

**CENTRO UNIVERSITÁRIO CARIOCA**

**ENGENHARIA DA COMPUTAÇÃO**

**YURI CONCEIÇÃO DE BRITO SCARANNI**

**UMA ABORDAGEM DIDÁTICA DO ECOSSISTEMA DA INFRAESTRUTURA DE CIÊNCIA DE DADOS**

**RIO DE JANEIRO**

**2021**

**YURI CONCEIÇÃO DE BRITO SCARANNI**

**UMA ABORDAGEM DIDÁTICA DO ECOSSISTEMA DA INFRAESTRUTURA DE CIÊNCIA DE DADOS**

Trabalho de Conclusão de Curso apresentado ao Centro Universitário Carioca, como requisito parcial para obtenção do grau de Bacharel em Engenharia da Computação.

Orientador: Prof. D. Sc. Sérgio Assunção Monteiro

RIO DE JANEIRO

2021

|  |
| --- |
| Yuri Conceição de Brito Scaranni  Uma Abordagem Didática do Ecossistema da Infraestrutura de Ciência de Dados / Yuri Conceição de Brito Scaranni. - Rio de Janeiro, 2021.  16 f.  Orientador: Sérgio Assunção Monteiro.  TCC (Graduação em Engenharia da Computação) – Centro universitário Unicarioca. Rio de Janeiro, 2021.  1. ETL. 2. Análise de dados. 3. Python. 4. Didático. 5. Bancos de dados. |

**YURI CONCEIÇÃO DE BRITO SCARANNI**

**UMA ABORDAGEM DIDÁTICA DO ECOSSISTEMA DA INFRAESTRUTURA DE CIÊNCIA DE DADOS**

Trabalho de Conclusão de Curso apresentado ao Centro Universitário Carioca, como requisito parcial para obtenção do grau de Bacharel em Engenharia da Computação.

BANCA EXAMINADORA

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Sérgio Monteiro Assunção, D.Sc - Orientador

Centro Universitário Carioca

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Neury Nunes Cardoso, D. Sc - Coordenador

Centro Universitário Carioca

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Alberto Tavares da Silva, D.Sc

Centro Universitário Carioca

**AGRADECIMENTOS**

Agradecemos a todos que me apoiaram desde o início deste sonho, aos meus amigos, aos meus professores, ao meu orientador prof. Sergio Monteiro Assunção por todo suporte no desenvolver deste trabalho e em especial a minha namorada e companheira de todos os momentos Evlyn Moraes.

**RESUMO**

A cada dia que passa observamos um aumento considerável no volume de dados que somos capazes de produzir. Seu armazenamento, antes com propósito único de gerar um histórico, ganhou mais significado quando o ramo de análise de dados cresceu e pôde mostrar informações extremamente úteis na tomada de decisão que antes passavam despercebidas. No meio desse processo, entre a geração do dado e a compreensão do mesmo fica a infraestrutura responsável por organizar esses dados se utilizando de processos automatizados para isso e é neste ponto que profissionais voltados para a ciência de dados se destacam, sendo os encarregados pela montagem de toda a engenharia envolvida no processo. Nesse contexto, o presente trabalho trata de explicar como as diversas ferramentas para processamento e análise de dados se contextualizam formando o ecossistema que viabiliza desenvolver aplicações para ciência de dados. Faremos uma abordagem tanto teórica, como prática.

**Palavras chave:** ETL, Análise de dados, Python, Didático, Bancos de dados.

**ABSTRACT**

With each passing day we observe a considerable increase in the volume of data we are able to produce. Its storage, previously with the sole purpose of generating a history, gained more significance when the data analysis industry grew and was able to show extremely useful information in decision making that previously went unnoticed. In the middle of this process, between data generation and understanding, is the infrastructure responsible for organizing this data using automated processes, and it is at this point that data science professionals stand out, being in charge of assembling all the engineering involved in the process. In this context, this paper aims to explain how the various tools for data processing and analysis are contextualized forming the ecosystem that enables the development of applications for data science. We will take a both theoretical and practical approach.

**Keywords:** ETL, Data analysis, Python, Didactic, Databases.

**LISTA DE ILUSTRAÇÕES**

[Figura 1 – Arquivo csv dos dados da covid19 17](#_Toc58093865)

[Figura 2 – Download ferramenta WAMP server 2](#_Toc58093866)0

[Figura 3 – WAMP server em execução na porta localhost 24](#_Toc58093867)

[Figura 4 – Página inicial do phpMyAdmin 24](#_Toc58093869)

[Figura 5 – Consulta a tabela criada dentro do banco de dados 24](#_Toc58093870)

[Figura 6 – Gráfico de mortes totais covid 25](#_Toc58093870)

[Figura 7 – Gráfico de mortes por milhão de habitantes 26](#_Toc58093870)

[Figura 8 – Gráfico de casos de covid 27](#_Toc58093870)

**LISTA DE CÓDIGOS**

[Código 1 – Implementação extração covid-19 ourworldindata 18](#_Toc58093865)

[Código 2 – Renomeando campos e excluindo colunas com Pandas 2](#_Toc58093866)0

[Código 3 – Criando base de dados 2](#_Toc58093866)2

[Código 4 – Inserindo arquivo em uma tabela no banco de dados 2](#_Toc58093867)3

[Código 5 – Código gráfico mortes totais por covid 2](#_Toc58093867)4

[Código 6 – Código gráfico mortes por milhão de habitantes 2](#_Toc58093867)6

[Código 7 – Código gráfico total de casos 2](#_Toc58093867)7

SUMÁRIO

[**1 INTRODUÇÃO** 12](#_Toc58367016)

