Documentar um código é uma prática importante para explicar o que o código faz, como ele funciona e como ele deve ser usado.

Javascript

```
// Função para ler o conteúdo de um arquivo
function lerArquivo(path) {
	try {
	return fs.readFileSync(path, 'utf8');
	} catch (error) {
	console.error(error);
	}
}
// Função para escrever dados em um arquivo
function escreverArquivo(path, data) {
	try {
	return fs.writeFileSync(path, data);
	} catch (error) {
	console.error(error);
	}
}
// Função para substituir caracteres incorretos em um campo específico dos dados
function substituirCaracteres(dados, campo) {
	return dados.map((item) =>  {
	item[campo] = item[campo].replaceAll('æ', 'a').replaceAll('ø', 'o');
	return item;
	});
}
// Função para corrigir o valor das vendas convertendo para inteiro
function corrigirValorVendas(dados) {
	return dados.map((venda) => {
	venda.vendas = parseInt(venda.vendas);
	return venda;
```

```
    });
}
// Função principal que executa as operações de leitura, tratamento e escrita de arquivos
function executa() {
    // Lendo os arquivos JSON
    const arquivoVeiculos = JSON.parse(lerArquivo('docs/broken_database_1.json'));
    const arquivoMarcas = JSON.parse(lerArquivo('docs/broken_database_2.json'));

    // Tratando os caracteres incorretos nos nomes dos veículos e marcas
    let arquivoVeiculosCorrigido = substituirCaracteres(arquivoVeiculos, 'nome');
    let arquivoMarcasCorrigido = substituirCaracteres(arquivoMarcas, 'marca');

    // Corrigindo o valor das vendas nos veículos
    arquivoVeiculosCorrigido = corrigirValorVendas(arquivoVeiculosCorrigido);

    // Escrevendo os arquivos corrigidos
    escreverArquivo('docs/arquivo_veiculos.json',
JSON.stringify(arquivoVeiculosCorrigido));
    escreverArquivo('docs/arquivo_marcas.json',
JSON.stringify(arquivoMarcasCorrigido));
}
// Executando a função principal para corrigir os arquivos
executa();
```

Neste código, documentei cada função com comentários explicativos sobre o que ela faz. Isso ajuda a entender o propósito e a funcionalidade de cada parte do código.

Documentação para o código SQL

1 *Maior Volume de Vendas por Marca:*

- Objetivo: Identificar a marca que teve o maior volume de vendas.
- Consulta:

```
    sql
    SELECT
```

```
v.marca,
v.id_marca,
SUM(v.vendas) as qtd_vendas
FROM venda_veiculo as v
GROUP BY v.id_marca
ORDER BY qtd_vendas DESC
LIMIT 1;
```

- Descrição: Esta consulta retorna a marca com o maior volume de vendas ao selecionar a marca, o ID da marca e a soma das vendas para cada marca. Os resultados são agrupados pelo ID da marca e ordenados em ordem decrescente de volume de vendas. Em seguida, é aplicado um limite de 1 para obter a marca com o maior volume de vendas.

2*Maior e Menor Receita Gerada por Veículo:*

- Objetivo: Identificar os veículos que geraram a maior e menor receita.

- Consulta:

```
sql
SELECT (select nome
from venda_veiculo
GROUP BY nome
ORDER BY sum(valor_veiculo) asc) as menor_receita,
(select
        nome
from venda_veiculo
GROUP BY nome
ORDER BY sum(valor_veiculo) DESC) as maior_receita;
```

3. *Média de Vendas por Ano e Marca:*

- Objetivo: Calcular a média de vendas por ano e marca.

- Consulta:

```
sql
SELECT
v.marca,
```

```
v.id_marca,
ROUND(AVG(v.vendas), 2) as media_vendas,
strftime('%Y', date(data)) AS ano
FROM venda_veiculo as v
GROUP BY v.id_marca;
```

- Descrição: Esta consulta calcula a média de vendas por ano e marca, arredondando o resultado para duas casas decimais. Ela seleciona a marca, o ID da marca, a média de vendas e o ano das vendas (extraído da data), agrupados pelo ID da marca.

4. *Marcas com Maior Receita e Menor Número de Vendas:*

- Objetivo: Identificar as marcas que geraram uma receita maior com um número menor de vendas.

- Consulta:

```
sql
SELECT
v.marca,
(SELECT sum(vv.valor_veiculo)
FROM venda_veiculo vv
WHERE vv.id_marca = v.id_marca) as receita,
(SELECT SUM(vvv.vendas) as qtd_vendas
FROM venda_veiculo as vvv
WHERE v.id_marca = vvv.id_marca) as qtd_vendas
FROM venda_veiculo v
GROUP BY v.id_marca
ORDER BY receita DESC, qtd_vendas;
```

- Descrição: Nesta consulta, é selecionada a marca juntamente com a receita total gerada por essa marca (calculada em uma subconsulta) e o número total de vendas para essa marca. Os resultados são agrupados pelo ID da marca e ordenados em ordem decrescente de receita e crescente de número de vendas.

5. *Relação entre os Veículos Mais Vendidos:*

- Objetivo: Verificar se existe alguma relação entre os veículos mais vendidos.

- Consulta:

```
sql

SELECT * FROM venda_veiculo v WHERE v.nome IN

(SELECT vv.nome

FROM venda_veiculo as vv

GROUP BY vv.nome

ORDER BY SUM(vv.vendas) DESC

LIMIT 3);
```

- Descrição: Esta consulta tenta verificar se existe alguma relação entre os veículos mais vendidos, selecionando todos os dados dos veículos cujo nome está entre os três veículos com maiores volumes de vendas.