

# Instituição Ada Tech - Programa Vem Ser Tech Dados 2023

## Projeto de Modelagem de Banco de Dados - Grupo 1

**Grupo 01:** Caio Vieira, Igor Cruz, Jared Leite, Leticia Carneiro, Sarah Rezende

**Professor:** Jorge Cristhian Chamby Díaz | **Módulo:** Banco de Dados | **Turma:** 1102

**GitHub:** [Projeto: Modelagem e Análise de Dados Veiculares](#)



---

## Projeto: Modelagem e Análise de Dados Veiculares

### Visão Geral do Projeto

Nosso projeto teve como objetivo demonstrar nossa experiência prática na criação de um banco de dados a partir de um cenário do mundo real, aplicando os conceitos aprendidos em modelagem de banco de dados e SQL. A capacidade de documentar e apresentar o trabalho de maneira clara e estruturada.

O projeto de **Modelagem e Análise de Dados Veiculares** tem como escopo informações relacionadas a veículos, especificamente valores de veículos obtidos por meio da tabela FIPE, Tabela de Informações Veiculares e Tabela de Salários Mínimos ao longo do

tempo. O processo de Extração, Transformação e Carga (ETL) de duas fontes de Dados foi realizado por meio de web scraping.

O propósito fundamental do nosso projeto é empregar consultas SQL para obter insights cruciais sobre a precificação e desvalorização de veículos. Estamos focados em identificar os carros de entrada, avaliando sua precificação em relação aos salários mínimos. Além disso, buscamos destacar os veículos que apresentam as maiores variações percentuais de preço, entre outros tópicos relevantes no contexto automotivo.

Por fim, nosso projeto reflete nosso comprometimento em aplicar conhecimentos teóricos na prática, destacando nossa habilidade em modelagem de banco de dados e análise de dados no contexto do mundo real.

## Dados de Entrada Originais

### Obtenção dos Dados

O processo de Extração, Transformação e Carga (ETL) foi conduzido por meio de *web scraping*<sup>1</sup>. A seguir, descreveremos o procedimento de obtenção dos dados para cada conjunto utilizado, empregando três fontes distintas de dados a “Tabela Fipe”, “Ficha Completa” e “Ipeadata”:

### 1ª Conjunto de Dados: Tabela FIPE - Carros Utilitários Pequenos

Desenvolvemos um [script](#) em Python para automatizar consultas no site da tabela FIPE. Este script permite a extração de dados detalhados sobre veículos, focando especificamente na categoria "*Consulta de Carros e Utilitários Pequenos*", fornecendo informações relevantes para análise. A implementação utiliza a biblioteca *Selenium* para interação automatizada com o site, enquanto o *BeautifulSoup* é empregado para análise HTML e extração de dados. A estrutura do código é organizada em métodos, simplificando a consulta e a manipulação eficiente das informações disponíveis na tabela FIPE relacionadas a veículos.

---

<sup>1</sup> **web scraping**: uma forma de mineração que permite a extração de dados de sites da web, convertendo-os em informação estruturada para posterior análise. Fonte: Wikipedia. Disponível em: [url - https://pt.wikipedia.org/wiki/Coleta\\_de\\_dados\\_web](https://pt.wikipedia.org/wiki/Coleta_de_dados_web). Acesso em [11/12/2023].



Imagem: Site da Tabela Fipe - Consulta De Carros E Utilitários Pequenos

Conduzimos o processo de Extração, Transformação e Carga (ETL) por meio de *web scraping*, visando extrair informações cruciais da fonte. Os dados extraídos englobam detalhes como mês de referência, código FIPE, marca, modelo, ano do modelo, autenticação, data da consulta e preço médio de comercialização.

IMPRIMIR            COPIAR URL	
Mês de referência:	novembro de 2019
Código Fipe:	017040-2
Marca:	Jeep
Modelo:	Renegade 1.8 4x2 Flex 16V Mec.
Ano Modelo:	2018 Gasolina
Autenticação	2x03dxdpnq5h
Data da consulta	quarta-feira, 27 de novembro de 2019 21:46
<b>Preço Médio</b>	<b>R\$ 62.406,00</b>

Imagem: Site Tabela Fipe

Destaca-se que a tabela atual, correspondente ao ano de 2023, apresenta aproximadamente 300 mil registros. Importante ressaltar que, para este trabalho, optamos por utilizar uma amostra específica, abrangendo 30 mil linhas, para análises mais específicas e eficientes.

Arquivo

Página Inicial

Inserir

Layout da Página

Fórmulas

Dados

Revisão

Exibir

Desenvolvedor

Ajuda

Colar

Formatar

Comentários

Compartilhamento

Calibri

11

A<sup>+</sup>

A<sup>-</sup>

Imagem: Site Tabela Fipec após a extração no Excel

## 2ª Conjunto de Dados: Informações veiculares - Ficha Completa

Foram criados diversos [scripts](#) em Python para realizar a extração de dados do site "[fichacompleta.com.br](https://fichacompleta.com.br)" por meio de web scraping. Utilizando a biblioteca *BeautifulSoup*, ele efetua uma série de requisições HTTP para URLs específicas, coletando informações cruciais como marcas, modelos e os links associados. Esse código é especialmente desenvolvido para coletar e organizar dados sobre veículos disponíveis no site mencionado.

Além disso, ao explorar a tabela de informações, implementamos um *web crawler*<sup>2</sup> para reunir todos os links relevantes. O web scraping, executado com a biblioteca BeautifulSoup, extraiu informações essenciais de cada página, contribuindo para uma coleta abrangente de dados.

Para evitar bloqueios automáticos e simular um comportamento humano durante a coleta de dados, incorporamos pausas no script usando a função *sleep* entre as requisições. Isso não apenas permitiu o acesso eficaz às informações desejadas, mas também ajudou a manter a conformidade com as políticas do site, promovendo uma abordagem ética no processo de web scraping.

<sup>2</sup> **Web Crawler:** é um [programa de computador](#) que navega pela [rede mundial](#) de uma forma metódica e automatizada. Fonte: Wikipedia. Disponível em: [url - https://pt.wikipedia.org/wiki/Rastreador\\_web](https://pt.wikipedia.org/wiki/Rastreador_web). Acesso em [11/12/2023]

### 3ª Conjunto de Dados: Ipeadata - Salário Mínimo

Obtivemos o conjunto de dados públicos relacionado aos salários mínimos no site <http://www.ipeadata.gov.br/>.

