INSTITUTO DE GESTÃO E TECNOLOGIA
DA INFORMAÇÃO

Fundamentos em Engenharia de Dados

Fernanda Farinelli

2022

Fundamento em Engenharia de Dados

Fernanda Farinelli

Sumário

Capítulo 1. Conceitos fundamentais de Big Data:

A evolução das tecnologias de informação e comunicação, assim como o surgimento da internet, mudou o dia a dia das pessoas trazendo as atividades humanas para o mundo virtual. A *World Wide Web*, a maior rede de informação global, em ritmo evolucionário, passou por fases que ficaram conhecidas como Web 1.0, Web 2.0 e Web 3.0 (SHIVALINGAIAH, NAIK, 2008). Vivemos cercados por uma grande quantidade de dados que apoiam as nossas decisões, tal disponibilidade oferece oportunidades para a obtenção de conhecimento, ou seja, ao submeter os dados a processos de análise, obtém-se informação e conhecimento útil nos processos decisórios das organizações (HEATH, BIZER, 2011).

Dados, tipos de dados e fontes de dados

Nas últimas décadas, os dados assumiram um papel vital na estratégia das empresas, tornando-se um dos grandes ativos existentes no patrimônio das organizações (DAMA, 2009, p. 1). Mas o que são dados? Veja no Quadro 1 o que são dados, informação e conhecimento.

Quadro 1 – Dados, informação e conhecimento.

Dado	Informação	Conhecimento
Simples observações sobre o estado do mundo.	Dados dotados de relevância e propósito.	Informação valiosa da mente humana.
Facilmente estruturado.	Requer unidade de análise.	Inclui reflexão, síntese e contexto.

Facilmente obtido por máquinas. Frequentemente quantificado. Facilmente transferível.	Exige consenso em relação ao significado. Exige necessariamente a mediação humana.	Difícil estruturação. Difícil captura em máquinas. Frequentemente tácito. Difícil transferência.

Fonte: Retirado de DAVENPORT (1998, p. 18).

Tal definição enfatiza o papel dos dados em representar fatos sobre o mundo, ou seja, *dados são fatos capturados, armazenados e expressos como texto, números, gráficos, imagens, sons ou vídeos* (DAMA, 2009, 2017). Nossos dados não se resumem a ações, podem ser objetos, localizações, quantidades, textos, imagens e áudios, ou qualquer coisa que possa ser digitalizada e armazenado em algum banco de dados (DAMA, 2009, 2017).

As atividades e ações virtuais e os diversos sistemas de informação produzem, coletam e analisam dados. Os dados podem vir de diferentes fontes de dados, ou seja, uma fonte é o local de onde o dado é coletado ou adquirido. Podem ser arquivos, banco de dados, portal de notícias ou até mesmo um *feed* de notícias. As fontes de dados podem ser qualquer dispositivo ou estrutura que forneça dados, localizada ou não no mesmo computador ou dispositivo que o programa de coleta.

Os dados podem assumir diferentes formatos ou tipos conforme sua origem, a Figura 1 apresenta um quadro sumarizando comparativamente estes tipos de dados.

Figura 1 – Comparativo dos tipos de dados.

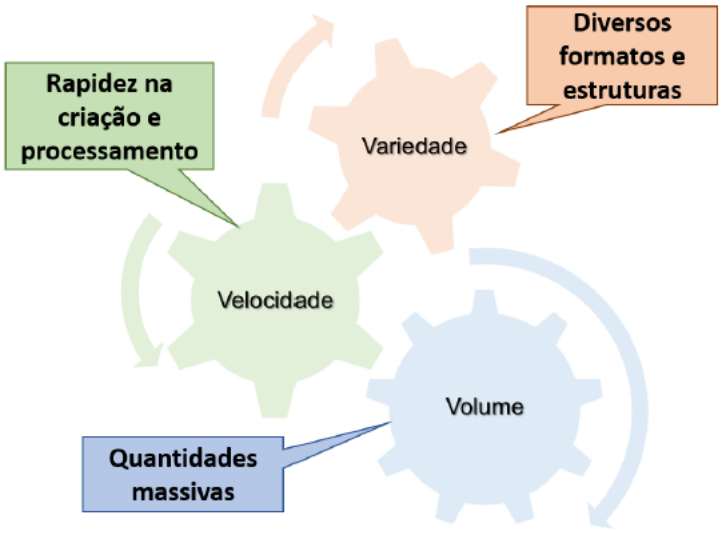
Os dados armazenados nos bancos de dados relacionais são considerados dados estruturados, como dados semiestruturados podemos citar os dados armazenados em arquivos XML e como dados não estruturados temos os dados originários de mídias sociais, imagens, vídeos e áudios. A maior parte dos dados que são coletados atualmente são dados não estruturados. Acredita-se que 95% dos dados gerados hoje são em formato não-estruturado (MAYER-SCHÖNBERGER, CUKIER, 2013:47).

Conceito e características de big data

Este volume de dados e suas diversas fontes e formatos levou ao fenômeno que ficou conhecido como *Big Data*. Termo cunhado em meados dos anos 90 por Michael Cox e David Ellsworth, cientistas da Nasa, que discutiram sobre os desafios da visualização de grandes volumes de dados, aos limites computacionais tradicionais de captura, processamento, análise e armazenamento (COX, ELLSWORTH, 1997). O termo *Big Data* é usado para descrever este grande conjunto de dados que desafia os métodos e ferramentas tradicionais para manipulação de dados considerando um tempo razoável de resposta. *Big Data* é caracterizado pela tríade volume, variedade e velocidade

(LANEY, 2001) ilustrado na Figura 2. Ainda temos duas importantes características, veracidade e valor (TAURION, 2013).

Figura 2 – Os 3Vs do Big Data.



O ***volume*** diz respeito à quantidade de dados que são produzidos e coletados pelas organizações. As organizações coletam dados de diversas fontes, implicando na ***variedade*** dos tipos (estruturados, semiestruturados e não estruturados) e formatos dos dados coletados. A ***velocidade*** diz respeito tanto ao quão rápido os dados estão sendo produzidos e quão rápido os dados devem ser tratados para atender a demanda da organização. As decisões são tomadas em tempo real. Temos ainda a ***veracidade*** ou confiabilidade dos dados, ou seja, eles devem expressar a realidade e ser consistentes. Enfim, o ***valor***, ou a utilidade dos dados ao negócio, como agregam valor (DAVENPORT, 2014; TAURION, 2013).

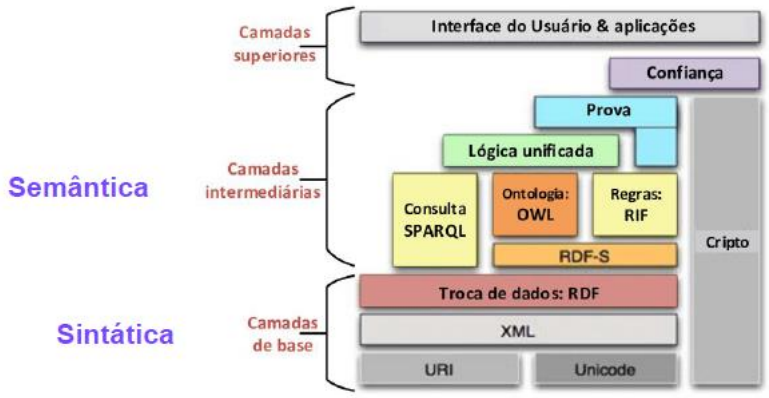
Web semântica

A Web Semântica é uma extensão da web que estrutura o significado de seu conteúdo de forma clara e bem definida, permitindo aos computadores interagir entre eles trocando informações. Sua principal motivação é ter uma web de dados, no qual tais dados sejam significativos tanto para os humanos quanto para as máquinas (BERNERS-LEE, 1998; BERNERS-LEE; HENDLER; LASSILA, 2001).

Na Web Semântica são os vocabulários que definem os conceitos e relacionamentos usados para descrever e representar uma área de interesse. Muitas vezes uma ontologia pode ser empregada quando se tem uma coleção de termos mais complexa e formal. O papel dos vocabulários, portanto, é o de auxiliar na integração dos dados, tratando os casos de ambiguidade de termos usados em diferentes bases de dados por exemplo.

A Figura 3 a seguir demonstra um exemplo de arquitetura de referência da web semântica:

Figura 3 - Arquitetura de Referência da Web semântica.



Fonte: (BERNERS-LEE; HENDLER; LASSILA, 2001).

Dados abertos e dados interligados

Dados abertos

O conceito de dados abertos remete a ideia de conteúdo aberto, ou seja, disponível para todos. Dados abertos são dados que podem ser livremente publicados na web, seguindo alguns padrões predefinidos, e a partir de sua publicação podem ser reutilizados e redistribuídos por qualquer pessoa ou aplicativo, sujeitos, no máximo, à exigência de atribuição da fonte e compartilhamento pelas mesmas regras (ISOTANI, BITTENCOURT, 2015; OKI, 2019).

Dados interligados Linked open data

Fundamenta-se na ideia de interligar dados na web ao invés de documentos. *Linked data* (LD), ou dados interligados, é uma forma de publicar dados na web de forma estruturada, de modo que uma pessoa ou máquina possa explorar estes dados. Relacionado à web semântica, propõe um conjunto de princípios, padrões e protocolos para serem adotados para publicar dados na web e para interligar os dados. As ligações permitem aos usuários da web navegar entre diferentes fontes. Além disso, as ferramentas de busca ficam aptas a indexar a web e fornece recursos de pesquisa mais sofisticados sobre o conteúdo rastreado (BERNERS-LEE, 2006; BIZER, HEATH, BERNERS-LEE, 2008, 2009; HEATH, BIZER, 2011).

Dados abertos interligados

Adicionalmente, temo o conceito de dados abertos interligados ou *Linked Open Data,* que remete a ideia de conteúdo aberto ou disponível para todos, mas com interconexões entre os dados, ou seja, são dados interligados que se encontram disponíveis livremente na web (BERNERS-LEE, 2006).

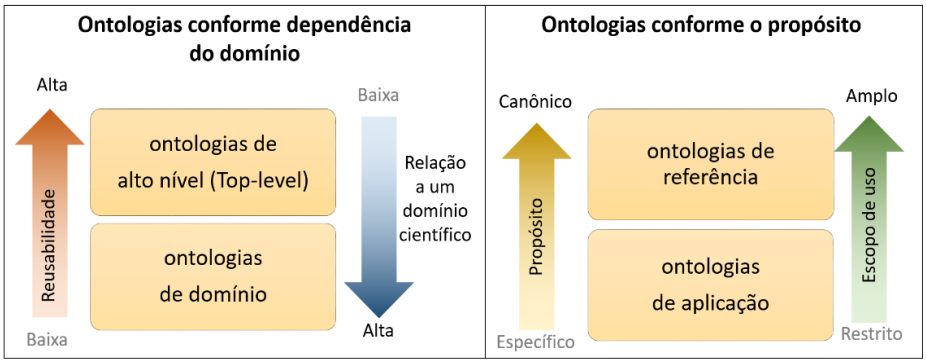
Ontologias

Ontologia é um termo polissêmico e objeto de pesquisa em diversas áreas, como: Filosofia, Ciência da Computação e Ciência da Informação. A palavra ontologia é derivada do grego, onde "*Onto*" exprime a noção do ser e "*Logia*" é algo dito ou a maneira de dizer. Ela pode ser entendida com disciplina filosófica ou como artefato representacional. Como disciplina da Filosofia, a Ontologia estuda a natureza da existência das coisas. Como artefato representacional, a ontologia representa conhecimento acerca de vários domínios de conhecimento, através da formalização das relações entre termos e conceitos (ALMEIDA, 2013).

As ontologias se classificam conforme descrito abaixo e ilustrado na Figura 4 (FARINELLI, 2017; FARINELLI; SOUZA, 2021).

- Ontologias de alto nível: descrevem conceitos amplos independentes de um domínio particular. Ex.: relacionadas a espaço, tempo, eventos etc.
- Ontologias de referência: descrevem conceitos relacionados a atividade ou tarefas genéricas, independentes de domínio. Ex.: diagnóstico.
- Ontologias de domínio: descrevem conceitos relacionados a domínios específicos, como direito, computação etc. É a categoria mais comum.
- Ontologias de aplicação: descrevem conceitos dependentes de um domínio e tarefa específicos.

Figura 4 – Classificação das ontologias.



Fonte: Traduzido de FARINELLI (2017).

Uma ontologia pode ser muito complexa, com milhares de conceitos ou muito simples, descrevendo apenas um ou dois conceitos. A especificação de uma ontologia inclui os seguintes elementos (FARINELLI, 2017):

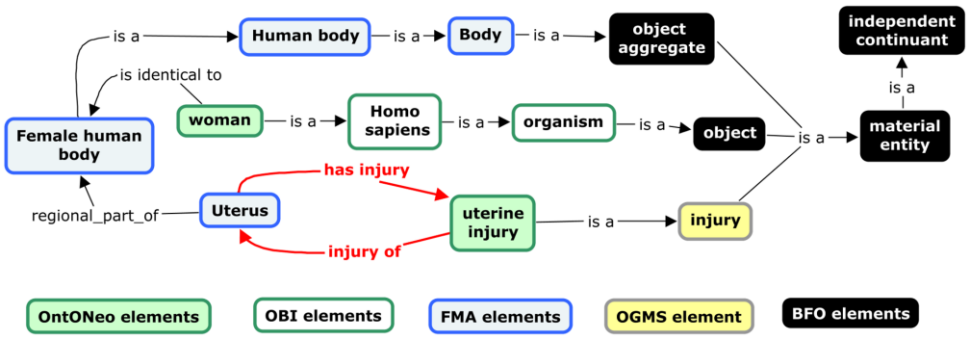
- Entidade: é algo que você deseja representar em um domínio particular. Qualquer coisa que exista, existiu ou irá existir. Ex.: eventos, processos e objetos.
- Instância ou indivíduos: representam uma unidade de objetos específicos de uma entidade, ou seja, indivíduos de um determinado universal.
- Atributos: propriedades relevantes da entidade/classe ou instância que ajudam a descrevê-la.
- Relacionamento: descreve o tipo de interação entre duas classes, duas instâncias ou uma classe e uma instância.
- Cardinalidade: uma medida do número de ocorrências de uma entidade associada a um número de ocorrências em outra.

- Axioma: uma proposição lógica que é considerado verdadeiro. Restringem a interpretação e o uso das classes envolvidas na ontologia.

As ontologias descrevem entidades sobre a perspectiva dos universais e particulares. Os particulares ou indivíduos são ocorrências únicas de algo existente na realidade, por exemplo, cada um de nós é uma única ocorrência ou indivíduo de um "homo sapiens". Os universais ou tipos são entidades reais que generalizam os particulares existentes no mundo, por exemplo, “homo sapiens” é uma entidade geral ou universal referente aos particulares que cada um de nós é.

Um exemplo de ontologia é mostrado na Figura 5. Nesta ontologia é descrita uma pequena fração do domínio médico obstétrico, especificamente para descrever a relação de lesão uterina com o corpo humano feminino. É possível se identificar as entidades ou classes representadas por elipses e as relações pelas setas.

Figura 5 – Parte da ontologia Ontoneo.



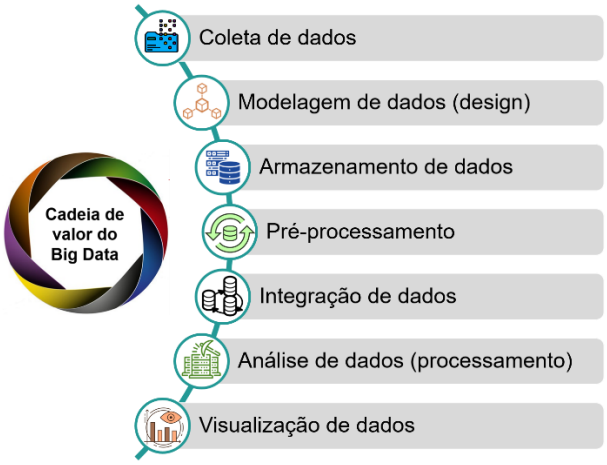
Fonte: FARINELLI (2017).

Pipeline de dados do Big Data e de Engenharia de Dados

Dados tradicionais e Big Data demandam processos de coleta, armazenamento, processamento, análise e visualização, porém a diferença se volta para as três características, volume, variedade e velocidade (TAURION, 2013). Ao longo das últimas

três décadas, surgiram várias abordagens para mineração de dados, dentre elas cita-se as metodologias KDD, CRISP-DM e SEMMA. O pipeline de dados é o conjunto de processos e atividades que uma organização executa para extrair informações dos seus dados, capazes de subsidiar seu processo de tomada de decisão. Um exemplo de pipeline fundamentado nos processos de mineração de dados KDD, SEMMA e CRISP-DM, além da cadeia de valor proposta por CURRY (2016), aqui nomeado de cadeia de valor do *Big Data*, é apresentada na Figura 6. Nesta cadeia de valor, o fluxo de informações é descrito como uma série de etapas necessárias para gerar valor e informações úteis dos dados.

Figura 6 – Cadeia de Valor de Big Data.



A grande responsabilidade de uma equipe de engenharia de dados é construir todo o pipeline de dados (data flow), principalmente nos processos de coleta, modelagem, pré-processamento ou preparação e armazenamento de dados. Cada um destes processos será tratado nos capítulos seguintes.

A Engenharia de Dados é a área responsável por planejar, criar, manter e evoluir toda a estrutura de dados de uma organização (NASCIMENTO, 2017). É neste cenário que o profissional de engenharia de dados ganha importância no mundo de Big Data. Este profissional deve:

- Entender a origem e natureza dos dados.
- Lidar o planejamento e desenvolvimento do esquema de dados.
- Definir estruturas confiáveis para suportar todo o fluxo de dados e implementar a coleta, preparação e armazenamento.
- Propor e implementar a estrutura de armazenamento e soluções de *Data-Warehouse*.
- Entender a necessidade de integração de dados.
- Propor e implementar a estrutura de integração e rotinas de ETL.

