

[返回首页](#)

成员元数据操作说明

本文将给出成员操作示例，在其他章节介绍某些特性时也有了一些成员操作的介绍，这里给出全面的成员操作，着重介绍各种操作所支持的批量操作。

成员支持的操作类型

多维数据库成员操作支持以下Action：create，alter，drop。

多维数据库成员操作Action：create

create 操作用于为指定维度新增成员，包括新增单个成员和批量新增成员两种模式。

1、新增单个成员模式：

新增单个成员时允许同时指定成员存储类型、成员屏蔽规则。

```
private void createSingleMember() {
    String cubeName = "myFirstCube";
    String dimensionName = "accounts";
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Member);
    commandInfo.setAction(CommandTypes.create);
    commandInfo.setName("account10");
    commandInfo.setOwnerUniqueName(cubeName + "." + dimensionName);

    // 指定成员的存储类型，如未指定，默认为 MemberStorageTypes.Stored
    commandInfo.setStorageType(MemberStorageTypes.DynamicCalc);

    // 指定成员的屏蔽规则，如未指定，默认为 all
    commandInfo.setMemberAggShieldRule("timePoint");

    OlapConnection olapConnection = createConnection();
    try{
        OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);
        cmd.executeNonQuery();
    } finally {
        olapConnection.Close();
    }
}
```

2、批量新增成员模式：

批量新增成员时允许同时指定成员存储类型、成员屏蔽规则、成员表达式。

```
private void createBatchMembers() {
    String cubeName = "myFirstCube";
    String dimensionName = "accounts";
```

```

MetadataCommandInfo commandInfo = new MetadataCommandInfo();
commandInfo.setMetadataType(MetadataTypes.Member);
commandInfo.setAction(CommandTypes.create);
commandInfo.setOwnerUniqueName(cubeName + "." + dimensionName);

List<MetadataItem> memberItems = commandInfo.getItems();

// 创建成员 account10
MemberMetadataItem account10 = new MemberMetadataItem();
account10.setName("account10");
account10.setStorageType(MemberStorageTypes.DynamicCalc);
account10.setMemberAggShieldRule("timePoint");
// 批量新增成员支持同时指定成员表达式
List<AggFactorMetadataItem> account10FactorItems = account10.getFactors();
// AggFactorMetadataItem 有两个构造：
// (String id, String name, AggOperators aggOperator)
// (String name, AggOperators aggOperator)
// 因子 id 是可选，如未指定因子 id，则需确保该因子 name 在当前表达式所有未指定 id 的因子中是唯一的，
// 如指定因子 id，允许存在多个 name 相同且指定 id 的因子，但需确保该因子的 id 在当前表达式所有指定 id 的因子中是唯一的。
account10FactorItems.add(new AggFactorMetadataItem("1", "account1", AggOperators.PLUS));
account10FactorItems.add(new AggFactorMetadataItem("account2", AggOperators.PLUS));

// 创建成员 account11
MemberMetadataItem account11 = new MemberMetadataItem();
account11.setName("account11");
account11.setStorageType(MemberStorageTypes.DynamicCalc);
account11.setMemberAggShieldRule("timePoint");
List<AggFactorMetadataItem> account11FactorItems = account11.getFactors();
// 因子允许是即将创建的成员，比如当前例子的 account10
account11FactorItems.add(new AggFactorMetadataItem("1", "account10", AggOperators.PLUS));

memberItems.add(account10);
memberItems.add(account11);

OlapConnection olapConnection = createConnection();
try{
    OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);
    cmd.executeNonQuery();
} finally {
    olapConnection.Close();
}
}

```

多维数据库成员操作Action：alter

alter 操作用于为修改指定成员属性，包括修改单个成员和批量修改成员两种模式。

批量修改成员需使用服务端5.3.0或以上版本，客户端2.1.3或以上版本。

1、修改单个成员模式：

修改单个成员允许修改指定成员存储类型、成员屏蔽规则、成员名称。

```

private void alterSingleMember() {
    String cubeName = "myFirstCube";
    String dimensionName = "accounts";
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Member);
    commandInfo.setAction(CommandTypes.alter);
    commandInfo.setName("account10");
    commandInfo.setOwnerUniqueName(cubeName + "." + dimensionName);

    // 修改成员的存储类型
    commandInfo.setStorageType(MemberStorageTypes.Stored);

    // 修改成员的屏蔽规则
    commandInfo.setMemberAggShieldRule("all");

    // 修改成员名称（该场景只在修改单个成员模式下支持）
    commandInfo.setNewName("account10_new");

    OlapConnection olapConnection = createConnection();
    try{
        OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);
        cmd.executeNonQuery();
    } finally {
        olapConnection.Close();
    }
}

```

2、批量修改成员模式：

批量修改成员允许修改指定成员存储类型、成员屏蔽规则，或者修复成员表达式。

特别注意该模式下对于成员表达式是修复动作，即会覆盖原有成员表达式。

```

private void alterBatchMembers() {
    String cubeName = "myFirstCube";
    String dimensionName = "accounts";
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Member);
    commandInfo.setAction(CommandTypes.alter);
    commandInfo.setOwnerUniqueName(cubeName + "." + dimensionName);

    List<MetadataItem> memberItems = commandInfo.getItems();

    // 修改成员 account10
    MemberMetadataItem account10 = new MemberMetadataItem();
    account10.setName("account10");
    // 修改成员的存储类型
    account10.setStorageType(MemberStorageTypes.DynamicCalc);
    // 修改成员的屏蔽规则
    account10.setMemberAggShieldRule("none");
    // 批量修改成员对于成员表达式是进行修复操作!!!
    // 即通过调用 repairFactors(List newFactors) 方法，将 newFactors 覆盖原有成员表达式
    List<AggFactorMetadataItem> newAccount10FactorItems = new ArrayList<>();
}

