

 2020 POSTGRESQL  
CONFERENCE CHINA

# 第十届PostgreSQL中国技术大会

开源 自研 新机遇

Greenplum中的多阶段聚集实现 - 李晓亮, 陈金豹



 2020 POSTGRES SQL  
CONFERENCE CHINA

# 第十届PostgreSQL 中国技术大会

开源 自研 新机遇

## 目录 / CONTENTS

- Greenplum简介
- 聚集在MPP架构中的问题
- Greenplum如何实现多阶段聚集
- Distinct-Qualified Aggregates的实现和优化
- Rollup的实现和优化



# 关于Greenplum

<https://github.com/greenplum-db/gpdb>

greenplum-db / gpdb

Watch 431 Star 4.3k Fork 1.2k

Code Issues 327 Pull requests 90 Actions Projects 6 Wiki Security

master Go to file Code

About Greenplum Database  
greenplum.org  
Readme  
View license

Releases 103  
6.12.0 Latest  
2 days ago  
+ 102 releases

Packages  
No packages published

Contributors 244  
+ 233 contributors

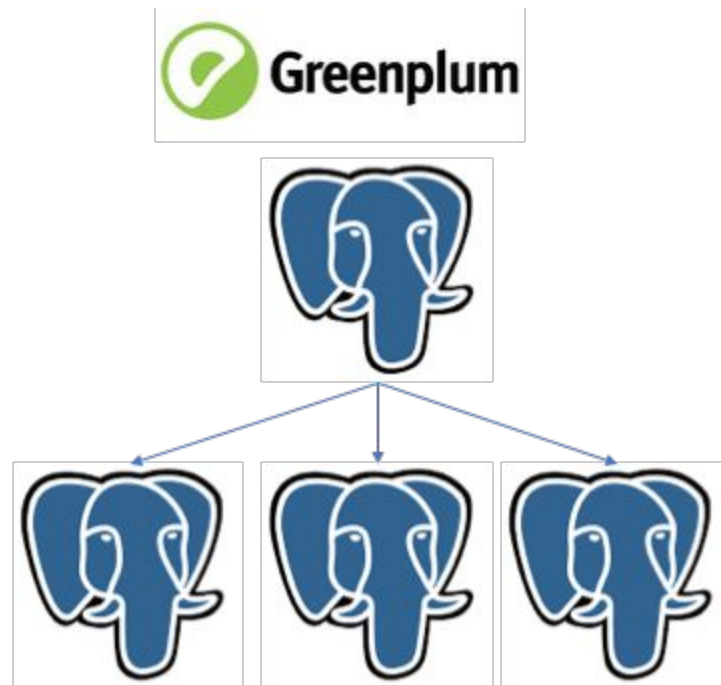
Commit	Message	Time
xiong-gang	Correctly seek to the end of buffile that contains ...	4 hours ago 62,958
.github	CONTRIBUTING.md: Add guidelines to run pgindent	3 years ago
concourse	Update greenplum-database-release default repo t...	3 days ago
config	Configure database with Python 3 by default	last month
contrib	Re-implement the ereport() macro using __VA_ARG...	29 days ago
doc	Merge with PostgreSQL version 12 (up to a point b...	last month
gpAux	Disable gpcloud by default in configure	14 days ago
gpMgmt	Redirect the error to log message	13 hours ago
gpcontrib	Replace Insist() with Assert()	24 days ago
gpdb-doc	Docs - add note regarding change in TRUNCATE b...	6 days ago
hooks	concourse git-hook: check for pipeline generation (...)	3 years ago
src	Correctly seek to the end of buffile that contains m...	4 hours ago
.dir-locals.el	Make Emacs perl-mode indent more like perltidy.	2 years ago
.editorconfig	Merging Orca .editorconfig into gpdb file	7 months ago
.git-blame-ignore-revs	Add ORCA formatting commit to .git-blame-ignore-...	last month
.gitattributes	Add XSL stylesheet to fix up SVG files	17 months ago
.gitignore	Merge with PostgreSQL version 12 (up to a point b...	last month