[**1.1 Objetivo**](#_Toc58367019) 12

[**1.2 Organização**](#_Toc58367019) **12**

[**2 REVISÃO BIBLIOGRÁFICA**](#_Toc58367043) 13

[**3 INTRODUÇÃO A CIÊNCIA DE DADOS**](#_Toc58367044) 14

[**3.1 Banco de dados**](#_Toc58367045) 14

[3.1.1 SQL 1](#_Toc58367046)5

[3.1.2 MySQL 1](#_Toc58367046)5

[3.1.3 phpMyAdmin 1](#_Toc58367046)5

[3.1.4 WAMP 1](#_Toc58367046)5

[**3.2 Python**](#_Toc58367045) 16

[3.2.1 Pandas 1](#_Toc58367046)6

[3.2.3 Matplotlib 1](#_Toc58367046)6

[3.2.3 Seaborn 1](#_Toc58367046)7

[**3.3 ETL**](#_Toc58367045) 17

[3.3.1 Extract 1](#_Toc58367046)7

[3.3.3 Transform 1](#_Toc58367046)7

[3.3.3 Load 1](#_Toc58367046)8

[**4 ESTUDO DE CASO**](#_Toc58367044) 18

[**4.1 Integração com ETL**](#_Toc58367045) 18

[4.1.1 Extração dos dados 1](#_Toc58367046)8

[4.1.2 Transformação 1](#_Toc58367046)9

[4.1.3 Carga 2](#_Toc58367046)1

[**4.2 Análise de dados Gráfica**](#_Toc58367045) 24

[**4.3 Análise de dados com banco de dados**](#_Toc58367045) x

[**5 CONCLUSÃO**](#_Toc58367044) x

[**BIBLIOGRAFIA**](#_Toc58367044) x

# Introdução

**A ciência de dados surgiu nos anos 1970 como um ramo da tecnologia que combinava a estatística com métodos científicos e análise de dados com o objetivo de extrair informações a partir de dados. A ciência de dados teve um crescimento muito grande nos últimos anos, muito por conta do avanço tecnológico no que diz respeito a bancos de dados, poder de processamento e machine learning.**

**O avanço dessas tecnologias permitiu que o conhecimento sobre dados e toda a arquitetura envolvida em sua extração e análise fossem disseminados com maior rapidez de forma que hoje se tornou uma ciência indispensável em qualquer empresa ou governo.**

1.1 Objetivo

**O objetivo desse trabalho é mostrar de forma prática e didática toda a infraestrutura contida na ciência de dados a partir de ferramentas conhecidas e como o funcionamento em conjunto dessas ferramentas pode gerar insights que por sua vez ajudam na tomada de decisão.**

1.1 Organização

**O trabalho foi dividido em cinco capítulos, sendo este primeiro capítulo uma introdução ao tema juntamente com o objetivo e organização do trabalho.**

**O segundo capítulo mergulha na história da ciência de dados, descrevendo eventos relevantes que a popularizaram.**

**O terceiro capítulo visa fazer uma introdução as tecnologias que estão diretamente envolvidas com a ciência de dados além de descrever o seu funcionamento.**

**O quarto capítulo é um estudo de caso onde a utilizamos as ferramentas descritas no capítulo três para realizar uma análise e armazenamento das informações sobre o coronavírus no mundo.**

**O quinto capítulo apresenta as conclusões do trabalho.**

# Revisão bibliográfica

**Ciência de dados, *Data Science*, se tornou uma disciplina importante no cenário global nos últimos anos e composta por disciplinas que já eram consolidadas há tempos como estatística, bancos de dados, mineração e análise de dados (van der Aalst W. 2016), com a combinação desses elementos foi possível transformar uma imensa quantidade de dados em informações valiosas pra empresas, pessoas e a sociedade como um todo.**

**Com o avanço tecnológico o mundo produziu mais dados, um exemplo disso se dá no estudo da IDC Digital Universe Study, de abril de 2014 que estimava que a quantidade de informação digital armazenada em 2014 ultrapassava a barreira dos 4 Zettabytes e cresceria 10 vezes mais nos próximos seis anos. Um valor que é representado por 2^70 bytes, para termos de comparação, 1 MB representa 2^20 bytes (van der Aalst W. 2016). Em meio a essa grande quantidade de dados temos aqueles gerados no nosso dia a dia, como redes sociais, voos, estudos, pesquisas, etc., basicamente tudo, boa parte deles é tida como dados não estruturados, comumente conhecida como BIG DATA, e afim de tratar estes dados a ciência de dados emerge.**

**Definida como um campo interdisciplinar visando transformar dados em valores úteis. Dados podem ser estruturados ou não estruturados, grandes ou pequenos, estáticos ou transmitidos em tempo real. Estes valores que o dado gera podem vir em forma de predições, decisões automatizadas ou qualquer tipo de visualização que entregue *insights.* A ciência de dados inclui a extração, preparação, exploração, transformação, armazenamento e recuperação de dados, inclui também a infraestrutura computacional, vários tipos de mineração e a apresentação desses resultados considerando os aspectos éticos, sociais, legais e comerciais (van der Aalst W. 2016).**

**A ciência de dados tem sua evolução quando se destaca da estatística comum para ser algo além no que diz respeito a análise de dados, como diz John Wilder Tukey (químico, estatístico e topólogo que desempenhou um papel fundamental no desenvolvimento e no estudo da estatística em meados do século XX, campo este que se beneficiou enormemente de suas contribuições) em 1962 em "The Future of Data Analysis", “por muito tempo acreditei ser um estatístico, mas na realidade tinha um interesse maior por analisar dados do que fazer inferências estatísticas” (Tukey, 1962). Tukey menciona também que a análise de dados e as partes das estatísticas que a compõem devem assumir características da ciência e não da matemática, era o começo da disseminação da ciência de dados. Até que em 1974 Peter Naur (astrônomo e cientista da computação dinamarquês e vencedor do concurso 2005 A.M. Prêmio Turing, a maior homenagem em ciência da computação) publica "Concise Survey of Computer Methods" na Suécia e nos Estados Unidos. Neste livro Naur lista os principais métodos contemporâneos de processamento de dados que eram úteis para muitas aplicações. Em seu livro Naur utiliza o conceito de dados definido pela International Federation for Information Processing (IFIP) em "Guide to Concepts and Terms in Data Processing" que afirma que o dado é uma representação de fatos ou ideias de uma maneira formalizada, capaz de ser comunicada ou manipulada por algum processo. A partir deste conceito Naur utiliza em seu livro o termo 'ciência de dados' algumas vezes além de oferecer uma definição para a mesma: "A ciência de lidar com dados, uma vez que tenham sido estabelecidos, enquanto a relação dos dados com o que eles representam é delegada a outros campos e ciências"(Naur, 1974).**

