FAZENDO ETL DO IPCA EM 2 CAMADAS, EXTRAINDO DADOS COM WEB SCRAPING NA 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 bigDGeneral_IPCA tem como objetivo baixar todo mês a tabela de IPCA do endereço https://www.idinheiro.com.br/tabelas/tabela-ipca/, extrair os dados mensais de IPCA, transformar em tabela estruturada e gravar em um arquivo parquet, fazer a ingestão em um bucket s3 na AWS para posterior catalogação desse arquivo, através do AWS Glue Data Catalog, sendo os dados posteriormente liberados para os Data Analysts e Data Scientysts que usarão ferramentas de insights para geração de apresentações gráficas para a empresa.

OBJETIVO:

Motivação:

Teve como motivador para que esse projeto bigDGeneral_IPCA fosse feito:
A necessidade de se ter o índice de inflação no Data Lake bigDVarejo e com isso conhecer a variação de preços praticados nas vendas ao consumidor.

Aplicação Prática:

Conhecer qual foi a inflação de determinado período relacionada aos preços de produtos voltados ao consumidor final, ao consumidor do varejo. Através desse conhecimento usar esses índices para se descontar o que foi perdido para a inflação, fazendo o alinhamento de qualquer valor no cálculo de sua variação.

Resultados Esperados:

Permitir a correção de qualquer valor ou índice descontando a variação da inflação do valor de produtos voltados ao consumidor.

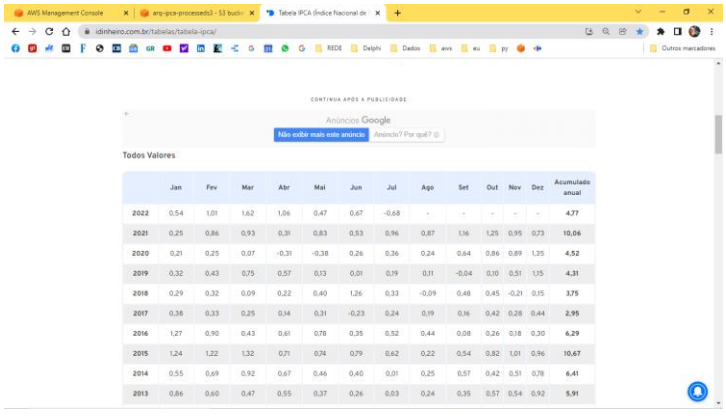
MÉTODOS:

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 bigDGeneral_IPCA:

- 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 tabelas de dados relativos à variação do IPCA:
  - Procurou-se encontrar na web uma tabela que resumisse todos os índices de IPCA existentes e atualizados do Brasil.

- Avaliar arquivos de dados disponibilizados:
  - Os dados da tabela table-all__value oferecidos pelo site https://www.idinheiro.com.br/tabelas/tabela-ipca/ foram analisados e validados se poderiam entregar o resultado de informação que era necessário para a comparação da perda de inflação ocorrida no período.



- *Verificar a viabilidade de baixar essa tabela via python:*
    - Outra verificação importante foi a de se certificar que a tabela eleita para o projeto pudesse ser baixada, utilizando funções python que fazem web scraping e que estivessem disponíveis continuamente, inclusive com as atualizações.

- *Definir quais os formatos de arquivos serão utilizados no armazenamento:*
    - A ideia é armazenar a tabela de IPCA mensal em arquivo parquet, que será catalogado 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. 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:*
    - Ficou definido que será necessário somente um bucket s3 para o Data Lake com o nome de arq-ipca-processeds3.

- *Definir quantas camadas serão necessárias para o processamento:*
    - O processamento será feito em 2 camadas, uma que usará o python fazendo a extração com web scraping da tabela de IPCA gravando em arquivo parquet e uma segunda camada que fará a catalogação dos dados através do aws Glue com suas ferramentas.

- *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 2 camadas ocorrerá no quinto dia do mês às 24:00.

PRODUTOS, SERVIÇOS E SISTEMAS:

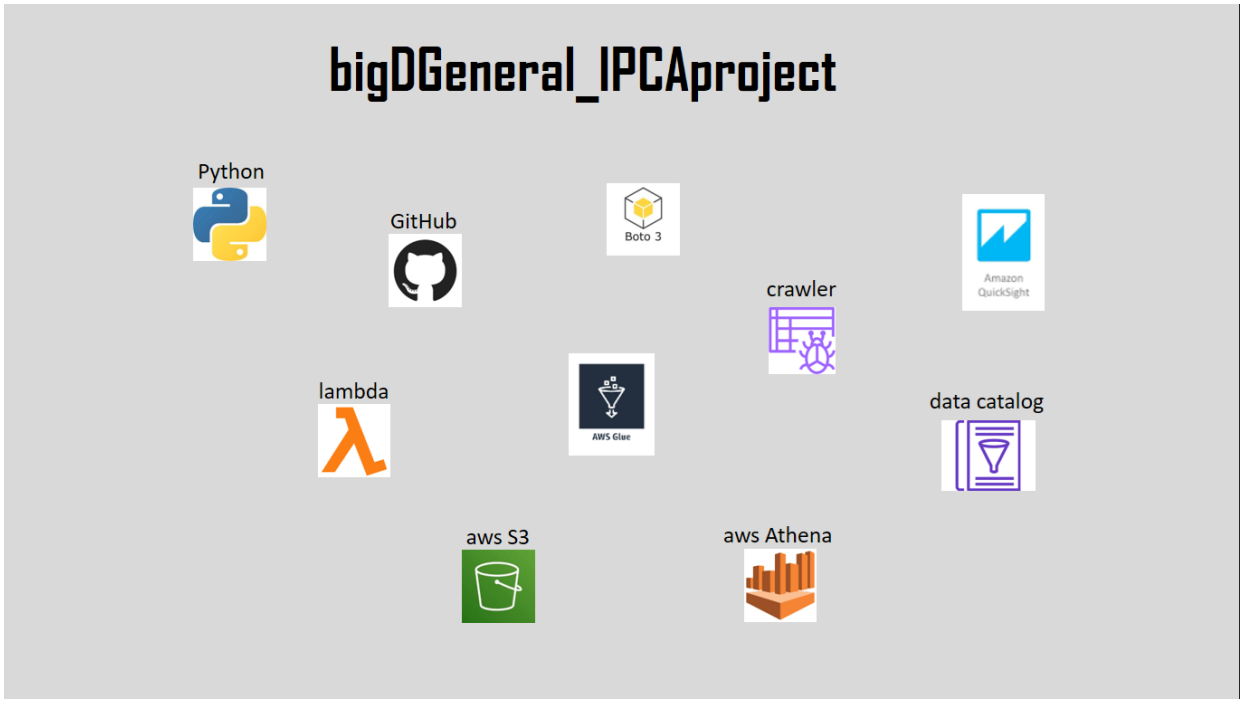


Fig-01

ESCOPO DA ARQUITETURA:

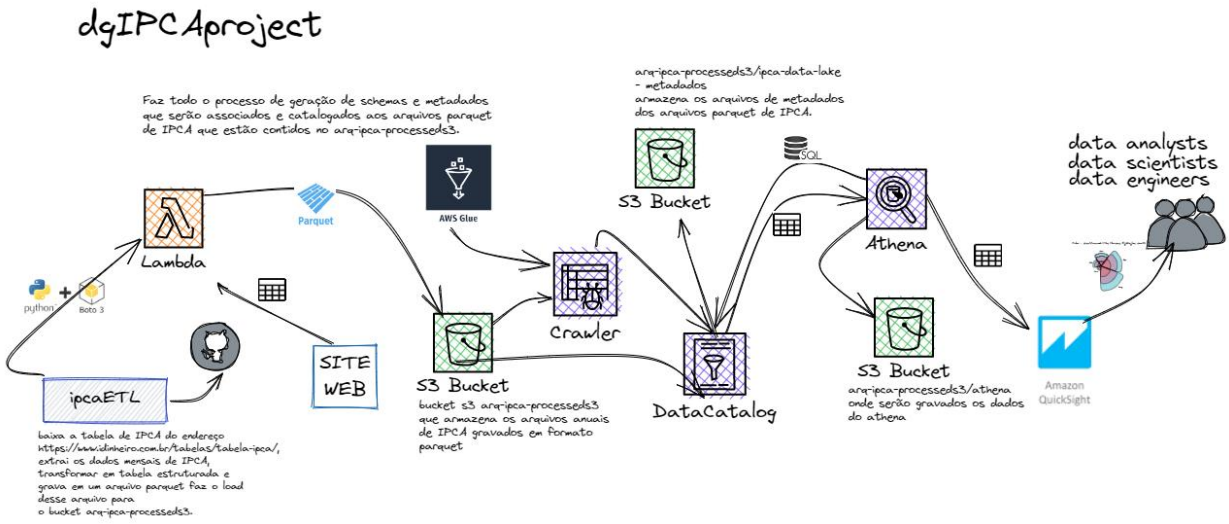


Fig-02

IMPLEMENTANDO AS CAMADAS COM CÓDIGO PYTHON:

Nesse projeto será utilizado 2 camadas em sua execução, uma camada usando código python que terá a função de fazer web scraping (extração de dados de uma página da web) da tabela de ipca, mensalmente, ofertada no site https://www.idinheiro.com.br/tabelas/tabela-ipca/ e uma segunda camada, utilizando os recursos da AWS, onde o arquivo ipca.pq (parquet) será catalogado pelo Glue Data Catalog, criando os metadados responsáveis por viabilizar o acesso aos dados que estão na tabela de ipca dentro do buckets s3 arq-ipca-processeds3.

Todo o código está disponível no endereço do github:
https://github.com/jairo2016/BigDGeneral_IPCAproject

IMPLEMENTANDO A CAMADA com web scraping:

```
    url = 'https://www.idinheiro.com.br/tabelas/tabela-ipca/'
    req = requests.get(url)
    if req.status_code == 200:
        content = req.content
        soup = BeautifulSoup(content, 'html.parser')
        tabela = soup.find("table",{"class":"table-all__value"})
        table_str = str(tabela)
        table_str = table_str.replace(',', '.')
        df = pd.read_html(table_str)[0]
```

Através da função requests.get, a pagina https://www.idinheiro.com.br/tabelas/tabela-ipca/ é carregada em memória para a variável req, daí então será possível extrair o conteúdo da página com BeautifulSoup e posteriormente, com soup.find, encontrar a tabela table-all__value que contém os valores do ipca mensalmente em vários anos. Essa tabela então é transformada em um df (data frame) para a transformação dos dados.

Os dados são transformados em uma tabela estruturada e gravados em arquivo parquet, para facilitar o acesso com sql:

```
def Monta_arq_ipca(tabela, PathArquivo):
    qtd_linhas = tabela.shape[0] - 1
    linha_cur= 0
    i=0

    registro= []
    ano= []
    mes= []
    perc= []

    while linha_cur <= qtd_linhas:

        coluna= 1
        nroMes= 1
        while coluna <= 13:
            valor= tabela.iloc[linha_cur, coluna]

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

            valor= str(valor)
```

```
            valor= valor.replace(' ', ',')

            try:
                valor= float(valor)
            except ValueError:
                print('valor inválido = ' + str(valor))
                break

            i= i+1
            registro.append(i)
            ano.append(str(tabela.iloc[linha_cur, 0]))
            mes.append(str(nroMes))
            perc.append(str(valor))

            nroMes= nroMes+1
            coluna= coluna+1

        linha_cur= linha_cur+1

    if i>0:
        df=pd.DataFrame({
                "registro":registro,
                "ano":ano,
                "mes":mes,
                "perc":perc,
                })

        print(df)

        df.to_parquet(PathArquivo + '.pq')
        df.to_string(PathArquivo + '.txt')
        return True

    else:
        return False
```

