Atitit 不错的ql选型 jpql

目录

[1. 目前jpql存在的问题与解决 1](#_Toc14906)

[2. QL的解析流程 1](#_Toc8348)

[2.1. Atitit 常见数据查询语言 GraphQL sql 1](#_Toc3029)

[2.2. Sql比较不错简单快速，但是对于后端是数据库的系统来说，安全是问题 2](#_Toc26427)

[2.3. JPQL和apijson是个不错的方案 2](#_Toc21836)

[3. 范例 2](#_Toc17409)

[3.1. Jpql转sql 2](#_Toc24600)

[4. 实现机制 3](#_Toc23131)

[4.1. 解析QL ast （jqpl hql） 3](#_Toc10858)

[4.2. 转换sql ast 3](#_Toc15972)

[4.3. Sql ast 生成sql 4](#_Toc29683)

[5. 其他ql模式 5](#_Toc17667)

[5.1. 受限sql 5](#_Toc27148)

[5.2. jpql over http param 注解模式 5](#_Toc10740)

[5.3. Apijson 5](#_Toc13863)

[6. 安全提示措施 5](#_Toc25338)

# 目前jpql存在的问题与解决

目前jpql的hb实现的比较繁琐，必须要预先配置数据源，且需要配合实体类。。。

需要解除与实体类的绑定，直接ast转换，转换生成sql

# QL的解析流程

Ql》》QL AST>>SQL AST>>SQL

这样经过ql 和sql ast双重解析过滤，比起直接使用sql安全性基本得到保障

可以sql也再次经过安全解析，进一步提升安全

## Atitit 常见数据查询语言 GraphQL sql

目录

1. 数据查询语言(DQL)、 1

1.1. Atitit jqpl的解析与相关类 1

1.2. Hql 1

1.3. JPQL 1

1.4. 认识APIJSON （推荐） 1

1.5. sql 3

1.6. Graphql 3

2. ref 3

## Sql比较不错简单快速，但是对于后端是数据库的系统来说，安全是问题

如果后端是web爬虫，或者es的化，文件系统，或其他非数据库类存储，是可以直接使用sql的

Graphql有些麻烦，资料相对不多，比sql资料少很多

## JPQL和apijson是个不错的方案

# 范例

## Jpql转sql

package org.chwin.firefighting.apiserver.QL;  
  
import lombok.SneakyThrows;  
import org.chwin.firefighting.apiserver.QL.JqplUtil;  
  
public class jqplTest {  
@SneakyThrows  
 public static void main(String[] args) {  
  
  
 // System.out.println(JqplAstTraver.Jqpl2sql(hqlAst)); and c3=256  
 // and c2=156 and c2=456 and c3=789 and c4=012  
  
 String hql = " from 表格1 where 字段1=123 ";  
 JqplUtil.*showAst*(hql);  
 System.*out*.println(JqplUtil.*Jqpl2sql*(hql));

输出

SELECT \* FROM 表格1 WHERE 字段1 = 123

# 实现机制

## 解析QL ast （jqpl hql）

\-[QUERY] Node: 'query'

+-[SELECT\_FROM] Node: 'SELECT\_FROM'

| \-[FROM] Node: 'from'

| \-[RANGE] Node: 'RANGE'

| \-[IDENT] Node: '\u8868\u683c1'

\-[WHERE] Node: 'where'

\-[EQ] Node: '='

+-[IDENT] Node: '\u5b57\u6bb51'

\-[NUM\_INT] Node: '123'

## 转换sql ast

D:\0wkspc\platform-top-run-coffey\platform-top-service\platform-top-service-finance\src\main\java\org\chwin\firefighting\apiserver\QL\ASTPrinterAti.java

public String nodeToString(AST ast) {  
 if (ast == null) {  
 return "{node:null}";  
 } else {  
 StringBuilder buf = new StringBuilder();  
 String tokenTypeName = this.getTokenTypeName(ast.getType());  
 JqplAstTraver1.processTokenTypename(tokenTypeName);  
  
  
  
  
 buf.append("[").append(tokenTypeName).append("] ");  
 buf.append(StringHelper.*unqualify*(ast.getClass().getName())).append(": ");  
 buf.append("'");  
 String text = ast.getText();  
 if (text == null) {  
 text = "{text:null}";  
 }  
  
  
  
 JqplAstTraver1. processByTokentypenameAndTxt(tokenTypeName, text);

## Sql ast 生成sql

SELECT \* FROM 表格1 WHERE 字段1 = 123

public JqplAstTraver() {  
  
  
 select\_where\_and.setSelectBody(new PlainSelect() {{  
  
 setSelectItems(new ArrayList() {{  
 add("\*");  
 }});  
 // setFromItem(new Table("TABLE1"));  
// setWhere(  
// new AndExpression(new EqualsTo() {{  
// setLeftExpression(new Column("c1"));  
// setRightExpression(new StringValue("aaa"));  
// }}, new EqualsTo() {{  
// setLeftExpression(new Column("c2"));  
// setRightExpression(new LongValue(789));  
// }}) {{  
//  
// }}  
// );  
 }}

# 其他ql模式

## 受限sql

限制其可以访问的数据库，表，sp，系统函数等。。只允许执行白名单内的东东 提升安全性

操作与访问重要或私人数据，必须提供userid，登录状态下使用。避免权限问题

## jpql over http param 注解模式

[http://api?@from=tab11&@where= 列名1:123](http://api?@from=tab11&@where=列名1=123) &@and= 列名2:456

注意urlencode

可读性更好的汉字模式,默认为查询操作

http://api?@操作表=用户表&@条件= 列名1:123 &@and= 列名2:456

http://api?@删除操作&@操作表=表1&@条件=记录id:123

http://api?@更新操作&@操作表=公告表& 标题=45666 @条件=id:123

## Apijson

# 安全提示措施

QL更加适合于查询与简单业务操作

重要业务操作，特别是重要update最好使用隐藏操作，不使用QL，只给前端提供SELECTID号和参数