Modelagem de Dados

Sistemas Gerenciadores de Bancos de Dados – SGBD

Quando projetamos um banco de dados, precisamos saber quais serão as aplicações que utilizarão o banco projetado. É necessário um dimensionamento para poder indicar o SGBD mais apropriado para o cliente que deseja o software. Portanto, os conceitos sobre SGBD são importantes para a compreensão do processo de modelagem que os dados devem passar, para depois serem armazenados nestes softwares.

- **Banco de Dados:** Conjunto de dados ou informações relacionadas entre si. Um banco de dados ou base de dados é um conjunto de arquivos integrados que são usados pelos sistemas de aplicação de uma determinada empresa. O termo faz referência ao armazenamento dos dados.
- **SGBD:** Sistema gerenciador de banco de dados. Um SGBD é um conjunto de softwares que possuem a finalidade de gerenciar as informações de um banco de dados. Devem organizar, acessar, controlar e proteger as informações contidas no banco de dados. O SGBD pode ser distribuído por diversos computadores, no mesmo local ou até em locais diferentes (espaços, cidades, países).
- **Transação:** Uma transação é um programa em execução ou processo que inclui um ou vários acessos ao banco de dados, como leitura ou atualização de registros. É uma unidade de execução de programa que acessa e, possivelmente, atualiza vários itens de dados. Geralmente, é o resultado da execução de um programa de usuário escrito em uma linguagem de manipulação de alto nível ou em uma linguagem de programação, como Java, C# ou em SQL, entre outras.

Veja abaixo as funções, características e funcionalidades dos SGBD’s:

As funções:

- Definição, recuperação e alteração de dados de um banco de dados.
- Pesquisa para recuperar uma determinada informação.
- Acesso, alteração e geração de relatórios das informações.
- Segurança a acessos não autorizados.
- Proteção e recuperação dos dados quando houver problemas de hardware ou software.
- Recurso de backup (cópia de segurança), possibilitando a recuperação posterior dos dados.
- Compartilhamento dos dados, controle da redundância dos dados e restrições de integridade.

As características:

- **Natureza Auto Descritiva:** Permite que um banco de dados não tenha somente os dados e sim a sua definição e estrutura.
- **Abstração de Dados:** Oferece aos usuários uma representação conceitual de dados, omitindo vários detalhes, por exemplo, como são armazenados ou como as operações são realizadas.
- **Visão (View):** Pode ser um subconjunto de um banco de dados. Pode ser resultante de pesquisas, que retornam parte das informações armazenadas.
- **Controle de Concorrência:** Fator primordial para que o compartilhamento de dados e as transações sejam realizados com sucesso para os usuários do banco de dados.

As funcionalidades:

- Permite inclusão, exclusão, seleção, ordenação e junção de registros e entidades.
- Possibilita a cópia e a exclusão de entidades.
- Estabelece relações entre as entidades e a criação de chaves.
- Permite a importação ou exportação de dados entre outras bases de dados.
- Possibilita a alteração da estrutura de campos e entidades.
- Permite consultas e relatórios da base de dados.
- Possibilita a criação de usuários com permissão de acesso individualizado.

Modelo Relacional

A proposta do modelo relacional baseia-se na ideia de que as informações em uma base de dados podem ser representadas em tabelas, nas quais as linhas representam as informações cadastradas. O modelo relacional tem três aspectos básicos:

- **Estrutura:** Os dados inseridos no banco de dados são reconhecidos como tabelas.
- **Integridade:** As tabelas precisam satisfazer as restrições de integridades.
- **Manipulação:** Operações que podem ser realizadas com as tabelas, com a intenção de juntar, selecionar, excluir, dentre outras operações.

Modelagem de Dados

Na fase inicial do projeto de banco de dados estão as etapas da modelagem destes, que caracterizam completamente as necessidades de dados dos prováveis usuários destes bancos.

Essas etapas possuem algumas funções que podem ser classificadas como:

- **Concepção:** Entendimento da necessidade do cliente com relação ao software, e quando serão estabelecidos os objetivos principais da solução desejada.
- **Elicitação:** Conversas com os usuários do software com o objetivo de colher mais informações sobre os procedimentos realizados e que deverão estar presentes no software.
- **Elaboração:** Criação de modelos para a formalização dos requisitos. Com o modelo é possível encontrar falhas ou esquecimentos dos clientes ou do próprio analista de sistemas.
- **Negociação:** Com o modelo apresentado, os clientes podem querer mais itens, e é necessário verificar as viabilidades das sugestões.

Os elementos que compõe um banco de dados são classificados em: Dados, hardware, software e usuários.

- **Dados:** Os dados armazenados nos bancos de dados serão compartilhados entre os diversos usuários dos sistemas. É em função deste armazenamento que algumas regras de modelagem deverão ser rigorosamente seguidas.
- **Hardware:** O hardware é um elemento importante, pois determina como e onde os dados serão processados e armazenados.
- **Software:** O sistema gerenciador de banco de dados é o software essencial de um banco de dados. É o mais importante por isolar o acesso dos dados pelos usuários leigos, do acesso ao seu armazenamento no hardware. Existem, ainda, os softwares de aplicação, que são ambientes próprios do SGBD, ou linguagens de programação, que permitem criar uma interface amigável entre o usuário e o SGBD.

O modelo relacional usa um conjunto de tabelas para representar tanto os dados como a relação entre eles.

- **Tabelas:** Cada tabela possui múltiplas colunas e cada uma possui um nome único. Também podem ser denominadas como: Entidades, cadastros ou arquivos. Na tabela Aluno foram armazenadas quatro informações: Matrícula, nome, data de nascimento (abreviado: Dt. Nasc.) e o curso no qual o aluno está matriculado. As informações encontram-se dispostas em colunas (cada coluna contém uma categoria de informação) e as linhas (em cada uma delas há informações de um determinado aluno).
- **Entidades:** Uma entidade é um objeto ou indivíduo do mundo real que possui existência própria e cujas características ou propriedades desejamos registrar (pessoa, carro, animal, etc.). Também pode representar um objeto com existência conceitual como: Um projeto, um departamento, um trabalho acadêmico, entre outros. As entidades possuem características próprias e que podem variar na quantidade conforme a necessidade de cada sistema. Suponha uma entidade chamada Animal: Nela armazena-se várias informações básicas de acordo com a necessidade do sistema a ser realizado. No caso de serviço para uma clínica veterinária, as informações armazenadas poderiam ser: Vacinas tomadas, peso, altura, histórico de doenças, etc. No caso de venda do animal, na entidade Animal, teríamos as seguintes informações: Data de nascimento, sexo, raça, porte e valor de venda.
- **Atributos:** As informações que são armazenadas em tabelas podem ser agrupadas e são chamadas de atributos ou campos ou colunas. No exemplo da tabela Aluno, há quatro campos: Matrícula, nome, data de nascimento e cursos. Cada coluna representa uma categoria de informação, portanto, nesta coluna somente será permitido inserir o campo desta categoria e na coluna nome somente o nome do aluno deverá ser digitado. Um conjunto de atributos logicamente dispostos em uma entidade (ou tabela) são conhecidos como registros, linhas ou tuplas. Uma tabela poderá ter milhares de registros.

O processo de modelagem de dados visa buscar informações para criar o banco de dados. Saber identificar corretamente uma entidade é um dos primeiros passos para obter sucesso no desenvolvimento do software.

Sistemas de Apoio à Decisão

Os dados podem influenciar na tomada de decisão das empresas e, para ajudar os gestores a tomar decisões e apontar problemas que possam ocorrer, existem os sistemas de apoio a decisão. Eles são sistemas que ajudam na análise de informação do negócio, baseados em informações, em estatísticas, em gráficos de consumo, etc., que são obtidos a partir de informações detalhadas armazenadas nos SGBDs.

OLTP (Online Transaction Processing ou Processamento de Transações em Tempo Real)

São operações realizadas no SGBD que permitem realizar consultas na base de dados de forma repetitiva, a nível operacional e administrativos. O OLTP permite consultas do dia a dia da empresa.

Permite consultas no dia a dia da empresa. As transações realizadas no banco utilizam comandos do SQL (Structured Query Language ou Linguagem de Consulta Estruturada) como: Insert, update e delete. Estas transações são realizadas em tempo real e não armazenam um histórico das consultas realizadas no banco de dados, dificultando o auxílio à tomada de decisões.

Exemplo: Uma secretária faz a matrícula de um aluno do sistema de uma determinada faculdade, inserindo as informações do discente (inserção). Caso o aluno se matricule num curso e se arrependa querendo trocá-lo, a secretaria poderá modificar (consulta de alteração).

OLAP (Online Analytical Processing ou Processsamento Analítico em Tempo Real)

É o processo de análise dos dados dos sistemas transacionais OLTP, que permite a múltipla análise da informação, possibilitando que gestores possam tomar decisões mais assertivas.

