CONSTRUINDO E IMPLEMENTANDO UM PROJETO DE BIG DATA COM ENGENHARIA DE DADOS

JAIRO BERNARDES DA SILVA JÚNIOR
www.linkedin.com/in/jairobernardesjunior

Pós-Graduado em Big Data e Ciência de Dados
Pós-Graduado em Engenharia de Sistemas
Pós-Graduado em Gestão de TI

Graduado em Física

DESCRIÇÃO:

O projeto bigDVarejoPMC, PMC significa Pesquisa Mensal de Comércio, diz respeito a uma parte de um projeto maior o bigDVarejo, um trabalho que será composto por inúmeros projetos menores que alimentarão um repositório de Data Lake voltado para comportar dados do varejo. Esses dados poderão ser de diferentes origens, tanto gerados pela empresa quanto de ambientes externos a ela, permitindo ser tratados e armazenados de forma que ofereçam informações úteis e que possam ser usadas no apoio à tomada de decisões.

OBJETIVO:

Motivação:

O que levou a planejar o bigDVarejoPMC inicialmente foi a necessidade de revelar qual a situação da empresa com relação a evolução de vendas do varejo comparada às vendas do país e para isso descobriu-se a oferta de arquivos, disponibilizados pelo IBGE, que entregam dados dos percentuais de crescimento de vendas no varejo agrupados por UF (unidade da federação) como também por categoria de comércio e de produto. Entende-se aqui que se uma empresa tem um crescimento de vendas aquém do crescimento ocorrido no seu meio ambiente de atuação, algo está errado e deve ser identificado e revisto para a devida correção do problema. Essa necessidade de entender melhor o que está acontecendo e o que pode acontecer também motiva a encontrar o problema o que será possível através do cruzamento de outros dados que ajudarão na formação do Data Lake.

Aplicação Prática:

Os valores percentuais de crescimento do PMC revelam uma posição passada de como se comportou o crescimento de vendas por UF, por categoria de comércio e por categoria de produtos no Brasil, dessa forma, pode-se saber se a empresa esteve participando desse crescimento de vendas de varejo ou se esteve deficitária no período em questão. Se comparado a evolução dos percentuais de crescimento de vendas da empresa, nos períodos passados, com os percentuais do Brasil, traçando uma curva de desempenho tanto da empresa quanto do país, e ainda levando em consideração os números por estado e categorias comerciais e de produtos pode-se verificar o sucesso e ou fracasso desse crescimento de vendas no varejo.

Resultados Esperados:

A evolução dos valores percentuais de vendas no varejo oferecidas pelo IBGE torna-se aqui uma informação importante para balizar se a empresa está desempenhando um papel positivo ou negativo frente ao crescimento de vendas no varejo do país, podendo, dessa forma, oferecer um diferencial para que a equipe de data analytics possa defender sua tese sobre a posição da organização frente a esses importantes dados estratégicos. As informações aqui em questão estão apoiadas em dados oficiais do comércio varejista e disponibilizados pelo IBGE. São dados que visam fornecer aos inúmeros interessados como evoluiu o crescimento de vendas no varejo durante os períodos passados fornecidos. No caso do Data Lake do bigDVarejo, tratado nesse projeto, tem-se o foco no desempenho da empresa em relação ao crescimento das vendas varejistas dentro do todo que é o Brasil. Espera-se que essa comparação do crescimento de vendas revele se a empresa teve uma participação positiva, participando desse crescimento ou negativa, ficando à margem da oportunidade de crescer em suas vendas no varejo, e ainda, com esse diagnóstico descritivo permitir que a equipe de Analytics possa buscar mais informações para enriquecer sua tese apresentada em insights que serão divulgadas para a gestão estratégica da companhia.

MÉTODOS:

De acordo com a motivação já descrita anteriormente deu-se o início para que os primeiros passos fossem dados, conforme o problema apresentado, em direção ao estudo e implementação do bigDVarejoPMC:

- Verificar se as solicitações apresentadas são viáveis:
  - De acordo com as necessidades apresentadas fez-se uma análise preliminar para se certificar da viabilidade do projeto.

- Pesquisar dados relativo à evolução de vendas do varejo no Brasil:

- Conforme o problema apresentado procurou-se, através da máquina de busca do google, por arquivos que poderiam conter dados para atender as necessidades apresentadas pela empresa.
  Os dados mais viáveis foram encontrados no site do IBGE (https://www.ibge.gov.br/estatisticas/economicas/comercio/9227-pesquisa-mensal-de-comercio.html?=&t=downloads), correspondendo a tabelas de percentual de crescimento de vendas agrupados por UF, categoria de comércio e de produto.

- *Avaliar arquivos de dados disponibilizados pelo IBGE:*
  - Os dados oferecidos pelo site do IBGE foram analisados e validados se poderiam entregar o resultado de informação que era necessário para a comparação das vendas da companhia com as vendas do país.

- *Verificar a viabilidade de baixar esses arquivos via python:*
  - Outra verificação importante foi a de se certificar que os arquivos eleitos para o projeto pudessem ser baixados através de API e que estivessem disponíveis continuamente, inclusive com as atualizações.

- *Definir quais os formatos de arquivos serão utilizados no armazenamento:*
  - Os arquivos originais com os dados brutos/crus (raw) são arquivos .xls, planilhas excel baixadas do site do IBGE, que serão transformadas em arquivos .csv e posteriormente em arquivos parquet. A ideia é armazenar os arquivos brutos, em .csv, para uma possível revisão de estrutura e os arquivos parquet com os 2 tipos de tabelas separadas, UF e Categorias de comércio e produto, que serão catalogados pelo Glue Data Catalog, podendo os dados e informações serem disponibilizados para o usuário final data analysts, data scientists ou mesmo data engineers.

- *Definir onde os arquivos serão armazenados, em local ou cloud:*
  - O local onde os arquivos ficarão armazenados tem como peso principal a continuidade do armazenamento com boa performance de acesso. Se tivesse a estrutura montada na empresa, com profissionais disponíveis para a manutenção dessa estrutura, poder-se-ia fazer uso da mesma, mas como está-se começando a montar o Data Lake e a empresa não tem nada disso ainda, optou-se por terceirizar a estrutura de armazenamento como também algumas ferramentas de acesso aos dados, assim elegeu-se o aws S3 para armazenamento, que oferece toda a manutenção da estrutura, com armazenamento distribuído e alta escalabilidade, deixando os engenheiros de dados livres para se preocuparem somente com o planejamento e operação do ELT.

- *Definir quantos S3 bucket serão criados para o armazenamento:*
  - Definido que o armazenamento dos arquivos ficará no S3 bucket e como cada bucket é praticamente uma pasta, um diretório, buscando organizar os arquivos por tipo, ficou definido que o Data Lake terá, inicialmente 3 buckets sendo definido os nomes: arquivosPMCrawS3 (dados brutos .xls), arquivosPMCprocessedS3 (dados transformados json) e arquivosPMCcuratedS3 (tabelas hive/parquet).

- *Definir quantas camadas serão necessárias para o processamento:*

  O processamento será feito em 3 partes distintas ou 3 camadas sequenciais que são:
  - *pmcBRONZE:*

Baixa os arquivos pmc.xls (pesquisa mensal de comércio) contendo o % de crescimento das vendas por categoria comercial e UF. Transforma esse arquivo para .csv e faz um load desse arquivo para o bucket arquivosPMCrawS3, onde ficarão armazenados para uma possível reestruturação dos dados e novo processamento.

- *pmcSILVER*:

  Transforma os arquivos .csv que estão no bucket arquivosPMCrawS3, separa as informações de UF das informações de categoria comercial e grava em arquivos parquet em 2 tipos de tabelas uma de UF e outra de categoria comercial e de produto. Faz o load dos arquivos parquet para o bucket arquivosPMCprocessedS3 para posterior catalogação.

- *pmcGOLD*:

  Os dados gravados em parquet no bucket arquivosPMCprocessedS3 agora são catalogados pelo Glue Data Catalog ficando acessíveis pelas ferramentas de pesquisas, no caso o aws athena e aws quickinsight.

- *Definir qual a linguagem será utilizada para o processamento dos dados:*

  Por conter uma enorme diversidade de bibliotecas para inúmeros fins, apresentar uma simplicidade de estrutura voltada para orientação a objetos, por ser uma linguagem que está sendo muito utilizada no mundo sendo uma tendência em manipulação de dados, apresentando funções voltadas para tal, por ser de fácil uso dentro da aws, optou-se em utilizar a linguagem python.

- *Definir onde serão executados os códigos:*

  Os códigos serão processados utilizando-se o serviço da aws Lambda, que é orientada a eventos com computação sem servidor, não é necessário definir um servidor para executar uma aplicação ficando transparente para nosso processamento, sendo mais uma preocupação para a equipe da aws Amazon manter o serviço funcionando com escalabilidade.

- *Definir onde será o repositório de hospedagem dos códigos:*

  Devido à experiência com o Git-Hub, por apresentar seus recursos de hospedagem e manutenção de versões de código com simplicidade e objetividade, pela sua divulgação e utilização na comunidade de desenvolvimento de software, definiu-se pela utilização dessa plataforma.

- *Definir as bibliotecas utilizadas:*
  - Para fazer upload dos arquivos baixados e gerados utilizar-se-á a biblioteca python boto3 (facilita o acesso aos serviços da aws).

- *Definir as características e configurações do scheduler*:
  - O processamento das 3 camadas serão executados às quartas e sextas-feiras às 20:00.

