EAD

Caderno de Referência de Conteúdo

CRC

1. INTRODUÇÃO

Seja bem-vindo!

Você iniciará o estudo da disciplina *Banco de Dados*, a qual fornecerá a você conteúdos específicos para a projeção de bancos de dados. Para facilitar sua compreensão, o conteúdo foi segmentado em oito unidades. Juntamente a este *Caderno de Referência de Conteúdo* (CRC), você também terá acesso ao *Caderno de Atividades e Interatividades* (CAI), com diversos exercícios e atividades práticas, as quais complementarão seu estudo.

No transcorrer da disciplina, você terá a oportunidade de conhecer conceitos fundamentais para a elaboração e a construção de bancos de dados, como o projeto conceitual, o projeto lógico e o projeto físico, e a manipulação e a manutenção de bancos de dados por meio de linguagem SQL básica e avançada.

Outro aspecto relevante para o desenvolvimento de banco de dados bem estruturados é a aplicação da normalização, tema estritamente relacionado ao desempenho (acesso aos dados). Com o conteúdo estudado nesta disciplina, além de criar bancos de dados estruturados, você saberá aplicar as fases de normalização até a 4ª forma normal.

Além disso, é imprescindível que você, como administrador de bancos de dados, acompanhe a evolução dos Sistemas Gerenciadores de Bancos de Dados (SGBDs), não tomando apenas esta disciplina como fonte de aprendizagem. É importante que você complemente sua formação consultando outras fontes, como livros, revistas e páginas web. Não se esqueça de compartilhar suas experiências por meio dos Fóruns e da Lista na Sala de Aula Virtual, pois, assim, você não só contribuirá para o aprendizado de outras pessoas, mas também solidificará seu conhecimento.

Após essa introdução aos conceitos principais da disciplina, apresentaremos, a seguir, no Tópico *Orientações para o estudo da disciplina*, algumas orientações de caráter motivacional, dicas e estratégias de aprendizagem que poderão facilitar seu estudo.

2. ORIENTAÇÕES PARA O ESTUDO DA DISCIPLINA

Abordagem Geral da Disciplina

Prof. Ms. Geraldo Henrique Neto

Neste tópico, apresenta-se uma visão geral do que será estudado nesta disciplina. Aqui, você entrará em contato com os assuntos principais deste conteúdo de forma breve e geral e terá a oportunidade de aprofundar essas questões no estudo de cada unidade. Desse modo, essa Abordagem Geral visa fornecer-lhe o conhecimento básico necessário a partir do qual você possa construir um referencial teórico com base sólida – científica e cultural – para que, no futuro exercício de sua profissão, você a exerça com competência cognitiva, ética e responsabilidade social. Vamos começar nossa aventura pela apresentação das ideias e dos princípios básicos que fundamentam esta disciplina.

A disciplina *Banco de Dados* tem como objetivo principal a iniciação do aluno na elaboração e na construção de bancos de dados. Nesta disciplina, você conhecerá as fases constituintes de um projeto bem estruturado de banco de dados, como também terá contato com a linguagem SQL.

Antes de iniciar o estudo sobre banco de dados, é importante definir o que é um banco de dados. Pois bem, o banco de dados se constitui em um sistema de armazenamento de dados cujo objetivo é registrar e salvaguardar informações relevantes que poderão ser acessadas quando necessário. Os bancos de dados são amplamente utilizados e constituem a parte essencial de quase todas as empresas, independentemente do seu ramo de atividade. A importância dos bancos de dados foi impulsionada nos últimos anos devido ao crescimento das aplicações web, das implantações de ERP's (*Enterprise Resourcing Process*), de BI (*Businnes Intelligence*), dentre outras. Todas essas tecnologias são dependentes do banco de dados por envolverem armazenamento de grandes volumes de dados, recuperação de dados no menor tempo possível, segurança de acesso, *backup* de dados em tempo real, dentre outras funcionalidades.

Conforme comentado, o banco de dados é extremamente importante para o armazenamento de dados. Mas, afinal, qual a definição de dados? Existe diferença entre dados e informações? Para o bom entendimento da disciplina, é fundamental entender a diferença entre esses dois conceitos. Os dados são considerados fatos brutos, o que indica que os fatos ainda não foram processados para revelar seu significado. Como exemplo, imagine que você queira saber o que os usuários de um laboratório de informática pensam a respeito do serviço prestado. Normalmente, você começaria entrevistando os usuários para avaliar o desempenho do laboratório. Você poderia usar um formulário na web, o que permitiria que os usuários respondessem as suas questões. Quando o formulário estiver preenchido, os dados, nesse exato momento, são considerados como brutos, armazenados em um repositório de dados. Embora você já tenha os fatos em mãos, eles não possuem nenhuma utilidade específica nesse formato. Portanto, é necessário transformar os dados brutos em dados lapidados, permitindo obter respostas rápidas e objetivas das questões oriundas do formulário, como: "Qual é a composição da base de clientes de nosso laboratório?".

Para revelar seu significado, os dados brutos são processados de maneira apropriada, gerando, assim, as informações. Esse processamento pode ser considerado simples (como exemplo podemos citar a organização dos dados para extrair padrões) ou complexo (como a realização de estimativas, empregando variáveis estatísticas). As informações necessitam conhecer seu contexto para revelar seu significado adequado.

As informações são consideradas fundamentais para a tomada de decisão nas empresas, seja governamental, de serviços ou filantrópicas. Um resumo dos dados extraídos das questões referentes a cada formulário de entrevista pode demonstrar os pontos fortes e fracos de um serviço, por exemplo, e auxiliar na tomada de decisões para melhor atender às necessidades de seus clientes.

Você já ouviu falar em FAT, FAT32, NTFS, SQL Server, Oracle e MySQL? Todas essas siglas têm um objetivo em comum: organizar e armazenar dados em sistemas computacionais. Elas fazem parte de dois sistemas para controle de informação: Sistema de Arquivos e Sistema Gerenciador de Banco de Dados (SGBD).

Sistema de arquivos é o método de organizar e armazenar informações seguindo uma estrutura lógica para alocação física dos arquivos em dispositivos de armazenamento, tais como: disco rígido ou CD-ROM. Em outras palavras, o sistema de arquivos é a estrutura que indica como os arquivos devem ser gravados e guardados em algum sistema de armazenamento.

Já, o SGBD é uma coleção de programas que permite ao usuário definir, construir e manipular bases de dados para as mais diversas finalidades. Os programas ou *softwares* SGBDs mais conhecidos são Microsoft SQL Server, MySQL, Oracle, FireBird e PostgreSQL. No Sistema Gerenciador de Banco de Dados, o acesso aos dados e o seu gerenciamento são realizados pelo próprio SGBD, o qual funciona como uma interface entre o banco de dados e os programas aplicativos, isto é, o SGBD está localizado entre o banco de dados físico e os usuários.

Um modelo de banco de dados nada mais é do que uma descrição dos tipos de informações que serão armazenadas em um banco de dados. Para que possamos construir um modelo de dados é necessário o uso de uma linguagem de modelagem de dados. Essas linguagens são classificadas em linguagens textuais ou gráficas. Cada representação de um modelo de dados por meio de uma linguagem de modelagem de dados recebe a denominação de esquema de banco de dados.

Em um projeto de banco de dados, comumente são utilizados dois níveis de abstração, sendo eles: o modelo conceitual e o modelo lógico. Um modelo conceitual é uma descrição do banco de dados de forma independente de implementação em um SGBD. O modelo conceitual registra quais dados podem aparecer no banco de dados, mas não registra como estes dados serão armazenados em nível do SGBD.

A técnica de modelagem conceitual mais difundida é a abordagem entidade-relacionamento (ER). Utilizando-se esta técnica, um modelo conceitual é usualmente representado por meio de um diagrama, denominado diagrama entidade-relacionamento (DER).

Um modelo lógico é uma descrição de um banco de dados no nível de abstração visto pelo usuário do SGBD, de forma que este modelo é dependente do SGBD que será usado. Em um modelo lógico devem ser definidas quais as tabelas contidas pelo banco e, para cada tabela, os nomes das colunas.

No decorrer das unidades, veremos que o minimundo representa o domínio relacionado aos dados que o banco deve armazenar. Um levantamento dos requisitos desse minimundo é

efetuado e, a partir da análise de requisitos, é criado o projeto conceitual, representado pelo modelo entidade-relacionamento, o qual não contém detalhes de implementação, tratando-se, portanto, de um modelo de dados de alto nível, e independente do SGBD a ser adotado. O próximo passo, diz respeito à criação do projeto lógico, que é realizada por meio do mapeamento do modelo entidade-relacionamento para o modelo relacional. A partir dessa fase, você já pode pensar em um modelo de dados de implementação do SGBD. E, para finalizar, o último passo corresponde à fase nomeada de projeto físico, quando são definidos as estruturas de armazenamento interno, os índices, além de outras atividades desenvolvidas paralelamente, tal como a implementação dos programas de aplicação.

Você já ouviu falar em modelo entidade-relacionamento? Pois bem, o modelo entidade--relacionamento (MER) foi criado por Peter Chen na década de 1970, podendo ser considerado um padrão para a modelagem conceitual. Esse modelo é baseado na percepção do mundo real, e tem como finalidade a construção de objetos denominados entidades e a promoção o relacionamento entre eles.

O MER tem a intenção de descrever, de maneira conceitual, os dados que serão utilizados em um sistema de informação. Essa descrição acontece na fase conceitual do projeto de banco de dados e será utilizada pelo sistema de informação em questão. O modelo possui conceitos intuitivos que permitem aos projetistas de bancos de dados capturarem os conceitos inerentes aos dados da aplicação, independente de qualquer tecnologia utilizada para desenvolvimento de bancos de dados. O esquema conceitual criado utilizando-se os conceitos do modelo entidade-relacionamento é denominado de diagrama entidade-relacionamento (DER).

O diagrama entidade-relacionamento é composto por entidades, atributos e relacionamentos. As entidades representam objetos do mundo real; os atributos representam suas características; e os relacionamentos representam a forma como esses objetos estão ligados entre si. Uma entidade é um objeto existente e distinto de qualquer outro objeto. Sua finalidade é representar um conjunto de objetos da realidade modelada. Por exemplo, uma pessoa chamada de José Ribeiro Neves possui um número de CPF único, CPF nº 123.321.987-00; este número de CPF é uma entidade, uma vez que identifica a pessoa de uma forma única em relação às outras pessoas. Uma entidade pode ser concreta, como uma pessoa ou uma empresa, ou abstrata, como um conceito.

Dessa maneira, você pode perceber que um conjunto de entidades é um grupo de entidades do mesmo tipo. O conjunto de alunos de sua turma, por exemplo, pode ser definido como o conjunto de entidades ALUNOS. A representação gráfica de uma entidade é obtida por meio de um retângulo identificado por um nome da entidade.

Uma das propriedades sobre as quais pode ser desejável manter associações é a associação entre objetos. Por exemplo, você pode achar válido conhecer apenas as pessoas de um determinado departamento. Entretanto, uma pessoa teria que estar associada/vinculada a um departamento para que esta classificação obtenha êxito. Um relacionamento não ocorre, necessariamente, entre entidades diferentes, podemos notar a existência do autorrelacionamento, ou seja, um relacionamento entre ocorrências de uma mesma entidade. Um detalhe importante em um autorrelacionamento é a identificação e compreensão do conceito de papel da entidade no relacionamento. Esse papel de entidade em relacionamento exerce a função que uma instância da entidade cumpre dentro de uma instância do relacionamento.

Você já deve ter percebido que um diagrama entidade-relacionamento pode definir ***n*** restrições com as quais o banco de dados deve estar de acordo. Uma dessas restrições é a cardina-

lidade do mapeamento, que expressa o número de entidades relacionadas a outras entidades por meio de um conjunto de relacionamentos. Basicamente, existem três tipos de relacionamentos entre entidades binárias: um-para-um (uma entidade de A está associada a apenas uma entidade de B, e uma entidade de B está associada a apenas uma entidade de A), um-para-muitos (uma entidade de A está associada a muitas entidades de B, entretanto, uma entidade de B está associada a apenas uma entidade de A), muitos-para-muitos (uma entidade de A está associada a qualquer quantidade de entidades de B, e uma entidade de B está associada a qualquer número de entidades de A).

Não nós limitamos apenas aos relacionamentos e seus atributos, podemos ainda abstrairmos propriedades à entidades por meio do uso do conceito de generalização/especialização. Isso nos permite atribuir propriedades específicas a um subconjunto de ocorrências (especialização) de uma entidade considerada genérica. Para exemplificar o conceito de generalização/especialização, imagine que a entidade CLIENTE seja segmentada em dois subconjuntos, esses representados pelas entidades nomeadas respectivamente de PESSOA FÍSICA e PESSOA JURÍDICA, em que cada uma possui propriedades específicas. A aplicação da generalização/especialização implica que a entidade PESSOA FÍSICA, além de possuir suas características específicas, possui também todas aquelas contidas na entidade CLIENTE, ou seja, a entidade PESSOA FÍSICA possui os seguintes atributos: CPF, sexo, nome e código, sendo os dois últimos provenientes da entidade genérica CLIENTE. De modo similar, a entidade PESSOA JURÍDICA possui os seguintes atributos: CGC, tipo de organização, bem como o nome e código, sendo estes, oriundos da entidade CLIENTE.

Vimos que um relacionamento é uma associação entre entidades, certo? Na modelagem ER não foi prevista a possibilidade de associar uma entidade a um relacionamento, ou então, de associar dois relacionamentos entre si. Na prática, quando construímos um novo DER ou modificamos um DER existente, surgem situações em que se deseja flexibilizar a associação de uma entidade a um relacionamento. Uma entidade associativa nada mais é que a redefinição de um relacionamento, que passa a ser tratado como se fosse também uma entidade.

A linguagem SQL (*Structured Query Language*) não é uma linguagem de programação de computadores criada com o propósito de desenvolver sistemas, como as linguagens Pascal, C, Basic, Cobol, Java, C/C++, C#, dentre outras. É uma linguagem declarativa, cuja finalidade é facilitar o acesso às informações por meio de consultas, atualizações e manipulações de dados armazenados em bancos de dados relacionais.

Devido a sua boa aceitação no mercado, surgiram os primeiros problemas operacionais, pois cada empresa passou a incorporar funcionalidades e comandos próprios à linguagem SQL, diferenciando-a da sua forma original. Na tentativa de reduzir esses problemas, surge, nesse cenário, o ANSI (American National Standards Institute), o qual passou a estabelecer, por meio de um Comitê, normativas e critérios para definição de padrões para a linguagem SQL, que a partir desse período começou a ser referenciada por ANSI/SQL.

Apesar dos inúmeros esforços de entidades e institutos de padronização para construir um padrão de trabalho adequado, infelizmente, ainda existem empresas que implementam rotinas, funções e comandos totalmente fora do padrão estabelecido, dificultando a vida dos profissionais da área de Tecnologia da Informação (TI). Em alguns casos isolados, utilizam mais de um Sistema de Gerenciamento de Banco de dados em um mesmo ambiente operacional.

A linguagem SQL utilizada no SGBD PostgreSQL é segmentada em quatro subconjuntos que formam a base das instruções: DDL (*Data Definition Language*), DML (*Data Manipulation*

Language), DCL (*Data Control Language*) e DQL (*Data Query Language*) – podendo, ainda, incluir os subconjuntos SRC (*Stored Routines Commands*).

O subconjunto DDL (*Data Definition Language*) oferece recursos para definir objetos e controlar os dados. São comandos responsáveis pela estruturação do banco de dados, por exemplo: a criação do próprio banco de dados (*database*), a criação das tabelas, dos índices, entre outros objetos. Você utilizará a maioria dos comandos que constituem a DDL, como: CREATE TABLE, CREATE VIEW, CREATE DATABASE, entre outros, durante a disciplina.

Já, o subconjunto DML (*Data Manipulation Language*) tem como objetivo promover mecanismos para manipulação e gerenciamento dos bancos de dados, inserindo, alterando e removendo os dados. Os comandos que formam esse subconjunto são: DELETE, INSERT, UNION e UPDATE.

No que se refere ao controle de acesso aos dados, o DCL (*Data Control Language*) oferece recursos de controle de acesso de usuários ao sistema e aos dados, promovendo regras para realização de consultas, inserções, modificações e exclusões de dados do banco de dados. Essa linguagem é formada por comandos como GRANT e REVOKE, entre outros.

O comando SELECT faz parte da DQL (*Data Query Language*), considerado por alguns autores como um comando pertencente ao subconjunto DML (*Data Manipulation Language*).

Os SRC (*Stored Routines Commands*) são comandos que permitem a utilização de códigos de sub-rotinas armazenados no SGBD formados por AFTER, AS, BEFORE, BEGIN, CALLED, CREATE, DECLARE, entre outros.

E por fim, o DTC (*Data Type Commands*) é o grupo de comandos responsáveis por estabelecer o tipo de dado que uma coluna armazenará em uma tabela específica. O DTC é formado pelos comandos BIGINT, BIGSERIAL, CHAR, CHARACTER, DATE, DECIMAL, DOUBLE, INTEGER, INT, MONEY, NUMERIC, entre outros.

Além dos comandos descritos anteriormente, existe um conjunto de funções predefinidas, com uma série de operadores, que promovem facilitadores nas diversas ações realizadas pelos aplicativos.

Antes mesmo de você explorar as fases de normalização de um banco de dados, você deverá conhecer as restrições de integridade conhecidas como dependências funcionais. A normalização está relacionada intimamente com aspectos importantes como desempenho do banco de dados.

Segundo Takai; Italiano e Ferreira (2005), a dependência funcional:

> [...] é uma propriedade do significado ou semântica dos atributos em um esquema de relação **R**. Utiliza-se o entendimento da semântica de atributos de R, isto é, como eles se relacionam, para especificar as dependências funcionais envolvidas em todas as instâncias da relação **r** (extensão) de **R**. As instâncias **r** satisfazem as restrições legais, pois obedecem as restrições de dependência funcional. Assim, a principal utilização das dependências funcionais é a de descrever um esquema de relação R especificando restrições sobre seus atributos que devem ser validados todas às vezes.

