Curso Técnico de Informática

MODELAGEM DE DADOS 1

Primeira Etapa

CAPITULO I

VISÃO GERAL DO GERENCIAMENTO DE BANCO DE DADOS

O computador é uma máquina como muitas outras, mas que possui algumas características que a distinguem das demais, como a velocidade do processamento e a capacidade de armazenamento. É um equipamento que, com o passar dos anos, se tornou de uso pessoal e profissional, podendo ser utilizado no lazer e no dia-a-dia, nas mais variadas tarefas. Olhando por um prisma mais direto, o computador possui três tarefas básicas: ***entrada***, ***processamento*** e ***saída de dados***.

A **entrada de dados** é responsável pela importação das informações para o computador e está relacionada diretamente à interação com o usuário, pois na maioria das vezes depende da informação digitada por ele através da forma mais simples de interação: o teclado. Já a **saída de dados** representa os dados informados na entrada, mas que, de alguma forma, foram transformados atendendo as expectativas do usuário, sendo apresentados por meio de algum dispositivo de saída, como impressora ou monitor.

O processo de transformação dos dados de entrada em dados de saída chama-se ***processamento***. Com o passar dos anos, uma nova tarefa tão importante quanto às outras foi adicionada a esse ambiente: o ***armazenamento dos dados***.

Tão importante quanto informar os dados, processá-los e apresentá-los, a sua organização e armazenamento no computador passou a ser uma tarefa essencial, principalmente pelo grande número de usuários que podem interagir com o equipamento ao mesmo tempo. Com o passar dos anos, o número de dados processados cresceu de forma assustadora e, de forma inversamente proporcional, o computador tem diminuído em dimensões físicas e aumentado em capacidade de armazenamento e processamento de dados.

Gerenciar e armazenar dados tornou-se uma tarefa complexa, que envolve muitas ferramentas e tecnologias. Com a grande evolução da computação ou da informática em geral, os equipamentos têm evoluído muito, o que tem tornado o seu custo mais acessível. Com isso, o uso do computador tem atingido os mais variados perfis de usuários.

A principal finalidade do computador é armazenar dados, que possam ser acessados a qualquer momento e em qualquer lugar do mundo. Quanto maior o número de computadores, maior o número de dados a serem gerenciados e processados. Como a tendência é que esse número aumente a cada ano, a complexidade dos dados armazenados deverá ser um grande problema na área de informática.

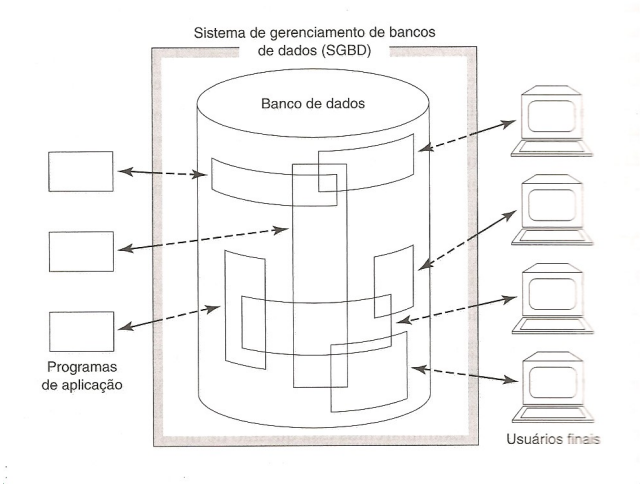
Em paralelo a esse grande número de novos usuários de computadores, bem como à expansão na facilidade de acesso aos computadores e às novas tecnologias em hardware, há também uma grande evolução na área de softwares, principalmente os que se destinam ao gerenciamento dos dados armazenados no computador. Esses softwares

ou aplicativos se destinam a uma área específica da informática, que se chama ***banco de dados***.

A área de banco de dados inclui ferramentas e principalmente pessoas especializadas, capazes de compreender as diferentes tecnologias do mercado, escolher a que melhor atende as suas expectativas e propor e modelar soluções aos diversos problemas que farão parte de suas atividades profissionais.

1.1- SISTEMA DE BANCO DE DADOS

Um *Sistema de Banco de Dados* é basicamente um sistema computadorizado de manutenção de registros; em outras palavras, é um sistema computadorizado cuja finalidade, geral é armazenar informações e permitir que os usuários busquem e atualizem essas informações quando as solicitar. As informações em questão podem ser qualquer coisa que tenha algum significado ao indivíduo ou à organização a que o sistema deve servir – ou seja, qualquer coisa que seja necessária para auxiliar no processo das atividades desse indivíduo ou dessa organização.



O *Sistema de Banco de Dados* envolve quatro componentes principais: dados, hardware, software e usuários.

Dados

Um banco de dados nada mais é do que um repositório de dados, que pode ser classificado como integrado ou compartilhado. O banco de dados integrado visa ao controle de total eliminação da redundância de dados geralmnete encontrada em sistemas de arquivos comuns (gerenciadores de arquivos); o compartilhado visa ao acesso de vários usuários a uma mesma parte dos dados, com finalidades distintas.

Hardware

Os componentes de hardware do sistema consistem em:

- *Volumes de aramazenamento secundário* – normalmente, discos magnéticos -, que são usados para manter os dados armazenados, juntamente com os dispositivos de E/S associados (unidades de discos, etc.), controladores de dispositivos, canais de E/s e assim por diante.

- *Processador(es) de hardware e memória principal associada*, que são usados para dar suporte à execução do software do *Sistema de Banco de Dados.*

Software

Entre o banco de dados físico – ou seja, os dados fisicamente armazenados – e os usuários do sistema existe uma camada de software, conhecida como *gerenciador de banco de dados* ou *servidor de bando de dados* ou, mais frequentemente, como ***sistema gerenciador de banco de dados*** (SGBD). Todas as requisições de acesso ao banco de dados são tratadas pelo SGBD; os recursos para acrescentar e remover arquivos (ou tabelas), buscar dados e atualizar dados em tais arquivos ou tabelas, e assim por diante, são facilidades fornecidas pelo SGBD

O SGBD é, de longe, o componente de software mais importante de todo o sistema mas não é o único. Outros componentes incluem utilitários, ferramentas de desenvolvimento de aplicações, recursos para auxiliar no projeto, geradores de relatórios e o gerenciador de transações. O termo SGBD também é usado para se referir, genericamente, a algum produto específico de algum fornecedor em particular.

Usuários

Consideramos três classes gerais de usuários:

- Os programadores de aplicações, responsáveis pela escrita de programas de aplicações de banco de dados em alguma linguagem de programação. Esses programas acessam o banco de dados emitindo a requisição apropriada ao SGBD.

- Os usuários finais acessam ao banco de dados interativamente.

- Os profissionais da área de banco de dados, à saber:

  Administrador de Dados que é a pessoa responsável pelos dados da empresa. Os dados constituem um dos bens mais valiosos da empresa, é imperativo que deva existir alguma pessoa que entenda esses dados e as necessidades da empresa com relação a esses dados, em um nível mais elevado da administração. É seus trabalho decidir que dados devem ser armazenados no banco de dados, além de estabelecer normas para manter e tratar esses dados, uma vez que venham a ser armazenados.

  Administrador de Banco de Dados (DBA) é o técnico responsável pela implementação das decisões do Administrador de Dados. É um profissional de tecnologia da informação. Seu trabalho é criar o banco de dados propriamente dito e implementar os controles técnicos necessários para por em prática as diversas decisões sobre normas tomadas pelo Administrador de Dados. Também é responsável por assegurar que o sistema operará com desempenho adequado e por oferecer vários outros serviços técnicos.

1.2- O QUE É UM BANCO DE DADOS?

No vasto mundo da tecnologia da informação, há poucas coisas tão fascinantes quanto à forma de armazenamento, controle e recuperação dos dados referentes a sistemas de informação.

A análise de um sistema de informação começa pela definição do objetivo a ser atingindo e das informações que se pretende armazenar a respeito do assunto que define o sistema.

A primeira providência a tomar é o contato direto com os dados pertinentes às informações de que se deseja tratar.

A utilização prática dessas informações, mesmo sabendo que parecem ser complexas, é de domínio relativamente fácil a partir do momento em que se decide sobre as configurações básicas para guardar, controlar e recuperar em tempo hábil os dados específicos que apóiam as informações desejadas.

Quando se abordam os conceitos de dados e informações, deve-se ter presente que dado é um conjunto de símbolos arrumados de forma a representar uma informação, e que a informação propriamente dita compreende os dados que foram inseridos e processados por um computador, relativos a um assunto determinado, vários assuntos, ramos de atividades, produtos ou serviços, que se tornam disponíveis a quem os requisitou.

A informação, portanto, é um dado ao qual se agrega valor e que afeta as empresas de forma efetiva, o que a transforma num bem tão precioso quanto o seu patrimônio. Para que o crescimento de uma empresa no mercado seja alcançado e para que ela consiga tratar as novas tecnologias apresentadas, terá que se preocupar muito em como organizar e recuperar suas informações.

Nos mais elementares campos do conhecimento humano, existem repositórios gigantescos de dados que permitem a recuperação de informações cruciais para o atendimento das necessidades das pessoas e das organizações. Documentos de identidade, registros de nascimento e de casamento, títulos eleitorais, carteiras de motoristas, entre outros, são exemplos conhecidos desses gigantescos repositórios.

Todas as vezes que alguém realiza uma transação num caixa eletrônico do seu banco, deseja obter informações que sejam geradas de forma rápida e restrita, sem redundância e, principalmente, íntegras.

A dificuldade em se manipular e controlar os imensos repositórios de dados existentes leva a estudos cada vez mais apurados das tecnologias aplicadas aos métodos de armazenamento de informações e seus impactos no uso de computadores.

Quando se estudam as formas de armazenamento, controle e recuperação de dados, é necessário fazer referência aos computadores envolvidos e aos relacionamentos existentes entre eles para a determinação dos tipos e da representação desses tipos por meio de modelos ou esquemas fáceis de serem tratados.

Uma coleção de informações relacionadas entre si, referentes a um mesmo assunto, e organizadas com o propósito de servir de base para que o usuário as recupere, tire conclusões e tome decisões, é o que define um ***banco de dados***.

O banco de dados, por si só, pode ser considerado como o equivalente eletrônico de um armário de arquivamento; ou seja, ele é um repositório ou recipiente para uma coleção de arquivos de dados computadorizados. Os usuários de um sistema podem

realizar (ou melhor, solicitar) que o sistema realize diversas operações envolvendo tais arquivos – por exemplo:

- Acrescentar novos arquivos ao banco de dados;
- Inserir dados em arquivos existentes;
- Buscar dados de arquivos existentes;
- Excluir dados de arquivos existentes;
- Alterar dados em arquivos existentes;
- Remover arquivos existentes do banco de dados.

De uma forma acadêmica pode-se definir um banco de dados como *"uma coleção de dados persistentes, usada pelos sistemas de aplicação de uma determinada empresa"*.

O termo empresa, aqui, é simplesmente um termo genérico para qualquer organização comercial, científica, técnica ou outra organização razoavelmente autônoma. Uma empresa poderia ser um único indivíduo (com um pequeno banco de dados pessoal) ou uma corporação ou grande empresa completa (com um enorme banco de dados compartilhado), ou qualquer coisa entre esses extremos (uma fábrica, uma universidade, um banco, um hospital, uma unidade do governo, etc.). Qualquer empresa precisa necessariamente manter muitos dados sobre a sua operação. Esses dados são persistentes. As empresas que mencionamos, anteriormente, normalmente incluiriam os seguintes itens (respectivamente) entre seus dados persistentes: sobre produtos, sobre alunos, sobre pacientes, sobre planejamento, etc.