Estes profissionais planejam e criam a infraestrutura necessária para receber o fluxo de dados desde sua aquisição, transformação, armazenamento, até o processamento e visualização de dados. São os responsáveis por prover a arquitetura que deve acolher o ciclo de vida dos dados. A estrutura projetada pelo engenheiro de dados geralmente envolve as operações listadas (PARUCHURI, 2017; TAMIR; MILLER; GAGLIARDI, 2015):

- Aquisição ou Ingestão (Data Ingestion): envolve atividades e estrutura para coleta dos dados de diversos formatos e origens.

- Processamento: envolve infraestrutura capaz de suportar o processamento dos dados, levando em conta o volume e a variedade, para obter os resultados finais desejados em tempo hábil.

- Armazenamento: envolve infraestrutura capaz de armazenar os dados coletados e os resultados dos dados processados ou analisados, visando inclusive a recuperação rápida destes dados.

- Acesso: envolve uma estrutura voltada para segurança cujo usuário esteja habilitado a acessar os resultados finais das análises.

A estrutura projetada dever ser projetada para ser escalável, para suportar o volume de dados e crescimento da demanda por armazenamento, além da necessidade de suportar o armazenamento e processamento dos dados dos diversos formatos, assim como garantir a confiabilidade, integridade e segurança dos dados. Esta estrutura, o projeto de arquitetura de dados, foca na transformação dos dados em um formato útil para análise. A engenharia de dados, assim com a ciência de dados, é um campo amplo abrangendo uma infinidade de habilidades, plataformas e ferramentas. No entanto, um único engenheiro de dados não precisa conhecer o espectro completo, em geral, ele trabalha em conjunto com outros engenheiros e analistas (PARUCHURI, 2017; TAMIR; MILLER; GAGLIARDI, 2015).

Visão geral dos principais tipos de análise de dados

Análise de dados é um conjunto de técnicas empregadas para a transformação de dados e informações em conhecimento para um propósito específico e que visa encontrar uma forma eficiente de conhecimento (padrões) em conjuntos de dados, seja

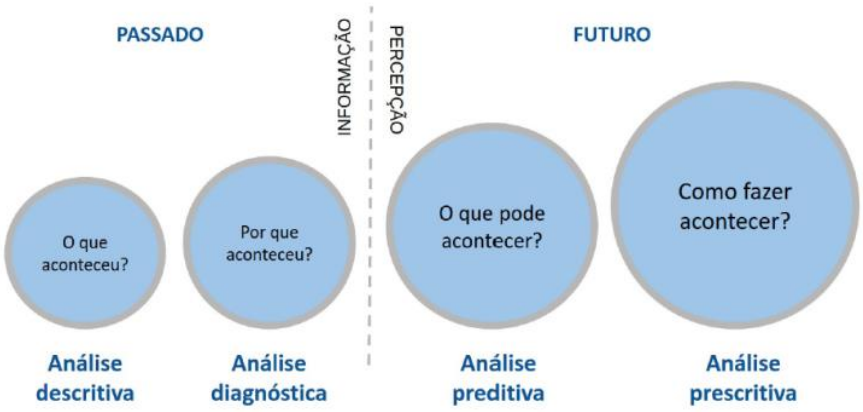
para compreender o comportamento de pessoas ou para identificar novas oportunidades de negócio (DAVENPORT, 2014; DAVENPORT; HARRIS, 2020).

Inicialmente, com o surgimento dos DWs, eram empregadas técnicas de análise afim de compreender o que aconteceu e o motivo pelo qual eventos aconteceram. Mais tarde, isso já não era suficiente, pois surgiu a necessidade de tentar prever o que poderia acontecer com os negócios antes de acontecer, e assim, antecipar algumas ações. Como o volume de dados ultrapassava a capacidade humana de interpretar e compreender tanta informação, foi necessário criar mecanismos automáticos para processar tantos dados (AMARAL, 2016).

Principais tipos de análise de dados

De acordo com a classificação do Gartner, existe uma cadeia de evolução em análise de dados, variando de descritiva a diagnóstica a preditiva e culminando com prescritiva. Esses tipos de análise de dados ajudam as organizações a compreender dois momentos sobre seus negócios: passado e futuro.

Figura 7: Tipos de análise de dados.



Fonte: Ferri (2019).

Análise descritiva

Análise descritiva de dados, às vezes descrita como análise exploratória, tem como objetivo é entender o cenário atual da organização a partir da análise de seus dados históricos. Trabalha com histórico de dados, cruzando informações com o objetivo de gerar um panorama claro e preciso dos temas relevantes para a empresa no presente momento a partir de seu passado. Em geral utilizam métricas e técnicas estatísticas simples ou avançadas para entender e explicar como os dados são buscando explicar o que está acontecendo ou aconteceu em uma determinada situação (TUKEY, 1977).

Análise diagnóstica

Algumas referências incluem a análise diagnostica como parte da descritiva, isto porque a análise diagnóstica visa explicar os eventos que ocorreram e foram descritos no modelo de análise anterior. Este modelo de análise tentar responder à pergunta "*Por que isso aconteceu?*". Neste modelo de análise o foco está na relação de causas e consequências percebidas ao longo do tempo, sobre de um determinado assunto ou evento, cruzando informações com o objetivo de entender quais fatores influenciaram o resultado atual.

Análise preditiva

A análise preditiva é utilizada para prever tendências baseadas nos dados. Segundo o Gartner, a análise preditiva é uma forma de análise avançada que examina dados ou conteúdo para responder à pergunta "*O que vai acontecer?*" Ou mais precisamente, "*O que é provável que aconteça?*" Este tipo de análise é o mais indicado para quem precisa prever algum tipo de comportamento ou resultado. Esta técnica busca analisar dados relevantes ao longo do tempo, buscando padrões comportamentais e suas variações de acordo com cada contexto, a fim de prever como

será o comportamento de seu público ou mercado no futuro, dadas as condições atuais. Muito útil para avaliar tendências de consumo e flutuações econômicas. É caracterizada por técnicas como análise de regressão, previsão, estatísticas multivariadas, correspondência de padrões, modelagem preditiva e previsão.

Análise prescritiva

A análise prescritiva vai um pouco além da preditiva, porém a lógica envolvida é semelhante. Esta forma de análise examina dados ou conteúdo para responder à pergunta "*O que deve ser feito?*" Ou "*O que podemos fazer para fazer algo acontecer?*". Um pouco mais profunda que a análise preditiva, a análise prescritiva traduz as previsões em planos viáveis para o negócio. Ou seja, foca em predizer as possíveis consequências para as diferentes escolhas que forem feitas, e desta forma, este tipo de análise pode recomendar melhores caminhos a serem seguidos. E é caracterizada por técnicas como análise de gráficos, simulação, processamento de eventos, redes neurais, mecanismos de recomendação, heurística e aprendizado de máquina.

Capítulo 2. Modelagem de Dados

Modelagem[1] é uma representação simples, normalmente gráfica, de estrutura de dados reais mais complexas, sendo esta um modelo de uma abstração de um objeto ou evento real de maior complexidade. Sua função é auxiliar na compreensão das complexidades do ambiente real. O ato de modelar é a atividade de criar representações que expressam objetos ou eventos do mundo real (CHEN, 1990; COUGO, 1997; HEUSER, 2009).

Um modelo é uma representação abstrata e simplificada de um sistema real, com a qual se pode explicar ou testar o seu comportamento, em seu todo ou em partes. Por exemplo, uma planta baixa, um manequim etc.

A modelagem de dados é o passo inicial do projeto na criação de um banco de dados, pois é nele que criamos um modelo de dados específico para um determinado domínio. O modelo de dados é uma representação formal dos dados que forma a estrutura de um banco de dados, descrevem os tipos de informações que estão armazenadas em um banco de dados. Estes modelos são ferramentas que permitem demonstrar como serão construídas as estruturas de dados que darão suporte aos processos de negócio, como esses dados estarão organizados e quais os relacionamentos que pretendemos estabelecer entre eles.

Abstração de dados

Dada a complexidade dos sistemas do mundo real, e os diferentes níveis de envolvidos com o projeto de banco de dados, o modelo de dados deve expressar a visão

[1] Os modelos criados nas videoaulas estão disponíveis em: https://github.com/FernandaFarinelli/Disciplinas_IGTI/tree/main/EngenhariaDeDados/Modulo1/ScriptsVideoaulas/Capitulo2

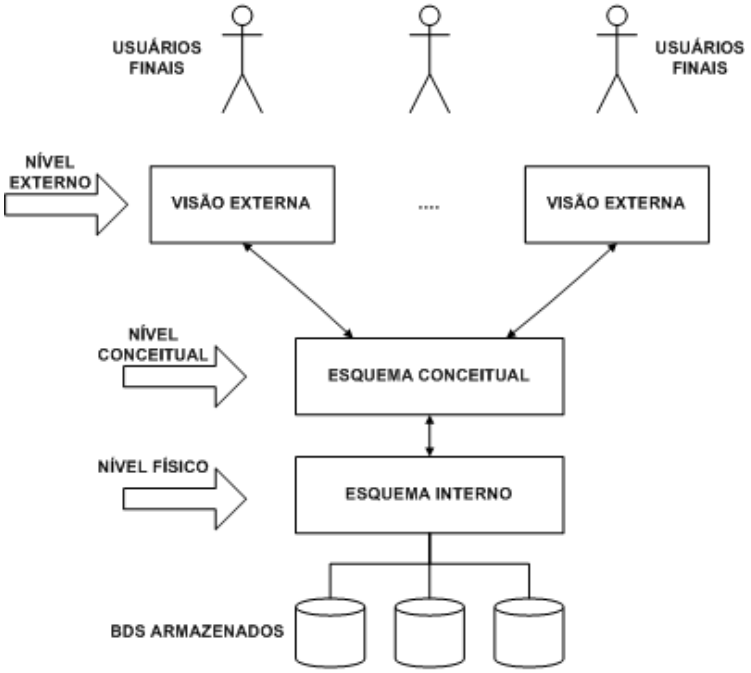
de mundo para estes diferentes níveis de usuário. Desta forma, é necessário construir uma abstração dos objetos e fenômenos do mundo real, de modo a obter uma forma de representação mais conveniente para cada envolvido. Abstração é o processo de identificar as qualidades ou propriedades importantes do problema que está sendo modelado. Neste sentido, consideramos três níveis de abstração descritos a seguir e ilustrados na Figura 8 (CHEN, 1990; COUGO, 1997; DAMA, 2017; ELMASRI; NAVATHE, 2005; HEUSER, 2009; SILBERSCHATZ; KORTH; SUDARSHAN, 2012):

- **Nível de visão do usuário:** o mais alto nível de abstração descrevendo apenas parte do banco de dados que são direcionadas para entendimento dos usuários finais. O nível de abstração das visões de dados é definido para simplificar esta interação com o sistema, que pode fornecer muitas visões para o mesmo banco de dados. Descreve os dados e as relações entre eles em geral e não a estrutura como são armazenados.

- **Nível conceitual ou lógico:** o próximo nível de abstração descreve quais dados estão armazenados de fato no banco de dados e as relações que existem entre eles. Neste nível o banco de dados inteiro é descrito em termos de um pequeno número de estruturas relativamente simples. Este nível de abstração é usado por administradores de banco de dados, que podem decidir quais informações devem ser mantidas no BD (entidades, atributos, relacionamentos), a tipologia dos dados e as principais restrições impostas sobre eles.

- **Nível físico:** o nível mais baixo de abstração, descreve a estrutura física dos dados e como estão realmente armazenados.

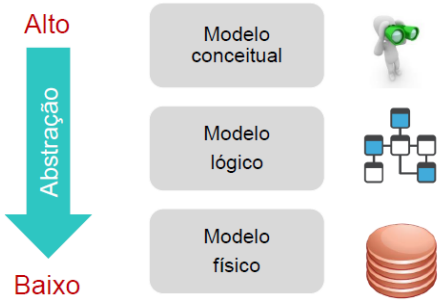
Figura 8: Arquitetura de referência dos níveis de abstração de dados.



Fonte: (ELMASRI; NAVATHE, 2005)

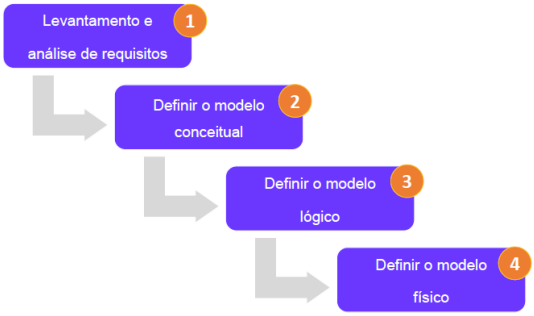
Dados esses níveis de abstração, os modelos de dados (Figura 9) também possuem diferentes níveis de detalhes aderentes aos diferentes níveis de abstração de representação de dados.

Figura 9: Abstração de dados versus modelos de dados.



O projeto de banco de dados é parte integrante no processo de desenvolvimento de sistemas de informação. O projeto tradicional de banco de dados (*top-down*), conforme ilustrado na Figura 10, consiste em Projeto Conceitual, Projeto Lógico e Projeto Físico, mas para isso, o projetista primeiro precisa conhecer os requisitos de dados que serão necessários ao projeto (HEUSER, 2009).

Figura 10: Etapas do projeto tradicional de banco de dados (top-down)



A etapa de levantamento e análise de requisitos consiste na coleta de informações sobre os dados, suas restrições e seus relacionamentos na organização. Geralmente ocorre por meio reuniões com os usuários; observação do funcionamento da organização que geram um documento com a especificação de requisitos. Muitas vezes, os requisitos são documentados em casos de uso que servem como fonte de dados ao projetista de banco de dados. Para compreensão das demais etapas, vamos primeiro compreender cada um dos modelos de dados.

Modelo conceitual

Representa ou descreve a realidade do ambiente do problema, constituindo-se em uma visão global de negócio dos principais dados e relacionamentos, independente

das limitações e especificações de implementação (tecnológicas ou paradigma de desenvolvimento). Ou seja, é um modelo que independe da implementação de SGBD. É uma descrição em alto nível, mas que tem a preocupação de capturar e retratar toda a realidade de uma organização, por isto é o modelo mais adequado para o envolvimento do usuário que não precisa ter conhecimentos técnicos (CHEN, 1990; COUGO, 1997). O modelo conceitual visa retratar os dados que devem aparecer em um projeto de banco de dados, mas não se preocupa como estes dados serão armazenados no seu SGBD.

Modelo entidade-relacionamento

O modelo entidade-relacionamento (MER) é um modelo de nível conceitual originalmente definido por Peter Chen em 1976, baseado na teoria relacional e teoria dos conjuntos, e passando a ser referência no processo de modelagem conceitual de dados. A construção deste modelo tem a finalidade de mostrar ao cliente os principais aspectos do banco de dados, assim como permitir uma interação mínima do usuário final com a tecnologia de banco de dados, e não se preocupando em representar como estes dados estarão realmente armazenados. Dessa forma, é possível a compreensão desse usuário de modo a garantir correção e respeito às regras de negócio por ele impostas. Este modelo possui basicamente três componentes principais: as entidades, os relacionamentos e os atributos (CHEN, 1990; COUGO, 1997; DAMA, 2017; ELMASRI; NAVATHE, 2005; HEUSER, 2009; SILBERSCHATZ; KORTH; SUDARSHAN, 2012).

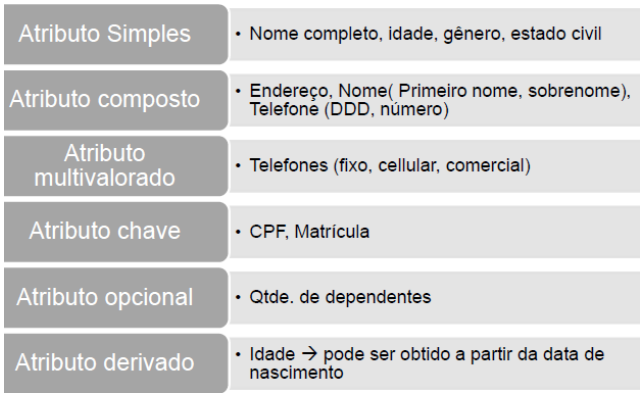
Entidade é um objeto do mundo real que pode ser distinguível dos outros objeto. Assim, um conjunto de entidades de mesma natureza apresenta as mesmas características ou atributos, abrigados sob um nome genérico, sobre as quais há necessidade de manter informações no banco de dados. Estes objetos podem ser concretos (existem fisicamente), como por exemplo pessoas e carros, ou abstratos (existência conceitual), por exemplo uma compra ou um curso.

As entidades se classificam como fortes ou fracas. As *entidades fortes* são aquelas cuja existência independe de outras entidades, ou seja, por si só elas já possuem total sentido de existir. Por exemplo, em um sistema de vendas, a entidade produto independe de quaisquer outras para existir. As *entidades fracas* são aquelas que dependem de outras entidades para existirem, pois individualmente elas não fazem sentido. Mantendo o mesmo exemplo, a entidade venda depende da entidade produto, pois uma venda sem itens não tem sentido.

Os ***atributos*** são propriedades ou características que descrevem uma entidade. Por exemplo, um carro pode ser caracterizado por uma cor e uma marca, uma pessoa pode ser caracterizada pelo nome e data de nascimento. Os atributos podem ser classificados pela sua cardinalidade como *monovalorado* ou *multivalorado*, ou seja, quantos atributos deste mesmo tipo uma entidade pode ter. Por exemplo, uma pessoa possui apena uma data de nascimento, entretanto ela pode ter nenhum, um ou múltiplos telefones de contato.

