Flink SQL 原理及使用入门

大数据以离线计算居多，大数据越实时越有价值。数据价值最大化的有效方式就是通过实时流计算技术（Flink/Spark 等）快速把计算结果反馈给用户，提高转化率，保证线下产品的正常运行。而 SQL 是通用语言，容易上手，下面就介绍下 Flink SQL 基本能力。

1. Get Started

Flink SQL 是 Flink 高层 API，语法遵循 ANSI SQL 标准。示例如下

复制代码

```
SELECT car_id, MAX(speed), COUNT(speed)
FROM drive_data
WHERE speed > 90
GROUP BY TUMBLE (proctime, INTERVAL '30' SECOND), car_id
```

Flink SQL 是在 Flink Table API 的基础上发展起来的，与上述示例对应的 Table API 示例如下

复制代码

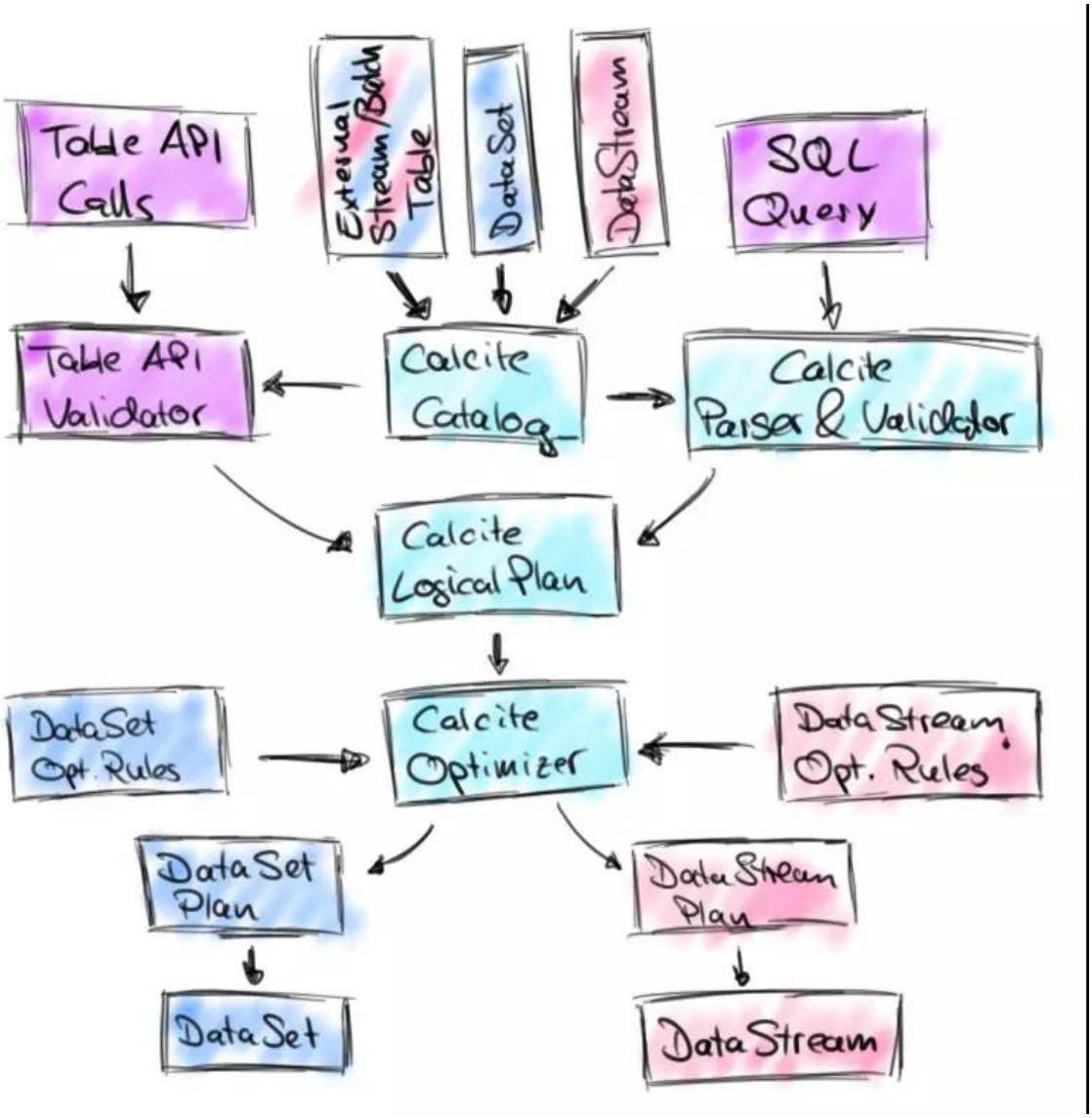
```
table.where('speed > 90)
  .window(Tumble over 30.second on 'proctime as 'w)
  .groupBy('w, 'car_id)
  .select('car_id, 'speed.max, 'speed.count)
```

