MITSUBISHI ELECTRIC
Changes for the Better

PostgreSQL Development Conference 2024

A Method for Scale-Out of Aggregation Using PostgreSQL's Built-in Sharding

2024/05/31

MITSUBISHI ELECTRIC CORPORATION

Yuki Fujii

Introduction

- Mitsubishi Electric actively contributes to OSS communities to quickly catch up the latest technology. I develop PostgreSQL's extensions such as fdw in my work.
- In 2021, Mr. Pyhalov of Postgres Professional posted a patch which allows PostgreSQL to scale out aggregation with built-in sharding★1.
- I have posted an improved version of this patch, which expands the scope of parallelizable aggregate functions.
- At the end of 2023, several committers pointed out technical issues with the patch I had posted.
- Today, I would like to explain to you the status of responses to these comments and discuss how to advance this patch.

★1 The thread is “Partial aggregate pushdown”.
https://www.postgresql.org/message-id/flat/cf744a8ee4d47bdabe1da9174d4f3dc9@postgrespro.ru

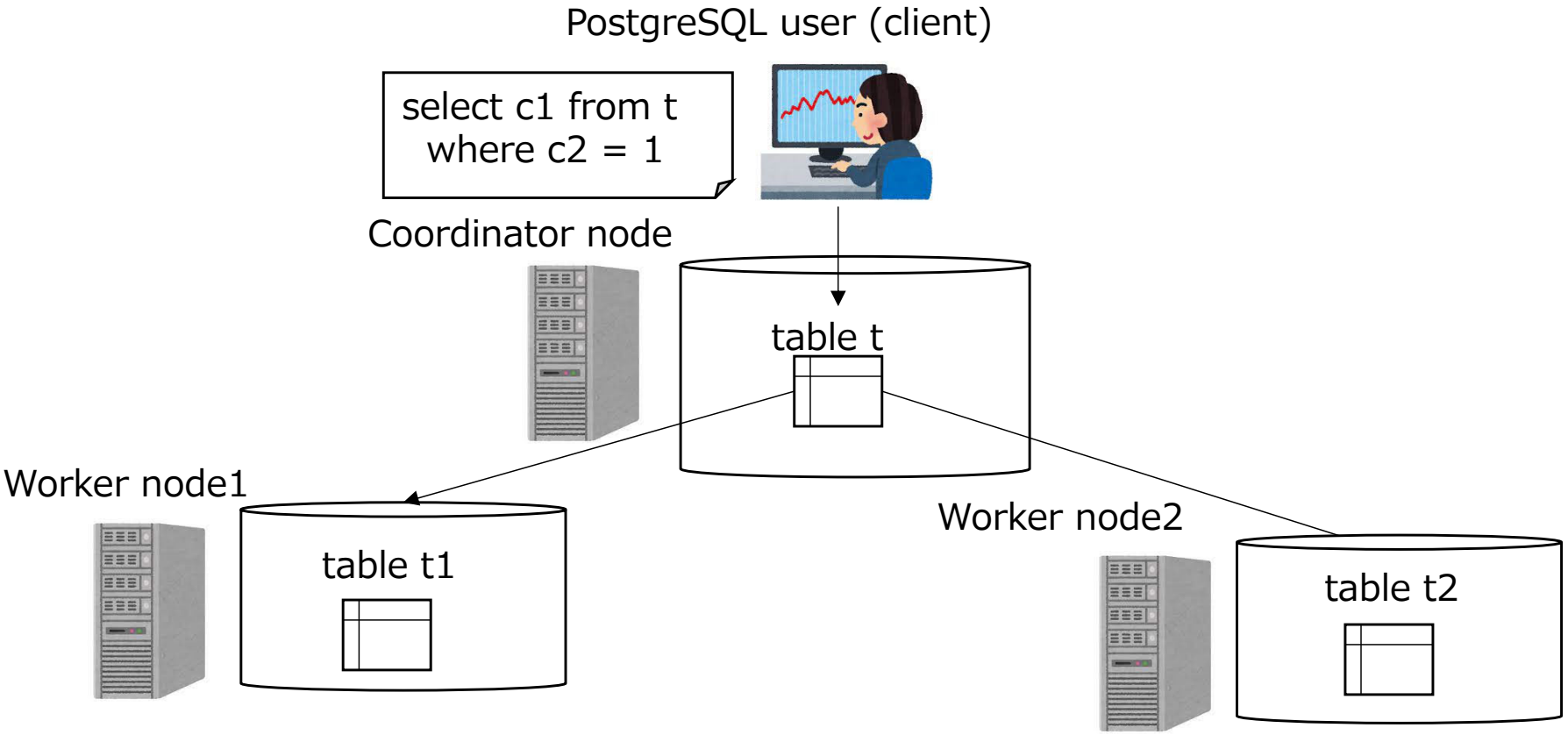
Table of contents

Built-in Sharding
1.1 Benefits

MITSUBISHI ELECTRIC
Changes for the Better

High-speed processing of large-scale data using multiple servers and PostgreSQL standard features★1.

- This feature allows for parallel reading and writing of data from one table across multiple physical servers transparently.
- The server cluster is composed of worker nodes that execute processes in parallel and the coordinator node that controls the workers.



★1 This means that we use only features in the PostgreSQL official repository.
This is a point of difference with other similar technologies, such as Citus

Built-in Sharding
1.2 Development Status

MITSUBISHI ELECTRIC
Changes for the Better

Parallel processing for aggregation is needed(see wiki★1),
but it is currently not supported.

A list of major PostgreSQL operations and their support status(PG16)

