ペタバイトデータをSQLでリアルタイム分析し、品質を向上させたユーザ事例

～現場の苦労を紹介～

TOSHIBA

東芝デジタルソリューションズ株式会社
ソフトウェアシステム技術開発センター
ソフトウェア開発部　　新名　博
2024/07/12

アジェンダ

01 工場IoT 品質管理システム

02 GridDBとは?

03 GridDBの仕組み

04 ユーザ事例　～現場の苦労 を紹介～

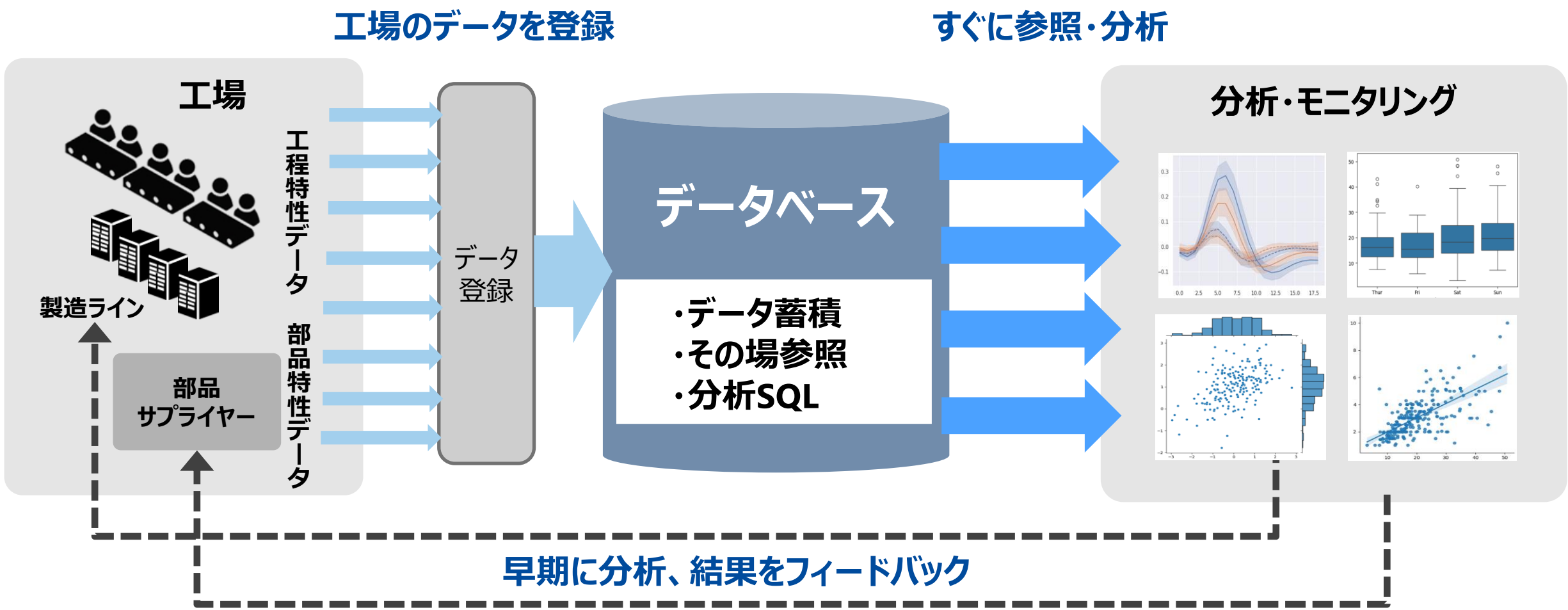
05 GridDBを利用したい

01

工場IoT 品質管理システム

導入事例　工場IoT 品質管理システム

工程データの蓄積・分析の連携をスムーズに。分析結果を早期に製造にフィードバック



導入効果　工場IoT 品質管理システム

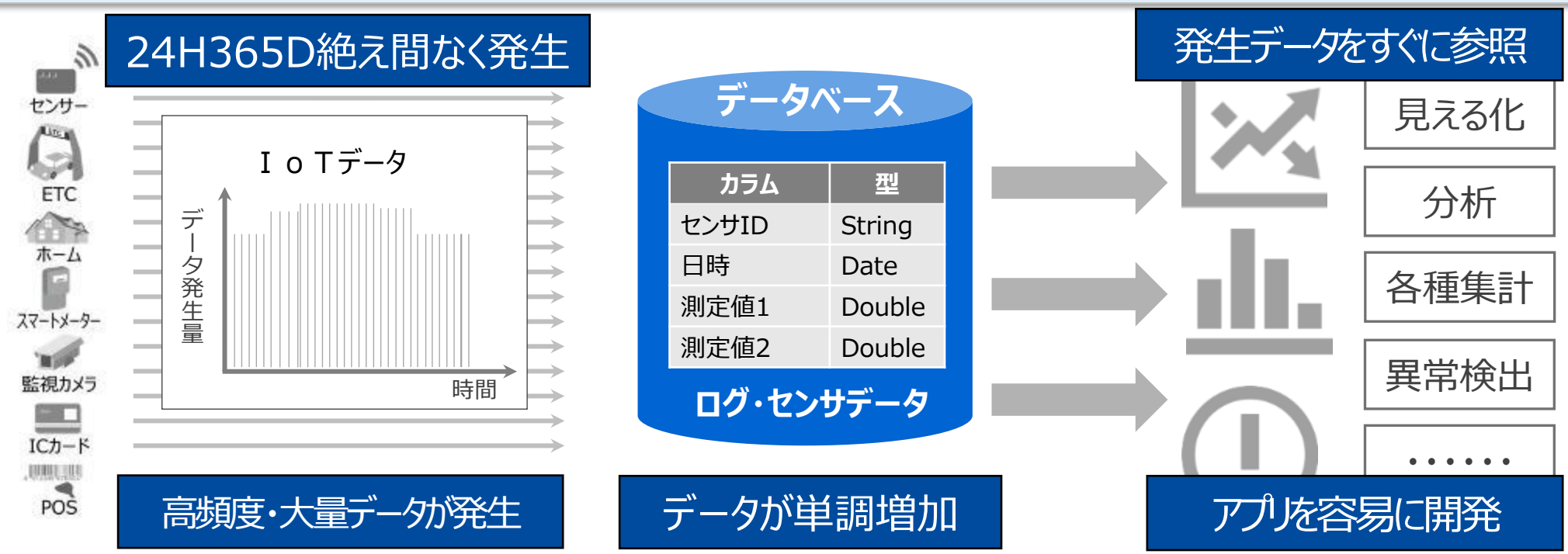
概要
- 品質管理システムで高性能DB専用機を使用 → GridDBへ切り替え

システムの課題
- 製造レコードを全件貯めることを目指しており、DB専用機では莫大なコストがかかる
  - ✓ データ蓄積量：**数ペタバイト**
  - ✓ 登録データ量：**数百GB/日**
  - ✓ 分析用SQLによるアクセス頻度：**数万回/日**

成果
- 高性能DB専用機以上の性能を標準的なサーバで実現し、**大幅なコストダウン**
- システム規模が大きくなったとき、安価な**スケールアウト(サーバを追加)**で対応した
  - サーバ4台→サーバ8台

IoTシステムのデータ管理に求められること

リアルタイムに分析して、素早く業務改善につなげたい



求められること	要件	NoSQLデータベース	RDB	GridDB
24H365D絶え間なく発生	高可用性	◎：クラスタによる冗長構成	◎：外付けで冗長構成	◎：クラスタによる冗長構成
高頻度で大量データが発生	低レイテンシ、高スループット	○：単純な操作が高速	○：高性能なHWが必要	◎：インメモリ指向で高速
大量データが単調増加	高スケーラビリティ	◎：クラスタでスケールアウト	△：スケールアップが困難	◎：クラスタでスケールアウト
発生後すぐにリアルタイム参照	一元管理で高速検索	△：ピンポイントの参照は高速	○：高性能なHWが必要	◎：分散検索で高速
アプリを容易に開発	慣れ親しんだ開発環境	×：固有の専用API	◎：SQL使用	◎：SQL使用可能

02

GridDBとは?

