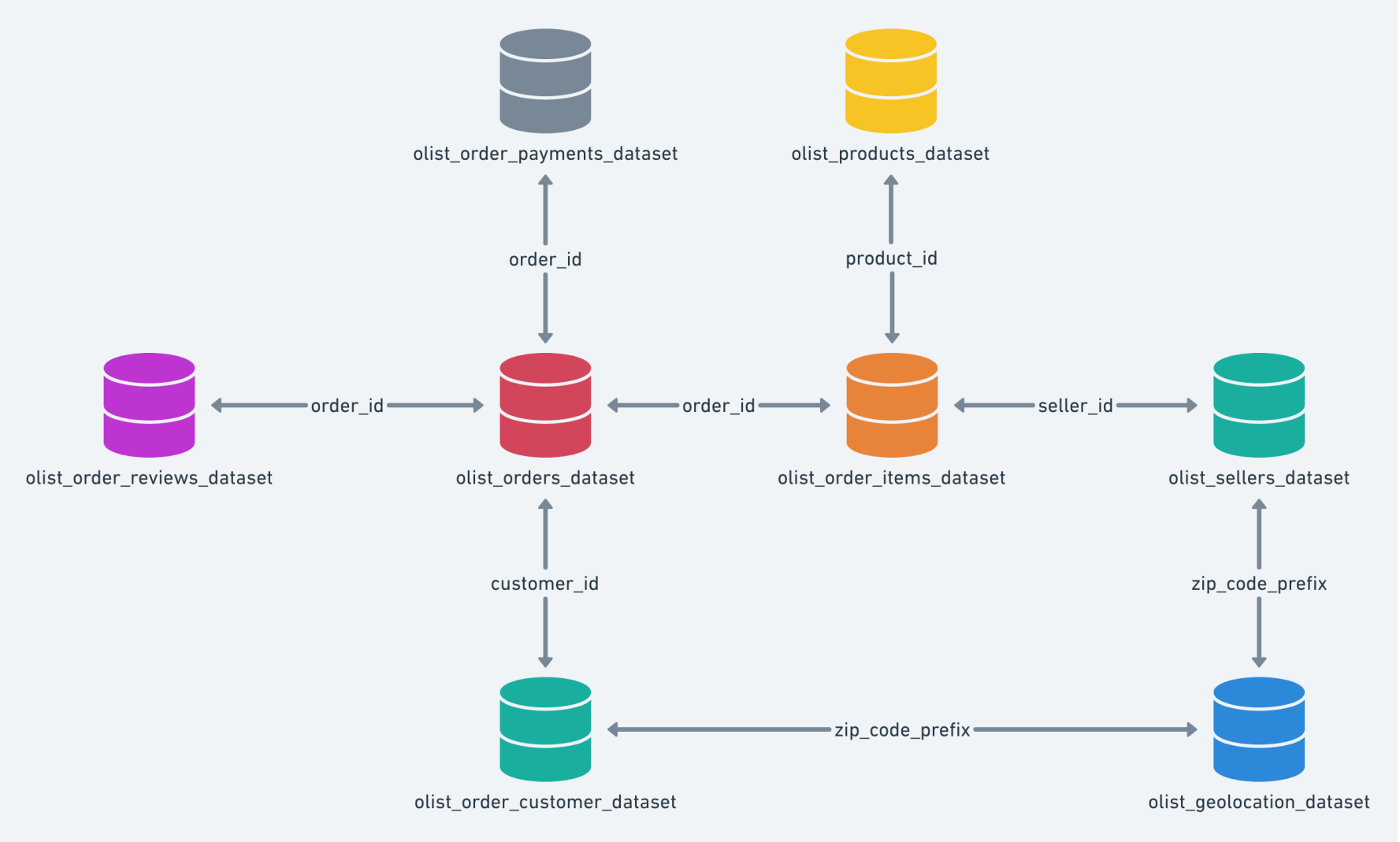
Teste Prático - Olist

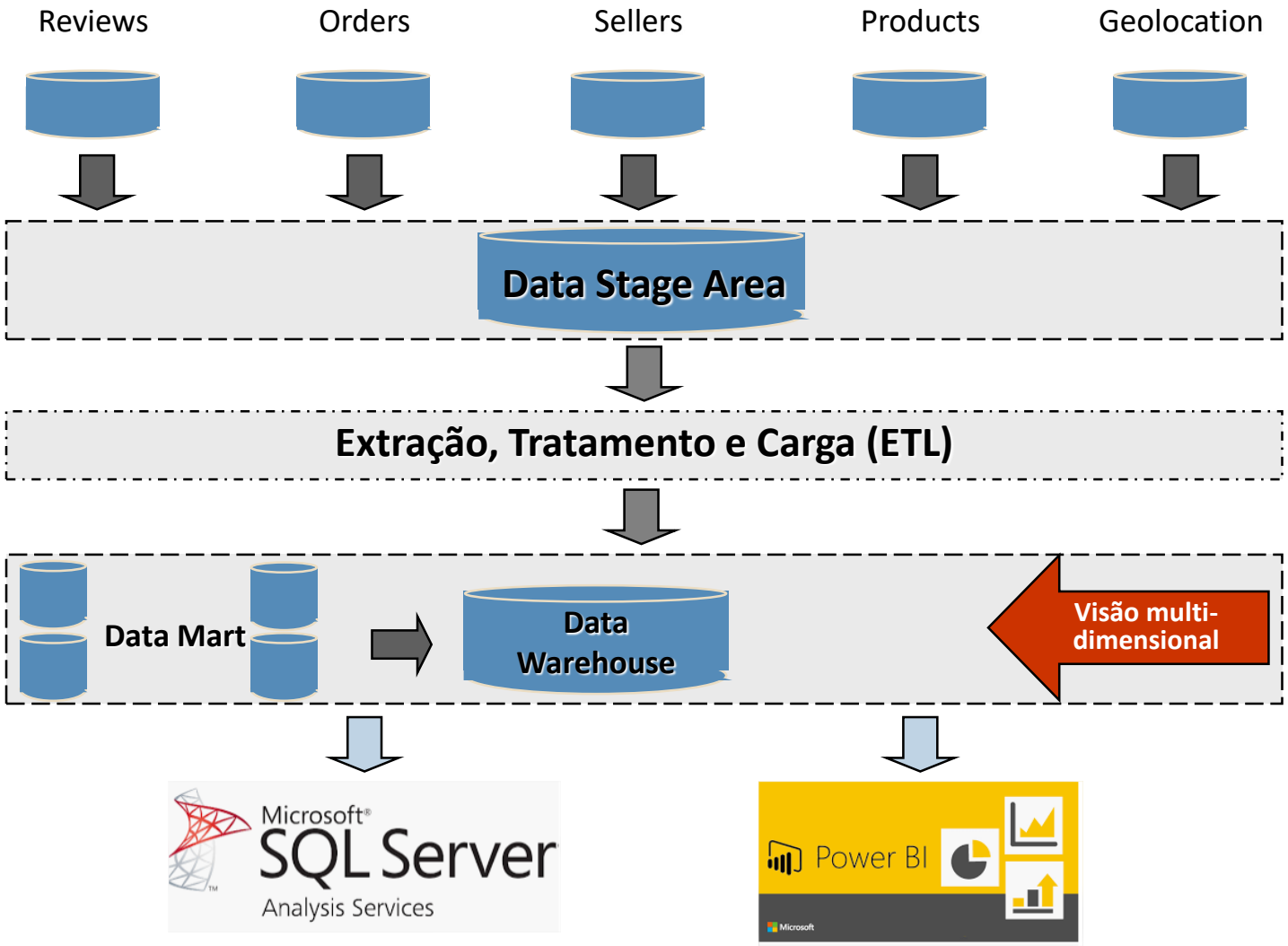
Data warehouse | Olist| Victor Aurelio Gomes Martens

