深度探索Flink SQL

贺小令 · 阿里巴巴 / 技术专家

Apache Flink Community China

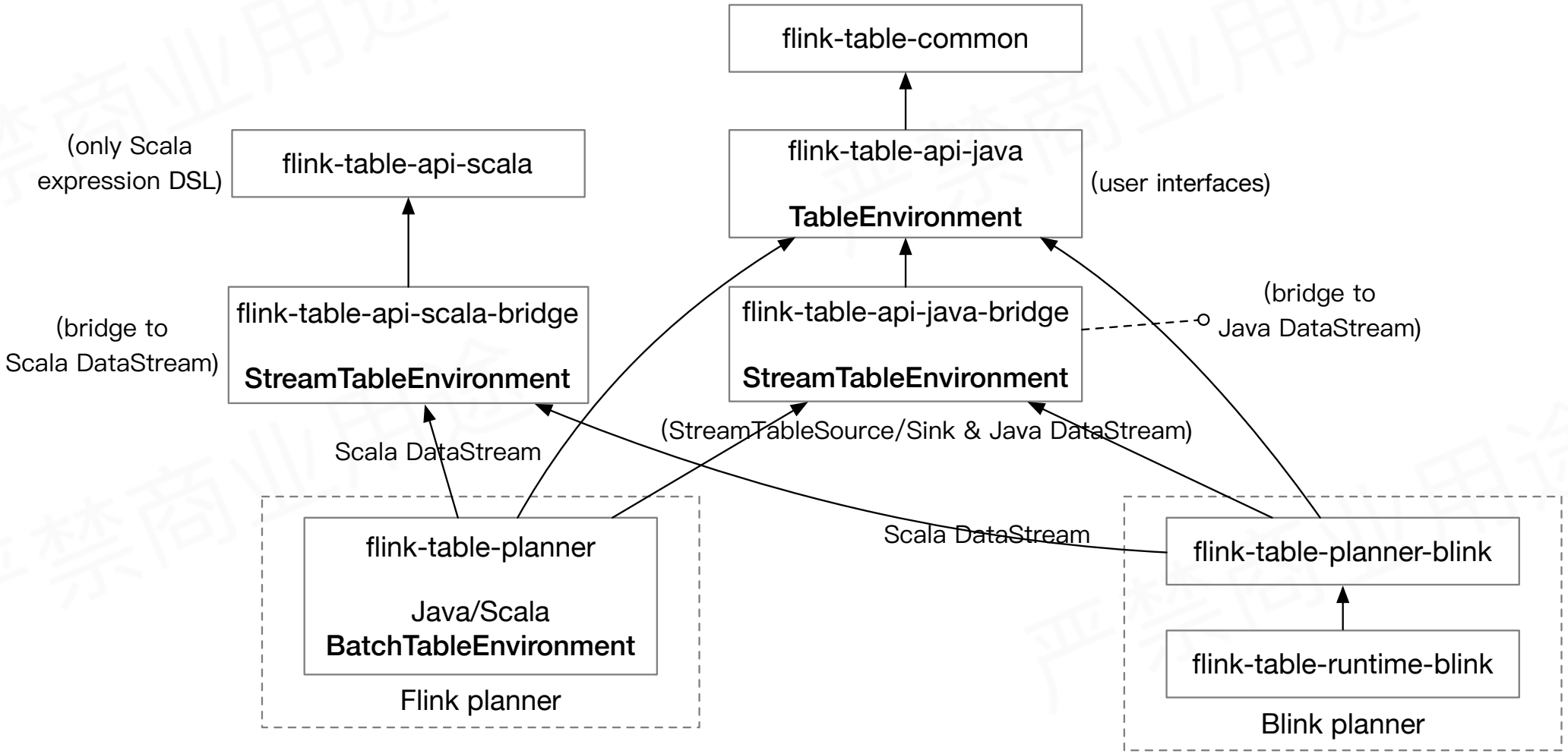
CONTENT

目录 >>

01

New TableEnvironment

TableEnvironments [FLIP-32]



