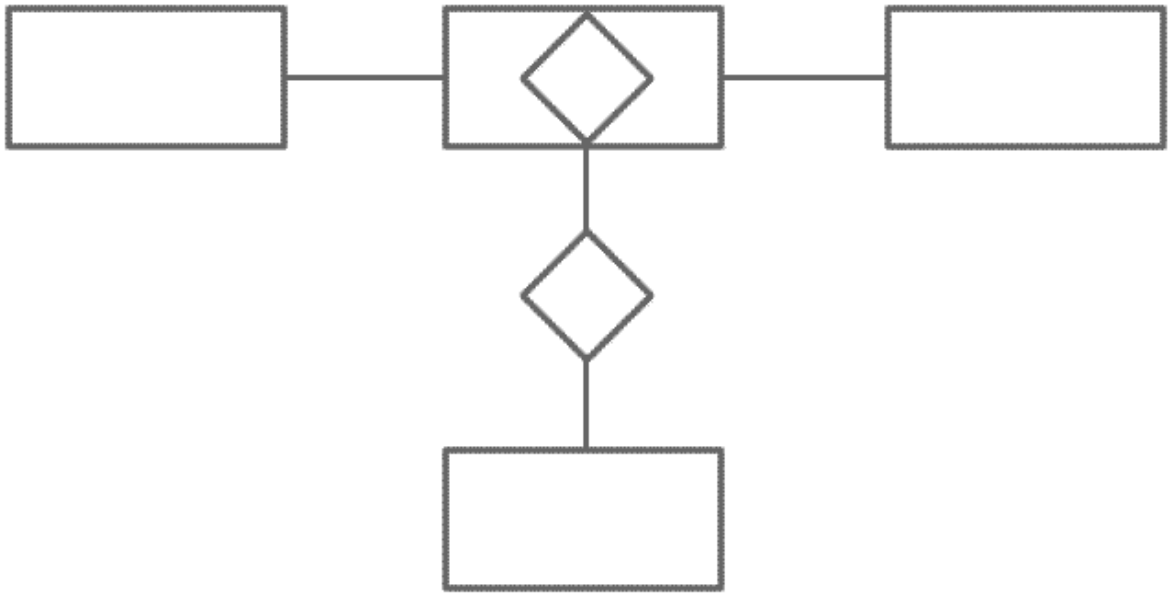
MODELAGEM EM BANCO DE DADOS



Prof. Marcos Alexandruk

INTRODUÇÃO

A humanidade sempre procurou manter registros históricos dos eventos mais importantes para que pudessem ser utilizados posteriormente.

Exemplos: Pinturas em cavernas, inscrições hieroglíficas, escritas cuneiformes e a imprensa (a partir do século XV).

Os computadores inventados e aperfeiçoados a partir do século XX, permitiram que os dados fossem armazenados e recuperados com grande rapidez e facilidade.

No início da década de 70 surgiram os SGBDs (Sistemas de Gerenciamento de Banco de Dados).

Pesquisas na área resultaram em um conjunto de técnicas, processos e notações para a modelagem ou projeto de banco de dados.

Observe, a seguir, alguns conceitos importantes:

INFORMAÇÃO:

- Acrescenta algo ao conhecimento da realidade a ser analisada. Por exemplo, a dosagem que um paciente precisa receber de um determinado remédio é uma INFORMAÇÃO. Este conhecimento pode ser (ou não) modelado (registrado).

DADO:

- É uma representação, um registro de uma informação. Este DADO pode ser registrado fisicamente através de um papel (receita médica), um disco magnético, etc.

BANCO DE DADOS:

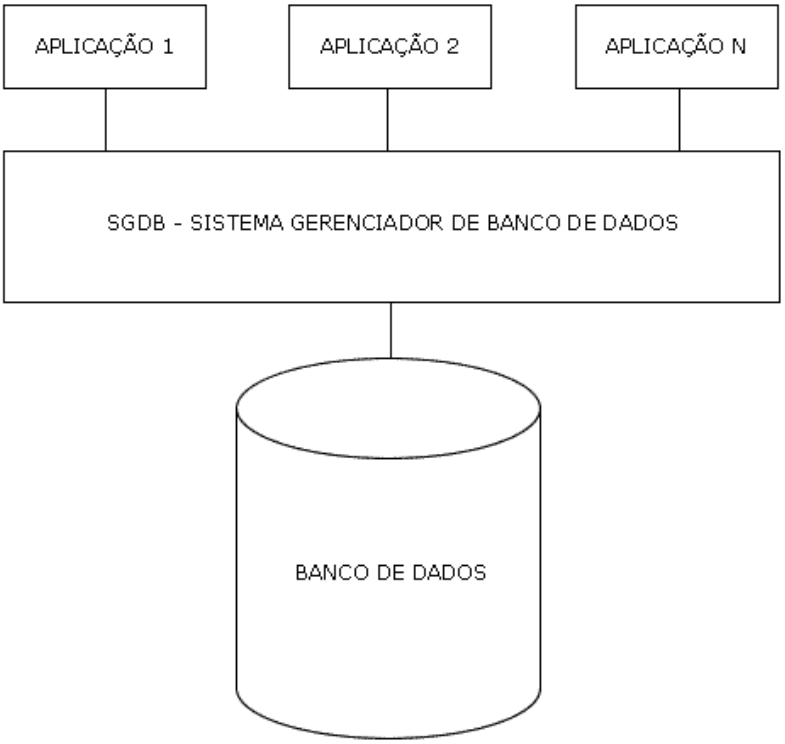
- coleção de dados integrados que tem por objetivo atender a uma comunidade de usuários.
- conjunto de dados persistentes e manipuláveis que obedecem a um padrão de armazemamento. Exemplos: *lista telefônica, dicionário, etc*.
- equivalente eletrônico de um armário de arquivamento, isto é, um repositório para uma coleção de arquivos de dados computadorizados.

SISTEMA DE GERÊNCIAMENTO DE BANCO DE DADOS (SGBD)

- software que incorpora as funções de definição, recuperação e alteração de dados em um banco de dados

Por que utilizar bancos de dados informatizados?

- Compacto (elimina arquivos de papéis);
- Rápido;
- Integrado (vários aplicativos utilizam o mesmo repositório de dados);
- Compartilhado (vários usuários podem acessar);
- Seguro (controle de acesso);
- Padronizado;
- Consistente;
- Suporte a transações.



MODELO DE DADOS

Descrição formal da estrutura de um banco de dados.

- MODELO HIERÁRQUICO

Organiza os dados de cima para baixo, como uma árvore. Cada registro é dividido em partes de registros denominados segmentos. O banco de dados se assemelha a um organograma com um segmento raiz e um número qualquer de segmentos subordinados. Os segmentos, por sua vez, são arranjados em estruturas multiniveladas, com um segmento superior ligado a um segmento subordinado em um relacionamento "pai-filho". Um segmento "pai" pode ter mais de um "filho", mas um segmento "filho" só pode ter um "pai".

- MODELO EM REDE

Os registros são organizados no banco de dados por um conjunto arbitrário de gráficos. Em outras palavras, um "filho" pode ter mais de um "pai". São mais flexíveis que os hierárquicos. Existem limitações práticas para o número de ligações, que podem ser estabelecidas entre os registros. Nem o modelo de gerenciamento de bancos de dados em rede nem o hierárquico podem criar facilmente novos relacionamentos entre os elementos de dados ou novos padrões de acesso sem grande esforço de programação.

- MODELO RELACIONAL

A introdução do sistema relacional (1970) foi o evento mais importante na história da área de banco de dados.

De modo abreviado (e informal), um sistema relacional é aquele no qual:

- Os dados são percebidos pelo usuário como tabelas
- Os operadores à disposição do usuário (por exemplo, para busca de dados) são operadores que geram tabelas "novas" a partir de tabelas "antigas". Por exemplo, há um operador de "restrição" (também conhecido como "seleção") para extrair um subconjunto das linhas de uma dada tabela, e outro operador, "projeção", que extrai um subconjunto das colunas.

Os primeiros produtos relacionais começaram a aparecer no final da década de 1970. Hoje a maioria dos sistemas de banco de dados é relacional:

- IBM: DB2
- Microsoft: SQL Server
- Oracle: 9i, 10g, 11g
- Sybase
- Informix
- MySQL
- Postgre

A linguagem de manipulação de dados em sistemas de bancos de dados relacionais é o SQL (Structured Query Language).

VISÃO DE DADOS

O maior benefício de um banco de dados é proporcionar ao usuário uma *visão abstrata* dos dados. Isto é, o sistema acaba por ocultar determinados detalhes sobre a forma de armazenamento e manutenção desses dados.

