Hive反模式的检测与修复

软件需求规约

版本 1.0

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 9/11/2020 | 1.0 | 建立需求分析文档 | 任姚丹珺, 贾兴国 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[1. 简介 4](#_Toc55853048)

[1.1 目的 4](#_Toc55853049)

[1.2 定义、首字母缩写词和缩略语 4](#_Toc55853050)

[1.3 参考资料 5](#_Toc55853051)

[2. 整体说明 5](#_Toc55853052)

[3. 功能需求 5](#_Toc55853053)

[3.1 <Use case 图> 5](#_Toc55853054)

[3.2 <Use case1 规约> 5](#_Toc55853055)

[3.3 <Use case2 规约> 5](#_Toc55853056)

[4. 非功能需求 6](#_Toc55853057)

[4.1 易用性 6](#_Toc55853058)

[4.2 可靠性 6](#_Toc55853059)

[4.3 性能 6](#_Toc55853060)

[4.4 可支持性 7](#_Toc55853061)

[4.5 设计约束 7](#_Toc55853062)

[5. 其它产品需求 7](#_Toc55853063)

[5.1 联机用户文档和联机帮助的需求 7](#_Toc55853064)

[5.2 接口需求 7](#_Toc55853065)

[5.2.1 用户界面 7](#_Toc55853066)

[5.2.2 硬件接口 7](#_Toc55853067)

[5.2.3 软件接口 7](#_Toc55853068)

[5.2.4 通信接口 7](#_Toc55853069)

[5.3 适用的标准 7](#_Toc55853070)

软件需求规约 (简化版)

# 简介

## 目的

本文档从软件需求的角度对系统进行综合概述。编写本文档是为了明确用户对Hive反模式检测与修复系统的功能需求和非功能性需求,并将这些需求用规范化的语言和规范化的结构完整、准确地表达清楚，同时提供全面的设计资料与设计思路，采用精确的软件框架结构设计。本文档用于记录并表述对系统的需求方面做出的重要决策，它对开发的后续阶段性工作起着指导作用。本文档的编写遵循IEEE需求文档的规范(ANSI/IEEE Std. 830-1984)。

本文档的预期读者对象为：

a) 最终用户：了解预期项目的功能和性能，并与开发方人员一起对整个需求进行讨论和协商。

b) 开发人员：根据该文档了解预期项目的功能，并据此进行项目策划、概要设计、详细设计，确定软件系统的体系结构、组成成分、数据组织、模块、内外部接口

c) 测试人员：用规范化的语言和规范化的结构描述的需求为测试提供了依据，从而有根据地对软件产品进行功能性测试和非功能性测试。

d) 其它相关人员：如用户文档编写人员、项目管理人员、项目运维人员等。

e) 老师（本项目董事会成员）：会对本文档提出一些修改建议以及点评。

## 定义、首字母缩写词和缩略语

表格1-术语定义表

|  |  |
| --- | --- |
| **术语** | **定义** |
| 用户角色 | 用户角色是指按照一定参考体系划分的用户类型，是能够代表某种用户特征、便于统一描述的众多用户个体的集合。 |
| 反模式 | 反模式指用来解决问题的带有共同性的不良方法。它们已经经过研究并分类，以防止日后重蹈覆辙，并能在研发尚未投产时辨认出来。 |
| HiveQL | hive是基于Hadoop构建的一套数据仓库分析系统，它将SQL语句转换为MapReduce任务运行，通过自己的SQL查询分析需要的内容 |
| 功能性需求 | 功能性需求规定开发人员必须在产品中实现的软件功能，用户利用这些功能来完成任务，满足业务需求。 |
| 非功能性需求 | 非功能性需求是指依一些条件判断系统运作情形或其特性，而不是针对系统特定行为的需求。 |
| 功能结构图 | 功能结构图是将系统的功能进行分解，按功能从属关系表示的图表 |
| 用例图（User Case） | 用例图是指由参与者（Actor）、用例（Use Case）以及它们之间的关系构成的用于描述系统功能的静态视图。 |

## 参考资料

[1] IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998.