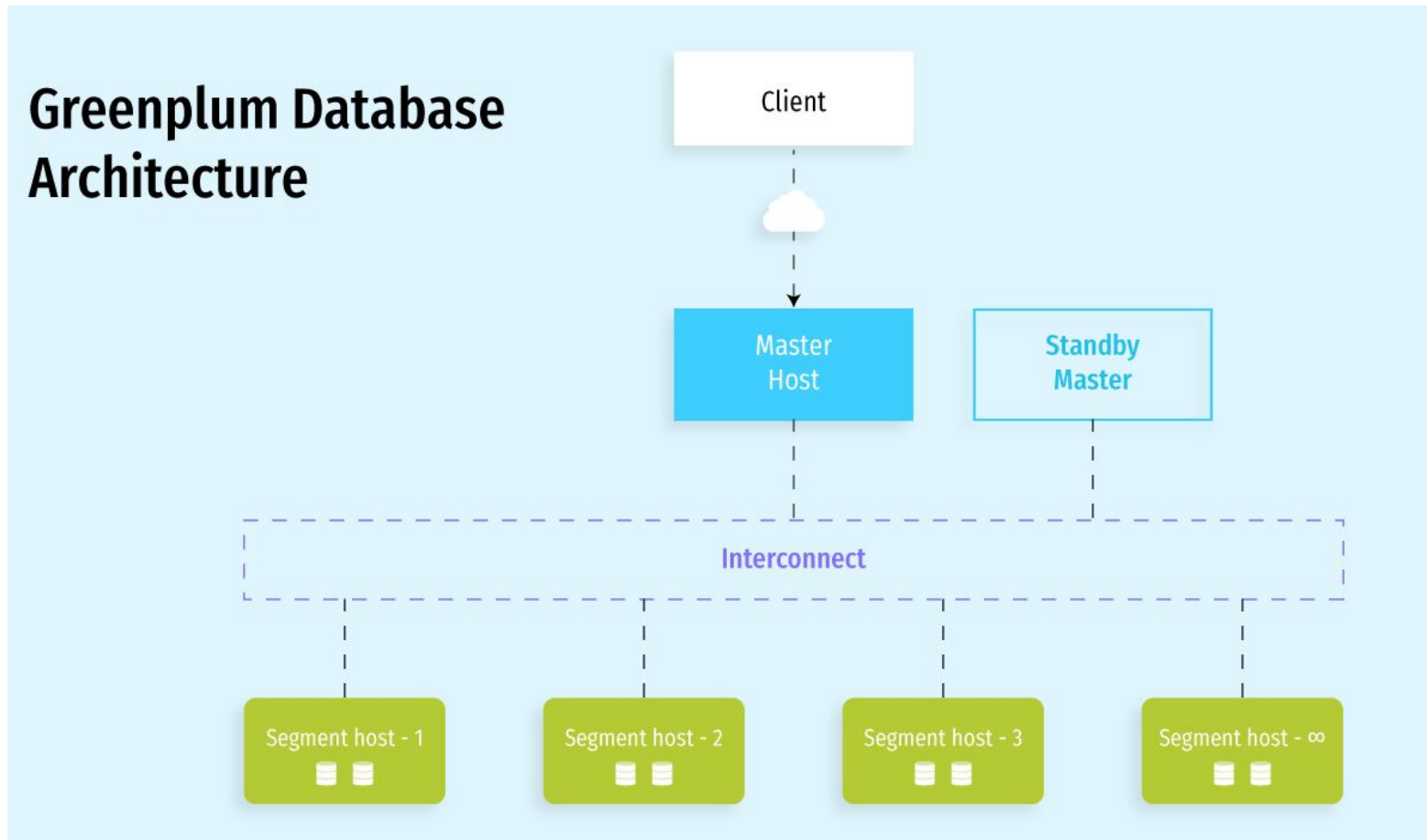


# 关于Greenplum



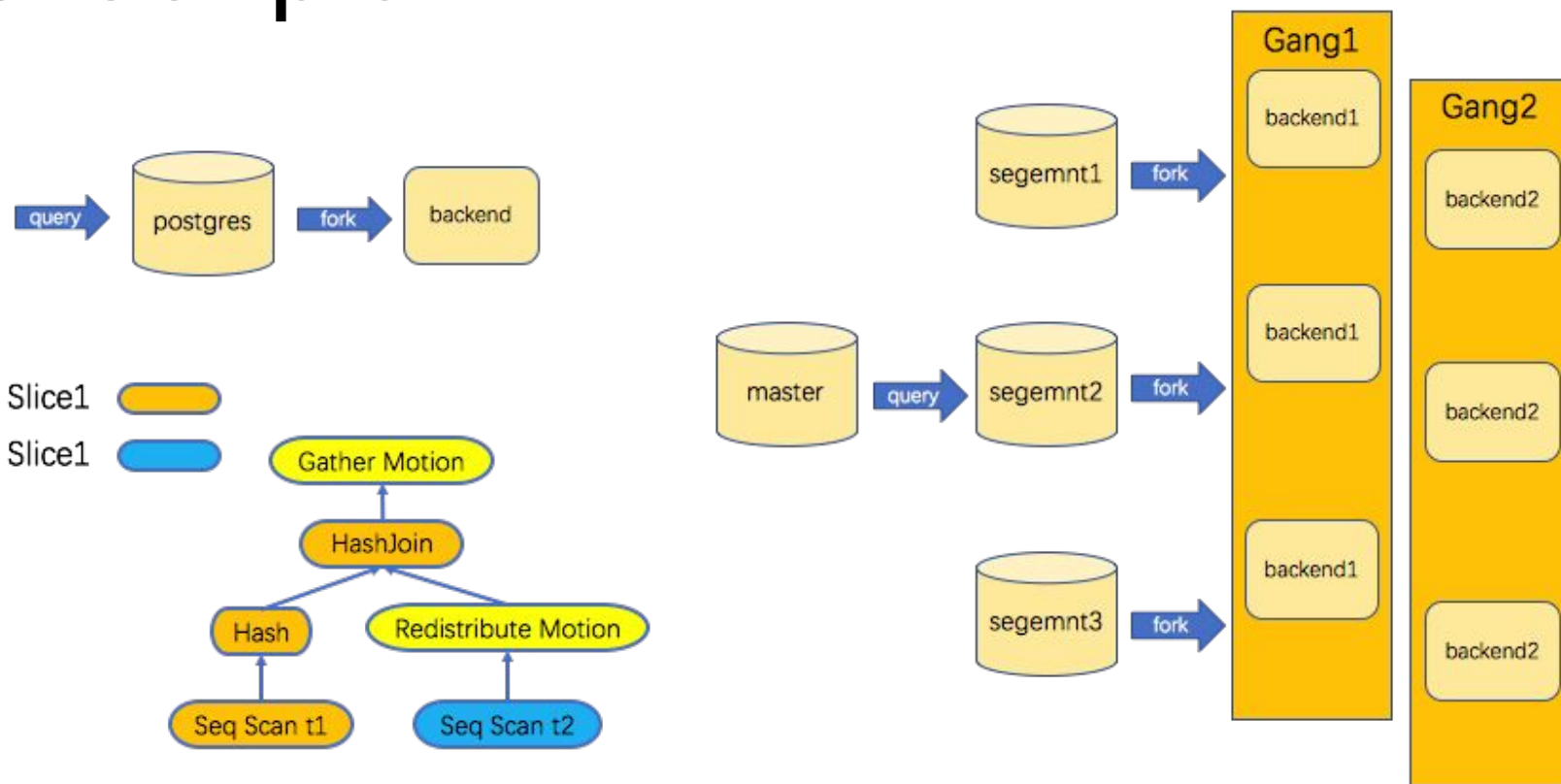


# 关于Greenplum





# 关于Greenplum







# 聚集在MPP架构中的问题

Aggregate functions perform calculations on a set of values and return a single value.

```
SELECT avg(t.b) FROM t GROUP  
BY t.a;
```





# 聚集在MPP架构中的问题

```
CREATE TABLE sales  
(  
    id int NOT NULL,  
    brand VARCHAR,  
    type INT,  
    quantity INT  
);
```

```
INSERT INTO sales  
(id, brand, type,  
quantity) VALUES  
(1, 'A', 1, 100),  
(2, 'A', 2, 200),  
(3, 'B', 1, 100),  
(4, 'B', 2, 300);
```







# 聚集在MPP架构中的问题

Postgre  
S

```
demo=# explain (costs false) select  
avg(quantity) from sales;
```

QUERY PLAN

---

Aggregate

-> Seq Scan on sales

```
demo=# explain (costs false)  
select avg(quantity) from sales  
group by brand;
```

