

Coleta e Obtenção de Dados

Leandro Lessa

2021

Coleta e Obtenção de Dados

Bootcamp Arquiteto(a) de Big Data

Leandro Lessa

© Copyright do Instituto de Gestão e Tecnologia da Informação.

Todos os direitos reservados.

Sumário

Capítulo 1. Introdução a coleta de dados	5
1.1. A crescente evolução dos dados	5
1.2. Conceito de dados	6
1.3. Conceito de Informação	8
1.4. Conceito de conhecimento.....	8
1.5. Características dos tipos de dados	8
1.5.1. <i>Dados estruturados</i>	9
1.5.2. <i>Dados não estruturados</i>	9
1.5.3. <i>Dados semiestruturados</i>	10
1.6. Coleta de dados.....	11
1.6.1. <i>Por que devemos coletar dados?</i>	12
1.6.2. <i>Vantagens da coleta de dados</i>	12
1.6.3. <i>Plano de coleta de dados</i>	13
1.6.4. <i>Métodos de coleta de dados</i>	15
Capítulo 2. Introdução à web semântica	18
2.1. Web Semântica.....	18
2.2. Ontologia.....	20
2.3. Inteligência coletiva.....	23
Capítulo 3. Introdução ao ambiente virtual Python	25
3.1. O que é o ambiente virtual	25
3.2. Criando uma virtualenv	26
3.3. Gerenciador de pacotes <i>pip</i>	27

Capítulo 4. Introdução ao MySQL	31
4.1. O que é MySQL	31
4.2. Características do MySQL	31
4.3. Linguagem SQL	32
4.4. Modelo de entidade e relacionamento	33
4.5. Empresas que utilizam o MySQL como banco de dados	35
4.6. MySQL Workbench	36
4.7. Instalação MySQL Workbench.....	37
 Capítulo 5. Coleta de dados na web.....	38
5.1. Mecanismos de coleta de dados na web	38
5.1.1. <i>Web Crawler</i>	39
5.1.2. <i>Web Scraping</i>	40
5.2. Bibliotecas mais populares para coleta de dados na web	41
5.3. API de coleta de dados	43
5.3.1. <i>APIs para mídias sociais</i>	43
 Capítulo 6. Introdução ao MongoDB	45
6.1. O que é MongoDB	45
6.2. Características do MongoDB	45
6.3. Vantagens de utilizar o MongoDB.....	47
6.4. Estrutura de armazenamento.....	47
6.5 Comandos básicos para consultas no MongoDB	48
6.6. Empresas que utilizam o MongoDB.....	50
 Referências.....	52

Capítulo 1. Introdução a coleta de dados

Este capítulo tem como objetivo apresentar os principais conceitos relacionados a dados e sua coleta. Entenderemos os diferentes tipos de dados e suas estruturas. Além disso, vamos discutir a importância e o objetivo de realizar uma boa coleta de dados.

1.1. A crescente evolução dos dados

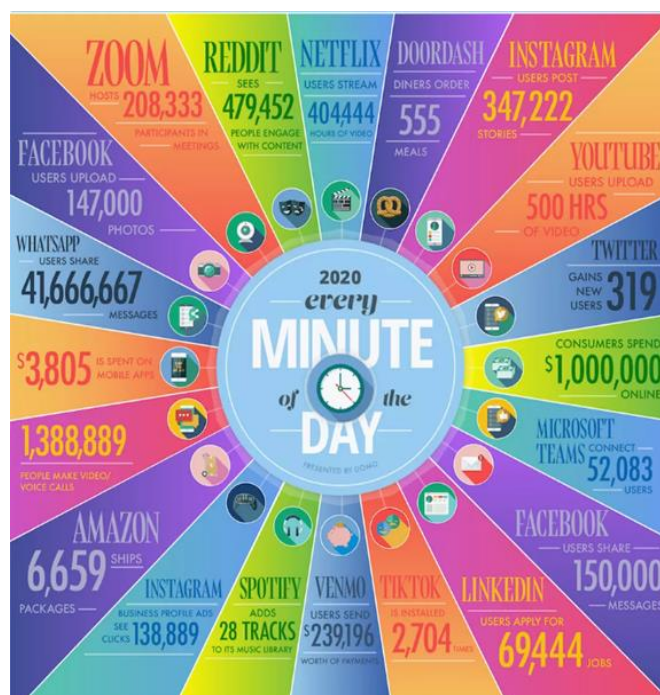
Estamos vivenciando uma crescente evolução dos dados no mundo inteiro. O número de dados produzidos a cada minuto é algo impressionante. Atualmente, tudo o que fazemos e consumimos está relacionado à coleta e obtenção dos dados.

Os dados são produzidos regularmente em cliques em anúncios, comportamentos em mídias sociais, compartilhamentos, viagens, transações, conteúdo de streaming e muito mais (DOMO, 2020).

É muito comum nos deparar com *Websites* que conseguem rastrear e coletar cliques e preferências de usuários, aplicativos que mensuram padrões como frequência cardíaca, distância percorrida, consumo de calorias. Temos também casas e carros inteligentes que conseguem identificar padrões de consumo e hábitos pessoais. A própria internet possui uma infinidade de dados distribuídos na rede (GRUS, 2016).

Anualmente, a Domo (empresa especializada em computação na nuvem) realiza um estudo que estima a quantidade de informação que é produzida a cada minuto todos os dias. Além de estimar esses dados, o estudo indica hábitos e consumos da população nas principais plataformas sociais. A Figura 1 mostra a quantidade de dados produzidos por minuto no ano de 2020.

Figura 1 - Dados produzidos por minuto em 2020.



Fonte: Domo (2020).

Conforme ilustrado na Figura 1, observamos as principais plataformas que mais produziram dados por minuto em 2020. Podemos tirar várias informações dessa imagem, porém vamos destacar que: a cada minuto, são produzidos mais de 347 mil *stories* no Instagram, 41 milhões de mensagens no WhatsApp e são gastos US\$ 1 milhão por minuto em compras on-line no mundo. Com essa infinidade de dados disponíveis na rede, é necessário criar mecanismos de coleta, obtenção e armazenamento desses dados.

1.2. Conceito de dados

Dados são os registros soltos, aleatórios e sem qualquer análise. São informações não tratadas que ainda não apresentam relevância. São códigos que isoladamente não possuem nenhum significado, mas quando agrupados podem transmitir uma mensagem ou representar algo ou até mesmo um conhecimento. Podemos definir "dado" como uma sequência de símbolos quantificados ou

quantificáveis. Os dados são tudo aquilo que pode ser quantificado como, por exemplo, imagens, sons, textos ou animações. Na Figura 2, temos uma exemplificação dessa ideia.

Figura 2 - Exemplo de dados.



Fonte: Imagem ilustrativa.

O que podemos dizer dessa imagem?

1. São bolas de golfe.
2. As bolas são de cor branca.
3. Percebemos que elas já foram usadas.
4. São do mesmo tamanho.

Podemos identificar várias características observando essa imagem. No entanto, imagine que essa imagem não foi disponibilizada. Apenas foi disponibilizado um dado como, por exemplo, um número ou uma cor. Perceba que não é possível saber o que ele significa ou o que ele representa, pode ser algo muito relevante ou pode ser nada. Porém, no momento que existir uma agregação com outro dado, ele passa a ser ou não uma informação. No exemplo ilustrado na Figura 2 podemos identificar os dados contidos na imagem, agregá-los, interpretá-los e, assim, obter

uma informação. Os dados podem ser do tipo numérico, textual, data e hora, bits e vários outros.

1.3. Conceito de Informação

Dizemos que a informação é o dado estruturado ou organizado que possui algum sentido. A informação é a matéria prima utilizada para o conhecimento, ela traz significado e compreensão sobre um determinado assunto ou situação. Se os dados agregados fazem sentido para quem o lê, então dizemos que existe um valor naquela informação. E é por meio da informação que podemos tomar decisões.

1. Requer unidade de análise.
2. Exige consenso em relação ao significado.
3. Exige, necessariamente, a mediação humana.

1.4. Conceito de conhecimento

Conhecimento é a informação processada e transformada em experiência pelo indivíduo. Ou seja, é o resultado de várias informações organizadas de forma lógica. O conhecimento é a capacidade, adquirida por alguém, de interpretar e operar sobre um conjunto de Informações. Se informação é o dado trabalhado, então o conhecimento é informação trabalhada.

1.5. Características dos tipos de dados

Antes de iniciar a coleta e posteriormente a análise dos seus dados, é necessário entender os diferentes tipos de dados. Existem 3 estruturas: dados estruturados, dados semiestruturados e dados não estruturados.

1.5.1. Dados estruturados

Os dados estruturados têm como característica ser bem definidos, inflexíveis, pensados antes da própria criação dos dados. Dessa forma, não é possível que tipos de dados diferentes das estruturas preestabelecidas sejam carregados. Por exemplo, se uma coluna de uma tabela for criada com o tipo de dado numérico, essa coluna não aceitará dados textuais. Ou seja, os registros daquela coluna devem seguir o mesmo formato dos outros registros daquela tabela (ELMASRI; SHAMKANT, 2019). A Tabela 1 apresenta um modelo de dado estruturado, referente à tabela Veículo que contém o cadastro de vendas de veículos. Observe que a tabela representa uma estrutura prévia definida.

Tabela 1 - Tabela de veículos.

Fabricante	Modelo	Ano	Valor venda
Fiat	Palio	2020	32.347
Volkswagen	Gol	2019	31.800
Ford	Fiesta Sedan	2015	27.572

Fonte: Elaborado pelo autor.

1.5.2. Dados não estruturados

Os dados não estruturados são o oposto dos dados estruturados. Eles não possuem uma estrutura pré-definida, alinhada ou padronizada. Os dados não estruturados se caracterizam por possuir uma estrutura flexível e dinâmica ou, até mesmo, nenhuma estrutura. Esses dados podem ser compostos por vários elementos diferentes, como: imagens, áudios, vídeos, gráficos e textos. Eles são difíceis de processar devido a sua complexibilidade e formatação. Os dados não estruturados podem ser encontrados em mídias sociais, e-mails, fotos, vídeos, chats, arquivos de logs, sensor de IoT, entre outros.