适用场景

	Flink Stream	Flink Batch	Blink Stream	Blink Batch	From/To DataStream	From/To DataSet	UDAF/ UDTF
TableEnvironment	✅	❌	✅	✅	❌	❌	❌ (Java/Scala 类型推导 还没统一)
Java/Scala StreamTableEnvironment	✅	❌	✅ 不支持 分段优化	❌	✅	❌	✅
Java/Scala BatchTableEnvironment	❌	✅	❌	❌	❌	✅	✅

Examples

- Blink Batch

```
EnvironmentSettings settings = EnvironmentSettings.newInstance().useBlinkPlanner().inBatchMode().build();
TableEnvironment tEnv = TableEnvironment.create(settings);
tEnv…
tEnv.execute(“job name”);
```

- Blink Stream

```
EnvironmentSettings settings = EnvironmentSettings.newInstance().useBlinkPlanner().inStreamingMode().build();
StreamExecutionEnvironment execEnv = …
StreamTableEnvironment tEnv =  StreamTableEnvironment.create(execEnv, settings);
tEnv…
```

- Flink Batch

```
ExecutionEnvironment execEnv = ...
BatchTableEnvironment tEnv = BatchTableEnvironment.create(execEnv);
tEnv...
```

- Flink Stream

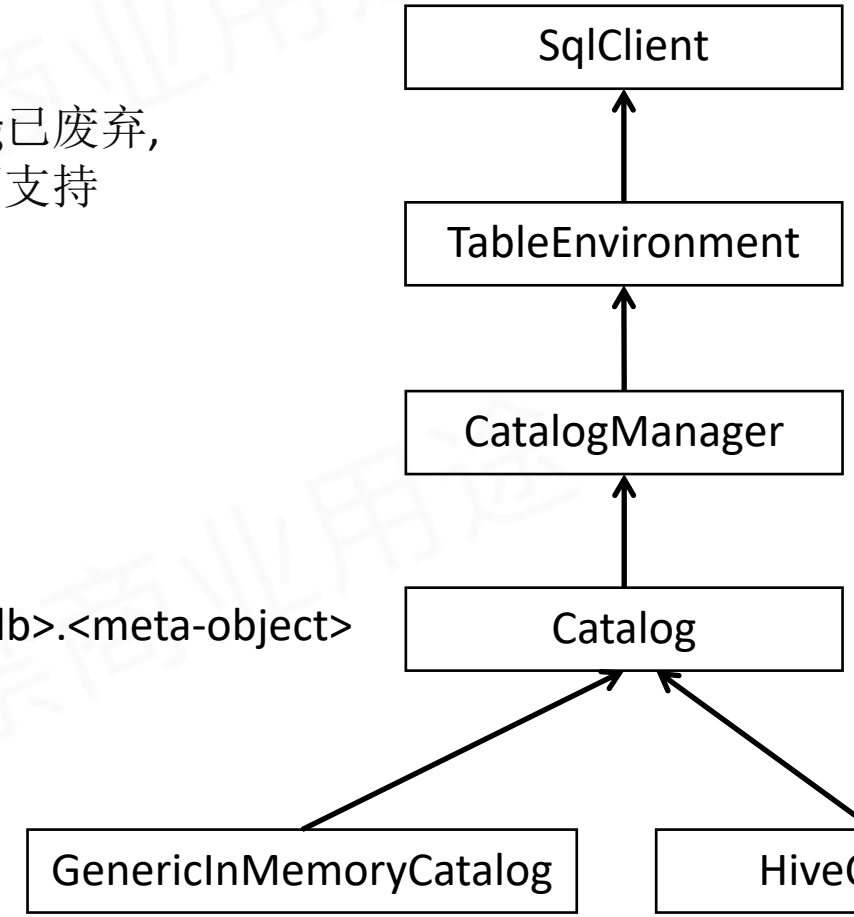
```
EnvironmentSettings settings = EnvironmentSettings.newInstance().useOldPlanner().inStreamMode().build();
TableEnvironment tEnv = TableEnvironment.create(settings);
tEnv…
tEnv.execute(“job name”);
```

02

New Catalog & DDL

New Catalog [FLIP-30]

ExternalCatalog已废弃,
Blink planner不支持



./sql_client.sh [mode] -e XXX.yaml
SHOW CATALOGS
USE CATALOG catalog_name

registerCatalog/listCatalogs/getCatalog/useCatalog

registerCatalog/getCatalogs/getCatalog/
getCurrentCatalog/setCurrentCatalog/
getCurrentDatabase/setCurrentDatabase