É difícil assinalar a real importância de um banco de dados sem o estudo e a prática dos conceitos relacionados, mas, em síntese, um banco de dados prevê a manutenção dos dados organizados, permitindo atualização, inclusão e exclusão de dados, garantindo consistência, integridade, acessos múltiplos, recuperação em caso de falhas e retornos tempestivos. Essas são características mais apreciadas e desejadas na utilização de um banco de dados.

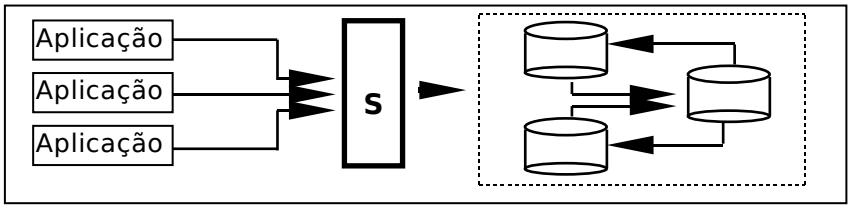
Um banco de dados deve conter dados dispostos numa ordem determinada por um esquema ou modelo, sempre atendendo a um propósito definido. Podem ser citados como exemplos a lista telefônica, a lista de códigos de endereçamento postal e outros.

1.3- SISTEMAS GERENCIADORES DE BANCO DE DADOS (SGBD)

Hoje em dia, a grande maioria dos programas comunica-se com os usuários utilizando interfaces gráficas de janelas. Entretanto, normalmente, os programas não contém todo o código referente a exibição dos dados da interface, mas utilizam gerenciadores de interface de usuário. Da mesma forma, para comunicar-se com processos remotos, usam gerenciadores de comunicação. Para manter grandes repositórios compartilhados de dados, ou seja, para manter banco de dados, são usados *Sistemas de Gerência de Bancos de Dados (SGBD)***.**

No *SGBD*, existe um gerenciador entre as aplicações e a base de dados comum a estas aplicações. Então reduz a *redundância*, evita a *inconsistência*, mantém a integridade e a transparência de dados, utiliza "*visões do usuário*", atende as exigências

de informações de todas as aplicações, facilita o *desenvolvimento* de novas aplicações e a aplicação das *restrições* de *segurança*.



É o software que trata de todo o acesso ao banco de dados. Conceitualmente, o que ocorre é o seguinte:

- Um usuário faz um pedido usando uma determinada sublinguagem de dados (geralmente SQL);
- O SGBD intercepta o pedido e o analisa;
- O SGBD, por sua vez, inspeciona o esquema externo, para esse usuário;
- O SGBD executa as operações necessárias sobre o banco de dados armazenado.

Um *SGBD* é uma coleção de programas que permitem ao usuário definir, construir e manipular Bases de Dados para as mais diversas finalidades.

Existem algumas regras básicas e claras para que um sistema de manipulação de dados possa ser considerado um SGBD. Fica implícito que se ao menos uma das características abaixo não estiver presente no nosso "candidato" a SGBD, este poderá ser um Gerenciador de Arquivo de altíssima qualidade, "quase" um SGBD, mas não um SGBD.

Regra	Explicação
Auto-Contenção	Um SGBD não contém apenas os dados em si, mas armazena completamente toda a descrição dos dados, seus relacionamentos e formas de acesso. Normalmente esta regra é chamada de Meta-Base de Dados.
Independência dos Dados	Quando as aplicações estiverem realmente imunes a mudanças na estrutura de armazenamento ou na estratégia de acesso aos dados, podemos dizer que esta regra foi atingida. Portanto, nenhuma definição dos dados deverá estar contida nos programas da aplicação. Quando você resolve criar uma nova forma de acesso, um novo índice, *se precisar alterar o código de seu aplicativo*, você não está lidando com um SGBD.
Abstração dos Dados	Em um SGBD real é fornecida ao usuário somente uma representação conceitual dos dados, o que não inclui maiores detalhes sobre sua forma de armazenamento real. O chamado Modelo de Dados é um tipo de abstração utilizada para fornecer esta representação conceitual. Neste modelo, um esquema das tabelas, seus relacionamentos e suas chaves de acesso são exibidas ao usuário, porém nada é afirmado sobre a criação dos índices, ou como serão mantidos, ou qual a relação existente entre as tabelas que deverá ser mantida íntegra. Assim se você desejar inserir um pedido em um cliente inexistente e esta entrada não for automaticamente rejeitada, você não está lidando com um SGBD.
Visões	Um SGBD deve permitir que cada usuário visualize os dados de forma diferente daquela existente previamente no Banco de Dados. Uma visão consiste de um subconjunto de dados do Banco de Dados, necessariamente derivados dos existentes no Banco de Dados.
Transações	Um SGBD deve gerenciar completamente a integridade referencial definida em seu esquema, sem precisar em tempo algum, do auxílio do programa aplicativo. Desta forma exige-se que o banco de dados tenha ao menos uma instrução que permita a gravação de uma série

	modificações simultâneas e uma instrução capaz de cancelar uma série modificações. Por exemplo, imaginemos que estejamos cadastrando um pedido para um cliente, que este deseje reservar 5 itens de nosso estoque, que estão disponíveis e portanto são reservados, porém existe um bloqueio financeiro (duplicatas em atraso) que impede a venda. A transação deverá ser desfeita com apenas uma instrução ao Banco de Dados, sem qualquer modificação suplementares nos dados. Caso você se obrigue a corrigir as reservas, através de acessos complementares, você não está lidando com um SGBD.
Acesso Automático	Em um Gerenciador de Arquivos uma situação típica é o chamado Dead-Lock, o abraço mortal. Esta situação indesejável pode ocorrer toda vez que um usuário travou um registro em uma tabela e seu próximo passo será travar um registro em uma tabela relacionada à primeira, porém se este registro estiver previamente travado por outro usuário, o primeiro usuário ficará paralisado, pois, estará esperando o segundo usuário liberar o registro em uso, para que então possa travá-lo e prosseguir sua tarefa. Se por hipótese o segundo usuário necessitar travar o registro travado pelo primeiro usuário, afirmamos que ocorreu um abraço mortal, pois cada usuário travou um registro e precisa travar um outro, justamente o registro anteriormente travado pelo outro. Imaginemos um caso onde o responsável pelos pedidos acabou de travar o Registro Item de Pedido, e, necessita travar um registro no Cadastro de Produtos, para indicar uma nova reserva. Se concomitantemente estiver sendo realizada uma tarefa de atualização de pendências na Tabela de Itens, e para tanto, previamente este segundo usuário travou a Tabela de Produtos, temos a ocorrência do abraço mortal. Se a responsabilidade de evitar esta ocorrência for responsabilidade da aplicação, você não está lidando com um SGBD.

As funções do SGBD devem incluir o suporte a, pelo menos, todos os itens a seguir:

- ***Definição de Dados***: o SGBD deve ser capaz de aceitar definições de dados em formato fonte e convertê-los para o formato objeto adequado.
- ***Manipulação de Dados***: O SGBD deve ser capaz de lidar com as requisições do usuário para buscar atualizar ou excluir dados existentes no banco de dados, ou para acrescentar novos dados ao banco de dados.
- ***Otimização e Execução***: O SGBD deve ser capaz de determinar um modo eficiente de implementar a requisição.
- ***Segurança e Integridade de Dados***: O SGBD, ou algum subsistema chamado, pelo SGBD, deve monitorar requisições de usuários e rejeitar toda a tentativa de violar as restrições de segurança e integridade definidas pelo DBA.
- ***Recuperação de Dados e Concorrência***: O SGBD deve impor certos controles de recuperação e concorrência.
- ***Dicionário de Dados***: Pode ser considerado um banco de dados isolado (mas um banco de dados do sistema, não um banco de dados do usuário); ele contém “dados sobre os dados” – ou seja, definições de outros objetos do sistema, em vez de somente “dados crus”.
- ***Desempenho***: É irrelevante dizer que o SGBD deve realizar todas as funções identificadas anteriormente de forma tão eficiente quanto possível.

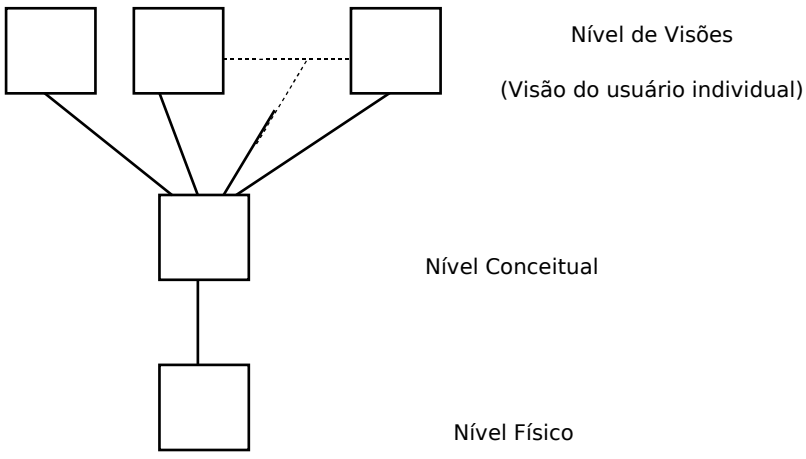
1.3.1- ABSTRAÇÃO DE DADOS

Um *SGBD* é composto de uma coleção de arquivos inter-relacionados e de um conjunto de programas que permitem aos usuários fazer acesso e modificar esses arquivos. O grande objetivo de um sistema de bancos de dados é prover os usuários uma visão *abstrata* dos dados, isto é, o sistema omite certos detalhes de como os dados são armazenados e mantidos. Entretanto, para que o sistema possa ser utilizado, os dados devem ser buscados eficientemente. Este conceito de abstração de dados tem direcionado o projeto de estruturas de dados complexos para a representação de dados em banco de dados. Uma vez que muitos dos usuários de banco de dados não são treinados em computação, a complexidade está escondida deles através de diversos níveis de abstração que simplificam a interação dos usuários com o sistema.

- **Nível físico**. O nível mais baixo de abstração descreve *como* os dados estão realmente armazenados. No nível físico são descritas a estrutura física de armazenamento do banco de dados, sua organização de arquivos e seus métodos de acesso.

- **Nível conceitual**. Neste nível de abstração descreve *quais* dados estão armazenados de fato no banco de dados e as relações que existem entre eles. O nível conceitual é a visão do conjunto de usuários, portanto a visão geral do banco de dados.

- **Nível de visões**. O mais alto nível de abstração descreve apenas parte do banco de dados. O nível de visões é o mais próximo dos usuários. Voltado para a forma como os dados são vistos pelos usuários individuais. Um determinado usuário, tanto pode ser um programador de aplicações como um usuário de terminal on-line ou um usuário final de qualquer grau de sofisticação. Descreve as partes do banco de dados que são do interesse de um ou vários usuários, escondendo as demais partes que não são do seu interesse.



1.3.2- CLASSIFICAÇÃO

Os sistemas gerenciadores de banco de dados podem ser classificados de diversas maneiras, como por modelo de dados, por número de usuários, por localização, entre outros.