1.5.3. Dados semiestruturados

Os dados semiestruturados são uma combinação entre os dados estruturados e os não estruturados. Esses dados possuem uma estrutura heterogênea, não sendo uma estrutura completamente rígida e nem exclusivamente flexível. Os dados presentes na Web possuem essa definição. Em muitos casos, os dados dispõem de uma definição regular (por exemplo, um catálogo de produtos), em outros, um padrão estrutural que pode ser identificado ou não existem informações descritivas relacionadas (por exemplo, um arquivo de imagem) (ABITEBOUL, S., 1997).

Os dados semiestruturados não estão devidamente estruturados em células ou colunas, no entanto, eles possuem elementos que facilitam a separação de campos e registros. Porém, não possuem elementos que facilitem a identificação dos campos e registros. Por exemplo, os dados semiestruturados podem ser armazenados em formato XML (do inglês *eXtensible Markup Language*) que é uma linguagem de marcação recomendada pela W3C para a criação de documentos com dados organizados hierarquicamente. Esses dados são inseridos através de nós com abertura e fechamento precedidos por um símbolo de '*'*'. A Figura 3 ilustra o armazenamento de dados semiestruturados em formato XML.

Figura 3 - Dados semiestruturados em formato XML.

```
< Pessoa >
  < nome > Leandro Less < / nome >
  < profissao > Professor < / profissao >
  < estado > MG < / estado >
  < cidade > Belo Horizonte < / cidade >
< / Pessoa >

< Pessoa >
  < nome > Daniele Lessa < / nome >
  < profissao > Assessora Administrativa < / profissao >
  < estado > MG < / estado >
  < cidade > Belo Horizonte < / cidade >
< / Pessoa >
```

Fonte: Elaborado pelo autor.

Em um banco de dados semiestruturado, as informações são guardadas e manipuladas na forma de XML, por exemplo, ao invés de tabelas (VELLOSO, 2017).

Um outro exemplo de dados semiestruturados é o armazenamento em formato JSON. A Figura 4 ilustra o armazenamento de dados semiestruturados em formato JSON:

Figura 4 - Dados semiestruturados em formato JSON.

```
[
  {
    "nome": "Leandro Lessa",
    "profissao": "Professor",
    "estado": "MG",
    "cidade": "Belo Horizonte",
  },
  {
    "nome": "Daniele Lessa",
    "profissao": "Assessora Administrativa",
    "estado": "MG",
    "cidade": "Belo Horizonte"
  },
]
```

Fonte: Elaborado pelo autor.

Perceba que existe uma identificação entre os campos (nome, profissão) e os registros (Leandro, professor) que permite extrair informações do arquivo.

1.6. Coleta de dados

A coleta de dados pode ser definida como o processo de coleta e medição de informações que visa obter conteúdo relevante de várias fontes. As coletas são realizadas em *websites*, formulários, entrevistas, questionários, sistemas e de pessoas em redes sociais. Os dados coletados podem ser utilizados para tarefas de pesquisas, estudos, planejamento, desenvolvimento ou experimentos.

Além disso, a coleta permite encontrar respostas a perguntas reais e obter novos *insights* que de outra forma não seriam imediatamente óbvios. Por exemplo, a

avaliação precisa dos dados coletados pode ajudar você a analisar as tendências atuais e até mesmo prever tendências futuras.

1.6.1. Por que devemos coletar dados?

Vivemos em uma época fortemente influenciada pelo uso dos dados. Cada instituição empresarial depende de informações perspicazes transmitidas por dados para tomar decisões que são essenciais para o crescimento organizacional. Cada instituição pode se tornar mais bem-sucedida e eficiente com o uso das percepções adquiridas por meio da coleta e análise dos dados. Ou seja, a coleta pode se tornar um potencial competitivo.

A partir do momento que uma instituição possui os dados corretos em mãos e fazem o uso de ferramentas que permitem processá-los e analisá-los, esses dados podem ser transformados em informações e, conseqüentemente, permitem que as tomadas de decisões sejam mais assertivas. No entanto, os dados coletados devem ser relevantes, precisos e contextualizados. Caso contrário, os dados podem ser enganosos e ter um impacto negativo nos negócios. Por essa razão, inicialmente precisamos ter cautela e tratar as conclusões obtidas por meio da coleta de dados como hipóteses, ao invés de declarações de fato. Devemos agir em pequenos passos incrementais, em vez de fazer mudanças radicais ou tomar grandes decisões de uma só vez.

1.6.2. Vantagens da coleta de dados

Abaixo contém algumas vantagens que podem ser obtidas por uma instituição que utiliza coleta de dados em seus processos.

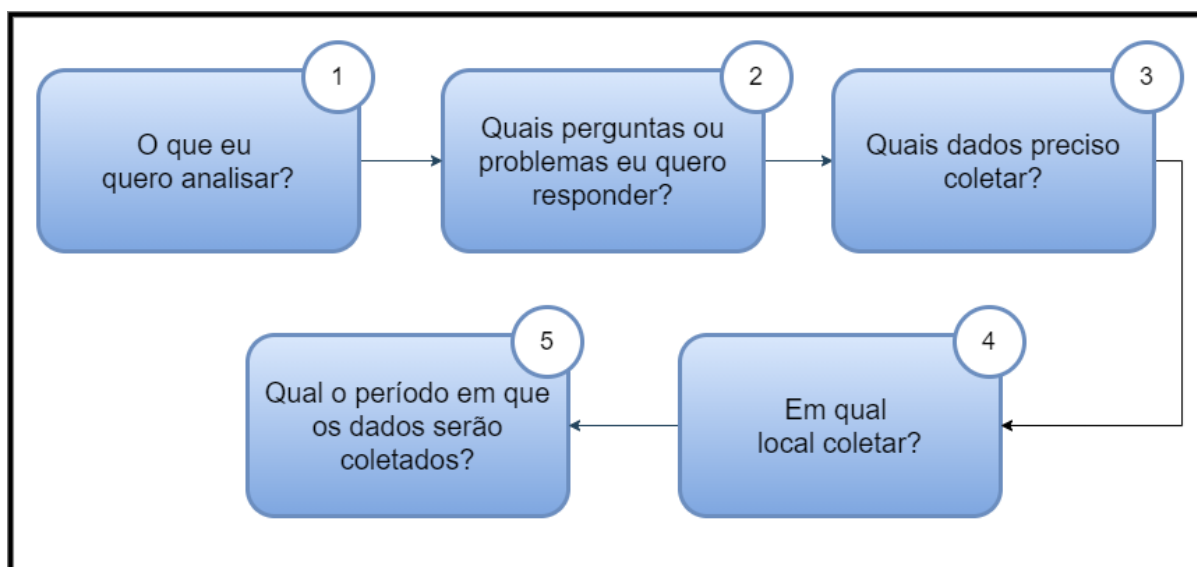
- Ajuda a concentrar sua energia e recursos limitados nas soluções que possuem maior impacto;
- Ajuda a entender melhor seus clientes;
- Ajuda a analisar melhor as tendências, de acordo com a forma como as opiniões e o comportamento dos clientes;
- Tomar decisões mais rápidas e eficazes;

- Realizar segmentação de públicos e selecionar estratégias de *marketing* dedicadas para cada segmento;
- Melhoria de produtos, serviços e processos baseados em dados de *feedback*;
- Ajuda a melhorar o relacionamento com o cliente.

1.6.3. Plano de coleta de dados

Antes de iniciar uma coleta de dados, temos que definir um plano para ela. Para que esse plano seja bem-sucedido, a primeira coisa que temos é estabelecer alguns passos a serem seguidos. Além disso, não podemos coletar dados sem antes ter definido claramente o objeto e o problema. A Figura 5 ilustra as etapas do plano de coleta de dados.

Figura 5 - Etapas do plano de coleta de dados.



Fonte: Elaborado pelo autor.

1. O que eu quero analisar?
2. Quais perguntas ou problemas eu quero responder?
3. Quais dados preciso coletar?
4. Em qual local coletar?

5. Qual o período em que os dados serão coletados?

Devemos analisar e verificar se os dados que decidimos coletar vão conseguir responder às perguntas que estamos propondo.

Por exemplo, vamos realizar uma coleta na *web* em um site que mostra a disputa de candidatos à prefeitura de alguma cidade. Nesse *site* é possível identificar vários dados referentes a cada candidato. Temos nome, partido, estado, cidade, número de votos recebidos, escolaridade, vida política, entre outros. Diante desse cenário, vamos criar um plano de coleta para extrair informações.

1. O que eu quero analisar?

- Pessoas que se candidataram à prefeitura da cidade de Belo Horizonte.

2. Quais perguntas ou problemas eu quero responder?

- Quantidade de partidos por candidato;
- Quantidade de concorrentes.

3. Quais dados preciso coletar?

- Nome do candidato;
- Partido;
- Cidade.

4. Em qual local coletar?

- Website http://divulga_resultados.com.br.

5. Qual o período em que os dados serão coletados?

- No mês de outubro do ano 2020.

Perceba nesse exemplo, que não coletamos todos os dados que estavam disponíveis no site. Nós apenas delimitamos o escopo e determinamos o que coletar.

Observe que ao seguir as etapas do plano de coleta, fica mais claro e fácil identificar o que e como coletar.

1.6.4. Métodos de coleta de dados

Existem vários métodos que possibilitam a coleta de dados. Abaixo vamos citar os mais conhecidos e utilizados (OLSEN, 2015).

Formulários e questionários: Esse método é um dos mais procurados devido à sua facilidade de criação, além de ser personalizável e poderoso, permitindo que a coleta seja segura com pouco esforço. Os questionários podem possuir perguntas fechadas ou abertas em sua construção, sendo que perguntas fechadas são que possuem múltiplas escolhas e perguntas abertas constituem por texto livre em linguagem natural. Esse método melhora a precisão dos dados em grande escala por estarem estruturados por padrão. Atualmente, com o avanço da tecnologia, ferramentas avançadas de coleta de dados estão disponíveis para os usuários de forma on-line permitindo a criação de seus próprios formulários eletrônicos.