A função da normalização é atuar como um filtro sobre as entidades e os relacionamentos, eliminando alguns elementos sem causar perda de informação nas entidades e nas relações. No decorrer do estudo desta disciplina, você estudará como isso é realizado por meio das **formas normais**, as quais foram introduzidas para atuar em cada caso em que a informação se encontra disponível para ser tratada, deixando os dados mais bem organizados dentro de um banco de dados. Ao aplicarmos a normalização, deseja-se evitar: grupos repetitivos de dados; variação temporal de certos atributos, dependências funcionais totais ou parciais em relação a uma

chave concatenada; redundâncias de dados desnecessários; perdas acidentais de informação; dificuldade na representação de fatos da realidade observada e dependências transitivas entre atributos.

A normalização tem sua atuação por meio das formas normais. Os primeiros estágios do processo de normalização são: primeira forma normal (1FN), segunda forma normal (2FN) e terceira forma normal (3FN). Nesses três estágios, sob um ponto de vista estrutural, a forma normal maior apresenta-se melhor sobre sua antecessora, logo a 3FN é melhor que a 2FN, que por sua vez é melhor que a 1FN.

Embora a normalização seja de grande relevância para o sucesso de um Projeto de Banco de Dados, não devemos assumir que seu nível alto seja sempre o mais adequado a se utilizar. O êxito do projeto dá-se, mediante a análise da demanda do usuário para com um desempenho satisfatório.

O objetivo da normalização é garantir que todas as tabelas atendam ao conceito de relações bem estabelecidas, ou seja, que tenham as seguintes características: cada tabela deve apresentar um único assunto, por exemplo, uma tabela de disciplina deverá possuir apenas as informações pertinentes à disciplina; nenhum item de dado deverá ser armazenado de forma desnecessária em mais de uma tabela, como mencionado anteriormente, a redundância, se houver necessidade desta, deve ser minuciosamente controlada; todos os atributos não pertinentes à tabela deverão ser dependentes da chave primária, garantindo assim a identificação exclusiva dos dados; e, para finalizar, todas as tabelas deverão estar livres das anomalias (anomalias de atualização, inserção e exclusão), de modo a garantir a integridade e a consistência dos dados.

Uma relação se encontra na **primeira forma normal** (1FN) se todos os seus atributos são monovalorados e atômicos. Entretanto, se utilizarmos a 1FN e a tabela resultante possuir chave primária com apenas um atributo, a tabela está automaticamente na 2FN.

Semelhante ao processo de normalização, um modelo antes de ser convertido à terceira forma normal (3FN) deve, primeiro, estar enquadrado na 2FN. Existe uma forma normal mais restritiva do que a 3FN, que é a forma normal de *Boyce-Codd* (BCNF), a qual foi proposta como uma forma mais simples que a 3FN. A BCNF é considerada mais restritiva, pois, à medida que toda relação está na BCNF, também estará na 3FN; entretanto, o inverso não é considerado verdadeiro.

É desejável que um projeto de banco de dados alcance a 3FN ou BCNF para todo esquema da relação. Alcançar o *status* de normalização somente na 1FN ou na 2FN não é considerado adequado, uma vez que essas formas foram desenvolvidas como caminho para a 3FN. Uma tabela é dita na 3FN se, e somente se, estiver na 2FN e não possuir, em hipótese alguma, dependências transitivas.

Não podemos pensar em um banco de dados como um sistema único, pois, dificilmente, ele fica isolado em uma máquina e é acessado apenas por uma pessoa, ou de apenas um lugar. Ao pensar em banco de dados, devemos ter em mente que informações armazenadas podem ser acessadas por muitas pessoas, as quais podem, ainda, estar em locais diferentes.

Em uma instituição financeira, como os muitos bancos que temos espalhados pelo Brasil (considerando apenas o território nacional), você pode ter uma conta que foi aberta em determinada agência, mas pode fazer as operações bancárias a partir de qualquer local em que haja um caixa eletrônico dessa instituição financeira.

Além disso, você pode ter uma conta com outra pessoa, isto é, duas pessoas podem fazer as mesmas operações na mesma conta bancária. Se ambas, a partir de lugares diferentes, resolverem fazer um saque, no mesmo momento, da mesma conta corrente, quem sacará primeiro? O usuário poderia receber uma informação incorreta e, muitas vezes, sem lógica.

Para que não ocorra esse caos no sistema, existe a transação, que é uma unidade de execução de programa que acessa e, possivelmente, atualiza vários itens de dados.

Analisando do ponto de vista do SGBD, uma transação é uma sequência de operações que são tratadas como um bloco único e indivisível (atômico), quanto à sua recuperação, ou seja, a execução parcial de transações não é permitida.

As responsabilidades do DBA variam a cada organização, mas, de modo geral, o DBA deverá planejar a estratégia de administração de dados. Sua função, de modo direto, consiste em definir, implementar e aplicar as políticas, padrões e procedimentos pertinentes a uma melhor conduta para o zelo das informações contidas em um banco de dados.

Sabe-se que um dos maiores ativos das organizações é o seu contingente informacional, e quando uma organização necessita acessar determinada informação que não está prontamente disponível, perdas podem ser geradas. Por isso, procedimentos de *backup* e restauração são imprescindíveis para assegurar a proteção e a constante disponibilidade das informações obtidas e acumuladas ao longo do tempo.

Os procedimentos de *backup* e restauração são muito importantes em qualquer SGBD utilizado. O DBA em questão deve garantir que os dados possam ser recuperados totalmente em caso de perda física ou de integridade dos dados.

As atividades do DBA inclui o gerenciamento de desastres, de modo a garantir a disponibilidade dos dados após um possível desastre, seja ele físico ou uma falha de integridade. Para tanto, é necessário um planejamento sistêmico dentro da organização, o que deverá compreender planos de testes e planejamento de ações, procedimentos necessários à recuperação.

A computação distribuída pode ser aplicada não somente a banco de dados, mas a quaisquer componentes que se encontram em computadores ou em locais diferentes e que, de alguma forma, se relacionam.

Um sistema de computação distribuída pode ser entendido como vários componentes que estão ligados por uma rede de computadores e que se relacionam para executarem tarefas em comum.

Segundo Elmasri e Navathe (2005, p. 579), banco de dados distribuídos pode ser definido como "uma coleção de múltiplos bancos de dados logicamente inter-relacionados, distribuídos por uma rede de computadores".

Um Sistema de Gerenciamento de Banco de Dados Distribuídos (SGBDD) tem por finalidade controlar o armazenamento e processamento de dados relacionados logicamente por meio de sistemas computacionais interconectados. Neste contexto, tanto os dados como as funções de processamento são distribuídos entre os diversos locais, também nomeados, eventualmente, de nós.

Bem, chegamos ao final de nossa abordagem e esperamos que você tenha aproveitado ao máximo os tópicos aqui apresentados, mesmo que de forma sucinta. É importante destacar que apenas o estudo teórico da linguagem SQL não será suficiente para o seu aprendizado: é fundamental que você pratique exaustivamente! Para tanto, crie o cenário instalando o SGBD PostgreSQL, conforme orientações descritas no Apêndice 1.

Glossário de Conceitos

O Glossário de Conceitos permite a você uma consulta rápida e precisa das definições conceituais, possibilitando-lhe um bom domínio dos termos técnico-científicos utilizados na área de conhecimento dos temas tratados na disciplina *Banco de Dados*. Veja, a seguir, a definição dos principais conceitos desta disciplina:

1) **Atributo**: abstração de uma propriedade de uma entidade ou de um relacionamento.
2) **Banco de Dados:** sistema de armazenamento de dados cujo objetivo é registrar e guardar informações importantes que poderão ser acessadas quando necessário.
3) **Banco de Dados Espaciais**: aquele que permite consultar objetos localizados em um espaço multidimensional. É o caso dos bancos de dados geográficos, que armazenam informações sobre mapas para localização de rios, cidades, estados, estradas, entre outros.
4) **Banco de Dados Especialistas**: também conhecido como sistemas baseados em conhecimento, é aquele que, por meio de técnicas aplicadas na área da Inteligência Artificial, incorpora raciocínio e inferência (capacidade de dedução).
5) **Banco de Dados Meteorológicos**: banco que armazena informações sobre o tempo.
6) **Banco de Dados Multimídias**: banco que armazena os dados sob a forma de imagem, clipes de filmes, músicas, textos falados ou escritos, entre outros.
7) **Banco de Dados Temporais**: é aquele que permite ao usuário consultar estados atuais e passados do banco de dados, pois armazena históricos das alterações.
8) **Base de Dados**: representado por um conjunto de banco de dados. Base de dados e banco de dados não são sinônimos.
9) **BI (*Business Intelligence* ou Inteligência de Negócios):** utiliza conceitos em que as informações são coletadas, armazenadas e analisadas, tendo como base fatos reais e/ou hipóteses. Esses sistemas auxiliam na gestão organizacional e no processo de tomada de decisões.
10) **Dado atômico**: tipo de dado considerado básico, ou seja, indivisível.
11) **Dado não atômico**: tipo de dado considerado complexo, divisíveis (fragmentados).
12) **Entidade**: abstração de um fato do mundo real para o qual se deseja manter seus dados no banco de dados.
13) **ERP's:** são sistemas de gestã**o** empresarial que possibilitam a integração de todos os dados e processos de uma empresa, melhorando o fluxo de informações.
14) **Gerenciamento de Banco de Dados**: utiliza Sistemas Gerenciadores de Banco de Dados (SGBDs).
15) **Gerenciamento de Base de Dados**: utiliza ferramentas de apoio integradas à tomada de decisão organizacional (ERP – *Enterprise Resource Planning*) ou *Datawarehouse*.
16) **Integridade Semântica**: garantia de dados sempre corretos com relação ao domínio de aplicação.
17) **Modelagem Conceitual**: nível mais alto de abstração cujo objetivo é representar os requisitos de dados do domínio da aplicação (independente do modelo de banco de dados).
18) **Modelagem Física**: constitui um esquema SQL para a modelagem lógica (depende exclusivamente do SGBD).
19) **Modelagem Lógica**: representação da modelagem conceitual em um modelo de banco de dados.
20) **Relacionamento**: abstração de uma associação entre (ocorrências de) entidades.
21) **Sistema Gerenciador de Banco de Dados** (**SGBD**): coleção de programas responsáveis pela criação e manutenção de banco de dados.

Esquema dos Conceitos-chave

Para que você tenha uma visão geral dos conceitos mais importantes deste estudo, apresentamos, a seguir (Figura 1), um Esquema dos Conceitos-chave da disciplina. O mais aconselhável é que você mesmo faça o seu esquema de conceitos-chave ou até mesmo o seu mapa mental. Esse exercício é uma forma de você construir o seu conhecimento, ressignificando as informações a partir de suas próprias percepções.

É importante ressaltar que o propósito desse Esquema dos Conceitos-chave é representar, de maneira gráfica, as relações entre os conceitos por meio de palavras-chave, partindo dos mais complexos para os mais simples. Esse recurso pode auxiliar você na ordenação e na sequenciação hierarquizada dos conteúdos de ensino.

Com base na teoria de aprendizagem significativa, entende-se que, por meio da organização das ideias e dos princípios em esquemas e mapas mentais, o indivíduo pode construir o seu conhecimento de maneira mais produtiva e obter, assim, ganhos pedagógicos significativos no seu processo de ensino e aprendizagem.

Aplicado a diversas áreas do ensino e da aprendizagem escolar (tais como planejamentos de currículo, sistemas e pesquisas em Educação), o Esquema dos Conceitos-chave baseia-se, ainda, na ideia fundamental da Psicologia Cognitiva de Ausubel, que estabelece que a aprendizagem ocorre pela assimilação de novos conceitos e de proposições na estrutura cognitiva do aluno. Assim, novas ideias e informações são aprendidas, uma vez que existem pontos de ancoragem.

Tem-se de destacar que "aprendizagem" não significa, apenas, realizar acréscimos na estrutura cognitiva do aluno; é preciso, sobretudo, estabelecer modificações para que ela se configure como uma aprendizagem significativa. Para isso, é importante considerar as entradas de conhecimento e organizar bem os materiais de aprendizagem. Além disso, as novas ideias e os novos conceitos devem ser potencialmente significativos para o aluno, uma vez que, ao fixar esses conceitos nas suas já existentes estruturas cognitivas, outros serão também relembrados.

Nessa perspectiva, partindo-se do pressuposto de que é você o principal agente da construção do próprio conhecimento, por meio de sua predisposição afetiva e de suas motivações internas e externas, o Esquema dos Conceitos-chave tem por objetivo tornar significativa a sua aprendizagem, transformando o seu conhecimento sistematizado em conteúdo curricular, ou seja, estabelecendo uma relação entre aquilo que você acabou de conhecer com o que já fazia parte do seu conhecimento de mundo (adaptado do *site* disponível em: <http://penta2.ufrgs.br/edutools/mapasconceituais/utilizamapasconceituais.html>. Acesso em: 11 mar. 2010).

Como poderá observar, esse Esquema oferecerá a você, como dissemos anteriormente, uma visão geral dos conceitos mais importantes deste estudo. Ao segui-lo, será possível transitar entre os principais conceitos desta disciplina e descobrir o caminho para construir o seu processo de ensino-aprendizagem. Dentre esses conceitos, há o de *Projeto de Banco de Dados*, o qual implica o conhecimento dos modelos conceitual, lógico e físico, além dos detalhes inclusos em cada modelo mencionado.

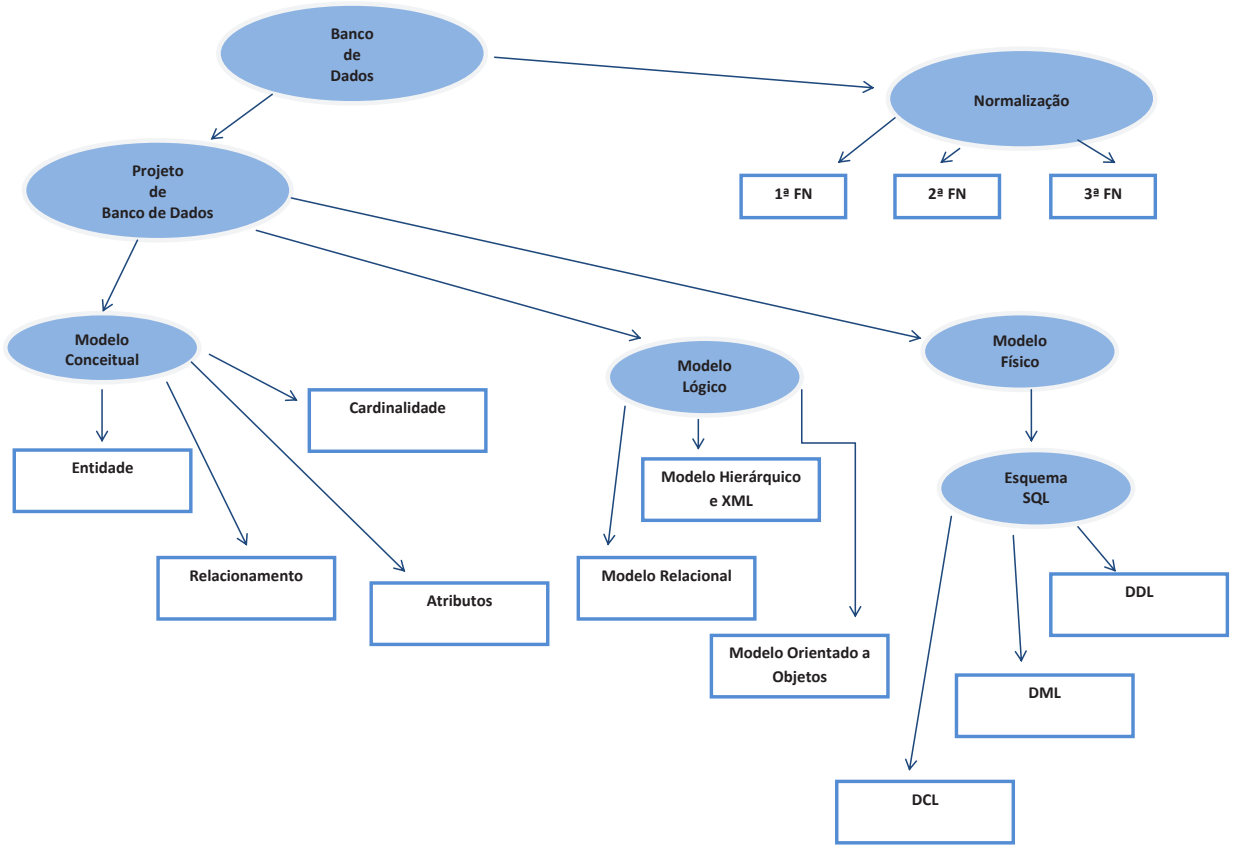


Figura 1 *Esquema dos Conceitos-chave da disciplina Banco de Dados*.

O Esquema dos Conceitos-chave é mais um dos recursos de aprendizagem que vem se somar àqueles disponíveis no ambiente virtual, por meio de suas ferramentas interativas, bem como àqueles relacionados às atividades didático-pedagógicas realizadas presencialmente no polo. Lembre-se de que você, aluno EaD, deve valer-se da sua autonomia na construção de seu próprio conhecimento.

Questões Autoavaliativas

No final de cada unidade, você encontrará algumas questões autoavaliativas sobre os conteúdos ali tratados, as quais podem ser de **múltipla escolha**, **abertas objetivas** ou **abertas dissertativas**.

Responder, discutir e comentar essas questões, bem como relacioná-las com a prática do ensino de Banco de Dados pode ser uma forma de você avaliar o seu conhecimento. Assim, mediante a resolução de questões pertinentes ao assunto tratado, você estará se preparando para a avaliação final, que será dissertativa. Além disso, essa é uma maneira privilegiada de você testar seus conhecimentos e adquirir uma formação sólida para a sua prática profissional.

