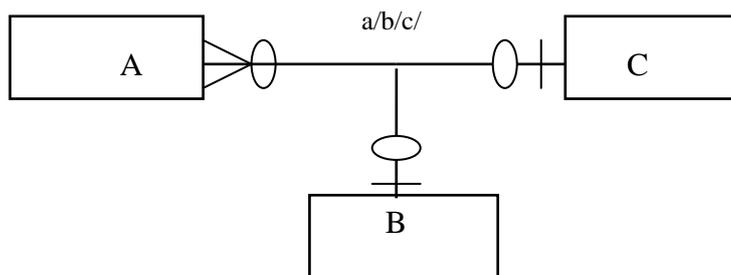
Identificador de uma Associação

As associações, se derem origem a uma tabela, também têm identificador. O identificador de uma associação é constituído pela **concatenação (junção) dos identificadores das entidades associadas**. Por exemplo, na associação de Cliente com Produto, o identificador da associação será normalmente o par de atributos: código_cliente, código_produto.

No entanto nem sempre é assim, como iremos ver posteriormente.

Grau de uma associação complexa

Tal como as associações binárias, as associações complexas podem ter vários graus de associação. Vejamos o seguinte exemplo que representa uma associação complexa entre as entidades A, B e C, com os graus de associação assinalados:



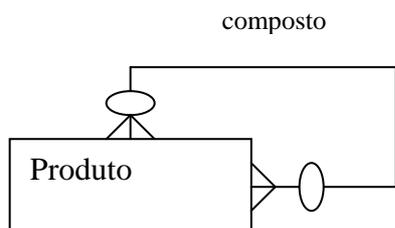
A interpretação dos graus de associação deve ser realizada fixando-se cada par de entidades e analisando qual o número de ocorrências da terceira entidade. Por exemplo:

- Para o par A, B só pode existir, no máximo, uma ocorrência de C;
- Para o par B, C podem existir, no máximo, várias N ocorrências de A;
- Para o par A, C só pode, no máximo, existir uma ocorrência de B.

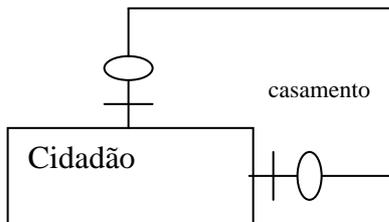
Graus de uma associação unária

A interpretação do grau das associações unárias apresenta algumas semelhanças com as associações binárias, pelo que não vamos debater muito a questão. Vimos anteriormente um exemplo de uma associação que representava o facto de um produto poder ter na sua composição outro produto (exemplo do computador com os respectivos componentes, disco, placa de som, etc).

Neste caso, a associação é de vários para vários, pois um determinado produto pode ter vários produtos seus componentes e, por sua vez, pode ser componente de vários outros: um computador pode ter vários produtos na sua composição, assim como um produto, (por exemplo, um determinado tipo de disco) também pode entrar na composição de diversos modelos de computador.



Vejamos um outro exemplo. A associação casamento, entre dois cidadãos, é uma associação unária de grau 1:1, porque um cidadão, de acordo com o regime legal existente em Portugal, só pode estar casado, num determinado momento no tempo, com um outro cidadão, e vice-versa. Não é possível uma pessoa estar casada com mais do que uma outra pessoa simultaneamente. Sendo assim, e se apenas se pretender guardar os dados do casamento actual esta situação representa-se do seguinte modo:



O facto do mínimo ser 0 (zero) reflecte a não obrigatoriedade do casamento.

Transformação do Modelo Entidade-Associação num Esquema Relacional

Introdução

Para informatizar um subsistema de informação de uma organização não basta definir graficamente os diferentes objectos de informação (entidades) relevantes para esse subsistema. É necessário especificar com detalhe quais os atributos, os tipos de dados que é necessário guardar. As diferentes entidades e associações identificadas no MEA vão dar origem a um conjunto de esquemas de relação, que irão posteriormente resultar nas tabelas de dados que irão ser construídas e manipuladas informaticamente num SGBD.

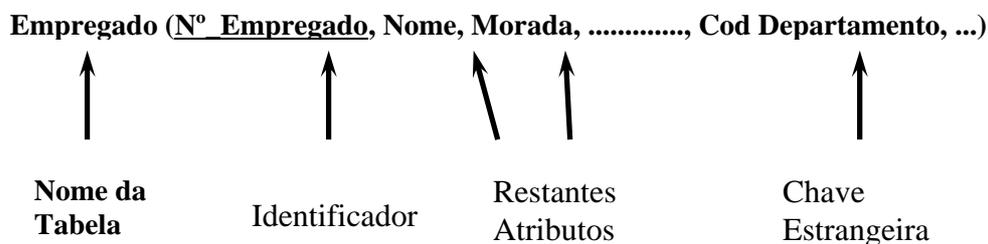
O esquema de tabelas

Ao nome da entidade, associação ou entidade associativa, seguido da lista dos seus atributos chama-se **esquema de relação** ou **tabela**, e é apresentado com a seguinte notação:

Nome_Tabela (Identificador, Descritor1, Descritor2,....., Chave Estrangeira1, Chave Estrangeira2, ...)

O atributo identificador (ou atributos, no caso de chaves compostas) é sublinhado, e separado por vírgulas dos restantes.

Exemplo:



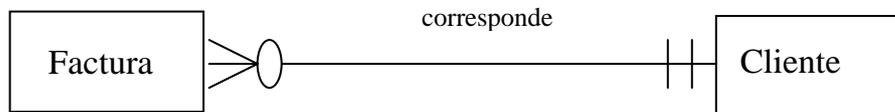
O **esquema relacional** é o conjunto dos esquemas de relação (também designados de tabelas, porque numa fase posterior vão dar origem a tabelas de dados)

Cada Entidade dá normalmente origem a um esquema de relação (ou tabela) com:

- **Identificador** da Entidade (também designado de Chave da Entidade);
- **Descritores** da Entidade - outros Atributos da Entidade que não o identificador, que apenas descrevem as diversas ocorrências da entidade;
- **Chaves Estrangeiras** - identificadores de outras Entidades que eventualmente lhe estejam associadas.

A ligação entre as tabelas é feita exactamente através das introdução dos respectivos identificadores noutras tabelas. Para tal, existem um conjunto de regras que veremos seguidamente.

Por exemplo, para guardar informação sobre as facturas de uma empresa, podemos ter a seguinte associação no MEA:



É uma associação de grau 1: N porque partimos do princípio que um cliente pode ter várias facturas associadas, mas cada factura corresponde a um só cliente. Por outro lado, a obrigatoriedade (o mínimo de 1 do lado do cliente) justifica-se para que não registemos facturas sem um cliente associado. O zero do lado da factura possibilita a existência de clientes que não têm (ainda...) facturas associadas, ou seja, clientes potenciais.

Sendo assim, poderíamos ter uma tabela de facturas com os seguintes atributos:

Factura (num factura, data, valor, código_cliente)

- O **num factura** é o identificador, por isso está sublinhado. Este permite distinguir de forma inequívoca todas as facturas: não poderão existir na base de dados duas facturas com números iguais.
- Os atributos **data** e **valor** são descritores, pois dão-nos informação mais detalhada e relevante sobre cada uma das facturas.
- O **código_cliente** é uma chave estrangeira. O **código_cliente** será naturalmente identificador de uma outra entidade associada (que é a entidade Cliente), que detém os dados relevantes sobre o cliente. Na tabela Factura, o **código_cliente** diz-nos apenas quem é o cliente a que aquela factura corresponde, permitindo estabelecer a relação entre as duas entidades.

Regras de derivação para o esquema relacional

É possível enunciar um conjunto de regras que permitem facilmente identificar qual a estrutura das tabelas que irão derivar, consoante as características das entidades e associações identificadas no Modelo Entidade-Associação. Essas regras, na sua forma original, a partir dos trabalhos de Codd (1970) e Chen (1976), eram mais complexas do que a versão simplificada que aqui apresentamos. Esta simplificação tem a ver com aquilo que é actualmente aplicado na prática, ao nível do desenvolvimento de Bases de Dados.

Associações Binárias

A associação entre duas Entidades (E1 e E2) pode ser resolvida de uma de três formas, que se excluem mutuamente:

- Colocando o identificador de E1 na tabela resultante de E2
- Colocando o identificador de E2 na tabela resultante de E1
- Criando uma nova tabela, cujo conteúdo são os dois identificadores.

Ou seja, são formas alternativas, e em cada situação, apenas uma delas é usada, dependendo do grau da associação.

Associações de Um para Vários (1:N)



Neste caso, cada entidade dará origem a uma tabela, sendo que a chave estrangeira será sempre colocada na tabela resultante da entidade do lado de maior grau (E2), dando assim origem a duas tabelas com a seguinte estrutura:

RE1 (C1, d11, ...,d1n)

RE2 (C2, d21,,d2n, C1)

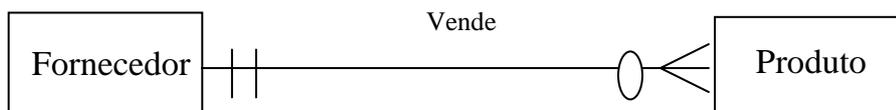
Sendo:

- C1 o identificador da Entidade 1;
- C2 o identificador da Entidade 2;
- d11 a d1n, os descritores da Entidade 1;
- d21 a d2n, os descritores da Entidade 2.

De notar que, neste caso, o grau mínimo não tem qualquer interferência na estrutura de tabelas resultante (pode ser Zero ou Um, de qualquer dos lados da associação).

Anteriormente, vimos alguns exemplos de associações binárias de 1:N.

Por exemplo, uma associação entre produto e fornecedor. Numa determinada situação, um tipo de produto só é fornecido por um fornecedor, mas um fornecedor pode fornecer vários produtos diferentes.



Neste caso, a teoria diz que devemos obter as seguintes tabelas:

Fornecedor (código_fornecedor, nome, morada, nºcontribuinte, ...)

Produto (código_produto, designação, peso, cor, ..., cod_fornecedor)

Repare que é no produto, a entidade que participa com grau N na associação, que deve ficar o código de fornecedor como chave estrangeira.

A atribuição dos graus das associações depende sempre da situação real que se analisa, e das necessidades de informação. Ou seja, noutras circunstâncias, poderia acontecer uma relação de Muitos para Muitos, em que um fornecedor pudesse fornecer vários produtos, mas também que o mesmo produto pudesse ser fornecido por fornecedores diversos.

Associações de Um para Um (1:1)

Este é um caso particular das associações de Um para Vários, e pode ser resolvido de forma análoga. **Neste caso, como ambas as entidades têm o mesmo grau máximo, a associação pode resolver-se colocando a chave estrangeira em qualquer das tabelas.**



RE1 (C1, d11,,d1n)

RE2 (C2, d21,, d2n, C1)

Ou, alternativamente:

RE1 (C1, d11,,d1n, C2)

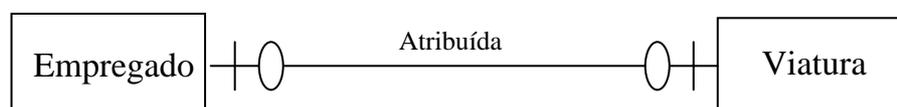
RE2 (C2, d21,, d2n,)

Sendo:

- RE1 o esquema de relação (tabela) referente à Entidade 1;
- RE2 o esquema de relação (tabela) referente à Entidade 2;
- C1 o identificador da Entidade 1;
- C2 o identificador da Entidade 2;
- d11 a d1n, os descritores da Entidade 1;
- d21 a d2n, os descritores da Entidade 2.

Vejamos um exemplo concreto:

Vamos partir do princípio que estamos a informatizar a atribuição de viaturas aos empregados de uma dada empresa, em que cada empregado tem atribuído um e só um automóvel e vice-versa. Sendo assim, temos uma situação deste tipo, associação binária de grau 1:1:



Neste caso, como o mínimo é zero em cada um dos lados, tal significa que pode haver, ainda que temporariamente, empregados sem viaturas atribuídas ou viaturas sem empregado atribuído. Neste caso, podem haver duas soluções:

Viatura (matricula, marca, modelo, ano_registro, cilindrada, ..., ID_empregado)

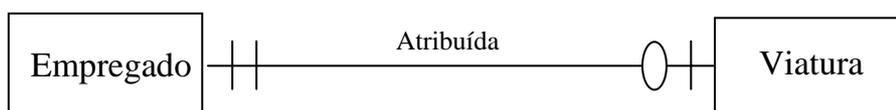
Empregado (ID_empregado, nome, morada, data_nascimento, nºcarta_condução, ...)

