

NoSQL/SQLデュアルインターフェースを備えた IoT向けデータベースGridDB ～ コマンドライン・インターフェース(CLI)を使ってみましょう ～



TOSHIBA

東芝デジタルソリューションズ株式会社 GridDBコミュニティ版担当

野々村 克彦

2022.03.11

Contents

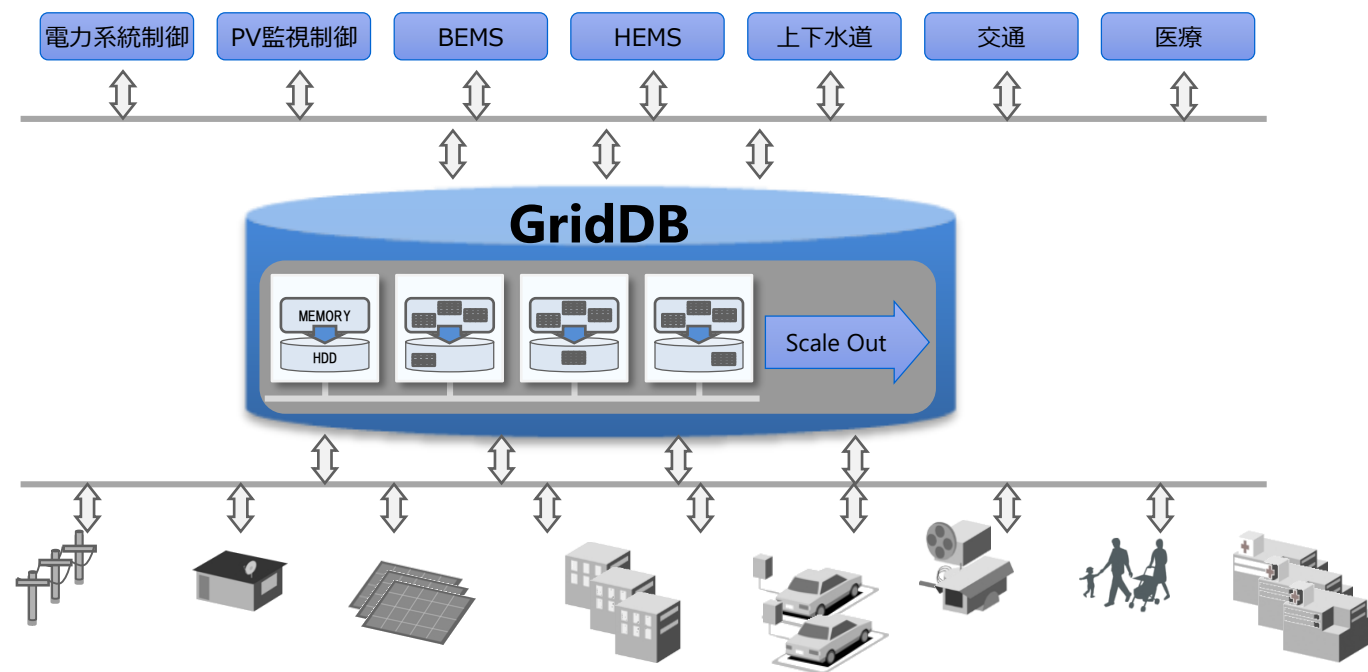
- 01 IoT向けデータベースGridDBの概要
- 02 コマンドライン・インターフェースによるGridDB操作
- 03 OSS活動
- 04 まとめ

