IMPORTANDO O PANDAS

```
import pandas as pd
```

IMPORTANDO E MOSTRANDO OS DADOS

```
dfProd=pd.read_csv("product.csv",sep=";")
dfProd.head(5)
```

	ProductID	ProviderID	CategoryID	Description
0	99POP	3	1	99POP
1	1	1	5	Taxi Comum
2	2	1	6	Executivo
3	46cec5c4d23e57bcba2122677eb8c759	4	5	Easy Taxi Corp (-15%)
4	5fc141256dc70a394d0ce4c5c1444dfc	4	5	Taxi

```
dfRide=pd.read_csv("ride_v2.csv",sep=";")
dfRide.head(5)
```

	RideID	UserID	Schedule	Create	RideStatusID	CompanyID	ProviderID	RideProviderID	price	Updat
0	1685755	e15b8cc3-5a67-4630-b89f-ee69f302b582	2025-02-10 14:31:10.8858446	2025-02-10 14:31:10.9084221	1	2	NaN	NaN	0.00	2025-02-14:31:10.90842
1	1685754	5c3fb011-0aea-429a-8305-b88953b77df1	2025-02-10 14:26:35.3411403	2025-02-10 14:26:35.4169873	2	230	5.0	NaN	30.45	2025-02-14:28:02.46569
2	1685753	d7e2f4dc-337f-45f5-b762-67b72b077abc	2025-02-10 14:23:45.2540905	2025-02-10 14:24:32.7058722	2	52	3.0	NaN	11.40	2025-02-14:24:46.50371
3	1685752	2125ed9c-89b8-4df6-9be6-53195397a269	2025-02-10 14:23:12.9838635	2025-02-10 14:23:12.9975475	8	230	36.0	1589157	45.79	2025-02-14:30:30.60311
4	1685751	72cbebfb-5d70-49ab-ab23-8e3b57c7e399	2025-02-10 14:19:30.5937678	2025-02-10 14:19:30.6117184	2	2	3.0	NaN	17.28	2025-02-14:24:45.97117

```
dfRideAdd=pd.read_csv("rideaddress_v1.csv",sep=";",low_memory=False)
dfRideAdd.head(5)
```

	RideAddressID	Address	Street	Number	Neighborhood	City	State	Lat	Lng	RideAddress
0	2334277	Rua João Pinheiro, 585 - Rua João Pinheiro - B...	Rua João Pinheiro	585	Rua João Pinheiro	NaN	Brasil	-26.329754299999998	-48.840427999999996	
1	2334278	Av. Dr. Nereu Ramos, 450 - Rocio Grande, São F...	Av. Dr. Nereu Ramos, 450 - Rocio Grande, São F...	450	NaN	NaN	NaN	-26.2554657	-48.6434197	
2	2334279	Rodovia Rafael da Rocha Pires, 1883 - Rodovia ...	Rodovia Rafael da Rocha Pires	1883	Rodovia Rafael da Rocha Pires	NaN	Brasil	-27.4919788	-48.528287999999996	
3	2334280	Angeloni Ingleses (Florianópolis) - Supermerca...	Angeloni Ingleses (Florianópolis) - Supermercado	6375	NaN	NaN	NaN	-27.4371486	-48.39824309999999	
4	2334281	Rua Barão do Rio Branco, 12 - Rua Barão do Rio...	Rua Barão do Rio Branco	12	Rua Barão do Rio Branco	NaN	Brasil	-19.8495799	-44.019915999999995	

```
dfRideEst=pd.read_csv("rideestimative_v3.csv",sep=";",low_memory=False)
dfRideEst.head(5)
```

	RideEstimativeID	RideID	ProductID	WaitingTime	Price	FareID	Selected	RideReasonSelectedEstimativeID	Fee
0	8619946	1183200	Flash	8	89.00	c6aaac64-5f89-4fc4-8b66-0251ec1c78a8	0	NaN	0.0
1	8619947	1183200	UberX	6	89.00	ff3cc941-93a8-4d0e-a274-bb988576d7d4	0	NaN	0.0
2	8619948	1183200	Comfort	10	116.50	d7708871-2f2c-447d-81e6-a2d121863a2f	0	NaN	0.0