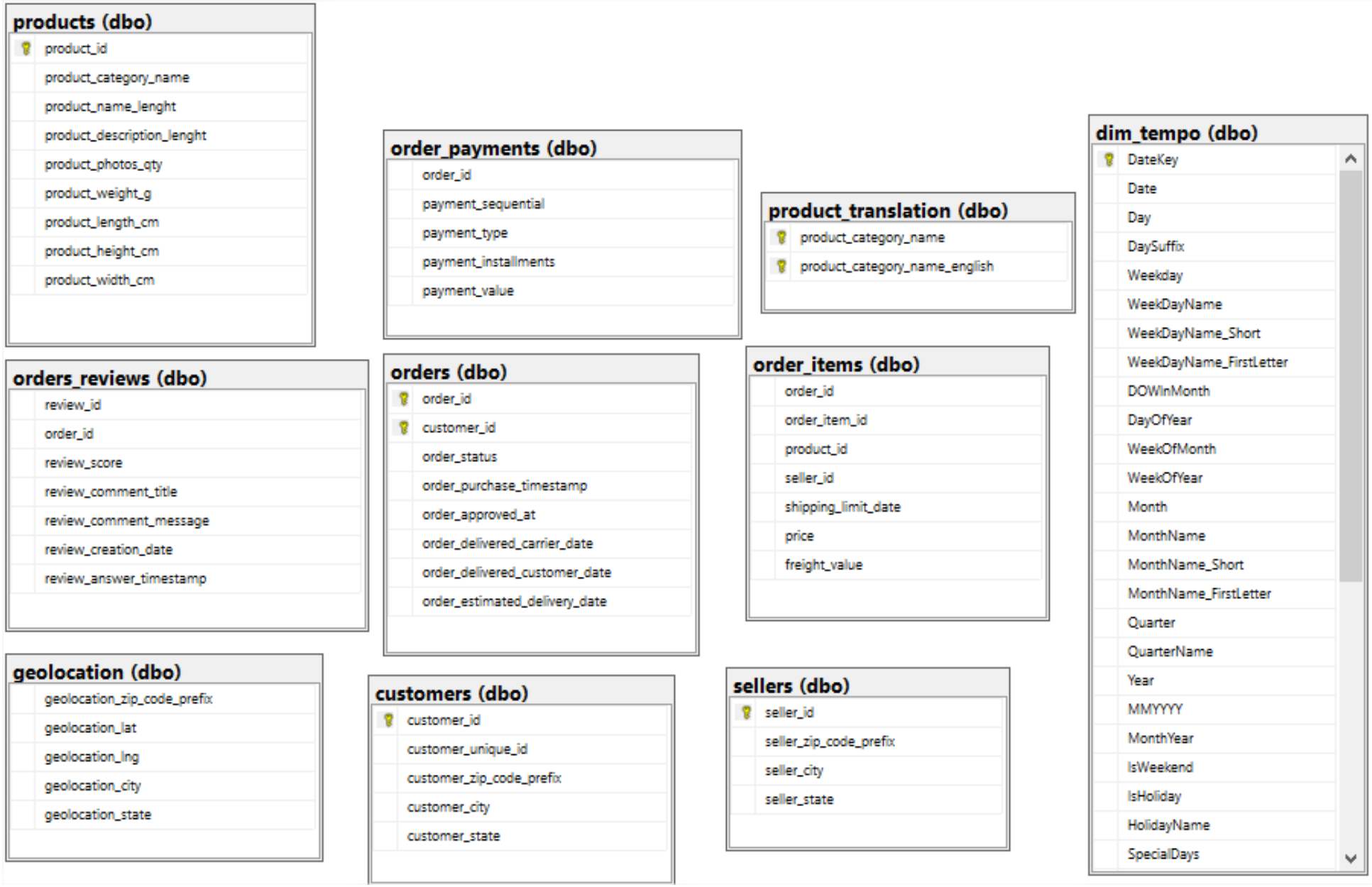
Datasets

Schema data base



Arquitetura




products (dbo)
product_id
product_category_name
product_name_lenght
product_description_lenght
product_photos_qty
product_weight_g
product_length_cm
product_height_cm
product_width_cm
order_payments (dbo)
order_id
payment_sequential
payment_type
payment_installments
payment_value
product_translation (dbo)
product_category_name
product_category_name_english
dim_tempo (dbo)
DateKey
Date
Day
DaySuffix
Weekday
WeekDayName
WeekDayName_Short
WeekDayName_FirstLetter
DOWInMonth
DayOfYear
WeekOfMonth
WeekOfYear
Month
MonthName
MonthName_Short
MonthName_FirstLetter
Quarter
QuarterName
Year
MMYYYY
MonthYear
IsWeekend
IsHoliday
HolidayName
SpecialDays
orders_reviews (dbo)
review_id
order_id
review_score
review_comment_title
review_comment_message
review_creation_date
review_answer_timestamp
orders (dbo)
order_id
customer_id
order_status
order_purchase_timestamp
order_approved_at
order_delivered_carrier_date
order_delivered_customer_date
order_estimated_delivery_date
order_items (dbo)
order_id
order_item_id
product_id
seller_id
shipping_limit_date
price
freight_value
geolocation (dbo)
geolocation_zip_code_prefix
geolocation_lat
geolocation_lng
geolocation_city
geolocation_state
customers (dbo)
customer_id
customer_unique_id
customer_zip_code_prefix
customer_city
customer_state
sellers (dbo)
seller_id
seller_zip_code_prefix
seller_city
seller_state

Datasets

- Para a conferência dos datasets, foi utilizado um algoritimo em Python para análise da base e sugestão de criação da tabela no sql server de forma a otimizar a criação dos campos.
- É gerado um HTML com as principais características de cada base, facilitando assim o entendimento da granularidade e frequência de cada campo.

dataset.py

work-at-olist-data-master > datasets > dataset.py > ...

```python
import pandas as pd
import numpy as np
import os
import pandas_profiling as pp
import csv, ast

def dataType(val, current_type):
  try:
      # Evaluates numbers to an appropriate type, and strings an error
      t = ast.literal_eval(val)
  except ValueError:
      return 'varchar'
  except SyntaxError:
      return 'varchar'
  if type(t) in [int, float]:
    if (type(t) in [int]) and current_type not in ['float', 'varchar']:
        # Use smallest possible int type
        if (-32768 < t < 32767) and current_type not in ['int', 'bigint']:
            return 'smallint'
        elif (-2147483648 < t < 2147483647) and current_type not in ['bigint']:
            return 'int'
        else:
            return 'bigint'
    if type(t) is float and current_type not in ['varchar']:
        return 'decimal'
  else:
      return 'varchar'

entries = os.listdir('datasets/')
for entry in entries:
    # if entry[len(entry)-3:] == "csv" and entry != 'olist_order_reviews_dataset.csv':
    if entry[len(entry)-3:] == "csv" and entry[:9] != 'cabeçalho':
        # print('Lendo o dataset: ', entry)
        dataset = entry[6-len(entry):-12]
        # print(dataset)
        f = open('datasets/'+entry, 'r')
        reader = csv.reader(f)
        longest, headers, type_list = [], [], []

        # if entry != 'olist_geolocation_dataset.csv':
        print('Lendo pandas csv...', entry)
        df = pd.read_csv('datasets/'+entry, sep=',')
        print('ProfileReport')
        profile = pp.ProfileReport(df, title=entry)
        print('Exportando html')
```

work-at-olist-data-master > datasets > dataset.py > ...

```python
        profile.to_file(output_file='datasets/html/' + entry + '.html')
        print('Profile report ok\n')

        # It iterates over the rows in our CSV, calls the function above, and populates the lists.
        for row in reader:
            if len(headers) == 0:
                headers = row
                for col in row:
                    longest.append(0)
                    type_list.append('')
            else:
                for i in range(len(row)):
                    # print('Range:', row[i], ' | Row ', i, ' | Len ', len(row[i]), ' | Longest ', longest[i] )
                    # NA is the csv null value
                    if type_list[i] == 'varchar' or row[i] == 'NA':
                        if len(row[i]) > longest[i]:
                            longest[i] = len(row[i])
                        pass
                    else:
                        var_type = dataType(row[i], type_list[i])
                        type_list[i] = var_type

        f.close()

        # Then use those lists to write the SQL statement
        statement = 'create table ' + dataset + '('

        campos = ''
        for i in range(len(headers)):
            campos = campos + headers[i].lower() + ", "
            if type_list[i] == 'varchar':
                statement = (statement + '\n{} varchar({}),').format(headers[i].lower(), str(longest[i]))
            else:
                statement = (statement + '\n' + '{} {}' + ',').format(headers[i].lower(), type_list[i])

        statement = statement[:-1] + ');'

        # Finally, the output!
        print(campos, '\n')
        print(statement, '\n')
```

resultado

```
Lendo pandas csv... olist_customers_dataset.csv
ProfileReport
Exportando html
Profile report ok

customer_id, customer_unique_id, customer_zip_code_prefix, customer_city, customer_state,

create table customers(
customer_id varchar(32),
customer_unique_id varchar(32),
customer_zip_code_prefix int,
customer_city varchar(32),
customer_state varchar(2));

Lendo pandas csv... olist_geolocation_dataset.csv
ProfileReport
```