01

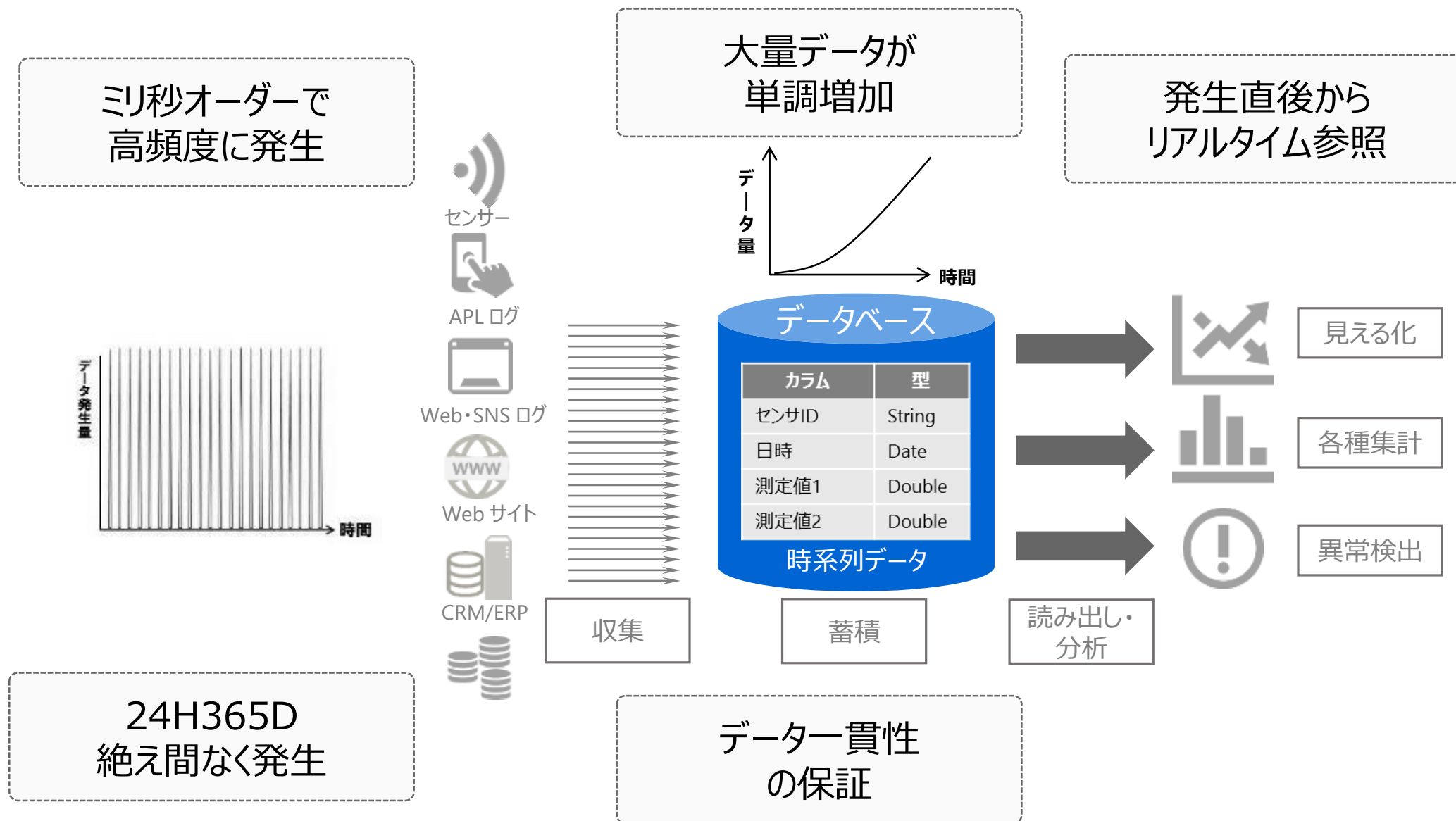
IoT向けデータベースGridDBの概要

IoT向けデータベースGridDB

- 日本発のビッグデータ／IoT向けデータベース
- V1.0製品化(2013年)、OSS化(2016年)、V4.6CE(2021年)
- 社会インフラ、製造業を中心に、高い信頼性・可用性が求められるシステムに適用されている



