



Sprint III – Engenharia de dados

Aluno : Arthur Ferreira de F. Lopes

Pós-graduação em ciência de dados e Analytics

Professores : Vitor Almeida e Silvo Alonso

Introdução

O número de restaurantes em Nova York está aumentando dia a dia. Muitos estudantes e profissionais ocupados confiam nesses restaurantes devido ao seu estilo de vida agitado. O serviço de entrega de comida online é uma ótima opção para eles. Fornece-lhes boa comida nos seus restaurantes preferidos. Uma empresa agregadora de alimentos, FoodHub, oferece acesso a vários restaurantes por meio de um único aplicativo de smartphone.

O aplicativo permite que os restaurantes recebam um pedido online direto de um cliente. O aplicativo designa um entregador da empresa para retirar o pedido após a confirmação do restaurante. O entregador então usa o mapa para chegar ao restaurante e aguarda o pacote de comida. Assim que a embalagem de comida é entregue ao entregador, ele confirma a retirada no aplicativo e se desloca até o local do cliente para entregar a comida. O entregador confirma a entrega no aplicativo após entregar a embalagem de comida ao cliente. O cliente pode avaliar o pedido no aplicativo. O agregador de alimentos ganha dinheiro arrecadando uma margem fixa do pedido de entrega dos restaurantes.

Coleta

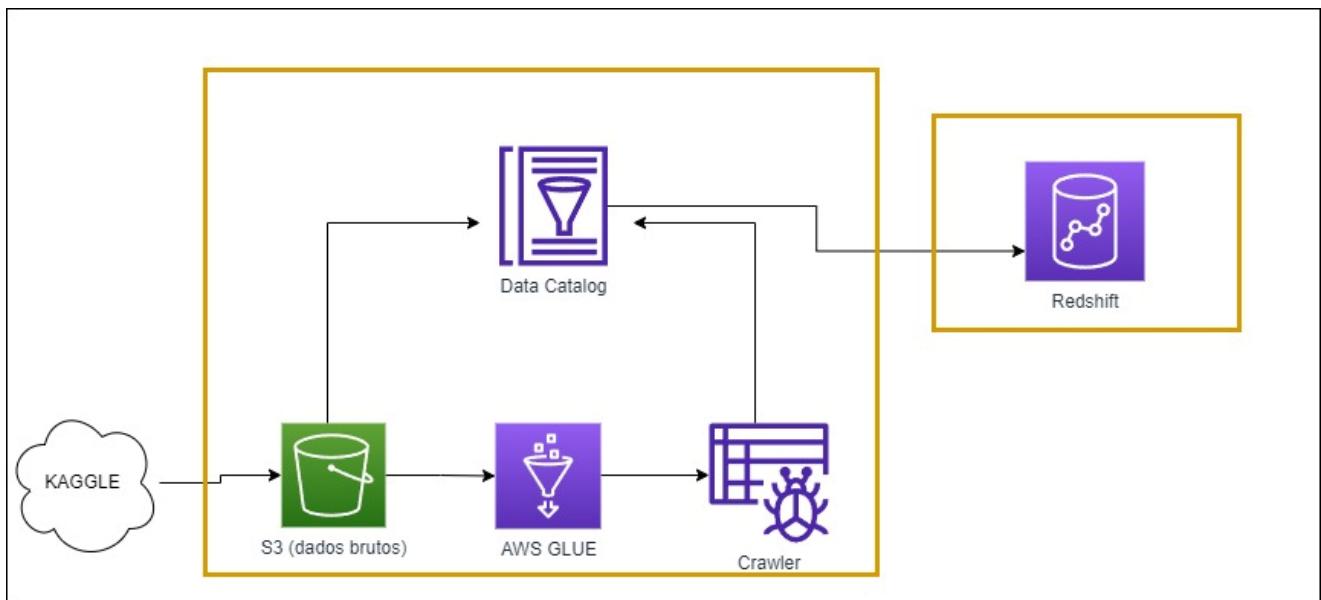
A coleta de dados foi realizada através do repositório de dados Kaggle. Por onde pude ter informações detalhadas do dataset que realizei o downloads.

<https://www.kaggle.com/datasets/ahsan81/food-ordering-and-delivery-app-dataset>

O dataset foi baixado em apenas um arquivo no formato CSV, onde é um formato que facilita o upload do mesmo na cloud AWS, a plataforma de escolhi para fazer o MVP

Modelagem

Um "pipeline de dados" refere-se a um conjunto de etapas sequenciais ou processos que são usados para coletar, processar, transformar e mover dados de uma fonte para um destino específico. Os pipelines de dados são comumente usados em engenharia de dados, análise de dados e em sistemas de processamento de dados em geral. Eles são projetados para automatizar e facilitar o fluxo de dados através de diferentes estágios. Abaixo, está representado o processo desde a coleta de dados até a utilização dos mesmos para responder alguma pergunta.



Catálogo de dados.

Um catálogo de dados é como uma lista organizada de informações sobre todos os dados que temos. É como um catálogo de uma biblioteca, mas em vez de livros, ele lista todos os nossos dados. Com um catálogo de dados, podemos encontrar rapidamente informações sobre onde os dados estão, como estão estruturados e quem os usa. Isso nos ajuda a gerenciar nossos dados de forma eficaz, economizando tempo e tornando mais fácil usar essas informações importantes para tomar decisões informadas. Em resumo, um catálogo de dados é como um guia útil para nossos dados, tornando mais simples encontrá-los e usá-los. Abaixo irei descrever os dados do dataset escolhido.

O dataset contém os diferentes dados relacionados a um pedido de comida em um aplicativo.

As descrições das colunas são :

order_id: ID exclusivo do pedido - Tipo : bigint

customer_id: ID do cliente que pediu a comida – Tipo: bigint

nome_restaurante: Nome do restaurante – Tipo : Varchar

cuisine_type: Culinária pedida pelo cliente – Tipo : Varchar

day_of_the_week: Indica se o pedido é feito em dia de semana ou final de semana (o dia da semana é de segunda a sexta e o final de semana é sábado e domingo) obs: Existe pedidos que não foram classificados pelo cliente – Varchar

rating: Classificação dada pelo cliente em 5 (Existe pedidos que o cliente não informou a classificação) – Tipo: Varchar – Valor minimo 3 – Valor Maximo 5

