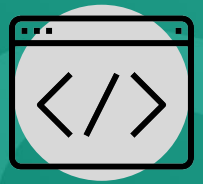
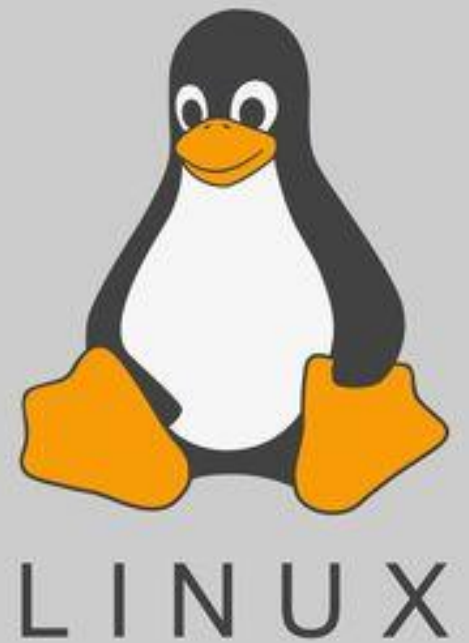
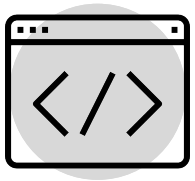
escola britânica de artes criativas & tecnologia

DE OLHO NO CÓDIGO





Linux



- Conheça o Linux
- Conheça o terminal Linux
- Crie os primeiros comandos no Linux
- Conheça as permissões
- Conheça o Shell Script

Conheça o Linux

- Os conteúdos vistos neste módulo são valiosos para um analista de dados, pois trazem **mais versatilidade e maior capacidade de trabalhar em ambientes diferentes**, sejam máquias Windows, MAC ou mesmo servidores Linux e ampliam as oportunidades de trabalho e colaboração.

- O Linux é conhecido por seu desempenho e eficiência e é considerado uma escolha sólida para tarefas de análise de dados que envolvem processamento intensivo. Além disso, muitas ferramentas e bibliotecas de análise de dados, como Python por exemplo, são bem suportadas no Linux. Ter familiaridade com o sistema facilita a integração de ferramentas essenciais.

Conheça o Linux

- Ao escolher um sistema operacional para análise de dados, é importante avaliar as necessidades do projeto. Se a análise de dados envolve a **execução de tarefas mais avançadas**, como modelagem de dados e uso intensivo de recursos, o Linux pode ser a melhor escolha devido à sua estabilidade e desempenho.

- Também é importante escolher uma distribuição adequada às necessidades do projeto. Distribuições populares, como Ubuntu, CentOS e Debian, são amplamente usadas para análise de dados.

Conheça o terminal Linux

- Utilizar o WSL em ambientes Windows fornece uma integração eficaz com ferramentas de análise de dados. Isso permite que os membros da equipe executem comandos Linux diretamente no Windows.

- Considere o uso de máquinas virtuais para criar ambientes Linux isolados, especialmente em ambientes Windows e macOS. Isso oferece maior controle e flexibilidade.

Crie os primeiros comandos no Linux

- Mantenha uma estrutura de diretórios bem organizada para armazenamento de dados e scripts. Use diretórios específicos para diferentes tipos de dados, facilitando a localização e a colaboração.

- Utilize scripts para automatizar tarefas repetitivas, como a limpeza de dados temporários ou a geração de relatórios diários. Isso economiza tempo e minimiza erros.

Crie os primeiros comandos no Linux

escola britânica de artes criativas & tecnologia

No exemplo abaixo, o script gera um relatório diário usando um script Python chamado '**script_analise.py**'. Ele cria um arquivo de relatório com a data atual no diretório de saída especificado. Isso é útil para automatizar a geração de relatórios regulares de análise de dados:

```
#!/bin/bash
# Este é um script para gerar um relatório diário
# Data atual
data_atual=$(date +"%Y-%m-%d")
# Diretório de saída para relatórios
output_dir="/home/usuario/relatorios"
# Gera o relatório com dados de análise
python script_analise.py > "$output_dir/relatorio_$data_atual.txt"

echo "Relatório diário gerado em:
$output_dir/relatorio_$data_atual.txt"
```

Veja a seguir a explicação de cada comando.

Crie os primeiros comandos no Linux

escola britânica de artes criativas & tecnologia

- **#!/bin/bash**: Ela é conhecida como shebang e indica ao sistema operacional que o script deve ser interpretado usando o Bash.

