Java查询-解析xml生成报表

1.需求背景

1.有些客户的系统里会通过存储过程等生成一些xml文件，客户需要解析这些xml文件生成报表。

2.xml文件层级格式必须规定好（该文档是基于2层结构的xml解析的-test.xml）。

3.该示例仅供参考，具体代码需根据xml格式稍作修改。

Java查询接口说明请见：IJavaQueryData。

注意：本示例来源于实际项目，原始版本是V6.1，可作为参考性质，也许是不能运行的。

2.部署扩展包

2.1.加载扩展包

readxml.ext，扩展包部署见扩展包部署。

2.2.扩展包代码

2.2.1 JavaQueryReadXml类

使用JAVA查询解析xml。

JavaQueryReadXml类

```
package cn.com.smartbi;

import java.io.*;
import java.util.*;
import java.util.Map.Entry;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import org.w3c.dom.*;
import smartbi.SmartbiException;
import smartbi.freequery.FreeQueryErrorCode;
import smartbi.freequery.metadata.IJavaQueryData;
import smartbi.freequery.metadata.JavaQueryConfig;
import smartbi.freequery.metadata.JavaQueryOutputField;
import smartbi.freequery.metadata.JavaQueryParameter;
import smartbi.freequery.querydata.CellData;
import smartbi.freequery.querydata.GridData;
import smartbi.net.sf.json.JSONObject;
import smartbi.util.StringUtil;
import smartbi.util.ValueType;
public class JavaQueryReadXml implements IJavaQueryData {
        private Map<String, JavaQueryConfig> configs = new LinkedHashMap<String, JavaQueryConfig>();
        private List<JavaQueryOutputField> outputFields;
        Element element = null;
        DocumentBuilder db = null;
        DocumentBuilderFactory dbf = null;
        public JavaQueryReadXml() {
                addConfig("FileName", "XML", "","", true);
                addConfig("parentNode", "", ":/xx/xx...", "", true);
        }
        private void addConfig(String name, String alias, String desc,
                        String defaultValue, boolean notNull) {
                JavaQueryConfig p = new JavaQueryConfig();
                p.setName(name);
                p.setAlias(alias);
                p.setDesc(desc);
                p.setValue(defaultValue);
```

```
        p.setNotNull(notNull);
        configs.put(name, p);
}
@Override
public List<JavaQueryConfig> getConfigs() {
        return new ArrayList<JavaQueryConfig>(configs.values());
}
@Override
public GridData getGridData(int arg0, int arg1) {
        Element parentNode = getParentNode(configs.get("parentNode").getValue());
        NodeList nodelist;
        List<List<CellData>> datas = new ArrayList<List<CellData>>();
        if(parentNode.getParentNode()==null||parentNode.getParentNode().getNodeType()==9){
                nodelist = parentNode.getChildNodes();
                 List<CellData> row = new ArrayList<CellData>();
                 for(JavaQueryOutputField j:outputFields){
                                for(int k = 0;k<nodelist.getLength();k++){
                                        if(nodelist.item(k).getNodeName() == j.getId()){
                                                CellData c = new CellData();
                                                c.setStringValue(nodelist.item(k).getTextContent());
                                                row.add(c);
                                        }
                                }
                        }
                 datas.add(row);
        }else{
                Element parent_parentNode = (Element) parentNode.getParentNode();
                NodeList pChildren = parent_parentNode.getChildNodes();
                nodelist =  parent_parentNode.getElementsByTagName(parentNode.getNodeName());
                for(int i = 0;i<nodelist.getLength();i++){
                        List<CellData> row = new ArrayList<CellData>();
                        NodeList childlist = nodelist.item(i).getChildNodes();
                        for(JavaQueryOutputField j:outputFields){
                                for(int k = 0;k<pChildren.getLength();k++){
                                        if(pChildren.item(k).getNodeName() == j.getId()){
                                                CellData c = new CellData();
                                                c.setStringValue(pChildren.item(k).getTextContent());
                                                row.add(c);
                                        }
                                }
                                for(int k = 0;k<childlist.getLength();k++){
                                        if(childlist.item(k).getNodeName() == j.getId()){
                                                CellData c = new CellData();
                                                c.setStringValue(childlist.item(k).getTextContent());
                                                row.add(c);
                                        }
                                }
                        }