Ou

Viatura (matricula, marca, modelo, ano_registro, cilindrada, ...,)

Empregado (ID_empregado, nome, morada, data_nascimento, nºcarta_condução,, matricula)

Neste caso, ambas as soluções funcionam. Assim, colocar a chave estrangeira de um lado ou de outro, terá apenas a ver com a o facto de se dever promover a economia de espaço em disco. Se, por exemplo, tivermos uma situação em que praticamente todas as viaturas estão atribuídas, e só uma pequena parte dos empregados têm carro, então a chave estrangeira deverá ser colocada na tabela de viaturas. Ou seja, a mesma solução em que exista obrigatoriedade da parte do empregado, isto é, que cada viatura esteja obrigatoriamente alocada a um empregado, mas que possam existir empregados sem viatura atribuída:



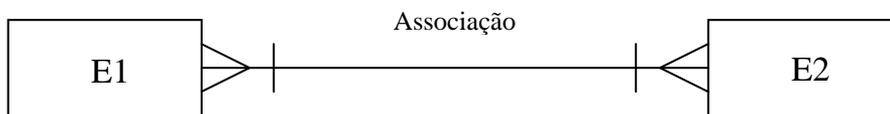
A solução mais lógica, neste caso, seria:

Viatura (matricula, marca, modelo, ano_registro, cilindrada, ..., ID_empregado)

Empregado (ID_empregado, nome, morada, data_nascimento, nºcarta_condução, ...)

De realçar que esta solução pode configurar-se como muito limitativa. Se, por exemplo, um empregado mudar de viatura, a base de dados só guarda o último registo, perdendo-se o anterior.

Associações de Vários para Vários (M:N)



Independentemente das obrigatoriedades, esta situação é sempre representada por três esquemas de relação, um para cada entidade e um para a associação.

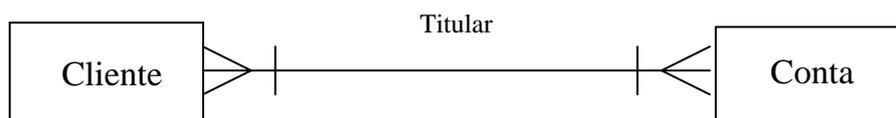
RE1 (C1, d11, ..., d1n)

RE2 (C2, d21, ..., d2k)

RA(C1, C2)

O identificador da tabela referente à associação é sempre a junção dos identificadores das entidades que estão associadas, neste caso C1 e C2.

Vejamos também um exemplo para este caso. Uma conta bancária pode ter um ou vários titulares (clientes), assim como um cliente pode abrir várias contas em seu nome no banco. Sendo assim esta será uma associação de M:N (no caso das associações de M:N não é importante analisar as obrigatoriedades porque a situação resolve-se sempre da mesma forma independentemente das obrigatoriedades).



Para este caso, a teoria diz que temos 3 tabelas:

Conta (nºconta, data_abertura, saldo, ...)

Cliente (nºcliente, nome, morada, telefone, BI, ...)

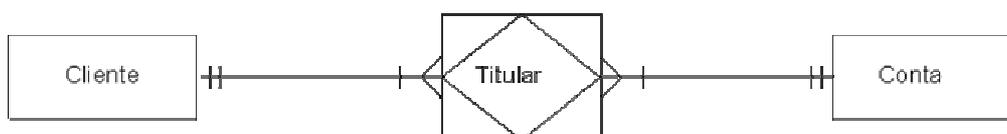
Titular(nºconta, nºcliente)

A tabela “Titular” poderá ter o seguinte conteúdo:

nºconta	nºcliente
01225500	3150
01225500	3200
01233333	3456
01233333	3150
01233333	3333
01235444	5555
...	...

Repare que, embora os atributos **nºconta** e **nº cliente** possam estar repetidos, a sua junção, que é a chave da tabela, nunca pode ter repetições. Na prática, isto quer dizer que não podemos registar na base de dados o mesmo cliente com a mesma conta duas vezes.

Eventualmente, esta associação poderá transformar-se numa entidade associativa se entendermos relevante definir qual é o grau de titularidade. O atributo que define a titularidade não é um atributo do cliente, porque varia consoante as contas que este tiver (ele pode ser primeiro titular numa conta e não em outra), nem é um atributo da conta, porque depende naturalmente do cliente (só um cliente é que é primeiro titular).



Conta (nºconta, data_abertura, balcão, saldo, ...)

Cliente (nºcliente, nome, morada, telefone, BI, ...)

Titular(nºconta, nºcliente, titularidade)

No atributo “titularidade”, iremos colocar a informação referente ao tipo de titularidade de cada cliente em cada conta. A tabela resultante ficaria com um aspecto semelhante ao seguinte:

nºconta	nºcliente	titularidade
01225500	3150	1º
01225500	3200	2º
01233333	3456	1º
01233333	3150	2º
01233333	3333	3º
01235444	5555	1º
...

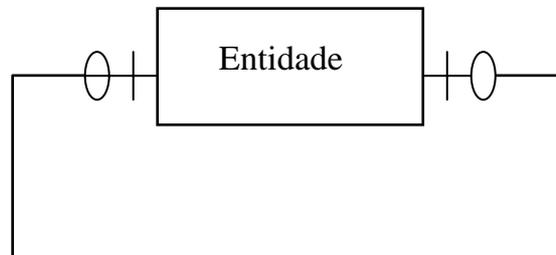
Repare que, neste caso, o cliente nº 3150 é 1º titular da conta nº 01225500, e 2º titular da conta nº 01233333.

Associações Unárias

A forma de derivação para o esquema relacional para as associações unárias obedece exactamente ao mesmo esquema de raciocínio que utilizámos para as associações binárias. A única diferença será a de que, neste caso, e uma vez que estamos a relacionar elementos de uma entidade com outros elementos da mesma entidade, a “chave estrangeira” é, na verdade, a chave da própria tabela, com o nome adaptado ao tipo de informação que guarda.

Vejamos alguns exemplos.

Associações Unárias, grau 1:1



Associação

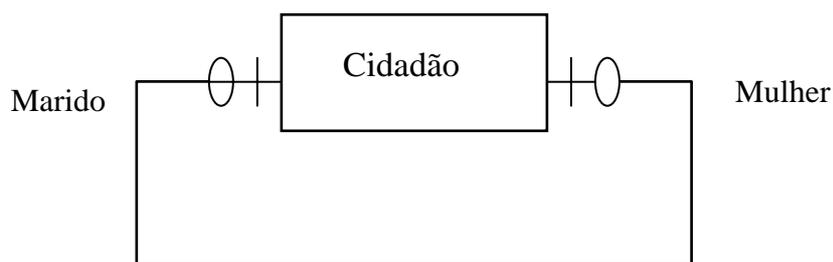
Esta situação pode ser representada por um só esquema de relação:

RE(C1,d1,...,dn, Cf)

C1 – Identificador da Entidade.

Cf – representa o identificador do outro elemento da entidade com a qual este está relacionado.

Vejamos o exemplo do casamento, anteriormente citado. A associação casamento é uma associação unária, porque é uma associação entre duas pessoas, ou dois cidadãos. Tem grau 1:1, porque um cidadão só pode estar casado, num determinado momento no tempo, com um outro cidadão, e vice-versa. Não tem obrigatoriedades porque poderão existir cidadãos que não sejam casados.



casamento

Este exemplo dá origem à seguinte tabela:

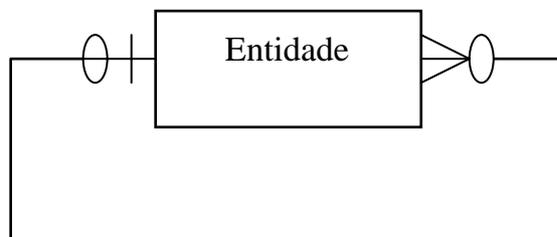
CIDADÃO (N°_Bi, Nome, Morada,, N°_Bi_Mulher)

Ou, alternativamente,

CIDADÃO (N°_Bi, Nome, Morada,, N°_Bi_Marido)

É indiferente a opção por qualquer uma destas situações. No primeiro caso, introduz-se na tabela o BI da mulher, no registo do respectivo marido, e no segundo faz-se o oposto.

Associações Unárias, grau 1:N

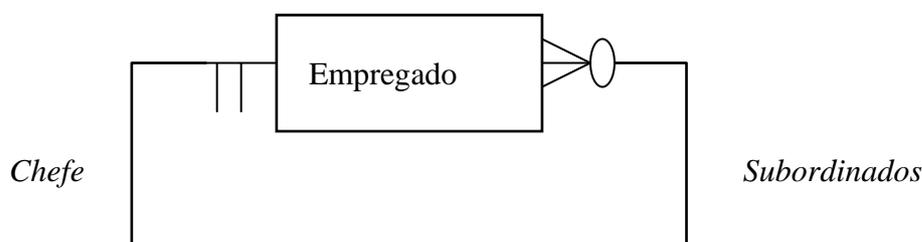


Associação

Esta situação é representada por um único esquema de relação, tal como a situação anterior:

RE(C1, d1,...,dn, Cf)

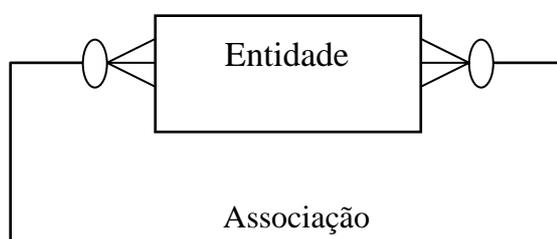
Vejamos o exemplo de uma relação de chefia entre funcionários. Cada funcionário tem obrigatoriamente um chefe, que também é funcionário, mas cada chefe poderá chefiar vários funcionários, por isso o grau de associação é de 1:N. Neste caso, é também necessário ter em atenção que o chefe máximo (topo da hierarquia) tem de ser considerado como chefe de si próprio.



Chefia

EMPREGADO (Nº Emp, nome, morada, BI, categoria,,Nº_Emp_Chefe)

Associações Unárias, grau M:N.



Associação

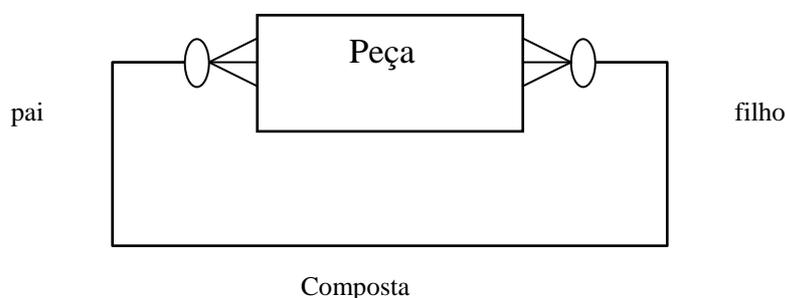
Independentemente das obrigatoriedades (grau mínimo da associação), esta situação é sempre representada por dois esquemas de relação, um para a entidade e outro para a associação, tal como nas associações binárias.

RE1(C1, df1, ..., dfn)

RA(Cp1,Cf1)

Em que Cp1 é a chave do elemento *pai*, e Cf2 é o identificador do elemento *filho*.

Vejamos o seguinte exemplo, em que temos uma entidade “peça”, com os seus atributos próprios, e que pode, por um lado, ser composta por peças mais pequenas (no caso, será o “pai”), e por outro lado, ser componente de outras peças maiores (o “filho”).

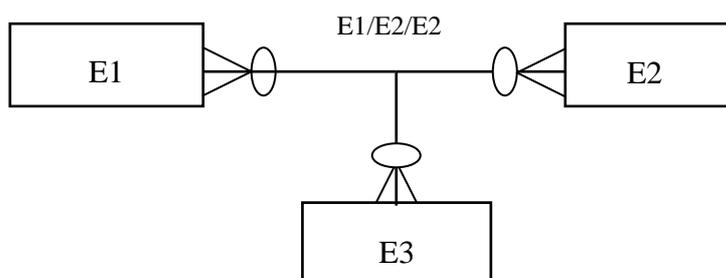


Peça (Cod_Peça, nome, tamanho,.....)

