**Projeto BD-ITAC**

Projeto de Big Data and IoT for Alerts and Crisis

**Instituto Tecnológico de Aeronáutica - ITA**

Relatório Sintético

Sprint 03

**Time de Desenvolvimento:**

**PO -** Marco Pellizzola

**SM -** Lineu Lima

**TMs -** Luciana Fogaça, Marcos Lopes, Larissa Menezes, Rodrigo Santana e João Siles



# 1 Introdução

O Projeto BD-ITAC 2016 refere-se ao desenvolvimento de um sistema integrado para o gerenciamento e alerta de crises e que tem por objetivo englobar os conceitos de Big Data e Internet of Things (IoT).

Diferente de outros sistemas, o BD-ITAC 2016 é um projeto acadêmico de alto nível de integração entre dispositivos e tecnologias, testável e desenvolvido de forma ágil.

O TS#06 (Time Scrum 06), responsável pela implantação da estrutura de banco de dados tem como objetivo suportar os demais times e integrar suas necessidades no que diz respeito à persistência de dados. Para tal, o time optou pela utilização de dois modelos distintos. O banco de dados relacional (MySQL) e o banco de dados não relacional (Cassandra).

**1.1 Objetivo da Sprint**

## Durante o Sprint#3, o time TS#06 teve como objetivo oferecer o suporte necessário às equipes do projeto, integrando as demandas de banco de dados de cada equipe e disponibilizando um Modelo Entidade-Relacionamento (MER) preliminar, que será validado na prática, durante a Sprint#3. Além do suporte às equipes, com o auxílio da ferramenta TestLink e empregando algumas das técnicas apresentadas em sala, como por exemplo o “*Pairwise Testing*”, o time 06 elaborou o plano de testes, cuja as execuções foram planejadas para o Sprint#2 e #3. Para manter-se aderente ao planejamento e garantir uma maior maturidade dos testes, o TS#06 investiu horas do sprint na automação dos testes, por meio das ferramentas Jmeter, para os testes do MySql e Cassandra-Stress, para os testes do Cassandra.

## **1.2 Sprint Backlog**

|  |  |
| --- | --- |
| **ID** | **User Story** |
|  |  |
|  |  |
|  |  |

## 

# 2 Desenvolvimento

## **2.1 Requisitos Funcionais:**

* Integração das modelagens requeridas pelos demais times;
* Criação de um Modelo Entidade-Relacionamento (MER);
* Scripts de banco de dados disponíveis para os times;
* Ferramenta instalada e configurada para testes automatizados do MySQL;
* Ferramenta instalada e configurada para testes automatizados do Cassandra.
* Plano de testes definido.

## **2.2 Integração:**

* O TS#06 teve integração com todos os outros times do projeto nesta sprint.

**2.3 TS06-US015**

O objetivo desta User Story foi garantir a atualização dos bancos de dados tanto relacional quanto não relacional de acordo com as demandas de cada time scrum. Para isso, esta atividade foi distribuída em tasks entre os integrantes do TS#06 que variaram entre criar as demandas de banco de dados, alterar as tabelas já criadas e excluir caso necessário.

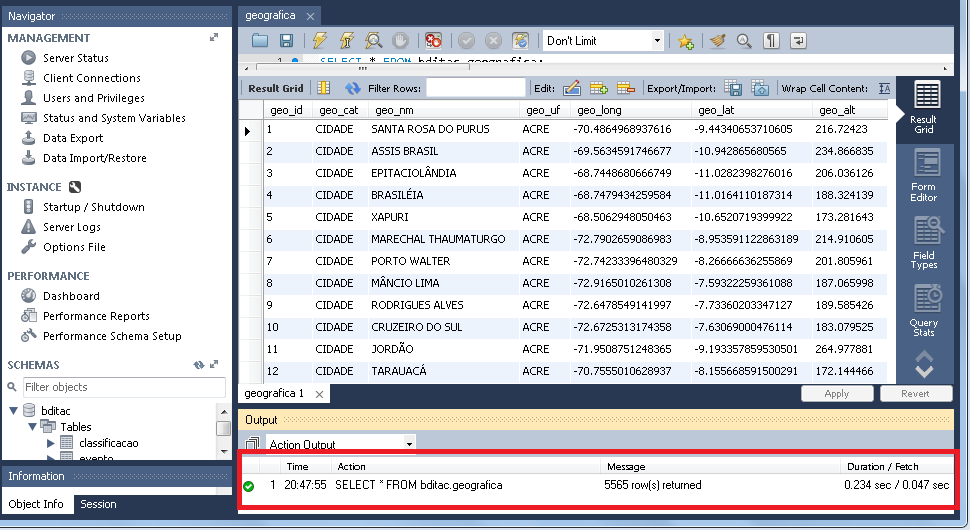
Com a necessidade de criar um modelo de relacionamento integrado, esta US necessitou da participação dos demais times em reuniões de integração e alinhamento de papéis e responsabilidades.

