



FULL DIGITAL PERFORMANCE

## Projeto Classificatório

Processo seletivo - Web Development

## O problema

Você é responsável por um software de gestão de estoque de produtos. Ao fazer uma alteração no sistema, uma rotina que não foi devidamente testada acabou quebrando todo o banco de dados. Por sorte, não houve perda completa dos dados, mas eles não estão mais no formato esperado pelo sistema. Sua missão nesse projeto é recuperar os dados e deixá-los no formato adequado novamente. Além disso, você precisará criar também alguns métodos para validação das correções.

O banco de dados utilizado é um banco de dados NoSQL, orientado a documentos. Não se assuste caso você não conheça esses nomes. Não iremos mexer diretamente com banco de dados, mas somente com o documento, em formato JSON, onde estão armazenados os dados de produto.

## Problemas detectados no banco de dados corrompido

### 1. Nomes

Todos os nomes de produto tiveram alguns caracteres modificados, houve substituição de todos os "a" por "æ", "c" por "ç", "o" por "ø", "b" por "ß". É preciso reverter essas substituições para recuperar os nomes originais.

Exemplo:

Original:

```
"name": "iPhone XS Max Prata, com Tela de 6,5, 4G, 64 GB  
e Câmera de 12 MP"
```

Corrompido:

```
"name": "iPhøne XS Mæx Prætæ, cøm Telæ de 6,5, 4G, 64 GB  
e Câmeræ de 12 MP"
```

## 2. Preços

Os preços dos produtos devem ser sempre do tipo number, mas alguns deles estão no tipo string. É necessário transformar as strings novamente em number.

Exemplo:

Original:

```
"price": 1250.00
```

Corrompido:

```
"price": "1250.00"
```

## 3. Quantidades

Nos produtos onde a quantidade em estoque era zero, o atributo "quantity" sumiu. Ele precisa existir em todos os produtos, mesmo naqueles em que o estoque é 0.

Exemplo:

Original:

```
"name": "Conjunto de Panelas Antiaderentes com 05 Peças  
Paris",  
"quantity": 0,  
"price": 192.84
```

Corrompido:

```
"name": "Conjunto de Panelas Antiaderentes com 05 Peças  
Paris",  
"price": 192.84
```

## Questões:

Para esse projeto, você utilizará o arquivo broken-database.json (disponível [aqui](#)) e irá fazer uma série de transformações até que ele volte ao formato original. Para isso será necessário desenvolver algumas funções e depois verificar se realmente foi recuperado. Você **deverá utilizar JavaScript** para resolver esse problema, caso não conheça nenhuma dessas linguagens, é uma ótima oportunidade para aprender! :)

### **1. Recuperação dos dados originais do banco de dados**

Você deverá criar uma função para ler o arquivo broken-database.json e criar três funções para percorrer o banco de dados corrompido e corrigir os três erros descritos anteriormente, além de uma função para exportar um arquivo .json com a saída.

Portanto serão 5 funções:

- a) Ler o arquivo Json;
- b) Corrigir nomes;
- c) Corrigir preços;
- d) Corrigir quantidades;
- e) Exportar um arquivo JSON com o banco corrigido;

Implementar e entregar as quatro funções em um mesmo arquivo 'resolucao.js' para correção. Enviar também para correção um arquivo no formato JSON com o banco de dados corrigido, ou seja, após passar pelas três funções de correção.

### **2. Validação do banco de dados corrigido**

Você deverá implementar funções para validar a sua recuperação do banco de dados. Todas essas funções deverão ter como input o seu banco de dados corrigido na questão 1. As funções de validação são:

- a) Uma função que imprime a lista com todos os nomes dos produtos, ordenados primeiro por categoria em ordem alfabética e ordenados por id em ordem crescente. Obs: é apenas uma saída, ordenada pelos dois fatores citados acima.

b) Uma função que calcula qual é o valor total do estoque por categoria, ou seja, a soma do valor de todos os produtos em estoque de cada categoria, considerando a quantidade de cada produto.

Essas funções também devem estar no arquivo 'resolucao.js'.

## Ambiente de teste:

Vamos testar a sua solução através do terminal utilizando comandos específicos.

Para o teste feito em JavaScript, utilizaremos:

```
$ node resolucao.js
```

Você pode aprender como instalar o node [aqui](#).

## Entrega ('resolucao.js'):

- Um arquivo chamado 'resolucao.js' contendo o código fonte de toda a implementação. Não esqueça de referenciar trechos de código utilizados da internet.
- Um arquivo chamado 'saida.json' contendo a resposta da execução do algoritmo.
- Um documento em formato PDF contendo uma breve explicação do algoritmo como um todo. Você deve abordar os pontos:
  - Uma breve explicação de cada funcionalidade,
  - O porquê da escolha da linguagem,
  - Tratamentos feitos no código para evitar bugs,
  - entre outras coisas que queira compartilhar.
- Após tudo pronto, coloque seu projeto em uma pasta pública em seu google drive ou em seu github e nos envie o link para que possamos avaliar o seu projeto.

Obs¹: A documentação é muito importante. Atente-se a isto.

Obs²: NÃO é escopo do teste desenvolver qualquer página .html.