Composta (Cod_Peça_Pai, Cod_Peça_Filho)

Associações Complexas

No caso das associações complexas, independentemente do grau da associação, o esquema relacional é sempre constituído por uma tabela para cada entidade que faz parte da associação, e uma outra tabela para a associação, contendo os identificadores das entidades



RE1 (C1, d11, ..., d1n)

RE2 (C2, d21, ..., d2k)

RE2 (C3, d31, ..., d3m)

RA(C1, C2, C3)

O identificador da tabela referente à associação é sempre a concatenação (junção) dos identificadores das entidades que estão associadas, neste caso C1, C2 e C2.

Normalização do esquema relacional.

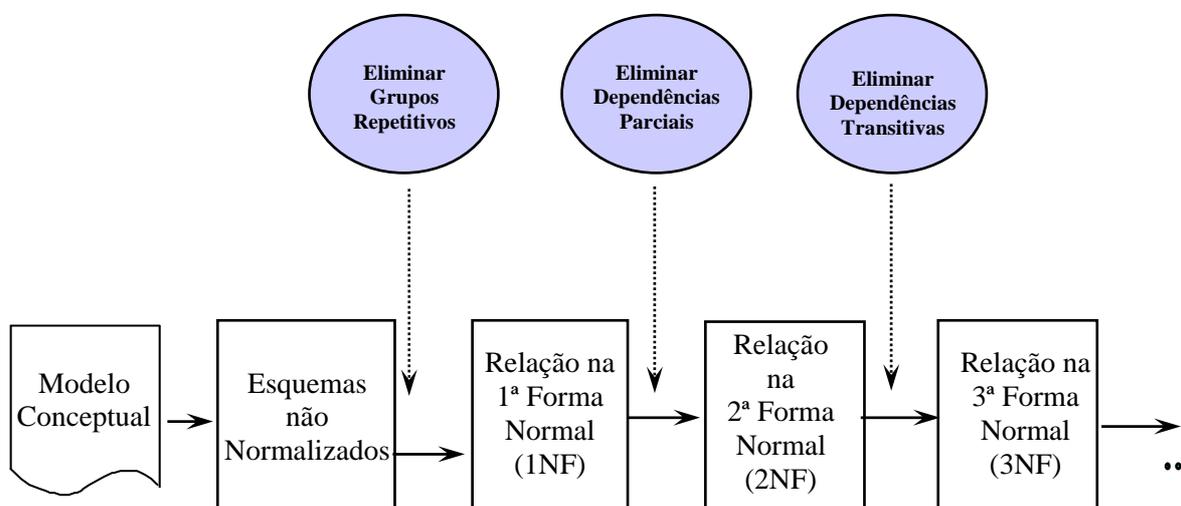
A **normalização** tem como objectivo modificar o conjunto de esquemas de relação obtido a partir do modelo entidade-associação, e transformá-lo num outro conjunto de esquemas equivalente, menos redundante e mais estável.

Após a construção do modelo conceptual dos dados (Modelo Entidade/Associação) é feita a transformação para um Esquema Relacional. Se a análise efectuada para o modelo entidade associação estiver correcta, então as tabelas deverão estar já normalizadas, ou seja, a normalização apenas refina o esquema de tabelas.

A teoria da normalização deve ser aplicada a cada um dos esquemas de relação (tabelas), por forma a validar a sua consistência e garantir a não redundância e estabilidade dos dados.

Esta matéria apresenta alguma complexidade e a apresentação que vamos fazer é relativamente superficial. Vamos apresentar apenas as três primeiras formas normais, sendo que a teoria da normalização pode ser aplicada até à 6ª forma normal.

O gráfico seguinte representa as diferentes fases do processo de normalização, para as três primeiras fases:



1ª Forma Normal

Uma relação está na 1ª Forma Normal se e só se não contém atributos ou grupos de atributos repetitivos.

Por outras palavras, esta forma normal impede que existam relações de Um para Vários dentro da mesma tabela. Um dos indicadores de que uma relação poderá não estar na 1ª forma normal é o facto de existirem nomes de atributos no plural.

Repare na seguinte tabela:

Livro(ISBN, título, preço, autores)

Como um livro pode ter vários autores (repare no nome do atributo escrito no plural) então a relação não está na 1ª forma normal. Ora, se a análise inicial tivesse sido bem feita, aplicando de uma forma correcta os conceitos que vimos anteriormente, ter-se-ia muito provavelmente identificado Autor como entidade.

Sendo também verdade que um autor pode escrever vários livros, então a relação entre Autor e livro é de Vários para Vários, pelo que o esquema, devidamente normalizado, seria o seguinte:

Livro(ISBN, título, preço)

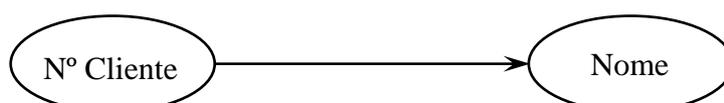
Autor(ID Autor, Nome)

Autoria (ID Autor, ISBN)

2ª Forma Normal

Uma relação está na 2ª Forma Normal se está na 1ª forma normal, e se todos os atributos descritores dependerem funcionalmente da totalidade do identificador. A validação desta forma normal só é, assim, aplicável em tabelas com chaves compostas.

Um conceito importante para perceber a normalização é o conceito de dependência funcional. Por vezes, dois atributos (ou dois grupos de atributos) estão intrinsecamente ligados entre si. Exemplo:



Num determinado momento, em qualquer ponto da Base de Dados onde figurem estes dois atributos, ao mesmo **nº de cliente** corresponderá necessariamente o mesmo **nome** (mas o inverso poderá não ser verdadeiro, ou seja, podem existir clientes com nomes iguais).

Nº Cliente → Nome_Cliente

Diz-se então que o **Nome_Cliente** "*depende*" do **NºCliente**, ou que **NºCliente** "*identifica*" o **Nome_Cliente**. Existe, por isso, uma **Dependência Funcional** entre os dois atributos.

Repare na seguinte relação:

Pedido_Livro (ID cliente, ISBN, Data-pedido, Nome-cliente, Título, Quantidade, Preço)

Aplicando a 2ª Forma Normal, teríamos:

Pedido_Livro (ID cliente, ISBN, Data_pedido, Quantidade)

Cliente(ID cliente, Nome-cliente)

Livro (ISBN, Título, Preço)

Na tabela Pedido_Livro, apenas a quantidade depende da totalidade do identificador, o título é um atributo do livro, depende apenas do ISBN do livro e o preço (corrente) também podemos considerar um atributo do livro, se partirmos do princípio que este é fixo e não varia de pedido para pedido. O nome do cliente, por seu lado, depende apenas do ID cliente.

3ª Forma Normal

Uma relação está na 3ª Forma Normal se está na 2ª Forma Normal, e se não existirem atributos descritores a dependerem funcionalmente de outros atributos descritores (não identificadores).

Repare no seguinte exemplo:

Empregado (nº empregado, nome, nº bi, categoria, vencimento base)

Esta tabela estaria normalizada se, na empresa, fosse atribuído a cada empregado, pessoalmente, um vencimento base. Se, por outro lado, na empresa em causa, o vencimento base dos funcionários dependesse da categoria em que se encontram, então a tabela não estaria na 3ª forma normal: um atributo depende funcionalmente de outro que não é chave. Na prática, esta situação resultaria em redundância de dados. Cada vez que se introduzisse uma empregado na base de dados, ter-se-ia que introduzir a respectiva categoria e o salário, que desta depende. Ou seja, se tivéssemos 300 empregados com a categoria Consultor, teríamos que introduzir 300 vezes o mesmo salário. E se, mais tarde, existisse uma actualização deste salário, esta teria que ser efectuada nos 300 registos.

A aplicação da terceira forma normal daria origem às seguintes tabelas:

Empregado (nº empregado, nome, nº bi, cod_cat)

Categoria(cod_cat, nome categoria, vencimento base)

Em suma, a 2ª e a 3ª formas normais determinam que todos os atributos descritores dependam unicamente da chave. Não podem depender de parte da chave (2ª forma normal), nem de outros atributos que não são chave (3ª forma normal).

Implementação de uma Base de Dados em Microsoft Access

Introdução

As estruturas de dados identificadas no Modelo Entidade-Associação irão dar origem a tabelas de dados num sistema de gestão de bases de dados (SGBD), como, por exemplo, o Access. Nesta sessão procedemos à simples implementação de alguns dos esquemas relacionais, para uma melhor compreensão e interiorização dos conceitos anteriormente estudados e simultaneamente expor algumas das funcionalidades dos sistemas de gestão de bases de dados. O objectivo deste capítulo é clarificar como é que as técnicas de análise de sistemas, descritas anteriormente, poderão contribuir para a implementação de sistemas informáticos, e desta forma complementar as matérias apresentadas.

Naturalmente, todo o trabalho que foi anteriormente desenvolvido na fase de análise deverá agora ser concretizado na implementação.

O que é o Access

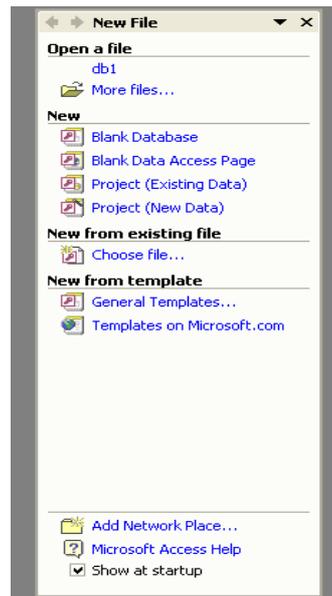
O Access é, como dissemos, um Sistema de Gestão de Bases de Dados (SGBD) relacional desenvolvido pela Microsoft. O conceito de SGBD foi anteriormente discutido neste manual. O Access foi desenvolvido em ambiente gráfico Windows, permitindo um acesso visual aos dados e trabalhá-los de forma simples e directa, independentemente da sua localização na estrutura de ficheiros. As suas capacidades de consulta e actualização são conseguidas através da utilização de duas “linguagens”: o SQL (*Structured Query Language*), uma linguagem *standard* dos SGBDs e muito potente; e o QBE (*Query By Example*), um interface gráfico que torna a consulta e manipulação de dados relativamente fácil, mesmo para não especialistas.

Como iniciar o Access

Depois de ter o *software* instalado, para começar a trabalhar com o Access, o utilizador deve proceder da seguinte forma:

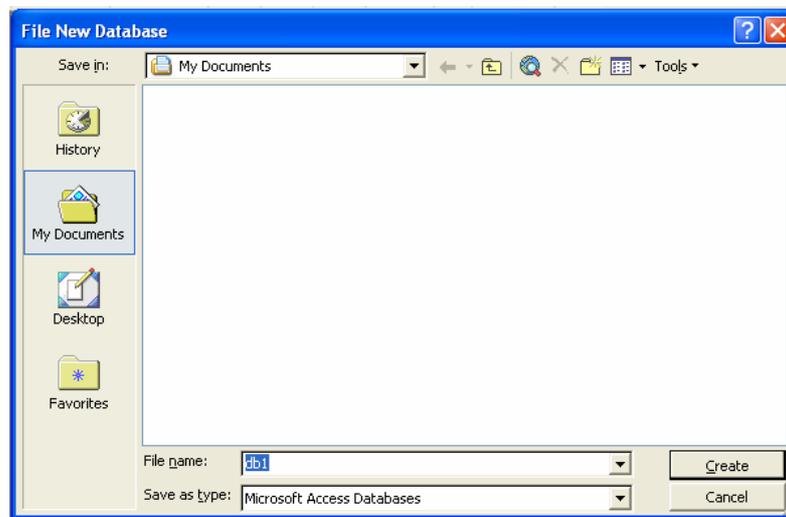
- No Windows, pressionar no botão **Iniciar** (*Start*) e deslocar o apontador do rato para a linha **Programas** (*Programs*). No sub-menu que surge no ecrã, deve seleccionar a opção **Microsoft Access**.

Seguidamente, aparece na parte direita do ecrã, a seguinte caixa de diálogo...