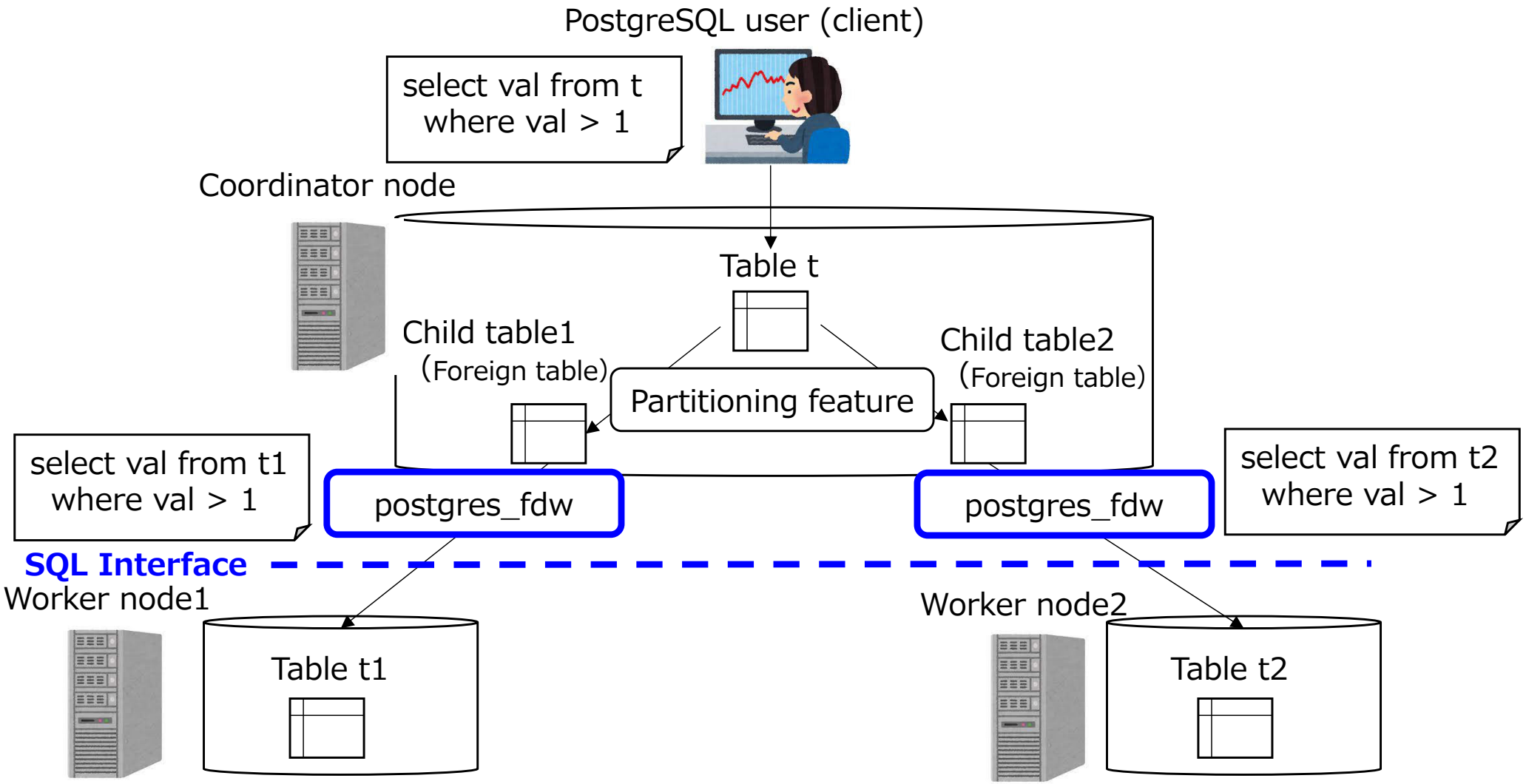
Operation	Parallel processing is supported?
Selection	Yes
Join	Yes
Sort	Yes
Aggregation	No
Subquery expression	No
Union or Intersection of sets	No

Focus

★1 PostgreSQL wiki, Built-in Sharding
https://wiki.postgresql.org/wiki/Built-in_Sharding

Built-in Sharding
1.3 Overall Mechanism

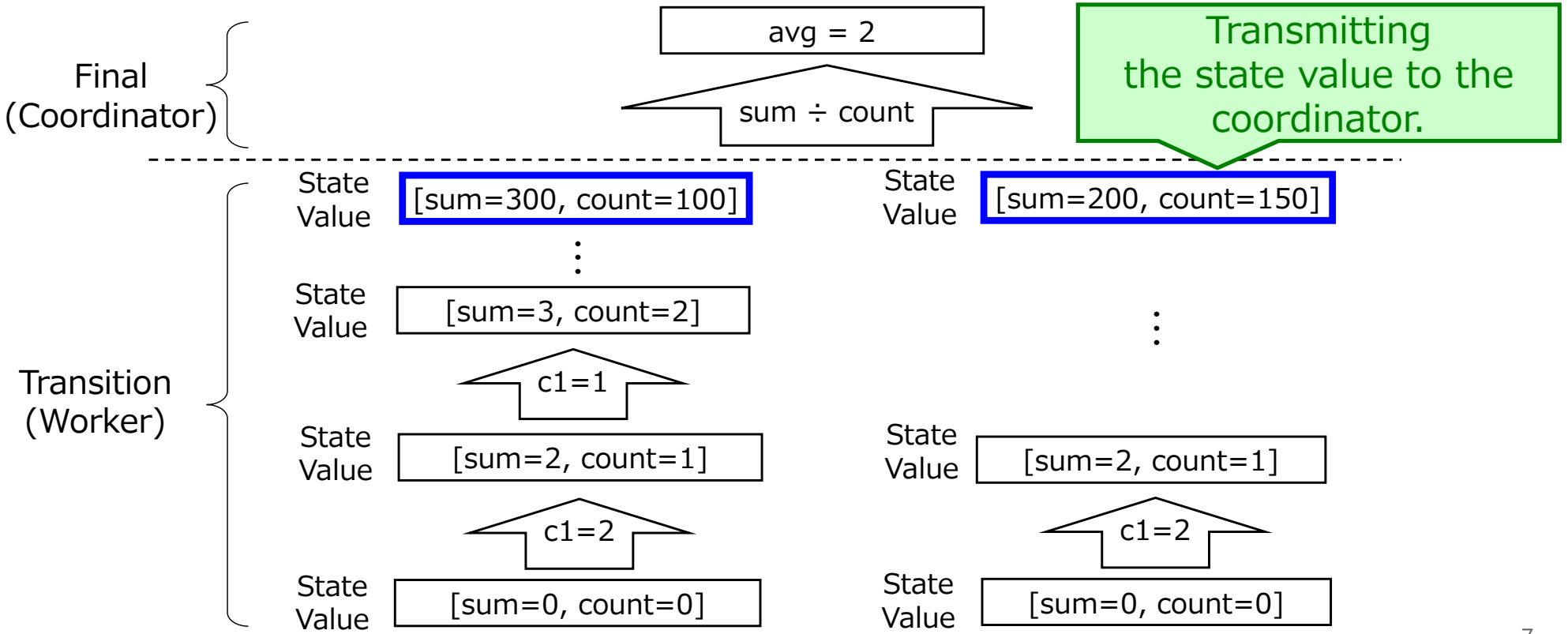
Partitioning the coordinator's table into multiple child tables, and linking them with the worker's table using SQL through postgres_fdw.



2. Parallel Execution for Aggregation
2.1 Overall Mechanism

Aggregating individually on the worker,
integrating them on the coordinator in the final process.

- In worker
  - The transition process is performed for each record, updates the state value, and transmits the state value to the coordinator.
- In coordinator
  - The final process generates return values from state values received from workers.