IoTデータの特長



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

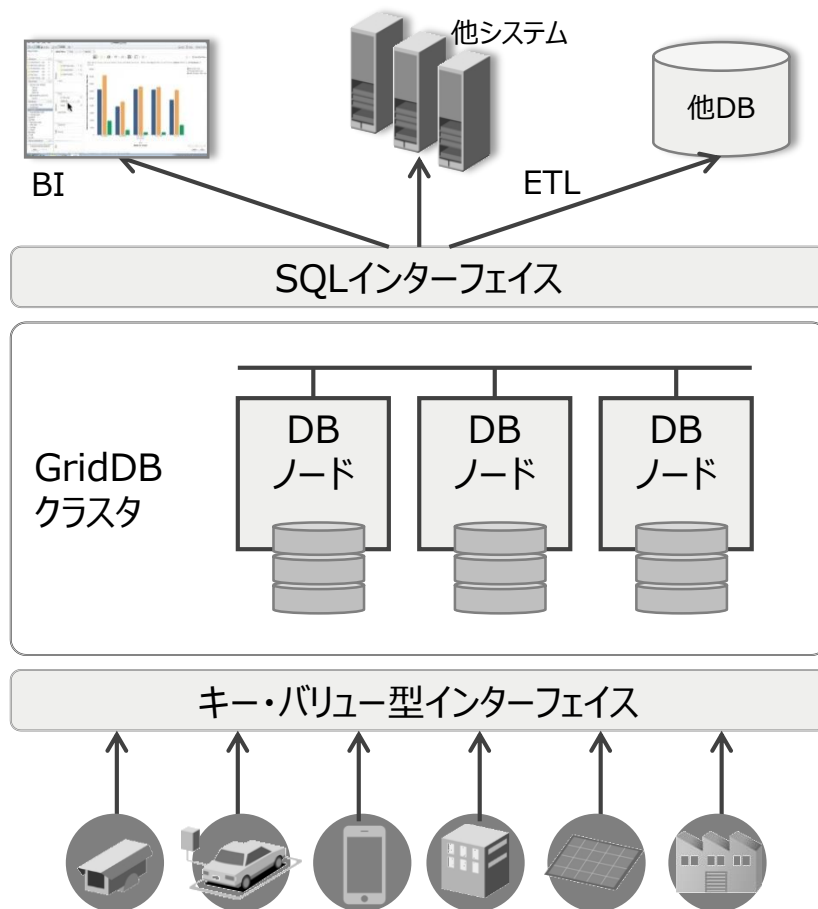
開発の俊敏性と 使いやすさ



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

時系列データ：時間とともに変化するデータ

NoSQLとSQLのデュアルインターフェイス



SQLインターフェイス

- 分散並列SQLデータベース
- 巨大テーブルに対するテーブルパーティショニング
- ジョインなど複数テーブルに対するSQL
- JDBC/ODBCドライバー

NoSQL(キー・バリュー型)インターフェイス

- 高可用、高スループット指向のKVS
- キーコンテナに対するCRUD、TQL
- バッチ処理(MultiPut/MultiGet/MultiQuery)
- Java/C/Python/Node.JS/Go APIなど