### Tratamento dos dados

#### Tratamento dos Dados da 2ª tabela - Informações dos Veículo

O web scraping, apesar de ser uma técnica poderosa para a extração de dados da web, apresenta diversos desafios que podem dificultar a obtenção de informações organizadas e prontas para uso.

Em nosso trabalho, enfrentamos esses desafios ao realizar web scraping para obter dados relacionados a tabela de informações do site "[fichacompleta.com.br](http://fichacompleta.com.br)". Fomos confrontados com a necessidade de adaptar nosso código para lidar com mudanças na estrutura do site e contornar bloqueios automáticos. O processo envolveu não apenas a extração inicial, mas também o tratamento e organização dos dados extraídos para garantir sua utilidade em análises subsequentes. Este esforço adicional foi crucial para garantir que os dados obtidos fossem confiáveis e efetivamente integrados às nossas análises.

Desenvolvemos um código que processa a coluna '**TechInfos**', extraindo as informações desejadas e gerando novas colunas específicas (**configuração, combustível, lugares, portas, porte, procedência e propulsão**). Posteriormente, removemos as colunas '**TechInfo**' e '**Equipamentos**'.

Nosso código em Python utiliza a biblioteca pandas para manipular e extrair informações da tabela *tech\_infos\_amostra.csv*. Após definir quais colunas seriam editadas, o código extraiu informações específicas da coluna escolhida *TechInfos* e criou novas colunas no *DataFrame* para cada informação extraída. Depois removeu as colunas originais (*TechInfos* e *Equipaments*) que continham informações agora armazenadas em colunas separadas. O código também realizou a alteração dos nomes das colunas "Brands", "Famillys", "Years" e "Models" que estavam em língua estrangeira para "Marca", "Família",

"Ano\_Modelo" e "Modelo". Exibindo assim o DataFrame resultante, agora com colunas adicionais derivadas da coluna original *TechInfos*.

```
import pandas as pd

# Carregando o arquivo CSV
TechInfos = 'tech_infos_amostra.csv'
df = pd.read_csv(TechInfos)

# Escolher o nome da coluna desejada
coluna_a_editar = 'TechInfos'
coluna_a_editar2 = 'Equipments'

# Extrair informações relacionadas às palavras-chave específicas
df['Configuração'] = df[coluna_a_editar].str.extract(r'Configuração:(.*?);')
df['Combustível'] = df[coluna_a_editar].str.extract(r'Combustível:(.*?);')
df['Lugares'] = df[coluna_a_editar].str.extract(r'Lugares:(.*?);')
df['Portas'] = df[coluna_a_editar].str.extract(r'Portas:(.*?);')
df['Porte'] = df[coluna_a_editar].str.extract(r'Porte:(.*?);')
df['Procedência'] = df[coluna_a_editar].str.extract(r'Procedência:(.*?);')
df['Propulsão'] = df[coluna_a_editar].str.extract(r'Propulsão:(.*?);')

# Excluir coluna original
df.drop(columns=[coluna_a_editar], inplace=True)
df.drop(columns=[coluna_a_editar2], inplace=True)

# Renomeando as colunas
df.rename(columns={'Brands': 'Marca', 'Famyls': 'Familia', 'Years': 'Ano_Modelo', 'Models': 'Modelo'},
          inplace=True)

# Exibir as informações resultante
print("\nDataFrame com informações mantidas:")
display(df)

tech_info_novo = 'tabela_tratada.csv'
```

Código em Python para o tratamento de dados da tabela de Informações.

### A tabela de informações veiculares antes do tratamento:

Observa-se abaixo que a coluna 'TechInfo' na tabela contém uma variedade de informações, resultando em um conjunto de dados desorganizado. O objetivo foi extrair apenas informações específicas, criar novas colunas relevantes para o projeto e remover as antigas que não contribuem para os objetivos estabelecidos.



```

Conteúdo da Coluna 'TechInfos':
0      FIPE 06/23:R$ 55.264,00;Ano:2016;Combustível:F...
1      FIPE 06/23:R$ 67.561,00;Ano:2014;Combustível:F...
2      FIPE 06/23:R$ 203.553,00;Ano:2020;Combustível:...
3      Ano:2003;Combustível:Diesel;Configuração:Picap...
4      FIPE 07/22:R$ 91.309,00;Ano:2011;Combustível:G...
...
82     FIPE 06/23:R$ 12.801,00;Ano:2001;Combustível:G...
83     FIPE 06/23:R$ 522.894,00;Ano:2022;Combustível:...
84     FIPE 06/23:R$ 105.213,00;Ano:2003;Combustível:...
85     Ano:2016;Combustível:Gasolina;Configuração:Sed...
86     FIPE 06/23:R$ 38.912,00;Ano:2005;Combustível:G...
Name: TechInfos, Length: 87, dtype: object

```

Imagem: conteúdo da coluna “TechInfos” antes do tratamento dos dados

	Brands	Familys	Years	Models	TechInfos	Equipments
0	Ford	Fiesta	2016	Fiesta TITANIUM 1.6 16V Flex Mec.	FIPE 06/23:R\$ 55.264,00;Ano:2016;Combustível:F...	Ajuste do volante em altura;Ajuste do volante ...
1	Honda	Civic	2014	Civic Sedan LXR 2.0 Flexone 16V Aut. 4p	FIPE 06/23:R\$ 67.561,00;Ano:2014;Combustível:F...	Ajuste do volante em altura;Ajuste do volante ...
2	Audi	A3 Sedan	2020	A3 Sedan Performance 2.0 TFSI S-tronic	FIPE 06/23:R\$ 203.553,00;Ano:2020;Combustível:...	Ajuste do volante em altura;Ajuste do volante ...
3	Ford	Ranger	2003	Ranger XLT 2.8 8v 135cv 4x4 CD TB Diesel	Ano:2003;Combustível:Diesel;Configuração:Picap...	NaN
4	Audi	Q7	2011	Q7 3.0 V6 TFSI Quat.Tip.5p/ Perf.(Hib.)	FIPE 07/22:R\$ 91.309,00;Ano:2011;Combustível:G...	NaN