Você encontrará, ainda, no final de cada unidade, um gabarito, que lhe permitirá conferir as suas respostas sobre as questões autoavaliativas de múltipla escolha.

As **questões de múltipla escolha** são as que têm como resposta apenas uma alternativa correta. Por sua vez, entendem-se por **questões abertas objetivas** as que se referem aos conteúdos matemáticos ou àqueles que exigem uma resposta determinada, inalterada. Já as **questões abertas dissertativas** obtêm por resposta uma interpretação pessoal sobre o tema tratado; por isso, normalmente, não há nada relacionado a elas no item Gabarito. Você pode comentar suas respostas com o seu tutor ou com seus colegas de turma.

Bibliografia Básica

É fundamental que você use a Bibliografia Básica em seus estudos, mas não se prenda só a ela. Consulte, também, as bibliografias apresentadas no *Plano de Ensino* e no item *Orientações para o estudo da unidade*.

Figuras (ilustrações, quadros...)

Neste material instrucional, as ilustrações fazem parte integrante dos conteúdos, ou seja, elas não são meramente ilustrativas, pois esquematizam e resumem conteúdos explicitados no texto. Não deixe de observar a relação dessas figuras com os conteúdos da disciplina, pois relacionar aquilo que está no campo visual com o conceitual faz parte de uma boa formação intelectual.

Dicas (motivacionais)

O estudo desta disciplina convida você a olhar, de forma mais apurada, a Educação como processo de emancipação do ser humano. É importante que você se atente às explicações teóricas, práticas e científicas que estão presentes nos meios de comunicação, bem como partilhe suas descobertas com seus colegas, pois, ao compartilhar com outras pessoas aquilo que você observa, permite-se descobrir algo que ainda não se conhece, aprendendo a ver e a notar o que não havia sido percebido antes. Observar é, portanto, uma capacidade que nos impele à maturidade.

Você, como aluno do curso de graduação na modalidade EaD, necessita de uma formação conceitual sólida e consistente. Para isso, você contará com a ajuda do tutor a distância, do tutor presencial e, sobretudo, da interação com seus colegas. Sugerimos, pois, que organize bem o seu tempo e realize as atividades nas datas estipuladas.

É importante, ainda, que você anote as suas reflexões em seu caderno ou no Bloco de Anotações, pois, no futuro, elas poderão ser utilizadas na elaboração de sua monografia ou de produções científicas.

Leia os livros da bibliografia indicada, para que você amplie seus horizontes teóricos. Coteje-os com o material didático, discuta a unidade com seus colegas e com o tutor e assista às videoaulas.

No final de cada unidade, você encontrará algumas questões autoavaliativas, que são importantes para a sua análise sobre os conteúdos desenvolvidos e para saber se estes foram significativos para sua formação. Indague, reflita, conteste e construa resenhas, pois esses procedimentos serão importantes para o seu amadurecimento intelectual.

Lembre-se de que o segredo do sucesso em um curso na modalidade a distância é participar, ou seja, interagir, procurando sempre cooperar e colaborar com seus colegas e tutores.

Caso precise de auxílio sobre algum assunto relacionado a esta disciplina, entre em contato com seu tutor. Ele estará pronto para ajudar você.

3. E-REFERÊNCIA

TAKAI, O. K.; ITALIANO, I. C.; FERREIRA, J. E. *Introdução a Banco de Dados*. 2005. Disponível em: <http://pt.scribd.com/doc/51228653/9/Arquiteturas>. Acesso em: 22 out. 2012.

4. REFERÊNCIA BIBLIOGRÁFICA

ELMASRI, R.; NAVATHE, S. B. *Sistemas de bancos de dados*. São Paulo: Pearson (Addison Wesley), 2005.

EAD

Introdução aos Sistemas de Banco de Dados

1

1. OBJETIVOS

- Compreender o conceito básico de Sistemas Gerenciadores de Bancos de Dados (SGBD), sua importância, utilização e aplicação.
- Conhecer os SGBDs mais utilizados no mercado, os problemas de armazenamento e recuperação de dados (redundâncias, inconsistências, integridade, compartilhamento e segurança) e os requisitos básicos que devem ser atendidos para um bom desempenho.

2. CONTEÚDOS

- Perspectiva histórica.
- Sistemas de arquivos *versus* SGBDs.
- Dados em SGBDs: descrição e armazenamento.
- Arquitetura em um SGBD.
- Consultas em um SGBD.

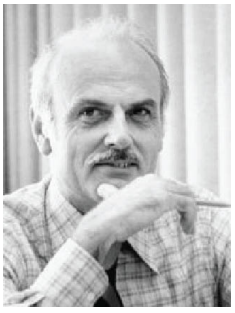
3. ORIENTAÇÕES PARA O ESTUDO DA UNIDADE

Antes de iniciar o estudo desta unidade, é importante que você leia as orientações a seguir:

1) Tenha sempre à mão o significado dos conceitos explicitados no Glossário e suas ligações pelo Esquema de Conceitos-chave para o estudo de todas as unidades deste CRC. Isso poderá facilitar sua aprendizagem e seu desempenho.
2) É importante que você fique sempre atento às informações contidas na unidade. Programe-se, organize seus estudos e participe ativamente na Sala de Aula Virtual. Ser disciplinado para estudar pode ajudar você a tirar o máximo de proveito em seu curso de Educação a Distância.
3) Antes de iniciar os estudos desta unidade, pode ser interessante conhecer um pouco sobre a evolução das formas de armazenamento de dados. Para saber mais, acesse os *sites* indicados nas E-referências.
4) O início do estudo desta unidade envolve alguns aspectos teóricos que serão fundamentais ao longo de todo o aprendizado. Dessa maneira, faça uso de um bloco de anotações para destacar os principais conceitos, tais como: banco de dados, dados e informações, sistemas de arquivos, Sistemas Gerenciadores de Bancos de Dados (SGBD) etc.
5) Lembre-se que **dado** é o que está armazenado no banco de dados e **informação** é o significado dos dados.
6) O *best-seller O mundo é plano*, de Thomas Friedman, mostra que, devido ao avanço tecnológico, uma informação inédita é disponibilizada para o mundo todo em segundos, colocando em igualdade de conhecimento populações de países desenvolvidos e países em desenvolvimento. Reflita sobre a importância dos dados e seu armazenamento no mundo globalizado.
7) Os conceitos apresentados nesta unidade serão trabalhados na prática nas unidades seguintes. Preocupe-se, a princípio, em compreender as diferenças entre os conceitos apresentados. Se necessário, anote os termos e suas definições em um caderno para retomá-los posteriormente.

Perspectiva Histórica

Em algum momento da história, as empresas descobriram que estava muito caro empregar um número grande de pessoas para fazer certos trabalhos, como armazenar e indexar (organizar) arquivos. Por este motivo, valia a pena empregar esforços e investir em pesquisas em busca de um meio mais barato e uma solução mecânica eficiente.



Na década de 1970, Ted Codd, um pesquisador renomado da IBM, publicou o primeiro artigo referente a bancos de dados relacionais. Este artigo discorria sobre o uso de cálculo e álgebra relacional, recursos que possibilitavam que usuários não técnicos manipulassem grande quantidade de informações. Codd projetava em sua mente um sistema computacional em que fosse factível ao usuário acessar os dados armazenados em tabelas através de comandos específicos. Na época, ninguém percebeu que as teorias obscuras de Codd desencadeariam uma revolução tecnológica comparável ao desenvolvimento dos computadores pessoais e da internet. Don Chamberlin, coinventor da SQL, a mais popular linguagem de computador utilizada pelos sistemas de bancos de dados de hoje, explica: "Havia aquele cara, Ted Codd, que usava um tipo de notação matemática estranha, mas ninguém a levava muito a sério". Então, Ted Codd organizou um simpósio e Chamberlin ouviu como ele conseguiu resumir cinco páginas de programas complicados em uma única linha. "E eu disse: Uau!", relembra Chamberlin.

O simpósio convenceu a IBM (*Internacional Business Machines*) a fundar o System R (Sistema R), um projeto de pesquisa que construiu um protótipo de banco de dados relacional e que levaria à criação da SQL (*Strutured Query Language*) e do DB2. Esta linguagem tornou-se um padrão na indústria para bancos de dados relacionais e, hoje em dia, é um padrão ISO (*International Organization for Standardization*). Os primeiros protótipos foram utilizados por muitas organizações, como o MIT Sloan School of Management (escola norte-americana conhecida na área de negócios). Novas versões do sistema foram testadas com empresas de aviação para rastreamento do manufaturamento de estoque. A IBM, no entanto, manteve o System R em segundo plano por vários e decisivos anos. O interesse da empresa voltava-se para o IMS, um sistema de banco de dados confiável, de alta tecnologia, que havia surgido em 1968. Sem perceber o potencial de mercado daquela pesquisa, a IBM permitiu que sua equipe publicasse seus trabalhos.

Entre os leitores estava Larry Ellison, que havia acabado de fundar uma pequena empresa. Recrutando programadores do System R e da Universidade da Califórnia, Ellison conseguiu colocar no mercado, bem antes da IBM, o primeiro banco de dados relacional com base em SQL, em 1979. Em 1983, a empresa laçou uma versão portátil do banco de dados, tendo um faturamento bruto anual de US$ 5.000.000 e alterou seu nome para Oracle. Impedida pela concorrência, a IBM finalmente lançou o SQL/DS (*Structured Query Language*/*Data System*), seu primeiro banco de dados relacional, em 1980 (imagem disponível em: <http://www.answers.com/topic/edgar-f-codd>. Acesso em: 18 maio 2012.).

4. INTRODUÇÃO À UNIDADE

Nesta unidade, você terá a oportunidade de estudar alguns conceitos de banco de dados, bem como aprender a importância dos Sistemas Gerenciadores de Banco de Dados em sistemas de informações. Para tanto, veja a definição a seguir:

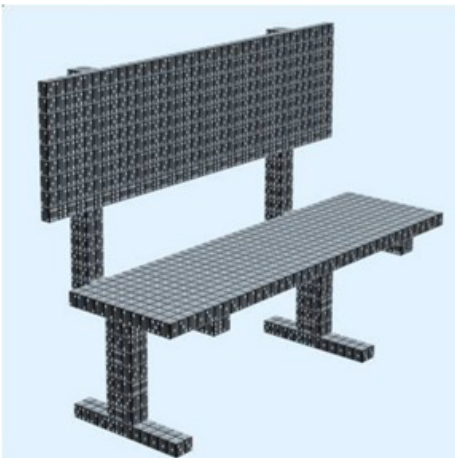


Figura 1 *Banco de Dados*.

Banco de dados é um sistema de armazenamento de dados cujo objetivo é registrar e guardar informações importantes que poderão ser acessadas quando necessário. Os bancos de dados são amplamente usados, formando uma parte essencial de quase todas as empresas.

Veja alguns exemplos de onde são empregados os bancos de dados:

1) **Instituições Financeiras (bancos)**: utilizam o banco de dados para o devido armazenamento de informações de seus clientes, suas respectivas contas, empréstimos e transações bancárias.
2) **Linhas Aéreas:** seu banco de dados armazena reservas e informações de horários dos voos. Sistemas computacionais de gerenciamento de linhas aéreas foram os primeiros aplicativos a utilizar o banco de dados de maneira geograficamente distribuída.
3) **Universidades:** empregam o banco de dados para manipular informações dos alunos, registros de cursos e notas.
4) **Transações de Cartão de Crédito:** constantemente utilizam bancos de dados para gerenciar compras com cartão de crédito e geração de faturas mensais.
5) **Telecomunicação:** o banco de dados é aplicado para manter registros de chamadas realizadas, gerar cobranças mensais, gerenciar saldos de cartões telefônicos pré-pagos e armazenar informações sobre *status* das redes de comunicações.
6) **Finanças:** na área financeira, o banco de dados é utilizado para armazenar informações pertinentes aos valores mobiliários ou vendas e compras de instrumentos financeiros, como ações e títulos. O banco de dados também é empregado no armazenamento e gerenciamento de dados oriundos do mercado financeiro em tempo real, cuja finalidade é permitir aos clientes realizar negócios *on-line* e às empresas, transações automatizadas.
7) **Vendas:** é imprescindível o uso de banco de dados para gerenciar e armazenar informações de cliente, produto e compra.
8) **Indústria:** utiliza o banco de dados para o gerenciamento da cadeia de suprimentos e para o controle da produção de itens, estoques e pedidos nas fábricas.
9) **Recursos Humanos:** emprega banco de dados para armazenar e gerenciar informações referentes aos funcionários, salários, descontos em folha de pagamento, benefícios, e também para a geração de contracheques.

Essa forma organizada de armazenamento permite a fácil manipulação dos dados, incluindo alterações, inserções, remoções, além das consultas.

É comum empregar os termos **banco de dados** e **base de dados** como sinônimos. De certa forma é um erro, pois o gerenciamento de uma base de dados utiliza, normalmente, ferramentas de apoio integradas à tomada de decisão organizacional; como exemplo, podem ser citados o ERP (*Enterprise Resource Planning*) e o *Data warehouse*. Entretanto, o gerenciamento de um banco de dados é o obtido por meio de sistemas de gerenciamento de bancos de dados.

Em geral, o gerenciamento eficiente de dados necessita do uso de um banco de dados computacional (digital), que pode ser considerado como uma estrutura compartilhada e integrada, a qual armazena um conjunto de:

- Dados brutos oriundos do usuário.
- Metadados, os quais permitem integrar e gerenciar os dados.

Os metadados fornecem uma descrição detalhada das características dos dados e do seu conjunto de relacionamentos, responsáveis por conectar os dados encontrados no banco de dados. Por exemplo, o componente de metadados armazena informações como o nome de cada elemento de dados, seu respectivo tipo (numérico, data ou texto) armazenado, a possibilidade ou não de deixar esse elemento vazio, dentre outras possibilidades. Desse modo, os metadados fornecem informações que complementam e expandem o valor e a utilização dos dados, ou seja, trazem uma representação mais completa dos dados no banco de dados.

A importância do banco de dados no cenário da Tecnologia da Informação (TI) aumentou consideravelmente nos últimos anos, impulsionada pelo crescimento das aplicações *web,* das implantações de **ERP's** (*Enterprise Resourcing Process*), de **BI** (*Business Intelligence*) etc. Todas essas tecnologias são dependentes do banco de dados por envolverem o armazenamento de grandes volumes de dados, a recuperação de dados no menor tempo possível (principalmente no comércio eletrônico), a segurança de acesso, o *backup* de dados em tempo real, dentre outras funções.

5. DADOS *VERSUS* INFORMAÇÕES

Para o bom entendimento da disciplina, é importante que você entenda a diferença entre dados e informações. Os dados são considerados fatos brutos, o que indica que os fatos ainda não foram processados para revelar seu significado. Como exemplo, imagine que você queira saber o que os usuários de um laboratório de informática pensam a respeito do serviço prestado. Normalmente, você começaria entrevistando os usuários para avaliar o desempenho do laboratório. Você poderia usar um formulário na *web*, o qual permitiria que os usuários respondessem às suas questões. Quando o formulário estiver preenchido, os dados, nesse momento considerados como brutos, são armazenados em um repositório de dados. Embora você já tenha os fatos em mãos, eles não possuem nenhuma utilidade específica nesse formato. Portanto, é necessário transformar os dados brutos em dados lapidados, o que permitirá a obtenção de respostas rápidas e objetivas das questões oriundas do formulário, como: "Qual é a composição da base de clientes de nosso laboratório?".

Para revelar seu significado, os dados brutos são processados de maneira apropriada, gerando, assim, as informações. Esse processamento pode ser considerado simples (a organização dos dados para extrair padrões, por exemplo) ou complexo (como a realização de estimativas, empregando variáveis estatísticas). As informações necessitam conhecer seu contexto para re-

velar seu significado adequado. O simples fato de obtermos uma temperatura média de 95°, por exemplo, pode ser considerado um dado específico sem significado algum, a menos que saibamos seu contexto: encontra-se em graus Fahrenheit ou Celsius? Está vinculada à temperatura de um *hardware* ou de um corpo humano?

Na maioria das vezes, as informações são pilares fundamentais para a tomada de decisões nas empresas, sejam elas governamentais, de serviços ou filantrópicas. No exemplo anterior, o resumo dos dados extraídos das questões referentes a cada formulário de entrevista pode demonstrar os pontos fortes e fracos do laboratório de informática, auxiliando, dessa forma, na tomada de decisões confiáveis para melhor atender às necessidades de seus clientes.

De acordo com as características e a aplicabilidade dos dados armazenados, os bancos são classificados de diferentes maneiras: podem se basear no número de usuários, na localização (ou localizações) e no tipo e extensão do uso esperado.

Veja a classificação dos bancos de dados por número de usuários:

1) **Banco de dados monousuário (de um único usuário):** suporta apenas um usuário por vez. Por exemplo, se o usuário Pedro estiver utilizando o banco de dados, os usuários Regina e Douglas precisam esperar o usuário Pedro finalizar sua consulta.
2) **Banco de dados multiusuário:** dá suporte a diversos usuários simultaneamente. Por exemplo, os usuários Pedro, Regina e Douglas poderão utilizar o banco de dados ao mesmo tempo.

Outra característica importante dos bancos de dados pode ser estabelecida levando em consideração sua localização. Observe:

1) **Banco de dados centralizado:** estabelece suporte a dados centralizados em um único local.
2) **Banco de dados distribuído:** suporta dados distribuídos por vários locais, normalmente, geograficamente distintos.

