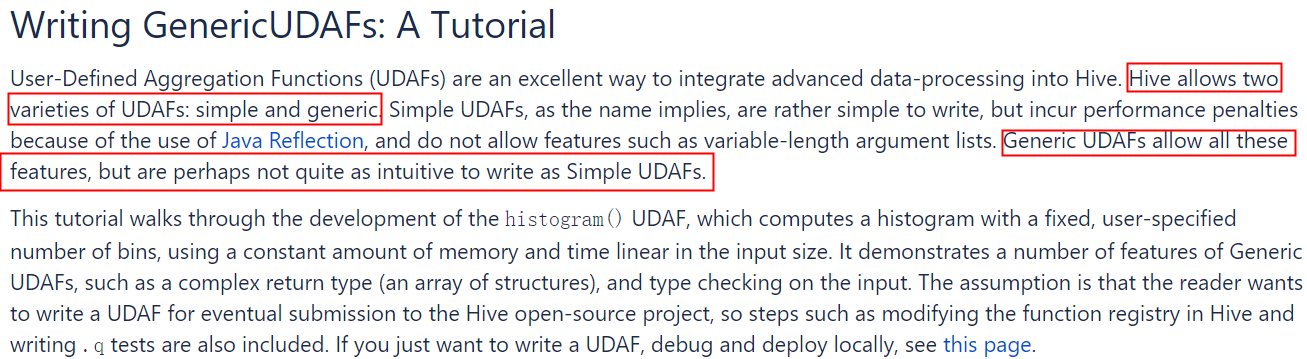
##### UDAF(Hive)问题总结

###### 简要说明

UDAF是Hive中用户自定义的聚集函数，UDAF实现有简单（UDAF）与通用（AbstractGenericUDAFResolver）两种方式，简单UDAF因为使用Java反射导致性能损失，而且有些特性不能使用，**已被弃用。**

**提示：**初期开发，采用简单UDAF语法实现逻辑功能，出现不能成功运行、批量补数据出现错误等问题，导致无法准确、有效定位具体问题，期间自定义函数经历至少3个版本。后续改写为通用接口（AbstractGenericUDAFResolver），约束各阶段输入、输出数据格式，问题得以解决。

**说明**：如果知晓如上问题点，那么实现UDAF通用接口自定义函数不难实现。但如果未发现上述问题点，一直围绕简单UDAF接口中的数据结构等问题进行调整，将很难彻底解决异常问题。

**Wiki:**

###### 开发步骤

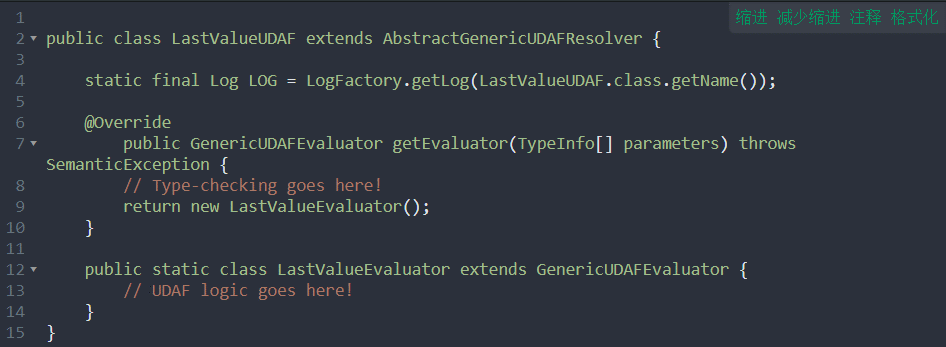
开发通用UDAF包含以下2个步骤：

1. 编写resolver类
2. 编写evaluator类

resolver负责类型检查，操作符重载。evaluator真正实现UDAF的逻辑。通常顶层UDAF类继承org.apache.hadoop.hive.ql.udf.GenericUDAFResolver2，里面编写嵌套类evaluator 实现UDAF的逻辑。

