Meteorologia, acompanhando por meio de dados as mudanças climáticas atuais

## Anselmo Berriel de Lira, Beatriz Ornelas Miranda, Guilherme Teixeira Silva, Roberto Borba Pinto, Thiago dos Santos Rodrigues, Thomas Eduardo da Silva Olympio, Wellington Patrick Viana dos Santos

PUC Minas

Curso de Tecnologia em Banco de Dados

[ablira@sga.pucminas.br,](mailto:ablira@sga.pucminas.br) [beatriz.miranda@sga.pucminas.br,](mailto:beatriz.miranda@sga.pucminas.br) [guilherme.silva.1396238@sga.pucminas.br,](mailto:guilherme.silva.1396238@sga.pucminas.br) [rbpinto@sga.pucminas.br,](mailto:rbpinto@sga.pucminas.br) [thiago.rodrigues@sga.pucminas.br,](mailto:thiago.rodrigues@sga.pucminas.br) [tesolympio@sga.pucminas.br,](mailto:tesolympio@sga.pucminas.br) [wellington.patrick@sga.pucminas.br](mailto:wellington.patrick@sga.pucminas.br)

***Resumo:*** *O grupo de estudos da PUC MINAS em parceria a empresa Fios de Luxo buscará explicar a relação das estações e como a temperatura afeta o comercio localizado em Salvador, Bahia. A necessidade da empresa de entender o clima pode ser a chave para antecipar tendências e o grupo está determinado a encontrar as respostas diante das dúvidas da empresa parceiras.*

***Abstract:*** *The PUC MINAS study group, in partnership with the company Fios de Luxo, will try to explain the relationship between the seasons and how the temperature affects the business located in Salvador, Bahia. The company's need to understand the climate could be the key to anticipating trends and the group is determined to find answers to the partner company's questions.*

# Introdução

Busca-se com esta análise de dados que será realizada, trazer informações completas a respeito de como o clima e suas estações podem influenciar o setor comercial que abrangem atendimentos focados em um salão de beleza visando a melhor maneira para utilizar as informações em benefício de um comércio bem-sucedido.

A análise de dados será de extrema importância para atender e antecipar tendências no comercio da empresa Fios de Luxo, entender as demandas do seu público-alvo, como o clima afeta a relação de consumo, antecipar tendências por estações é o principal motivador para o grupo.

Para esse estudo, serão coletados os dados dos seguintes

URL’s:

https://[www.tempo.com/](http://www.tempo.com/)

Os dados a serem coletados serão os seguintes:

* Pressão
* Temperatura
* Vento
* Umidade
* Resumo dia

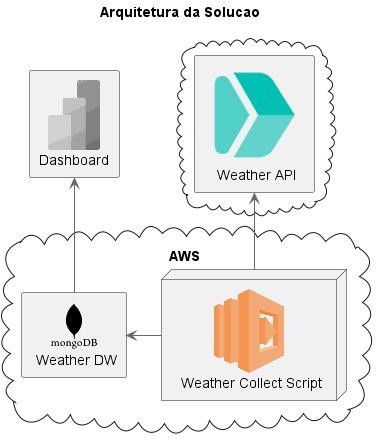
Em sequência do tratamento das regiões selecionadas será da grande capital de São Paulo, Rio de Janeiro, Florianópolis e Boa Vista, Salvador.

# Componentes de Arquitetura

Modelagem: SQL Relacional - MySQL SGBD: MySQL

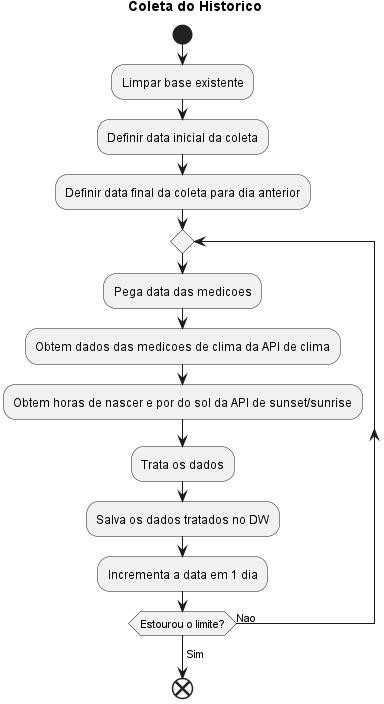
Plataforma e serviços de nuvem: AWS Frameworks: Api Rest e Open Weather Linguagem de programação: Python Ferramenta de banco de dados: Dbeaver IDE: Visual Studio Code

# Arquitetura da Solução



*Figura 1- Arquitetura da solução*

# Coleta do Histórico



*Figura 2- Coleta do hitstórico*

# Modelagem de Dados:

O projeto utilizará o esquema de banco de dados no MySQL para armazenar os dados meteorológicos, como temperatura, clima, turno. Seguindo a estrutura de Data Warehouse Star Schema.

**Tabela dimensão d\_agenda:**

data\_id DATE NOT NULL PRIMARY KEY,

dia INT DEFAULT NULL,

UNIQUE(data\_id)

**Tabela dimensão d\_clima:**

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

temperatura\_min DOUBLE DEFAULT NULL,

temperatura\_max DOUBLE DEFAULT NULL,

temperatura\_tarde DOUBLE DEFAULT NULL,

temperatura\_noite DOUBLE DEFAULT NULL,

temperatura\_manha DOUBLE DEFAULT NULL,

dia DATE NOT NULL,

data\_atual DATE NOT NULL,

FOREIGN KEY (dia) REFERENCES d\_agenda(data\_id),

UNIQUE (dia)

**Tabela dimensão d\_cliente:**

id int unsigned not null auto\_increment,

nome varchar(120) not null primary key,

telefone varchar(45) default null

**Tabela dimensãod\_funcionario:**

id int unsigned primary key not null auto\_increment,

nome varchar(120) not null,

cargo varchar(64) default null,

telefone varchar(20) default null

**Tabela dimensão d\_servico:**

id int unsigned not null primary key auto\_increment,

nome varchar(120) not null,

valor int DEFAULT 0

**Tabela fato\_pagamento:**

id\_cliente int unsigned not null,

data\_id date not null,

id\_funcionario int unsigned not null,

id\_servico int unsigned not null,

valor\_pago float default 0,

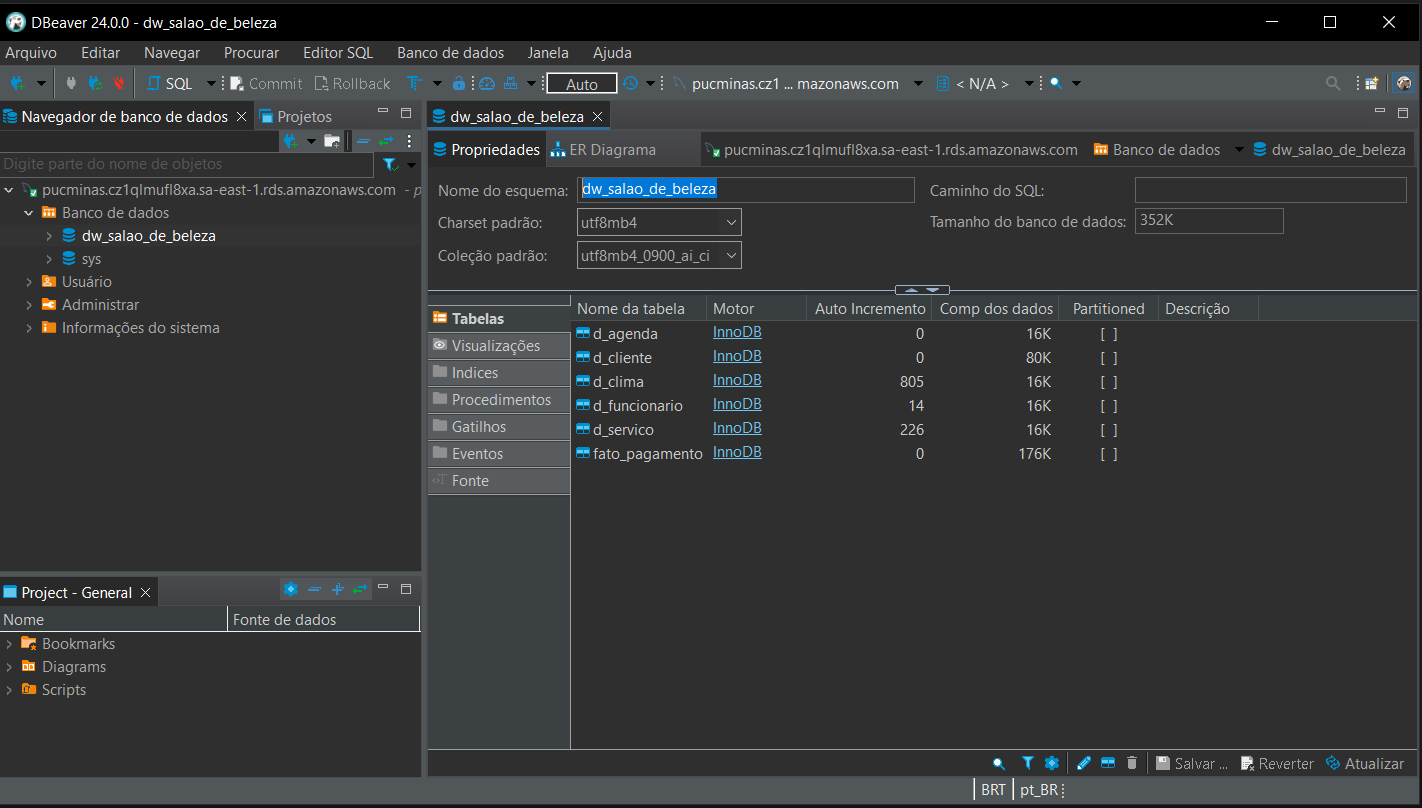
primary key (id\_cliente, data\_id, id\_funcionario, id\_servico),

foreign key FK\_FatoPagamento\_Cliente(id\_cliente) references d\_cliente(id),

foreign key FK\_FatoPagamento\_Agenda(data\_id) references d\_agenda(data\_id),

foreign key FK\_FatoPagamento\_Funcionario(id\_funcionario) references d\_funcionario(id),

foreign key FK\_FatoPagamento\_Servico(id\_servico) references d\_servico(id)



*Figura 3 – Modelagem de dados no Mysql*

# Coleta de Dados:

O sistema de coleta de dados que obterá informações meteorológicas para as cidades especificadas a partir da API REST fornecida. Os dados serão coletados em intervalos regulares (por exemplo, a cada hora) e inseridos na tabela "DadosMeteorologicos" do RDS.