Atualmente, a maneira mais usual de classificar os bancos de dados baseia-se no modo como estes serão utilizados e na sensibilidade ao tempo das informações nele coletadas. Podemos detalhar essa classificação como:

1) **Bancos de dados operacionais:** projetados especialmente para suportar operações diárias de uma empresa. Também podem ser referenciados como bancos de dados transacionais ou de produção.
2) ***Data Warehouse* (armazém de dados):** tem a finalidade de armazenar os dados utilizados para a geração de informações necessárias à tomada de decisões táticas e estratégicas. Essas decisões exigem que os dados sejam "lapidados" (manipulação de dados), a fim de extrair informações úteis para a formulação de decisões, previsões de vendas, posicionamento no mercado, dentre outros. Sua estrutura difere-se muito de um banco de dados operacional ou transacional, promovendo facilitadores para a recuperação desses dados.
3) **Bancos de dados temporais**: são aqueles que permitem ao usuário a consulta dos estados atuais e passados do banco de dados, pois armazenam históricos das alterações. Veja alguns exemplos de aplicações, em que o controle e acesso a informações temporais são fundamentais: área médica (evolução do paciente), área empresarial (aplicações financeiras, controle de produção, gerenciamento de vendas, gestão de pessoas), controle acadêmico, sistemas de informações geográficas, sistemas de reservas, entre outros.
4) **Bancos de dados espaciais**: são aqueles que permitem consultas a objetos localizados em um espaço multidimensional. É o caso dos bancos de dados geográficos, que

armazenam informações sobre mapas para localização de rios, cidades, estados, estradas, entre outros.

5) **Bancos de dados meteorológicos**: são aqueles que armazenam informações sobre o tempo.
6) **Bancos de dados de multimídias**: são aqueles que armazenam os dados sob a forma de imagens, vídeos, músicas, textos falados ou escritos, entre outros.
7) **Bancos de dados especialistas**: também conhecidos como **sistemas baseados em conhecimento**, são aqueles que, por meio de técnicas aplicadas na área da Inteligência Artificial, incorporam raciocínio e inferência (capacidade de dedução).

Os bancos de dados também podem ser classificados de acordo como os dados são estruturados. Dados não estruturados são aqueles que existem em seu estado original (estado bruto), no formato em que foram coletados. Entretanto, esse formato não permite o processamento adequado para produzir informações. Já os dados estruturados são o resultado da obtenção de dados não estruturados, como também sua formatação, facilitando, assim, o armazenamento, a manipulação e a geração de informações. O formato (estrutura) é utilizado baseando-se no tipo de processamento que desejamos realizar nos dados. Alguns dados podem não estar prontos (não estruturados) para determinados tipos de processamento, todavia, podem estar prontos (estruturados) para outros tipos. Podemos utilizar, como exemplo, o valor de um dado específico, como o valor 140070131, o qual pode referir-se a um CEP, um valor de vendas ou um código de um determinado produto. Por um lado, caso represente um CEP ou um código de produto e for armazenado como texto, não será permitido a realização de cálculos matemáticos com ele. Por outro lado, se o mesmo valor representar uma transação de vendas, torna-se necessário formatá-lo como numérico.

Contudo, a maioria dos dados encontrados são classificados como semiestruturados, que são aqueles parcialmente processados.

Recentemente, as necessidades de armazenamento e gerenciamento de dados não estruturados e semiestruturados estão sendo atendidas pela nova geração de bancos de dados, nomeados de XML. A XML (sigla em inglês para *Extensible Markup Language*) é considerada uma linguagem especial utilizada para representar e manipular elementos de dados em formato textual. Os bancos de dados em XML suportam o armazenamento e gerenciamento de dados semiestruturados em XML. A Tabela 1 realiza a comparação dos Sistemas de Gerenciamento de Bancos de Dados mais conhecidos no mercado.

Tabela 1 Tipos de bancos de dados.

PRODUTO	NÚMERO DE USUÁRIOS		LOCALIZAÇÃO DE DADOS		UTILIZAÇÃO DE DADOS		XXML
	Monousuário	Multiusuário	Centralizado	Distribuído	Operacional	Data Warehouse	
PostgreSQL	X	X	X	X	X	X	X*
MS SQL Server	X**	X	X	X	X	X	X
DB2	X**	X	X	X	X	X	X
MySQL	X	X	X	X	X	X	X*
Oracle (SGBDR)	X**	X	X	X	X	X	X

* Promove suporte apenas a funções de XML. Os dados em XML são armazenados em grandes objetos de texto.

** O fornecedor oferece versão específica do SGBD.

Ao analisarmos a Tabela 1, podemos perceber as especificidades que cada produto apresenta em relação à classificação dos bancos de dados.

6. SISTEMAS DE ARQUIVOS *VERSUS* SGBDS

Você já ouviu falar em FAT, FAT 32, NTFS? E SQL Server, Oracle e MySQL?

Todas essas siglas têm um objetivo em comum: organizar e armazenar dados em sistemas computacionais. Elas fazem parte de dois sistemas para controle de informação, o Sistema de Arquivos e o Sistema Gerenciador de Banco de Dados (SGBD).

Sistema de arquivos é o método de organizar e armazenar informações seguindo uma estrutura lógica para alocação física dos arquivos em dispositivos de armazenamento, tais como: disco rígido ou CD-ROM. Em outras palavras, o sistema de arquivos é a estrutura que indica como os arquivos devem ser gravados e guardados em algum sistema de armazenamento. A maioria dos sistemas de arquivos utiliza recursos de armazenamento de dados que se apresentam como um conjunto de blocos com tamanho fixo, denominado de setores, cada qual com 512 *bytes*.

O controle de acesso aos discos de armazenamento é realizado por sistemas operacionais como MS-DOS, Windows, Linux ou Unix. O *software* dos sistemas de arquivos é responsável por organizar esses setores em arquivos e diretórios, além de manter a informação sobre a qual setor pertence um arquivo e qual setor está disponível.

Existem sistemas de arquivos que você provavelmente utiliza no seu dia a dia, conhecidos como FAT (*File Allocation Table*), FAT32 e NTFS (*New Technology File System*), presentes no sistema operacional Windows. A diferença entre eles está relacionada às limitações da tecnologia que foram superadas e à necessidade de melhorias, como segurança e capacidade de armazenamento. Devido à confiabilidade presente no sistema de arquivos, o FAT e o FAT32 foram incorporados ao sistema de arquivos NTFS.

Os sistemas de arquivos funcionam por meio de uma espécie de tabela, que contém indicações da localização das informações de cada arquivo. Quando um arquivo é salvo em um disquete, por exemplo, o FAT divide a área do disco em pequenos blocos. Assim, um arquivo ocupa vários desses blocos, mas eles não precisam estar em uma sequência. Os blocos de determinados arquivos podem estar em várias posições diferentes. Por isso, existe a necessidade de uma tabela para indicar cada bloco.

Observe, na Figura 2, a representação de um sistema de arquivos. É importante lembrar que aplicativos são programas computacionais desenvolvidos que utilizam uma linguagem de programação com a finalidade de solucionar problemas e auxiliar nas atividades computacionais.

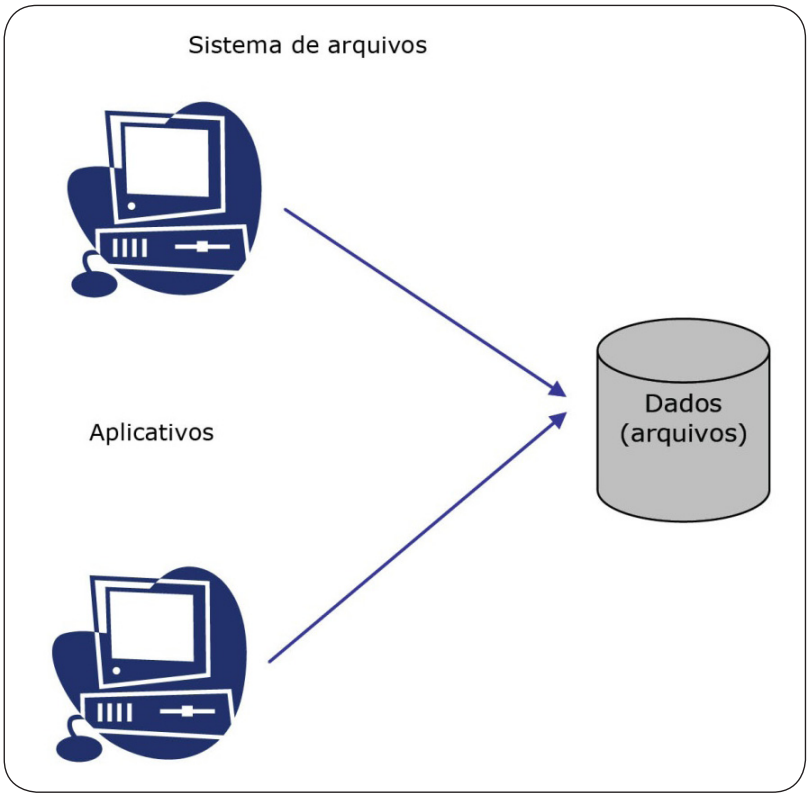


Figura 2 *Sistema de Arquivos.*

Arquitetura de sistemas é a forma como o sistema terá os seus componentes organizados e estruturados para realizar a tarefa determinada. A arquitetura contém um plano e uma lógica, em que o sistema desenvolvido executará suas atividades.

O sistema de arquivos foi uma das primeiras arquiteturas de sistemas para armazenamento e manipulação de dados e geração de informação, e, ao longo do tempo, sofreu alterações e foi adaptado conforme a evolução tecnológica. No entanto, alguns problemas ainda são encontrados, tais como:

1) A manutenção: ela é prejudicada, pois a estrutura de arquivos é definida e padronizada no próprio código do aplicativo.
2) O compartilhamento de um arquivo por vários programas: apresenta dificuldades para gerenciar o acesso a esses arquivos e seu controle.
3) As incompatibilidades nos aplicativos: o desenvolvimento de arquivos e programas de um mesmo sistema operacional, ao ser realizado isoladamente por diferentes programadores, e até mesmo em linguagens diferentes, causa tais incompatibilidades, prejudicando as funcionalidades do sistema.
4) A inconsistência (aplicativos com informações incorretas), o isolamento de dados (formatos distintos), a redundância (dados repetidos) e os problemas com segurança (fácil acesso aos registros do sistema).
5) A falta de gerenciamento para acessos concorrentes (vários usuários acessando a mesma informação) aos dados e recuperação de dados.

Observe, a seguir, um exemplo que engloba a inconsistência e a falta de gerenciamento de acesso. Suponha que dois alunos estejam consultando o acervo de uma biblioteca digital à procura do mesmo livro. A biblioteca possui um único exemplar disponível para empréstimo, e essa informação é apresentada para os dois alunos que consultam o acervo. Ambos realizam a consulta e fazem a reserva no mesmo instante. Isso causará um problema no sistema, pois haverá inconsistência de dados. A inconsistência e a falta do gerenciamento da informação geraram

uma falha: o sistema não foi capaz de bloquear a reserva de um dos estudantes e dizer que o livro já havia sido reservado para o outro.

Você percebe a necessidade das informações serem atualizadas em tempo real? Caso contrário, o que aconteceria com os sistemas bancários, os sistemas que realizam vendas pela internet ou reservas de voos aéreos?

Com as exigências e necessidades de novos conceitos e estruturas nos sistemas de arquivos, percebeu-se um aumento significativo nos custos de equipamentos, manutenção e tempo de trabalho para atender todas essas novas demandas. Dessa forma, os Sistemas de Gerenciamento de Banco de Dados (SGBD) surgiram como uma evolução dos Sistemas de Arquivos, criando novas estruturas de dados com o objetivo de gerenciar todo o armazenamento de informações.

O SGBD é uma coleção de programas que permite ao usuário definir, construir e manipular bases de dados para as mais diversas finalidades. Os programas ou *softwares* SGBDs mais conhecidos são Microsoft Access, MySQL, Oracle, FireBird, SQL Server e PostgreSQL.

No Sistema de Gerenciamento de Banco de Dados (SGBD), o acesso aos dados e seu gerenciamento são realizados pelo próprio sistema, o qual funciona como uma **interface** (ponto que delimita e estabelece a relação entre dois sistemas independentes) entre o banco de dados e os programas aplicativos, isto é, o SGDB está localizado entre o banco de dados físico e os usuários.

Um Sistema Gerenciador de Banco de Dados é responsável por receber as requisições provenientes de diversas aplicações e realizar a tradução, quebrando a complexidade das solicitações e permitindo o atendimento a essas requisições adequadamente. O SGBD consegue ocultar dos usuários e aplicativos grande parte da complexidade existente em um banco de dados. Normalmente, os aplicativos computacionais que interagem com os bancos de dados por meio de um Sistema Gerenciador de Banco de Dados podem ser implementados pelos programadores utilizando diversas linguagens de programação, como por exemplo, o Visual Basic, Dot Net, Java, PHP ou C++. Além do uso de linguagens de programação específicas, existe a possibilidade de desenvolver aplicativos baseando-se exclusivamente em ferramentas proprietárias dos Sistemas Gerenciadores de Banco de Dados, como por exemplo, o Forms e Reports da Oracle.

Como vimos anteriormente, os dados constituem um material bruto fundamental a partir do qual as informações são obtidas, e requerem um excelente método para gerenciá-los. Você descobrirá que o SGBD torna o gerenciamento de dados mais eficientes e eficazes. Como exemplo, mencionamos suas diversas vantagens na utilização em aplicações computacionais. O SGBD:

1) Permite o compartilhamento dos dados para diversas aplicações e usuários.
2) Integra as visões dos dados dos diferentes usuários em um único repositório de dados.
3) Fornece modelos adequados para melhor aplicar as políticas de privacidade e segurança de dados.
4) Reduz a inconsistência dos dados, que ocorre quando diferentes versões dos mesmos dados aparecem em locais distintos. Por exemplo, quando o departamento de *marketing* de uma determinada empresa armazena o nome de uma funcionária como "Gisele Ap. da Silva" e o departamento de recursos humanos, por sua vez, armazena o nome da mesma funcionária como "Gisele A. Silva".
5) Dados bem gerenciados e com acesso aprimorado permitem a geração de informações de melhor qualidade, que, por sua vez, auxiliam na tomada de decisões.

As vantagens da utilização de um SGBD não se limitam aos itens listados. Certamente, você descobrirá inúmeras utilidades ao conhecer os detalhes dos bancos de dados e seu projeto adequado.

O SGBD apresenta uma visualização única e integrada ao usuário (ou aplicativos), conforme pode ser visualizado na Figura 3.

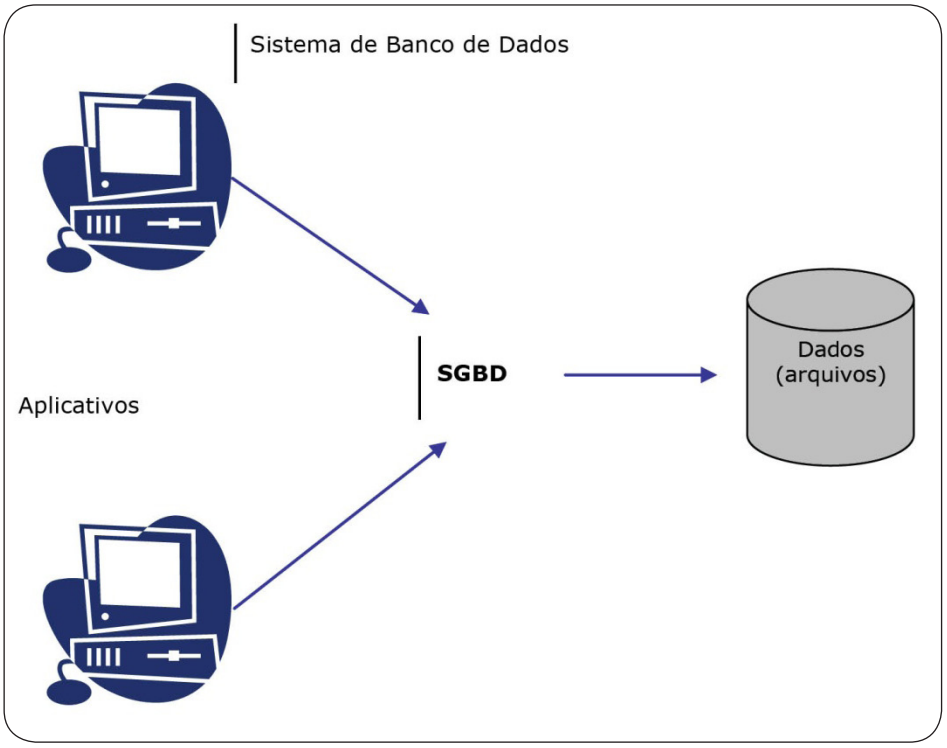


Figura 3 *Sistema de Banco de Dados.*

O SGBD solucionou problemas dos sistemas de arquivos, tais como: integração de dados (mesmo local), dados duplicados, independência de dados e aplicativos e representação das perspectivas do usuário.

Desenvolvedores, administradores e usuários que trabalham diretamente com SGBDs devem ter conhecimento e domínio dos recursos para usufruir de todas as vantagens que o sistema proporciona, como:

1) Rapidez no acesso às informações presentes no banco de dados.
2) Redução de problemas de integridade e redundância.
3) Diminuição do esforço humano no desenvolvimento.
4) Utilização dos dados e controle integrado de informações distribuídas fisicamente.

Desconhecer o funcionamento do SGBD pode acarretar o mau desenvolvimento e afetar a segurança de acesso às informações.

Uma das comparações realizadas entre o sistema de arquivos e o SGBD está relacionada ao desempenho. O sistema de arquivos é programado para uma aplicação específica, o que gera um bom desempenho. Já o SGBD é programado para aplicações mais genéricas, podendo atender a aplicações diferentes.

Em outras palavras, podemos dizer que os SGBDs vieram para eliminar todo o trabalho realizado anteriormente por um programador de aplicação, responsável pelo controle do acesso, da integridade e da redundância dos dados; contudo, ainda há alguns itens que são melhor atendidos pelos sistemas de arquivos.

7. DADOS EM SGBDS: DESCRIÇÃO E ARMAZENAMENTO

Os dados são informações que podem ser gravadas e que possuem um significado implícito. O banco de dados (BD) é uma coleção de dados relacionados que:

- Representa aspectos do mundo real (minimundo ou universo de discurso). Portanto, as mudanças que ocorrem no mundo real devem ser refletidas no BD.
- Descreve uma coleção lógica e coerente de dados com algum significado inerente. Uma organização randômica, ou seja, aleatória, de dados não pode ser considerada um BD.
- Constrói em atendimento a uma proposta específica.