Processo interativo de criar, gerenciar, analisar e gerar relatórios sobre os dados de banco de dados. Os dados coletados são armazenados em uma tabela multidimensional (ou arrays multidimensionais) para posterior análise de algoritmos e softwares específicos. Para fazer as análises, os dados são coletados do OLTP e isso acontece de acordo com a necessidade da empresa. O OLAP é a capacidade de analisar grandes volumes de informações dentro de um Data Warehouse, também faz referência às ferramentas analíticas utilizadas no BI para a visualização das informações gerenciais e dá suporte para as funções de análises do negócio organizacional.

Exemplo: No final do bimestre, o diretor desta mesma faculdade precisa de um relatório com o perfil dos alunos nos últimos três anos. No relatório, serão analisados: Idade, curso, semestres cancelados e situação financeira. Essas informações servirão para a tomada de decisão de oferta de novos cursos e para definição do perfil de aluno desejado.

Data Warehouse

Data Warehouse ou Depósito de Dados é um tipo especial de banco de dados. É um arquivo ou repositório de informações obtidas de várias origens (de vários bancos de dados) e armazenadas em um único local (preserva o banco de dados original da empresa). É orientado por assunto, integrado e não volátil, permitindo consultas para ajudar na tomada de decisão.

Data Mining

Data Mining ou Mineração de Dados refere-se à descoberta de novas informações em função de regras ou padrões em grandes quantidades de dados e pode ser aplicado em pesquisas científicas ou em empresas com o objetivo de aumentar significativamente a lucratividade.

BI

Business Intelligence (Inteligência de Negócios) é o processo de coleta, análise, monitoria e compartilhamento de informações para a gestão de negócios. O BI analisa dados brutos operacionais para encontrar informação útil e auxiliar na tomada de decisão.

Redundância

Redundância significa repetição. O grande número de dados e a modelagem de um banco de dados pode levar a redundâncias, ocasionando problemas futuros. O controle da redundância de um banco de dados é uma tarefa que deve ser realizada a partir da sua modelagem.

Análise dos Atributos

A mesma informação pode ser armazenada em várias entidades diferentes, levando muitas vezes à inconsistência do bando de dados.

Backup do Banco de Dados

As cópias físicas do banco de dados geram redundância. Esse tipo de redundância é controlada, visto que as cópias são necessárias para manter a segurança do banco de dados.

Segurança

A segurança da informação ganhou notoriedade nestas últimas décadas. Riscos de ataques internos e externos podem afetar os dados e a própria administração da empresa. As empresas precisam criar regras que possam garantir a segurança aos seus dados, surgindo, assim, a política de segurança da empresa. As regras são criadas conforme as necessidades de cada organização, estabelecendo critérios de acesso físico e remoto ao banco de dados.

Políticas de Segurança

Backup

Regras para Backup

Em caso de falhas, a cópia backup do banco de dados é realizada.

- **Responsabilidades:** Quem fará o backup? Quem terá acesso a ele?
- **Meios:** De que forma o backup será feito? Qual mídia ou nuvem usar? Qual software? Qual hardware?
- **Período:** Qual o intervalo dos backups? Diariamente, semanalmente, mensalmente?
- **Retenção:** Quanto tempo o backup deve ficar armazenado na mesma mídia?
- **Armazenamento:** Onde serão armazenados os backups? Quais locais seguros deverão ser indicados?

Senhas

Senhas de Acesso

Controlam o acesso ao banco de dados.

- **Integridade:** Garantia que as informações serão mantidas de forma íntegra e sem modificações indevidas de pessoas não autorizadas.
- **Confiabilidade:** Garantia que as informações armazenadas no banco somente serão acessadas por pessoas autorizadas previamente.
- **Disponibilidade:** Garantia de disponibilizar a informação somente às pessoas com permissão de acesso e de modificação.

Permissões

Autenticação do Usuário

Permissão para acessar e executar somente os trabalhos com autorização prévia, conforme o seu nível hierárquico dentro da empresa.

- Somente leitura dos dados.
- Inserir novos dados.
- Atualizar novos dados.
- Excluir dados.

Modelos de Banco de Dados

A utilização de aplicativos em computadores, tablets ou smartphones, para os mais diversos fins, está fazendo parte do dia a dia de todos. Várias empresas estão usando aplicações que oferecem serviços diferenciados para seus clientes, e o sucesso dessas aplicações dependem de um sistema muito bem estruturado, sendo uma das etapas primordiais nesse processo: A modelagem do banco de dados.

Modelagem de Dados

A modelagem de dados, de acordo com Coronel e Rob (2011), é um processo iterativo e progressivo: Começando com uma compreensão simples do domínio do problema e, conforme essa compreensão se desenvolve, o nível de detalhes do modelo também se amplia.

Uma das etapas mais importantes no desenvolvimento de um software é a elaboração de um projeto de banco de dados. Cougo (1997) descreve que um modelo de dados é um detalhamento dos tipos de informações que serão guardadas em um banco de dados. Para a construção de modelos de dados, usa-se uma linguagem de modelagem de dados que pode ser classificada de acordo com a forma de apresentar modelos, em linguagens textuais ou gráficas.

Abreu e Machado (2004) afirmam que o projeto de um sistema de informações é uma atividade complexa que inclui planejamentos, especificações e desenvolvimento de vários componentes. É necessário estabelecer uma sequência de atividades para guiar o processo de modelagem do banco de dados, sendo elas:

- Análise dos requisitos.
- Modelo conceitual.
- Modelo lógico.
- Modelo físico.

Levantamento e Análise de Requisitos

O processo de inicialização de um banco de dados de sucesso começa no procedimento de levantamento e de análise de requisitos, de acordo com Coronel e Rob (2011). Assim, nesta etapa, são levantadas as necessidades do cliente.

Modelagem Conceitual

Assim que os requisitos do software forem levantados, o próximo passo é a modelagem conceitual. Não contém detalhes sobre como será representado em meio físico, representa as informações no nível da realidade do que será modelado. Navathe e Ramez (2005) afirmam que a modelagem conceitual é uma descrição concisa das informações que o software deverá possuir, de acordo com seus requisitos. É uma representação do que precisa ser realizado (não é a solução do problema).

Segundo Coronel e Rob (2011), o modelo conceitual traz algumas vantagens importantes: Fornece a visão de nível macro, de forma simplificada e independente de hardware e software, não sendo necessária a realização de adaptações em função de determinado SGBD ou equipamento que serão adotados posteriormente.

Modelagem Lógica

O modelo lógico do banco de dados é a etapa em que mapeamos o conceito de modelos de entidade e relacionamento com o foco na criação do banco de dados. Nesta etapa as entidades são transformadas em tabelas para armazenas as informações, são estabelecidos os relacionamentos, decididas as regras e determinados os tipos de dados para cada campo da tabela, como descrevem Korth, Silberschatz e Sudarshan (2012).

O modelo lógico se transforma em um modelo mais abstrato (conceitual) para um modelo com mais detalhes de implementação, ou seja, mais próximo do que será de fato implementado. Cougo (1997) define como modelo lógico de dados aquele em que os objetos, suas características e seus relacionamentos sejam representados de acordo com as regras de implementação e limites impostos por alguma tecnologia de determinado SGBD.

Modelagem Física

Na última fase do projeto do banco de dados é realizada a modelagem física. Navathe e Ramez (2005) afirma que é justamente nesta fase que são determinadas as estruturas de armazenamento interno, as chaves (ou índices) e estabelecidos os diversos caminhos de acessos a base de dados. Esse modelo descreve o detalhamento ao nível do SGBD, nível físico de criação dos componentes do banco de dados.

Exemplificando os Conceitos

Uma escola de ensino fundamental bilíngue necessita de um software para seu gerenciamento acadêmico. Após algumas entrevistas, o analista de sistemas levantou os seguintes requisitos essenciais para o projeto de banco de dados:

- A escola possui diversos departamentos (são grandes áreas de conhecimento: Matemática, estudo da linguagem, etc.).
- Um departamento pode oferecer diversas disciplinas, mas a disciplina irá pertencer somente a um departamento.
- Um aluno pode estar matriculado em um único curso.
- Uma mesma disciplina pode constar no currículo de diversos cursos.
- Todo professor pertence a um departamento e ele poderá lecionar em diversas disciplinas.

Modelo Conceitual

Professores
Código, nome, formação, endereço, telefone

Curso

Código, nome, sigla

Disciplinas
Código, denominação, sigla, ementa

Departamentos
Código, denominação

Aluno
Matrícula, nome, endereço, telefone, filiação e data de nascimento

As primeiras etapas da modelagem do banco de dados (conceitual e lógica) são de grande relevância para atender às necessidades do cliente, enquanto a última etapa, de modelagem física, está voltada diretamente ao SGBD escolhido para ser utilizado na criação do banco de dados. Cada etapa possui a sua importância, mas projetar um banco de dados é vital para o sucesso do software que está sendo desenvolvido.