但强烈建议继承AbstractGenericUDAFResolver，隔离将来hive接口的变化。GenericUDAFResolver和GenericUDAFResolver2接口的区别是，后面的允许evaluator实现可以访问更多的信息，例如DISTINCT限定符，通配符FUNCTION(\*)。

###### UDAF代码框架示例



###### GenericUDAFEvaluator

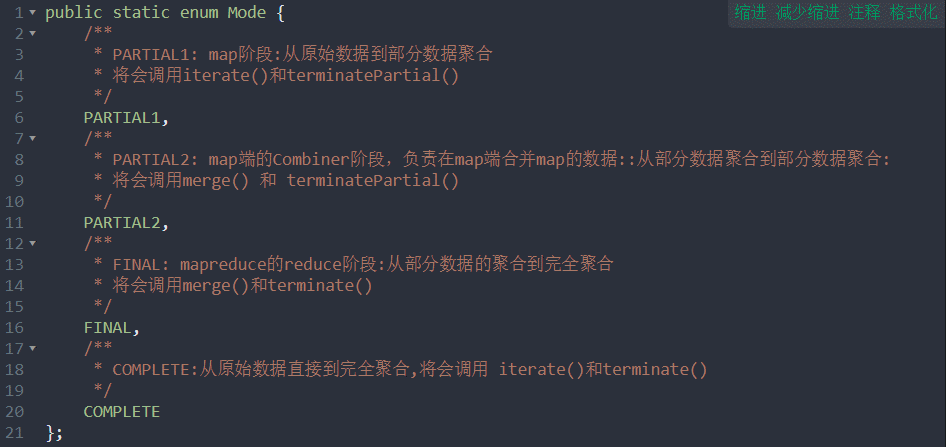
UDAF逻辑处理主要发生在Evaluator中，需要实现该抽象类的几个方法。

1. ObjectInspector接口

主要用于解耦数据使用与数据格式，使得数据流在输入输出端切换不同的输入输出格式，不同的Operator上使用不同的格式。