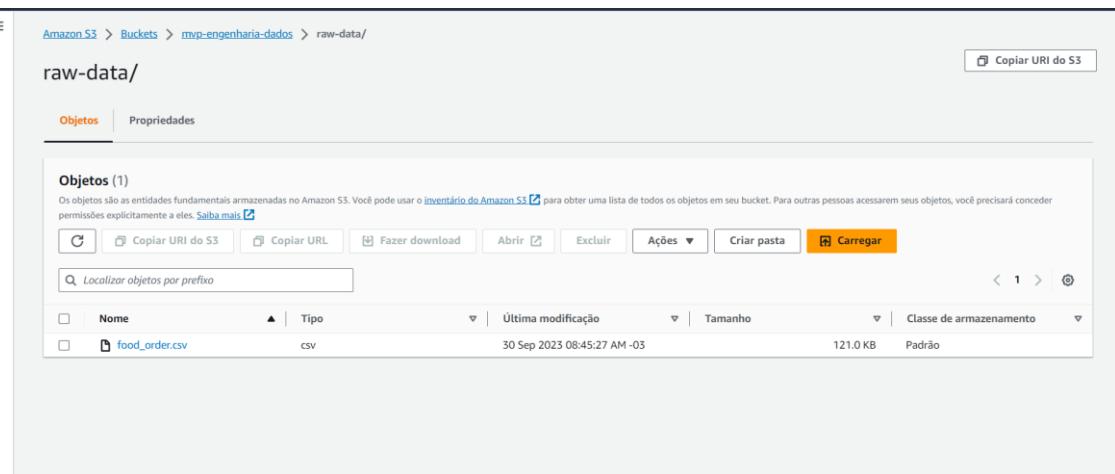
food_preparation_time: Tempo (em minutos) que o restaurante leva para preparar a comida. Isto é calculado tomando a diferença entre os carimbos de data/hora da confirmação do pedido do restaurante e a confirmação de retirada do entregador. – Tipo: bigint - Valor minino 20 – Valor máximo – 35

delivery_time: Tempo (em minutos) que o entregador leva para entregar a embalagem de comida. Isso é calculado tomando a diferença entre os carimbos de data/hora da confirmação de coleta do entregador e as informações de entrega – Tipo : bigint – Valor mínimo 15 – Valor máximo 33

Carga dos dados

Como visto acima, os dados foram obtidos através de um repositório na internet e que somente depois do downloads pude iniciar a carga dos dados , abaixo, irei explicar cada passo do pipeline :

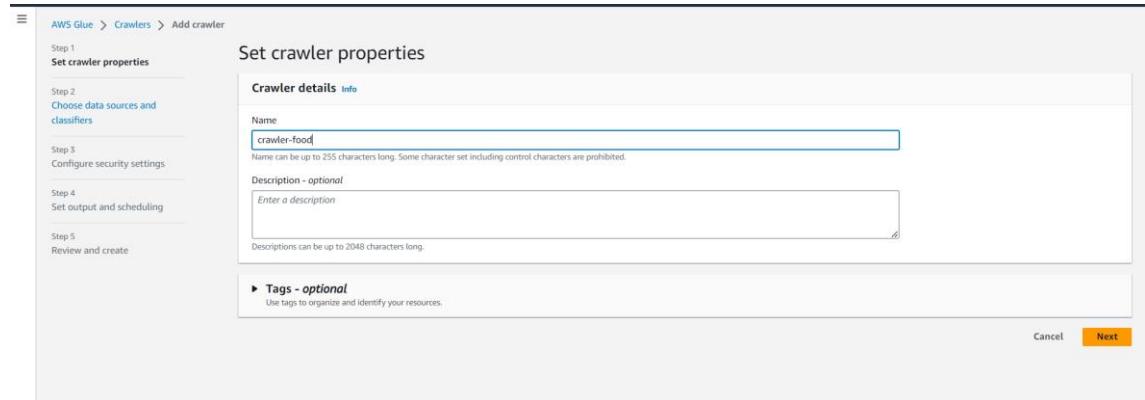
- 1- A Ingestão de dados no S3 (bucket) foi feita através do upload no arquivo na plataforma. Realizei a criação da pasta (mvp-engdados) e subir o arquivo CSV na casa raw-data, dessa forma consigo organizar os arquivos brutos, aqueles que ainda não foram processados.



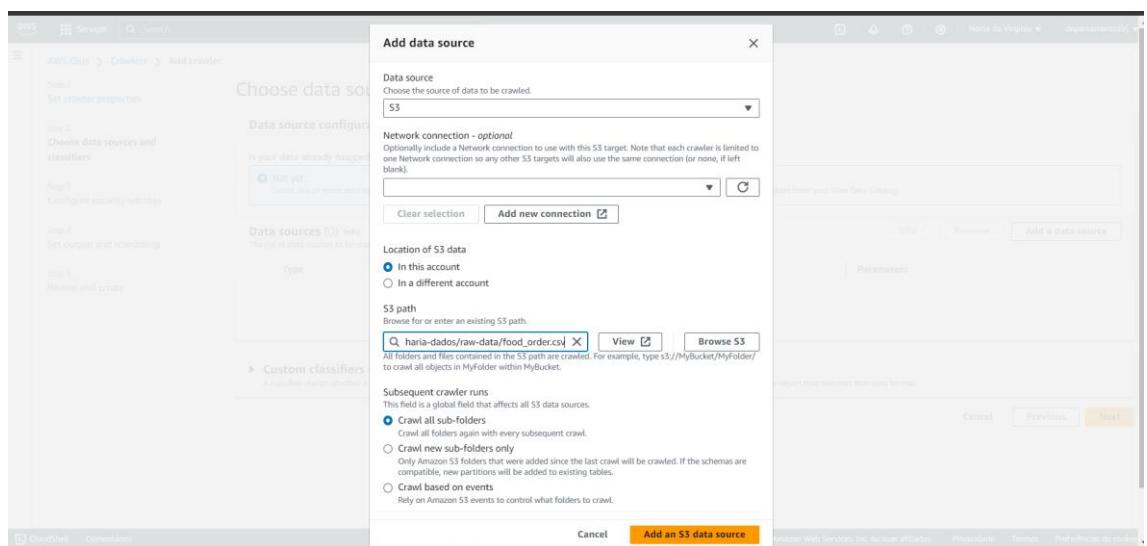
The screenshot shows the Amazon S3 console interface. The URL in the address bar is [Amazon S3 > Buckets > mvp-engenharia-dados > raw-data/](#). The page title is "raw-data/". On the right, there is a "Copiar URI do S3" button. Below the title, there are tabs for "Objetos" and "Propriedades", with "Objetos" selected. A sub-header "Objetos (1)" is shown. A note below it says: "Os objetos são as entidades fundamentais armazenadas no Amazon S3. Você pode usar o [inventário do Amazon S3](#) para obter uma lista de todos os objetos em seu bucket. Para outras pessoas acessarem seus objetos, você precisará conceder permissões explicitamente a elas. [Saiba mais](#)". Below this are buttons for "Copiar URI do S3", "Copiar URL", "Fazer download", "Abrir", "Excluir", "Ações", "Criar pasta", and "Carregar". A search bar "Localizar objetos por prefixo" is present. A table lists one object: "food_order.csv" (CSV type, 121.0 KB, last modified 30 Sep 2023 08:45:27 AM -03). The table has columns for checkbox, Name, Tipo, Última modificação, Tamanho, and Classe de armazenamento.

	Nome	Tipo	Última modificação	Tamanho	Classe de armazenamento
<input type="checkbox"/>	food_order.csv	CSV	30 Sep 2023 08:45:27 AM -03	121.0 KB	Padrão

- 2- Para a próxima etapa, foi necessário criar um crawler , que nada mais é do que uma ferramenta do AWS GLUE onde é possível processar e catalogar os dados e os metadados.