GridDBについて

GridDBは、東芝デジタルソリューションズが開発・販売する
NoSQL型のDBMSです。

ビッグデータやIoTシステム向けに特化して作られたDBMSです。

GridDBの特徴

時系列データ指向

高頻度で大規模な時系列データを効率よくリアルタイム処理する時系列データ指向

ペタバイト級の高い処理能力

ペタバイト規模のデータを扱うためにさまざまな工夫を組み込み、高い処理能力を実現

高い信頼性と柔軟な拡張性

障害の発生時やサーバ増設においてもノンストップ運用を実現する高い信頼性と柔軟な拡張性

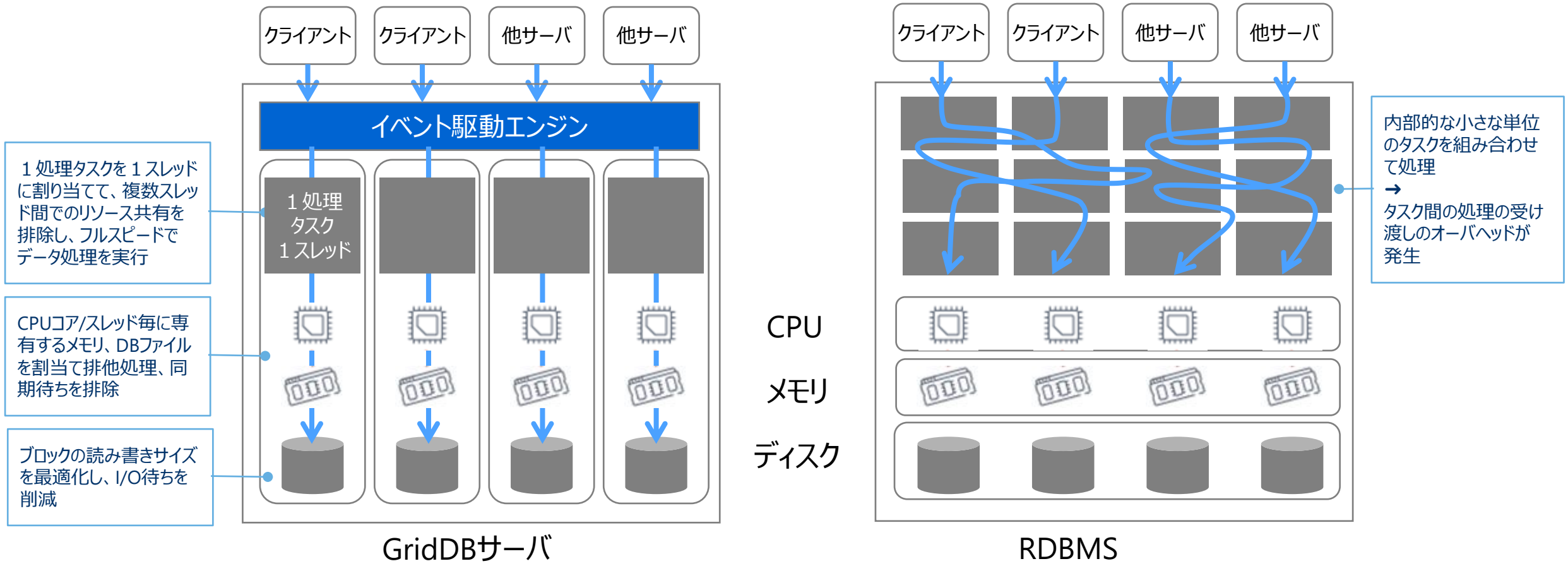
開発の容易性

NoSQLインターフェースだけではなく、SQLインターフェースを用意し、開発の俊敏性と使いやすさを実現

NoSQLの高速性とRDBの開発容易性を備えたクラスタ型データベース

ペタバイト級の高い処理能力

排他を排除することでCPUをフル活用



開発の俊敏性と使いやすさ

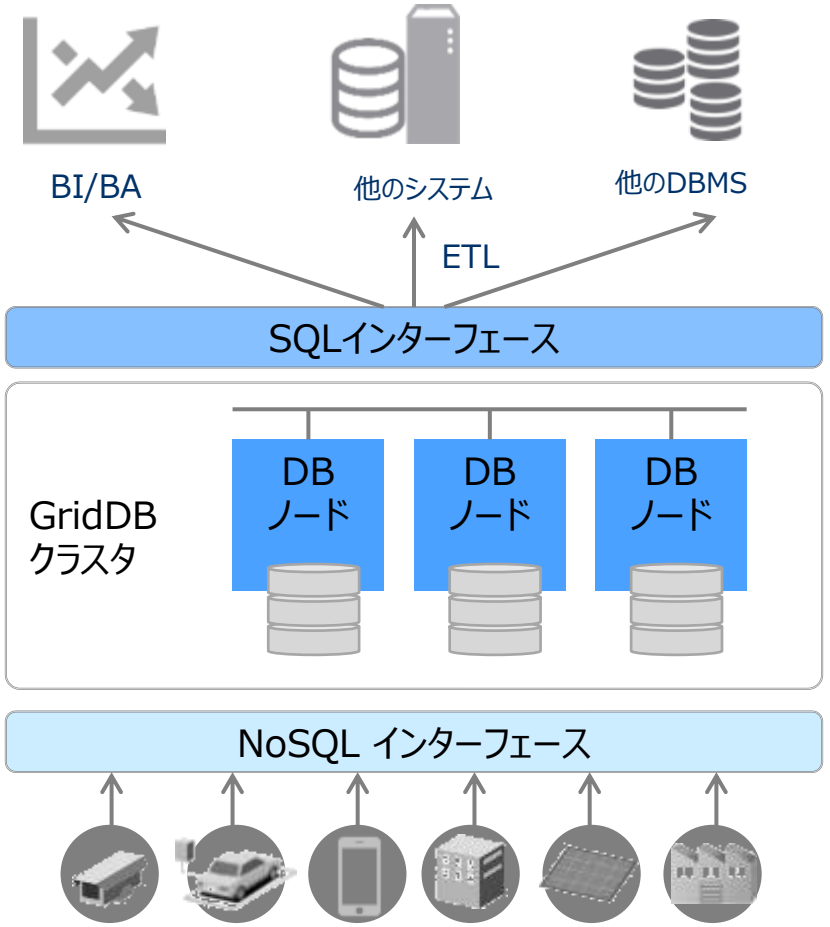
NoSQLとSQLのデュアルインターフェースを提供

NoSQL インターフェース

- 高速・高スループットな登録・検索・更新が可能
- Java / C / Python / Go / Node.js クライアント

SQL インターフェース

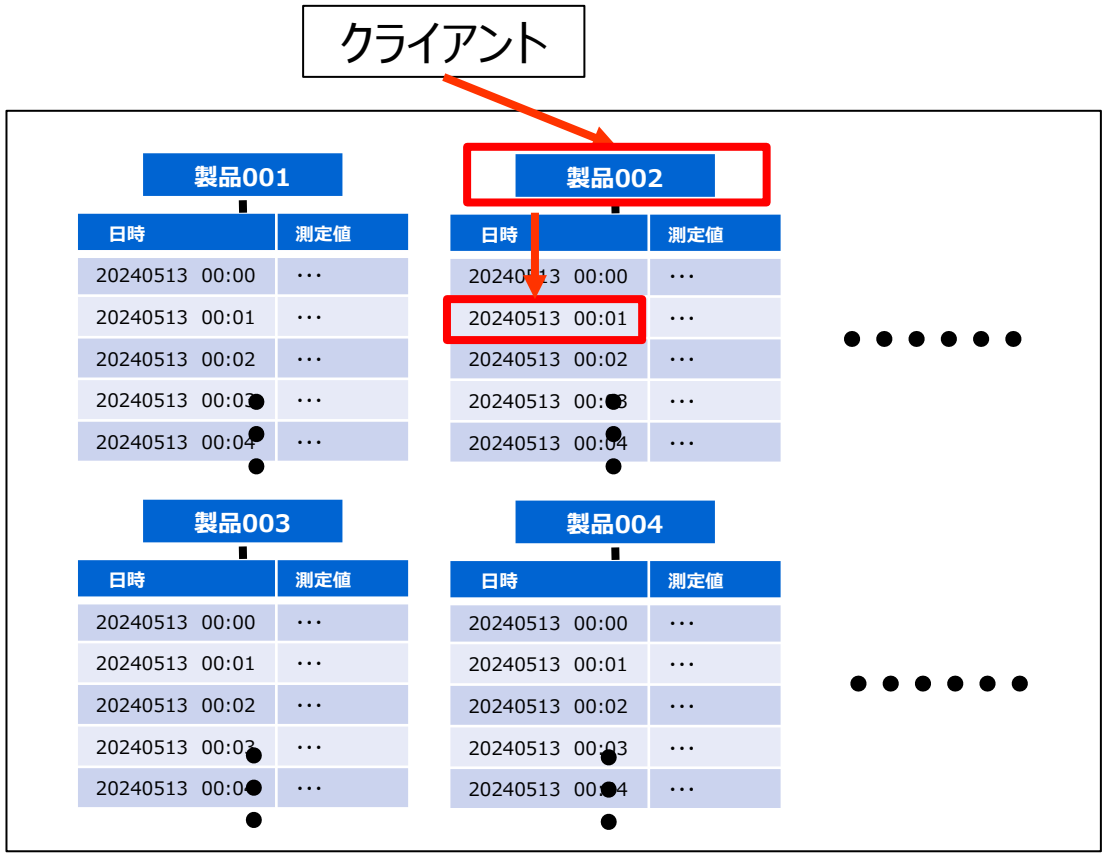
- 複雑な検索が可能
- 標準化されたSQLなので、他ソフトウェアとの連携が容易
- JDBC / ODBCドライバ



NoSQLとSQL(データモデル)

