Flink Table API&SQL编程指南之时间属性(3)

Flink总共有三种时间语义：*Processing time*(处理时间)、*Event time*(事件时间)以及*Ingestion time*(摄入时间)。关于这些时间语义的具体解释，可以参考另一篇文章Flink的时间与watermarks详解。本文主要讲解Flink Table API & SQL中基于时间的算子如何定义时间语义。通过本文你可以了解到：

- 时间属性的简介
- 处理时间
- 事件时间

时间属性简介

Flink TableAPI&SQL中的基于时间的操作(如window)，需要指定时间语义，表可以根据指定的时间戳提供一个逻辑时间属性。

时间属性是表schama的一部分，当使用DDL创建表时、DataStream转为表时或者使用TableSource时，会定义时间属性。一旦时间属性被定义完成，该时间属性可以看做是一个字段的引用，从而在基于时间的操作中使用该字段。

时间属性像一个时间戳，可以被访问并参与计算，如果一个时间属性参与计算，那么该时间属性会被雾化成一个常规的时间戳，常规的时间戳不能与Flink的时间与水位线兼容，不能被基于时间的操作所使用。

Flink TableAPI & SQL所需要的时间属性可以通过Datastream程序中指定，如下：

```
final StreamExecutionEnvironment env = StreamExecutionEnvironment.getExec
utionEnvironment();

env.setStreamTimeCharacteristic(TimeCharacteristic.ProcessingTime); // 默
认

// 可以选择:
// env.setStreamTimeCharacteristic(TimeCharacteristic.IngestionTime);
// env.setStreamTimeCharacteristic(TimeCharacteristic.EventTime);
```

处理时间

基于本地的机器时间，是一种最简单的时间语义，但是不能保证结果一致性，使用该时间语义不需要提取时间戳和生成水位线。总共有三种方式定义处理时间属性，具体如下

DDL语句创建表时定义处理时间

处理时间的属性可以在DDL语句中被定义为一个计算列，需要使用PROCTIME()函数，如下所示：

```
CREATE TABLE user_actions (
  user_name STRING,
  data STRING,
  user_action_time AS PROCTIME() -- 声明一个额外字段，作为处理时间属性
) WITH (
  ...
);

SELECT TUMBLE_START(user_action_time, INTERVAL '10' MINUTE), COUNT(DISTI
NCT user_name)
FROM user_actions
GROUP BY TUMBLE(user_action_time, INTERVAL '10' MINUTE); -- 10分钟的滚动窗
口
```

DataStream转为Table的过程中定义处理时间

在将DataStream转为表时，在schema定义中可以通过.proctime属性指定时间属性，并将其放在其他schema字段的最后面，具体如下：

```
DataStream<Tuple2<String, String>> stream = ...;
// 声明一个额外逻辑字段作为处理时间属性
Table table = tEnv.fromDataStream(stream, "user_name, data, user_action_
time.proctime");

WindowedTable windowedTable = table.window(Tumble.over("10.minutes").on(
"user_action_time").as("userActionWindow"));
```

使用TableSource

自定义TableSource并实现`DefinedProctimeAttribute`接口，如下：

```
// 定义个带有处理时间属性的table source
public class UserActionSource implements StreamTableSource<Row>, DefinedP
roctimeAttribute {

        @Override
        public TypeInformation<Row> getReturnType() {
```

```
                String[] names = new String[] {"user_name" , "data"};
                TypeInformation[] types = new TypeInformation[] {Types.S
TRING(), Types.STRING()};
                return Types.ROW(names, types);
        }

        @Override
        public DataStream<Row> getDataStream(StreamExecutionEnvironment
execEnv) {
                // 创建stream
                DataStream<Row> stream = ...;
                return stream;
        }

        @Override
        public String getProctimeAttribute() {
        // 该字段会追加到schema中，作为第三个字段
                return "user_action_time";
        }
}

// 注册table source
tEnv.registerTableSource("user_actions", new UserActionSource());

WindowedTable windowedTable = tEnv
        .from("user_actions")
        .window(Tumble.over("10.minutes").on("user_action_time").as("use
rActionWindow"));
```

