

[返回首页](#)

作者：许方友；时间：2024-07-24

## 功能的背景

维度成员允许设置 `自定义属性` 和 `标签`。

通过 `自定义属性` 和 `标签`，可以设置成员的自定义信息，并应用在其他功能上，例如 [缓慢变化](#) 等。

## 功能介绍

每个标准成员的 `自定义属性` 目前提供如下类型的自定义属性：

10 个 `int` 类型属性（`intC0`、`intC1...`）；

10 个 `string` 类型的属性（`stringC0`、`stringC1...`）；

32 个 `boolean` 类型的属性（`boolC0`、`boolC1...`）。

每个成员可以打上多个自定义的 `标签`。

明面上看，`自定义属性` 功能更加强大，可以涵盖 `标签` 的功能，而 `标签` 无法涵盖 `自定义属性` 的功能。

之所以提供两个功能，是考虑到 `自定义属性` 占用内存大，必须配合复杂的索引才能提供良好的性能，只能用在成员数据量比较小的情况；

而 `标签` 功能的特点是内存占用小，其内部存储天然就是索引结构的（位图），这样非常方便在运行时检索，更容易支持百万甚至上亿的成员定义标签，对于稀疏的标签也比较好的支持。

## 自定义属性介绍

只允许在标准成员上定义自定义属性（`intC0`、`intC1`、`stringC0`），而 [简单成员](#) 则不允许设置自定义属性。

## 自定义属性设置

```
/**
 * 非批量模式下设置自定义属性
 * @param type 创建成员（create）和修改成员（alter）均支持
 * @param dimension 维度名称
 * @param member 成员名称
 * @param intC0 第一个 int 类型的自定义属性
 * @param intC1 第二个 int 类型的自定义属性
 * @param stringC0 第一个 string 类型的自定义属性
 */
public static void singleSetCustomProperty(CommandTypes type, String dimension,
String member, int intC0, int intC1, String stringC0) {
    OlapConnection olapConn = OlapConnectionUtil.getConnection(cubeName);
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Member);
    commandInfo.setOwnerUniqueName(cubeName + "." + dimension);
    commandInfo.setAction(type);
    commandInfo.setName(member);
    commandInfo.setIntC0(intC0); // 设置自定义属性 intC0
    commandInfo.setIntC0(intC1); // 设置自定义属性 intC1
    commandInfo.setStringC0(stringC0); // 设置自定义属性 stringC0

    OlapCommand command = new OlapCommand(olapConn, commandInfo);
    command.executeNonQuery();
}
```

```

}

/**
 * 批量模式下设置自定义属性
 * @param type 创建成员（create）和修改成员（alter）均支持
 * @param dimension 维度名称
 * @param members 成员名称
 * @param intC0s 第一个 int 类型的自定义属性
 * @param intC1s 第二个 int 类型的自定义属性
 * @param stringC0s 第一个 string 类型的自定义属性
 */
public static void batchSetCustomProperty(CommandTypes type, String dimension,
String[] members, int[] intC0s, int[] intC1s, String[] stringC0s) {
    OlapConnection olapConn = OlapConnectionUtil.getConnection(cubeName);
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Member);
    commandInfo.setOwnerUniqueName(cubeName + "." + dimension);
    commandInfo.setAction(type);
    List<MemberMetadataItem> items = new ArrayList<>();
    for (int index = 0; index < members.length; index++) {
        MemberMetadataItem item = new MemberMetadataItem(members[index]);
        item.setIntC0(intC0s[index]); // 设置自定义属性 intC0
        item.setIntC1(intC1s[index]); // 设置自定义属性 intC1
        item.setStringC0(stringC0s[index]); // 设置自定义属性 stringC0
        items.add(item);
    }
    commandInfo.getItems().addAll(items);
    OlapCommand command = new OlapCommand(olapConn, commandInfo);
    command.executeNonQuery();
}

```

## 自定义属性查询

通过自定义方法 `getMemberCustomPropertyAndTag`，可基于维度以及成员清单，通过成员查找模式进行查询

成员查找模式：提供成员清单（使用 JSON 规范，例如：["member1","member2"]），获取所提供成员的自定义属性和打上的标签，如传入 [] 则获取所有成员的自定义属性和打上的标签。

管理

aggFunCube    追踪    管理    性能分析    慢语句查询    自定义方法    异步任务中心 正常

执行自定义方法

getMemberCustomPropertyAndTag

cubeName ?

aggFunCube

Dimension ?

orgs

Tags ?