PRODUTOS, SERVIÇOS E SISTEMAS:

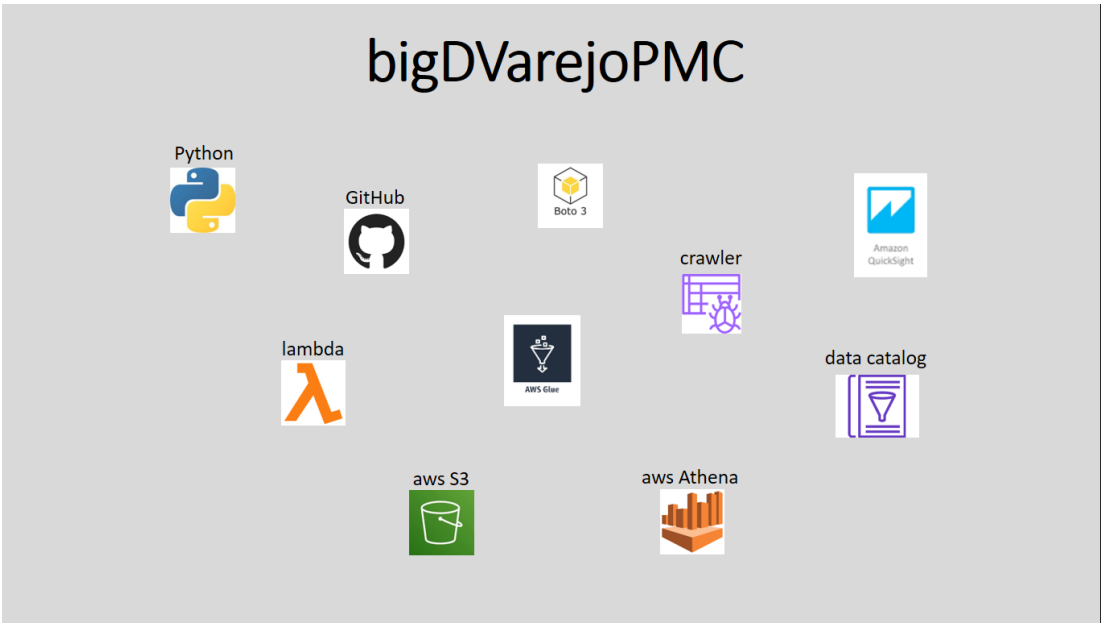


Fig-01

ESCOPO DA ARQUITETURA:

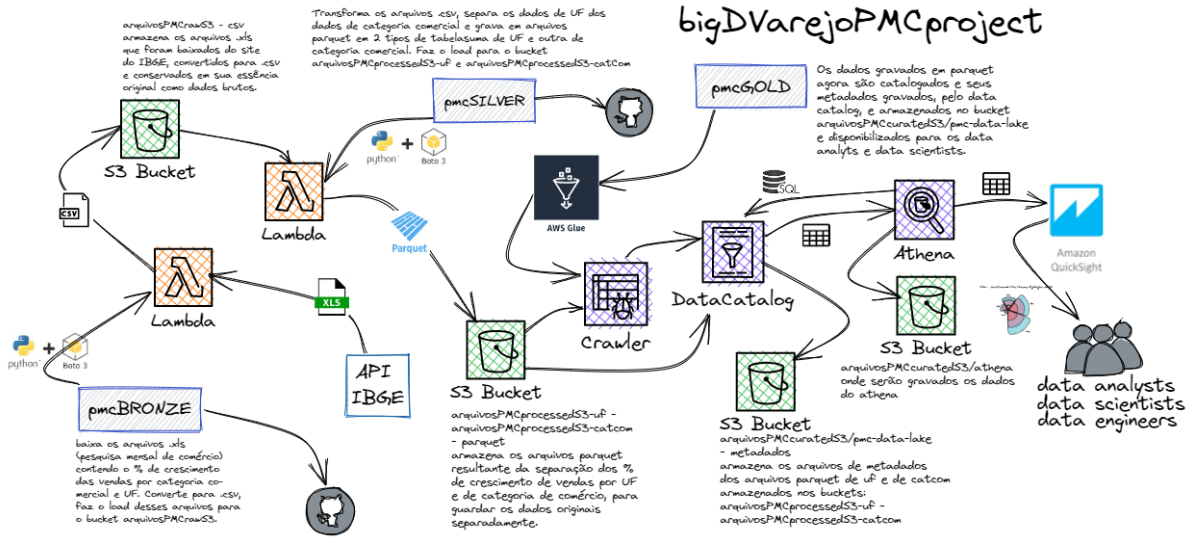


Fig-02

IMPLEMENTANDO AS CAMADAS COM CÓDIGO PYTHON:

Nas duas camadas iniciais pmcBRONZE e pmcSILVER foi utilizado a linguagem python que fará a extração e as primeiras transformações dos dados e arquivos para disponibilizar os mesmos para a camada pmcGOLD, onde os arquivos parquet serão catalogados pelo Glue Data Catalog, criando os metadados que serão responsáveis por viabilizar o acesso aos dados que estão nas tabelas parquet dentro dos buckets s3 (fig-02).

Todo o código está disponível no endereço do github:

https://github.com/jairo2016/bigDVarejo_PMCproject

IMPLEMENTANDO A CAMADA pmcBRONZE:

A camada pmcBRONZE será responsável pela extração dos dados do site do IBGE utilizando o endereço http:
https://ftp.ibge.gov.br/Comercio_e_Servicos/Pesquisa_Mensal_de_Comercio/Tabelas/2018/pmc_201801_00.xls

Os nomes dos arquivos seguem uma ordem de 12 meses e para cada mês uma sequência de 1 a 13 será montada para a leitura do arquivo .xls que conterá os percentuais de crescimento de vendas no varejo do Brasil, sendo esses dados apresentados agrupados por UF (unidade da federação) e por categoria de comércio e produtos.

Os dados são baixados e carregados em memória:

```
    with eventlet.Timeout(None):
        arqXLS = requests.get(url, verify = False)
        with open(tmpAux + url[85:103], 'wb') as pmcArq:
            pmcArq.write(arqXLS.content)
```

Os dados são convertidos em arquivo .csv:

```
def ConvertArquivoParaCSV(url, tmpAux):
    arquivoxls = pd.read_excel(tmpAux + url[85:103])
    arquivoxls.to_csv(tmpAux + url[85:99] + 'csv', index=False)
```

Os arquivos .csv, que contêm os dados crus (raw), são carregados para o bucket s3 arquivos-pmc-raws3 onde ficarão disponíveis para o processamento da camada pmcSILVER:

```
def UploadCSVfile_arquivosPMCrawS3(NomeBucketS3, nomeArquivo, pathArquivo):
    client = boto3.client(
        service_name='s3',
        aws_access_key_id='xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx',
        aws_secret_access_key='xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx',
        region_name='eu-west-1' # voce pode usar qualquer regiao
        )

    client.upload_file(pathArquivo, NomeBucketS3, nomeArquivo)
```

Por fim o controle do último arquivo baixado é feito gravando a URL do mesmo em um arquivo ultimoBaixado.txt, para que continue a partir de onde parou no próximo processamento:

```
def SalvaUltimaURL(patharquivoUltimoBaixado, ultimaURL):
    arquivo = open(patharquivoUltimoBaixado,'w')
    arquivo.write(ultimaURL)
    arquivo.close
```

IMPLEMENTANDO A CAMADA pmcSILVER:

Os dados que estão nos arquivos crus .csv no bucket s3 arquivos-pmc-raws3 agora precisam ser lidos e transformados em 2 arquivos parquet, um com a tabela de percentual de crescimento de vendas no varejo por UF (unidade da federação) e outro de percentual de crescimento de vendas no varejo por categoria de comércio e produtos, serão armazenados nos buckets s3 arquivos-pmc-processeds3-uf e arquivos-pmc-processeds3-catcom respectivamente, ficando disponíveis para a camada pmcGOLD feita sob o AWS Glue.

Os arquivos .csv são lidos do bucket s3 arquivos-pmc-raws3 e carregados em memória:

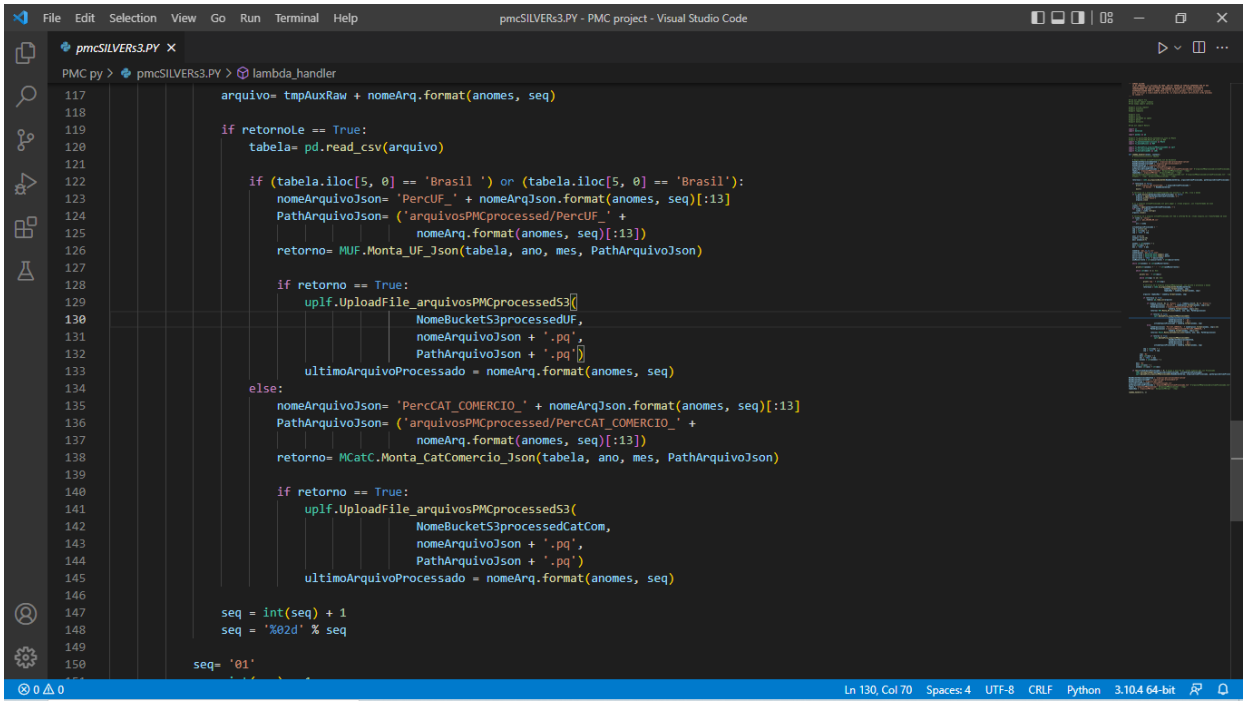
```
def Le_arquivosBucketS3(NomeBucketS3, arquivoUltimoProcessado,
patharquivoUltimoProcessado):
    client = boto3.client(
        service_name='s3',
        aws_access_key_id='xxxxxxxxxxxxxxxxxxxxxxxxx',
        aws_secret_access_key='xxxxxxxxxxxxxxxxxxxxxxxxx',
        region_name='eu-west-1' # voce pode usar qualquer regiao
        )

    retorno = False

    try:
        client.download_file(NomeBucketS3, arquivoUltimoProcessado,
patharquivoUltimoProcessado)
        retorno = True
    except botocore.exceptions.ClientError as e:
        if e.response['Error']['Code'] == "404":
            retorno = False

    return retorno
```