LIMPANDO OS DADOS

```
print(len(dfProd))
print(len(dfRide))
print(len(dfRideAdd))
print(len(dfRideEst))
print(len(dfProd.dropna()))
print(len(dfRide.dropna()))
print(len(dfRideAdd.dropna()))
print(len(dfRideEst.dropna()))
```

```
237
500000
1000000
2000000
237
14781
121971
132236
```

ADICIONANDO UMA COLUNA PARA O DIA DA SEMANA

```
import pandas as pd

# Convertendo a coluna Schedule para formato de data (caso ainda não esteja)
dfRide['Schedule'] = pd.to_datetime(dfRide['Schedule'])

# Criando os dias da semana em português usando um dicionário de tradução
dias_em_portugues = {
    "Monday": "Segunda-feira",
    "Tuesday": "Terça-feira",
    "Wednesday": "Quarta-feira",
    "Thursday": "Quinta-feira",
    "Friday": "Sexta-feira",
    "Saturday": "Sábado",
```

```
    "Sunday": "Domingo"
}

# Adicionando a coluna 'Dia' com os dias traduzidos
dfRide['Dia'] = dfRide['Schedule'].dt.day_name().map(dias_em_portugues)

# Visualizando o resultado
dfRide.head()
```

	RideID	UserID	Schedule	Create	RideStatusID	CompanyID	ProviderID	RideProviderID	price	Upd
0	1685755	e15b8cc3-5a67-4630-b89f-ee69f302b582	2025-02-10 14:31:10.885844600	2025-02-10 14:31:10.9084221	1	2	NaN	NaN	0.00	2025-(14:31:10.908
1	1685754	5c3fb011-0aea-429a-8305-b88953b77df1	2025-02-10 14:26:35.341140300	2025-02-10 14:26:35.4169873	2	230	5.0	NaN	30.45	2025-(14:28:02.465
2	1685753	d7e2f4dc-337f-45f5-b762-67b72b077abc	2025-02-10 14:23:45.254090500	2025-02-10 14:24:32.7058722	2	52	3.0	NaN	11.40	2025-(14:24:46.503
3	1685752	2125ed9c-89b8-4df6-9be6-53195397a269	2025-02-10 14:23:12.983863500	2025-02-10 14:23:12.9975475	8	230	36.0	1589157	45.79	2025-(14:30:30.603
4	1685751	72cbebfb-5d70-49ab-ab23-8e3b57c7e399	2025-02-10 14:19:30.593767800	2025-02-10 14:19:30.6117184	2	2	3.0	NaN	17.28	2025-(14:24:45.971

RENOMENANDO A COLUNA RIDEADDRESSTYPEID PARA MELHORAR O ENTENDIMENTO

```
dfRideAdd = dfRideAdd.rename(columns={'RideAddressTypeID': 'OrigDest'})
dfRideAdd.head(5)
```

	RideAddressID	Address	Street	Number	Neighborhood	City	State	Lat	Lng	OrigDest	I
0	2334277	Rua João Pinheiro, 585 - Rua João Pinheiro - B...	Rua João Pinheiro	585	Rua João Pinheiro	NaN	Brasil	-26.329754299999998	-48.840427999999996	1	1
1	2334278	Av. Dr. Nereu Ramos, 450 - Rocio Grande, São F...	Av. Dr. Nereu Ramos, 450 - Rocio Grande, São F...	450	NaN	NaN	NaN	-26.2554657	-48.6434197	2	1
2	2334279	Rodovia Rafael da Rocha Pires, 1883 - Rodovia ...	Rodovia Rafael da Rocha Pires	1883	Rodovia Rafael da Rocha Pires	NaN	Brasil	-27.4919788	-48.528287999999996	1	1
3	2334280	Angeloni Ingleses (Florianópolis) - Supermerca...	Angeloni Ingleses (Florianópolis) - Supermercado	6375	NaN	NaN	NaN	-27.4371486	-48.39824309999999	2	1
		Rua Barão do									