上述示例使用 Scala 代码，结合隐式转换和中缀表示等 Scala 语法，Table API 代码看起来非常接近 SQL 表达。

2. 架构原理

老版本的 Table API 通过类似链式调用的写法，构造一棵 Table Operator 树，并对各个树节点做代码生成，转化成 Flink 低层 API 调用代码，即 DataStream/DataSet API。

从 2016 年开始，开源社区已经有大量 SQL-on-Hadoop 的成熟解决方案，包括 Apache Hive、Apache Impala、Apache Drill 等等，都依赖 Apache Calcite 提供的 SQL 解析优化能力，Apache Calcite 当时已经是一个非常流行的业界标准 SQL 解析和优化框架。于此同时，随着在实时分析领域中 Flink 的应用场景增加，对 SQL API 的呼声渐高，于是社区开始在 Apache Calcite 的基础上构建新版本的 Table API，并增加 SQL API 支持。



新版本的 Table & SQL API 在原有的 Table API 基础上，由 Calcite 提供 SQL 解析和优化能力，将 Table API 调用和 SQL 查询统一转换成 Calcite 逻辑执行计划（Calcite RelNode 树），并对此进行优化和代码生成，最终同样转化成 Flink DataStream/DataSet API 调用代码。

3. DDL & DML

完整的 SQL 语法由 DDL（data definition language）和 DML（data manipulation language）两部分组成。Flink SQL 目前只支持 DML 语法，而包含数据流定义的 DDL 语法仍需通过代码实现。

国内各大公有云厂商中，华为云和阿里云都提供了基于 Flink SQL 的实时流计算服务，各自定义了一套 DDL 语法，语法大同小异。以华为云为例，数据流定义以`CREATE STREAM`为关键字，具体的 DDL 写法示例如下

复制代码

```
CREATE SOURCE STREAM driver_behavior (car_id STRING, speed INT, collect_time LONG)
WITH (
  type = "kafka",
  kafka_bootstrap_servers = "10.10.10.10:3456,10.10.10.20:3456",
  kafka_group_id = "group1",
```

```
  kafka_topic = "topic1",
  encode = "csv",
  field_delimiter = ","
) TIMESTAMP BY collect_time.ROWTIME;

CREATE SINK STREAM over_speed_warning (message STRING)
WITH (
  type = "smn",
  region = "cn-north-1",
  topic_urn = "urn:smn:cn-north-1:38834633fd6f4bae813031b5985dbdea:warning",
  message_subject = "title",
  message_column = "message"
);
```

DDL 中包含输入数据流和输出数据流定义，描述实时流计算的数据上下游生态组件，在上述例子中，输入流（`SOURCE STREAM`）类型是 Kafka，`WITH`子句描述了 Kafka 消费者相关配置。输出流（`SINK STREAM`）类型是 SMN，是华为云消息通知服务的缩写，用于短信和邮件通知。

数据从 Kafka 流入，向 SMN 服务流出，而中间的数据处理逻辑由 DML 实现，具体的 DML 写法示例如下

复制代码

```
INSERT INTO over_speed_warning
SELECT "your car speed (" || CAST(speed as CHAR(20)) || ") exceeds the maximum speed.
FROM (
  SELECT car_id, MAX(speed) AS speed, COUNT(speed) AS overspeed_count
  FROM driver_behavior
  WHERE speed > 90
  GROUP BY TUMBLE (collect_time, INTERVAL '30' SECOND), car_id
)
WHERE overspeed_count >= 3;
```

以上 DML 语句，描述了在 30 秒内车辆累计超速三次时，向作为输出流的下游 SMN 组件输出告警消息。DML 语句中`INSERT INTO`关键字后紧接着输出流名，而`FROM`关键字后紧接着输入流名，`SELECT`子句表达输出的内容，`WHERE`子句表达输出需要满足的过滤条件。上述例子使用到了 SQL 子查询，外层`FROM`后跟着一整个`SELECT`子句，为了方便理解，我们也可以把子查询语法转化成等价的临时流定义表达，在华为云实时流计算服务的 DDL 语法中支持了这种特性，与上述 DML 写法等价的示例如下