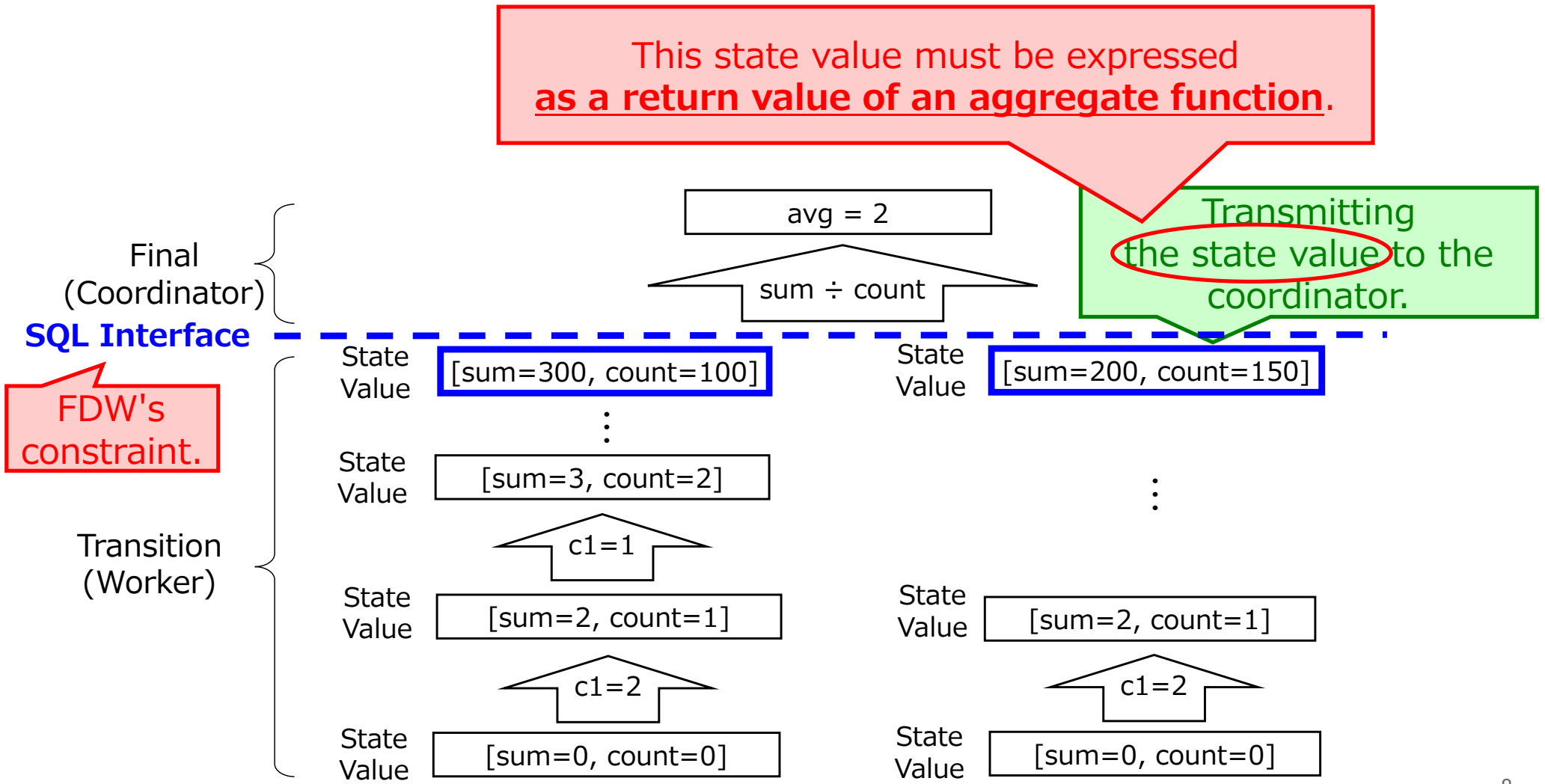


The flow of the average (avg) processing

2. Parallel Execution for Aggregation
2.2 Challenges in Built-in Sharding(1/2)

MITSUBISHI ELECTRIC
Changes for the Better

The state value generated by each worker
must be expressed as a return value of an aggregate function.



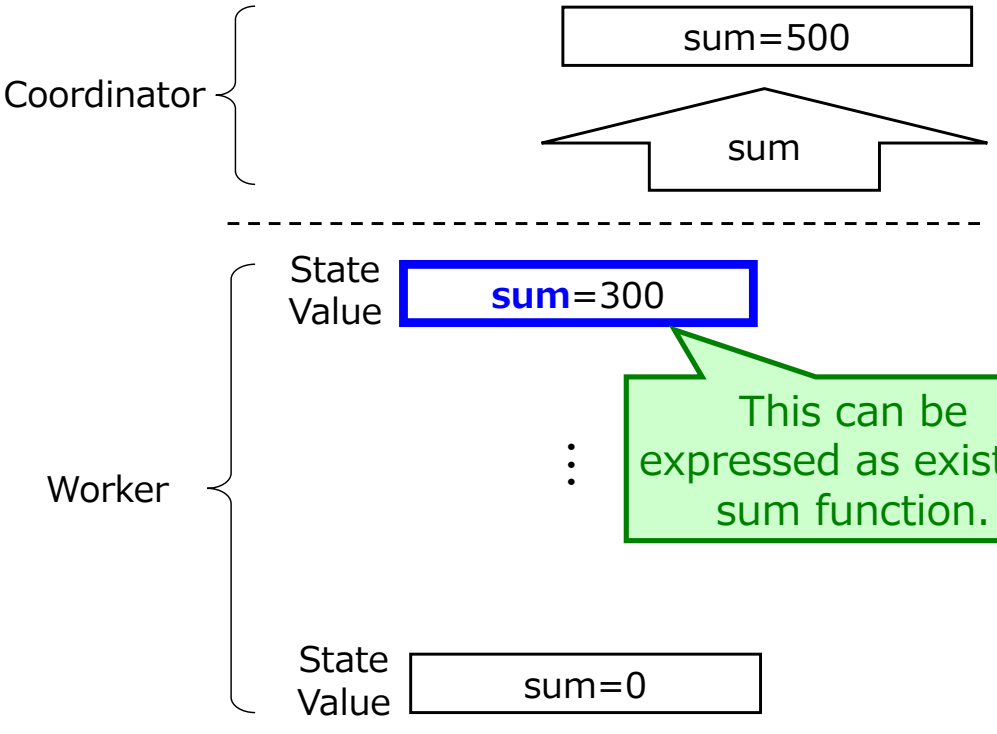
The flow of the average (avg) processing