ABSTRAÇÃO DE DADOS

NÍVEL FÍSICO:

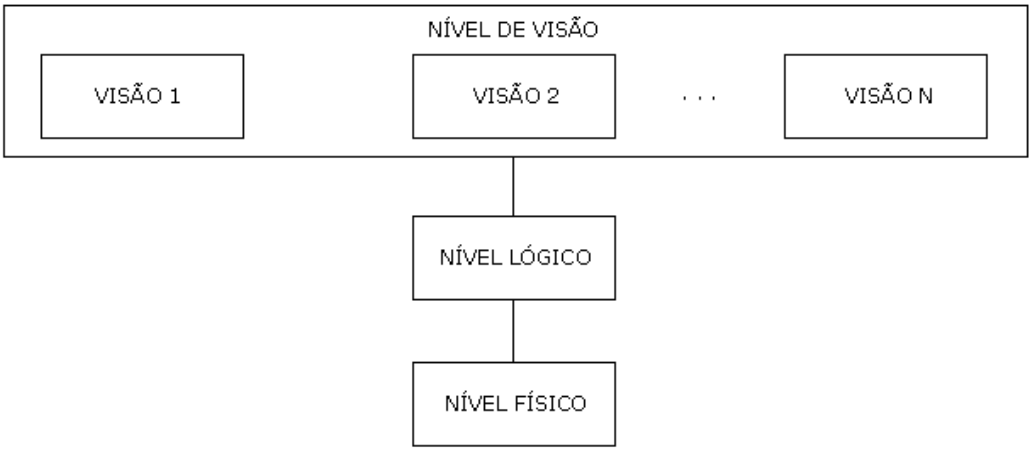
- É o nível mais baixo de abstração.
- Descreve *como* os dados estão de fato armazenados.

NÍVEL LÓGICO:

- Descreve *quais* dados estão armazenados e quais os relacionamentos entre eles.
- Utilizado pelos administradores de banco de dados que precisam decidir quais informações devem pertencer ao banco de dados.

NÍVEL DE VISÃO:

- É o nível mais alto de abstração.
- Descreve apenas parte do banco de dados.
- Muitos usuários utilizam apenas parte do banco de dados. Assim, para que estas interações sejam simplificadas, um nível de visão é definido.
- O sistema pode proporcionar diversas visões do mesmo banco de dados.



Os três níveis de abstração de dados

INDEPENDÊNCIA DE DADOS:

- É a capacidade de modificar a definição dos esquemas em determinado nível, sem afetar o esquema do nível superior:
- **Independência de dados física:** é a capacidade de modificar o esquema físico sem que, com isso, qualquer programa ou aplicação precise ser reescrito.
- **Independência de dados lógica:** é a capacidade de modificar o esquema lógico sem que, com isso, qualquer programa ou aplicação precise ser reescrito.
- A independência lógica é mais difícil de ser alcançada que a independência física, uma vez que as aplicações são mais fortemente dependentes da estrutura lógica dos dados do que de seu acesso.

ARQUITETURAS DE SISTEMAS DE BANCOS DE DADOS

SISTEMAS CENTRALIZADOS

Sistemas de banco de dados centralizados são aqueles executados sobre um único sistema computacional que não interagem com outros sistemas.
Tais sistemas podem ter a envergadura de um sistema de banco de dados de um só usuário executado em um computador pessoal até sistemas de alto desempenho em sistemas de grande porte.

SISTEMAS CLIENTE-SERVIDOR

As funcionalidades de um banco de dados podem ser superficialmente divididas em duas categorias:
- back-end: gerencia as estruturas de acesso, desenvolvimento e otimização de consultas, controle de concorrência e recuperação.
- front-end: consiste em ferramentas como formulários, gerador de relatórios e recursos de interfaces gráficas.

A interface entre o front-end e o back-end é feita por meio da SQL ou de um programa de aplicação.

SISTEMAS PARALELOS

Sistemas paralelos imprimem velocidade ao processamento e à I/O por meio do uso em paralelo de diversas CPUs e discos.
Suprem a demanda de aplicações que geram consultas em bancos de dados muito grandes (da ordem de terabytes) ou que tenham de processar um volume enorme de transações por segundo (da ordem de milhares de transações por segundo).

Há diversos modelos arquitetônicos para máquinas paralelas:
- Memória compartilhada: Todos os processadores compartilham uma mesma memória.
- Disco compartilhado: Todos os processadores compartilham o mesmo disco. Sistemas com discos compartilhados são, às vezes chamados de *clusters*.
- Ausência de compartilhamento: Os processadores não compartilham nem memória nem disco.
- Hierárquico: É um híbrido das arquiteturas anteriores.

SISTEMAS DISTRIBUÍDOS

Em um sistema de banco de dados distribuído o banco de dados é armazenado em diversos computadores. Os computadores comunicam-se uns com os outros por intermédio de redes de alta velocidade ou linhas telefônicas. Eles não compartilham memória principal ou discos.
Os computadores em um sistema de banco de dados distribuídos podem variar, quanto ao tamanho e funções, desde estações de trabalho até sistemas de grande porte (mainframe).

VISÃO GERAL DA ESTRUTURA DO SISTEMA DE BANCO DE DADOS

COMPONENTES DE PROCESSAMENTO DE CONSULTAS:

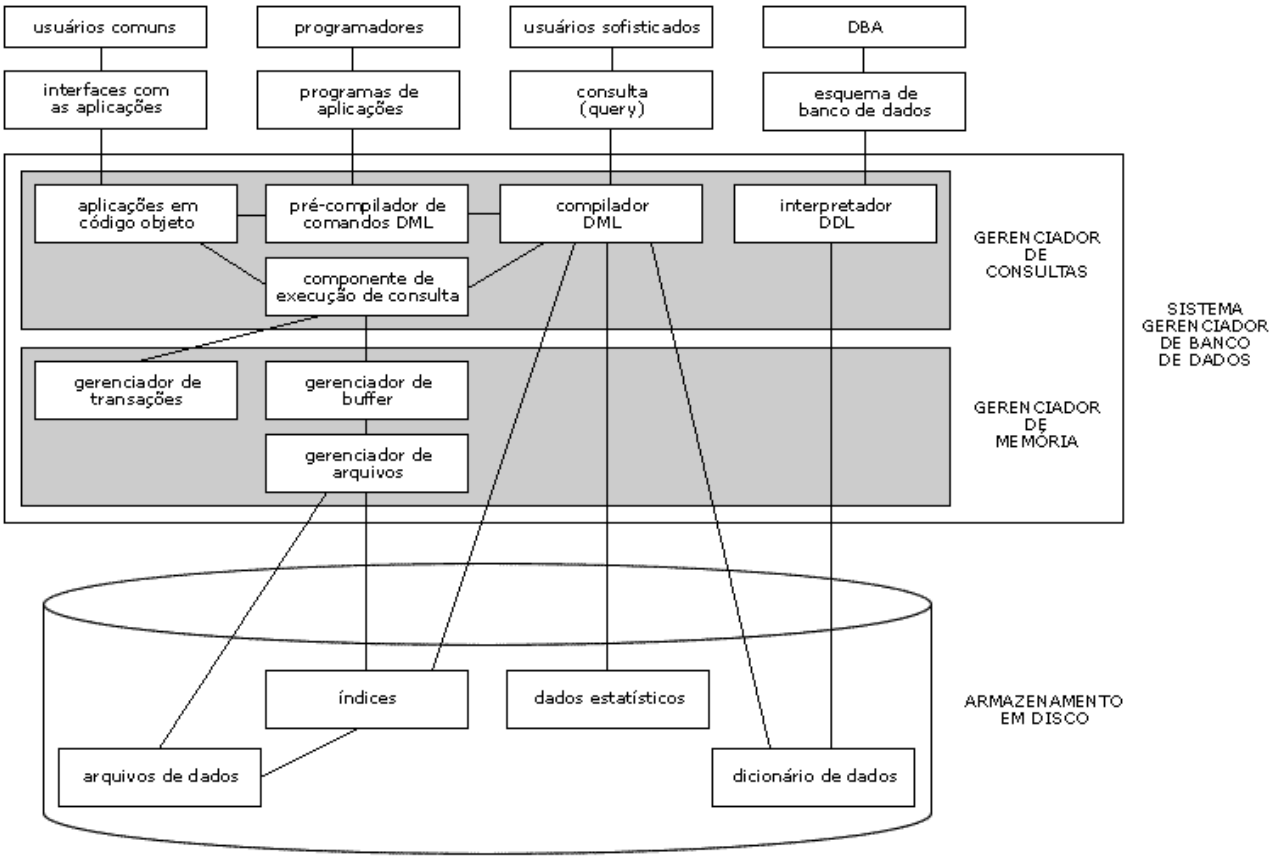
- **Compilador DML**: Traduz comandos DML da linguagem de consulta em instruções de baixo nível, inteligíveis ao componente de execução de consultas.
- **Pré-compilador para comandos DML**: Inseridos em programas de aplicação que convertem comandos DML em chamadas e procedimentos normais da linguagem hospedeira. Interage com o compilador DML de modo a gerar o código apropriado.
- **Interpretador DDL**: Interpreta os comandos DDL e registra-os em um conjunto de tabelas que contêm *metadados*.
- **Componentes para tratamento de consultas**: Executam instruções de baixo nível geradas pelo compilador DML.

COMPONENTES DE ADMINISTRAÇÃO DE ARMAZENAMENTO DE DADOS:

- **Gerenciamento de autorização e integridade**: Testam o cumprimento das regras de integridade e a permissão ao usuário no acesso ao dado.
- **Gerenciamento de transações**: Garante que o banco de dados permanecerá em estado consistente (correto) a despeito de falhas no sistema e que transações concorrentes serão executadas sem conflitos em seus procedimentos.
- **Administração de arquivos**: Gerencia a alocação de espaço no armazenamento em disco e as estruturas de dados usadas para representar as informações armazenadas em disco.
- **Administração de buffer**: Responsável pela intermediação de dados do disco para a memória principal e pela decisão de quais dados colocar em memória cache.