Os atributos ainda podem ser considerados simples ou compostos, ou seja, os atributos *simples* são atômicos ou indivisíveis, por exemplo a idade de uma pessoa, e os atributos *compostos* são aqueles que podem ser divididos em dois ou mais atributos, como por exemplo o endereço de uma pessoa que pode ser dividido em tipo do logradouro, nome do logradouro, número, complemento, bairro, código postal etc. Ainda podemos dizer que um atributo é *derivado* quando o valor deste depende do valor de um ou mais atributos. Por exemplo, o atributo idade, ele é determinado pela data de nascimento e pela data corrente, ou seja, ele deriva da diferença entre a data que está sendo medida e a data de nascimento.

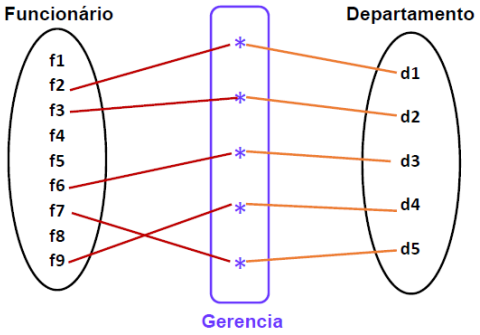
Figura 11: Exemplo de atributos por tipo



Os ***relacionamentos*** são as associações entre as entidades, ou seja, refere-se a como entidade se relaciona com outra entidade. Em geral representam ações que ocorrem entre duas ou mais entidades. Os relacionamentos se classificam conforme sua cardinalidade, de acordo com a lista abaixo:

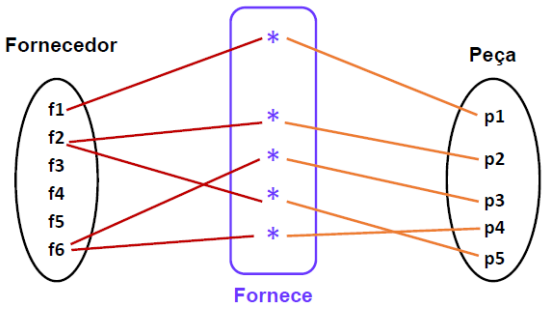
- Relacionamento 1..1 (*um para um*): cada uma das duas entidades envolvidas referenciam obrigatoriamente apenas uma unidade da outra. Por exemplo, em uma base de currículos, cada usuário cadastrado pode possuir apenas um currículo, ao mesmo tempo em que cada currículo só pertence a um único usuário cadastrado. Ou um funcionário pode ser gerente de um único departamento, e um departamento pode ter apenas um funcionário o gerenciando (Figura 12).

Figura 12: Exemplo de relacionamento 1:1



- Relacionamento 1..n ou 1..* (*um para muitos*): uma das entidades envolvidas pode referenciar várias unidades da outra, porém, do outro lado cada uma das várias unidades referenciadas só pode estar ligada a uma unidade da outra entidade. Por exemplo, em um sistema de plano de saúde, um usuário pode ter vários dependentes, mas cada dependente só pode estar ligado a um usuário principal. Ou um fornecedor pode fornecer várias peças (N), mas uma peça específica é fornecida por um único fornecedor (Figura 13).

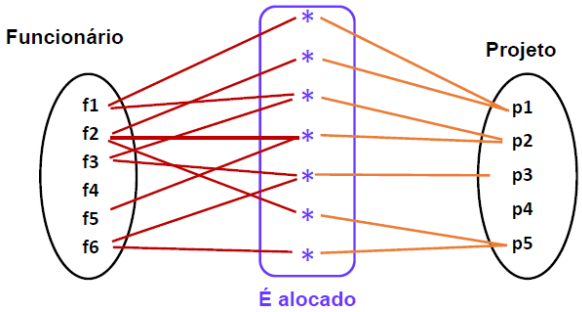
Figura 13: Exemplo de relacionamento 1:N ou N:1



- Relacionamento n..n ou *..* (*muitos para muitos*): neste tipo de relacionamento cada entidade, de ambos os lados, podem referenciar múltiplas unidades da outra. Por exemplo, em um sistema de biblioteca, um título pode ser escrito por vários autores, ao mesmo tempo em que um autor

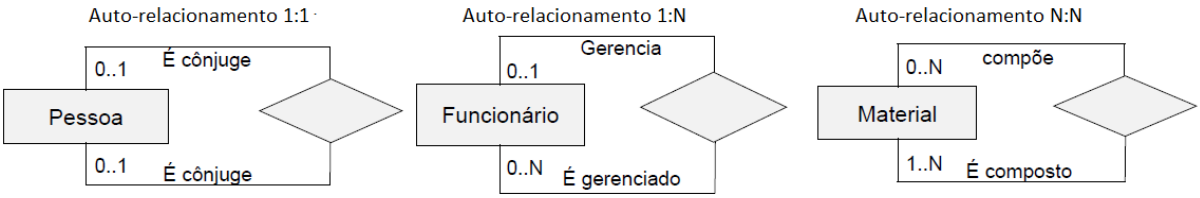
pode escrever vários títulos. Ou um funcionário pode ser alocado para trabalhar em múltiplos (N) projetos e um projeto pode ter múltiplos (N) funcionários designados para trabalhar nele (Figura 14).

Figura 14: Exemplo de relacionamento N:N



Nem sempre um relacionamento associa entidades diferentes, podemos ter o auto-relacionamento, onde uma entidade se relaciona com ela mesmo (Figura 15).

Figura 15: Exemplo de auto-relacionamento.



Por exemplo, vamos considerar uma entidade Pessoa e casamento, sendo que uma pessoa pode ser casada com uma outra pessoa, ou seja, um auto-relacionamento 1:1. Já no caso de uma entidade FUNCIONÁRIO onde nosso modelo representa o conceito de que um funcionário possui um gerente que é também funcionário, sendo que este gerente pode gerenciar múltiplos funcionários (1:N). Por fim, considere uma entidade MATERIAL onde um material pode ser composto por outros múltiplos materiais e, por sua vez, um material pode compor outros múltiplos materiais (N:N).

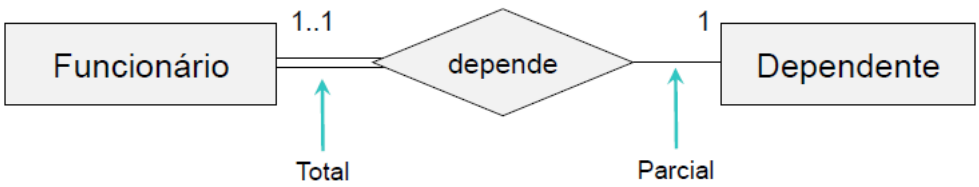
É importante notar que em um relacionamento N:N, em geral, criamos uma terceira entidade que é chamada de entidade associativa. Ela surge normalmente quando existe a necessidade de associar uma entidade a um relacionamento existente (Figura 16). O exemplo abaixo possui duas entidades, MÉDICO e PACIENTE, e um relacionamento chamado CONSULTA.

Figura 16: Exemplo de entidade associativa.

Outro conceito interessante é a ideia de participação no relacionamento. Este conceito especifica se a existência de uma entidade depende de ela estar relacionada com outra entidade através de um relacionamento. Assim, a participação pode ser:

- Total: a entidade existe somente quando ela se relaciona com outra entidade;
- Parcial: a existência de uma entidade independe do fato dela estar relacionada com outra.

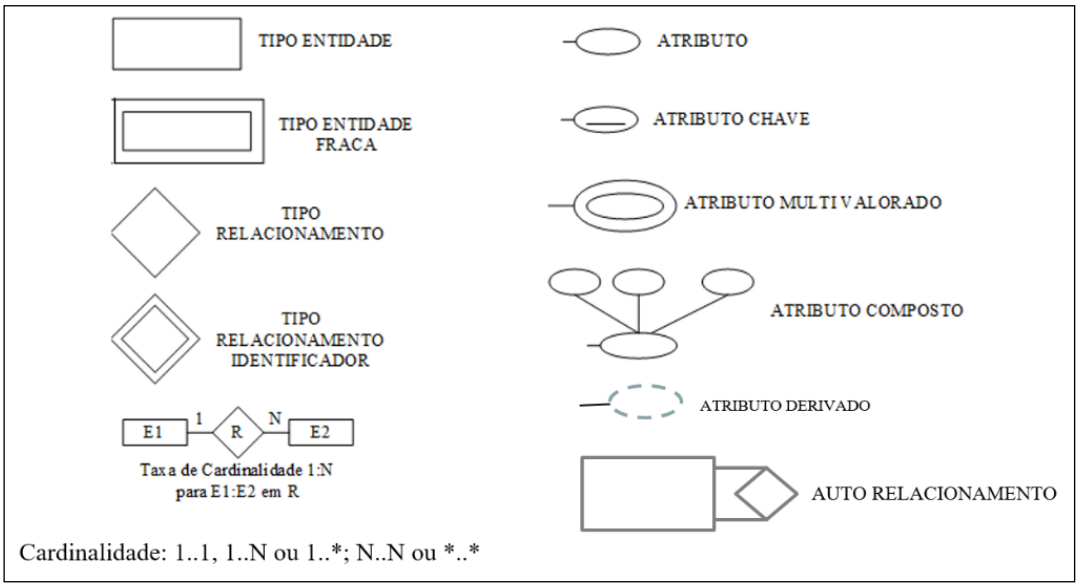
Figura 17: Exemplo de participação das entidades no relacionamento.



No exemplo da Figura 17 observamos que a entidade DEPENDENTE tem participação total em relação à entidade FUNCIONÁRIO, pois só existe dependente quando ela se relaciona com um funcionário. No caso da participação de funcionário é parcial, pois o funcionário existe independente de existir dependente.

O MER é graficamente representado pelo Diagrama de entidade-relacionamento (DER) e pode ser usada a notação de Peter Chen apresentada na Figura 18 a seguir.

Figura 18: Formas usadas na representação dos elementos do DER



Para criar um DER pode utilizar a ferramenta brModelo[2] desenvolvida para ensino de modelagem de banco de dados relacionais com base na metodologia defendida por Carlos A. Heuser no livro *Projeto de Banco de Dados*. Veja a referência Heuser (2009) e a versão aberta do livro[3]. A notação utilizada na ferramenta é ilustrada no DER a seguir na Figura 19 e Figura 20:

[2] Disponível para download em: http://www.sis4.com/brModelo/

[3] Disponível em: http://files.profrossana.webnode.com/200000190-473a348353/Projeto_de_Banco_de_Dados_-_Carlos_Alberto_Heuser.pdf

Figura 19: Exemplo de DER criado no brModelo.

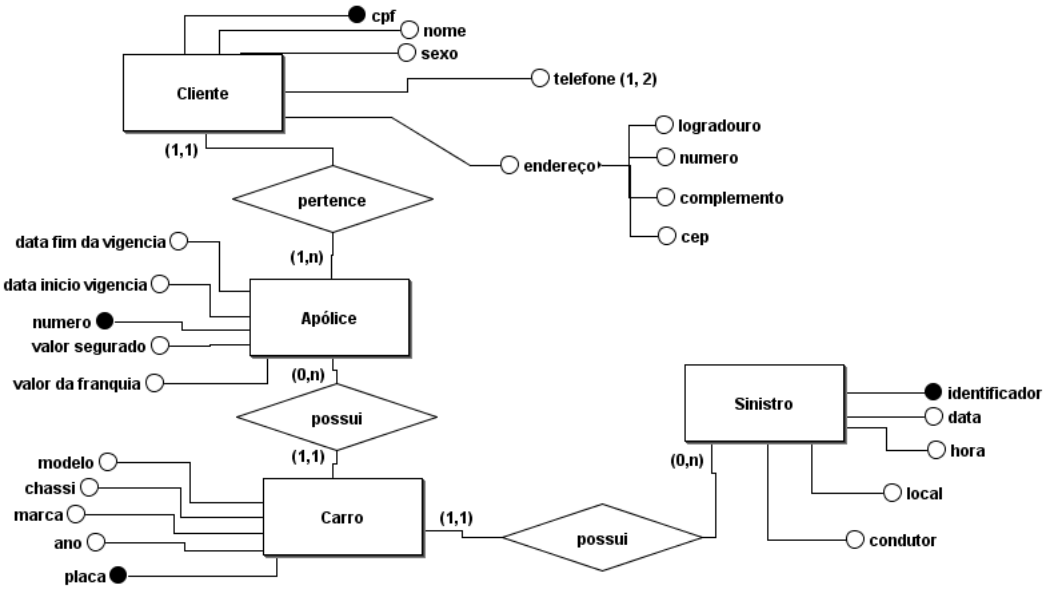
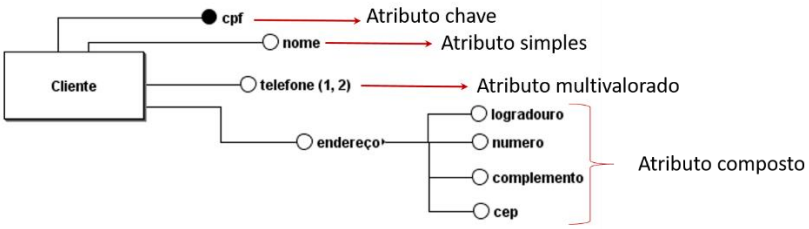


Figura 20: Notação de atributos do brModelo.

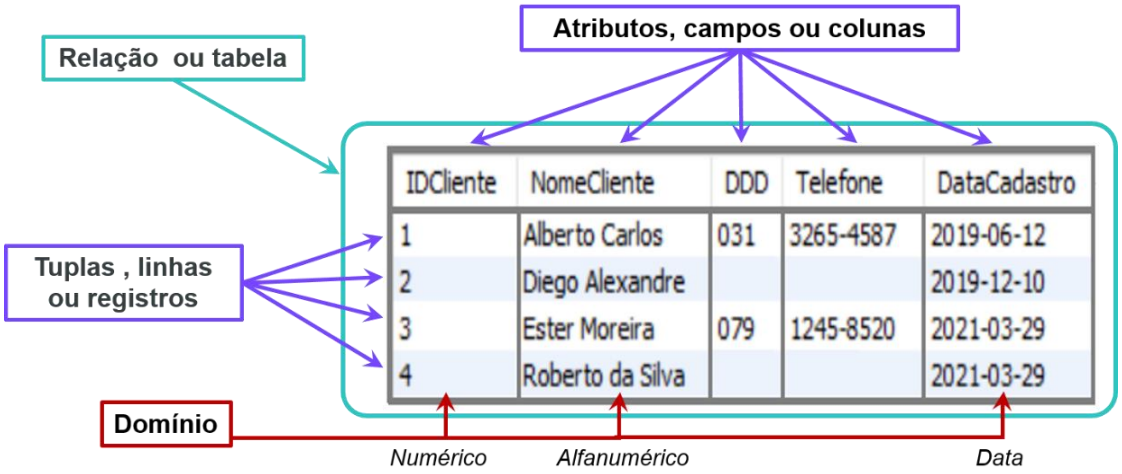


Modelo lógico

Um modelo derivado do modelo conceitual que leva em consideração algum paradigma tecnológico - paradigma hierárquico, em rede, relacional, orientado a objetos -, mas não uma ferramenta em si. O modelo lógico descreve as estruturas que estarão contidas no banco de dados, mas sem considerar ainda nenhuma característica específica de SGBD, resultando em um esquema lógico de dados (CHEN, 1990; COUGO, 1997; DAMA, 2017; ELMASRI; NAVATHE, 2005; HEUSER, 2009; SILBERSCHATZ; KORTH; SUDARSHAN, 2012).

O modelo relacional é um modelo lógico que visa representar os dados necessários a um sistema usando o paradigma relacional definido por Codd em 1970. Na sua estrutura fundamental apresentada na Figura 21, o modelo relacional possui a **relação** (entidade no modelo conceitual e tabela no modelo físico) e as associações ou **relacionamentos** entre as relações. Uma relação é constituída por um ou mais **atributos** (campos ou colunas no modelo físico) que traduzem o tipo de dados a armazenar. Cada instância ou ocorrência do esquema de dados é chamada de **tupla** (registro ou linha no modelo físico). Um atributo possui ainda um **domínio** vinculado a ele, ou seja, um conjunto de valores atômicos que o atributo pode assumir (ELMASRI; NAVATHE, 2005; HEUSER, 2009; SILBERSCHATZ; KORTH; SUDARSHAN, 2012).

Figura 21: Elementos de um modelo relacional.



Desta forma, em um modelo relacional, as estruturas do banco de dados relacional são vistas como um conjunto de relações ou tabelas, onde cada instância da relação é referenciada como uma tupla. Além disso, uma relação é caracterizada pelo seu conjunto de atributos que possuem um domínio específico. As relações se associam pelos relacionamentos. Além disso, no modelo relacional podem ser definidas restrições

de integridade que ajudam a garantir a coerência e consistência dos valores que os dados que serão armazenados (ELMASRI; NAVATHE, 2005). As regras de integridade são:

- *Integridade de Chave*: Define que os valores da chave primária e alternativa devem ser únicos e não nulos.

- *Integridade de Domínio:* Define os valores que podem ser assumidos pelos campos de uma coluna. Pode ser o tipo do dado, como por exemplo Numérico e Alfanumérico, ou pode ser os possíveis valores que vai assumir, por exemplo no caso de estado civil os únicos valores permitidos poderiam ser: solteiro, casado, separado, divorciado e viúvo.

- *Integridade de Vazio:* Especifica se os campos de uma coluna podem ou não serem vazios (nulos ou não nulos). Note que nulo é ausência de valores, ou seja, atribuir um valor numérico equivalente a zero, é diferente de não atribuir valor nenhum.

