Tópicos em Engenharia de Software: Relatório Técnico

Hugo Poletto Alacoque Gomes

https://github.com/HugoPoletto34/Trabalho-4-Topicos-Eng-Software

Introdução

A atividade consiste em avaliar a qualidade de um software que processa dados em Big Data. Os testes em software são importantes para garantir que o software funcione corretamente e atenda aos requisitos do usuário. Em ambientes de Big Data, os testes são ainda mais importantes, pois os sistemas são complexos e lidam com grandes quantidades de dados.

Os testes em Big Data podem ser desafiadores, pois os sistemas são geralmente distribuídos e heterogêneos. No entanto, os testes são essenciais para garantir a confiabilidade e a segurança desses sistemas.

1 Desenvolvimento de Testes Unitários

1.1 Função avgAgeCountry

O processo de implementação dos testes para a função ‘avgAgeCountry‘ consideram os seguintes cenários:

1.1.1 Parâmetro de dados vazio

Caso a função avgAgeCountry receba o parâmetro de dados vazio, deverá retornar um objeto vazio.

1.1.2 Valores de idade ausentes ou nulos

Em casos de valores de idades ausentes ou nulos, deverá lançar uma exceção.

1.1.3 Campo ‘country‘ ausente ou nulo

Em casos do campo country ausente ou nulo, deverá lançar uma exceção.

1.2 Outras Funções de Processamento

As outras funções desenvolvidas são:

- qualPaisComMaiorMediaDeIdade(): Essa função retorna o país com a maior média de idade. Ela é importante para identificar os países com a população mais velha.

- qualPaisComMenorMediaDeIdade(): Essa função retorna o país com a menor média de idade. Ela é importante para identificar os países com a população mais jovem.

Ambas as funções são importantes para a análise de dados demográficos. Elas podem ser usadas para identificar tendências e padrões na população de diferentes países.

A função qualPaisComMaiorMediaDeIdade() pode ser usada para identificar países com uma população envelhecida. Isso pode ser um indicador de problemas sociais, como a falta de mão de obra e o aumento do custo de saúde.

A função qualPaisComMenorMediaDeIdade() pode ser usada para identificar países com uma população jovem. Isso pode ser um indicador de crescimento econômico e prosperidade.

1.3 Função de Transformação

A função de transformação é o segundo parâmetro não obrigatório que converte a idade para o que for definido na função antes de fazer a média. Por padrão, a função recebe a idade por parâmetro e retorna ela mesma. Esse paramâmetro traz a possibilidade de visualizarmos a média das idades no formato desejado, sendo em meses, dias, horas, etc. Nos testes, a função de transformação recebe a idade de anos para meses multiplicando a idade por 12.

2 Objetivos dos Testes

Os objetivos dos testes desenvolvidos são:

Prevenir erros na leitura de arquivos JSON: Os testes test-read-json-file-success, test-read-json-file-file-not-found e test-read-json-file-invalid-json verificam se a função read-json-file() está funcionando corretamente, mesmo em situações de erro, como arquivo não encontrado ou JSON inválido.

Prevenir erros na função avgAgeCountry(): Os testes test-avarage-age-function, test-data-null-avarage-function, test-zero-ages, test-exception-when-none-ages, test-exception-when-none-country e test-exception-when-void-country verificam se a função avgAge-Country() está funcionando corretamente em todos os casos possíveis, incluindo dados vazios, idades zeradas, ausência de idades ou países, e países vazios.

Prevenir erros na função qualPaisComMaiorMediaDeIdade(): Os testes test-qual-Pais-Com-Maior-Media-De-Idade e test-qual-Pais-Com-Maior-Media-De-Idade-empty verificam se a função qualPaisComMaiorMediaDeIdade() está funcionando corretamente, mesmo com dados vazios.

Prevenir erros na função qualPaisComMenorMediaDeIdade(): Os testes test-qual-Pais-Com-Menor-Media-De-Idade e test-qual-Pais-Com-Menor-Media-De-Idade-empty verificam se a função qualPaisComMenorMediaDeIdade() está funcionando corretamente, mesmo com dados vazios.