Modelo de Entidade

O modelo de entidade – relacionamentos (ou MER) foi desenvolvido para aperfeiçoar o projeto do banco de dados, permitindo a especificação do modelo conceitual, conforme afirmam Korth, Silberschatz e Sudarshan (2012). É o modelo mais utilizado pelos sistemas gerenciadores de banco de dados e foi elaborado por Edgar F. Codd em 1970 mas foi a partir de 1987 que começou a ser adotado pelas empresas de desenvolvimento de software.

Modelagem Relacional

O modelo lógico, ou seja, o modelo relacional de um banco de dados é criado a partir do levantamento de requisitos e do modelo conceitual. O processo de mapeamento dos dados entre os modelos (conceitual-lógico) é chamado de modelagem relacional.

A abordagem relaciona, segundo Abreu e Machado (2009), parte do princípio que as informações em uma base de dados podem ser consideradas como relações matemáticas e devem ser representadas em formas de tabelas.

A representação gráfica da modelagem relacional é a forma de representação dos componentes do modelo lógico de um banco de dados. Esta representação é uma parte muito importante da compreensão do esquema do banco de dados. Uma representação simples e intuitiva é fundamental para o entendimento e comunicação das pessoas envolvidas na criação do modelo do banco de dados. De acordo com Korth, Silberschatz e Sudarshan (2012), existe uma série de notações alternativas para a realização da modelagem, sendo que as mais utilizadas são as de Peter Chen, IDEF1X, James Martin (com o famoso Pé de Galinha) e a notação UML.

Tabela

O modelo relaciona é um conceito matemático conhecido como relação, no qual dois conjuntos numéricos possuem seus termos relacionados entre si. No modelo conceitual, um conjunto é chamado de entidade, já no modelo lógico é chamado de tabela.

Cada tabela é definida com um conjunto de atributos que descrevem suas características particulares, esses atributos também são conhecidos como campos.

Cada linha de uma tabela representa um conjunto de campos e são conhecidos como registros ou tuplas. Uma tabela pode ter milhares de registros (milhares de linhas), quem limita a quantidade de linhas de uma tabela é o SGBD.

Um banco de dados é formado por um conjunto de tabelas que estão relacionadas entre si. Cada tabela do banco de dados deve ter um nome único e significativo, por exemplo: Uma tabela que guarda informações de automóveis, pode ter como nome “automóvel” e não “tabela_A”.

De acordo com Coronel e Rob (2011), as tabelas possuem algumas características. Veja elas:

- A tabela é vista com o uma estrutura composta de linhas e colunas (bidimensional).
- Cada linha ou registro representa uma única ocorrência da entidade no interior do conjunto da entidade.
- Cada coluna da tabela representa um atributo e possui nome diferente (dos demais atributos da mesma tabela).
- Cada intersecção entre linha e coluna representa um único valor (é o dado da tabela).
- Todos os valores em uma coluna devem possuir o mesmo formato.
- A ordem das colunas e das linhas é insignificante para um SGBD.
- Cada tabela deve representar um atributo ou uma combinação de atributos que identifique exclusivamente cada linha (chamado de chave, que será estudado mais adiante).

Entidades

Cada tabela que representa uma entidade do modelo conceitual pode ser classificada em: Entidade forte ou entidade fraca. Conforme Cougo (1997), esta distinção se dá com a análise de existências de duas condições básicas: Dependência de existência ou dependência de identificador. Uma entidade é fraca se um desses dois tipos de dependência existir no relacionamento entre duas entidades.

Korth, Silberschatz e Sudarshan (2012) e Navathe e Ramez (2005) classificam as entidades em alguns grupos:

- **Entidades Fortes:** É uma tabela autônoma que não depende de outra para sua existência. Aluno, curso, cliente, empresa, paciente. Na análise de requisitos é facilmente encontrada, visto que são substantivos fortes e significativos.
- **Entidades Fracas ou Dependentes:** É uma tabela que necessita de outra para realmente existir e somente existe por causa da entidade forte. A tabela dependente só há porque existe a tabela funcionário, pois, para que exista um dependente cadastrado, é preciso que tenha um funcionário que “possui” esse dependente. Se não existisse a tabela funcionário, certamente não existiria a tabela dependente.
- **Entidades Agregadas:** É criada quando temos um conjunto de campos que se repetem em mais de uma entidade. Aluno e professor possuem em comum dados do endereço, para evitar repetições, podemos criar uma nova entidade agregada chamada endereço para guardar os endereços.

- **Entidades Subordinadas:** Representa uma especialização em que uma entidade supertipo possui várias entidades subordinadas que são especializadas com atributos específicos. Podemos ter dois tipos de clientes: Pessoa física e pessoa jurídica. Ambos têm campos em comum que ficariam na entidade supertipo cliente e, nas entidades pessoa física e pessoa jurídica, somente estariam os campos específicos de cada tipo de cliente.

Relacionamento

As entidades podem ser conectadas. Essa conexão entre as entidades é chamada de relacionamento.

Um relacionamento descreve uma associação entre entidades como afirma Coronel e Rob (2011).

Os relacionamentos envolvendo tabelas fracas resultam em uma tabela associativa e que deve ser representada por meio de um losango com bordas duplas.

Quando temos um relacionamento entre duas entidades, o número de ocorrências de uma entidade que está associada, com ocorrências de outra entidade, determina o grau de relacionamento ou cardinalidade entre as tabelas, como afirma Abreu e Machado (2009).

O grau de relacionamento é a quantidade de entidades que estão ligadas ao relacionamento e pode ser de vários tipos. Veja abaixo:

- **Relacionamento Unário:** Uma entidade se relaciona com ela mesma.
- **Relacionamento Binário:** É um relacionamento que liga dois tipos diferentes de entidades. É o evento mais comum dos tipos de relacionamentos.
- **Relacionamento Ternário:** É um relacionamento em que três entidades estão conectadas por um mesmo relacionamento.
- **Relacionamento Quaternário:** É um relacionamento em que quatro tabelas estão conectadas.
- **Relacionamento n-ário:** É um relacionamento acima de quatro tabelas envolvidas. Este tipo de relacionamento é o menos aconselhável visto que a possibilidade de redundâncias no banco pode ser maior.

Cardinalidade

A cardinalidade atribui um valor específico ao relacionamento, expressando a faixa de ocorrências permitidas (mínimas e máximas) entre as tabelas e, de acordo com Coronel e Rob (2011), pode ser de vários tipos.

Autorrelacionada

O autorrelacionamento é um tipo de relacionamento unário, envolvendo somente uma tabela. Nele os elementos de uma entidade se relacionam a outros elementos dessa mesma entidade, por exemplo, com a tabela funcionário. Todo funcionário possui um chefe ou superior que, por sua vez, também é um funcionário e que supervisiona vários empregados.

Um-Para-Um

A cardinalidade um-para-um (1 para 1) tem como características que cada tabela terá somente uma única ocorrência da outra tabela, por exemplo, em uma agência de empregos um candidato pode cadastrar somente um currículo e o currículo pertence somente a um candidado.

Um-Para-Muitos

Na cardinalidade um-para-muitos (1 para N) ou muitos-para-um (N para 1), uma das entidades pode referenciar várias unidades da outra, porém, do outro lado só pode ser referenciada uma única vez. Por exemplo, o funcionário precisa informar a cidade de seu nascimento e ele só pode indicar uma úncia cidade. Mas a mesma cidade pode ser referenciada várias vezes por outros funcionários. Conforme a figura, devemos ler da seguinte forma: Do lado direito: Um funcionário pode estar associado a no mínimo 1 e no máximo 1 cidade. Do lado esquerdo: Uma cidade possui no mínimo 1 funcionário e no máximo N.

Muitos-Para-Muitos

No relacionamento muitos-para-muitos (N para N), cada entidade, de ambos os lados, pode referenciar múltiplas ocorrências. O relacionamento resultante da cardinalidade N para N geralmente é um verbo, por exemplo: Atender, consultar, realizar, entre outros. Por exemplo: Um aluno pode cursar vários cursos e o mesmo curso pode possuir vários alunos. De acordo com a figura, devemos ler da seguinte forma: Do lado direito: Em um curso pode cursar 0 ou N alunos. Do lado esquerdo: Um aluno pode cursar 1 ou N cursos.

A definição da cardinalidade é de fundamental importância para o sucesso do banco de dados. Após realizar uma modelagem de um banco de dados, considere sempre mostrar o diagrama para seus colegas, pois a opinião deles pode levantar problemas que na hora de modelar foram esquecidos. E, claro, o cliente deverá sempre ser consultado nos casos em que a cardinalidade for muito duvidosa.