- *Integridade Referencial:* Define que os valores dos campos que aparecem numa chave estrangeira devem aparecer obrigatoriamente na chave primária (ou alternativa) da tabela referenciada.

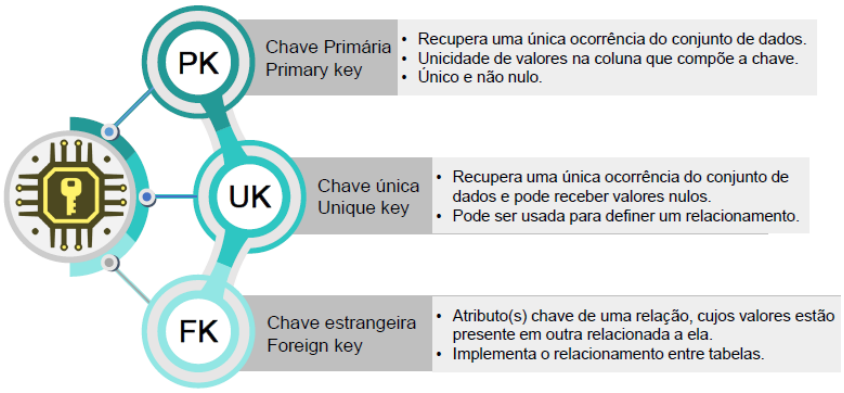
- *Integridade de Unicidade:* Define que o valor do campo pode repetir ou se são únicos. Por exemplo, o cpf é um valor único para cada pessoa no Brasil, não pode repetir para pessoas distintas, mas a idade, por exemplo, pode repetir, podemos ter vários clientes de uma mesma idade no nosso cadastro.

- *Integridade de valor padrão:* Define que quando o valor do campo não for informado ele assume um valor padrão predefinido.

- *Integridade semântica:* Integridade definida pelo usuário: Restrições definidas conforme os requisitos que delimitam o valor dos dados.

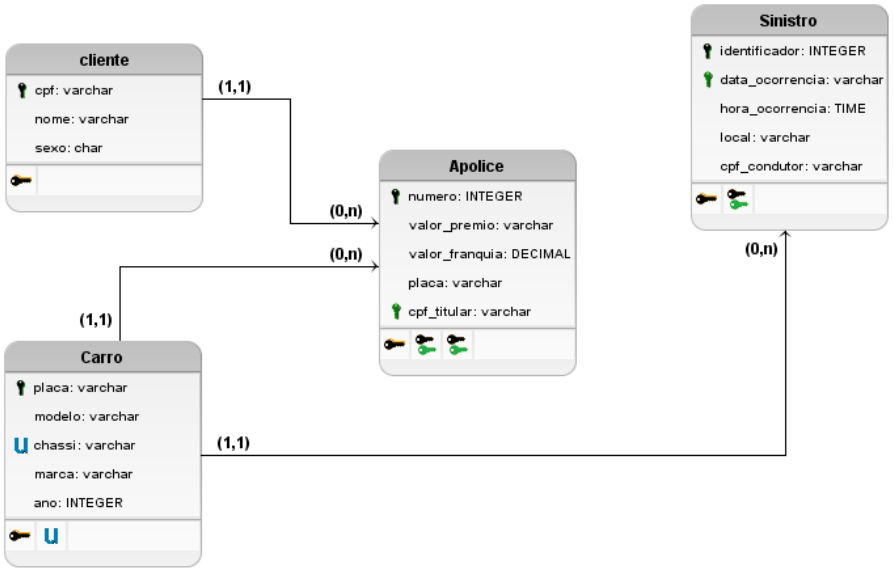
Ainda temos o conceito de chaves no modelo relacional, conforme descrito na Figura 22.

Figura 22: Conceito de chaves.



O processo de modelagem relacional, quando não derivado de um modelo conceitual como o MER, envolve, antes de tudo, o entendimento sobre o negócio que será modelado. Isso pode ser pela compreensão dos requisitos do sistema, pela compreensão das regras de negócio, ou os dois no melhor dos casos.

Figura 23: Exemplo de modelo relacional criado no brModelo.



Modelo de dados físico

O modelo de dados físico, como mencionado anteriormente, descreve, os dados e as estruturas que serão armazenadas do ponto de vista da tecnologia que foi escolhida para implementar o modelo de dados. É criado por meio de alguma linguagem, que descreve como será feita a armazenagem dos dados no SGBD. Neste nível de modelagem, se escolhe qual SGBD será usado, levando em consideração o modelo lógico adotado. Por exemplo, para um modelo lógico relacional, pode ser escolhido um SGBD relacional (CHEN, 1990; COUGO, 1997; DAMA, 2017; ELMASRI; NAVATHE, 2005; HEUSER, 2009; SILBERSCHATZ; KORTH; SUDARSHAN, 2012).

Assim, para melhor compreender o modelo físico relacional, antes é necessário entender o que são os SGBD relacionais.

Fundamentos de banco de dados

Os sistemas de informação não são nada sem os dados e estes precisam estar armazenados em algum repositório. O conceito de armazenamento de dados é muitas vezes relacionado à persistência de dados, ou seja, o dado deve ser persistido ou armazenado em local não volátil, de forma que eles possam ser recuperados posteriormente para consulta e análise. Isso significa armazenar estes dados em um local que possa garantir a integridade dos dados por um período de tempo indeterminado, até que eles sejam atualizados ou descartados propositalmente.

Um banco de dados é “uma coleção de dados inter-relacionados, representando informações sobre um domínio específico”. Um sistema de banco de dados como o conjunto de quatro componentes básicos: dados, hardware, software e usuários (ELMASRI, NAVATHE, 2005; SILBERSCHATZ, KORTH, SUDARSHAN, 2012).

Para suportar as necessidades dos sistemas de bancos de dados, foram criados os Sistema Gerenciadores de Banco de Dados (SGBD ou em inglês Data Base Management System - DBMS). SGBDs são sistemas ou softwares utilizados para gerir os bancos de dados, permitindo: i) criar, modificar e eliminar bases de dados; ii) realizar as operações básicas com os dados (inserir, alterar, excluir e consultar); iii) garantir a segurança de acesso aos dados; iv) garantir a integridade de dados, controle de concorrência e possibilidades de recuperação e tolerância a falhas (SILBERSCHATZ, KORTH, SUDARSHAN, 2012).

Os objetivos de um sistema de banco de dados são o de isolar o usuário dos detalhes internos do banco de dados (promover a abstração de dados) e promover a independência dos dados em relação às aplicações, ou seja, tornar independente da aplicação a estratégia de acesso e a forma de armazenamento. Existem diversos paradigmas tecnológicos de SGBDs, como por exemplo os SGBDs em rede, hierárquicos, relacionais, NoSQL.

Sistemas gerenciadores de bancos de dados relacionais

Os bancos de dados relacionais são os mecanismos de persistência de dados mais adotado por empresas nas últimas décadas. É um SGBD que implementa o modelo relacional de dados. Fundamentado na teoria de conjuntos e nas possíveis relações entre os conjuntos, permitindo operações de junção, união, retorno seletivo de dados e diversas outras operações matemáticas. Até recentemente, este modelo foi considerado o mais adequado ao solucionar os vários problemas que se colocam no nível da concepção e implementação da base de dados para tratamento de dados estruturados (ELMASRI, NAVATHE, 2005; HEUSER, 2008).

Os bancos de dados relacionais são adequados para solucionar problemas que se colocam no nível da concepção e normalmente o uso mais comum deste tipo de

banco é para implementar funcionalidades do tipo CRUD (do inglês *Create*, *Read*, *Update* e *Delete*), ou seja, criar ou inserir, ler ou selecionar, alterar e excluir um dado, por meio da linguagem SQL. As funcionalidades dos sistemas de informação fazem interface com os bancos de dados utilizando estas quatro operações básicas (CRUD) e geralmente por múltiplas aplicações em paralelo. Além disso, uma determinada funcionalidade pode envolver múltiplas operações, assim temos o conceito de transação em um SGBD. Uma transação é uma sequência de operações executadas como uma única unidade lógica de trabalho.

Modelo de dados relacional físico

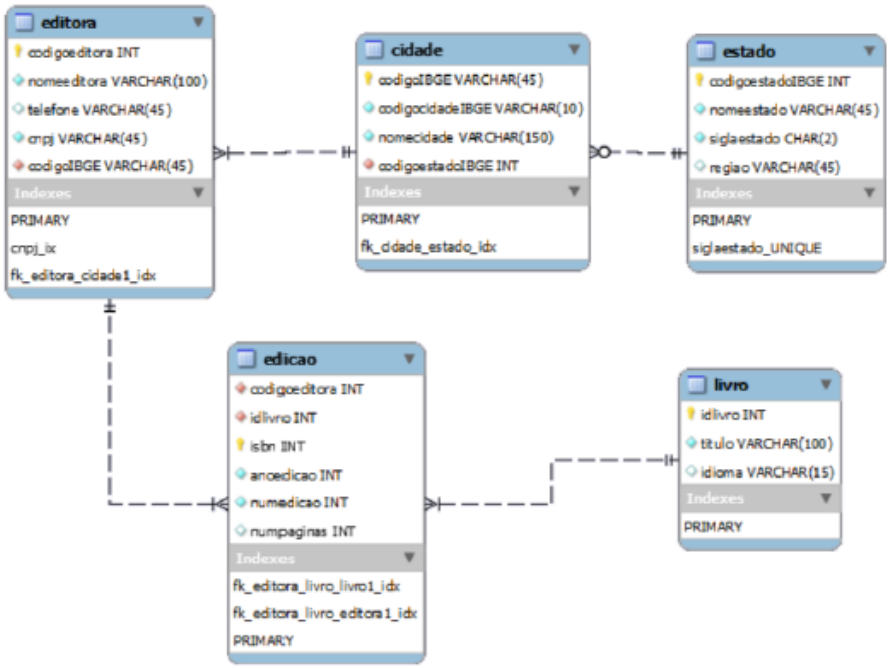
Modelo derivado do modelo lógico relacional, que descreve as estruturas físicas de armazenamento de dados, tais como: tamanhos de campos, índices, tipos de dados, nomenclaturas etc. Este modelo leva em consideração limites impostos pelo SGBD relacional escolhido e pelos requisitos não funcionais dos programas que acessam os dados (ELMASRI, NAVATHE, 2005).

Por exemplo, no caso de escolher o SGBD Oracle, deve-se ter em mente que ele implementa tipos de dados diferentes do SQL Server. No Oracle temos um tipo de dados chamado VARCHAR2 e no SQL Server o tipo de dados semelhante seria o VARCHAR. Outra diferença é que no SQL Server podemos criar uma coluna com a opção autoincremento, já no Oracle não temos esta opção, mas temos um objeto chamado SEQUENCE para gerar dados sequenciais. É importante ter em mente que estas diferenças alteram o modelo físico, pois são características específicas do SGBD escolhido.

Apresentamos na Figura 24 um exemplo de modelo relacional desenhado com o apoio da ferramenta MySQL Workbench[4]. Conforme a figura, observamos 5 tabelas chamadas *livro, editora, edição, cidade, estado*.

Cada uma delas possui uma chave primária que é identificada com o desenho de uma "chave" em frente ao nome do atributo identificado como chave. Por exemplo, na tabela *livro* o atributo PK é *idlivro*. Os demais atributos que não aceitam nulo (NOT NULL) são precedidos por um losango azul. No caso dos losangos vermelhos, eles identificam que o atributo é não nulo e também uma chave estrangeira, por exemplo o atributo *idlivro* na tabela *edicao.* Por fim, os atributos onde o losango não é preenchido indicam que é um atributo opcional que aceita valores nulos (NULL).

Figura 24 – Exemplo de modelo relacional.



[4] Disponível para download em: http://fabforce.eu/dbdesigner4/

Capítulo 3. Linguagem SQL

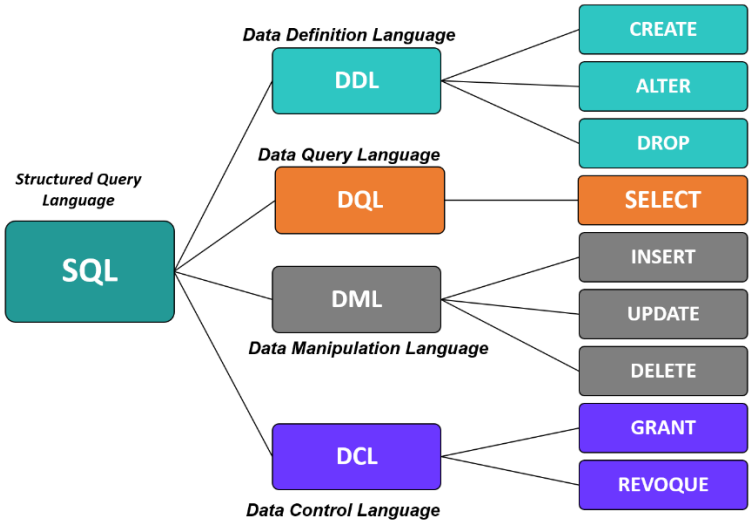
A Linguagem Estruturada de Consultas mais conhecida como linguagem SQL (do inglês *Structured Query Language*) surgiu no início dos anos 70 com o objetivo de fornecer uma interface mais amigável ao usuário para acesso aos bancos de dados. Foi um grande sucesso, sendo que a maioria dos SGBDs relacionais atuais a utilizam. A linguagem SQL é uma linguagem criada para interagir com os bancos de dados relacionais, pois, assim como os próprios bancos de dados, a linguagem é criada com o conceito de teoria de conjuntos. SQL serve para criar tanto as estruturas como os dados nos bancos de dados (W3SCHOOLS, 2022).

- SQL pode executar consultas em um banco de dados;
- SQL pode recuperar dados de um banco de dados;
- SQL pode inserir, atualizar e excluir registros em um banco de dados;
- SQL pode criar, alterar e excluir novos objetos no bancos de dados;
- SQL pode definir permissões em tabelas, procedimentos e visualizações.

A linguagem SQL[5] é dividida em sub-linguagens, conforme ilustrado na Figura a seguir que apresenta os principais comandos de cada uma (ELMASRI; NAVATHE, 2005; SILBERSCHATZ; KORTH; SUDARSHAN, 2012).

[5] Os scripts usados dos nas videoaulas dos capítulos 3 e 4 estão disponíveis em:
https://github.com/FernandaFarinelli/Disciplinas_IGTI/tree/main/EngenhariaDeDados/Modulo1/ScriptsVideoaulas/Capitulo3
https://github.com/FernandaFarinelli/Disciplinas_IGTI/tree/main/EngenhariaDeDados/Modulo1/ScriptsVideoaulas/Capitulo4

Figura 25 – Divisões da linguagem SQL.



- Linguagem de Definição de Dados: DDL (*Data Definition Language*) é a parte da linguagem SQL utilizada para criação ou definição dos elementos ou estrutura do banco de dados, assim como a modificação e remoção das estruturas. Ou seja, permite criar, alterar e excluir as estruturas como tabelas, índices e procedimentos.

- Linguagem de Consulta de Dados: DQL (*Data Query Language*) é um conjunto de instruções usado para consultar dados nas estruturas dos objetos armazenados.

- Linguagem de Manipulação de Dados: também conhecida como DML (*Data Manipulation Language*), é o subconjunto da linguagem SQL, utilizada para realizar as operações de manipulação de dados, como inserir, atualizar e apagar dados.

- Linguagem de Controle de Dados: DCL (*Data Control Language*) é a parte da linguagem SQL que controla os aspectos destinados a autorização de acesso aos dados para manipulação de dados dentro do BD.

Linguagem de Definição de Dados

Um esquema físico de banco de dados é determinado por um conjunto de definições, no caso dos SGBDs relacionais estas definições são expressas em linguagem SQL, por meio de sua sub-linguagem chamada linguagem de definição de dados (DDL). A linguagem de definição de dados (DDL) é uma linguagem de computador usada para criar e modificar a estrutura de objetos de banco de dados em um banco de dados. Esses objetos de banco de dados incluem visões, esquemas, tabelas, índices, procedimentos armazenados, funções, gatilhos (triggers) etc. (ELMASRI; NAVATHE, 2005; SILBERSCHATZ; KORTH; SUDARSHAN, 2012).

Em suma, é possível criar, alterar e excluir uma estrutura ou objeto no SGBD. A sintaxe básica é (W3SCHOOLS, 2022):

- CREATE objeto (Para criar um novo objeto);
- ALTER objeto (Para alterar/modificar um objeto já existente);
- DROP objeto (Para excluir um objeto existente).

Cada objeto criado tem sua sintaxe específica, conforme apresentado a seguir.

Banco de dados e esquema de banco de dados[6]:

Criar:

```
CREATE DATABASE nome_banco_de_dados;

CREATE SCHEMA nome_do_esquema;
```

[6] Pratique em: https://www.w3schools.com/sql/sql_create_db.asp

Excluir:

```
DROP DATABASE nome_banco_de_dados;

DROP SCHEMA nome_do_esquema;
```

Alterar:

```
ALTER DATABASE nome_banco_de_dados
    opção_de_alteração;

ALTER SCHEMA nome_banco_de_dados
    opção_de_alteração;
```

OBSERVAÇÃO:

1. No MySQL, fisicamente, um esquema é sinônimo de um banco de dados. Você pode substituir a palavra-chave SCHEMA por DATABASE na sintaxe SQL do MySQL, por exemplo, utilizando CREATE SCHEMA no lugar de CREATE DATABASE.

2. Na criação de um banco de dados /esquema ainda é possível passar um conjunto de parâmetros para a criação, e neste caso cada SGBD possui seu próprio conjunto de opções que podem ser alteradas.