QUERY PLAN

---

HashAggregate

Group Key: brand

-> Seq Scan on sales





# 聚集在MPP架构中的问题

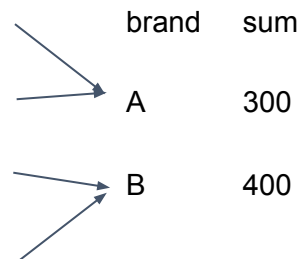
PostgreSQL

A 200  
B 100  
B 300  
A 100



A 200  
A 100  
B 300  
B 100

transfunc



finalfunc

brand	sum	N	type	avg
A	300	2	A	150
B	400	2	B	200







# Greenplum如何实现多阶段聚集

```
CREATE TABLE sales  
(  
    id int NOT NULL,  
    brand VARCHAR,  
    type INT,  
    quantity INT  
);
```

```
INSERT INTO sales (id,  
brand, type, quantity)  
VALUES  
(1, 'A', 2, 100),  
(2, 'A', 1, 200),  
(3, 'B', 2, 100),  
(4, 'B', 1, 300),  
(5, 'A', 2, 200),  
(6, 'A', 1, 400),  
(7, 'B', 2, 100),  
(8, 'B', 1, 400);
```





# Greenplum如何实现多阶段聚集

Greenplu  
m

```
demo=# explain (costs false) select avg(quantity) from sales group by brand;
```

## QUERY PLAN

---

Gather Motion 3:1 (slice2; segments: 3)

-> **GroupAggregate**

Group Key: sales.brand

-> Redistribute Motion 3:3 (slice1; segments: 3)

Hash Key: sales.brand

-> **GroupAggregate**

Group Key: sales.brand

-> Seq Scan on sales



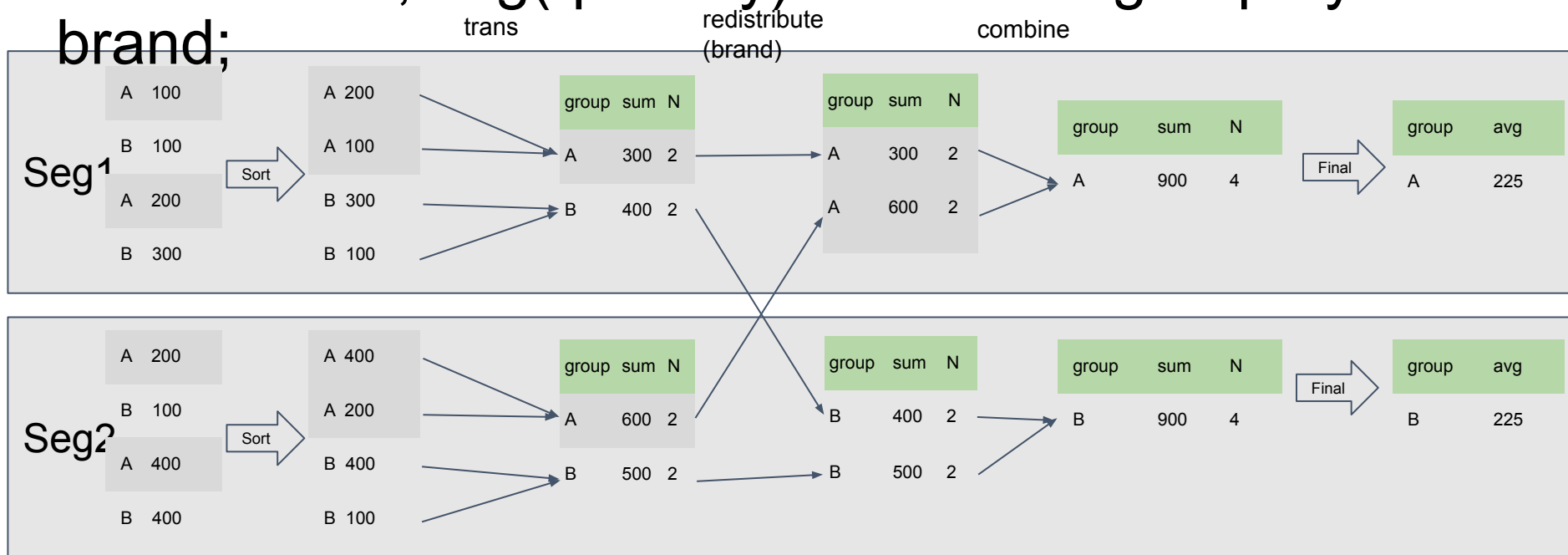




# Greenplum如何实现多阶段聚集

Greenplum

select brand, avg(quantity) from sales group by brand;