2. Parallel Execution for Aggregation
2.2 Challenges in Built-in Sharding(2/2)

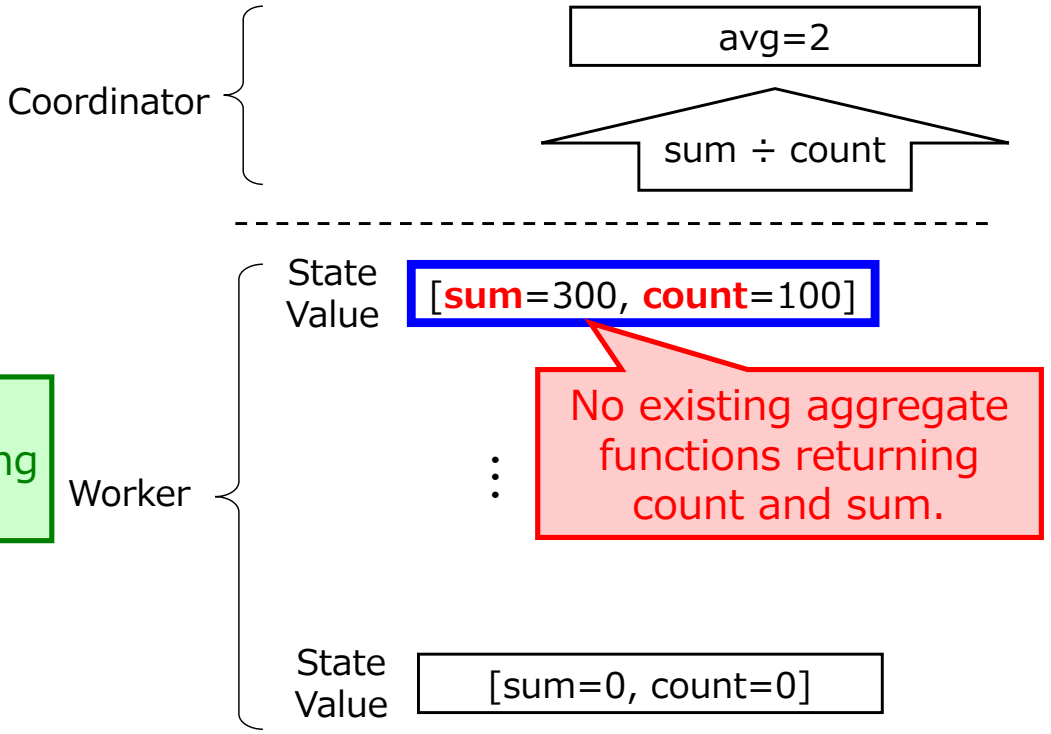
MITSUBISHI ELECTRIC
Changes for the Better

There is a bad case, in which a state value cannot be expressed as a return value of any aggregate function.

Good Case : Total (sum)

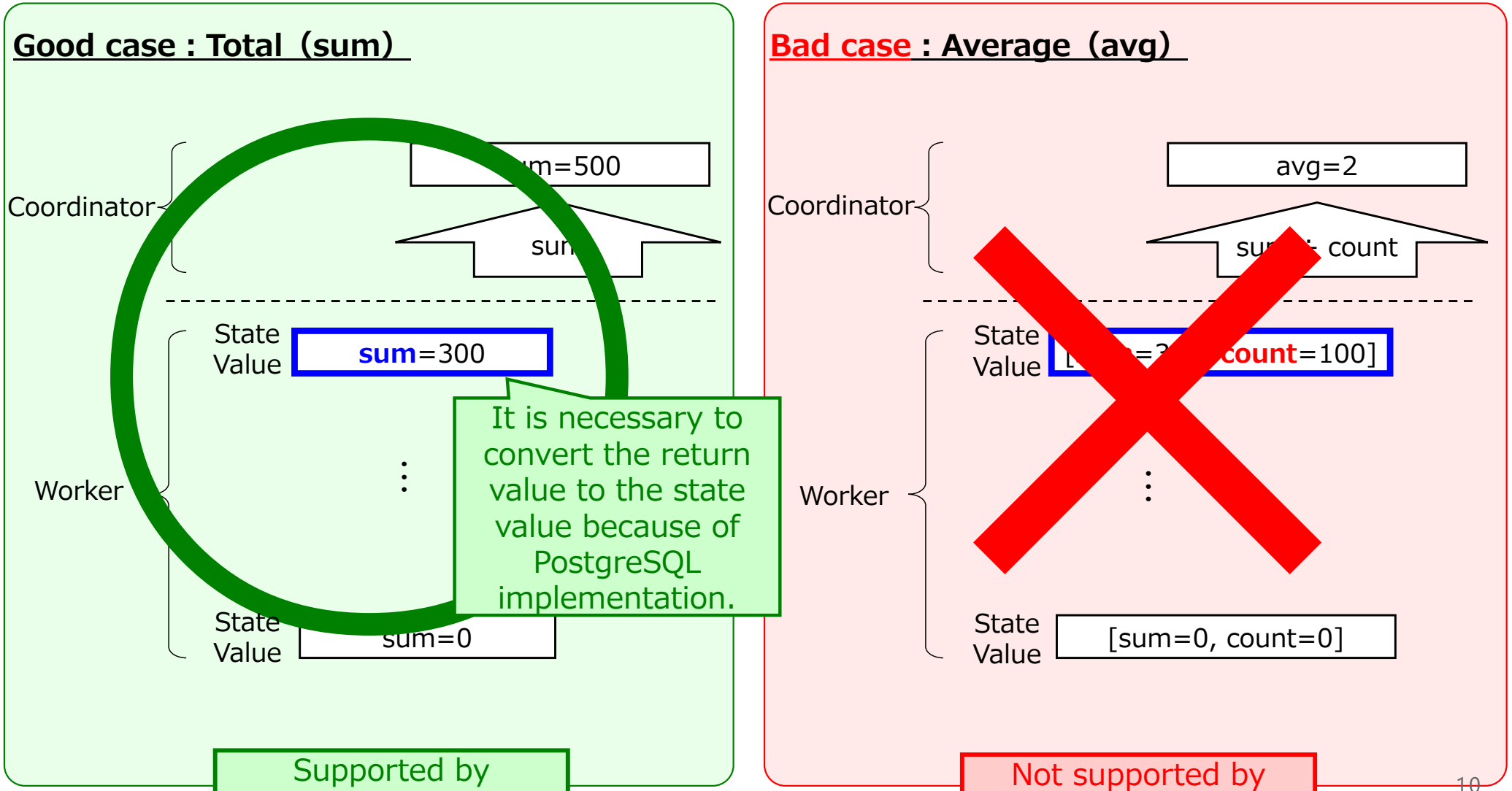


Bad case : Average (avg)



Existing Patch

Mr. Pyhalov posted a patch which supports only good cases.

