数据分割屏蔽规则文档

1. 修订记录

|  |  |  |
| --- | --- | --- |
| **日期** | **修订** | **描述** |
| 2020-12-09 | 许方友 | 创建 |
|  |  |  |

1. 数据分割屏蔽规则说明

数据分割屏蔽规则支持的版本：

服务端：5.4及以上

客户端：2.1.4及以上

数据分割屏蔽规则是基于现有动态计算功能，仅用于合并报表应用特殊的场景，即某一年的数据分割成不启用动态计算功能的历史数据和启用动态计算功能的新数据。

数据分割屏蔽规则设置在具体的年【Year】成员上，例如需要指定（FY2020，M\_M10）前的数据不启用动态计算功能，需要在FY2020设置屏蔽规则“past10Months”，而FY2020前的年份如FY2019，FY2018等需要设置other类型屏蔽规则，屏蔽除了年以外的维度，年维度不屏蔽是考虑到各个年份可能会聚合到AllYear成员上。

数据分割屏蔽规则需要指定不启用动态计算功能的所有**“完整”**的**存储型**期间【Period】成员，这里**“完整”**的含义是这个期间成员从逻辑上完整包含了不启用动态计算的数据。例如需要指定前10个月的数据不启用动态计算功能，期间的结构如下：



1. **假设在多维数据库中该维度所有成员均设置为存储型：**

新增数据分割屏蔽规则“past10Months”，需要指定的完整期间成员如下：M\_M01、M\_M02、M\_M03、M\_M04、M\_M05、M\_M06、M\_M07、M\_M08、M\_M09、M\_M10、M\_Q1、M\_Q2、M\_Q3、M\_HF1，而M\_Q4、M\_HF2、M\_YearTotal不是完整期间成员是因为其逻辑上包含需要启用动态计算的数据。

后续如果执行聚合指令（M\_Q4 = M\_M10 + M\_M11 + M\_M12），会将M\_M10存在数据库的数据与M\_M11、M\_M12的动态计算结果进行聚合。

1. **假设在多维数据库中全年、半年、季度均设置为动态计算型，月份设置为存储型：**

可以新增数据分割屏蔽规则“past10Months”，需要指定的完整期间成员如下：M\_M01、M\_M02、M\_M03、M\_M04、M\_M05、M\_M06、M\_M07、M\_M08、M\_M09、M\_M10，而M\_Q1、M\_HF1、M\_YearTotal等属于动态计算型成员而无需设置。

后续查询M\_Q4的数据，将进行动态计算，也是将M\_M10存在数据库的数据与M\_M11、M\_M12的动态计算结果进行聚合。

1. 数据分割屏蔽规则操作Action：create

create 操作用于新增数据分割屏蔽规则。

1. 新增数据分割屏蔽规则：

private void createDataPartitionRule() {  
 String cubeName = "myFirstCube";  
 MetadataCommandInfo commandInfo = new MetadataCommandInfo();  
 commandInfo.setMetadataType(MetadataTypes.*AggShieldRule*);  
 commandInfo.setAction(CommandTypes.*create*);  
 commandInfo.setOwnerUniqueName(cubeName);  
  
 String ruleName = "past10Months";  
 AggShieldRuleDataPartitionMetadataItem item = new AggShieldRuleDataPartitionMetadataItem(ruleName,  
 Arrays.*asList*("M\_M01", "M\_M02", "M\_M03", "M\_M04", "M\_M05", "M\_M06",  
 "M\_M07", "M\_M08", "M\_M09", "M\_M10", "M\_Q1", "M\_Q2", "M\_Q3", "HF1"));  
  
 commandInfo.getItems().add(item);  
  
 OlapConnection olapConnection = createConnection();  
 try{  
 OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);  
 cmd.executeNonQuery();  
 } finally {  
 olapConnection.Close();  
 }  
}

1. 数据分割屏蔽规则操作Action：repair

repair 操作用于为修复指定数据分割屏蔽规则，这里的修复是对原有屏蔽规则进行覆盖

1. 修复数据分割屏蔽规则：

private void repairDataPartitionRule() {  
 String cubeName = "myFirstCube";  
 MetadataCommandInfo commandInfo = new MetadataCommandInfo();  
 commandInfo.setMetadataType(MetadataTypes.*AggShieldRule*);  
 commandInfo.setAction(CommandTypes.*repair*);  
 commandInfo.setOwnerUniqueName(cubeName);  
  
 // 对已存在的数据分割屏蔽规则 “past10Months” 进行修复  
 String ruleName = "past10Months";  
 AggShieldRuleDataPartitionMetadataItem item = new AggShieldRuleDataPartitionMetadataItem(ruleName,  
 Arrays.*asList*("M\_M01", "M\_M02", "M\_M03", "M\_M04", "M\_M05", "M\_M06",  
 "M\_M07", "M\_M08", "M\_M09", "M\_M10"));  
  
 commandInfo.getItems().add(item);  
  
 OlapConnection olapConnection = createConnection();  
 try{  
 OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);  
 cmd.executeNonQuery();  
 } finally {  
 olapConnection.Close();  
 }  
}

1. 数据分割屏蔽规则操作Action：drop

drop 操作用于为删除指定数据分割屏蔽规则。

1. 删除数据分割屏蔽规则：

private void dropDataPartitionRule() {  
 String cubeName = "myFirstCube";  
 MetadataCommandInfo commandInfo = new MetadataCommandInfo();  
 commandInfo.setMetadataType(MetadataTypes.*AggShieldRule*);  
 commandInfo.setAction(CommandTypes.*drop*);  
 commandInfo.setOwnerUniqueName(cubeName);  
  
 // 删除指定的屏蔽规则，只需指定名称和类型即可。  
 String ruleName = "past10Months";  
 AggShieldRuleDataPartitionMetadataItem item = new AggShieldRuleDataPartitionMetadataItem(ruleName, Collections.*emptyList*());  
  
 commandInfo.getItems().add(item);  
  
 OlapConnection olapConnection = createConnection();  
 try{  
 OlapCommand cmd = new OlapCommand(olapConnection, commandInfo);  
 cmd.executeNonQuery();  
 } finally {  
 olapConnection.Close();  
 }  
}