<catalog>.<db>.<meta-object>

create/drop/list/alter/exists
Database/Table/Partition/Function/Statistics

DDL (in tableEnv#sqlUpdate)

- CREATE TABLE

```
CREATE TABLE [[catalog_name.]db_name.]table_name (
  a int comment ’column comment',
  b bigint,
  c varchar
) comment 'table comment'
[partitioned by (b)]
WITH (
  update-mode = 'append',
  connector.type = 'kafka',
  …
)
```

计算列，watermark定义还不支持

with属性参考各个connector factory里定义:
CsvAppendTableSourceFactory
KafkaTableSourceSinkFactory
…

- CRATE VIEW

  CREATE VIEW view_name AS SELECT xxx

- DROP TABLE

  DROP TABLE [IF EXISTS] [[catalog_name.]db_name.]table_name

- DROP VIEW

  DROP VIEW [IF EXISTS] view_name

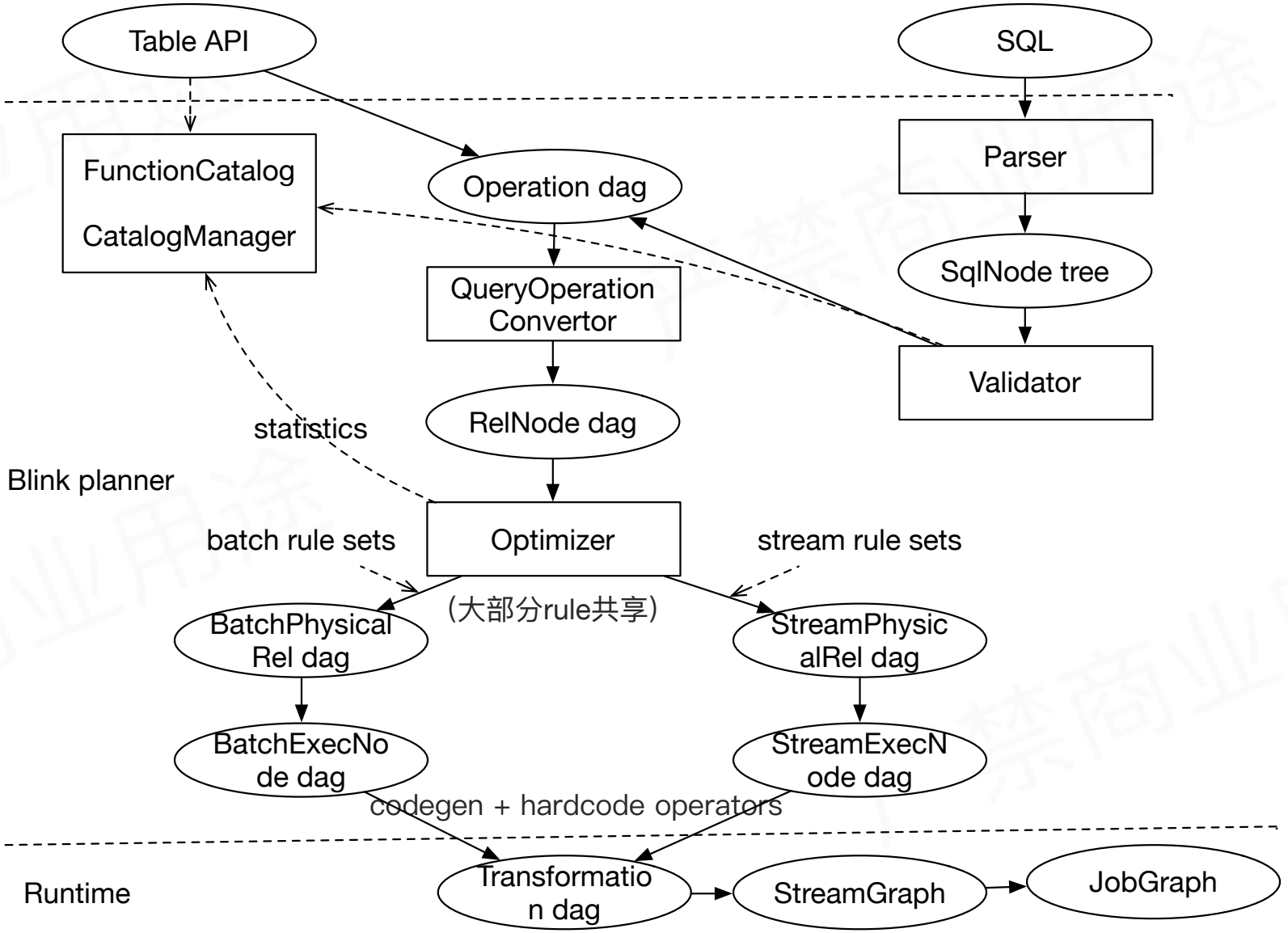
DDL (in SqlClient)