**COLOCAR MER FINAL AQUI!!!**  
Figura 01 - Modelagem de dados

**2.4 TS06-US10**

O objetivo desta US foi disponibilizar tabelas com grande volume de dados para que os demais times pudessem comparar e definir em suas aplicações qual seria o banco de dados mais adequado para certos tipos de dados.

Como subproduto desta US, o TS#06 pode verificar se o relacionamento das tabelas estava coerente, exercitando as tabelas com diversos inputs e verificando se o banco de dados estava se comportando conforme o esperado.

  
Figura 02 - Exemplo de consulta a uma tabela com grande volume de dados.

**2.5 TS06-US11**

Para esta User Story, pesquisou-se ferramentas de automatização de testes de banco de dados. Assim sendo, optou-se pela utilização da ferramenta Apache Jmeter para a realização dos testes relativos ao banco de dados relacional (MySQL). Segundo site do fabricante, a mesma é “open-source”, aplicação 100% Java, desenvolvida para medir performance. Pode-se fazer testes de carga em banco de dados por meio de requisições JDBC.

Para os testes do banco de dados não-relacional (Cassandra), por sua vez, considerou-se utilizar também o Jmeter. Isso é possível por meio da instalação de um plugin. No entanto, a documentação encontrada relativa ao assunto é bastante limitada e a equipe não conseguiu fazer a configuração necessária para a automação dos testes. Tendo isto em vista, decidiu-se utilizar uma outra ferramenta, chamada Cassandra-stress a qual é também baseada em Java e, como o nome sugere, utilizada para testes de carga. A mesma foi instalada e corretamente configurada.

A seguir pode-se visualizar *screenshots* de ambas as ferramentas, com exemplos de testes automatizados realizados nos bancos de dados do projeto.

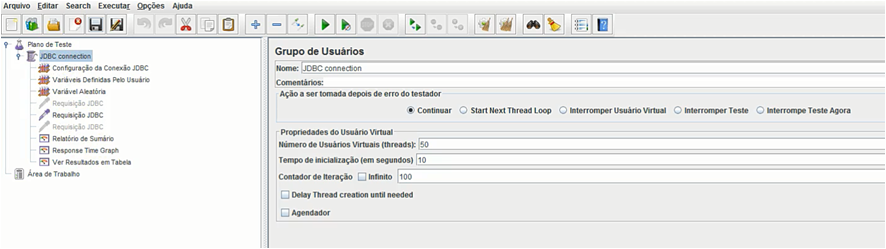


Figura 03 - Tela inicial do Jmeter com um plano de teste automatizado configurado para fazer select, insert e delete de um grande número de dados

