**Documentação e Manual do Usuário para Scripts de Monitoramento e Análise**

Este manual cobre a utilização dos seguintes scripts Python:

1. monitor\_space.py
2. coletor\_logs.py
3. latency\_monitor.py
4. plot\_latency\_data.py
5. grafico\_uso.py
6. SQL.py
7. .BAT

**1. monitor\_space.py (Monitoramento de Espaço em Disco via SSH)**

**Visão Geral**

Este script conecta-se a um servidor remoto via SSH, executa o comando df -m no diretório /srv/data\_pool para verificar o uso de disco e salva a porcentagem de uso em um arquivo CSV.

**Pré-requisitos**

* **Python 3**
* **paramiko**: Biblioteca Python para SSH.
* **Chave SSH privada**: Para autenticação sem senha no servidor remoto. A chave deve estar no formato OpenSSH (.pem). Se você tem uma chave .ppk do PuTTY, use puttygen.exe para convertê-la (Conversions > Export OpenSSH key).
* **Acesso SSH**: O usuário especificado deve ter permissão para conectar via SSH e executar o comando df -m /srv/data\_pool no servidor remoto.

**Como Instalar/Configurar**

1. **Instalar paramiko**:

Bash

pip install paramiko

1. **Preparar a Chave SSH**:
   * Garanta que sua chave SSH (.pem) tenha as permissões corretas (somente leitura para o proprietário). Em sistemas Linux/macOS, use: chmod 400 /caminho/para/sua/chave.pem
   * No Windows, verifique as permissões de arquivo manualmente para garantir que apenas sua conta tenha acesso.

**Como Usar**

Execute o script pela linha de comando, fornecendo os detalhes do servidor como argumentos:

Bash

python monitor\_space.py \

--servidor\_nome "NomeDoServidor" \

--servidor\_ip "192.168.1.100" \

--usuario "seu\_usuario\_ssh" \

--chave\_ssh "/caminho/para/sua/chave.pem" \

--porta\_ssh 2222

**Parâmetros:**

* --servidor\_nome (obrigatório): Um nome amigável para o servidor (usado no CSV).
* --servidor\_ip (obrigatório): O endereço IP ou hostname do servidor.
* --usuario (obrigatório): O nome de usuário SSH para login.
* --chave\_ssh (obrigatório): O caminho completo para o seu arquivo de chave privada SSH (.pem).
* --porta\_ssh (opcional): A porta SSH. Padrão: 2222.

**Saída/Resultados**

* **Console**: Mensagens de status da conexão e execução do comando.
* **Arquivo CSV**: Um arquivo CSV chamado [servidor\_nome]\_uso\_disco.csv (ex: Desenvolvimento\_uso\_disco.csv) será criado (ou atualizado) no mesmo diretório do script. Ele conterá a data e hora da coleta, o nome do servidor e a porcentagem de uso do disco.
* A white screen with a black border

  AI-generated content may be incorrect.

**Solução de Problemas**

* **paramiko.AuthenticationException**: Usuário ou chave SSH incorretos. Verifique --usuario e --chave\_ssh.
* **paramiko.SSHException**: Problemas na conexão SSH (ex: servidor não disponível, firewall bloqueando a porta). Verifique o IP, porta e conectividade.
* **FileNotFoundError: Arquivo da chave SSH '...' não encontrado**: O caminho para --chave\_ssh está incorreto ou o arquivo não existe.
* **Nenhuma saída do comando 'df -m ...'**: O comando remoto não retornou dados. Verifique se o caminho /srv/data\_pool está correto no servidor e se o usuário tem permissão para acessá-lo.
* **Erro ao executar o comando: permissão negada**: O usuário SSH não tem permissão para executar df ou acessar o diretório.