Imagem: conteúdo da tabela de informações veiculares com destaque da coluna “TechInfos” antes do tratamento dos dados

### A tabela de informações veiculares depois do tratamento:

	Marca	Familia	Ano_Modelo	Modelo	Configuração	Combustível	Lugares	Portas	Porte	Procedência	Propulsão
0	Ford	Fiesta	2016	Fiesta TITANIUM 1.6 16V Flex Mec.	Hatch	Flex	5	4	Compacto	Nacional	Combustão
1	Honda	Civic	2014	Civic Sedan LXR 2.0 Flexone 16V Aut. 4p	Sedã	Flex	5	4	Médio	Nacional	Combustão
2	Audi	A3 Sedan	2020	A3 Sedan Performance 2.0 TFSI S-tronic	Sedã	Gasolina	5	4	Médio	Nacional	Combustão
3	Ford	Ranger	2003	Ranger XLT 2.8 8v 135cv 4x4 CD TB Diesel	Picape	Diesel	5	4	Médio	Importado	Combustão
4	Audi	Q7	2011	Q7 3.0 V6 TFSI Quat.Tip.5p/ Perf.(Hib.)	SUV	Gasolina	7	4	Grande	Importado	NaN

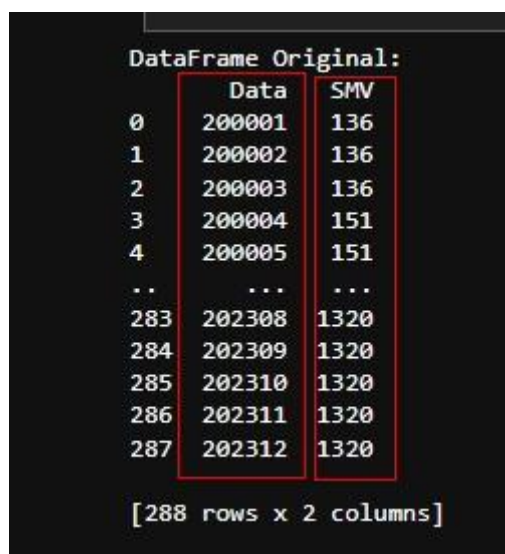
Imagem: conteúdo da tabela de informações veiculares com após o tratamento dos dados

Após o tratamento da tabela utilizando o código em Python e a criação de novas colunas a partir dos dados da coluna 'TechInfos', obtivemos uma tabela mais organizada e funcional.

### Tratamento da 3ª Tabela - Ipeadata (Salário Mínimo)

Aprimoramento da qualidade dos dados, considerando a presença de informações anteriores ao período do Plano Real. Não foi necessário realizar a normalização, sendo suficiente o tratamento de apenas duas colunas.

O objetivo principal do tratamento desta tabela foi efetuar a transformação dos dados na coluna “**Data**” para o formato **DATE**, enquanto na coluna “Salários Mínimos Vigentes” buscamos o tipo **int**.”



```

DataFrame Original:
  Data  SMV
0  200001  136
1  200002  136
2  200003  136
3  200004  151
4  200005  151
..    ...  ...
283 202308 1320
284 202309 1320
285 202310 1320
286 202311 1320
287 202312 1320
[288 rows x 2 columns]

```

Imagem: visualização da tabela “Ipeadata” antes do tratamento

Incluímos '01' ao final dos valores na coluna “**Data**” para permitir a conversão eficiente dos dados para o formato **DATE**. Após a conclusão da transformação, renomeamos a coluna 'SMV' para '**Salário Mínimo Vigente**'.



```

import pandas as pd

# Arquivo CSV
arquivo_csv = 'Pasta1.CSV'
df = pd.read_csv(arquivo_csv, sep=';')

# Adiciona "01" ao final de cada valor na coluna 'Data'
df['Data'] = df['Data'].apply(lambda x: str(x) + '01')

# Converte a coluna 'Data' para o tipo de dados datetime
df['Data'] = pd.to_datetime(df['Data'], format='%Y-%m-%d')

# Converte a coluna 'SMV' para o tipo de dados integer
df['Salario_Minimo_Vigente'] = pd.to_numeric(df['SMV'], errors='coerce', downcast='integer')

# Remove a coluna antiga 'SMV'
df.drop(columns=['SMV'], inplace=True)

# Exibe o DataFrame com as colunas 'Data' e 'Salario_Minimo_Vigente' modificadas
print(df)

```

Imagem: visualização do script que trata a tabela “ipeadata”

A “Ipeadata” após o tratamento:

	Data	Salario_Minimo_Vigente
1	2000-01-01	136
2	2000-02-01	136
3	2000-03-01	136
4	2000-04-01	151
5	2000-05-01	151
6	2000-06-01	151
7	2000-07-01	151
8	2000-08-01	151
9	2000-09-01	151
10	2000-09-01	151

Imagem: visualização da tabela “Ipeadata” depois do tratamento

## Modelagem Feita

Utilizamos os princípios aprendidos em aula referentes à primeira, segunda e terceira formas normais para aprimorar a modelagem do nosso banco de dados. Este processo incluiu a aplicação de técnicas específicas para garantir a eficiência e consistência da estrutura de dados. Para documentar e visualizar essas modificações, empregamos uma tabela no formato Excel, a qual está disponível para [acesso](#) no nosso repositório do GitHub. Esse documento proporciona uma visão transparente das transformações realizadas, contribuindo para a compreensão detalhada da evolução do design do banco de dados.