Em seguida o arquivo parquet ipca.pq, gerado na transformação, é carregado (load) para o bucket s3 arq-ipca-processeds3 onde será catalogado pelo Glue Data Catalog na próxima camada.

```
def UploadFile_file_ipca_processedS3(NomeBucketS3, nomeArquivo, pathArquivo):
    client = boto3.client(
        service_name='s3',
        aws_access_key_id='xxxxxxxxxxxxxxxxxxxx',
        aws_secret_access_key=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx,
        region_name='eu-west-1' # voce pode usar qualquer regiao
        )

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

IMPLEMENTANDO A CAMADA de catalogação e disponibilização dos dados:

A camada de catalogação 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 do arquivo parquet ipca.pq que está armazenado no bucket s3 arq-ipca-processeds3. Foram utilizadas as informações que estão na fig-03 abaixo tal como name, location e description.

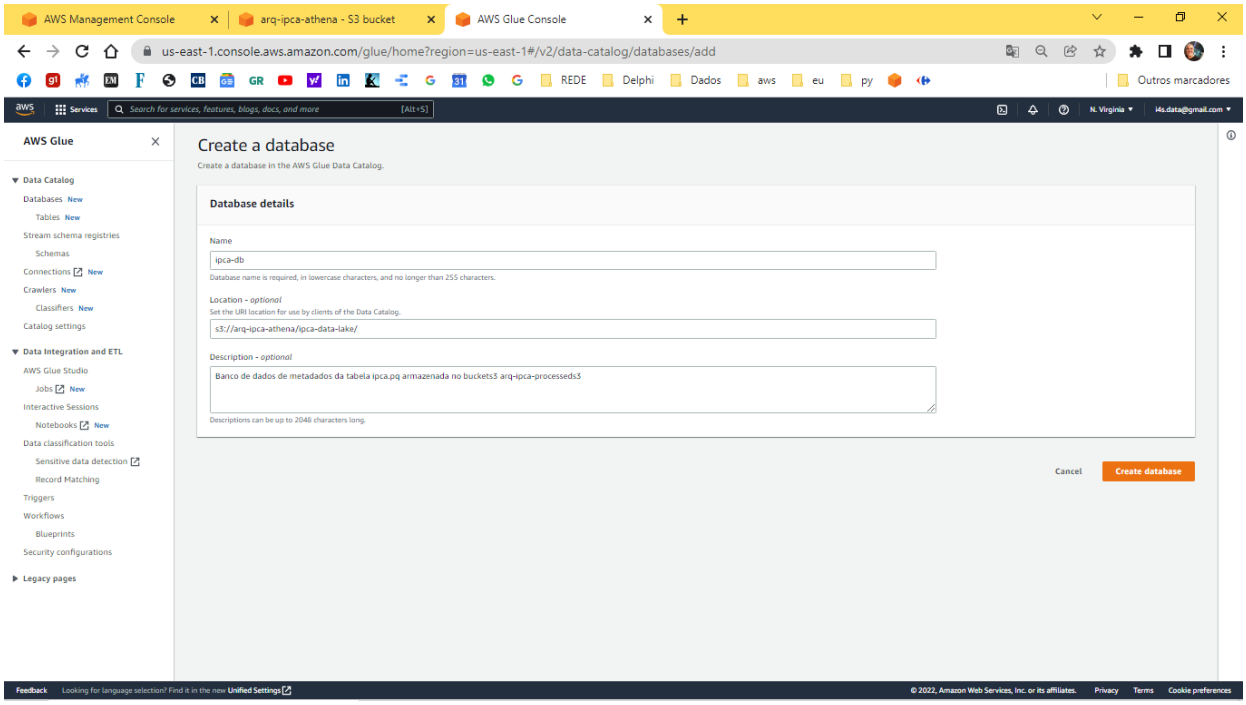


Fig-03

Em seguida cria-se um crawler que irá fornecer dados para a geração dos schemas das tabelas de metadados catalogados e apontados para a tabela parquet que está nos bucket arq-ipca-processeds3. Foram utilizadas as informações da fig-04.