Exemplos práticos são questionários de papel ou eletrônicos, como *google forms* ou *survey monkey*.

Entrevista: Esse método pode ser realizado pessoalmente, por meio de ligação telefônica ou *chat* na web em tempo real. Perguntas abertas são realizadas com mais frequência. É um método caro de coleta, uma vez que não pode haver espaço para erros (BELEI et al., 2008).

Observação: As informações coletadas pelo pesquisador podem ser baseadas nos julgamentos que eles fazem da observação, mas não precisam ser sempre tendenciosas. A observação ajuda a anotar as mudanças que acontecem em tempo real, o que nem sempre é possível com outros métodos. São muito utilizadas para formar hipóteses. Porém, nem sempre são viáveis, pois depende da situação e resultados tendenciosos podem ser esperados.

Por exemplo, você pode observar como o humor do seu público muda em tempo real.

Documentos e registros: Faz uso dos dados já existentes para coletar informações. Não é necessário “perder tempo” buscando informações, uma vez que grande parte da pesquisa já está documentada. É um dos métodos mais econômicos, embora não seja tão eficiente quanto os outros métodos.

Exemplo prático deste método são os registros financeiros.

Grupo focal: Se enquadram em coleta de dados qualitativos. Esse tipo de pesquisa envolve um grupo de indivíduos que fornece *feedback* e respostas às perguntas abertas feitas a eles. O principal propósito desse método é coletar opiniões coletivas, e não apenas individuais (OLSEN, 2015).

Exemplo de pergunta desse método: “Qual foi a funcionalidade favorita no desenvolvimento desse aplicativo?” O grupo discute entre eles e expõe uma opinião comum.

Histórias Orais: Se caracteriza por um método que define a coleta entre as experiências e pensamentos de pessoas que fizeram parte de um evento estabelecido. São baseados em um único evento ou fenômeno.

Exemplo: Reportagem com pessoas envolvidas no atentado de 11 de setembro nos EUA.

Pesquisa de Combinação: Se caracteriza pela combinação dos métodos de grupo focal e entrevistas. O principal objetivo é melhorar a participação dos entrevistados, a fim de obter dados sobre tópicos sensíveis sem muita dificuldade. Esse método de pesquisa protege o anonimato, o que permite respostas imparciais e precisas, aumentando assim a qualidade dos dados. Sua desvantagem é que consome muito tempo para ser executado.

Rastreamento on-line: Método que utiliza recursos eletrônicos para coletar dados de clientes e potenciais clientes. É realizado pelo próprio provedor de hospedagem ou *softwares* de análises. Pode utilizar *pixels* de busca e *cookies* para rastrear atividades de visitantes em um *site* ao longo de várias sessões (OLSEN, 2015).

Exemplo: Visitar *sites* da *web*, autorizar dados de *cookies*.

Análise de marketing on-line: Método que se caracteriza em coletar dados através de campanhas de *marketing* veiculadas por meios de mídias sociais, *e-mails*, páginas na *web*, anúncios em diversos destinos, entre outros.

Exemplo: A ferramenta pode dizer quem clicou em seu anúncio, quantas vezes foi clicado, de qual dispositivo foi clicado, de qual região você obteve o máximo de cliques e assim por diante.

Monitoramento de mídia social: Com a presença das redes sociais no negócio as plataformas digitais não apenas constroem a marca, mas são utilizadas para coletar dados de pessoas.

Exemplo: é possível medir o envolvimento de seus clientes com suas postagens por meio do número de impressões, curtidas, compartilhamentos e comentários. Além disso, pode analisar as atividades dos seguidores e segmentá-los baseado na interação na página das redes sociais.

Capítulo 2. Introdução à web semântica

Neste capítulo, vamos abordar os principais conceitos de web semântica e suas características.

Objetivo da *web* semântica é aprimorar o serviço de busca e a exibição de dados para os usuários.

2.1. Web Semântica

Nos últimos anos, a utilização da internet se tornou cada vez mais acessível. Isso possibilitou o crescimento explosivo da quantidade de dados e informações disponíveis na *web* em todo o mundo. Dia após dia, conteúdos são criados, informações são compartilhadas e dados são hospedados. Em nenhum momento da história houve com tamanha intensidade tantos dados produzidos. No entanto, apesar do grande número de dados na *web* e dos mecanismos de busca disponíveis para suas pesquisas, os usuários muitas das vezes não conseguem encontrar conteúdos relevantes e satisfatórios (SOUZA; ALVARENGA, 2004).

Por exemplo, como conseguiríamos encontrar com exatidão o que procuramos sendo que existem diversas fontes diferentes de informações espalhadas no mundo inteiro, levando em consideração que se encontram palavras que significam coisas diferentes em contextos diferentes? Todo esse excesso de dados faz com que a busca se torne poluída e, assim, as respostas e os resultados que um usuário procura podem ser interferidos por outros resultados que não lhe são úteis.

Preocupado com o grande crescimento desenfreado da internet, Tim Berners-Lee, o criador da *World Wide Web*, juntamente com James Hendler e Ora Lassila propõem o conceito da Web semântica, em 2001, através de um artigo na revista *Scientific American* intitulado: “Web Semântica: um novo formato de conteúdo para a Web que tem significado para computadores vai iniciar uma revolução de novas possibilidades” (BERNERS-LEE et al., 2001).

Para começarmos a entender o conceito de *web* semântica, precisamos pensar que, ao realizarmos uma pesquisa na internet geralmente utilizamos palavras, e os resultados se relacionam com as palavras lançadas. A grande questão é que a internet não foi pensada para "compreender" a pesquisa do indivíduo, apenas correlacionar os dados e enviá-los como resposta. Nesse sentido, a *web* semântica vem como um aprimoramento da *web* tradicional que busca estruturar o conteúdo de forma clara e definida, permitindo que os computadores interajam entre si, gerando resultados que sejam significativos para os humanos e para as máquinas

A ideia é que a *web* semântica ou Web 3.0 possa servir de auxílio aos usuários, como assistentes pessoais. Primeiramente, ela deve aprender os padrões do usuário, o que geralmente ele consome na *web* através de pesquisas, compras ou outros registros relacionados como agendas on-line e, a partir desse ponto, gerar respostas mais objetivas e coesas, filtrando outros resultados que ele possa não gostar. Desta forma, os resultados são diretamente relacionados ao perfil do usuário (SOUZA; ALVARENGA, 2004).

Nesse sentido, você poderia pesquisar: "onde eu poderia ir no final de semana?" e obter de fato uma resposta satisfatória que leva em consideração o seu perfil.

O problema que temos hoje é que os sistemas de buscas atuais não possuem entendimento da pesquisa que se está fazendo. Quando pesquisamos no *Google* qualquer questão, ele não retorna um resultado baseado no entendimento da pergunta. Há, na verdade, uma associação de palavras-chave ao que se procura. Por exemplo, se você buscar "laranja" pode ter como resultado tanto a fruta quanto objetos da cor laranja, ou se colocar "*apple*", terá maçãs ou produtos e serviços dessa empresa de tecnologia.

O grande lance da *web* semântica é reduzir a incidência de pesquisas infrutíferas ou com resultados que não possuem ligação com o que o indivíduo está procurando e aprender a responder dúvidas.

O conceito não está focado na interface visual, ou seja, para muitos leigos não haveria diferença alguma, pois a informação continua disposta da mesma forma na página. A diferença irá residir em uma mudança na programação de como o conteúdo é apresentado e de que forma e quais conteúdos retornam para você enquanto resposta à sua pesquisa. Essa mudança fará que os computadores consigam entender o que desejamos com aquela pesquisa. O maior desafio é criar essas tecnologias que entendam o significado das palavras e termos dentro do contexto em que vivemos.

A *web* semântica não visa ser um novo tipo de rede de informações, mas um projeto para fazer com que a rede de informações na internet seja mais "humanizada", aplicando conceitos inteligentes à *web* atual. Dessa forma, poderemos ter um tipo de conteúdo mais qualificado, definido e bem quantificado para que o usuário tenha um acesso mais eficiente, tornando o browser um assistente que orienta o usuário através de uma interação inteligente.

2.2. Ontologia

Um termo muito importante para a *Web Semântica* é a Ontologia. A ontologia é um modelo de dados que representa um conjunto de conteúdos em um determinado campo e domínio e como esses conteúdos se relacionam. Essa ideia e conceito é muito utilizada no campo da IA (Inteligência Artificial).

Objetivo da ontologia é a organização dos conteúdos, o aperfeiçoamento das buscas, a construção de bases de conhecimento e a padronização do vocabulário para determinados domínios (PICKLER, 2007).

A ontologia analisa os indivíduos (objetos na visão mais básica), suas classes, tipos, conjuntos e coleções. Também analisa as propriedades dos indivíduos, seus atributos e a forma com que eles se relacionam.

Na *web* semântica as páginas são ligadas pelas ontologias, pois, dessa forma, você conseguirá obter conexões lógicas entre os elementos da pesquisa. A criação automatizada desse processo é o grande desafio da ontologia.