                        datas.add(row);
                }
        }

        GridData d = new GridData();
        List<String> headers = new ArrayList<String>();
        for (JavaQueryOutputField of : outputFields)
                headers.add(of.getName());
        d.setStringHeaders(headers);
        d.setData(datas);
        return d;
}
@Override
public List<JavaQueryOutputField> getOutputFields() {
        return outputFields;
}
@Override
public List<JavaQueryParameter> getParameters() {
        return new ArrayList<JavaQueryParameter>();
}
@Override
public int getRowCount() {
        return Integer.MAX_VALUE;
```

```
	}
	@Override
	public void init() {
		Element parentNode = getParentNode(configs.get("parentNode").getValue());
		if(parentNode==null){
			throw new SmartbiException(FreeQueryErrorCode.JAVA_QUERY_DEFINE_CLASS_ERROR).
setDetail("");
		}
		NodeList childNodes = parentNode.getChildNodes();
		outputFields = new ArrayList<JavaQueryOutputField>();
		if(parentNode.getParentNode()==null||parentNode.getParentNode().getNodeType()==9){
			addOutput(childNodes);
		}else{
			Element pNode = (Element) parentNode.getParentNode();
			NodeList pChildren = pNode.getChildNodes();
			addOutput(pChildren);
			addOutput(childNodes);
		}
	}
	@Override
	public void loadConfigs(String configStr) {
		if (StringUtil.isNullOrEmpty(configStr))
			return;
		JSONObject obj = JSONObject.fromString(configStr);
		configs.get("FileName").setValue(
				obj.has("FileName") ? obj.getString("FileName") : null);
		configs.get("parentNode").setValue(
				obj.has("parentNode") ? obj.getString("parentNode") : null);
	}
	@Override
	public String saveConfigs() {
		JSONObject json = new JSONObject();
		for (JavaQueryConfig config : configs.values())
			json.put(config.getName(), config.getValue());
		return json.toString();
	}
	@Override
	public void setConfigValue(String key, String value) {
		configs.get(key).setValue(value);
	}
	@Override
	public void setConfigValues(Map<String, String> configValues) {
		for (Entry<String, String> config : configValues.entrySet())
			configs.get(config.getKey()).setValue(config.getValue());
	}
	@Override
	public void setParameterValue(String arg0, String arg1, String arg2) {
	}

	@Override
	public void close() {
	}

	public void addOutput(NodeList childNodes){
		for(int j = 0; j < childNodes.getLength(); j++){
			Node e = childNodes.item(j);
			if (e.getNodeType() == 1 && isValidateNode(e)) {
				JavaQueryOutputField jf = new JavaQueryOutputField();
				jf.setId(e.getNodeName());
				jf.setName(e.getNodeName());
				jf.setAlias(e.getNodeName());
				jf.setDataType(ValueType.STRING);
				outputFields.add(jf);
			}
		}
	}
	//
	private boolean isValidateNode(Node e) {
		if (e.getChildNodes().getLength() == 1) {
			Node child_node = e.getChildNodes().item(0);
			if (child_node.getNodeType() == 3||child_node.getNodeType() == 4) { //4CDATA3text
```

```
                        return true;
                    } else {
                        return false;
                    }
                } else if(e.getChildNodes().getLength()==0){
                    return true;
                }else{
                    return false;
                }
        }
        //
        private Element getParentNode(String str) {
                File f = new File(configs.get("FileName").getValue());
                try {
                    dbf = DocumentBuilderFactory.newInstance();
                    // dbdocumentBuilderFatorydocumentBuildr
                    db = dbf.newDocumentBuilder();
                    // DOMdocument
                    Document dt = db.parse(f);
                    // element
                    element = dt.getDocumentElement();
                    if(str.indexOf("/") != -1){
                        String[] node_val = str.split("/");
                        Element parentNode = element;
                        for(int i = 0; i < node_val.length; i++){
                            NodeList childNodes = parentNode.getChildNodes();
                            for(int j = 0; j < childNodes.getLength(); j++){
                                if(node_val[i].equals(childNodes.item(j).getNodeName())){
                                    parentNode = (Element) childNodes.item(j);
                                    break;
                                }
                            }
                        }
                        return parentNode;
                    }else{
                        return element;
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                    return element;
                }
        }
}
```