**Com o passar dos anos a ciência de dados foi se consolidando no mundo a ponto de se tornar hoje um ponto chave para qualquer negócio, outras áreas do conhecimento dentro do ramo foram surgindo como o aprendizado de máquina, *Machine Learning*, e a análise preditiva, *Predictive analytics,* transformando a ciência de dados em um dos ramos mais promissores atualmente.**

# Introdução a Ciência de Dados

**Quando estamos falando de ciência de dados é comum acreditar que ela se trata apenas de estatística e análise, mas, além desses dois, outras ferramentas são extremamente importantes quando estamos analisando a infraestrutura da ciência de dados.**

3.1 Banco de dados

**Um banco de dados é uma coleção organizada de informações - ou dados - estruturados, geralmente alocados em um sistema de computador. Um banco de dados é geralmente controlado por um sistema de gerenciamento de banco de dados (DBMS). Juntos, os dados e o DBMS, juntamente com os aplicativos associados a eles, são chamados de sistema de banco de dados, geralmente abreviados para apenas banco de dados. Sendo assim o local onde armazenamos todo tipo de dado para ser consultado a qualquer momento, de qualquer lugar que consiga estabelecer uma conexão com o servidor no qual o banco de dados está. Dentre os sistemas gerenciais de bancos de dados (SGBD) mais famosos são Oracle, MySQL, SQL Server.**

3.1.1 SQL

**Structured Query Language, é uma linguagem de programação em bancos de dados padronizada e universal, já que serve para todos os SGBD, com pequenas variações de termos ou sintaxe entre eles. SQL é uma linguagem utilizada para consultar (essas consultas são chamadas de Query) e visualizar dados dentro do banco de dados, além de alterar estrutura de tabelas, criar bases de dados, inserir e atualizar valores, excluir tabelas e registros, em resumo, SQL é a linguagem que se comunica com o banco de dados executando todos os comandos dentro dele.**

3.1.2 MySQL

**Um dos Sistemas de gerenciamento de bancos de dados mais utilizados no mundo hoje, MySQL utiliza o SQL como sua linguagem como interface. Popularizou-se por ser uma aplicação do tipo open source e por ter uma fácil integração com muitas linguagens de programação.**

3.1.3 PhpMyAdmin

**PhpMyAdmin é uma ferramenta gratuita escrita na linguagem PHP utilizada para administração do banco de dados MySQL através da Web. Consegue prover uma interface gráfica que facilita a iteração com o banco de forma que é possível criar e deletar tabelas, bases de dados, além de conceder permissões e consultar tabelas.**

3.1.4 WAMP

**WAMP, também conhecido como WAMP server, é um acrônimo para Windows Apache MySQL Php (Para sistemas Linux é conhecido como LAMP – Linux Apache MySQL Php). Ele é um ambiente de desenvolvimento Web do Windows. Ele permite que você crie aplicativos web com Apache2, PHP e um banco de dados MySQL. WAMP possibilita a criação de um servidor local e executa automaticamente o banco MySQL na máquina, além disso ele já traz instalado o phpMyAdmin que é um cliente web para bancos de dados, desta forma é possível executar um sistema completo de banco de dados localmente.**

3.2 Python

**O Python é uma linguagem de programação de alto nível criada em 1991 por Guido van Rossum, simples e objetiva a linguagem se popularizou nos últimos anos por isso além da sua imensa flexibilidade, disponibilidade de bibliotecas e grande comunidade. Em Python é multiparadigma, seja ele funcional, orientado a objetos, estruturada, possui tipagem dinâmica e é open source.**

**Dentre as várias linguagens disponíveis Python se destaca no ramo da ciência de dados por ser robusto para programação de forma geral e possuir muitas bibliotecas de apoio para análise, tratamento e visualização de dados (ex.: Numpy, Pandas, Matplotlib e Scipy).**

**Com Python é possível fazer desde a extração, tratamento, até a inserção dos dados em um banco de dados, tornando uma linguagem extremamente poderosa.**

3.2.1 Pandas

**O Pandas é um dos pacotes mais utilizados no que diz respeito ao tratamento de dados, ele é um pacote Python que fornece estruturas de dados rápidas, flexíveis e expressivas projetadas para tornar o trabalho com dados estruturados (tabulares, multidimensionais, potencialmente heterogêneos) e de séries temporais fácil e intuitivo. O principal objetivo do pacote é ser o bloco de construção fundamental de alto nível para fazer análises de dados práticas e do mundo real em Python. Além disso, tem o objetivo mais amplo de se tornar a ferramenta de análise / manipulação de dados de código aberto mais poderosa e flexível disponível em qualquer idioma. O Pandas, portanto, colabora muito com milhares de métodos e funções para tratar e estruturar dados, facilitando a vida do programador.**

3.2.2 Matplotlib

**Matplotlib é uma biblioteca da linguagem Python utilizada na criação de gráficos e visualização de dados. Ela é uma biblioteca bastante conhecida pela comunidade da linguagem e devido a facilidade na criação e manipulação de gráficos se tornou extremamente popular sendo utilizada inclusive por outras bibliotecas como base.**

3.2.3 Seaborn

**Seaborn é uma biblioteca da linguagem Python muito utilizada na visualização de dados, ela é baseada em outra biblioteca, Matplotlib, com o diferencial de apresentar gráficos de alto nível de forma mais atrativa.**

3.3 ETL

**O ETL, processo de ETL, é o nome dado a um dos processos mais comuns de exploração de dados e integração de dados, onde o dado atravessa algumas etapas até que possa ser utilizado em algum DataWarehouse, banco de dados tradicional ou visualizador de dados.**

3.3.1 Extract

**Diz respeito a parte de extração de um ETL, a obtenção do dado em si. Várias ferramentas podem estar envolvidas variando de acordo com cada processo. A forma mais comum de se ver uma extração é através de uma linguagem de programação que se conecta com uma origem ou simplesmente faz o download dessa origem, dentre as linguagens mais conhecidas que realizam essa operação o Python é de longe a mais consolidada no processo, muito devido a sua facilidade em lidar com dados e seu tratamento além da imensa gama de bibliotecas que ele possui.**