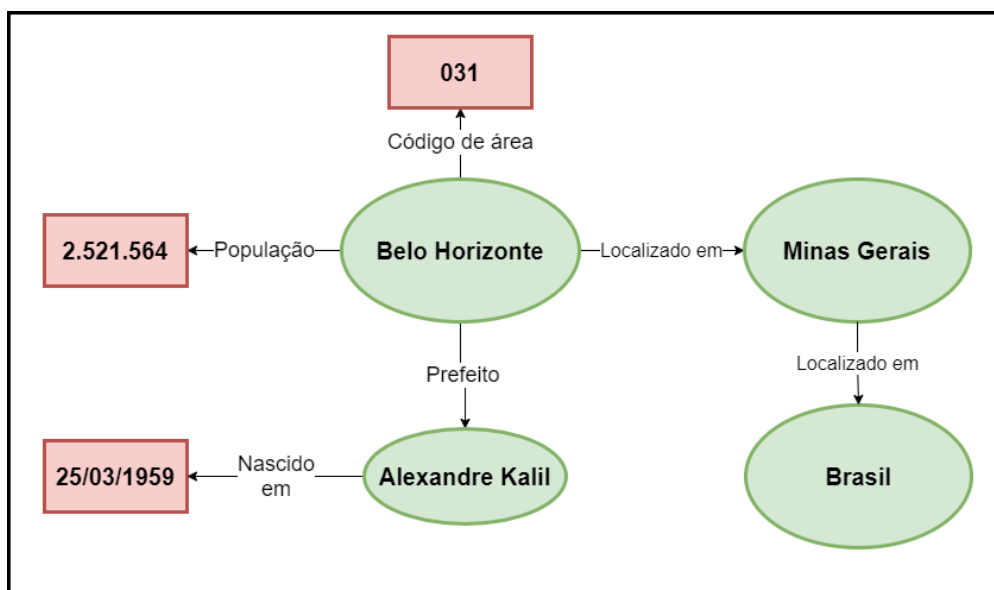
Esse processo pode ser criado manualmente, porém será muito trabalhoso e de alto custo, mas existem tecnologias que auxiliam isso como, por exemplo, o processamento probabilístico da linguagem, onde se analisa e correlaciona diversos termos das pesquisas para obter e entender a lógica, conceito e significado do que está sendo dito (PICKLER, 2007).

Para exemplificar, podemos pensar na companhia aérea Azul. Se fizermos uma pesquisa apenas com "Voos Azul", o processamento probabilístico da linguagem vai analisar o texto e os documentos relacionados buscando informações ligadas à companhia e não a cor azul. Nesse exemplo, os documentos que tiverem a ver com pintura, *design* e cores não serão apresentados.

Outra tecnologia que auxilia o processo de obtenção das conexões lógicas entre os elementos da pesquisa é o RDF (W3C, 2014), uma linguagem utilizada para representar informações na internet, sejam arquivos de dados ou metadados. O RDF cria um conjunto, como um banco de informações que representa um conjunto de conhecimentos que podem ser divididos em partes menores que são ou não interligadas e relacionadas entre si. Ele é composto pelos recursos, propriedades e indicações.

Recurso é tudo aquilo que possui URL, ou seja, pode ser identificado na *web*. A propriedade armazena esse recurso que tem um nome específico e a indicação faz a união entre os dois, juntamente com o valor. Na figura abaixo nós conseguiremos entender melhor esse conceito:

Figura 6 - Modelo RDF.



Fonte: Elaborado pelo autor.

É dessa forma que o computador consegue manipular o conhecimento e, na prática, é isso que é a *web* semântica: a manipulação mecânica da informação através de um computador onde, através das associações realizadas, pareça que ele está realmente entendendo o que estamos dizendo.

Através do RDF não é necessário estruturar as informações de forma hierárquica ou em tabelas, pois os recursos são flexíveis. Eles também podem ser representados em mapas conceituais, onde são convencionados os nomes dos objetos e sujeitos para que eles estejam relacionados na *web* semântica. Para isso é necessário o uso de vocabulários específicos em domínios específicos. Isso significa que alguns termos farão sentido se estiverem sendo usados ou falados em determinados contextos.

Com o mesmo objetivo temos também a linguagem das *tags*, que é o HTML. Ele, por meio de todas as *tags* usadas na confecção das páginas, auxilia na busca de resultados mais úteis e relevantes. Por exemplo, é comum ter o padrão de utilizar a *tag* H1 para destacar termos e a *tag* P para parágrafos. No momento da busca isso poderá ser utilizado como um filtro/selecionador do que pode ser mais importante.

As *tags* padronizadas ajudam no rastreamento das informações dentro das páginas, e por se tratar de uma tecnologia já existente, seria apenas necessária a padronização e o aperfeiçoamento para torná-la eficiente dentro da *web* semântica.

2.3. Inteligência coletiva

O conceito de Inteligência Coletiva foi criado pelo filósofo e sociólogo Pierre Lèvy que estuda o impacto da internet na sociedade.

Para Lèvy, a internet minimiza ou anula o entendimento de inteligência individual e se exalta a coletividade (LÈVY, 2007).

É necessário encarar esse conceito como um projeto em desenvolvimento, pois há sempre algo sendo acrescentado, descoberto ou testado. A cultura está em constante evolução e, nesse sentido, a inteligência coletiva colabora para a criação de novas tecnologias.

A internet é em si um ambiente composto de conhecimentos advindos de toda parte, pois no compartilhamento há a troca de conhecimento.

Esse processo está diretamente ligado aos programas de computador, pois para a criação e o aperfeiçoamento dos mesmos, geralmente há a reunião de vários especialistas, pesquisadores e desenvolvedores e cria-se então o ambiente da inteligência coletiva. Todos juntos, pensando em como atingir um objetivo em comum (LÈVY, 2007).

Para Lèvy, o ser humano é incapaz de produzir conhecimento sozinho e sem o auxílio de qualquer máquina.

Os princípios da inteligência coletiva são:

1. Primeiramente reconhecer que todo ser humano tem algum conhecimento;
2. No entanto, nenhum ser humano possui conhecimento de tudo;

3. É necessário compreender que cada indivíduo possui conhecimento em suas particularidades;
4. É preciso compartilhar ideias para que obtenha melhoria significativa através da colaboração.

Capítulo 3. Introdução ao ambiente virtual Python

Os aplicativos Python geralmente usam pacotes e módulos que não vêm como parte da biblioteca padrão. Os aplicativos às vezes precisarão de uma versão específica de uma biblioteca, porque ele pode exigir que um *bug* específico seja corrigido ou o aplicativo pode ser escrito usando uma versão obsoleta da interface da biblioteca.

Isso significa que pode não ser possível para uma instalação do Python atender aos requisitos de cada aplicativo. Por exemplo, se o aplicativo ~A~ precisa da versão 1.0 de um módulo específico, mas o aplicativo ~B~ precisa da versão 2.0, então os requisitos estão em conflito e a instalação da versão 1.0 ou 2.0 deixará incapaz de ser executado um dos aplicativos.

A solução para esse problema é criar um ambiente virtual, uma árvore de diretório independente que contém uma instalação do Python para uma versão específica do Python, além de vários pacotes adicionais.

3.1. O que é o ambiente virtual

Os ambientes virtuais, também conhecidos como “*virtualenvs*”, são uma forma de isolar diversos ambientes de desenvolvimento, permitindo ao desenvolvedor utilizar versões específicas de diversos pacotes sem impactar instalações de outras aplicações ou sistemas.

Quando criamos uma *virtualenv* é iniciado uma instalação do Python completa, com o executável do Python, pip e *setup-tools*.

Além disso, quando utilizamos ambientes separados por projeto, fica muito mais prático e fácil trabalhar. Muitas vezes é útil ter um ou mais ambientes Python onde você pode experimentar diferentes combinações de pacotes sem afetar sua instalação principal. Um ambiente virtual também é útil quando você precisa trabalhar

em um sistema compartilhado e não tem permissão para instalar pacotes, pois poderá instalá-los no ambiente virtual (PYTHON, 2021).

A imagem abaixo ilustra 3 ambientes virtuais isolados uns dos outros.

Figura 7 - Exemplo de estrutura de um ambiente virtual.



Fonte: Elaborada pelo autor.

Como podemos observar, existem 3 ambientes virtuais: coleta de dados, classificador e recomendador. Esses ambientes virtuais possuem instâncias diferentes umas das outras para executar o projeto que foi desenvolvido.

3.2. Criando uma virtualenv

O módulo utilizado para criar e gerenciar os ambientes virtuais é chamado de "venv". Quando uma *env* é criada, geralmente cria-se a versão do Python mais recente disponível no sistema operacional. Dessa forma, se você tiver várias versões do Python instaladas em seu computador, poderá selecionar a versão específica a ser utilizada na criação do ambiente virtual.

Após ter instalado o Python na máquina, vamos abrir o *prompt* de comando DOS e executar o seguinte comando para instalar a virtualenv:

```
>> pip install virtualenv
```

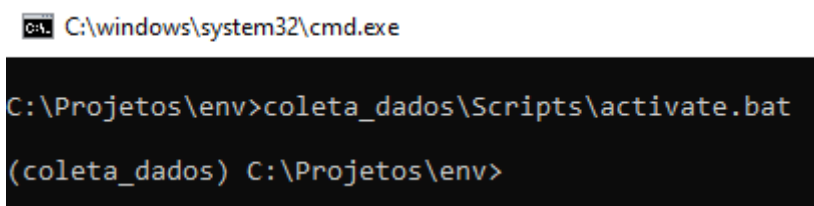
Em seguida, vamos criar um ambiente virtual chamado: "coleta_dados" com a versão do python3.

```
>> virtualenv -p python3 coleta_dados
```

Pronto. A *virtualenv* já está criada. Para acessar a virtual *env* no ambiente *Windows* execute o comando:

```
>> coleta_dados\Scripts\activate.bat
```

Figura 8 - Exemplo de ativação da máquina virtual no *Windows*.



```
C:\windows\system32\cmd.exe

C:\Projetos\env>coleta_dados\Scripts\activate.bat
(coleta_dados) C:\Projetos\env>
```

Fonte: Elaborado pelo autor.

Se você utilizando MAC OSX ou Unix, execute o comando:

```
>> coleta_dados/bin/activate
```

3.3. Gerenciador de pacotes *pip*

O programa chamado *pip* pode ser utilizado para instalar, atualizar e remover pacotes.

Para esse curso, vamos utilizar alguns comandos do *pip*, com por exemplo: *pip install*, *pip list*, *pip freeze*.