- CREATE VIEW
- DROP VIEW
- SHOW CATALOGS/DATABASES/TABLES/FUNCTIONS
- USE CATALOG xxx
- SET xxx=yyy
- DESCRIBE table_name
- EXPLAIN SELECT xxx

03

Blink planner

主要流程



改进

功能相关

- 更完整的SQL语法的支持（subquery, over, group sets等）
- 更丰富、高效的算子
- 更完善的cost模型，对接了catalog中的statistics
- join reorder
- shuffle service (only for batch)
- …

改进

性能相关

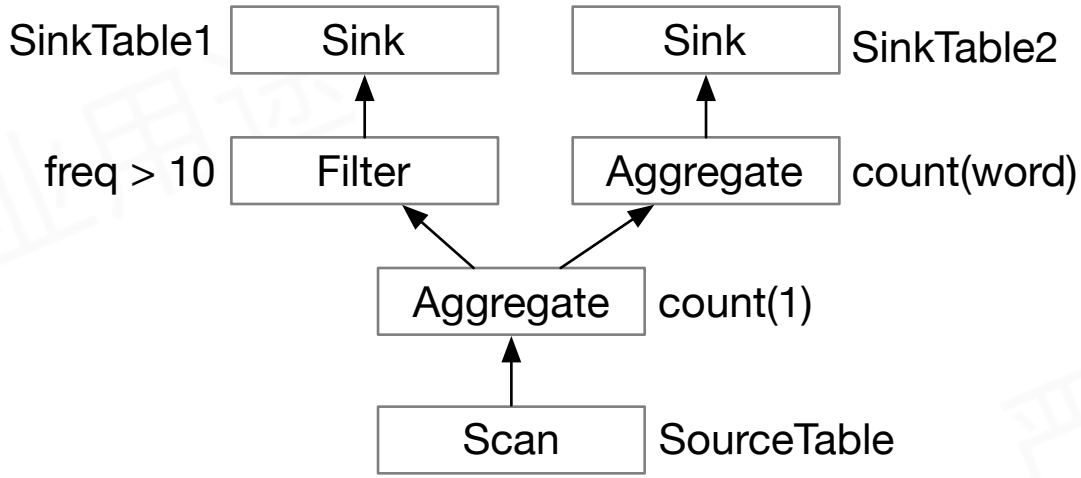
- 分段优化 & sub-plan reuse
- 更丰富的优化rule
- 更高效的数据结构BinaryRow
- mini-batch (only for stream)
- 省多余的shuffle & sort (for batch now)
- …

分段优化

create view MyView as select word, count(1) as freq from SourceTable group by word;