HTML gerado olist_customers_dataset.csv

Overview

Dataset info

Number of variables	5
Number of observations	99441
Missing cells	0 (0.0%)
Duplicate rows	0 (0.0%)
Total size in memory	29.6 MiB
Average record size in memory	312.3 B

Variables types

CAT	4
NUM	1

Toggle Reproduction Information
Toggle Warnings

Warnings

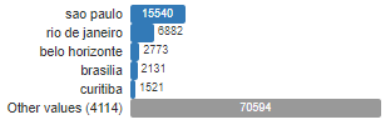
`customer_city` has a high cardinality: 4119 distinct values	Warning
`customer_id` has a high cardinality: 99441 distinct values	Warning
`customer_unique_id` has a high cardinality: 96096 distinct values	Warning

Variables

customer_city
Categorical

HIGH CARDINALITY

Distinct count	4119
Unique (%)	4.1%
Missing	0
Missing (%)	0.0%
Memory size	777.0 KiB

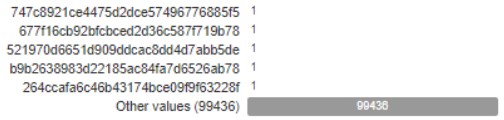


Toggle details

customer_id
Categorical

UNIQUE
HIGH CARDINALITY

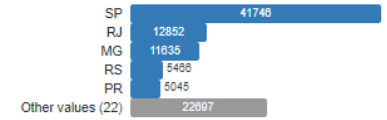
Distinct count	99441
Unique (%)	100.0%
Missing	0
Missing (%)	0.0%
Memory size	777.0 KiB



Toggle details

customer_state
Categorical

Distinct count	27
Unique (%)	< 0.1%
Missing	0
Missing (%)	0.0%
Memory size	777.0 KiB

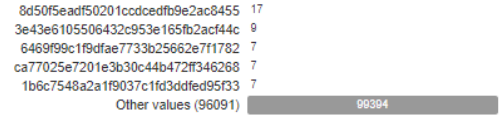


Toggle details

customer_unique_id
Categorical

HIGH CARDINALITY

Distinct count	96096
Unique (%)	96.6%
Missing	0
Missing (%)	0.0%
Memory size	777.0 KiB

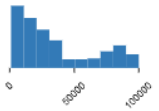


Toggle details

customer_zip_code_prefix
Real number (ℝ≥0)

Distinct count	14995
Unique (%)	15.1%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%

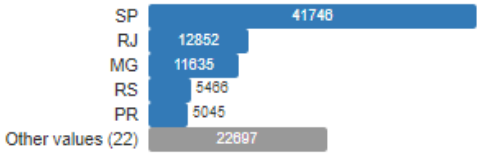
Mean	35137.47418
Minimum	1003
Maximum	99990
Zeros	0
Zeros (%)	0.0%
Memory size	777.0 KiB



Toggle details

customer_state
Categorical

Distinct count	27
Unique (%)	< 0.1%
Missing	0
Missing (%)	0.0%
Memory size	777.0 KiB



Toggle details

Common Values | Composition | Length

Value	Count	Frequency (%)
SP	41746	42.0%
RJ	12852	12.9%
MG	11635	11.7%
RS	5466	5.5%
PR	5045	5.1%
SC	3637	3.7%
BA	3380	3.4%
DF	2140	2.2%
ES	2033	2.0%
GO	2020	2.0%
Other values (17)	9487	9.5%

Datasets

- Para os datasets de olist_order_reviews_dataset.csv e olist_geolocation_dataset.csv foi necessária uma limpeza da base antes da utilização.

clean_dataset.py

work-at-olist-data-master > datasets > clean_dataset.py > ...

```python
import pandas as pd
import csv

i = 'datasets/olist_order_reviews_dataset.csv'
df = pd.read_csv(i, sep=',', index_col = False)
df['review_comment_message'].replace('\n' ,'', regex=True, inplace = True)
df['review_comment_message'].replace('\"' ,'', regex=True, inplace = True)
df.to_csv(i, index = False)


i = 'datasets/olist_geolocation_dataset.csv'
df = pd.read_csv(i, sep=',', index_col = False)
df['geolocation_zip_code_prefix'].drop_duplicates()
df.to_excel(i + '.xlsx', index=False, engine='xlsxwriter')
df.to_csv(i, index = False)
```

Microsoft Azure

Para a construção do data warehouse

- Sql Server
- Data Factory
- Blob Storage
- Analysis Services

Serviços Utilizados

Nome	Tipo
srvvmolist	SQL Server
dfvmolist	Data factory (V2)
savmolist	Conta de armazenamento
sqlvmolist (srvvmolist/sqlvmolist)	Banco de dados SQL
asvmolist	Analysis Services

Instância do SQL Server

Data Factory: responsável pela orquestração (ETL) dos datasets csv com o SQL Server

Blob Storage: responsável pelo armazenamento dos datasets (data lake)

sqlvmolist: banco de dados do SQL Server.

Analysis Services: responsável pela construção do data warehouse para ser consumido no power bi/demais ferramentas.

SQL Server com o banco de dados “sqlvmolist”

Grupo de recursos...	: rgvmolist	Administrador do servid...	: vmolist
Status	: Disponível	Firewalls e redes virtuais	: Mostrar configurações de firewall
Local	: Sul do Brasil	Administrador do Active...	: Não configurado
Assinatura (alterar)	: Avaliação Gratuita	Nome do servidor	: srvvmolist.database.windows.net
ID da Assinatura	: a877df22-65eb-4579-b018-52fc2d66ac4c		
Marcações (alterar)	: Projeto : Olist		

Notificações (0) | Recursos (6)

Todos | Segurança (4) | Desempenho (1) | Recuperação (1)

Administrador do Active Directory
Permite que você gerencie de forma centralizada a identidade e o acesso aos seus bancos de dados SQL do Azure.
NÃO CONFIGURADO

Segurança de dados avançada
Descoberta e Classificação de Dados, Avaliação de Vulnerabilidades e Proteção Avançada contra Ameaças.
NÃO CONFIGURADO

Ajuste automático
Monitora e ajusta o banco de dados automaticamente para otimizar o desempenho.
CONFIGURADO

Auditoria
Rastreie eventos do banco de dados e grave-os em um log de auditoria no armazenamento do Azure.
NÃO CONFIGURADO

Grupos de failover
Gerencia automaticamente a replicação, a conectividade e o failover de um conjunto de bancos de dados.
NÃO CONFIGURADO

Transparent Data Encryption
Criptografia em repouso para bancos de dados, backups e logs.
CHAVE GERENCIADA PELO SERVIÇO

Recursos disponíveis

Filtrar por nome | Todos os tipos

1 banco de dados

Nome	Tipo	Status	Camada de preços
Banco de dados SQL			
sqlvmolist	Banco de dados SQL	Online	Standard S0: 10 DTUs

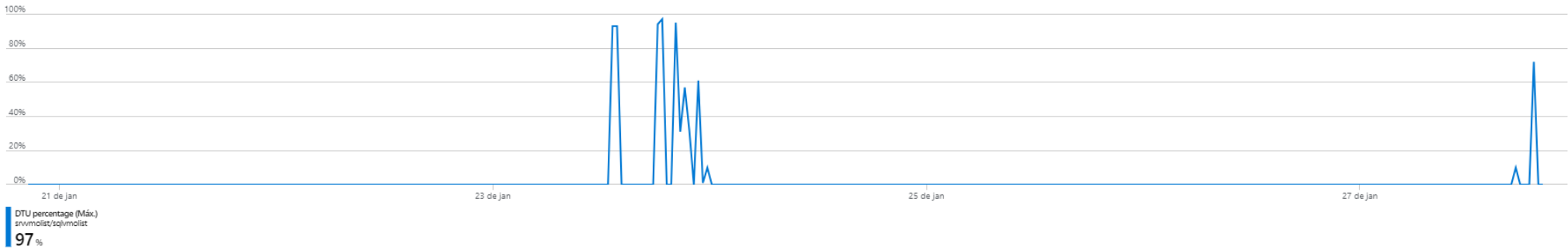
SQL Server com histórico de utilização

Grupo de recursos...	: rgvmolist	Nome do servidor	: srvvmolist.database.windows.net
Status	: Online	Pool elástico	: Nenhum pool elástico
Local	: Sul do Brasil	Cadeias de conexão	: Mostrar cadeias de conexão do banco de dados
Assinatura (alterar)	: Avaliação Gratuita	Camada de preços	: Standard S0: 10 DTUs
ID da Assinatura	: a877df22-65eb-4579-b018-52fc2d66ac4c	Ponto de restauração m...	: 2020-01-20 00:00 UTC
Marcações (alterar)	: Projeto : Olist		

Mostrar dados para o último: 1 hora 24 horas 7 dias

Tipo de agregação: M...