Um Sistema Gerenciador de Banco de Dados (SGBD) é uma coleção de programas que permite aos usuários criarem e manterem um banco de dados. Ele foi definido para ser um sistema de *software* de propósito geral que facilita os processos de definição, construção, manipulação e compartilhamento de bancos de dados entre vários usuários e aplicações.

São características oferecidas pelos SGBDs:

1) **Rapidez**: agilidade na execução das consultas *on-line*.
2) **Disponibilidade total**: significa que todas as vezes que uma informação for solicitada, ela deve estar disponível e atualizada.
3) **Flexibilidade**: facilidade na implementação de mudanças.
4) **Integridade**: garantia da consistência dos dados quando atualizações são efetuadas no banco.

Para atingir os objetivos propostos pelo SGBD, você verá que é necessário ter uma estrutura com níveis bem definidos e utilizar um modelo de dados adequado para descrever o banco de dados.

Níveis de Abstração de Dados

Normalmente, quando constituímos um projeto de banco de dados, iniciamos a partir de uma visão abstrata do ambiente como um todo, acrescentando detalhes à medida que o projeto evolui para a sua implementação. Utilizar níveis de abstração de dados pode ser extremamente útil no que diz respeito à integração de visões múltiplas de dados, como podemos presenciar em distintos níveis de uma empresa.

O Comitê de Planejamento e Exigência de Padrões (SPARC) do Instituto Nacional Americano de Padrões (ANSI) definiu, no início da década de 1970, uma estrutura de modelagem com base em níveis de abstração de dados. A arquitetura ANSI/SPARC define três níveis de abstração de dados: externo, conceitual e interno.

Modelo Externo

Esse modelo é considerado a perspectiva dos usuários finais do ambiente de dados. Usuários finais é o termo dado às pessoas que fazem uso de aplicativos para manipular os dados, gerando, assim, as informações necessárias para o cenário empresarial.

Normalmente, as empresas são segmentadas em diversas unidades comerciais, como: vendas, finanças e *marketing*, em que cada unidade pode estar sujeita a restrições e exigências específicas. Dessa maneira, os usuários finais podem visualizar apenas seus subconjuntos de dados específicos, separados das outras unidades da empresa.

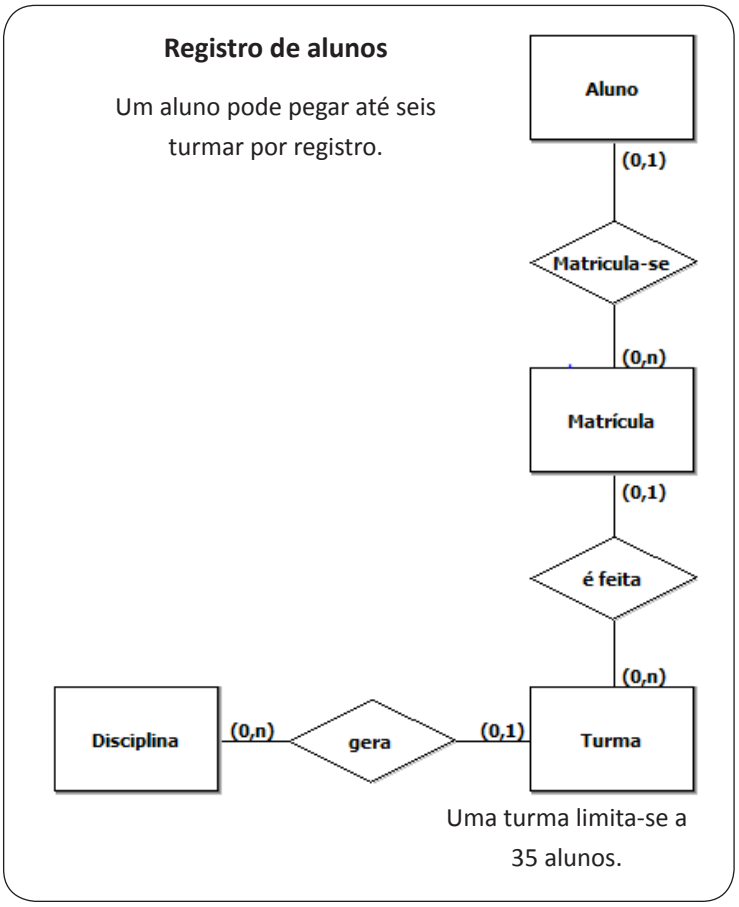


Figura 4 *Modelos externos de uma instituição acadêmica Sistema de Banco de Dados.*

Na Figura 4, imagine os esquemas externos de duas unidades comerciais correspondentes a uma instituição acadêmica: registro de alunos e agendamento de turmas. Cada esquema externo inclui as entidades, relacionamentos e restrições determinadas pela unidade comercial (departamento). Observamos que, embora as visões das aplicações sejam isoladas, cada visão compartilha sua unidade com a outra (os esquemas externos pertinentes ao registro de alunos e o agendamento de turmas compartilham as entidades TURMA e DISCIPLINA).

Em detalhes, podemos descrever as entidades-relacionamentos representados pela Figura 4:

1) Um PROFESSOR ensina muitas TURMAs, porém, cada TURMA é ensinada por apenas um PROFESSOR. Ou seja, existe um relacionamento **1:M** entre PROFESSOR e TURMA.
2) Uma TURMA pode receber a MATRÍCULA de diversos alunos, e para cada aluno é permitida a MATRÍCULA em várias TURMAs, formando, assim, um relacionamento **M:N** entre ALUNO e TURMA.
3) Para cada DISCIPLINA é permitida a geração de muitas TURMAs, mas cada TURMA se refere apenas a uma DISCIPLINA, conforme pode ser observado na cardinalidade mínima e máxima.
4) Por fim, uma TURMA normalmente necessita de uma SALA, mas uma SALA pode ser reservada para várias TURMAs. Ou seja, cada sala de aula pode ser utilizada por diversas turmas, uma em cada horário, por exemplo. Dessa forma, podemos identificar a existência de um relacionamento **1:M** entre SALA e TURMA.

Modelo Conceitual

Após a identificação das visões externas, utilizamos o modelo conceitual, o qual é representado graficamente pelo diagrama de entidade e relacionamento (DER) que, na prática, constitui a planta básica do banco de dados. O modelo conceitual tem como objetivo realizar a integração de todas essas visões externas (entidades, relacionamentos e restrições) em uma visão global de todos os dados da empresa.

O ER (entidade-relacionamento) é o modelo conceitual mais utilizado. Dentre as vantagens apresentadas pelo modelo conceitual, podemos destacar:

- Fornecimento de uma visão macroscópica de fácil entendimento sobre o ambiente de dados.
- Independência de *software*: o modelo não depende da tecnologia do Sistema de Gerenciamento de Banco de Dados (SGBD) utilizado em sua implementação, ou seja, as alterações provenientes do *software* não refletirão sobre o projeto de banco de dados no nível conceitual.
- Independência de *hardware*: o modelo não depende do *hardware* utilizado em sua implementação, ou seja, as alterações provenientes do *hardware* não refletirão sobre o projeto de banco de dados no nível conceitual.

Modelo Interno

Nessa fase, normalmente já definimos qual tecnologia de Sistema de Gerenciamento de Banco de Dados (SGBD) será utilizada. O modelo interno realiza o mapeamento do modelo conceitual para o SGBD específico.

Quando utilizamos o modelo relacional (o qual será detalhado adiante), escolhemos um banco de dados que suporta esse tipo para implementar o modelo interno, o qual se resume no mapeamento do modelo conceitual para as tabelas do modelo relacional. O esquema interno é constituído pela SQL (linguagem padrão) quando selecionamos o banco de dados relacional. Como exemplo, a Figura 5 apresenta o modelo interno implementado, criando-se as tabelas PROFESSOR, DISCIPLINA, TURMA, ALUNO, MATRÍCULA e SALA.

Devido à dependência do modelo interno a um *software* de SGBD específico, dizemos que ele é dependente de *software*. Qualquer alteração realizada na tecnologia do SGBD exigirá que o modelo interno sofra algum tipo de alteração a fim de se adequar às novas características e exigências de implementação do modelo de banco de dados. Em alguns casos, é possível alterarmos o modelo interno sem interferir no modelo conceitual, obtendo, assim, a independência lógica. Entretanto, o modelo interno possui independência de *hardware*, ou seja, esse modelo não sofrerá nenhum tipo de alteração/modificação pela escolha do computador em que o *software* (SGBD) será instalado.

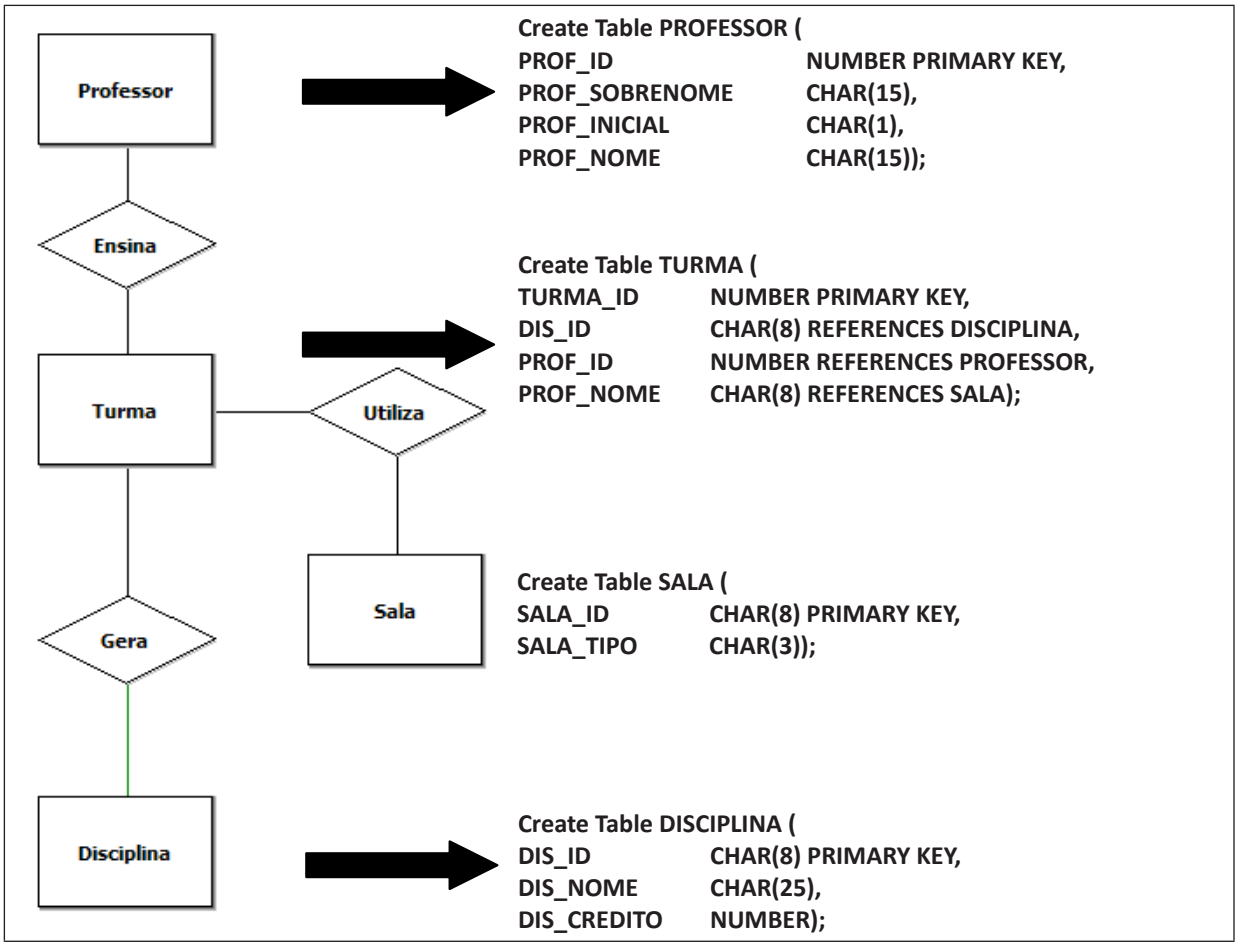


Figura 5 *Modelo Interno (Instituição Acadêmica)*.

Modelo Físico

Este modelo trabalha nos níveis mais baixos de abstração, ou seja, descreve a maneira como os dados são gravados em meios de armazenamento, como discos e fitas magnéticas. O modelo físico carece da definição dos dispositivos de armazenamento físico, como também seus métodos de acesso a esse meio. Consequentemente, podemos dizer que esse modelo é dependente tanto do *software* (SGBD) como do *hardware*.

Mesmo que no modelo relacional o projetista do banco de dados não precise se preocupar com as características inerentes ao armazenamento físico dos dados, a implementação de um modelo relacional poderá exigir sintonização (*tuning*) refinada no nível físico para otimizar o desempenho**.**

Sabemos que o modelo físico depende da tecnologia do SGBD, dos mecanismos de acesso aos arquivos e dos tipos de dispositivos utilizados para efetuar o armazenamento. Eventualmente, quando é possível realizar qualquer tipo de alteração no modelo físico sem influenciar o modelo interno, ocorre o que chamamos de independência física.

Modelo de Dados

Um modelo de dados é uma representação relativamente simples, em geral gráfica, de estruturas mais complexas de dados reais. Normalmente, o modelo é uma abstração de um objeto ou até mesmo um evento real com um maior grau de complexidade. Tem como função principal auxiliar na compreensão das regras de negócios complexos existentes em um ambiente real. Em

um cenário de bancos de dados, um modelo representa estruturas de dados e suas características, como: relações, restrições, transformações, dentre outros elementos que possuem o mesmo objetivo, ou seja, dar suporte ao problema específico de um determinado domínio de negócios.

Habitualmente, os projetistas de bancos de dados confiam no bom senso na hora de desenvolver bons modelos. Por exemplo, se os estudantes de uma turma têm de criar, individualmente, um modelo de dados para uma locadora de filmes, é muito provável que cada um apresente um modelo diferente. Qual estaria correto? A resposta é simples: "aquele que atender a todas as necessidades do usuário final", podendo haver mais de uma solução correta.

Um modelo de dados bem desenvolvido pode inclusive promover uma compreensão aprimorada da organização para a qual o banco está sendo desenvolvido. Em resumo, os modelos de dados são uma ferramenta de comunicação. Esse aspecto importante da modelagem foi sintetizado claramente por um cliente, cuja reação foi a seguinte: "Eu criei esta empresa, trabalhei nela por anos e esta é a primeira vez em que eu realmente entendi como todas as partes se encaixam na prática".

Da mesma forma que a planta de uma casa é uma abstração, o modelo de dados é, de modo similar, uma abstração; não é possível obter os dados necessários a partir do modelo. Se dificilmente se construirá uma boa casa sem uma planta, é também improvável criar um bom banco de dados sem a prévia criação de um modelo de dados adequado.

Objetos Básicos

Os blocos básicos de construção de todos os modelos de dados são as entidades, os atributos, os relacionamentos e as restrições.

Uma **entidade** é algo (pessoa, local, objeto ou evento) sobre o qual são coletados e armazenados dados. Ela representa um tipo particular de objeto no mundo real. Por isso, as entidades são **distinguíveis**, ou seja, cada ocorrência de entidade é única e distinta. Por exemplo, uma entidade **Cliente** teria muitas ocorrências de clientes distinguíveis, como Manoel Ribeiro, Pedro José da Silva, Dinorá Fernandes Junqueira etc. As entidades podem ser objetos físicos, como clientes e produtos, mas também podem ser abstrações, como rotas de voo ou apresentações musicais.

Um **atributo** é considerado uma característica de uma entidade. Por exemplo, uma entidade **Cliente** seria caracterizada pelos atributos nome, sobrenome, telefone, endereço e limite de crédito. Os atributos são equivalentes aos campos nos sistemas de arquivos.

Um **relacionamento** descreve uma associação (conectividade) entre entidades (uma ou **n**). Por exemplo, um relacionamento entre clientes e corretores pode ser descrito da seguinte maneira: um corretor pode atender muitos clientes, mas cada cliente pode ser atendido apenas por um corretor. Os modelos de dados utilizam três tipos de relacionamentos: um para muitos (**1:M** ou **1..***), muitos para muitos (**M:N** ou ***..***) e um para um (**1:1** ou **1..1**).

Veja, a seguir, exemplos que ilustram as distinções entre os três tipos de relacionamentos permitidos.

- **Relacionamento um para muitos** (**1:M** ou **1..***)**:** um pintor faz várias pinturas, mas cada uma é criada por apenas um artista; o pintor (uma entidade) relaciona-se com as pinturas (várias entidades). Portanto, podemos definir o relacionamento PINTOR pinta PINTURA como sendo **1:M**. De modo similar, um cliente (entidade) pode gerar muitas faturas (várias entidades), mas cada fatura é gerada apenas por um cliente. O relacionamento CLIENTE gera FATURA também seria identificado como **1:M**.

- **Relacionamento muitos para muitos** (**M:N** ou ***..***)**:** um funcionário pode aprender várias habilidades profissionais e cada habilidade profissional pode ser aprendida por vários funcionários. Esse tipo nos permite identificar o relacionamento FUNCIONÁRIO aprende HABILIDADE como **M:N**. De modo similar, um aluno pode frequentar várias turmas e cada turma pode ser frequentada por vários alunos, conferindo-se, assim, a identificação **M:N** ao relacionamento expresso por ALUNO frequenta TURMA.
- **Relacionamento um para um** (**1:1** ou **1..1**)**:** a estrutura de gerenciamento de uma empresa de varejo pode exigir que cada uma de suas lojas seja gerenciada por um único funcionário. Por sua vez, cada gerente de loja, que é um funcionário, gerencia apenas uma loja. Portanto, o relacionamento FUNCIONÁRIO gerencia LOJA é identificado como **1:1**.