3. No caso da alteração, cada SGBD possui seu próprio conjunto de opções que podem ser alteradas. A seguir, você pode consultar as opções para diferentes SGBDs:

    a. MySQL: https://dev.mysql.com/doc/refman/8.0/en/alter-database.html

    b. ORACLE: https://docs.oracle.com/cd/E11882_01/server.112/e41084/statements_1004.htm#SQLRF00802

c. MS SQL Server: https://docs.microsoft.com/pt-br/sql/t-sql/statements/alter-database-transact-sql?view=sql-server-ver15&tabs=sqlpool

d. PostgreSQL: https://www.postgresql.org/docs/9.1/sql-alterdatabase.html

4. Em alguns SGBDs, como o MySQL, pode ser usada a cláusula IF NOT EXIST para evitar erro na criação, caso o objeto já exista.

```
CREATE DATABASE IF NOT EXIST nome_banco_de_dados;

CREATE SCHEMA IF NOT EXIST nome_banco_de_dados;
```

Tabela

Criar[7]**:**

```
CREATE TABLE nome_da_tabela ( lista de opções para
criação );
```

A Figura 26 a seguir apresenta um exemplo utilizando o padrão ANSI.

[7] Pratique em: https://www.w3schools.com/sql/sql_create_table.asp

Figura 26: Exemplo sintaxe de criação de tabela.

```
CREATE TABLE tabela (
  atrib1 tipo [(tamanho)] [NOT NULL | DEFAULT valor]
              [CHECK (condição)],
  atrib2 tipo [(tamanho)] [NOT NULL | DEFAULT valor]
              [CHECK (condição)],
    ...
  [CONSTRAINT nome da restrição] PRIMARY KEY (<atributos chave primária>),
  [CONSTRAINT nome da restrição] UNIQUE (< atributos chave candidata>),
  [CONSTRAINT nome da restrição] FOREIGN KEY (<atributos chave estrangeira>)
                REFERENCES tabelaRef [(<chave primária>)]
                    [ON DELETE CASCADE | SET NULL | SET DEFAULT]
                    [ON UPDATE CASCADE | SET NULL | SET DEFAULT],
   [CONSTRAINT nome da restrição] CHECK (condição)
);
```

Alterar[8]:

ALTER TABLE *nome_da_tabela ação;*

Onde *<ação>*:

ADD nome_nova_coluna tipo[<restrições de coluna>]

ADD [CONSTRAINT nome_da_crestrição]<restrição de tabela>

DROP nome_coluna [CASCADE | RESTRICT]

DROP CONSTRAINT nome_da_restrição

SQL Server / MS Access

ALTER COLUMN nome_coluna tipo [<restrições de coluna>];

[8] Pratique em: https://www.w3schools.com/sql/sql_alter.asp

```
ALTER COLUMN nome_coluna DROP DEFAULT;

ALTER COLUMN nome_coluna SET DEFAULT <valor>;
```

My SQL / Oracle (atá versão 9):

```
MODIFY COLUMN nome_coluna tipo[<restrições de coluna>]
```

Oracle (a partir da versão 10):

```
MODIFY nome_coluna tipo[<restrições de coluna>]
```

Na criação e alteração, a lista de opções/ações pode variar conforme o SGBD escolhido, assim, deve-se consultar a documentação de cada SGBD para verificar a sintaxe completa do comando.

Excluir[9]:

```
DROP TABLE nome_da_tabela [CASCADE | RESTRICT];
```

Índice[10]

Criar:

```
CREATE INDEX nome_do_indice
ON nome_da_tabela (nome_coluna, ...);

Ou

CREATE UNIQUE INDEX nome_do_indice
ON nome_da_tabela (nome_coluna, ...);
```

[9] Pratique em: https://www.w3schools.com/sql/sql_drop_table.asp

[10] Pratique em: https://www.w3schools.com/sql/sql_create_index.asp

Excluir:

MS Access:

```
DROP INDEX nome_do_indice ON nome_da_tabela;
```

SQL Server:

```
DROP INDEX nome_da_tabela.nome_do_indice;
```

DB2/Oracle:

```
DROP INDEX nome_do_indice;
```

MySQL:

```
ALTER TABLE nome_da_tabela DROP INDEX
nome_do_indice;
```

Visão[11]

Criar:

```
CREATE VIEW nome_da_view AS

<comando de consulta (DQL)>;
```

Excluir:

```
DROP VIEW nome_da_view;
```

[11] Pratique em: https://www.w3schools.com/sql/sql_view.asp

O procedimento armazenado e a função são rotinas compostas por um conjunto de comando escritos em linguagem SQL, que são armazenado como objeto de banco de dados, e assim podem ser executadas em um banco de dados de uma só vez, podendo ser reutilizado repetidamente por meio de chamada ao procedimento. A diferença entre estas rotinas é que a função retorna um valor calculado dentro dela, e a procedure não. A sintaxe de criação do procedimento armazenado e da função pode variar conforme o SGBD, consulte a documentação para saber como criar. A sintaxe básica é:

Criar:

```
CREATE PROCEDURE nome (lista opcional de parâmetros)
AS

<conjunto de instruções SQL>;

CREATE FUNCTION nome (lista opcional de parâmetros)
AS

<conjunto de instruções SQL>

RETURNS <valor>;
```

Excluir:

```
DROP PROCEDURE nome_da_procedure;

DROP FUNCTION nome_da_procedure;
```

Executar procedure:

```
EXEC nome_da_procedure;   Ou   CALL
nome_da_procedure;
```

Triggers ou gatilho

Um Trigger é um procedimento armazenado no banco de dados que é chamado automaticamente sempre que ocorre um evento especial no banco de dados. Este evento pode ser uma inclusão, alteração ou exclusão de dados em tabelas. A sintaxe de criação do procedimento armazenado e da função pode variar conforme o SGBD, consulte a documentação para saber como criar. A sintaxe básica é:

Criar:

```
CREATE TRIGGER nome ON nome_da_tabela
[BEFORE | AFTER] [INSERT | UPDATE | DELETE]
<conjunto de instruções SQL>;
```

Excluir:

```
DROP TRIGGER nome;
```

Linguagem de Manipulação de Dados

A linguagem de manipulação de dados (*Data Manipulation Language*, DML) é a sub-linguagem da linguagem SQL que permite aos usuários manipular dados, ou seja, incluir dados em uma tabela, alterar e excluir dados já existentes na tabela. Para os exemplos de DML será usado a seguinte tabela:

```
CREATE TABLE `editora` (
  `codigo_editora` int NOT NULL AUTO_INCREMENT,
  `nome_editora` varchar(100) NOT NULL,
  `telefone` varchar(45) NULL,
  `codigo_cidade` varchar(45) NULL,
  PRIMARY KEY (`codigo_editora`)
);
```

Os comandos básicos DML são:

- INSERT: inclui/insere/adiciona novas informações no banco de dados;
- UPDATE: modifica/altera informações armazenadas no banco de dados;
- DELETE: exclui/deleta informações do banco de dados.

Inserir

É o comando DML usado para inserir dados em uma tabela já existente no seu banco de dados. A sintaxe básica do comando possui a seguinte estrutura:

```
INSERT INTO nome_tabela[(lista_atributos)]
    VALUES(lista_valores_atributos)
    [, (lista_valores_atributos)];
```

Exemplos:

```
INSERT INTO `editora`(`codigo_editora`,`nome_editora`,`telefone`,`codigo_cidade`)
VALUES (1,"Casa dos Espiritos",NULL,3550308);

INSERT INTO `editora`(`nome_editora`,`telefone`,`codigo_cidade`)
VALUES ("Editora Lê","3145678952",3106200);

INSERT INTO `editora`(`nome_editora`,`telefone`,`codigo_cidade`)
VALUES ("Moderna",NULL,3550308),("Objetiva","3199684562",3106200),("Manole","1124537859",4314902);
```

```
INSERT INTO `livraria`.`editora` VALUES (6,"Novo Conceito",NULL,1302603);

INSERT INTO `livraria`.`editora`(`nome_editora`,`codigo_cidade`) VALUES ("Benvirá",5300108);

INSERT INTO `livraria`.`editora` VALUES (DEFAULT,"Scipione",NULL,5300108);

INSERT INTO `livraria`.`editora`(`nome_editora`,`telefone`,`codigo_cidade`) VALUES ("Atica",NULL,3550308);
INSERT INTO `livraria`.`editora`(`nome_editora`,`telefone`,`codigo_cidade`) VALUES ("Campus",NULL,3304557);
INSERT INTO `livraria`.`editora`(`nome_editora`,`telefone`,`codigo_cidade`) VALUES ("Objetiva","145263987",3304557);
INSERT INTO `livraria`.`editora`(`nome_editora`,`telefone`,`codigo_cidade`) VALUES ("Bookman","214578963",3304557);
INSERT INTO `livraria`.`editora` (`nome_editora`, `telefone`, `codigo_cidade`) VALUES (13,'Record', '124578256', '3550308');
```

OBSERVAÇÃO: Caso a tabela possua uma coluna de autoincremento, essa coluna pode ser omitida, pois o próprio SGBD vai atribuir o valor. Esse tipo de coluna existe em SGBDs como o MySQL, MariaDB, MS SQL Server.

Alterar

É o comando DML usado para modificar dados existentes em uma tabela, sejam dados individuais ou grupos de dados, ou seja, você pode atualizar um único dado ou vários dentro de uma tabela em seu banco de dados. A sintaxe básica do comando possui a seguinte estrutura:

```
UPDATE nome_tabela
SET nome_atributo_1 = valor [{,nome_atributo_n = valor}]
[WHERE condição];
```

Exemplos:

```
UPDATE `livraria`.`editora` SET `nome_editora` = "Novatec" WHERE `codigo_editora` = 11;

UPDATE `livraria`.`editora` SET `nome_editora` = "Novatec ltda", `telefone` = NULL
WHERE `codigo_editora` = 11;

UPDATE `livraria`.`editora` SET `telefone` = "Não informado" WHERE `telefone` IS NULL;
```

Excluir

É o comando DML usado para excluir dados existentes em uma tabela. A sintaxe básica do comando possui a seguinte estrutura:

```
DELETE FROM nome_tabela [WHERE condição];
```

Exemplo:

```
DELETE FROM editora WHERE  codigo_editora = 11;
```

OBSERVAÇÃO: Note que se não informada a condição nos comandos UPDATE e DELETE, a ação será aplicada em todos os dados/registros da tabela.

```
UPDATE editora SET `telefone` = "Não informado";

DELETE FROM editora;
```

Linguagem de Controle de Dados

A Linguagem de Controle de Dados, ou do inglês *Data Control Language* (DCL), é uma sub-linguagem da linguagem SQL, usada para controlar o acesso aos dados e objetos de dados em um banco de dados. Os comandos DCL permitem conceder e revogar permissões ou privilégios de acesso a objetos do banco de dados para usuários e papéis (roles).

Permissões ou privilégios

Os privilégios são os tipos de ações que um usuário ou role pode realizar no banco de dados. Os tipos comuns de privilégios são:

- Privilégio em Nível de Conta: Este nível de privilégio atua em um contexto mais alto, ou seja, no nível da conta de usuários, sem entrar nos detalhes de qual serão os privilégios de atuação nas tabelas (relações) do banco de dados. Podemos chamar de privilégios de primeiro nível. Neste nível encontramos os seguintes privilégios:
  - CREATE SCHEMA: Criação de esquemas na base de dados;
  - CREATE TABLE: Criação de tabelas;

- CREATE VIEW: Criação de visões;
- ALTER: Para aplicar mudanças de esquema, como por exemplo, a inclusão ou remoção de atributos nas relações (tabelas);
- DROP: Para exclusão de relações (tabelas) ou visões(views);
- MODIFY: Para inserir, excluir ou atualizar tuplas (linhas);
- SELECT: Para recuperar informações no banco de dados.

- Privilégio em Nível de Objeto: Neste nível, o controle de privilégios é exercido nos acessos aos objetos individuais no banco de dados. A conta usada para criar o objeto é considerada o proprietário ou owner do objeto, e assim recebe todos os privilégios para atuar neste objeto. Os demais usuários devem receber a permissão para atuar neste objeto. Estes privilégios ou permissões são:
  - Privilégio Leitura: Ou seja, privilégio para poder consultar o objeto, em geral uma tabela ou visão. É o privilégio de SELECT.
  - Privilégios de Modificações: Privilégio que concede ao usuário a possibilidade de modificar os dados existentes em uma tabela e/ou visão. Neste caso, a modificação pode ser de inclusão, alteração e exclusão por meios dos privilégios de INSERT, UPDATE e DELETE, respectivamente. Para os privilégios INSERT e UPDATE, na maior parte dos SGBDs é possível especificar quais atributos da tabela poderão ser modificados.
  - Privilégios de Referência: Refere-se à possibilidade de referenciar uma tabela, ou seja, especificar restrições de integridade de chave estrangeira. No geral, este privilégio é o REFERENCE.

– Privilégios de execução: Refere-se à possibilidade de executar uma procedure ou função. O privilégio é o EXECUTE.

– Para conceder todos os privilégios, existe o privilégio ALL PRIVILEGES.

Os dois comandos mais usados na DCL são GRANT e REVOKE. O GRANT concede o privilégio ao usuário ou role e o REVOKE revoga o privilégio. A sintaxe do comando é:

Conceder:

```
GRANT<lista de privilegios> ON<objeto>
TO<lista de usuários/roles>
[WITH GRANT OPTION] [GRANTED BY grantor];
```

Onde:

- Lista de usuários/roles: pode ser um conjunto separado por vírgulas ou PUBLIC.
- O garantidor (grantor) é o usuário corrente (CURRENT_USER) ou o papel (CURRENT_ROLE), é opcional no comando.
- WITH GRANT OPTION: Indica que quem recebeu o privilégio pode propagar a outros usuários ou roles.

Exemplos:

```
GRANT ALL PRIVILEGES ON exemplo TO 'dba';
GRANT SELECT ON exemple.* TO 'read';
GRANT INSERT, UPDATE, DELETE ON exemple.* TO 'write';
GRANT SELECT, INSERT, UPDATE, DELETE ON exemple.* TO 'read_write';

GRANT 'read_write' TO 'developer';
GRANT CREATE, DROP, REFERENCES, ALTER, EXECUTE, CREATE VIEW, TRIGGER ON *.* TO developer;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON exemple.* TO 'usuario1';
GRANT 'developer' TO 'usuario2'@'localhost';

GRANT INSERT ON example.* TO 'usuario3'@'localhost';
GRANT 'read' TO 'usuario3'@'localhost';
```

Revogar:

```
REVOKE[GRANT OPTION FOR] <lista de privilegios>
ONobjeto
FROM<usuários/roles> [RESTRICT|CASCADE] ;
```

Exemplos:

```
REVOKE DELETE FROM 'read_write';

REVOKE 'read' FROM 'usuario3'@'localhost';

REVOKE INSERT, UPDATE, DELETE ON exemple.* FROM 'usuario1';
```

Role ou papel

Uma role ou papel é um agrupamento de permissões que pode ser concedida a usuários ou outras roles. Sua utilização ajuda a administrar as permissões de objetos no banco de dados, já que unifica um conjunto de permissões não necessitando executar concessões e revogações individuais.

Criar um novo papel.

```
CREATE ROLE <role>[,<lista de roles>];
CREATE ROLE 'developer';
CREATE ROLE 'dba', 'read', 'write','read_write';
```

Exclui um papel.

```
DROP ROLE <role>[,<lista de roles>];
```

```
DROP ROLE 'developer';
DROP ROLE 'dba', 'read', 'write','read_write';
```

Usuário

Cria um novo usuário.

```
CREATE USER <usuario>[@<hostname>]
IDENTIFIED BY PASSWORD(<senha>)];
CREATE USER 'usuario1' IDENTIFIED BY 'userpw1';
CREATE USER 'usuario2'@'localhost' IDENTIFIED BY 'userpw2';
```

Exclui um usuário.

```
DROP USER <usuario>[@<hostname>];
DROP USER 'usuario1';
DROP USER 'usuario2'@'localhost';
```

Linguagem de Consulta de Dados

A Linguagem de Consulta de Dados ou *Data Query Language* (DQL) é a sub-linguagem do SQL responsável por realizar consultas (leitura) aos dados de uma determinada tabela do banco de dados. Ela possui apenas um único comando: SELECT. É possível consultar uma ou mais tabelas utilizando um único comando SQL-DQL.

Consultando uma única tabela

A sintaxe básica de consulta de uma única tabela é:

```
SELECT lista de colunas      ou   SELECT *
FROM tabela;                      FROM tabela;
```

Onde * significa que são todas as colunas da tabela.