- 2.1- Nessa etapa será necessário localizar o arquivo ou pasta que contém os dados e informar a localização



2.2 - – Será necessário configuração algumas permissões de segurança e liberar no IAM para que prosseguíssemos no processo.

Configure security settings

IAM role [Info](#)

Existing IAM role
AWSGlueServiceRole-glue

[Create new IAM role](#) [Update chosen IAM role](#) [View](#)

Only IAM roles created by the AWS Glue console and have the prefix "AWSGlueServiceRole-." can be updated.

Lake Formation configuration - optional

Allow the crawler to use Lake Formation credentials for crawling the data source. [Learn more](#)

Use Lake Formation credentials for crawling S3 data source
Checking this box will allow the crawler to use Lake Formation credentials for crawling the data source. If the data source is registered in another account, you must provide the registered account ID. Otherwise, the crawler will crawl only those data sources associated to the account. Only applicable to S3, Glue Catalog, Iceberg, and Hudi data sources.

► Security configuration - optional

Enable at-rest encryption with a security configuration.

[Cancel](#) [Previous](#) [Next](#)

2.2- Teremos também que criar ou apenas informar o local de saída dos dados processados. No meu caso criei uma pasta com o nome “MVP-database”

AWS Glue > Crawlers > Add crawler

Step 1 Set crawler properties

Step 2 Choose data sources and classifiers

Step 3 Configure security settings

Step 4 Set output and scheduling

Step 5 Review and create

Set output and scheduling

Output configuration [Info](#)

Target database
mvp-database

[Clear selection](#) [Add database](#)

Table name prefix - optional
Type a prefix added to table names

Maximum table threshold - optional
This field sets the maximum number of tables the crawler is allowed to generate. In the event that this number is surpassed, the crawl will fail with an error. If not set, the crawler will automatically generate the number of tables depending on the data schema.
Type a number greater than 0

► Advanced options

Crawler schedule
You can define a time-based schedule for your crawlers and jobs in AWS Glue. The definition of these schedules uses the Unix-like cron syntax. [Learn more](#)

Frequency
On demand

[Cancel](#) [Previous](#) [Next](#)

2.3- Após o crawler criado será necessário executá-lo para que ele possa rodar e fazer o catalogo dos dados.

The screenshot shows the AWS Glue Crawler interface. At the top, it says 'Crawlers' and provides a brief description: 'A crawler connects to a data store, progresses through a prioritized list of classifiers to determine the schema for your data, and then creates metadata tables in your data catalog.' Below this is a table titled 'Crawlers (2) Info'. The table has columns: Name, State, Schedule, Last run, Last run timestamp, Log, and Table changes from last run. Two entries are listed: 'crawler-food' (Ready, Succeeded, September 30, 2023 at 13:..., View log, 1 created) and another 'crawler-food' entry (Ready). There are buttons for 'Action', 'Run', and 'Create crawler'.

3 – Após os dados serem catalogados, ele ficam armazenados no “data catalog”, onde é possível obter informações da tabela processada, como mostrado a imagem abaixo. É possível ter informações do metadados.

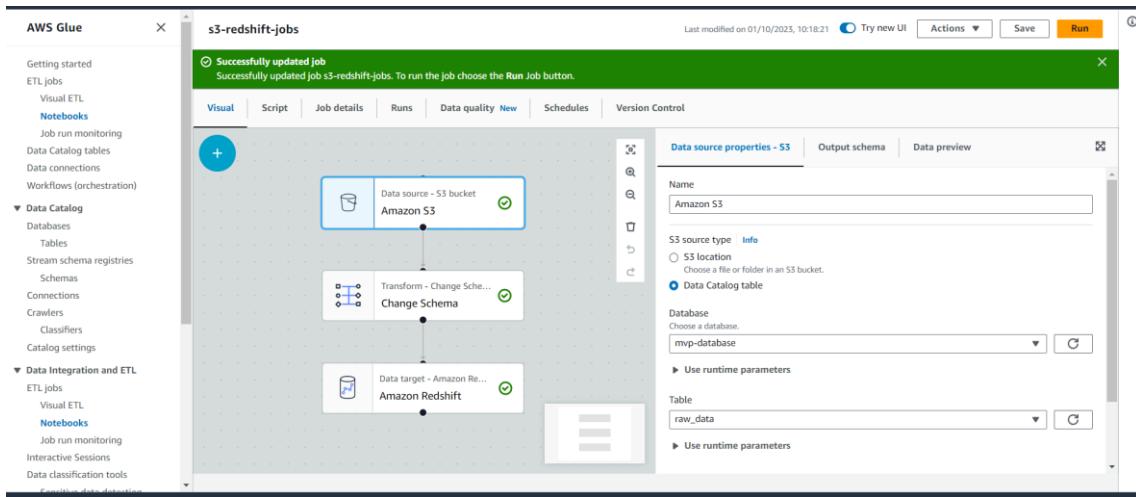
The screenshot shows the AWS Glue Data Catalog Tables interface. On the left, there's a sidebar with navigation links like 'Getting started', 'ETL jobs', 'Data connections', 'Workflow (orchestration)', 'Data Catalog', 'Databases', and 'Tables'. The main area shows a table named 'Location' with columns: Location (redacted), Connection (-), Deprecated (-), and Last updated (September 30, 2023 at 13:17:39). Below this is a detailed view of the 'Schema' for the table 'Location'. It shows 9 columns: order_id (bigint), customer_id (bigint), restaurant_name (string), cuisine_type (string), cost_of_the_order (double), day_of_the_week (string), rating (string), food_preparation_time (bigint), and delivery_time (bigint). There are buttons for 'Edit schema as JSON' and 'Edit schema'.

4 – Agora com os dados já processados será necessário criar uma conexão para podermos conectar no redshift através do Job e assim injetar nossos dados dentro do banco. Obs: É sempre uma boa prática testar o conector criado antes de executar o job que irá ser utilizado.