3.3.2 Transform

**A transformação abrange toda a etapa de limpeza e manipulação do dado, uma vez que o mesmo pode estar em uma codificação ruim para leitura, com mais colunas do que as necessárias, não traduzido, etc. A tecnologia responsável pela transformação ainda é a linguagem de programação, geralmente também Python. Ao fim desta etapa é comum a geração de um novo arquivo pronto para leitura imediata ou inserção em uma base de dados.**

3.3.3 Load

**Nesta etapa, de carga, o dado extraído já está pronto para ser repassado ao seu destino para então ser consultado, a linguagem de programação continua atuando ao abrir conexão com bancos de dados para então fazer a inserção dos dados no banco, ou agindo para fazer upload do arquivo em alguma plataforma que disponibilize a leitura/consulta.**

# Estudo de caso

4.1 Integração com ETL

4.1.1 Extração dos dados

Um processo de ETL bem estruturado garante uma boa massa de dados, que por sua vez garantem uma análise mais real e melhor, este tipo de processo é mais utilizado para inserção dos dados em uma base para futuras análises. O primeiro passo neste tipo de integração é analisar bem a fonte dos dados e escolher a melhor forma de extrai-lo. Uma das formas mais comuns, e que utilizaremos neste trabalho, é a extração de arquivos direto da Web, para isso utilizaremos a linguagem Python com a biblioteca requests, instalada através do comando pip install requests,e a extração será dos dados da covid-19 disponibilizados diariamente em arquivo .csv pelo site *ourworldindata.org*, um site que disponibiliza dados empíricos sobre todo tipo de assunto e é mantido sob a tutela da Universidade de Oxford.

Código fonte 1 – Implementação extração covid-19 ourworldindata.

import requests

name\_dataset = 'covid19.csv'

source\_url = "https://covid.ourworldindata.org/data/owid-covid-data.csv"

print('>>>> Extração iniciada <<<<')

data = requests.get(source\_url)

if data.status\_code != 200:

print('Erro ao extrair!')

with open(f'download/{name\_dataset}', 'w', encoding='utf-8') as f:

f.write(data.text)

print('>>>> Extração finalizada <<<<')

Após a extração podemos ver o arquivo salvo na pasta ‘download’ do nosso projeto como mostra a Figura 1.

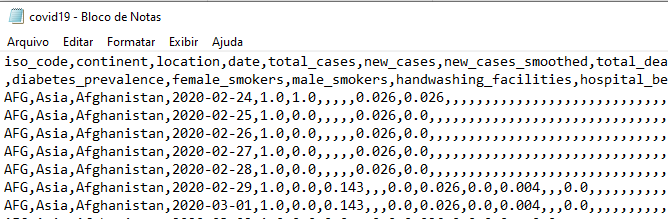


Figura 1 – Arquivo csv dos dados da covid19.

4.1.2 Transformação

A transformação dos dados é a etapa na qual devemos ter em mente aquilo que pretendemos carregar para nossa futura análise, alguns dados precisam ser tratados, traduzidos ou até removidos. Uma das ferramentas mais utilizadas para isso é a biblioteca Pandas do Python. Com a capacidade de transformar dados em uma estrutura organizada, denominado Dataframe, o Pandas, que pode ser instalado através do comando pip install pandas, é um aliado fantástico e leve para o tratamento de dados. Faremos alguns tratamentos nestes dados como traduzir o nome dos campos e excluir algumas colunas que não farão parte da análise, mantendo apenas as mais relevantes.

import pandas as pd

campos\_nome = {

"iso\_code":"codigo\_iso",

"continent":"continente",

"location":"pais",

"date":"data",

"total\_cases":"casos\_totais",

"new\_cases":"novos\_casos",

"total\_deaths":"mortes\_totais",

"new\_deaths":"novas\_mortes",

"total\_cases\_per\_million":"total\_casos\_por\_milhao",

"new\_cases\_per\_million":"novos\_casos\_por\_milhao",

"total\_deaths\_per\_million":"total\_mortes\_por\_milhao",

"new\_deaths\_per\_million":"novas\_mortes\_por\_milhao",

"reproduction\_rate":"indice\_reproducao",

"new\_tests":"novos\_testes",

"total\_tests":"total\_testes",

"total\_tests\_per\_thousand":"total\_testes\_por\_mil",

"new\_tests\_per\_thousand":"novos\_testes\_por\_mil",

"total\_vaccinations":"total\_vacinas",

"people\_vaccinated":"pessoas\_vacinadas",

"people\_fully\_vaccinated":"pessoas\_vacinadas\_dose\_completa",

"new\_vaccinations":"novas\_vacinas\_aplicadas",

"total\_vaccinations\_per\_hundred":"total\_vacinacao\_por\_centena",

"people\_vaccinated\_per\_hundred":"pessoas\_vacinadas\_por\_centena",

"people\_fully\_vaccinated\_per\_hundred":"pessoas\_vacinadas\_dose\_completa\_por\_centena",

"population":"populacao",

"population\_density":"densidade\_populacional",

"median\_age":"media\_idade",

"aged\_65\_older":"idade\_acima\_65",

"aged\_70\_older":"idade\_acima\_70",

"gdp\_per\_capita":"pib\_per\_capita",

"extreme\_poverty":"pobreza\_extrema",

"life\_expectancy":"expectativa\_vida"

}

df = pd.read\_csv(f'download/covid19.csv', sep=',', encoding='utf-8')

df = df.filter(campos\_nome.keys()) # Filtrando colunas

df = df.rename(columns=campos\_nome) # Renomeando colunas

df.to\_csv('download/covid19\_renamed.csv', sep=',', index=False)

Código fonte 2 – Renomeando campos e excluindo colunas com Pandas.

4.1.3 Carga

Após os dados estarem extraídos e tratados é o momento de inseri-los no banco de dados. O banco escolhido aqui é o MySQL, utilizado através do servidor local criado pela ferramenta WAMP server. A interface com os dados será feita pelo phpMyAdmin também disponibilizado pelo WAMP. Para efetuar a carga precisamos antes criar a base de dados e então através do Pandas iremos inserir os dados do arquivo em uma tabela gerada também pelo Pandas.