[2] 王安生. 软件工程化[M]. 北京：清华大学出版社,2014.

[3] 叶伟剑.浅谈软件需求分析过程[J].大众科技.2005(04)

[4] 张博.ANTLR 4权威指南[M]。北京：机械工业出版社，2017.

# 整体说明

在这一章节中，会对Hive反模式的检测与修复系统进行概括性的描述，包括系统功能的初步概况、用户角色的划分以及对系统的假设与约束。

## 产品总体效果

Hive反模式的检测与修复系统是为了方便程序员对自己编写的HiveQL的正确性以及性能进行检测而开发的工具。目的是通过反模式的静态检测和动态监测，精准定位出HiveQL中的不良代码，并给出恰当的修改意见，从而方便程序员即使在工期紧张时，也能快速生产出高质量的代码。

如图1所示，本系统的用户在登录系统后，可以方便地上传HiveQL代码，并获取反模式检测与修复的分析结果。基本业务流程是：用户登录系统后，上传HiveQL代码，系统会同时执行两项工作：

1. 将代码发送到Hive集群真实地运行，通过跟踪任务运行时的log信息，获得anti-pattern的动态检测结果。
2. 生成HiveQL对应的AST树，通过规则匹配检测，获得静态检测结果。

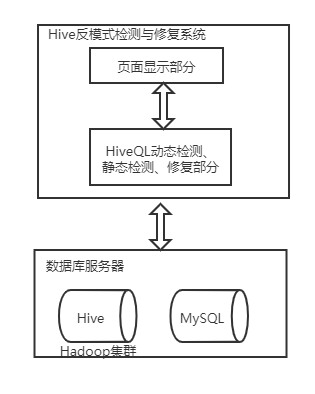
在获得静态检测和动态检测的anti-pattern检测结果后，系统会自动修复上传的HiveQL代码。

图 1 系统块图

## 产品功能

Hive反模式检测与修复系统可以对程序员提交的HiveQL代码进行实时的检测与修复，减少了人工检查代码错误并修复的时间，并能够提高代码质量。本系统主要支持的功能有：

1. 用户通过交互式页面提交HiveQL代码，请求静态检测并修复；
2. 用户通过交互式页面提交HiveQL代码，请求动态监测并修复；
3. 用户对数据库相关的配置项做出设置；
4. 用户获得系统返回的检测与修复结果。

## 用户特征

表 2-用户特征表

|  |  |
| --- | --- |
| **用户角色名称** | **特性** |
| 程序员 | 使用HiveQL读写Hadoop集群数据的程序员，可以上传HiveQL代码、设置数据库的配置项、获得系统的检测与修复结果。 |

## 约束

本项目在开发和运行的全过程中，严格遵循相关法律及政策。

开发本项目时，集群运行在实验室服务器上，cpu为 Intel(R) Xeon(R) W-2140B CPU @ 3.20GHz，内存128g，Hadoop版本2.7.4，hive版本为2.3.5，使用SQL存储hive元数据。

本项目用到的开发工具为Intellig IDEA，前端使用VUE进行编写，使用HTML、JavaScript、CSS语言。后端使用Java语言进行编写。

本系统数据加密传输，系统页面通过浏览器进行展示，本系统能够支持IE10以上、火狐、谷歌、搜狗浏览器。

## 假设与依赖关系

1. 本系统可以通过修改配置项，修改Hadoop数据库集群的配置。
2. 本系统能在30s内给出静态检测结果与修复，对动态检测，时长会根据实际代码变化，最大不超过180s。
3. 本项目能通过提交HiveQL代码到集群上运行，跟踪和检测代码的动态执行结果，从而根据输出的log对反模式进行动态检测。
4. 本项目在静态检测时仅关注HiveQL代码的AST树匹配，如果HiveQL代码无法被正确解析，则不在本系统讨论范围内。
5. 系统在正式使用前对需求分析中的各指标进行充分和完全的调试。
6. 系统在开发过程中严格遵循本文档要求。
7. 本系统的最大用户并发量不超过200。