Tags

Members ?

["orgs\_1"]

执行

执行并下载

Result

```
1 [{
2   "member": "orgs_1",
3   "tags": ["tag1"],
4   "customProperties": {
5     "string00": "0",
6     "int01": "1"
7   }
8 }]
```

```
/**
 * 使用成员查找模式进行查找
 * @param connection 数据库链接对象
 * @param dimensionName 维度名称
 * @param members 成员清单
 */
private static String getMemberCustomPropertyAndTag(OlapConnection connection,
String dimensionName, String members) {
    PropertyBag propertyBag = new PropertyBag();
    propertyBag.set("Dimension", dimensionName);
    propertyBag.set("Members", members);
    FunctionCommandInfo functionCommandInfo = new
FunctionCommandInfo("getMemberCustomPropertyAndTag", propertyBag);
    OlapCommand cmd = new OlapCommand(connection, functionCommandInfo);
    PropertyBag retPropertyBag = cmd.executeFunction();
    return retPropertyBag.get("result");
}

public static void main(String[] args) {
    OlapConnection olapConn = OlapConnectionUtil.getConnection(cubeName);
    String result = getMemberCustomPropertyAndTag(olapConn, "orgs", "
['orgs_1']");
    System.out.println(result);
}
```

输出结果：

```
[{"member": "orgs_1", "tags": ["tag1"], "customProperties": {"stringC0": "0", "intC1": "1"}}]
```

## 标签介绍

标准成员和 [简单成员](#) 都支持标签功能。

标签作为新设计的功能，虽然仍然是成员上的功能，发送的仍然是元数据命令，但**不再需要元数据写锁**。

## 标签的定义

标签的定义是在 [维度](#) 上的，其有如下规则：

1. 可以同时创建多个标签，多个标签使用，进行分割，如 "tag1, tag2, tag3"；
2. 一旦创建了标签，都会一直保留在数据库中；
3. 再次更新标签时，重复的tag一样不会重建，例如数据库中原有 tag1, tag2，现在更新标签 "tag1, tag3"，那么更新完成后数据库将会有如下标签：tag1, tag2, tag3；

```
/**
 * 标签的定义
 * @param type 创建维度（create）和修改维度（alter）均支持
 * @param dimension 维度名称
 * @param tags 标签清单
 */
public static void createTags(CommandTypes type, String dimension, String[]
tags) {
    OlapConnection olapConn = OlapConnectionUtil.getConnection(cubeName);
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Dimension);
    commandInfo.setOwnerUniqueName(cubeName);
    commandInfo.setAction(type);
    commandInfo.setName(dimension);
    String tagDefinitions = String.join(",", tags); // "tag1,tag2,tag3"
    commandInfo.setTagDefinitions(tagDefinitions); // 设置标签定义
    OlapCommand command = new OlapCommand(olapConn, commandInfo);
    command.executeNonQuery();
}
```