OUTRAS ESTRUTURAS NECESSÁRIAS COMO PARTE DA IMPLEMENTAÇÃO FÍSICA DO SISTEMA:

- **Arquivo de dados**: Armazena o próprio banco de dados.
- **Dicionário de dados**: Armazena os metadados relativos à estrutura do banco de dados.
- **Índices**: Proporcionam acesso rápido aos dados associados a valores determinados.
- **Estatísticas de dados**: Armazenam informações estatísticas usadas pelo processador de consultas para seleção de meios eficientes para execução de uma consulta.



MODELAGEM DE BANCO DE DADOS

O projejo de um banco de dados ocorre geralmente em três etapas:

MODELO CONCEITUAL

- É a primeira etapa do projeto de um sistema de aplicação em banco de dados.
- Representa ou descreve a realidade do ambiente do problema, constituindo-se em uma visão global dos principais dados e relacionamentos (estruturas de informações), independente das restrições de implementação.
- É uma descrição em alto nível (macro definição) mas que tem a preocupação de capturar e retratar toda a realidade de uma organização.
- O resultado de um *modelo conceitual* é um esquema que representa a realidade das informações existentes, assim como as estruturas de dados que representam estas informações.

MODELO LÓGICO

- Tem seu início a partir do modelo conceitual, levando em consideração três abordagens atualmente disponíveis: Relacional, Hierárquica e Rede.
- O modelo lógico descreve as estruturas que estarão contidas no banco de dados, mas sem considerar ainda nenhuma característica específica de SGBD, resultando em um esquema lógico de dados.

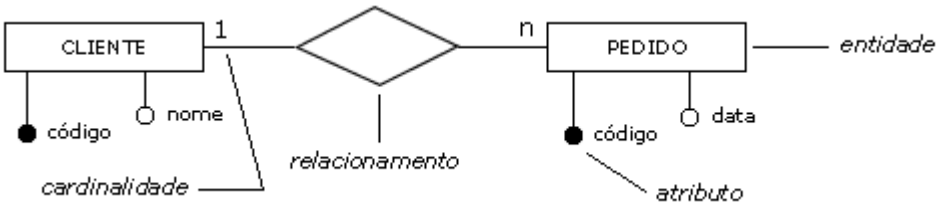
MODELO FÍSICO

- Parte do modelo lógico e descreve as estruturas físicas de armazenamento de dados, tais como: tamanhos de campos, índices, tipos de dados, nomenclaturas, etc.
- Este modelo detalha o estudo dos métodos de acesso do SGDB, para elaboração dos índices de cada informação colocada nos modelos conceitual e lógico.
- É a etapa final do projeto de banco de dados, na qual será utilizada a linguagem de definição de dados (DDL), para a realização da montagem do mesmo no nível de dicionário de dados.

MODELO CONCEITUAL

Descrição do banco de dados de forma independente de implementação em um SGBD. Registra que dados podem aparecer no banco de dados, mas não registra como estes dados estão armazenados no SGDB.

DIAGRAMA ENTIDADE-RELACIONAMENTO



O Modelo Entidade-Relacionamento foi definido por Peter Chen em 1976, e teve como base a teoria relacional criada por E. F. Codd (1970).

Um modelo ER é um modelo formal, preciso, não ambíguo. Isto significa que diferentes leitores de um mesmo modelo ER devem sempre entender exatamente o mesmo. Tanto é assim, que um modelo ER pode ser usado como entrada de uma ferramenta CASE (Computer Aided Software Engineering) na geração de um banco de dados relacional.

ENTIDADE: Representa um conjunto de objetos (tudo que é perceptível ou manipulável) da realidade modelada sobre os quais deseja-se manter informações no banco de dados.

ATRIBUTOS: Dados que são associados a cada ocorrência de uma entidade ou de um relacionamento.

IDENTIFICADOR DE ENTIDADE: Atributo ou conjunto de atributos e relacionamentos cujos valores distinguem uma ocorrência da entidade das demais.

CARDINALIDADE: Número (mínimo, máximo) de ocorrências de entidade associadas a uma ocorrência da entidade em questão através do relacionamento.

Cardinalidade mínima: É o número mínimo de ocorrências de entidade que são associadas a uma ocorrência de uma entidade através de um relacionamento.

A cardinalidade mínima 1 recebe a denominação de *associação obrigatória*, já que ela indica que o relacionamento deve obrigatoriamente associar uma ocorrência de entidade a outra. A cardinalidade mínima 0 (zero) recebe a denominação de *associação opcional*.

Cardinalidade máxima: É o número máximo de ocorrências de entidade que são associadas a uma ocorrência de uma entidade através de um relacionamento. Apenas duas cardinalidades máximas são relevantes: a cardinalidade máxima 1 e a cardinalidade máxima n (muitos).

DIAGRAMA DE OCORRÊNCIAS: Para *fins didáticos*, pode ser útil construir um *diagrama de ocorrências*. Neste as ocorrências de entidades são representadas por círculos brancos e ocorrências de relacionamentos por círculos pretos. As ocorrências de entidades participantes de uma ocorrência de relacionamento são indicadas pelas linhas que ligam o círculo preto aos círculos brancos.

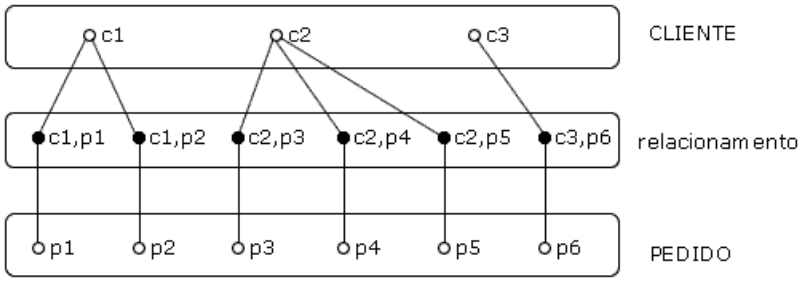


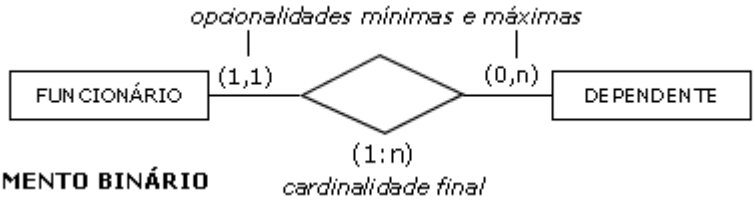
DIAGRAMA DE OCORRÊNCIAS

RELACIONAMENTO BINÁRIO

Um relacionamento binário é aquele envolve **duas** ocorrências de entidade.

Podemos classificar os relacionamentos binários em:

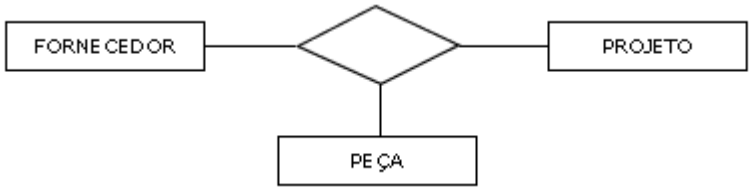
- **1:1 (um-para-um):** cada elemento de uma entidade relaciona-se com um e somente um elemento de outra entidade.
- **1:n (um-para-muitos):** um elemento da entidade 1 relaciona-se com muitos elementos da entidade 2, mas cada elemento da entidade 2 somente pode estar relacionado a um elemento da entidade 1.
- **n:n (muitos-para-muitos):** em ambos os sentidos encontramos um grau de relacionamento de um-para-muitos, o que o caracteriza no contexto geral como um relacionamento de muitos-para-muitos.



RELACIONAMENTO BINÁRIO

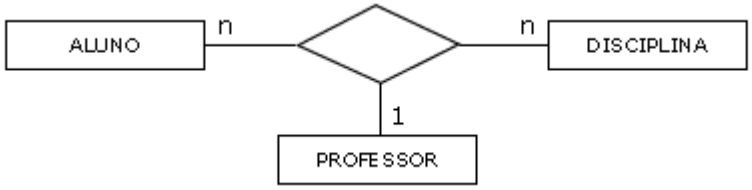
RELACIONAMENTO TERNÁRIO

Relacionamento entre múltiplas entidades, expressam um fato em que todas as entidades ocorrem simultaneamente, ou seja, todas as ocorrências do relacionamento possuem, sempre, ligações com todas as entidades envolvidas no relacionamento.
O exemplo a seguir queremos garantir que a seguinte situação seja representada de forma apropriada: **x** fornece a peça **y** para o projeto **z**. Esta condição somente pode ser representada através de um relacionamento ternário.



Observe o que ocorreria se tentássemos substituir o relacionamento ternário acima por três relacionamentos binários:
x fornece a peça **y**: Não podemos inferir que a peça **y** será utilizada no projeto **z**.
y é utilizada no projeto **z**: Não podemos inferir que **y** foi fornecida por **x**.
x fornece para o projeto **z**: Não podemos inferir **x** fornece a peça **y** para o projeto **z**.