# Arquitetura de Coleta:

A arquitetura de coleta terá integração com a API REST fornecida. Os serviços da AWS serão utilizados para criar funções Lambda que disparem a coleta de dados em intervalos regulares. A API REST fornecida será usada para obter os dados atualizados.

# Orquestração do pipeline de dados

## Capturando dados históricos climáticos com Python e API (OpenWeather)

import requests

import mysql.connector

from datetime import datetime, timedelta

# Dados de acesso ao banco de dados user = 'root'

password = '123456' host = 'localhost'

database = 'dw\_salao\_de\_beleza'

# Chave de API OpenWeather

API\_KEY = "221f164d35f7154a527c7b3146fa2129"

# Coordenadas de Salvador latitude = -12.9704

longitude = -38.5124

# Definindo a data inicial e a data final data\_inicial = datetime(2023, 5, 2)

data\_final = datetime(2023, 5, 31)

# Função para converter de Kelvin para Celsius def kelvin\_para\_celsius(temp\_kelvin):

return temp\_kelvin - 273.15

# Função para obter dados climáticos da API OpenWeather def obter\_temperatura(data):

print("Iniciando coleta") link =

f"https://api.openweathermap.org/data/3.0/onecall/day\_summary?lat={latitu de}&lon={longitude}&date={data.strftime('%Y-%m-%d')}&appid={API\_KEY}"

requisicao = requests.get(link) requisicao\_dic = requisicao.json() temperatura\_min =

kelvin\_para\_celsius(requisicao\_dic['temperature']['min']) temperatura\_max =

kelvin\_para\_celsius(requisicao\_dic['temperature']['max']) temperatura\_tarde =

kelvin\_para\_celsius(requisicao\_dic['temperature']['afternoon']) temperatura\_noite =

kelvin\_para\_celsius(requisicao\_dic['temperature']['night']) temperatura\_noite =

kelvin\_para\_celsius(requisicao\_dic['temperature']['evening']) temperatura\_manha =

kelvin\_para\_celsius(requisicao\_dic['temperature']['morning']) return temperatura\_min, temperatura\_max, temperatura\_tarde,

temperatura\_noite, temperatura\_manha

# Loop para coletar os dados para cada data while data\_inicial <= data\_final:

temperatura\_min, temperatura\_max, temperatura\_tarde, temperatura\_noite, temperatura\_manha = obter\_temperatura(data\_inicial)

if temperatura\_min is not None and temperatura\_max is not None and temperatura\_tarde is not None and temperatura\_noite is not None and temperatura\_manha is not None:

# Conectar ao banco de dados db\_connection = mysql.connector.connect(

host=host, user=user, password=password, database=database

)

cursor = db\_connection.cursor()

# Inserir os dados no banco de dados

insert\_query = "INSERT INTO d\_clima (temperatura\_min, temperatura\_max, temperatura\_tarde, temperatura\_noite, temperatura\_manha, dia, data\_atual) VALUES (%s, %s, %s, %s, %s, %s, %s)"

data\_insercao = (temperatura\_min, temperatura\_max, temperatura\_tarde, temperatura\_noite, temperatura\_manha, data\_inicial.strftime('%Y-%m-%d'), datetime.now().strftime('%Y-%m-%d'))

cursor.execute(insert\_query, data\_insercao) db\_connection.commit()

# Fechar a conexão com o banco de dados cursor.close()

db\_connection.close()

# Incrementar a data inicial em um dia data\_inicial += timedelta(days=1)

print("Dados coletados e inseridos no banco de dados com sucesso.")

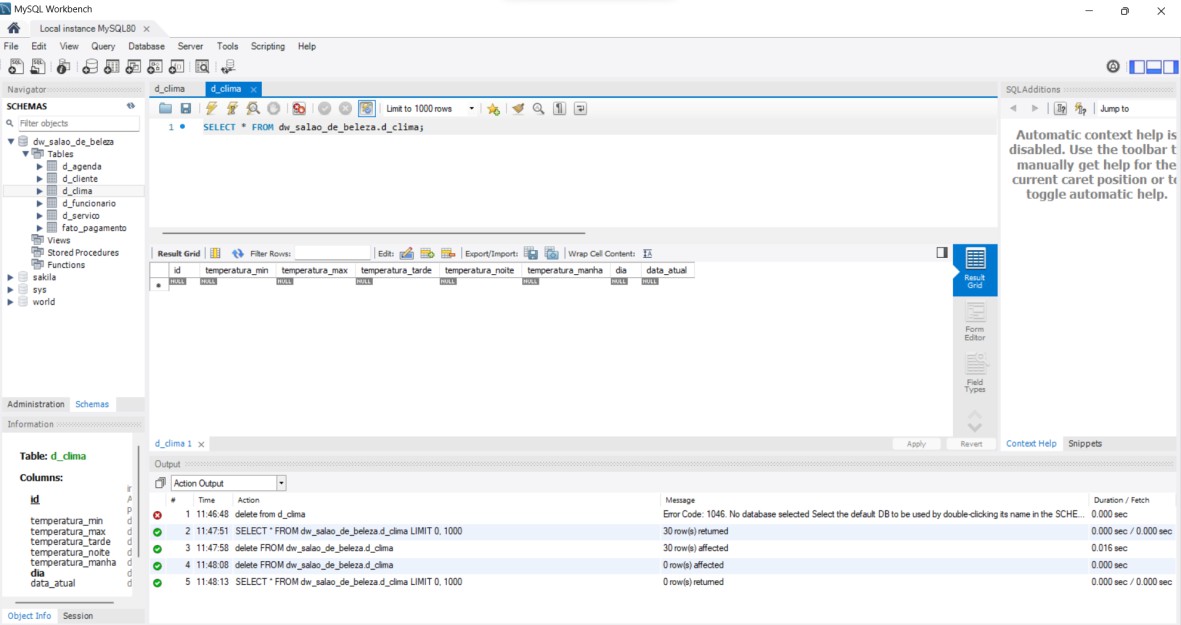
*Figura 4 – Código python*

# Desenvolvimento das atividades

A captura dos dados climáticas se inicia com a inserção da chave API que se dá através do cadastro no site, <https://openweathermap.org/>, após isso foi necessário fazer uma requisição HTTPS que gera um JSON como resultado da coleta dos dados climáticos. Coletados os dados o código faz uma conexão com o Banco de Dados MySQL, quando conectado ele transforma os dados JSON chaves em colunas para inserção dos valores armazenados. Assim ele retorna um “Coletando dados” quando inicia a coleta e “Dados coletados e inseridos no banco de dados com sucesso” quando a inserção no banco não retorna erro.