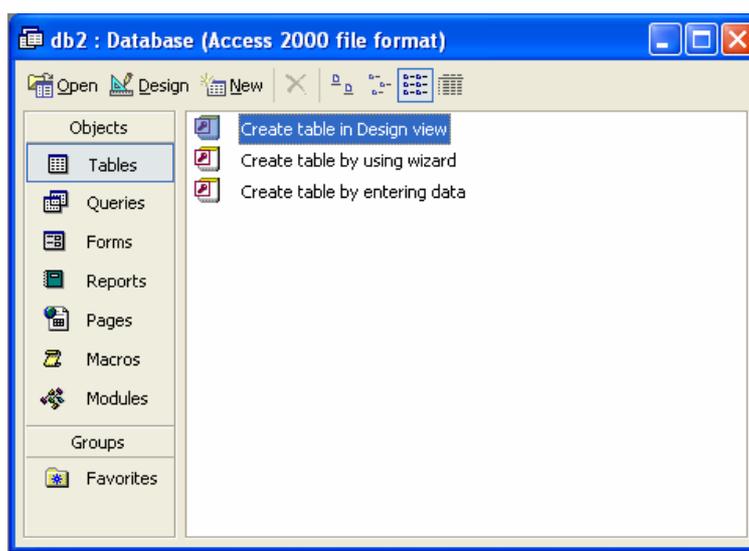
Aqui, deverá assinalar a opção **Blank Database**, para criar uma nova base de dados.

Surge, de imediato, uma outra janela, de forma a que possa atribuir um nome e definir a localização da sua nova base de dados, como podemos observar na figura seguinte:



De notar que o Access é, neste sentido, diferente das outras aplicações do Office (Word e Excel) uma vez que se deve atribuir, antes de começar a trabalhar, o nome ao ficheiro de base de dados, que irá assumir a extensão “.mdb”. Posteriormente, todos os objectos (Tabelas, Formulários, Queries, ...) serão gravados dentro deste ficheiro.

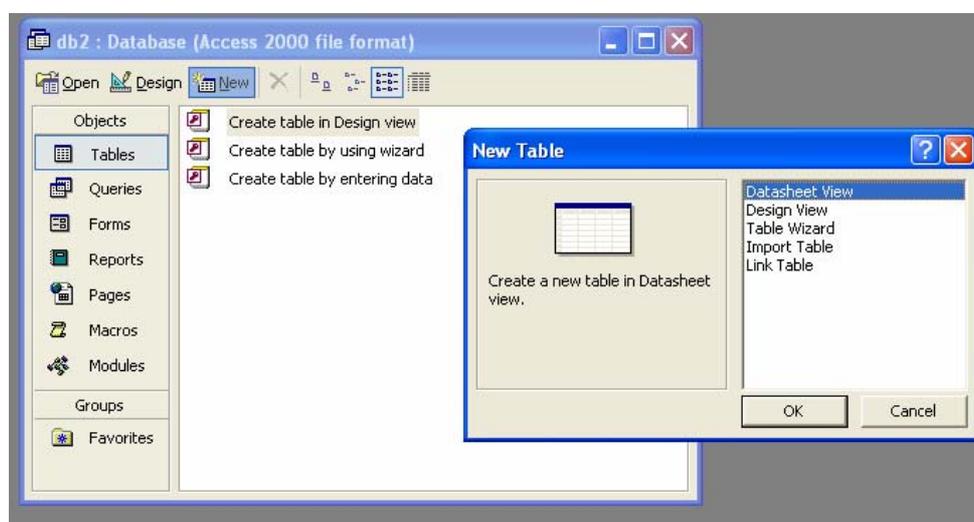
Após a atribuição do nome à nova base de dados, aparece a seguinte caixa de diálogo:



É nesta janela que tem a possibilidade de escolher se quer construir ou alterar tabelas de dados (*Tables*), elaborar estruturas de interrogação da base de dados (*Queries*), definir formulários para introdução ou visualização de dados (*Forms*), construir relatórios para o utilizador (*Reports*), criar páginas (*Pages*), automatizar sequências de comandos (*Macros*) ou escrever na linguagem de programação do Access módulos de código para as aplicações informáticas (*Modules*).

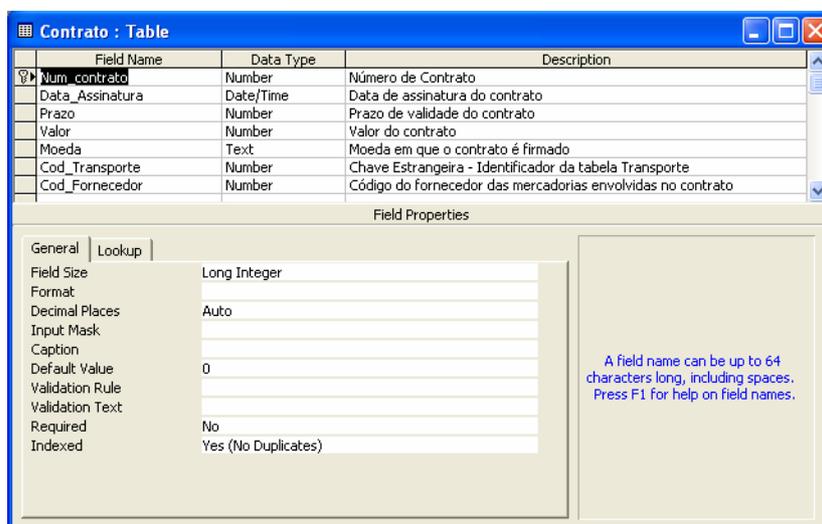
Criação de Tabelas

Uma tabela é uma estrutura de dados organizada por linhas e colunas. Como tivemos oportunidade de discutir anteriormente, uma base de dados é, normalmente, constituída por várias tabelas devidamente relacionadas entre si. Para criar uma tabela, posicione-se na opção **Tables** da janela da base de dados, prima o botão **New**, execute o comando **Insert, Table** do menu, ou simplesmente assinala a opção *Create Table in Design View*.



Após a execução deste comando aparecerá uma estrutura onde terá de definir as características das colunas de cada tabela da sua base de dados.

No exemplo seguinte, estamos a observar a estrutura da tabela **contrato** do caso Transimporta. Os atributos **num_contrato**, **data_assinatura**, **prazo**, **valor**, etc, são aqui definidos.



Field Name	Data Type	Description
Num_contrato	Number	Número de Contrato
Data_Assinatura	Date/Time	Data de assinatura do contrato
Prazo	Number	Prazo de validade do contrato
Valor	Number	Valor do contrato
Moeda	Text	Moeda em que o contrato é firmado
Cod_Transporte	Number	Chave Estrangeira - Identificador da tabela Transporte
Cod_Fornecedor	Number	Código do fornecedor das mercadorias envolvidas no contrato

Field Properties	
General	Lookup
Field Size	Long Integer
Format	
Decimal Places	Auto
Input Mask	
Caption	
Default Value	0
Validation Rule	
Validation Text	
Required	No
Indexed	Yes (No Duplicates)

Na figura anterior, referente à estrutura da tabela contrato, podem ser identificadas quatro áreas:

- **Field Name** - Local onde se deve inserir o nome de cada uma das colunas (também designadas de campos ou atributos) da tabela de dados, como o número_contrato, data_assinatura, valor, etc.
- **Data Type** - Define-se, para cada campo, qual o tipo de dados que este poderá conter. O Access apresenta as seguintes opções:

Text: Dados alfanuméricos, isto é caracteres, que podem incluir símbolos numéricos. (até 255 caracteres).

Memo: Dados do tipo alfanumérico mas com capacidade até 64.000 caracteres (normalmente utilizado para descrições detalhadas, observações, ...)

Number: Dados numéricos que poderão ser do tipo:

Byte: valores de 0 a 255.

Integer: números compreendidos entre -32.768 e 32.767.

Long Integer: números inteiros entre -2.147.483.648 a 2.147.483.647.

Single: números decimais, no intervalo de (-3,402823E38) a 3,402823E38.

Double: números decimais de -1.7976931346232E308 a 1.7976931346232E308 (10 dígitos de precisão).

Date/Time: Para campos de Data ou Hora.

Currency: Dados do tipo monetário, para os quais existe uma grande variedade de formatos.

Autonumber: Número sequencial, automaticamente inserido pelo Access.

Yes/No: Campo com duas opções: Verdadeiro ou Falso / Sim ou Não.

OLE Object: Para introdução de um objecto OLE (*Object Linking and Embedding*), ou seja, um objecto de outra aplicação do Windows (Word, imagem, etc)

Hyperlink: Refere um endereço que permita o acesso a um ficheiro ou página HTML.

Lookup Wizard: Campo que permite a criação de uma lista de valores, através de uma *Text box*, *List Box* ou de uma *Combo Box*.

Description: Nesta área, pode ser inserido um texto descrevendo o significado da coluna da tabela. O seu preenchimento é opcional.

Field Properties: Propriedades das colunas da tabela, como por exemplo, o tamanho do campo, o formato, o número de casa decimais, etc. As propriedades variam consoante o tipo de dados escolhido.

Após a definição da estrutura das tabelas, deve identificar-se que atributos serão identificadores ou chaves. No exemplo, o **Num_contrato** tem o símbolo de uma chave à esquerda, indicando que é identificador. Todas as tabelas devem ter um identificador. Pode-se definir o identificador no menu, escolhendo a opção **Edit, Primary Key**, com o cursor a seleccionar o campo (atributo) respectivo, ou mais simplesmente, assinalando o ícone , na barra de ferramentas. Se não indicar nenhuma chave primária da primeira vez que gravar a tabela, o Access questiona se pretende ou não criar a chave primária. Caso responda afirmativamente, irá ser criado automaticamente um campo designado **ID**, do tipo **Autonumber**.

Após a definição da estrutura das tabelas é necessário introduzir os dados. Para tal, podemos premir o botão **Open** ou assinalar directamente, o ícone , a partir da vista de estrutura.

A designação de cada coluna (atributo) aparece na primeira linha da estrutura e depois os dados poderão ser directamente introduzidos, linha a linha, como se de uma folha de cálculo se tratasse.

Exemplo: Caso Transimporta

Para praticar e assimilar os conceitos do modelo Entidade Associação, e verificar o bom funcionamento da Base de Dados que dele resulta, recomenda-se que se implementem essas estruturas de dados e se introduzam alguns dados de teste, por forma a obter listagens, em função de necessidades específicas de informação.

Para tal, vamos usar o caso “Transimporta”. As figuras seguintes representam as estruturas das seis tabelas e os dados que foram introduzidos, a título de exemplo:

Tabela Mercadoria

Estrutura:

Field Name	Data Type	Description
Cod_Mercadoria	Number	Identificado da Mercadoria
Designacao	Text	Nome da Mercadoria
Unidade_Medida	Text	Unidade na qual é medida a quantidade de mercadoria

Dados:

Cod_Mercadoria	Designacao	Unidade_Medida
1	Melão	kilos
2	Batatas	kilos
3	Tecido de linho	metros
4	Alcatifa	metros quadrados
5	Bananas	kilos
6	Vinho	litros
0		

Tabela Fornecedor

Estrutura:

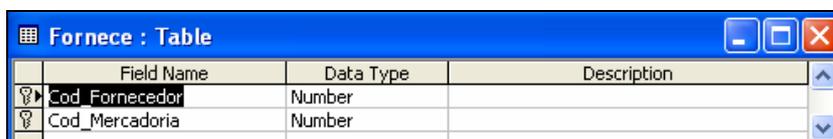
Field Name	Data Type	Description
Cod_Fornecedor	Number	
Nome	Text	
Morada	Text	
Telefone	Text	
Fax	Text	
OBS	Memo	

Dados:

Cod_Fornecedor	Nome	Morada	Telefone	Fax	OBS
1	American Sell	Main Street 123 - Dallas	12345678	2347777	
2	Softa	Plaza del Sol, 45 - Madrid	8901234	6789023	
3	TFT Export	Oxford Street 111 - London	12341234	43214321	
0					

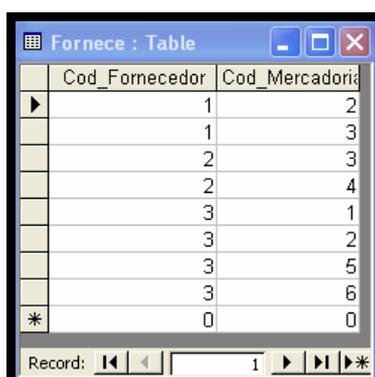
Tabela *Fornece*

Estrutura:



Field Name	Data Type	Description
Cod_Fornecedor	Number	
Cod_Mercadoria	Number	

Dados:

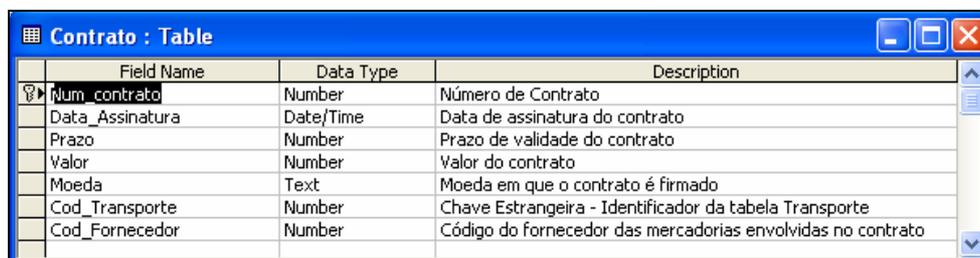


Cod_Fornecedor	Cod_Mercadoria
1	2
1	3
2	3
2	4
3	1
3	2
3	5
3	6
*	0

Esta tabela deriva da relação de Vários para Vários entre as tabelas de **Fornecedor** e de **Mercadoria**, que nos diz quais as mercadorias que cada fornecedor pode fornecer. Por exemplo, no caso do fornecedor 1 (American Sell), este pode fornecer as mercadorias 2 – Batatas e 3 – Tecido de Linho. Tem uma chave múltipla, constituída pelos dois códigos, que se define na vista de estrutura, seleccionando os dois campos antes da assinalar o ícone , como se pode verificar na figura acima representada.

