Recuperação dos dados originais do banco de dados

lerArquivoJson(caminhoArquivo)

Lógica e Implementação:

- Esta função é crucial para iniciar o processo de correção. Ela utiliza o módulo `fs` (File System) do Node.js para ler arquivos do sistema de arquivos.
- Utiliza o método `readFileSync` para ler os dados do arquivo JSON de forma síncrona, garantindo que o script não prossiga até que os dados sejam completamente carregados.
- Após a leitura do arquivo, a função utiliza `JSON.parse` para converter a string JSON em um objeto JavaScript. Isso permite manipular os dados de forma mais eficiente no script.
- Implementa tratamento de erros usando `try-catch` para lidar com possíveis exceções, como arquivos não encontrados ou dados corrompidos, garantindo que o script não falhe e forneça feedback útil em caso de erro.

corrigirNomesEMarcas(dados)

Lógica e Implementação:

- Esta função foca em corrigir os nomes de veículos e marcas que estão corrompidos devido a caracteres alterados.
- Usa o método `map` para iterar sobre cada item no array de dados. Isso garante que todas as entradas sejam verificadas e corrigidas se necessário.
- Dentro da função `map`, utiliza-se expressões regulares e o método `replace` para localizar e substituir caracteres corrompidos ("æ" e "ø") pelos caracteres corretos ("a" e "o").
- A função verifica a existência das propriedades `nome` e `marca` em cada item e aplica a correção apenas se essas propriedades existirem, evitando assim erros de referência a propriedades indefinidas.

corrigirVendas(dados)

Lógica e Implementação:

- O objetivo desta função é garantir que os dados numéricos estejam no tipo correto (números), especialmente os dados de vendas que foram incorretamente formatados como strings.
- Assim como na função anterior, utiliza `map` para iterar sobre cada item.
- Em cada item, verifica se o valor de `vendas` é do tipo string usando `typeof`. Se for string, usa `parseInt` para converter o valor para um número inteiro.
- Essa abordagem garante a consistência dos tipos de dados, o que é essencial para análises e operações de dados subsequentes.

exportarDadosCorrigidos(dados, caminhoArquivo)

Lógica e Implementação:

- Após a correção dos dados, esta função se encarrega de salvar os dados corrigidos de volta em um arquivo JSON.
- Utiliza `JSON.stringify` para converter o objeto JavaScript de volta para uma string JSON. O segundo e terceiro parâmetros de `JSON.stringify` são usados para formatar a saída, tornando-a mais legível.
- Usa `fs.writeFileSync` para escrever a string JSON no arquivo especificado. Isso é feito de maneira síncrona para garantir que a operação seja concluída antes que o script prossiga.
- Implementa tratamento de erros para lidar com problemas potenciais durante a gravação do arquivo, garantindo que qualquer problema seja comunicado ao usuário.

Tratamentos Feitos no Código para Evitar Bugs:

- Utilização de `try-catch` para manipulação de erros, garantindo que o programa não falhe abruptamente.
- Verificações condicionais para assegurar que apenas os dados necessários sejam alterados.
- Testes unitários para cada função, garantindo a precisão antes de processar o banco de dados completo.

Outros Pontos Relevantes:

- A modularidade do código permite fácil manutenção e atualizações futuras.
- As habilidades demonstradas neste projeto estão alinhadas com a manipulação e análise de dados, fundamentais para a vaga de Web Analytics.
- Este desafio reflete minha capacidade de lidar com dados corrompidos e transformá-los em informações úteis, uma competência vital para a geração de insights em um ambiente orientado por dados.