A partir do arquivo .csv em memória os dados das tabelas de UF e de categoria de comércio e produtos são separados em 2 tabelas distintas:



Em seguida, é gerado um arquivo parquet com dados da UF e outro com dados de categoria de comércio e produtos que serão armazenados nos buckets s3 arquivos-pmc-processeds3-uf e arquivos-pmc-processeds3-catcom:

```
def Monta_CatComercio_Json(tabela, ano, mes, PathArquivoJson):
    qtd_linhas = 19 #tabela.shape[0] - 1
    linhaXLS= 5
    i=0

    registro= []
```

```
    descricao= []
    anomes= []

    m3mensal= []

    while linhaXLS <= qtd_linhas:
        valor= tabela.iloc[linhaXLS, 1]

        if str(valor) == '-':
            valor = 0

        try:
            float(valor)
        except ValueError:
            break

        try:
            testa= tabela.iloc[5, 7]
        except IndexError:
            break

        i= i+1
        registro.append(i)
        descricao.append(tabela.iloc[linhaXLS, 0])
        anomes.append(str(ano) + str(mes))

        m3mensal.append(str(tabela.iloc[linhaXLS, 6]).replace('- ','0'))

        linhaXLS=linhaXLS+1

    if i>0:
        df=pd.DataFrame({
                "registro":registro,
                "classe_comercio":descricao,
                "ano_mes":anomes,

                "m3mensal":m3mensal,
                })

        df.to_parquet(PathArquivoJson + '.pq')
        #df.to_string(PathArquivoJson + '.txt')
        #df.to_json(PathArquivoJson + '.json')
        #df.to_csv(PathArquivoJson + '.csv')
        return True
    else:
        return False
```

Cada arquivo parquet gerado é carregado no bucket s3 sendo que para as tabelas de UF guardado no bucket s3 arquivos-pmc-processeds3-uf e para as tabelas de categoria de comércio e produtos guardado no bucket s3 arquivos-pmc-processeds3-catcom:

```
def UploadFile_arquivosPMCprocessedS3(NomeBucketS3, nomeArquivo, pathArquivo):
    client = boto3.client(
```

```
        service_name='s3',
        aws_access_key_id='xxxxxxxxxxxxxxxxxxxxxxxxxx',
        aws_secret_access_key='xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx',
        region_name='eu-west-1' # voce pode usar qualquer regiao
        )

    client.upload_file(pathArquivo, NomeBucketS3, nomeArquivo)
```

Por fim o controle do último arquivo processado é feito gravando o nome do mesmo em um arquivo ultimoProcessado.txt, para que continue a partir de onde parou no próximo processamento:

```
def SalvaUltimaURL(patharquivoUltimoProcessado, ultimaURL):
    arquivo = open(patharquivoUltimoProcessado,'w')
    arquivo.write(ultimaURL)
    arquivo.close
```

Agora os arquivos de percentuais de crescimento de vendas de UF e de categoria de comércio e produtos estão disponíveis para a camada pmcGOLD para serem catalogados pelo Glue Data Catalog e finalmente estarem disponíveis para os profissionais de insights.

IMPLEMENTANDO A CAMADA pmcGOLD:

A camada pmcGOLD é toda implementada no serviço AWS na nuvem através do Glue da amazon.

Primeiramente cria-se um database no AWS Glue que irá conter as tabelas de metadados, que conterão informações sobre os dados dos arquivos parquet nos buckets arquivos-pmc-processeds3-uf e arquivosPMCprocesseds3-catcom. Foram utilizadas as informações que estão na fig-03 abaixo tal como name, location e description.

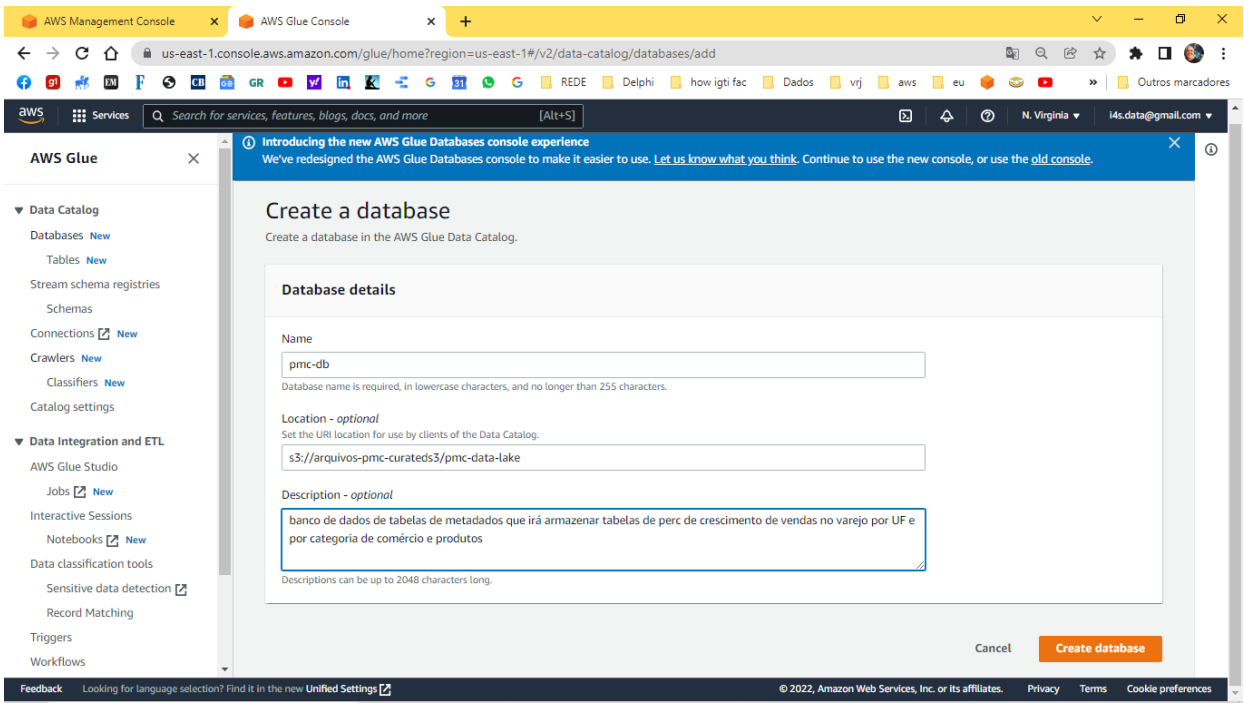


Fig-03

Em seguida cria-se um crawler que irá fornecer dados para a geração dos schemas das tabelas de metadados que serão catalogados e apontados para as tabelas parquet que estão nos buckets s3 arquvios-pmc-processeds3-uf e arquivos-pmc-processeds3-catcom. Foram utilizadas as informações da fig-04. Primeiramente será feito o procedimento para os arquivos de percentuais de UF, sendo que depois, o mesmo procedimento deve ser feito para os arquivos de categoria de comércio e produtos.