Tabela *Contrato*

Estrutura:



Field Name	Data Type	Description
Num_contrato	Number	Número de Contrato
Data_Assinatura	Date/Time	Data de assinatura do contrato
Prazo	Number	Prazo de validade do contrato
Valor	Number	Valor do contrato
Moeda	Text	Moeda em que o contrato é firmado
Cod_Transporte	Number	Chave Estrangeira - Identificador da tabela Transporte
Cod_Fornecedor	Number	Código do fornecedor das mercadorias envolvidas no contrato

Dados:

	Num_contrato	Data_Assinatura	Prazo	Valor	Moeda	Cod_Transporte	Cod_Fornecedor
▶ +	2	16-12-1996	90	1120000	USD	1	2
+	105	27-04-1997	30	40000	USD	2	1
+	122	07-07-1997	90	80000	GBP	4	3
+	204	03-08-1998	360	13000	USD	3	1
+	302	12-06-1998	60	23000	GBP	5	3
*	0		0	0		0	0

Record: 1 of 5

O **Cod_Transporte** e o **Cod_fornecedor** são chaves estrangeiras, resultantes da associação com a tabela de Transporte e de Fornecedor. Note que, **no caso das chaves estrangeiras, o tipo de dados (Data Type) deve ser o mesmo do que têm nas tabelas originais**. Isto é, se, por exemplo, a chave na tabela de fornecedores é numérica (ou **AutoNumber**), então também deverá ser numérica como chave estrangeira, em qualquer das relações em que participe.

Tabela Linha de contrato
Estrutura:

Field Name	Data Type	Description
Num_Contrato	Number	
Cod_Mercadoria	Number	
Preco	Number	
Quantidade	Number	

Dados:

	Num_Contrato	Cod_Mercadoria	Preco	Quantidade
	12	3	15	8000
	12	4	20	50000
	105	2	2	40000
	122	1	1	10000
	122	5	2	10000
	122	6	2	25000
	204	2	3	13000
▶	302	5	1	23000
*	0	0	0	0

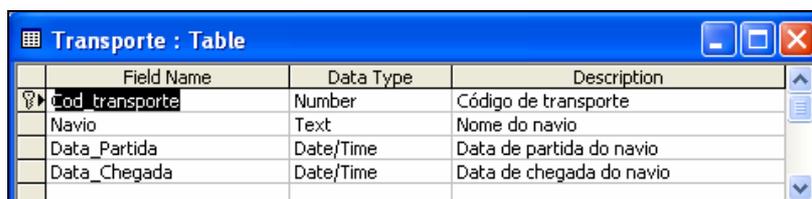
Record: 8 of 8

Esta tabela resulta de uma entidade - associativa entre **Contrato** e **Mercadoria**, e, por isso, os campos **Num_Contrato** e **Cod_Mercadoria** constituem a respectiva chave, procedendo da mesma forma que foi referida para a tabela **Fornece**. É uma associação de Vários para

Vários, que nos indica quais as mercadorias que fazem parte de cada contrato, mas com atributos próprios, o **Preço** e a **Quantidade**, uma vez que estes, para a mesma mercadoria, podem variar de contrato para contrato. A tabela Fornece é também originada por uma relação de Vários para Vários, mas não tem atributos próprios (não é uma Entidade Associativa).

Tabela *Transporte*

Estrutura:



Field Name	Data Type	Description
Cod transporte	Number	Código de transporte
Navio	Text	Nome do navio
Data_Partida	Date/Time	Data de partida do navio
Data_Chegada	Date/Time	Data de chegada do navio

Dados:



	Cod transporte	Navio	Data_Partida	Data_Chegada
+	1	O Explorador	01-01-1997	15-01-1997
+	2	O Navegador	12-05-1997	23-05-1997
+	3	Asia	15-07-1998	17-07-1998
+	4	O Mercador	13-08-1998	18-08-1998
+	5	Marbella	14-09-1998	15-09-1998
*	0			

Record: 1 of 5

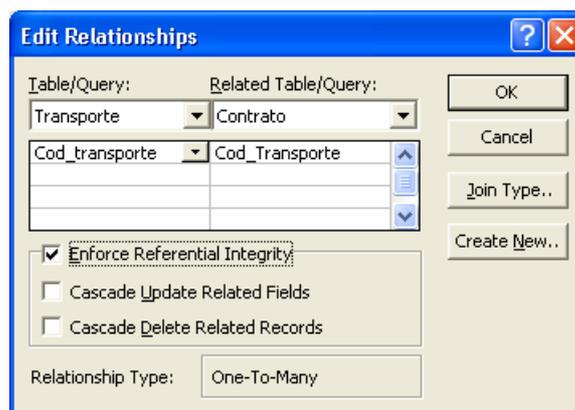
Associações entre tabelas

Após a criação das várias tabelas, o passo seguinte é estabelecer associações entre os dados que estão distribuídos pelas diferentes tabelas. Essas associações devem ser estabelecidas de acordo com o modelo entidade-associação, que foi previamente elaborado. Para tal, é necessário começar por abrir uma janela, onde se podem seleccionar as tabelas da base de dados. Essa janela poderá ser activada acedendo no **menu** à opção **Tools, Relationships** (e depois **Show Table**):

Depois de ter as tabelas seleccionadas, para estabelecer relações entre estas basta premir o rato no campo de ligação e arrastá-lo para o mesmo campo da outra tabela. Por exemplo, para relacionar as tabelas **Transporte** e **Contrato**, basta arrastar o campo comum **Cod transporte** (código de transporte), de uma tabela para cima do mesmo campo, na **outra tabela**. Surge a caixa de diálogo representada na figura seguinte, na qual é já identificado o grau da relação em causa (Um para Vários). Esta identificação tem a ver com o facto do campo, na tabela **Transporte**, ser chave, não admitindo duplicações e, assim, o grau, deste lado da associação, só pode ser **Um**. Na tabela **Contrato**, o campo

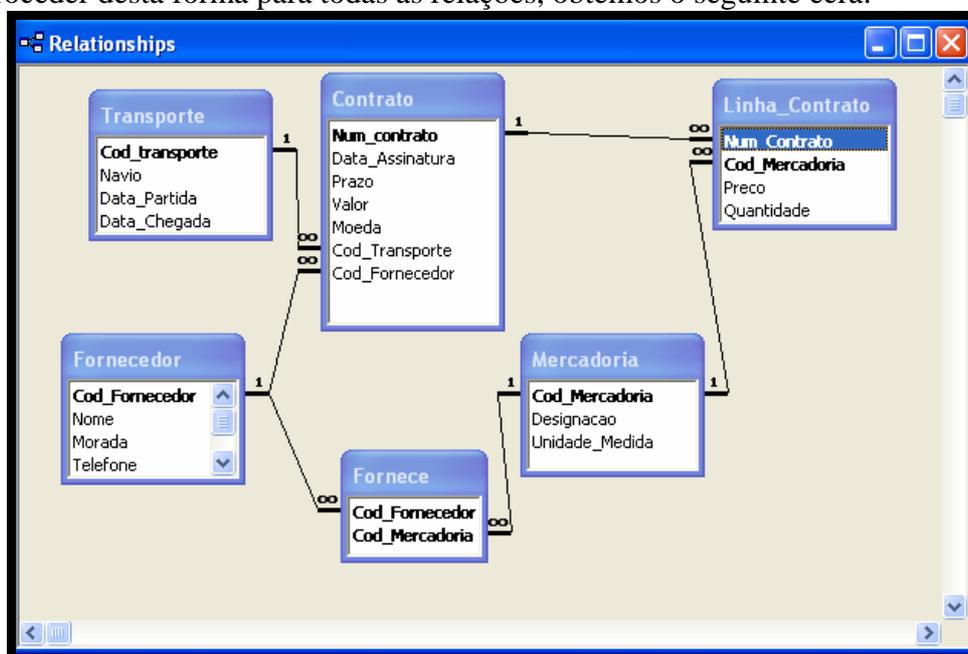
Cod_transporte é uma chave estrangeira e está definido por forma a admitir duplicações, pelo que o grau da associação é **Vários**.

Deverá também ser assinalada a caixa referente à Integridade Referencial (**Enforce Referential Integrity**). Desta forma, o Access irá controlar a integridade dos dados, não permitindo, neste caso, que sejam introduzidos, na tabela **Contrato**, códigos de transporte que não existam, isto é, não tenham sido previamente definidos na tabela **Transporte**.



Entre as tabelas **Contrato** e **Linha_Contrato** o campo comum é o *Num_contrato* (número de contrato).

Após proceder desta forma para todas as relações, obtemos o seguinte ecrã:

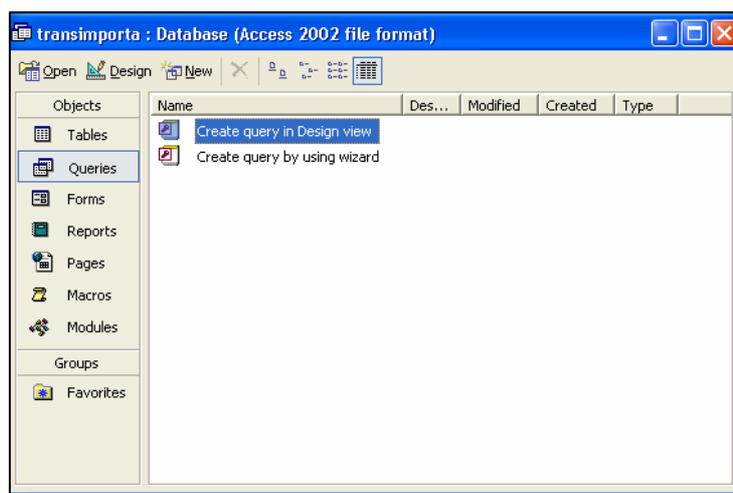


Se, por acaso, se enganar na definição de alguma relação, para editá-la basta fazer um **duplo clique** na mesma. Para a eliminar, seleccione a relação com um **clique** do rato e pressione a tecla de **Delete**.

Questionar a base de dados – a obtenção de listagens

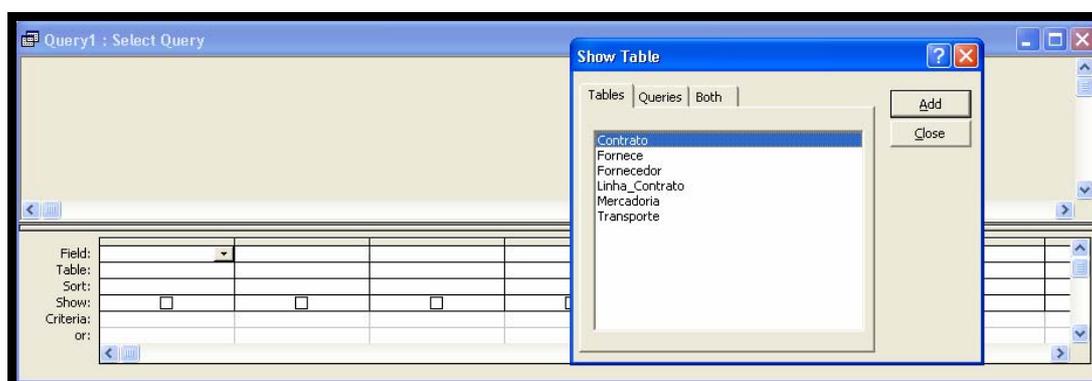
Vamos agora explicar como é que poderíamos interrogar a base de dados, obtendo listagem de dados incluídos em várias tabelas, utilizando o QBE (*Query By Example*).