# Greenplum如何实现DQA

Greenplu  
m

```
demo=# explain (costs false, verbose) select brand, avg(distinct quantity) from sales group by brand;
```

## QUERY PLAN

---

```
Gather Motion 3:1 (slice2; segments: 3)
  Output: sales.brand, (avg(sales.quantity))
  -> HashAggregate
    Group Key: sales.brand
    -> HashAggregate
      Group Key: sales.brand, sales.quantity
      -> Redistribute Motion 3:3 (slice1; segments: 3)
        Hash Key: sales.brand
        -> HashAggregate
          Group Key: sales.brand, sales.quantity
          -> Seq Scan on public.sales
```



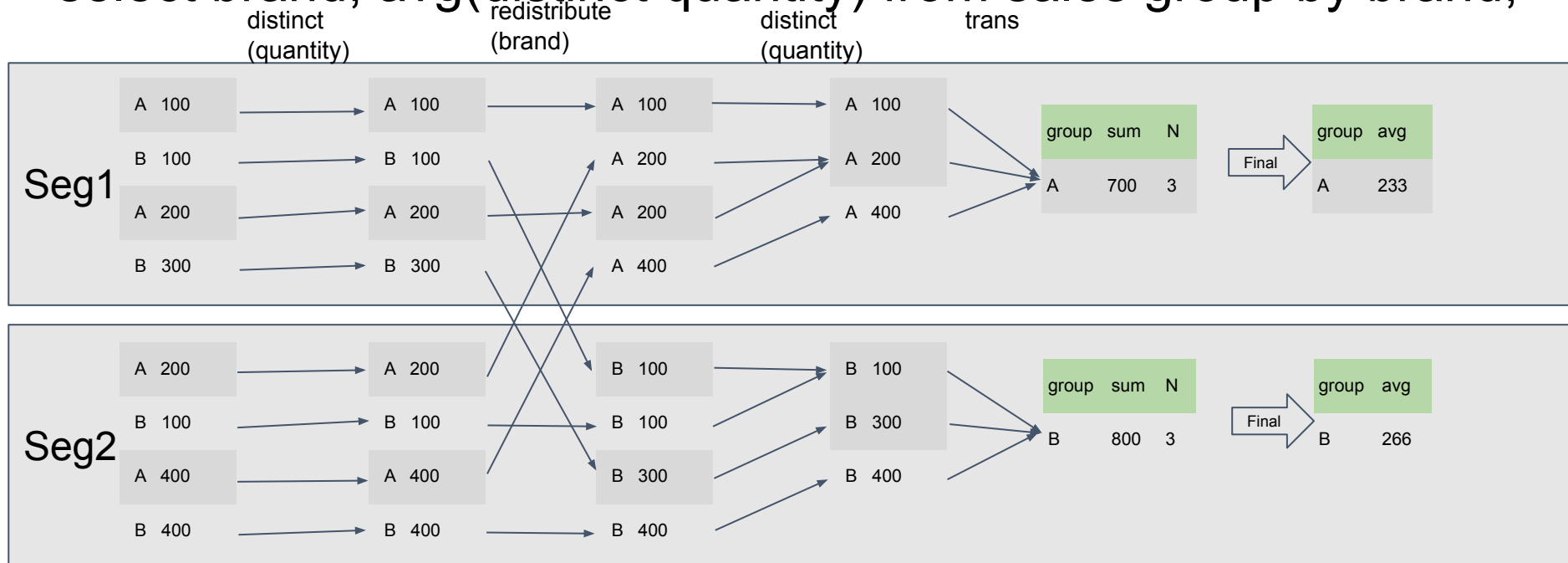




# Greenplum如何实现DQA

Greenplu

select brand, avg(distinct quantity) from sales group by brand;<sup>m</sup>

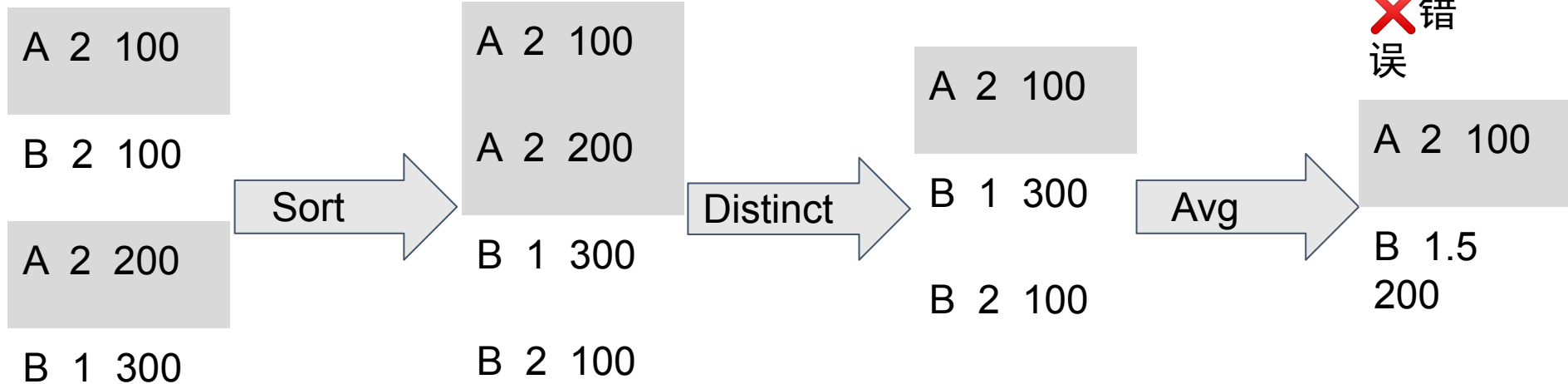




# Greenplum如何实现Multi DQA

Greenplum

demo=# explain (costs false, verbose) select brand, avg(distinct type), avg(distinct quantity) from sales group by brand;

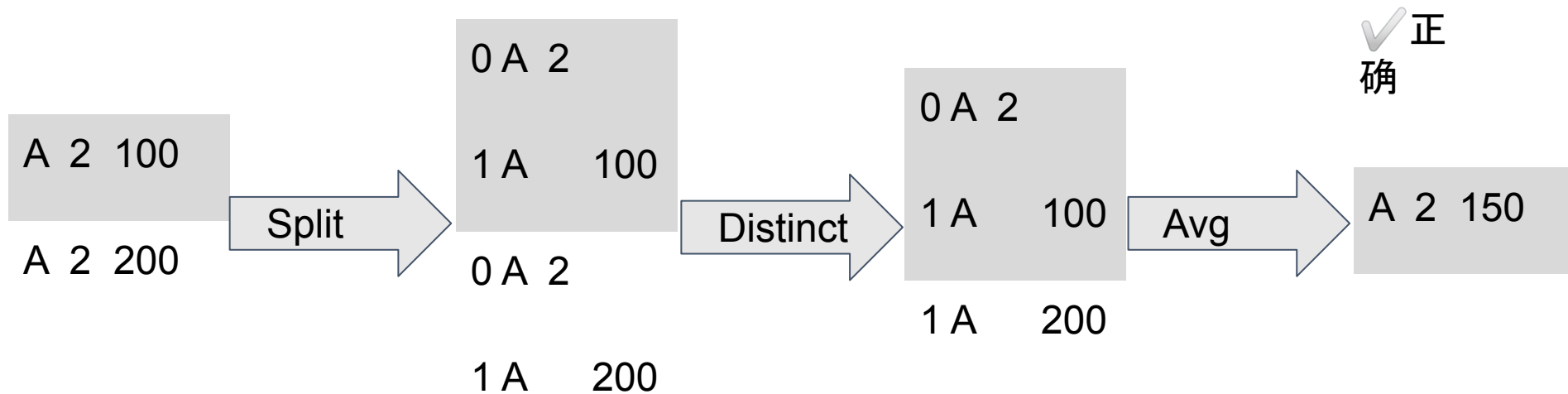




# Greenplum如何实现Multi DQA

Greenplum

demo=# explain (costs false, verbose) select brand, avg(distinct type), avg(distinct quantity) from sales group by brand;