Diagrama de Entidade-Relacionamento – DER

O diagrama de entidade-relacionamento tem como objetivo a preparação para a implementação física do banco de dados no sistema gerenciador de banco de dados. Sendo assim, agora, você entrará de fato no processo de modelagem de dados, criando os diagramas de entidade-relacionamento.

A fase de modelagem física requer a aplicação de comandos para a criação das tabelas e campos dependendo diretamente da modelagem lógica. É de fundamental importância ter o projeto lógico (o mais correto possível) para a criação física do banco de dados.

Modelo Lógico/Relacional

O modelo de dados lógico ou relacional trabalha com esquemas compostos de tabelas. Para que uma tabela realmente exista, é necessário que possua diversas propriedades, que nada mais são do que os seus atributos ou campos. Cada campo, por sua vez, possui uma descrição de seu tipo de dado e tem as seguintes características:

- Contém todas as tabelas e os relacionamentos entre elas.
- Descrição de todos os atributos de cada tabela.
- Identificação de um atributo chave para cada tabela.
- Determinação de relacionamentos por meio de chaves.

Definições

Korth, Silberschatz e Sudarshan (2012) afirmam algumas definições no modelo relacional:

- **Relação:** O termo relação é designado para se referir a uma tabela.
- **Tupla:** Para a linha é utilizado o termo tupla, referindo-se a uma determinada linha da tabela.
- **Atributo:** O atributo é conhecido como a coluna de uma tabela.
- **Instância de Relação:** O termo instância de relação é designado a um determinado conjunto de tuplas ou a um determinado número de registros (algumas linhas selecionadas de uma tabela).

Chave de Tabela

Em um determinado banco de dados existe uma tabela de clientes com centenas de registros cadastrados. Será que existe a possibilidade de procurar um determinado cliente pelo seu nome e aparecer mais de um registro como resultado? Há possibilidade de duplicidade de nomes em uma tabela? Se a possibilidade de repetição existe, como garantir que o cliente certo seja encontrado?

Para solucionar esse dilema, no banco de dados relacional, há a necessidade de estabelecer um ou mais campos para ser uma chave de identificação do registro armazenado. Alguns valores do registro até poderão ser repetidos, como duas ou mais pessoas com exatamente o mesmo nome, mas, obrigatoriamente, deve haver um campo na tabela que nunca se repete. Este campo será o campo chave de tabela.

Coronel e Rob (2011) explicam que uma chave consiste em um ou mais atributos que determinam a existência de outros. Elas são utilizadas para estabelecer os relacionamento entre as tabelas e para estabelecer a integridade referencial dos dados. Os tipos de chaves em uma tabela podem ser:

- Chave primária.
- Chave concatenada ou composta.
- Chave substituta ou surrogada.
- Chave secundária.
- Chave estrangeira.

Chave Primária

Um dos fundamentos primordiais de um banco de dados é que cada tabela exista uma chave primária, que também é conhecida como Primary Key (ou somente PK).

Devemos escolher um dos campos da tabela para ser a chave primária. No exemplo foi escolhido o campo CPF como chave primária, pois temos certeza que uma pessoa não pode ter o CPF igual ao de outra pessoa.

Na forma textual, a chave primária deve sempre ficar em evidência, em negrito, sublinhada e com sinal # na sua frente. Observe como ficaria a tabela: Cliente (#CPF, Nome, RG, Dt Nasc, Cidade). A ordem dos campos não prejudica em nada os campos das tabelas, porém, para fins didáticos e para facilitar a identificação, nos exemplos, a chave primária ficará sempre como o primeiro campo da tabela.

Chave Composta

A figura mostra o uso do RG, como esse número pode se repetir dependendo do órgão expedidor, não seria uma boa escolha para a chave primária.

Assim, caso quiséssemos realmente usar o RG, deveríamos solicitar também o órgão expedidor do documento, essas duas informações em conjunto não se repetem, e é justamente esse o conceito de chave concatenada ou composta. Um ou mais campos que não podem se repetir.

A forma textual da tabela cliente demonstrada ficaria da seguinte forma: Cliente (#RG, #Orgão Expedidor, Nome, CPF, Dt Nasc).

Chave Surrogada

A chave substituta ou surrogada e que também é conhecida como surrogate key é uma chave primária criada exclusivamente para impedir que os registros da tabela venham a repetir e que não tenha um campo para ser a chave primária. Este tipo de chave, é um campo criado de valor inteiro e ele tem um autoincremento.

A cada registro adicionado na tabela, o campo recebe um incremento e assim sucessivamente (o próprio SGBD se encarrega do incremento). Este tipo de campo numa pode ser alterado e também nunca pode ser reaproveitado, em casos em que o registro for apagado, não há como reaproveitar o valor de uma chave substituta.

A forma textual da tabela: Encomenda (#Código, Encomenda, Quantidade, Preço Unitário).

Chave Secundária

De acordo com Coronel e Rob (2011), a chave secundária é uma chave utilizada para fins de recuperação de informação. É uma chave que auxilia na recuperação de um registro caso, por exemplo, um paciente de um consultório tenha o CPF como chave primária mas não tem acesso ao número. Com a chave secundária, podemos usar algum mecanismos de busca utilizando o campo do sobrenome ou a data de nascimento para realizar uma pesquisa no banco de dados.

Chave Estrangeira

Korth, Silberschatz e Sudarshan (2012) descrevem a chave estrangeira (também conhecida como foreign key – FK) como sendo uma chave primária de outra tabela. É com esta chave que ocorrem os relacionamentos entre as tabelas de um banco de dados.

Com o relacionamento entre as tabelas cliente e cidade, teremos a seguinte forma textual entre as duas tabelas: Cidade (#Cod Cidade, Cidade) Cliente (#CPF, Nome, Dt Nasc, &Cod Cidade).

Integridade Referencial

Segundo Coronel e Rob (2011), a integridade referencial tem como exigência básica a sua existência em uma outra tabela, como chave primária. Estabelecer a integridade referencial é justamente garantir que, ao relacionar uma tabela com outra, haverá a garantia de que a chave estrangeira tenha sido cadastrada (primeiramente) como chave primária de outra tabela que compõe o relacionamento. E, para estabelecer a integridade referencial precisamos seguir alguns passos:

1. Observar no diagrama os relacionamentos. Procure por cardinalidades do tipo N nas tabelas.
2. Existe um ou mais cardinalidades do tipo N? Se sim, haverá chaves estrangeiras. Podem haver vários Ns nas tabelas e, consequentemente, várias chaves estrangeiras.

3. A tabela do lado 1 deverá receber novos campos, para criar o relacionamento. Insira a chave primária da tabela correspondente ao relacionamento do lado N.

Notações

Para representar as cardinalidades no modelo gráfico de um diagrama de entidade-relacionamentos, podemos utilizar diversas notações. Uma notação gráfica é a forma de como algo é criado, desenhado ou projetado, são os padrões que deverão ser adotados no desenvolvimento de algo. Cougo (1997) descreve algumas notações.

Ciclo de Vida de um Banco de Dados

Em um projeto de banco de dados, assim como no desenvolvimento de qualquer software, há um ciclo de vida que irá determinar o início do projeto até o seu final (que neste caso é a manutenção ou a evolução do banco de dados).

Veja as seis fases e suas respectivas ações do ciclo de vida de um banco de dados:

- **Estudo Inicial do Banco de Dados:** Estudo dos requisitos do problema e suas restrições, definição dos objetivos, escopo e fronteiras do banco de dados.
- **Projeto do Banco de Dados:** Criação do projeto conceitual, escolha do SGBD que deverá ser usado, criação do projeto lógico e físico do banco de dados.
- **Implementação e Carga:** Instalação do SGBD, criação do banco de dados, carregamento ou conversão dos dados que serão armazenados no banco.
- **Teste e Avaliação:** Realização de testes na base de dados para encontrar possíveis erros.
- **Operação:** O banco entra em funcionamento nos aplicativos desenvolvidos em paralelo.
- **Top-Down:** Assim que entra em operação, o banco de dados deve sempre receber manutenção para ficar o máximo possível em plena operação, e a evolução do banco de dados acontece assim que novas necessidades do usuário surgem.

Estratégias de Modelagem

Como estratégia de modelagem em um diagrama de entidade-relacionamento há duas abordagens clássicas tradicionais que podem ser adotadas. São eles:

- **Top-Down:** Identifica-se primeiro os conjuntos de dados e, após, são definidos elementos de cada um desses conjuntos. O processo envolve a identificação de diferentes tipos de entidades e a definição de cada atributo. Esta técnica é utilizada em bancos de dados maiores.
- **Bottom-Up:** Primeiramente são identificados os elementos de dados, ou seja, os itens, e em seguida os itens são agrupados em conjunto de dados. Neste caso os atributos são identificados primeiro, e depois, ao agrupá-los, teremos as tabelas. Geralmente, esta técnica é utilizada em bancos de dados pequenos.
- **Middle-Up-Down:** As abordagens top-down e bottom-up acabam se complementando, muitas vezes um analista ou projetista de banco de dados acaba aplicando as duas técnicas no mesmo banco de dados a ser modelado, surgindo então uma abordagem mista que foi denominada de middle-up-down.