	A	B	C	D	E	F	G	H	I	J	K	L
15		jan/23	038001-6	Acura	NSX 3.0	1994 Gasolina	vmtpw1wz6w	domingo, 28 de maio de 2023 08:5	R\$ 43.113,00			
16		jan/23	038001-6	Acura	NSX 3.0	1993 Gasolina	t20vg81yyj	domingo, 28 de maio de 2023 08:5	R\$ 41.680,00			
17		jan/23	038001-6	Acura	NSX 3.0	1992 Gasolina	tdbfhagmpx	domingo, 28 de maio de 2023 08:5	R\$ 39.830,00			
18												
19												
20		1FN										
21				depende spensz de	depende spensz de		depende de	depende de	depende de			
22		Mes_Ref [PK]	Cod_FIPE [PK]	Marca	Modelo	Ano_Modelo [PK]	Autenticacao	Data_Consulta	Preco_Medio	Combustivel [PK]		
23		jan/23	038003-2	Acura	Integra GS 1.8	1992	gw9php06hn	domingo, 21 de maio de 2023 08:5	R\$ 12.109,00	Gasolina		
24		jan/23	038003-2	Acura	Integra GS 1.8	1991	gl13xf0qxr	domingo, 28 de maio de 2023 08:5	R\$ 11.150,00	Gasolina		
25		jan/23	038002-4	Acura	Legend 3.2/3.5	1998	nszj2bm6dz	domingo, 28 de maio de 2023 08:5	R\$ 27.689,00	Gasolina		
26		jan/23	038002-4	Acura	Legend 3.2/3.5	1997	mkip5rnkxz9	domingo, 28 de maio de 2023 08:5	R\$ 24.425,00	Gasolina		
27		jan/23	038002-4	Acura	Legend 3.2/3.5	1996	l2svd8p80x	domingo, 28 de maio de 2023 08:5	R\$ 23.153,00	Gasolina		
28		jan/23	038002-4	Acura	Legend 3.2/3.5	1995	k6blay14by	domingo, 28 de maio de 2023 08:5	R\$ 20.811,00	Gasolina		
29		jan/23	038002-4	Acura	Legend 3.2/3.5	1994	kvw3n0l3x5	domingo, 28 de maio de 2023 08:5	R\$ 19.921,00	Gasolina		
30		jan/23	038002-4	Acura	Legend 3.2/3.5	1993	iy4d21ayd0	domingo, 28 de maio de 2023 08:5	R\$ 17.553,00	Gasolina		
31		jan/23	038002-4	Acura	Legend 3.2/3.5	1992	jfm0mx3xwv	domingo, 28 de maio de 2023 08:5	R\$ 16.146,00	Gasolina		
32		jan/23	038002-4	Acura	Legend 3.2/3.5	1991	h55g6f172w	domingo, 28 de maio de 2023 08:5	R\$ 15.509,00	Gasolina		
33		jan/23	038001-6	Acura	NSX 3.0	1995	v66mfkhd30	domingo, 28 de maio de 2023 08:5	R\$ 44.691,00	Gasolina		
34		jan/23	038001-6	Acura	NSX 3.0	1994	vmtpw1wz6w	domingo, 28 de maio de 2023 08:5	R\$ 43.113,00	Gasolina		
35		jan/23	038001-6	Acura	NSX 3.0	1993	t20vg81yyj	domingo, 28 de maio de 2023 08:5	R\$ 41.680,00	Gasolina		
36		jan/23	038001-6	Acura	NSX 3.0	1992	tdbfhagmpx	domingo, 28 de maio de 2023 08:5	R\$ 39.830,00	Gasolina		
37												
38												
39		2FN										
40												
41		Mes_Ref [PK]	Cod_FIPE [PK/FK]	Ano_Modelo [PK]	Combustivel [PK]	Preco_Medio	Autenticacao	Data_Consulta		Cod_FIPE [PK]	Marca	Modelo
42		jan/23	038003-2	1992	Gasolina	R\$ 12.109,00	gw9php06hn	domingo, 21 de maio de 2023 08:57		038003-2	Acura	Integra GS 1.8
43		jan/23	038003-2	1991	Gasolina	R\$ 11.150,00	gl13xf0qxr	domingo, 28 de maio de 2023 08:53		038002-4	Acura	Legend 3.2/3.5
44		jan/23	038002-4	1998	Gasolina	R\$ 27.689,00	nszj2bm6dz	domingo, 28 de maio de 2023 08:53		038001-6	Acura	NSX 3.0
45		jan/23	038002-4	1997	Gasolina	R\$ 24.425,00	mkip5rnkxz9	domingo, 28 de maio de 2023 08:53				
46		jan/23	038002-4	1996	Gasolina	R\$ 23.153,00	l2svd8p80x	domingo, 28 de maio de 2023 08:53				
47		jan/23	038002-4	1995	Gasolina	R\$ 20.811,00	k6blay14by	domingo, 28 de maio de 2023 08:53				
48		jan/23	038002-4	1994	Gasolina	R\$ 19.921,00	kvw3n0l3x5	domingo, 28 de maio de 2023 08:53				
49		jan/23	038002-4	1993	Gasolina	R\$ 17.553,00	iy4d21ayd0	domingo, 28 de maio de 2023 08:53				

Aqui está o nosso diagrama ER (Entidade-Relacionamento), que representa o modelo físico do banco de dados. Ele destaca as entidades, seus atributos e os relacionamentos entre elas.

## Diagrama Entidade-Relacionamento

Projeto - Modelagem e Análise de Dados  
Veiculares | Grupo 1 | Turma 11021  
Módulo III Bancos de Dados