Utilização de computação



Armazenamento de dados do banco de dados

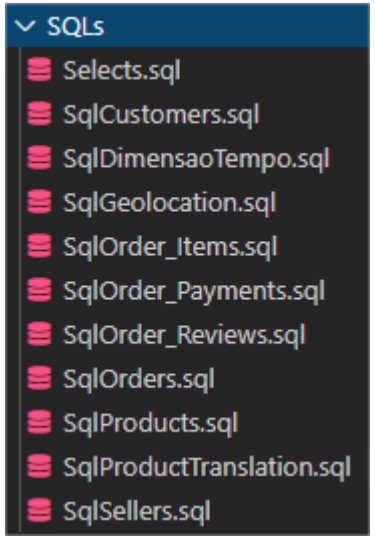
Espaço usado
107 MB

Espaço alocado
144 MB

Tamanho máximo do armazen...
10 GB

SQL Server

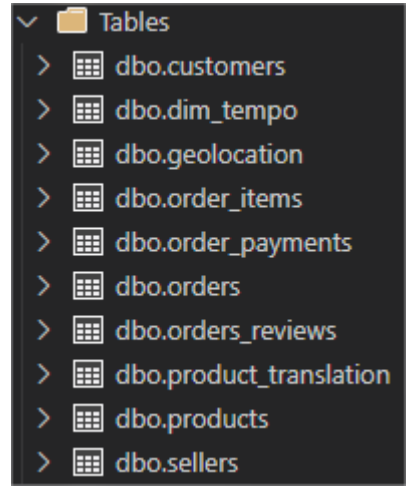
As tabelas foram geradas a partir do mapeamento de cada dataset.



work-at-olist-data-master > datasets > SQLs > SqlCustomers.sql

```sql
drop table if exists dbo.customers;
GO

create table customers(
    customer_id varchar(32) NOT NULL,
    customer_unique_id varchar(32),
    customer_zip_code_prefix int,
    customer_city varchar(32),
    customer_state varchar(2)
    PRIMARY KEY (customer_id)
);
GO
```



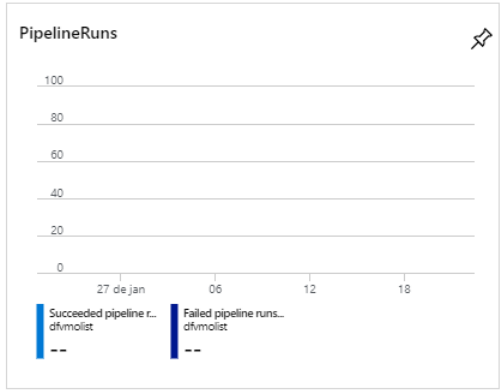
Data Factory

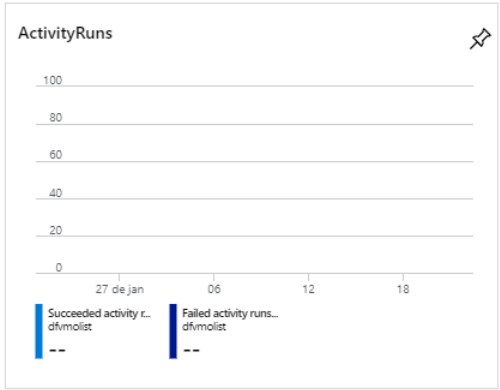
Grupo de recursos...	: rgvmolist	Tipo	: Data factory (V2)
Status	: Succeeded	Introdução	: Início rápido
Local	: Sul do Brasil		
Assinatura (alterar)	: Avaliação Gratuita		
ID da Assinatura	: a877df22-65eb-4579-b018-52fc2d66ac4c		