**2. coletor\_logs.py (Coleta de Logs e Comandos via SSH)**

**Visão Geral**

Este script conecta-se a um servidor remoto via SSH, executa uma série de comandos pré-definidos e filtra logs específicos (buscando por "ERROR", "SEVERE", etc.) para a data atual. Os resultados são compilados em um arquivo HTML formatado, que é automaticamente aberto no navegador.

**Pré-requisitos**

* **Python 3**
* **paramiko**: Biblioteca Python para SSH.
* **webbrowser**: Módulo nativo do Python (não requer instalação adicional).
* **Chave SSH privada**: No formato OpenSSH (.pem).
* **Acesso SSH**: O usuário especificado deve ter permissão para conectar via SSH e executar os comandos e ler os arquivos de log definidos no script.

**Como Instalar/Configurar**

1. **Instalar paramiko**:

Bash

pip install paramiko

1. **Preparar a Chave SSH**: Siga as mesmas instruções de monitor\_space.py.

**Como Usar**

Execute o script pela linha de comando, fornecendo os detalhes do servidor como argumentos:

Bash

python coletor\_logs.py \

--servidor\_nome "NomeDoServidor" \

--servidor\_ip "192.168.1.100" \

--usuario "seu\_usuario\_ssh" \

--chave\_ssh "/caminho/para/sua/chave.pem" \

--porta\_ssh 2222

**Parâmetros:**

* --servidor\_nome (obrigatório): Um nome amigável para o servidor (usado no título do HTML).
* --servidor\_ip (obrigatório): O endereço IP ou hostname do servidor.
* --usuario (obrigatório): O nome de usuário SSH para login.
* --chave\_ssh (obrigatório): O caminho completo para o seu arquivo de chave privada SSH (.pem).
* --porta\_ssh (opcional): A porta SSH. Padrão: 2222.

**Saída/Resultados**

* **Console**: Mensagens de status da conexão e execução dos comandos.
* **Arquivo HTML**: Um arquivo HTML chamado [servidor\_nome]\_logs\_[timestamp].html (ex: Desenvolvimento\_logs\_2025-06-02\_11-30-00.html) será gerado no mesmo diretório do script. Este arquivo contém os resultados dos comandos executados e os logs de erro filtrados.
* **Navegador Web**: O arquivo HTML gerado será automaticamente aberto no seu navegador padrão.
* A screenshot of a computer

  AI-generated content may be incorrect.
* A screenshot of a computer

  AI-generated content may be incorrect.
* A screenshot of a computer program

  AI-generated content may be incorrect.

**Solução de Problemas**

* **Erros de Conexão/Autenticação**: Siga as mesmas dicas de monitor\_space.py.
* **Erro ao executar comando '...'**: O comando remoto não pode ser executado. Verifique a sintaxe do comando, o caminho, e as permissões do usuário SSH no servidor remoto.
* **Nenhum log de erro filtrado encontrado para a data de hoje**: Não foram encontradas linhas que correspondam aos filtros de grep para a data atual nos arquivos de log especificados. Isso pode ser normal se não houver erros recentes. Verifique os caminhos dos logs no script.
* **HTML vazio ou incompleto**: Pode indicar que os comandos remotos falharam em produzir saída ou que as permissões de leitura dos arquivos de log são insuficientes.

**latency\_monitor.py (Processamento de Logs HTML para CSV)**

* **Visão Geral:** Este script foi aprimorado para escanear o diretório atual em busca de **todos os arquivos HTML que seguem o padrão de nomenclatura \*\_logs\_\*.html**. Para cada arquivo encontrado, ele extrai informações detalhadas de latência de replicação (como data, hora, latência em segundos, operações aplicadas de insert/delete, etc.) contidas em um bloco <pre> específico (normalmente do hardwareEventLog.bin). As informações extraídas são então organizadas e salvas em um novo arquivo CSV correspondente, nomeado como [NomeDoServidor]\_latency.csv, onde [NomeDoServidor] é derivado do nome do arquivo HTML de log ou do título HTML.
* **Pré-requisitos:**
  + Python 3
  + pandas: Para manipulação e salvamento de dados em CSV.
  + beautifulsoup4: Para análise e extração de dados do HTML.
  + lxml (ou html5lib): Um parser para Beautiful Soup.
* **Como Instalar/Configurar:**
  + Instalar bibliotecas: pip install pandas beautifulsoup4 lxml
  + Colocar os arquivos \*\_logs\_\*.html (gerados por outras ferramentas ou scripts, como o coletor\_logs.py se ele gera esse formato) no mesmo diretório do script latency\_monitor.py.
* **Como Usar:**
  + Execute o script diretamente, sem argumentos:

Bash

python latency\_monitor.py

* + O script automaticamente detectará e processará todos os arquivos \*\_logs\_\*.html presentes no diretório.
* **Saída/Resultados:**
  + **Console**: Mensagens de status indicando quais arquivos HTML estão sendo processados, quaisquer erros ou avisos, e a confirmação da criação de cada arquivo CSV.
  + **Arquivos CSV**: Para cada arquivo HTML de log processado ([NomeDoServidor]\_logs\_\*.html), um arquivo CSV correspondente será gerado no mesmo diretório, nomeado como [NomeDoServidor]\_latency.csv. Esses arquivos CSV contêm as colunas: Date, Time, Subscription, Latency\_Seconds, Latest\_Commit\_RBA\_LRSN, Open\_Transactions, Earliest\_Open\_RBA\_LRSN, Parsed\_Insert, Parsed\_Update, Parsed\_Delete, Applied\_Insert, Applied\_Delete, Host\_Name.
  + A screenshot of a computer

    AI-generated content may be incorrect.

**plot\_latency\_data.py (Análise e Geração de Gráficos de Latência)**

* **Visão Geral:** Este script foi aprimorado para analisar dados de latência de replicação de arquivos CSV gerados (normalmente pelo latency\_monitor.py), gerar gráficos visuais e emitir alertas. Ele agora processa dados filtrados para os **últimos 10 dias** a partir da data de execução do script. Para cada servidor e cada assinatura detectada nos dados, ele gera **gráficos de latência e de operações aplicadas separados**. Finalmente, um **relatório HTML completo** é criado para cada servidor, incorporando todos os seus gráficos e alertas relevantes, e é **aberto automaticamente** no navegador padrão.
* **Pré-requisitos:**
  + Python 3
  + pandas: Para manipulação de dados.
  + matplotlib: Para a geração de gráficos.
  + webbrowser: Módulo padrão do Python para abrir URLs em navegadores.
  + **Dados de Entrada**: Arquivos CSV no formato [NomeDoServidor]\_latency.csv (ou [NomeDoServidor]\_detailed\_latency\_data.csv). Estes arquivos são gerados tipicamente pelo script latency\_monitor.py.
* **Como Instalar/Configurar:**
  + Instalar bibliotecas: pip install pandas matplotlib
  + Certifique-se de que os arquivos \*\_latency.csv (ou \*\_detailed\_latency\_data.csv) que você deseja analisar estejam no mesmo diretório do script plot\_latency\_data.py.
* **Como Usar:**
  + Execute o script diretamente, sem argumentos de linha de comando:

Bash

python plot\_latency\_data.py

* + O script irá procurar por todos os arquivos \*\_latency.csv (renomeando-os temporariamente para \*\_detailed\_latency\_data.csv para processamento interno e depois restaurando o nome original) no diretório, processar cada um individualmente, gerar os gráficos PNG e o relatório HTML correspondente, e abrir cada relatório HTML em uma nova aba do seu navegador.
* **Saída/Resultados:**
  + **Console**: Mensagens de status sobre o carregamento de dados, a geração de alertas (se a latência exceder o limite de 3600 segundos) e a confirmação da criação e abertura de cada arquivo de gráfico PNG e HTML.
  + **Arquivos PNG**: Para cada servidor e cada assinatura detectada nos dados ([NomeDaAssinatura]), dois arquivos PNG serão gerados no diretório, contendo apenas dados dos últimos 10 dias:
    - [NomeDoServidor]\_[NomeDaAssinatura]\_latency\_graph\_last\_10\_days.png: Gráfico de latência.
    - [NomeDoServidor]\_[NomeDaAssinatura]\_applied\_operations\_graph\_last\_10\_days.png: Gráfico de operações aplicadas (inserções e exclusões).
  + **Arquivos HTML**: Para cada servidor processado, um arquivo HTML será gerado (ex: [NomeDoServidor]\_relatorio\_latencia\_operacoes.html). Este arquivo servirá como um relatório completo, incorporando todos os gráficos PNG relevantes e listando os alertas de latência para os últimos 10 dias.
  + **Navegador Web**: Cada arquivo HTML gerado será automaticamente aberto em uma nova aba do seu navegador padrão.
  + A screenshot of a computer

    AI-generated content may be incorrect.  
    A graph on a white background

    AI-generated content may be incorrect.

**5. grafico\_uso.py (Geração de Gráfico de Uso de Disco)**

**Visão Geral**

Este script lê dados de uso de disco de um arquivo CSV (gerado pelo monitor\_space.py), processa esses dados e gera um gráfico de linha que visualiza o uso de disco ao longo do tempo para um servidor específico.

**Pré-requisitos**

* **Python 3**
* **pandas**: Biblioteca para manipulação de dados.
* **matplotlib**: Biblioteca para criação de gráficos.
* **Arquivo CSV de uso de disco**: Gerado pelo script monitor\_space.py, no formato [servidor\_nome]\_uso\_disco.csv.

**Como Instalar/Configurar**

1. **Instalar bibliotecas**:

Bash

pip install pandas matplotlib

1. **Garantir o CSV**: Certifique-se de ter executado o script monitor\_space.py para o servidor desejado e que o arquivo CSV correspondente esteja no mesmo diretório de grafico\_uso.py.

**Como Usar**

Execute o script pela linha de comando, fornecendo o nome do servidor cujo gráfico você deseja gerar:

Bash

python grafico\_uso.py \

--servidor\_nome "NomeDoServidor"

**Parâmetros:**

* --servidor\_nome (obrigatório): O nome do servidor para o qual o arquivo CSV existe e cujo gráfico você deseja gerar. Deve corresponder ao nome usado no monitor\_space.py.

**Saída/Resultados**

* **Console**: Mensagens de status do carregamento e geração do gráfico.
* **Arquivo PNG**: Um arquivo de imagem PNG chamado [servidor\_nome]\_uso\_disco.png (ex: Desenvolvimento\_uso\_disco.png) será gerado no mesmo diretório do script, contendo o gráfico de uso de disco.
* A screenshot of a computer screen

  AI-generated content may be incorrect.

**Solução de Problemas**

* **Erro: Arquivo CSV '...' não encontrado**: O script não encontrou o arquivo CSV para o servidor\_nome especificado. Verifique se o nome está correto e se o CSV está no diretório certo.
* **Erro: O arquivo CSV '...' está vazio ou mal formatado**: O CSV existe, mas está vazio ou suas colunas não estão no formato esperado (Data, Servidor, Uso%). Verifique o conteúdo do CSV.
* **Não há dados válidos de uso de disco para '...' após o pré-processamento**: O script conseguiu ler o CSV, mas os dados de "Uso%" não puderam ser convertidos para números, ou há linhas com dados inválidos. Verifique a coluna "Uso%" no CSV.
* **Nenhum dado encontrado para o servidor '...' no arquivo '...'**: O CSV contém dados, mas nenhuma linha corresponde ao servidor\_nome especificado no argumento.