Modelagem Conceitual

A modelagem conceitual em um projeto de banco de dados é considerada de alto nível, pois possui como finalidade ser de fácil compreensão entre os usuários envolvidos na modelagem do banco. Assim que o modelo lógico começa a ser implementado, o modelo conceitual servirá de apoio à construção do esquema do banco de dados. Durante ou mesmo ao término do esquema conceitual, as operações básicas do modelo de dados podem ser usadas para especificar as operações de alto nível do usuário e servem para verificar se o modelo possui todos os requisitos listados pelo cliente.

O foco da modelagem conceitual é detalhar e discutir o funcionamento do negócio do cliente, e não o uso de determinada tecnologia, descartando informações de como os dados serão armazenados e depois recuperados em bancos de dados.

A fim de criar um modelo lógico do banco de dados mais coeso, são necessárias várias revisões na descrição do modelo conceitual (de alto nível):

- **Tabelas:** Em substantivos, objetos reais, objetos que possam armazenar informações.
- **Campos:** Em características específicas de algum objeto ou adjetivos.
- **Relacionamentos:** Em verbos que “ligam” uma tabela à outra.
- **Cardinalidades:** É a quantidade de vezes que cada tabela pode estar relacionada com outra.

Algumas normas precisam ser adotadas durante a criação do modelo lógico do banco de dados, na criação do diagrama de entidade-relacionamentos.

Veja as principais normas:

- **Em Casos de Relacionamento 1 para N:** A chave primária do lado 1 sempre deverá estar na tabela do lado N como uma chave estrangeira.
- **Em Casos de Relacionamento N para N:** O relacionamento passa a ser implementado como tabela própria e que possui campos específicos, relacionados entre as duas tabelas que deram origem a esta nova tabela, que se chama tabela associativa.
- **As Tabelas Devem Ter o Número de Chaves Primárias Reduzido ao Mínimo Possível:** Ou seja, sempre que possível, uma tabela deverá ter somente um identificador único, evitando chaves alternativas.

Dicionário de Dados

Todo modelo de banco de dados produzido precisa ser documentado. Isso é importante por vários motivos: Organização, apresentação e controle (caso seja necessária a inclusão de campos ou mesmo entidades). Após a elaboração do DER, deve ser realizado o mapeamento das entidades, juntamente com a documentação completa das tabelas, neste caso, um dicionário de dados.

É necessário criar o dicionário de dados para estabelecer uma padronização e uma documentação sobre cada tabela criada.

Um dicionário de dados deve ter várias informações, como:

- Descrição dos nomes de: Tabelas, relações e atributos.
- Tipos dos dados (seu domínio) e seus respectivos tamanhos.
- Descrição detalhada das chaves utilizadas.
- Nomes dos usuários com suas permissões sobre a tabela.

O nível de detalhamento do dicionário de dados é opcional, mas ele acaba se tornando um documento fundamental, sendo muito requisitado quando o banco de dados apresenta problemas e precisa sofrer manutenção.

Exemplificação

Uma imobiliária especializada em aluguel de casas e apartamentos localizados no litoral de Santa Catarina necessita de um software para ajudar no gerenciamento de aluguéis e oferecer melhores ofertas para seus clientes.

Após diversos contratos com a imobiliária, foram estabelecidos os 5 requisitos que deveriam ser atendidos pelo banco de dados:

1. Cada imóvel deverá ter registrado: Seu tipo (casa ou apartamento), quantidade de quartos e banheiros, se o imóvel possui vista para o mar, preço da diária.
2. As informações dos proprietários e dos inquilinos deverão ser armazenadas separadamente. Os proprietários podem ter vários imóveis que podem ser alugados para vários inquilinos.
3. Além das informações sobre o município ao qual o imóvel pertence, deverá também ser informado o nome da praia mais importante (próxima) ao imóvel.
4. Os móveis são todos os itens que compõe a mobília. Os itens mais verificados são: Cama, geladeira, freezer, televisor, ar-condicionado, entre outros. Neste caso, é importante que seja informada a quantidade de cada item.
5. Deverá ser realizado e registrado um contrato que deverá ser exclusivo para os alugueis com os inquilinos e os imóveis alugados.

DER Imobiliária

O diagrama de entidade-relacionamento demonstra uma solução inicial para a modelagem do banco de dados da imobiliária.

Para melhor entendimento do diagrama da imobiliária, considere os seguintes itens:

- **Entidades:** Com os requisitos listados pode-se identificar primeiramente as entidades que se destacam: Imóvel, tipo de imóvel, cidade, praia, proprietário, inquilino, contrato de aluguel e mobília.
- **Notação:** Para expressar graficamente o diagrama de entidade-relacionamentos, utilizamos a notação “pé de galinha”.
- **Cardinalidade:** O lado das setas ligando as tabelas que apresenta três pontar representa o lado N, o que possui uma ponta é o lado 1.
- **PK – FK:** As siglas PK e FK em alguns campos representam, respectivamente: Primary key ou a chave primária da tabela. Foreign key ou a chave estrangeira da tabela.

Nesta aula você pôde conhecer mais alguns aspectos do processo de modelagem de dados, como: As estratégias de modelagem top-down e bottom-up e a documentação do desenvolvimento de um diagrama de entidade-relacionamentos. Em sua vida profissional você conhecerá e acabará usando modelos diferentes, que foram adaptados pelas empresas de acordo com suas necessidades de desenvolvimento.

Orientação a Objetos

Atualmente a maioria das linguagens de programação é orientada a objetos, como as linguagens Java e C#, que apesar de serem diferentes, compartilham dos mesmos conceitos e paradigmas da programação orientada a objetos, tendo como base a reutilização de componentes em outros softwares, pois pode-se reaproveitar códigos, classes, partes inteiras de um sistema em outro sistema. Assim, um software desenvolvido nesta linguagem utilizando classes pode perfeitamente armazenar as informações em bases de dados relacionais.

Para fazer a modelagem com UML, uma linguagem de modelagem unificada muito utilizada na programação orientada a objetos, é preciso planejamento e padronização dos processos de desenvolvimento de um software.

Objeto

Um objeto pode ser uma pessoa, alguma coisa, um processo real ou abstrato e que possua informações e funcionalidades. Pode representar um cliente, uma bicicleta, um DER de um sistema, um botão de um software.

Classe

Uma classe é o agrupamento das informações comuns de um objeto. Ela deve ser exclusivamente criada para guardar informações de um determinado objeto.

Uma classe pode ser conceituada como um tipo de dado.

Ela é formada por dados e comportamentos, dividida em três partes: Nome, atributos e métodos.

- **Nome:** Refere-se ao que a classe representa, ao ler o nome da classe já devemos identificar o que ela irá armazenar.
- **Atributos:** São as características do objeto utilizados para a definição dos dados.
- **Métodos:** São os comportamentos que um objeto poderá assumir, funções que manipulam os atributos.

Instância

Uma instância de uma classe é a criação de um objeto baseado em uma determinada classe. Ao instanciarmos um objeto de uma classe, criamos esses objetos na memória.

UML

“Mas o que isso tem a ver com a modelagem de dados?”

A resposta é que os objetos serão persistidos em algum banco de dados e por isso também precisam ser modelados de acordo com as regras que estamos aprendendo. Para auxiliar nesse processo de modelagem de classes existe a Linguagem de Modelagem Unificada – UML (Unified Modeling Language), uma ferramenta que auxilia na modelagem de sistemas orientados a objetos.

A UML originalmente foi proposta para realizar a modelagem de sistemas orientados a objetos, mas ela pode ser utilizada como uma notação em projetos de banco de dados, padronizando a modelagem de todo o sistema.

Existem diversos diagramas que compõem a linguagem UML, entre eles o diagrama de classes, que é muito semelhante ao diagrama entidade-relacionamento e pode ser usado para representar o projeto lógico de um banco de dados.

A cardinalidade utilizada em um diagrama de classe é semelhante ao diagrama de entidade-relacionamentos, porém, podemos determinar o número exato de ocorrências: 1..5 – uma ocorrência no mínimo e cinco no máximo e o N (de muitos no DER) é representado pelo sinal de asterisco (*).

Quando uma classe é instanciada (ação de criá-la), tem-se um objeto na memória de trabalho do computador. Nesse caso, os atributos da classe são “preenchidos” com dados, no diagrama de objetos, para mostrar as informações que serão armazenadas na classe.