Para realizar a instalação de uma biblioteca, basta executar o seguinte comando:

```
>> pip install requests
```

O comando abaixo instala a biblioteca do *requests*:

Figura 9 - Instalação biblioteca requests.

```
(coleta_dados) C:\Projetos\env>pip install requests
Collecting requests
  Using cached requests-2.25.1-py2.py3-none-any.whl (61 kB)
Requirement already satisfied: certifi>=2017.4.17 in c:\projetos\env\coleta_dados\lib\site-packages (from requests) (2020.12.5)
Requirement already satisfied: idna<3,>=2.5 in c:\projetos\env\coleta_dados\lib\site-packages (from requests) (2.10)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\projetos\env\coleta_dados\lib\site-packages (from requests) (1.26.3)
Requirement already satisfied: chardet<5,>=3.0.2 in c:\projetos\env\coleta_dados\lib\site-packages (from requests) (4.0.0)
Installing collected packages: requests
Successfully installed requests-2.25.1

(coleta_dados) C:\Projetos\env>
```

Fonte: Elaborado pelo autor.

Caso queira instalar uma versão específica de uma biblioteca, podemos executar o comando:

```
>> python -m pip install requests=2.6.0
```

Figura 10 - Verificando versão instalada

```
(coleta_dados) C:\Projetos\env>python -m pip install requests==2.6.0
Collecting requests==2.6.0
  Downloading requests-2.6.0-py2.py3-none-any.whl (469 kB)
    |████████████████████████████████████████| 469 kB 1.7 MB/s
Installing collected packages: requests
  Attempting uninstall: requests
    Found existing installation: requests 2.25.1
    Uninstalling requests-2.25.1:
      Successfully uninstalled requests-2.25.1
Successfully installed requests-2.6.0

(coleta_dados) C:\Projetos\env>
```

Fonte: Elaborado pelo autor.

Para desinstalar, utilize o comando:

```
>> pip uninstall requests
```

Figura 11 - Desinstalando biblioteca.

```
C:\windows\system32\cmd.exe

(coleta_dados) C:\Projetos\env>pip uninstall requests
Found existing installation: requests 2.25.1
Uninstalling requests-2.25.1:
  Would remove:
    c:\projetos\env\coleta_dados\lib\site-packages\requests-2.25.1.dist-info\*
    c:\projetos\env\coleta_dados\lib\site-packages\requests\*
Proceed (y/n)? y
  Successfully uninstalled requests-2.25.1

(coleta_dados) C:\Projetos\env>
```

Fonte: Elaborado pelo autor.

O comando *pip list* exibe todas as bibliotecas instaladas na *virtualenv*.

Figura 12 - Verificando bibliotecas instaladas.

```
C:\windows\system32\cmd.exe

(coleta_dados) C:\Projetos\env>pip list
Package            Version
-----
certifi             2020.12.5
chardet             4.0.0
idna                2.10
numpy               1.20.1
pandas              1.2.3
pip                 21.0.1
python-dateutil     2.8.1
pytz                2021.1
requests            2.25.1
setuptools          53.0.0
six                 1.15.0
urllib3             1.26.3
wheel               0.36.2

(coleta_dados) C:\Projetos\env>_
```

Fonte: Elaborado pelo autor.

Você pode criar uma cópia de todas as bibliotecas instaladas na sua *virtual env*. Para isso, vamos utilizar o comando *pip freeze*. Por padrão, é comum colocar esta lista em um arquivo *requirements.txt*.


```
>> pip freeze > requirements.txt
```

Para instalar todas as bibliotecas em uma outra *virtualenv* sem precisar instalar uma biblioteca por vez e executar o comando:

```
>> pip install -r requirements.txt
```

O *pip* existe muito mais comandos que podem ajudar no gerenciamento das bibliotecas. Você pode consultar a documentação completa em:

```
https://docs.python.org/3/
```

Capítulo 4. Introdução ao MySQL

Neste capítulo, vamos abordar os principais conceitos e características do banco de dados MySQL.

4.1. O que é MySQL

O MySQL é um sistema gerenciador de banco de dados relacional de código aberto (*open source*) usado na maioria das aplicações gratuitas com um funcionamento baseado em um modelo do tipo cliente-servidor. Utiliza a linguagem SQL (*Structure Query Language* – Linguagem de Consulta Estruturada), que é a linguagem mais popular para inserir, acessar e gerenciar o conteúdo armazenado num banco de dados (DU BOIS, 2008).

Atualmente, é o SGBD (sistema gerenciador de banco de dados) mais popular da *Oracle*.

4.2. Características do MySQL

Existem várias características que podemos atribuir ao banco de dados MySQL. Abaixo segue algumas das qualidades que o MySQL possui.

1. É gratuito;
2. Possui código aberto. Ou seja, é *open source*;
3. Altamente disseminado entre usuários e empresas no mundo;
4. Desempenho (possui alta resposta às requisições em grandes cargas de informações);
5. Possui segurança (controle de acesso, níveis de permissões, *backups*);

6. Integridade (possui mecanismos para não exclusão de registros quando infringe alguma restrição);
7. Possui alta disponibilidade, inclusive para aplicações *web*;
8. Seu código é estável e confiável;
9. Possui uma grande comunidade de usuários e de administradores;
10. Possui diversos conteúdos publicados na *web* sobre seu conteúdo e funcionamento.

4.3. Linguagem SQL

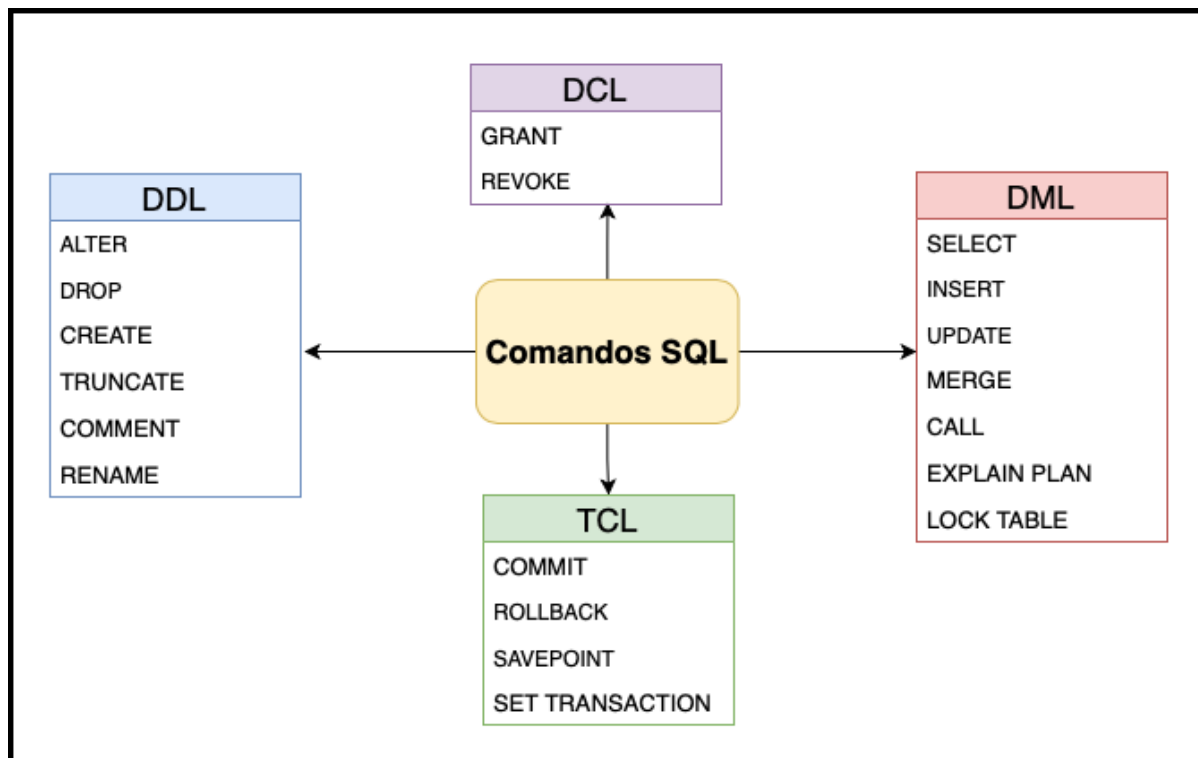
O MySQL possui a linguagem SQL (*Structured Query Language*) como linguagem de programação para executar comandos no banco de dados relacional (CARDOSO; CARDOSO, 2013).

A Linguagem SQL pode ser classificada em 4 tipos classes:

- **Linguagem de Definição de Dados** (*DDL – Data Definition Language*): comandos para a criação do banco de dados e dos objetos no banco de dados, como tabelas, índices, *constraints* etc., e para alteração e exclusão de objetos.
- **Linguagem de Manipulação de Dados** (*DML – Data Manipulation Language*): comandos para inserir, atualizar, deletar e consultar dados.
- **Linguagem de Controle de Dados** (*DCL – Data Control Language*): comandos para gerenciar permissões de acesso para usuários e objetos, ou seja, permissões para executar comandos DDL, DML, DCL e TCL.
- **Linguagem de Controle de Transação** (*TCL – Transaction Control Language*): comandos para realizar controle das transações de dados no banco de dados.

A Figura13 ilustra os comandos pertencentes a cada classe da linguagem SQL.

Figura 13 - Classes SQL.