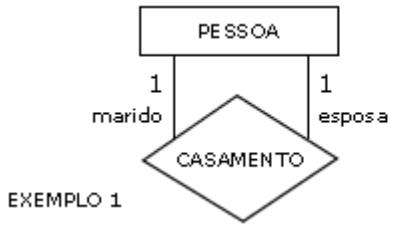
Veja este outro caso que expressa a cardinalidade num relacionamento ternário:

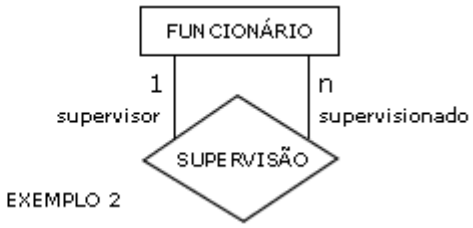


No exemplo acima, cada par de ocorrências (aluno, disciplina) está associado a no máximo um professor.
A um par (aluno, professor) podem estar associadas muitas disciplinas, ou em outros termos, um professor pode ministrar a um determinado aluno várias disciplinas.
A um par (disciplina, professor) podem estar associados muitos alunos, ou em outros termos, um professor pode uma determinada disciplina a vários alunos.

AUTO RELACIONAMENTO

Relacionamento entre ocorrências de uma mesma entidade.





Exemplo 1: **1:1** - uma ocorrência de pessoa exerce o papel de marido e outra ocorrência de pessoa exerce o papel de esposa.

Exemplo 2: **1:n** - uma ocorrência de funcionário exerce o papel de supervisor e outras ocorrências de funcionário exercem o papel de supervisionado.

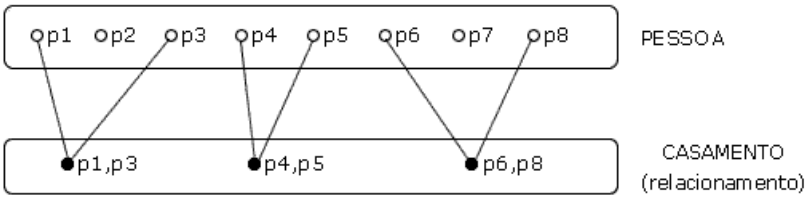
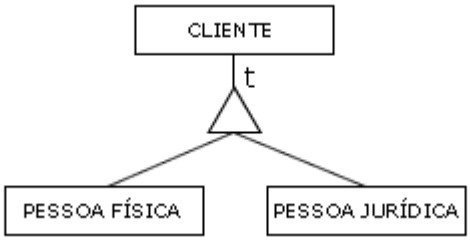


DIAGRAMA DE OCORRÊNCIAS

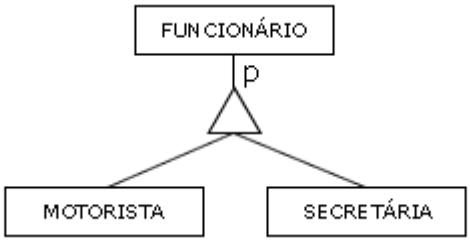
GENERALIZAÇÃO/ESPECIALIZAÇÃO

Através deste conceito é possível atribuir propriedades particulares a um subconjunto das ocorrências especializadas de uma entidade genérica.

Especialização total: para cada ocorrência da entidade genérica existe sempre uma ocorrência em uma das entidades especializadas.

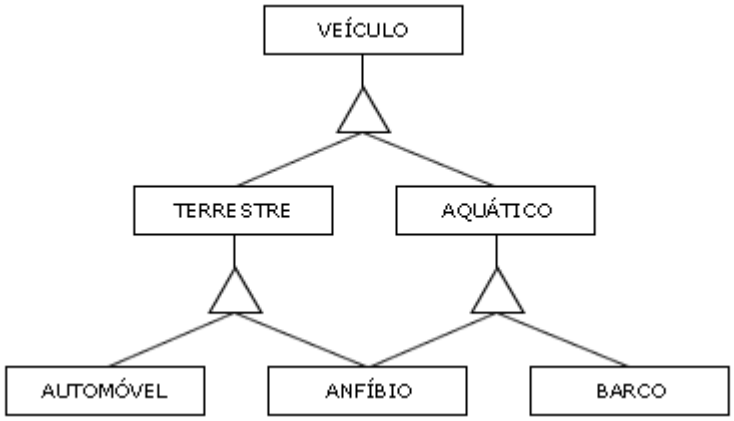


Especialização parcial: nem toda ocorrência da entidade genérica possui uma ocorrência correspondente em uma entidade especializada.



MÚLTIPLOS NÍVEIS E HERANÇA MÚLTIPLA

É admissível que uma mesma entidade seja especialização de diversas entidades genérica (herança múltipla).
No diagrama abaixo o exemplo de herança múltipla aparece na entidade ANFÍBIO.



HERANÇA DE PROPRIEDADES

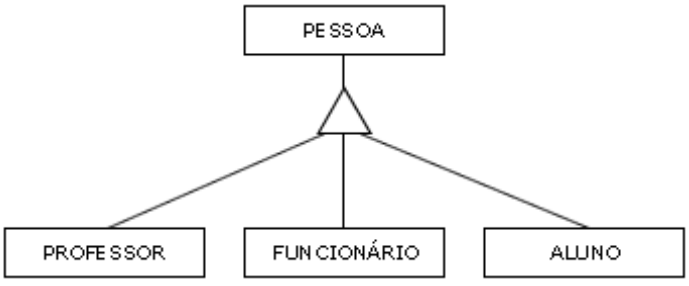
Herdar propriedades significa que cada ocorrência da entidade especializada possui, além de suas propriedades (atributos, relacionamentos e generalizações/especializações) também as propriedades da ocorrência da entidade genérica correspondente.

GENERALIZAÇÃO/ESPECIALIZAÇÃO EXCLUSIVA

Significa que uma ocorrência de entidade genérica aparece, para cada hierarquia generalização/especialização, no máximo uma vez.

GENERALIZAÇÃO/ESPECIALIZAÇÃO NÃO EXCLUSIVA

Neste caso, uma ocorrência da entidade genérica pode aparecer em múltiplas especializações.
No exemplo abaixo, considera-se o conjunto de pessoas vinculadas a uma universidade. Neste caso a especialização não é exclusiva, já que a mesma pessoa pode aparecer em múltiplas especializações. Uma pessoa pode ser professor de um curso e ser aluno em outro curso (pós-graduação, por exemplo). Por outro lado, uma pessoa pode acumular o cargo de professor em tempo parcial com o cargo de funcionário, ou, até mesmo, ser professor de tempo parcial em dois departamentos diferentes, sendo portanto duas vezes professor.



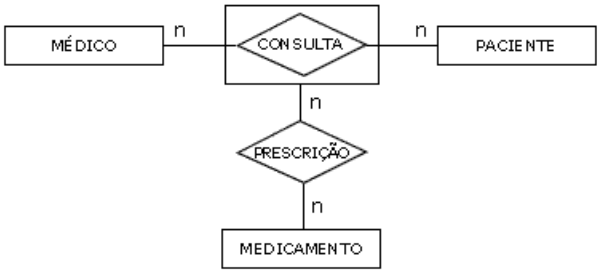
O principal problema que este tipo de generalização/especialização apresenta é que neste caso as entidades especializadas não podem herdar o identificador da entidade genérica. No caso, o identificador de *pessoa* não seria suficiente para identificar *professor*, já que uma pessoa pode ser múltiplas vezes professor.

ENTIDADE ASSOCIATIVA

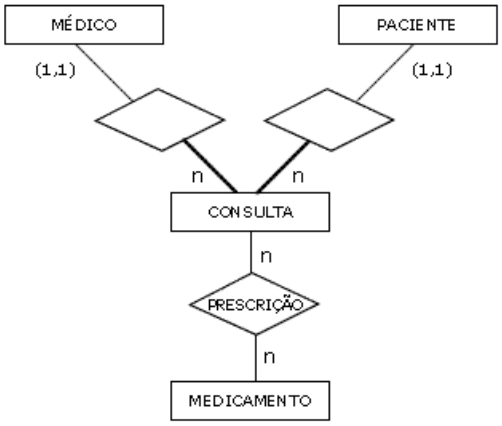
Um relacionamento é uma associação entre entidades. Na modelagem Entidade-Relacionamento não foi prevista a possibilidade de associar uma entidade com um relacionamento ou então de associar dois relacionamentos entre si.

Uma entidade associativa nada mais é que a redefinição de um relacionamento, que passa a ser tratado como se fosse também uma entidade.

Representamos graficamente uma entidade associativa conforme o exemplo abaixo:



ENTIDADE ASSOCIATIVA



SUBSTITUINDO RELACIONAMENTO POR ENTIDADE

SOFTWARE

Uma opção interessante para desenvolver modelos conceituais:

brModelo (Desenvolvido por Carlos Henrique Cândido - UFSC)

http://www.sis4.com/brModelo/download.aspx

EXERCÍCIOS

Sistema Locadora

Uma locadora de vídeos possui aproximadamente 2000 DVDs, cujo empréstimo deve ser controlado.
Cada DVD possui um número. Para cada filme, é necessário saber seu título e sua categoria (comédia, drama, aventura, ...). Cada filme recebe um identificador próprio. Deve-se controlar também que filme cada DVD contém. Para cada filme há pelo menos um DVD, e cada DVD contém somente um filme.
Os clientes podem desejar encontrar os filmes estrelados pelo seu ator predileto. Por isso, é necessário manter a informação dos atores que estrelam em cada filme. Nem todo filme possui estrelas. Para cada ator os clientes às vezes desejam saber o nome real, bem como a data de nascimento.
A locadora possui muitos clientes cadastrados. Somente clientes cadastrados podem alugar DVDs. Para cada cliente é necessário saber seu nome e seu sobrenome, seu telefone, seu email e seu endereço. Além disso, cada cliente recebe um número de associado.
Finalmente desejamos saber que DVDs cada cliente tem emprestados. Um cliente pode ter vários DVDs em um instante do tempo. Não são mantidos registros históricos de aluguéis.