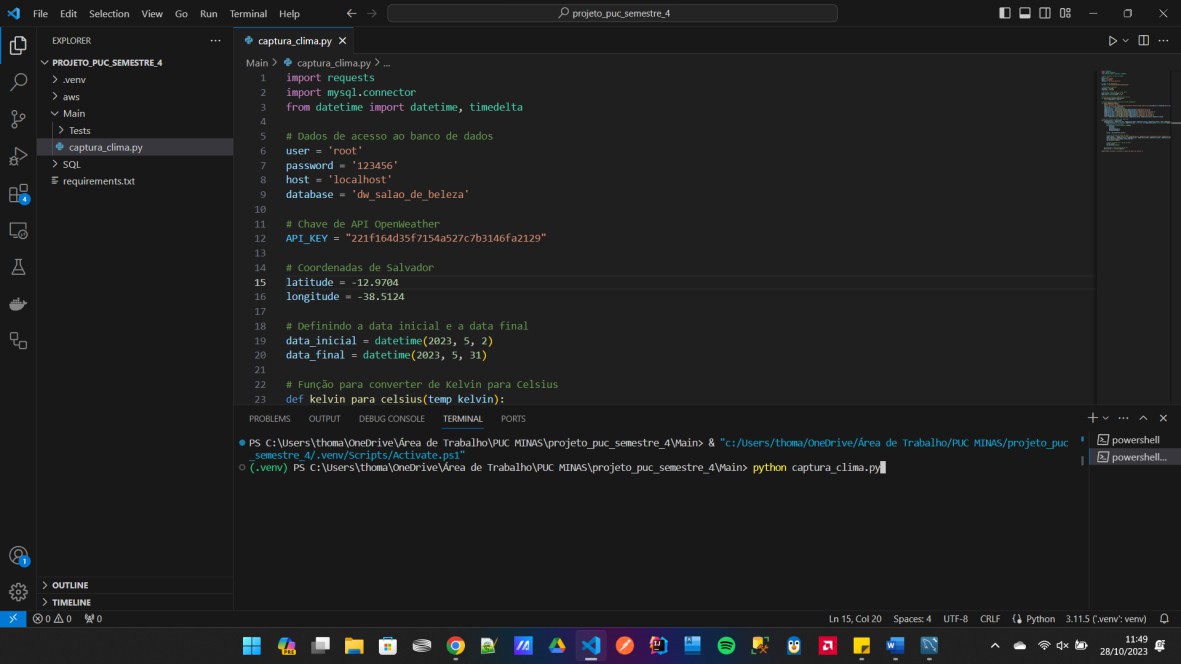
# Tabela dimensão D\_clima

A tabela *“d\_clima”* iniciou sem dados

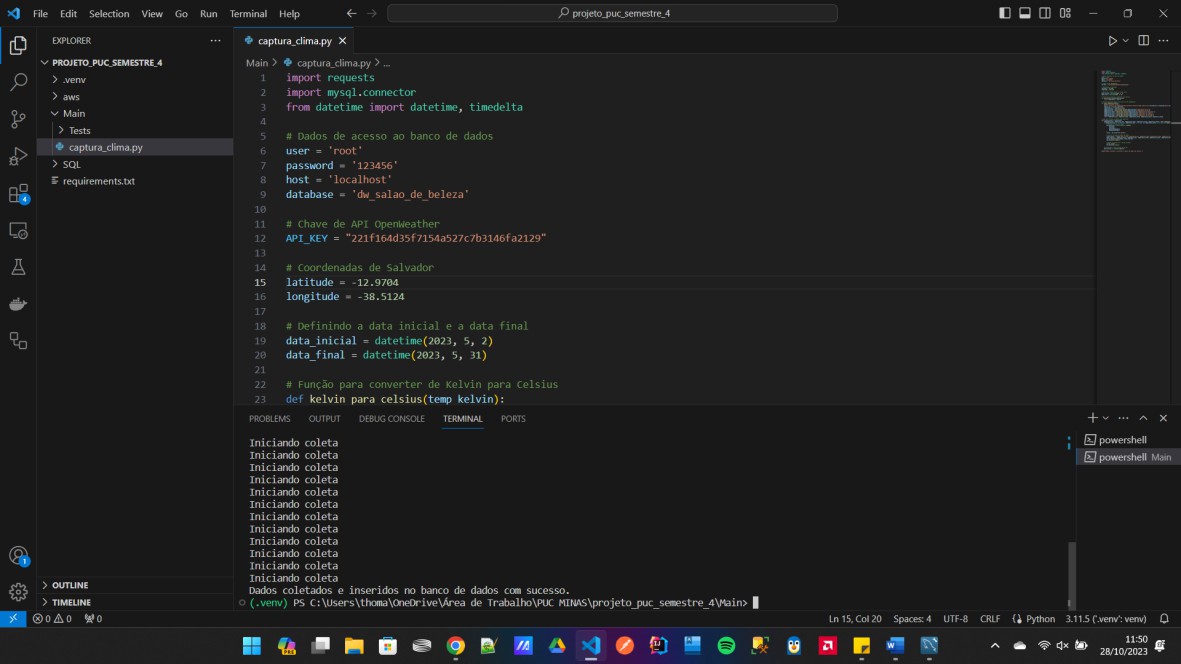


*Figura 5 - Exibição da tabela d\_clima no MySQL*

# Executando o código em Python

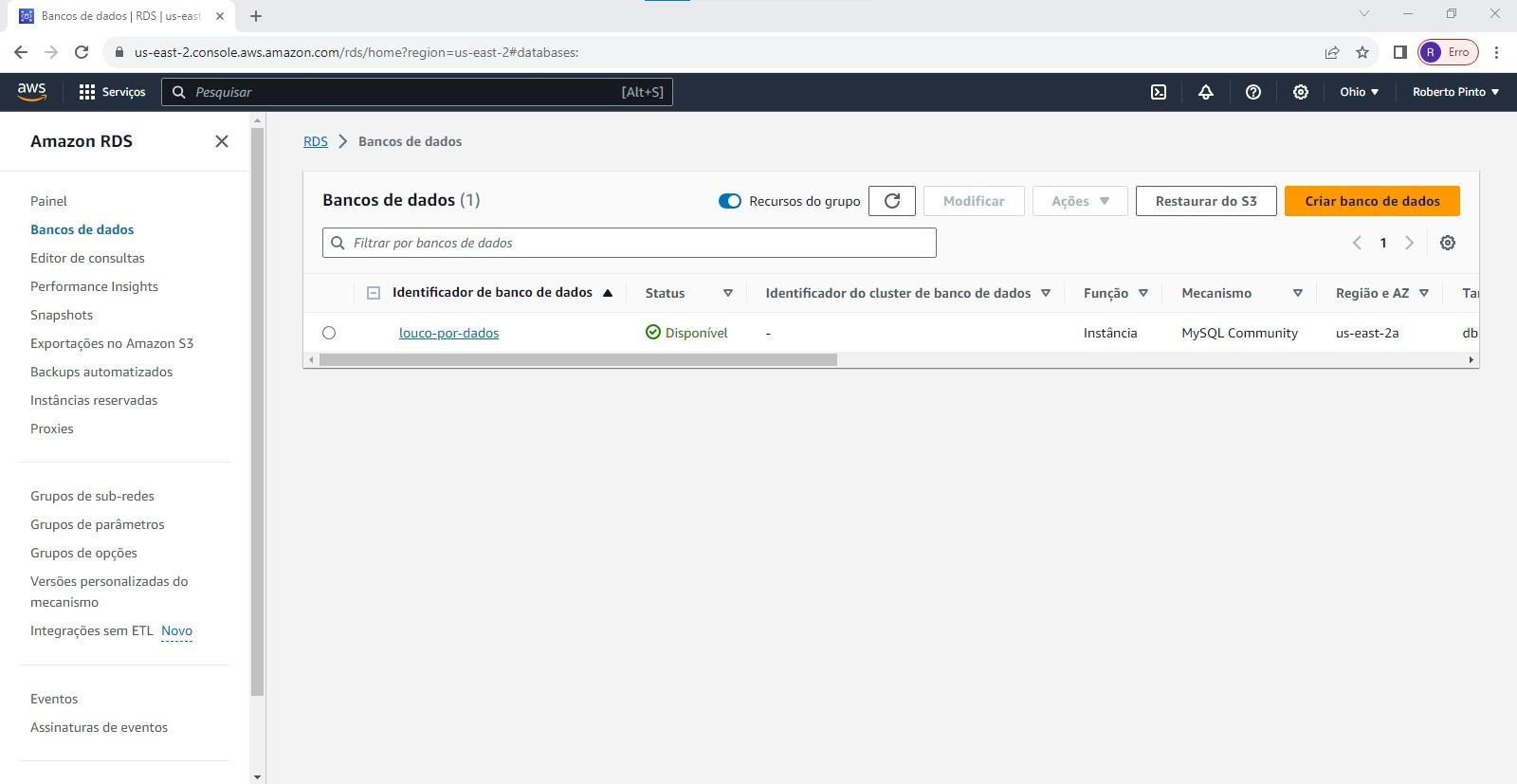


*Figura 6 – Teste no Visual Studio Code*



*Figura 7 - Continuação dos testes no Visual Studio Code*

# Configuração em ambiente AWS



*Figura 8 – Criação do banco AWS*

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

*Figura 9 - Identificação das instâncias conectadas ao AWS*

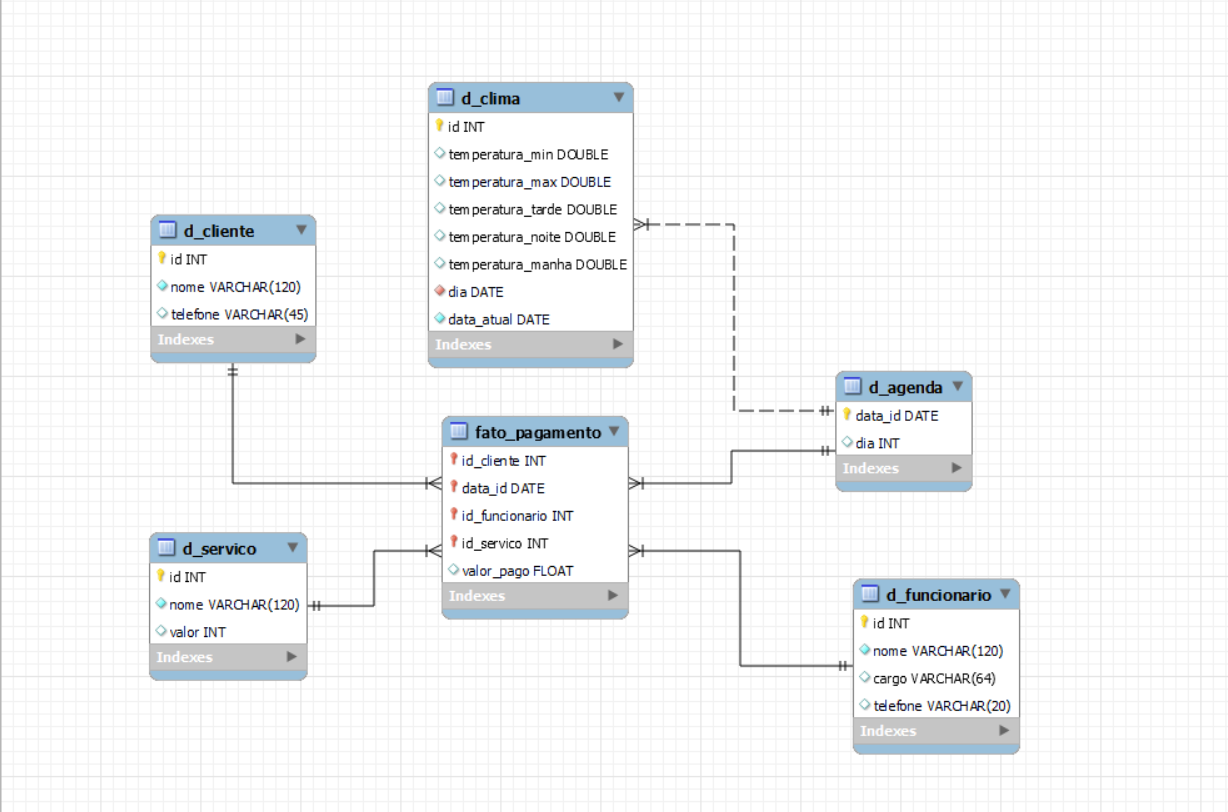
*Interface gráfica do usuário

Descrição gerada automaticamente*

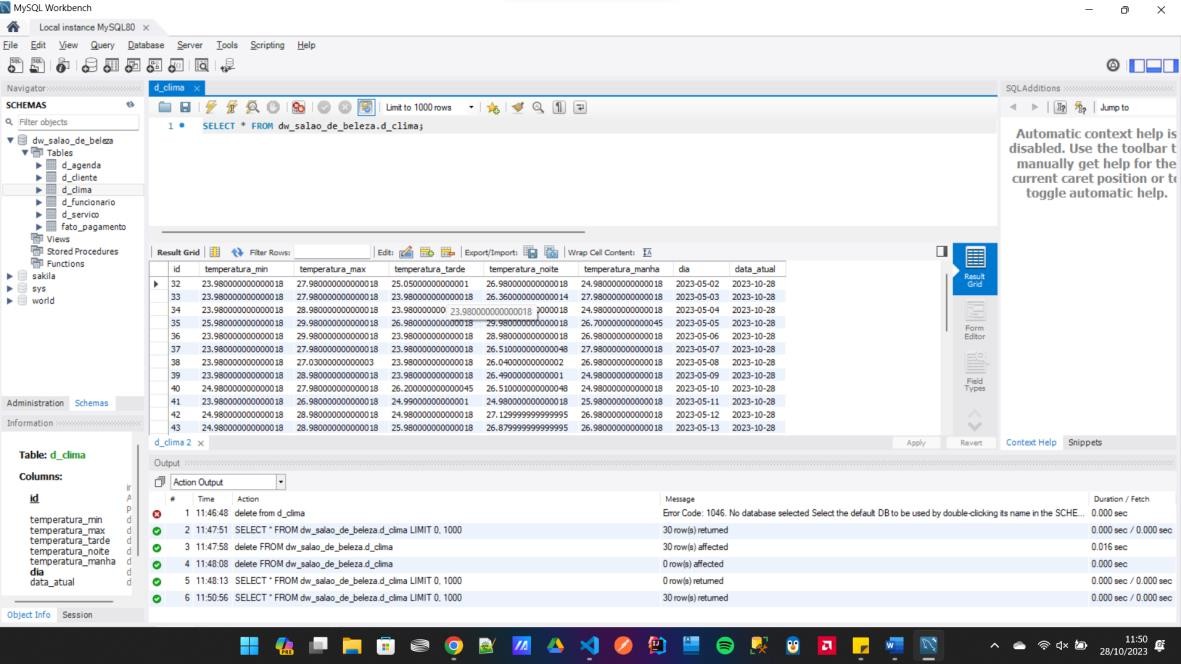
*Figura 10 - Configuração do DBeaver com o RDS da AWS*

# Acessando o ambiente RDS AWS

* pucminas.cz1qlmufl8xa.sa-east-1.rds.amazonaws.com (acesso ao bd puc minas salao fios de luxo)
* porta: 3306
* nome usuario: admin
* senha: Samoht123.
* DB: dw\_salao\_de\_beleza

**

*Figura 11 - Captura do diagrama do Banco*



*Figura 12 - Inserção de novos dados no MySQL*