- NoSQL：コンテナ　機器などの管理単位毎に表現し、コンテナを絞込み
- SQL：　テーブル　全情報をひとまとめに表現し、レコードを絞込み

コンテナ表現



テーブル表現

03

GridDBの仕組み

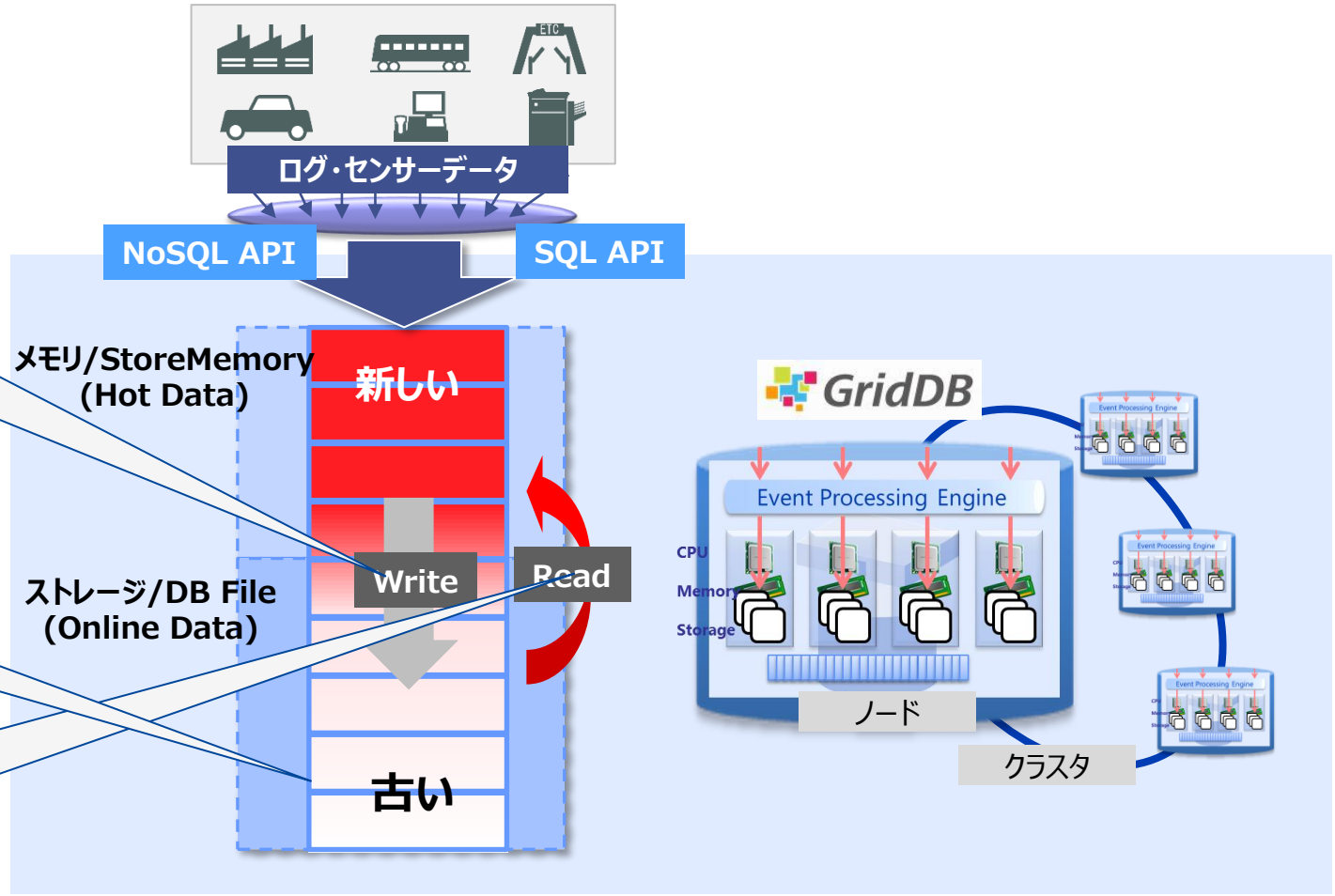
GridDB処理イメージ

時々刻々と増加するデータをインメモリで受け止め、ストレージに永続化

新しいデータは登録直後はメモリ上にある。データの永続化のため、ログとデータはストレージにも書き込まれる。

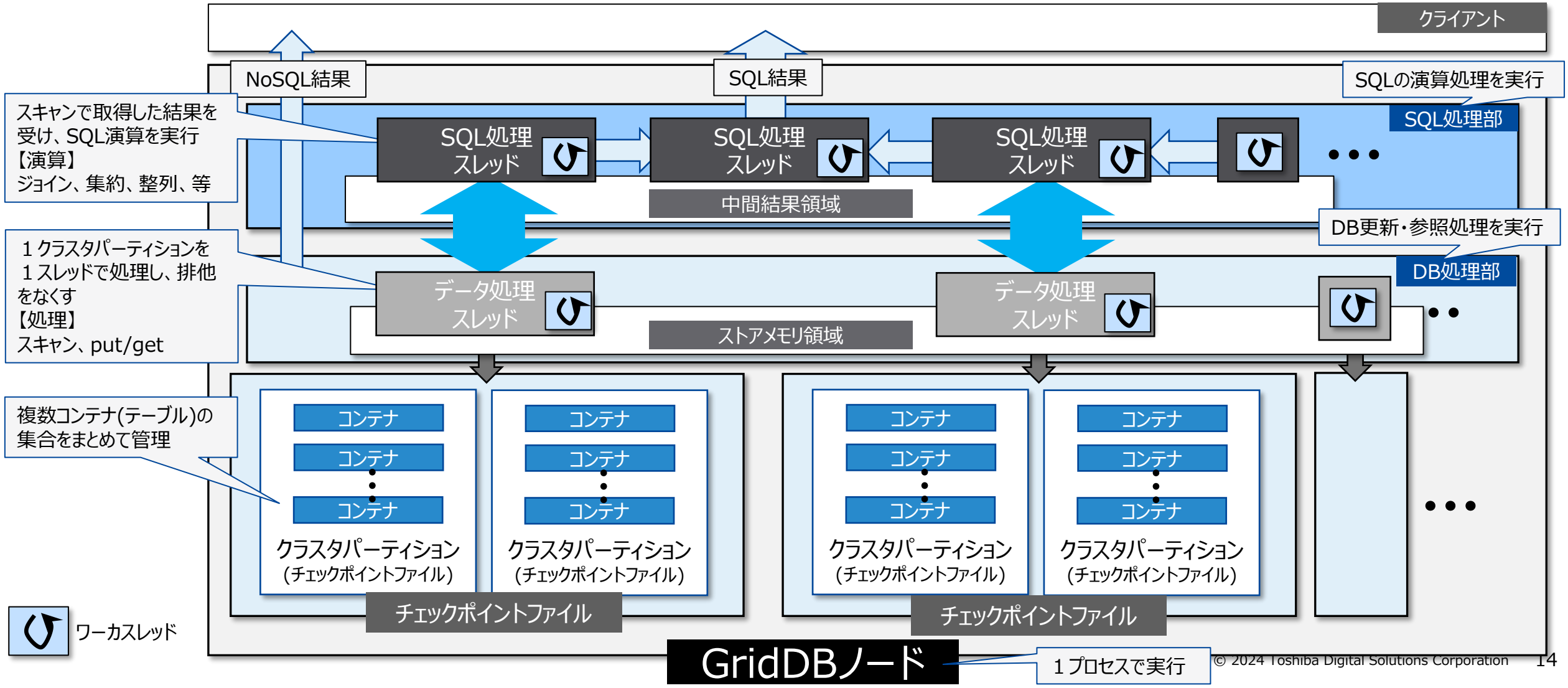
長期間参照されない古いデータはストレージにのみ残る状態になる。

古いデータはストレージに追い出されるが、アプリからの要求があった場合には、ストレージからのデータ読込みも行う。



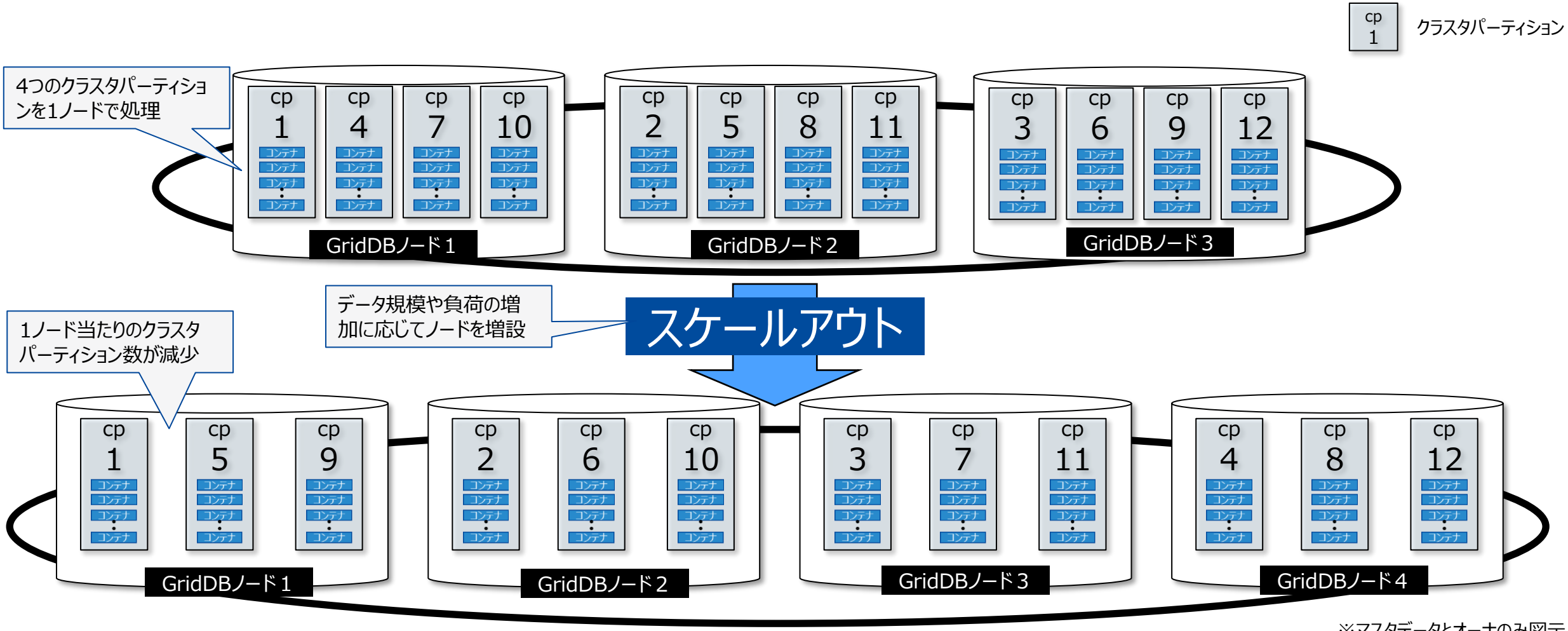
GridDBの動き ／ NoSQL と SQL

NoSQL層のコンテナデータ処理、SQL層の演算処理が並列に動作



クラスタ内のデータ配置 ／ クラスタパーティションとクラスタ

クラスタパーティション(cp)をノードに分散配置してスケールアウト



※マスタデータとオーナのみ図示

主なデータベースチューニング項目

データのサイジング、性能要件を加味して、計算機リソースに合わせて設定

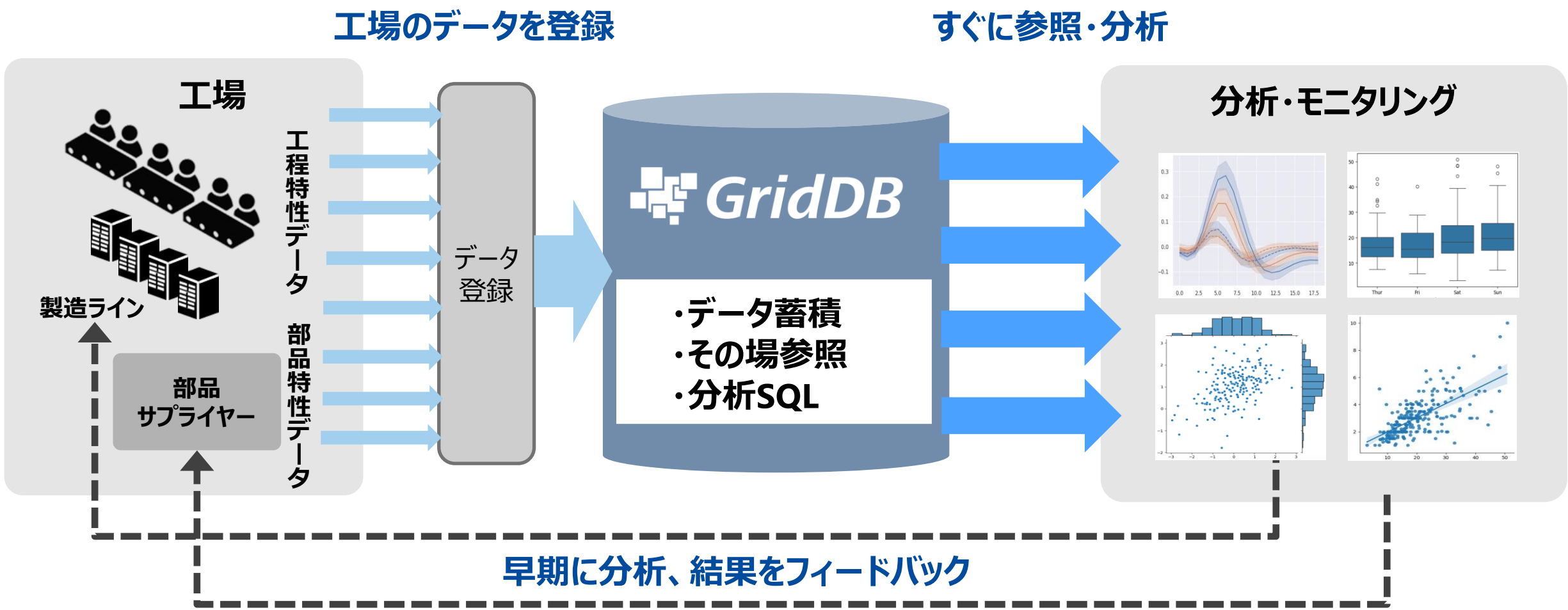
分類	項目	パラメータ	既定値	意味
データストア	クラスタパーティション数	/dataStore/partitionNum	128	・データを管理するクラスタパーティションの数 ・スケールアウトを考慮し、コア数とノード台数の公倍数が目安