2.2.2 JavaQueryDefine.js.patch

将上面创建的JAVA查询类加入到新建JAVA查询对象界面的**类名**选项，供选择。

JavaQueryDefine.js.patch

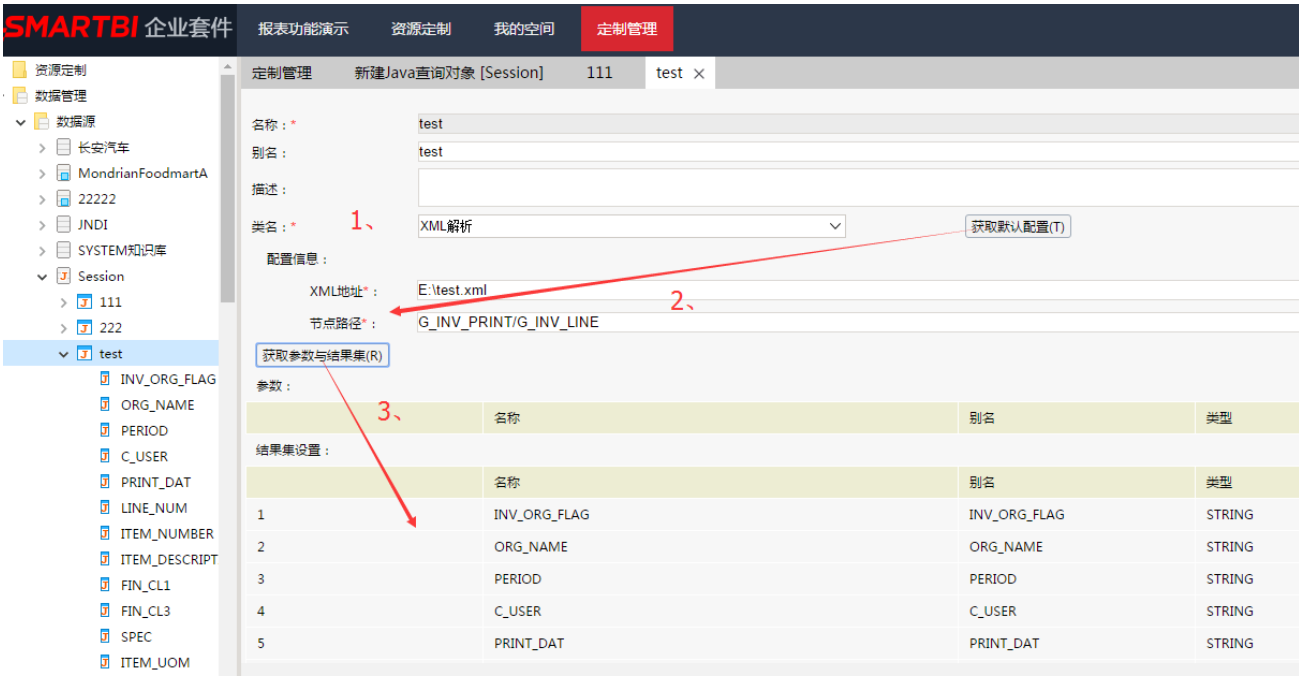
```
//
JavaQueryDefine.prototype.render_old = JavaQueryDefine.prototype.render;
JavaQueryDefine.prototype.render = function(id, dsId, queryId) {
        this.render_old(id, dsId, queryId);
        this.javaQueryClassName.insertItems([["cn.com.smartbi.JavaQueryReadXml","XML"]]);
}
```