A discussão precedente identificou cada relacionamento em duas direções, ou seja, os relacionamentos são bidirecionais (grau dois ou binário). Veja:

- Um CLIENTE pode gerar diversas faturas.
- Cada uma das várias FATURAS é gerada apenas por um CLIENTE.

A **restrição** é uma limitação imposta aos dados. As restrições são importantes, pois ajudam a assegurar a integridade dos dados. Elas normalmente são expressas na forma de regras. Por exemplo:

- O salário de um funcionário possui valores entre R$ 1.000,00 e R$ 5.000,00.
- A média de nota de um aluno deve estar entre o intervalo 0,0 e 10,0.
- Cada turma deve ter um e apenas um professor.

Como é possível identificar, de maneira adequada, as entidades, os atributos, os relacionamentos e as restrições? A primeira etapa é identificar as regras de negócio para o domínio do problema que está sendo modelado.

Uma regra de negócio é uma descrição breve, precisa e sem ambiguidade de uma política, procedimento ou princípio em uma determinada organização. As regras de negócio se aplicam a qualquer tipo de organização, seja ela grande ou pequena – uma empresa, uma instituição pública, um grupo religioso ou um laboratório de pesquisa – que armazene e manipule dados para gerar informações importantes na tomada de decisões.

Regras de negócio adequadas são utilizadas para definir entidades, atributos, relacionamentos e restrições. Sempre que você vir uma descrição de relacionamento como "um corretor pode atender muitos clientes, mas cada cliente pode ser atendido por apenas um corretor", você verá as regras de negócio em ação.

Essas regras descrevem, em linguagem simples, as principais características dos dados conforme vistos pela empresa. Os seguintes itens são exemplos de regras de negócio:

- Um cliente pode gerar muitas faturas.
- Uma fatura é gerada por apenas um cliente.
- Uma seção de treinamento não pode ser agendada para menos de 10 funcionários ou mais de 30.

Observe que essas regras de negócio estabelecem entidades, relacionamentos e restrições. Por exemplo, as duas primeiras regras estabelecem duas entidades (CLIENTE e FATURA) e um relacionamento **1:M** entre elas. A terceira regra de negócio estabelece uma restrição (não menos de 10 pessoas e não mais de 30), duas entidades (FUNCIONÁRIO e TREINAMENTO) e um relacionamento entre FUNCIONÁRIO e TREINAMENTO.

Convertendo Regras de Negócio em Modelos de Dados

As regras de negócio constituem um cenário adequado para a identificação correta dos objetos do modelo de dados, como suas entidades, atributos, relacionamentos e restrições. Em um cenário real, os nomes são utilizados para identificar objetos. Se o ambiente de negócio desejar rastrear os objetos, haverá regras específicas para eles. Geralmente, um substantivo em uma regra de negócio será traduzido como entidade no modelo de dados, e um verbo que associe substantivos será traduzido como um relacionamento entre entidades. Por exemplo, a regra de negócio "um CLIENTE pode GERAR várias FATURAS" contém dois substantivos (CLIENTES e FATURAS) e um verbo (GERAR) que associa os substantivos. Conclui-se a partir dessa regra que:

- CLIENTE e FATURA são objetos de interesse para o ambiente e devem ser representados por suas respectivas **entidades**.
- Há um **relacionamento** GERAR entre as entidades CLIENTE e FATURA.

Para identificar adequadamente o tipo, deve-se considerar que os relacionamento são bidirecionais, ou seja, valem em ambos os sentidos. Por exemplo, a regra "um CLIENTE pode GERAR várias FATURAS" é complementada pela regra "uma FATURA é **gerada** por apenas um CLIENTE". Nesse caso, o relacionamento é um para muitos (**1:M**). O CLIENTE é o lado **um** e a FATURA, o lado **muitos**. Normalmente, para fazer essa identificação do tipo de relacionamento, devemos fazer duas perguntas:

- Quantas **instâncias** de B são **relacionadas** a uma **instância** de A?
- Quantas **instâncias** de A são **relacionadas** a uma **instância** de B?

É possível avaliar o relacionamento entre ALUNO e DISCIPLINA fazendo as seguintes perguntas:

- Em quantas disciplinas um aluno pode se matricular? Resposta: várias disciplinas.
- Quantos alunos podem se matricular em uma disciplina? Resposta: vários alunos.

Portanto, o relacionamento entre ALUNO e DISCIPLINA é de muitos para muitos (**M:N**).

A busca por um melhor gerenciamento de dados gerou vários modelos que tentaram resolver as falhas fundamentais do sistema de arquivos. Na Tabela 2 podemos ter uma visão geral dos principais modelos de dados, em ordem cronológica.

Tabela 2 Detalhes sobre a evolução dos principais modelos de dados.

GERAÇÃO	Época	MODELO	EXEMPLOS	COMENTÁRIOS
Primeira	Década de 1960 a 1970	Sistema de arquivos	VMS/VSAM	Utilizado pela IBM nos sistemas de *mainframe*. Sem utilizar relacionamentos, gerenciava os registros.
Segunda	Década de 1970	Modelo de dados hierárquico e em rede	IMS ADABAS IDS-II	Caracterizavam os primeiros bancos de dados.
Terceira	Década de 1970 até o presente	Modelo de dados relacional	DB2 Oracle MS SQL Server MySQL PostgreSQL	Conceitos básicos simples e objetivos. Modelagem entidade-relacionamento (ER) e suporte à modelagem relacional de dados.

GERAÇÃO	Época	MODELO	EXEMPLOS	COMENTÁRIOS
Quarta	Década de 1980 até o presente	Orientado a objetos relacionais estendidos	Versant Caché FastObjects.Net Objectivity/DB DB/2 UDB	Promove suporte a dados complexos. Produtos relacionais estendidos com suporte a *warehouse* de dados e objetos. Propagação de banco de dados na web.
Próxima geração	Do presente ao futuro	XML	dbXML Tamino DB2 UDB Oracle 10g MS SQL Server PostgreSQL	Promove organização e gerenciamento de dados não estruturados. Modelos relacionais e de objetos adicionam suporte a documentos em XML.

Modelo Hierárquico

Desenvolvido na década de 1960, o modelo hierárquico tinha por objetivo gerenciar grandes quantidades de dados provenientes de projetos complexos. Podemos mencionar como exemplo o foguete Apollo, que aterrissou na Lua em 1969. Sua estrutura lógica é representada por uma estrutura semelhante à de uma árvore, visualizada de cima para baixo, onde identificamos seus níveis ou segmentos. Um segmento é semelhante ao tipo de registro em um sistema de arquivos qualquer. Internamente, por hierarquia, a camada considerada superior (raiz) é identificada como **pai** do segmento imediatamente abaixo dela. Na Figura 6 podemos visualizar o segmento considerado **Raiz** como o pai dos segmentos do **Nível 1** que, por sua vez, são pais dos segmentos do **Nível 2**, e assim sucessivamente. Os segmentos encontrados abaixo de outros segmentos são identificados como **filhos**. Resumidamente, podemos considerar que o modelo hierárquico representa um conjunto de relacionamentos um para muitos (**1:M**) entre um **pai** e seus **filhos**, ou seja, cada **pai** pode ter **muitos filhos**, entretanto, cada **filho possui apenas um pai**.

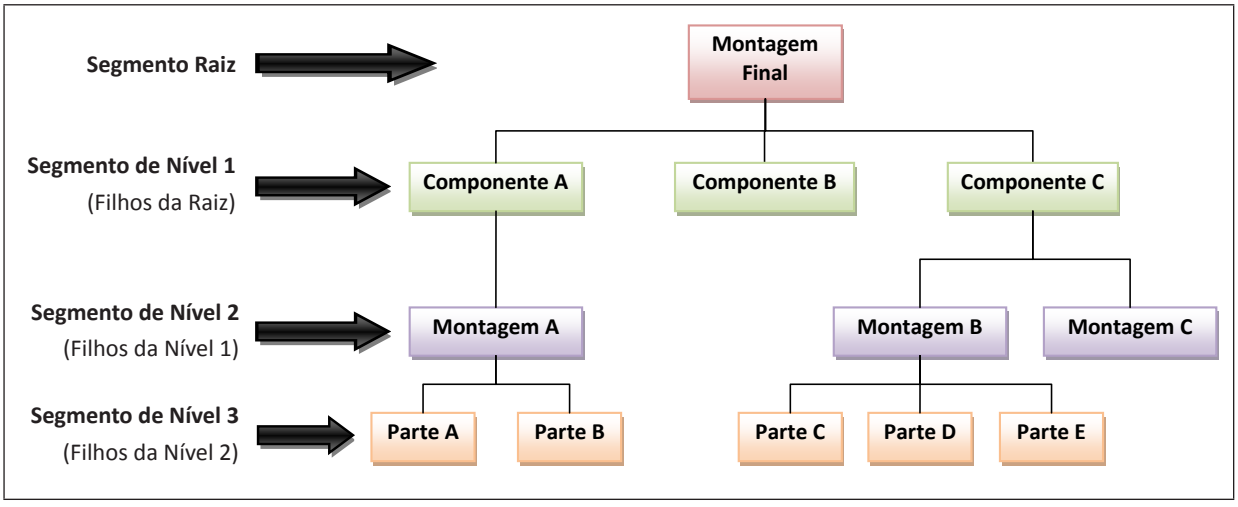


Figura 6 *Estrutura Hierárquica*.

O banco de dados hierárquico tornou-se referência na década de 1970, somando uma série de vantagens em relação aos sistemas de arquivo. Consequentemente, gerou uma base significante para o desenvolvimento de aplicações comerciais. Porém, o modelo hierárquico apresentava diversas limitações, entre elas a dificuldade de implementação e gerenciamento,

a falta de independência estrutural e o fato de a maioria dos relacionamentos de dados mais comuns se adaptarem à forma 1:M.

Modelo em Rede

O modelo em rede foi criado para representar, exclusivamente, relacionamentos de dados complexos com maior eficiência (em comparação ao modelo hierárquico), otimizando o desempenho dos bancos de dados e determinando um padrão para os mesmos.

No modelo em rede, em geral, o usuário visualiza o banco de dados em rede como uma coleção de registros em relacionamentos **1:M**. Ao contrário do modelo hierárquico, esse modelo permite que um registro tenha mais de um pai. Um exemplo desse relacionamento é apresentado na Figura 7.

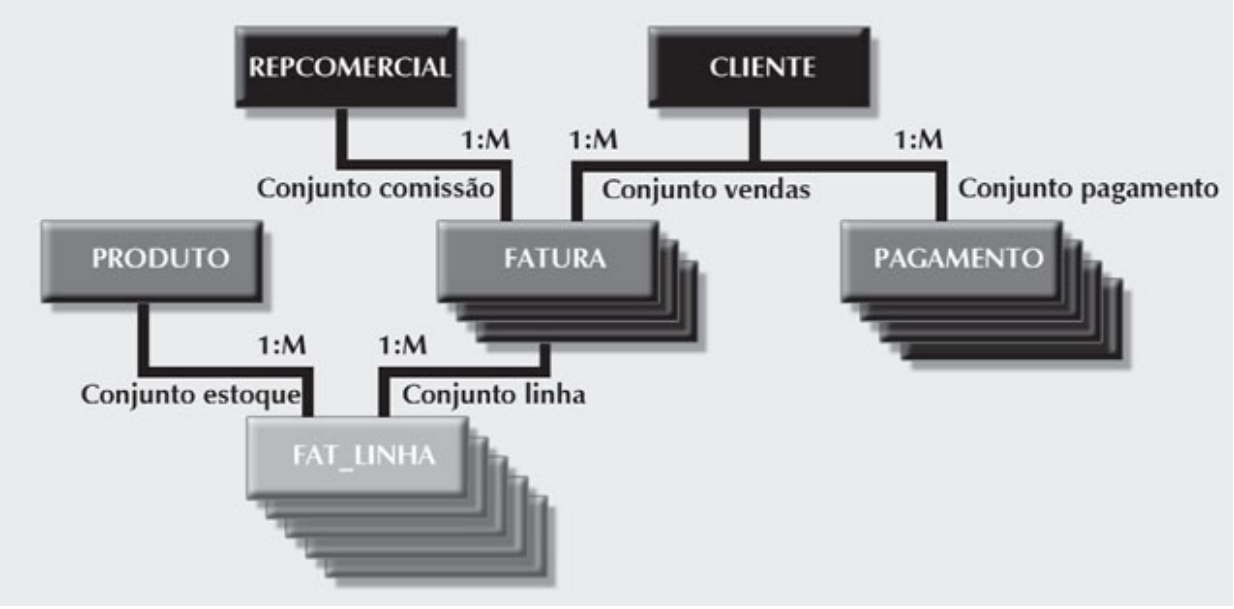


Figura 7 *Modelo de dados de rede.*

A Figura 7 ilustra um modelo de dados em rede de uma empresa de vendas qualquer. Nesse modelo podemos identificar os tipos de registros: CLIENTE, REPCOMERCIAL, FATURA, FAT_LINHA, PRODUTO e PAGAMENTO. Observe que a FATURA é propriedade tanto do REPCOMERCIAL como do CLIENTE. De maneira similar, FAT_LINHA possui dois proprietários, PRODUTO e FATURA.

A ausência de um recurso de consulta *ad hoc* não permitia aos desenvolvedores a geração do código necessário para a produção de simples relatórios. Outra desvantagem considerável é a independência de dados totalmente limitada: qualquer tipo de alteração estrutural, por mais simples que fosse, poderia devastar todos os aplicativos que obtinham dados do banco. Devido a essas restrições, os modelos hierárquicos e em rede foram, consequentemente, substituídos pelo modelo de dados relacional na década de 1980.

Modelo Relacional

O modelo relacional foi apresentado, em 1970, por Codd (da IBM), em seu famoso artigo *A Relational Model Data for Large Shared Data Banks* (Um modelo relacional de dados para grandes bancos de dados compartilhados).

A base do modelo relacional é calcada no conceito matemático conhecido como relação. Na tentativa de diminuir a complexidade da teoria matemática abstrata, podemos pensar uma relação (também chamada de tabela) como sendo uma matriz composta por linhas e colunas.

O modelo relacional é normalmente implementado por meio de um Sistema de Gerenciamento de Banco de Dados Relacionais (SGBDR). Na prática, o SGBDR executa as mesmas funções básicas fornecidas pelos Sistemas de Gerenciamento de Banco de Dados Hierárquico e de Rede, incorporando também outras funções que tornam o modelo relacional mais simples de entender e implantar.

Uma das vantagens mais significantes do SGBDR é a maneira de ocultar do usuário as complexidades do modelo relacional, gerenciando todos os detalhes físicos e permitindo que o usuário apenas visualize o banco de dados relacional como uma coleção de tabelas nas quais os dados são armazenados.

Por meio do compartilhamento de um determinado atributo comum (valor de uma coluna), as tabelas são relacionadas entre si. A Figura 8 apresenta a tabela nomeada de CLIENTE que contém um número de corretor, que, por sua vez, também está contida na tabela CORRETOR.

Nome da Tabela: CORRETOR

AGENT_CODE	AGENT_NAME	AGENT_FNAME	AGENT_INITIAL	AGENT_AREACODE	AGENT_PHONE
501	Alby	Alex	B	713	123-3456
502	Hahn	Leah	F	615	245-5455
503	Okon	John	T	615	234-2353

Ligação por meio de AGENT_CODE

Nome da Tabela: CLIENTE

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREA_CODE	CUS_PHONE	CUS_INSURE_AMT	ATEND_CODE
10010	Ramas	Alfred	A	713	123-3456	100	502
10011	Dunne	Leona	K	615	245-5455	250	501
10012	Smith	Kathy	W	615	234-2353	313	502
10013	Olowski	Paul	F	456	456-3434	452	502
10014	Orlando	Myron		324	342-5633	346	501
10015	O'Brian	Amy	B	657	345-3212	466	503
10016	Brown	James	G	864	345-3456	233	502
10017	Williams	George		342	345-5644	322	503
10018	Farriss	Anne	G	567	246-3344	34	501
10019	Smith	Olette	K	343	247-3456	455	503

Figura 8 *Relacionamento entre tabelas.*

O relacionamento entre as tabelas CLIENTE e CORRETOR permite que você relacione o cliente com seu corretor de vendas, mesmo que os dados dos clientes estejam armazenados em uma tabela e os dos corretores estejam localizados em outra. Por exemplo, é possível vincular facilmente o cliente Leona Dunne com seu respectivo corretor, ora identificado como Alex Alby, pois, para o cliente Leona Dunne, o código AGENT_CODE (código do corretor) da tabela CLIENTE é 501 (que, por sua vez, corresponde ao AGENT_CODE de Alex Alby na tabela CORRETOR).

Podemos afirmar que o modelo relacional tornou-se a base fundamental de uma revolução real dos bancos de dados existentes hoje em dia, especialmente pelo fato de incorporar a poderosa e flexível linguagem de consulta conhecida como SQL (*Structured Query Language*), a qual permite que o usuário especifique o que deve ser feito sem a necessidade de especificar como se deve fazê-lo.

Modelo Entidade-Relacionamento

Embora o modelo relacional represente um aprimoramento em relação aos modelos hierárquico e em rede, alguns recursos ainda eram escassos para que ele fosse avaliado como uma ferramenta de projeto eficiente. Como é mais objetivo representar estruturas gráficas em vez de descrevê-las em texto, os projetistas de bancos de dados preferem utilizar a ferramenta gráfica, pois esta permite que as entidades e seus relacionamentos sejam visualizados de maneira simplificada. Dessa forma, o modelo de entidade-relacionamento (ER) ou MER (ERM, sigla em inglês para *Entity Relationship Model*) tornou-se um padrão aceito mundialmente para modelagem de dados.

O famoso Peter Chen apresentou pela primeira vez, em 1976, o modelo de dados ER, o qual tratava da representação gráfica clara e intuitiva de entidades e seus respectivos relacionamentos em uma estrutura de banco de dados. Tal representação tornou-se popular, pois complementava de forma satisfatória os conceitos do modelo relacional, o qual foi mesclado com o MER para construir uma base sólida do projeto de bancos de dados fortemente estruturado. Os modelos ER são, geralmente, simbolizados por um diagrama de entidade-relacionamento (DER), que utiliza representações gráficas para modelar os componentes do banco de dados.