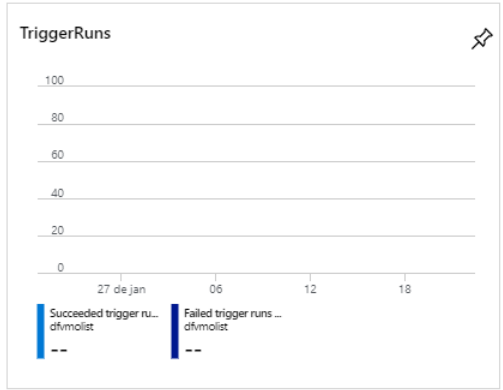
Documentação

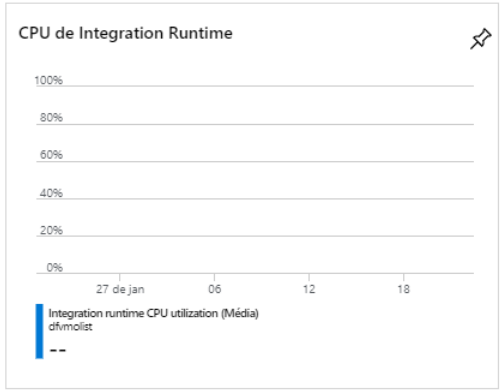
Autor & Monitor

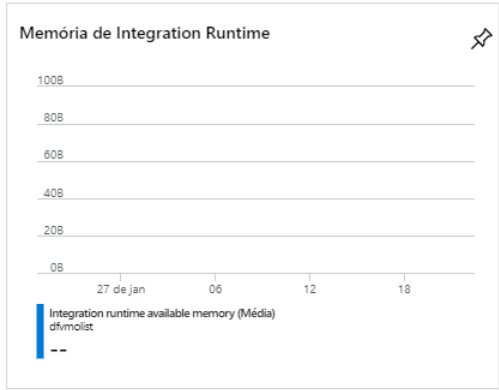
Monitoramento





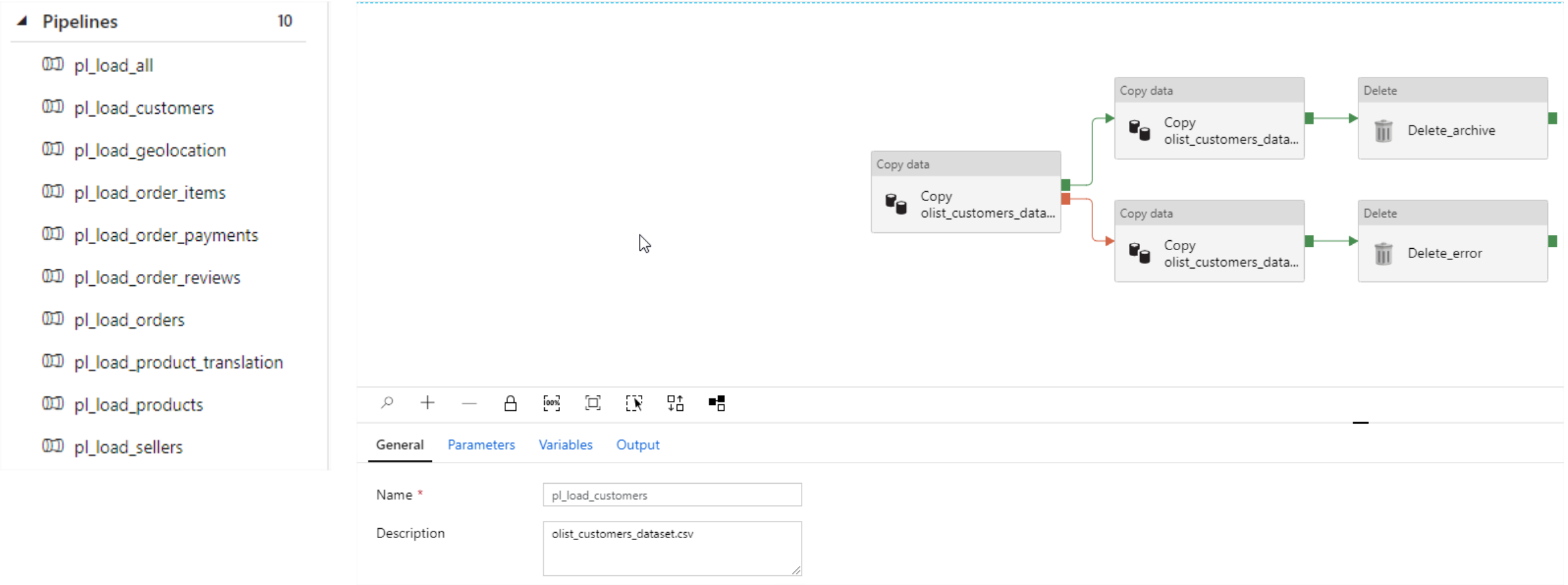






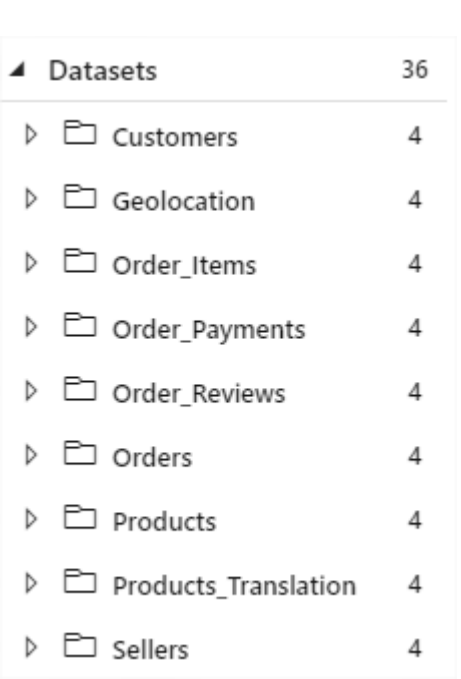
Data Factory - Pipeline

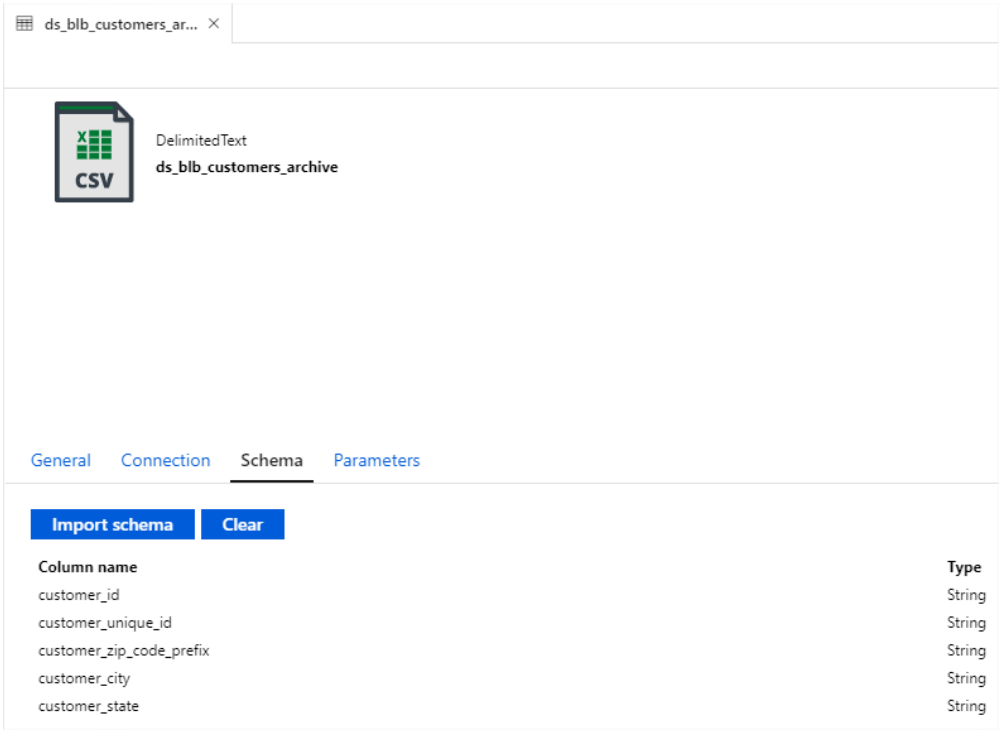
Pipeline serve para automatizar as implementações através de fluxos, nesse caso de copy data.

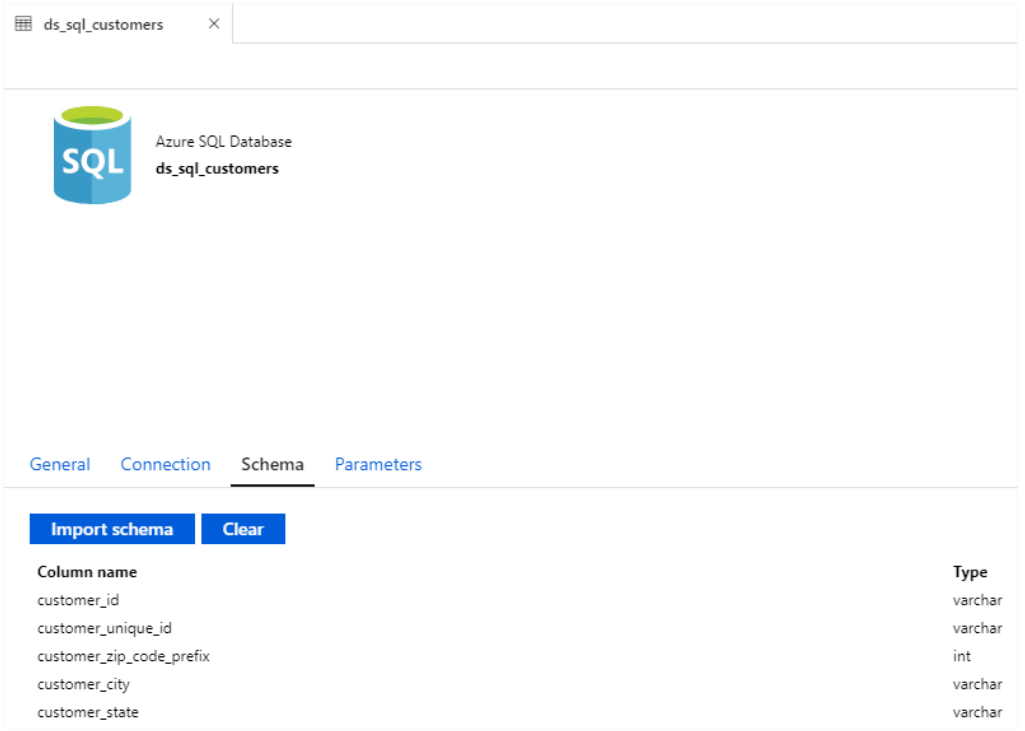


Data Factory - Datasets

Os datasets servem para configurar as bases de origem e as tabelas de destinos onde os dados serão inseridos/atualizados.

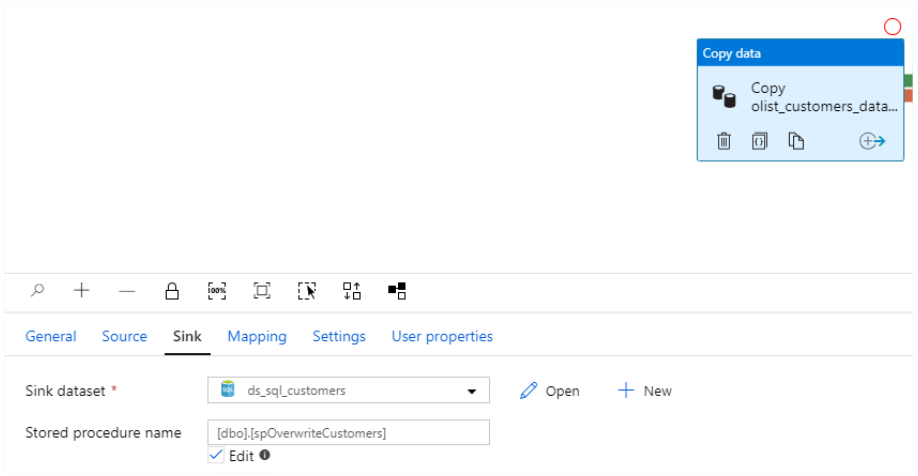






Data Factory – Pipeline com stored procedure

Serve para tratar os dados se necessário e realizar o match para insert ou updated.