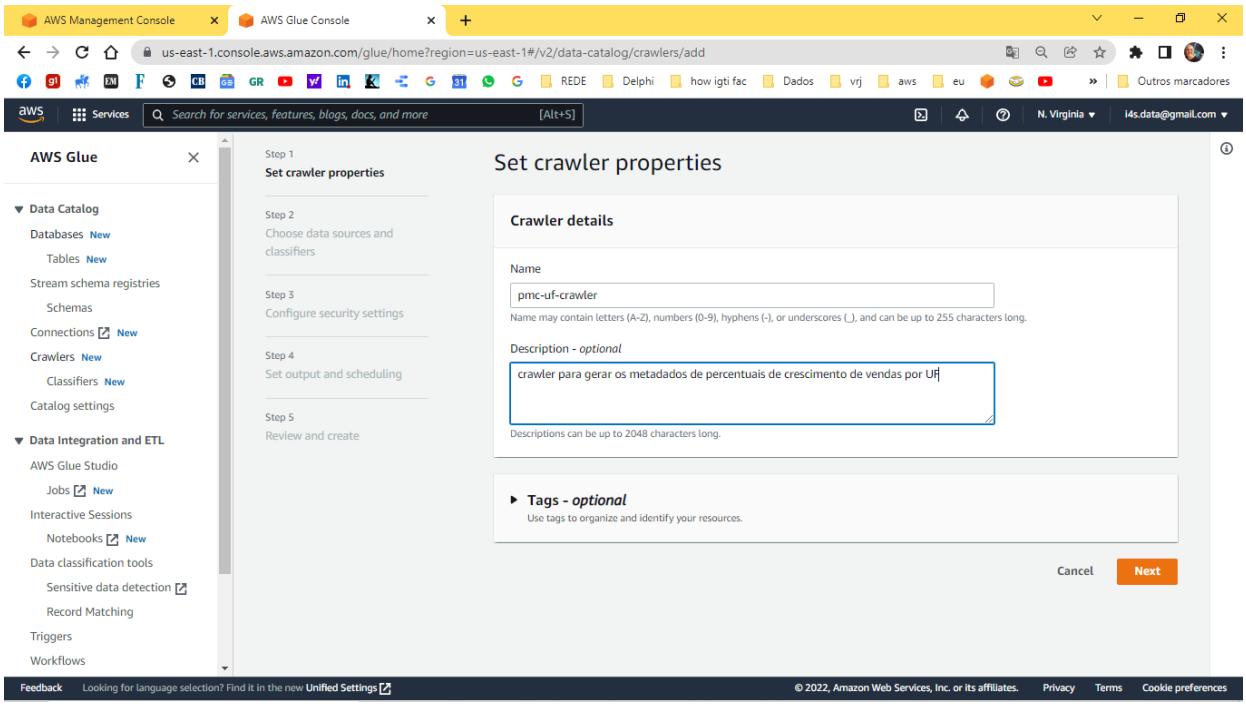


Fig-04

Adiciona-se agora uma fonte de dados (data source) que vai fornecer os dados para a montagem dos schemas, no nosso caso a fonte de dados é o bucket s3 arquivos-pmc-processseds3-uf que contêm os arquivos parquet transformados. Foram utilizadas as informações da fig-05.

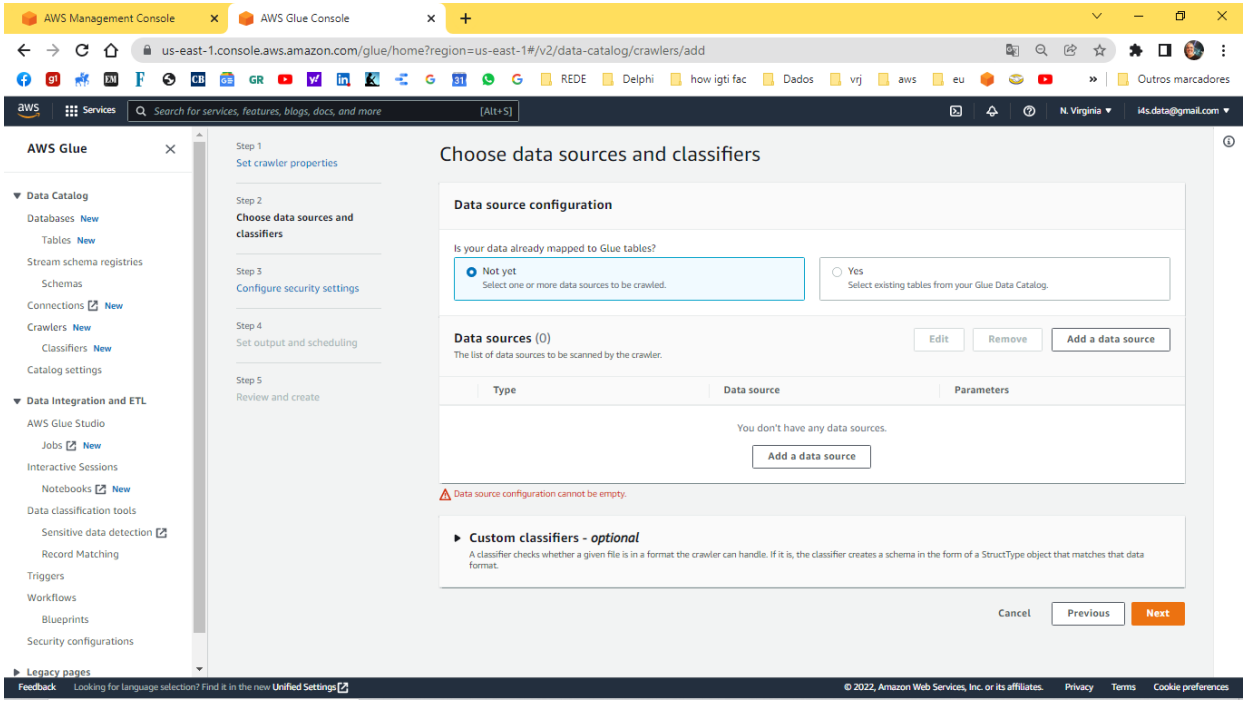


Fig-05

Configurando o data source foram utilizados os dados da fig-06 abaixo.

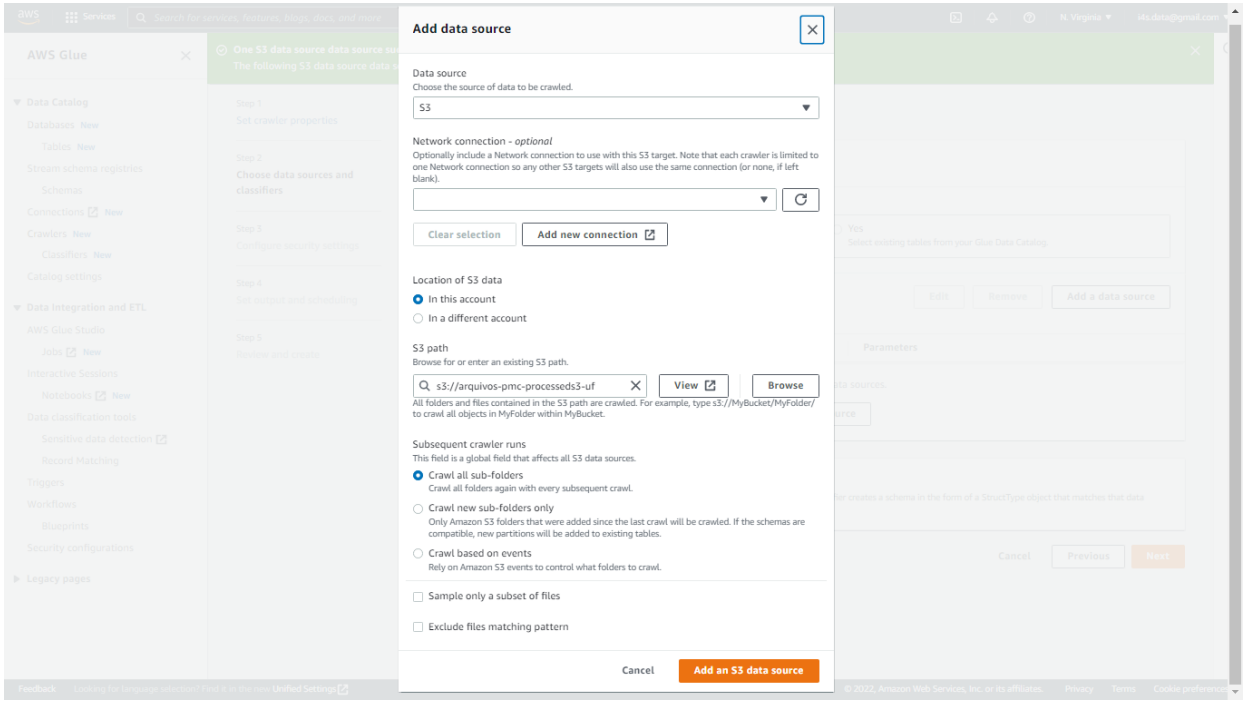


Fig-06

Capturado o data source, segue como na fig-07.