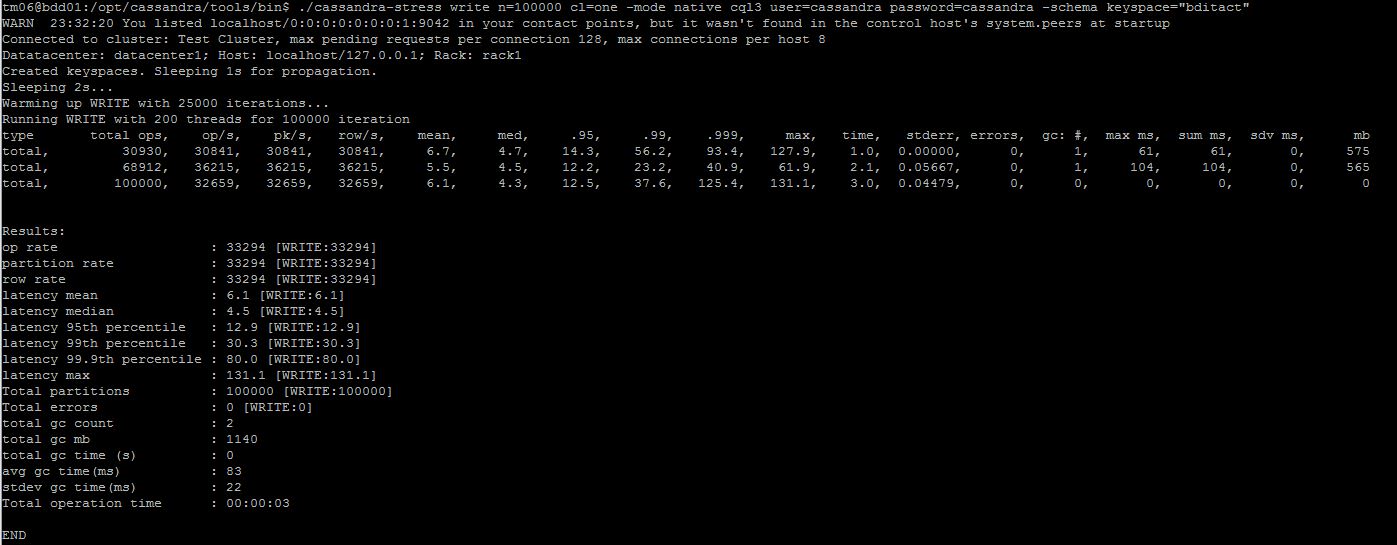


Figura 04 - Testes de carga sendo rodados no Cassandra-stress (Inserção de 100 mil registros)

Para a criação do Plano e Caso de Testes, além da gestão do processo, foi utilizada a ferramenta *open source* Testlink (versão 1.9.14 Padawan). Esta é direcionada para a gestão de todo o processo de testes de forma organizada e extremamente simples, sendo uma das ferramentas *open source* de gestão do processo de testes mais utilizada atualmente.

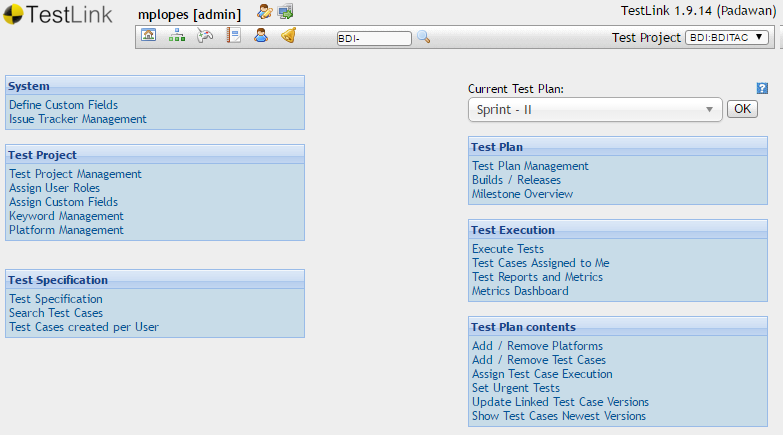


Figura 05 - Página inicial do Projeto de Testes: BD:BDITAC | Plano de Testes: Sprint II