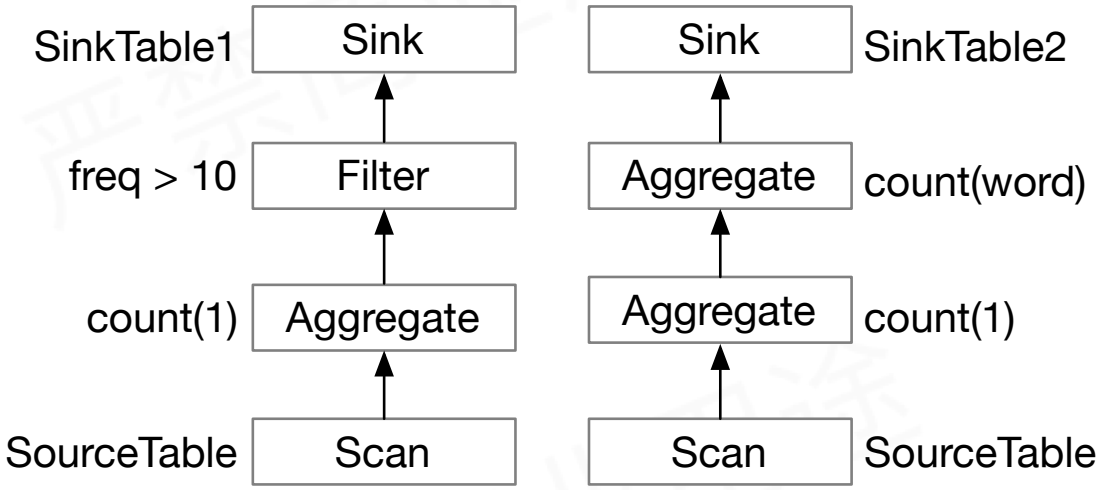
insert into SinkTable1 select * from MyView where freq > 10;

insert into SinkTable2 select count(word) as freq2, freq from MyView group by freq;

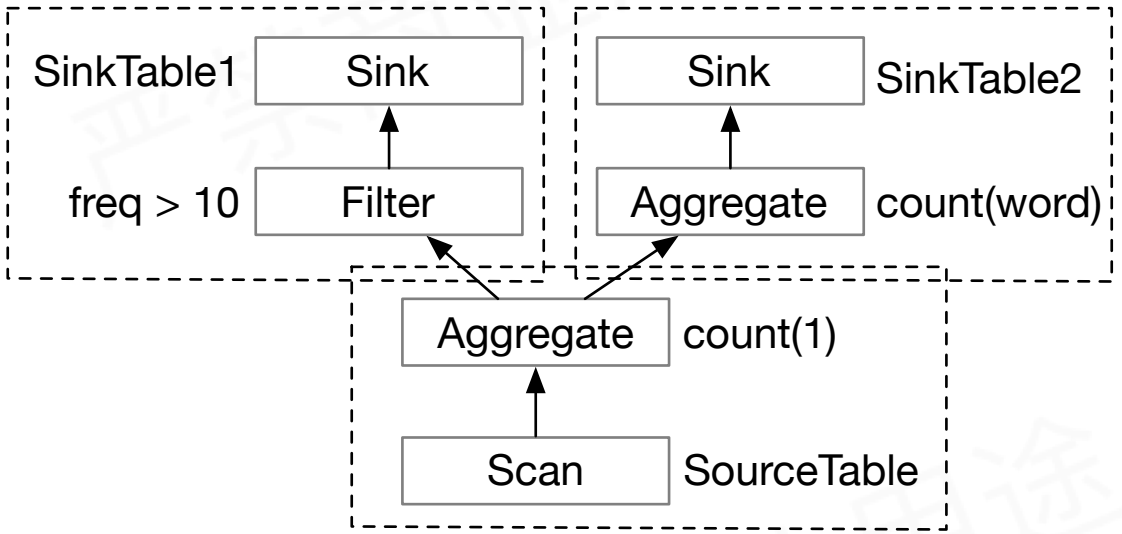


(原始的逻辑tree)