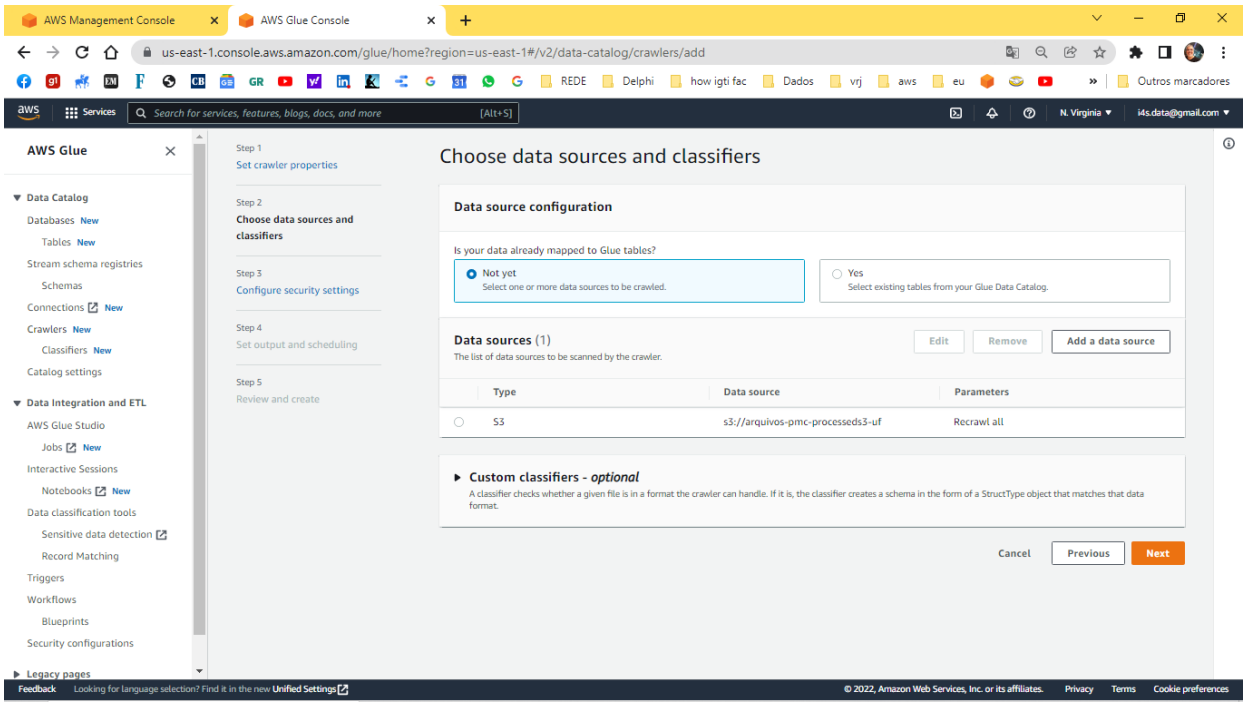


Fig-07

Criando ou escolhendo uma IAM role (regra), caso ainda não foi criado uma regra click no botão 'create new IAM role' e crie uma. Foram utilizados os dados da fig-08.

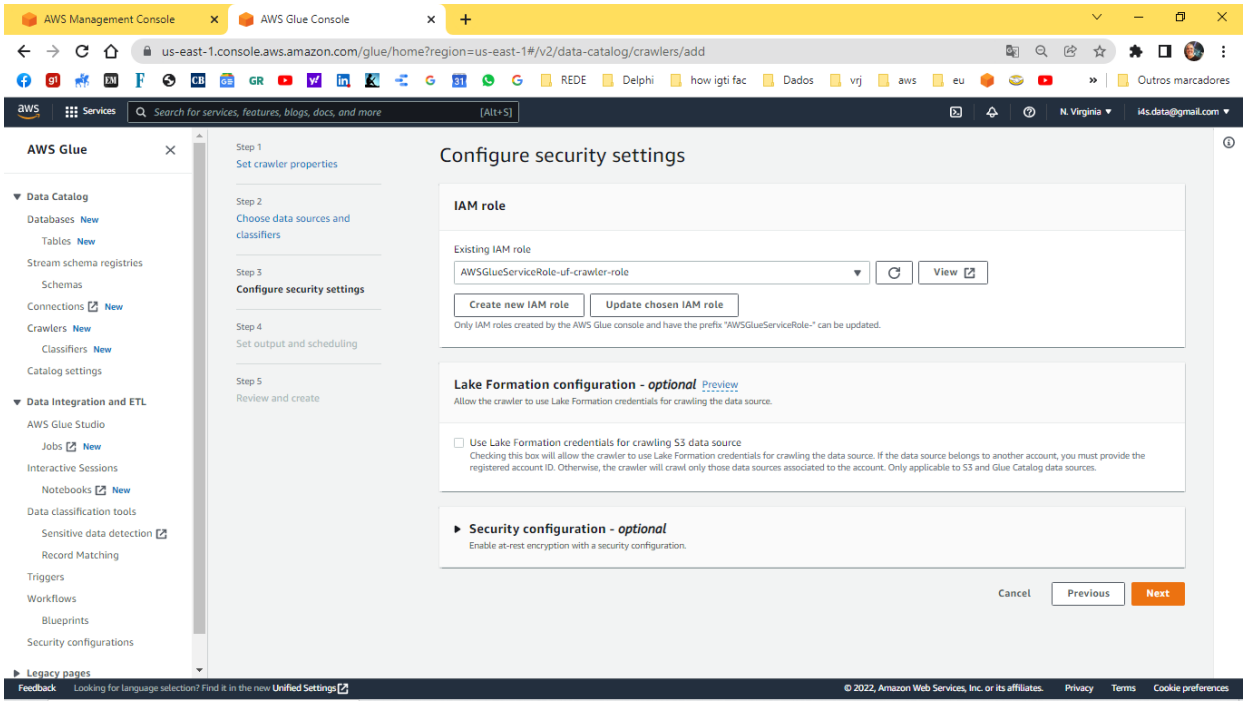


Fig-08

Informe o database que foi criado anteriormente como na fig-09.

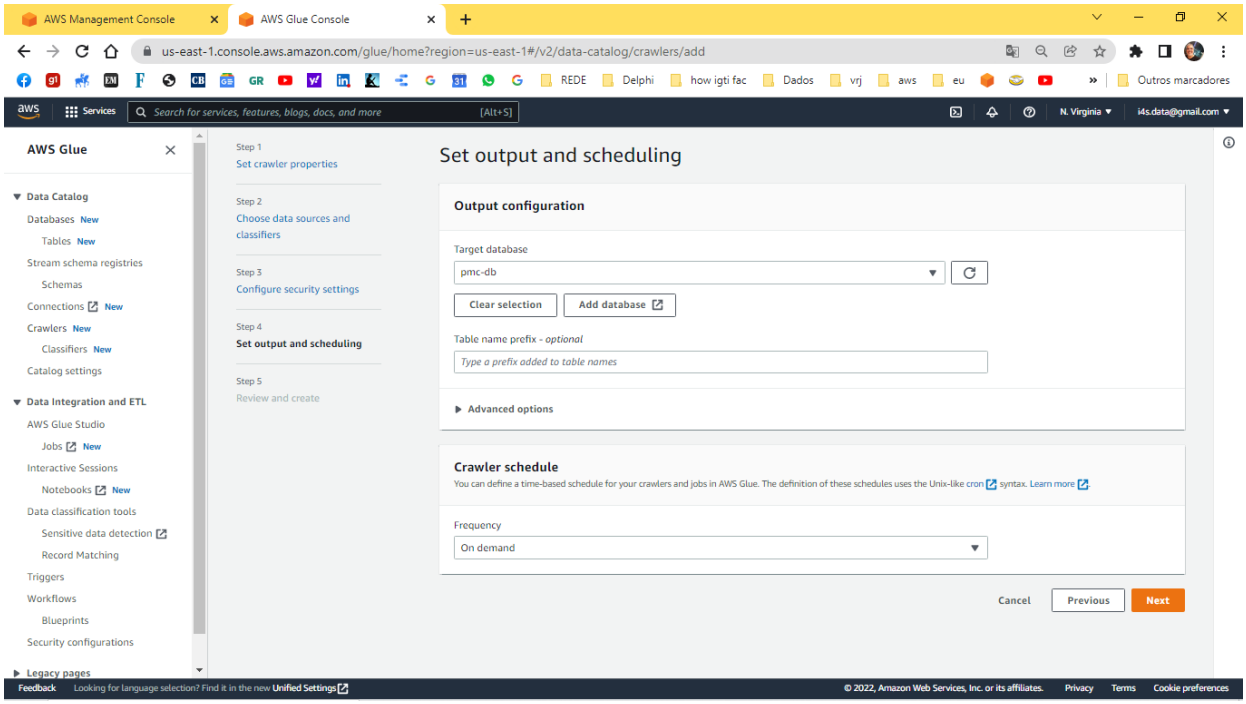


Fig-09

Agora basta conferir os dados e criar o crawler pmc-uf-crawler como na fig-10.

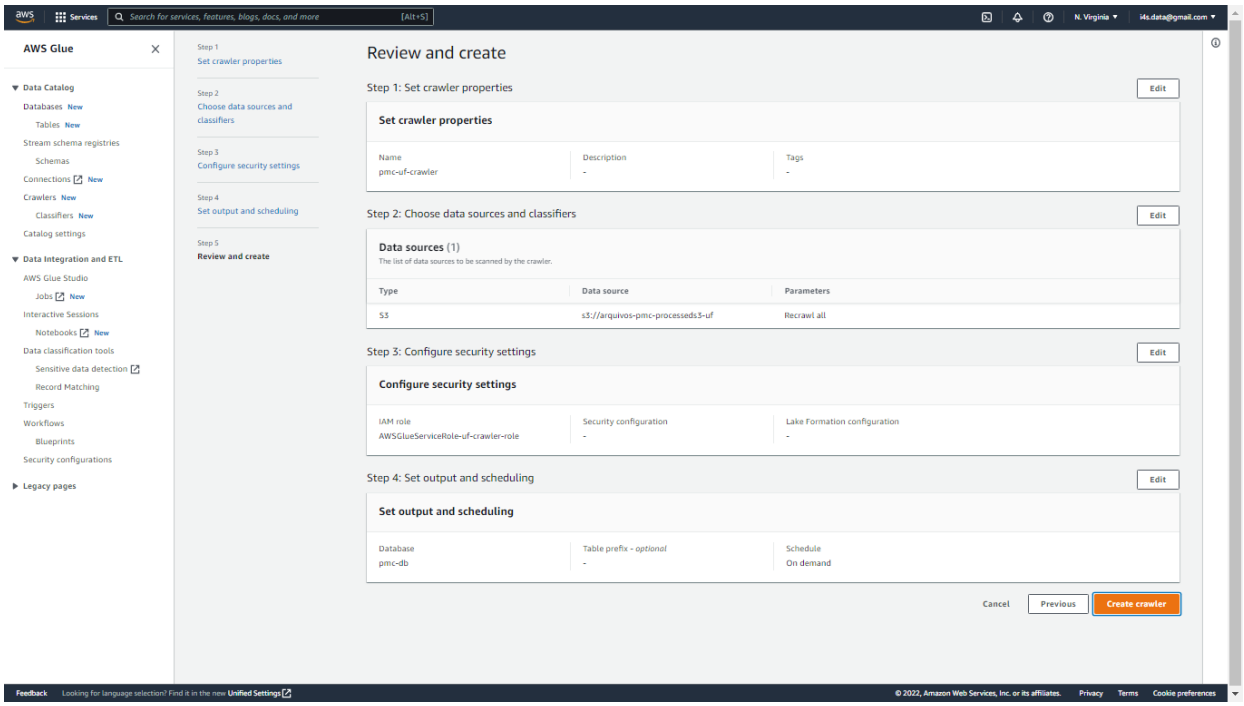


Fig-10

Agora com o crawler pmc-uf-crawler criado é preciso executar o mesmo para que a tabela de metadados com seu schema seja gerado, clicar na tecla run como na fig-11.