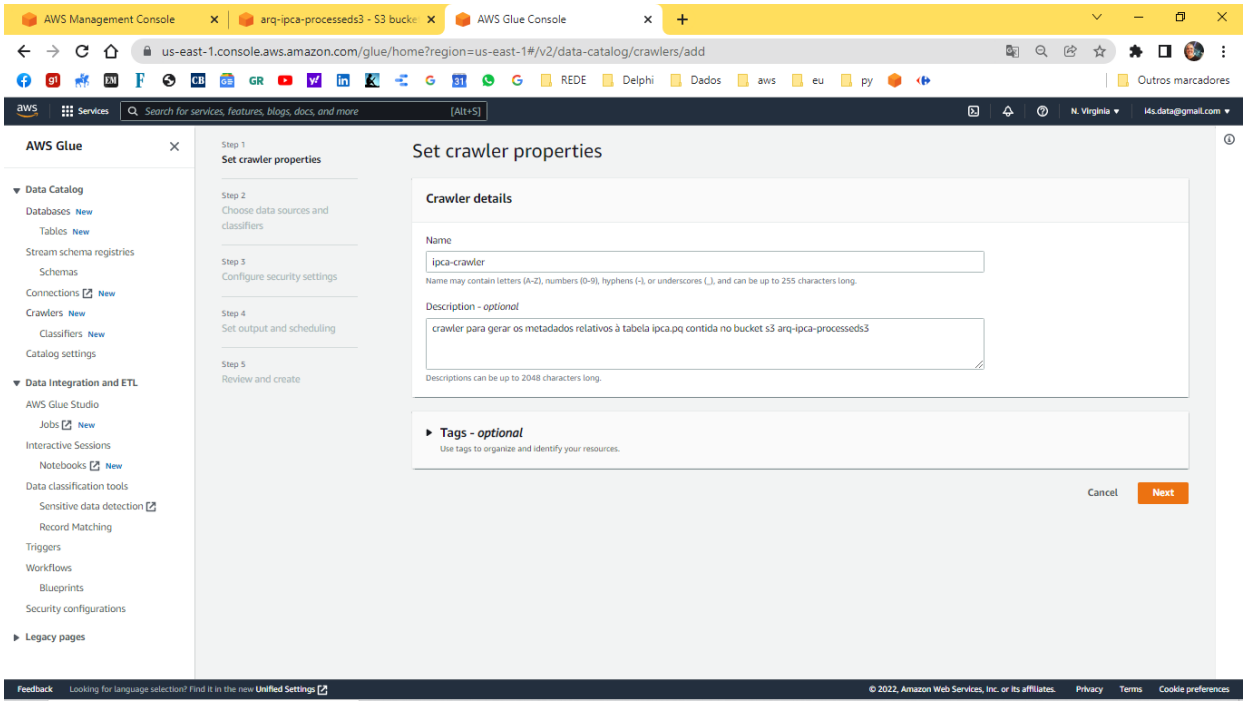


Fig-04

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 arq-ipca-processeds3 que contêm o arquivo ipca parquet transformado. Foram utilizadas as informações da fig-05.

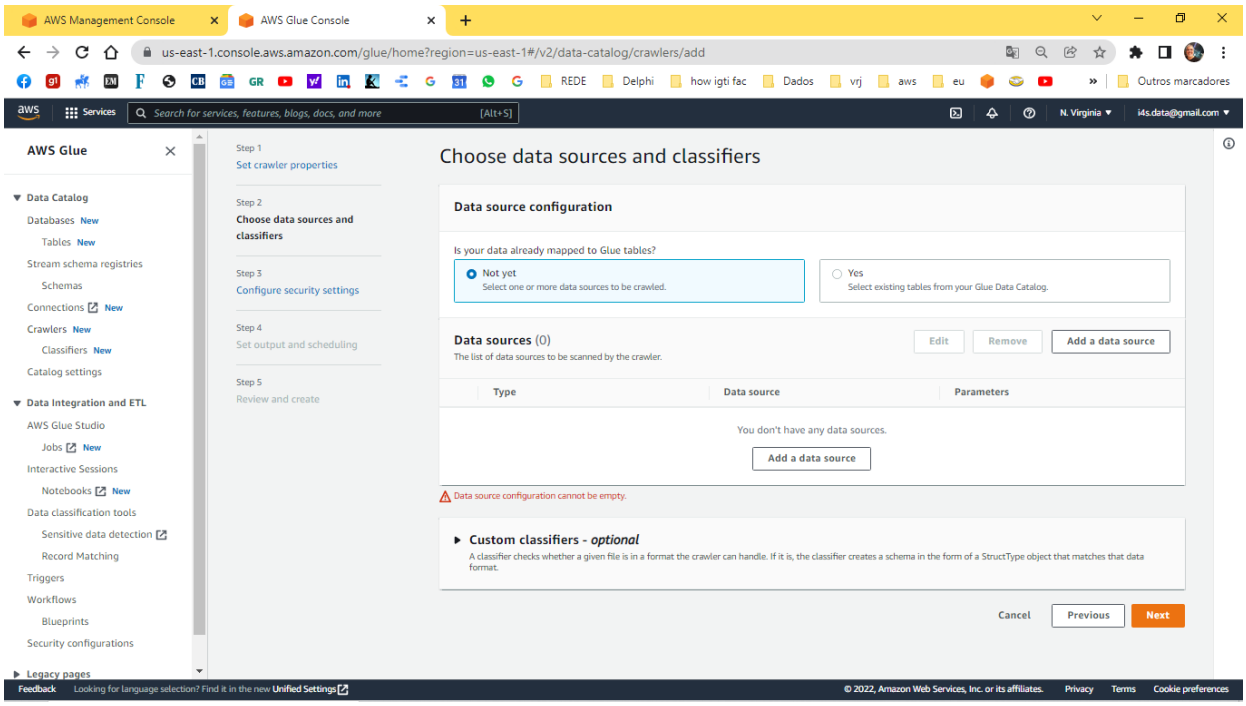


Fig-05

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

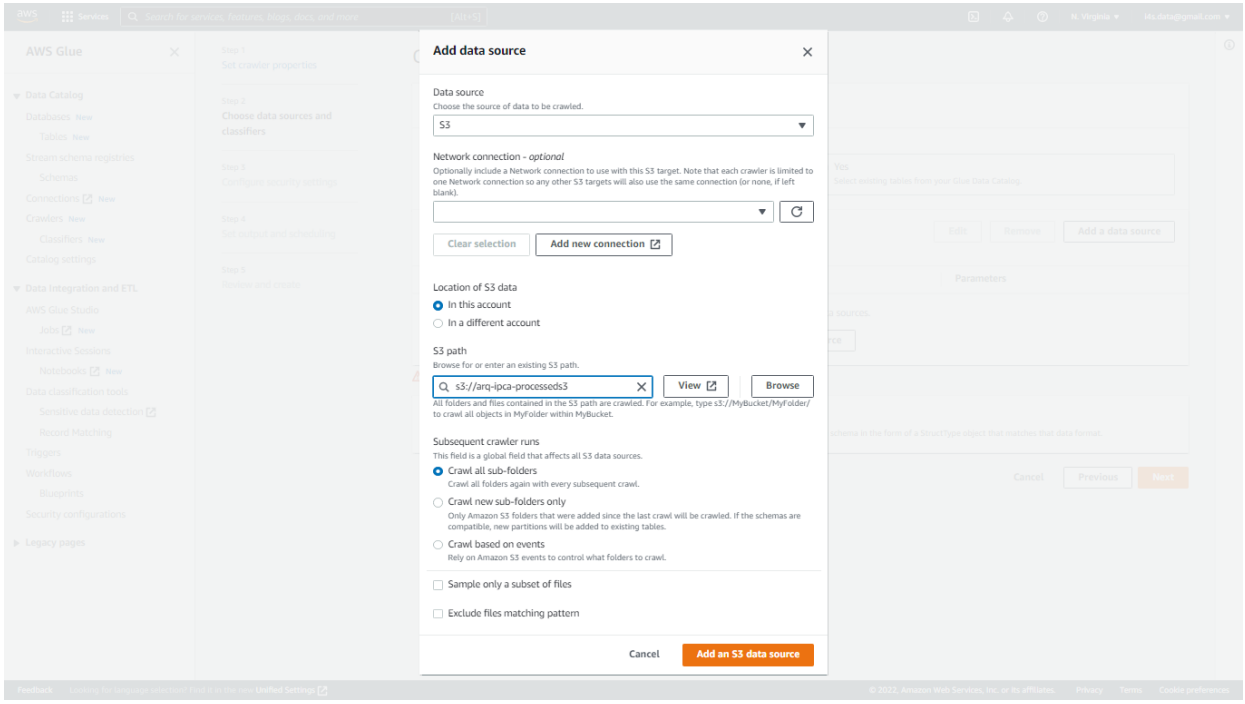


Fig-06

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

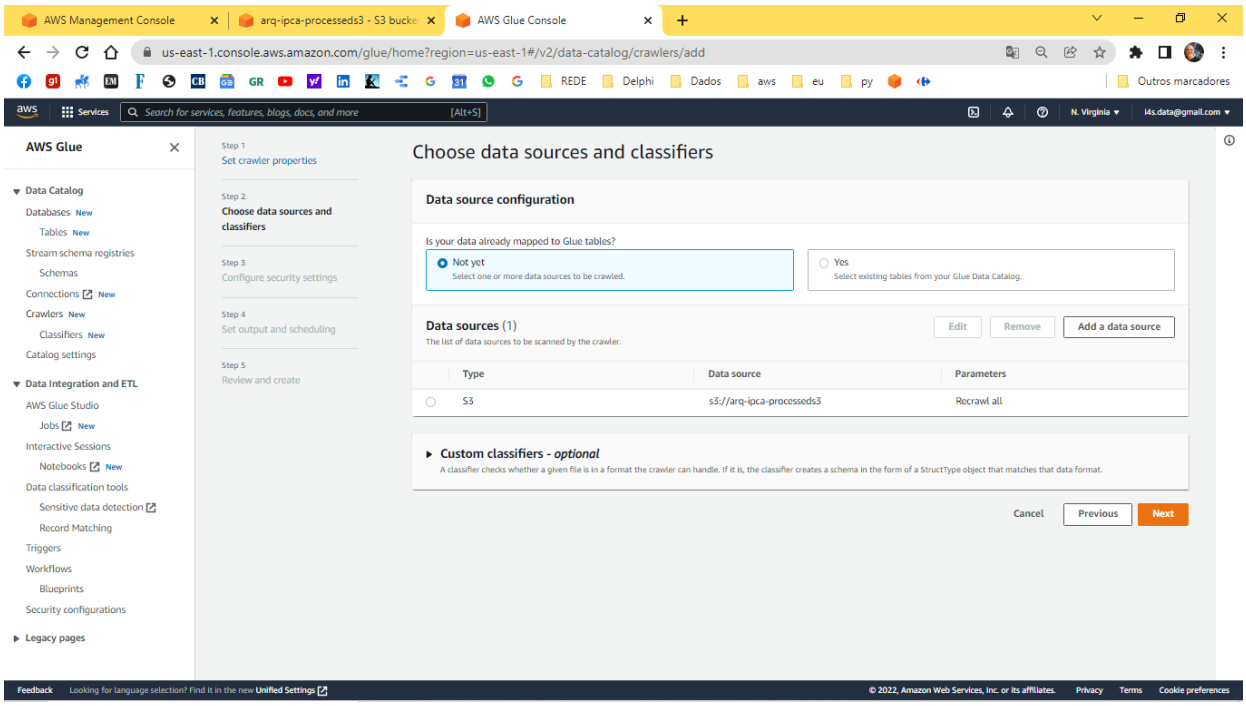


Fig-07

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.

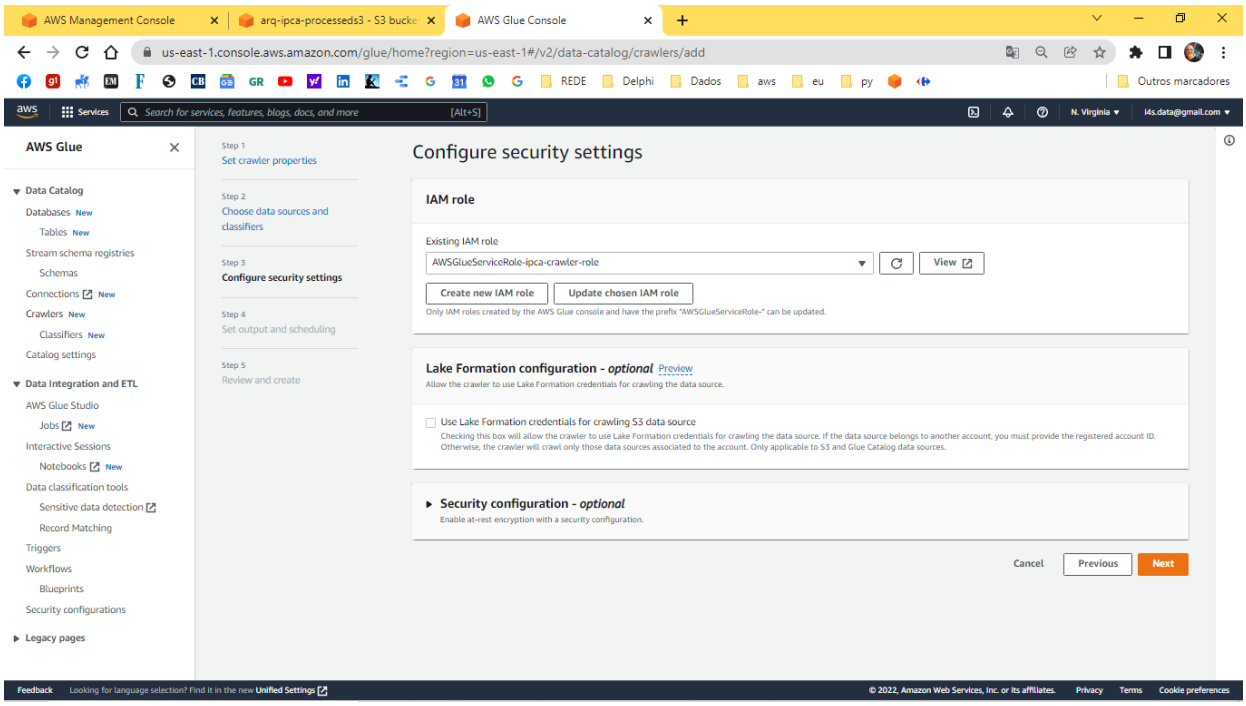


Fig-08