Exemplos:

```
SELECT *
FROM editora;
```

```
SELECT editora.codigo_editora, editora.nome_editora, editora.telefone, editora.codigo_cidade
FROM editora;
```

RESULTADO

codigo_editora	nome_editora	telefone	codigo_cidade
1	Casa dos Espiritos	NULL	3550308
2	Editora Lê	3145678952	3106200
3	Moderna	NULL	3550308
4	Objetiva	3199684562	3106200
5	Manole	1124537859	4314902
NULL	NULL	NULL	NULL

```
SELECT editora.codigo_editora, editora.nome_editora
FROM editora;
```

RESULTADO

codigo_editora	nome_editora
1	Casa dos Espiritos
2	Editora Lê
3	Moderna
4	Objetiva
5	Manole

```
SELECT editora.nome_editora, editora.codigo_cidade
FROM editora;
```

RESULTADO

nome_editora	codigo_cidade
Casa dos Espiritos	3550308
Editora Lê	3106200
Moderna	3550308
Objetiva	3106200
Manole	4314902

Consultando de valores distintos

A cláusula DISTINCT indica para o comando que deve retornar apenas valores diferentes, ou seja, não duplicados. A sintaxe básica é:

```
SELECT DISTINCT lista de colunas
FROM tabela;
```

Consultando de valores ordenados

A cláusula ORDER BY indica para o comando que o resultado retornado deve ser ordenado de forma ascendente (ASC) ou descendente (DESC). Por padrão, quando usado, o resultado é exibido de forma ascendente. A sintaxe básica é:

```
SELECT DISTINCT lista de colunas
FROM tabela
ORDER BY coluna1,coluna2, ... ASC|DESC
```

Exemplos: Considere a seguinte tabela de editora:

codigo_editora	nome_editora	telefone	codigo_cidade
1	Casa dos Espiritos	NULL	3550308
2	Editora Lê	3145678952	3106200
3	Moderna	NULL	3550308
4	Objetiva	3199684562	3106200
5	Manole	1124537859	4314902
6	Casa dos Espiritos	456789321	45879
7	Moderna	14523698574	48759632

Listar os nomes distintos das editoras:

```
SELECT DISTINCT nome_editora
FROM editora;
```

nome_editora
Casa dos Espiritos
Editora Lê
Moderna
Objetiva
Manole

Listar os nomes distintos das editoras ordenador por ordem alfabética:

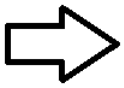
```
SELECT DISTINCT nome_editora
FROM editora
ORDER BY nome_editora;

SELECT DISTINCT nome_editora
FROM editora
ORDER BY nome_editora ASC;
```

nome_editora
Casa dos Espiritos
Editora Lê
Manole
Moderna
Objetiva

Listar os nomes distintos das editoras ordenador por ordem alfabética decrescente:

```
SELECT DISTINCT nome_editora
FROM editora
ORDER BY nome_editora DESC;
```



nome_editora
Objetiva
Moderna
Manole
Editora Lê
Casa dos Espiritos

Operadores Aritméticos e relacionais

Os operadores aritméticos são responsáveis pela execução de operações matemáticas simples. São os operadores a seguir:

+	Adição
-	Subtração
*	Multiplicação
/	Divisão

Já os operadores relacionais são utilizados quando precisamos fazer comparações entre dois valores. São os operadores a seguir:

>	Maior que
<	Menor que
=	Igual a
<>	Diferente de
>=	Maior ou igual a
<=	Menor ou igual a

Exemplos: Considere a seguinte tabela:

titulo_livro	preco
Pelas Ruas de Calcutá	36.1
Devoted - Devoção	27.2
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.7
P.s. - Eu Te Amo	23.5
O Que Esperar Quando Você Está Esperando	37.8
As Melhores Frases Em Veja	23.9
Bichos Monstruosos	24.9
Casas Mal Assombradas	27.9
Memórias Póstumas de Brás Cubas	22.5
Dom Casmurro	25.9
Dom Casmurro	35.9
Quincas Borba	35.9

```
SELECT titulo_livro, preco, preco + 10 AS adição, preco - 10 AS subtração, preco * 0.1 AS multiplicação, preco / 10 AS divisão
FROM livro;
```

titulo_livro	preco	adição	subtração	multiplicação	divisão
Pelas Ruas de Calcutá	36.1	46.099998474121094	26.099998474121094	3.6099998474121096	3.6099998474121096
Devoted - Devoção	27.2	37.20000076293945	17.200000762939453	2.7200000762939456	2.720000076293945
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.7	42.70000076293945	22.700000762939453	3.2700000762939454	3.2700000762939454
P.s. - Eu Te Amo	23.5	33.5	13.5	2.35	2.35
O Que Esperar Quando Você Está Esperando	37.8	47.79999923706055	27.799999237060547	3.779999923706055	3.779999923706055
As Melhores Frases Em Veja	23.9	33.89999961853027	13.899999618530273	2.3899999618530274	2.3899999618530274
Bichos Monstruosos	24.9	34.89999961853027	14.899999618530273	2.4899999618530275	2.4899999618530275
Casas Mal Assombradas	27.9	37.89999961853027	17.899999618530273	2.7899999618530273	2.7899999618530273
Memórias Póstumas de Brás Cubas	22.5	32.5	12.5	2.25	2.25
Dom Casmurro	25.9	35.89999961853027	15.899999618530273	2.5899999618530276	2.589999961853027
Dom Casmurro	35.9	45.900001525878906	25.900001525878906	3.5900001525878906	3.5900001525878906
Quincas Borba	35.9	45.900001525878906	25.900001525878906	3.5900001525878906	3.5900001525878906

```
SELECT titulo_livro, preco
FROM livro
WHERE preco > 30;
```

titulo_livro	preco
Pelas Ruas de Calcutá	36.1
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.7
O Que Esperar Quando Você Está Esperando	37.8
Dom Casmurro	35.9
Quincas Borba	35.9

```
SELECT titulo_livro, preco
FROM livro
WHERE preco < 30;
```

titulo_livro	preco
Devoted - Devoção	27.2
P.s. - Eu Te Amo	23.5
As Melhores Frases Em Veja	23.9
Bichos Monstruosos	24.9
Casas Mal Assombradas	27.9
Memórias Póstumas de Brás Cubas	22.5
Dom Casmurro	25.9

Filtros de seleção (condição)

É possível filtrar o resultado utilizando uma condição de seleção por meio da cláusula WHERE. A sintaxe básica é:

```
SELECT lista de colunas
FROM tabela
WHERE condição de seleção;
ou
SELECT lista de colunas
FROM tabela
WHERE NOT condição de seleção;
```

Exemplos: Considere a seguinte tabela de editora:

codigo_editora	nome_editora	telefone	codigo_cidade
1	Casa dos Espiritos	NULL	3550308
2	Editora Lê	3145678952	3106200
3	Moderna	NULL	3550308
4	Objetiva	3199684562	3106200
5	Manole	1124537859	4314902
6	Casa dos Espiritos	456789321	45879
7	Moderna	14523698574	48759632

Listar o nome da editora de código igual a 4.

```
SELECT *
FROM editora
WHERE codigo_editora = 4;
```

codigo_editora	nome_editora	telefone	codigo_cidade
4	Objetiva	3199684562	3106200

```
SELECT nome_editora
FROM editora
WHERE codigo_editora = 4;
```

nome_editora
Objetiva

Listar o nome das editoras onde o código é diferente de 4.

```
SELECT *
FROM editora
WHERE NOT codigo_editora = 4;
```

codigo_editora	nome_editora	telefone	codigo_cidade
1	Casa dos Espiritos	NULL	3550308
2	Editora Lê	3145678952	3106200
3	Moderna	NULL	3550308
5	Manole	1124537859	4314902
6	Casa dos Espiritos	456789321	45879
7	Moderna	14523698574	48759632

Listar as editoras no qual o telefone é nulo ou vazio (NULL).

```
SELECT *
FROM editora
WHERE telefone IS NULL;
```

codigo_editora	nome_editora	telefone	codigo_cidade
1	Casa dos Espiritos	NULL	3550308
3	Moderna	NULL	3550308

Listar as editoras no qual o telefone não é nulo ou vazio (NOT NULL).

```
SELECT *
FROM editora
WHERE telefone IS NOT NULL;
```

codigo_editora	nome_editora	telefone	codigo_cidade
2	Editora Lê	3145678952	3106200
4	Objetiva	3199684562	3106200
5	Manole	1124537859	4314902
6	Casa dos Espiritos	456789321	45879
7	Moderna	14523698574	48759632

Combinação de múltiplos filtros (condições)

Pode ser aplicado múltiplas condições em uma consulta, para isso, podemos usar os operadores lógicos AND e OR.

A sintaxe básica é:

```
SELECT lista de colunas
FROM tabela
WHERE condição1 AND condição2 AND condição3 ...;
ou
SELECT lista de colunas
FROM tabela
WHERE condição1 OR condição2 OR condição3 ...;
Ou
SELECT lista de colunas
FROM tabela
WHERE condição1 AND condição2 OR condição3 ...;
```

Exemplos: Considere a seguinte tabela de editora:

codigo_editora	nome_editora	telefone	codigo_cidade
1	Casa dos Espiritos	NULL	3550308
2	Editora Lê	3145678952	3106200
3	Moderna	NULL	3550308
4	Objetiva	3199684562	3106200
5	Manole	1124537859	4314902
6	Casa dos Espiritos	456789321	45879
7	Moderna	14523698574	48759632

Listar as editoras no qual o telefone não é nulo ou vazio (NOT NULL) e o código da editora é maior que 5.

```
SELECT *
FROM editora
WHERE telefone IS NOT NULL
AND codigo_editora > 5;
```

codigo_editora	nome_editora	telefone	codigo_cidade
6	Casa dos Espiritos	456789321	45879
7	Moderna	14523698574	48759632

Listar as editoras no qual o telefone não é nulo ou vazio (NOT NULL) ou o código da editora é menor que 5.

```
SELECT *
FROM editora
WHERE telefone IS NOT NULL
OR codigo_editora < 5;
```

codigo_editora	nome_editora	telefone	codigo_cidade
1	Casa dos Espiritos	NULL	3550308
2	Editora Lê	3145678952	3106200
3	Moderna	NULL	3550308
4	Objetiva	3199684562	3106200
5	Manole	1124537859	4314902
6	Casa dos Espiritos	456789321	45879
7	Moderna	14523698574	48759632

Operador BETWEEN:

Seleciona os dados dentro de um intervalo de valores, sendo estes valores incluídos nos resultados. Considere a tabela editora para os exemplos. A sintaxe básica é:

```
SELECT lista de colunas
```

```
FROM tabela
WHERE condição1 BETWEEN valor1 AND valor2;
```

Listar as editoras no qual o código da editora é maior ou igual 5 e menor ou igual 7.

```
SELECT * FROM editora
WHERE codigo_editora BETWEEN 4 AND 6;
```

codigo_editora	nome_editora	telefone	codigo_cidade
4	Objetiva	3199684562	3106200
5	Manole	1124537859	4314902
6	Casa dos Espiritos	456789321	45879

Operador LIKE:

Seleciona os dados que possuam uma determinada sequência de caracteres. Considere a tabela editora para os exemplos. A sintaxe básica é:

```
SELECT lista de colunas
FROM tabela
WHERE coluna LIKE 'caracter%'; => Inicia com o caractere
      ou
SELECT lista de colunas
FROM tabela
WHERE coluna LIKE '%caracter%'; => Tem um caractere ou string
```

Listar as editoras no qual o nome da editora inicia com a letra 'M'. (Note que para alguns SGBDs é case sensitive).

```
SELECT * FROM editora
WHERE nome_editora LIKE 'M%';
```

codigo_editora	nome_editora	telefone	codigo_cidade
3	Moderna	NULL	3550308
5	Manole	1124537859	4314902
7	Moderna	14523698574	48759632

Listar as editoras no qual o nome da editora possua a letra 'l'.

```
SELECT * FROM editora
WHERE nome_editora LIKE '%l%';
```

codigo_editora	nome_editora	telefone	codigo_cidade
2	Editora Lê	3145678952	3106200
5	Manole	1124537859	4314902

Funções de agregação

Uma função de agregação executa um cálculo em um conjunto de valores e retorna um único valor. As principais funções de agregação são:

- MIN = Valor Mínimo de um conjunto de valores;
- MAX = Valor Máximo de um conjunto de valores;
- AVG = Média Aritmética de um conjunto de valores;
- SUM = Total (Soma) de um conjunto de valores;
- COUNT = Quantidade de dados de uma tabela.

A sintaxe básica é:

Mínimo

```
SELECT MIN(column_name)
FROM table_name
WHERE condition;
```

Máximo

```
SELECT MAX(column_name)
FROM table_name
WHERE condition;
```

Média

```
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

Somatório

```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

Contagem/quantidade

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

Exemplos: Considere a seguinte tabela:

titulo_livro	preco
Pelas Ruas de Calcutá	36.1
Devoted - Devoção	27.2
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.7
P.s. - Eu Te Amo	23.5
O Que Esperar Quando Você Está Esperando	37.8
As Melhores Frases Em Veja	23.9
Bichos Monstruosos	24.9
Casas Mal Assombradas	27.9
Memórias Póstumas de Brás Cubas	22.5
Dom Casmurro	25.9
Dom Casmurro	35.9
Quincas Borba	35.9

```
SELECT min(preco), max(preco), avg(preco), sum(preco), count(preco)
FROM livro;
```

min(preco)	max(preco)	avg(preco)	sum(preco)	count(preco)
22.5	37.8	29.516666730244953	354.20000076293945	12

Funções de agregação: Agrupamento e Condição

Podemos usar as cláusulas GROUP BY e HAVING combinadas às funções de agrupamento para, respectivamente, agrupa linhas que possuem os mesmos valores em linhas de resumo e aplicar condições às funções. A sintaxe básica é:

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s);
```

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition;
```

Exemplos: Considere a seguinte tabela:

titulo_livro	preco
Pelas Ruas de Calcutá	36.1
Devoted - Devoção	27.2
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.7
P.s. - Eu Te Amo	23.5
O Que Esperar Quando Você Está Esperando	37.8
As Melhores Frases Em Veja	23.9
Bichos Monstruosos	24.9
Casas Mal Assombradas	27.9
Memórias Póstumas de Brás Cubas	22.5
Dom Casmurro	25.9
Dom Casmurro	35.9
Quincas Borba	35.9

Listar o nome do livro e seu menor preço. Listar o nome do livro e seu maior preço.

```
SELECT titulo_livro, min(preco)
FROM livro
GROUP BY titulo_livro;
```

titulo_livro	min(preco)
Pelas Ruas de Calcutá	36.1
Devoted - Devoção	27.2
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.7
P.s. - Eu Te Amo	23.5
O Que Esperar Quando Você Está Esperando	37.8
As Melhores Frases Em Veja	23.9
Bichos Monstruosos	24.9
Casas Mal Assombradas	27.9
Memórias Póstumas de Brás Cubas	22.5
Dom Casmurro	25.9
Quincas Borba	35.9

```
SELECT titulo_livro, max(preco)
FROM livro
GROUP BY titulo_livro;
```

titulo_livro	max(preco)
Pelas Ruas de Calcutá	36.1
Devoted - Devoção	27.2
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.7
P.s. - Eu Te Amo	23.5
O Que Esperar Quando Você Está Esperando	37.8
As Melhores Frases Em Veja	23.9
Bichos Monstruosos	24.9
Casas Mal Assombradas	27.9
Memórias Póstumas de Brás Cubas	22.5
Dom Casmurro	35.9
Quincas Borba	35.9

Listar o nome do livro e a média de preço.

```
SELECT titulo_livro, avg(preco)
FROM livro
GROUP BY titulo_livro;
```

titulo_livro	avg(preco)
Pelas Ruas de Calcutá	36.099998474121094
Devoted - Devoção	27.200000762939453
Xô, Bactéria! Tire Suas Dúvidas Com Dr. Bactéria	32.70000076293945
P.s. - Eu Te Amo	23.5
O Que Esperar Quando Você Está Esperando	37.79999923706055
As Melhores Frases Em Veja	23.899999618530273
Bichos Monstruosos	24.899999618530273
Casas Mal Assombradas	27.899999618530273
Memórias Póstumas de Brás Cubas	22.5
Dom Casmurro	30.90000057220459
Quincas Borba	35.900001525878906

Listar o nome do livro e a média de preço onde a média de preço for menor que 30.

```
SELECT titulo_livro, avg(preco)
FROM livro
GROUP BY titulo_livro
HAVING avg(preco) < 30;
```

titulo_livro	avg(preco)
Devoted - Devoção	27.200000762939453
P.s. - Eu Te Amo	23.5
As Melhores Frases Em Veja	23.899999618530273
Bichos Monstruosos	24.899999618530273
Casas Mal Assombradas	27.899999618530273
Memórias Póstumas de Brás Cubas	22.5

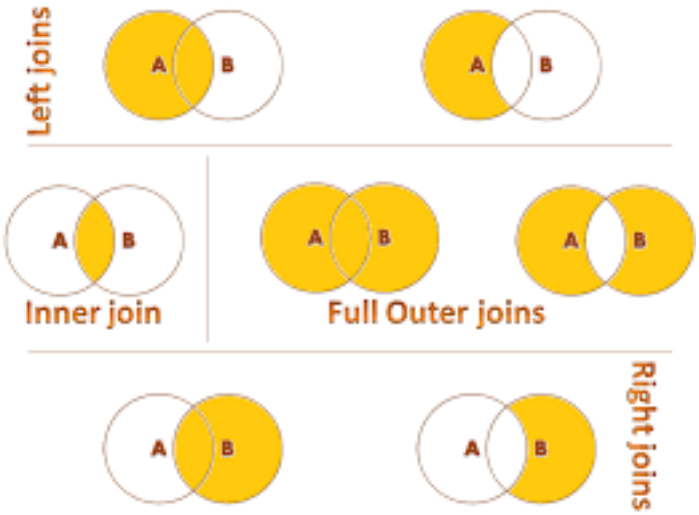
Para consultar múltiplas tabelas é necessário passar uma condição de junção entre as tabelas. Em geral, esta junção se dá pelo relacionamento existente entre as tabelas, ou seja, pela(s) colunas comuns às duas ou mais tabelas. Todas as cláusulas ensinadas até aqui podem também ser combinadas às consultas por múltiplas tabelas.

Existem diferentes tipos de condições de junção ou JOINs no SQL[12], ilustradas na Figura 27:

- (INNER) JOIN: Retorna registros que possuem valores correspondentes nas duas tabelas.
- LEFT (OUTER) JOIN: Retorna todos os registros da tabela da esquerda e registros correspondentes da tabela da direita.
- RIGHT (OUTER) JOIN: Retorna todos os registros da tabela da direita e os registros correspondentes da tabela da esquerda.
- FULL (OUTER) JOIN: Retorna todos os registros quando houver uma correspondência na tabela esquerda ou direita.

[12] Pratique em: https://www.w3schools.com/sql/sql_join.asp

Figura 27: Tipos de junções SQL.