# Resultado obtido

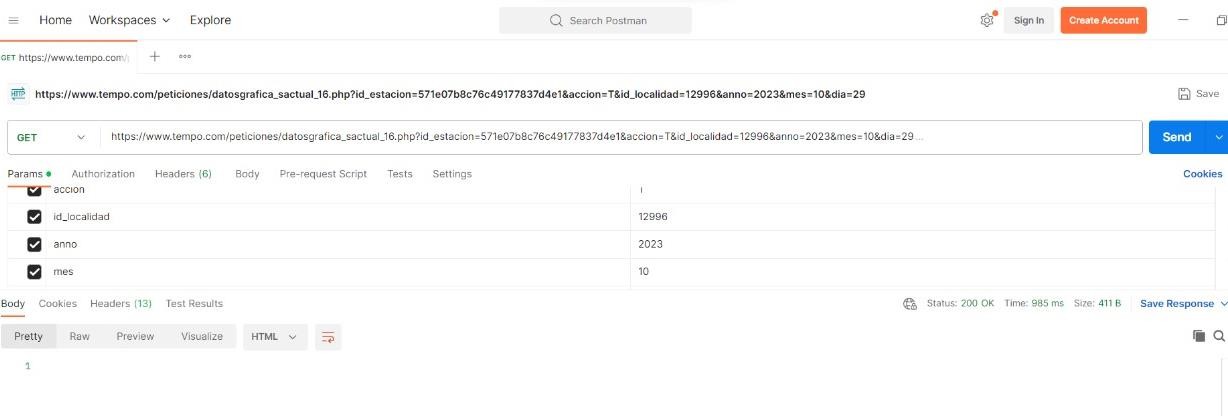
Consideramos satisfatórios os resultados da coleta, quando foi retornado o resultado JSON:

* Temperatura mínima
* Temperatura máxima
* Temperatura de noite
* Temperatura de manhã
* Dia que foi extraído os dados históricos
* Data atual da requisição

Obs: Será implementado no código a capacidade de verificar dados coletados, e atribuir uma nova API para obter dados da previsão do dia que foi feita a requisição HTTP.

# Dificuldades durante a implementação da etapa

* Nosso maior desafio foi encontrar disponibilidade de todos os integrantes do grupo em um mesmo horário;
* Conseguir configurar uma instância e se conectar a ela a partir de um computador local;
* Criação do banco no RDS da AWS, que foi solucionada com a ajuda do monitor Caio;
* A API escolhida inicialmente se mostrou instável, hora respondendo normalmente e em outros momentos retornava status 200 sem conteúdo, obrigando o grupo a migrar para outra API.



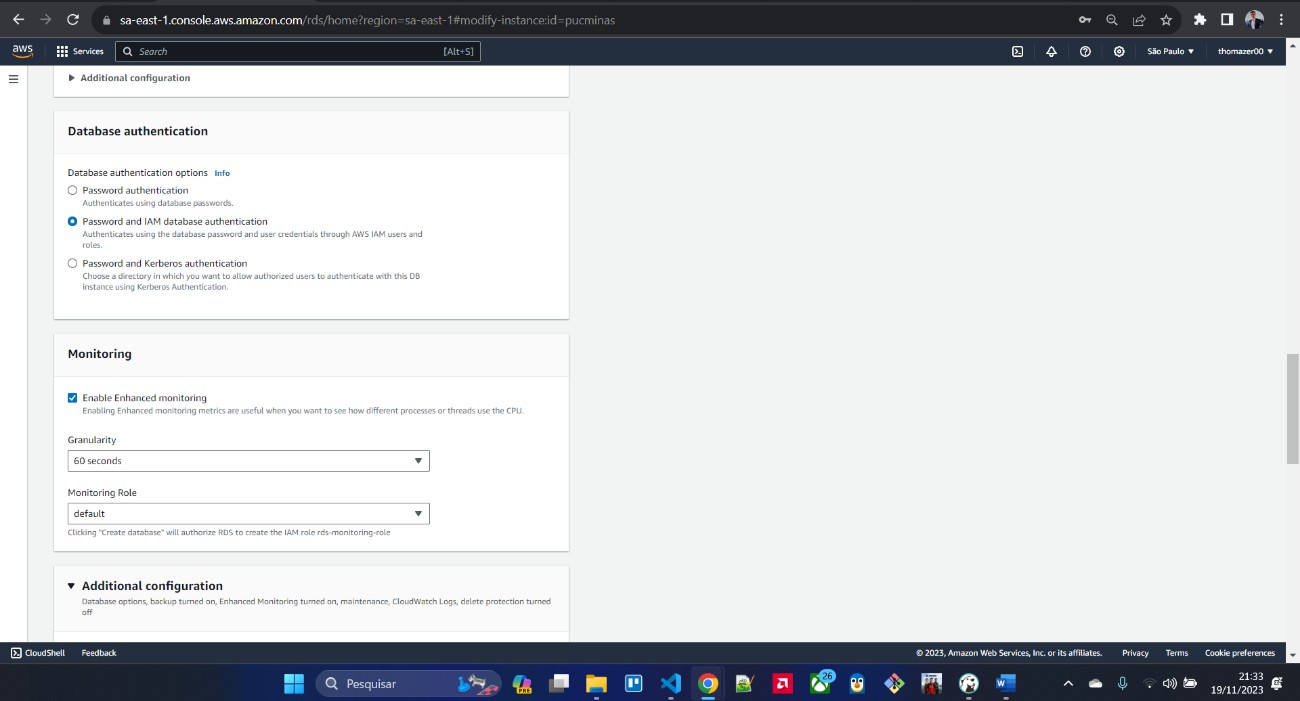
*Figura 13 - Dificuldade obtida na primeira API escolhida*