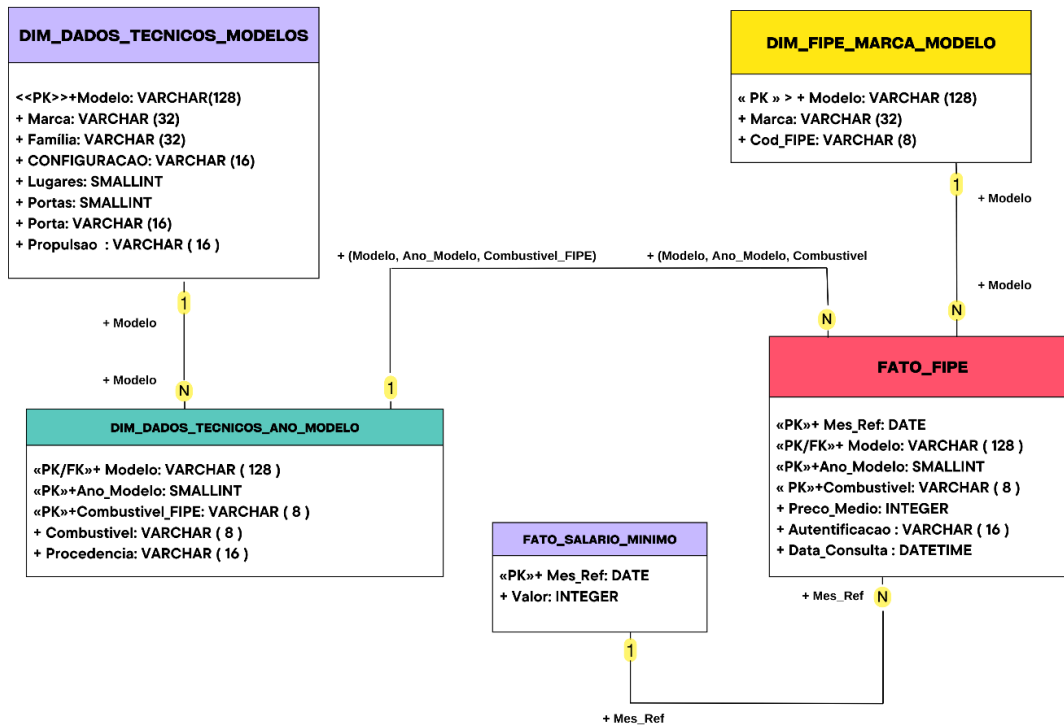


Imagem: visualização do diagrama Entidade-Relacionamento

## Modelo Entidade-Relacionamento (ER):

Abaixo está descrito o modelo detalhado de dados referente à representação estruturada do Banco de Dados, referente a informações relacionadas a modelos de veículos, preços, dados técnicos e salários mínimos. O modelo é composto por diversas entidades inter-relacionadas, cada uma representando aspectos específicos dos dados a serem armazenados.

### Entidade: DIM\_DADOS\_TECNICOS\_MODELOS

- **Atributos:**
  - **Modelo:** Identificador único do modelo técnico (VARCHAR(128)).

- **Marca:** Marca do veículo (VARCHAR(32)).
  - **Família:** Grupo ou família à qual o modelo pertence (VARCHAR(32)).
  - **Configuracao:** Configuração específica do modelo (VARCHAR(16)).
  - **Lugares:** Número de lugares no veículo (SMALLINT).
  - **Portas:** Numero de portas no veículo (SMALLINT).
  - **Porta:** Tipo de porta no veículo (VARCHAR(16)).
  - **Propulsao:** Tipo de propulsão do veículo (VARCHAR(16)).
- **Chave Primária:**
    - **Modelo:** Identificador único do modelo.
- **Cardinalidade:**
    - Relacionamento com "Dados Técnicos Modelos Ano Modelo" é de 1 para N (1:N).

#### **Entidade: DIM\_DADOS\_TECNICOS\_YEAR\_MODELO**

- **Atributos:**
  - **Modelo:** Identificador do modelo (VARCHAR(128)).
  - **Ano\_Modelo:** Ano específico do modelo (SMALLINT).
  - **Combustivel\_FIPE:** Código FIPE do combustível (VARCHAR(8)).
  - **Combustível:** Tipo de combustível (VARCHAR(8)).
  - **Procedência:** Procedência do modelo (VARCHAR(16)).
- **Chaves:**
  - **Chave Primária:** (Modelo, Ano\_Modelo, Combustivel FIPE).
  - **Chave Estrangeira:** Modelo (referenciando outra tabela).
- **Cardinalidade:**
  - Relacionamento com "Fato\_Fipe" é de 1 para N (1:N).
  - Relacionamento com "Dados Técnicos Modelos" é de 1 para N (1:N).

#### **Entidade: FATO\_FIPE**

- **Atributos:**

- **Mes\_Ref:** Mês de referência (DATE).
- **Modelo:** Identificador do modelo (VARCHAR(128)).
- **Ano\_Modelo:** Ano específico do modelo (SMALLINT).
- **Combustivel:** Tipo de combustível (VARCHAR(8)).
- **Preco\_Medio:** Preço médio do veículo (INTEGER).
- **Autenticacao:** Código de autenticação (VARCHAR(16)).
- **Data\_Consulta:** Data e hora da consulta (TIMESTAMP).
- **Chaves:**
  - **Chave Primária:** (Mes\_Ref, Modelo, Ano\_Modelo, Combustivel).
  - **Chave Estrangeira:** Modelo (referenciando outra tabela).
- **Cardinalidade:**
  - Relacionamento com "Dados Técnicos Modelos Ano Modelo" é de 1 para N (1:N).
  - - Relacionamento com "Fipe Marca Modelo" é de 1 para N (1:N).
  - - Relacionamento com "Fato Salário Mínimo" é de 1 para N (1:N).

#### **Entidade: DIM\_FIPE\_MARCA\_MODELO**

- **Atributos:**
  - **Modelo:** Identificador do modelo (VARCHAR(128)).
  - **Marca:** Marca do modelo (VARCHAR(32)).
  - **Cod\_FIPE:** Código FIPE do modelo (VARCHAR(8)).
- **Chaves:**
  - **Chave Primária:** Modelo.
- **Cardinalidade:**
  - Relacionamento com "Fato\_Fipe" é de 1 para N (1:N).

#### **Entidade: FATO\_SALARIO\_MINIMO**

- **Atributos:**
  - **Mes\_Ref:** Mês de referência (DATE).

- **Valor:** Valor do salário mínimo (INTEGER).
- **Chaves:**
  - **Chave Primária:** Mes\_Ref.
- **Cardinalidade:**
- O Relacionamento com "Fato Salário Mínimo" é de 1 para N (1:N).

Este modelo de dados visa proporcionar uma estrutura organizada e relacionada para armazenar e recuperar informações sobre modelos de veículos, preços associados, dados técnicos e valores de salário mínimo em diferentes períodos.