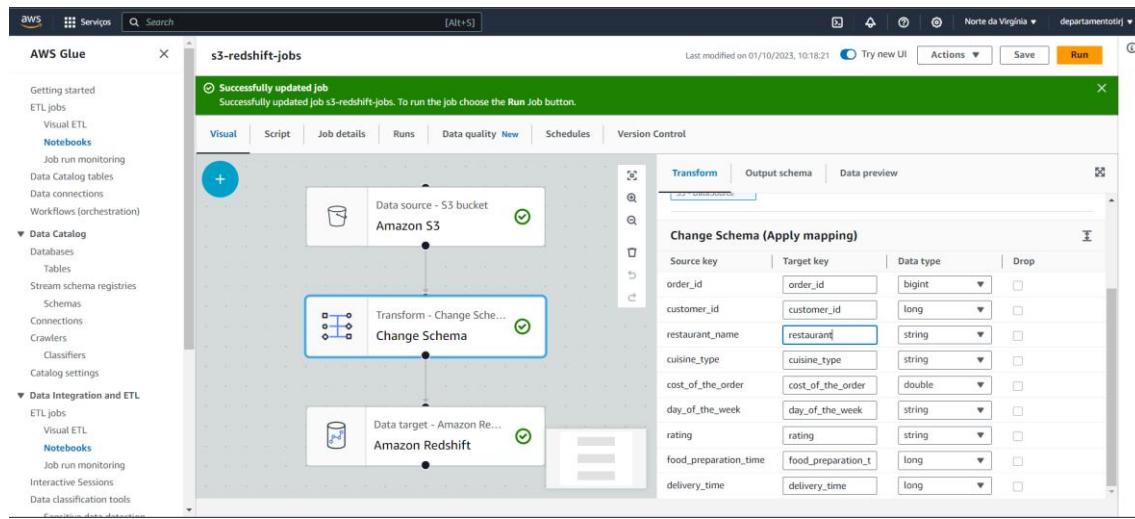
The screenshot shows the AWS Glue Connectors interface. The sidebar includes 'Getting started', 'ETL jobs', 'Data connections', 'Workflow (orchestration)', 'Data Catalog', 'Databases', 'Tables', 'Stream schema registries', 'Schemas', 'Connections', 'Crawlers', 'Classifiers', and 'Catalog settings'. The main area shows 'Marketplace connectors' and 'Custom connectors'. A 'Test Connection' dialog box is open, showing the message 'Successfully connected to the data store with connection connection-redshift.' with a 'View log' button. Below this is a 'Connections (1) Info' table with one entry: 'connection-redshift' (JDBC, Sep 25, 2023). There are buttons for 'Actions', 'Create connection', and 'Create job'.

5 – Agora que já subimos o arquivo no S3, já criamos o crawler para catalogar os dados, conseguimos visualizar os dados catalogados e a conexão com o conector foi validada, agora é o momento de criar o Job e subir os dados do S3 para o redshift.

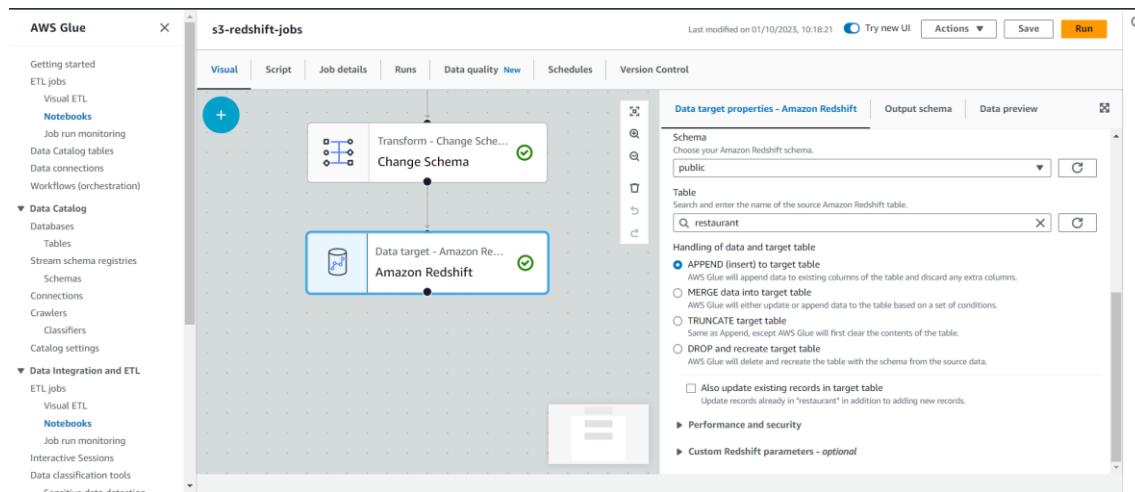
5.1- Para iniciar a criação do Job, adicionei o S3(bucket) onde estão salvo os dados processados e optei por escolher a opção de catalogo de dados, onde está salva a table processada pelo crawler



5.2 – Nessa etapa irei aplicar algumas transformações no nome de cada e coluna e também no tipo de dados de cada uma delas.



5.3 – Na etapa para salva a saída dos dados, foi necessário selecionar o conector entre o s3 e o Redshift, que foi criado anteriormente, e também selecionar a tabela dentro do redshift que esse novos dados irão pertencer. No meu caso , criei uma tabela chamada “restaurant” dentro do schema “Public” e optei pela opção do Job dar um insert dos dados na tabela especificada.



5.4 - Para finalizar o Job e coloca-lo em produção, Precisei acessar a aba “Job Details” e selecionar a IAM ROLE que permite obter esses acessos.

AWS Glue

s3-redshift-jobs

Last modified on 01/10/2023, 10:18:21 Try new UI Actions Save Run

Visual Script Job details Runs Data quality New Schedules Version Control

Basic properties Info

Name: s3-redshift-jobs

Description - optional:

IAM Role: AWSGlueServiceRole-glue

Type: Spark

Glue version: Glue 4.0 - Supports spark 3.3, Scala 2, Python 3

Language:

Getting started
ETL jobs
Visual ETL
Notebooks
Job run monitoring
Data Catalog tables
Data connections
Workflows (orchestration)

Data Catalog
Databases
Tables
Stream schema registries
Schemas
Connections
Crawlers
Classifiers
Catalog settings

Data Integration and ETL
ETL jobs
Visual ETL
Notebooks
Job run monitoring
Interactive Sessions
Data classification tools

5.5 – Agora basta rodar o Job e aguardar que ele faça a ingestão de dados no redshift.

AWS Glue

s3-redshift-jobs

Last modified on 01/10/2023, 10:18:21 Try new UI Actions Save Run

Visual Script Job details Runs Data quality New Schedules Version Control

Job runs (1/4) Info Last updated (UTC): October 1, 2023 at 14:16:38

Filter job runs by property:

Run status	Retries	Start time	End time	Duration	Capacity (DPUUs)	Worker type	Glue version
Succeeded	0	09/30/2023 14:16:35	09/30/2023 14:20:18	3 m 12 s	10 DPUUs	G.1X	4.0
Failed	0	09/30/2023 14:05:19	09/30/2023 14:09:11	3 m 55 s	10 DPUUs	G.1X	4.0
Failed	0	09/30/2023 12:50:14	09/30/2023 12:54:58	4 m 4 s	10 DPUUs	G.1X	4.0
Succeeded	0	09/30/2023 12:27:39	09/30/2023 12:31:09	3 m 3 s	10 DPUUs	G.1X	4.0

Table View Card View

09/30/2023 14:16:35

Job name: s3-redshift-jobs

Id: [REDACTED]

Run status: Succeeded

Glue version: 4.0

Getting started
ETL jobs
Visual ETL
Notebooks
Job run monitoring
Data Catalog tables
Data connections
Workflows (orchestration)

Data Catalog
Databases
Tables
Stream schema registries
Schemas
Connections
Crawlers
Classifiers
Catalog settings

Data Integration and ETL
ETL jobs
Visual ETL
Notebooks
Job run monitoring
Interactive Sessions
Data classification tools

6 – Após a execução do Jobs, é possível visualizar e realizar consulta dentro do redshift.