Por Modelo de Dados

Modelo	***Descrição***
Modelo Hierárquico	Este modelo de banco de dados é descrito por um diagrama de estrutura de árvore, com dois componentes básicos: as caixas correspondentes aos tipos de registro e as linhas que representam as ligações entre os tipos de registro. Pode ser representado diretamente quando os dados são definidos com relacionamentos hierárquicos, isto é, um segmento raiz, localizado no topo da estrutura, que se relaciona com outros segmentos de cima para baixo, da esquerda para a direita, denominado de diagrama de estrutura de dados ou de árvore (relacionamento do topo para baixo e das folhas para dentro). Este modelo se tornou viável quando a estrutura dos discos de armazenamento se tornou endereçável, possibilitando a representação hierárquica das informações coma utilização do endereçamento físico daqueles dispositivos. Todos os bancos de dados apresentam vantagens e desvantagens em sua utilização, e com o banco de dados modelo hierárquico não é diferente. Ele é um modelo bastante simplificado, representa naturalmente sistemas que possuem uma hierarquia bem definida, não representando sistemas que não possuem tais características, além de aumentar os índices de redundância (duplicidade), possuir tempos de consulta e processamento reconhecidamente altos.
Modelo de Rede	Pode ser visto como uma extensão mais completa do modelo hierárquico, apenas livre das dificuldades encontradas no modelo anterior. Os controles existentes quanto aos recursos de armazenamento, estrutura e caminhos utilizados para navegar de um registro a outro tornam o modelo de rede mais completo. O banco de dados em rede é composto de uma estrutura mais completa, possui as propriedades básicas de registros, conjuntos e ocorrências, e utiliza a linguagem de definição de banco de dados (DDL) e a linguagem de manipulação de dados (DML), além de permitir uma evolução mais eficiente do modelo. A estrutura é formada de entidade (registros), atributos (itens de dados), tipos de registros e ocorrência do registro. Tanto no modelo hierárquico quanto o de rede são chamados de sistemas de navegação, pois as aplicações devem ser construídas para atravessar um conjunto de registros interligados previamente. A maior vantagem do modelo de rede frente ao hierárquico é um melhor controle sobre a redundância de dados.
Modelo Relacional	O banco de dados mais utilizado atualmente é o banco de dados relacional, principalmente pelas suas fortes características de seguranças, compartilhamento e integridade dos dados, fatores primordiais para uma administração eficaz da informação de qualquer empresa. Com o próprio nome diz, esse tipo de banco de dados é composto de relações entre entidades, denominadas tabelas, seguindo o mesmo conceito matemático de relação. As tabelas se relacionam por meio de chaves denominadas estrangeiras, que podem ser simples, formadas por apenas um atributo, ou compostas, formadas por vários atributos. Por atributo compreende-se a representação de um tipo de informação. Para que um modelo seja considerado relacional, os dados devem ser armazenados em tabelas, nenhum ponteiro ou ligação deve ser visível ao usuário, a linguagem de consulta deve ser relacionalmente completa e as consultas podem ser expressas sem uso de iterações ou recursões. Cada linha da tabela, formada por um conjunto de colunas, representa um registro (ou *tupla*). Os registros não precisam necessariamente conter dados em todas as colunas, ou seja, seus valores podem ser nulos. As colunas de uma tabela são também chamadas de *campos*. Os campos possuem características que definem o dado que será armazenado. Os sistemas de banco de dados possuem normas que irão reger os dados que são armazenados. Em um banco de dados pode existir uma ou centenas de tabelas. O limitador é imposto exclusivamente pela ferramenta computacional ou SGBD utilizado. Cada coluna de uma tabela obedece às regras definidas na sua criação. Cada coluna recebe um tipo de dado, que representa o conjunto de valores que podem ser armazenados em cada uma das colunas. Basicamente, cada coluna representa o tipo de cada dado, ou seja, as regras de entrada dos dados, evitando a incompatibilidade. Já as linhas representam todos os dados cadastrados, ou seja, um conjunto de colunas ou campos. As linhas representam o número de registros cadastrados na tabela e as colunas representam os tipos de dados ou campos cadastrados. Alguns campos da tabela, além do tipo de dados, possuem uma propriedade adicional. Esses campos recebem o nome de chave. Existem dois tipos de chaves: *chave primária* (É a chave que identifica cada registro, dando-lhe unicidade. Essa chave nunca de se repetirá dentro da estrutura da tabela) e *chave estrangeira* (É a chave formada pela chave primária de outra tabela e a chave de um campo da tabela externa que recebe o relacionamento. Essa chave, de forma resumida, define um relacionamento entre as tabelas e pode ocorrer repetidas vezes).
Modelo Orientado a Objetos	Trata-se basicamente de um sistema em que a unidade de armazenamento é vista como um objeto, ou seja, ela tem propriedades e não campos. A principal

Modelo	Descrição
	característica desse tipo de banco de dados é a abstração dos dados. Combina os benefícios e conceitos da programação orientada a objetos com a funcionalidade dos banco de dados. A vantagem é a lógica contida no objeto e a possibilidade de ser reutilizado várias vezes em diversas aplicações.

Por Número de Usuários

Monousuários é o tipo de banco de dados, construído para utilização de apenas um usuário de cada vez, diferentemente dos bancos multiusuários, que garantem a utilização dos dados de forma compartilhada por vários usuários ao mesmo tempo.

Por Localização

Banco de dados distribuído é uma estrutura com dois ou mais arquivos armazenados em diferentes locais, em uma rede de computadores, responsável pelo armazenamento e recuperação das informações de forma transparente para os usuários. As características principais que um banco de dados distribuído deve possuir são descritas a seguir:

- Tolerância a falhas;
- Independência em relação ao site central;
- Autonomia para as transações locais;
- Independência de localização;
- Independência de replicação;
- Independência de hardware e de rede;
- Processamento distribuído de consultas;
- Independência do sistema operacional;
- Gerenciamento de transações distribuídas;
- Controle de consistência.

A implementação de um banco de dados distribuído não constitui tarefa fácil, cabendo uma análise bem-feita sobre as necessidades de se usar ou não essa estrutura, optando-se, em muitos casos, pela utilização de banco de dados centralizado, onde todos os programas e processamentos ocorrem no computador hospedeiro, que traz como maior vantagem a segurança da centralização.

As vantagens apresentadas pelo banco de dados em relação à arquitetura tradicional de arquivos são:

- ***Redução e eliminação de redundâncias***. Na arquitetura tradicional de arquivos, cada sistema possui seus próprios arquivos, provocando desperdício de espaço de armazenamento quando as informações são duplicadas. Com a utilização de banco de dados, os dados comuns a mais de um sistema podem ser compartilhados, o que permite o acesso a uma única fonte da informação.
- ***Eliminação de inconsistências***. Como a informação está armazenada em um único local, compartilhada por vários sistemas, e é atualizada por apenas uma fonte, os usuários utilizam informações mais confiáveis.
- ***Compartilhamento de dados***. Permite que mais de uma aplicação ou usuário, independentemente da operação que está sendo realizada, tenha acesso aos dados simultaneamente e de uma forma mais segura. O compartilhamento também diminui os custos quando da implementação de novas aplicações ou sistemas.

- ***Restrição de segurança***. Permite definir perfis de utilização dos dados, a partir de níveis de acesso definidos pelo gestor da informação, impedindo que pessoas não autorizadas utilizem ou atualizem determinada informação.
- ***Reforço de padrões***. Garante que os dados armazenados sigam um padrão definido no que tange a nomenclatura e formato de armazenamento, facilitando a utilização dos dados por vários sistemas.
- ***Independência dos dados***. O armazenamento dos dados é facilitado por possuir definições diferentes para cada tabela do banco.
- ***Integridade dos dados***. Permite que a integridade dos dados seja mantida por regras predefinidas, impedindo que uma chave de uma tabela não tenha correspondência em outra.
- ***Restauração dos dados***. Possibilita um controle da restauração dos dados em caso de falha de ambiente ou de um aplicativo, mantendo a integridade dos dados.
- ***Reorganização dos dados***. Permite que os dados sejam reorganizados de forma a eliminar espaços inutilizados, redimensionando as tabelas e, por conseqüência, melhorando a performance de acesso.
- ***Relacionamentos***. Ocorre a implementação de relacionamentos complexos entre os dados.

CAPITULO II

MODELOS DE BANCO DE DADOS

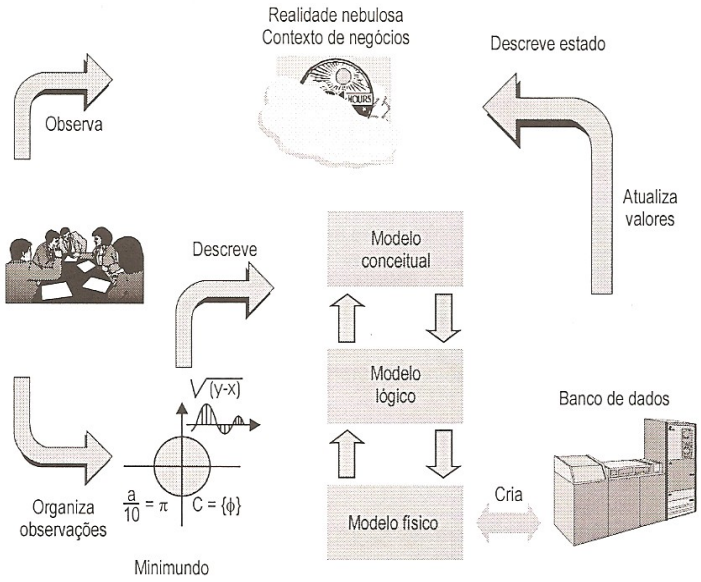
Um modelo de (banco de) dados é uma descrição dos tipos de informações que estão armazenadas em um banco de dados.

Modelo de Dados
=
Descrição formal da estrutura de um banco de dados

Para construir um modelo de dados usa-se uma *linguagem de modelagem de dados*. Estas linguagens podem ser classificadas de acordo com a forma de apresentar modelos, em linguagens *textuais* ou linguagens *gráficas*. Existem linguagens de modelagem para descrever modelos de dados em diferentes níveis de abstração e com diferentes objetivos. Cada representação de um modelo de dados através de uma linguagem de modelagem de dados recebe a denominação de *esquema de banco de dados*.

De acordo com a intenção do modelador, um banco de dados pode ser modelado (descrito) em vários níveis de abstração. Um modelo de dados que servirá para explicar a um usuários leigo em informática qual é a organização de um banco de dados provavelmente não conterá detalhes sobre a representação em meio físico, das informações. Já um modelo de dados usado por um técnico para otimizar a performance de acesso ao banco de dados conterá mais detalhes de como as informações estão organizadas internamente e, portanto, será menos abstrato.

Assim como é possível, construir modelos de dados em vários níveis de abstração , também, é possível usar diferentes técnicas, aplicando diferentes conceitos aos construir modelos. Ao conjunto de conceitos usados na construção de um modelo denominamos de *abordagem de modelagem*.



2.1- MODELO CONCEITUAL

Representa e descreve a realidade do ambiente do problema, constituindo-se em uma visão global dos principais dados e seus relacionamentos (estruturas de informação), completamente independente dos aspectos de sua implementação tecnológica. Quando falamos em modelo conceitual, estamos nos referindo aquela que deve ser a primeira etapa de um projeto de um banco de dados.