## **Consultas SQL para Criação de Tabelas e Relacionamentos:**

Abaixo está o conjunto de consultas **SQL** que foram necessárias para criar todas as tabelas do banco de dados, considerando os relacionamentos entre as entidades:

Query	Query History
1	<b>CREATE TABLE IF NOT EXISTS</b> dim_fipe_marca_modelo(
2	modelo VARCHAR(128) NOT NULL,
3	marca VARCHAR(32) NOT NULL,
4	cod_fipe VARCHAR(8) NOT NULL,
5	<b>PRIMARY KEY</b> (modelo)
6	);
7	
8	<b>CREATE TABLE IF NOT EXISTS</b> fato_fipe(
9	mes_ref DATE NOT NULL,
10	modelo VARCHAR(128) NOT NULL,
11	ano_modelo SMALLINT NOT NULL,
12	combustivel VARCHAR(8) NOT NULL,
13	preco_medio INTEGER NOT NULL,
14	autenticacao VARCHAR(16) NOT NULL,
15	data_consulta TIMESTAMP NOT NULL,
16	<b>PRIMARY KEY</b> (mes_ref,modelo,ano_modelo,combustivel),
17	<b>FOREIGN KEY</b> (modelo) <b>REFERENCES</b> dim_fipe_marca_modelo(modelo)
18	);
19	
20	<b>CREATE TABLE IF NOT EXISTS</b> dim_dados_tecnicos_modelo(
21	modelo VARCHAR(128) NOT NULL,
22	marca VARCHAR(32),
23	familia VARCHAR(32),
24	configuracao VARCHAR(16),
25	lugares SMALLINT,
26	portas SMALLINT,
27	porte VARCHAR(16),
28	propulsao VARCHAR(16),
29	<b>PRIMARY KEY</b> (modelo)
30	);
31	

Query	Query History
16	<b>PRIMARY KEY</b> (mes_ref,modelo,ano_modelo,combustivel),
17	<b>FOREIGN KEY</b> (modelo) <b>REFERENCES</b> dim_fipe_marca_modelo(mod
18	);
19	
20	<b>CREATE TABLE IF NOT EXISTS</b> dim_dados_tecnicos_modelo(
21	modelo VARCHAR(128) NOT NULL,
22	marca VARCHAR(32),
23	familia VARCHAR(32),
24	configuracao VARCHAR(16),
25	lugares SMALLINT,
26	portas SMALLINT,
27	porte VARCHAR(16),
28	propulsao VARCHAR(16),
29	<b>PRIMARY KEY</b> (modelo)
30	);
31	
32	<b>CREATE TABLE IF NOT EXISTS</b> dim_dados_tecnicos_ano_modelo(
33	modelo VARCHAR(128) NOT NULL,
34	ano_modelo SMALLINT NOT NULL,
35	combustivel VARCHAR(8),
36	combustivel_fipe VARCHAR(8) NOT NULL,
37	procedencia VARCHAR(16),
38	<b>PRIMARY KEY</b> (modelo,ano_modelo,combustivel_fipe),
39	<b>FOREIGN KEY</b> (modelo) <b>REFERENCES</b> dim_dados_tecnicos_modelo
40	);
41	
42	<b>CREATE TABLE IF NOT EXISTS</b> fato_salario_minimo(



## Inserção de Dados nas Tabelas:

Na fase de inserção de dados, desenvolvemos **scripts** em Python para converter as tabelas em arquivos de inserção (insert files). Abaixo vamos ver alguns exemplos do tratamento feitos com Python para que os Dados pudessem ser inseridos em nosso Banco.

Neste exemplo, a função *trataDateRefFipe* converte uma data no formato "mês/ano" (por exemplo, 'janeiro/2000') para o formato "YYYY-MM-DD" e a retorna como uma string.

```
def trataDateRefFipe(txt):
    mes_replace= {'janeiro':'01', 'fevereiro':'02', 'março':'03', 'abril':'04', 'maio':'05',
    'junho':'06', 'julho':'07',
    'agosto':'08', 'setembro':'09', 'outubro':'10', 'novembro':'11', 'dezembro':'12'
    }

    try:
        mesDesc, ano = txt.split('/')
        mes = mes_replace.get(mesDesc)

        if mes:
            return trataTexto(ano+'-'+mes+'-'+01)
        else:
            return 'NULL'
    except:
        return 'NULL'

trataDateRefFipe('janeiro/2000')

# Resultado "'2000-01-01'"
```

Já neste exemplo, o script Python está gerando comandos SQL do tipo INSERT para a tabela *fato\_fipe* e escrevendo esses comandos em um arquivo chamado *insert\_fato\_fipe.txt*.

```

inícioInsert='''INSERT INTO fato_fipe
VALUES
'''
with open('insert_fato_fipe.txt', 'a',encoding="utf-8") as file:
    file.write(inícioInsert)
    file.close()

tamanho = len(FATO_FIPE)
for row in FATO_FIPE.iterrows():
    data = trataDateRefFipe(row[1][0])
    modelo = trataTexto(row[1][1])
    ano_modelo, combustivel = trataAno_Modelo(row[1][2])
    preco=trataPreco(row[1][3])
    validacao = trataTexto(row[1][4])
    data_consulta = trataDataConsulta(row[1][5])

    linhaTratada = f"({data},{modelo},{ano_modelo},{combustivel},{preco},{validacao},{data_consulta})"
    if row[0] < tamanho-1:
        linhaTratada = linhaTratada+',\n'
    else:
        linhaTratada = linhaTratada+';'
    with open('insert_fato_fipe.txt', 'a',encoding="utf-8") as file:
        file.write(linhaTratada)
        file.close()

DIM_FIPE_MARCA_MODELO = AmostraFipe.groupby(['Modelo','Marca','Cod_FIPE']).count()
# [['Modelo','Marca','Cod_FIPE']]

DIM_FIPE_MARCA_MODELO

```

Vocês podem visualizar todos os tratamentos e ter acesso integral aos scripts por meio do [arquivo](#) disponível em nosso repositório no GitHub, lá é possível encontrar todos os arquivos e notebooks de tratamento dos dados.