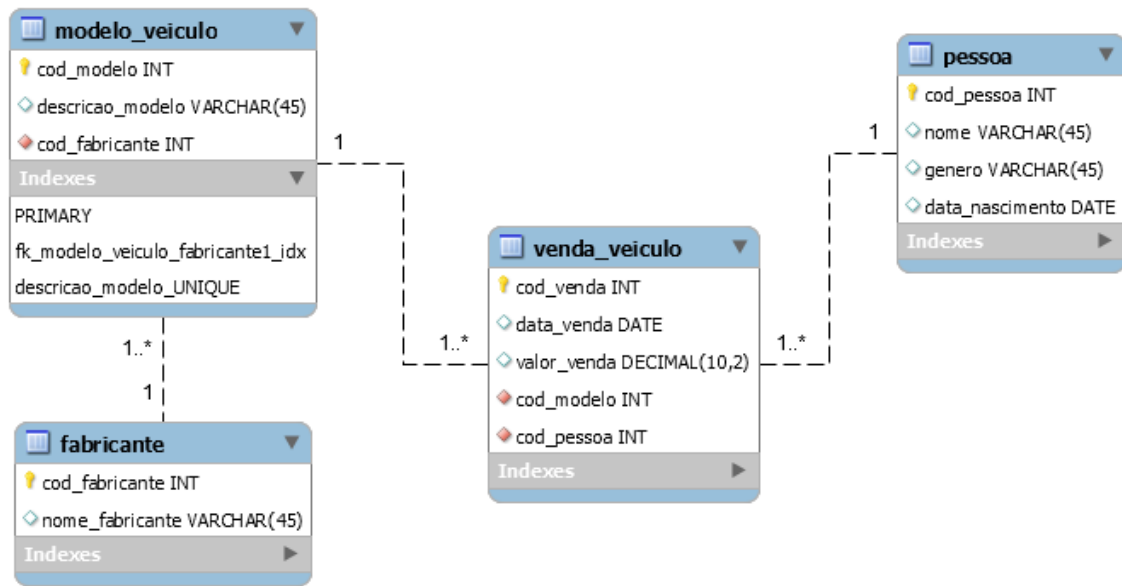
Fonte: Elaborado pelo autor.

4.4. Modelo de entidade e relacionamento

O modelo de entidade e relacionamento é um modelo conceitual utilizado para descrever as entidades (objetos) inseridos em um domínio de negócios, com seus atributos (características) e como eles se relacionam entre si. Desta forma, em uma modelagem de um modelo de banco de dados relacional temos: as tabelas que são denominadas de entidades e as suas colunas denominamos atributos.

A Figura 14 ilustra o modelo de entidade e relacionamento de um banco de dados baseado na venda de veículos.

Figura 14 - Modelo de entidade e relacionamento.



Fonte: Elaborado pelo autor.

Podemos perceber os relacionamentos entre as tabelas que compõem o banco de dados de vendas de veículos. A Figura 15 ilustra uma consulta em SQL que retorna os dados de nome, endereço e data da venda de uma pessoa.

Figura 15 - Exemplo de consulta SQL.

```

SELECT  pes.nome,
        pes.endereco,
        pve.data_venda as data_da_venda
FROM    pessoa as pes
JOIN    vendas_veiculo as vve
ON      pes.cod_pessoa = vve.cod_pessoa
WHERE   nome = 'Leandro Lessa'
  
```

Fonte: Elaborado pelo autor.

4.5. Empresas que utilizam o MySQL como banco de dados

Por ser um banco muito popular no mundo inteiro e devido as suas características, muitas empresas optaram por desenvolver suas soluções para ele. A Figura 16 ilustra algumas empresas que utilizam o MySQL como banco de dados em suas aplicações.

Figura 16 - Empresas que utilizam o MySQL.



Fonte: <https://www.mysql.com/>.

4.6. MySQL Workbench

MySQL *Workbench* é uma ferramenta gráfica para trabalhar com servidores e bancos de dados MySQL. O MySQL *Workbench* oferece suporte total ao servidor MySQL versões 5.6 e superiores. Também é compatível com versões mais antigas do servidor MySQL 5.x, exceto em certas situações (como exibir a lista de processos) devido a alterações nas tabelas do sistema. Não é compatível com as versões 4.x do servidor MySQL (MYSQL, 2020).

A funcionalidade do MySQL *Workbench* cobre cinco tópicos principais:

1. Desenvolvimento SQL: permite criar e gerenciar conexões com servidores de banco de dados. Além de permitir que você configure parâmetros de conexão, o MySQL *Workbench* oferece a capacidade de executar consultas SQL nas conexões de banco de dados usando o Editor SQL integrado.
2. Modelagem de Dados (*Design*): permite que você crie modelos de seu esquema de banco de dados graficamente, faça engenharia reversa e direta entre um esquema e um banco de dados ativo e edite todos os aspectos de seu banco de dados usando o Editor de Tabelas abrangente. O Editor de Tabelas fornece recursos fáceis de usar para editar tabelas, colunas, índices, gatilhos, particionamento, opções, inserções e privilégios, rotinas e visualizações.
3. Administração do servidor: permite administrar instâncias do servidor MySQL administrando usuários, realizando backup e recuperação, inspecionando dados de auditoria, visualizando a integridade do banco de dados e monitorando o desempenho do servidor MySQL.
4. Migração de dados: permite que você migre do *Microsoft SQL Server*, *Microsoft Access*, *Sybase ASE*, *SQLite*, *SQL Anywhere*, *PostgreSQL* e outras tabelas, objetos e dados RDBMS para o MySQL. A migração também oferece suporte à migração de versões anteriores do MySQL para as versões mais recentes.

5. MySQL *Enterprise Support*: Suporte para produtos empresariais, como MySQL *Enterprise Backup*, MySQL *Firewall* e MySQL *Audit*.

4.7. Instalação MySQL Workbench

O MySQL Workbench pode ser instalado facilmente através de linha de comando ou através de pacotes executáveis. Abaixo segue os links de acesso com a documentação e passo a passo para instalação dos principais sistemas operacionais.

1. **Windows:** <https://dev.mysql.com/doc/workbench/en/wb-windows.html>.
2. **LINUX:** <https://dev.mysql.com/doc/workbench/en/wb-linux.html>.
3. **MAC:** <https://dev.mysql.com/doc/workbench/en/wb-mac.html>.

Capítulo 5. Coleta de dados na web

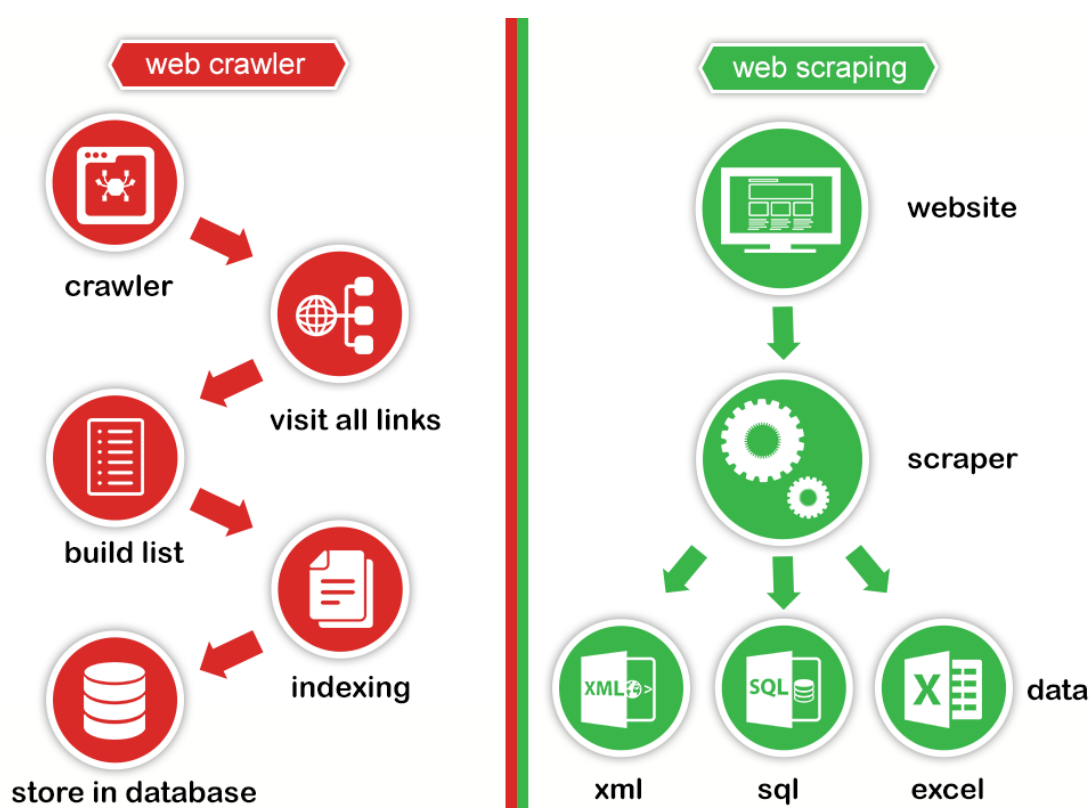
Neste capítulo, vamos abordar os principais conceitos e características da coleta de dados na Web.

5.1. Mecanismos de coleta de dados na web

A coleta de dados na *web* é uma forma de mineração que permite extrair dados de sites hospedados na *web* de forma automática, convertendo-os em informação estruturada para posteriormente ser utilizada para análise e tomada de decisões.

Existem dois mecanismos de coleta de dados na *web*: *Web Crawler* e *Web Scraping*.

Figura 17 - Diferenças entre Web Crawler e Web Scraping.



Fonte: <http://proweb scraping.com/web-scraping-vs-web-crawling/>.

5.1.1. Web Crawler

Web Crawler ou rastreador é um *software* de computador desenvolvido que realiza uma varredura em páginas da *web* de maneira sistemática, extraindo conteúdos relevantes. A principal função de um *Web Crawler* é examinar links pela internet e construir uma espécie de mapa que contém os links dos sites que se relacionam entre si (LESSA, 2019).

O *Web Crawler* pode exercer várias funções como:

1. Avaliar e comparar sites de concorrentes;
2. Coletar informações para atualizar base de dados para motores de busca;
3. Criar cópia das páginas visitadas para a indexação;
4. Realizar tarefas de manutenção automatizadas como, por exemplo, checagem de links, e-mails e validação de códigos HTML entre outros (LESSA, 2019).

Exemplos de *Web Crawling*: *Google*, *Yahoo* ou *Bing*.

Esses buscadores realizam o rastreamento das páginas da *web* e usam as informações para indexá-las.

