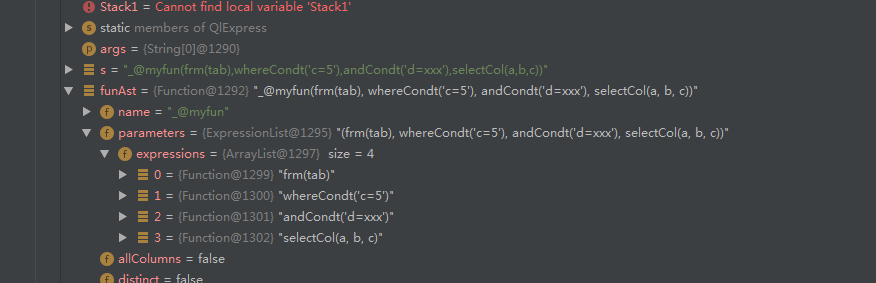
Atitit 表达式QL解析ast法



## 使用sqlparse解析器 可以吧函数式表达。

有点，支持函数，支持常见sql表达式

。问题，不能使用sql专用解析关键词

From where and 等。。

Atitit fastjson 过滤相关字段

package org.chwin.firefighting.apiserver.QL;  
  
import com.alibaba.fastjson.JSON;  
import com.alibaba.fastjson.serializer.SerializerFeature;  
import com.alibaba.fastjson.serializer.SimplePropertyPreFilter;  
import lombok.SneakyThrows;  
import net.sf.jsqlparser.expression.Function;  
import net.sf.jsqlparser.parser.CCJSqlParserUtil;  
import net.sf.jsqlparser.statement.Statement;  
import net.sf.jsqlparser.statement.select.PlainSelect;  
import net.sf.jsqlparser.statement.select.Select;  
import net.sf.jsqlparser.statement.select.SelectExpressionItem;  
import net.sf.jsqlparser.statement.select.SelectItem;  
  
import java.util.List;  
  
public class QlExpress {  
 @SneakyThrows  
 public static void main(String[] args) {  
 String s = "f1('abc',f2(123,456))";  
 String sql = "select " + s;  
 Statement Statement1 = CCJSqlParserUtil.*parse*(sql);  
 Select select = (Select) Statement1;  
 List<SelectItem> li = ((PlainSelect) select.getSelectBody()).getSelectItems();  
 SelectExpressionItem sei = (SelectExpressionItem) li.get(0);  
 Function funAst = (Function) sei.getExpression();  
 SimplePropertyPreFilter filter = new SimplePropertyPreFilter(Function.class, "name","parameters");  
 System.*out*.println(JSON.*toJSONString*(funAst, filter,SerializerFeature.*PrettyFormat*));  
 }  
}

"C:\Program Files\Java\jdk1.8.0\_211\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2019.1.2\lib\idea\_rt.jar=54749:C:\Program Files\JetBrains\IntelliJ IDEA 2019.1.2\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\Owner\AppData\Local\Temp\classpath1100288608.jar org.chwin.firefighting.apiserver.QL.QlExpress

{

"name":"f1",

"parameters":{

"expressions":[

{

"notExcapedValue":"abc",

"value":"abc"

},

{

"name":"f2",

"parameters":{

"expressions":[

{

"stringValue":"123",

"value":123

},

{

"stringValue":"456",

"value":456

}

]

}

}

]

}

}

package org.chwin.firefighting.apiserver.QL;  
  
import com.alibaba.fastjson.JSON;  
import com.alibaba.fastjson.serializer.SerializerFeature;  
import com.alibaba.fastjson.serializer.SimplePropertyPreFilter;  
import lombok.SneakyThrows;  
import net.sf.jsqlparser.JSQLParserException;  
import net.sf.jsqlparser.expression.Function;  
import net.sf.jsqlparser.parser.CCJSqlParserUtil;  
import net.sf.jsqlparser.statement.Statement;  
import net.sf.jsqlparser.statement.select.PlainSelect;  
import net.sf.jsqlparser.statement.select.Select;  
import net.sf.jsqlparser.statement.select.SelectExpressionItem;  
import net.sf.jsqlparser.statement.select.SelectItem;  
import org.jetbrains.annotations.NotNull;  
  
import java.util.List;  
  
public class QlExpress {  
 @SneakyThrows  
 public static void main(String[] args) {  
 String s = "f1('abc',f2(123,456))";  
 Function funAst = *getFunctionAst*(s);  
 String x = *toJsonString*(funAst);  
 System.*out*.println(  
 x);  
  
 }  
  
 @NotNull  
 private static String toJsonString(Function funAst) {  
 SimplePropertyPreFilter filter = new SimplePropertyPreFilter(Function.class, "name", "parameters");  
 return JSON.*toJSONString*(funAst, filter, SerializerFeature.*PrettyFormat*);  
 }  
  
 private static Function getFunctionAst(String s) throws JSQLParserException {  
 String sql = "select " + s;  
 Statement Statement1 = CCJSqlParserUtil.*parse*(sql);  
 Select select = (Select) Statement1;  
 List<SelectItem> li = ((PlainSelect) select.getSelectBody()).getSelectItems();  
 SelectExpressionItem sei = (SelectExpressionItem) li.get(0);  
 return (Function) sei.getExpression();  
 }  
}