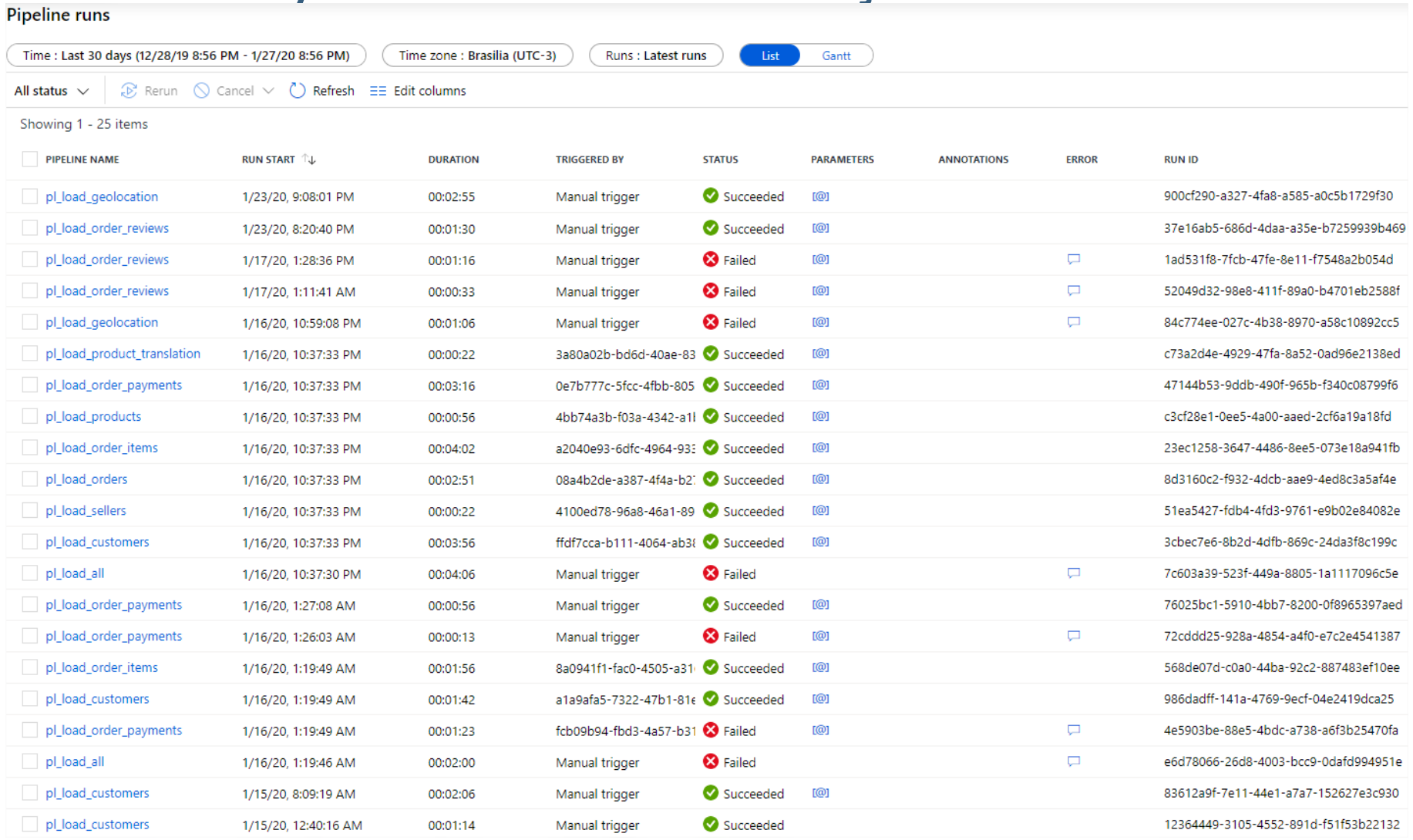



```sql
CREATE TYPE [dbo].[CustomersType] AS TABLE(
    customer_id varchar(32) NOT NULL,
    customer_unique_id varchar(32),
    customer_zip_code_prefix int,
    customer_city varchar(32),
    customer_state varchar(2)
    PRIMARY KEY (customer_id)
)
GO

CREATE PROCEDURE [dbo].[spOverwriteCustomers] @Customers dbo.CustomersType READONLY
    AS
    SET NOCOUNT ON

    BEGIN
    MERGE dbo.customers AS target
    USING (Select customer_id, customer_unique_id, customer_zip_code_prefix, customer_city, customer_state from @Customers) AS source
            (customer_id, customer_unique_id, customer_zip_code_prefix, customer_city, customer_state)
    ON (target.customer_id      = source.customer_id)
    WHEN MATCHED THEN
        UPDATE SET customer_id = source.customer_id,
        customer_unique_id = source.customer_unique_id,
        customer_zip_code_prefix = source.customer_zip_code_prefix,
        customer_city = source.customer_city,
        customer_state = source.customer_state
    WHEN NOT MATCHED THEN
        INSERT (customer_id, customer_unique_id, customer_zip_code_prefix, customer_city, customer_state)
        VALUES (source.customer_id, source.customer_unique_id, source.customer_zip_code_prefix, source.customer_city, source.customer_state);
    END
    SET NOCOUNT OFF
GO
```

Data Factory – Monitor de execução



Pipeline runs

Time : Last 30 days (12/28/19 8:56 PM - 1/27/20 8:56 PM) | Time zone : Brasilia (UTC-3) | Runs : Latest runs | List | Gantt

All status | Rerun | Cancel | Refresh | Edit columns

Showing 1 - 25 items

PIPELINE NAME	RUN START	DURATION	TRIGGERED BY	STATUS	PARAMETERS	ANNOTATIONS	ERROR	RUN ID
pl_load_geolocation	1/23/20, 9:08:01 PM	00:02:55	Manual trigger	Succeeded	[@]			900cf290-a327-4fa8-a585-a0c5b1729f30
pl_load_order_reviews	1/23/20, 8:20:40 PM	00:01:30	Manual trigger	Succeeded	[@]			37e16ab5-686d-4daa-a35e-b7259939b469
pl_load_order_reviews	1/17/20, 1:28:36 PM	00:01:16	Manual trigger	Failed	[@]			1ad531f8-7fcb-47fe-8e11-f7548a2b054d
pl_load_order_reviews	1/17/20, 1:11:41 AM	00:00:33	Manual trigger	Failed	[@]			52049d32-98e8-411f-89a0-b4701eb2588f
pl_load_geolocation	1/16/20, 10:59:08 PM	00:01:06	Manual trigger	Failed	[@]			84c774ee-027c-4b38-8970-a58c10892cc5
pl_load_product_translation	1/16/20, 10:37:33 PM	00:00:22	3a80a02b-bd6d-40ae-83	Succeeded	[@]			c73a2d4e-4929-47fa-8a52-0ad96e2138ed
pl_load_order_payments	1/16/20, 10:37:33 PM	00:03:16	0e7b777c-5fcc-4fbb-805	Succeeded	[@]			47144b53-9ddb-490f-965b-f340c08799f6
pl_load_products	1/16/20, 10:37:33 PM	00:00:56	4bb74a3b-f03a-4342-a1l	Succeeded	[@]			c3cf28e1-0ee5-4a00-aaed-2cf6a19a18fd
pl_load_order_items	1/16/20, 10:37:33 PM	00:04:02	a2040e93-6dfc-4964-933	Succeeded	[@]			23ec1258-3647-4486-8ee5-073e18a941fb
pl_load_orders	1/16/20, 10:37:33 PM	00:02:51	08a4b2de-a387-4f4a-b27	Succeeded	[@]			8d3160c2-f932-4dcb-aae9-4ed8c3a5af4e
pl_load_sellers	1/16/20, 10:37:33 PM	00:00:22	4100ed78-96a8-46a1-89	Succeeded	[@]			51ea5427-fdb4-4fd3-9761-e9b02e84082e
pl_load_customers	1/16/20, 10:37:33 PM	00:03:56	ffdf7cca-b111-4064-ab3	Succeeded	[@]			3cbec7e6-8b2d-4dfb-869c-24da3f8c199c
pl_load_all	1/16/20, 10:37:30 PM	00:04:06	Manual trigger	Failed				7c603a39-523f-449a-8805-1a1117096c5e
pl_load_order_payments	1/16/20, 1:27:08 AM	00:00:56	Manual trigger	Succeeded	[@]			76025bc1-5910-4bb7-8200-0f8965397aed
pl_load_order_payments	1/16/20, 1:26:03 AM	00:00:13	Manual trigger	Failed	[@]			72cddd25-928a-4854-a4f0-e7c2e4541387
pl_load_order_items	1/16/20, 1:19:49 AM	00:01:56	8a0941f1-fac0-4505-a31	Succeeded	[@]			568de07d-c0a0-44ba-92c2-887483ef10ee
pl_load_customers	1/16/20, 1:19:49 AM	00:01:42	a1a9afa5-7322-47b1-81e	Succeeded	[@]			986dadff-141a-4769-9ecf-04e2419dca25
pl_load_order_payments	1/16/20, 1:19:49 AM	00:01:23	fcb09b94-fbd3-4a57-b31	Failed	[@]			4e5903be-88e5-4bdc-a738-a6f3b25470fa
pl_load_all	1/16/20, 1:19:46 AM	00:02:00	Manual trigger	Failed				e6d78066-26d8-4003-bcc9-0dafd994951e
pl_load_customers	1/15/20, 8:09:19 AM	00:02:06	Manual trigger	Succeeded	[@]			83612a9f-7e11-44e1-a7a7-152627e3c930
pl_load_customers	1/15/20, 12:40:16 AM	00:01:14	Manual trigger	Succeeded				12364449-3105-4552-891d-f51f53b22132