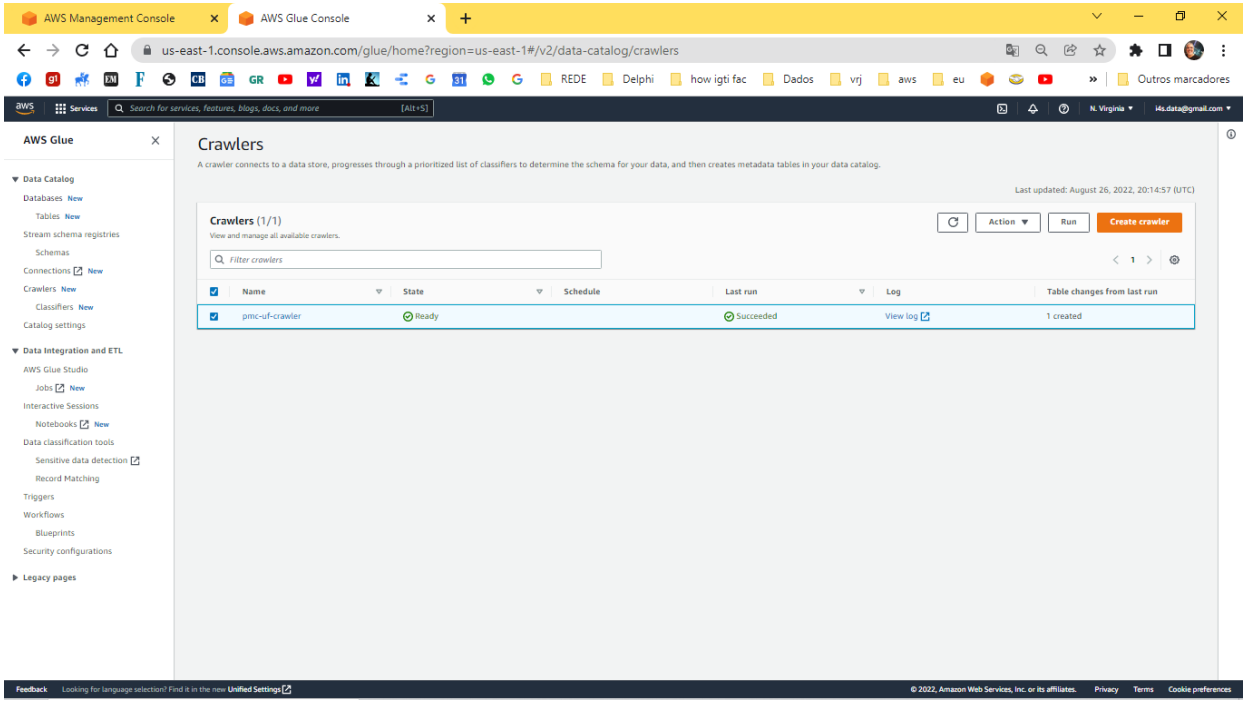


Fig-11

Após a execução do pmc-uf-crawler verfica-se em tables que uma tabela arquivos_pmc_processed foi criada no pmc-db(database) e esta tabela contém o schema e os metadados, que foram catalogados pelo data catalog e que estão apontados para os arquivo parquet no bucket s3 arquivos-pmc-processeds3-uf que contêm os dados de percentuais de crescimento das vendas do varejo no Brasil por UF. Veja na fig-12.

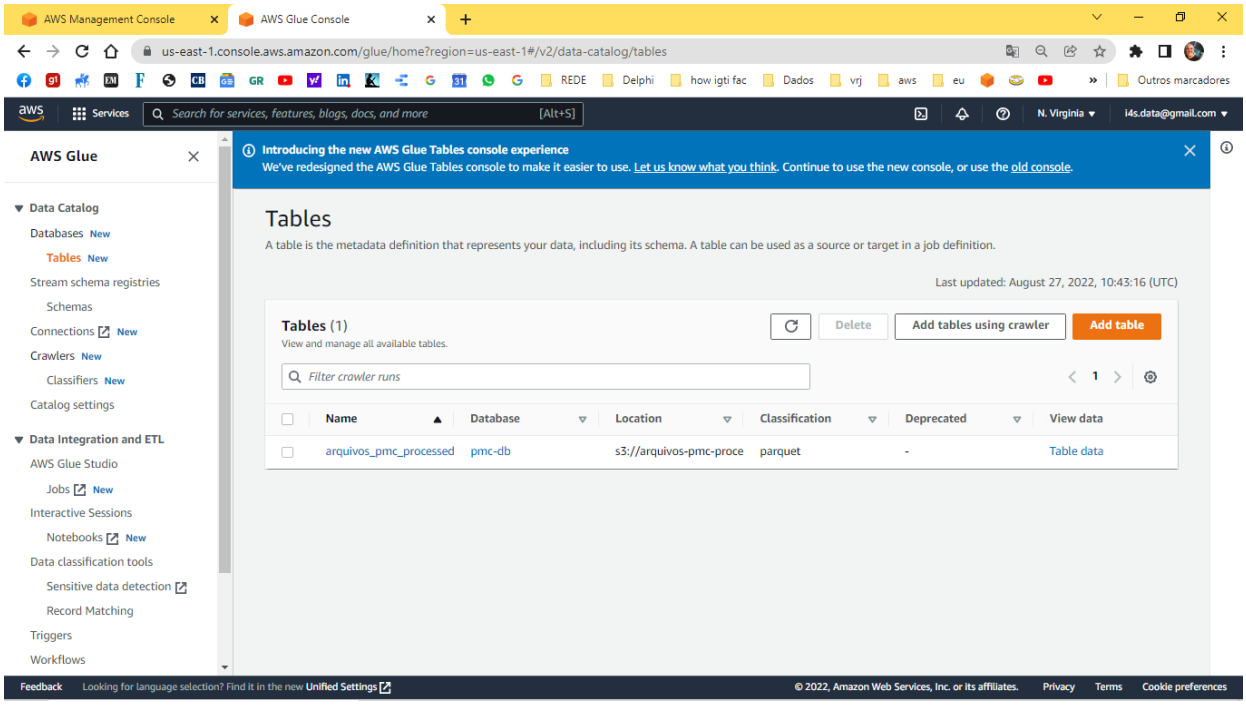


Fig-12

Aqui o schema gerado e a configuração da tabela arquivos_pmc_processess3_uf. Fig-13.

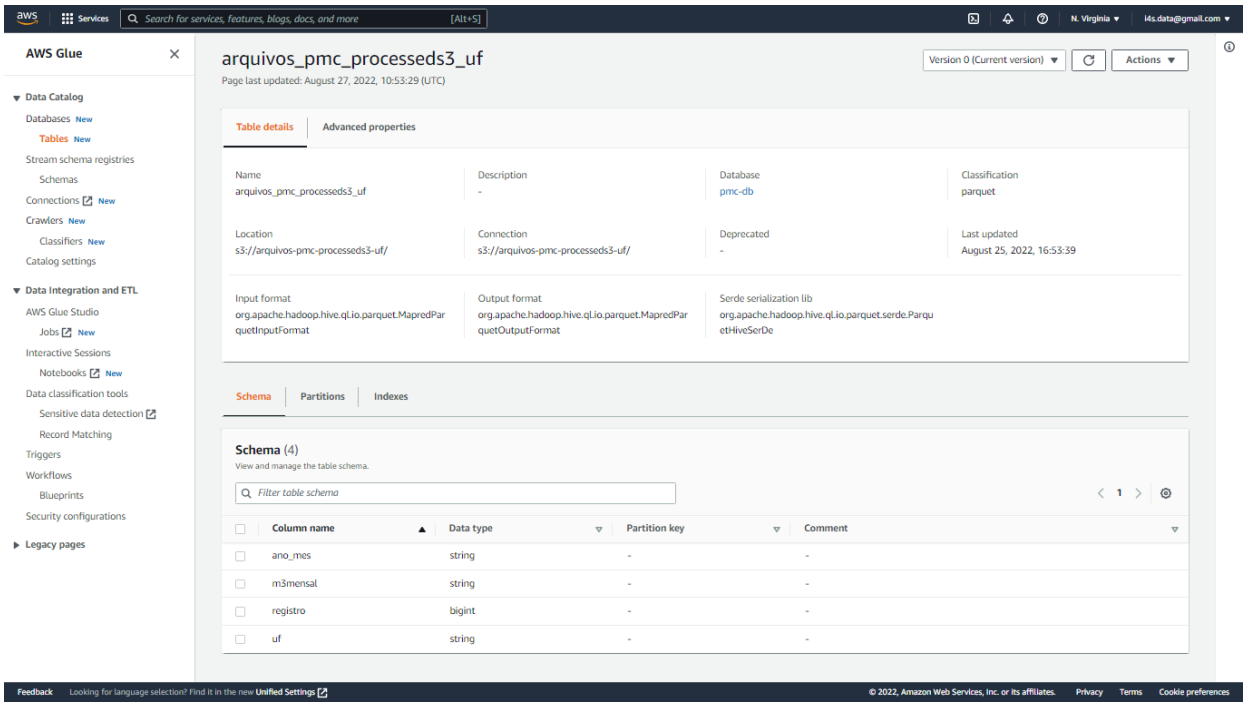


Fig-13

Agora, para acessar os dados que estão nos arquivos parquet catalogados no data catalog, define-se um workgroup no athena informando o bucket s3 aquivos-pmc-curateds3/athena onde serão salvos os arquivos de controle (metadados) do athena. Fig-14.