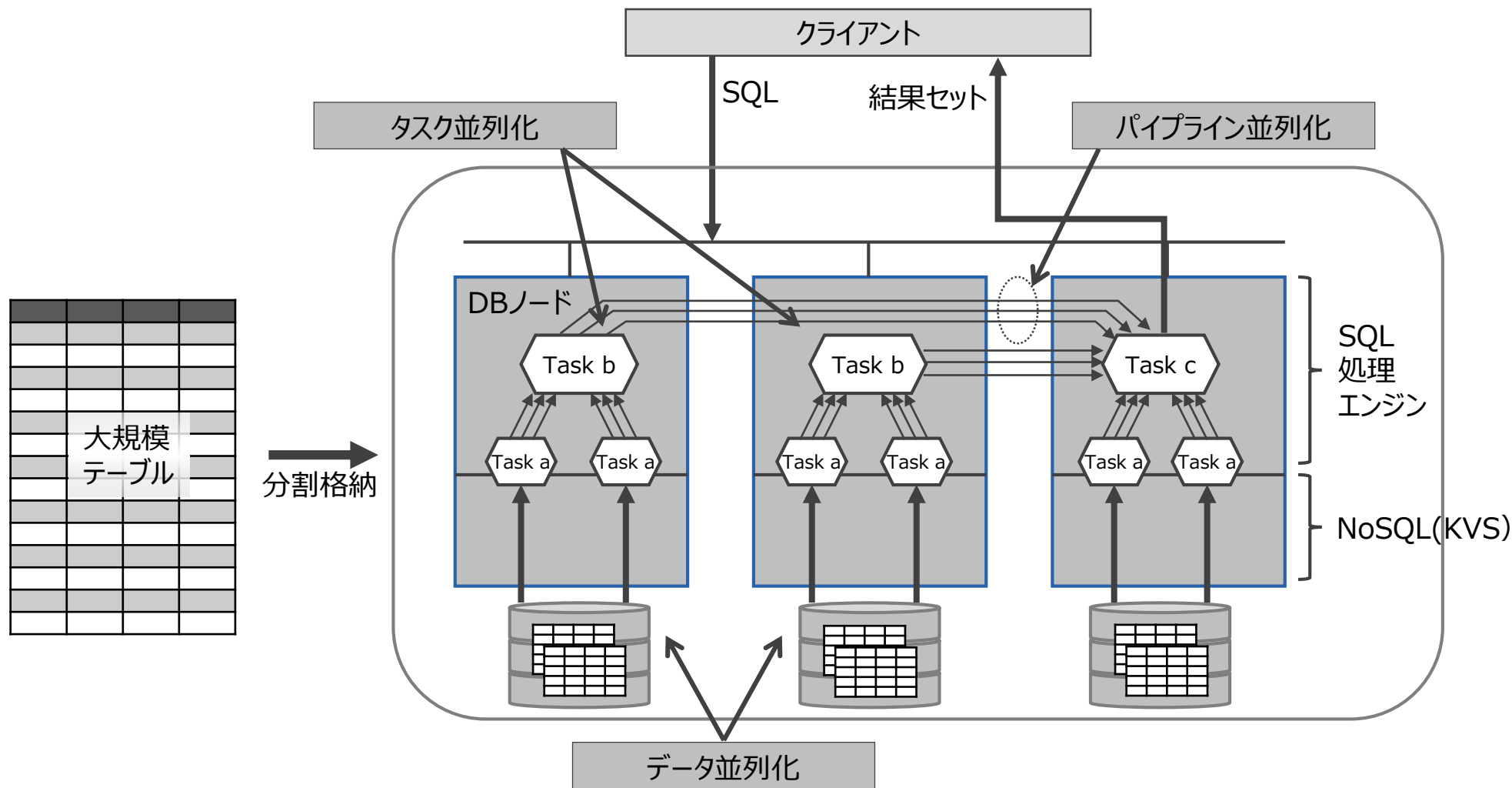
- **データ定義言語 (DDL)** : CREATE(DROP) DATABASE/USER/TABLE/INDEX/VIEW
- **データ制御言語 (DCL)** : GRANT/REVOKE/SET PASSWORD
- **データ操作言語 (DML)** : SELECT/INSERT/UPDATE/DELETE
- **句**
 - FROM/GROUP BY/HAVING/ORDER BY/WHERE/LIMIT/OFFSET
 - JOIN
 - 内部結合 [NATURAL] [INNER] JOIN
 - 左外部結合 [NATURAL] LEFT [OUTER] JOIN
 - クロス結合 [NATURAL] CROSS JOIN
 - UNION/INTERSECT/EXCEPT
- **演算子、関数、CASEなど**

テーブルパーティショニング

- データ登録数が多い巨大なテーブルのデータを分散配置することで、プロセッサの並列実行を可能とし、巨大テーブルのアクセスを高速化するための機能
 - ハッシュパーティショニング
 - ✓ 選択基準：散らすべきキーにランダム性が高く、キーの間に処理上の関連性が無い場合
 - インターバルパーティショニング
 - ✓ 選択基準：散らすべきキーの数値的な範囲で散らしたい場合
 - インターバルハッシュパーティショニング
 - ✓ 選択基準：インターバルパーティショニングでは力不足の場合