The screenshot shows the AWS Redshift Query Editor interface. On the left, the sidebar displays the database schema with a tree view of tables, views, functions, and stored procedures under the 'dev' and 'public' schemas. In the main area, a query is being run:

```
1 SELECT *
2 FROM "dev"."public"."restaurant";
```

The results of the query are displayed in a table titled 'Result 1 (100)'. The table has eight columns: order_id, customer_id, rating, cost_of_the_order, delivery_time, restaurant, and cuisine_type. The data includes various restaurant names and their characteristics. At the bottom of the results table, it says 'Elapsed time: 38 ms Total rows: 100'.

order_id	customer_id	rating	cost_of_the_order	delivery_time	restaurant	cuisine_type
1477607	386995	Not given	9.75	25	Han Dynasty	Chinese
1477087	206039	5	12.66	15	Shake Shack	American
1478263	201535	Not given	9.51	29	Blue Ribbon Fried Chicken	American
1477410	39011	Not given	9.07	24	Mamoun's Falafel	Mediterranean
1478359	361616	3	24.2	32	Blue Ribbon Brooklyn	American
1477249	351561	Not given	20.13	31	The Kali Roti Company	Indian
1476650	88201	5	8.35	23	J. G. Melon	American
1477904	48131	4	16.1	29	RedFarm Hudson	Chinese
1477740	129798	4	25.27	24	TAO	Japanese

Análise de dados

A análise de dados é o processo de examinar informações para encontrar respostas, identificar tendências e tomar decisões mais inteligentes. Ela envolve olhar para números, fatos e figuras para descobrir o que eles nos dizem. A análise de dados é uma ferramenta valiosa em muitos campos, ajudando a resolver problemas, melhorar processos e impulsionar o sucesso. É como usar um superpoder para entender e agir com base nos dados disponíveis. Com isso, desenvolvi algumas query que é possível extrair informações para novas tomadas de decisões como mostrarei abaixo.

1 – Listando todas informações do dataset e entender como ele está sendo distribuído.
O dataset possui 1898 pedidos.

SELECT

*

FROM

"dev"."public"."restaurant";

The screenshot shows the AWS Redshift query editor interface. On the left, the sidebar displays the schema structure: 'Serverless: default-workgroup' containing 'awsdatabatalog' and 'dev' which further contains 'public' and 'sample_data_dev'. Under 'public', there is a single table named 'restaurant'. The main area shows a query window with the following SQL code:

```
1 SELECT
2 *
3 FROM
4 "dev"."public"."restaurant";
```

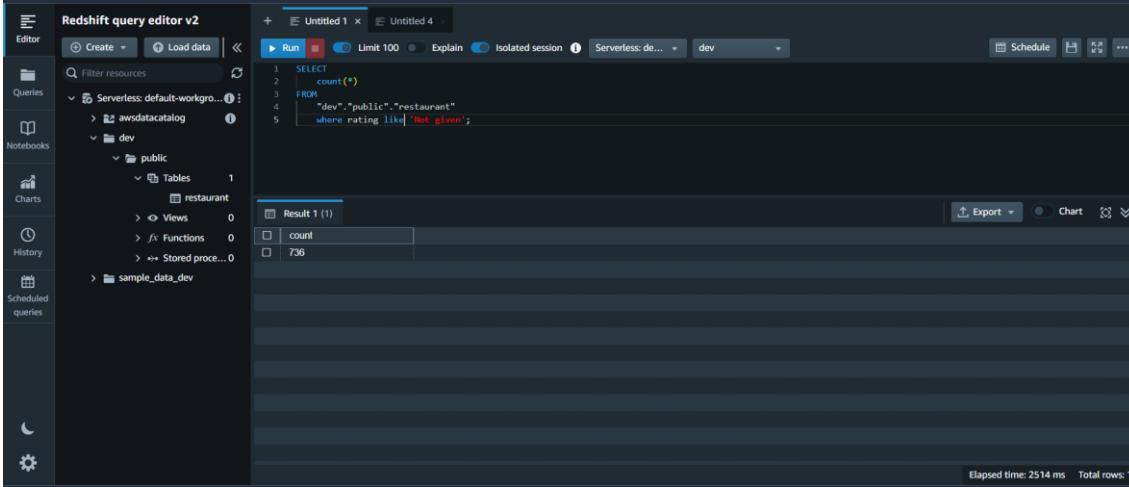
Below the query window is a results table titled 'Result 1 (100)'. The table has 10 columns: order_id, customer_id, rating, cost_of_the_order, delivery_time, restaurant, and cuisine_type. The first few rows of data are:

order_id	customer_id	rating	cost_of_the_order	delivery_time	restaurant	cuisine_type
1477607	386995	Not given	9.75	25	Han Dynasty	Chinese
1477087	206039	5	12.66	15	Shake Shack	American
1478263	201535	Not given	9.51	29	Blue Ribbon Fried Chicken	American
1477410	39011	Not given	9.07	24	Mamoun's Falafel	Mediterran
1478359	361616	3	24.2	32	Blue Ribbon Brooklyn	American
1477249	351561	Not given	20.13	31	The Kali Roll Company	Indian
1476650	88201	5	8.35	23	J. G. Melon	American
1477904	48131	4	16.1	29	RedFarm Hudson	Chinese
1477740	129798	4	25.27	24	TAO	Japanese
1477281	309513	5	22.36	27	Otto Enoteca Pizzeria	Italian
1477021	231463	Not given	14.46	29	Mamoun's Falafel	Mediterran
1477835	129518	4	15.33	29	Blue Ribbon Fried Chicken	American
1477111	229946	3	12.13	25	Chipotle Mexican Grill \$1...	Mexican

At the bottom of the results table, it says 'Elapsed time: 186 ms Total rows: 100'.

2 – Todos os pedidos possuem alguma tipo de classificação entre 3 e 5 e se o usuário não quiser dar alguma classificação, a classificação fica registrado no banco de dados como “Not given”.

Nessa análise, percebi que a proporção de pedidos não classificados “**Not given**” chegou a ter uma porcentagem expressiva de 39% que corresponde num total de 736 pedidos.



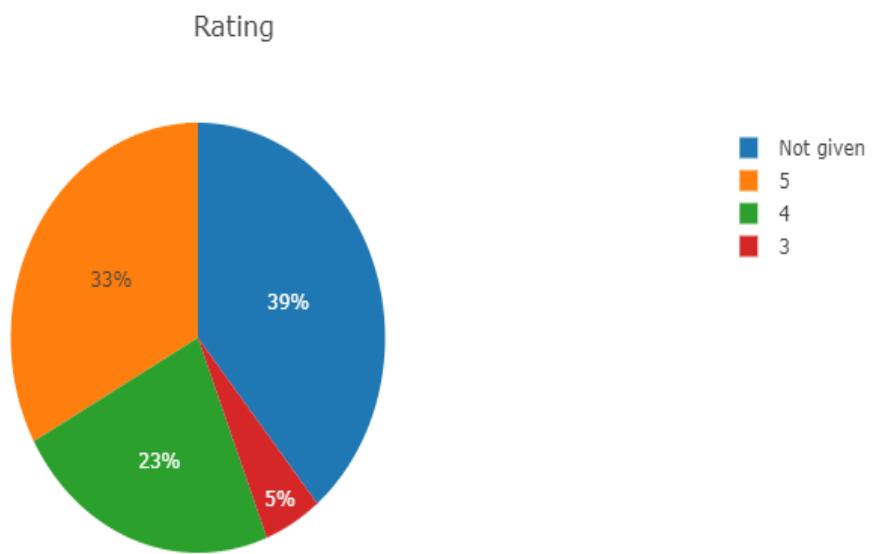
The screenshot shows the Redshift query editor interface. On the left is the sidebar with options like Editor, Queries, Notebooks, Charts, History, and Scheduled queries. The main area has tabs for Untitled 1 and Untitled 4. Untitled 1 contains a SQL query:

```
1 SELECT
2     count(*)
3     FROM
4     "dev"."public"."restaurant"
5     WHERE rating like 'not given';
```