Escolhendo a opção *Queries* aparece um ecrã que serve de *interface* para elaborar estas “estruturas de interrogação”:

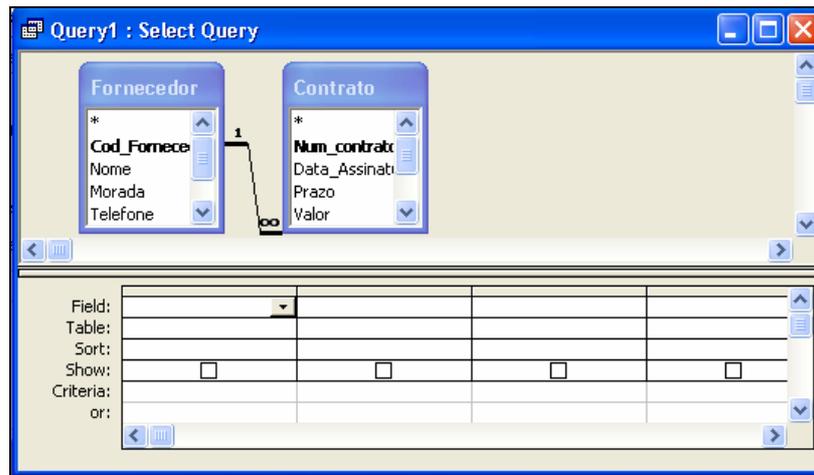


Para criar uma nova *Query*, deve-se premir com o rato o botão *New*, ou assinalar a opção *Create query in Design View*.

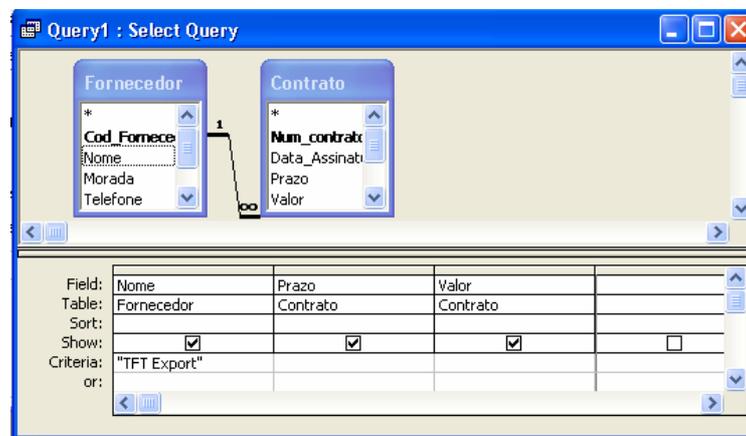
Seguidamente aparece um ecrã para a concepção de uma nova *Query*, onde é necessário definir quais as tabelas em que se encontra a informação que pretendemos listar, através da opção *Add* :



Vejamos um exemplo simples: pretende-se uma listagem do valor e prazo de validade dos contratos referentes ao fornecedor “TFT Export”. Neste caso, devemos assinalar as tabelas Fornecedor e Contrato, onde se encontram estes dados.



Na zona de tabelas, aparecem as tabelas seleccionadas e as respectivas relações, herdadas das definidas em *Tools – Relationships*. Basta agora arrastar os campos pretendidos para a zona de baixo, neste caso, o campo **Nome** da tabela **Fornecedor** e os campos **Prazo** e **Valor** da tabela **Contrato**. Se não for definido nenhum critério, quando executamos o *query* aparecem todos os fornecedores com contratos e os respectivos prazos e valores. Como, neste caso, só pretendemos os da “TFT Export”, então deveremos, nas linhas de **Crítério**, e por baixo do campo **Nome** da Tabela **Fornecedor**, escrever, simplesmente “TFT Export”:



Para executar o *Query*, basta assinalar o botão , que alterna entre a vista de estrutura e a vista de listagem, o botão , equivalente ao comando *Query-Run*, obtendo os seguintes resultados:

	Nome	Prazo	Valor
▶	TFT Export	90	80000
	TFT Export	60	23000
*			

Record: 1 of 2

As *Queries* mais frequentes são as de *Select*, para visualização de dados, que vão ser objecto do nosso estudo. No entanto, esta ferramenta permite também efectuar outras operações:

- Actualizar dados, recorrendo a uma rotina - *Update Queries*;
- Acrescentar dados - *Append Queries*;
- Para criar novas tabelas - *Make-Table Queries*;
- Organizar e cruzar os dados – *Crosstab Queries*;
- Apagar dados – *Delete Queries* .

Define-se em *Field* o nome do campo que queremos listar e em *Table* a designação da tabela à qual esse campo pertence.

A opção *Sort* serve para ordenar por ordem crescente (*Ascending*) ou decrescente (*Descending*).

O indicador *Show* é utilizado para definir se as colunas irão ou não aparecer na listagem.

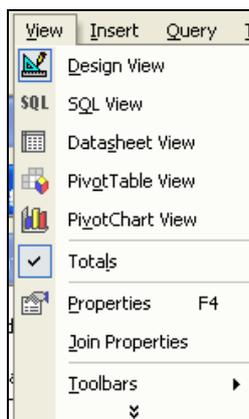
Na zona de critério (*Criteria*), vamos colocar restrições à listagem. Nesta área, **os campos de caracteres (alfanuméricos) devem ser apresentados entre aspas** (ex: “TFT”), **os campos de data entre cardinais** (#12-08-2005#) e **os valores numéricos são escritos naturalmente**. No entanto, se estes caracteres não forem introduzidos, o *Access* trata de os acrescentar, em função do tipo de campo. Podem ser usados caracteres especiais, as chamadas “wildcards” -o “*”, que substitui qualquer cadeia de caracteres e o “?”, que simula qualquer carácter, mas numa determinada posição. Por exemplo, o critério “A*” (que o *Access* traduz para Like “A*”) lista todos os nomes que comecem por A.

Também podemos colocar os operadores lógicos *and*, *or* e *not*, para a conjugação de vários critérios.

A zona de critério tem várias linhas, sendo que os critérios que lá colocamos funcionam da seguinte forma:

- Critérios na mesma linha - funcionam com o “E” (*and*) lógico.
- Critérios em linha distintas - funcionam com o “Ou” (*or*) lógico.

Outra potencialidade com bastante interesse, passa por se poderem obter directamente, a partir das *queries*, somatórios e outras funções estatísticas, tais como médias, máximos e mínimos. Para isso, basta activar, a partir da vista de estrutura do *Query*, activar a opção *Totals*, a partir do Menu *View*.



Aparece uma nova linha na definição do *Query*, em que a opção **Group By**, aparece por defeito:



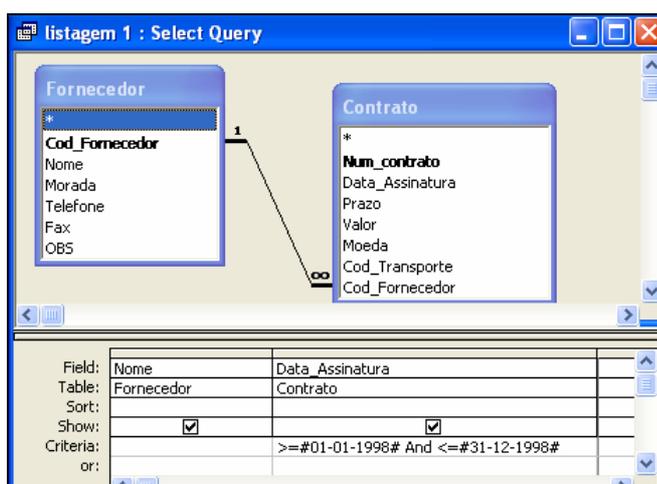
Esta opção agrupa os registos relativamente aos quais se irão fazer, entre outras, as seguintes operações estatísticas:

- **Sum** – Soma
- **Avg** – Média aritmética
- **Min** – Mínimo
- **Max** – Máximo
- **Count** – Número de registos
- **StDev** – Desvio Padrão
- **Var** – Variância.

A utilização dos *queries* desta ferramenta é relativamente simples. Se os dados estiverem devidamente estruturados, a elaboração dos *queries* é rápida. Se quisermos obter a mesma listagem mais do que uma vez, basta gravarmos o *query*, que este fica disponível para outras sessões.

Vejam agora alguns exemplos, usando a base de dados do caso **Transimporta**:

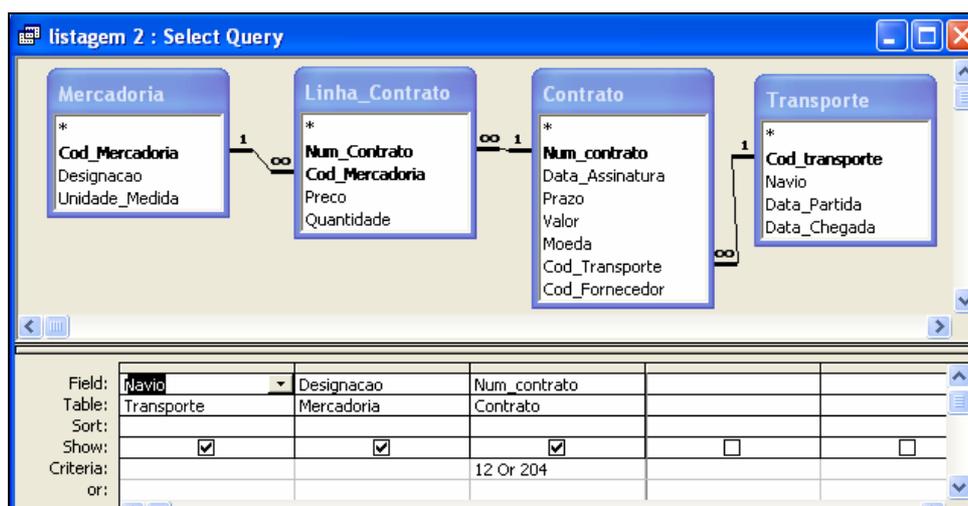
1. Listar o nome dos fornecedores com os quais estabelecemos contratos em 1998.



Das tabelas **Contrato** e **Fornecedor**, retiramos os campos **Data_Assinatura** e **Nome**, respectivamente. Sendo o campo **Data_Assinatura** um campo tipo **Data**, então o critério adequado para obtermos os registos do ano de 1998, poderá ser o de estar compreendida entre o primeiro e o último dia do ano.

Nome	Data_Assinatura
American Sell	03-08-1998
TFT Export	12-06-1998

- Listar o nome do navio e a designação das mercadorias envolvidas nos contratos números 12 e 204.



Outra alternativa seria colocar o nº 12 na primeira linha do critério, e o nº 204 por baixo, na segunda linha. Note que, embora não conste na listagem do Query nenhum campo da tabela Linha de Contrato, esta tem que estar presente na estrutura do Query, para fazer a ligação entre as tabelas Mercadoria e Contrato.

	Navio	Designacao	Num_contrato
▶	O Explorador	Tecido de linho	12
	O Explorador	Alcatifa	12
	Asia	Batatas	204
*			

Record: 1 of 3

3. Listar a data de partida do navio que transporta as mercadorias do contrato 105.

Field:	Data_Partida	Num_contrato	
Table:	Transporte	Contrato	
Sort:			
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Criteria:		105	
or:			

	Data_Partida	Num_contrato
▶	12-05-1997	105
*		

Record: 1

4. Listar o nome e morada dos fornecedores que firmaram com a empresa contratos em dólares durante o ano de 1997.

Field:	Nome	Data_Assinatura	Moeda	Year([contrato],[data_assinatura])
Table:	Fornecedor	Contrato	Contrato	
Sort:				
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Criteria:			"USD"	1997
or:				

Nesta solução, são introduzidos os critérios da moeda e da data na mesma linha, conjugando assim um “E” lógico. Para obter o ano de 1997, é utilizada a função *Year*

(data), que retira o ano de uma determinada data, em alternativa à solução proposta no exemplo 1.

