O seguinte código(localizado no arquivo resolucao.js) tem como objetivo resolver um problema específico de corrupção de dados. Foram implementadas as funções “corrigirNomes()”, que tem como objetivo a recuperação de caracteres originais do banco, “corrigirPrecos()”, que tem como objetivo recuperar o tipo original do atributo “price” do banco, “corrigirQuantidades()”, que tem como objetivo recuperar o atributo “quantity” ausente em alguns produtos do banco, “lerJson()”, que tem como objetivo ler um arquivo passado ao programa para posterior tratamento dos dados, “gravarJson()”, que tem como objetivo gravar os dados tratados em um arquivo, além das funções de validação dos resultados “ordenarProdutos()”, que tem como objetivo imprimir todos os nomes dos produtos ordenados por categoria em ordem alfabética e id em ordem crescente, nessa ordem, e “calcularValorEstoque()”, que tem como objetivo calcular o valor total do estoque por categoria.

Todo o código foi implementado em JavaScript, isso pode-se atribuir ao fato de o banco de dados a ser tratado estar no formato JSON(JavaScript Object Notation**)**, sendo assim, é uma boa opção fazer uso dessa linguagem, uma vez que JSON é derivado de JavaScript, o que faz com que tenham excelente integração.

Alguns tratamentos foram implementados, mas se resumem apenas a informar o erro ao usuário. Pelo objetivo do código se tratar de uma finalidade bem específica, os tratamentos foram implementados apenas em lugares mais passíveis de bugs, sendo estes as funções de leitura e escrita de arquivos(readFileSync() e writeFileSync()), as funções são executadas dentro de um try, caso algum erro ocorra ele é direcionado para o catch() e é exibido ao usuário, no caso da função de leitura readFileSync() foi adicionado um tratamento para capturar o código de erro ENOENT, que indica que não foi possível acessar o arquivo informado, dessa forma é possível exibir uma mensagem mais amigável ao usuário em caso de erro.

Segue abaixo as funcionalidades do código e como devem ser utilizadas:

lerJson(string):

Recebe uma string informando a localização de um arquivo JSON. Retorna um JavaScript Object correspondente ao JSON lido.

corrigirNomes(object):

Recebe um JavaScript Object correspondente a um JSON. Retorna um JavaScript Object com os caracteres “a”, “c”, “o” e “b” tratados.

corrigirPrecos(object):

Recebe um JavaScript Object correspondente a um JSON. Retorna um JavaScript Object com todos os atributos “price” do objeto em tipo numérico.

corrigirQuantidades(object):

Recebe um JavaScript Object correspondente a um JSON. Retorna um JavaScript Object com todos os produtos contendo o atributo “quantity”(valor default = 0).

gravarJson(object):

Recebe um JavaScript Object correspondente a um JSON. Não possui retorno, cria/altera o arquivo “saida.json”, gravando nele os dados do object recebido na chamada da função; os dados são gravados em formato legível por humanos.

ordenarProdutos(object):

Recebe um JavaScript Object correspondente a um JSON previamente tratado pelas funções anteriores. Não possui retorno, imprime no console os nomes(atributo “name”) dos produtos, esses por sua vez estão ordenados por categoria(atributo "category”) em ordem alfabética e por id em ordem crescente.

calcularValorEstoque(object):

Recebe um JavaScript Object correspondente a um JSON previamente tratado pelas funções anteriores. Não possui retorno, imprime no console o valor total do estoque por categoria(no formato “’categoria’: ‘valor total’”, sendo o valor total de cada categoria o somatório dos valores de todos os produtos da mesma categoria(sendo o preço de cada classe de produto a multiplicação do seu valor pela quantidade de produtos existentes em estoque).