	ブロックサイズ	/dataStore/storeBlockSize	64KB	・データベースのブロックサイズ ・I/Oのバランス(索引と大規模テーブルスキャン)で調整
	ストアメモリサイズ	/dataStore/storeMemoryLimit	1024MB	・データ管理用メモリ領域(≒バッファ)の上限サイズ ・高頻度にアクセスするホットデータ領域をサイジングして調整
	並列度	/dataStore/concurrency	4	・ノード全体でのコンテナ・テーブルスキャンの並列実行数 ・稼働させる計算機のコア数を目安に設定
	ブロック圧縮	/dataStore/storeCompressionMode	NO_COMPRESSION	・データブロックの圧縮有無や圧縮アルゴリズムの設定 ・性能要件(DBサイズや応答特性)に合わせて設定
チェックポイント	実行周期	/checkpoint/checkpointInterval	60s	・チェックポイント(メモリ上のデータブロック永続化)実行周期 ・性能要件(応答特性やリカバリ時間)に合わせて設定
SQL	SQLストアメモリ	/sql/storeMemoryLimit	1024MB	・SQL処理で使用する中間データのメモリ領域の上限サイズ ・SQLの中間結果をサイジングして調整
	並列度	/sql/concurrency	4	・ノード全体でのSQLの演算(JOINなど)の並列実行数 ・稼働させる計算機のコア数を目安に設定
クラスタ	クラスタパーティション配置規則	/cluster/goalAssignmentRule	DEFAULT	・クラスタパーティション配置表の割当規則 ・可用性や性能を加味して配置規則を設定