Blob Storage

Onde ficam disponibilizados os datasets para serem atualizados no sql server

Archive: arquivamento dos datasets no dia e hora que foram atualizados no sql server

Local: archive / 2020 / 01 / 17

Pesquisar blobs por prefixo (diferenciar maiúsculas de minúsculas)

Nome	Modificado	Camada de acesso	Tipo de blob	Tamanho
[..]				
olist_customers_dataset._2020-01-17_01:37:33.CSV	16/01/2020 22:41:24	Principal (Inferidos)	Blob de blocos	9.21 MiB
olist_order_items_dataset._2020-01-17_01:37:33.CSV	16/01/2020 22:41:31	Principal (Inferidos)	Blob de blocos	15.92 MiB
olist_order_payments_dataset._2020-01-17_01:37:33.CSV	16/01/2020 22:40:45	Principal (Inferidos)	Blob de blocos	6.48 MiB
olist_orders_dataset_2020-01-17_01:37:33.CSV	16/01/2020 22:40:18	Principal (Inferidos)	Blob de blocos	18.2 MiB
olist_products_dataset_2020-01-17_01:37:33.CSV	16/01/2020 22:38:25	Principal (Inferidos)	Blob de blocos	2.82 MiB
olist_sellers_dataset_2020-01-17_01:37:33.CSV	16/01/2020 22:37:51	Principal (Inferidos)	Blob de blocos	186.95 KiB
product_category_name_translation_2020-01-17_01:37:33.CSV	16/01/2020 22:37:51	Principal (Inferidos)	Blob de blocos	2.83 KiB

Error: arquivamento dos datasets
que ocorreram erros na atualização para o sql server

Local: error / 2020 / 01 / 23

Pesquisar blobs por prefixo (diferenciar maiúsculas de minúsculas)

Nome
[..]
olist_geolocation_dataset_2020-01-23_23:28:23.CSV
olist_geolocation_dataset_2020-01-23_23:42:16.CSV
olist_geolocation_dataset_2020-01-23_23:48:14.CSV

Inbound: onde os datasets são carregados para
que o data factory possa carregar no SQL server

Local: inbound

Pesquisar blobs por prefixo (diferenciar maiúsculas

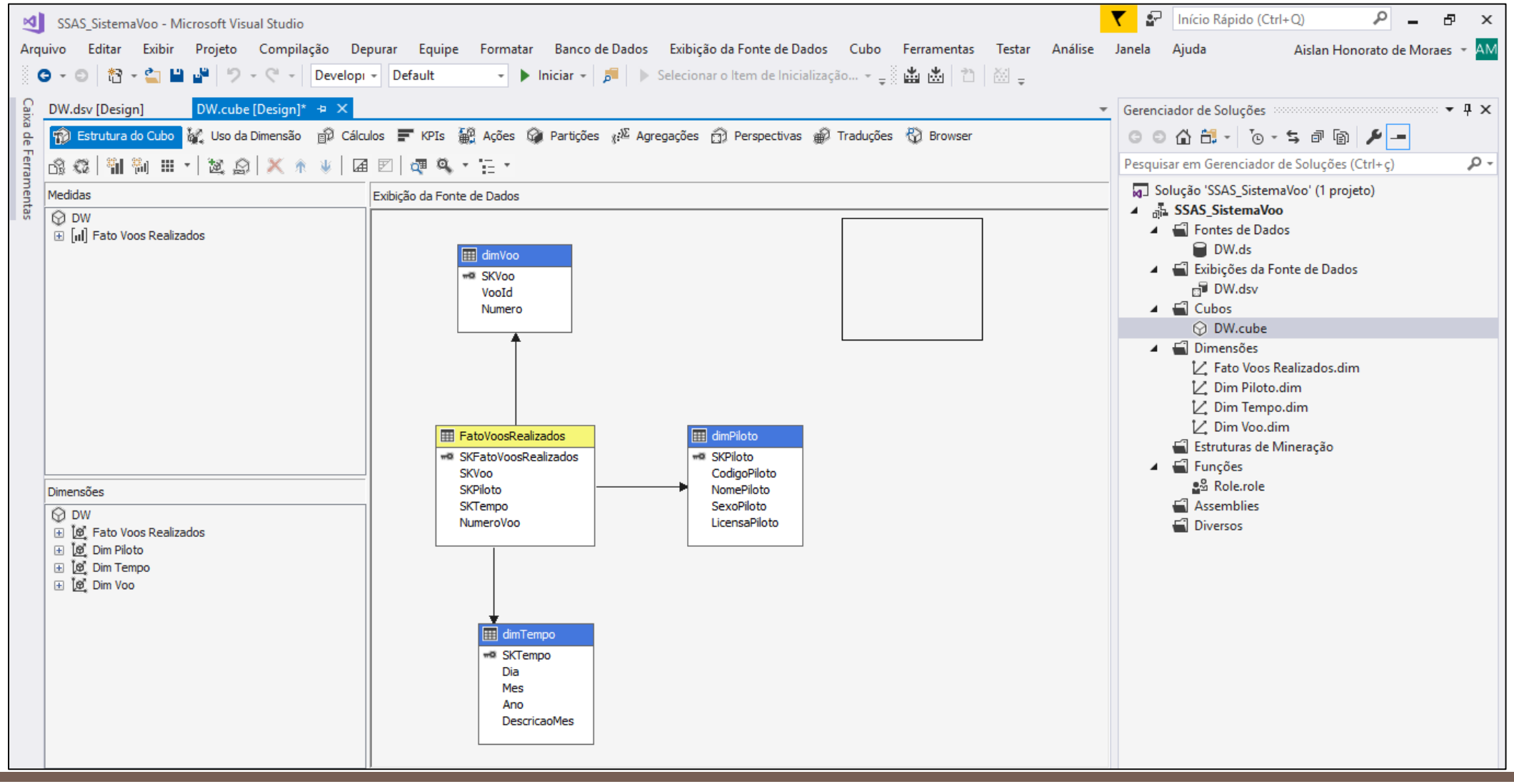
Nome
olist_customers_dataset.csv
olist_geolocation_dataset.csv
olist_order_items_dataset.csv
olist_order_payments_dataset.csv
olist_order_reviews_dataset.csv
olist_orders_dataset.csv
olist_products_dataset.csv
olist_sellers_dataset.csv
product_category_name_translation.csv