The results pane shows the output of the query:

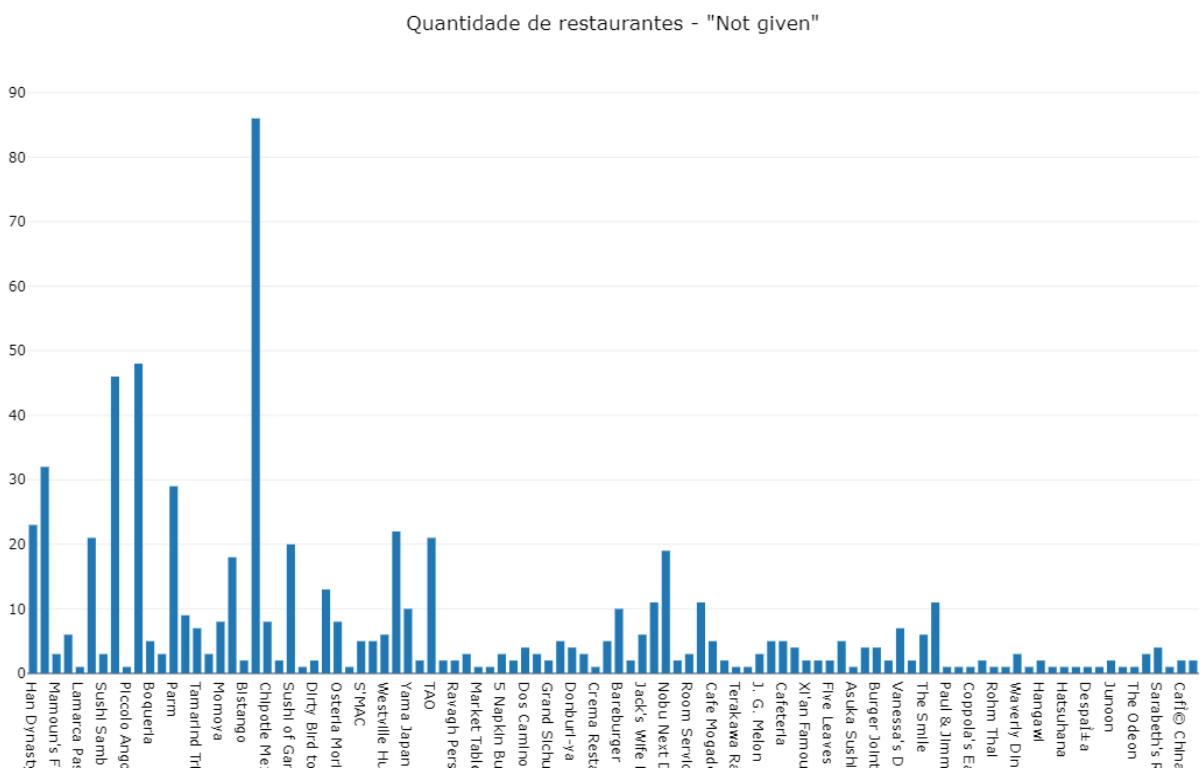
count	736
count	736

At the bottom right of the results pane, it says "Elapsed time: 2514 ms Total rows: 1".



3 - Para entender melhor o motivo dessa alta taxa, listei todo os restaurantes e verifiquei a quantidade de pedidos com a classificação "Not given" de cada um deles e com isso.

O primeiro restaurante da lista é o **Shake Shack** e possui 86 pedidos sem classificação e o restaurante que está na **10º Sushi of Gari 46** possui 20 pedidos nessa classificação. Existe exatamente 134 restaurantes nesta lista.



SELECT

```
    restaurant, count(order_id) as order_id  
FROM  
    "dev"."public"."restaurant"  
    where rating like 'Not given'  
    group by restaurant  
    order by order_id desc;
```

The screenshot shows the AWS Redshift query editor interface. The left sidebar displays the schema with a table named 'restaurant'. The main area contains a query window with the following SQL code:

```

1 SELECT
2     restaurant, count(order_id) as order_id
3 FROM
4     "dev"."public"."restaurant"
5     where rating like 'Not given'
6     group by Restaurant
7     order by order_id desc;

```

The results pane shows a table titled 'Result 1 (100)' with two columns: 'restaurant' and 'order_id'. The data includes:

restaurant	order_id
Shake Shack	86
The Meatball Shop	48
Blue Ribbon Sushi	46
Blue Ribbon Fried Chicken	32
Parm	29
Han Dynasty	23
Blue Ribbon Sushi Bar & ...	22
RedFarm Hudson	21
TAO	21
Sushi of Gari 46	20
Nobu Next Door	19
RedFarm Broadway	18
Rubirosa	13

At the bottom right of the results pane, it says 'Elapsed time: 2732 ms Total rows: 100'.

Listando a quantidade de restaurante com possui a classificação “ Not given”

SELECT

count(distinct restaurant)

FROM

"dev"."public"."restaurant"

where rating like 'Not given';

The screenshot shows the AWS Redshift query editor interface. The left sidebar displays the schema with a table named 'restaurant'. The main area contains a query window with the following SQL code:

```

1 SELECT
2     count(distinct restaurant)
3 FROM
4     "dev"."public"."restaurant"
5     where rating like 'Not given';
6

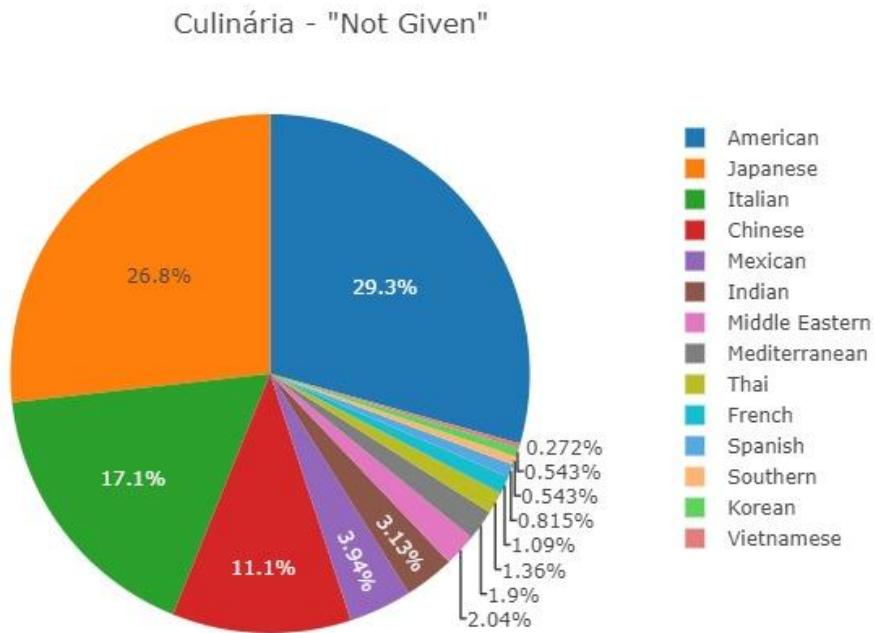
```