Etapas de um *Web Crawler*:

Abaixo seguem as etapas realizadas em um rastreamento:

1. Seleção de uma URL ou uma lista de URLs iniciais;
2. Adicione a URL em lista de URLs ou qualquer estrutura de armazenamento;
3. Escolha a URL;
4. Obtenha a página da *Web* correspondente a essa URL;
5. Analise essa página da *web* para encontrar novos links de URL;
6. Adicione todas as URLs recém-encontradas à lista de URLs;

7. Vá para o passo 3 e reitere até que a lista esteja vazia.

5.1.2. Web Scraping

Web Scraping é um *software* de computador desenvolvido que realiza extração de dados de determinados sites de forma automatizada.

A raspagem de dados envolve localizar os dados e depois extraí-los. O *Web Scraping* não "copia e cola", mas busca de forma direta os dados de maneira precisa. A coleta de dados não se limita apenas à *web*, mas em qualquer lugar no qual os dados estejam armazenados (PATEL, 2020).

Por exemplo, você quer trabalhar com inteligência de preços. Você extrairia o preço de vários produtos específicos da *Amazon* ou de qualquer outro site de comércio eletrônico. Além disso, o *Web Scraping* pode ser utilizado para comparar preços, monitorar o tempo, pesquisar tendências que estão em alta no mercado, entre outras coisas.

Etapas de um *Web Scraping*:

O processo de coleta de dados de um *Web Scraping* pode ser classificado em 3 etapas:

1. Solicitação-Resposta:

- O primeiro passo é solicitar ao site de destino o conteúdo de uma URL específica.
- Em seguida, o *Web Scraping* obtém as informações solicitadas em formato HTML.

2. Analisar e Extrair:

- Quando se trata de análise, geralmente se aplica a qualquer linguagem de computador. É o processo de tomar o código como texto e produzir uma estrutura na memória com a qual o computador possa entender e trabalhar.

- Simplificando, a análise HTML é basicamente receber código HTML e extrair informações relevantes, como o título, parágrafos, cabeçalhos na página, links, texto em negrito etc.

3. Dados de *Download*:

- A parte final é onde você baixa e salva os dados em um formato CSV, JSON ou em um banco de dados para que possam ser recuperados e usados manualmente ou empregados em qualquer outro programa.

5.2. Bibliotecas mais populares para coleta de dados na web

A seguir, vamos citar as principais bibliotecas utilizadas por desenvolvedores que usam a linguagem Python para coleta de dados na *web*.

Beautiful Soup

O *Beautiful Soup* é uma biblioteca Python para extrair dados de arquivos HTML e XML. Ele funciona com seu analisador favorito para fornecer maneiras idiomáticas de navegar, pesquisar e modificar a árvore de análise. Geralmente, economiza horas ou dias de trabalho dos programadores (SOUP, 2021).

Vantagens:

- É fácil de aprender e dominar. Por exemplo, se quisermos extrair todos os links da página da *web*. Pode ser simplesmente realizado por um simples código.
- Tem uma boa documentação abrangente que nos ajuda a aprender as coisas rapidamente.
- Ele tem um bom apoio da comunidade para descobrir os problemas que surgem enquanto trabalhamos com esta biblioteca (PALAKOLLU, 2019).

Selenium

O *Selenium* é um conjunto de ferramentas para automação de navegador da *web* que usa as melhores técnicas disponíveis para controlar remotamente instâncias do navegador e emular a interação de um usuário com o navegador.

Ele permite que os usuários simulem atividades comuns realizadas pelos usuários finais; inserindo texto em campos, marcando valores suspensos e caixas de seleção e clicando em links em documentos. Ele também fornece muitos outros controles, como movimento do mouse, execução arbitrária de *JavaScript* e muito mais.

Embora usado principalmente para testes *front-end* de sites, o *Selenium* é em seu núcleo uma biblioteca de agentes de usuário do navegador. As interfaces são onipresentes para sua aplicação, o que incentiva a composição com outras bibliotecas para se adequar ao seu propósito (SELENIUM, 2021).

Scrapy

Scrapy é uma estrutura rápida de rastreamento e raspagem da *web* de alto nível, usada para rastrear sites e extrair dados estruturados de suas páginas. Ele pode ser usado para uma ampla gama de propósitos, desde mineração de dados até monitoramento e testes automatizados (SCRAPY, 2020).

As principais características do *Scrapy* são:

1. Suporte interno para extrair dados de fontes HTML usando expressão XPath e expressão CSS.
2. É uma biblioteca portátil, ou seja, (escrita em Python e executada em *Linux*, *Windows*, *Mac* e *BSD*)
3. Pode ser facilmente extensível.
4. É mais rápido do que outras bibliotecas de raspagem existentes. Ele pode extrair os sites com 20 vezes mais rapidez do que outras ferramentas.

5. Ele consome muito menos memória e uso da CPU.
6. Isso pode nos ajudar a construir um aplicativo robusto e flexível com várias funções.
7. Ele tem um bom suporte da comunidade para os desenvolvedores, mas a documentação não é muito boa para os iniciantes.

5.3. API de coleta de dados

API (*Application Programming Interface*) é um conjunto de definições e protocolos usados no desenvolvimento e na [integração](#) de *software* de aplicações. Uma API é um *software* intermediário que permite que serviços distintos se comuniquem um com os outros sem precisar saber como eles foram implementados (HAT, 2021).

Uma API pode ser considerada por um conjunto de soluções e rotinas que utilizam padrões de programação para acessar um aplicativo, serviço ou plataforma baseada na *web*. Através das APIs, os aplicativos podem se comunicar uns com os outros sem conhecimento ou intervenção dos usuários. A API liga as diversas funções em um site de maneira que possam ser utilizadas em outras aplicações.

Uma maneira de obter dados da *web* é através das APIs. Através dos serviços e funcionalidades é possível coletar conteúdo específico sobre um determinado assunto e até mesmo conteúdo de mídias sociais.

5.3.1. APIs para mídias sociais

Atualmente, as mídias sociais disponibilizam APIs para coleta de dados dentro da sua plataforma. Esses serviços podem ser utilizados através de cadastro na própria plataforma de forma gratuita ou, em alguns casos, é necessário realizar investimento financeiro para obter acesso.

Cada mídia social, através de suas regras, define o que e quem pode extrair informações da sua plataforma. Deste modo, para ter acesso aos dados

disponibilizados, o usuário deverá ter um *token* para permissão de acesso. Caso ocorra descumprimento das regras impostas pela política de acesso ou mal-uso das APIs, a mídia social tem o poder de bloquear perfis que estiverem utilizando a ferramenta.

Por isso, é muito importante conhecer e respeitar as regras de acesso definidas para utilização das APIs de cada mídia social. É muito comum que as mídias sociais restrinjam aplicações que descumprem os termos de sua utilização. Sobre as questões de segurança, com o uso de dados pessoais impostas pela LGPD (Lei Geral de Proteção de Dados), muitas das APIs têm sido mais rigorosas para permitirem acesso aos usuários à sua plataforma e, ao mesmo tempo, têm suprimido coleta de dados que são considerados sensíveis.

Abaixo seguem alguns APIs das mais populares mídias sociais entre os usuários brasileiros:

- *Twitter:* [https://developer.twitter.com/en](https://developer.twitter.com/en;);
- *Facebook:* https://developers.facebook.com/docs/graph-api?locale=pt_BR;
- *LinkedIn:* <https://www.linkedin.com/developers/apps>;
- *Instagram:* <https://www.instagram.com/developer/register/>;
- *Google Data API:* <https://developers.google.com/gdata/docs/directory>;
- *YouTube:* <https://developers.google.com/youtube/v3/>.

Capítulo 6. Introdução ao MongoDB

Neste capítulo, vamos abordar os principais conceitos e características do banco de dados não relacional MongoDB.

6.1. O que é MongoDB

O MongoDB é um sistema gerenciador de banco de dados não relacional NOSQL, desenvolvido em C++ e baseado em documentos. É um banco de dados gratuito e *open source* de alta performance e flexível. O MongoDB é um banco de dados orientado a documentos, ou seja, os dados armazenados na sua estrutura são documentos, diferente dos bancos de dados baseados no modelo relacional, no qual cada dado é armazenado em linhas e colunas de tabelas.

MongoDB armazena dados em documentos flexíveis do tipo JSON que possui estrutura de chave-valor, o que significa que os campos podem variar de documento para documento e a estrutura de dados pode ser alterada ao longo do tempo (MONGODB, 2021).

6.2. Características do MongoDB

Abaixo contém algumas características do MongoDB:

1. Gratuito e *open source*;
2. Orientado a documentos: JSON:
 - Não existem estruturas de tabelas e sim armazenamento de documentos.
3. Multiplataforma;
4. Indexados:

- Pode realizar consultas de conteúdo dentro dos documentos.

5. Não possui integridade referencial:

- É possível inserir um dado em uma coleção que não existe, ela é imediatamente criada;
- Não precisa criar um modelo de banco de dados primeiro para depois armazenar um dado.

6. Utiliza linguagem NoSQL:

- É uma classe definida de banco de dados que fornece um mecanismo para armazenamento e recuperação de dados que são modelados de formas diferentes das relações tabulares usadas nos bancos de dados relacionais.

7. Não possui *schemas*:

- Um banco de dados no MongoDB possui muitas coleções e cada coleção é um conjunto de documentos. Não possuir *schemas* significa que dentro de uma coleção podem existir documentos com estruturas diferentes;
- Por exemplo: Documentos JSON armazenados em coleção podem ter atributos diferentes.

8. Possui escalabilidade horizontal e vertical:

- Horizontal: Aumento da capacidade do sistema adicionando-se mais servidores;
- Vertical: Aumento da capacidade de um sistema trocando o hardware de um servidor ou adicionando no servidor existente mais memória CPU.