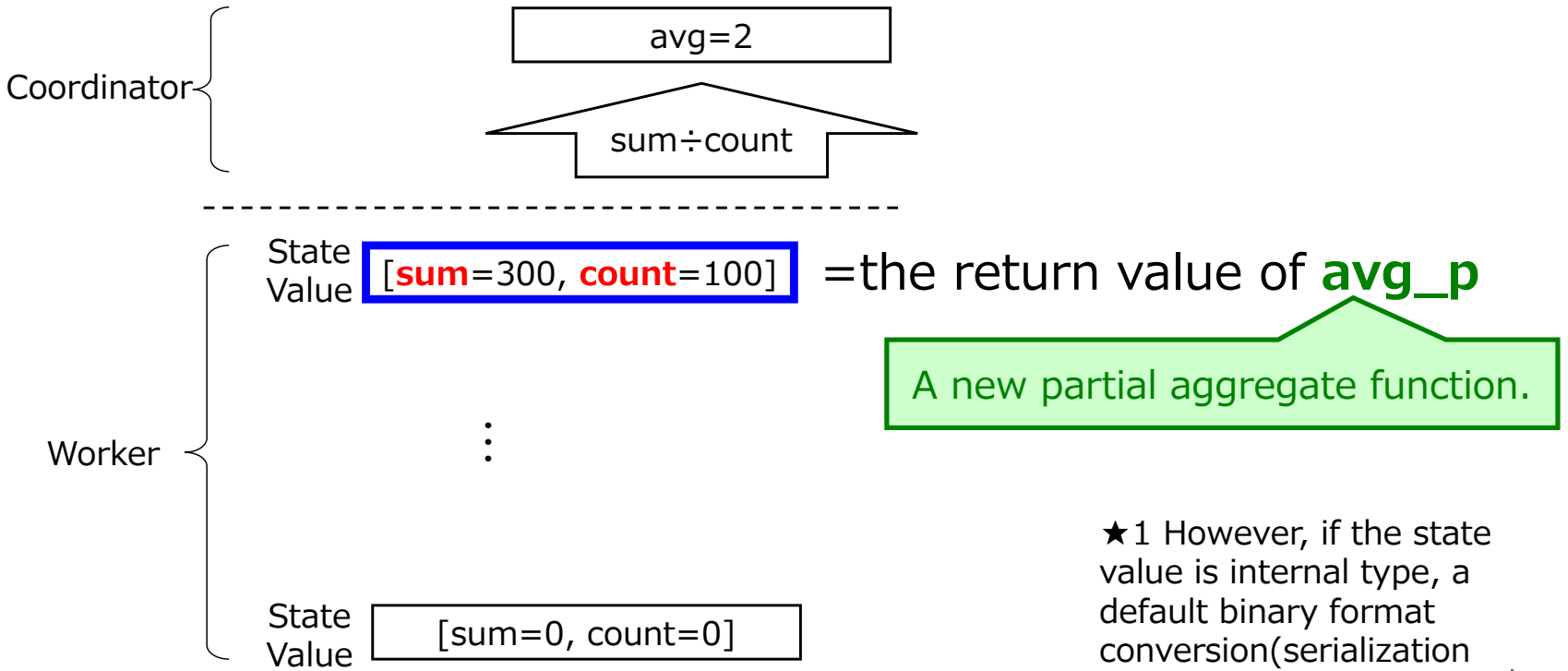


4. Proposal Patch
4.2 Approach

Defining new aggregate functions which return the state value (Partial aggregate function).

- Processing of partial aggregate function
  - Returning the state value after all transitions.
  - Having the same transition as the original aggregate function.
  - Having no final function★1.

Resolved case : Average (avg)



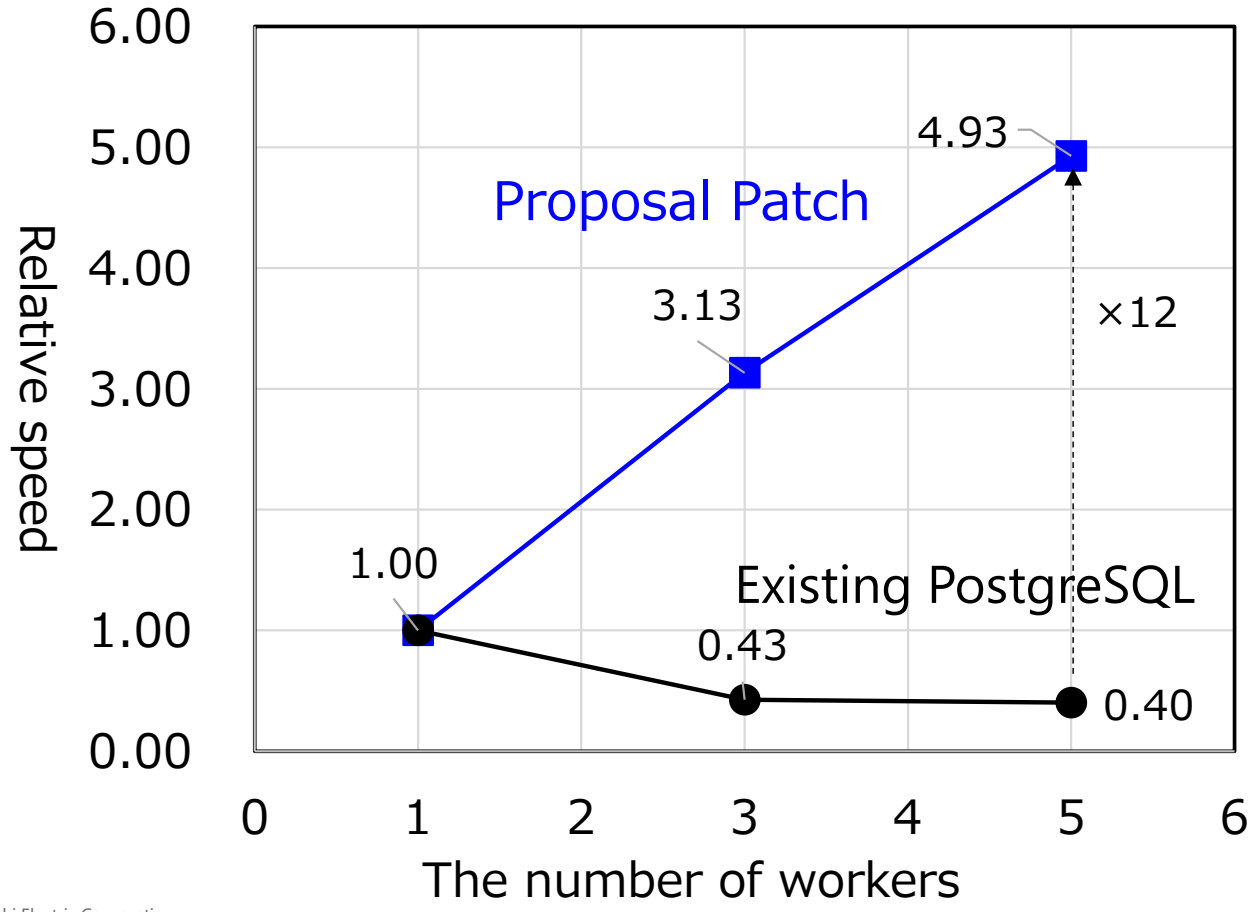
★1 However, if the state value is internal type, a default binary format conversion(serialization function) is necessary

4. Proposal Patch
4.2 Performance Evaluation

Aggregation speed increases proportionally to the number of workers.

■ Result★1

➢ With existing PostgreSQL, aggregation speed decrease when the number of workers is greater than 1. The reason for that is because the existing PostgreSQL requires transmitting all target data from the worker to the coordinator.



★1 Used a query calculating an average to one table(76GB) (TPC-H query1).

Refer to Supplement1 for the evaluation environment, Supplement2 for the PostgreSQL settings.