04

ユーザ事例　～現場の苦労を紹介～

導入事例　工場IoT 品質管理システム

工程データの蓄積・分析の連携をスムーズに。分析結果を早期に製造にフィードバック



ペタバイトサイズのデータ規模感

多数の大規模テーブルへの高スループット登録、と、複雑SQLの両立が必要

システムのデータ規模の例	
データ蓄積期間	5年
製品あたりの特性値数	50万
年間製品生産数	5000万

取りこぼし無く登録

工場のデータを登録

すぐに参照・分析

大規模テーブルに対して、多様な検索

工場

登録：10,000件/秒

検索：10,000回/日

析・モニタリング

工程特性データ

部品特性デ

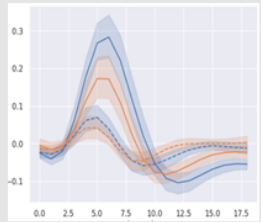
部品
サプライヤー

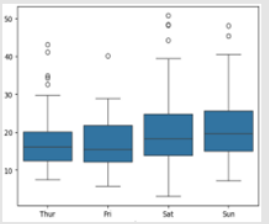
データ
登録

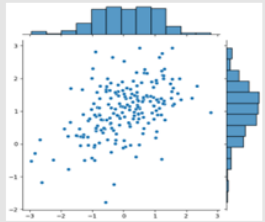
GridDB

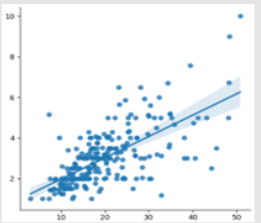
テーブル数：5000

レコード数：1.5兆件







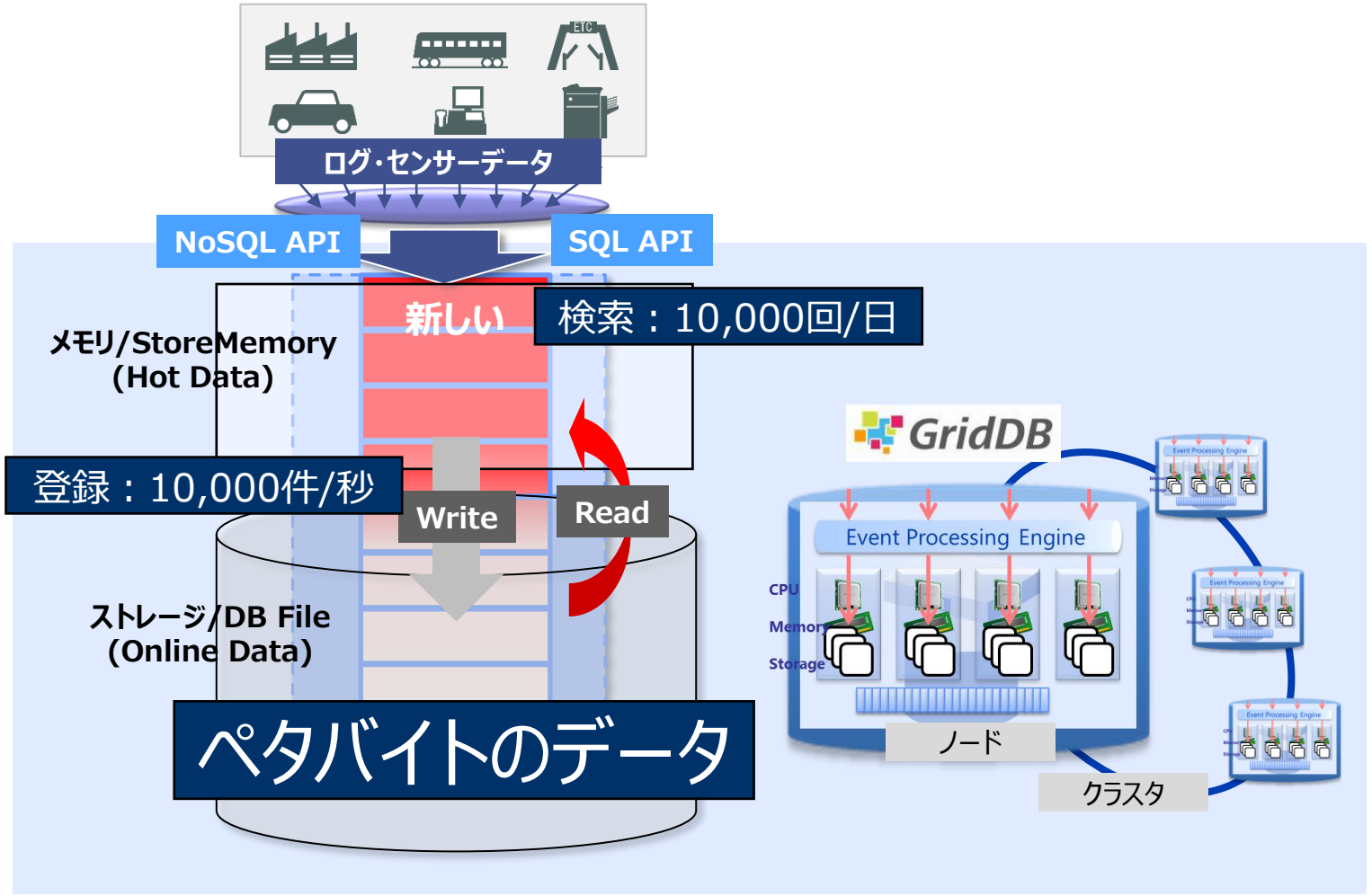


高負荷環境での
多数の同時実行

現場の苦労　理想と現実

【理想的な状態】
・直近のデータはインメモリに。
・古いデータは低頻度の参照。
　→ ホットデータを中心に高速処理

【現実の大規模データ管理の課題】
① 直近データだけでもリソースが多く必要
　　→**～20TB/1カ月を参照**
② 古いデータも頻繁・大量に参照
　　→**前年データとの比較が頻繁**
③ データ件数が大量でも高速に処理
　　→**250億件/1カ月の演算処理**
④ 障害への備えも必要
　　→**障害があっても応答性能を維持**



インメモリ効果、並列・分散効果を最大限発揮しないと要件を達成できない

時々刻々と蓄積される大規模IoTデータ管理の課題への対応

GridDBの基本機能とチューニングにより課題を解消

基本的なIoTシステムの要件

	時間方向のデータアクセス	ペタバイト規模のデータ管理	耐障害性	開発容易	①大量リソースが必要	②頻繁に過去データ参照	③大量レコード処理	④障害時の性能安定
時系列データ指向	●							
スケールアウト		●						
クラスタ機能(冗長化)			●					
SQLインターフェース				●				
1．パーティショニングテーブル					●	●		
2．SQLチューニング(区間指定)					●	●		
3．SQLチューニング(並列処理数ヒント)							●	
4．占有ブロック設定					●	●		
5．パーティション配置指定							●	●

ペタバイト規模での課題への対処
➔ 今回お話しする範囲

インメモリ効果、分散・並列効果をフル活用

1．テーブル定義：パーティショニングテーブル

大きなテーブルのデータは、インターバルパーティショニングで分割して管理

インターバル
パーティショニングキー

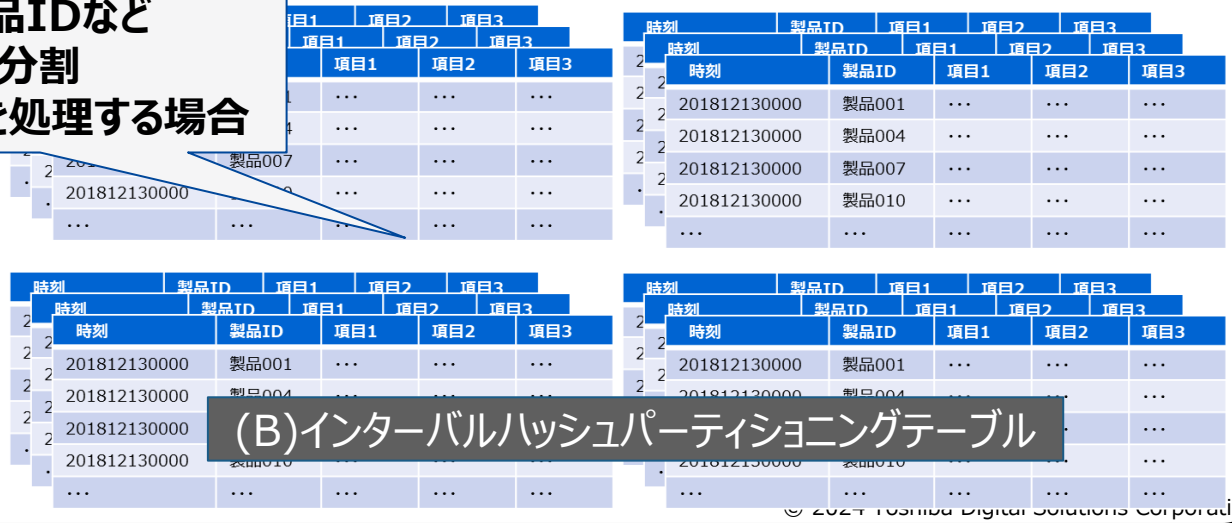
IoTデータのテーブル

日時	製品ID	測定値1	測定値2	測定値3
20240513 00:00	製品001	…	…	…
20240513 00:00	製品002	…	…	…
20240513 00:00	製品003	…	…	…
20240513 00:00	製品004	…	…	…
20240513 00:00	製品005	…	…	…
20240513 00:01	製品001	…	…	…
20240513 00:01	製品002	…	…	…
20240513 00:01	製品003	…	…	…
20240513 00:01	製品004	…	…	…
20240513 00:01	製品005	…	…	…
20240513 00:02	製品001	…	…	…
20240513 00:02	製品002	…	…	…
20240513 00:02	製品003	…	…	…
20240513 00:02	製品004	…	…	…
20240513 00:02	製品005	…	…	…
20240513 00:03	製品001	…	…	…
20240513 00:03	製品002	…	…	…
…	…	…	…	…