Então iremos baixar e executar o WAMP server, disponível em *wampserver.com/en/*, como mostra a Figura 2.

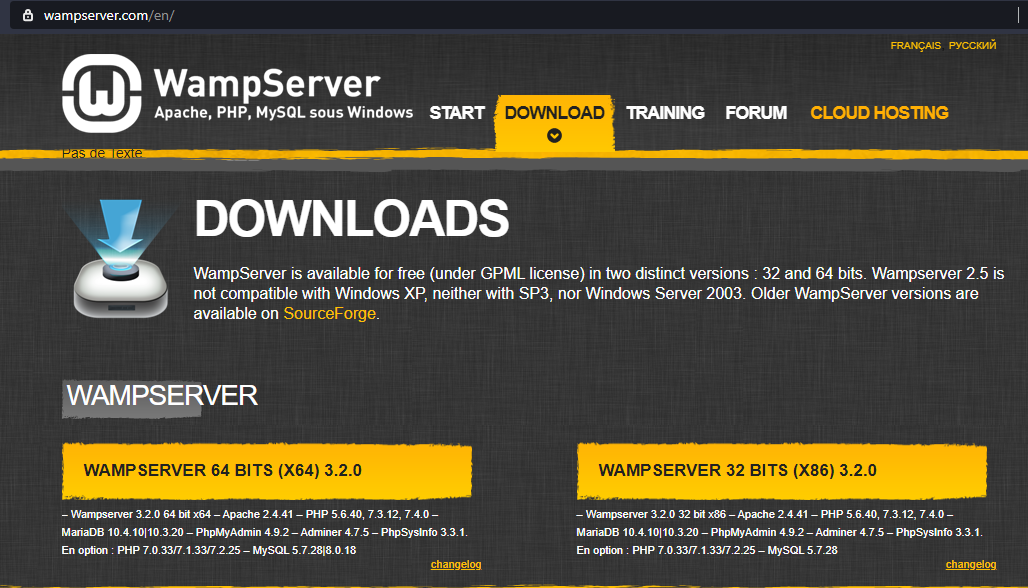


Figura 2 – Download ferramenta WAMP server.

Terminado o download e a instalação, o WAMP server, figura 3, está pronto para administrar nosso servidor local de MySQL.

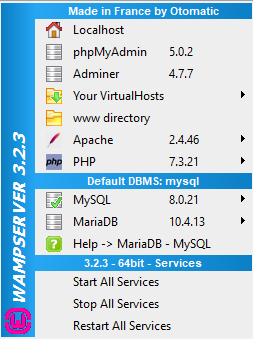


Figura 3 – WAMP server em execução na porta localhost.

Agora podemos acessar pelo browser a URL *localhost/phpmyadmin/* e fazer login na base de dados MySQL. Por padrão o login de acesso é ***root*** e o password em branco. Na aba superior é possível ver as opções gerais de administração e escrita de SQL e a esquerda as bases de dados existentes, como na figura 4.



Figura 4 – Página inicial do phpMyAdmin.

Na parte superior em SQL, podemos escrever nosso primeiro código SQL para criar uma nova base de dados que abrigará nossa tabela com os dados que geramos nos passos de extração e transformação.

CREATE DATABASE IF NOT EXISTS analise\_covid;

Código fonte 3 – Criando base de dados.

Com a base de dados criada é possível agora executar o script Python que realiza a inserção dos dados. A inserção é feita utilizando duas bibliotecas, Sqlalchemy, instalada através do comando pip install sqlalchemy, e a biblioteca PyMySQL, instalada através do comando pip install pymysql.

from sqlalchemy import create\_engine

import pandas as pd

# Dados de conexão local

conn = {'host': 'localhost', 'user': 'root', 'pass': '', 'database':'analise\_covid'}

# Criando conexão

engine = create\_engine(f'mysql+pymysql://{conn["user"]}:{conn["pass"]}@{conn["host"]}/{conn["database"]}')

# Abrindo arquivo de dados transformados

data = pd.read\_csv('download/covid19\_renamed.csv', sep=',', encoding='utf-8', index\_col=False)

# Insere os dados

data.to\_sql(name='covid\_19', con=engine, schema=conn["database"], if\_exists='replace', index=False)

Código fonte 4 – Inserindo arquivo em uma tabela no banco de dados.

Com a carga dos dados feita é possível realizar consultas SQL e analisa-los através do phpMyAdmin, como mostra a figura 5.

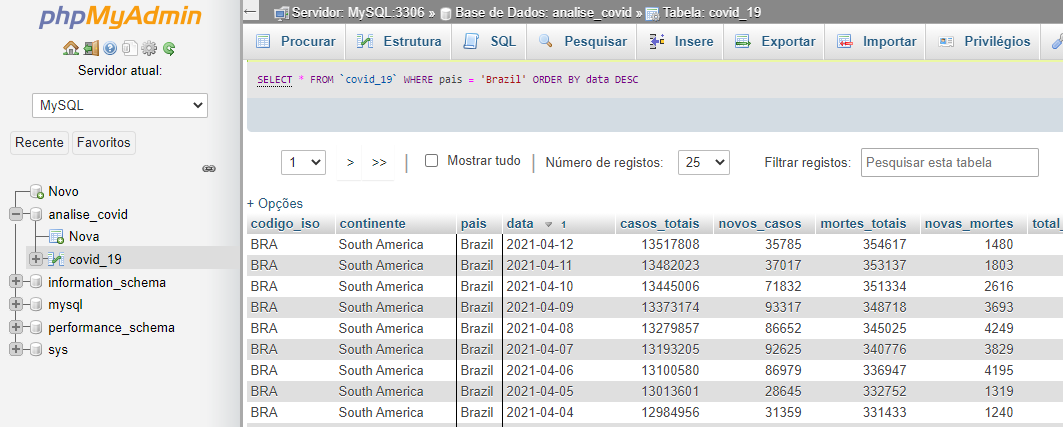


Figura 5 – Consulta a tabela criada dentro do banco de dados.