复制代码

```
CREATE TEMP STREAM over_speed_info (car_id STRING, speed INT, overspeed_count INT);
```

```
INSERT INTO over_speed_info
SELECT car_id, MAX(speed) AS speed, COUNT(speed) AS overspeed_count
FROM driver_behavior
WHERE speed > 90
GROUP BY TUMBLE (collect_time, INTERVAL '30' SECOND), car_id;

INSERT INTO over_speed_warning
SELECT "your car speed (" || CAST(speed as CHAR(20)) || ") exceeds the maximum speed.
FROM over_speed_info
WHERE overspeed_count >= 3;
```

通过`TEMP STREAM`语法定义临时流，可以将带有子查询的 SQL 语法平铺表达，串接数据流逻辑，更容易理解。

4. 语法

Flink SQL 的核心部分是 DML 语法，基础的 DML 语法包含笛卡尔积（单表情况下只有 Scan 操作）、选择（Filter）和投影（Projection）三个数据操作部分，三者分别对应`FROM`子句、`WHERE`子句和`SELECT`子句，这三个部分的顺序代表了 DML 语句的逻辑执行顺序。较为进阶的语法包含聚合、窗口和连接（`JOIN`）等常用语法，以及排序、限制和集合等非常用语法。下表简单列举 Flink SQL 基础和常用的进阶 DML 语法句式并加以说明，其他语法元素和内建函数等详细内容，可参考 Flink SQL 文档

- 基础语法

操作	样例	备注
GROUPBY AGGREGATION	SELECT MAX(speed) FROM drive_data GROUP BY car_id	
GROUPBY WINDOW AGGREGATION	SELECT car_id, MAX(speed) FROM drive_data GROUP BY TUMBLE(proctime, INTERVAL '1' MINUTE), car_id	GroupBy 窗口每个聚合周期输出一批聚合结果
OVER WINDOW AGGREGATION	SELECT MAX(speed) OVER (PARTITION BY car_id ORDER BY proctime RANGE BETWEEN INTERVAL '30' SECOND PRECEDING AND CURRENT ROW) FROM drive_data	Over 窗口每进入一条数据就输出一条聚合结果，且所有的投影属性的 Over 窗口必须一致

- 聚合语法

操作	样例	备注
GROUPBY AGGREGATION	SELECT MAX(speed) FROM drive_data GROUP BY car_id	
GROUPBY WINDOW AGGREGATION	SELECT car_id, MAX(speed) FROM drive_data GROUP BY TUMBLE(proctime, INTERVAL '1' MINUTE), car_id	GroupBy 窗口每个聚合周期输出一批聚合结果
OVER WINDOW AGGREGATION	SELECT MAX(speed) OVER (PARTITION BY car_id ORDER BY proctime RANGE BETWEEN INTERVAL '30' SECOND PRECEDING AND CURRENT ROW) FROM drive_data	Over 窗口每进入一条数据就输出一条聚合结果，且所有的投影属性的 Over 窗口必须一致

- 连接语法

操作	样例	备注
INNER EUIQ-JOIN	SELECT * FROM drive_data INNER JOIN car_info ON drive_data.car_id = car_info.id	当前只支持等值连接
TIME-WINDOWED JOIN	SELECT * FROM drive_data d, camera_data c WHERE d.car_id = c.car_id AND d.proctime BETWEEN c.proctime - INTERVAL '30' SECOND AND c.proctime	
TABLE JOIN	SELECT * FROM drive_data INNER JOIN car_info ON drive_data.car_id = car_info.id	流表 Join 语法和流流 Join 语法类似，Flink SQL 目前不支持流表 Join，华为云实时流计算服务的 SQL 语法是支持的

5. 场景

目前 Flink SQL 的应用广泛，可以用在 IoT、车联网、智慧城市、日志分析、ETL、实时大屏、实时告警、实时推荐等等。在 IoT 和车联网等行业对 Flink 有更高的要求，如时间地理函数、CEP SQL、StreamingML 等，各个云厂商都有不同程度的实现，华为云实时流计算在这方面特性最为丰富。

本文转载自 华为云产品与解决方案 公众号。

原文链接： https://mp.weixin.qq.com/s/au-X4obr31ivTuZpuVdlMA