ハッシュ
パーティショニングキー

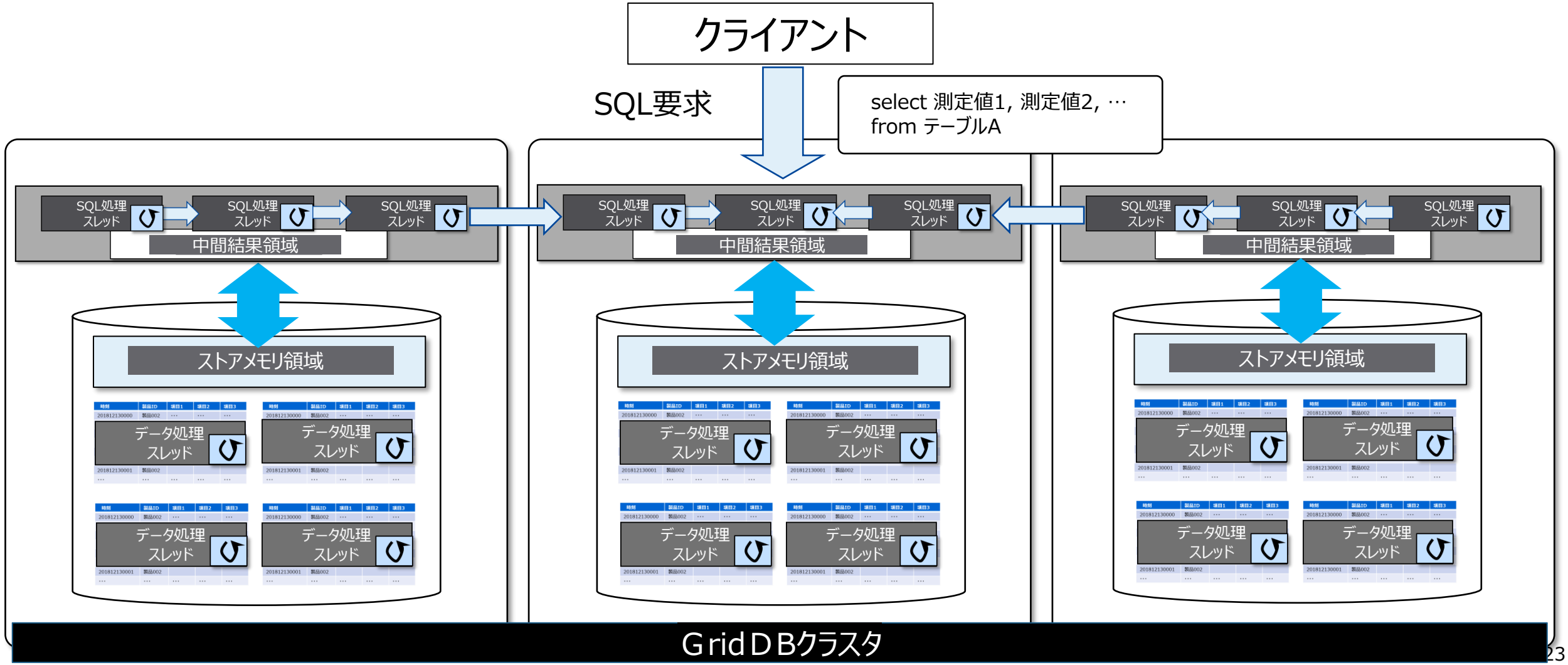
蓄積されるデータの、
日時がパーティショニングキー
→日時で絞込みが効く場合

日時に加え、製品IDなど
別キーでハッシュ分割
→大量スキャンを処理する場合



1．テーブル定義：パーティショニングテーブルのスキャンの動き

パーティションのスキャンがノード毎、データ処理スレッドごとに並列に動作



2．SQLチューニング：パーティションプルーニングによる絞込み

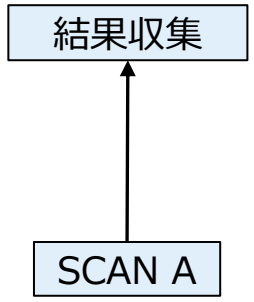
パーティショニングキー(日時)による絞込みで必要なデータのみインメモリに

SQLとプランの例

SQL

```
select 測定値1, 測定値2, …
from テーブルA
```

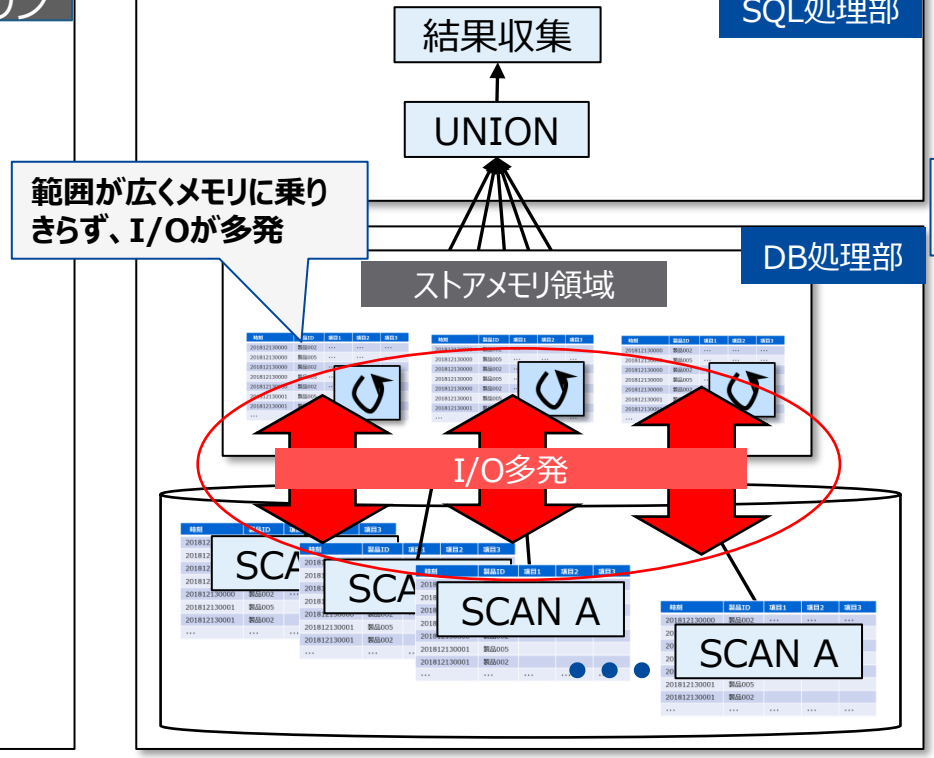
プラン



(A)SQL実行状態
(全範囲が検索対象)

SQL

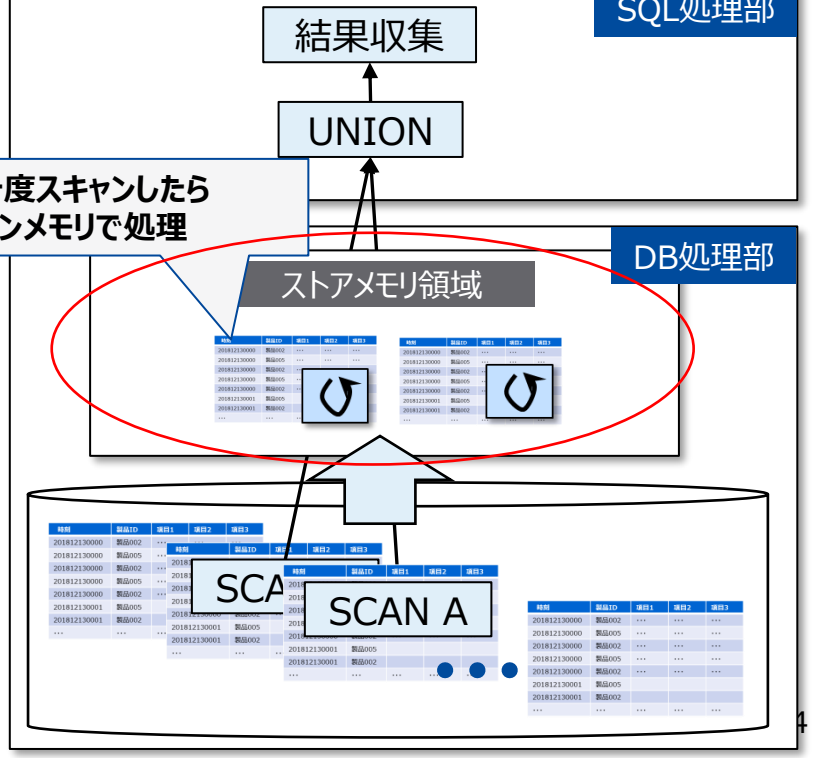
```
select 測定値1, 測定値2, …
from テーブルA
```



(B)必要な日時範囲を検索
(パーティションプルーニングで絞込み)

SQL

```
select 測定値1, 測定値2, …
from テーブルA
where 日時 BETWEEN TIMESTAMP('2024-06-…')
      AND TIMESTAMP('2024-07-…')
```



3．SQLチューニング：演算並列処理数のヒント指定

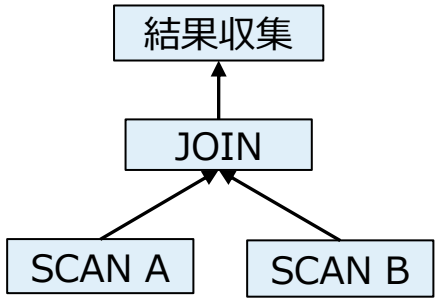
ヒント句でSQL演算の並列処理数を調整し、最大限CPUを活用

SQLとプランの例

SQL

```
select *
from A inner join B on A.id = B.id
          A.ts = TIMESTAMP('…')  and
          B.ts = TIMESTAMP('…')  and
          B.b = 'BBB'
```