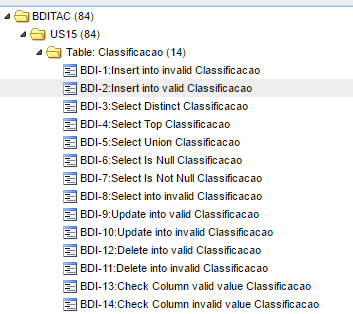


Figura 06 - Plano de Testes: Sprint II

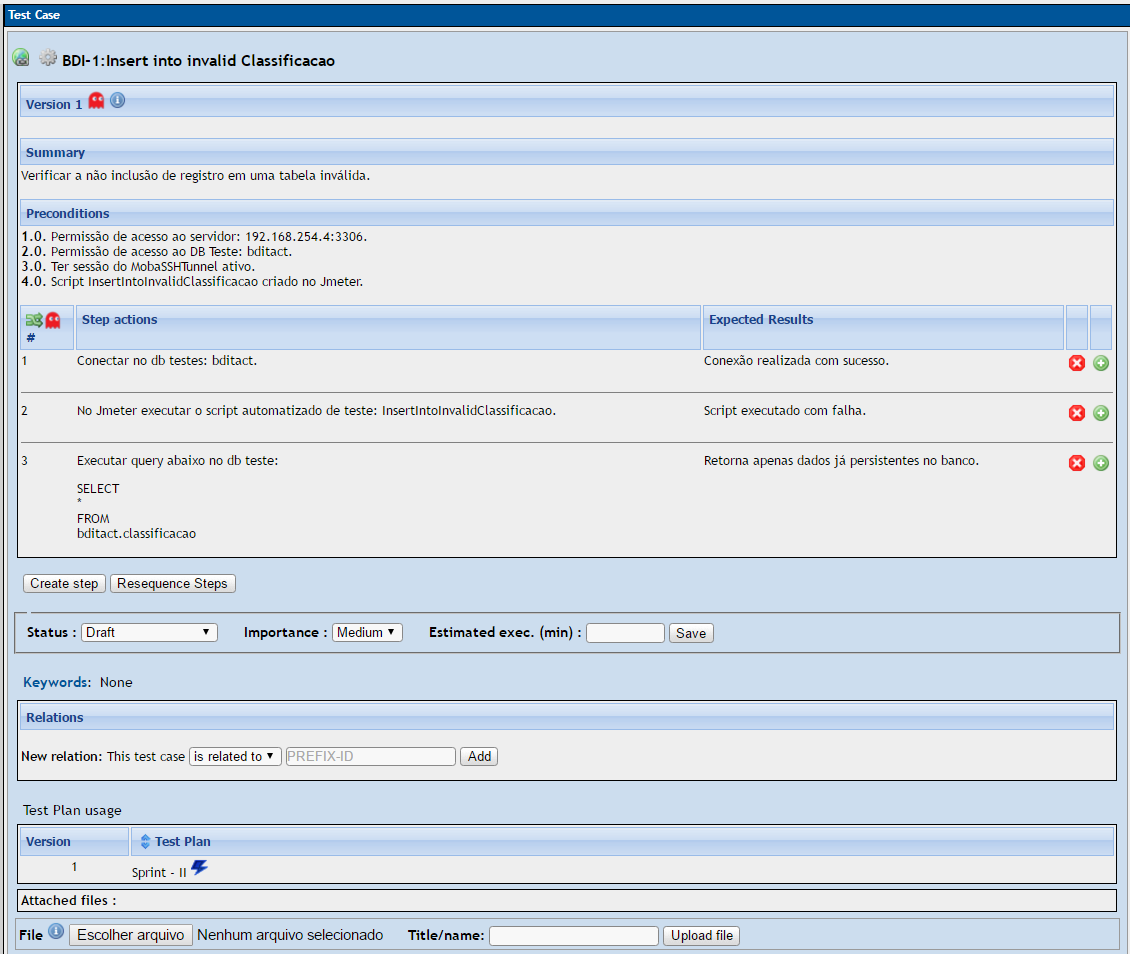
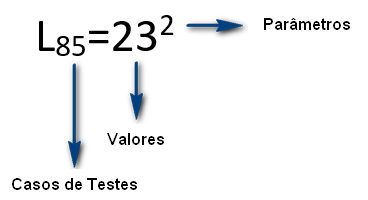