分段优化（续）



Flink planner按每个Sink单独优化，各个Sink的计算链路相互独立，有重复计算

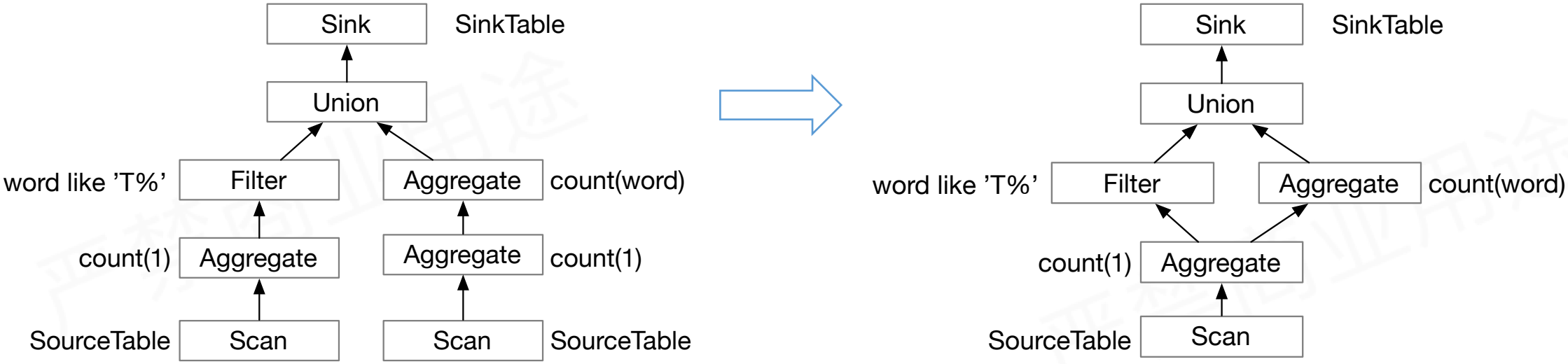


Blink planner先分段（能优化的最大公共子图），每段独立优化

分段优化解的是多Sink优化问题（DAG优化）

sub-plan reuse

insert into SinkTable
select freq from (select word, count(1) as freq from SourceTable group by word) t where word like 'T%'
union all
select count(word) as freq2 from (select word, count(1) as freq from SourceTable group by word) t group by freq;



sub-plan reuse (续)

相关配置

- *table.optimizer.reuse-sub-plan-enabled*，默认开启
- *table.optimizer.reuse-source-enabled*，默认开启

在Batch模式下，sub-plan reuse可能造成死锁(hash-join, nested-loop-join先读build端再读probe端)，框架会将probe端数据落盘来解死锁。落盘会有额外开销，此时用户可根据情况来调整配置。

sub-plan reuse解的是优化结果的子图复用问题

agg 分类

- group agg
- over agg
- window agg
- table agg

select count(a) from t group by b

select count(a) over (partition by b order by c) from t

select count(a) from t group by tumble(ts, interval ’10’ second), b

tEnv.scan(“t”).groupBy(‘a).flatAggregate(flatAggFunc(‘b as (‘c, ‘d)))

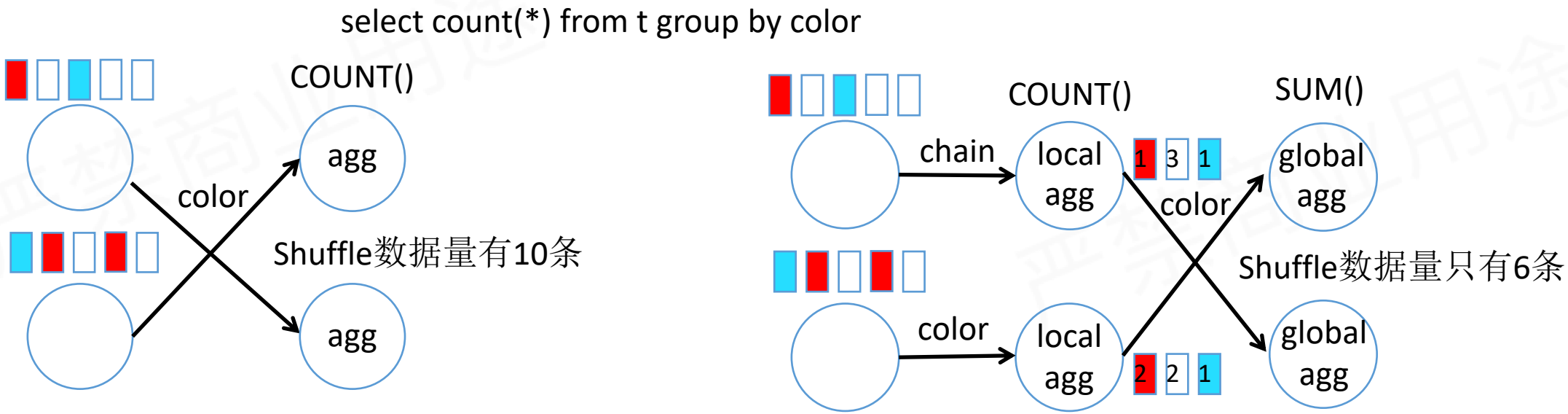
group agg 优化

- local/global
  - 减少网络shuffle数据
  - agg function可merge (sum(a) -> local sum(a) + global sum(local result))
- distinct agg
  - 改写为两层agg
  - Stream下是为了解数据热点问题 (state需要存所有input数据)