- **data_atual=$(date +"%Y-%m-%d")**: A variável ***data_atual*** é atribuída com a data atual formatada no formato "ano-mês-dia" usando o comando ***date***. O resultado é armazenado na variável ***data_atual***.

- **output_dir="/home/usuario/relatorios"**: A variável ***output_dir*** é atribuída com o caminho do diretório onde os relatórios serão armazenados.

- **python script_analise.py > "$output_dir/relatorio_$data_atual.txt"**: Executa o script Python ***script_analise.py*** e redireciona a saída para um arquivo de relatório no diretório especificado pela variável ***output_dir***. O arquivo é nomeado como "relatorio_ano-mes-dia.txt", utilizando a data atual.

  - **python script_analise.py**: Chama o interpretador Python para executar o script ***script_analise.py***.

  - **> "$output_dir/relatorio_$data_atual.txt"**: Redireciona a saída padrão para o arquivo de relatório, criando-o ou sobrescrevendo-o se já existir.

- **echo "Relatório diário gerado em: $output_dir/relatorio_$data_atual.txt"**: Exibe uma mensagem indicando o caminho completo do relatório gerado.

Conheça as permissões

- Ao conceder permissões de arquivo ou diretório, siga o **princípio do menor privilégio**, ou seja, conceda apenas as permissões necessárias para executar uma tarefa específica. Isso minimiza riscos de segurança.
- Mantenha um registro da documentação de permissões de arquivos e diretórios, para que a equipe possa entender quem tem acesso a quais recursos e por quê.

Conheça o Shell Script

escola britânica de artes criativas & tecnologia

- O Shell Script pode ser usado de diversas formas na análise de dados. Os scripts podem ser personalizados para atender às necessidades específicas do projeto, automatizando tarefas, facilitando a manipulação de dados e tornando o processo de análise mais eficiente. Por exemplo, para ler dados de um arquivo CSV, você pode usar comandos como **cat**, **awk** e **while read**.

- Suponha que temos um arquivo chamado **dados.csv** com duas colunas separadas por vírgula:

```
#!/bin/bash

while IFS=',' read -r coluna1 coluna2; do
    echo "Coluna1: $coluna1, Coluna2: $coluna2"
done < dados.csv
```

- **IFS=',':** Define o separador como vírgula para a leitura.
- **read -r coluna1 coluna2**: Lê as colunas do arquivo CSV.
- **echo "Coluna1: $coluna1, Coluna2: $coluna2"**: Exibe os dados processados.

Conheça o Shell Script

escola britânica de artes criativas & tecnologia

- Suponha que queremos filtrar linhas com um valor específico na segunda coluna do CSV:

```
#!/bin/bash

while IFS=',' read -r coluna1 coluna2; do
  if [ "$coluna2" == "valor_desejado" ]; then
    echo "Coluna1: $coluna1, Coluna2: $coluna2"
  fi
done < dados.csv
```

if ["$coluna2" == "valor_desejado"]: Condição para verificar se a segunda coluna atende ao critério de filtragem.

Conheça o Shell Script

escola britânica de artes criativas & tecnologia

- Vamos calcular a média de valores em uma coluna:

```
#!/bin/bash

soma=0
contador=0

while IFS=',' read -r _ coluna2; do
  soma=$((soma + coluna2))
  contador=$((contador + 1))
done < dados.csv

media=$((soma / contador))
echo "Média da Coluna2: $media"
```

soma=$((soma + coluna2)): Atualiza a soma dos valores na coluna.

contador=$((contador + 1)): Incrementa o contador para calcular a média.

Conheça o Shell Script

escola britânica de artes criativas & tecnologia

- Utilize comandos como ***gnuplot*** para criar gráficos simples a partir dos dados:

```
#!/bin/bash

echo "set terminal png" > script.gp
echo "set output 'grafico.png'" >> script.gp
echo "plot 'dados.csv' using 2 with linespoints title 'Coluna2'" >> script.gp

gnuplot script.gp
```

- **echo "set terminal png" > script.gp**: Configura o terminal de saída para PNG.
- **echo "set output 'grafico.png'" >> script.gp**: Define o nome do arquivo de saída.
- **echo "plot 'dados.csv' using 2 with linespoints title 'Coluna2'" >> script.gp**: Cria um gráfico de linhas e pontos usando a segunda coluna.

escola britânica de artes criativas & tecnologia

Bons estudos!