O objetivo do modelo conceitual é descrever de forma simples e facilmente compreendida pelo usuário final as informações de um contexto de negócios, as quais devem ser armazenadas em um banco de dados. É uma descrição de alto nível, mas que tem a preocupação de captar e retratar a realidade de uma organização, processo de negócio, setor, repartição, departamento, etc.

Devemos destacar que o modelo conceitual não retrata nem é vinculado aos aspectos ligados a abordagem de banco de dados que será utilizado e tampouco se preocupa com as formas de acesso ou estruturas físicas implementadas por SGBD específico.

O resultado de um modelo conceitual é um esquema gráfico que representa a realidade das informações existentes em um determinado contexto de negócios, assim como as estruturas de dados em que estão organizadas essas informações.

O modelo conceitual nunca deve ser construído com considerações sobre processos de negócios. O foco deve ser dirigido sempre ao entendimento e representação de uma realidade de um contexto.

2.2- MODELO LÓGICO

Ele somente tem o seu início após a criação do modelo conceitual. A partir deste ponto vamos considerar uma das abordagens possíveis da tecnologia de SGBDs (relacional, hierárquico, rede e orientado a objetos) para a estruturação e estabelecimento da lógica dos relacionamentos existentes entre os dados definidos no modelo conceitual.

O modelo lógico descreve em formato as estruturas que estarão no banco de dados de acordo com as possibilidades permitidas pela sua abordagem, mas sem considerar, ainda, nenhuma característica específica de SGBD. Isso resulta em um esquema lógico de dados sob a óptica de uma das abordagens citadas, através do emprego de uma técnica de modelagem de dados orientada às restrições de cada abordagem.

2.3- MODELO FÍSICO

O modelo físico será construído a partir do modelo lógico e descreve as estruturas físicas de armazenamento de dados, tais como:

- Tipo e tamanho de campos;
- Índices;
- Domínio de preenchimento desses campos;
- Nomenclaturas;
- Exigência de conteúdo;
- Gatilhos; etc.

Estas são projetadas de acordo com os requisitos de processamento e uso mais econômico dos recursos computacionais. Esse modelo detalha o estudo dos métodos de acesso ao SGBD para a criação dos índices necessários para cada informação colocada nos modelos conceitual e lógico.

2.4- PROJETO DE BANCO DE DADOS

O projeto de um novo banco de dados dá-se em três fases:

- **Modelagem Conceitual**
  Nesta primeira fase, é construído um modelo conceitual, na forma de um diagrama entidade-relacionamento. Este modelo independe da implementação.

- **Projeto Lógico**
  Objetiva transformar o modelo conceitual obtido na primeira fase em um modelo lógico. O modelo lógico define como o banco de dados será implementado em um SGBD específico.

- **Projeto Físico**
  Nesta etapa, o modelo de BD é enriquecido com detalhes que influenciam no desempenho do BD, mas não interferem em sua funcionalidade. Alterações neste modelo não afetam as aplicações que usam o BD.

CAPITULO III

BANCO DE DADOS RELACIONAL

Criado por Edgar F. Codd, nos anos 70, começou a ser realmente utilizado nas empresas a partir de 1987. A abordagem relacional está baseada no principio de que as informações em uma base de dados podem ser consideradas relações matemáticas e que estão representadas de maneira uniforme com o uso de tabelas bidimensionais.

Este princípio coloca os dados direcionados a estruturas mais simples de armazenamento de dados, que são as tabelas, e nas quais a visão do usuário é privilegiada.

3.1- TEORIA RELACIONAL

A abordagem relacional representa uma forma de descrever um banco de dados através de conceitos matemáticos simples: a teoria dos conjuntos.

Voltada, principalmente, a melhorar a visão dos dados pelos usuários, a abordagem relacional faz com que os usuários vejam o banco de dados como um conjunto de tabelas bidimensionais, originadas em linhas e colunas.

São conjuntos de dados vistos segundo um conjunto de TABELAS, e as operações que as utilizam são feitas por linguagens que o manipulam, não sendo procedurais, ou seja, manipulando conjuntos de uma só vez.

O conceito principal vem da tória dos conjuntos atrelado à concepção de que não é relevante ao usuário saber *onde os dados estão nem como os dados estão* (transparência).

Os usuários manipulam estes objetos dispostos em linhas e colunas das tabelas que possuem informações sobre o mesmo assunto de forma estruturada e organizada.

Para lidar com estes objetos, o usuário conta com um conjunto de operadores e funções de alto nível, constantes na álgebra relacional.

A Teoria Relacional possui premissas que definem uma tabela de dados:

- Cada uma das tabelas é chamada de relação;
- O conjunto de uma linha e suas colunas é chamado tupla;
- Cada coluna dessa tabela tem um nome e representa um domínio da tabela;
- Não há duas linhas iguais;
- Usamos nomes para fazer referência às colunas;
- Cada tabela tem um nome próprio, distinto de qualquer outra tabela no banco de dados.

Exemplo 1

Tabela de CDs

Nº do CD	Dt da Gravação	Título do Conteúdo	Responsável	Local onde está guardado

1	24/01/2001	Clipart	Samir	Estojo Verde
3	13/02/2000	IRRF 2000	Felipe	Caixa de Documentos
2	14/12/2000	Backup Textos	Felipe	Estojo Azul
4	25/01/2000	Fotos Gramado	Samir	Caixa de Álbum 3

Podemos observar a tabela de CDs que os conceitos listados anteriormente em relação às tabelas relacionais estão destacados:

- A ordem das linhas é irrelevante, pois o CD de número 2 vem após o CD de número 3. Se observarmos mais detalhadamente, veremos que o CD número 3 tem uma data inclusive menor que o CD de número 2;
- Nenhuma linha se repete nesta tabela;
- A ordem das colunas também não tem nenhum destaque, ou importância, pois não estão em nenhuma ordem lógica;
- Todas as colunas têm um nome, que identifica o seu conteúdo, ou melhor, o significado do valor de seu conteúdo.

Exemplo 2

Tabela Funcionários

Nome	Sexo	Matrícula	Departamento	Cargo	Salário
João Carlos	M	373	TI Operações	Operador	3.000,00
Carlos Brito	M	872	TI Programação	Programador I	3.500,00
Silvia Morais	F	963	TI Análise	Analista Sistemas II	5.500,00
Claudia Tereza	F	161	TI Gerência	Secretaria	1.500,00
Pedro Julio	M	292	RH	Diretor	6.000,00
Pedro Julio	M	574	TI Análise	Analista Sistemas I	4.500,00

Vamos analisar a tabela acima:

- A coluna matrícula não indica nenhuma ordem para as linhas, confirmando o conceito de que a ordem das linhas é irrelevante;
- Todas as colunas têm um nome que significa algo sobre o assunto, o que é bem evidente;
- A disposição das colunas não tem nenhuma finalidade de classificação, tampouco indica ordem de leitura dos dados;
- A modificação da ordem das colunas não acarreta prejuízo nenhum do assunto ou da representatividade da tabela;

Salário	Nome	Sexo	Matrícula	Departamento	Cargo
3.000,00	João Carlos	M	373	TI Operações	Operador
3.500,00	Carlos Brito	M	872	TI	Programador I

				Programação	
5.500,00	Silvia Morais	F	963	TI Análise	Analista Sistemas II
1.500,00	Claudia Tereza	F	161	TI Gerência	Secretaria
6.000,00	Pedro Julio	M	292	RH	Diretor
4.500,00	Pedro Julio	M	574	TI Análise	Analista Sistemas I

O importante é que toda vez que nos referimos à tabela funcionários visualizemos os dados que ela contém, e simbolizando valores possíveis, pois assim vamos mentalizando e nos abstraindo em algum objeto que a tabela representa.

3.2- CARACTERÍSTICAS PRINCIPAIS DE UMA RELAÇÃO (TABELA)

Vamos definir a concepção técnica de uma relação (tabela).

NumReg	NomeFunc	DtAdmissão			Sexo	Fone	CodDepto
		Dia	Mês	Ano			
101	Luis Sampaio	10	08	2003	M	2565-8974	D5
104	Carlos Pereira	02	08	2004	M	31331-4649	D6
134	José Alves	23	03	2002	M	2386-8897	D1
121	Luiz Paulo Souza	10	05	2001	M	2241-5896	D5
25	Marta Silveira	05	12	2002	F	2693-5521 2594-8523 2594-8522	D5
139	Ana Luiza Magalhães	12	01	2003	F	4545-8899	D6
123	Pedro Sergio Doto	29	01	2003	M	2496-8855	D3
148	Larissa Silva	01	06	2002	F	4289-9675	D6
115	Roberto Fernandes	15	06	2003	M	2685-8132	D5
22	Sergio Nogueira	10	10	2000	M	2594-3122	D4

Atributo Composto

Atributo Multivalorado

Todos os atributos (colunas) de uma relação devem ser atômicos, isto é, indivisíveis em termos de valores e componentes.

Isso quer dizer que não existem colunas do tipo subgrupo, todas são itens elementares não subdivididos em nenhuma hipótese. E, também, não é permitida a

existência da múltipla ocorrência de valores (multivalorados) em nenhum de seus atributos (colunas).

É importante a compreensão de que cada linha de uma tabela representa um objeto, um assunto que é descrito pelos valores de cada uma dessas colunas.

O esquema de uma relação, ou seja, a sua definição pode ser interpretada como uma declaração, ou um tipo de afirmação.

O exemplo de uma tabela funcionário apresenta:

- Número de registro (NumReg), nome de funcionário (NomeFunc), data de admissão (DtAdmissão), sexo (Sexo), telefone (Fone) e departamento (CodDepto);
- Cada tupla (linha) da relação (tabela) deve ser interpretada como fato ou uma ocorrência particular dessa afirmação;

NumReg	NomeFunc	DtAdmissão	Sexo	Fone	CodDepto
101	Luis Sampaio	10/08/2003	M	2565-8974	D5
104	Carlos Pereira	02/08/2004	M	31331-4649	D6
134	José Alves	23/03/2002	M	2386-8897	D1
121	Luiz Paulo Souza	10/05/2001	M	2241-5896	D5
25	Marta Silveira	05/12/2002	F	2693-5521	D5
139	Ana Luiza Magalhães	12/01/2003	F	4545-8899	D6
123	Pedro Sergio Doto	29/01/2003	M	2496-8855	D3
148	Larissa Silva	01/06/2003	F	4289-9675	D6
115	Roberto Fernandes	15/06/2002	M	2685-8132	D5
22	Sergio Nogueira	10/10/2000	M	2594-3122	D4

3.3- DOMÍNIO

Representa o conjunto de valores atômicos admissíveis de um componente (coluna) de uma relação (tabela).

Exemplo

- Telefone: conjunto de 10 dígitos;
- Idade_Empregado: 16 ≥ idade ≤ 70;
- Departamentos: conjunto de departamentos de uma empresa.

A cada domínio está associado um tipo de dados ou formato.

Exemplo

- Telefone: (ddd) dddd-ddddd, onde d={0,1,2,3,4,5,6,7,8,9}
- Idade_Empregado: número inteiro entre 16 e 70

Restrições de domínio são estabelecidas determinando-se domínios de valores para cada coluna de uma tabela. Normalmente são estabelecidos e definidos os valores que uma coluna de uma tabela pode ter, isto é, o domínio da coluna. Permitem, por exemplo:

- Verificar os valores inseridos em um banco de dados;
- Testar consultas para garantir que as comparações tinham sentido;
- Em geral, o domínio é especificado por tipos primitivos de dados, tais como: interger, float, char, date, Money, time, etc.