# 功能需求

## Use case 图

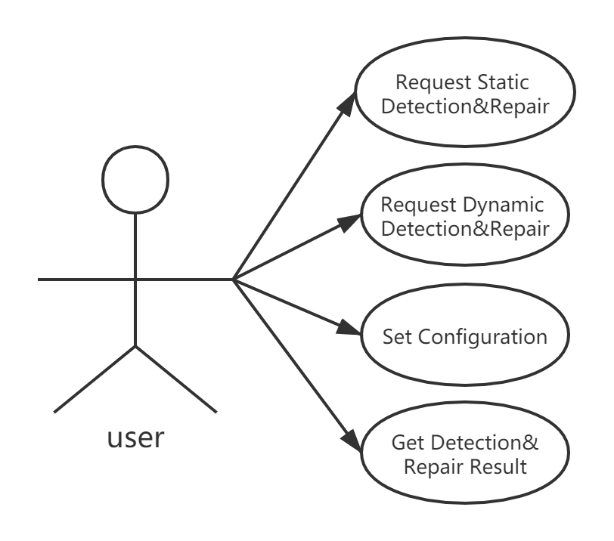


图 2 系统用例图

## Request Static Detection & Repair

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | UC01 | 用例名称 | Request Static Detection & Repair |
| 描述 | 请求反模式的静态检测与修复 | | |
| 执行者 | 程序员 | | |
| 前置条件 | 程序员用户已登陆 | | |
| 后置条件 | 用户获取静态检测修复结果 | | |
| 基本流 | 1. 用户选择静态检测功能 2. 系统跳转到静态检测页面 3. 用户复制粘贴HiveQL代码 4. 用户点击确定按钮提交HiveQL代码到静态检测系统 5. 系统提示提交静态检测成功 | | |
| 备选流 | 1a. 未查询到该用户，系统提示相关信息  3a. 用户粘贴的HiveQL代码不符合语法规范，系统显示相应的提示信息  4a. 提交检测失败，系统显示提交失败原因  5a. 系统提示提交静态检测失败 | | |

## Request Dynamic Detection & Repair

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | UC02 | 用例名称 | Request Dynamic Detection & Repair |
| 描述 | 请求反模式的动态检测与修复 | | |
| 执行者 | 程序员 | | |
| 前置条件 | 程序员用户已登陆 | | |
| 后置条件 | 用户获取动态检测修复结果 | | |
| 基本流 | 1. 用户选择动态检测功能 2. 系统跳转到动态检测界面 3. 用户复制粘贴HiveQL代码 4. 用户点击确定按钮提交HiveQL代码到动态检测系统 5. 系统提示提交动态检测成功 | | |
| 备选流 | 1a. 未查询到该用户，系统提示相关信息  3a. 用户粘贴的HiveQL代码不符合语法规范，系统显示相应的提示信息  4a. 提交检测失败，系统显示提交失败原因  5a. 系统提示提交动态检测失败 | | |

## Set Configuration

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | UC03 | 用例名称 | Set Configuration |
| 描述 | 设定数据库服务器相关配置项 | | |
| 执行者 | 程序员 | | |
| 前置条件 | 程序员用户已登陆 | | |
| 后置条件 | 用户获取动态检测修复结果 | | |
| 基本流 | 1. 用户选择设置配置项功能 2. 系统跳转到修改配置项界面 3. 用户填写需要修改的配置项 4. 用户点击确定按钮将修改 5. 系统修改数据库服务器中的相关配置项 | | |
| 备选流 | 1a. 未查询到该用户，系统提示相关信息  3a. 用户对配置项作出的修改不符合规范，系统显示相应的提示信息  4a. 提交配置项修改失败，系统显示提交失败原因  5a. 系统修改数据库服务器中的相关配置项失败 | | |