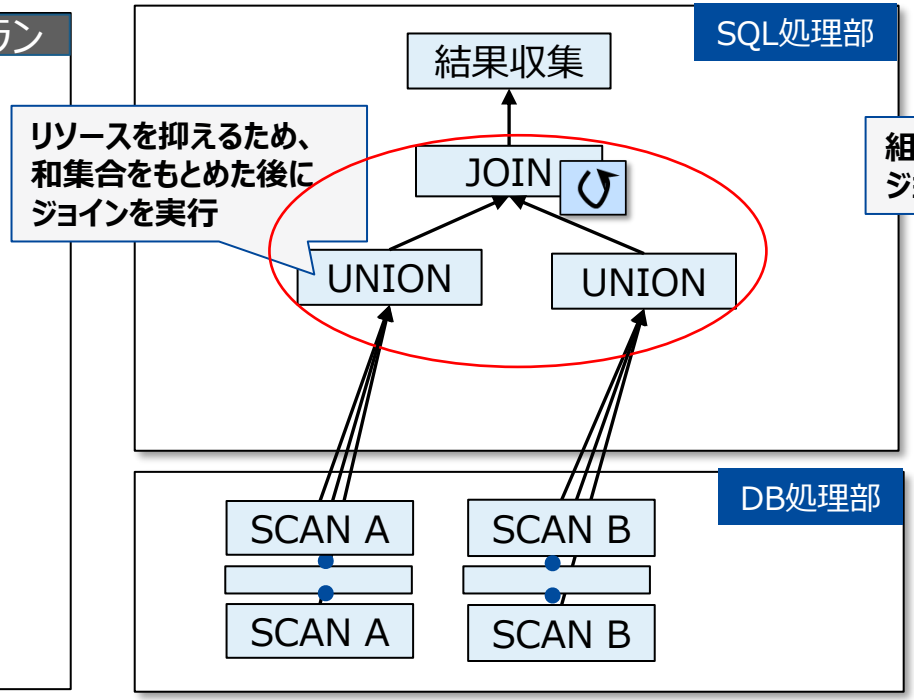
プラン



(A)通常のSQL実行の状態

SQL

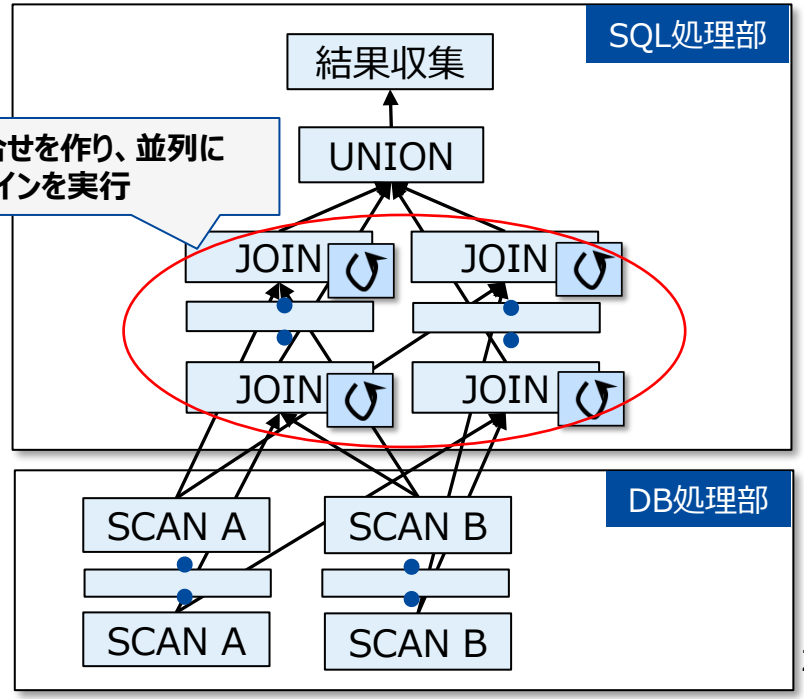
```
select *
from A inner join B on AA.id = B.id
          A.ts = TIMESTAMP('…')  and
          B.ts = TIMESTAMP('…')  and
          B.b = 'BBB'
```



(B)ヒント指定時のSQL実行状態
(並列処理数を増加)

SQL

```
/*+ MaxDegreeOfExpansion(演算数) */
select *
from A inner join B on AA.id = B.id
          A.ts = TIMESTAMP('…')  and
          B.ts = TIMESTAMP('…')  and
          B.b = 'BBB'
```