Também, podem ser descritos pela definição de subconjuntos de tipos primitivos ou de listas enumeradas, ou seja, listas de valores possíveis de existir na coluna.

3.4- CHAVE PRIMÁRIA

Em toda e qualquer tabela existente em um banco de dados relacional haverá sempre uma coluna ou um conjunto de colunas concatenadas, cujos valores são únicos na tabela, isto é, nunca se repete aquele valor em nenhuma outra linha da tabela.

Essa coluna ou conjunto de colunas concatenadas identifica uma única linha da tabela. Então dizemos que essa coluna ou conjunto de colunas forma a chave primária da tabela.

NumReg	NomeFunc	DtAdmissão	Sexo	Fone	CodDepto
101	Luis Sampaio	10/08/2003	M	2565-8974	D5
104	Carlos Pereira	02/08/2004	M	31331-4649	D6
134	José Alves	23/03/2002	M	2386-8897	D1
121	Luiz Paulo Souza	10/05/2001	M	2241-5896	D5
25	Marta Silveira	05/12/2002	F	2693-5521	D5
139	Ana Luiza Magalhães	12/01/2003	F	4545-8899	D6
123	Pedro Sergio Doto	29/01/2003	M	2496-8855	D3
148	Larissa Silva	01/06/2003	F	4289-9675	D6
115	Roberto Fernandes	15/06/2002	M	2685-8132	D5
22	Sergio Nogueira	10/10/2000	M	2594-3122	D4

Na tabela acima, qual coluna ou qual conjunto de colunas que concatenadas formam um identificador único para cada linha desta tabela?

Analisando a referida tabela, podemos ver que existe somente um valor de matrícula para cada linha, que não se repete, logo podemos determinar que matrícula é a chave primária da tabela funcionários.

3.5- VALORES NULOS

Por definição, todas as linhas de uma tabela têm de ser distinguíveis umas das outras, devem possuir um identificador único.

Um identificador de chave primária nulo significa dizer que existe uma ocorrência (linha) na tabela sem identificação única, ou não distinguível.

Se uma tabela relacional tem uma chave primária, que é um valor único para cada linha da tabela, logo esse valor não pode em hipótese alguma estar nulo, ou seja, ser desconhecido.

Estendendo este conceito, destacamos que nenhuma das colunas que participam da composição da chave primária pode ter valor nulo, pois o resultado da concatenação seria nulo, em uma operação de concatenação de valores.

3.6- REGRA DE INTEGRIDADE DE IDENTIDADE

A identificação de uma linha de uma tabela não pode ser feita por um valor desconhecido, motivo pelo qual a chave primária de uma tabela não pode possuir nenhum elemento de sua composição com valor nulo.

3.6.1- CHAVE PRIMÁRIA

- Coluna ou concatenação de colunas;
- Valor único na tabela;
- Cada linha tem um valor diferente na chave primária;
- Não existem valores nulos na chave primária.

Atributos cujos valores no mundo real podem ser duplicados não devem ser definidos como chaves primárias de uma tabela (Nome, por exemplo).

Em geral, uma tabela pode ter mais de uma chave que possua a capacidade de identificação única das linhas da tabela. Nesse caso, cada uma dessas chaves da tabela é chamada de ***CHAVE CANDIDATA***.

Uma das chaves será definida como primária e as outras ficam como chaves alternativas à chave primária.

Em geral, entre todas as chaves candidatas, escolhe-se para chave primária aquela com o menor número de atributos ou mais significativa conceitualmente na identificação de cada fato ou ocorrência (linha) de uma tabela.

Esquema de uma Tabela

- A definição de uma tabela é realizada por um formato denominada esquema da tabela;
- Esse formato destaca a chave primária por um sublinhado no nome da coluna: para a tabela de funcionários que apresentamos o esquema seria Funcionário (NumReg, NomeFunc, DtAdmissao, Sexo, Fone, CodDepto).

3.6.2- CHAVE ESTRANGEIRA

Uma tabela relacional é uma estruturação dos dados por assunto, organizada em tabelas com linhas e colunas, e cada linha é a representação de uma ocorrência de um objeto, um assunto, descrita por valores em cada coluna. Dessas colunas já sabemos que uma ou um conjunto delas forma o que definimos como o identificados único de cada linha que é a chave primária da tabela.

Exemplo

Vamos analisar as propriedades que as colunas de uma tabela relacional podem ter e as regras dessas propriedades

Existem três tabelas nesse pequeno banco de dados.

São as tabelas referentes aos produtos de nossa cozinha, neste caso somente os alimentos. Uma que chamaremos de tabela de Estoque de alimentos, uma de Fornecedores e uma de Unidade de armazenamento.

Uma característica importante nas tabelas relacionais é que elas têm muitas vezes colunas comuns.

Estoque de Alimentos

Alimento	Quantidade	Data Validade	Fabricante	Unidade
Feijão	2	20/08/2004	2	1
Leite	3	12/07/2004	4	2
Açúcar	5	12/08/2004	1	1
Arroz	3	10/10/2004	6	1
Azeite	2	12/03/2004	5	6
Café	1	12/12/2004	3	1

Fornecedores

Fabricante	NomeFab
2	Coral
4	CCPL
1	União
6	Tio João
5	Galo
3	Pilão

Unidades de Armazenamento

Unidade	Descrição

1	Kg
2	Litro
3	Peça
4	Envelope
5	Pote 500g
6	Vidro 500g

Esquema do Banco de Dados

- Estoque de Alimentos {Alimento, Quantidade, Data Validade, Fabricante, Unidade}
- Fornecedores {Fabricante, NomeFabricante}
- Unidades de Armazenamento {Unidade, Descrição}

Observe que a tabela de Estoque de Alimentos tem as colunas Número do Fabricante e Código da Unidade, que também existem respectivamente nas tabelas Fornecedores e Unidade de Armazenamento.

Esta é uma característica que em primeiro lugar tem como objetivo inicial evitar que sejam inseridos na tabela de alimentos, por exemplo, valores relativos a um mesmo fornecedor de duas maneiras: Tio João e T. João, fazendo com que apresentássemos o mesmo produto como sendo de dois fornecedores diferentes.

Isso ajuda a eliminar ou diminuir erros de entrada de dados nos sistemas, e manter a consistência do banco de dados, pois utilizamos o número do fornecedor em lugar de, talvez, digitar o seu nome.

O que significa uma tabela ter coluna ou colunas que existem em outra tabela?

Analisando as tabelas e o esquema do banco de dados, observamos que cada tabela tem uma chave primária.

A tabela Estoque de Alimentos tem como chave primária a coluna Alimento.

A tabela Fornecedores tem como chave primária a coluna Fabricante, e a tabela Unidade de Armazenamento tem como chave primária a coluna Unidade.

O que significa quando temos um campo que é chave primária de uma tabela que faz parte dos campos de outra tabela?
Isso é o que denominamos de ***chave estrangeira***.

É uma referência de um elemento de uma tabela a um elemento de outra tabela, uma relação entre as tabelas, uma ligação lógica entre elas.

Então, no exemplo, a coluna Fabricante na tabela Estoque de Alimentos é uma chave estrangeira, assim como Unidade também é outra chave estrangeira nesta tabela.

Uma tabela pode ter tantas chaves estrangeiras quantas forem as suas associações a outras tabelas.

Uma tabela pode ter um conjunto de atributos que contêm valores com o mesmo domínio de um conjunto de atributos que formam a chave primária de uma outra tabela.

3.7- INTEGRIDADE REFERENCIAL

Quando colocamos uma coluna como chave estrangeira em uma tabela, assumimos responsabilidade com o banco de dados por assim dizer.

As colunas pertencentes à chave estrangeira da tabela A devem ter o mesmo domínio das colunas pertencentes à chave primária da tabela B.

O valor de uma chave estrangeira em uma tabela A deve ser um valor de chave primária da tabela B, ou então ser nulo.

Sintetizando: uma tabela contém uma chave estrangeira, então o valor dessa chave só pode ser:

- Nulo – neste caso pode, pois representa a inexistência de referência para uma linha da tabela.
- Igual ao valor de alguma chave primária na tabela referenciada.

Como seria ter uma tabela com chave estrangeira nula? Vejamos:

NumReg	NomeFunc	DtAdmissão	Sexo	CdCargo	CodDepto
101	Luis Sampaio	10/08/2003	M	C3	D5
104	Carlos Pereira	02/08/2004	M	C4	D6
134	José Alves	23/03/2002	M	C5	D1
121	Luiz Paulo Souza	10/05/2001	M	C3	D5
123	Pedro Sergio Doto	29/01/2003	M	C1	Nulo
115	Roberto Fernandes	15/06/2002	M	C3	D5
22	Sergio Nogueira	10/10/2000	M	C2	D4

Na linha DE Pedro Sergio Doto o valor para CodDepto está nulo, o que pode significar que ele ainda não está alocado a nenhum departamento, ou foi deslocado de algum departamento.

O que importa é que ele não tem departamento assinalado, o que é uma situação válida.

O que não pode haver é um valor de chave estrangeira que não exista como chave primária de nenhuma linha da tabela referenciada, no caso a tabela Departamento.

Na definição de uma chave estrangeira somente podemos nos referenciar a uma chave primária de uma outra tabela? Nem sempre isso é verdade.

Na criação de uma tabela estrangeira, além de podemos nos referenciar a um campo chave primária de outra tabela, também podemos nos referenciar a uma coluna que tenha sido definida como única, uma chave candidata.

Qual a razão da integridade referencial? O que ela implica?

Existe um conjunto de regras de operação para um banco de dados que coloca restrições, regras de atualização das tabelas do banco de dados, de forma a garantir e manter a integridade referencial. São elas:

Regra	Definição
Deleção Restrita	Ao excluir (deletar) a tabela pai , se ela possuir filhos relacionados (ou seja, se o departamento tiver funcionários), a exclusão é impedida. Isso evita que o bando de dados fique inconsistente, ou seja, linhas de Funcionário com valor de chave estrangeira inexistente como chave primária na tabela associada. Outras opções para garantir a integridade

	de referências do banco de dados seriam excluir todos os filhos em cascata, fazendo com que todos os funcionários referenciem um departamento-padrão ou fazer com que todos os funcionários fiquem sem departamento.
Inclusão e Linha Restrita	Ao inserir um funcionário, caso seja obrigatório que ele já possua departamento associado, verifica se ele está relacionado a um departamento existente na tabela Departamento, senão impede a operação.
Atualização e Linha Restrita	Ao atualizar a chave estrangeira de uma tabela, verifica se existe uma linha da tabela associada que possua como chave primária o novo valor da chave estrangeira, senão impede essa operação. A opção cascata é sempre perigosa de ser utilizada em banco de dados, pois existe o risco de perder todos os dados existentes em uma determinada tabela se optar por apagar as suas linhas que estão associadas a uma determinada linha que será deletada da tabela que possui a chave primária referenciada.

Exemplo

Vamos fazer uma simulação de tabelas. Suponha que nossa tabela tenha somente os funcionários do departamento de vendas (D5).

