一、接口说明

1、装载:

function:BlockData load(datasourceInfo dsInfo, int cityID, List<String>columns) throws Exception

parameters:

dsInfo:数据源信息

cityID:城市信息(如果为0，则取所有城市)

columns:提取列名链表

return:

返回装载的数据块

Description:

1. 装载按指定的列和城市id来装载数据来构建数据行(GroupArrayingData)；
2. 所有行装载并存放到BlockData中，同时根据指定的行信息需要构建新的“字段描述信息”(GroupingArrayDataDescriptor)
3. 装载出错以异常形式抛出

2、存储

function: void save(datasourceInfo dsInfo, BlockData blockData) throws Exception

parameters:

dsInfo:数据源信息

blockData:数据块

return:

无

Description:

1、保存前，所有的GroupArrayingData数据需要按city id排序；

2、保存内容有：列的数量(全部列)、列的名称、列的类型、城市和列对应的数据偏移位置二维交叉表、数据内容；

3、装载出错以异常形式抛出

二、数据文件命名规则:

1、文件存储位置及格式说明：

./[neType]/[neLevel]/[gran]/[yyyymmddhh24].bin

neType:网络类型CDMA/WCDMA/GSM

datasourceInfo.getNetType()

nelevel:网元级别BSC/BTS/CELL

datasourceInfo.getNeLevel()

gran:时间粒度HOUR/WEEK/MONTH

datasourceInfo.getGranularity()

yyyymmddhh24:数据时间

datasourceInfo.getDataTime()

2、城市ID:

1、保存的时候，需要能从GroupArrayingData中读取到行的city id;

2、载入时，不再从datasourceInfo.getCityId()读取，根据load传入的参数决定；

三、存储结构

|  |  |  |
| --- | --- | --- |
| **存储块** | **描述** | **备注** |
| 文件头基本信息 | {记录数:[4]} {BlockData的附加信息:[random]} | BlockData的附加信息有:indexKey、exportKey |
| 列的描述 | {列的数量:[2]} {  {列名:[random]} {类型:[1]}  {列名:[random]} {类型:[1]}  ... } |  |
| 每个城市的每列存储偏移位置(预留最后写入) 预留长度=2+城市数量\*(2+(列的数量\*16)) | {城市数量:[2]} {  {城市ID:[2]} {当前城市的记录数:[4]}  {  {第0列偏移位置:[8]} {数据块尺寸:[4]}  {第1列偏移位置:[8]} {数据块尺寸:[4]}  {第2列偏移位置:[8]} {数据块尺寸:[4]}  ……  } } |  |
| 数据块 | //xx城市xx列 {  {从0行至n行的数据内容...} }  //xx城市xx列 {  {数据内容...} } …… | 方案一：  每个String的存储格式:{长度:[4]} + {byte[]}  如果String == null, 长度则存入Integer.MAX\_VALUE |