事件时间

基于记录的具体时间戳，即便是存在乱序或者迟到数据也会保证结果的一致性。总共有三种方式定义处理时间属性，具体如下

DDL语句创建表时定事件时间

事件时间属性可以通过 WATERMARK语句进行定义，如下：

```
CREATE TABLE user_actions (
  user_name STRING,
  data STRING,
  user_action_time TIMESTAMP(3),
  -- 声明user_action_time作为事件时间属性，并允许5S的延迟
  WATERMARK FOR user_action_time AS user_action_time - INTERVAL '5' SECO
ND
) WITH (
  ...
```

```
);

SELECT TUMBLE_START(user_action_time, INTERVAL '10' MINUTE), COUNT(DISTI
NCT user_name)
FROM user_actions
GROUP BY TUMBLE(user_action_time, INTERVAL '10' MINUTE);
```

DataStream转为Table的过程中定义事件时间

当定义Schema时通过.rowtime属性指定事件时间属性，必须在DataStream中指定时间戳与水位线。例如在数据集中，事件时间属性为event_time，此时Table中的事件时间字段中可以通过’event_time. rowtime‘来指定。

目前Flink支持两种方式定义EventTime字段，如下：

```
// 方式1:
// 提取timestamp并分配watermarks
DataStream<Tuple2<String, String>> stream = inputStream.assignTimestamps
AndWatermarks(...);

// 声明一个额外逻辑字段作为事件时间属性
// 在table schema的末尾使用user_action_time.rowtime定义事件时间属性
// 系统会在TableEnvironment中获取事件时间属性
Table table = tEnv.fromDataStream(stream, "user_name, data, user_action_
time.rowtime");

// 方式2:

// 从第一个字段提取timestamp并分配watermarks
DataStream<Tuple3<Long, String, String>> stream = inputStream.assignTime
stampsAndWatermarks(...);

// 第一个字段已经用来提取时间戳，可以直接使用对应的字段作为事件时间属性
Table table = tEnv.fromDataStream(stream, "user_action_time.rowtime, use
r_name, data");

// 使用:

WindowedTable windowedTable = table.window(Tumble.over("10.minutes").on(
"user_action_time").as("userActionWindow"));
```

使用TableSource

另外也可以在创建TableSource的时候，实现DefinedRowtimeAttributes接口来定义EventTime字段，在接口中需要实现getRowtimeAttributeDescriptors方法，创建基于EventTime的时间属性信息。

```java
// 定义带有rowtime属性的table source
public class UserActionSource implements StreamTableSource<Row>, DefinedRowtimeAttributes {

        @Override
        public TypeInformation<Row> getReturnType() {
                String[] names = new String[] {"user_name", "data", "user_action_time"};
                TypeInformation[] types =
                    new TypeInformation[] {Types.STRING(), Types.STRING(), Types.LONG()};
                return Types.ROW(names, types);
        }

        @Override
        public DataStream<Row> getDataStream(StreamExecutionEnvironment execEnv) {

        // 创建流，基于user_action_time属性分配水位线
                DataStream<Row> stream = inputStream.assignTimestampsAndWatermarks(...);
                return stream;
        }

        @Override
        public List<RowtimeAttributeDescriptor> getRowtimeAttributeDescriptors() {
        // 标记user_action_time字段作为事件时间属性
        // 创建user_action_time描述符，用来标识时间属性字段
                RowtimeAttributeDescriptor rowtimeAttrDescr = new RowtimeAttributeDescriptor(
                        "user_action_time",
                        new ExistingField("user_action_time"),
                        new AscendingTimestamps());
                List<RowtimeAttributeDescriptor> listRowtimeAttrDescr = Collections.singletonList(rowtimeAttrDescr);
                return listRowtimeAttrDescr;
        }
}

// register表
tEnv.registerTableSource("user_actions", new UserActionSource());

WindowedTable windowedTable = tEnv
        .from("user_actions")
        .window(Tumble.over("10.minutes").on("user_action_time").as("userActionWindow"));
```

小结

本文主要介绍了如何在Flink Table API和SQL中使用时间语义，可以使用两种时间语义：处理时间和事件时间。分别对每种的时间语义的使用方式进行了详细解释。