# Greenplum如何实现Rollup

Greenplu  
m

```
select
  brand, type,
  avg(quantity)
from
  sales
group by
  grouping sets (
    (brand, type),
    (brand),
    (type),
    ()
  );
```

```
select
  brand, type, avg(quantity)
from
  sales
group by
  brand, type
union all
select
  brand, NULL,
  avg(quantity)
from
  sales
group by
  brand
union all
```

```
select
  brand, type, avg(quantity) from
  sales
group by
  rollup (brand, type);
```

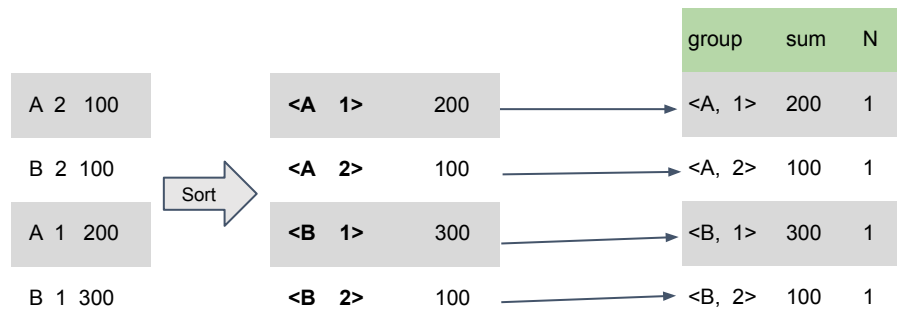




# Greenplum如何实现Rollup

Greenplum

select brand, type, avg(quantity) from sales group by rollup (brand, type);

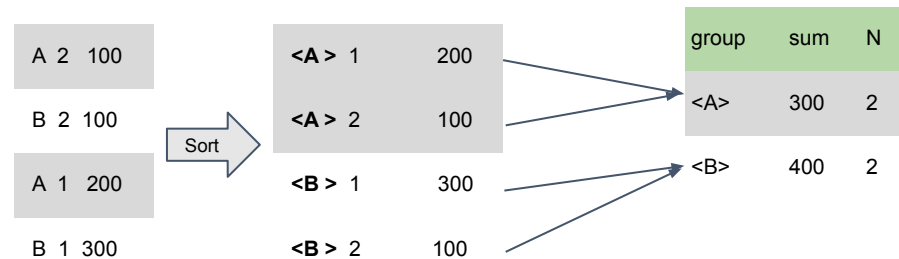




# Greenplum如何实现Rollup

Greenplu  
m

select brand, type, avg(quantity) from sales group by  
rollup (brand, type);