local/global agg

必要条件

- agg的所有agg function都是mergeable (实现了merge方法)
- *table.optimizer.agg-phase-strategy*为AUTO或TWO_PHASE
- Stream下，mini-batch开启；Batch下，AUTO会根据cost选择



distinct agg

- Batch下，强制改写

第一层求distinct值和非distinct agg function的值，第二层求distinct agg function的值

- Stream下，必要条件
  - 必须是支持的agg function:

    avg/count/min/max/sum/first_value/last_value/concat_agg/single_value
  - *table.optimizer.distinct-agg.split.enabled*开启(默认关闭)

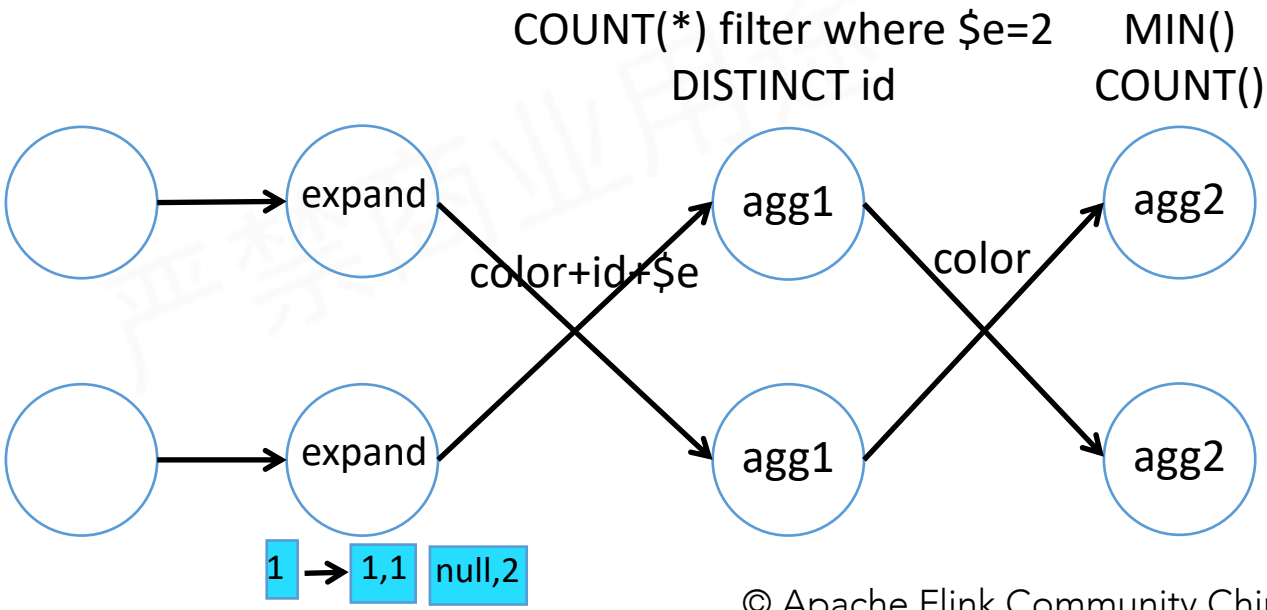
distinct agg (batch)

```
select color, count(distinct id), count(*)
from t group by color
```

可手工改写为

```
select color, count(id), min(cnt) from (
  select color, id, count(*) filter (where $e=2)  as cnt
  from (
    select color, id, 1 as $e from t  -- for distinct id
    union all
    select color, null as id, 2 as $e from t  -- for count(*)
  ) group by color, id, $e
) group by color
```