The results pane shows a table titled 'Result 1 (1)' with one column: 'count'. The data includes:

count
134

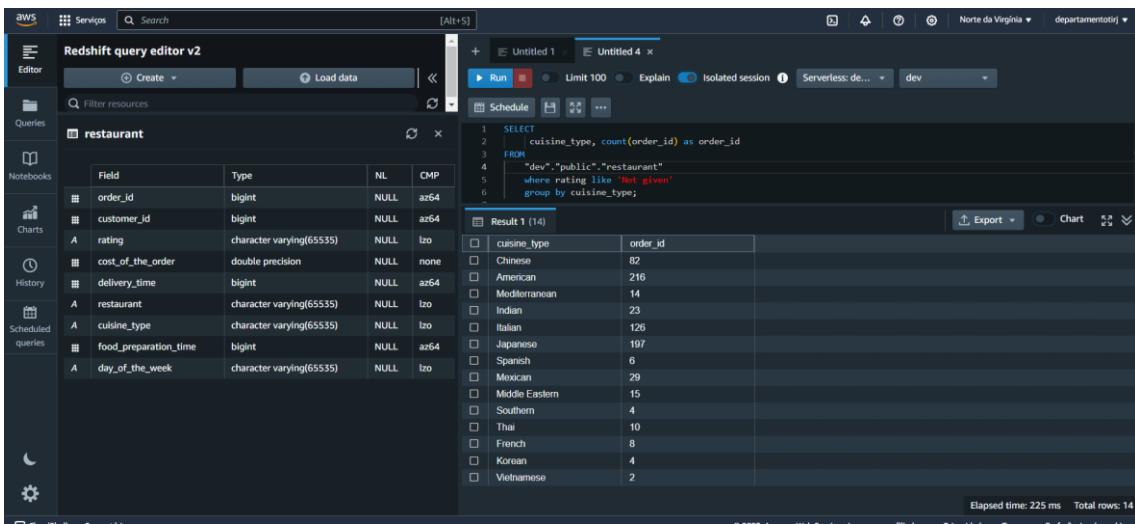
At the bottom right of the results pane, it says 'Elapsed time: 3281 ms Total rows: 1'.

4 – Após analisar os restaurantes, agora diminuir o filtro e pesquisei quais são os tipos de culinária que estão com índice ele levado de “Not given”



SELECT

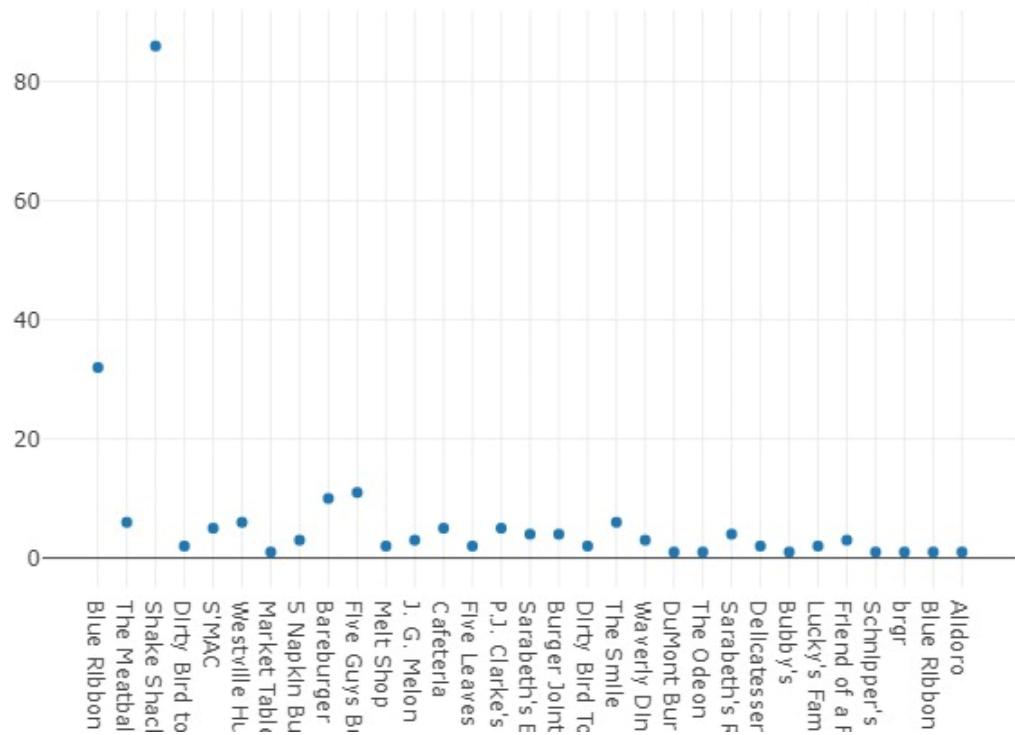
```
cuisine_type, count(order_id) as order_id  
FROM  
"dev"."public"."restaurant"  
where rating like 'Not given'  
group by cuisine_type;
```



The screenshot shows the AWS Redshift query editor interface. On the left, there's a sidebar with various navigation links: Editor, Queries, Notebooks, Charts, History, and Scheduled queries. The main area has tabs for 'Untitled 1' and 'Untitled 4'. The current tab, 'Untitled 4', contains the SQL query provided above. Below the query is a table titled 'Result 1 (14)' which lists the cuisine types and their corresponding order counts. At the bottom right of the results table, it says 'Elapsed time: 225 ms Total rows: 14'.