4.1.3 Análise de dados via cliente SQL

A análise dos dados inseridos em um banco de dados é extremamente comum e vital em profissões ligadas a ciência de dados e para isso é necessário a utilização da linguagem SQL para que seja possível obter algumas análises interessantes em cima dos dados. Agora que temos nossos dados dentro da base de dados MySQL, o que precisamos fazer é utilizar o editor de texto SQL do PhpMyAdmin para realizar nossas consultas, ou querys.

Para começar, iremos até nosso PhpMyAdmin e iremos explorar os dados que lá estão, abrindo o editor de SQL e inserindo a query:

SELECT

data

,pais

,casos\_totais

,mortes\_totais

FROM

analise\_covid.covid\_19

WHERE

data = (CURRENT\_DATE - INTERVAL(1) DAY)

AND continente IS NOT NULL

ORDER BY 3 DESC

LIMIT 10;

Código fonte 5 – Query para top 10 países no ranking de casos por covid.

Esta query trará como resultado os países com maior número de casos e mortos por covid no até um dia antes da data atual, se limitando apenas a países e ignorando continentes e ordenando pelo número de casos como indica a figura 6.

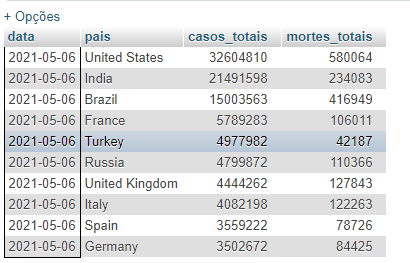


Figura 6 – Retorno da consulta feita sobre a tabela covid19.

Uma análise através do banco nos dá de forma simples e rápida valores exatos e uma noção mais realista dos dados, nesta query a seguir faremos uma análise temporal, olhando a média de novos casos de covid em cada região(continente) ignorando os valores globais (World e International).

SELECT

DATE\_FORMAT(data,'%Y-%m') AS Data,

pais AS Pais,

ROUND(AVG(novos\_casos), 2) AS media\_novos\_casos\_por\_continente

FROM

analise\_covid.covid\_19

WHERE

continente IS NULL

AND pais NOT IN ('World', 'International')

GROUP BY 2,1

ORDER BY 1 DESC

Código fonte 6 – Query de média de novos casos por continente.

O que podemos notar a partir dos dados é que atualmente a região mais controlada é a Oceania seguida da África, com menor índice de novos casos em maio de 2021, em contrapartida, Ásia e América do Sul estão entre os piores (figura 7), o que vai diretamente de encontro com o que acompanhamos no mundo atualmente, onde os EUA, embora líderes de casos, controlaram a doença e não apresentam grande incidência de novos casos enquanto Brasil e Índia que completam o pódio elevam os números devido a situação não controlada que os dois países vivenciam.

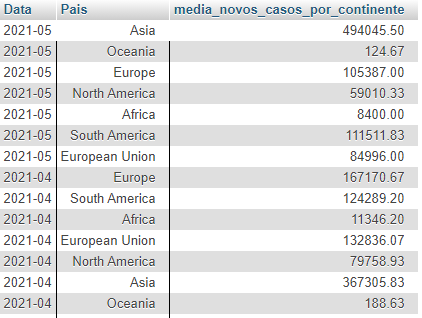


Figura 7 – Média de novos casos mensal por região (continente)

Este é apenas um pequeno exemplo do tipo de insight que olhar os dados de forma direta traz, e a grande vantagem por trás desse método de análise é a praticidade que a linguagem SQL trás, poder manipular dados de diversas formas e com inúmeras funções torna a análise numérica simples e direta.

4.2 Análise de dados gráfica com Python

A análise através de gráficos da uma dimensão mais visual dos dados que lidamos e muitas vezes permitem insights mais diretos, apenas ao olhar já é possível formular conclusões. Neste sentido o Python é de longe a melhor linguagem para realizar este tipo de trabalho pois a linguagem possui muitas bibliotecas de visualização extremamente poderosas como o Seaborn, Matplotlib entre outras.

As análises gráficas que se seguem neste tópico fazem uso do arquivo csv com dados sobre a covid19 disponibilizado pela ourworldindata.org já previamente extraído e transformado no tópico anterior. Para isso utilizaremos as seguintes bibliotecas.

import seaborn as sns

from datetime import timedelta

import pandas as pd

import matplotlib.pyplot as plt

Código fonte 8 – Importação de bibliotecas python

Mortes totais:

Nesta análise iremos olhar dentro da nossa massa de dados por colunas que representem o índice de mortes totais. Organizando pela quantidade de óbitos totais, para uma demonstração mais visual usaremos os primeiros 50 resultados.

def mortes\_covid\_top\_50():

# Abrindo arquivo e estabelecendo 'data' como tipo datetime

df = pd.read\_csv('download/covid19\_renamed.csv', sep=',')

df['data'] = pd.to\_datetime(df['data'])

# Extraindo somente o último dia disponível

yesterday = max(df['data']) - timedelta(days=1)

df = df.loc[df['data'] == yesterday]

# Removendo linhas onde o continente

# Não está preenchido

df = df.dropna(subset=['continente'])

# Ordenando pelo campo 'mortes\_totais' e limitando

# Aos 50 primeiros registros

df = df.sort\_values(by='mortes\_totais', ascending=False).head(50)

# Plotagem do gráfico de barras

sns.set\_theme(style="darkgrid")

palette = sns.color\_palette("Wistia", 50)

sns.barplot(data=df,

x='pais',

y='mortes\_totais',

palette=palette)

plt.xlabel("Países")

plt.ylabel("Mortes totais")

plt.title("Ranking mortes covid")

plt.xticks(rotation=90)

plt.show()

Código fonte 7 – Código gráfico mortes totais por covid.

Com a execução do código fonte 7 obtemos este gráfico da figura 9.