Expand算子

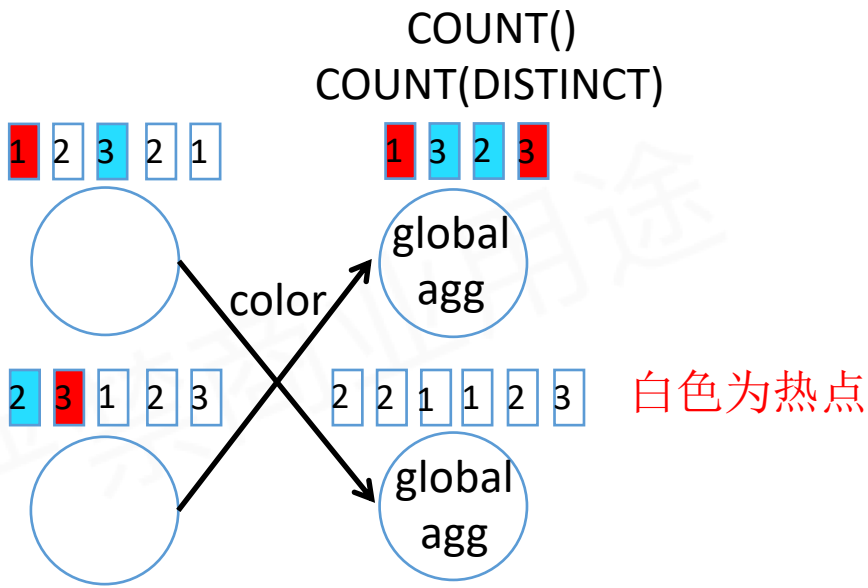


distinct agg (stream)

```
select color, count(distinct id), count(*)
from t group by color
```

可手工改写为

```
select color, sum(dcnt), sum(cnt) from (
   select color, count(distinct id) as dcnt, count(*) as cnt
   from t  group by color, mod(hash_code(id), 1024)
) group by color
```



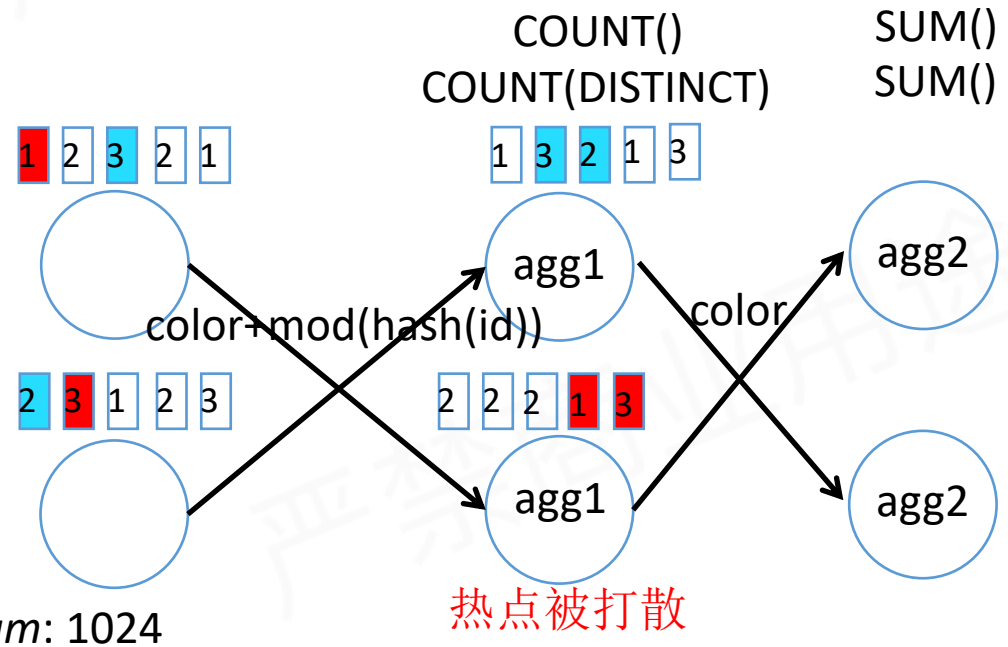


table.optimizer.distinct-agg.split.bucket-num: 1024
(如配置太小可能数据没被完全打散，还可能有热点)

THANKS

关注 Apache Flink 社区微信公众号 Vererica

由 Apache Flink Community China 运营管理，旨在联合国内的 Flink 大 V，向国内宣传和普及 Flink 相关的技术。

持续输出 Flink 最新社区动态：

- Release 及新特性解读
- Meetup
- 入门教程
- 应用案例
- 源码解析