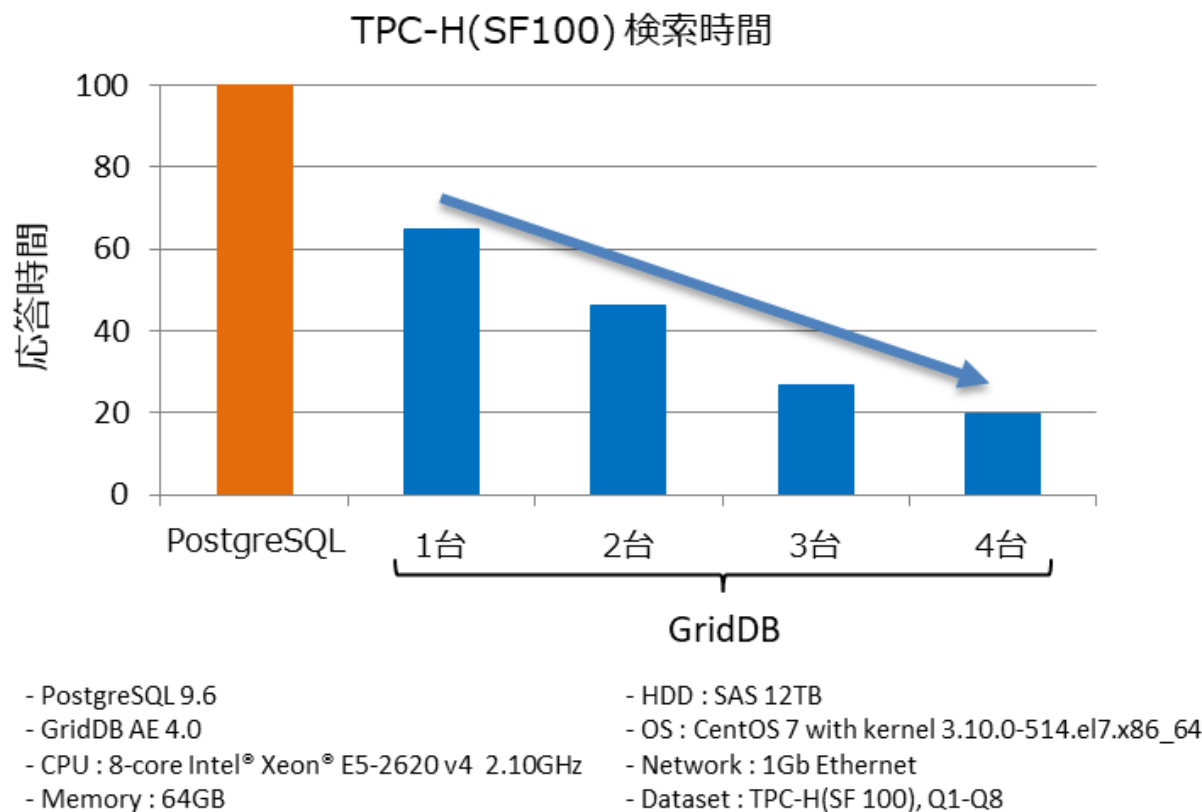
```
-- ハッシュ
CREATE TABLE a3 (code INT, ts TIMESTAMP, dest STRING NOT NULL)
  PARTITION BY HASH(dest) PARTITIONS 10
-- インターバル
CREATE TABLE a1 (code INT, ts TIMESTAMP NOT NULL, dest STRING)
  PARTITION BY RANGE(ts) EVERY(1,DAY)
-- インターバルハッシュ
CREATE TABLE a4 (code INT NOT NULL, ts TIMESTAMP, dest STRING)
  PARTITION BY RANGE(ts) EVERY(1,DAY)
  SUBPARTITION BY HASH(dest) SUBPARTITIONS 2
```

テーブルパーティショニングとSQLにおける分散並列処理



SQL性能

- TPC-H(Transaction Processing Performance Council)
- SQLのスケールアウト効果



主な適用事例

- 社会インフラ、製造業を中心に、高い信頼性・可用性が求められるシステムに適用

- フランス リヨン 太陽光発電 監視・診断システム
発電量の遠隔監視、発電パネルの性能劣化を診断
- 電力会社 低圧託送業務システム
スマートメータから収集される電力使用量を集計し、需要量と発電量のバランスを調整
- HDD製造会社 品質管理システム
製造装置のセンサーデータを長期に渡って蓄積・分析し、品質分析・改善に適用
- 半導体製造ライン 履歴管理システム
製造履歴や品質履歴、材料データなどを横串で分析し、製品の品質管理やトレーサビリティに利用
- 半導体製造ライン 異常検知システム
製造ラインのセンサーデータをリアルタイムにAIで分析し、製造ラインの異常を検知
- デンソー ファクトリー IoT
工場のDigitalTwinを実現し、生産性向上
- DENSO International Americaの次世代の車両管理システム
車両の各種センサーデータを用いる車両管理システムのPoC
-