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

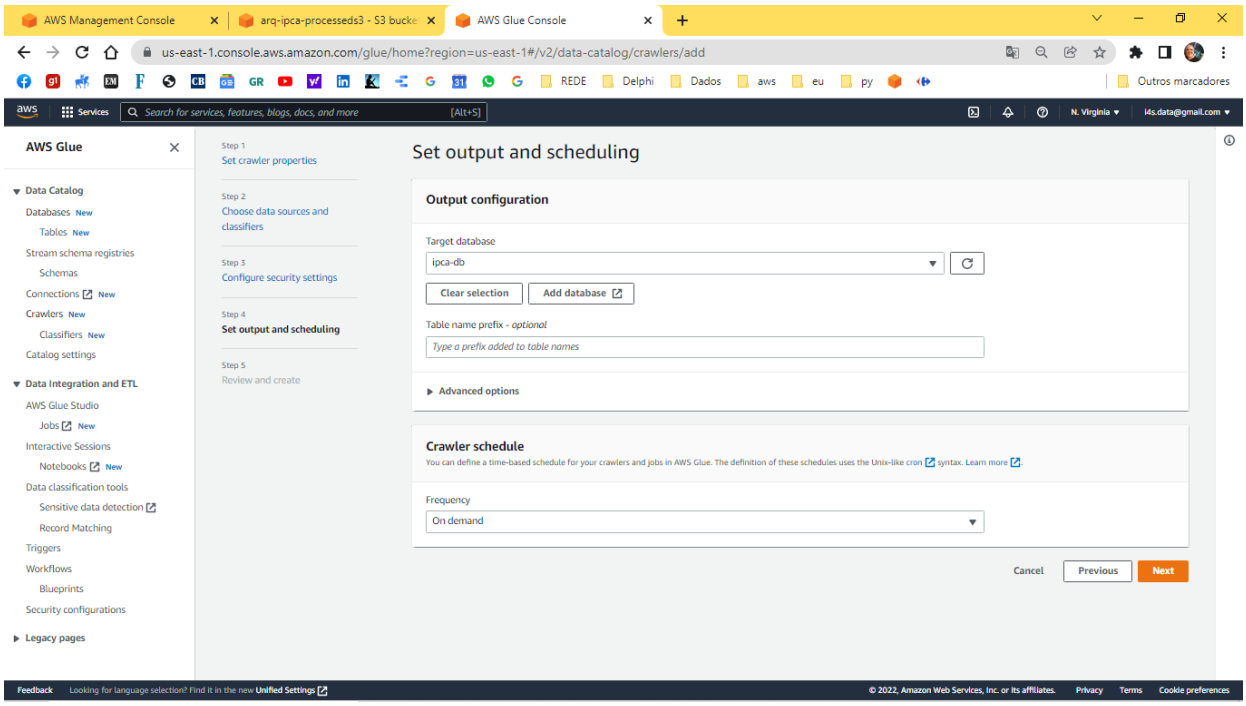


Fig-09

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

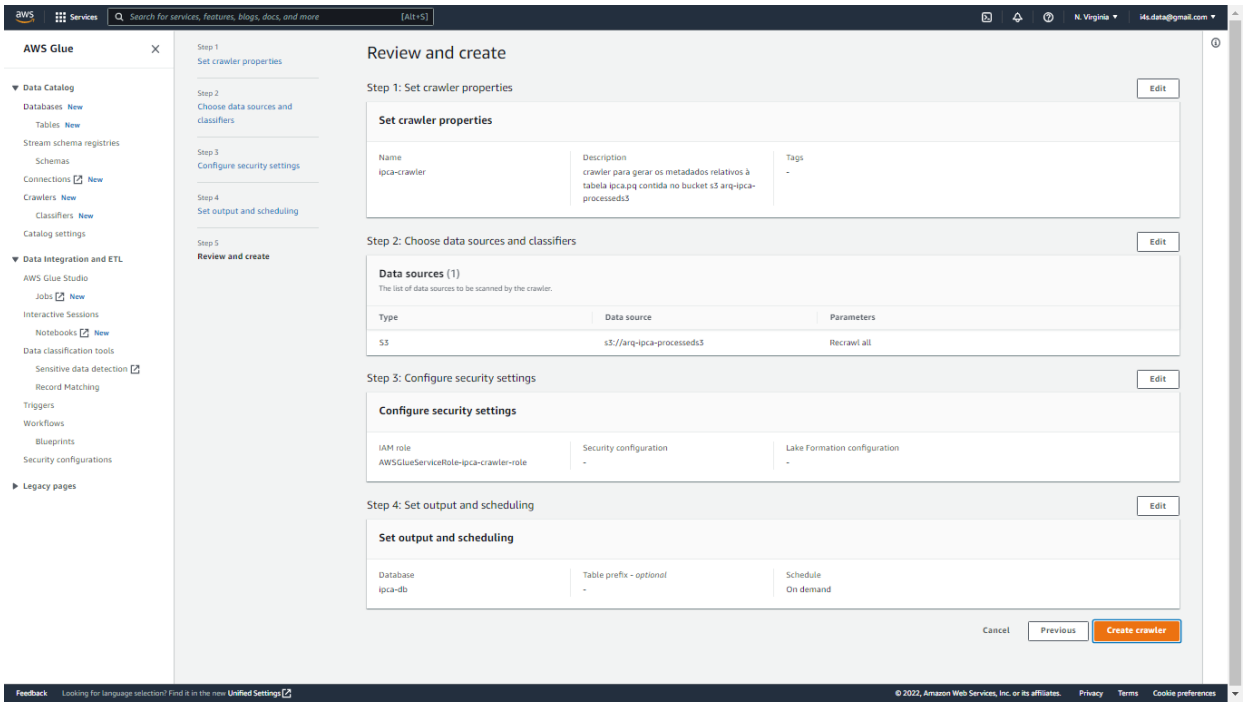


Fig-10

Agora com o crawler ipca-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.

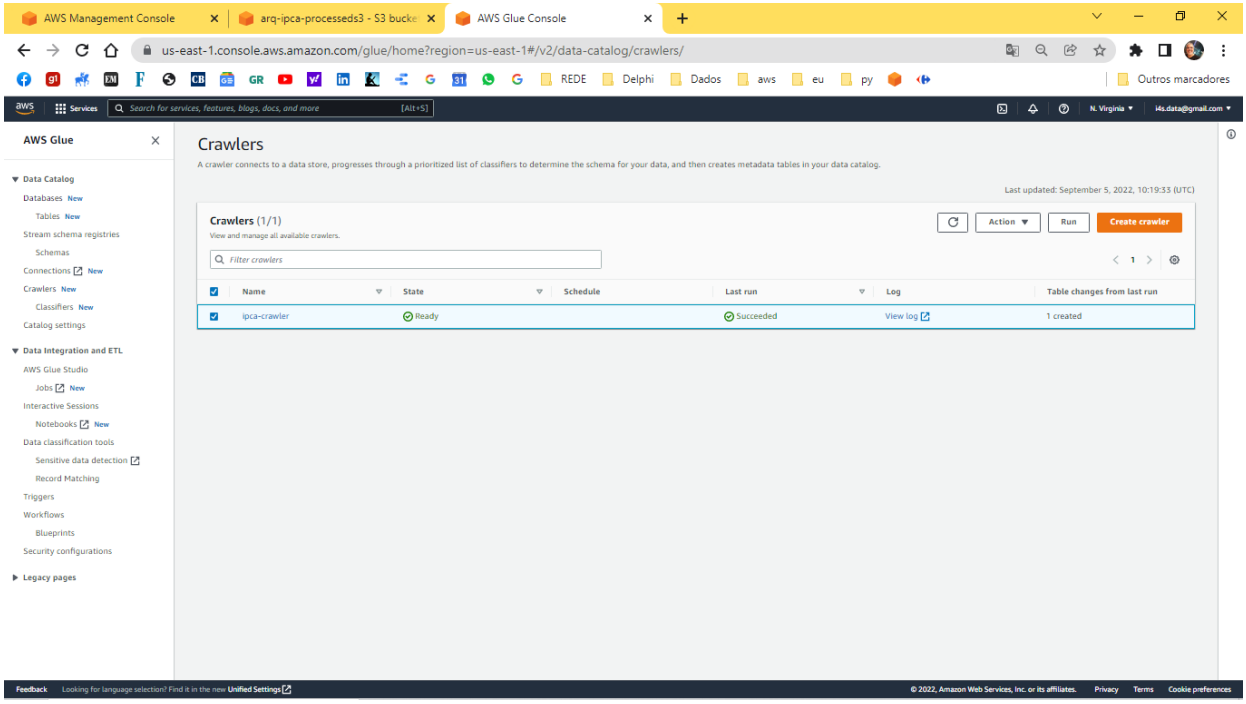


Fig-11

Após a execução do ipca-crawler verfica-se em tables que uma tabela arq_ipca_processeds3 foi criada no ipca-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 arq-ipca-processeds3 que contêm os dados de percentuais de ipca no arquivo ipca.pq. Veja na fig-12.