Paralelamente, elaboramos comandos SQL correspondentes, permitindo a inserção de dados nas tabelas recém-criadas. Essa abordagem assegura que o banco de dados seja inicialmente povoado com um conjunto de informações, proporcionando uma base sólida para fins de demonstração e análise. Eis aqui um exemplos:

## Insert da Tabela Fato Fipe

Query	Query History
2685	
2686	<b>INSERT INTO</b> fato_fipe
2687	<b>VALUES</b>
2688	('2023-10-01','100 2.8 V6',1995,'Gasolina',13139,'g9c153pmhmc','2023-10-06 22:21:00'),
2689	('2023-10-01','100 2.8 V6',1994,'Gasolina',12753,'g4vrhxslv1c','2023-10-06 22:21:00'),
2690	('2023-10-01','100 2.8 V6',1993,'Gasolina',10857,'gfmmmg9pxc','2023-10-06 22:21:00'),
2691	('2023-10-01','100 2.8 V6 Avant',1995,'Gasolina',14328,'hq9692b9k6c','2023-10-06 22:21:00'),
2692	('2023-10-01','100 2.8 V6 Avant',1994,'Gasolina',12853,'g503nx9n7zc','2023-10-06 22:21:00'),
2693	('2023-10-01','100 S-4 2.2 Avant Turbo',1995,'Gasolina',23489,'l6qzfdcwflc','2023-10-06 22:21:00'),
2694	('2023-10-01','100 S-4 2.2 Avant Turbo',1994,'Gasolina',21247,'lcfzkh4f5nc','2023-10-06 22:21:00'),
2695	('2023-10-01','80 2.0',1995,'Gasolina',15405,'h4ypgl7f7jc','2023-10-06 22:22:00'),
2696	('2023-10-01','80 2.0',1994,'Gasolina',12459,'g1dw7zmlmc','2023-10-06 22:22:00'),
2697	('2023-10-01','80 2.0 Avant',1995,'Gasolina',15457,'h5klcdsvzqc','2023-10-06 22:22:00'),
2698	('2023-10-01','80 2.0 Avant',1994,'Gasolina',13098,'g8w46109xxc','2023-10-06 22:22:00'),
2699	('2023-10-01','80 2.6/ 2.8',1995,'Gasolina',23215,'l3jc3rptgtc','2023-10-06 22:22:00'),
2700	('2023-10-01','80 2.6/ 2.8',1994,'Gasolina',20304,'k0c97zfl2zc','2023-10-06 22:22:00'),
2701	('2023-10-01','80 2.6/2.8 Avant',1996,'Gasolina',27515,'nqx9mcf4lhc','2023-10-06 22:22:00'),
2702	('2023-10-01','80 2.6/2.8 Avant',1995,'Gasolina',26843,'ngl1tk1tncs','2023-10-06 22:22:00'),
2703	('2023-10-01','80 2.6/2.8 Avant',1994,'Gasolina',20825,'k6hpm96b9tc','2023-10-06 22:22:00'),
2704	('2023-10-01','80 2.8 Cabriolet',1999,'Gasolina',59711,'1xfkyg50t5c','2023-10-06 22:22:00'),
2705	('2023-10-01','80 2.8 Cabriolet',1998,'Gasolina',58254,'1cbz3ncm7vc','2023-10-06 22:22:00'),
2706	('2023-10-01','80 2.8 Cabriolet',1997,'Gasolina',56833,'0tphxd0r4bc','2023-10-06 22:22:00'),
2707	('2023-10-01','80 2.8 Cabriolet',1996,'Gasolina',55446,'z9fcs5wpy9c','2023-10-06 22:22:00'),
2708	('2023-10-01','80 2.8 Cabriolet',1995,'Gasolina',54093,'zsklsyzfrcc','2023-10-06 22:22:00'),
2709	('2023-10-01','80 2.8 Cabriolet',1994,'Gasolina',48040,'xhlh8dr954c','2023-10-06 22:22:00'),
2710	('2023-10-01','80 S2 Avant',1995,'Gasolina',23240,'l3tgs33168c','2023-10-06 22:22:00'),
2711	('2023-10-01','80 S2 Avant',1994,'Gasolina',20506,'k2rq9fttfzc','2023-10-06 22:22:00'),
2712	('2023-10-01','A1 1.4 TFSI 122cv S-tronic 3p',2014,'Gasolina',72781,'6vn3ybqx8vrl','2023-10-06 22:22:00'),
2713	('2023-10-01','A1 1.4 TFSI 122cv S-tronic 3p',2013,'Gasolina',70685,'51284fl6gsnc','2023-10-06 22:22:00'),
2714	('2023-10-01','A1 1.4 TFSI 122cv S-tronic 3p',2012,'Gasolina',70685,'51284fl6gsnc','2023-10-06 22:22:00'),
2715	('2023-10-01','A1 1.4 TFSI 122cv S-tronic 3p',2011,'Gasolina',70685,'51284fl6gsnc','2023-10-06 22:22:00'),
Data Output Messages Notifications	
Total rows: 8 of 8 Query complete 00:00:00.143 Ln 2714, Col 1	

## Consultas

A seguir, apresentamos algumas das consultas SQL que aproveitam as capacidades do banco de dados, extraindo informações relevantes e interessantes de nosso conjunto de dados.

**Quais são os carros de menor preço de cada marca, no mês de dezembro de 2023, e quantos salários mínimos representam o valor deles?**

Query

Query History

1

/\*

2

Consulta 01:

3

4

Quais são os carros de menor preço de cada marca, no mês de dez/23,

5

quantos salários mínimos representam o valor deles?

6

7

\*/

8

WITH

9

modelos\_zeroKm\_marca\_dez\_2023 AS(

10

SELECT d.marca

11

,f.modelo

12

,f.preco\_medio

13

,f.mes\_ref

14

FROM fato\_fipe f

15

INNER JOIN dim\_fipe\_marca\_modelo d

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑

🗑

📥

📥

📈

	<div>Marca</div> <div>character varying (32)</div>	<div>Modelo de Entrada (menor preço) por Marca</div> <div>character varying (128)</div>	<div>Valor Médio (FIPE)</div> <div>text</div>	<div>Qtd Salários Mínimos</div> <div>text</div>
1	Fiat	MOBI LIKE 1.0 Fire Flex 5p.	68.723,00	52,06
2	VW - VolksWagen	Gol 1.0 Flex 12V 5p	78.144,00	59,20
3	Toyota	YARIS XL Sedan 1.5 Flex 16V 4p Aut.	97.193,00	73,63
4	Honda	CITY Sedan EX 1.5 Flex 16V 4p Aut.	121.211,00	91,83
5	BYD	Dolphin EV (Elétrico)	149.897,00	113,56
6	Ford	Ranger Black 2.2 4x2 CD Diesel Aut.	209.934,00	159,04
7	Audi	A3 Sportb. S-Line 2.0 TFSI S-Tron.(Hib.)	284.662,00	215,65
8	BMW	X1 SDRIVE 18i GP 1.5 TB Aut.	301.838,00	228,67

A análise dos dados revela que o Fiat Mobi Like apresenta um valor médio de **R\$ 68.723,00**, o que, ao ser relacionado com o **salário mínimo vigente**, representa aproximadamente **52,06** salários mínimos. Essa informação levanta preocupações legítimas sobre a acessibilidade dos veículos no mercado.