**6. SQL.py (Execução de Consultas DB2 e Geração de Relatório HTML)**

**Visão Geral**

Este script conecta-se a um banco de dados DB2 usando JDBC (via jaydebeapi), executa uma série de consultas SQL pré-definidas e gera um relatório HTML com os resultados de cada consulta. O relatório HTML é automaticamente aberto no navegador.

**Pré-requisitos**

* **Python 3**
* **Java Runtime Environment (JRE)** ou **Java Development Kit (JDK)**: jaydebeapi precisa de uma JVM para funcionar.
* **jaydebeapi**: Biblioteca Python para conectar a bancos de dados via JDBC.
* **Driver JDBC DB2**: Um arquivo .jar (ex: db2jcc4.jar) necessário para a conexão com o DB2. Você precisará obtê-lo do seu ambiente DB2 ou do site da IBM.
* **Credenciais de acesso ao DB2**: Usuário e senha com permissão para executar as consultas.

**Como Instalar/Configurar**

1. **Instalar jaydebeapi**:

Bash

pip install jaydebeapi

1. **Configurar Java**: Certifique-se de que o Java está instalado e configurado corretamente no seu sistema (a variável de ambiente JAVA\_HOME pode ser necessária em alguns casos).
2. **Obter o Driver JDBC**: Coloque o arquivo db2jcc4.jar (ou o nome do seu driver) em um local acessível no seu sistema, e forneça o caminho completo para ele.

**Como Usar**

Execute o script pela linha de comando, fornecendo os detalhes da conexão JDBC:

Bash

python SQL.py \

--driver\_path "/caminho/para/db2jcc4.jar" \

--jdbc\_url "jdbc:db2://seu\_host\_db2:50100/SEU\_DB" \

--db\_user "seu\_usuario\_db2" \

--db\_pass "sua\_senha\_db2" \

--jdbc\_driver "com.ibm.db2.jcc.DB2Driver"

**Parâmetros:**

* --driver\_path (obrigatório): O caminho completo para o arquivo .jar do driver JDBC (ex: db2jcc4.jar).
* --jdbc\_url (obrigatório): A URL de conexão JDBC para o seu banco de dados DB2.
* --db\_user (obrigatório): O nome de usuário do banco de dados DB2.
* --db\_pass (obrigatório): A senha do usuário do banco de dados DB2.
* --jdbc\_driver (obrigatório): A classe do driver JDBC (para DB2, geralmente com.ibm.db2.jcc.DB2Driver).

**Saída/Resultados**

* **Console**: Mensagens de status da execução e sucesso.
* **Arquivo HTML**: Um arquivo HTML chamado SQL\_RESULTADO.html será gerado no mesmo diretório do script. Este arquivo contém cada consulta SQL executada e a tabela de resultados correspondente.
* **Navegador Web**: O arquivo HTML gerado será automaticamente aberto no seu navegador padrão.

**Solução de Problemas**

* **Erro ao executar consulta: no JDBC driver jar file found**: O driver\_path está incorreto ou o arquivo .jar não existe.
* **Erro ao executar consulta: ClassNotFoundException: com.ibm.db2.jcc.DB2Driver**: O nome da classe do driver JDBC ( jdbc\_driver ) está incorreto ou o driver .jar não está no driver\_path fornecido.
* **Erro ao executar consulta: CommunicationsException ou similar**: Problemas de rede ou firewall impedindo a conexão com o banco de dados. Verifique a jdbc\_url (host, porta, nome do banco) e a conectividade.
* **Erro ao executar consulta: SQLCODE -...**: Erro de SQL. A consulta pode estar incorreta ou o usuário não tem permissão para executá-la ou acessar as tabelas. Verifique a sintaxe SQL e as permissões do usuário db\_user.
* **jaydebeapi.DatabaseError: Authentication failed**: As credenciais do banco de dados ( db\_user , db\_pass ) estão incorretas.
* **jaydebeapi.Error: JVM not found**: O Python não conseguiu encontrar uma instalação Java. Verifique se o JRE/JDK está instalado e configurado no PATH do sistema.
* A screenshot of a computer

  AI-generated content may be incorrect.
* A screenshot of a computer

  AI-generated content may be incorrect.
* A screenshot of a computer

  AI-generated content may be incorrect.