Reactions to Proposal Patch

5.1 The first comment

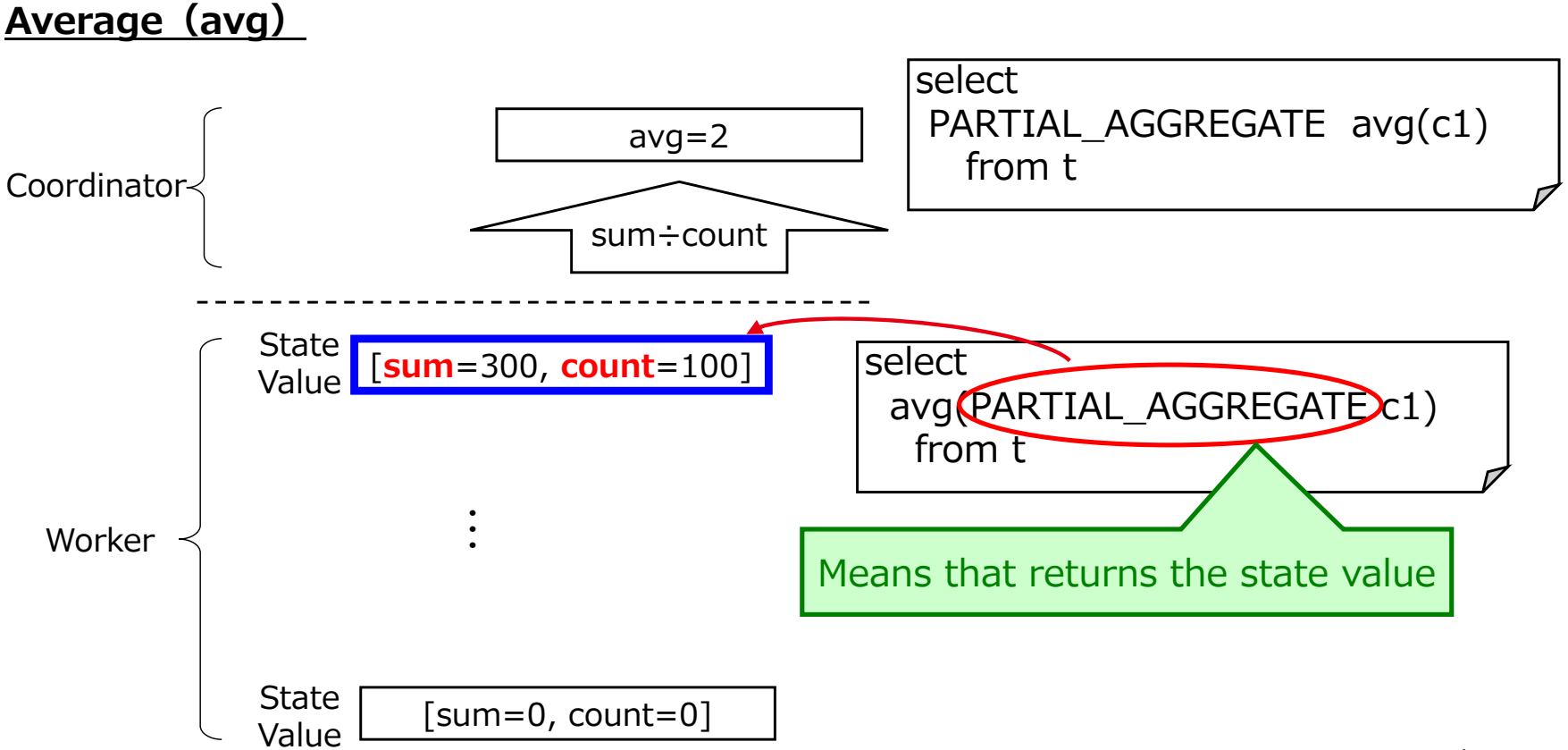
Adding a new partial aggregate function for each aggregate function is complicated. (by Mr. Haas of EnterpriseDB)

- Issue1: Big catalog size
  Theoretically, the size of the catalogs(pg_aggregate, pg_proc) will double.
- Issue2: Many additional codes
  Many codes are needed for managing partial aggregate functions.
- Issue3: Manual definition of partial aggregate functions
  There are many tasks for developers and the potential for mistakes.
  While the definition processes can be automated, it needs for additional codes.

Reactions to Proposal Patch
5.2 Solution to the first comment

By adding a new SQL keyword to the aggregate expression, partial aggregate functions have become unnecessary.

- I have prototyped the method proposed by Mr. Haas to add the following SQL keyword.
- The patch's size has been reduced to a quarter compared to the version I initially posted.

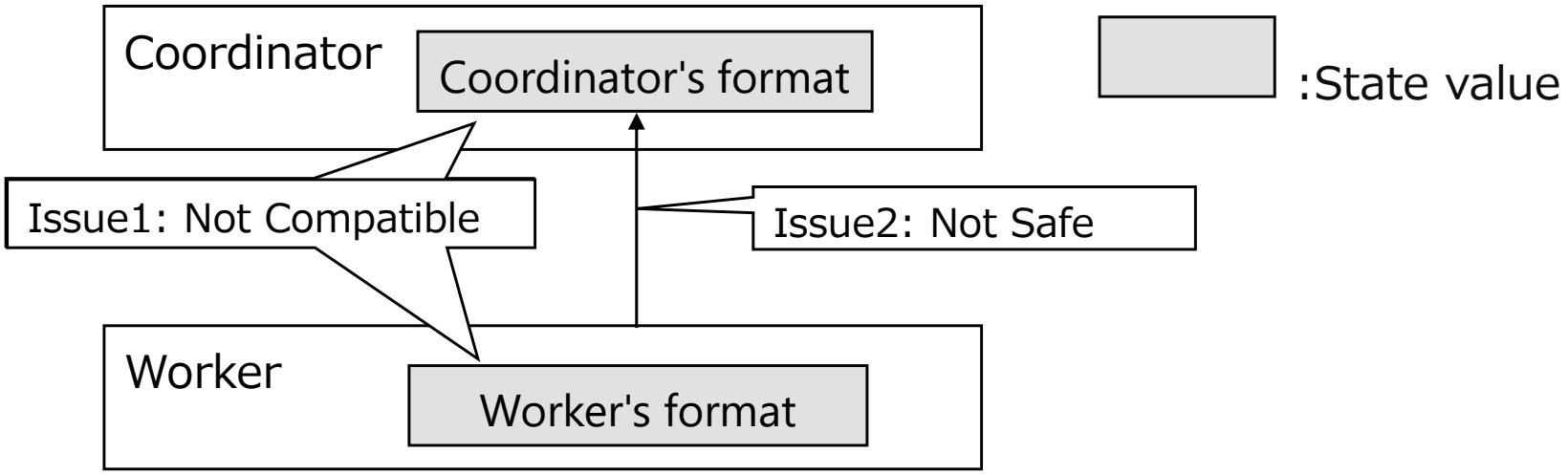


Reactions to Proposal Patch
5.3 The second comment

Ensure safety and compatibility for transmitting state value. (by Mr. Haas)

- Issue1: Compatibility
  - Cause1: The difference of PostgreSQL versions.
  - Cause2: The difference of server settings. Ex. Server encodings.
- Issue2: Safety
  There is a possibility for change of the state value during transmission due to security attacks or communication errors.
- Status
  I have considered solutions. Not yet implemented. **Need consensus**.