GridDB Community Edition

高頻度・大量に発生する時系列データの蓄積とリアルタイムな活用をスムーズに実現する次世代の
オープンソースデータベース



GridDB Enterprise Edition

高頻度・大量に発生する時系列データの蓄積とリアルタイムな活用をスムーズに実現し、ビジネスを大きく成長させるために
最適化された次世代のデータベース



GridDB Cloud

高頻度・大量に発生する時系列データの蓄積とリアルタイムな活用をスムーズに実現する
クラウドデータベースサービス

各エディションの違い

- インターフェイスはほぼ同じ
- クラスタ構成の有無の違い

項目	機能	Community Edition	Enterprise Edition	Cloud
サポート			✓	✓
プロフェッショナルサービス			✓	✓
データ管理	時系列コンテナ	✓	✓	✓
	コレクションコンテナ	✓	✓	✓
	索引	✓	✓	✓
	アフィニティ	✓	✓	✓
	テーブルパーティショニング	✓	✓	✓
クエリ言語	TQL	✓	✓	✓
	SQL	✓	✓	✓
NoSQLインタフェース	Java	✓	✓	✓
	C言語	✓	✓	✓
NewSQL(SQL) インタフェース	JDBC	✓	✓	✓
	ODBC		✓	✓
WebAPI		✓	✓	✓
時系列データ	時系列分析関数	✓	✓	✓
	期限付き解放機能		✓	✓
	長期アーカイブ		✓	✓
クラスタリング	機能クラスタ構成		✓	✓
	分散データ管理		✓	✓
	レプリケーション		✓	✓
運用管理	ローリングアップグレード		✓	
	オンラインバックアップ		✓	✓
	エクスポート / インポート		✓	✓
	運用管理GUI		✓	✓
セキュリティ	運用コマンド		✓	✓
	信暗号化 (TLS/SSL)		✓	✓
オンプレミス環境	認証機能 (LDAP)		✓	
	オンプレミス環境	✓	✓	
クラウドサービス				✓

02

GridDB コマンドライン・インターフェースによるGridDB操作

GridDB コマンドライン・インターフェース(CLI)

- 対話的に操作できるお手軽なツール（コマンドインタプリタ、gs_shとも呼ばれる）
 - 初心者向け
 - （Python/Javaなどで）作成したアプリの動作確認用、など

GridDBの学習の流れ：

1． GridDBサーバのインストール、設定、起動など

← CLIによるGridDB操作

2． Python/Javaなどのクライアントライブラリによるアプリ開発

3． バッチ処理、テーブルパーティショニングなど

	JDBCドライバ (コンパイル言語)	Javaクライアント (コンパイル言語)	Pythonクライアント (スクリプト言語)	コマンドライン・インターフェース(CLI)
お手軽さ	△	△	○	◎
操作範囲	○	○	○	△
インターフェース(I/F)	SQL	NoSQL	NoSQL	NoSQL/SQL

ソフトウェアスタック

コマンドライン・インターフェース (NoSQL/SQL I/F)

Javaクライアント
(NoSQL I/F)

JDBCドライバ
(SQL I/F)

Python/Node.JS/Go/PHPクライアント
(NoSQL I/F)

Cクライアント
(NoSQL I/F)

サーバ

①Java環境によるSQLインターフェースの利用

- SQLWorkbench/J (<https://www.sql-workbench.eu/>)