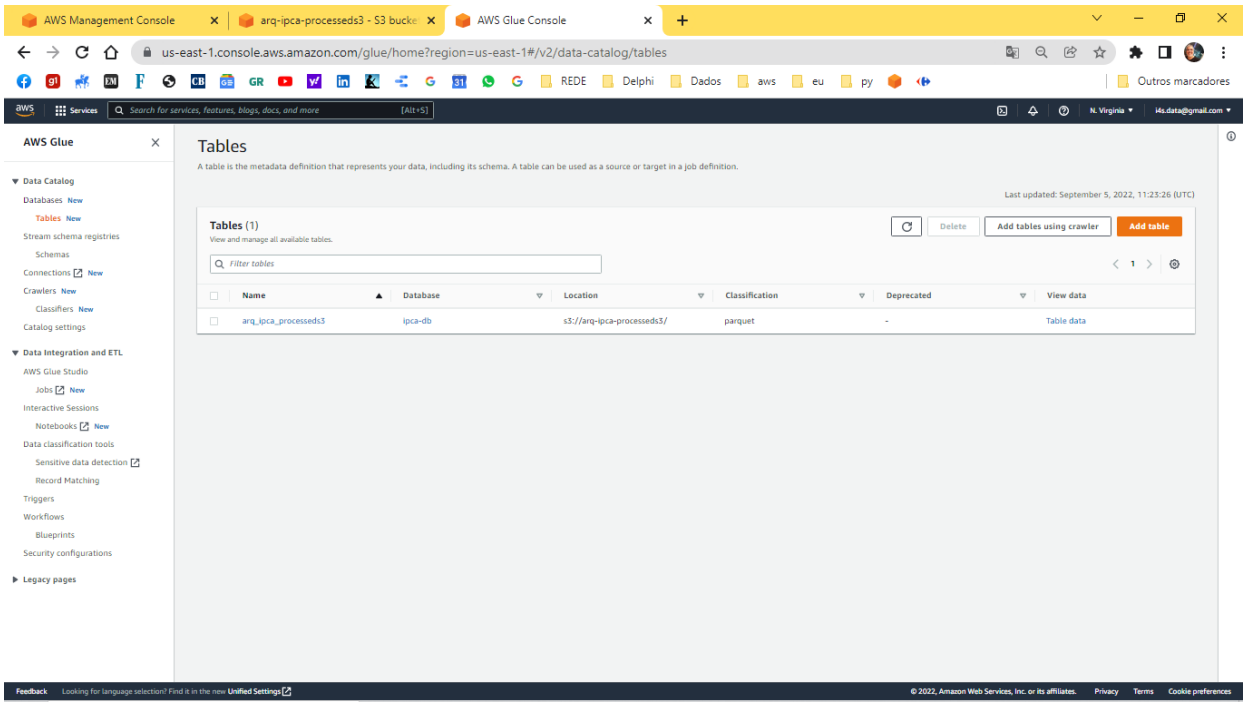


Fig-12

Aqui o schema gerado e a configuração da tabela processeds3. Fig-13.

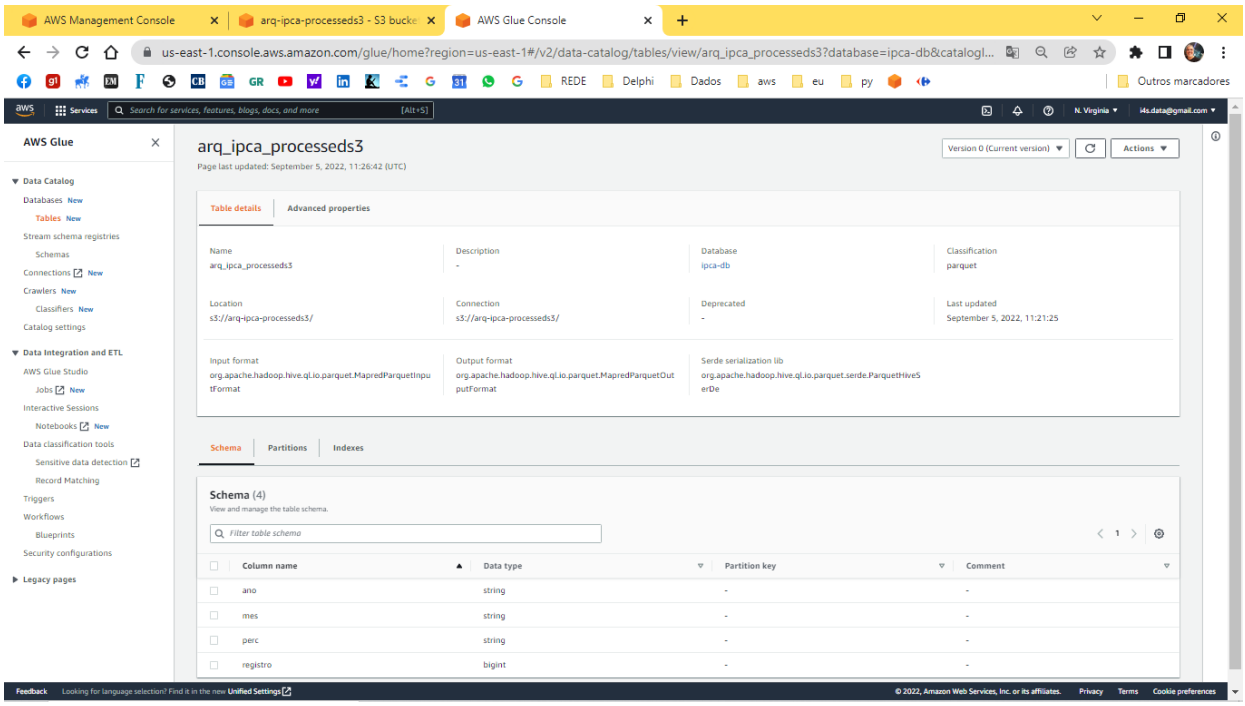


Fig-13

Agora, para acessar os dados que estão no arquivo parquet catalogados no data catalog, define-se um workgroup no athena informando o bucket s3 arq-ipca-processeds3/athena onde serão salvos os arquivos de controle (metadados) do athena. Fig-14.

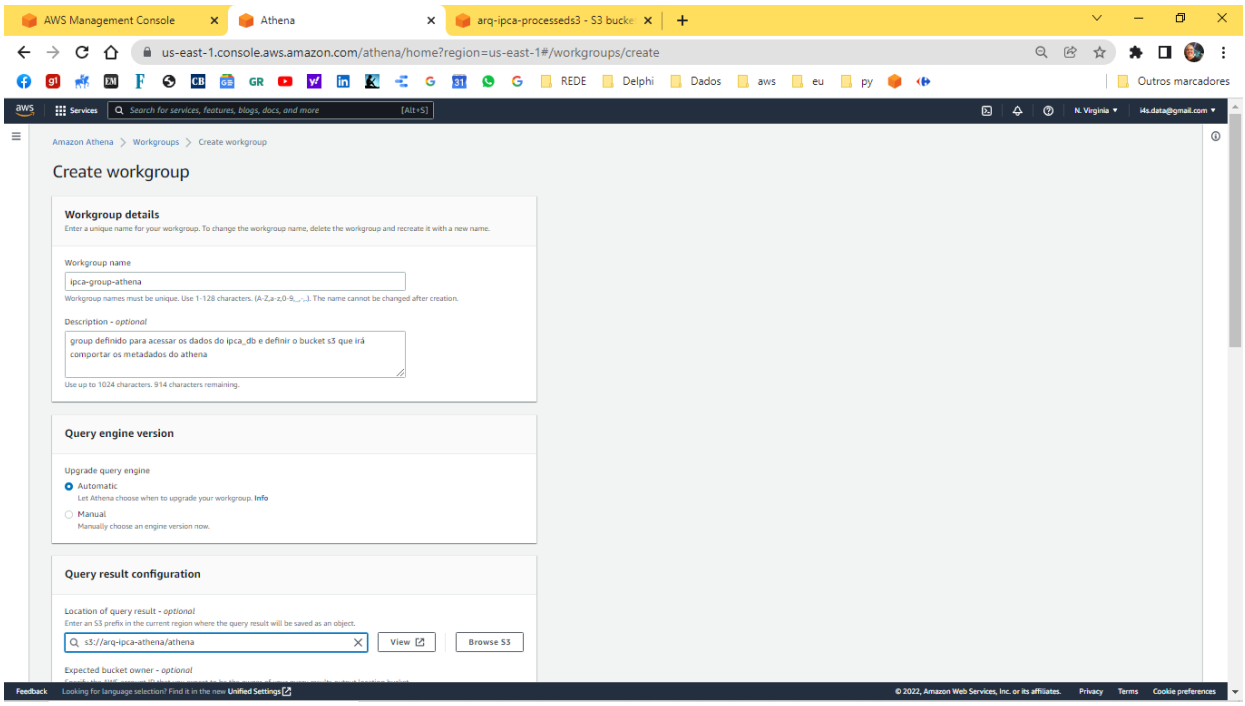


Fig-14

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

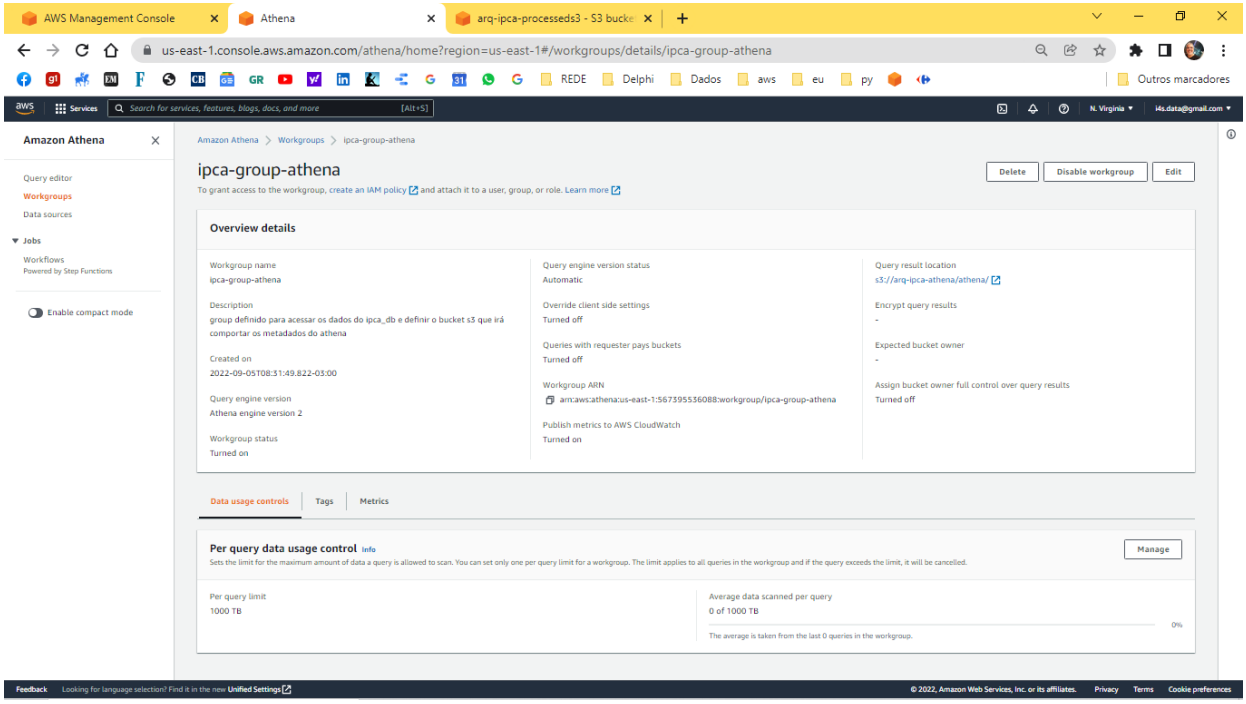


Fig-15