Figura 07 - Caso de Teste: BDI-1: Insert into invalid *Classificacao*

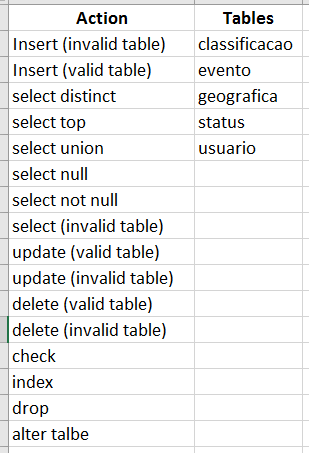
Foi utilizada também a técnica de caixa preta - Pairwise Testing na elucidação dos cenários de testes possíveis, conforme os valores e parâmetros abaixo, teríamos:

L=232 = 529 combinações possíveis

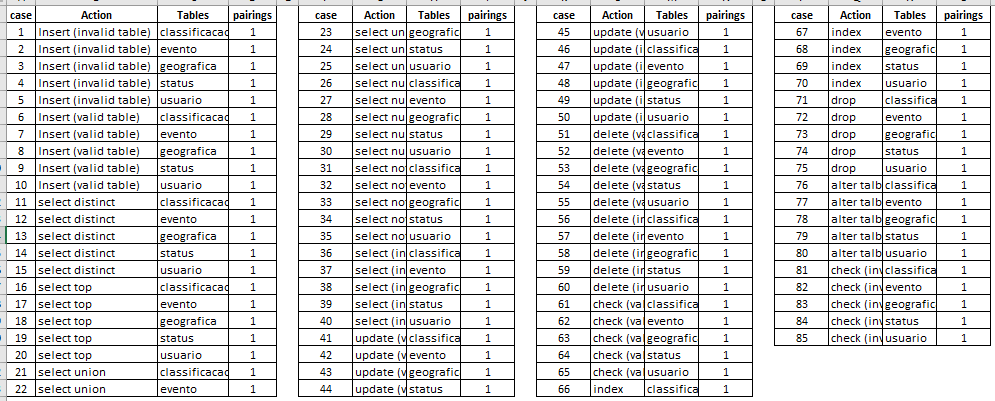
Aplicando Pairwise Testing - Orthogonal Array por cobinação, onde a ordem não é importante, temos:



Valores e parâmetros alvo de testes:



Utilizando a ferramenta Allpairs temos

:

## **2.6 Burndown Chart**

O *Burndown chart* indica o progresso do desenvolvimento durante uma *Sprint*. É através dele que é medido o andamento das atividades. O *Burndown chart* é atualizado diariamente pelo Scrum master e, ao final da Sprint, além de saber se as atividades foram realizadas, pode-se observar também o desempenho e organização da equipe através da cadência com que as atividades são encerradas.

A figura abaixo mostra o *burndown chart* do TS#06 no final da Sprint 3.

**COLOCAR BURNDOWN CHART FINAL AQUI!!!**

Figura 08 - Burndown Chart Sprint #3 TS#06

## **2.7 Kanban**

Para realização do Kanban, o TS#06 utilizou um conjunto de cartões no trello:

* "Sprint#03 Backlog", com as *User Stories* e as tarefas gerais para o Sprint 3;
* Na categoria "To Do", as User Stories foram divididas em tasks e atreladas a um ou mais membros do time cuja responsabilidade é garantir a execução das tarefas conforme definido no *Definition of Done*.;
* As tarefas em "Doing", indicam que a tarefa já foi iniciada;
* A última categoria ("Done") indica que uma tarefa já foi realizada e testada para o sprint atual.

**COLOCAR PRINT ATUALIZADO DO TRELLO AQUI!!!**

Figura 09 - Kanban Sprint #3 TS#06

# **3 Síntese das Atividades Individuais Realizadas no Sprint**

# 

# 4 Recomendações de melhorias das disciplinas ministradas no 1o Semestre de 2016

# 

# Referências

Projeto BD-ITAC, 2016. <https://sites.google.com/site/interdproj2016/>

Projeto BDIC-DM, 2015. <https://sites.google.com/site/projetobdic32/>

Apache Jmeter. <http://jmeter.apache.org/>

CassandraStress.<https://docs.datastax.com/en/cassandra/2.1/cassandra/tools/toolsCStress_t.html>

Pairwise Testing. <http://pairwise.org/>

Allpairs Test Case Generation Tool. http://www.satisfice.com/tools.shtml

Testlink Project. http://testlink.org/

**Apêndices**