Veja, a seguir, os componentes que constituem o modelo ER:

- **Entidade**: representada no DER por um retângulo, e sua identificação é feita por um substantivo, escrito de maneira centralizada. Normalmente, é escrito em letras maiúsculas e no singular: PROFESSOR em vez de PROFESSORES e FUNCIONÁRIO em vez de FUNCIONÁRIOS. Quando utilizamos o DER no modelo relacional, é frequente que uma entidade seja mapeada para uma tabela relacional, onde cada linha é nomeada como instância de entidade ou ocorrência de entidade no modelo ER.

  Cada entidade é definida por um conjunto de atributos que descrevem suas características específicas. Por exemplo, a entidade FUNCIONÁRIO possuirá como atributos o nome e data de nascimento.

- **Relacionamento**: tem como principal objetivo realizar a associação entre os dados. A grande parte dos relacionamentos faz a conexão (relaciona) entre duas entidades. Sua identificação, em geral, é realizada por um verbo. São exemplos: um PINTOR **pinta** várias PINTURAS; um FUNCIONÁRIO **aprende** várias HABILIDADES; um FUNCIONÁRIO **gerencia** uma LOJA.

Nas Figuras 9 e 10 podem ser visualizados os diferentes tipos de relacionamento. Os dois casos fazem uso de notações de ER: a notação original de Peter Chen e a notação Crow's Foot (pé de galinha), considerada a notação mais atual.

A Figura 9 apresenta a notação de Peter Chen, em que as conectividades (relacionamentos) são descritas próximas a cada entidade. Graficamente, os relacionamentos são representados por um losango, o qual é conectado às entidades relacionadas por meio de uma reta. A identificação de um relacionamento é escrito no interior do losango.

Já a Figura 10 ilustra a notação pé de galinha. Esse nome é consequência da utilização do símbolo de três pontas, o qual representa o lado **muitos** do relacionamento. Podemos observar que as conectividades oriundas do DER básico que usa a notação pé de galinha são representadas por símbolos. No exemplo, o **1** é representado por um curto segmento de reta e o **M**, por uma forca de três "pés de galinha". Também podemos notar que o nome do relacionamento é escrito sobre a reta do relacionamento.

No exemplo constituinte das Figuras 9 e 10, as entidades e relacionamentos são demonstrados em um formato horizontal, embora possam ser representados verticalmente. A localização, como a ordem de representação das entidades, é irrelevante.

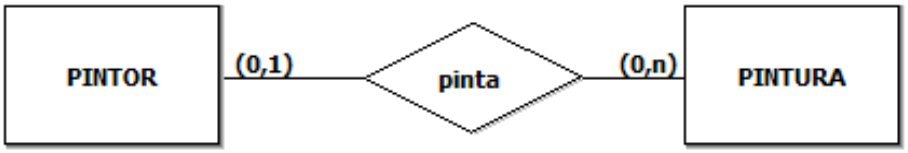


Figura 9 *Apresentação da notação de Peter Chen.*

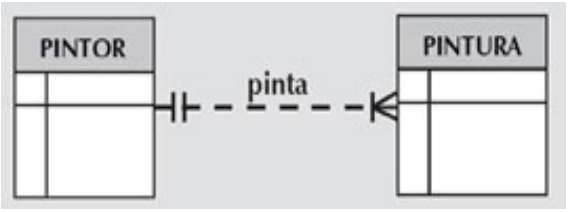


Figura 10 *Apresentação da notação* Pés de Galinha (Crow´s Foot).

Modelo Orientado a Objetos

No modelo de dados orientado a objetos, os dados e seus respectivos relacionamentos são contidos em uma única estrutura, conhecida como objeto. Como não poderia deixar de ser, o modelo de dados orientado a objetos constitui a base para o SGBDOO.

Diferente do que ocorre com uma entidade, o objeto também inclui, internamente, informações pertinentes aos relacionamentos entre os fatos, assim como informações sobre os relacionamentos com outros objetos. Esse tipo de modelo permite que um determinado objeto possua todas as ações que podem ser executadas sobre ele, como a alteração, a busca ou a impressão de seus dados.

Os seguintes componentes constituem o modelo de dados orientado a objetos:

1) **Objeto**: abstração de uma entidade real, podendo ser considerado similar à entidade no modelo de ER.
2) **Atributos**: descrevem propriedades de um objeto. Exemplo: o objeto PESSOA inclui os atributos Nome, RG, CPF e Data de Nascimento.
3) **Classe**: representa um conjunto de objetos comuns, os quais compartilham os atributos (estrutura) e seus comportamentos (métodos).
4) **Método**: representa uma ação real. Exemplo: a ação de encontrar uma PESSOA selecionada, alterar o nome da PESSOA ou até mesmo realizar a impressão de seu endereço. Portanto, os métodos correspondem aos procedimentos da linguagem de programação tradicional.
5) **Hierarquia de classes:** assemelha-se a uma estrutura de árvore de cima para baixo, em que cada classe possui apenas um pai. Por exemplo, as classes CLIENTE e EMPREGADO herdam características compartilhadas de sua classe pai PESSOA (semelhante ao modelo hierárquico).
6) **Herança**: destina-se à capacidade de um objeto herdar os atributos e seus respectivos métodos das classes superiores (pai). Por exemplo, podemos citar as duas classes CLIENTE e FUNCIONÁRIO, as quais podem ser criadas como subclasses da mesma classe PESSOA (pai ou superclasse), herdando todos os atributos e métodos de PESSOA.

A UML (*Unified Modeling Language*, ou seja, Linguagem de Modelagem Unificada) é uma linguagem baseada em conceitos de orientação a objetos que descreve um conjunto de diagra-

mas e símbolos utilizados para modelar graficamente um sistema computacional. Ele é usado, também, para representar o diagrama de classe dos modelos de dados orientados a objetos.

A Figura 11 apresenta os objetos necessários para um cenário simples de faturamento, bem como o diagrama de classes equivalente em UML e seu respectivo modelo de ER. Os objetos da FATURA incluem todos os objetos a ela relacionados. Observamos que seus relacionamentos (**1** e **M**) representam a conectividade dos objetos relacionados com a FATURA, em que o número **1,** próximo ao objeto CLIENTE, indica que cada FATURA se relaciona única e exclusivamente com apenas um CLIENTE. Enquanto a letra **M,** localizada próxima ao objeto LINHA, indica que cada FATURA contém muitas LINHAS.

O diagrama de classes em UML utiliza três classes distintas de objetos (CLIENTE, FATURA e LINHA) e dois relacionamentos. É possível notar que as conexões dos relacionamentos são representadas pelos símbolos **1..1**, **0..*** e **1..***, em que os relacionamentos são expostos na duas extremidades permitindo a reprodução de diversos "papéis" que os objetos possam executar. Já o modelo ER faz uso, também, de três entidades segmentadas e dois relacionamentos para constituir o mesmo cenário.

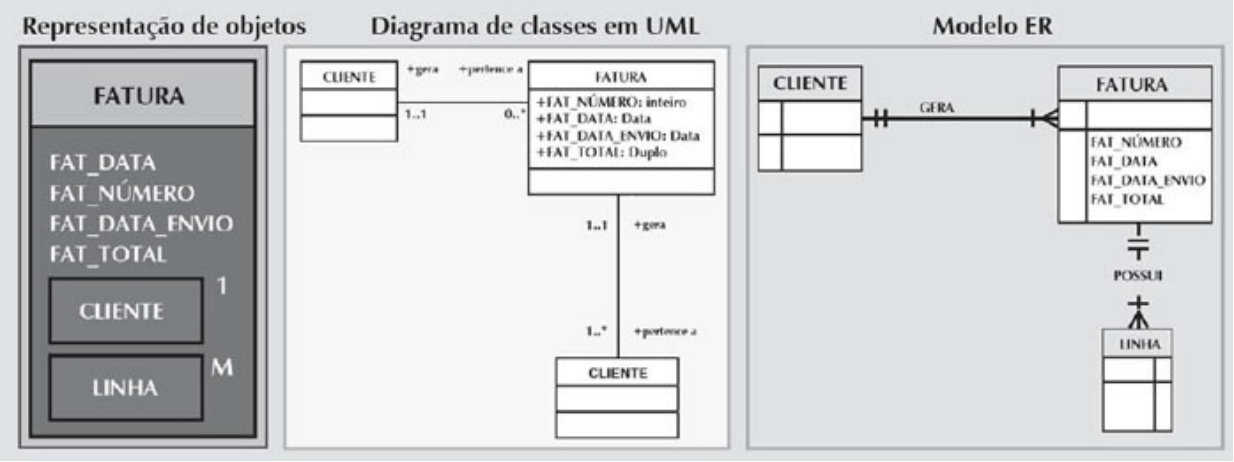


Figura 11 *Diferenças entre os modelos de OO, UML e ER*.

Modelo de Dados Relacionais Estendido

Devido à complexidade incremental das aplicações computacionais, o modelo de dados relacionais estendido foi criado por meio da utilização dos melhores recursos do modelo orientado a objeto (OO) em um ambiente estrutural de banco de dados relacional. Um SGBD baseado em um modelo de dados relacional estendido é, geralmente, representado como um Sistema de Gerenciamento de Bancos de Dados Relacionais/Objeto (SGBDR-O).

Esse modelo destina-se, exclusivamente, a aplicações comerciais, enquanto o modelo de dados orientado a objeto concentra-se em aplicações científicas e/ou aplicações computacionais específicas (a manipulação e o gerenciamento de imagens médicas, por exemplo).

Para que todo o esquema, as regras armazenadas e controladas pelos SGBDs funcionem corretamente, é necessário conhecermos os componentes de processamento de consulta e administração de memória. Os componentes de processamento de consultas são *softwares* que promovem interface, de mais alto nível, entre os usuários – sejam eles DBAs (*Database Administrators*), usuários finais ou programas de aplicações (também conhecidos como *front-end*). Os componentes incluem:

1) **Compilador DML** (*Data Manipulation Language*): realiza a tradução dos comandos DML em instruções de **baixo nível** para o componente de execução de comandos.

Possui também a responsabilidade de otimizar, transformando a requisição do usuário em uma equivalente, porém mais eficiente, de acordo com o projeto implementado pelo banco de dados. É importante lembrar que os **dados de baixo nível** são dados no formato binário ou no formato de linguagem de máquina (a linguagem *Assembly*, por exemplo).

2) **Pré-compilador para comandos DML**: o pré-compilador atua paralelamente com o compilador DML, traduzindo comandos DML manipulados em programas de aplicação, gerando a codificação adequada.
3) **Interpretador DDL** (*Data Definition Language*): realiza a interpretação dos comandos DDL, armazenando esses registros (**metadados**) em tabelas internas apropriadas. **Metadado** é uma abstração do dado capaz, por exemplo, de indicar se determinada base de dados existe, quais os atributos de uma tabela e suas características (tamanho e/ou formato).
4) **Pré-compilador DML** (*Data Manipulation Language* ou Linguagem de Manipulação de Dados): compila comandos DML em rotinas da linguagem do ***host***. Precisa interagir com o processador de consultas para gerar o código apropriado. ***Host*** ou Hospedeiro é qualquer computador capaz de interpretar e executar rotinas de linguagens que convertam o código-fonte em um código- objeto apropriado, isto é, o compilador da linguagem interage com o processador, o qual irá dizer sob quais regras o código deverá ser criado e alocado fisicamente, para que este possa ser executado corretamente pelo *host*.
5) **Componentes para tratamento de consultas**: tem como objetivo principal executar instruções de baixo nível geradas pelo compilador DML.

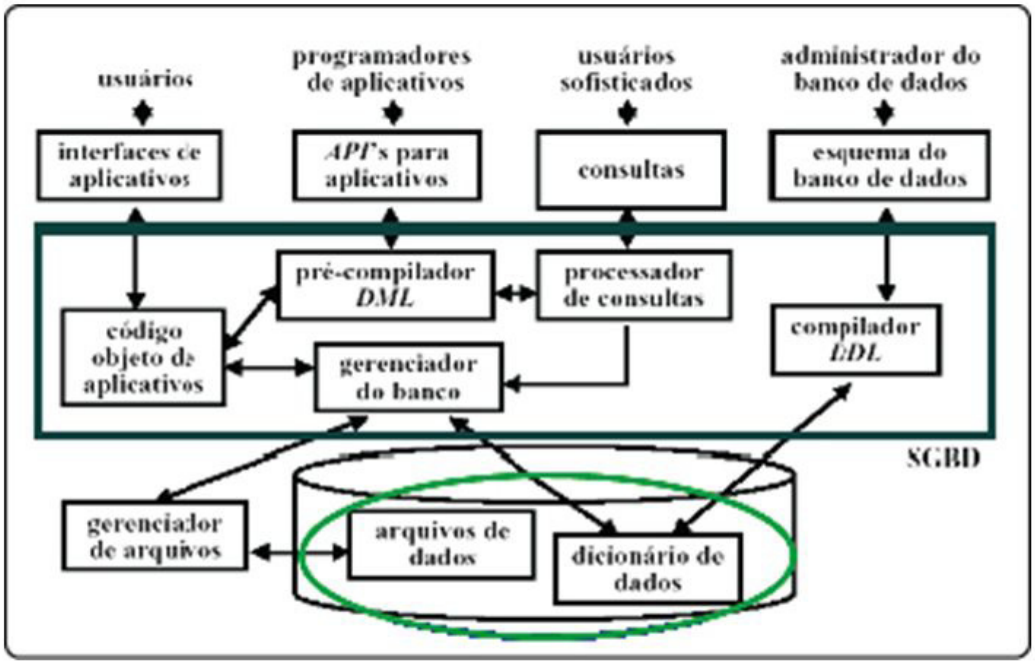


Figura 12 *Sistema Gerenciador de Banco de Dados.*

Os componentes de administração de memória são responsáveis pelo gerenciamento de memória e arquivos do Sistema Gerenciador de Banco de Dados e estabelecem a comunicação em baixo nível entre os componentes de processamento de consulta e o repositório de dados. Esses componentes são constituídos pelo:

1) **Gerenciador de Autorizações e Integridade:** aplicativos que realizam os testes adequados para garantir o cumprimento das regras de integridade e as permissões dos usuários para acessar os dados.

2) **Gerenciador de Transações:** garante que os arquivos de dados sejam mantidos de maneira consistente, independentemente de falhas no sistema ou transações concorrentes (executadas paralelamente), evitando conflitos entre os procedimentos.
3) **Gerenciador de Arquivos:** gerencia a alocação de espaço de armazenamento em disco, como suas estruturas utilizadas para representar as informações.
4) **Gerenciador de *Buffer*:** gerencia os dados oriundos do disco (meio de armazenamento), colocando-os na memória principal ou na memória cache.

Além das estruturas apresentadas anteriormente, existem outras estruturas em discos, implementadas pelo Sistema Gerenciador de Bancos de Dados, consideradas relevantes. São elas:

1) **Arquivos de Dados:** seria o banco de dados propriamente dito.
2) **Dicionário de Dados:** realiza o armazenamento das informações relativas à estrutura do banco de dados. O dicionário de dados é frequentemente requisitado por várias aplicações que constituem o SGBD. É importante destacar que o dicionário de dados é um espaço reservado dentro de um banco de dados, utilizado para armazenar informações sobre o próprio banco de dados. Um dicionário de dados pode conter informações como: informações do banco de dados, procedimentos SQL armazenados, permissões de usuários, estatísticas do usuário, desempenho e crescimento.
3) **Arquivos de Índices**: aperfeiçoam o acesso aos dados, auxiliando os algoritmos de consulta, por meio da indexação de determinados dados contidos no arquivo de dados.
4) **Estatística de Dados:** arquivo responsável pelo armazenamento de variáveis referentes às estatísticas relativas aos dados. Essas variáveis são utilizadas pelo processador de consultas para selecionar os meios mais eficientes na realização de uma consulta específica.

A centralização dos recursos em um SGBD (Figura 12) aumenta a vulnerabilidade do sistema, pois uma falha poderá ocasionar a interrupção das atividades do sistema.

8. ARQUITETURA EM UM SGBD

Vejamos o que Takai; Italiano e Ferreira (2005) dizem sobre a arquitetura em um Sistema Gerenciador de Banco de Dados.

> As primeiras arquiteturas utilizavam ***mainframes*** para executar o processamento principal e de todas as funções do sistema, incluindo os programas aplicativos, os programas de interface com o usuário, bem como a funcionalidade dos SGBDs. Esta é a razão pela qual a maioria dos usuários fazia acesso aos sistemas via terminais que não possuíam poder de processamento, apenas a capacidade de visualização. Todos os processamentos eram feitos remotamente, apenas as informações a serem visualizadas e os controles eram enviados do *mainframe* para os terminais de visualização, conectados a ele por redes de comunicação.
>
> Como os preços do *hardware* foram decrescendo, muitos usuários trocaram seus terminais por computadores pessoais (PC) e estações de trabalho. No começo, os SGBDs usavam esses computadores da mesma maneira que usavam os terminais, ou seja, o SGBD era centralizado e toda sua funcionalidade, execução de programas aplicativos e processamento da interface do usuário eram executados em apenas uma máquina.