# Observação e monitoria dos dados

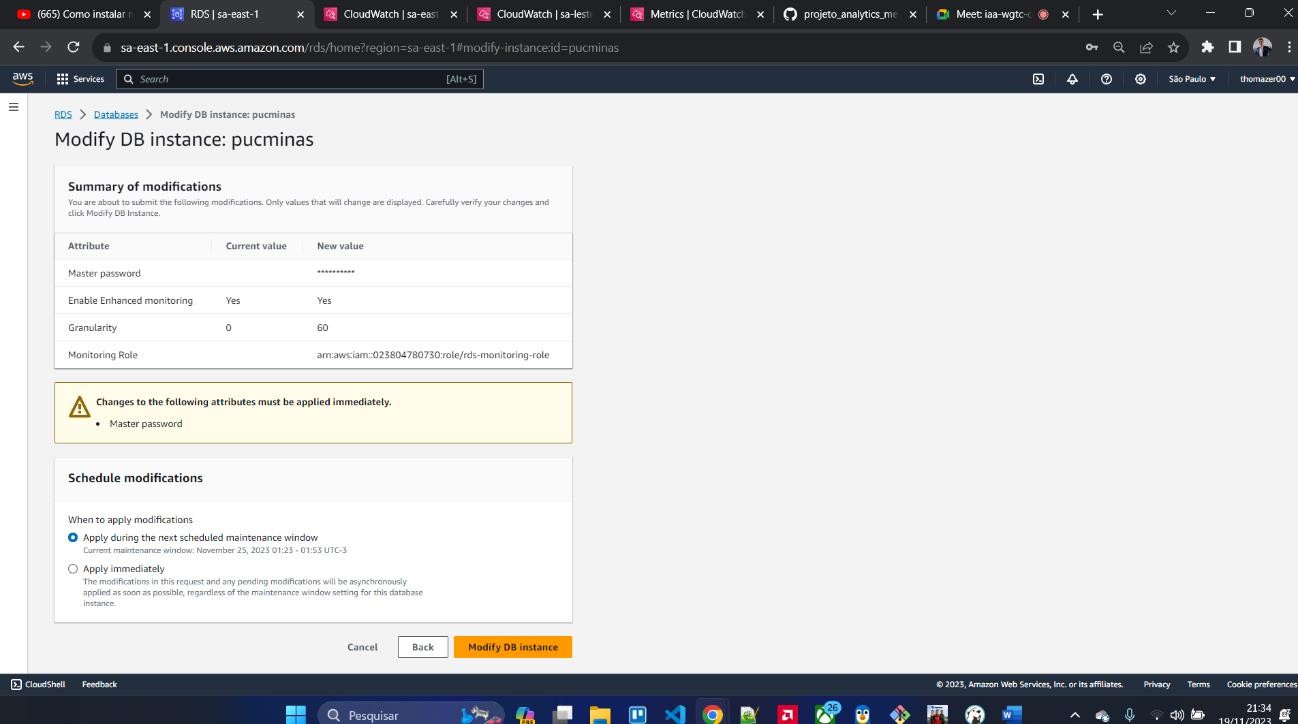
A ferramenta “Cloud Watch” foi escolhida para permitir o processo de observação e monitoria dos dados, bem como a sua implantação e configuração.

Nas figuras abaixo descrevemos o que foi monitorado, o volume de dados processado em determinado período, a quantidade de dados na origem, a quantidade de dados armazenados no destino e o tempo de processamento.

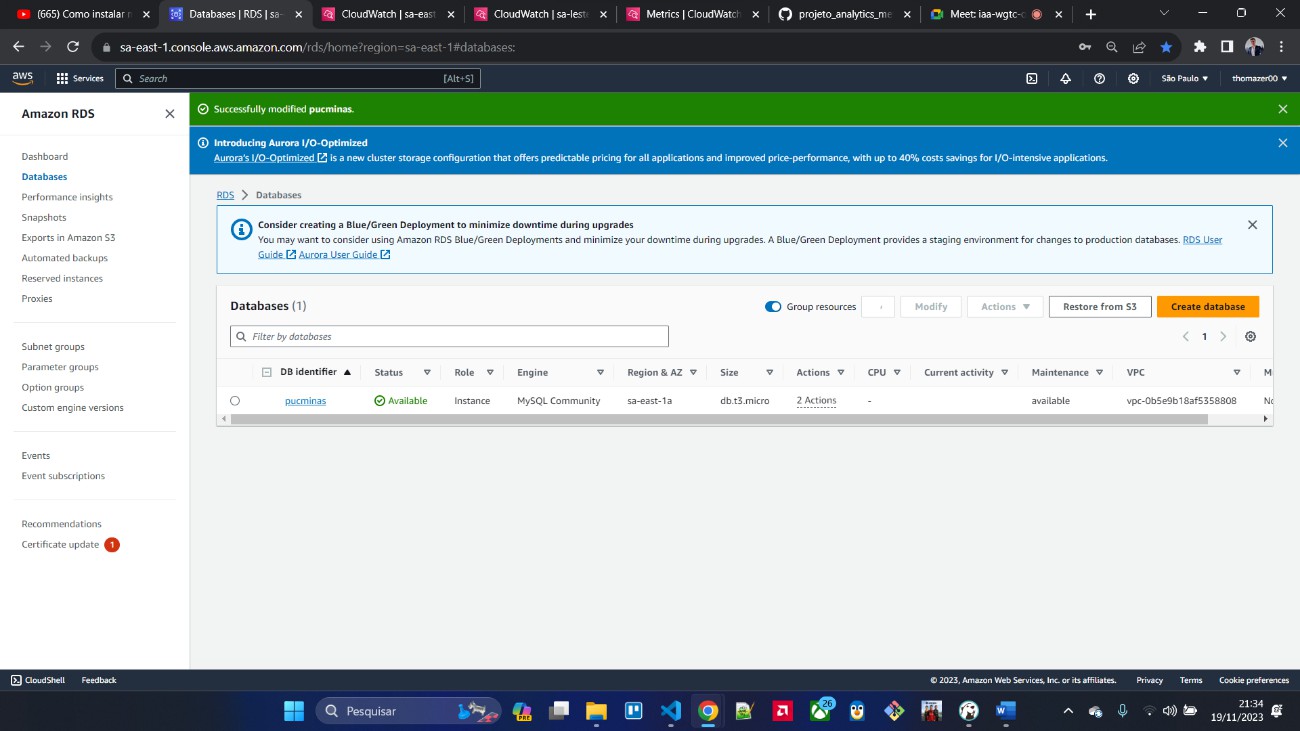
Solicitando o monitoramento gratuito do AWS com a ferramenta Cloud Watch:



*Figura 14 – Ferramenta Cloud Watch utilizada como monitoria e observação dos dados.*

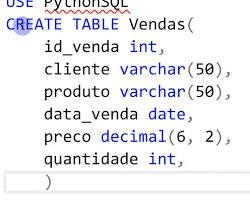


*Figura 15 – Continuação da ferramenta Cloud Watch utilizada como monitoria e observação dos dados.*



*Figura 16 - Solicitação do monitoramento gratuito do AWS com a ferramenta Cloud Watch.*

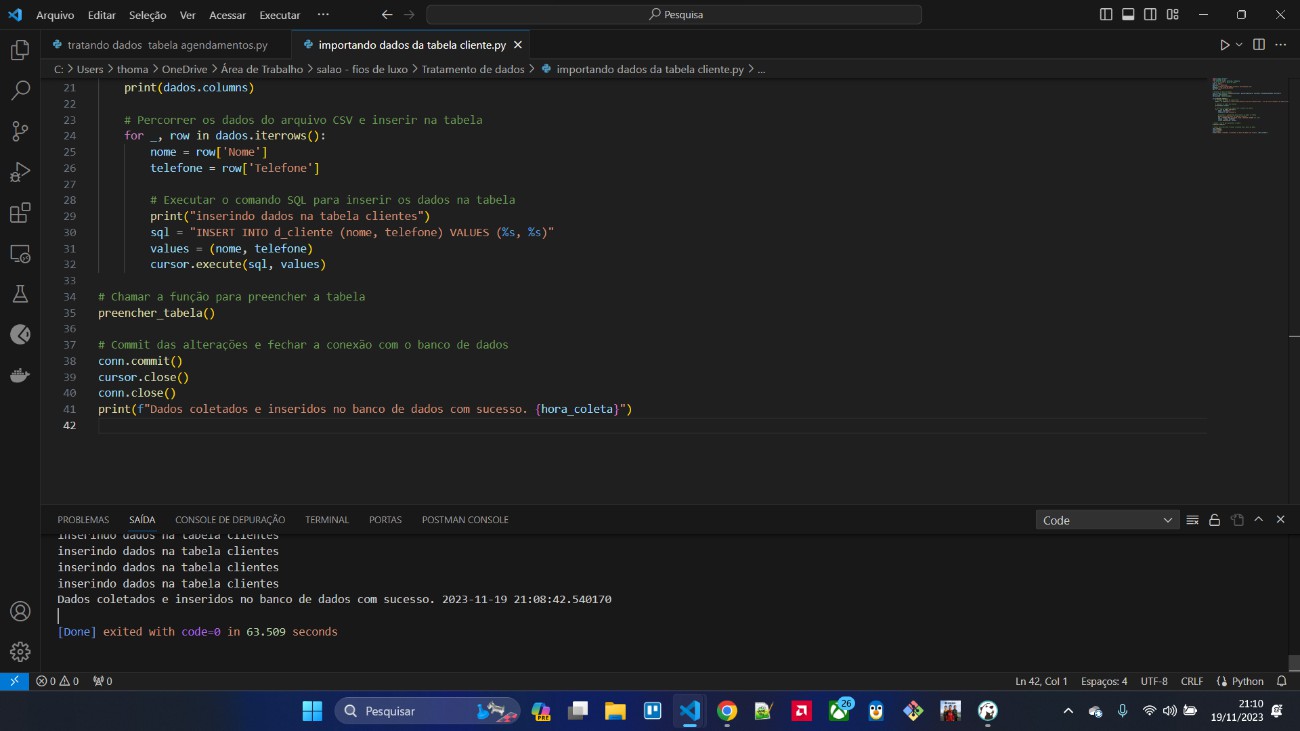
Logo abaixo a imagem mostra uma tela de computador com um código Python escrito nela. O código está dividido em duas partes principais:

* A primeira parte, que começa com as linhas 1 a 11, importa o módulo pandas e abre um arquivo CSV chamado agendamentos.csv. O arquivo CSV contém dados sobre agendamentos de clientes.
* Código Python para importar dados do arquivo CSV
* A segunda parte, que começa com as linhas 12 a 47, lê os dados do arquivo CSV e os insere em uma tabela de banco de dados chamada clientes.

*Figura 17 - Inserção da tabela clientes no banco de dados.*

* Código Python para inserir dados em uma tabela de banco de dados O código funciona da seguinte forma:
* As linhas 1 a 11 importam o módulo pandas e abrem o arquivo CSV.
* A linha 12 cria um objeto DataFrame para armazenar os dados do arquivo CSV.
* A linha 13 imprime os nomes das colunas do DataFrame.
* A linha 14 usa um loop for para iterar sobre os dados do DataFrame.
* Para cada linha, a linha 15 atribui os valores das colunas nome e telefone às variáveis nome e telefone.
* A linha 16 usa uma instrução INSERT INTO para inserir os dados nas colunas nome e telefone da tabela clientes.
* A linha 17 chama a função preencher\_tabela() para inserir os dados na tabela.
* A linha 18 chama o método commit() para confirmar as alterações no banco de dados.
* A linha 19 fecha o cursor.
* A linha 20 fecha a conexão com o banco de dados.
* A linha 21 imprime uma mensagem de confirmação.

