MongoDB

-
-
-
-
-
-

MongoDBArrayObject

MongoDB8.4+

ColumnResolver

```
public interface ColumnResolver extends Mutable {

    String MARK_STRING = "ColumnResolver";

    int CURRENT_LEVEL = 1;


    boolean checkIfConditionMeet(ColumnResolver resolver);

    /**
     *
     *
     * @param cell doc
     * @return truefalse
     */
    boolean accept(Object cell);

    /**
     *
     *
     * @param key
     * @param cell
     */
    List<FieldObject> expandColumn(String key, Object cell);


    /**
     *
     *
     * @param doc                 MongoDB
     * @param columnNames
     * @param columnResolverMap
     * @param rowDataCollections
     * @param rowData
     */
    void expandData(Document doc, List<String> columnNames,
                    Map<FieldObject, ColumnResolver> columnResolverMap,
                    List<List<Object>> rowDataCollections,
                    List<Object> rowData);
}
```

JSON

```
{
    "name" : "",
    "age" : 15,
    "language" : [
        "",
        "",
        ""
    ],
    "region" : [
        "",
        "",
        ""
    ],
    "country" : "US"
}
```

languageregioncom.fr.plugin.db.mongo.expand.ColumnResolver

```
public class ArrayColumnResolver extends AbstractColumnResolver {

    @Override
    public boolean accept(Object cell) {
        return cell instanceof ArrayList
                && !(((ArrayList) cell).get(0) instanceof Document);
    }


    @Override
    public List<FieldObject> expandColumn(String key, Object cell) {
        List<FieldObject> data = new ArrayList<FieldObject>();
         data.add(FieldObject.create(key));
         return data;
    }

    @Override
    public void expandData(Document doc, List<String> columnNames, Map<FieldObject, ColumnResolver>
columnResolverMap, List<List<Object>> rowDataCollections, List<Object> rowData) {
        Map<Integer, List<Object>> group = new HashMap<Integer, List<Object>>();
        List<Object> standard = null;
        int maxLength = 0;
        for (Map.Entry<FieldObject, ColumnResolver> entry : columnResolverMap.entrySet()) {
            if (!checkIfConditionMeet(entry.getValue())) {
                continue;
            }
            FieldObject fieldObject = entry.getKey();
            Object data = doc.get(fieldObject.getOriginalName());
            if (data instanceof ArrayList) {
                List<Object> array = (List<Object>) data;
                if (standard == null) {
                    standard = array;
                }
                maxLength = Math.max(maxLength, array.size());
                group.put(fieldObject.getIndex(), array);
            }
        }

        if (standard != null) {
            for (int i = 0; i < maxLength; i++) {
                List<Object> row = new ArrayList<Object>(Arrays.asList(new Object[rowData.size()]));
                Collections.copy(row, rowData);
                for (FieldObject fieldObject : columnResolverMap.keySet()) {
                    List<Object> array = group.get(fieldObject.getIndex());
                    row.set(fieldObject.getIndex(), array.size() > i ? array.get(i) : null);
                }
                rowDataCollections.add(row);
            }
        }
    }
}
```

JSON


```
{
    "TEST_ID" : "8376",
    "Section_Name" : "",
    "ItemList" : [
        {
            "Item_Name" : "AAA",
            "Item_Value" : "123"
        },
        {
            "Item_Name" : "BBB",
            "Item_Value" : "456"
        },
        {
            "Item_Name" : "CCC",
            "Item_Value" : "789"
        }
    ]
}
```


2. TEST_ID(字符串)	3. Section_Name(字符串)	4. Item_Name(字符串)	5. Item_Value(字符串)
8376	大三李	AAA	123
8376	大三李	BBB	456
8376	大三李	CCC	789

com.fr.plugin.db.mongo.expand.ColumnResolver

JSON

```
public class ArrayMapColumnResolver extends AbstractColumnResolver {

    @Override
    public boolean accept(Object cell) {
        return cell instanceof ArrayList && ((ArrayList) cell).get(0) instanceof Document;
    }

    @Override
    public List<FieldObject> expandColumn(String key, Object cell) {
        List<FieldObject> expand = new ArrayList<FieldObject>();
        if (cell instanceof ArrayList) {
            Document document = (Document) ((ArrayList) cell).get(0);
            for (String name : document.keySet()) {
                expand.add(FieldObject.create(key, name));
            }
        }
        return expand;
    }

    @Override
    public void expandData(Document doc,
                           List<String> columnNames,
                           Map<FieldObject, ColumnResolver> columnResolverMap,
                           List<List<Object>> rowDataCollections,
                           List<Object> rowData) {
        boolean expanded = false;
        List<List<Object>> rowList = new ArrayList<List<Object>>();
        for (Map.Entry<FieldObject, ColumnResolver> entry : columnResolverMap.entrySet()) {
            if (!checkIfConditionMeet(entry.getValue())) {
                continue;
            }

            FieldObject fieldObject = entry.getKey();
            List<Document> children = (ArrayList<Document>) doc.get(fieldObject.getOriginalName());
            for (int i = 0, size = children.size(); i < size; i ++) {
                Document document = children.get(i);
                if (!expanded) {
                    List<Object> row = new ArrayList<Object>(Arrays.asList(new Object[rowData.size()]));
                    Collections.copy(row, rowData);
                    row.set(fieldObject.getIndex(), document.get(fieldObject.getCurrentName()));
                    rowDataCollections.add(row);
                    rowList.add(row);
                } else {
                    rowList.get(i).set(fieldObject.getIndex(), document.get(fieldObject.getCurrentName()));
                }
            }
            expanded = true;
        }
    }
}
```

```
<dependence>
    <Item key="com.fr.solution.plugin.db.mongo" type="plugin"/>
</dependence>
<extra-core>
    <ColumnResolver class="com.fr.plugin.db.mongo.expand.impl.ArrayColumnResolver"/>
</extra-core>
```

MongoDB