- Java環境 & JDBCドライバ経由で動くSQL実行ツール

※特定の国の政府機関は利用できない制限があるが、日本は利用制限に含まれていない

②Python環境によるSQLインタフェースの利用

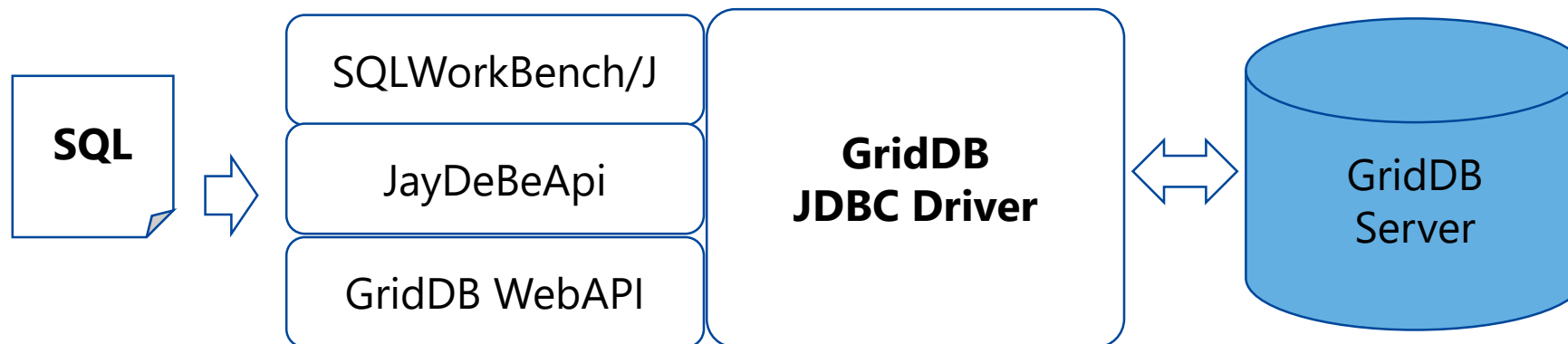
- JayDeBeApi (<https://github.com/baztian/jaydebeapi>)

- Python標準のデータベースAPIの仕様であるDB-API(PEP 249)の1実装

- <https://www.python.org/dev/peps/pep-0249/>

③WebAPIによるSQLインタフェースの利用

- GridDB WebAPI ※SQLはSELECT文のみ



コマンドライン・インターフェースによるGridDB利用方法の内容

＜利用方法の流れ＞

1. 各種ソフトウェアのインストール
2. GridDBサーバの設定・起動・クラスタ構成
3. コマンドライン・インターフェースの起動
4. コマンド実行

＜動作環境例＞

- Ubuntu
- 同一マシンに全ソフトウェアをインストール。ローカル実行
- GridDBのクラスタ名はmyCluster(デフォルト)
- GridDB管理者の名前はadmin、パスワードはadmin

Dockerfile

```
FROM Ubuntu:bionic
```

```
RUN apt-get update && apt-get install -y wget python2.7 default-jre vim  
RUN ln -sf /usr/bin/python2.7 /usr/bin/python2
```

各種ソフトウェアのインストール

1. GridDBサーバ (Javaクライアント含む)のインストール

```
$ wget https://github.com/griddb/griddb/releases/download/v4.6.1/griddb_4.6.1_am64.deb
```

```
$ dpkg -i griddb_4.6.1_amd64.deb
```

2. GridDB JDBCドライバのインストール

```
$ wget https://repo1.maven.org/maven2/com/github/griddb/gridstore-jdbc/4.6.0/gridstore-jdbc-4.6.0.jar -O /usr/share/java/gridstore-jdbc.jar
```

3. GridDB CLIのインストール

```
$ wget https://github.com/griddb/cli/releases/download/v4.6.0/griddb_cli_4.6.0_am64.deb
```

```
$ dpkg -i griddb-cli_4.6.0_amd64.deb
```

(設定ファイルのダウンロード)

```
$ wget https://raw.githubusercontent.com/griddb/cli/main/sample/init.gsh
```

※GridDBサーバ、Javaクライアント : <https://github.com/griddb/griddb>

※GridDB JDBCドライバ : <https://mvnrepository.com/artifact/com.github.griddb/gridstore-jdbc>

※GridDB CLI : <https://github.com/griddb/cli>

GridDBサーバの設定・起動・クラスタ構成

1. gsadmユーザでログイン

```
$ su - gsadm
```

2. GridDB管理者(admin)のパスワード設定

```
$ gs_passwd admin
```

```
> admin
```

3. GridDBサーバの起動、（1ノードでの）クラスタ構成

```
$ gs_startnode
```

```
$ gs_joincluster -u admin/admin -c myCluster
```

・Stat確認

```
$ gs_stat -u admin/admin
```

GridDBコマンドライン・インターフェース起動

1. 初期設定ファイルの設定(コピー)

```
$ cp /init.gsh .gsshrc
```

2. コマンドライン・インターフェース起動

```
$ gs_sh
```

```
>
```

init.gsh

```
# ノード変数の定義
setnode node0 127.0.0.1 10040
# クラスタ変数の定義
setcluster cluster0 myCluster 239.0.0.1 31999 $node0
setclustersql cluster0 myCluster 239.0.0.1 41999
# ユーザ情報の定義
setuser admin admin
```