## 给成员打上标签

支持给一批成员打上标签，无论这些成员之前有没有打过该标签，命令执行完后都会打上该标签。

```
/**
 * 给成员打上标签
 * @param dimension 维度名称
 * @param members 成员清单
 * @param tag 标签名称
 */
public static void tagMembers(String dimension, String[] members, String tag) {
    OlapConnection olapConn = OlapConnectionUtil.getConnection(cubeName);
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Tag); // 元数据类型为标签
    commandInfo.setOwnerUniqueName(cubeName + "." + dimension);
    commandInfo.setAction(CommandTypes.create); // 打标签的类型为 create
}
```

```
// 设置标签清单，包括标签名称及需要打上该标签的成员，支持批量操作，一个命令可以包含多个标签清单
TagMetadataItem item = new TagMetadataItem(tag);
item.setMembers(Arrays.asList(members));
commandInfo.getItems().add(item);

OlapCommand command = new OlapCommand(olapConn, commandInfo);
command.executeNonQuery();
}
```

## 给成员删除标签

支持给一批成员去除标签，无论这些成员之前有没有打过该标签，命令执行完后都会去除该标签。

```
/**
 * 给成员去除标签
 * @param dimension 维度名称
 * @param members 成员清单
 * @param tag 标签名称
 */
public static void untagMembers(String dimension, String[] members, String tag)
{
    OlapConnection olapConn = OlapConnectionUtil.getConnection(cubeName);
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Tag); // 元数据类型为标签
    commandInfo.setOwnerUniqueName(cubeName + "." + dimension);
    commandInfo.setAction(CommandTypes.drop); // 打标签的类型为 drop
    // 设置标签清单，包括标签名称及需要去除该标签的成员，支持批量操作，一个命令可以包含多个标签清单
    TagMetadataItem item = new TagMetadataItem(tag);
    item.setMembers(Arrays.asList(members));
    commandInfo.getItems().add(item);

    OlapCommand command = new OlapCommand(olapConn, commandInfo);
    command.executeNonQuery();
}
```

## 标签查询

通过自定义方法 `getMemberCustomPropertyAndTag`，基于维度以及成员清单，通过成员查找模式进行查询(与自定义属性查询方式一致)

成员查找模式：提供成员清单（使用 JSON 规范，例如：["member1","member2"]），获取所提供成员的自定义属性和打上的标签，如传入 [] 则获取所有成员的自定义属性和打上的标签。

通过函数接口 `getMemberCustomPropertyAndTag`，基于维度以及标签清单，通过标签查找模式进行查询

标签查找模式：提供标签清单（使用 JSON 规范，例如：["tag1","tag2"]），获取打上这些标签的成员清单，如传入 [] 则获取所有标签的成员清单。

管理

aggFunCube 追踪 管理 性能分析 慢语句查询 自定义方法 异步任务中心 正常

执行自定义方法

getMemberCustomPropertyAndTag x

cubeName ?

aggFunCube

Dimension ?

orgs

Tags ?

[tag1]

Members ?

Members

执行 执行并下载

Result

1 [[  
2     "tag": "tag1",  
3     "members": ["orgs\_1", "orgs\_2"]  
4 ]]

```
/**
 * 使用标签查找模式进行查找
 * @param connection 数据库链接对象
 * @param dimensionName 维度名称
 * @param tags 标签清单
 */
private static String getMemberCustomPropertyAndTag(OlapConnection connection,
String dimensionName, String tags) {
    PropertyBag propertyBag = new PropertyBag();
    propertyBag.set("Dimension", dimensionName);
    propertyBag.set("Tags", tags);
    FunctionCommandInfo functionCommandInfo = new
FunctionCommandInfo("getMemberCustomPropertyAndTag", propertyBag);
    OlapCommand cmd = new OlapCommand(connection, functionCommandInfo);
    PropertyBag retPropertyBag = cmd.executeFunction();
    return retPropertyBag.get("result");
}
public static void main(String[] args) {
    OlapConnection olapConn = OlapConnectionUtil.getConnection(cubeName);
    String result = getMemberCustomPropertyAndTag(olapConn, "orgs", "[tag1]");
    System.out.println(result);
}
```

输出结果：

```
[{"tag": "tag1", "members": ["orgs_1", "orgs_2"]}]
```

上一篇：[简单成员](#)

下一篇：[成员函数表达式及成员标签表达式](#)