A sintaxe básica é para consultas em múltiplas tabelas é:

```
SELECT lista de colunas
FROM tabela1 [INNER | LEFT | RIGHT | FULL ] JOIN
tabela2
ON tabela1.coluna = tabela2.coluna;
```

Exemplos: Considere as tabelas a seguir:

DEPARTAMENTO

cod_depto	nome_depto
1	Recursos humanos
2	Informática
3	Marketing
4	Infraestrutura
5	Jurídico
6	Compras

FUNCIONARIO

id_func	nome_func	nascimento	salario	depto
1	José Silva	1979-03-12	5450.00	NULL
2	Maria Souza	1982-05-28	5600.00	3
3	João Nogueira	1975-04-15	6500.00	4
4	Paulo Silva Amorim	1985-10-19	5500.00	1
5	Ana Paula Almeida	1979-11-25	6500.00	4
6	Silvana Abdala	1979-05-04	5450.00	2
7	Jair Nogueira	1989-11-15	5300.00	NULL
8	Almir Nunes	1990-01-01	5450.00	2

INNER JOIN

```
SELECT *
FROM departamento, funcionario
WHERE cod_depto = depto;
```

```
SELECT *
FROM departamento INNER JOIN funcionario
ON cod_depto = depto;
```

cod_depto	nome_depto	id_func	nome_func	nascimento	salario	depto
3	Marketing	2	Maria Souza	1982-05-28	5600.00	3
4	Infraestrutura	3	João Nogueira	1975-04-15	6500.00	4
1	Recursos humanos	4	Paulo Silva Amorim	1985-10-19	5500.00	1
4	Infraestrutura	5	Ana Paula Almeida	1979-11-25	6500.00	4
2	Informática	6	Silvana Abdala	1979-05-04	5450.00	2
2	Informática	8	Almir Nunes	1990-01-01	5450.00	2

LEFT (OUTER) JOIN

```
SELECT *
FROM departamento LEFT JOIN funcionario
ON cod_depto = depto;
```

cod_depto	nome_depto	id_func	nome_func	nascimento	salario	depto
1	Recursos humanos	4	Paulo Silva Amorim	1985-10-19	5500.00	1
2	Informática	6	Silvana Abdala	1979-05-04	5450.00	2
2	Informática	8	Almir Nunes	1990-01-01	5450.00	2
3	Marketing	2	Maria Souza	1982-05-28	5600.00	3
4	Infraestrutura	3	João Nogueira	1975-04-15	6500.00	4
4	Infraestrutura	5	Ana Paula Almeida	1979-11-25	6500.00	4
5	Jurídico	NULL	NULL	NULL	NULL	NULL
6	Compras	NULL	NULL	NULL	NULL	NULL

RIGHT (OUTER) JOIN

```
SELECT *
FROM departamento RIGHT JOIN funcionario
ON cod_depto = depto;
```

cod_depto	nome_depto	id_func	nome_func	nascimento	salario	depto
NULL	NULL	1	José Silva	1979-03-12	5450.00	NULL
3	Marketing	2	Maria Souza	1982-05-28	5600.00	3
4	Infraestrutura	3	João Nogueira	1975-04-15	6500.00	4
1	Recursos humanos	4	Paulo Silva Amorim	1985-10-19	5500.00	1
4	Infraestrutura	5	Ana Paula Almeida	1979-11-25	6500.00	4
2	Informática	6	Silvana Abdala	1979-05-04	5450.00	2
NULL	NULL	7	Jair Nogueira	1989-11-15	5300.00	NULL
2	Informática	8	Almir Nunes	1990-01-01	5450.00	2

FULL (OUTER) JOIN

```
SELECT *
FROM departamento FULL JOIN funcionario
ON cod_depto = depto;
```

cod_depto	nome_depto	id_func	nome_func	nascimento	salario	depto
NULL	NULL	1	José Silva	1979-03-12	5450.00	NULL
3	Marketing	2	Maria Souza	1982-05-28	5600.00	3
4	Infraestrutura	3	João Nogueira	1975-04-15	6500.00	4
1	Recursos humanos	4	Paulo Silva Amorim	1985-10-19	5500.00	1
4	Infraestrutura	5	Ana Paula Almeida	1979-11-25	6500.00	4
2	Informática	6	Silvana Abdala	1979-05-04	5450.00	2
NULL	NULL	7	Jair Nogueira	1989-11-15	5300.00	NULL
2	Informática	8	Almir Nunes	1990-01-01	5450.00	2
5	Jurídico	NULL	NULL	NULL	NULL	NULL
6	Compras	NULL	NULL	NULL	NULL	NULL

Capítulo 4. Data warehouse e modelagem dimensional

Um armazém de dados ou data warehouse (DW)[13] é um esquema de banco de dados organizado em estruturas lógicas dimensionais, construído para atender sistemas de apoio à tomada de decisão, e possibilitando o seu processamento analítico por sistemas como os *Online Analytical Processing* (OLAP) e *Data Mining*. A definição original é dada por William H. Inmon (Bill Inmon) ao caracterizar os DW como um conjunto de dados baseado em assuntos, integrado, não volátil e variável em relação ao tempo, de apoio às decisões gerenciais (BARBIERI, 2001, 2011; DAMA, 2009, 2017).

- ***Orientado a assuntos:*** significa que os dados do DW dão informações sobre um assunto particular em vez de sobre operações contínuas da companhia. Por exemplo, um DW que lida com vendas de produtos a diferentes tipos de clientes, ou atendimentos e diagnósticos de pacientes.

- ***Integrado:*** os dados que são reunidos no DW partem de uma variedade de origens e assim podem apresentar diferentes nomenclaturas, formatos e estruturas das fontes de dados, estes dados precisam ser transformados em um único esquema ou formato para prover uma visão unificada e consistente da informação.

- ***Não volátil:*** os dados de um DW são estáveis, ou seja, não são modificados como em sistemas transacionais (exceto para correções), mas somente carregados e acessados para leituras, e em regra geral nunca removidos. Isso capacita à análise de séries históricas dos negócios.

[13] Os scripts e modelos das videoaulas se encontram disponíveis em: https://github.com/FernandaFarinelli/Disciplinas_IGTI/tree/main/EngenhariaDeDados/Modulo1/ScriptsVideoaulas/Capitulo5

- ***Variável em relação ao tempo:*** todos os dados no DW são identificados com um período de tempo particular, que é referente ao fato em análise.

Outro conceito relevante para entendermos sobre DW é o conceito de *data mart* ou "mercado de dados". Segundo Inmon, um *data mart* corresponde às necessidades de informações de uma determinada comunidade de usuários. Ou seja, *data mart* se refere a um subconjunto do DW que contém dados sobre um setor específico da empresa (departamento, direção, serviço, gama de produto etc.), um assunto específico (compras, vendas, estoque, contábil etc.), ou diferentes níveis de sumarização (Vendas Anual, Vendas Mensal).

Modelo dimensional de dados

A modelagem dimensional é uma técnica de modelagem, focada na elaboração do modelo lógico usado para projetos de DW e data marts. No modelo multidimensional, ou dimensional como às vezes é chamado, o foco não é a coleta dos dados e a normalização das estruturas de dados, mas sim a consulta aos dados. Esta é uma das grandes diferenças dos modelos dimensionais para os modelos relacionais. Ou seja, para melhorar o desempenho, há redundância planejada dos dados, compensando os gastos com armazenamento e atualização das informações. A redundância de dados aqui se torna uma vantagem competitiva (BARBIERI, 2001, 2011). O modelo dimensional permite visualizar dados abstratos de forma simples e relacionar informações de diferentes setores da empresa. Esta abordagem lida basicamente com três elementos: fatos, dimensões e métricas.

Os **fatos** são os assuntos que serão analisados, ou seja, acontecimentos do negócio que são merecedores de análise e controle na organização. A tabela de fatos é a principal tabela de um modelo dimensional, onde as métricas estão armazenadas. É composta por uma chave primária (formada por uma combinação única de valores de

chaves de dimensão) e pelas métricas de interesse para o negócio. A tabela de fatos deve representar uma unidade do processo do negócio, e não devem misturar assuntos diferentes numa mesma tabela.

As **dimensões** são entidades do negócio que apresentam alguma influência sobre o fato em análise, assim, são variáveis de análise de um acontecimento. Estes dois elementos são representados no modelo como tabelas. A tabela de dimensão é composta de atributos e contém a descrição do negócio. Seus atributos são fontes das restrições de consultas, agrupamento dos resultados e cabeçalhos para relatórios. Ela possui aspectos pelos quais se pretende observar as métricas relativas ao processo modelado. Imagine que está analisando uma compra (o fato) e será analisado conforme as dimensões Produto (o que foi comprado?), Filial (onde foi comprado?), Cliente (quem comprou?) e Tempo (quando comprou?).

As **métricas** são as medidas que são analisadas em um assunto de acordo com as variáveis de análise e são representadas no modelo como atributos da tabela fato. As métricas podem ser de diferentes tipos:

- Aditiva: permitem operações matemáticas como adição, subtração e média de valores independente das suas dimensões de análise. Por exemplo, a quantidade e valores de produtos. Podem ser sumarizados por data (dia, mês ou ano), local, clientes sem perder o sentido da informação.

- Semiaditiva: pode ser somada ou sumarizadas por parte das suas dimensões de análise. Por exemplo, saldo de conta bancária ou saldo de itens em estoque. O saldo é um valor que reflete a situação atual da conta, que pode ter o saldo credor ou devedor e pode variar por dia ao longo do mês. Assim, não faz sentido somar os saldos de todos os dias de um mês para uma determinada conta bancária.

- Não aditiva: não podem ser somadas e não podem ter agregações, pois perdem a veracidade do valor. Por exemplo, valores percentuais. Não faz sentido algum somar o percentual de vendas de um produto com o outro.

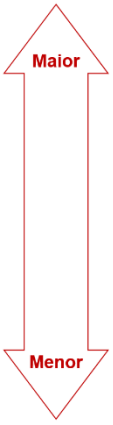
Outro fator importante a ser observado no modelo dimensional refere-se à granularidade do fato a ser analisado, ou seja, o nível de detalhe ou de resumo contido nas unidades de dados existentes no fato em análise. Um grão corresponde a um registro na tabela fato, e o detalhamento do fato afeta diretamente o volume de dados armazenado e consequentemente o tempo de resposta de uma consulta. Quanto mais detalhe existir, mais baixo será o nível de granularidade, consequentemente, quanto menos detalhe existir, mais alto será o nível de granularidade (BARBIERI, 2001, 2011).

Para exemplificar a ideia de granularidade, considere o exemplo da Figura , podemos analisar a criminalidade por tipo anualmente ou a cada semestre. Quando avaliamos por ano estamos avaliando os crimes com menos detalhes, ou seja, maior granularidade.

Figura 28 – Exemplo de granularidade.

Ocorrência	2017	2018	2019
Assaltos à mão armada	123	109	158
Furtos em residências	75	90	101
Furtos de veículos	243	250	332
Assassinatos	89	77	167
Estupros	2	24	69

Ocorrência	1º sem 2017	2º sem 2017	1º sem 2018	2º sem 2018	1º sem 2019	2º sem 2019
Assaltos à mão armada	50	73	45	64	70	88
Furtos em residências	40	35	60	30	71	30
Furtos de veículos	121	123	120	130	165	167
Assassinatos	40	49	37	40	67	100
Estupros	2	0	14	10	30	39



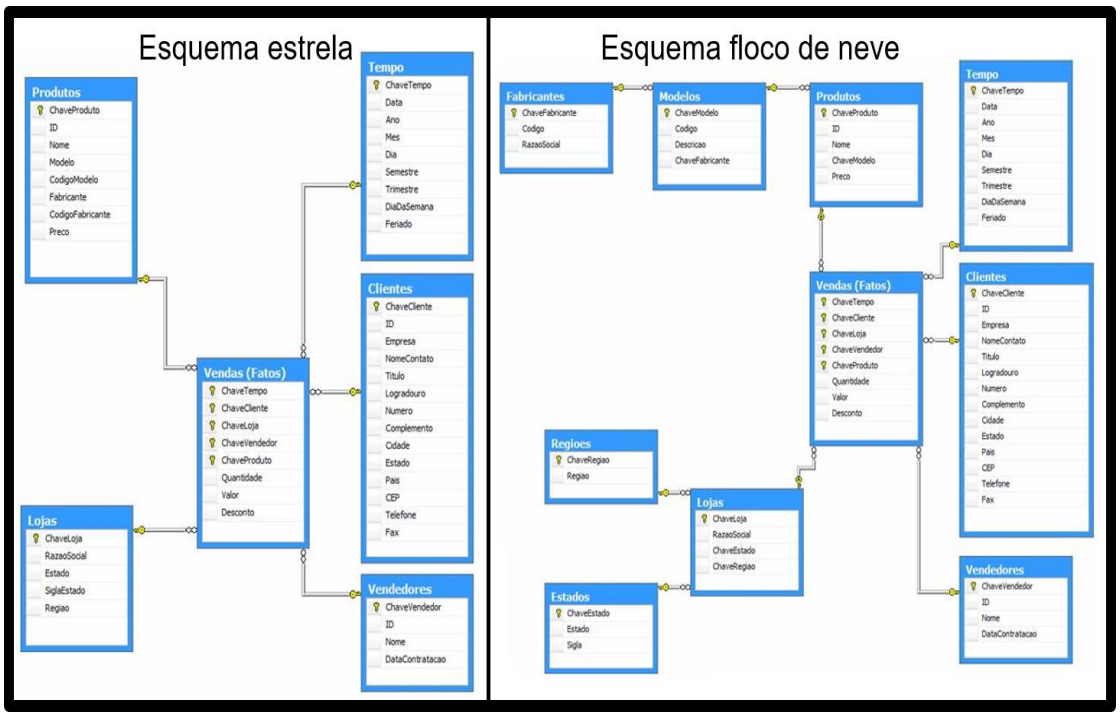
Além disso, no modelo multidimensional, os Fatos e dimensões podem ser dispostos segundo diferentes configurações: Esquema Estrela (*Star Schema*), o esquema

original, onde as dimensões não são normalizadas. Ou Esquema Floco de Neve, (*Snow flake*), onde as dimensões são estruturadas de forma normalizadas, ou decompostas de ou mais dimensões que possuem hierarquias (BARBIERI, 2001, 2011).

Por exemplo, conforme Figura 29, a dimensão *lojas* no esquema estrela está desnormalizada, enquanto no modelo floco de neve ela foi normalizada dando origem às dimensões *estados* e *regiões*. O mesmo ocorre com a dimensão *produtos,* que após ser normalizada deu origem às dimensões *fabricantes* e *modelos*.

No esquema estrela a navegação pelas tabelas é mais simples, fácil e rápida, porém desperdiça espaço repetindo as mesmas descrições ao longo de toda a estrutura. O Esquema Floco de Neve reduz a redundância, mas aumentam a complexidade do esquema e consequentemente a compreensão por parte dos usuários. Dificultam as implementações de ferramentas de visualização dos dados. A decisão pela adoção destes dois esquemas deve considerar o tempo de resposta com o uso ou não da normalização e espaço em disco, pensando também em estratégias como uso de fatos agregados e alteração na granularidade dos dados.

Figura 29 – Exemplo de esquema estrela versus floco de neve[14].



Para maiores detalhes sobre DW e modelagem dimensional, sugere-se a leitura do material disponível no Blog: https://rafaelpiton.com.br/blog/.

Extração, Transformação e Carga (ETC)

O processo conhecido como Extração, Transformação e Carga (ETC), mais conhecidos como ETL, sigla originária do inglês *Extract, Transform and Load*, tem sua origem com os projetos de DW, e envolve a coleta, a padronização e a consolidação de dados. ETC se referem às múltiplas rotinas construídas e executadas com a finalidade de sistematizar a coleta ou extração dos dados originados dos diversos sistemas

[14] Modelos retirados de:
https://msdn.microsoft.com/pt-br/library/cc518031.aspx#XSLTsection126121120120

organizacionais, o tratamento (limpeza, transformação, integração, padronização) dos dados e a carga destes dados nas estruturas analíticas (DW).

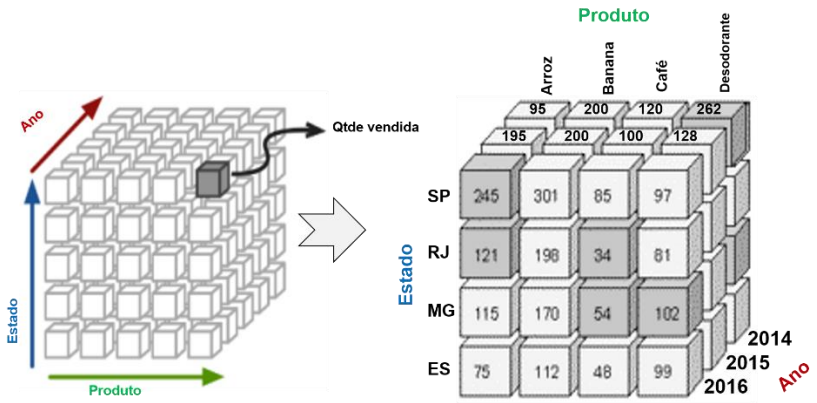
Online Analytical Processing

Em processos de analises de dados é comum a necessidade de agrupar, agregar e juntar dados. Essas operações em bancos de dados relacionais consomem muitos recursos. Para otimizar estes cruzamentos de dados surgiu o *OnLine Analytical Processing* (OLAP), processamento online analítico, que se refere a um conjunto de técnicas de análise de dados desenvolvidas para analisar dados em data warehouses (BARBIERI, 2001, 2011).