Agora, na opção de query do athena, pode-se acessar os dados que estão no arquivo parquet ipca.pq no bucket s3 arq-ipca-processeds3 através do data catalog que, com os metadados, permite recuperar os percentuais de ipca mensais, utilizando a linguagem SQL. Fig-16.

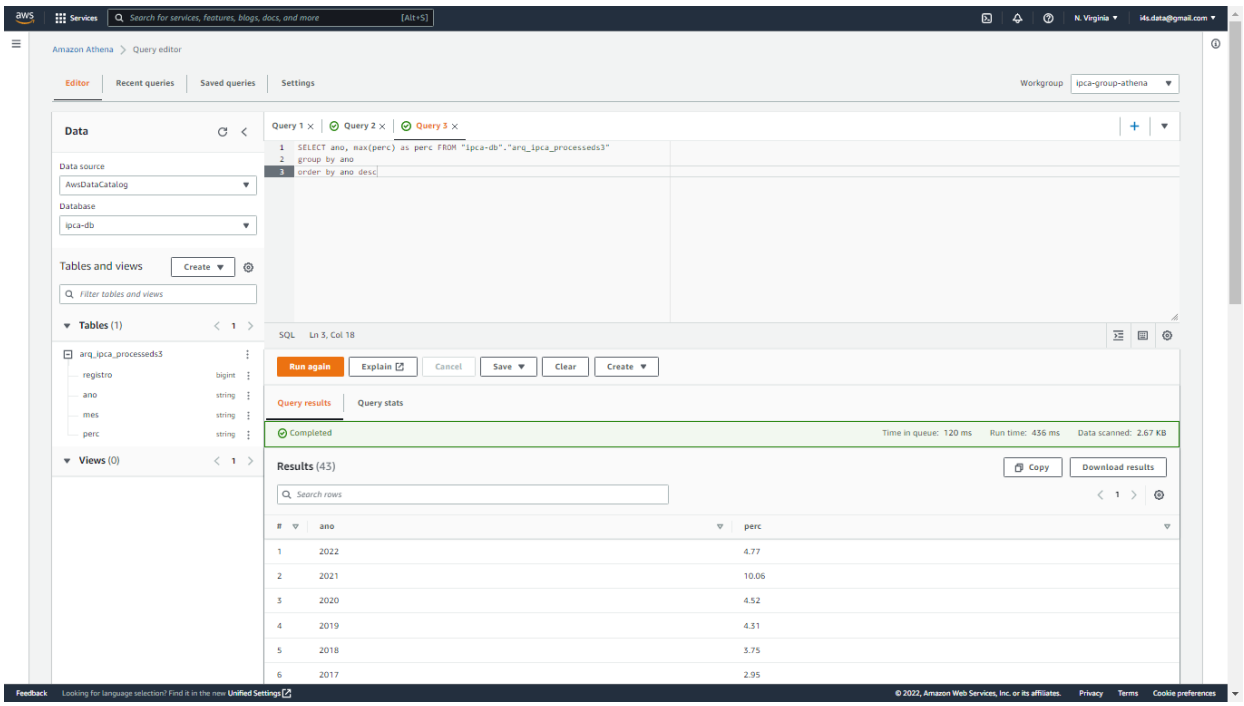


Fig-16

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.

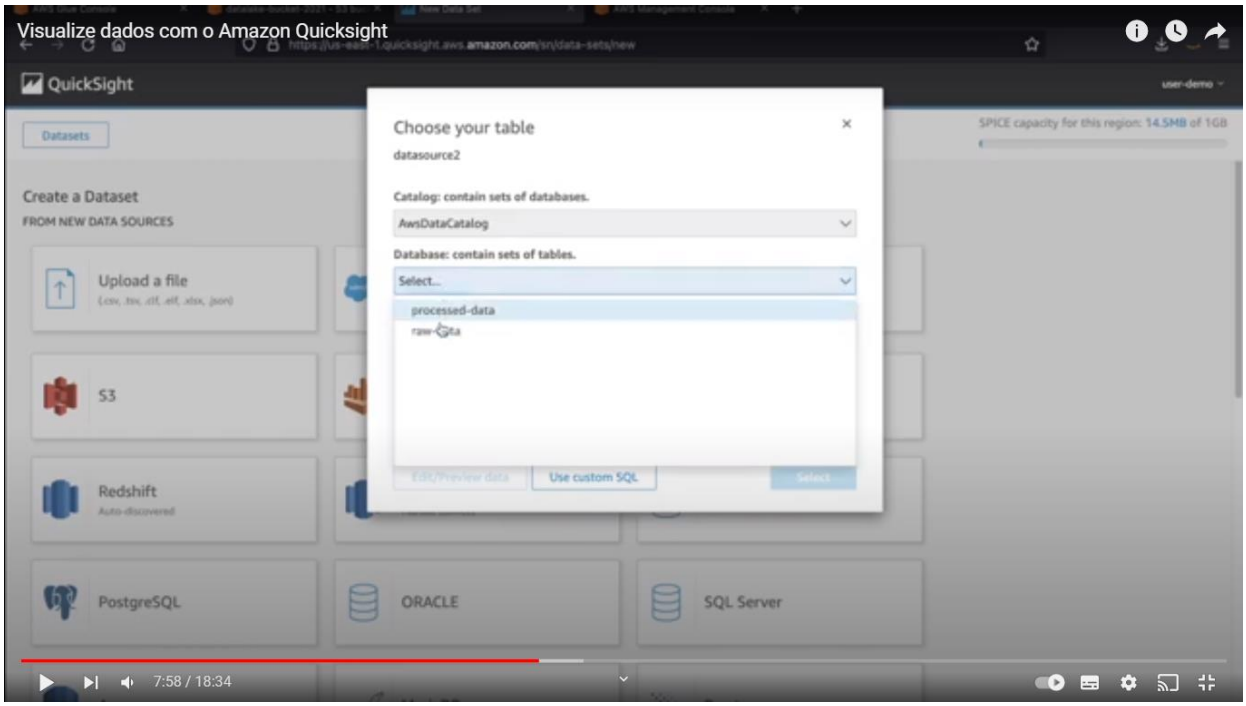


Fig-17

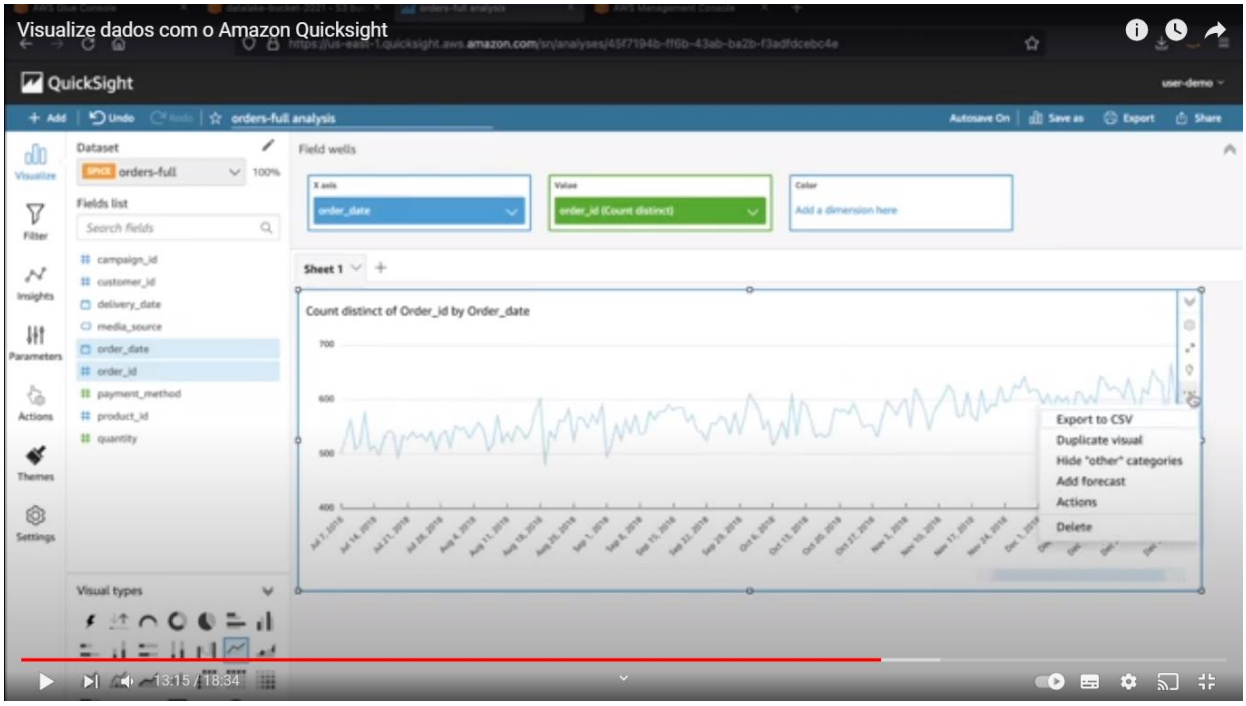


Fig-18

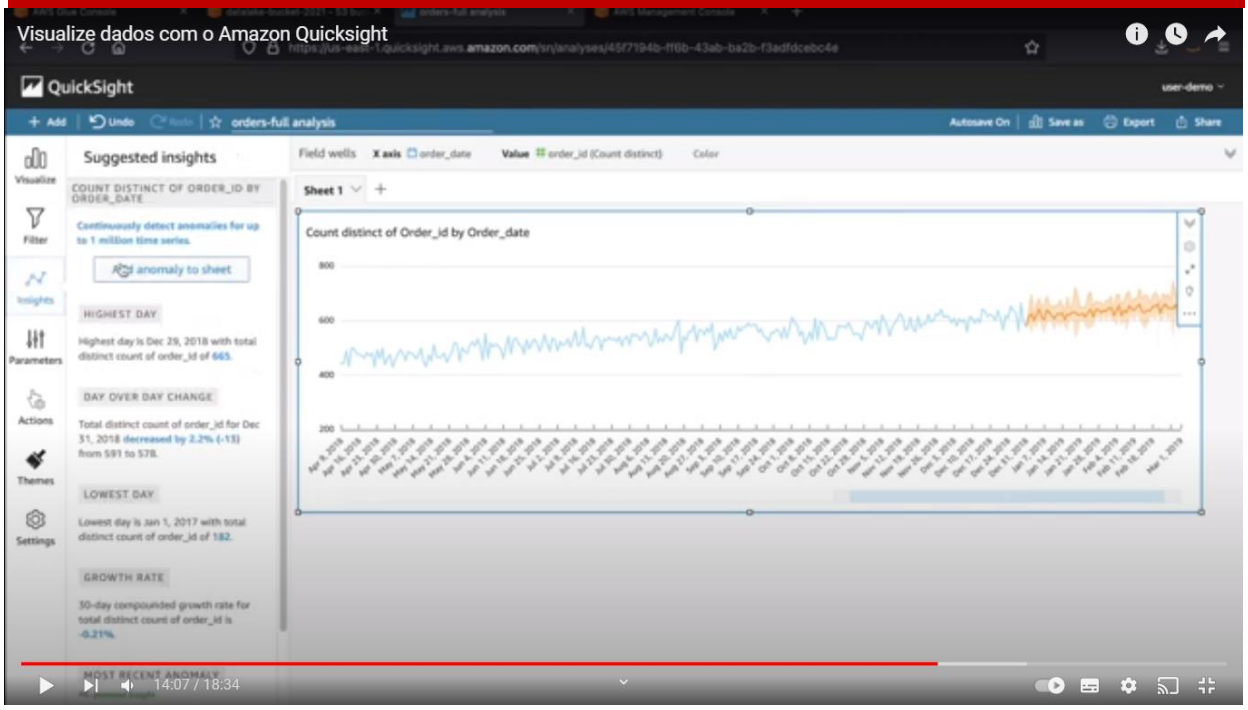

Visualize dados com o Amazon Quicksight
QuickSight
Suggested insights
Count distinct of Order_id by Order_date
14:07 / 18:34

Fig-19