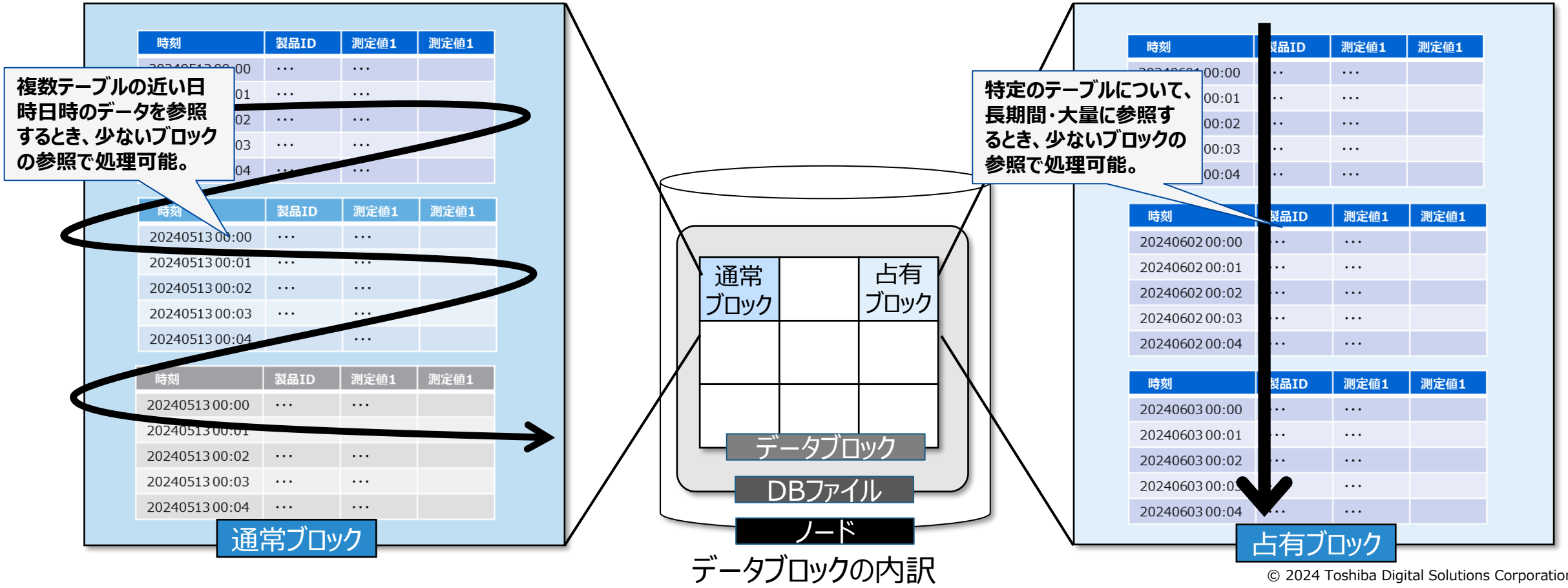


4．データ格納方法の選択：ブロック設定

テーブルの参照パタンに合わせてデータ格納し、メモリ効率とI/O効率を向上

(A)通常ブロック設定
・登録日時が近いデータを１ブロック格納
・複数テーブルの近い日時の参照向き

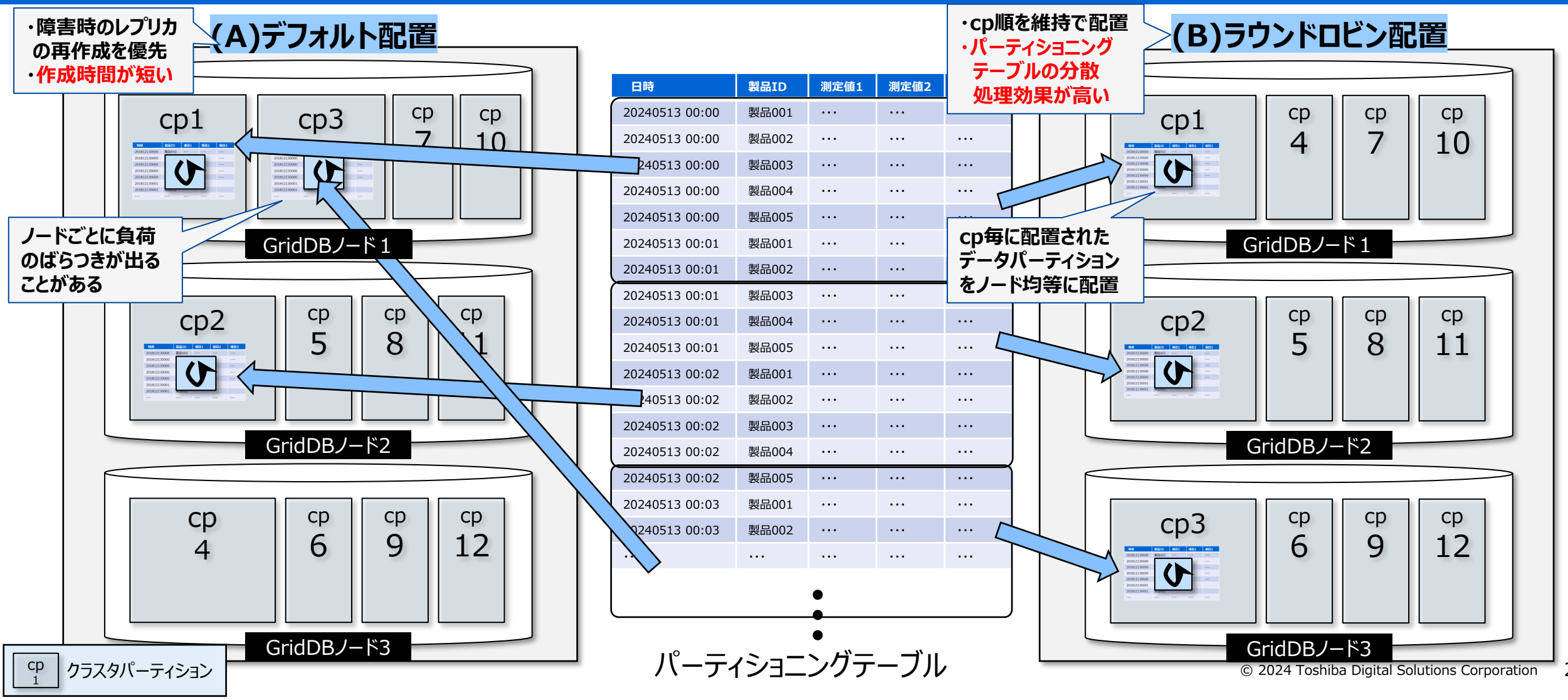
(B)占有ブロック設定
・１テーブルのデータを１ブロック格納
・大きなテーブルの広い範囲の参照向き



データブロックの内訳

5．ノード分散方法の選択　クラスタパーティション配置指定

ラウンドロビン配置：データパーティションを均等に配置して分散効果を最大化

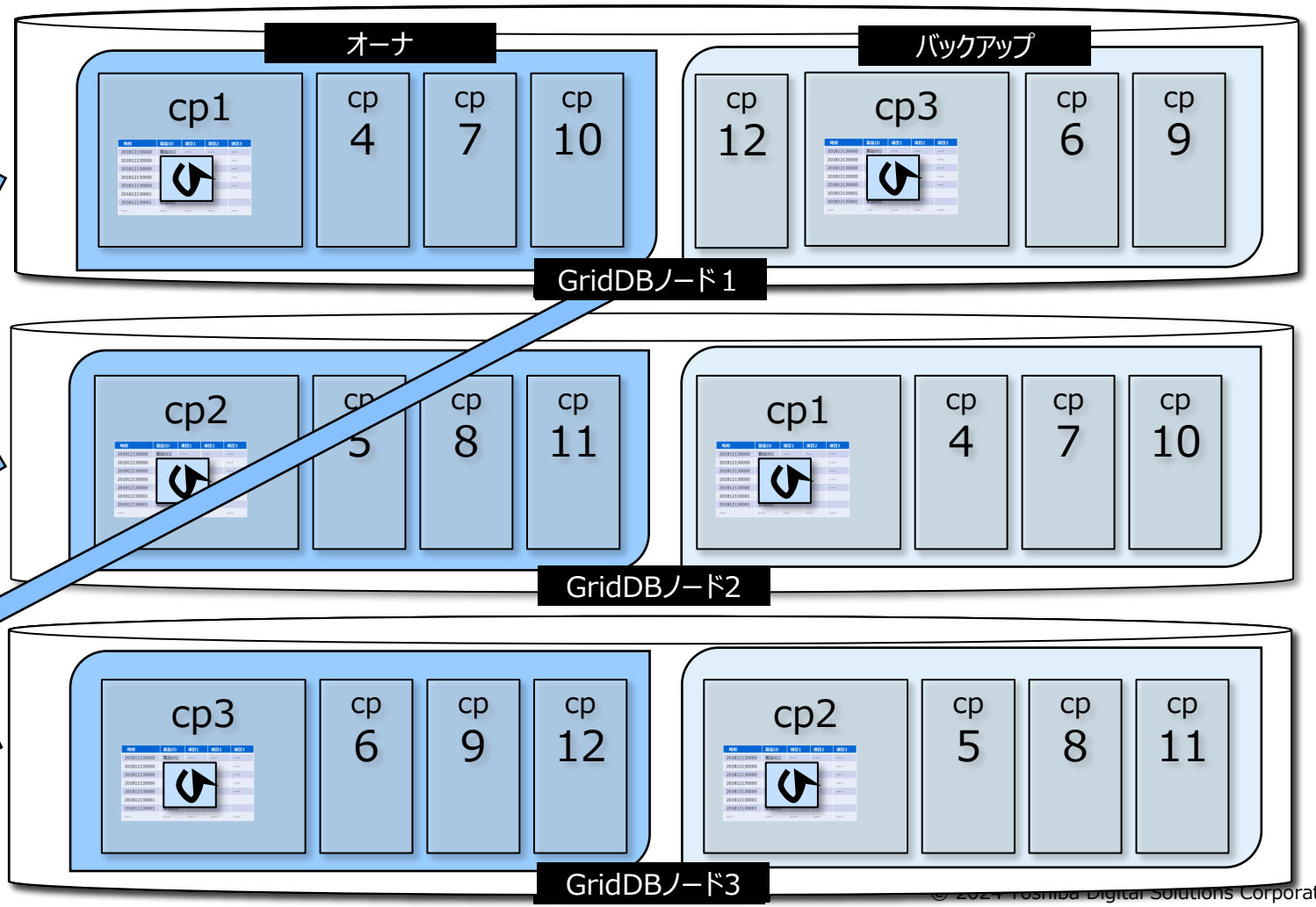


5．クラスタパーティション配置　ノード分散

ラウンドロビン配置：障害発生しても復旧後は均等な配置を維持

パーティショニングテーブル

日時	製品ID	測定値1	測定値2	測定値3
20240513 00:00	製品001	…	…	…
20240513 00:00	製品002	…	…	…
20240513 00:00	製品003	…	…	…
20240513 00:00	製品004	…	…	…
20240513 00:00	製品005	…	…	…
20240513 00:01	製品001	…	…	…
20240513 00:01	製品002	…	…	…
20240513 00:01	製品003	…	…	…
20240513 00:01	製品004	…	…	…
20240513 00:01	製品005	…	…	…
20240513 00:02	製品001	…	…	…
20240513 00:02	製品002	…	…	…
20240513 00:02	製品003	…	…	…
20240513 00:02	製品004	…	…	…
20240513 00:02	製品005	…	…	…
20240513 00:03	製品001	…	…	…
20240513 00:03	製品002	…	…	…
…	…	…	…	…



05

GridDBを利用したい

ラインアップ

商用版

Enterprise Edition

http://griddb.com

商用ライセンス
商用サポート

クラウドデータベース サービス

http://cloud.griddb.com

商用サービス
商用サポート（無料プランを除く）

オープンソース

Community Edition

https://github.com/griddb

AGPL3.0/ Apach2.0
コミュニティサポート

クラウドデータベースサービス

共用環境無料プラン	専用環境プラン		
フリー *Free*	スタンダード *Standard*	プロフェッショナル *Professional*	エンタープライス *Enterprise*
月額料金　無料 クレジットカード登録不要	月額料金 295,000円～ （税抜）	月額料金 365,000円～ （税抜）	月額料金 495,000円～ （税抜）
● 共有 vCPU ● 共有メモリ ● 10 GB ストレージ ● ３ノード以上で構成 ● 1ユーザのWeb APIリクエスト数 3,000回／10分 ● コミュニティサポート	● 4 vCPU ● 16 GB メモリ ● 1 TB ストレージ (オプションで追加可能) ● １ノードまたは３ノード構成を用意 (オプションでノード追加可能) ● 商用サポート	● 8 vCPU ● 32 GB メモリ ● 1 TB ストレージ (オプションで追加可能) ● １ノードまたは３ノード構成を用意 (オプションでノード追加可能) ● 商用サポート	● 16 vCPU ● 64 GB メモリ ● 1 TB ストレージ (オプションで追加可能) ● １ノードまたは３ノード構成を用意 (オプションでノード追加可能) ● 商用サポート

共用環境無料プランでデータ量や処理量が増えていった場合は、専用環境プランに容易に移行ができます。

デベロッパーズサイト　その他

GridDB　公式サイト		http://griddb.com
GridDB Cloud 公式サイト		http://cloud.griddb.com
Github サイト		https://github.com/griddb
GridDB Developers サイト		https://griddb.net
X		@griddb_jp
facebook		https://www.facebook.com/griddb/
linedin		https://www.linkedin.com/company/griddb/
slack		https://griddb.slack.com/
stackoverflow		https://stackoverflow.com/questions/tagged/griddb
youtube		https://www.youtube.com/@GridDB

TOSHIBA