## Get Detection & Repair Results

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | UC04 | 用例名称 | Get Detection & Repair Results |
| 描述 | 获取检测与修复结果 | | |
| 执行者 | 程序员 | | |
| 前置条件 | 程序员用户已登陆，检测与修复已完成 | | |
| 后置条件 | 用户获取到检测与修复姐 | | |
| 基本流 | 1. 用户选择获取检测与修复结果功能 2. 系统跳转至检测与修复结果页面 3. 用户选择需要查看的检测工作项 4. 系统显示对应的检测与修复结果 | | |
| 备选流 | 1a. 未查询到该用户，系统提示相关信息  3a. 用户选择的工作项还没有完成，系统显示相应提示  4a. 系统未检测到反模式，用户提交的代码没有反模式  4b. 系统检测反模式失败，提示失败原因 | | |

# 非功能需求

## 易用性

[此节应包括所有影响易用性的需求。例如，

•指出普通用户和高级用户要高效地执行特定操作所需的培训时间

•指出典型任务的可评测任务次数或根据用户已知或喜欢的其他系统确定新系统的易用性需求

•指出在符合公认的易用性标准（如 IBM 的 CUA 标准和 Microsoft 的 GUI 标准）方面的需求]

## 可靠性

[对系统可靠性的需求应在此处说明。以下是一些建议：

• 可用性—指出可用时间百分比 ( xx.xx%)、使用小时数、维护访问权、降级模式操作等。

• 平均故障间隔时间 (MTBF) – 通常表示为小时数，但也可表示为天数、月数或年数。

• 平均修复时间 (MTTR) — 系统在发生故障后可以暂停运行的时间。

• 精确度 — 指出系统输出要求具备的精密度（分辨率）和精确度（按照某一已知的标准）。

• 最高错误或缺陷率—通常表示为每千行代码的错误数目 (bugs/KLOC) 或每个功能点的错误数目 (bugs/function-point)。

• 错误或缺陷率—按照小错误、大错误和严重错误来分类。需求中必须对“严重”错误进行界定，例如：数据完全丢失或完全不能使用系统的某部分功能。]

## 性能

[此节应概述系统的性能特征。其中需包括具体的响应时间。

• 对事务的响应时间（平均、最长）

• 吞吐量，例如每秒处理的事务数

• 容量，例如系统可以容纳的客户或事务数

• 降级模式（当系统以某种形式降级时可接受的运行模式）

• 资源利用情况，如内存、磁盘、通信等]

## 可支持性

[此节应列出将提高所构建系统的可支持性或可维护性的所有需求，其中包括编码标准、命名约定、类库、维护访问权和维护实用程序。]

## 设计约束

[此节应列出所构建系统的所有设计约束。设计约束代表已经批准并必须遵循的设计决定。其中包括软件语言、软件流程需求、开发工具的指定用途、构架及设计约束、购买的构件、类库等。]

# 其它产品需求

## 联机用户文档和联机帮助的需求

[如果存在对联机用户文档、帮助系统、关于声明的帮助等的需求，请在此说明。]

## 接口需求

[此节规定应用程序必须支持的接口/界面。它应非常具体，包含协议、端口和逻辑地址等，以便于按照接口/界面需求开发并检验软件。]

### 用户界面

[c。]

### 硬件接口

[此节指出软件所支持的所有硬件接口，其中包括逻辑结构、物理地址、预期行为等。]

### 软件接口

[此节说明软件系统中与其他构件之间的软件接口。这些构件可以是购入的构件、取自其他应用程序重新利用的构件，也可以是为此 **SRS** 范围之外的子系统开发，但该软件应用程序必须与之交互的构件。]

### 通信接口

[说明与其他系统或设备（如局域网、远程串行设备等）的所有通信接口。]

## 适用的标准

[通过引用，此节说明了所有适用的标准以及适用于所述系统的相应标准的具体部分。例如，其中可以包括法律、质量及法规标准；业界在易用性、互操作性、国际化、操作系统相容性等方面的标准。]