Funcionário

NumReg	NomeFunc	DtAdmissão	Sexo	CdCargo	CodDepto
101	Luis Sampaio	10/08/2003	M	C3	D5
104	Carlos Pereira	02/08/2004	M	C4	D6
134	José Alves	23/03/2002	M	C5	D1
121	Luiz Paulo Souza	10/05/2001	M	C3	D5
123	Pedro Sergio Doto	29/01/2003	M	Nulo	D3
115	Roberto Fernandes	15/06/2002	M	C3	D5
22	Sergio Nogueira	10/10/2000	M	C2	D4

Departamento

CodDepto	NumDepto	RamalTel
D1	Assist. Técnica	2246
D2	Estoque	2589
D3	Administraçã o	2772
D4	Segurança	1810
D5	Vendas	2599
D6	Cobrança	2688

Se estabelecemos para a tabela Departamento a regra de cascata, então se apagarmos (deletar) a linha cuja chave primária é =”D5”, o resultado será a tabela funcionário como apresentada em seguida:

NumReg	NomeFunc	DtAdmissão	Sexo	CdCargo	CodDepto

Vazia, completamente sem linhas. A regra de cascata provoca que todas as linhas de tabelas associadas a essa chave primária sejam deletadas, apagadas para evitar que fiquem existindo no banco de dados referências às linhas inexistentes em uma tabela.

Mas não é somente o caso de deleção completa da tabela que preocupa, pois trabalhamos com nossa tabela original de funcionários:

NumReg	NomeFunc	DtAdmissão	Sexo	CdCargo	CodDepto
101	Luis Sampaio	10/08/2003	M	C3	D5
104	Carlos Pereira	02/08/2004	M	C4	D6
134	José Alves	23/03/2002	M	C5	D1
121	Luiz Paulo Souza	10/05/2001	M	C3	D5
123	Pedro Sergio Doto	29/01/2003	M	Nulo	D3
115	Roberto Fernandes	15/06/2002	M	C3	D5
22	Sergio Nogueira	10/10/2000	M	C2	D4

Se executarmos a mesma operação de deleção da linha relativa a Departamento de Vendas (D5).

Essa tabela perderá todas as linhas que estavam associadas pelo valor de chave estrangeira =”D5”, perdendo os dados de alguns funcionários:

Funcionário

NumReg	NomeFunc	DtAdmissão	Sexo	CdCargo	CodDepto
104	Carlos Pereira	02/08/2004	M	C4	D6
134	José Alves	23/03/2002	M	C5	D1
123	Pedro Sergio Doto	29/01/2003	M	Nulo	D3
22	Sergio Nogueira	10/10/2000	M	C2	D4

CAPITULO IV

ABORDAGEM ENTIDADE-RELACIONAMENTO

A técnica de modelagem de dados mais difundida e utilizada é a abordagem entidade-relacionamento (ER). Nesta técnica, o modelo de dados é representado graficamente através de um diagrama de entidade-relacionamento (DER). A abordagem ER foi criada por Peter Chen em 1976, podendo se considerada como um padrão de fato para a modelagem conceitual. Mesmo as técnicas de modelagem orientada a objetos,

que têm surgido nos *últimos anos, como a UML, baseiam-se nos conceitos da abordagem ER.*

4.1- ENTIDADE

O conceito fundamental da abordagem ER é o conceito de *entidade*.

ENTIDADE
=
Conjunto de objetos da realidade modelada sobre os quais deseja-se manter informações no banco de dados

Uma entidade representa um conjunto de objetos da realidade modelada. Como o objetivo de um modelo ER é modelar de forma abstrata um banco de dados, interessam-nos os objetos sobre os quais deseja-se manter informações. A entidade pode representar objetos concretos da realidade (uma pessoa, um automóvel, etc) quanto objetos abstratos (um departamento, um endereço, etc).

Em um DER, uma entidade é representada através de um retângulo que contêm o nome da entidade. Alguns exemplos são mostrados abaixo:

Como dito acima, cada retângulo, cada entidade representa um conjunto de objetos sobre os quais deseja-se guardar informações. Assim, no exemplo acima, o primeiro retângulo designa o conjunto de todas as pessoas sobre as quais se deseja manter informações do banco de dados, enquanto o segundo retângulo designa o conjunto de todos os departamentos sobre os quais se deseja manter informações. Caso seja necessário referir um objeto particular (uma determinada pessoa ou um determinado departamento) fala-se em *ocorrência de entidade*. Mas recentemente, por influência da programação orientada a objetos, usa-se “instância”.

Da forma como está apresentado, o modelo do exemplo acima indica apenas quais os conjuntos de objetos sobre os quais deseja-se manter informações, mas não quais as informações que devem ser mantidas para cada objeto. Estas informações são definidas pelas propriedades das entidades, dadas pelos relacionamentos, atributos e generalizações/especializações (este último só veremos na disciplina Modelagem de Dados 2).

4.2- RELACIONAMENTO

4.2.1- CONCEITUAÇÃO

Uma das propriedades sobre as quais pode ser desejável informações é a *associação* entre objetos. Exemplificando, pode ser desejável saber quais pessoas estão associadas a quais departamentos em uma organização. A propriedade de entidade que específica as associações entre objetos é o relacionamento.

RELACIONAMENTO
=
Conjunto de associações entre ocorrências de entidades

Em um DER, um relacionamento é representado através de um losângulo, ligado por linhas aos retângulos representativos das entidades que participam do relacionamento. O exemplo abaixo, apresenta um DER contendo duas entidades, PESSOA e DEPARTAMENTO, e um relacionamento, LOTAÇÃO.

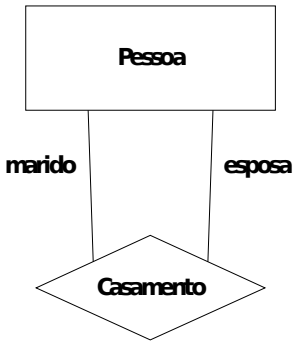
Este modelo expressa que o bando de dados mantém informações sobre:

- Um conjunto de objetos classificados como pessoas (entidade PESSOA);
- Um conjunto de objetos classificados como departamento (entidade DEPARTAMENTO); e,
- Um conjunto de associações, cada uma ligando um departamento a uma pessoa (relacionamento LOTAÇÃO).

Da mesma forma que fizemos com entidades, quando quisermos nos referir a *ocorrências ou instâncias de relacionamentos*. No caso do relacionamento LOTAÇÃO, uma ocorrência da entidade PESSOA e por uma determinada ocorrência da entidade DEPARTAMENTO.

Não necessariamente um relacionamento associa entidades diferentes. O exemplo abaixo mostra um DER que contém, um *auto-relacionamento*, devido a isto, é necessário um conceito adicional, o de *papel* da entidade no relacionamento.

PAPEL DA ENTIDADE EM RELACIONAMENTO
=
Função que uma instância da entidade cumpre dentro de uma instância do relacionamento



No caso do relacionamento de casamento, uma ocorrência de pessoa exerce o papel de marido e a outra ocorrência de pessoa exerce o papel de esposa. Papéis são anotados no DER como mostrado no exemplo anterior. No caso de relacionamentos entre entidades diferentes, no caso de LOTAÇÂO, não é necessário indicar papéis das entidades, já que eles são óbvios.

4.2.2- CARDINALIDADE DE RELACIONAMENTOS

Para fins de projeto de banco de dados, uma propriedade importante de um relacionamento é a quantas ocorrências de uma entidade podem estar associadas a uma determinada ocorrência através do relacionamento. Esta propriedade é chamada de *cardinalidade* de uma entidade em um relacionamento. Há duas cardinalidades a considerar: a cardinalidade *máxima* e a cardinalidade *mínima*.

Cardinalidade (mínima,máxima) de entidade em relacionamento
=
Número(mínimo,máximo) de ocorrências de entidade associadas a uma ocorrência da entidade

4.2.3- CARDINALIDADE MÁXIMA

Para exemplificar o conceito de cardinalidade, vamos considerar as cardinalidades máximas do exemplo abaixo:

- Entidade EMPREGADO tem cardinalidade, máxima *1* no relacionamento LOTAÇÂO. Isso significa que uma ocorrência de EMPREGADO pode estar associado a no máximo uma ocorrência de DEPARTAMENTO ou, em outros termos, que um empregado pode estar lotado em no máximo um departamento.

- Entidade DEPARTAMENTO tem cardinalidade máxima 120 no relacionamento LOTAÇÂO. Isso significa que uma ocorrência de DEPARTAMENTO pode estar associado a no máximo 120 ocorrências de EMPREGADO ou, em outros termos, que um departamento pode ter nele lotado no máximo 120 empregados.

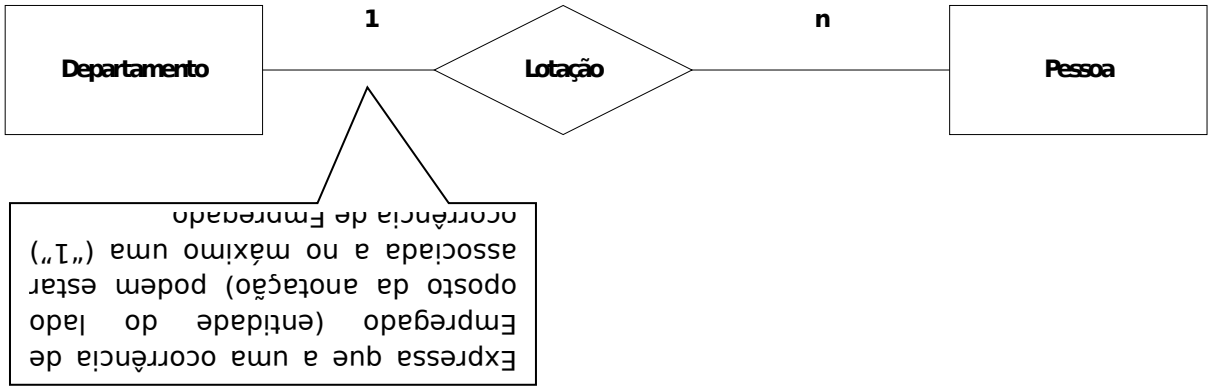
Para o projeto de banco de dados, especialmente de banco de dados relacionais, não é necessário distinguir entre diferentes cardinalidades máximas maiores que um. Por este motivo, apenas duas cardinalidades máximas são geralmente consideradas:

- A cardinalidade máxima um (1) e;
- A cardinalidade máxima ilimitada, usualmente chamada de cardinalidade máxima “muitos” e referida pela letra n.

Assim no exemplo acima, diz-se que a cardinalidade máxima da entidade DEPARTAMENTO no relacionamento LOTAÇÃO é n.

Em um DER, a cardinalidade máxima é representada conforme o exemplo anterior modificado. Observe a convenção usada. À primeira vista, ela pode parecer pouco natural, já que a cardinalidade vai anotada ”do outro lado” do relacionamento ao qual se refere. Exemplificando, a cardinalidade máxima da entidade EMPREGADO no relacionamento lotação é anotada junto ao símbolo da entidade DEPARTAMENTO.