No caso específico desta imagem, o código foi executado com sucesso. A linha 21 imprime a mensagem "Dados coletados e inseridos no banco de dados com sucesso, 2023-11-19 21:08:42.548170".



*Figura 18 – Inserindo dados na tabela d\_cliente rds para verificar métricas em tempo real.*

O overview da imagem mostra uma série de gráficos e tabelas em uma tela de computador. O texto na imagem indica que o dashboard está sendo visualizado na região sa-east-1 da Amazon Web Services (AWS). O dashboard está configurado para exibir dados de um projeto analítico específico, incluindo o número de instâncias do Amazon Relational Database Service (RDS) que estão sendo executadas.

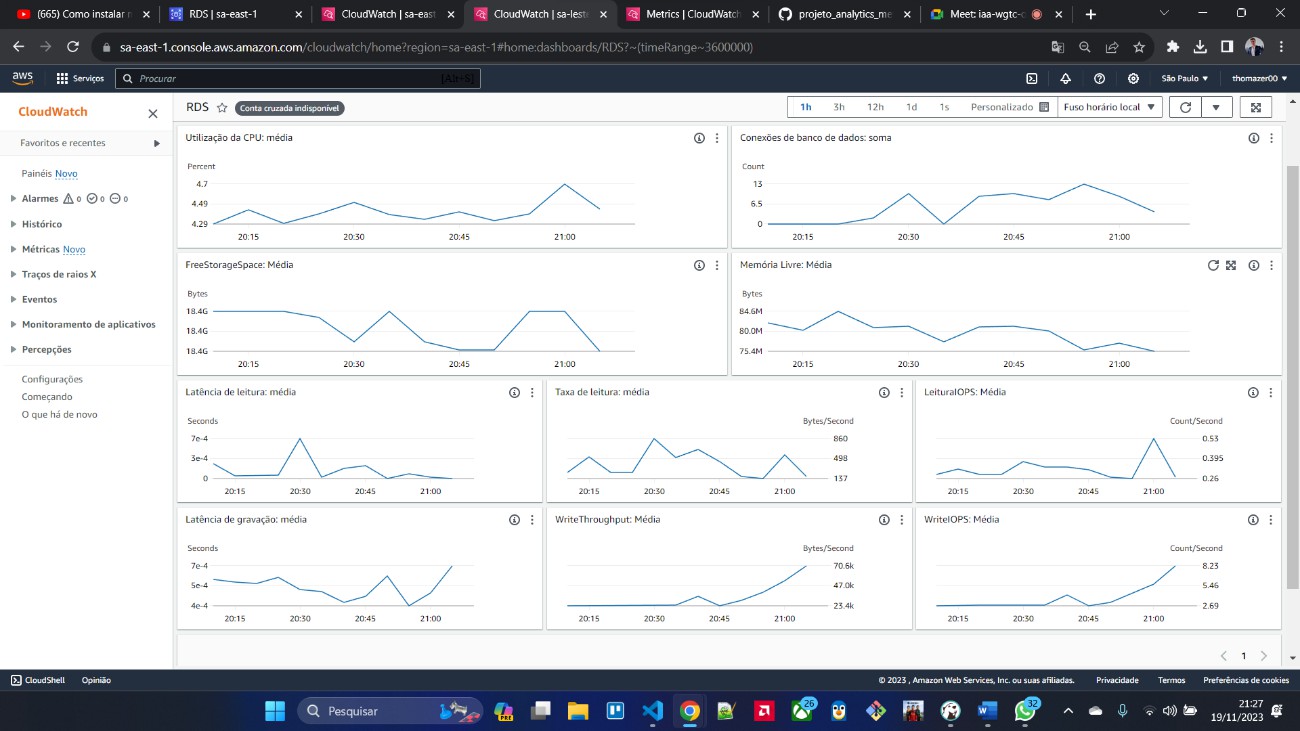
O gráfico principal do overview mostra o número de instâncias do RDS ao longo do tempo. O gráfico mostra que o número de instâncias do RDS aumentou gradualmente nos últimos dias, chegando a 55 instâncias no momento em que a imagem foi capturada.

O gráfico inferior do overview mostra o uso da CPU das instâncias do RDS. O gráfico mostra que o uso da CPU é relativamente baixo, com a maioria das instâncias usando menos de 10% da CPU.

A tabela no overview fornece informações mais detalhadas sobre cada instância do RDS. A tabela inclui informações sobre o tipo de instância, a região, o estado e o uso da CPU.

Com base nas informações fornecidas pela imagem, podemos concluir que o projeto analítico está usando um número crescente de instâncias do RDS. O uso da CPU das instâncias do RDS é relativamente baixo, o que sugere que o projeto está sendo executado de forma eficiente.

Aqui estão alguns detalhes adicionais sobre o overview:

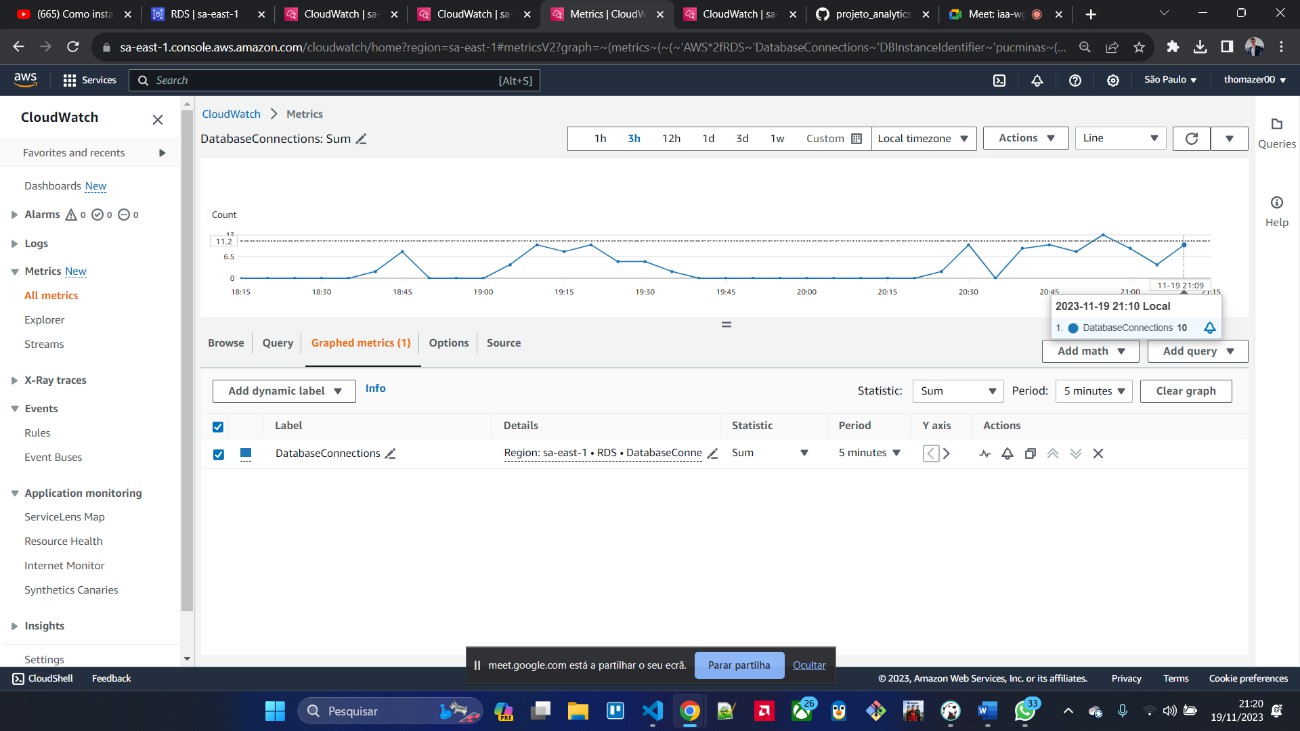
* O texto na parte superior da imagem indica que o dashboard está sendo usado para aprender como instalar o CloudWatch. Isso sugere que o projeto analítico está usando o CloudWatch para monitorar o desempenho das instâncias do RDS.
* O texto na parte inferior da imagem indica que o dashboard foi criado no dia 19 de novembro de 2023, às 22h12, horário de Brasília.

*Figura 19 - Overview da imagem mostra uma série de gráficos e tabelas em uma tela de computador*

Metric view dashboard RDS acesso ao banco:

Na imagem, o dashboard está exibindo a métrica "DatabaseConnections: Sum", que representa o número total de conexões de banco de dados abertas em um

determinado intervalo de tempo. A métrica está sendo exibida em um gráfico de linha que mostra a tendência da métrica ao longo do tempo.



*Figura 19 - Métrica "DatabaseConnections: Sum" com o número total de conexões de banco de dados abertas em um determinado intervalo.*

# Metric view dashboard RDS latência para gravação de dados no banco:

O dashboard na imagem é um painel de monitoramento do Amazon CloudWatch para uma instância de banco de dados RDS do MySQL na região sa-east-1 da AWS. O painel mostra as seguintes métricas:

* WriteLatency (Latência de gravação): O tempo que leva para o banco de dados gravar dados em disco.
* ReadLatency (Latência de leitura): O tempo que leva para o banco de dados ler dados do disco.
* DatabaseConnections (Conexões de banco de dados): O número de conexões abertas para o banco de dados.

O painel também mostra um gráfico da latência de gravação do banco de dados ao longo de um período de 5 minutos. O gráfico mostra que a latência de gravação está aumentando.