Uma crítica relevante pode ser direcionada ao fato de que o valor do Fiat Mobi Like parece exorbitante em comparação com o poder aquisitivo médio da população, representado pelos salários mínimos. Isso pode indicar um desafio significativo para muitas pessoas que buscam adquirir um veículo, levantando questões sobre a equidade no acesso a esses bens.

A discrepância entre os valores dos veículos e os salários mínimos também destaca a necessidade de se analisar e abordar questões relacionadas à desigualdade econômica, uma vez que o acesso a tais bens, como veículos, desempenha um papel crucial na mobilidade e qualidade de vida das pessoas.

Em resumo, a crítica recai sobre a disparidade entre os valores dos veículos e a capacidade financeira da população, sinalizando a importância de considerar políticas e práticas que promovam a acessibilidade e a equidade no setor automotivo.

## Em dezembro de 2023, quantos modelos Zero KM tem cada marca e qual valor médio deles?

Query

Query History

128

/\*

129

Consulta 03:

130

Em dezembro de 2023, quantos modelos Zero KM tem cada marca e qual valor médio deles?

132

\*/

133

WITH

134

modelos\_zeroKm\_marca\_dez\_2023 AS (

135

SELECT d.marca

136

,f.modelo

137

,f.preco\_medio

138

,f.mes\_ref

139

FROM fato\_fipe f

140

INNER JOIN dim\_fipe\_marca\_modelo d

141

ON f.modelo = d.modelo

142

Data Output

Messages

Notifications

	Marca character varying (32)	Qty Modelos bigint	Valor Médio (R\$) text
1	Audi	42	569.126,52
2	BMW	39	614.322,82
3	BYD	9	304.935,44
4	Fiat	49	149.840,94
5	Ford	36	298.661,64
6	Honda	12	189.269,50
7	Toyota	33	244.195,61
8	VW - VolksWagen	35	160.466,66

A análise da consulta realizada em dezembro de 2023 revela informações sobre a disponibilidade de modelos Zero KM de diversas marcas e seus respectivos valores médios. Destacamos dois exemplos: a marca Audi apresenta um total de 42 modelos Zero KM, com um valor médio de R\$ 596.126,52. Por outro lado, a BMW oferece 39 modelos Zero KM, com um valor médio de R\$ 614.322,82. O ato de comparar ressalta nuances nas estratégias de precificação e na oferta de modelos por parte dessas marcas renomadas. Essa análise é crucial para consumidores e empresas do setor automotivo ao tomarem decisões informadas sobre compra e posicionamento no mercado.

## Qual é o sedan mais caro de outubro de 2023? Qual foi sua desvalorização entre out/23 e nov/23?

Query

Query History

56

/\*

57

Consulta 02:

58

59

Qual o sedã de maior valor comercializado em nov/23 (valor médio - FIPE)?

60

Qual foi sua desvalorização percentual entre nov/23 e dez/23?

61

--Vale lembrar que as tabelas são uma amostragem pequena e não representam a realidade

62

\*/

63

64

WITH

65

daddos\_tecnicos\_seda AS(

66

SELECT a.modelo

67

,a.ano\_modelo

68

,a.combustivel\_fipe

69

,b.marca

70

FROM dim dados tecnicos ano modelo a

Data Output

Messages

Notifications

	Marca character varying (32)	Modelo character varying (128)	Ano do Modelo smallint	Preço Nov/23 text	Preço Dez/23 text	Desvalorização (%) text
1	BMW	330i Sport 2.0 TB 16V 4p	2020	253.941,00	250.988,00	1,16

Com base nas informações, concluímos que o sedã mais caro em outubro de 2023, foi a BMW, modelo 2020, apresentou um valor de R\$ 253.941,00 neste mês. Ao compararmos esse valor com o preço registrado em novembro de 2023, que foi de R\$ 250.988,00, observamos uma desvalorização de R\$ 2.953,00 no período de outubro a novembro de 2023.

## Considerações Finais

A realização deste projeto foi uma experiência de aprendizado significativa que nos permitiu aplicar os conhecimentos adquiridos em sala de aula em um ambiente prático. O projeto nos permitiu explorar novos horizontes e expandir nossa compreensão sobre modelagem de Banco de Dados e SQL.

Gostaríamos de expressar nossa gratidão ao professor Jorge, por nos proporcionar a oportunidade de realizar este projeto e pela orientação valiosa que nos foi oferecida ao longo

das aulas. Foi uma jornada colaborativa que nos ajudou a fortalecer nossos laços e a compreender a importância do trabalho em equipe.

Atenciosamente,

Grupo 1 - Turma 1102 | Dados.

## Referências

Leite, Jared. "**WebScraping\_FIPE Repository**." GitHub, [https://github.com/jaredleite/WebScraping\\_FIPE/tree/main](https://github.com/jaredleite/WebScraping_FIPE/tree/main).

Leite, Jared. "**WebScraping\_CarTechData**." GitHub, [https://github.com/jaredleite/WebScraping\\_CarTechData](https://github.com/jaredleite/WebScraping_CarTechData).

Ipeadata. "**Ipeadata - Banco de Dados Macroeconômicos**." Disponível em: <http://www.ipeadata.gov.br/>. Acesso em: [11/12/2023].

Fipe - Fundação Instituto de Pesquisas Econômicas. "**Tabela FIPE - Preços Médios de Veículos**." Disponível em: <https://veiculos.fipe.org.br/>. Acesso em: [29/07/2023].

Wikipedia. "**Coleta\_De\_Dados**" Disponível em: [https://pt.wikipedia.org/wiki/Coleta\\_de\\_dados\\_web](https://pt.wikipedia.org/wiki/Coleta_de_dados_web). Acesso em 11 de dezembro de 2023.

**Ficha Completa**. Disponível em: <https://www.fichacompleta.com.br/carros/>. Acesso em: [29/07/2023].