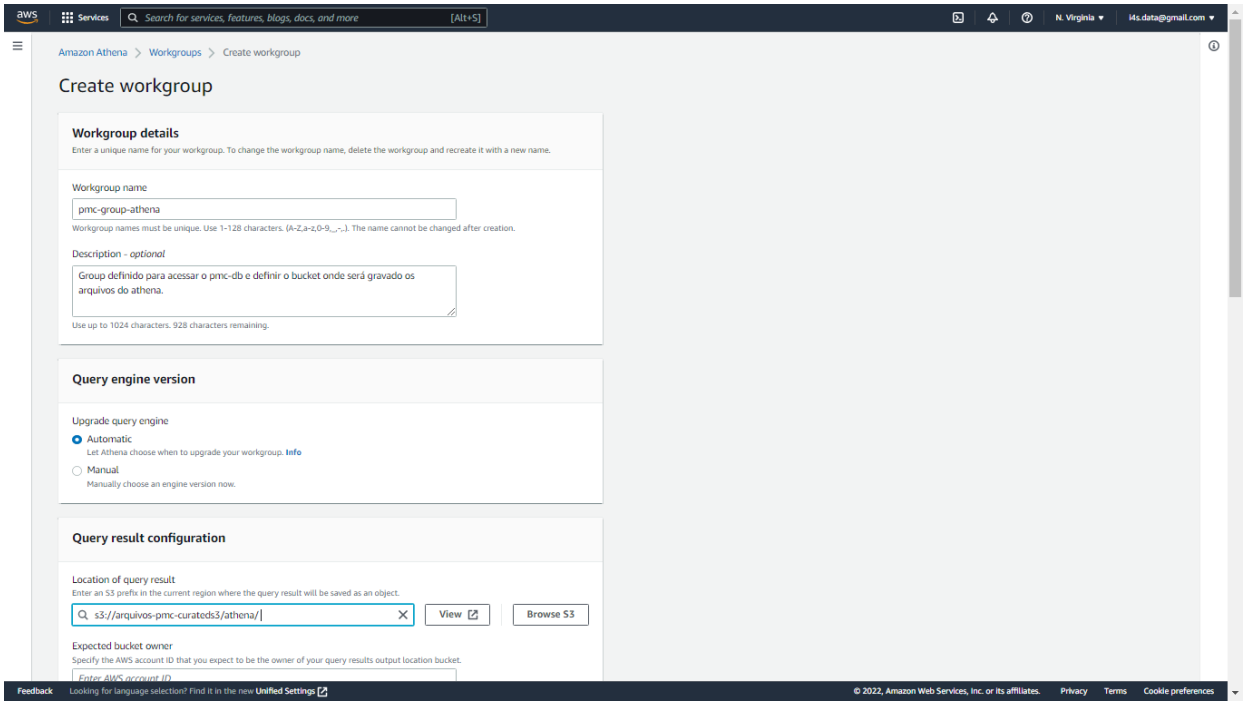


Fig-14

Configuração do group pmc-group-athena. Fig-15.

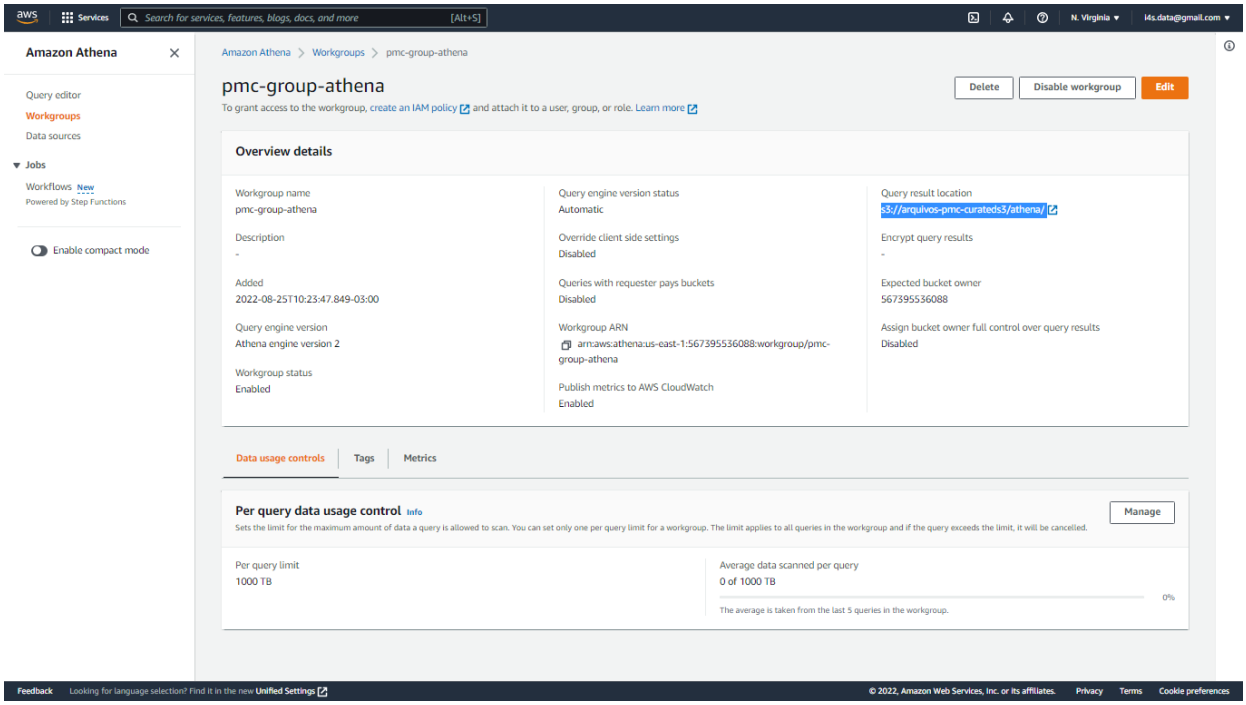


Fig-15

Agora, na opção de query do athena, pode-se acessar os dados que estão nos arquivos parquet no bucket s3 arquivos-pmc-processeds3-uf através do data catalog que, com os metadados, permite recuperar os percentuais de crescimento de vendas do varejo por UF, utilizando a linguagem SQL. Fig-16.

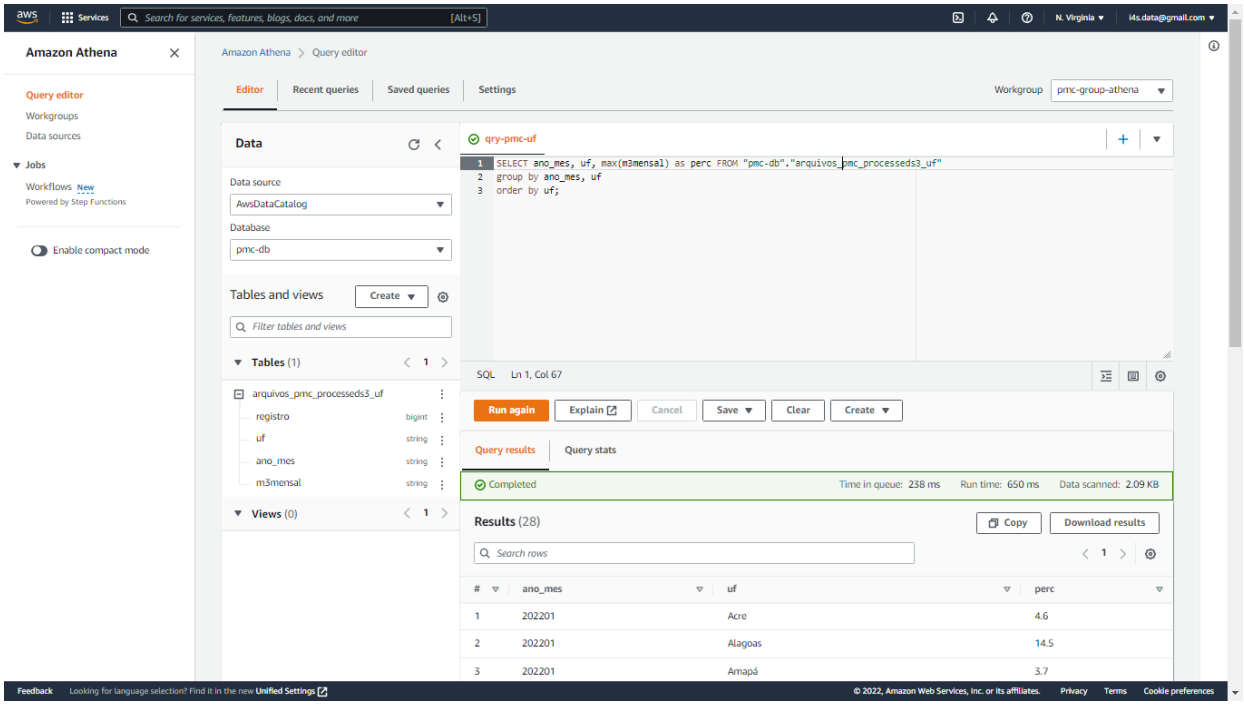


Fig-16

Em seguida, esses dados acessados pelo athena, através do data catalog, podem ser trabalhados e apresentados pelo quickinsight gerando apresentações gráficas e storytelling. Fig-17, fig-18 e fig-19.