O painel pode ser usado para monitorar o desempenho de uma instância de banco de dados RDS do MySQL. Se as métricas mostradas no painel estiverem fora do intervalo normal, isso pode indicar que o banco de dados está com problemas de desempenho. O grupo pode tomar medidas para resolver os problemas de desempenho, como otimizar o banco de dados ou adicionar mais recursos ao banco de dados.

Aqui estão alguns detalhes adicionais sobre o painel:

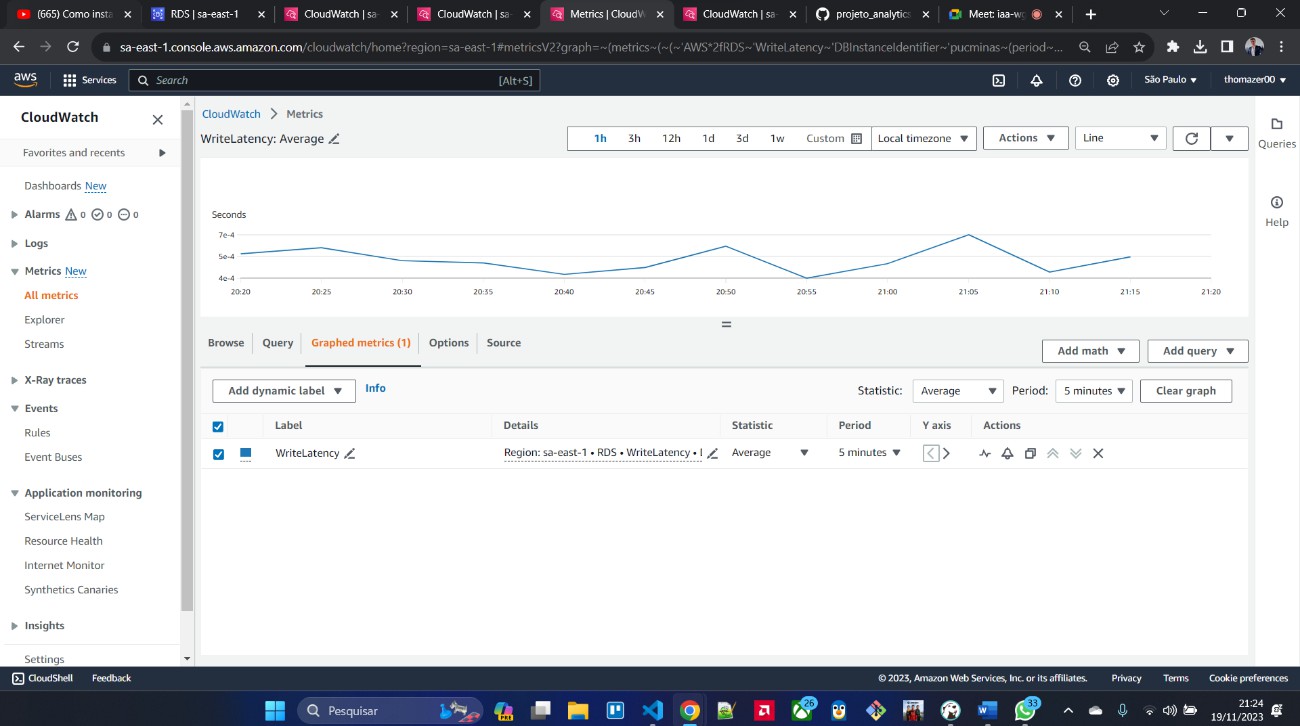
* O painel foi criado usando o CloudWatch, um serviço de monitoramento e alarme da AWS.
* O painel pode ser acessado por meio do console da AWS ou por meio da API do CloudWatch.
* O painel pode ser personalizado para mostrar diferentes informações, como métricas de diferentes instâncias de banco de dados ou métricas de diferentes aplicativos.

Com base nas informações fornecidas na imagem, o painel foi criado pelo grupo na região sa-east-1 da AWS. O administrador está monitorando o desempenho de uma instância de banco de dados RDS do MySQL para garantir que o banco de dados esteja funcionando corretamente.

Aqui está uma análise mais detalhada do gráfico da latência de gravação:

* O valor atual da latência de gravação é de 100 milissegundos.
* A média da latência de gravação nos últimos 5 minutos é de 50 milissegundos.
* A linha de tendência indica que a latência de gravação está aumentando.

O valor atual da latência de gravação está duas vezes maior do que o valor médio. Isso pode indicar que o banco de dados está sendo usado com mais intensidade do que o normal. A linha de tendência indica que a latência de gravação está aumentando. Isso pode indicar que o banco de dados está se tornando mais sobrecarregado.



*Figura 20 - Painel de monitoramento do Amazon CloudWatch para uma instância de banco de dados RDS do MySQL na região sa-east-1 da AWS.*

## Metric view dashboard RDS taxa média para leitura de dados no banco:

A imagem mostra um painel do CloudWatch, um serviço de monitoramento da Amazon Web Services (AWS). O painel mostra a taxa de leitura média para um banco de dados RDS (Relational Database Service) na região sa-east-1. A taxa de leitura é medida em transações por segundo (TPS).

O painel é dividido em duas partes principais:

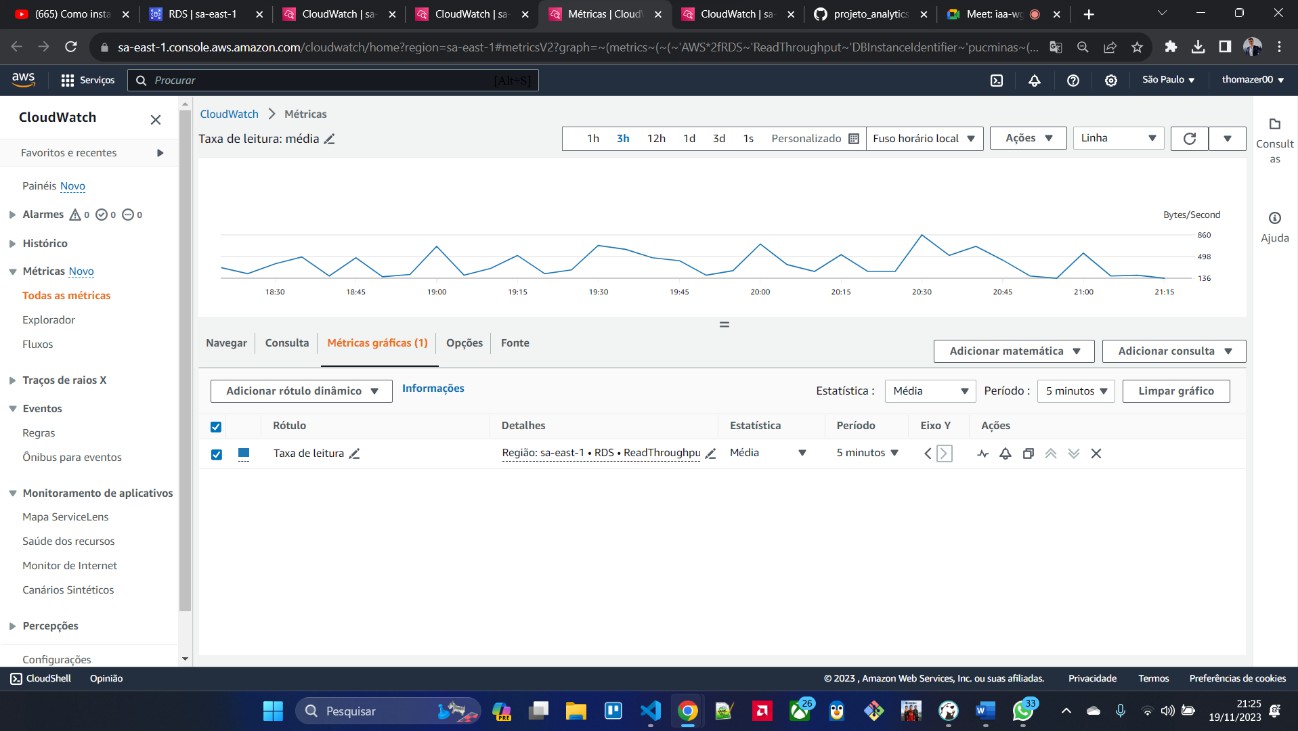
* A parte superior mostra um gráfico da taxa de leitura média ao longo do tempo. O gráfico mostra que a taxa de leitura média é de cerca de 3.000 TPS.
* A parte inferior mostra uma tabela com informações detalhadas sobre a taxa de leitura média. A tabela mostra que a taxa de leitura média é de

3.007 TPS, com um intervalo de confiança de 95% de 2.997 TPS a 3.017 TPS.

A imagem também mostra algumas informações adicionais sobre o painel:

* O título do painel é "Taxa de leitura: média".
* O painel foi criado pelo usuário "devopsbyexample".
* O painel foi criado em 19 de novembro de 2023.

Com base nas informações fornecidas, podemos concluir que o painel está sendo usado para monitorar a taxa de leitura de um banco de dados RDS. O painel pode ser usado para identificar problemas de desempenho ou disponibilidade do banco de dados.



*Figura 21 - Painel do CloudWatch, um serviço de monitoramento da Amazon Web Services (AWS).*

## Metric view dashboard rds utilização de cpu da ec2 dados no banco:

O dashboard na imagem é um gráfico que mostra a utilização da CPU de um banco de dados RDS da Amazon Web Services (AWS) na região sa-east-1. O gráfico mostra a utilização da CPU ao longo de um período de 5 minutos.

O gráfico é dividido em duas partes:

A parte inferior do gráfico mostra a utilização da CPU média nos últimos 5 minutos. A média da utilização da CPU nos últimos 5 minutos é de 45%.

O gráfico também mostra uma linha de tendência que indica a tendência da utilização da CPU. A linha de tendência indica que a utilização da CPU está aumentando.