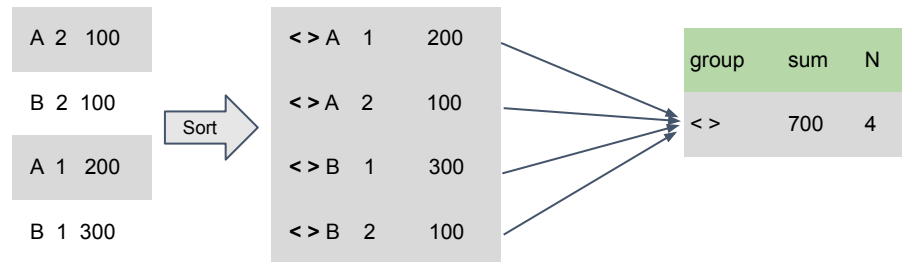




# Greenplum如何实现Rollup

Greenplum

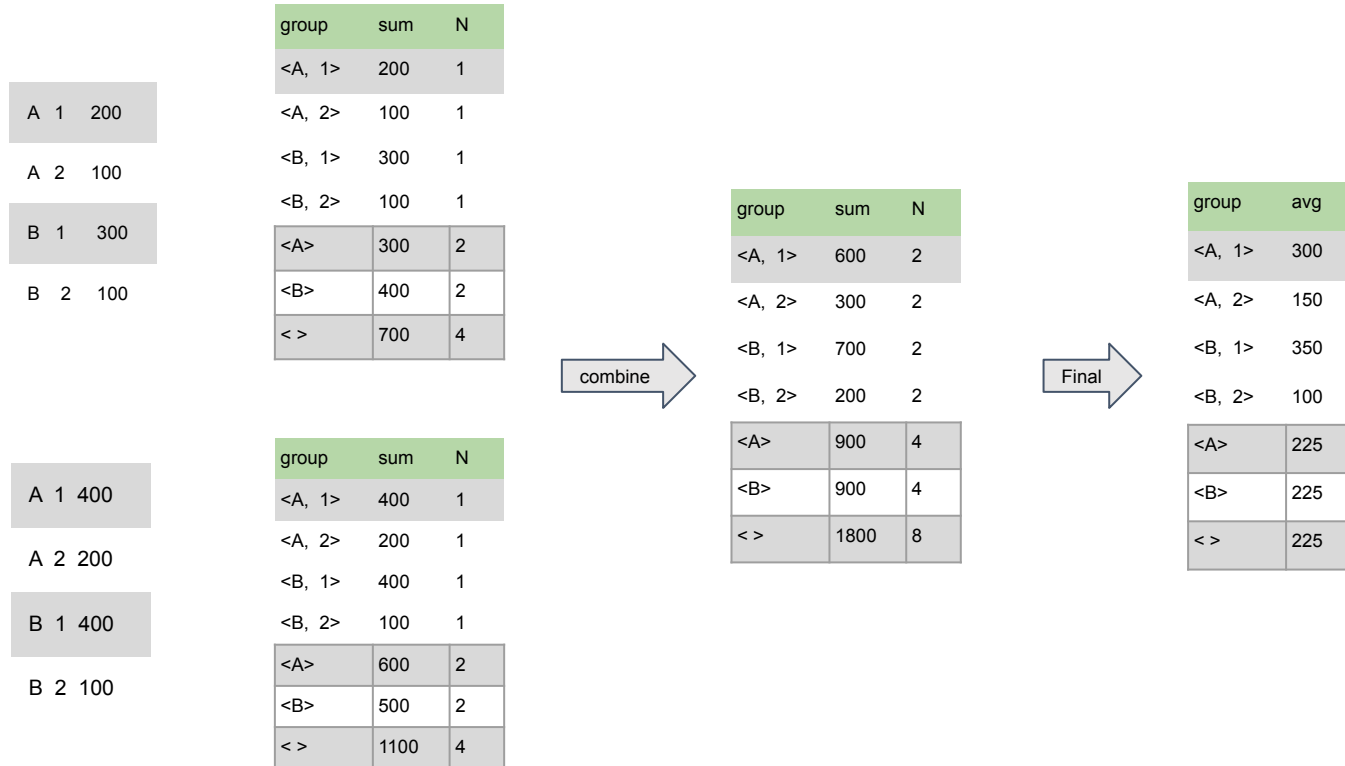
```
select brand, type, avg(quantity) from sales group by  
rollup (brand, type);
```





# Greenplum如何实现Rollup

Greenplum





# Greenplum如何实现Rollup

```
explain (costs off) select brand, type, avg(quantity) from sales group by rollup (brand, type);  
QUERY PLAN
```

Greenplu  
m

-----  
Gather Motion 3:1 (slice2; segments: 3)

-> **HashAggregate**

Group Key: partial\_aggregation.brand, partial\_aggregation.type, partial\_aggregation."grouping",  
partial\_aggregation."group\_id"

-> Subquery Scan on partial\_aggregation

-> Redistribute Motion 3:3 (slice1; segments: 3)

Hash Key: "rollup".brand, "rollup".type

-> **GroupAggregate**

Group Key: "rollup"."grouping", "rollup"."group\_id", "rollup".brand, "rollup".type

-> Subquery Scan on "rollup"

-> **GroupAggregate**

Group Key: rollup\_1.brand, rollup\_1."grouping", rollup\_1."group\_id", rollup\_1.type

-> Subquery Scan on rollup\_1

-> **GroupAggregate**

Group Key: sales.brand, sales.type

-> Sort

Sort Key: sales.brand, sales.type

-> Seq Scan on sales







# GREENPLUM DATABASE®



**微信技术讨论群**  
微信搜索添加“gp\_assistant”  
加入技术讨论



**微信公众号**  
搜索添加“Greenplum中文社区”  
技术干货、行业热点、活动预告



2020 POSTGRESQL CONFERENCE CHINA  
第十届PostgreSQL中国技术大会

# THANKS

谢谢观看



GREENPLUM  
DATABASE

开源 自研 新机遇