Na programação orientada a objetos as classes podem se relacionar com outras classes através de hierarquias. Uma subclasse pode herdar as características de outra classe (neste caso seria a classe mãe ou superclasse). Na estrutura de herança as classes compartilham as suas funções e características comuns e as subclasses podem receber outras particularidades exclusivas.

A herança tem como princípio fundamental a possibilidade de criar subclasses que podem herdar características da classe mãe. Esta mesma classe analogia pode ser aplicada nos modelos de entidade-relacionamentos, e então temos dois processos:

- **Generalização:** Processo em que várias entidades (tabelas) são agrupadas em uma única entidade genérica.
- **Especialização:** É o processo inverso, no qual novas entidades são criadas com atributos que acrescentam detalhes para a entidade genérica.

Em um diagrama de entidade-relacionamento generalização e especialização são um tipo de relacionamento entre entidades que determina que uma entidade contém a outra, isto quer dizer que uma entidade superior contém um ou mais conjuntos de entidades inferiores.

Uma estrutura de generalização e especialização pode ser classificada em parcial ou total.

- **Generalização-Especialização:** Há uma tabela generalizada chamada Funcionário e as tabelas especializadas chamadas Professor e Secretária. Nesse cenário as entidades Professor e Secretária possuem atributos da entidade Funcionário mais as suas específicas. A estrutura de generalização e especialização é representada pelo triângulo que une as entidades.
- **Generalização-Especialização Parcial:** Uma generalização e especialização parcial, no exemplo com um “p” ao lado do triângulo, indica que nem todo funcionário é professor ou secretária e neste caso não é toda ocorrência da entidade generalizada que possui uma entidade especializada correspondente.
- **Generalização-Especialização Total:** Podemos ter uma generalização e especialização total quando a cada ocorrência da entidade generalizada deva existir obrigatoriamente uma existência da entidade especializada. No exemplo temos um “t” que indica a totalidade e uma seta dupla apontando para a entidade generalizada, indicando que obrigatoriamente um funcionário será ou professor ou secretária.

Exemplificação

Para uma determinada universidade é necessário guardar as informações das disciplinas ministradas por professores e quais as disciplinas que um determinado aluno cursa.

Modelo Conceitual

Para melhor entendimento do modelo conceitual com generalização e especialização, considere as seguintes entidades:

- **Tabela Funcionário:** Nome, endereço, RG, CPF, data de nascimento, data de admissão, data de demissão, salário, nome da mãe, nome do pai.
- **Tabela Professor:** Nome, endereço, RG, CPF, data de nascimento, nome da mãe, nome do pai, valor da hora da aula, quantidade de horas de aula.
- **Tabela Aluno:** Nome, endereço, RG, CPF, data de nascimento, data de entrada na faculdade, data de formatura, salário, nome da mãe, nome do pai.
- **Tabela Disciplina:** A tabela Disciplina estará relacionada com a tabela Professor e com a tabela Aluno.
- **Tabela Pessoa:** A tabela Pessoa foi criada para inserir os atributos que se repetem nas tabelas Funcionário, Professor e Aluno, quais sejam: Nome, endereço, RG, CPF, data de nascimento, nome da mãe, nome do pai.

O diagrama de classes pode ser utilizado como ferramenta gráfica para criar o modelo lógico de banco de dados.

A modelagem de dados abstraiu os conceitos da programação orientada a objetos como forma de evolução e acompanhamento dos conceitos mais atuais de programação. Nesta aula você conheceu a aplicação de herança (generalização e especialização) no modelo de entidade-relacionamentos.

Ferramentas CASEs

As ferramentas CASEs (Computer Aided Software Engineering), ou Engenharia de Software Auxiliada por Computador, são ferramentas que apresentam uma série de serviços que auxiliam no desenvolvimento de softwares e podem minimizar o tempo de desenvolvimento de softwares e podem minimizar o tempo de desenvolvimento do software modelado.

As primeiras ferramentas Cases surgiram na década de 1980 e eram classificadas em:

- **Lower CASE:** Com suporte nas fases de análise e projeto de sistemas.
- **Upper CASE:** Com suporte nas fases de construção e análise de sistemas.

Hoje em dia as ferramentas Cases são classificadas como Integrated CASE com a união das ferramentas Lower CASE ou Upper CASE, atendendo praticamente a todas as fases de um projeto de sistemas.

Todos os tipos de ferramentas CASEs possuem em comum a possibilidade de representação gráfica de elementos do projeto e que podem ser: O diagrama de entidade-relacionamentos, os diagramas de classes, casos de usos, etc.

As ferramentas CASEs são utilizadas para automatizar várias tarefas, por exemplo:

- **Geração de Códigos:** De forma automática, os códigos podem ser gerados a partir do diagrama gráfico.
- **Geração de Documentação:** Permitindo padronização nos processos.
- **Efetuar Testes:** Possibilitar validações nas especificações formais de desenvolvimento.

- **Geração de Relatórios:** Permitindo o acompanhamento do planejamento e do gerenciamento do projeto.

As ferramentas CASEs para banco de dados possuem as seguintes características, conforme Coronel e Rob (2011):

- **Diagramas:** Suporte à criação de diagramas gráficos.
- **Notação:** Utilização de alguma notação de modelagem de banco de dados.
- **Scripts:** Geração de scripts SQL (Structured Query Language – Linguagem de Consulta Estruturada).
- **Forward Engineer:** Permite a partir do DER (modelo gráfico) conectar de forma automática o banco de dados e criar automaticamente o meio físico.
- **Reverse Engineer:** Permite que a partir do modelo físico criado no banco de dados, possa ser gerado o modelo gráfico (o DER) do banco de dados.
- **Dicionário de Dados:** Documentação, conforme os atributos são criados nas tabelas, a ferramenta CASE já cria o dicionário de dados de forma automática.

Exemplos de Ferramentas CASEs

Existem várias ferramentas CASEs disponíveis de forma freeware (com opções básicas) e há as ferramentas proprietárias (pagas) com muitos recursos.

Veja a seguir algumas delas.

Astah

O Astah é uma ferramenta CASE para criar diagramas UML. Ela é ideal para desenvolvedores Java, pois gera os scripts em Java, acelerando o processo de desenvolvimento do software. Possui duas versões:

- **Community:** Gratuita para projetos UML (com algumas limitações).
- **Professional:** Versão completa e paga (ou disponível de forma trial).

Uma das vantagens da ferramenta Astah é a possibilidade de criar automaticamente o dicionário de dados, basta exportar para a ferramenta Microsoft Excel e o dicionário de dados é gerado com todas as tabelas.

Workbench

O MySQL Workbench é de propriedade da empresa Oracle e é uma ferramenta CASE gratuita que gera scripts para o SGBD MySQL. O foco desta ferramenta é a modelagem física do banco de dados, acelerando o processo de criação da base de dados. No relacionamento utiliza a notação “pé-de-galinha”.

Draw.IO

Draw.IO é uma ferramenta fácil e que tem como requisito estar conectado à internet. A ferramenta ainda disponibiliza uma grande quantidade de templates de modelos para servirem de exemplos de modelagem. O uso da ferramenta é bem intuitivo, do lado esquerdo estão as categorias de diagrama, você pode acessar a “Entity Relation” para ter acesso às opções de tabela, e basta clicar em uma opção para adicioná-la à área de trabalho. Os campos podem ser editados diretamente na tabela

adicionada. Para relacionar as tabelas escolha as opções de seta e ligue a chave primária diretamente sobre a chave estrangeira.

Lucidchart

Outra ferramenta CASE online é Lucidchart. No site é possível criar de forma gratuita um modelo com até 60 objetos, acima deste limite é necessário pagar. Essa ferramenta é ideal para projetos pequenos e médios. Nela, você utilizará as ferramentas da categoria padrão e entidade-relacionamentos. Na categoria de entidade-relacionamentos você deverá selecionar o modelo de tabela que desejar e inserir os campos diretamente na figura da tabela. Para criar o relacionamento entre as tabelas, você usará a ferramenta de "seta" na categoria, ligando sempre a chave primária com a sua chave estrangeira (na outra tabela). No exemplo foi utilizada a notação "pé-de-galinha". Um dos destaques da ferramenta Lucidchart é a possibilidade de gerar scripts para os seguintes bancos de dados: MySQL, PostreSQL, SQL Server e Oracle. Também é possível exportar o DER para uma imagem.

Exemplificação

"Após diversas situações de emergência devido a inundações, geadas e vendavais, a defesa civil de uma cidade no sul do país precisa de um banco de dados com informações sobre as vítimas, para poder controlar as emergências que ocorreram e a quantidade de pessoas envolvidas. Uma emergência pode atingir cidades vizinhas e a defesa civil pode intervir em outras cidades."