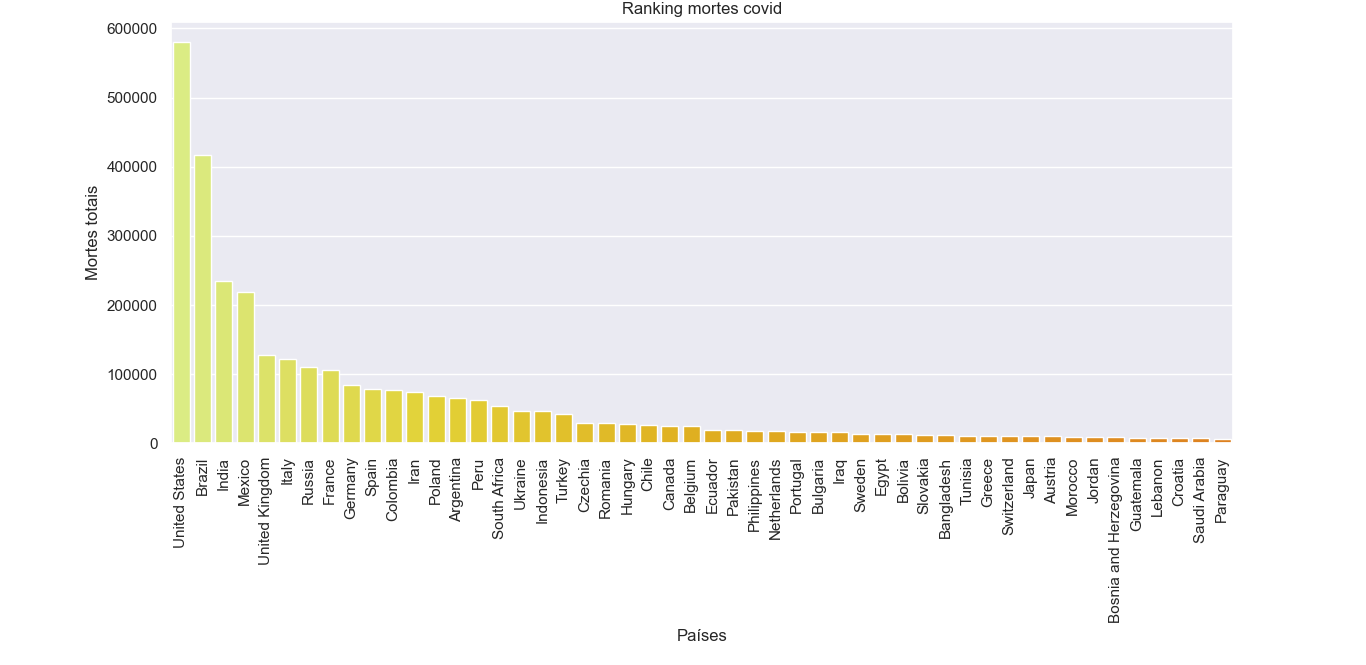


Figura 9 – Gráfico de mortes totais covid.

De forma bem clara é possível enxergar toda a situação global em um instante. E ver quais são os países mais afetados pela.

Recordistas de casos de covid:

Nesta segunda análise voltamos nossa visão para os países com maior índice de infectados pelo covid19 ao longo dos meses, utilizamos na visualização apenas 5 países para ter uma noção mais nítida das linhas.

def casos\_totais\_ao\_longo\_do\_tempo():

# Abrindo arquivo e estabelecendo 'data' como tipo datetime

df = pd.read\_csv('download/covid19\_renamed.csv', sep=',')

df['data'] = pd.to\_datetime(df['data'])

# Removendo linhas onde o continente

# Não está preenchido

df = df.dropna(subset=['continente'])

# Selecionando paises que até a data válida

# mais próxima eram os top 5 recordistas de casos

yesterday = max(df['data']) - timedelta(days=1)

df\_drop = df.loc[df['data'] == yesterday]

df\_drop = df\_drop.sort\_values(by='casos\_totais', ascending=False).head(5)

df\_drop = df\_drop['pais'].drop\_duplicates()

df = df[df['pais'].isin(df\_drop)].sort\_values(by='data')

# Transformando data comum em apenas ano e mês

df['data'] = df['data'].dt.to\_period('M').apply(str)

# Plotagem do gráfico de linhas

sns.set\_theme(style="whitegrid")

grap = sns.lineplot(data=df, x='data', y='casos\_totais', hue='pais')

# Criando unidade Mi(milhões) no eixo Y

from matplotlib.ticker import FuncFormatter

f = lambda x, pos: f'{x / 10 \*\* 6:,.0f} Mi'

grap.yaxis.set\_major\_formatter(FuncFormatter(f))

plt.xlabel("Ano-Mês")

plt.ylabel("Total de casos")

plt.title("Linha temporal do total de casos de covid")

plt.show()

Código fonte 8 – Código gráfico infectados covid.

O resultado é o gráfico da figura 10 que mostra os 5 países recordistas em número de infectados.