Sistema Escola

Uma escola de informática mantém em seu catálogo vários cursos livres com duração entre trinta e sessenta dias. Cada curso é recebe um código identificador.
Professores são contratados para ministrar um ou mais cursos e, portanto é necessário saber quais cursos cada professor está habilitado a ministrar. Os professores recebem um número de matrícula. A escola deseja manter também registrado o endereço, telefone, email, bem como os números dos seguintes documentos RG, CPF e Carteira de Trabalho de todos os seus professores.
Há várias turmas para cada curso e cada turma está alocada em uma sala. Cada turma, identificada por um código, tem apenas um professor.
Um aluno pode matricular-se simultaneamente em vários cursos e, portanto, pertencer a mais de uma turma. No momento da matrícula o aluno recebe um RA (válido para um ou mais cursos). A escola mantém registrado o nome, endereço, telefone, email, RG e CPF de todos os seus alunos.

Sistema Cinema

Projetar um modelo de dados que atenda às necessidades de controle dos cinemas e filmes em uma determinada empresa de distribuição de filmes.
Regras de negócio que serão a base para o desenvolvimento do DER:
A empresa de distribuição possui vários cinemas em várias localidades.
Cada cinema possui identificação única, um nome fantasia, um endereço, etc.
Cada filme é registrado com um título original, tempo de duração, etc.
Existirá um único diretor para cada filme.
Cada filme terá vários atores.
As sessões possuem horários variados.
Os atores de um filme podem atuar em diversos filmes, assim como o diretor de um filme pode também ser ator neste filme, ou ainda mais, ser ator em outro filme. Um ator possui número, nome, nacionalidade e idade.

Sistema Museu de Arte

Projete o DER para um museu de arte, conforme as seguintes informações coletadas:
O museu tem uma coleção de objetos de arte. Cada objeto tem um número, um artista (se conhecido), um ano (quando foi criado, se conhecido), um título e uma descrição.

Os objetos de arte são categorizados de diversas formas de acordo com o seu tipo. Há três tipos principais: pintura, escultura e estátua, mais um chamado outros para comodar objetos que não se enquadram em algum dos três tipos principais.
Uma pintura tem um tipo (óleo, guache, etc.), material (papel, tela, madeira, etc.) e estilo (moderno, abstrato, etc.).
Uma escultura tem o material do qual foi criado (madeira, pedra, etc.) e estilo.
Um objeto de arte na categoria outros tem um tipo (impressão, foto, etc.) e estilo.
Um objeto de arte também é categorizado como coleção permanente que é de propriedade do museu (com as informações de data de aquisição, se está exposto ou guardado e preço) ou empréstimo, do qual tem informações da coleção (da qual foi emprestado), data de empréstimo e data de devolução.
Cada objeto de arte tem também a descrição de seu país/cultura de origem (iteliano, egípcio, americano, etc.) e época (renascimento, moderna, antiga, etc.).
O museu mantém informações sobre artistas, quando conhecidos: nome, data de nascimento, data de falecimento (se não-vivo), país de origem, época, estilo principal e descrição. Nome é assumido como sendo único.
Diferentes exibições ocorrem, cada uma tem um nome, data de início e data de término, e é relacionada a todos os objetos de arte que estão à mostra durante a exibição.
Mantêm-se também informações de outras coleções com a qual o museu interage, incluindo nome (único), tipo (museu, pessoal, etc.), descrição, endereço, telefone e pessoa de contato.

Sistema Empresa

Uma empresa é organizada em departamentos. Cada departamento possui um nome e um código único, e o departamento pode ter várias localidades (cidades). Os projetos existentes na empresa são, obrigatoriamente, controlados por um departamento, e cada projeto possui um nome, um código único e uma única localização (cidade), que pode ser diferente das possíveis localidades do departamento que o controla. Alguns departamentos não possuem projetos sobre sua responsabilidade, como por exemplo o "departamento pessoal".
No caso dos empregados da empresa é armazenado número de matricula, nome, endereço, salário, sexo e data de nascimento. Quase todos os empregados tem um outro empregado que é o seu supervisor direto, e consequentemente, somente alguns são supervisores, conforme a sua hierarquia na empresa. Em função da cadeia hierárquica existem empregados que não possuem supervisores.
A maioria dos empregados são alocados a um departamento, ou seja, pode até existir um empregado sem departamento, mas todo departamento deve possuir empregados alocados a ele, além disso, todo departamento tem um chefe que o gerencia, a partir de uma data, pois a empresa implementa um sistema de rodízio na chefia dos departamentos, o rodízio na chefia determina que um empregado só pode ser chefe de somente um departamento.
Um empregado pode trabalhar em mais de um projeto, mesmo que não seja do seu departamento, dedicando algumas horas por semana em cada um dos projetos. E, é claro, alguns empregados, como os do "departamento pessoal", não estão empenhados em nenhum projeto. Por outro lado, todo projeto tem pelo menos um ou mais empregados trabalhando nele.
A empresa oferece alguns benefícios sociais aos dependentes dos seus empregados, caso ele possua. Para tanto, é mantido para cada dependente do empregado o nome do dependente, o sexo, a data de nascimento e o grau de parentesco.

Sistema Agenda

Deseja-se construir uma agenda de endereços de pessoas e empresas onde trabalham. As pessoas da agenda possuem endereços para fins postais e telefones, que podem ser residenciais, comerciais, fax, celular ou de outro tipo. Anota-se no telefone DDD, prefixo e número. Telefones do tipo fixo são associados a endereços e telefones do tipo móvel são associados a pessoas. A cada endereço associa-se um código de endereço(único), rua, número, bairro, CEP, todo endereço de pessoa pode ser classificado dentre os tipos residência própria, residência com os pais, residência com parentes, residência com amigos, de referência ou outro, sendo que, um endereço pode pertencer a mais de uma pessoa. Para toda pessoa da

agenda armazena-se seu código seqüencial na agenda, seu nome. Uma pessoa pode ser amiga de outras pessoas e têm armazenadas a data de início da amizade entre elas, ou se a pessoa for parente de outras pessoas deve armazenar o tipo do parentesco. Alem disso, pessoas têm armazenadas, o seu sexo e sua data de nascimento e a profissão. Sendo que algumas pessoas podem trabalhar em uma empresa da agenda. Da Empresa, armazena-se a razão social da empresa, a inscrição estadual, o CGC, o ramo de dedicação da empresa e o proprietário da empresa que é uma pessoa armazenada na agenda. As empresas da agenda possuem um único endereço, e em uma empresa trabalham várias pessoas da agenda, sendo que a existência de uma empresa está condicionada a existir uma pessoa na agenda que trabalha nela.

Sistema Imobiliária

Uma imobiliária lida com venda de imóveis urbanos. Para qualquer imóvel têm-se registradas a sua inscrição, preço de venda, área total e área construída. Todo imóvel tem localização num endereço. A cada endereço associa-se um código de endereço, rua, número, bairro, CEP e os telefones associados (se existirem). Uma pessoa pode assumir um dos seguintes papéis em relação a imobiliária: corretor, proprietário de imóvel ou comprador. Sobre o proprietário do imóvel têm-se CPF, nome, estado civil e, se for casado, o nome do cônjuge. Um proprietário pode ter vários imóveis a venda na imobiliária. Sobre os compradores têm-se CPF, nome, profissão e uma lista de preferências de imóveis a adquirir. Sobre os corretores da imobiliária têm-se número do CRECI, nome e data de admissão. Um corretor negocia com um comprador a venda de um imóvel. E, é claro, um corretor negocia outros imóveis com outros compradores, podendo um mesmo comprador adquirir um outro imóvel com o mesmo comprador e com outros compradores. Sobre a venda são necessárias as seguintes informações: data da venda, valor da venda e valor da comissão.

Sistema campeonato de futebol

Na construção de um banco de dados para administrar times, jogos e campeonatos de futebol, cada time tem um nome (único) e uma quantidade de jogadores que jogam para o time, a partir de uma data inicial e final do contrato.
Nos jogos do time, cada um desses jogadores é escalado, e, é preciso saber qual foi a sua escalação no jogo (o número da camiseta do jogador). Para cada jogador tem-se o nome, o apelido, a posição, o salário e o número de registro na federação.
Um time participa de jogos com outros times dentro de campeonatos. Um jogo é realizado em estádio numa certa data (dia e hora) e produz um resultado, registrando, também, o público presente e a renda do jogo. Cada jogo realizado tem um número de ordem em função do campeonato, ou seja, o número de ordem serve para identificar um jogo dentro do campeonato que ele pertence.
Os estádios tem nome (único), cidade, capacidade de público e o(s) time(s) que mandam jogo naquele estádio, sendo que os times só possuem um estádio onde eles mandam seus jogos.
Em um jogo válido pelo campeonato deve ter sempre um juiz da federação, sobre os juízes que apitam os jogos tem-se os nome, número de registro na federação, nome da mãe, classe, data que começou como juiz e para quais campeonatos está designado, e claro durante um campeonato temos vários juizes escalados.
Para um campeonato tem-se o nome (único), quantidade de times e descrição, e para cada campeonato precisa-se ter os times que participaram do campeonato, bem como a classificação de cada time e o time que foi o campeão.