**7. BAT (Execução em todos os ambientes)**

@echo off

REM Navega até a pasta onde os scripts estão localizados (se necessário)

cd /d "C:\Users\c1311101\scripts\build\exe.win-amd64-3.11"

@echo off

setlocal

:: =========================================================

:: Script para criar um diretorio com a data atual e mover

:: todos os arquivos que correspondem a \*\_logs\_\*.html para la.

:: =========================================================

echo.

echo Iniciando a organizacao dos arquivos de log HTML...

:: Obter a data atual no formato YYYY-MM-DD de forma consistente

:: (funciona independentemente das configuracoes regionais do Windows)

for /f "skip=1 tokens=1-6" %%a in ('wmic path Win32\_LocalTime Get Day^,Hour^,Minute^,Month^,Second^,Year /value') do (

if "%%a" NEQ "" (

set %%a

)

)

:: Formatar Mes e Dia com zero a esquerda se necessario

if "%Month:~0,1%"==" " set "Month=0%Month:~1%"

if "%Day:~0,1%"==" " set "Day=0%Day:~1%"

:: Construir a string da data no formato YYYY-MM-DD

set "CURRENT\_DATE\_FOLDER=%Year%-%Month%-%Day%"

:: Definir o nome do diretório de destino

set "TARGET\_DIR=%CURRENT\_DATE\_FOLDER%\_logs\_backup"

echo.

echo Criando diretorio: "%TARGET\_DIR%"

mkdir "%TARGET\_DIR%"

:: Verificar se o diretorio foi criado com sucesso

if not exist "%TARGET\_DIR%" (

echo.

echo Erro: Nao foi possivel criar o diretorio "%TARGET\_DIR%".

echo Certifique-se de ter permissoes de escrita no diretorio atual.

pause

exit /b 1

)

echo.

echo Movendo arquivos \*\_logs\_\*.html para "%TARGET\_DIR%"...

:: O /Y suprime o prompt para confirmar a sobrescrita de arquivos existentes.

:: Remova o /Y se voce quiser ser perguntado.

move /Y \*\_logs\_\*.html "%TARGET\_DIR%\"

move /Y \*.png "%TARGET\_DIR%\"

move /Y \*.csv "%TARGET\_DIR%\"

move /Y \*relatorio\*.html "%TARGET\_DIR%\"

:: Verificar se a operacao de mover encontrou problemas

:: (Esta e uma verificacao simples; o comando 'move' pode retornar 0 mesmo se

:: alguns arquivos nao forem encontrados, mas eh util para diagnostico basico.)

if exist \*\_logs\_\*.html (

echo.

echo Aviso: Alguns arquivos \*\_logs\_\*.html podem nao ter sido movidos.

echo Verifique se ha erros na saida acima.

) else (

echo.

echo Todos os arquivos \*\_logs\_\*.html encontrados foram movidos com sucesso para "%TARGET\_DIR%".

)

echo.

echo Operacao de organizacao de logs concluida.

REM pause

REM exit /b 0

REM Executa o primeiro comando monitor\_space

monitor\_space.exe --servidor\_nome "CEC2C" --servidor\_ip "10.22.214.17" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "HOM-IDAA0002" --servidor\_ip "10.22.199.53" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0001-node01-CEC1B" --servidor\_ip "10.22.236.19" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0002-node01-CEC2B" --servidor\_ip "10.22.214.172" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0003-onZ-D2G1-LP1036" --servidor\_ip "10.22.194.4" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0003-onZ-D3G1-LP1031" --servidor\_ip "10.22.208.7" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0003-onZ-D3G4-LP1032" --servidor\_ip "10.22.208.8" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0003-onZ-D8G1-LP1037" --servidor\_ip "10.22.194.5" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0004-onZ-D0G1-LP2037" --servidor\_ip "10.22.214.167" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0004-onZ-D2G1-LP2036" --servidor\_ip "10.22.214.166" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0004-onZ-D3G1-LP2031" --servidor\_ip "10.22.214.161" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

monitor\_space.exe --servidor\_nome "IDAA0004-onZ-D3G4-LP2032" --servidor\_ip "10.22.214.161" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