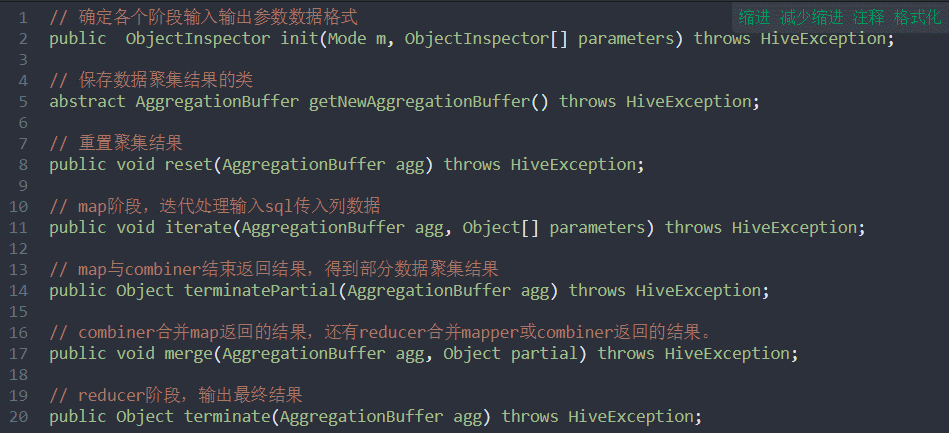
1. GenericUDAFEvaluator内部类Model

代表UDAF在mapreduce的各个阶段。

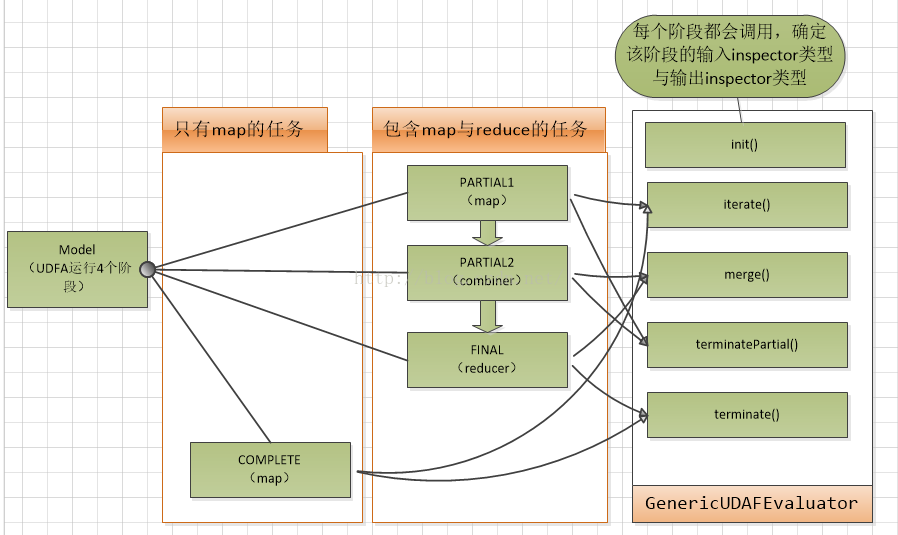


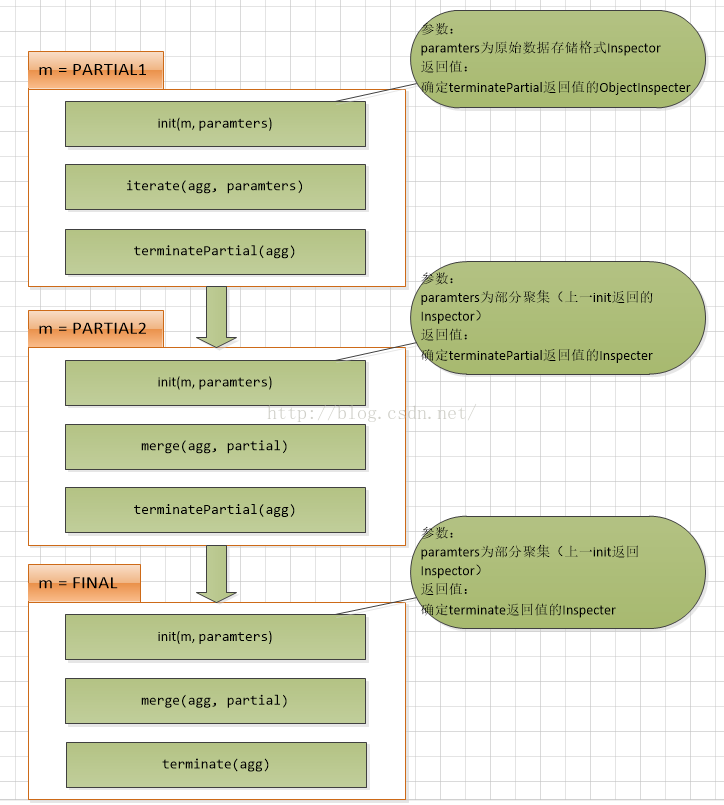
一般情况下，完整UDAF逻辑是一个mapreduce过程，如果有mapper和reducer，就会经历PARTIAL1(mapper)，FINAL(reducer)；如果还有combiner，那就会经历PARTIAL1(mapper)，PARTIAL2(combiner)，FINAL(reducer)。