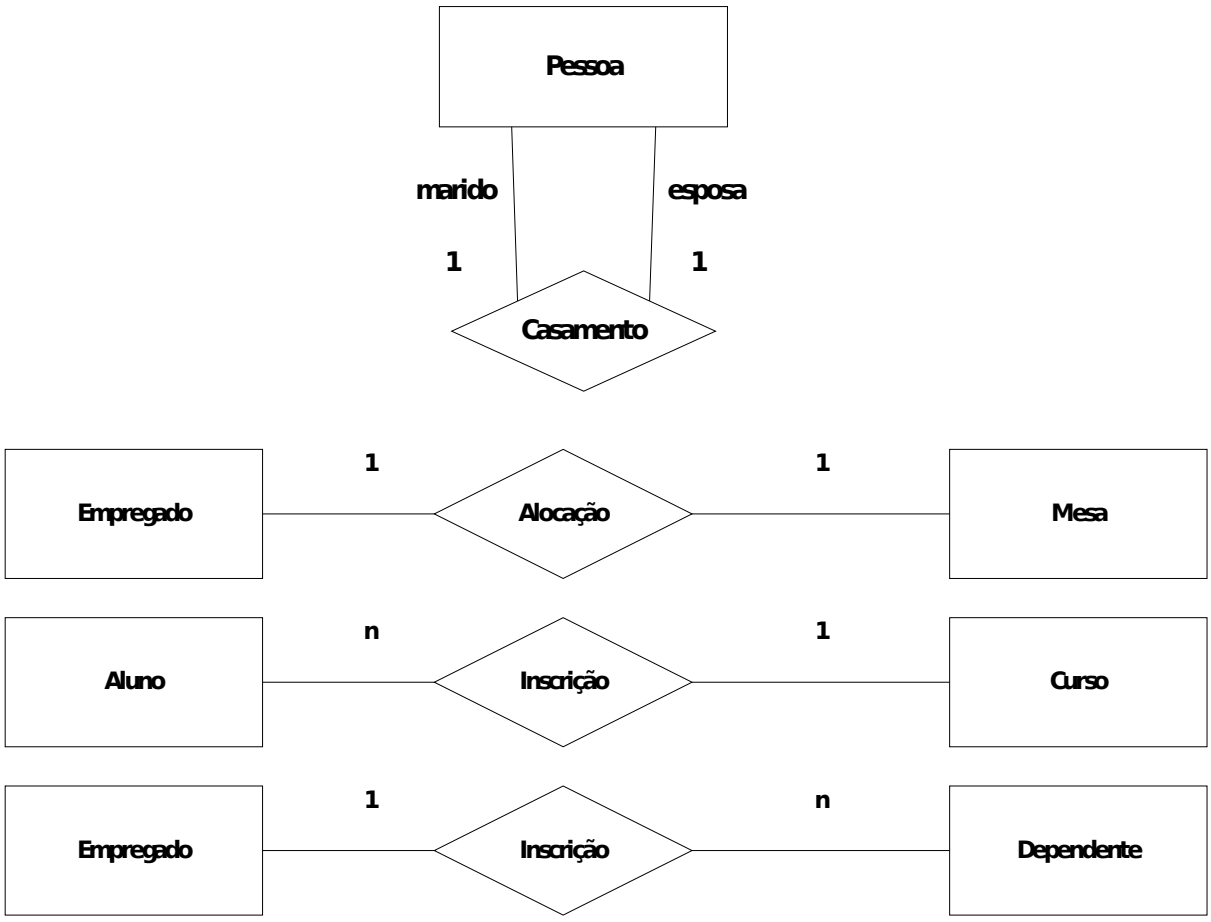
Expressa que a uma ocorrência de Departamento (entidade do lado oposto da anotação) podem estar associadas muitas (“n”) ocorrências de Empregado

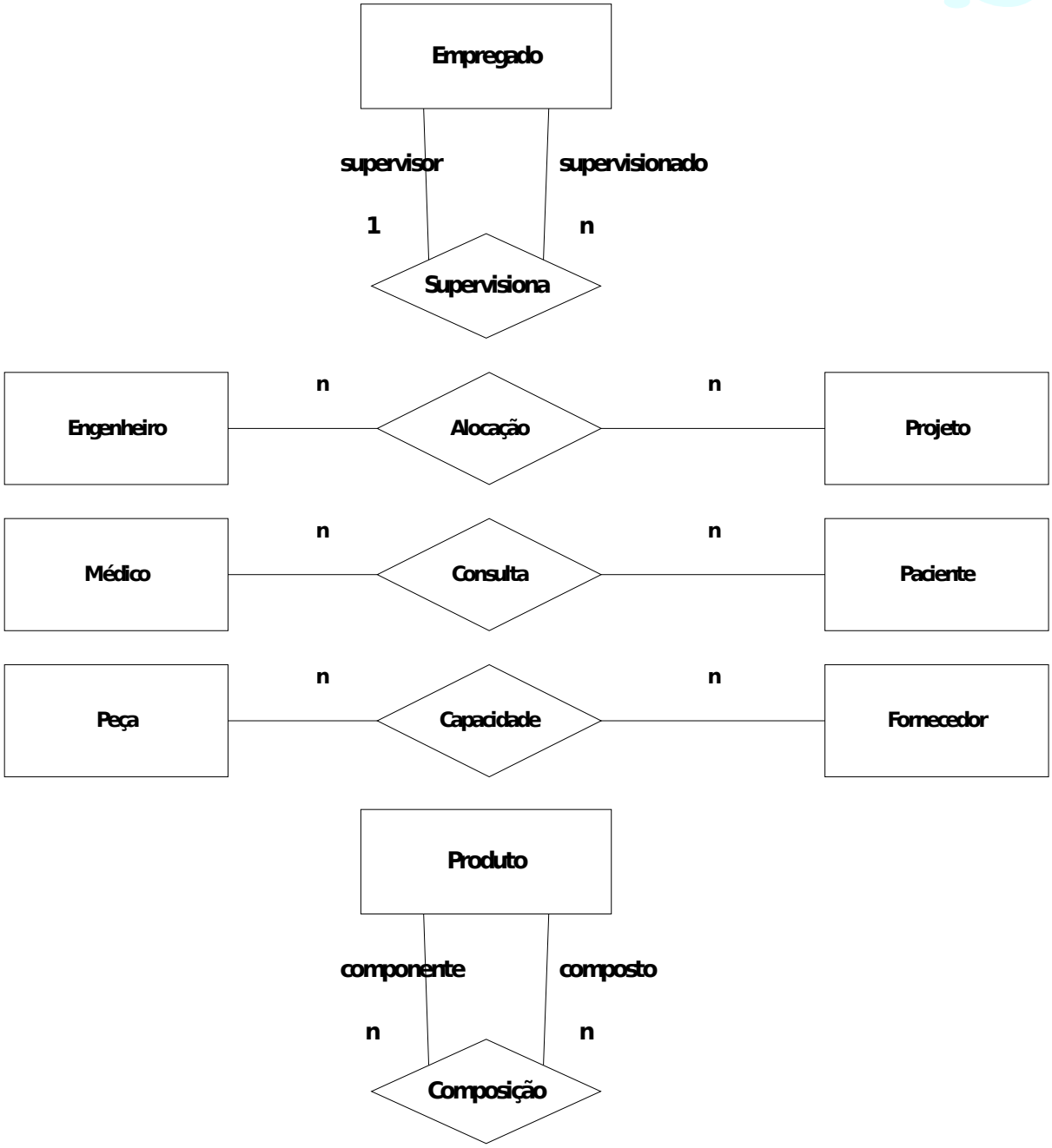


Classificação de Relacionamentos Binários

A cardinalidade máxima pode ser usada para classificar relacionamentos binários. Uma relacionamento binário é aquele cujas ocorrências contêm duas ocorrências de entidade como todos vistos até aqui. Podemos classificar os relacionamentos binários em n:n, 1:n e 1:1.

A seguir apresentamos alguns exemplos de relacionamentos binários.





4.2.5- CARDINALIDADE MÍNIMA

Além da cardinalidade máxima, outra informação que pode ser representada pó um modelo ER é o número mínimo de ocorrências de entidades associadas a uma ocorrência de uma entidade através de um relacionamento. Para fins de projeto de banco de dados, consideram-se apenas duas cardinalidades mínimas: a cardinalidade mínima 0 e a cardinalidade mínima 1.

A cardinalidade mínima 1 também recebe a denominação de "*associação obrigatória*", já que ela indica que o relacionamento deve obrigatoriamente associar uma ocorrência da entidade em questão. Com base na mesma linha da raciocínio, a cardinalidade mínima, o recebe a denominação "*associação opcional*".

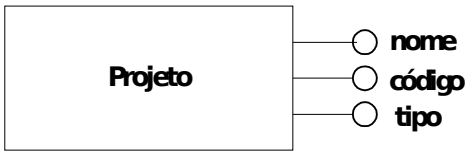
A cardinalidade mínima é anotada no diagrama junto a cardinalidade máxima, conforme o exemplo abaixo. Nesta figura, aparece novamente o exemplo da alocação de empregados a mesas. Aqui, a cardinalidade mínima é usada para especificar que cada empregado deve ter a ele alocada obrigatoriamente uma mesa (cardinalidade mínima 1) e que uma mesa pode existir sem que a ela esteja alocado um empregado (cardinalidade 0).

4.4- ATRIBUTO

O modelo ER permite a especificação de propriedades de entidades. Uma propriedade é participar de um relacionamento. Outra propriedade é ter um atributo. O conceito de atributo serve para associar informações a ocorrências de entidades ou de relacionamentos.

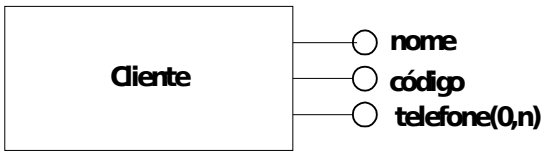
ATRIBUTO
=
Dado que associado a cada ocorrência de uma entidade ou de um relacionamento

Atributos são representados graficamente conforme o exemplo abaixo. A figura expressa que cada ocorrência de PROJETO tem associado exatamente um nome, um código e um tipo.



Na prática, muitas vezes os atributos não são representados graficamente, para não sobrecarregar os diagramas, já que entidades podem possuir um grande número de atributos. Prefere-se usar uma representação textual que aparece separadamente do diagrama ER.

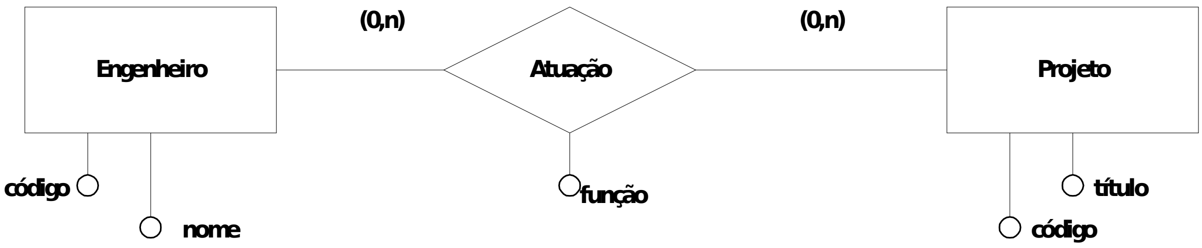
Um atributo pode possuir uma cardinalidade, de maneira análoga, a uma entidade em um relacionamento. A cardinalidade de um atributo define quantos valores deste atributo podem estar associados a uma ocorrência de entidade/relacionamento a qual ele pertence. A representação gráfica da cardinalidade de atributos é derivada da representação da cardinalidade de entidades em relacionamentos, conforme o exemplo abaixo.