Prevenir erros na transformação de dados na função avgAgeCountry(): O

teste test-avgAgeCountry-with-transform verifica se a função avgAgeCountry() está funcionando corretamente quando uma transformação é aplicada aos dados.

Prevenir erros na função avgAgeCountry() com dados vazios: O teste test-avgAgeCountry-with-transform-empty verifica se a função avgAgeCountry() está funcionando corretamente com dados vazios, mesmo quando uma transformação é aplicada aos dados.

2.1 Relevância do teste unitário e a liberdade para criar diferentes asserts

Os testes unitários são importantes para garantir a qualidade do software, pois eles permitem verificar o comportamento de cada unidade de código de forma isolada. Isso ajuda a prevenir erros e a garantir que o software funcione corretamente conforme o esperado.

A liberdade para criar diferentes asserts permite que os desenvolvedores testem o código de forma completa e abrangente. Por exemplo, é possível testar o valor retornado por uma função, a exceção lançada por uma função ou até mesmo o comportamento interno de uma função.

2.2 A intencão de cada teste

A intenção de cada teste é verificar o comportamento de uma unidade de código específica em diferentes cenários. Por exemplo, o teste test-read-json-file-success verifica se a função read-json-file() é capaz de ler um arquivo JSON com sucesso.

O teste test-read-json-file-file-not-found verifica se a função read-json-file() lança uma exceção quando o arquivo não é encontrado. O teste test-read-json-file-invalid-json verifica se a função read-json-file() lança uma exceção quando o arquivo JSON é inválido.

Os testes unitários devem ser escritos de forma clara e concisa, para que sejam fáceis de entender e manter. Além disso, os testes unitários devem ser executados regularmente para garantir que o software continue funcionando corretamente após as alterações.

3 Reflexão sobre Testes em Big Data

Escrever testes para ambientes de Big Data apresenta desafios específicos que requerem atenção e estratégia. Um dos principais desafios é lidar com o volume de dados, uma vez que os testes precisam ser capazes de manipular grandes quantidades de informações, o que, por sua vez, pode resultar em tempos de execução mais longos. Além disso, a distribuição dos dados em diferentes nós de um cluster pode complicar a execução de testes em um ambiente local, tornando essencial a adaptação dos testes para essa realidade.

A escalabilidade é outra questão crucial, visto que os testes precisam ser dimensionados para ambientes com grandes volumes de dados, a fim de garantir que o software funcione de maneira eficiente e confiável.

Testar funções que processam grandes volumes de dados é fundamental para assegurar que elas operem corretamente e não apresentem problemas de desempenho que possam impactar a qualidade do sistema.

Nesse contexto, o PySpark surge como uma ferramenta para otimizar o desempenho de aplicações de Big Data. Ele oferece recursos como paralelismo, que executa operações em vários nós do cluster simultaneamente, armazenamento em cache para acesso mais rápido aos dados e compressão para economizar espaço em disco.

Para realizar um teste de desempenho, pode-se seguir o seguinte procedimento:

1. Criar um conjunto de dados de teste com um volume semelhante ao que será utilizado no ambiente de produção.
2. Executar a função em um cluster PySpark com diferentes configurações de paralelismo e cache.
3. Medir o tempo de execução da função para cada configuração.
4. Os resultados do teste de desempenho podem ser usados para identificar as configurações que oferecem o melhor desempenho, permitindo assim otimizar o software para ambientes de Big Data.

4 Conclusão

A atividade permitiu aprender sobre a importância dos testes para o desenvolvimento de software em ambientes de Big Data. Os testes são essenciais para garantir a qualidade e o desempenho das funções de processamento de dados.

Ao escrever testes, é importante considerar os desafios específicos de ambientes de Big Data, como o volume de dados, a distribuição dos dados e a escalabilidade dos testes. A utilização de recursos de otimização de desempenho, como paralelismo, cache e compressão, pode ajudar a melhorar o desempenho das funções de processamento de dados. Os testes de desempenho são uma ferramenta importante para avaliar o desempenho das funções de processamento de dados em diferentes cenários.