REM Executa o segundo comando grafico

grafico\_uso.exe --servidor\_nome "CEC2C"

grafico\_uso.exe --servidor\_nome "HOM-IDAA0002"

grafico\_uso.exe --servidor\_nome "IDAA0001-node01-CEC1B"

grafico\_uso.exe --servidor\_nome "IDAA0002-node01-CEC2B"

grafico\_uso.exe --servidor\_nome "IDAA0003-onZ-D2G1-LP1036"

grafico\_uso.exe --servidor\_nome "IDAA0003-onZ-D3G1-LP1031"

grafico\_uso.exe --servidor\_nome "IDAA0003-onZ-D3G4-LP1032"

grafico\_uso.exe --servidor\_nome "IDAA0003-onZ-D8G1-LP1037"

grafico\_uso.exe --servidor\_nome "IDAA0004-onZ-D0G1-LP2037"

grafico\_uso.exe --servidor\_nome "IDAA0004-onZ-D2G1-LP2036"

grafico\_uso.exe --servidor\_nome "IDAA0004-onZ-D3G1-LP2031"

grafico\_uso.exe --servidor\_nome "IDAA0004-onZ-D3G4-LP2032"

REM Aguarda 5 segundos antes de abrir o navegador (opcional)

timeout /t 5 >nul

REM Abre uma página da web no Google Chrome

start firefox "file:///C:\Users\c1311101\scripts\build\exe.win-amd64-3.11/espaco\_utilizado.html"

REM Executa o segundo comando SQL

SQL.exe --driver\_path "db2jcc4.jar" --jdbc\_url "jdbc:db2://GWDB2.BB.COM.BR:50100/BDB2P04" --db\_user "meuMFUSER" --db\_pass "meuMFPASS" --jdbc\_driver "com.ibm.db2.jcc.DB2Driver"

REM Executa o segundo comando dwa\_error

coletor\_logs.exe --servidor\_nome "CEC2C" --servidor\_ip "10.22.214.17" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "HOM-IDAA0002" --servidor\_ip "10.22.199.53" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0001-node01-CEC1B" --servidor\_ip "10.22.236.19" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0002-node01-CEC2B" --servidor\_ip "10.22.214.172" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0003-onZ-D2G1-LP1036" --servidor\_ip "10.22.194.4" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0003-onZ-D3G1-LP1031" --servidor\_ip "10.22.208.7" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0003-onZ-D3G4-LP1032" --servidor\_ip "10.22.208.8" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0003-onZ-D8G1-LP1037" --servidor\_ip "10.22.194.5" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0004-onZ-D0G1-LP2037" --servidor\_ip "10.22.214.167" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0004-onZ-D2G1-LP2036" --servidor\_ip "10.22.214.166" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0004-onZ-D3G1-LP2031" --servidor\_ip "10.22.214.161" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

coletor\_logs.exe --servidor\_nome "IDAA0004-onZ-D3G4-LP2032" --servidor\_ip "10.22.214.161" --usuario "root" --chave\_ssh "minha\_chave.pem" --porta\_ssh 2222

REM monitora latencia e le \*\_LOGS\_ e GERA \*\_LATENCY.CSV

latency\_monitor.exe

REM plot\_latency\_data.exe LE \*\_LATENCY.CSV e GERA GRAFICOS e ALERTAS

plot\_latency\_data.exe