Sistema Bancário

Em sistema bancário simplificado temos: Clientes, onde cada cliente tem CPF, RG, nome, endereço, telefone e estado civil. Um cliente pode ter mais de uma conta em agências distintas. As agências possuem código da agência, nome, endereço e nome do gerente. Sobre as contas tem-se número da conta e saldo atualizado. Uma conta é gerenciada por uma única agência. Os clientes podem movimentar suas contas, na movimentação deve constar sobre o tipo (crédito ou débito), quantia, data e hora.

MODELO RELACIONAL

CHAVE PRIMÁRIA: Atributo através do qual seja possível identificar determinado registro. Uma chave primária não pode ser repetida, ou seja, o conjunto de valores que constituem a chave primária deve ser único dentro de uma tabela.

- Chave primária simples: apenas um atributo (campo) compõe a chave primária.
- Chave primária composta: mais de um atributo compõe a chave primária.

CHAVE ÚNICA: Utilizada quando determinado campo não deve ser repetido e não é chave primária. Aumenta a consistância do banco de dados.

Exemplo: Cadastro de clientes. Cada cliente recebe um código único, que é a chave primária. Para maior segurança e consistência podemos optar que o campo RG também seja único, evitando que o mesmo cliente seja cadastrado duas vezes.

CHAVE ESTRANGEIRA: Utilizada quando queremos que o valor de um atributo seja validado a partir do valor de atributo de uma outra tabela. Criamos assim uma relação de dependência (um relacionamento) entre as tabelas.

Exemplo: Antes de efetuar o cadastro de um pedido de venda, é necessário que o cliente em questão conste no cadastro de clientes.

RELACIONAMENTOS: Associação estabelecida entre campos comuns de duas tabelas. Dessa forma permitimos o estabelecimento de correspondência entre registros de diferentes tabelas.

Relacionamento um-para-um (1-1):

```
FUNCIONARIO                          ARMARIO
CodigoFuncionario (1) ----|          NumeroArmario
NomeFuncionario           |--- (1) CodigoFuncionario
EnderecoFuncionario                  DataCadastro
TelefoneFuncionario
```

Relacionamento um-para-muitos (1-N):

```
CLIENTE                          PEDIDO
CodigoCliente (1) ---|           NumeroPedido
NomeCliente          |---- (N) CodigoCliente
EnderecoCliente                  DataPedido
TelefoneCliente                  CodigoProduto
```

Relacionamento muitos-para-muitos (N-N): Não é possível efetuar de forma direta. Esse tipo de relacionamento só é possível definindo uma terceira tabela (tabela de associação ou tabela de detalhes). Essa tabela deve possuir chave primária composta de dois campos e as chaves estrangeiras provenientes das duas tabelas primárias. Concluindo, um relacionamento de muitos-para-muitos deve ser dividido em dois relacionamentos de um-para-muitos com uma terceira tabela.

```
PEDIDO                          ITENSPEDIDO                      PRODUTO
NumeroPedido (1) -------- (N) NumeroPedido          |---- (1) CodigoProduto
CodigoCliente                 CodigoProduto (N) ---|          DescProduto
DataPedido                    Quantidade                       ValorProduto
                                                               EstoqueProduto
```

NOTAÇÃO RESUMIDA PARA MODELOS LÓGICOS RELACIONAIS

Notação compacta, útil para discussões sobre a estrutura geral do banco de dados, utilizada quando não se deseja entrar no nível maior de detalhamento.

```
Departamento(CodDept, Nome)
Funcionario(CodFunc, Nome, CodDept, CPF)
            CodDept referencia Departamento
```

INTEGRIDADE DE DADOS

INTEGRIDADE DE DOMÍNIO: Zela pelos valores ideais e necessários para um atributo. Para isso definimos algumas regras de validação por meio de expressões compostas de valores constantes.

- Não permitir um estoque negativo
- Impedir uma data de nascimento superior à data atual
- Não permitir que o valor de um produto seja negativo

INTEGRIDADE DE ENTIDADE: Tem o objetivo de validar os valores permitidos a partir de valores já inseridos na própria entidade. Após uma "auto-consulta" a entidade vai permitir ou não a gravação do novo registro.

- Não permitir duas pessoas com o mesmo CPF
- Impedir que seja locada uma fita que já está locada

INTEGRIDADE REFERENCIAL: Zela pela consistência dos registros de uma entidade a partir de valores provenientes de outras entidades, isto é, determinado registro vai "depender" diretamente de um registro de outra tabela.

- Um registro em uma tabela pai pode ter um ou mais registros em uma tabela filho.
- Um registro em uma tabela filho sempre tem um registro coincidente em uma tabela pai.
- Para a inclusão de um registro em uma determinada tabela filho, é necessário que exista um registro pai coincidente.
- Um registro pai só poderá ser excluído se não possuir nenhum registro filho.

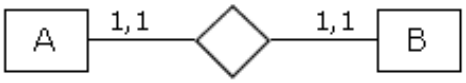
NOMENCLATURA DE CAMPOS

- Não utilizar caracteres especiais (exceto o underscroe "_");
- Começar com uma letra e não com um número;
- Evitar acentuação e "ç";
- Não utilizar espaços.

ORACLE - TIPOS DE DADOS	
Tipo	Descrição
char	Cadeia de caracteres de tamanho fixo n. O default é 1 e o máximo é 255.
varchar2 (n)	Cadeia de caracteres de tamanho variável. Tamanho máximo 4.000.
long	Cadeia de caracteres de tamanho variável. Tamanho máximo 2 GB. Só pode existir um campo deste tipo por tabela.
raw e long raw	Equivalente ao varchar2 e long, respectivamente. Utilizado para armazenar dados binários (sons, imagens, etc.) Limite: 2.000 bytes.
number (p,e)	Valores numéricos. Ex: number (5,2) armazena -999,99 a +999,99.
date	Armazena data e hora (inclusive minuto e segundo). Ocupam 7 bytes.

CONSTRAINTS / RESTRIÇÕES	
Nome	Uso
null	Informa se o campo pode receber valores nulos. Caso não possa, deve ser precedido de not.
unique	Indica que os valores na coluna, ou conjunto de colunas, não pode ser repetido. Cria um índice automaticamente.
check	Especifica os valores que uma coluna pode assumir.
primary key	Identifica a chave primária da tabela. Cria um índice automaticamente.
foreign key	Identifica a chave estrangeira da tabela. Implementada pela cláusula references.

MAPEAMENTO E-R (CONCEITUAL) PARA RELACIONAL (LÓGICO)



A1	A2	. . .	B1	B2

Fusão de tabelas

AK	BF

BK	

BF - NOT NULL

PK de B será FK de A

AK	

BK	AF

AF - NOT NULL

PK de A será FK de B

AK	

BK	

Pode ser NULL

Indiferente

AK	

BK	AF

AF - NOT NULL

PK de A será FK de B

AK	

BK	AF

Pode ser NULL

PK de A será FK de B

AK	

BK	AF

AF - NOT NULL

PK de A será FK de B

AK	

BK	AF

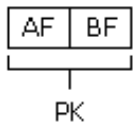
Pode ser NULL

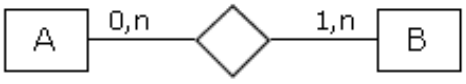
PK de A será FK de B

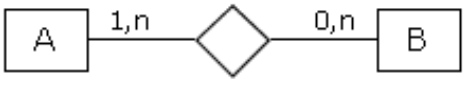
AK	

BK	AF

PK de A e PK de B
serão FK em R

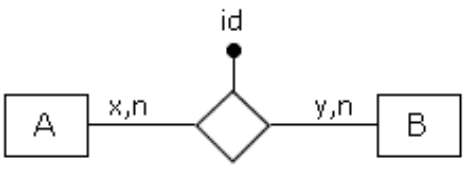






Cardinalidade máxima n:n
solução única para os quatro casos

RELACIONAMENTO COM ATRIBUTO IDENTIFICADOR

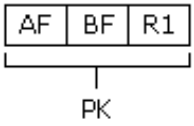


x=0 ou 1 y=0 ou 1

AK	

BK	AF

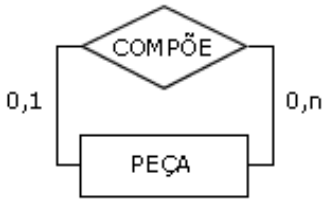
PK de A e PK de B
serão FK em R



R1 -> id

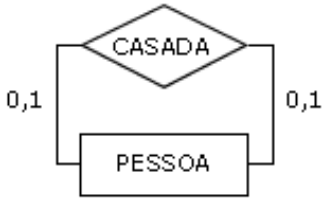
MAPEAMENTO E-R (CONCEITUAL) PARA RELACIONAL (LÓGICO)