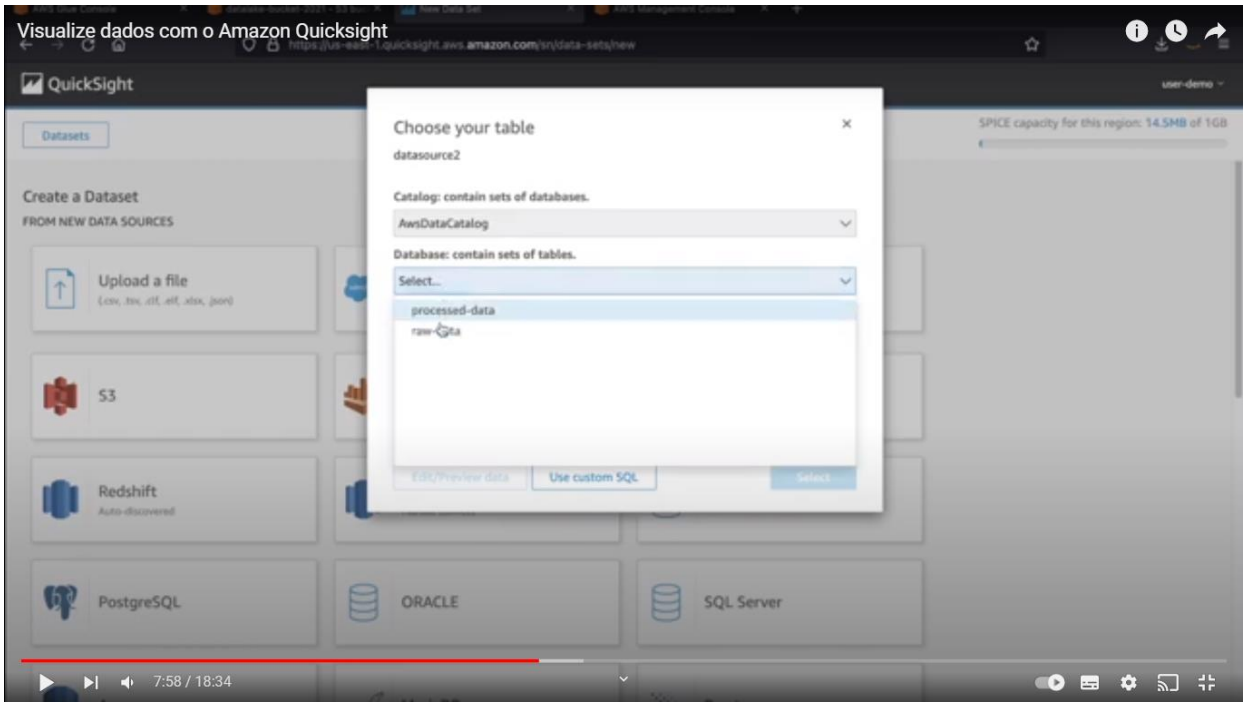


Fig-17

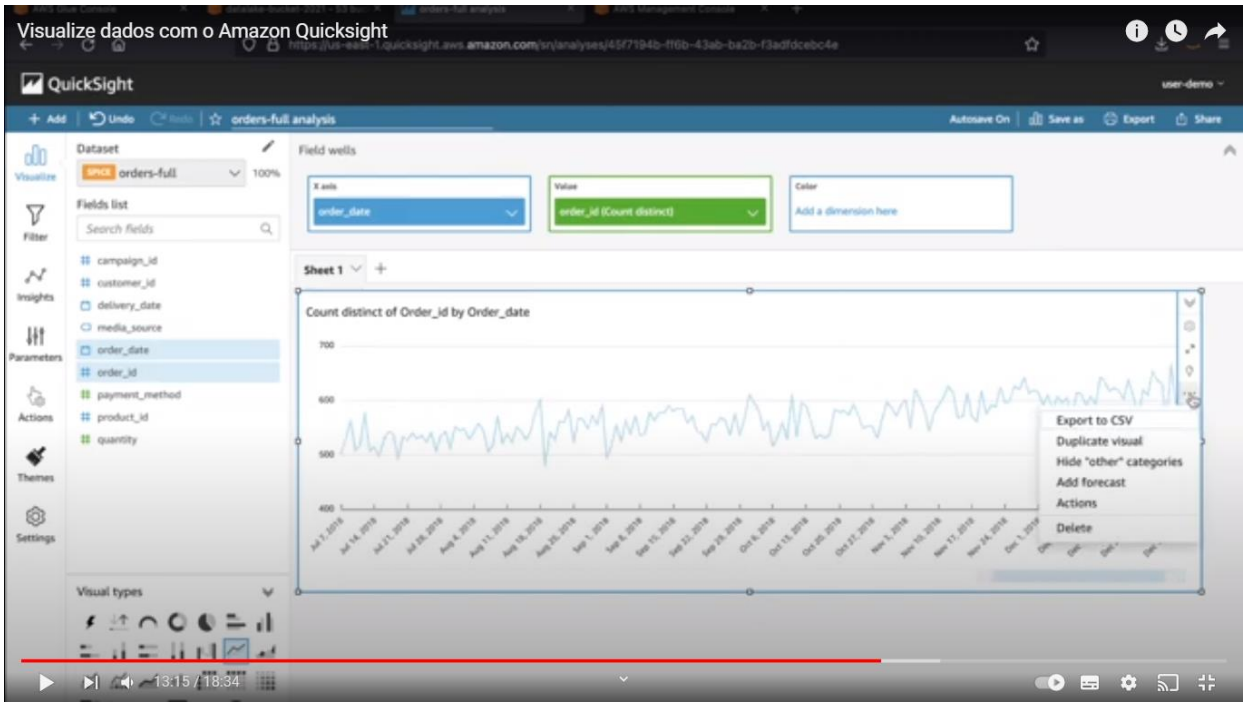

Visualize dados com o Amazon Quicksight
QuickSight
orders-full analysis
Dataset
orders-full
Fields list
Field wells
Sheet 1
Count distinct of Order_id by Order_date
Export to CSV
Duplicate visual
Hide "other" categories
Add forecast
Actions
Delete
Visual types
13:15 / 18:34

Fig-18

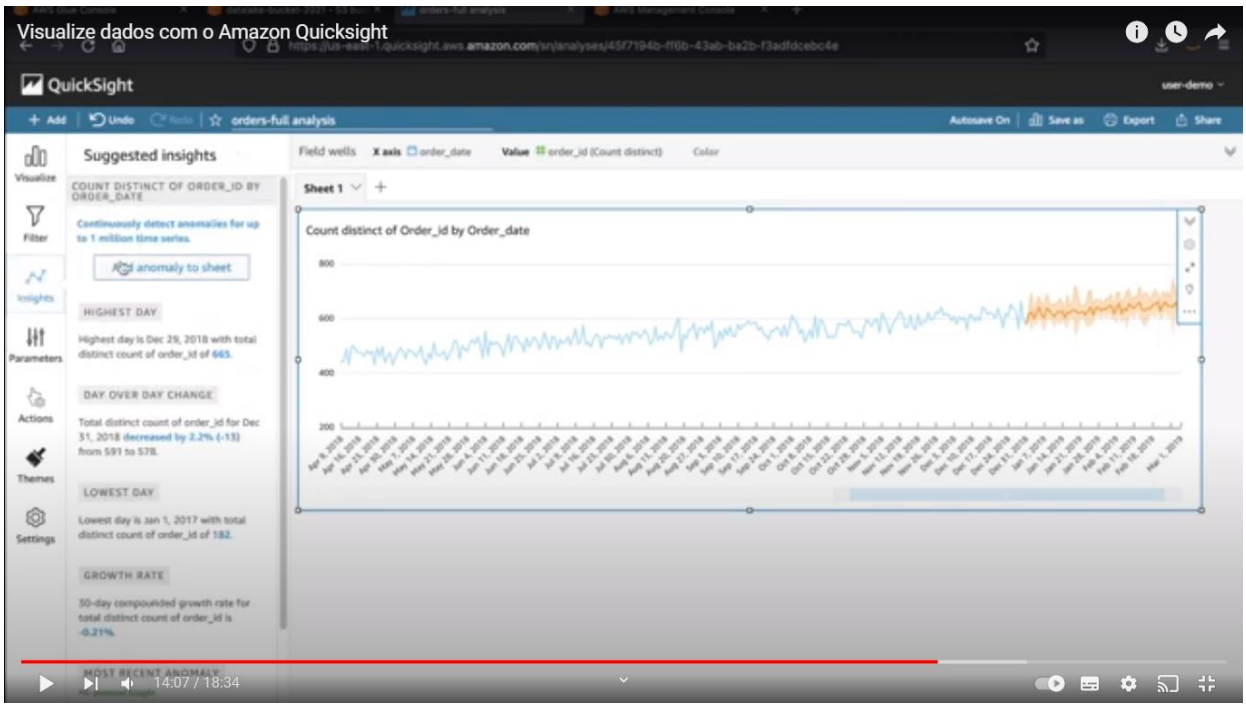

Visualize dados com o Amazon Quicksight
QuickSight
orders-full analysis
Suggested insights
Field wells
Sheet 1
Count distinct of Order_id by Order_date
14:07 / 18:34

Fig-19