São 5 requisitos para realizar o banco de dados:

1. É necessário anotar o endereço da família em situação de risco e seu bairro.
2. Deve haver um cadastro de cada membro de uma família, seus dados pessoais e uma foto são fundamentais.
3. A família poderá ser classificada por tipo: Em risco, alto risco e extremo risco.
4. Deve haver uma tabela para cadastro das emergências classificadas como: Inundação, vendaval, desmoronamentos, entre outras opções.
5. Uma família pode sofrer com diversas emergências e uma emergência pode atingir diversas famílias. É necessário armazenar a data da emergência e os dados causados pela emergência ocorrida.

Diagrama Entidade-Relacionamentos

Um diagrama de entidade-relacionamentos geralmente possui muitas entidades e utilizar uma ferramenta CASE para criar as tabelas e relacionamentos facilita muito o trabalho do desenvolvedor.

Para criar o diagrama de entidade-relacionamentos, foi utilizada a ferramenta CASE online Lucidchart.

Conceitos de Normalização de Dados

Os danos que a redundância pode causar e que geram mais problemas é a repetição da mesma informação em várias tabelas, ocasionando, além de duplicidade, possíveis erros em relatórios. E pior que a repetição é a inconsistência no banco de dados, que nada mais é do que a mesma informação sendo cadastrada de forma errônea.

A seguir, observe as três tabelas: Cliente, Fornecedor e Funcionário.

Resolvendo a Redundância e a Inconsistência Entre as Tabelas

No caso do campo Cidade é mais simples criarmos uma tabela chamada Cidade, relacionando as tabelas que precisam dessa informação.

Redundância Controlada

Às vezes, é interessante manter uma informação redundante no banco de dados. Por questões de desempenho de alguma pesquisa ou software, a redundância controlada pode ocorrer. Ela é recomendada quando o campo recebe uma grande quantidade de consultas e poucas alterações. É uma redundância que sabemos que existe, mas é necessária.

Observe que, na tabela Funcionário, há o código do departamento (CodDepartamento) e o nome do departamento (NomeDepartamento), fato este que causa a redundância.

Normalização

A normalização pode ser definida como um processo de organizar os dados em um modelo de banco de dados. Para realizar essas tarefas, são criadas tabelas e estabelecidos relacionamentos entre tabelas de acordo com as regras empregadas, a fim de proteger os dados e tornar o banco de dados mais maleável, adaptando-se para acabar com a redundância e a dependência inconsistente (campo errado na tabela errada) dos dados que serão armazenados no banco de dados.

A normalização de dados é um processo rígido e formal que deve ser seguido passo a passo examinando os campos de uma tabela, com o objetivo de evitar irregularidades observadas na inclusão, exclusão e alteração de registros.

Regras Básicas da Normalização

Uma das regras básicas da normalização é verificar se determinado campo realmente pertence à tabela.

Tabela Produto não Normalizada

Na tabela Produto não normalizada, observe que nos campos TipoProduto e Fornecedor a mesma informação foi inserida erroneamente, entre os registros.

No caso da tabela de produtos, há os campos:

- **TipoProduto:** Podemos criar uma tabela para armazenar os tipos de produtos.
- **CodForn e Fornecedor:** Devemos criar uma tabela chamada fornecedor.

Tabela Produto Normalizada

Veja, a seguir, a normalização com a criação de novas tabelas e campos como chaves estrangeiras:

- **Tabela Produto:** Foram retirados da tabela Produto alguns campos, para que esta seja normalizada. As informações que permanecem e que agora fazem parte de outra tabela ficaram como chaves estrangeiras das tabelas.

- **Tabela TipoProduto:** A tabela TipoProduto foi criada e será nesta que todos os tipos de produtos deverão ser criados, para depois serem escolhidos na tabela Produto (via chave estrangeira).
- **Tabela Fornecedor:** A tabela Fornecedor foi criada para receber as informações dos fornecedores e, através da chave estrangeira na tabela Produto, vincularemos o Fornecedor ao Produto, evitando que o nome deste fornecedor seja cadastrado de forma errada.

Geralmente, os analistas de sistemas já criam esses tipos de tabelas relacionadas automaticamente a um modelo de banco de dados. O exemplo é para dar ênfase à necessidade da normalização, principalmente quando estamos começando na modelagem de banco de dados.

Técnicas de Normalização

As técnicas de normalização se baseiam na dependência funcional entre os atributos de uma entidade do banco de dados e nas chaves primárias. A forma normal é uma regra que deve ser satisfeita por uma entidade para que ela seja avaliada como uma tabela “projetada com exatidão”. São várias formas normais, com regras que vão se tornando mais rigorosas, com o objetivo de averiguar nas tabelas a existência de redundância ou dependências funcionais. No entanto, pelo menos quatro formas normais são consideradas essenciais para a construção de um bom projeto de banco de dados:

- Primeira forma normal ou 1FN.
- Segunda forma normal ou 2FN.
- Terceira forma normal ou 3FN.
- Quarta forma normal ou 4FN.

Dependência Funcional

A dependência funcional consiste em uma restrição entre dois ou mais conjuntos de atributos de uma mesma tabela ou relacionamento (Alves, 2014).

Um exemplo de dependência funcional é a tabela Aluno. Nela temos a matrícula e o nome do aluno. O nome do aluno depende diretamente da matrícula do aluno.

Para melhor entendimento do esquema de relacionamento, explore a seguir:

- Os três primeiros campos, MatriculaAluno, CodigoCurso e CodigoDisciplina, representam a chave primária da tabela (no caso, uma chave composta ou concatenada). Neste exemplo o conjunto dos três campos formam um índice único (formando a chave primária).
- Campo MatriculaAluno: Possui como dependência funcional os campos NomeAluno e DataMatricula.
- Campo CodigoCurso: Possui como dependência funcional o campo NomeCurso.
- Campo CodigoDisciplina: Possui como dependência funcional o campo NomeDisciplina.
- Campos MatriculaAluno, CodigoCurso, CodigoDisciplina: Determinam o valor da NotaProva.

A dependência funcional pode ser classificada em: Transitiva ou indireta, total ou parcial:

- **Transitiva ou Indireta:** Acontece quando um determinado campo da tabela, além de depender da chave primária da tabela, depende de outro campo ou de outros campos que são

integrantes da mesma tabela. Exemplo de tabela que precisará sofrer um processo de normalização para resolver essa dependência entre os campos com esse tipo de dependência.
- **Total ou Completa:** Ocorre quando um atributo que não faz parte da chave primária depende diretamente de todos os outros atributos que fazem parte da chave primária. Sempre ocorre quando a tabela possui chaves concatenadas (mais de uma chave primária). Exemplo em que os campos Cidade e o Bairro são chaves concatenadas e o campo FiscalResponsavel depende dos dois campos para “existir”.
- **Parcial:** Ocorre quando um campo ou atributo que não faz parte da chave primária tem dependência funcional de apenas alguns dos atributos que fazem parte da chave primária. Exemplo no qual a tabela Medição de Temperatura possui três chaves primárias: UF, Cidade e Regiao. O campo Temperatura possui uma dependência funcional parcial apenas de parte das chaves primárias, visto que se o campo Regiao não existisse ou se fosse removido não afetaria o campo Temperatura.

Normalizar é converter uma tabela em tabelas de graus e cardinalidades menores até que quase não haja redundâncias nem dependências funcionais. Você poderá perceber que o objetivo principal da normalização não é eliminar totalmente as inconsistências, mas sim controlá-las. Identificar as dependências funcionais nas tabelas é o primeiro passo para saber que precisamos normalizar as tabelas em um banco de dados.

Normalização de Dados

Um banco de dados mal projetado pode exigir muito tempo de espera por resultados e o pior poderá acontecer: Erros duplicados e imprecisos. A normalização de tabelas, segundo Coronel e Rob (2011), é um método para avaliar e corrigir estruturas de tabelas com o propósito de reduzir as redundâncias de dados, diminuindo desta forma a possibilidade de erros e anomalias em uma tabela.

Para atingir os objetivos da normalização, os autores indicam que as tabelas precisam ter as seguintes propriedades:

- **Assunto da Tabela:** Cada tabela deverá tratar somente de um único assunto, por exemplo: Uma tabela com informações sobre remédio não poderá ter informações de um médico.
- **Campos nas Tabelas:** O mesmo campo não poderá ser armazenado, desnecessariamente, em mais de uma tabela, sendo esta uma garantia que não será necessário atualizar o mesmo campo, em mais de uma tabela.
- **Chave Primária:** Os campos de uma tabela são dependentes da chave primária desta tabela e de mais nenhum campo.
- **Integridade e Consistência:** A tabela deverá estar livre de anomalias de inserção, atualização e exclusão, garantindo a integridade e a consistência dos dados, por exemplo: Na tabela Cliente será necessário informar a cidade de seu nascimento, para tanto não devemos deixá-lo informar a cidade, mas escolher a cidade dentre as cidades previamente cadastradas ou através de uma busca CEP (que trará o endereço completo).