效果图：

3. 生成报表

3.1. 配置java查询对象

如图：



第一步：新建java查询对象，类名选择XML解析

第二步：获取默认配置，填写xml地址和要解析的节点路径

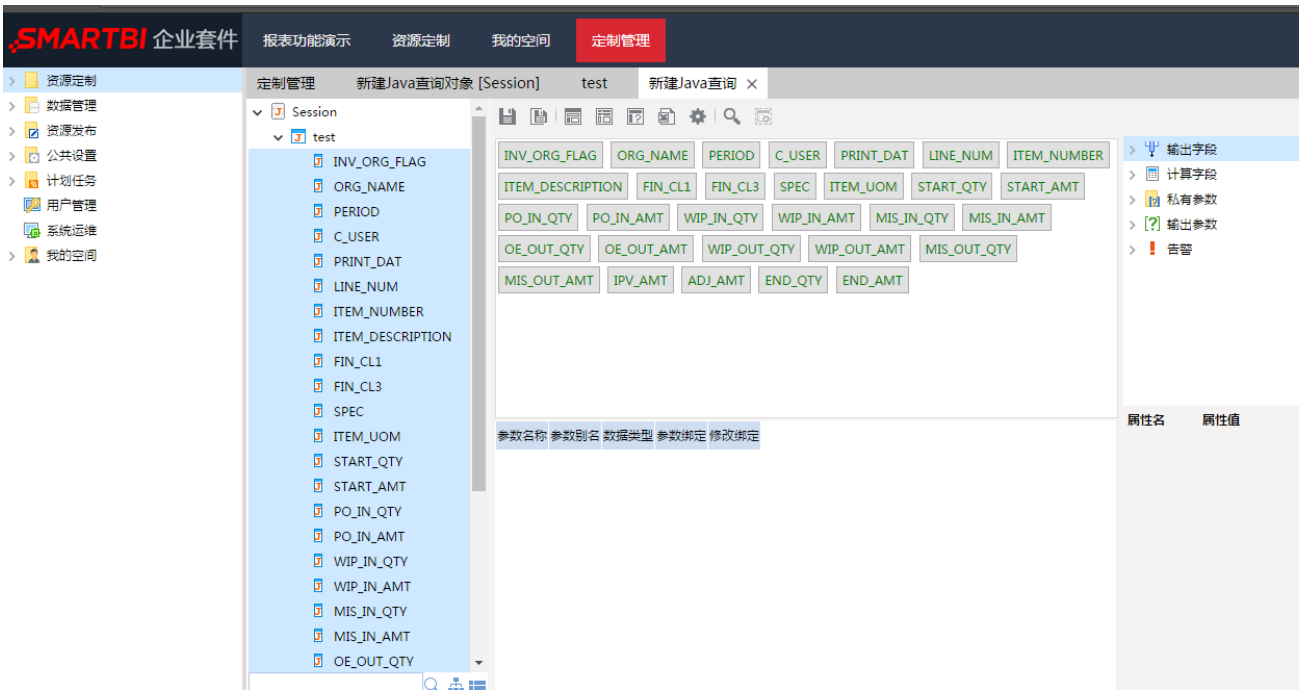
xml地址：将test.xml放到E盘目录下，地址为 E:\test.xml

节点路径：第一层数据为G_INV_PRINT节点，第二层数据为G_INV_LINE节点，要解析到第二层，故节点路径为 G_INV_PRINT/G_INV_LINE