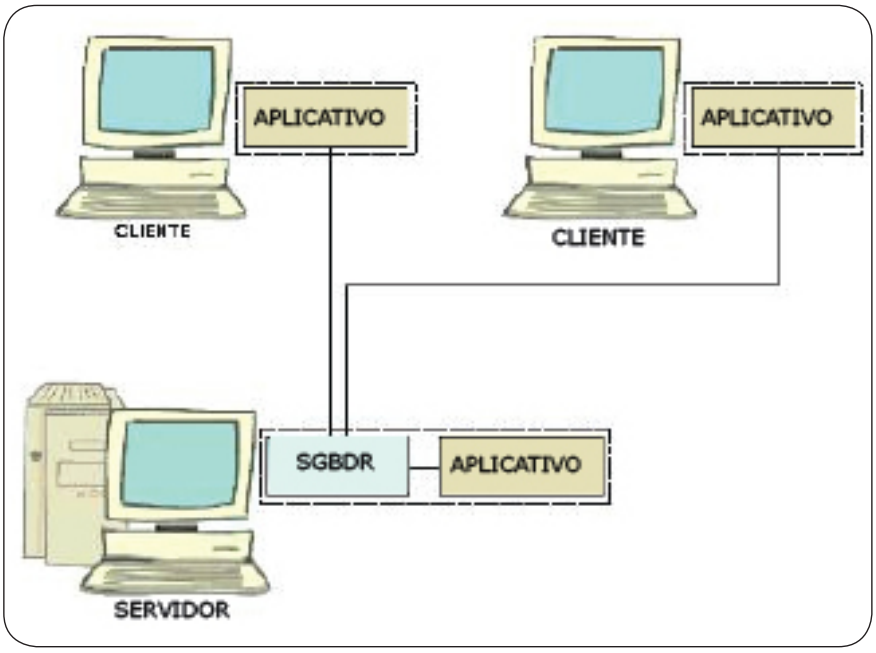


Figura13 *Arquitetura Cliente-Servidor.*

Gradualmente, os SGBDs começaram a explorar a disponibilidade do poder de processamento no lado do usuário, o que levou à arquitetura cliente-servidor.

A arquitetura cliente-servidor foi desenvolvida para dividir ambientes de computação onde um grande número de PCs, estações de trabalho, servidores de arquivos, impressoras, servidores de banco de dados e outros equipamentos estão conectados juntos por uma rede.

[...]Desta maneira, a arquitetura cliente-servidor foi incorporada aos SGBDs comerciais. Diferentes técnicas foram propostas para se implementar essa arquitetura, sendo que a mais adotada pelos Sistemas Gerenciadores de Banco de Dados Relacionais (SGBDR) comerciais é a inclusão da funcionalidade de um SGBD centralizado no lado do servidor. As consultas e a funcionalidade transacional permanecem no servidor, sendo que este é chamado de servidor de consulta ou servidor de transação. É assim que um servidor SQL é fornecido aos clientes. Cada cliente tem que formular suas consultas SQL, prover a interface do usuário e as funções de interface usando uma linguagem de programação.

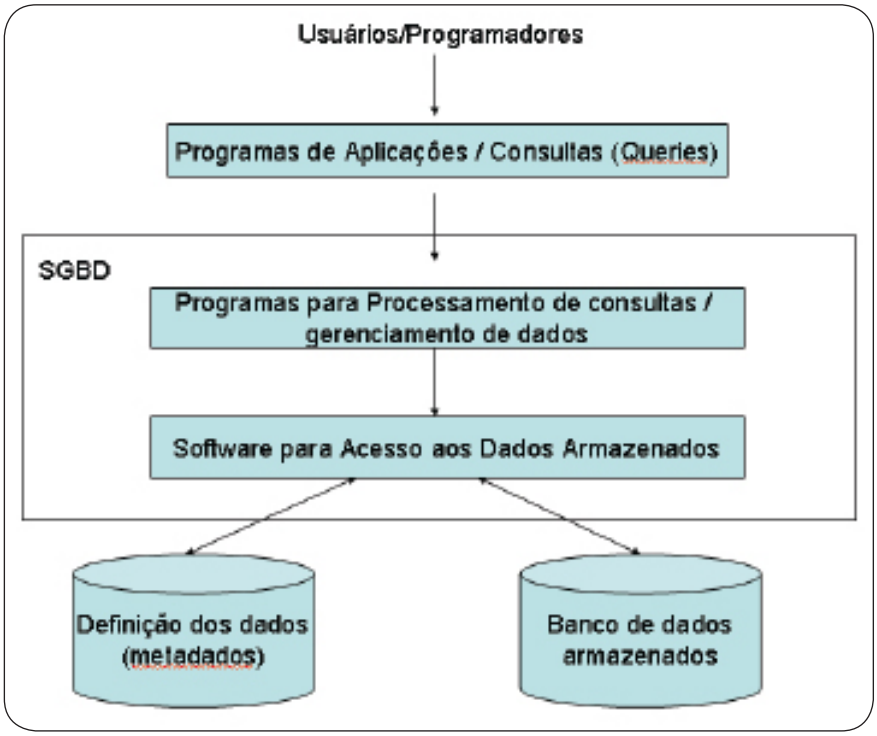


Figura 14 *Sistemas Gerenciadores de Banco de Dados Relacionais.*

Comumente o servidor SQL também é chamado de *back-end machine* e o cliente de *front-end machine*. Como o SQL provê uma linguagem padrão para os SGBDRs, esta criou o ponto de divisão lógica entre o cliente e o servidor.

Atualmente, existem várias tendências para arquitetura de Banco de Dados, nas mais diversas direções.

Por isso, as arquiteturas de SGBDs podem ter as seguintes configurações:

Plataformas centralizadas. Na arquitetura centralizada, existe um computador com grande capacidade de processamento, o qual é o hospedeiro do SGBD e emuladorres para os vários aplicativos. Esta arquitetura tem como principal vantagem a de permitir que muitos usuários manipulem grande volume de dados. Sua principal desvantagem está no seu alto custo, pois exige ambiente especial para *mainframes* e soluções centralizadas.

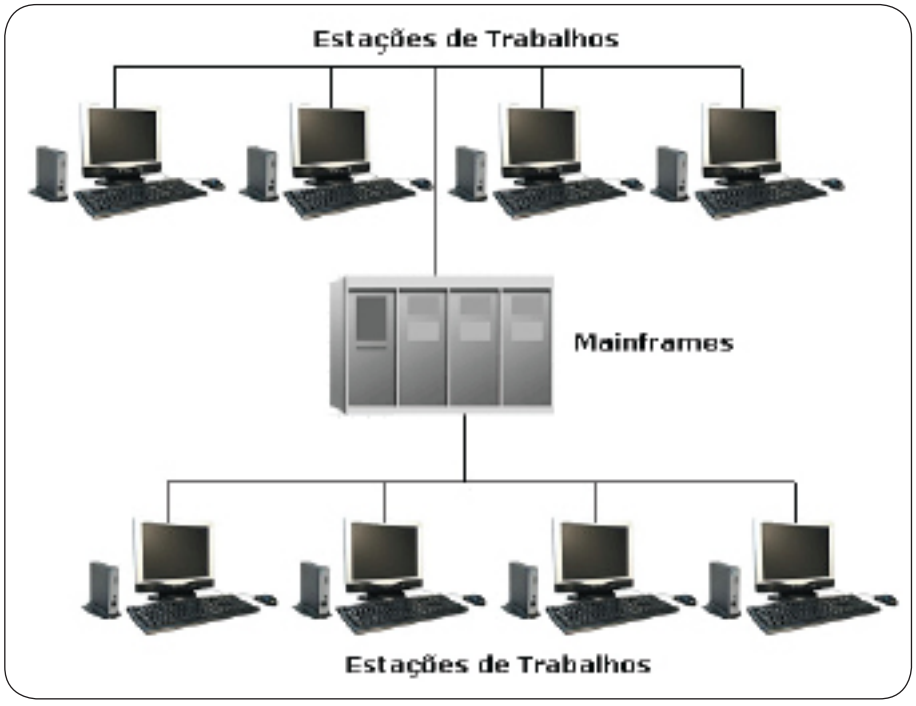


Figura 15 *Plataformas Centralizadas.*

Sistemas de Computador Pessoal – PC. Os computadores pessoais trabalham em sistema *stand-alone*, ou seja, fazem seus processamentos sozinhos. No começo esse processamento era bastante limitado, porém, com a evolução do *hardware*, tem-se hoje PCs com grande capacidade de processamento. Eles utilizam o padrão *Xbase* e, quando se trata de SGBDs, funcionam como hospedeiros e terminais. Desta maneira, possuem um único aplicativo a ser executado na máquina. A principal vantagem desta arquitetura é a simplicidade.

Figura 16 *Sistemas de Computador Pessoal.*

Banco de Dados Cliente-Servidor. Na arquitetura Cliente-Servidor, o cliente (*front_end*) executa as tarefas do aplicativo, ou seja, fornece a interface do usuário (tela, e processamento de entrada e saída). O servidor (*back_end*) executa as consultas no DBMS e retorna os resultados ao cliente. Apesar de ser uma arquitetura bastante popular, são necessárias soluções sofisticadas de *software* que possibilitem: o tratamento de transações, as confirmações de transações (*commits*), desfazer transações (*rollbacks*), linguagens de consulta (*stored procedures*) e gatilhos (*triggers*). A principal vantagem desta arquitetura é a divisão do processamento entre dois sistemas, o que reduz o tráfego de dados na rede.

Banco de Dados Distribuídos (N camadas). Nesta arquitetura, a informação está distribuída em diversos servidores. [...] Cada servidor atua como no sistema cliente-servidor, porém as consultas oriundas dos aplicativos são feitas para qualquer servidor indistintamente. Caso a informação solicitada seja

mantida por outro servidor ou servidores, o sistema encarrega-se de obter a informação necessária, de maneira transparente para o aplicativo, que passa a atuar consultando a rede, independentemente de conhecer seus servidores.

Exemplos típicos dessa configuração são as bases de dados corporativas, em que o volume de informação é muito grande e, por isso, deve ser distribuído em diversos servidores.

[...]A característica básica é a existência de diversos programas aplicativos consultando a rede para acessar os dados necessários, porém, sem o conhecimento explícito de quais servidores dispõem desses dados.

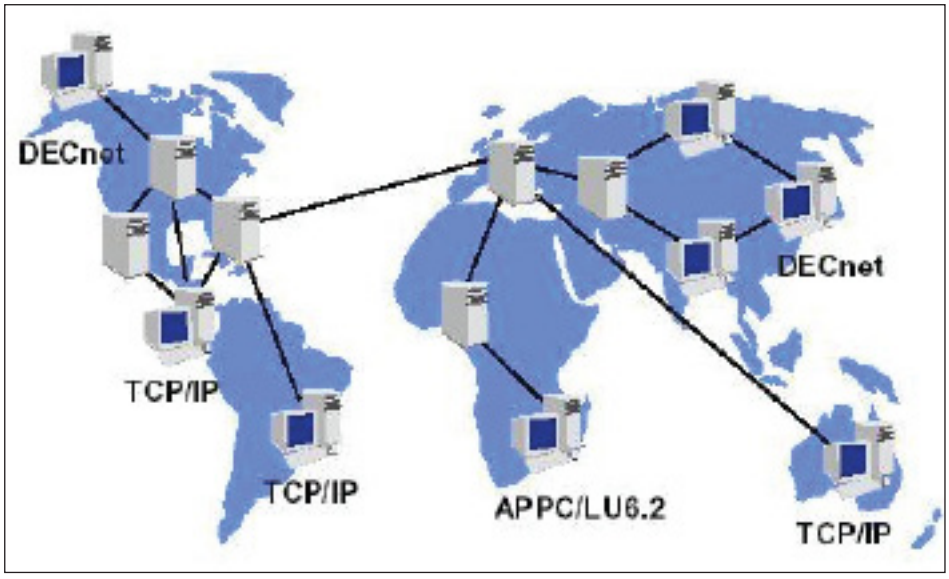


Figura 17 *Banco de Dados Distribuídos*.

9. CONSULTAS EM UM SGBD

Os Sistemas de Gerenciamento de Bases de Dados foram criados com o objetivo de manusear corretamente os dados que gerenciam, e as funções mais executadas em um SGBD são: preservar os dados que são guardados em memória e fornecer recursos que possibilitem a sua recuperação.

A responsabilidade dos SGBDs no que diz respeito às informações é grande, pois nenhum dado pode sofrer alterações em seu armazenamento ou em sua recuperação. É comum um dado ter um caráter modificado devido ao mau funcionamento da memória ou do *hardware,* por onde trafegam as informações.

A estrutura de consulta é o fator de maior influência no desempenho de um sistema e representa grande parte dos problemas, uma vez que todas as funcionalidades que recuperam dados o fazem por meio de consultas.

Como parte dos desenvolvedores não tem experiência para determinar qual a melhor forma de se estruturar uma consulta, visto que isso, em geral, depende da plataforma em uso, esta tarefa quase sempre fica a cargo dos DBAs (*Database Administrator* ou Administrador de Banco de Dados).

Existem duas formas de realizar a consulta em uma base de dados:

- utilizando uma linguagem específica de trabalho com base de dados como, por exemplo a SQL (*Structured Query Language*);
- realizando a consulta por meio de exemplo – QBE (*Query By Example*).

Mas o que seria realizar uma consulta, quando falamos em banco de dados?

Realizar uma **consulta é fazer uma pergunta** ao banco de dados, especificando alguns critérios, como: "Quais os nomes dos professores cuja disciplina é Matemática?".

Observe como seria a pergunta em SQL:

```
SELECT NomeProfessor FROM Professores WHERE Disciplina= matemática.
```

Observe o raciocínio, de acordo com o *site* Murall (2012), disponível em <http://murall.com.br/banco-de-dados/>. Acesso em: 23 out. 2012.

> O que acontece se a pergunta for feita de forma errada ou tão confusa, capaz de fazer o sistema a se perder na resposta?
>
> Se a pergunta for incorreta ou confusa o sistema se perde ou, muitas vezes demora em responder ao que se deseja, causando queda do desempenho de busca de informações do banco de dados, gerando assim, demora no fornecimento da informação ou erro por não encontrar as respostas. Vale lembrar que existe um recurso que auxilia na melhoria do desempenho de um dado no processo de busca. Esse recurso é conhecido por ÍNDEX ou ÍNDICE, que trabalha como indicador da posição que apresenta a informação que está sendo procurada. O tempo de acesso às linhas de uma tabela é acelerado com esse recurso, pois apresenta ponteiros que indicam o local exato da tabela.

10. QUESTÕES AUTOAVALIATIVAS

Sugerimos que você procure responder, discutir e comentar as questões a seguir que tratam da temática desenvolvida nesta unidade.

A autoavaliação pode ser uma ferramenta importante para você testar o seu desempenho. Se você encontrar dificuldades em responder a essas questões, procure revisar os conteúdos estudados para sanar as suas dúvidas. Esse é o momento ideal para que você faça uma revisão desta unidade. Lembre-se de que, na Educação a Distância, a construção do conhecimento ocorre de forma cooperativa e colaborativa; compartilhe, portanto, as suas descobertas com os seus colegas.

Confira, a seguir, as questões propostas para verificar o seu desempenho no estudo desta unidade:

1) O que é um banco de dados?

2) Sabemos que banco de dados e base de dados não são sinônimos. Explique sua diferença.

3) Defina e exemplifique dados e informação.

4) Como os bancos de dados são classificados? Cite alguns tipos de bancos de dados.

5) Defina um banco de dados temporal.

6) O que é um Sistema Gerenciador de Banco de Dados (SGBD)?

7) Defina os seguintes conceitos:
   a) Modelo Conceitual.
   b) Modelo Lógico.
   c) Modelo Físico.

8) Um desenvolvedor recebe um documento que detalha de maneira precisa e estruturada um banco de dados. O desenvolvedor deverá criar um *software* para acessar o banco de dados por meio de um SGBD. Esse documento é um modelo conceitual, um modelo lógico ou um modelo físico?

9) Defina o modelo entidade-relacionamento (MER), detalhando seus principais componentes.

11. CONSIDERAÇÕES

Nesta unidade, você teve a oportunidade de compreender a importância e a necessidade indiscutível de um Sistema de Gerenciamento de Banco de Dados no mundo atual, o qual depende da tecnologia para realizar o controle das informações.

Você tem ideia da quantidade de informações que é trocada, armazenada e buscada, diariamente, em todos os setores da sociedade? Pode-se dizer que é algo imensurável. É fundamental, portanto, que existam regras, arquiteturas, estruturas com níveis bem definidos e modelos de dados para organizar e controlar essa grande quantidade de informação.

Como futuro projetista de banco de dados, é necessário que você conheça e domine os modelos conceituais de banco de dados. Na próxima unidade, você estudará o modelo entidade-relacionamento, um modelo conceitual amplamente difundido e utilizado pelos projetistas de bancos de dados.

12. E-REFERÊNCIAS

Lista de figuras

Figura 1 *Banco de Dados*. Disponível em: <http://segundoepmedici.blogspot.com.br/2011/08/banco-de-dados-br-modelo.html>. Acesso em: 01 out. 2012.

Figura 17 *Banco de Dados Distribuídos*. Disponível em: <http://www.devmedia.com.br/articles/viewcomp.asp?comp=5530>. Acesso em: 01 out. 2012.

Sites pesquisados

MURALL. *Banco de Dados*. Disponível em: <http://murall.com.br/banco-de-dados/>. Acesso em: 23 out. 2012.

TAKAI, O. K.; ITALIANO, I. C.; FERREIRA, J. E. *Introdução a Banco de Dados*. 2005. Disponível em: <http://pt.scribd.com/doc/51228653/9/Arquiteturas>. Acesso em: 22 out. 2012.

13. REFERÊNCIAS BIBLIOGRÁFICAS

ELMASRI, R.; NAVATHE, S. B. *Sistemas de bancos de dados.* São Paulo: Pearson (Addison Wesley), 2005.

KORTH, H.; SILBERCHATZ, A. *Sistemas de bancos de dados.* 3. ed. São Paulo: Makron Books, 1998.

PRESSMAN, R. S. *Engenharia de software*. São Paulo: Makron Books, 1995.

RAMAKRISHNAN, R.; GEHRKE, J. *Database management systems.* 2. ed. Boston: McGraw-Hill, 2000.

ROB, P.; CORONEL, C. *Sistemas de Banco de Dados*: Projeto, Implementação e Administração. 8. ed. São Paulo: Cengage Learning, 2011.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. *Sistema de Banco de Dados*. 5. ed. Rio de Janeiro: Elsevier, 2006.