6.3. Vantagens de utilizar o MongoDB

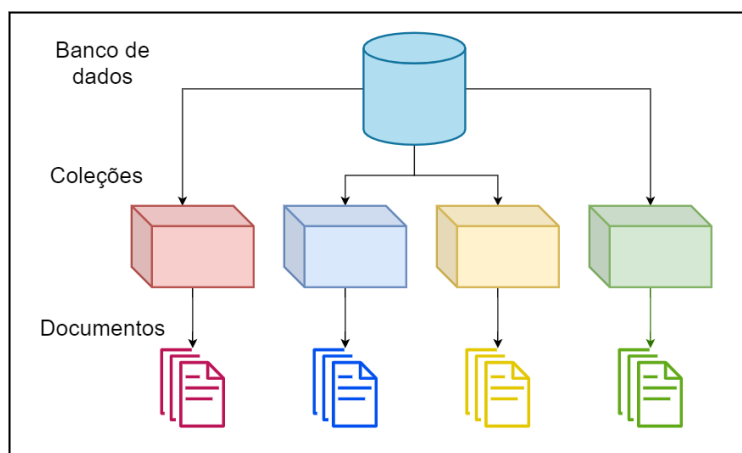
- Bom desempenho e facilidade para criar consultas;
- As consultas são mais fáceis de escrever, uma vez que não possui transações e junções entre instâncias;
- Melhor eficiência de performance. Uma única consulta traz todos os dados do documento;
- Possui escalabilidade e flexibilidade na manipulação de grandes massas de dados;
- Facilidade de migração dos dados de banco de dados relacionais.

6.4. Estrutura de armazenamento

A estrutura de armazenamento de dados do MongoDB pode ser dividida em 3 níveis. No primeiro nível encontramos a instância do banco de dados Mongo. No segundo nível encontramos as coleções e no terceiro nível temos os documentos.

A Figura 18 ilustra a estrutura de armazenamento do MongoDB.

Figura 18 - Estrutura de armazenamento.



Fonte: Elaborada pelo autor.

Podemos perceber que cada banco de dados pode conter uma ou várias coleções, cada coleção, por sua vez, possui documentos. Se fizermos um comparativo entre a estrutura do banco de dados relacional, as coleções seriam as tabelas e os documentos as linhas.

6.5 Comandos básicos para consultas no MongoDB

A seguir estão alguns comandos de manipulação de dados utilizados no MongoDB.

Criando um banco de dados:

```
> Use dbIGTI
```

```
>> switched to db dbIGTI
```

Criando uma coleção chamada cliente:

```
>> db.createcollection("forum_debates")
```

Inserindo um documento:

```
>> db.forum_debates.insert( {nome: "Leandro Lessa",  
postagem: "Bom trabalho pessoal"})
```

Inserindo lista de documentos:

```
>> db.forum_debates.insert([  
  
  {nome: "Leandro Lessa", post: "bom trabalho pessoal"},  
  
  {nome: "Leandro Lessa", post: "Trabalho Prático atualizado"},  
  
  {nome: "Daniele Lessa", post: "Tive dúvidas na resolução do  
trabalho"}  
  
  ])
```

Recuperar todos os documentos da coleção clientes:

```
>> db.forum_debates.find()
```

Recuperar o primeiro elemento da coleção clientes:

```
>> db.forum_debates.findOne()
```

Retorna os documentos de forma estruturada:

```
>> db.forum_debates.find().pretty()
```

Ler documentos específicos:

```
>> db.forum_debates.find({nome:"Leandro Lessa"})
```

Ler documentos com condição AND (basta separar as condições por vírgula):

```
>> db.forum_debates.find({nome: "Leandro Lessa", post: "Trabalho Prático atualizado"})
```

Operadores lógicos para realização de consultas:

\$eq = igual

\$gt = maior que

\$gte = maior ou igual que

\$lt = menor que

\$lte = menor ou igual que

\$ne = diferente de

\$in = contém

\$nin = Não contém

Consulta baseada em operadores lógicos:

Vamos utilizar o operador menor ou igual que **\$lte**.

Para entender o uso do operador, vamos adicionar registros em uma nova coleção.

```
>> db.createcollection("pessoa")

>> db.pessoa.insert([

    {nome: "Jose Riberio", idade: 25},

    {nome: "Ana Maria", idade: 22},

    {nome: "Joaquim do Amaral ", idade: 21}

])
```

```
>>db.pessoa.find({idade: {$lte: 22}})
```

Desta forma, se recupera todos os documentos na coleção pessoa com idade menor ou igual a 22 anos.

Ler documentos com condição OR:

```
>> db.pessoa.find({$or:[ {nome: "Jose Ribeiro"}, {nome:
"Joaquim do Amaral "}]})
```

6.6. Empresas que utilizam o MongoDB

O banco de dados MongoDB, por possuir características de alto desempenho, ser flexível, ter confiabilidade e escala econômica, entre outras características, tem chamado atenção de muitas empresas no mundo inteiro. Muitas dessas empresas têm migrado as soluções de seus negócios para o armazenamento no MongoDB.

Podemos citar a *SEGA Hardlight* que migrou para o servidor MongoDB a fim de simplificar as operações e melhorar a experiências de milhões de jogadores móveis. A *SAP* utiliza o MongoDB como banco central de dados que sustenta o sistema de gerenciamento de conteúdo da plataforma de serviços da empresa.

Para mais exemplos de aplicações do MongoDB nas empresas, acesse: <https://www.mongodb.com/who-uses-mongodb>.

A Figura 19 ilustra algumas empresas que já utilizam o MongoDB como banco de dados em seus negócios.

Figura 19 - Empresas que utilizam o MongoDB.



Fonte: <https://www.mongodb.com/who-uses-mongodb>.

Referências

ABITEBOUL, Serge. Querying Semi-Structured. In: *Data. Proc. ICDT 97*, 1970.

BELEI, Renata Aparecida et al. *O Uso de Entrevista, Observação e Videogravação em Pesquisa Qualitativa*. 2008. Disponível em: <<https://periodicos.ufpel.edu.br/ojs2/index.php/caduc/article/view/1770>>. Acesso em: 12 ago. 2021.

BERNERS-LEE, Tim; HENDLER, James; LASSILA, Ora. The Semantic Web: A New Form of Web Content That is Meaningful to Computers Will Unleash a Revolution of New Possibilities. In: *Scientific American*, 2001.

CARDOSO, Virgínia; CARDOSO, Giselle. *Linguagem SQL: fundamentos e práticas*. São Paulo: Saraiva, 2013. 196 p.

DOMO. Data Never Sleeps 8.0. 2020. Disponível em: <<https://www.domo.com/learn/data-never-sleeps-8>>. Acesso em: 12 ago. 2021.

DUBOIS, Paul. *MySQL: developer's library*. 4. ed. Pearson Education, 2008.

ELMASRI, Ramez; SHAMKANT, Navathe B. *Sistemas de Bancos de Dados*. Pearson, 2019.

GRUS, Joel. *Data Science do Zero: primeiras regras com o python*. Alta Books, 2016.

HAT, Red. *O que é API?*. 2021. Disponível em: <<https://www.redhat.com/pt-br/topics/api/what-are-application-programming-interfaces>>. Acesso em: 12 ago. 2021.

LESSA, Leandro Figueira. *Explorando Perfis Profissionais do LinkedIn para Recomendação de Cursos de Pós-Graduação*. Pontifícia Universidade Católica de Minas Gerais, 2019.

LÉVY, Pierre. *A inteligência coletiva: por uma antropologia do ciberespaço*. 5. ed. São Paulo: Edições Loyola, 2007.

MONGODB. *What Is MongoDB?*. Disponível em: <<https://www.mongodb.com/what-is-mongodb>>. Acesso em: 12 ago. 2021.

MYSQL. *MySQL: mysql workbench manual*. MySQL Workbench Manual. 2020. Disponível em: <<https://www.mysql.com/>>. Acesso em: 12 ago. 2021.

OLSEN, Wendy. *Coleta de Dados: debates e métodos fundamentais em pesquisa social*. Penso, 2015.

PALAKOLLU, Sri Manikanta. *Scrapy Vs Selenium Vs Beautiful Soup for Web Scraping*. Medium: Analytics Vidhya 2019. Disponível em: <<https://medium.com/analytics-vidhya/scrapy-vs-selenium-vs-beautiful-soup-for-web-scraping-24008b6c87b8>>. Acesso em: 12 ago. 2021.

PATEL, Hiren. *Web Scraping vs Web Crawling: What's the Difference?*. DZone, 2020. Disponível em: <<https://dzone.com/articles/web-scraping-vs-web-crawling-whats-the-difference>>. Acesso em: 12 ago. 2021.

PICKLER, Maria Elisa Valentim. Web Semântica: ontologias como ferramentas de representação do conhecimento. In: *Perspect. ciênc. inf.*, Belo Horizonte, v. 12, n. 1, p. 65-83, abr. 2007. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1413-99362007000100006&lng=en&nrm=iso>. Acesso em: 12 ago. 2021.

PYTHON, Software Foundation. *Virtual Environments and Packages*. 2021. Disponível em: <<https://docs.python.org/3/tutorial/venv.html>>. Acesso em: 12 ago. 2021.

SCRAPY. *Scrapy 2.4 documentation*. 2020. Disponível em: <<https://docs.scrapy.org/en/latest/>>. Acesso em: 12 ago. 2021.

SELENIUM. *The Selenium project and tools*. 2021. Disponível em: <https://www.selenium.dev/documentation/en/introduction/the_selenium_project_and_tools/>. Acesso em: 12 ago. 2021.

SOUP, Beautiful. *Documentação Beautiful Soup*. 2021. Disponível em: <<https://www.crummy.com/software/BeautifulSoup/bs4/doc.ptbr/>>. Acesso em: 12 ago. 2021.

SOUZA, Renato Rocha; ALVARENGA, Lídia. A Web Semântica e suas contribuições para a ciência da informação. In: *Ci. Inf.*, Brasília, v. 33, n. 1, p. 132-141, abr. 2004. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-19652004000100016&lng=en&nrm=iso>. Acesso em: 12 ago. 2021.

VELLOSO, Fernando de Castro. *Informática - Conceitos Básicos*. 8. ed. Rio de Janeiro: Elsevier, 2017.

W3C. *RDF: resource description framework*. Resource Description Framework. 2014. Disponível em: <<https://www.w3.org/RDF/>>. Acesso em: 12 ago. 2021.