	Nome	Data_Assinatura	Moeda
▶	American Sell	27-04-1997	USD
*			

Record: 1 of 1

- Listar o nome de todos os fornecedores que firmaram com a empresa contratos em dólares. Incluir também na listagem todos aqueles que firmaram contratos em 1997 independentemente da moeda.

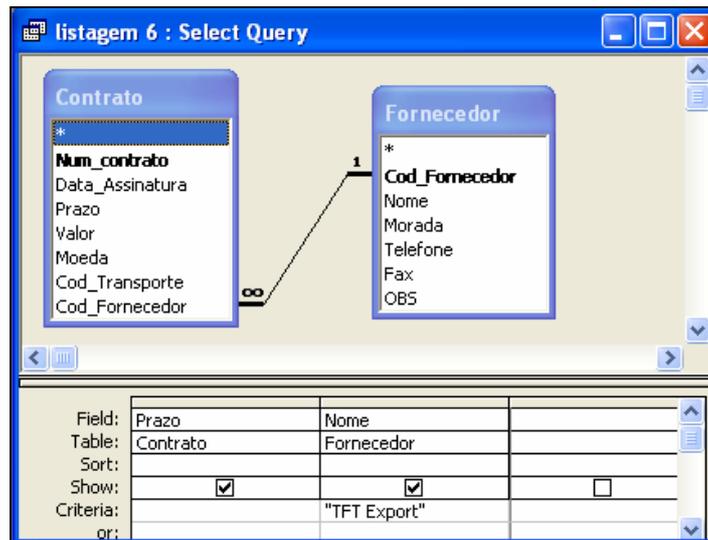
Field:	Nome	Data_Assinatura	Moeda	Year([contrato].[data_assinatura])
Table:	Fornecedor	Contrato	Contrato	
Sort:				
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Criteria:			"USD"	
or:				1997

Neste caso, os critérios são colocados em linhas diferentes, para obter o “Ou” lógico: os contratos em dólares, conjuntamente com os do ano de 1997.

	Nome	Data_Assinatura	Moeda
▶	Sofita	16-12-1996	USD
	American Sell	27-04-1997	USD
	TFT Export	07-07-1997	GBP
	American Sell	03-08-1998	USD
*			

Record: 1 of 4

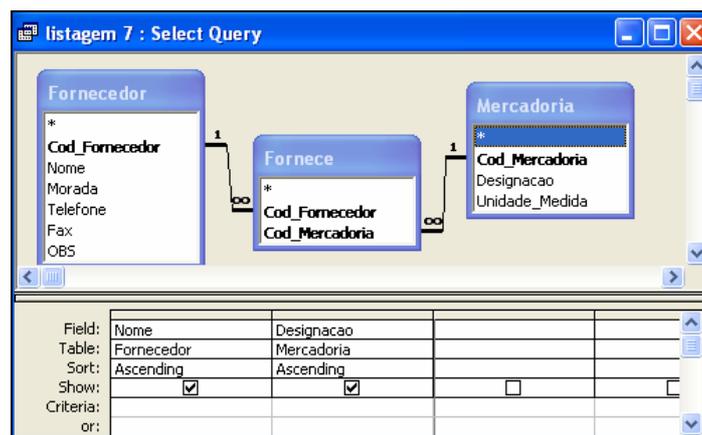
- Listar o prazo de validade dos contratos feitos com o fornecedor “TFT Export”.



Prazo	Nome
90	TFT Export
60	TFT Export

Record: 2 of 2

Listagem das mercadorias que podem ser fornecidas pelos diversos fornecedores, ordenadas alfabeticamente por fornecedor.

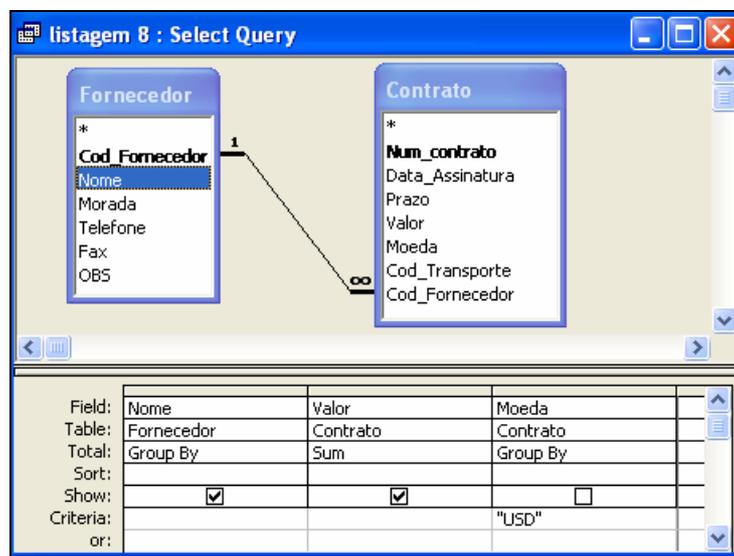


Note que, neste *Query*, é definida a ordem primária de listagem no campo mais à esquerda, e a secundária na coluna seguinte. Na prática, a listagem vai aparecer ordenada por ordem alfabética de Fornecedores e, dentro de cada um destes, por ordem alfabética de Mercadorias.



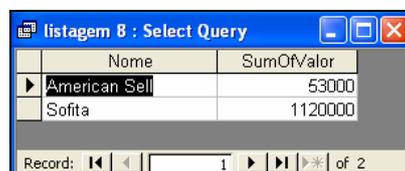
Nome	Designacao
American Sell	Batatas
American Sell	Tecido de linho
Sofita	Alcatifa
Sofita	Tecido de linho
TFT Export	Bananas
TFT Export	Batatas
TFT Export	Melão
TFT Export	Vinho
*	

7. Listar o valor total dos contratos em USD, por fornecedor.



Field:	Nome	Valor	Moeda
Table:	Fornecedor	Contrato	Contrato
Total:	Group By	Sum	Group By
Sort:			
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Criteria:			"USD"
or:			

Neste *query*, foi activada a opção **Totals**, a partir do Menu **View**, que faz aparecer uma linha onde se definem o tipo de operações que se pretendem fazer. Neste caso, é seleccionada a opção **Sum** (somatório), para ser aplicada no campo **Valor**. O campo **Moeda**, que serve apenas para definir o critério, foi, como pode verificar, retirado da listagem (opção **Show**)



Nome	SumOfValor
American Sell	53000
Sofita	1120000

Exemplo – Caso Gestão de contas de um Banco Comercial.

Vamos começar por criar a base de dados, à qual vamos chamar Banco Comercial, e aí definir a seguinte estrutura de tabelas:

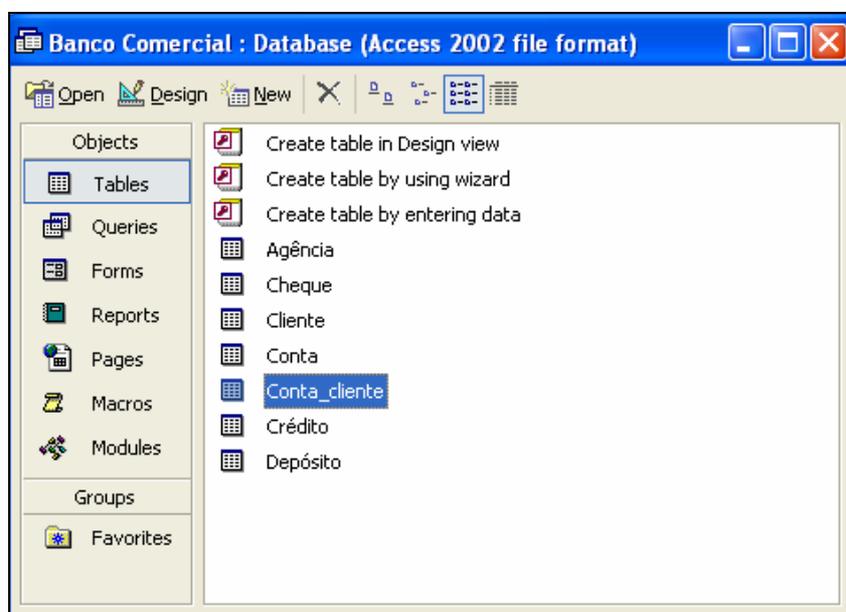


Tabela *Conta*

Estrutura:

Field Name	Data Type	Description
num_conta	AutoNumber	Número da Conta - identificador
data_abertura	Date/Time	Data de abertura da contra
tipo	Text	Define o tipo de conta (À ordem ou a Prazo)
saldo	Currency	Saldo corrente da conta
cod_agencia	Number	Chave estrangeira - identificador da Agência

Dados:

	num_conta	data_abertura	tipo	saldo	cod_agencia
+	1	04-03-2001	Ordem	1.245,00 €	1
+	2	24-09-2002	Ordem	272,00 €	2
+	3	17-06-2004	Ordem	3.452,00 €	3
+	4	26-05-2002	Prazo	65.389,00 €	1
+	5	14-01-2003	Ordem	365,00 €	4
+	6	14-12-2004	Ordem	0,00 €	5
+	7	03-11-2002	Ordem	1.297,00 €	1
+	8	28-06-2003	Ordem	9.452,00 €	1
+	9	03-04-2003	Ordem	98.237,00 €	2
+	10	16-08-2004	Prazo	3.827.324,00 €	2
*	(AutoNumber)			0,00 €	0

A descrição dos campos é, como já foi dito, opcional, e deverá ser preenchida para evitar dúvidas ou ambiguidades sobre o conteúdo das tabelas. Para as tabelas seguintes, parece-nos que o nome do campo é suficientemente esclarecedor, pelo que vamos omitir a descrição.

Tabela *Cliente*

Estrutura:

Field Name	Data Type	Description
num_cliente	AutoNumber	
nome	Text	
morada	Text	
telefone	Number	
BI	Number	
N_Contribuente	Number	

Dados:

num_cliente	nome	morada	telefone	BI	N_Contribuente
1	João Sousa	R. Das Lajes, 14, LISBOA	213457819	125441326	197653419
2	Andreia Silva	Av. Dos Aliados, 245, R/C Dto, PORTO	228763092	213542817	196723542
3	Joana Reis	R. João das Regras, 12, 1ºEsq, COIMBRA	234675182	156420198	154274982
4	António Ferreira	Alameda das Tílias, 34, 10º Dto, PORTIMÃO	965431890	128765198	187652239
5	Susana Lopes	R. Do Carmo, 43, 1º, SETÚBAL	918276428	109897119	129807819
6	Vanda Stuart	Av. Da Liberdade, 123, 8º, ALCOCHETE	938487829	298127019	508276376
7	Joaquim Tavares	Av. Marconi, 98 - 8º Esq, LISBOA	287674738	109927766	982927376
8	Gabriel Alves	R. Das Trinas, 2 - 9º Frt, CASTELO BRANCO	238872979	138862368	573287423
*	(AutoNumber)		0	0	0

Record: 1 of 8

Tabela *Conta_Cliente*

Estrutura:

Field Name	Data Type	Description
num_conta	Number	
num_cliente	Number	

Dados:

num_conta	num_cliente
1	1
1	3
1	4
2	3
3	2
4	1
5	4
6	2
7	7
8	5
9	6
*	0

Record: 1

Tabela Agência

Estrutura:

Field Name	Data Type	Description
cod_agencia	AutoNumber	
Localidade	Text	

Dados:

cod_agencia	Localidade
1	Lisboa
2	Porto
3	Coimbra
4	Faro
5	Portimão
* (AutoNumber)	

Record: 1 of 5

Tabela Cheque

Estrutura:

Field Name	Data Type	Description
num_cheque	Number	
valor	Currency	
data_emissao	Date/Time	
local_emissao	Text	
num_conta	Number	

Dados:

num_cheque	valor	data_emissao	local_emissao	num_conta
201982349	234.122,00 €	19-04-2004	LISBOA	1
233243222	4.572,00 €	09-10-2003	COIMBRA	3
234987237	234,00 €	09-04-2003	FARO	9
236416399	82.397,00 €	27-01-2004	PORTO	6
283092302	2.308,00 €	19-10-2004	ÉVORA	8
327473466	32,00 €	02-09-2003	PORTIMÃO	4
0	0,00 €			0