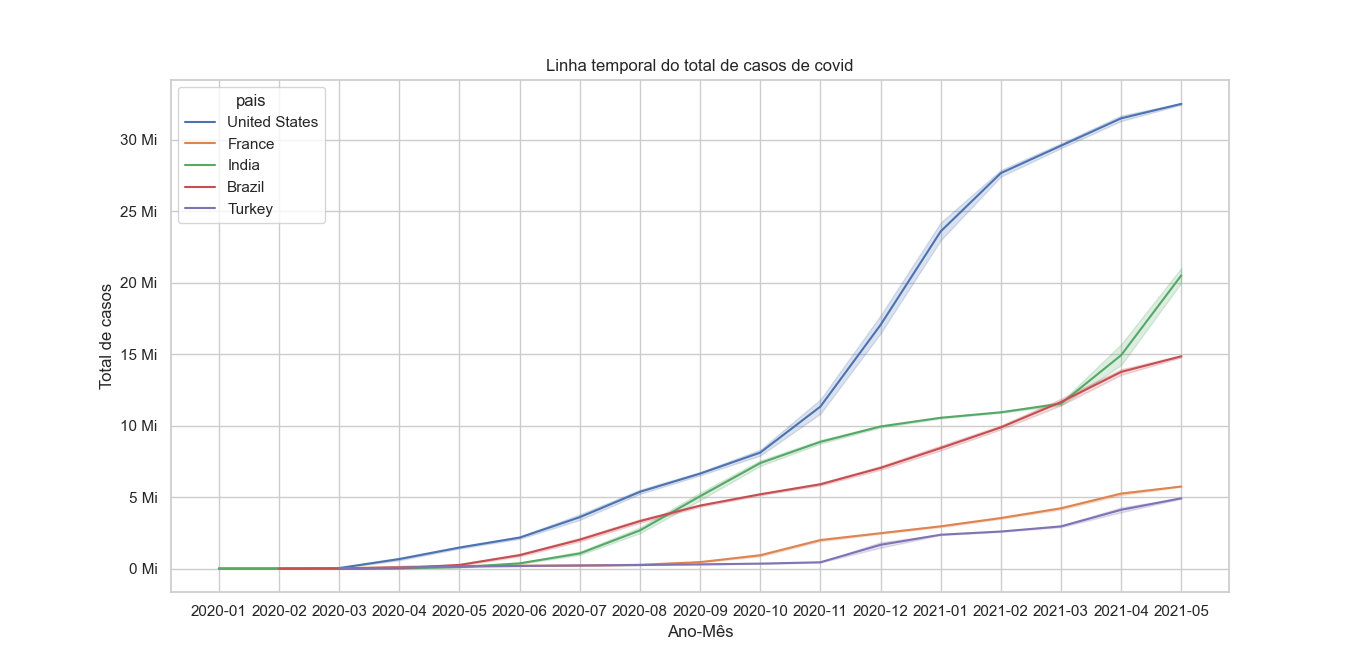


Figura 10 – Gráfico de total de infectados covid.

Um dado importante disso é que nem todos 5 primeiros no ranking de mortos estão entre os 5 primeiros no ranking de infectados. O que nos leva a elaboração de mais um gráfico para responder a este questionamento e nos levar a uma certeza, o número de infectados está ou não diretamente relacionado ao número de vítimas fatais?

Para isso iremos usar um gráfico de relação, que segue o código a seguir:

Código fonte 9 – Código gráfico relação mortes x infectados.

def relacao\_casos\_x\_mortes():

# Abrindo arquivo e estabelecendo 'data' como tipo datetime

df = pd.read\_csv('download/covid19\_renamed.csv', sep=',')

df['data'] = pd.to\_datetime(df['data'])

# Removendo linhas onde o continente

# Não está preenchido

df = df.dropna(subset=['continente'])

# Selecionando paises que até a data válida

# mais próxima eram os top 5 recordistas de casos

yesterday = max(df['data']) - timedelta(days=1)

df\_drop = df.loc[df['data'] == yesterday]

df\_drop = df\_drop.sort\_values(by='casos\_totais', ascending=False)

df\_drop = df\_drop['pais'].drop\_duplicates()

df = df[df['pais'].isin(df\_drop)].sort\_values(by='data')

# Filtrando colunas

yesterday = max(df['data']) - timedelta(days=1)

df = df.loc[df['data'] == yesterday]

df = df.filter(['data', 'pais', 'mortes\_totais', 'casos\_totais'])

# Plotando relação

sns.set\_theme(style="ticks")

grap = sns.scatterplot(data=df, x='mortes\_totais', y='casos\_totais')

# Criando unidade Mi(milhões) no eixo Y

from matplotlib.ticker import FuncFormatter

f = lambda x, pos: f'{x / 10 \*\* 6:,.0f} Mi'

grap.yaxis.set\_major\_formatter(FuncFormatter(f))

plt.xlabel("Total de mortos")

plt.ylabel("Total de casos")

plt.title("Linha temporal do total de casos de covid")

plt.show()

O resultado nos mostra que, sim, há uma relação linear entre o número de infectados e o número de mortos, há alguns pontos fora da curva, mas boa parte dos países onde o número de casos é baixo, o número de mortos também é e o efeito contrário acontece nos demais países, onde quanto maior o número de infectados maior o número de óbitos como mostra a figura 11. Isso nos traz a algumas conclusões como, evitar a infecção em massa é a medida fundamental já que a relação mortos x infectados se mostrou linear, conseguimos concluir também que países com populações maiores como EUA, Índia e Brasil sofreram mais com a doença e que há pontos em comum entre estes países que sofreram mais como notícias já divulgadas em 2021 sobre atrasos na vacinação e falta de cumprimento de medidas sanitárias para evitar a contaminação.

4.3 Análise de dados com ferramentas de Data Science

4.3.1 Databricks

Tem crescido no mercado de ciência de dados o número de ferramentas capaz de sintetizar tudo que o ramo precisa, são conhecidas como ferramentas de ETL ou Ambiente de Data Science, dentre eles se destaca o Databricks. O databricks é uma ferramenta completa voltada para cientistas e engenheiros de dados, cada vez mais utilizada nas grandes empresas o Databricks entrega agilidade, poder de computação e facilidade na hora de realizar as principais tarefas como extrair dados e plotar gráficos.

O Databricks tem seu funcionamento bem parecido com a ferramenta já conhecida Jupyter, através de notebooks é possível escrever blocos de código que são executados de forma independente, com o diferencial que o Databricks suporta a escrita de diversas linguagens de programação dentro do mesmo notebook, possibilitando a sinergia entre Python, SQL, Scala, Shell Scrip e Markdown.

Outro grande ponto positivo dentro do Databricks é sua grande escalabilidade devido o uso da engine Spark de forma nativa. Através do Spark o Databricks consegue realizar um processamento paralelo, o que acelera o desenvolvimento e programação já que grandes massas de dados podem ser lidas rapidamente.

O Databricks possui versões gratuita, Community, e paga, com isto, podemos explorar toda a capacidade desta excepcional ferramenta de forma gratuita.

# Conclusão

# Bibliografia

**van der Aalst W. (2016) Data Science in Action. In: Process Mining. Springer, Berlin, Heidelberg.** [**https://doi.org/10.1007/978-3-662-49851-4\_1**](https://doi.org/10.1007/978-3-662-49851-4_1)

**FORBES. A VERY SHORT HISTORY OF DATA SCIENCE. DISPONÍVEL EM: https://www.forbes.com/sites/gilpress/2013/05/28/a-very-short-history-of-data-science/?sh=1e363ef455cf. ACESSO EM: 13 MAR. 2021.**