Para aplicar as regras da normalização, um dos alvos a ser observado são os campos (ou atributos) que fazem parte das tabelas. Conforme Korth, Silberschatz e Sudarshan (2012), podemos classificar os atributos de um modelo entidade-relacionamentos como:

- **Atributo Simples ou Atômico:** É o atributo que não é divisível e que possui um sentido único, como o RG ou o CPF de uma pessoa, que não podem ser divididos em dois outros campos.

- **Atributo Composto:** É um atributo que pode ser dividido em várias partes. Um bom exemplo é o endereço. Podemos dividir esse atributo em: Rua, número, complemento, bairro.
- **Atributo Monovalorado:** É um atributo que possui apenas um valor para a tabela, por exemplo, a matrícula de um aluno, dado este que não poderá se repetir na tabela.
- **Atributo Multivalorado:** É um atributo que pode receber mais de uma informação. O melhor exemplo é o telefone, que pode receber mais de um valor.
- **Atributo Derivado:** O valor deste atributo pode vir de outra tabela ou de outros campos. Por exemplo, para um médico, a idade (em anos e dias) é necessária, pois ela pode ser calculada a partir da data de nascimento e da data de atendimento no ato da consulta médica.
- **Atributo Chave:** É o atributo escolhido ou criado para indicar o registro (a linha) da tabela.

Primeira Forma Normal – 1FN

A 1FN possui a seguinte regra:

- Uma tabela estará na 1FN, se e somente se todos seus atributos forem atômicos, não possuindo grupos repetitivos ou colunas que possuam mais de um valor.

Para estar na 1FN os seguintes passos devem ser realizados:

- Identificar a chave primária da tabela.
- Identificar a coluna que possua dados repetidos.
- Remover a coluna que tenha dados repetidos.
- Criar uma nova tabela para armazenar os dados repetidos.
- Criar um relacionamento entre a tabela que está sendo normalizada e a sua tabela secundária.

Para deixar a tabela Funcionario na Primeira Forma Normal (1FN):

- **Chave Primária:** É melhor criar um campo novo e indicar como chave primária. Uma boa sugestão é o campo: Matrícula ou código do funcionário.
- **Dados Repetidos:** Os campos: Cidade e departamento.

Para normalizar a tabela Funcionario para que fique na 1FN devemos fazer o seguinte:

- **Primeiro:** Criar uma tabela chamada Cidade.
- **Segundo:** Inserir a chave estrangeira da tabela Cidade na tabela Funcionario.

Verificando mais uma vez os campos da tabela, observe o campo “idade”. Não está errado guardar, porém, a cada mudança de idade de um funcionário, uma atualização deverá ser realizada na tabela. Isso não é nem de longe prático e nem ideal para um banco de dados.

- **Criar Como Campo:** Data de nascimento ao invés do campo idade.

Segunda Forma Normal – 2FN

A 2FN deve obedecer à seguinte regra:

- Uma tabela está na 2FN, se e somente se estiver na 1FN e se todas as suas colunas que não são chaves dependem exclusivamente da chave primária (de toda a chave primária e não só de parte dela).

Para estar na 2FN devemos aplicar as seguintes ações:

- Identificar as colunas que não são funcionalmente dependentes da chave primária da tabela.
- Remover o campo da tabela e criar uma nova tabela com esses dados.

Para aplicar a 2FN, devemos criar uma tabela Departamento e inserir a chave estrangeira na tabela Funcionario. As tabelas irão ficar da seguinte forma:

- Departamento (#codDepart, Departamento).
- Funcionario (#matriculaFunc, nome, dtNascimento, valorDaHora, dtAdmissao, &codDepart, &idCidade).

Uma maneira de deixar a sua tabela sempre na 1FN é criar a chave primária e analisar o conteúdo que cada campo irá armazenar. Outra dica importante é sobre os campos: Cidade e Estado. Tanto Cidade como Estado sempre serão tabelas. Você nunca deverá deixar como um simples campo em uma tabela, se não o usuário poderá armazenar qualquer coisa como nome de cidade e de estado, gerando inconsistência nos dados armazenados.

Normalização de Dados

A normalização é um processo que visa diminuir a redundância no banco de dados. O objetivo é identificar e reduzir de forma gradual as anomalias que podem aparecer em tabelas ou nos relacionamentos. De forma geral, precisamos retirar um ou mais campos de uma tabela e criar novas tabelas para receber esses campos retirados.

O procedimento de normalização proporciona a quem for modelar um banco de dados as seguintes ações:

- **Análise dos Relacionamentos Entre as Tabelas:** Uma estrutura formal para a análise dos relacionamentos entre as tabelas, com base em suas chaves (primárias e estrangeiras) e nas dependências funcionais entre os campos da tabela.
- **Formas Normais:** Um conjunto de testes de formas normais para ser realizado em cada esquema de relação, de forma que o modelo de banco de dados seja normalizado no grau desejado, aplicando as formas normais até o limite que for mais conivente para a modelagem do banco de dados.

O método de normalização envolve a aplicação de um conjunto de regras, chamadas de formas normais. Ao avaliarmos um banco de dados e examinarmos que ele respeita as regras da primeira forma normal podemos desta forma afirmar que o banco está na 1FN (Primeira Forma Normal).

- **1FN:** Uma tabela está na 1FN quando todos os campos contêm apenas um valor correspondente singular e não existem grupos de atributos repetidos, isto quer dizer que não haverá repetições ou campos que tenham mais de um valor. O primeiro passo é identificar a chave primária da tabela, após, é necessário reconhecer os campos repetitivos e removê-los da tabela, e em seguida cria-se uma nova tabela, para receber os campos que estão repetindo na tabela que está sendo modelada.

- **2FN:** Uma tabela está na 2FN quando atender os requisitos da primeira forma normal e se os campos que estão na tabela, que não sejam chaves, dependerem da chave primária em sua totalidade e não em parte dela, causando irregularidades e prevenindo a redundância. Para solucionar, é necessário identificar esses campos que dependem parcialmente da chave primária e criar uma nova tabela para esses campos, criando um relacionamento entre as duas tabelas.

Tabela Cliente Não Normalizada na 2FN

A tabela Cliente está na 1FN e precisa que a 2FN seja aplicada:

- Ao ser normalizada a tabela Cliente para a 2FN, constata-se que a NotaFiscal pode ser uma nova tabela.
- É necessário criar uma chave estrangeira na tabela Cliente, fazendo a referência à tabela NotaFiscal. O relacionamento entre as duas tabelas (Cliente e NotaFiscal) é de 1 para N.

Terceira Forma Normal – 3FN

Uma tabela estará na 3FN somente se estivar na 2FN e todos os campos forem independentes, o que quer dizer que não poderá haver dependências funcionais entre os campos e que todos os campos dependem da chave primária da tabela. Os campos da tabela precisam depender unicamente da chave primária da tabela.

Para aplicar a terceira forma normal é necessário:

- Reconhecer os campos que são funcionalmente dependentes das outras colunas não chaves.
- Eliminar as colunas dependentes.

Na tabela Funcionario, o campo Descricao, refere-se à descrição do cargo que o funcionário ocupa, e neste caso será necessário aplicar a 3FN na tabela.

Para aplicar a 3FN na tabela Funcionario:

- Retiramos o campo Descricao, que é um campo que detalha o cargo e deve estar em uma tabela apropriada – no caso será a tabela Cargo.

O relacionamento entre as duas tabelas (Funcionario e Cargo) é 1 para N.

Quarta Forma Normal – 4FN

Para realizar a normalização de uma tabela na 4FN é necessário que a tabela esteja na 3FN. A tabela somente estará na 4FN se não existir dependência multivalorada.

De acordo com Navathe e Ramez (2005), em uma tabela na 4FN, além de estar na 3FN, todo campo precisa ser atômico (não pode ser dividido em vários campos).

Para aplicar a quarta forma normal é necessário:

- Primeiro identificar os campos multivalorados (que causam repetições).
- Criar uma tabela para cada grupo multivalorado.
- Criar uma chave primária para a nova tabela.

- Inserir a chave estrangeira na tabela que está sendo normalizada (na 4FN) para criar o relacionamento entre as tabelas.

Algumas perguntas sobre a tabela Funcionario precisam de respostas:

- Caso um funcionário tenha mais de um filho, como será o cadastro?
- Vamos cadastrar o funcionário duas vezes?
- E se o funcionário tiver cinco filhos?

A redundância na tabela Funcionario aparecerá com certeza. E para resolver esse problema, será necessário dividir a tabela Funcionario em duas tabelas: Funcionario e Dependente.

A tabela Func-Depend foi criada para relacionar os funcionários que possuem mais de um dependente e os dependentes que estão relacionados com mais de um funcionário (gerando assim um relacionamento N para N).