1. GenericUDAFEvaluator方法



1. Model与Evaluator关系





1. UDAF示例代码

|  |
| --- |
| @Description(name = "letters", value = "\_FUNC\_(expr) - 返回该列中所有字符串的字符总数")  public class TotalNumOfLettersGenericUDAF extends AbstractGenericUDAFResolver {      @Override      public GenericUDAFEvaluator getEvaluator(TypeInfo[] parameters)              throws SemanticException {          if (parameters.length != 1) {              throw new UDFArgumentTypeException(parameters.length - 1,                      "Exactly one argument is expected.");          }          ObjectInspector oi = TypeInfoUtils.getStandardJavaObjectInspectorFromTypeInfo(parameters[0]);          if (oi.getCategory() != ObjectInspector.Category.PRIMITIVE){              throw new UDFArgumentTypeException(0,                              "Argument must be PRIMITIVE, but "                              + oi.getCategory().name()                              + " was passed.");          }          PrimitiveObjectInspector inputOI = (PrimitiveObjectInspector) oi;          if (inputOI.getPrimitiveCategory() != PrimitiveObjectInspector.PrimitiveCategory.STRING){              throw new UDFArgumentTypeException(0,                              "Argument must be String, but "                              + inputOI.getPrimitiveCategory().name()                              + " was passed.");          }          return new TotalNumOfLettersEvaluator();      }      public static class TotalNumOfLettersEvaluator extends GenericUDAFEvaluator {          PrimitiveObjectInspector inputOI;          ObjectInspector outputOI;          PrimitiveObjectInspector integerOI;          int total = 0;          @Override          public ObjectInspector init(Mode m, ObjectInspector[] parameters)                  throws HiveException {              assert (parameters.length == 1);              super.init(m, parameters);               //map阶段读取sql列，输入为String基础数据格式              if (m == Mode.PARTIAL1 || m == Mode.COMPLETE) {                  inputOI = (PrimitiveObjectInspector) parameters[0];              } else {              //其余阶段，输入为Integer基础数据格式                  integerOI = (PrimitiveObjectInspector) parameters[0];              }               // 指定各个阶段输出数据格式都为Integer类型              outputOI = ObjectInspectorFactory.getReflectionObjectInspector(Integer.class,                      ObjectInspectorOptions.JAVA);              return outputOI;          }          /\*\*           \* 存储当前字符总数的类           \*/          static class LetterSumAgg implements AggregationBuffer {              int sum = 0;              void add(int num){                  sum += num;              }          }          @Override          public AggregationBuffer getNewAggregationBuffer() throws HiveException {              LetterSumAgg result = new LetterSumAgg();              return result;          }          @Override          public void reset(AggregationBuffer agg) throws HiveException {              LetterSumAgg myagg = new LetterSumAgg();          }          private boolean warned = false;          @Override          public void iterate(AggregationBuffer agg, Object[] parameters)                  throws HiveException {              assert (parameters.length == 1);              if (parameters[0] != null) {                  LetterSumAgg myagg = (LetterSumAgg) agg;                  Object p1 = ((PrimitiveObjectInspector) inputOI).getPrimitiveJavaObject(parameters[0]);                  myagg.add(String.valueOf(p1).length());              }          }          @Override          public Object terminatePartial(AggregationBuffer agg) throws HiveException {              LetterSumAgg myagg = (LetterSumAgg) agg;              total += myagg.sum;              return total;          }          @Override          public void merge(AggregationBuffer agg, Object partial)                  throws HiveException {              if (partial != null) {                  LetterSumAgg myagg1 = (LetterSumAgg) agg;                  Integer partialSum = (Integer) integerOI.getPrimitiveJavaObject(partial);                  LetterSumAgg myagg2 = new LetterSumAgg();                  myagg2.add(partialSum);                  myagg1.add(myagg2.sum);              }          }          @Override          public Object terminate(AggregationBuffer agg) throws HiveException {              LetterSumAgg myagg = (LetterSumAgg) agg;              total = myagg.sum;              return myagg.sum;          }      }  } |

###### 经典示例推荐

1）文章

1. [Hadoop Hive UDF Tutorial - Extending Hive with Custom Functions](https://blog.matthewrathbone.com/2013/08/10/guide-to-writing-hive-udfs.html#the-complex-api)
2. [ObjectInspector](https://blog.csdn.net/czw698/article/details/38407817)

1）项目

1. [hive-extension-examples](1.%09https:/github.com/rathboma/hive-extension-examples)

* Siry@20190121