AUTO-RELACIONAMENTO - 1:N



cod_pk	descricao	cod_fk
11	correia	
12	parafuso	
21	freio	12
22	carburador	12

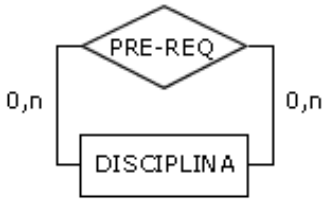
AUTO-RELACIONAMENTO - 1:1



unique

cod_pk	nome	cod_cj	
101	Antonio	201	esposa
102	Benedito	202	
201	Maria	101	marido
202	Claudia	102	
203	Sueli		

AUTO-RELACIONAMENTO - N:N

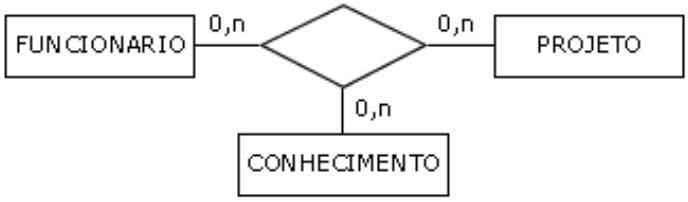


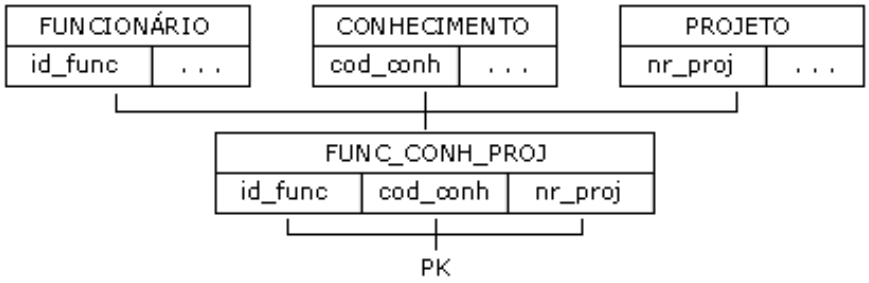
cod	disciplina
101	Cálculo I
102	Cálculo II
201	Física I
202	Física II

PK

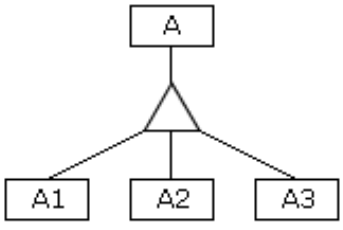
cod	cod_pre
102	101
202	201
202	101

RELACIONAMENTO TERNÁRIO





GENERALIZAÇÃO / ESPECIALIZAÇÃO



A	
codigo	colunas comuns p/ A1, A2, A3

A1	
codigo	colunas exclusivas p/ A1

A2	
codigo	colunas exclusivas p/ A2

A3	
codigo	colunas exclusivas p/ A3

NORMALIZAÇÃO

- Conceito introduzido em 1970 por E. F. Codd (1FN).
- Processo matemático formal com fundamento na teoria dos conjuntos.

O processo de normalização aplica uma série de regras sobre as tabelas de um banco de dados para verificar se estas foram corretamente projetadas.

Objetivos principais:

- Garantir a integridade dos dados, evitando que informações sem sentido sejam inseridas.
- Organizar e dividir as tabelas da forma mais eficiente possível, diminuindo a redundância e permitindo a evolução do banco de dados.

São seis as formas normais mais conhecidas:

- **1FN** – 1ª Forma Normal
- **2FN** – 2ª Forma Normal
- **3FN** – 3ª Forma Normal
- **FNBC** – Forma Normal de Boyce e Codd
- **4FN** – 4ª Forma Normal
- **5FN** – 5ª Forma Normal

Nota: As três primeiras formas normais atendem à maioria dos casos de normalização.

Uma forma normal engloba todas as anteriores, i.e., para que uma tabela esteja na 2FN, ela obrigatoriamente deve estar na 1FN e assim por diante.

Normalmente após a aplicação das regras de normalização, algumas tabelas acabam sendo divididas em duas ou mais tabelas. Este processo colabora significativamente para a estabilidade do modelo de dados e reduz consideravelmente as necessidades de manutenção.

Conceitos úteis:

1. Chaves

- **Chave candidata:** Atributo ou conjunto de atributos que são únicos para cada registro. Para cada tabela podemos ter uma ou várias chaves desse tipo. Exemplo: *codigo* e *cpf*.
- **Chave primária:** Entre as chaves candidatas, escolhemos uma para ser o identificador principal da tabela. Este atributo passa a ser chamado de chave primária (PK – Primary Key).
- **Chaves alternativas:** São as chaves candidatas que não foram definidas como chave primária.
- **Chave estrangeira:** É o atributo ou conjunto de atributos que faz a ligação com a chave primária de outra tabela (FK – Foreign Key).

2. Dependência Funcional (DF)

Sempre que um atributo **X** identifica um atributo **Y**, dizemos que entre eles há uma **dependência funcional**. Temos, portanto, que X é o **determinante** e que Y é o **dependente**. A representação é: X➔Y (lê-se X determina Y ou Y é dependente de X).

cidade➔estado

estado➔país

Em outras palavras, estado é funcionalmente dependente de cidade e país é fincionalmente dependente de estado. Ou ainda, estado é dependente de cidade e país é dependente de estado. E por último, cidade determina estado e estado determina país.

3. Trivialidade

A dependência funcional trivial indica que um determinante com mais de um atributo pode determinar seus próprios membros quando isolados.

{banco, agência} ➔ banco DF trivial: banco é parte do determinante

{banco, agência} ➔ agência DF trivial: agência é parte do determinante

Quando um determinante identifica outro atributo qualquer, temos uma dependência funcional não trivial (essa DF é a que nos interessa no processo de normalização):

{banco, agência} ➔ cidade DF não trivial: cidade não faz parte do determinante

4. Transitividade

Se um atributo X determina Y e se Y determina Z, podemos dizer que X determina Z de forma transitiva. i.e., existe uma dependência funcional transitiva de X para Z.

cidade ➔ estado

estado ➔ país

ciade ➔ país (cidade determina país de forma transitiva)

5. DF (Dependência Funcional) irredutível à esquerda

Dizemos que o lado esquerdo de uma dependência funcional é irredutível quando o determinante está em sua forma mínima. Temos a forma mínima quando não é possível reduzir a quantidade de atributos determinantes sem perder a dependência funcional.

{cidade, estado} ➔ país (não está na forma irredutível à esquerda, pois podemos ter somente o estado como determinante)

estado ➔ país (forma irredutível à esquerda)

Nota: *Nem sempre estar na forma irredutível à esquerda significa possuir um determinante com apenas uma coluna.*

6. DMV (Dependência Multivalorada)

A DMV é uma ampliação da Dependência Funcional (DF). Na DMV o valor de um atributo determina um conjunto de valores de um outro atributo.

É representada por X➔➔Y (X multideterina Y ou Y é multidependente de X).

DF: {CPF}➔{Nome} Temos somente um nome para cada CPF

DMV: {CPF}➔➔{Dependente} Temos vários dependentes para cada pessoa

PRIMEIRA FORMA NORMAL:

Uma Tabela está na Primeira Forma Normal quando seus atributos não contêm grupos de repetição (tabelas aninhadas).

Proj(CodProj,Desc(CodFunc,Nome,Cargo,Salario,DtInicio))	*Não está na 1FN*
Proj(CodProj,Desc) ProjFunc(CodProj,CodFunc,Nome,Cargo,Salário,DtInicio)	*Está na 1FN*

SEGUNDA FORMA NORMAL:

Ocorre quando a chave primária é composta por mais de uma coluna. Neste caso, devemos observar se todos as colunas que não fazem parte da chave dependem de todos os colunas que compõem a chave. Se alguma coluna depender somente de parte da chave composta (dependência funcional parcial), então esta coluna deve pertencer a outra tabela.

ProjFunc(CodProj,CodFunc,Nome,Cargo,Salario,DtInicio)	*Não está na 2FN*
ProjFunc(CodProj,CodFunc,DtInicio) Func(CodFunc,Nome,Cargo,Salario)	*Está na 2FN*

TERCEIRA FORMA NORMAL:

Uma tabela está na Terceira Forma Normal quando cada coluna não chave depende diretamente da chave primária, isto é, quando não há dependências funcionais transitivas ou indiretas. Uma dependência funcional transitiva acontece quando uma coluna não chave primária depende funcionalmente de outra coluna ou combinação de colunas não chave primária.

Func(CodFunc,Nome,Cargo,Salario)	*Não está na 3FN*
Func(CodFunc,Nome,Cargo) Cargo(Cargo,Salario)	*Está na 3FN*

ÁLGEBRA RELACIONAL

- Desenvolvida para descrever operações sobre uma base de dados relacional
- Cada operador toma uma ou duas relações como sua entrada e produz uma nova relação como sua saída
- Linguagem da consulta teórica, usuários não a utilizam diretamente
- É usada internamente em todos os SGBDRs

Características:

- Constituída de cinco operadores fundamentais:
  - Seleção σ (sigma)
  - Projeção π (pi)
  - Produto cartesiano X
  - Diferença –
  - União $\cup$
- Três operadores derivados:
  - Intersecção $\cap$
  - Junção $\bowtie$
  - Divisão :

SELEÇÃO

Seleciona todas as tuplas que satisfazem à condição de seleção de uma relação R.