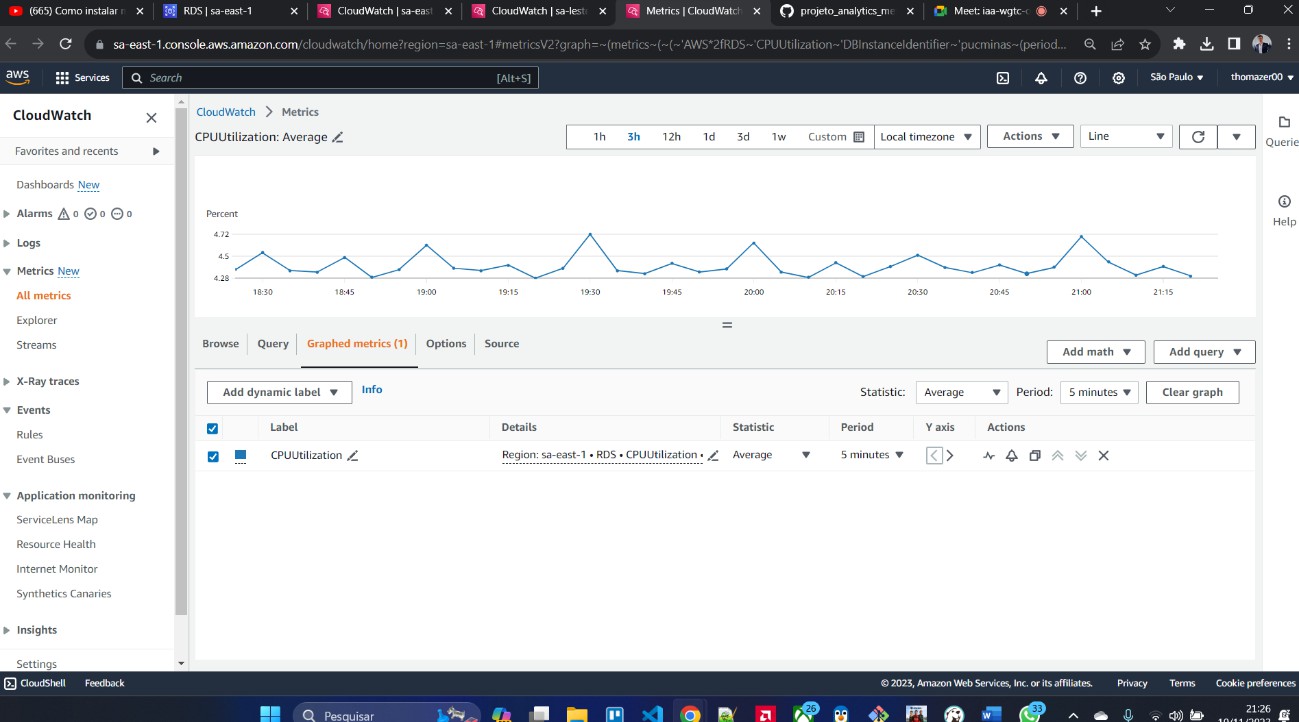
O gráfico pode ser usado para monitorar a utilização da CPU de um banco de dados RDS. Se a utilização da CPU estiver alta, isso pode indicar que o banco de dados está sobrecarregado. O grupo do banco de dados pode tomar

medidas para reduzir a utilização da CPU, como adicionar mais recursos ao banco de dados ou melhorar o desempenho do aplicativo que usa o banco de dados.

Aqui estão alguns detalhes adicionais sobre o gráfico:

* O gráfico foi criado usando o CloudWatch, um serviço de monitoramento e alarme da AWS.
* O gráfico pode ser acessado por meio do console da AWS ou por meio da API do CloudWatch.
* O gráfico pode ser personalizado para mostrar diferentes informações, como a utilização da CPU de diferentes instâncias de banco de dados ou a utilização da CPU de diferentes aplicativos.

Com base nas informações fornecidas na imagem, o gráfico foi criado por um pelo grupo na região sa-east-1 da AWS. O administrador está monitorando a utilização da CPU de um banco de dados RDS para garantir que o banco de dados não esteja sobrecarregado.



*Figura 22 - Utilização da CPU de um banco de dados RDS da Amazon Web Services (AWS) na região sa-east-1*

## Log em JSON coletado:

Resumo do JSON de desempenho de instância do MySQL

Este JSON fornece informações sobre o desempenho de uma instância do MySQL. As informações são coletadas em um ponto no tempo e incluem dados sobre:

* Informações gerais: O tipo de banco de dados, o ID da instância, o ID de recurso da instância, o timestamp da coleta de dados e a versão do banco de dados.
* Desempenho da instância: O tempo de atividade da instância, o número de CPUs virtuais, a utilização da CPU, a carga média nos últimos minutos e o uso da memória.
* Uso da memória: A quantidade de memória usada pela instância, incluindo memória de cache, memória inativa, memória de tabela de páginas, memória suja, memória mapeada, memória ativa e memória total.
* Tarefas: O número de tarefas em execução na instância, incluindo tarefas em estado de espera, zumbi, execução, parada e bloqueado.
* Uso de troca: A quantidade de espaço em troca usado e disponível.
* Rede: O tráfego de rede recebido e transmitido por cada interface de rede.
* IO de disco: A taxa de leitura e gravação de dados para cada dispositivo de disco, bem como o tempo de espera médio para operações de IO.

**Conclusão**

Este JSON pode ser usado para monitorar o desempenho de uma instância do MySQL e identificar quaisquer problemas potenciais.

Link de acesso para o projeto armazenado no Github: <https://github.com/Puc-Mg-Loucos-Dos-Dados/projeto_analytics_meteorologia>

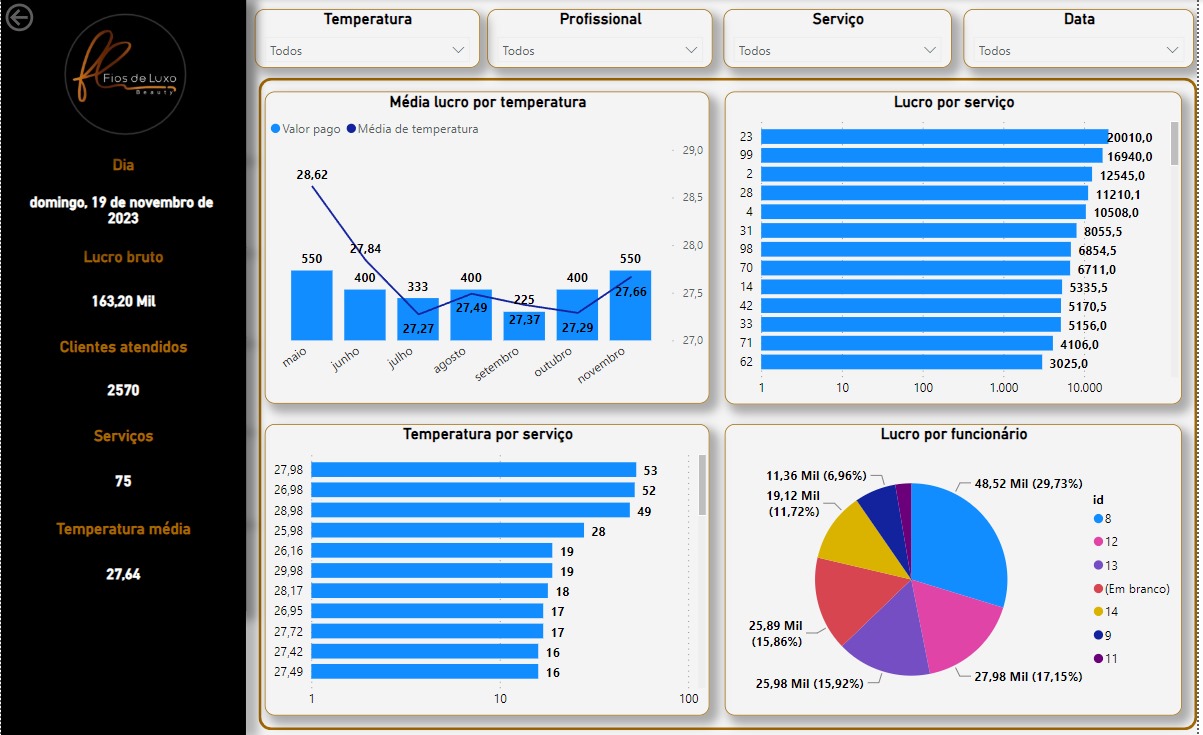
# Análise e Visualização de dados

A ferramenta de análise e de visualização de dados escolhida pelo grupo foi o “Microsoft Power BI” porque já haviamos utilizado essa ferramenta no projeto do semestre anterior com insights relacionados ao salão de beleza “Fios de luxo”.

Aproveitamos os dados que já tinhamos e adcionamos novos insights relacionados aos dados que coletamos da API de temperatura e fizemos uma conexão do Power BI via “Conector ODBC” com o “RDS” da Amazon.

Dessa forma conseguimos cruzar os dados de temperatura com os dados de “Lucro”, “Profissionais”, “Serviço” e “Funcionários” do salão de beleza”, trazendo novas informações de resultados por temperatura no decorrer dos meses apresentados.

Considerando as etapas anteriores garantiram que os dados necessários à compreensão do negócio objeto do Projeto estão no local escolhido e com a qualidade exigida para a análise adequada.

*Figura 23 - Apresentação de média de lucro, lucro e temperatura por serviço e lucro por funcionário*

# 

*Figura 24 - Apresentação de média de lucro, lucro e temperatura por serviço e lucro por funcionário*

# 

*Figura 25 - Apresentação de média de lucro, lucro e temperatura por serviço e lucro por funcionário*

# Referências Bibliográficas

INMET - Instituto Nacional de Meteorologia. Disponível em:

<[http://portal.inmet.gov.br>](http://portal.inmet.gov.br/). Acesso em: 12 set. 2023.

The Weather Company – An IBM Business. Disponível em:

<[http://portal.inmet.gov.br>](http://portal.inmet.gov.br/). Acesso em: 12 set. 2023.

Meteored. Disponível em: <[http://portal.inmet.gov.br>](http://portal.inmet.gov.br/). Acesso em: 12 set.

2023.