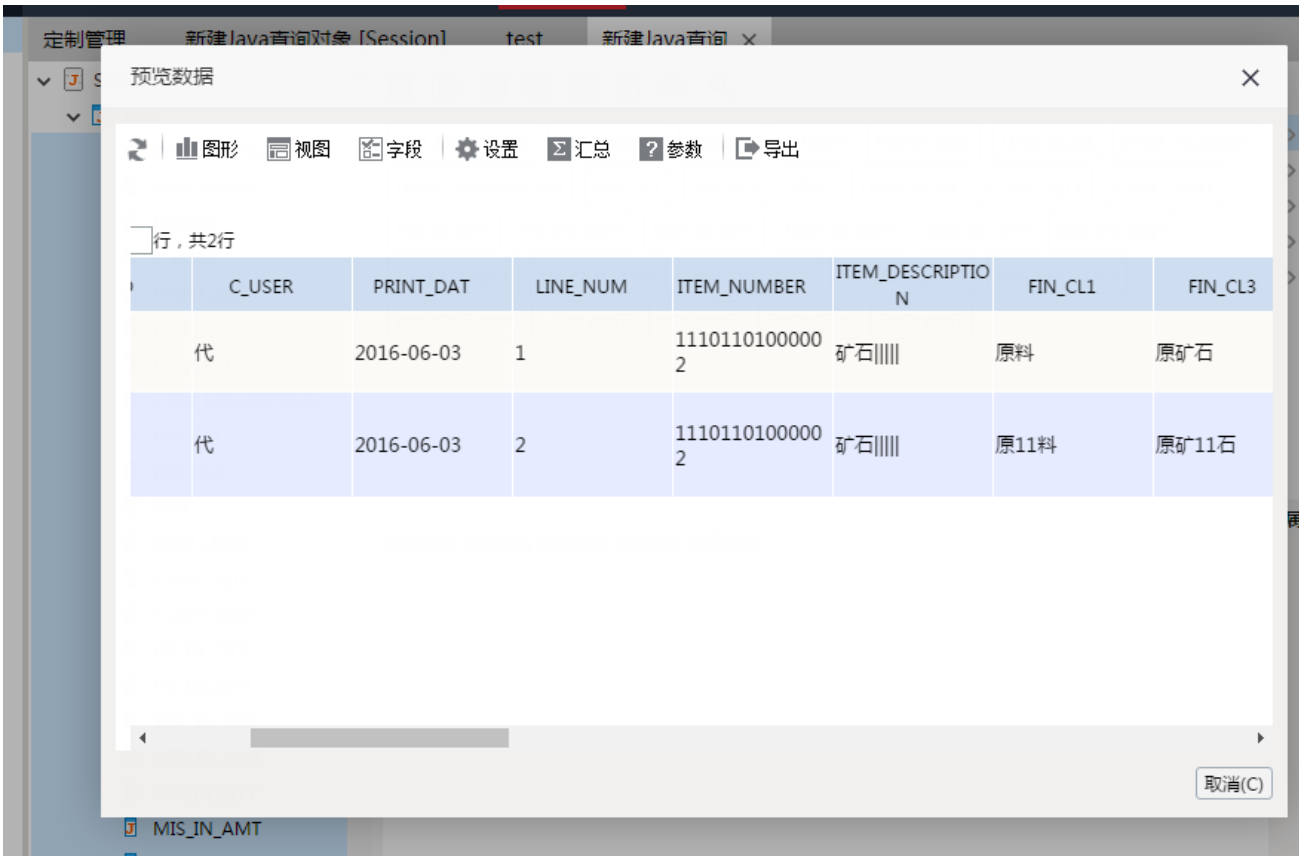
第三步：获取参数和结果集（由于该示例未添加参数，如需参数，可修改java查询类）

第四步：保存

3.2. 创建java查询数据集



3.3. 生成报表



4. 相关资源下载（EPPR-7727）

test.xml

readxml.ext

ReadXml.rar