Issues relateted to transmitting state value

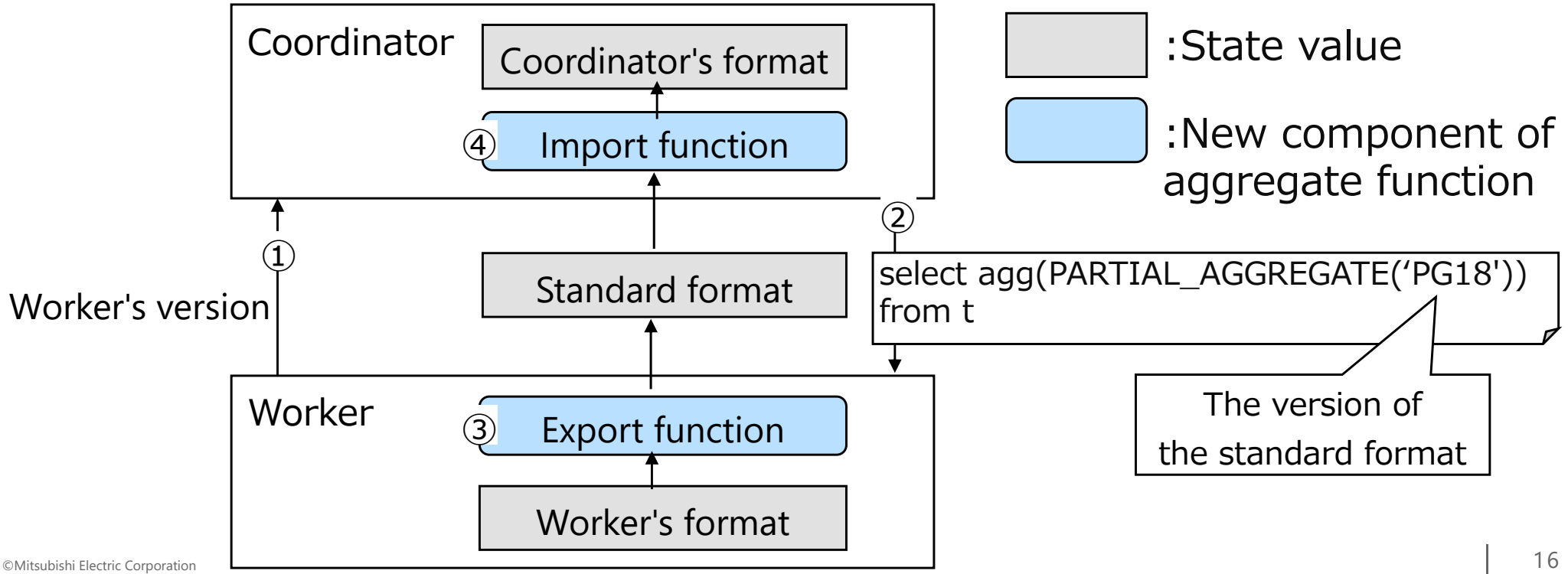


6 Transmitting state value
6.1 Proposal considered(1/2)

By adding export and import functions for each aggregate function,
I can resolve the differences between nodes.

- I define the standard format for transmitting state value,
  which is fixed for each version of PostgreSQL.
- The coordinator decides the standard format version based on the worker's version.
  (next slide for details)

Overall System of my approach

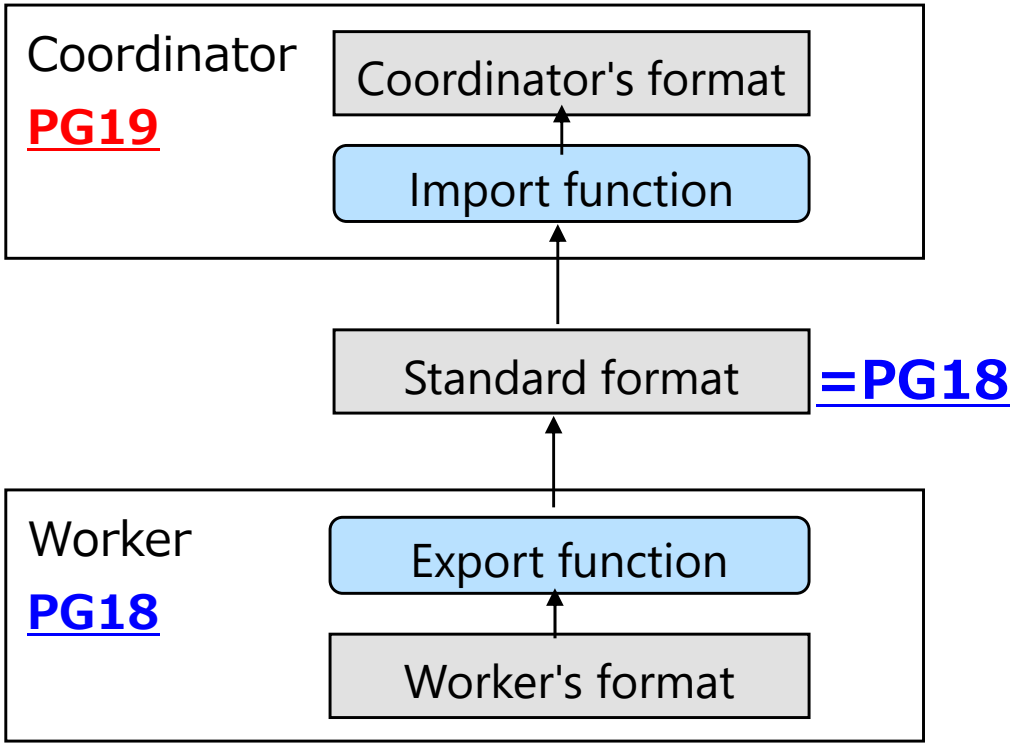


Transmitting state value
6.1 Proposal considered(2/2)

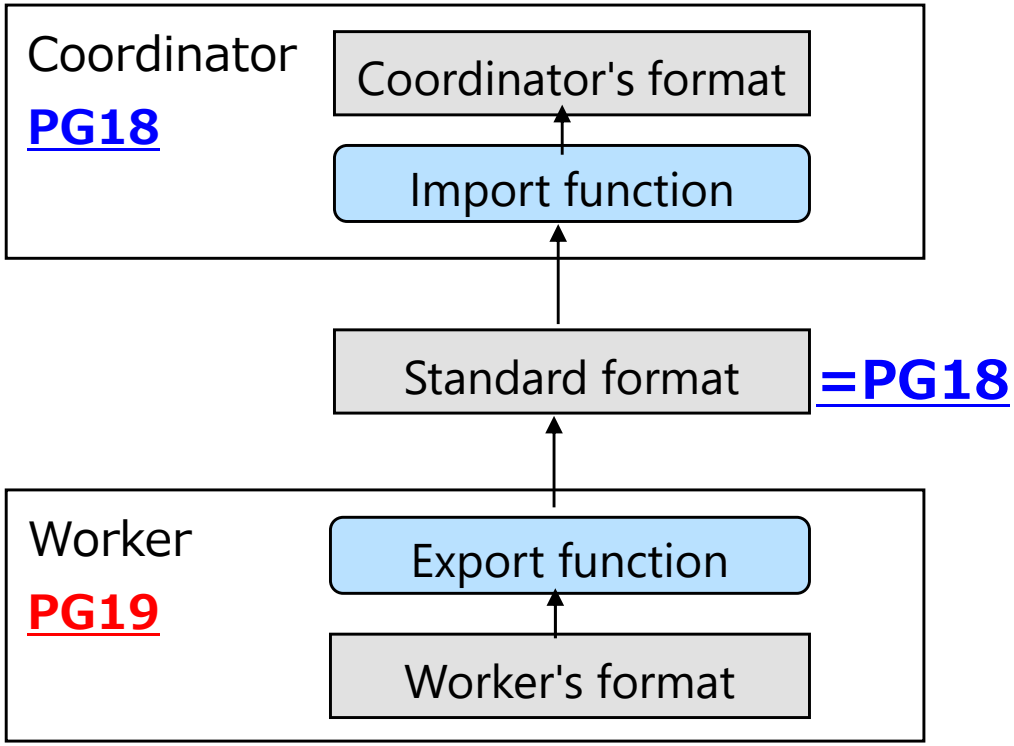
We need to make sure the compatibility of import/export functions, when the coordinator's version is different from the worker's.