Uploads atuais

Ignorar: Concluído Tudo

product_category_name_translat...	3 KiB / 3 KiB
olist_sellers_dataset.csv	160 KiB / 160 KiB
olist_products_dataset.csv	2 MiB / 2 MiB
olist_orders_dataset.csv	17 MiB / 17 MiB
olist_order_reviews_dataset.csv	14 MiB / 14 MiB
olist_order_payments_dataset.csv	5 MiB / 5 MiB
olist_order_items_dataset.csv	14 MiB / 14 MiB
olist_geolocation_dataset.csv	57 MiB / 57 MiB
olist_customers_dataset.csv	8 MiB / 8 MiB

Analysis Services (Exemplo apenas)



Microsoft PowerBi

Para a construção do dashboard

Dashboard Olist

Vendas

olist

BUSCA RÁPIDA

product_category_name

Search

review_comment_message

Search

FILTROS

product_category_name

Todos

review_score

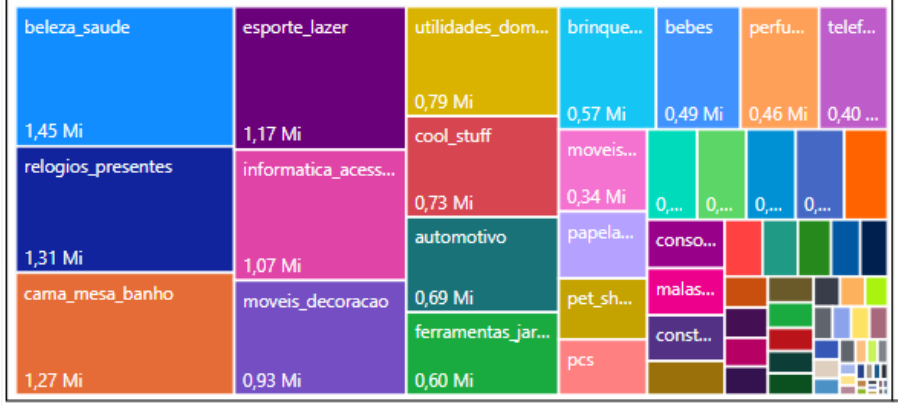
Todos

Ano

(Em branco) | 2016 | 2017 | 2018

TOTAL		Created		Processing		Shipped		Delivered	
103886	16,01 Mi	5	689,00	319	69,40 Mil	1166	177,21 Mil	100756	15,42 Mi
		0,0%	0,0%	0,3%	0,4%	1,1%	1,1%	97,0%	96,3%

Total Category Name



Review Comment Message

Top States

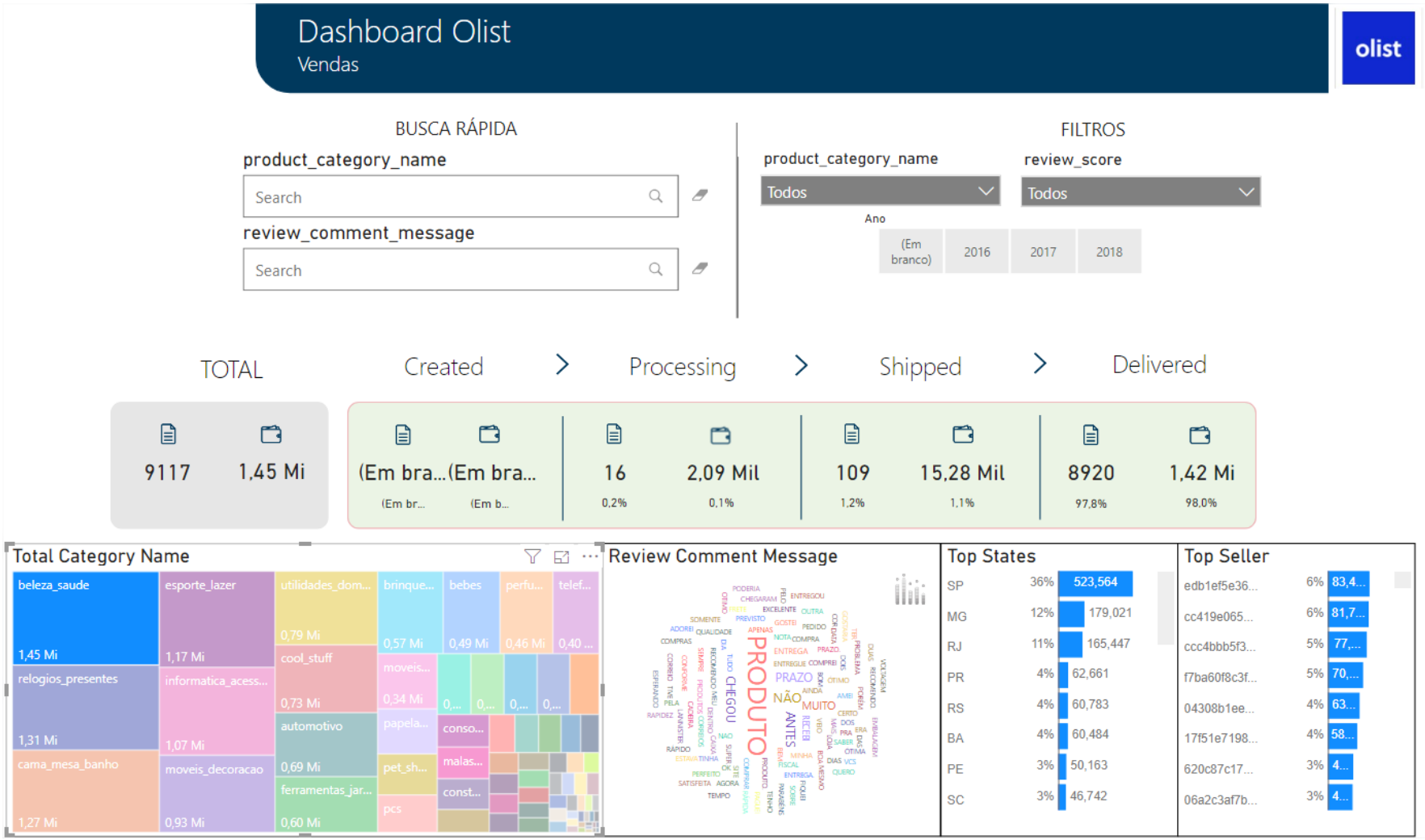
SP	37%	5,996,717
RJ	13%	2,142...
MG	12%	1,871,...
RS	6%	893,063
PR	5%	810,255
SC	4%	624,425
BA	4%	615,285
GO	2%	349,355

Top Seller

4869f7a5df...	2%	253,...
4a3ca9315...	2%	248,...
7c67e1448...	2%	241,...
53243585a...	1%	235...
fa1c13f261...	1%	20...
da8622b14...	1%	19...
7e93a43ef3...	1%	18...
1025f0e2d4...	1%	17...

payment_value por geolocation_state

Filtrando beleza_saude



Filtrando beleza_saude

Machine Learning

Para a construção do modelo de previsão de venda

Em construção

- O foco do teste aplicado é para a vaga de engenheiro de dados, porém como sou estudante de Pós em Ciência de Dados, deixei aberto esse slide como um desafio pessoal para o a aplicação das técnicas aprendidas.

```
#CONEXÃO COM O AZURE
import pyodbc as odbc
import pandas as pd

server = '[redacted].database.windows.net'
database = 'sql[redacted]'
username = 'vm[redacted]'
password = '[redacted]'
driver= '{ODBC Driver 17 for SQL Server}'
```

```
# CONSULTA dbo.v_Machine
SQL = "SELECT * FROM dbo.v_Machine;"
cnxn = odbc.connect('DRIVER='+driver+';SERVER='+server+';PORT=1433;DATABASE='+database+';Uid='
+username+';Pwd='+password+';Encrypt=yes;TrustServerCertificate=no;Connection Timeout=30;')
df = pd.read_sql_query(SQL, cnxn)
```

```
# Importando os modulos
from sklearn.model_selection import train_test_split # Dividir os dados em treino e teste
from sklearn.metrics import r2_score # Avaliar o r2 do modelo
from sklearn.preprocessing import LabelEncoder # Converter os dados categoricos em numericos
from sklearn.model_selection import GridSearchCV # Testar os melhores parametros
from xgboost import XGBRegressor # Modelo para Regressao

# Modulos para analise expolratoria
from IPython.core.pylabtools import figsize
import matplotlib.pyplot as plt
```

Login para o Microsoft Azure

- Favor solicitar