R

A	B
a1	b1
a2	b2

$\sigma_{(A='a1')}(R)$

A	B
a1	b1

PROJEÇÃO

Produz uma nova relação com apenas alguns atributos de R, removendo as tuplas duplicadas.

R

A	B
a1	b1
a2	b2

$\pi_{(B)}(R)$

B
b1
b2

PRODUTO CARTESIANO

Produz uma nova relação com todas as possíveis tuplas resultantes da combinação de duas tuplas, uma de cada relação envolvida na operação.

R

A	B
a1	b1
a2	b2

S

C	D
c2	d2
c3	d3

(R X S)

A	B	C	D
a1	b1	c2	d2
a1	b1	c3	d3
a2	b2	c2	d2
a2	b2	c3	d3

DIFERENÇA

Produz uma nova relação com todas as tuplas em R que não estão em S. *R e S devem ter número iguais de atributos.*

R

A	B
a1	b1
a2	b2

S

A	B
a2	b2
a3	b3

(R - S)

A	B
a1	b1

UNIÃO

Produz uma nova relação com a união das tuplas em R e em S. *R e S devem ter número iguais de atributos.*

R

A	B
a1	b1
a2	b2

S

A	B
a2	b2
a3	b3

(R ∪ S)

A	B
a1	b1
a2	b2
a3	b3

INTERSECÇÃO

Produz uma nova relação com a intersecão das tuplas em R e em S, ou seja, com as tuplas que aparecem em R e em S. *R e S devem ter número iguais de atributos.*

R

A	B
a1	b1
a2	b2

S

A	B
a2	b2
a3	b3

(R ∩ S)

A	B
a2	b2

JUNÇÃO

Junção de R com S = (R X S) [expressão de seleção]

R

A	B
a1	b1
a2	b2

S

C	D
b2	d2
b1	d3

R X S [B = C]

A	B	C	D
a1	b1	b1	d3
a2	b2	b2	d2

JUNÇÃO NATURAL

Quando a condição de junção for a igualdade do valor de um atributo comum e o atributo comum aparecer só uma vez no resultado.

R

A	B
a1	b1
a2	b2

S

C	D
b2	d2
b1	d3

R * S

A	B	D
a1	b1	d3
a2	b2	d2

DIVISÃO

Produz uma nova relação S contendo todas as tuplas de A (dividendo) que aparecem em R (mediador) com todas as tuplas de B (divisor).

A

A
a1
a2
a3
a4
a5

R

A	B
a1	b1
a1	b2
a1	b3
a1	b4
a2	b1
a2	b2
a3	b2
a4	b2
a4	b4

B

B
b1

S

A
a1
a2

B

B
b2
b4

S

A
a1
a4

B

B
b1
b2
b3
b4

S

A
a1

SOFTWARE

WinRDBI (Windows Relational DataBase Interpreter)

http://www.eas.asu.edu/~winrdbi/

Arizona State University

- Relational Algebra
- Domain Relational Calculus
- Tuple Relational Calculus
- SQL

CÁLCULO RELACIONAL

- Alternativa à Álgebra Relacional
- Logicamente equivalentes
- Baseia-se em um ramo da lógica matemática chamado *cálculo de predicados*

Referências bibliográficas:

Cálculo relacional como base para uma linguagem de consulta: J.L.Kuhns: "Answering Questions by Computer: A Logical Study." (1967)

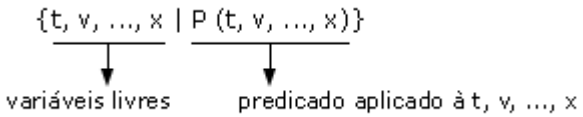
Cálculo de predicados adaptados especificamente a banco de dados relacionais: E.F.Codd: "A Relational Model of Data for Large Shared Data Banks." (1970)

- Cálculo Relacional: descreve qual é o problema (não procedural).
- Álgebra Relacional: Prescreve um procedimento para resolver o problema (procedural).

CÁLCULO RELACIONAL DE TUPLA

É baseado na especificação de um conjunto de variáveis de tuplas. Cada variável de tupla pode assumir como seu valor qualquer tupla da relação especificada.

Forma geral:



- **Variável livre:** Assume velores de tuplas de uma ou mais relações. Constitui a resposta da consulta.
- **Predicado:** Expressão lógica que, se verdadeira para determinados valores das variáveis livres t, v, ..., x, retorna os valores destas variáveis na resposta da consulta.

EXEMPLOS – SELEÇÃO E PROJEÇÃO:

1. Buscar os dados dos pacientes que estão com sarampo:

```
{p | p ∈ Pacientes ∧ p.doenca = 'sarampo'}
```

2. Buscar o número e a capacidade dos ambulatórios do terceiro andar

```
{a.numero, a.capacidade | a ∈ Ambulatorios ∧ a.andar = 3}
```

EXEMPLOS – PRODUTOS OU JUNÇÃO:

1. Buscar o nome dos médicos que têm consulta marcada e as datas das suas consultas:

```
{m.nome, c.data | m ∈ Medicos ∧ c ∈ Consultas ∧ m.crm = c.crm}
```

2. Buscar os nomes dos médicos ortopedistas e o número e andar dos ambulatórios onde eles atendem:

```
{m.nome, a.numero, a.andar | m ∈ Medicos ∧ m.especialidade = 'ortopedia' ∧
a ∈ Ambulatorios ∧ m.numero = a.numero}
```

QUANTIFICADORES EXISTENCIAIS:

∃ exists

∀ forall

- QUANTIFICADOR EXISTENCIAL

∃ **v (P)**

Existe pelo menos um valor *v* que torna *P* verdadeira.

- QUANTIFICADOR UNIVERSAL

∀ **v (P)**

Para todos os valores de *v*, *P* é verdadeira.

EXEMPLO: Suponha que a variável *v* percorra o conjunto *Senado*, e suponha que *P* seja a FBF* *"v é mulher"*, então:

∃ v (P) → verdadeiro

∀ v (P) → falso

*FBF – Fórmula Bem Formada (WFF – Well Formed Formula)

EXEMPLOS – QUANTIFICADOR EXISTENCIAL:

1. Buscar o nome dos médicos que atendem em ambulatórios do segundo andar:

```
{m.medico | m ∈ Medicos ∧ ∃ a ∈ Ambulatorios (a.andar = 2 ∧ m.numero = a.numero)}
```

2. Buscar o nome e o problema dos pacientes de têm consulta marcada com o médico João da Silva:

```
{p.nome, p.problema | p ∈ Pacientes ∧ ∃ c ∈ Consultas (p.rg = c.rg ∧ ∃ m ∈ Medicos (c.crm = m.crm ∧ c.nome = 'Joao da Silva'))}
```

EXEMPLOS – QUANTIFICADOR UNIVERSAL:

1. Buscar o nome dos médicos que têm consulta marcada com todos os pacientes:

```
{m.medico | m ∈ Medicos ∧ ∀ p ∈ Pacientes (∃ c ∈ Consultas (p.rg = c.rg ∧ c.crm = m.crm))}
```

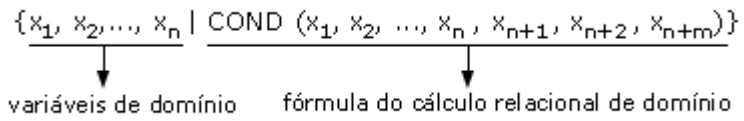
2. Buscar o nome dos pacientes de têm consulta marcada com todos os médicos ortopedistas:

```
{p.nome | p ∈ Pacientes ∧ ∀ m ∈ Medicos (m.especialidade = 'ortopedia' → ∃ c ∈ Consultas (c.crm = m.crm ∧ c.rg = p.rg))}
```

CÁLCULO RELACIONAL DE DOMÍNIO

No CRD as variáveis estendem-se sobre valores únicos de domínios de atributos. Para formar uma relação de grau n para um resultado de consulta, faz-se necessário criar n variáveis de domínio, uma para cada atributo.

Forma geral:



- **Variáveis de domínio:** Abrangem os valores únicos dos domínios dos atributos. Devemos ter *n* destas variáveis de domínio, uma para cada atributo.
- **Fórmula:** São constituídas a partir de átomos, variáveis e quantificadores.

EXEMPLOS:

1. Buscar o nome da agência, número da conta que têm saldo superior a 1200 reais:

$\{a, c, s \mid a, c, s \in Contas \wedge s > 1200\}$

2. Buscar os nomes de todos os clientes que tenham uma conta em todas as agências localizadas na Aclimação:

$\{ c \mid \forall x, y, z (x, y, z \in Agencias \wedge y = 'Aclimacao' \rightarrow \exists a, b (x, a, b \in Contas \wedge c, a \in Clientes)\}$

NOTA: *Enquanto a SQL (baseada no Cálculo Relacional de Tupla) estava sendo desenvolvida pela IBM Research em San Jose, Califórnia, outra linguagem, chamada QBE (Query By Example (baseada no Cálculo Relacional de Domínio) estava sendo desenvolvida quase simutaneamente pela IBM Research em Yorktown Heights, Nova York.*