```

```

newAccount10FactorItems.add(new AggFactorMetadataItem("1", "account11",
AggOperators.SUBTRACT));
account10.repairFactors(newAccount10FactorItems);

// 创建成员 account11
MemberMetadataItem account11 = new MemberMetadataItem();
account11.setName("account11");
// 修改成员的存储类型
account11.setStorageType(MemberStorageTypes.Stored);
// 修改成员的屏蔽规则
account11.setMemberAggShieldRule("none");
// 批量修改成员对于成员表达式是进行修复操作!!!
// 即通过调用 repairFactors(List newFactors) 方法, 将 newFactors 覆盖原有成员表达式
List<AggFactorMetadataItem> newAccount11FactorItems = new ArrayList<>();
newAccount11FactorItems.add(new AggFactorMetadataItem("1", "account11",
AggOperators.SUBTRACT));
newAccount11FactorItems.add(new AggFactorMetadataItem("account2", AggOperators.SUBTRACT));
account11.repairFactors(newAccount11FactorItems);

memberItems.add(account10);
memberItems.add(account11);

OlapConnection olapConnection = createConnection();
try{
    OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);
    cmd.executeNonQuery();
} finally {
    olapConnection.Close();
}
}

```

多维数据库成员操作Action : drop

drop 操作用于删除指定成员，包括删除单个成员和批量删除成员两种模式。

可设置命令级别的参数：AllowDropMemberWhenRowReferenced，类型：boolean，以显示表明如果即将删除的成员在数据库中被数据引用，是否允许强制删除；默认为不允许（false）

1、删除单个成员模式：

```

private void dropSingleMember() {
    String cubeName = "myFirstCube";
    String dimensionName = "accounts";
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Member);
    commandInfo.setAction(CommandTypes.drop);
    // 删除成员 account10
    commandInfo.setName("account10");
    commandInfo.setOwnerUniqueName(cubeName + "." + dimensionName);

    // 如果即将删除的成员在数据库中被数据引用，是否允许强制删除，默认为不允许（false）
    commandInfo.setAllowDropMemberWhenRowReferenced(true);
}

```

```

OlapConnection olapConnection = createConnection();
try{
    OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);
    cmd.executeNonQuery();
} finally {
    olapConnection.Close();
}
}

```

2、批量删除成员模式：

```

private void dropBatchMember() {
    String cubeName = "myFirstCube";
    String dimensionName = "accounts";
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Member);
    commandInfo.setAction(CommandTypes.drop);
    commandInfo.setOwnerUniqueName(cubeName + "." + dimensionName);

    // 如果即将删除的成员在数据库中被数据引用，是否允许强制删除，默认为不允许 ( false)
    commandInfo.setAllowDropMemberWhenRowReferenced(true);

    List<MetadataItem> memberItems = commandInfo.getItems();
    // 删除成员 account10 和 account11
    memberItems.add(new MemberMetadataItem("account10"));
    memberItems.add(new MemberMetadataItem("account11"));

    OlapConnection olapConnection = createConnection();
    try{
        OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);
        cmd.executeNonQuery();
    } finally {
        olapConnection.Close();
    }
}

```

3、（ add on 2021-05-07 ）删除成员时新命级别参数allowDropMemberWithNullData，表示是否允许删除被度量值均为 null 的数据引用的成员，默认为false，**该功能在服务端版本6.3.0及以上，客户端版本2.3.2及以上支持**。删除单个成员和批量删除成员的指令中均支持该参数。1) 当成员有数据引用，如果删除该成员的指令将allowDropMemberWithNullData设置为false则抛异常；2) 当成员有数据引用且其数据的度量值均为null，如果删除该成员的指令将allowDropMemberWithNullData设置为true则可将成员删除；3) 当成员有数据引用且其数据的度量值不全为null，如果删除该成员的指令将allowDropMemberWithNullData设置为true则抛异常，并显示前十笔度量值为非null的维度组合；

```

private void dropSingleMember() {
    String cubeName = "myFirstCube";
    String dimensionName = "accounts";
    MetadataCommandInfo commandInfo = new MetadataCommandInfo();
    commandInfo.setMetadataType(MetadataTypes.Member);
    commandInfo.setAction(CommandTypes.drop);
    // 删除成员 account10

```

```
commandInfo.setName("account10");
commandInfo.setOwnerUniqueName(cubeName + "." + dimensionName);

// 是否允许删除被度量值均为 null 的数据引用的成员
commandInfo.setAllowDropMemberWithNullData(true);
OlapConnection olapConnection = createConnection();
try{
    OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);
    cmd.executeNonQuery();
} finally {
    olapConnection.Close();
}
}
```

上一篇：[获取Cube元数据](#)

下一篇：[计算引擎](#)