O processo de data warehousing é o processo de extrair dados de sistemas transacionais e transformá-los em informação organizada em um formato amigável. O conceito central do OLAP é a ideia de um cubo de dados. Baseia-se na ideia de analisar um acontecimento por múltiplas dimensões que surge a ideia de modelo multidimensional, que usa a metáfora do cubo (uma figura de três dimensões) para falar do conjunto de dados que será coletado (BARBIERI, 2001, 2011), conforme ilustrado na Figura 30 a seguir.

Por exemplo, em um data mart de vendas, algumas das dimensões relevantes são o momento da venda (dia, mês e ano), local da venda, produto e vendedor. Na Figura 30 consideramos apenas as três primeiras dimensões. Cada ponto em um cubo de dados armazena uma métrica consolidada dos valores de quantidade vendida. Em geral, essas dimensões são hierárquicas; o momento da venda pode ser organizado como uma hierarquia dia-mês-trimestre-ano, ou local poderia ser a loja, cidade, estado.

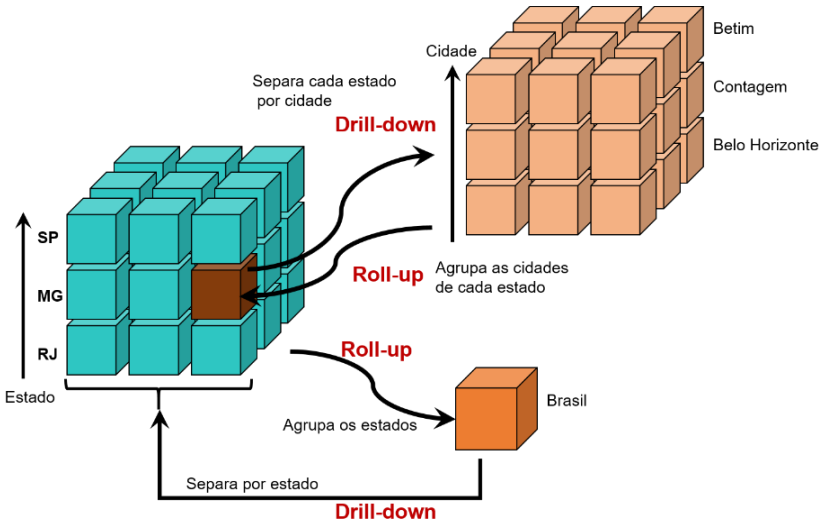
Figura 30 – Exemplo de cubo.



Na figura acima observamos que no ano de 2016, a Banana foi o produto mais vendido no estado de São Paulo, seguido pelo Arroz. Além disso, quando avaliamos a quantidade vendida de cada produto no estado de São Paulo em cada ano, observamos que os produtos Café e Desodorante tiveram uma queda nas vendas.

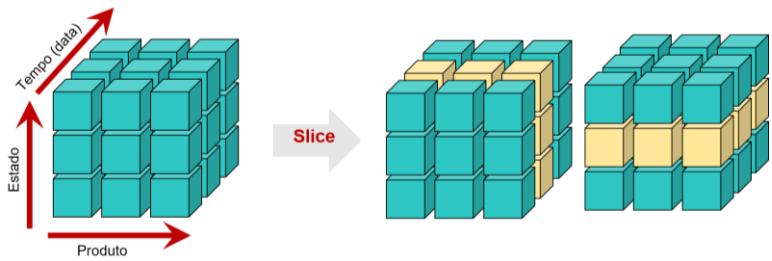
As operações OLAP típicas incluem drill down, roll-up, pivot e slice and dice, que são as formas de interagir com o cubo de dados para análise de dados multidimensionais. A operação roll-up também é conhecida como “consolidação” ou “agregação”, na operação drill-down é o oposto do processo de roll-up, ou seja, os dados são fragmentados em partes menores. Estas operações podem ser realizadas ao longo de cada dimensão e você pode “subir ou descer” dentro do detalhamento do dado, como, por exemplo, analisar uma informação tanto diariamente quanto anualmente, partindo da mesma base de dados. Ou seja, você pode navegar no cubo por diferentes níveis de granularidade. A operação de roll-up (Figura 31) não é limitado pelo grão máximo e os dados podem ser agregados mesmo após se chegar a este limite superior. Já na operação de drill-down (Figura 31) é limitado pelo grão mínimo e navega no sentido de explorar maior granularidade do cubo.

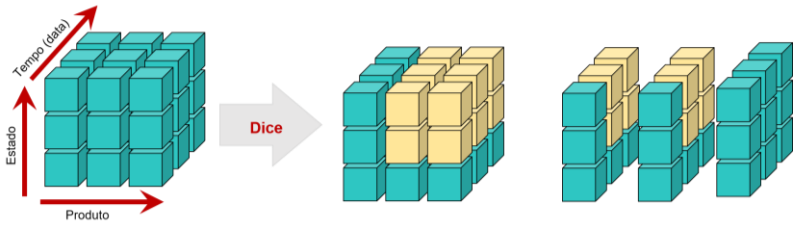
Figura 31 – Operações OLAP: *roll-up* e *drill-down*.



Nas operações slice and dice (Figura 32), também conhecidas como seleção e projeção, a ideia é gerar um sub-cubo limitando a análise a uma parte dos dados. No caso do slice, a análise é de apenas uma fatia, ou seja, restringe os valores de uma dimensão aplicando um filtro de seleção de dados, mas não diminui a cardinalidade do cubo. Já na operação dice, ocorre a redução das dimensões ou a cardinalidade de um cubo por meio da eliminação ou filtro em uma ou mais dimensões. Em suma, estas operações consistem na rotação do cubo, possibilitando a combinação de quaisquer dimensões.

Figura 32 – Operações OLAP: *slice*.





Na operação pivot, você gira os eixos de dados para fornecer uma apresentação de dados substituta, oferecendo assim diferentes perspectivas sobre o mesmo conjunto de dados. Por exemplo, ao invés de ver os dados na dimensão Produto x Estado é possível trocar para Estado x Produto.

Os dados de um DW promoverem a capacidade de analisar dados estruturados, padronizados e centralizados, fomentando os processos de gerência e tomada de decisão. Entretanto, o Big Data trouxe novos desafios que os DW's não conseguem atender em sua plenitude, principalmente pela diversidade de estruturas e formatos de dados, para suprir tal demanda, surge o *Data lake*.

Referências

ALMEIDA, M. B. d. Revisiting ontologies: A necessary clarification. In: *Journal of the American Society for Information Science and Technology*, v. 64, n. 8, p. 1682-1693, 2013. Disponível em: <http://onlinelibrary.wiley.com/doi/10.1002/asi.22861/pdf>. Acesso em: 25 jan. 2022 .

AMARAL, F. *Introdução à Ciência de Dados: mineração de dados e big data*. Alta Books Editora, 2016.

AZEVEDO, A. I. R. L.; SANTOS, M. F. *KDD, SEMMA and CRISP-DM*: a parallel overview. IADS-DM, 2008.

BARBIERI, C. P. *BI2 - Business Intelligence: Modelagem e Qualidade*. 1. ed. Rio de Janeiro: Elsevier, 2011. p. 416.

BARBIERI, C. P. *BI-business intelligence: modelagem e tecnologia*. Axcel Books, 2001.

BDW, B. D. W. *Introduction about NoSQL Data Models*. Big Data World. online. 2019 2014.

BENEVENUTO, F.; ALMEIDA, J. M.; SILVA, A. S., 2011, Campo Grande, Brasil. *Explorando redes sociais online*: Da coleta e análise de grandes bases de dados às aplicações. Sociedade Brasileira de Computação, 2011. p. 63-102.

BERNERS-LEE, T. Linked Data. *w3*, 2006. Disponível em: <https://www.w3.org/DesignIssues/LinkedData.html>. Acesso em: 25 jan. 2022.

BERNERS-LEE, T. *Semantic web road map*. 1998.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The semantic web. In: *Scientific american*, v. 284, n. 5, p. 28-37, 26 abr. 2001.

BIZER, C.; HEATH, T.; BERNERS-LEE, T. Linked data: Principles and state of the art. In: *World Wide Web Conference*, 2008.

BIZER, C.; HEATH, T.; BERNERS-LEE, T. Linked data-the story so far. In: *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, p. 205-227, 2009.

CHEN, P. P. *Modelagem de dados*: a abordagem entidade-relacionamento para projeto lógico. Makron Books do Brasil, 1990.

CHEN, P. P. The entity-relational model toward a unified view of data. In: *ACM Trans, on Database Systems*, 1, n. 1, p. 1-49, 1976.

COUGO, P. S. *Modelagem conceitual e projeto de banco de dados*. 1 ed. 18ª Reimp. ed. Rio de Janeiro: Elsevier, 1997.

COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T.; BLAIR, G. *Sistemas Distribuídos-: Conceitos e Projeto*. Bookman Editora, 2013.

COX, M.; ELLSWORTH, D. Application-controlled demand paging for out-of-core visualization. In: *IEEE Computer Society Press*. English, Phoenix, AZ, USA, 235-ff, 1997. Disponível em: <https://www.nas.nasa.gov/assets/pdf/techreports/1997/nas-97-010.pdf>. Acesso em: 25 jan. 2022.

CURRY, E. The Big Data Value Chain: Definitions, Concepts, and Theoretical Approaches. *In*: CAVANILLAS, J. M.; CURRY, E. e WAHLSTER, W. (Ed.). *New Horizons for a Data-Driven Economy: A Roadmap for Usage and Exploitation of Big Data in Europe*. Springer, 2016. p. 29-37. Disponível em: <https://link.springer.com/content/pdf/10.1007%2F978-3-319-21569-3_3.pdf>. Acesso em: 25 jan. 2022.

DAMA. *DAMA-DMBOK: Data management body of knowledge.* 2 ed. Bas King Ridge, New Jersey, USA: Technics Publications LLC, 2017. p. 624.

DAMA. *The DAMA Guide to The Data Management Body of Knowledge (DAMA-DMBOK Guide)*. 1 ed. Bas King Ridge, New Jersey, USA: Technics Publications LLC, 2009. p. 406.

DATA SCIENCE GUIDE, D. *Exploratory data analysis*. Data science guide. online. 2021.

DAVENPORT, T. H. *Big data no trabalho: derrubando mitos e descobrindo oportunidades*. 1ª 'edição'. Rio de Janeiro: Elsevier, 2014. p. 232.

DAVENPORT, T. H. *Ecologia da informação: por que só a tecnologia não basta para o sucesso na era da informação*. Tradução ABRÃO, B. S. 2 ed. São Paulo: Futura, 1998. p. 292.

DAVENPORT, T. H.; HARRIS, J. G. *Competição analítica: vencendo através da nova ciência*. Alta Books, 2020.

ELMASRI, R.; NAVATHE, S. B. *Sistemas de banco de dados*. 4 ed. São Paulo: Addison Wesley, 2005.

EMC oferece solução de armazenamento e análise de Data Lake. *Canaltech*, 03 abr. 2015. Disponível em: <https://canaltech.com.br/infra/EMC-oferece-solucao-de-armazenamento-e-analise-de-Data-Lake/>. Acesso em: 25 jan. 2022.

FARINELLI, F. *Improving semantic interoperability in the obstetric and neonatal domain through an approach based on ontological realism*. Orientador: ALMEIDA, M. B. d. 2017. 256 f. Doctoral (Doctor in Information Science) - School of Information Science Federal University of Minas Gerais at Brazil, Belo Horizonte. Disponível em: <http://www.bibliotecadigital.ufmg.br/dspace/handle/1843/BUBD-AX2J5B>. Acesso em: 25 jan. 2022.

FARINELLI, F.; SOUZA, A. D. d. Ontologias de alto nível: porque precisamos e como usar. In: *Fronteiras da Representação do Conhecimento*, v. 1, n. 1, p. 174-202, 2021.

FERRI, E. Análise Preditiva ou Prescritiva? Sua empresa precisa de ambos. *Medium*, 2019. Disponível em: <https://medium.com/@edselferri>. Acesso em: 25 jn. 2022.

FOWLER, M. Nosql Definition. *martinFowler.com*, 2016. Disponível em: <https://martinfowler.com/bliki/NosqlDefinition.html>. Acesso em: 25 jan. 2022.

GUIA de Dados Abertos. *Open Data Handbook*, 2019. Disponível em: <http://opendatahandbook.org/guide/pt_BR/>. Acesso em: 25 jan. 2022.

HEATH, T.; BIZER, C. Linked data: Evolving the web into a global data space. In: *Synthesis lectures on the semantic web: theory and technology*, v. 1, n. 1, p. 1-136, 2011.

HECKERT, N. A.; FILLIBEN, J. J. *NIST/SEMATECH e-Handbook of Statistical Methods*; Chapter 1: Exploratory Data Analysis. 2003.

HEUSER, C. A. *Projeto de Banco de Dados.* 6. ed. Porto Alegre: Bookman, 2008. p. 282.

ISOTANI, S.; BITTENCOURT, I. *Dados Abertos Conectados: Em busca da Web do Conhecimento*. Novatec Editora, 2015. p. 175.

KAUSAR, M. A.; DHAKA, V. S.; SINGH, S. K. Web crawler: a review. In: *International Journal of Computer Applications*, 63, n. 2, 2013.

LANEY, D. 3D data management: Controlling data volume, velocity and variety. In: *META Group Research Note*, v. 6, p. 70-73, 2001.

MAYER-SCHÖNBERGER, V.; CUKIER, K. *Big data: a revolution that will transform how we live, work, and think*. Boston: Houghton Mifflin Harcourt, 2013. p. 242.

MEDRI, W. *Análise exploratória de dados*. Apostila do curso de especialização em Estatística. Londrina: Universidade Estadual de Londrina 2011.

MITCHELL, R. *Web scraping with Python*: Collecting more data from the modern web. O'Reilly, 2018.

MUNZERT, S.; RUBBA, C.; MEIßNER, P.; NYHUIS, D. *Automated data collection with R: A practical guide to web scraping and text mining*. John Wiley & Sons, 2014.

NASCIMENTO, J. P. B. A carreira dos profissionais de Ciência de Dados, Engenharia de Dados e Machine Learning. *IGTI*, 2017. Disponível em: <http://igti.com.br/blog/carreira-big-data-engenharia-dados-machine-learning/>. Acesso em: 11 mai. 2021.

PARUCHURI, V. What is Data Engineering?. *Dataquest*, 2017. Disponível em: <https://www.dataquest.io/blog/what-is-a-data-engineer/>. Acesso em: 25 jan. 2022.

REHMAN, M. H. u.; CHANG, V.; BATOOL, A.; WAH, T. Y. Big data reduction framework for value creation in sustainable enterprises. In: *International Journal of Information Management*, 36, n. 6, Part A, p. 917-928, 2016.

SADALAGE, P. J.; FOWLER, M. *NoSQL distilled*: a brief guide to the emerging world of polyglot persistence. Pearson Education, 2013.

SCIME, A. *Web Mining*: applications and techniques. IGI Global, 2005.

SHAFIQUE, U.; QAISER, H. A comparative study of data mining process models (KDD, CRISP-DM and SEMMA). In: *International Journal of Innovation and Scientific Research*, 12, n. 1, p. 217-222, 2014.

SHIVALINGAIAH, D.; NAIK, U. Comparative Study of Web 1.0, Web 2.0 and Web 3.0. In: *6th International CALIBER*, 2008, Allahabad, Índia. p. 499-507.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. *Sistemas de banco de dados*. 6. ed. São Paulo: Elsevier, 2012.

SILVA, Edilberto Magalhães. *Descoberta de conhecimento com o uso de text mining*: cruzando o abismo de moore. 2002. 175 f. Dissertação (Mestrado em Informática) - Universidade Católica de Brasília, Brasília, 2002. Disponível em: <https://bdtd.ucb.br:8443/jspui/handle/123456789/1462>. Acesso em: 25 jan. 2022.

SOUSA, F. R.; MOREIRA, L. O.; MACÊDO, J. A. F. d.; MACHADO, J. C. Gerenciamento de dados em nuvem: Conceitos, sistemas e desafios. In: *Topicos em sistemas colaborativos, interativos, multimidia, web e bancos de dados, Sociedade Brasileira de Computacao*, p. 101-130, 2010.

SOUSA, F. R.; MOREIRA, L. O.; MACHADO, J. C. Computação em nuvem: Conceitos, tecnologias, aplicações e desafios. In: *II Escola Regional de Computação Ceará, Maranhão e Piauí (ERCEMAPI)*, p. 150-175, 2009.

SQL Tutorial. *W3 School*, online, 2022. Disponível em: <https://www.w3schools.com/sql/>. Acesso em: 25 jan. 2022.

STROHBACH, M.; DAUBERT, J.; RAVKIN, H.; LISCHKA, M. Big Data Storage. *In*: CAVANILLAS, J. M.;CURRY, E. e WAHLSTER, W. (Ed.). *New Horizons for a Data-Driven Economy: A Roadmap for Usage and Exploitation of Big Data in Europe*. Cham: Springer International Publishing, 2016. p. 119-141.

TAMIR, M.; MILLER, S.; GAGLIARDI, A. *The Data Engineer*. 2015.

TAN, A.-H., 1999, *Text mining: The state of the art and the challenges*. sn. 65-70. Disponível em: <http://www.ntu.edu.sg/home/asahtan/papers/tm_pakdd99.pdf>. Acesso em: 11 mai. 2021.

TAURION, C. *Big data*. Rio de Janeiro: Brasport, 2013.

TAURION, C. *Cloud computing-computação em nuvem*. Brasport, 2009.

THELWALL, M. A web crawler design for data mining. In: *Journal of Information Science*, 27, n. 5, p. 319-325, 2001.

TUKEY, J. W. *Exploratory data analysis*. Reading, Mass., 1977.

VANDEN BROUCKE, S.; BAESENS, B. From web scraping to web crawling. In: *Practical Web Scraping for Data Science*: Springer, 2018. p. 155-172.

WHITE, T. *Hadoop: The definitive guide*. O'Reilly Media, 2012.