CALCULANDO A DISTÂNCIA COM BASE NAS LATITUDES E LONGITUDES

```
import pandas as pd

# Filtrando os dados por OrigDest para separar origem e destino
dfOrig = dfRideAdd[dfRideAdd['OrigDest'] == 1][['RideID', 'Lat', 'Lng']].rename(columns={'Lat': 'Lat_1', 'Lng': 'Lng_1'})
dfDest = dfRideAdd[dfRideAdd['OrigDest'] == 2][['RideID', 'Lat', 'Lng']].rename(columns={'Lat': 'Lat_2', 'Lng': 'Lng_2'})

# Fazendo o merge para juntar origem e destino em um único DataFrame
dfRideAdd_unificado = pd.merge(dfOrig, dfDest, on='RideID', how='inner')

# Combinando Lat_1 e Lng_1 em uma única coluna
dfRideAdd_unificado['Coordenadas_1'] = dfRideAdd_unificado.apply(lambda row: f"{row['Lat_1']}, {row['Lng_1']}", axis=1)

# Combinando Lat_2 e Lng_2 em uma única coluna
dfRideAdd_unificado['Coordenadas_2'] = dfRideAdd_unificado.apply(lambda row: f"{row['Lat_2']}, {row['Lng_2']}", axis=1)
```

```
dfRideAdd_unificado['Coordenadas_2'] = dfRideAdd_unificado.apply(lambda row: f'{row['Lat_2']}, {row['Lng_2']}', axis=1)

# Visualizando o resultado final
dfRideAdd_unificado[['RideID', 'Coordenadas_1', 'Coordenadas_2']].head()
```

	RideID	Coordenadas_1	Coordenadas_2
0	1183200	-26.329754299999998, -48.840427999999996	-26.2554657, -48.6434197
1	1183201	-27.4919788, -48.528287999999996	-27.4371486, -48.39824309999999
2	1183202	-19.8495799, -44.019915999999995	-19.936899, -43.9401603
3	1183203	-23.9624233, -46.25465759999999	-23.8373074, -46.1321725
4	1183204	-10.9198019, -37.077441799999995	-10.9071288, -37.0877194

```
from geopy.distance import geodesic
import pandas as pd

# Garantindo que as coordenadas estejam no formato numérico (substituir ',' por '.')
dfRideAdd_unificado['Lat_1'] = dfRideAdd_unificado['Lat_1'].str.replace(',', '.').astype(float)
dfRideAdd_unificado['Lng_1'] = dfRideAdd_unificado['Lng_1'].str.replace(',', '.').astype(float)
dfRideAdd_unificado['Lat_2'] = dfRideAdd_unificado['Lat_2'].str.replace(',', '.').astype(float)
dfRideAdd_unificado['Lng_2'] = dfRideAdd_unificado['Lng_2'].str.replace(',', '.').astype(float)

# Função para calcular a distância em km usando geopy
def calcular_distancia(row):
    coord_origem = (row['Lat_1'], row['Lng_1'])
    coord_destino = (row['Lat_2'], row['Lng_2'])
    return geodesic(coord_origem, coord_destino).kilometers

# Calculando a distância e adicionando ao DataFrame
dfRideAdd_unificado['Distância_km'] = dfRideAdd_unificado.apply(calcular_distancia, axis=1)

# Visualizando o resultado final
dfRideAdd_unificado[['RideID', 'Coordenadas_1', 'Coordenadas_2', 'Distância_km']].head()
```

	RideID	Coordenadas_1	Coordenadas_2	Distância_km
0	1183200	-26.329754299999998, -48.840427999999996	-26.2554657, -48.6434197	21.327087
1	1183201	-27.4919788, -48.528287999999996	-27.4371486, -48.39824309999999	14.217724
2	1183202	-19.8495799, -44.019915999999995	-19.936899, -43.9401603	12.774728
3	1183203	-23.9624233, -46.25465759999999	-23.8373074, -46.1321725	18.643943
4	1183204	-10.9198019, -37.077441799999995	-10.9071288, -37.0877194	1.796511

```
dfRide.head()
```