No caso de a cardinalidade ser (1,1), ela pode ser omitida do diagrama. Assim, o exemplo acima expressa que nome e código são atributos *obrigatórios* (cardinalidade mínima 1 – cada entidade possui no mínimo um valor associado) e *monovalorados*

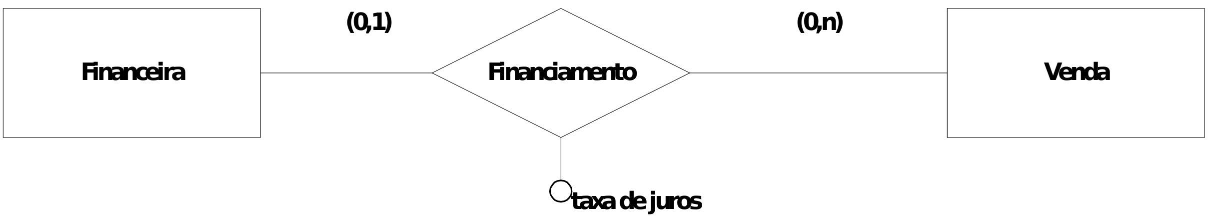
(cardinalidade máxima 1 – cada entidade possui no máximo um valor associado). Já o atributo telefone, é um atributo *opcional* (cardinalidade mínima 0) e *multivalorado* (cardinalidade máxima n).

Assim como entidades também relacionamentos podem possuir atributos. O exemplo abaixo, mostra um DER no qual um relacionamento, ATUAÇÂO, possui um atributo, a função que um engenheiro exerce dentro de um projeto.



Esta função não pode ser considerada atributo de ENGENHEIRO, já que um engenheiro pode atuar em diversos projetos, exercendo diferentes funções. Também, não é atributo de PROJETO, já que, em um projeto, podem atuar diversos engenheiros com funções diferentes.

Outro exemplo de atributo em relacionamento, agora em um relacionamento 1:n, é mostrado abaixo. Este diagrama modela vendas a prazo que são relacionadas a uma financeira, através do relacionamento FINANCIAMENTO. Os atributos número de parcelas e taxa de juros são atributos do relacionamento. Estes dois atributos poderiam ter sido incluídos na entidade VENDA. Neste caso, seriam atributos opcionais, já que nem toda venda é a prazo e possui estes atributos. Assim, preferiu-se usar o modelo do exemplo, exatamente para explicar o fato de os atributos número de parcelas e taxa de juros pertencerem somente a vendas a prazo.

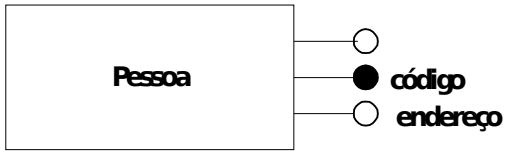


4.4.1- IDENTIFICANDO ENTIDADES

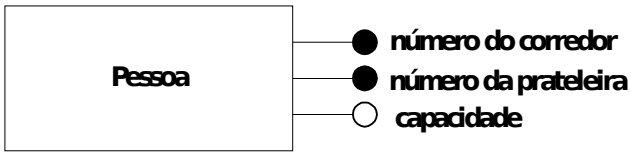
Cada entidade deve possuir um identificador.

Identificador de entidade
=
Conjunto de um ou mais atributos e relacionamentos cujos valores servem para distinguir uma ocorrência da entidade das demais ocorrências da mesma entidade

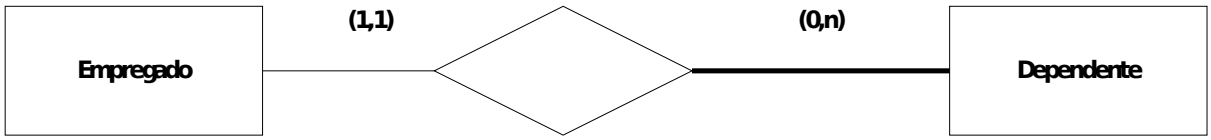
O … uto como identifica… ulo preto. No exemplo abaixo, o atributo *código* é identificador. Isso significa que cada pessoa possui um código diferente. Já os atributos *nome* e *endereço* não são identificadores – o mesmo nome (ou o mesmo endereço) pode ser associado a pessoas diferentes.



O exemplo abaixo apresenta o identificador da entidade composto por dois atributos. Considera-se um almoxarifado de uma empresa de ferragens organizado como segue. Os produtos ficam armazenados em prateleiras. Estas prateleiras encontram-se em armários organizados em corredores. Os corredores são numerados sequencialmente a partir de um e as prateleiras são numeradas sequencialmente a partir de um, dentro de um corredor. Assim, para identificar uma prateleira é necessário conhecer seu número e o número do corredor em que se encontra. Para cada prateleira deseja-se saber sua capacidade em metros cúbicos.

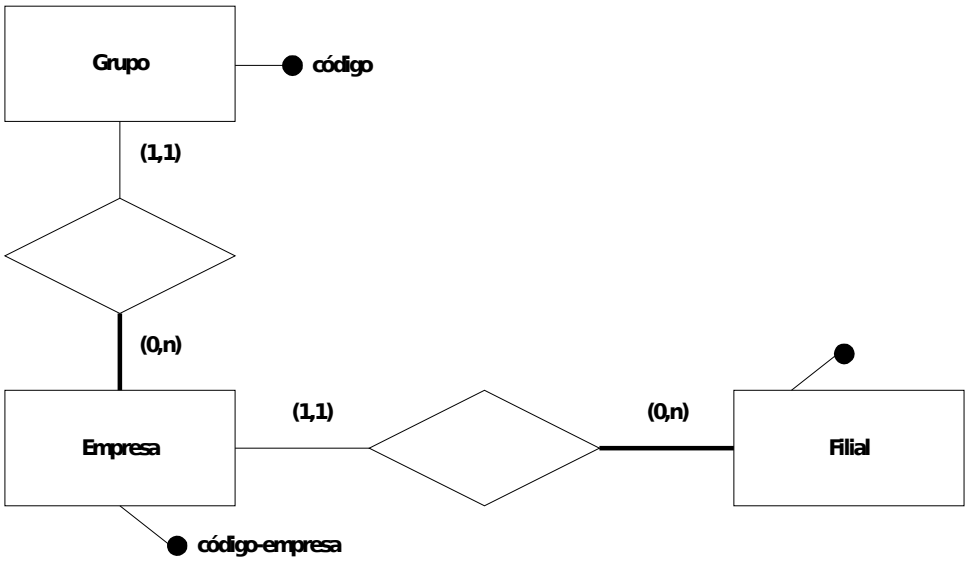


Finalmente, há casos em que o identificador de uma entidade é composto não somente por atributos da própria entidade, mas também por relacionamentos dos quais a entidade participa (relacionamento identificador). Um exemplo deste caso é mostrado abaixo. Este modelo envolve empregados de uma organização, relacionados com seus dependentes para fins de imposto de renda. Cada dependente está relacionado a exatamente um empregado. Um dependente é identificado pelo empregado ao qual ele está relacionado e por um número de sequência que distingue os diferentes dependentes de um mesmo empregado. No DER, o relacionamento usado como identificador é indicado por uma linha mais densa.



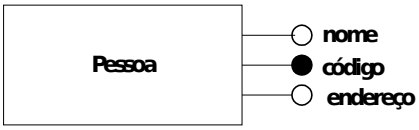
Nesse caso, alguns autores dizem que a entidade DEPENDENTE é uma *entidade fraca*. O termo “fraca” deriva do fato de a entidade somente existir quando relacionada a outra entidade e de usar, como parte de seu indicador, entidades relacionadas.

Outro exemplo de relacionamento identificador é apresentado no exemplo abaixo, que contém um fragmento de um DER sobre empresas. No exemplo, é representada a divisão de grupos de empresas em empresas e de empresas em filiais de empresas. Para identificar um grupo de empresas é usando um código. Já uma empresa é identificada pelo grupo ao qual está relacionada e por um número da empresa dentro do grupo. Finalmente, uma filial é identificada pela empresa a qual está vinculada e por um número de filial dentro da empresa.

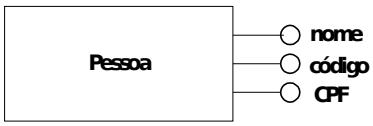


O identificador de uma entidade, seja ele simples, composto por diversos atributos ou composto por identificadores externos, deve obedecer a duas propriedades:

- O identificador dever *mínimo*. Isso significa que o identificador de uma entidade deve ser composto de tal forma que retirando um dos atributos ou relacionamentos que o compões, ele deixa de ser identificador na entidade PESSOA, do exemplo abaixo, o par código e nome poderia ser usado para distinguir uma ocorrência de PESSOA das demais. Entretanto, estes atributos não formam um identificador mínimo, já que código é suficiente para distinguir as ocorrências de PESSOA.



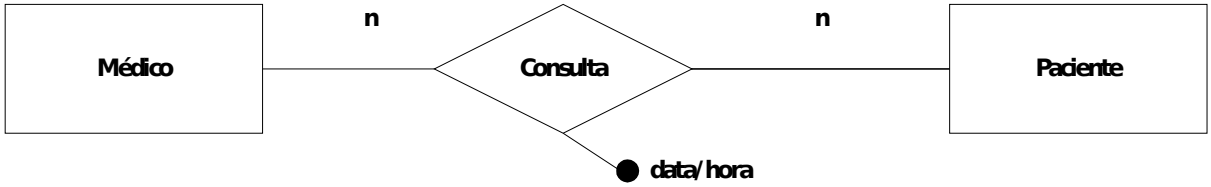
- Para fins de projeto de bando de dados relacional, cada entidade deve possuir um *único* identificador. Em alguns casos, diferentes conjuntos de atributos podem servir para distinguir as ocorrências da entidade. Exemplificando, a entidade EMPREGADO no exemplo abaixo poderia possuir como identificador tanto o atributo código, quanto o atributo CPF. Cabe ao modelador decidir qual dos dois atributos será usado como identificador na entidade.



4.4.2- IDENTIFICANDO RELACIONAMENTO

Em princípio, uma ocorrência de relacionamento diferencia-se das demais ocorrências do mesmo relacionamento pelas ocorrências de entidades que dela participam. Exemplificando, uma ocorrência de ALOCAÇÃO (exemplo abaixo) é identificada pela ocorrência de ENGENHEIRO e pela ocorrência de PROJETO que ela relaciona. Em outros termos, para cada par (engenheiro, projeto) há no máximo um relacionamento de alocação.

Entretanto, há casos nos quais entre as mesmas ocorrências de entidade podem existir diversas ocorrências de relacionamentos. Um exemplo é o relacionamento CONSULTA entre as entidades MÉDICO e PACIENTE, abaixo. Para um determinado médico e um determinado paciente pode haver diversas consultas. Neste caso, é necessário algo que distinga uma consulta entre um médico e seu paciente das demais consultas entre este médico e este paciente. A diferenciação ocorre através de *atributos identificadores de relacionamentos*. No caso do relacionamento pode ser data/hora.



Assim, um relacionamento é identificado pelas entidades dele participantes, bem como pelos atributos identificadores eventualmente definidos.

BIBLIOGRAFIA

Date, C. J. **Introdução a Sistemas de Banco de Dados**. Editora Campus, 8 ª edição, 2004, RJ
HEUSER, Carlos Alberto. **Projeto de Banco de Dados**. Editora Bookman, 6 ª edição, 2009, SP.
MACHADO, Felipe Nery Rodrigues. **Banco de Dados: Projeto e Implementação**. Editora Érica, 2 ª edição, 2008, SP.
MEDEIROS, Marcelo. **Banco de Dados para Sistemas de Informação**. Editora Visual Books, 1ª edição, 2006, SC.
MECENAS, Ivan & OLIVEIRA, Viviane de. **Banco de Dados: do modelo conceitual à implementação**. Editora Alta Books, 1ª edição, 2005, RJ.