- The standard format version is the minimum of the coordinator's and worker's versions.

Coordinator's version > Worker's version



Coordinator's version <= Worker's version



MITSUBISHI ELECTRIC
Changes for the Better

Of the 131 built-in aggregate functions,
it is necessary to add export or import functions for 72 of them.

Necessity of export/import for each built-in aggregate function★1

Necessary?	Arg's type	Example	Count
No	not pseudo	min, max	59
Yes (not hard)	not pseudo	avg, count	67
Yes (hard)	pseudo	array_agg, any_value, range_ intersect_agg	5
Total			131

This type needs to accept many actual data types, including user-defined types

★1 The target PostgreSQL version is 16

- Import functions have to check validation, in addition to convert the state value.

The types of validation check for import

No.	Explanation
1	Checking the number of data items of the state value. Ex. Number of data items of avg's state value must be 2(count and sum).
2	Checking the ranges for each data item of the state value. Ex. The number of records must be nonnegative for avg.

7 Points for discussion

MITSUBISHI ELECTRIC
Changes for the Better

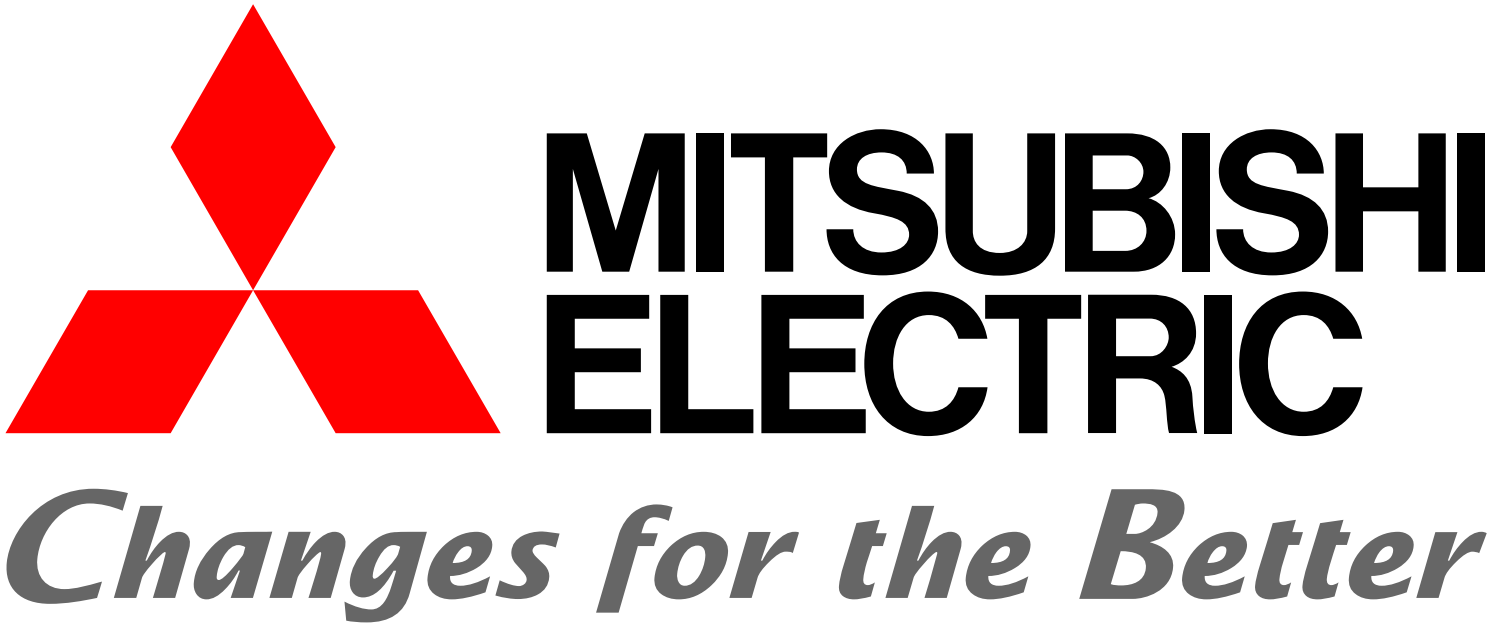
- Transmitting state value
  - ✓ Could you accept my proposal?
  - ✓ I think it would be difficult to completely implement my proposal at once. Would it be possible to commit a patch with the following constraints first?
    - The server versions of the coordinator and the worker match.
    - Supported built-in aggregate functions are a few subset (Ex. avg, sum, count, min, max).
- Aggregation in worker
  - ✓ The SQL keyword to be added is non-standard, but are they acceptable?

8 Acknowledgement

MITSUBISHI ELECTRIC
Changes for the Better

- I would like to express my sincere gratitude to the following developers for their extremely valuable comments on the basic parts of the patch.
  - Mr. Pyhalov (Postgres Professional)
  - Mr. Haas (EnterpriseDB)
  - Mr. Momjian (EnterpriseDB)

  Among others.


MITSUBISHI
ELECTRIC
Changes for the Better

[Supplement1] H/W, OS Setting

- Using AWS EC2
- The Coordinator and the workers are in the same subnet
- Settings of coordinator and workers are the following table

Item	Explanation
PostgreSQL	16Dev※1
Our patch	v17(The latest version at 2022/12/15)
OS	Amazon Linux release2(Kernel 5.10)
EC2 instance type	m6in.xlarge
The number of vCPUs	4
DRAM	16GB
EBS Type	gp2
Storage	SSD, 1TB

※1 コミットIDは8b6b043ceef29a0a7a462b748da398511832efcf(2022/12/15時点の最新版)

[Supplement2] PostgreSQL Setting

MITSUBISHI ELECTRIC
Changes for the Better

- Table
  One child table per one EC2 instance
- PostgreSQL configuration parameters

Item	Value
shared_buffers	4096MB
work_mem	1024MB
max_parallel_workers_per_gather	4
enable_partitionwise_aggregate	on

- postgres_fdw configuration parameters

Item	Value
async_capable	true
server_version	160000