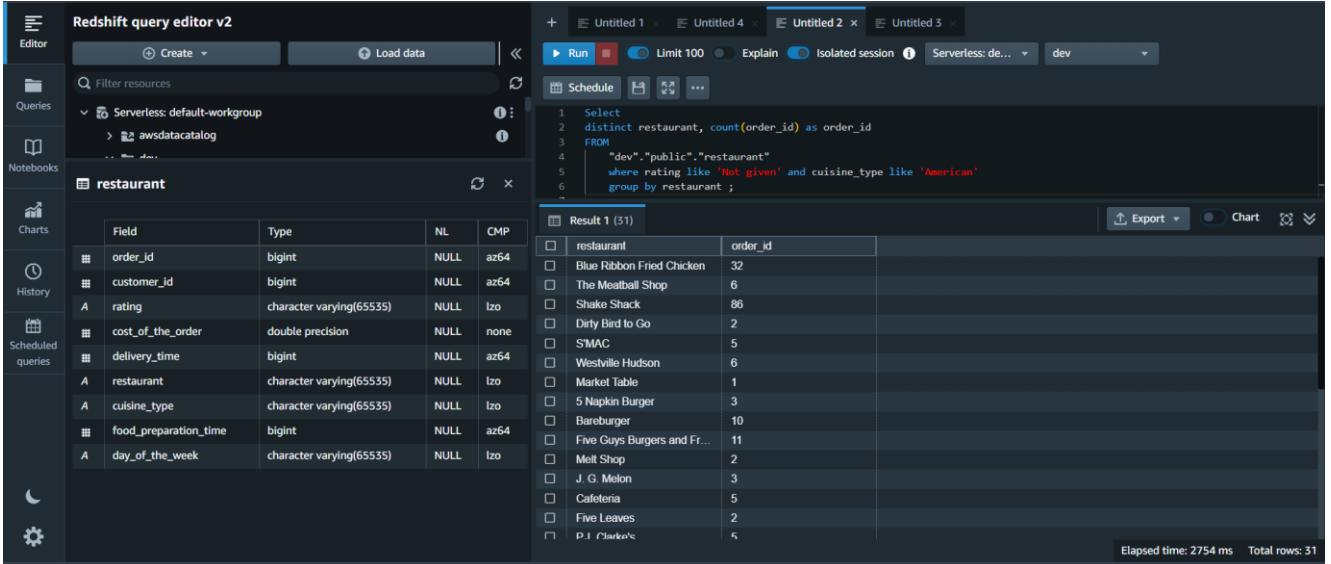
cuisine_type	order_id
Chinese	82
American	216
Mediterranean	14
Indian	23
Italian	126
Japanese	197
Spanish	6
Mexican	29
Middle Eastern	15
Southern	4
Thai	10
French	8
Korean	4
Vietnamese	2

5 – Continuando com a analise mais a fundo, verifiquei pode existem 31 restaurantes que vendem comida America, em um total de 134 restaurantes e que o restaurante **Shake Shack** possui o maior índice de pedido de culinária américa sem classificação.



Select

```
distinct restaurant, count(order_id) as order_id  
FROM  
"dev"."public"."restaurant"  
where rating like 'Not given' and cuisine_type like 'American'  
group by restaurant ;
```

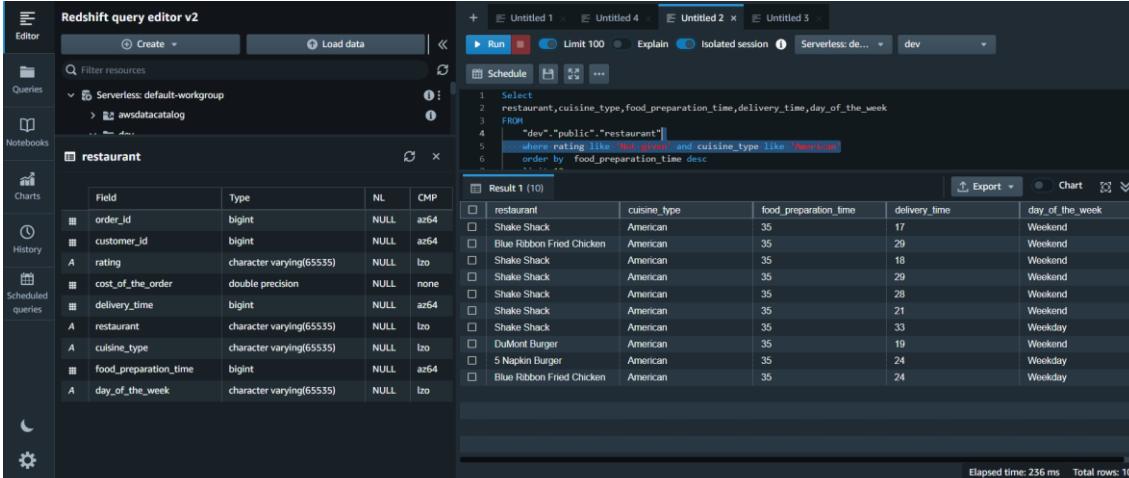


The screenshot shows the Redshift query editor interface. On the left, there's a sidebar with icons for Editor, Queries, Notebooks, Charts, History, and Scheduled queries. The main area has tabs for Untitled 1, Untitled 4, Untitled 2 (which is active), and Untitled 3. The Untitled 2 tab contains the SQL query provided above. Below the query is a table titled "Result 1 (51)" showing the results of the query. The results list various restaurants along with their order counts. The table has columns: restaurant, order_id. The results are as follows:

restaurant	order_id
Blue Ribbon Fried Chicken	32
The Meatball Shop	6
Shake Shack	86
Dirty Bird To Go	2
SMAC	5
Westville Hudson	6
Market Table	1
5 Napkin Burger	3
Bareburger	10
Five Guys Burgers and Fr...	11
Melt Shop	2
J. G. Melon	3
Cafeteria	5
Five Leaves	2
P.J. Clarke's	5

At the bottom right of the editor, it says "Elapsed time: 2754 ms Total rows: 31".

6 – A última análise feita em cima dos dados, me mostrou que o motivo do alto índice de “Not Given” em cima da Culinária Americana está vindo do Restaurante “ Shake Shack” que em outra analise aparecia de forma significativa. Analise também mostrou que entre 10 pedidos com maior tempo de preparo, 6 foram feito pela Shake Shack



The screenshot shows the Redshift query editor interface, similar to the previous one. The sidebar and tabs are identical. The Untitled 2 tab contains a modified version of the previous query, including an ORDER BY clause for food_preparation_time desc. Below the query is a table titled "Result 1 (10)" showing the top 10 results. The table has columns: restaurant, cuisine_type, food_preparation_time, delivery_time, day_of_the_week. The results are as follows:

restaurant	cuisine_type	food_preparation_time	delivery_time	day_of_the_week
Shake Shack	American	35	17	Weekend
Blue Ribbon Fried Chicken	American	35	29	Weekend
Shake Shack	American	35	18	Weekend
Shake Shack	American	35	29	Weekend
Shake Shack	American	35	28	Weekend
Shake Shack	American	35	21	Weekend
Shake Shack	American	35	33	Weekday
DuMont Burger	American	35	19	Weekend
5 Napkin Burger	American	35	24	Weekday
Blue Ribbon Fried Chicken	American	35	24	Weekday

At the bottom right of the editor, it says "Elapsed time: 236 ms Total rows: 10".

```
Select  
restaurant,cuisine_type,food_preparation_time,delivery_time,day_of_the_week  
FROM  
"dev"."public"."restaurant"  
where rating like 'Not given' and cuisine_type like 'American'  
order by food_preparation_time desc  
limit 10;
```

Conclusão

Com a análise feita em cima do dataset, objetivo para entender o motivo da alta classificação dos pedido com “ Not given” se deu pela Culinaria américa vim do Restaurante Shake Shack, onde nele o tempo de preparação dos alimentos alcançou o máximo de 35 min, tempo que se reflete na maioria dos pedidos do restaurante.