	RideID	UserID	Schedule	Create	RideStatusID	CompanyID	ProviderID	RideProviderID	price	Upd
0	1685755	e15b8cc3-5a67-4630-b89f-ee69f302b582	2025-02-10 14:31:10.885844600	2025-02-10 14:31:10.9084221	1	2	NaN	NaN	0.00	2025-(14:31:10.908
1	1685754	5c3fb011-0aea-429a-8305-b88953b77df1	2025-02-10 14:26:35.341140300	2025-02-10 14:26:35.4169873	2	230	5.0	NaN	30.45	2025-(14:28:02.465
2	1685753	d7e2f4dc-337f-45f5-b762-67b72b077abc	2025-02-10 14:23:45.254090500	2025-02-10 14:24:32.7058722	2	52	3.0	NaN	11.40	2025-(14:24:46.503
3	1685752	2125ed9c-89b8-4df6-9be6-53195397a269	2025-02-10 14:23:12.983863500	2025-02-10 14:23:12.9975475	8	230	36.0	1589157	45.79	2025-(14:30:30.603
4	1685751	72cbebfb-5d70-49ab-ab23-8e3b57c7e399	2025-02-10 14:19:30.593767800	2025-02-10 14:19:30.6117184	2	2	3.0	NaN	17.28	2025-(14:24:45.971

CRIANDO UMA COLUNA PARA CALCULAR O TEMPO DAS CORRIDAS MINUTOS

```
import pandas as pd
```

```
# Filtrando os dados válidos onde 'Car' não é NULL ou vazio
dfRide_valid = dfRide[dfRide['Car'].notnull() & (dfRide['Car'] != '')].copy()

# Verificando alguns valores iniciais para confirmar o formato
print("Exemplo de valores nas colunas Create e Updated:")
print(dfRide_valid[['Create', 'Updated']].head(10))

# Convertendo as colunas 'Create' e 'Updated' para datetime (caso necessário)
dfRide_valid.loc[:, 'Create'] = pd.to_datetime(dfRide_valid['Create'], errors='coerce')
dfRide_valid.loc[:, 'Updated'] = pd.to_datetime(dfRide_valid['Updated'], errors='coerce')

# Calculando o tempo da viagem em minutos
dfRide_valid.loc[:, 'Tempo_Viagem_Min'] = (dfRide_valid['Updated'] - dfRide_valid['Create']).dt.total_seconds() / 60

# Visualizando os resultados
dfRide_valid[['RideID', 'Car', 'Create', 'Updated', 'Tempo_Viagem_Min']].head()
```

```
Exemplo de valores nas colunas Create e Updated:
                          Create                     Updated
3   2025-02-10 14:23:12.997547500 2025-02-10 14:30:30.603112300
28  2025-02-10 13:38:44.160459900 2025-02-10 13:49:33.126425600
60  2025-02-10 13:00:29.307441800 2025-02-10 13:19:38.118900200
79  2025-02-10 12:40:24.548489900 2025-02-10 12:55:18.448089800
160 2025-02-10 11:35:35.182254200 2025-02-10 12:04:31.409630400
162 2025-02-10 11:33:39.787264000 2025-02-10 12:05:31.521144100
433 2025-02-07 19:18:16.274998100 2025-02-07 20:19:41.009869300
487 2025-02-07 17:52:13.337529100 2025-02-07 18:11:16.570670300
506 2025-02-07 17:14:34.268628800 2025-02-07 17:26:31.986561500
516 2025-02-07 17:05:39.246366500 2025-02-07 17:55:57.867399600
```

	RideID	Car	Create	Updated	Tempo_Viagem_Min
3	1685752	VW VIRTUS CL / BRANCA	2025-02-10 14:23:12.997547500	2025-02-10 14:30:30.603112300	7.293426
28	1685727	TOYOTA YARIS/PRATA	2025-02-10 13:38:44.160459900	2025-02-10 13:49:33.126425600	10.816099