Record: 1 of 6

Tabela *Crédito*

Estrutura:

Field Name	Data Type	Description
num_authorized	AutoNumber	
valor	Currency	
data_authorized	Date/Time	
num_conta	Number	

Dados:

num_authorized	valor	data_authorized	num_conta
1	20.000,00 €	18-11-2004	1
2	100.000,00 €	23-11-2004	6
3	50.000,00 €	01-01-2005	1
4	40.000,00 €	23-01-2005	2
5	350.000,00 €	22-02-2005	4
6	250.000,00 €	30-05-2005	6
(AutoNumber)	0,00 €		0

Record: 7 of 7

Tabela *Depósito*

Estrutura:

Field Name	Data Type	Description
num_deposito	AutoNumber	
data_deposito	Date/Time	
montante	Currency	
nome	Text	
num_conta	Number	

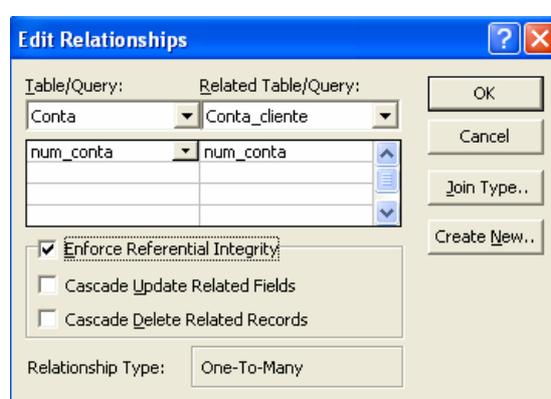
Dados:

num_deposito	data_deposito	montante	nome	num_conta
1	02-08-2003	2.398,00 €	Silvério Borges	3
2	04-07-2003	347.273,00 €	Antónia Gama	7
3	12-08-2003	28.175,00 €	Marta Dias	1
4	23-02-2003	98.983,00 €	José Castelo	6
(AutoNumber)		0,00 €		0

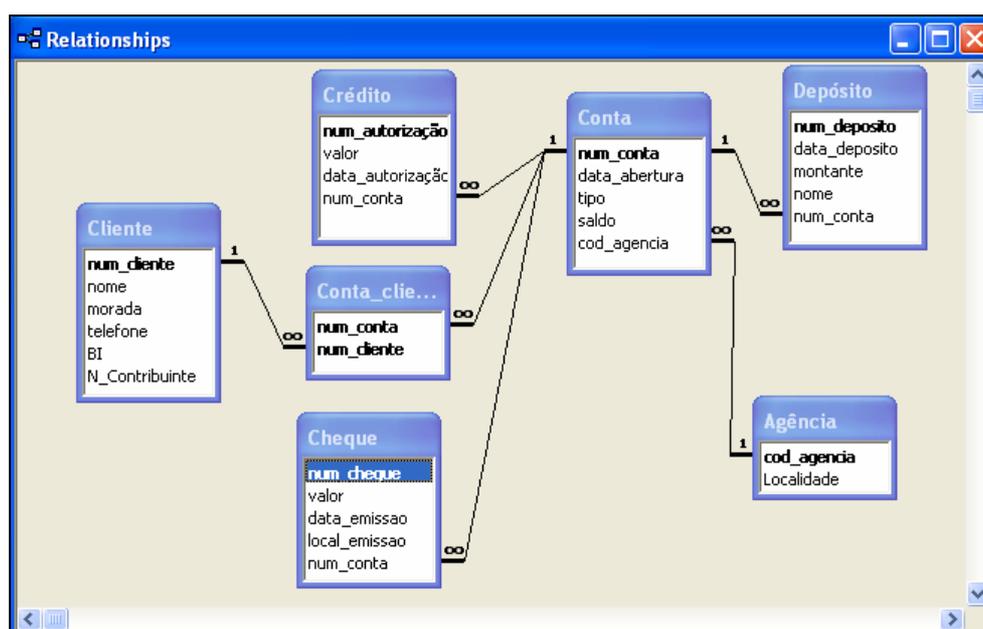
Record: 1 of 4

Uma vez criadas as tabelas, vamos agora proceder ao seu relacionamento, replicando no Access o Modelo Entidade-Associação que elaborámos previamente, através do comando *Tools- Relationships*.

Embora seja possível proceder ao relacionamento entre as tabelas após o seu preenchimento, tal não é, no entanto aconselhável. Se relacionarmos as tabelas antes da introdução de dados, e assinalarmos a opção **Enforce Referential Integrity**, o Access irá garantir a integridade dos dados, nomeadamente no que concerne às chaves estrangeiras. No caso da relação apresentada na figura seguinte, a garantia da integridade referencial entre as tabelas **Conta** e **Conta – Cliente** faz com que não seja possível introduzir no campo `num_conta`, que é chave estrangeira na tabela `conta_cliente`, números de contas que não tenham sido previamente criados na tabela **Conta**. Ou seja, na prática, evita que um cliente seja titular de contas que não existem.



Após o relacionamento de todas as tabelas do modelo, da forma que aprendemos anteriormente, obtemos um quadro semelhante ao seguinte:



Vamos, agora, interrogar a base de dados e procurar obter as seguintes listagens:

1 – Listagem das Contas das quais é titular o cliente nº1, João Sousa, e as respectivas datas de abertura

Field:	num_cliente	nome	num_conta	data_abertura
Table:	Cliente	Cliente	Conta	Conta
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:	1			

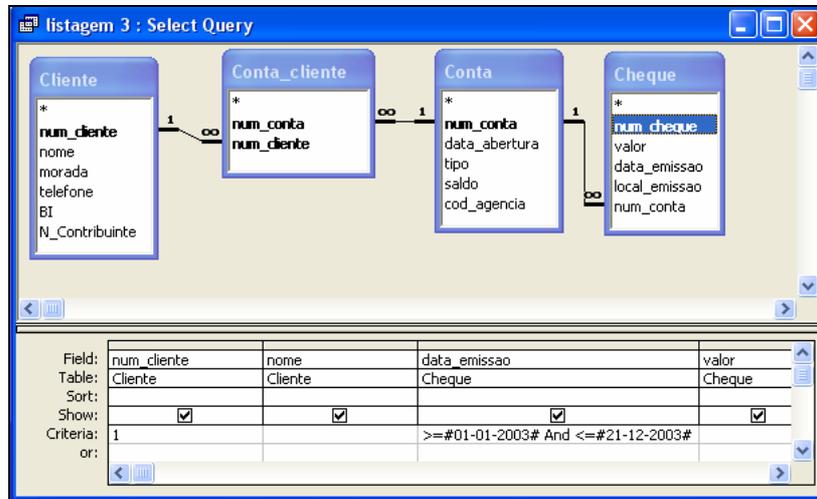
num_cliente	nome	num_conta	data_abertura
1	João Sousa	1	04-03-2001
1	João Sousa	4	26-05-2002

2 – Listagem dos depósitos efectuados na agência de Lisboa.

Field:	Localidade	num_deposito	montante
Table:	Agência	Depósito	Depósito
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:	"Lisboa"		

Localidade	num_deposito	montante
Lisboa	2	347.273,00 €
Lisboa	3	28.175,00 €

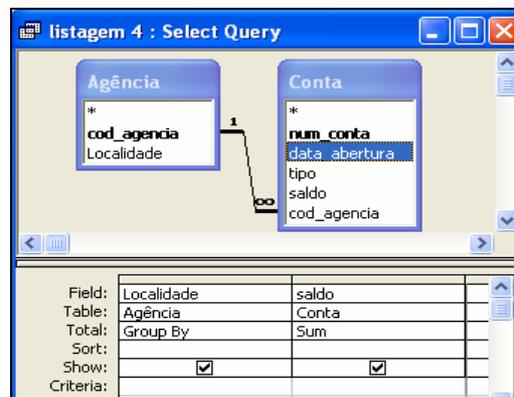
3 - Listagem dos cheque emitidos pelo cliente nº 1 , João Sousa, durante o ano de 2003.



num_cliente	nome	data_emissao	valor
1	João Sousa	02-09-2003	32,00 €

Record: 1 of 1

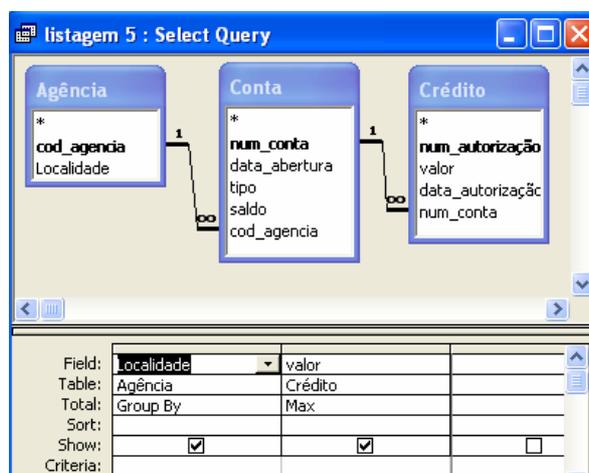
4 – Listagem dos saldos totais das contas, por Agência.



Localidade	SumOfsaldo
Coimbra	3.452,00 €
Faro	365,00 €
Lisboa	77.383,00 €
Portimão	0,00 €
Porto	3.925.833,00 €

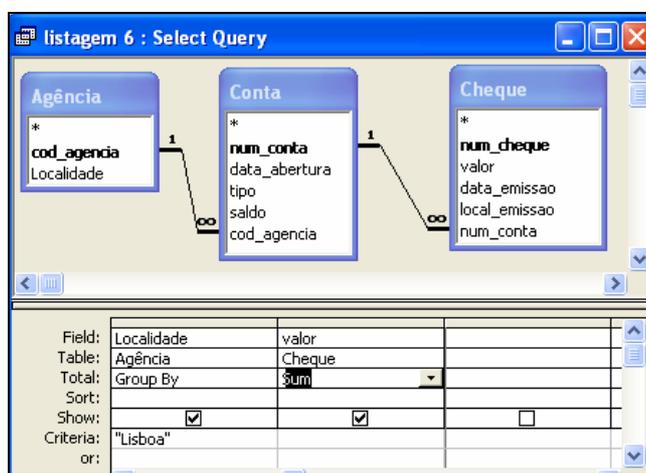
Record: 1 of 5

5 - Listagem do valor mais elevado de crédito concedido, por Agência.



Localidade	MaxOfvalor
Lisboa	350.000,00 €
Portimão	250.000,00 €
Porto	40.000,00 €

6 – Total do valor dos cheques emitidos pela Agência de Lisboa



tipo	Localidade	SumOfvalor
Ordem	Lisboa	236.430,00 €

Bibliografia

- CHEN, P. (1976). “The Entity-Relationship Model - Towards a Unified View of Data”, *ACM Transactions on Database Systems*, 1(1), 9-36.
- CODD, E. F., (1985), “Is Your DBMS Really Relational”, *Computer World*, Outubro.
- CODD, E.F. (1970). “A Relational Model of Data for Large Shared Data Banks”, *Communications of the ACM*, 13(6), 377-387.
- DATE, C.J. *An Introduction to Database Systems*, 5ª edição, Addison-Wesley, Reading, 1990.
- DOWNS, E., CLARE, Peter e COE, Ian, (1992). *Structured Systems Analysis and Design Method: Application and Context*, 2ª edição, Prentice Hall, Hemel Hempstead.
- LUCAS, Ana. *Curso de Técnicas de Análise e Desenho de Sistemas*, Info-Excelência, Lisboa, 1993.
- RIVAS, Filipe Gómez-Pallete (1984). *Estructuras Organizativas e Información en la Empresa*, Asociación para el Progreso de la Dirección, Madrid.
- RUMBAUGH, J.; Jacobson, I e Booch, G. (1999). *The Unified Modeling Language Reference Manual*, Addison-Wesley, Reading (Mass.)
- SERRANO, António, CALDEIRA, Mário e GUERREIRO, António (2004). *Gestão de Sistemas e Tecnologias de Informação*, FCA.
- TUDOR, D. J. e TUDOR, I. J. (1995). *Systems Analysis and Design Methods: A Comparison of Structured Methods*, NCC Blackwell.
- YOURDON, Edward (1989). *Modern Structured Analysis*, 3ª edição, Prentice Hall, Englewood Cliffs.