実行例（１）

#接続

```
> connect $cluster0
```

テーブル(コンテナ作成)

```
> createcollection mycontainer c0 long c1 long  
(sql) create table t1 (c0 long, c1 long);
```

データ登録

```
> putrow mycontainer 1 2  
(sql) insert into t1 values(1, 2);
```

※SQL文の先頭が下記文字列のいずれかである場合、コマンド名sqlを省略することができます。

select update insert replace delete create drop alter grant revoke pragma explain

実行例 (2)

検索

> tql mycontainer select *;

(sql) select * from t1;

> get

切断

> disconnect

※TQL : 1テーブルに対するSELECT文に相当

実行例（3）

テーブル（コンテナ）一覧

> showcontainer

(sql) select * from "#tables";

ステータス確認

> showevent

> showconnection

クエリの実行計画表示

> tqlexplain mycontainer select *;

(sql)

explain select * from t1;

getplantxt

getplanjson

※ここまでは非常に簡単な操作になりますが、
更にテーブルパーティショニングの操作も可能です。

主なコマンド一覧（1）

変数定義

コマンド	引数	説明
setnode	[テキストファイル名]	ノード変数を定義します。
setcluster	クラスタ変数名 クラスタ名 マルチキャストアドレス ポート番号 [ノード変数...]	クラスタ変数を定義します。
setclustersql	クラスタ変数名 クラスタ名 SQLアドレス SQLポート番号	クラスタ構成にSQLの接続先を定義します。
setuser	ユーザ名 パスワード [OSユーザgsadmのパスワード]	クラスタにアクセスするユーザおよびパスワードを定義します。
show	[変数名]	変数の定義内容を表示します。
save	[スクリプトファイル名]	変数定義をスクリプトファイルに保存します。
load	[スクリプトファイル名]	スクリプトファイルを読み込み、実行します。

接続・切断

コマンド	引数	説明
connect	クラスタ変数 [データベース名]	GridDBクラスタに接続します。
disconnect		GridDBクラスタから切断します。

主なコマンド一覧（2）

コンテナ操作

コマンド	引数	説明
createcollection	コンテナ名 カラム名 タイプ [カラム名 タイプ ...]	コンテナ(コレクション)を作成します。
dropcontainer	コンテナ名	コンテナを削除します。
putrow	コンテナ名 値 [値...]	コンテナにrowを登録します。
removerow	コンテナ名 rowキー値 [rowキー値...]	コンテナのrowを削除します。
createindex	コンテナ名 カラム名 索引タイプ...	指定カラムに索引を作成します。
tql	コンテナ名 クエリ ;	TQL検索を実行し、検索結果を保持します。
sql	SQL文 ;	SQL文を実行し、検索結果を保持します。
get	[取得件数]	検索結果を取得し、標準出力に表示します。
tqlexplain	コンテナ名 クエリ ;	指定TQL文の実行計画を表示します。

※SQL文の先頭が下記文字列のいずれかである場合、コマンド名sqlを省略することができます。
select update insert replace delete create drop alter grant revoke pragma explain

主なコマンド一覧（3）

ステータス・一覧表示

コマンド	引数	説明
showcontainer (showtable)	[コンテナ名(テーブル名)]	コンテナ(テーブル)情報を表示します。
searchcontainer	[コンテナ名]	コンテナ名からコンテナを検索します。
showsql	[クエリID]	実行中のSQL処理を表示します。
showevent		実行中のイベント一覧を表示します。
showconnection		コネクションの一覧を表示します。

実行計画

コマンド	引数	説明
getplantxt	[テキストファイル名]	実行計画をテキスト形式で表示します。
getplanjson	[JSONファイル名]	実行計画をJSON形式で表示します。

03

GridDBのOSS活動



- **GridDBをGitHub上にソース公開(2016/2)**

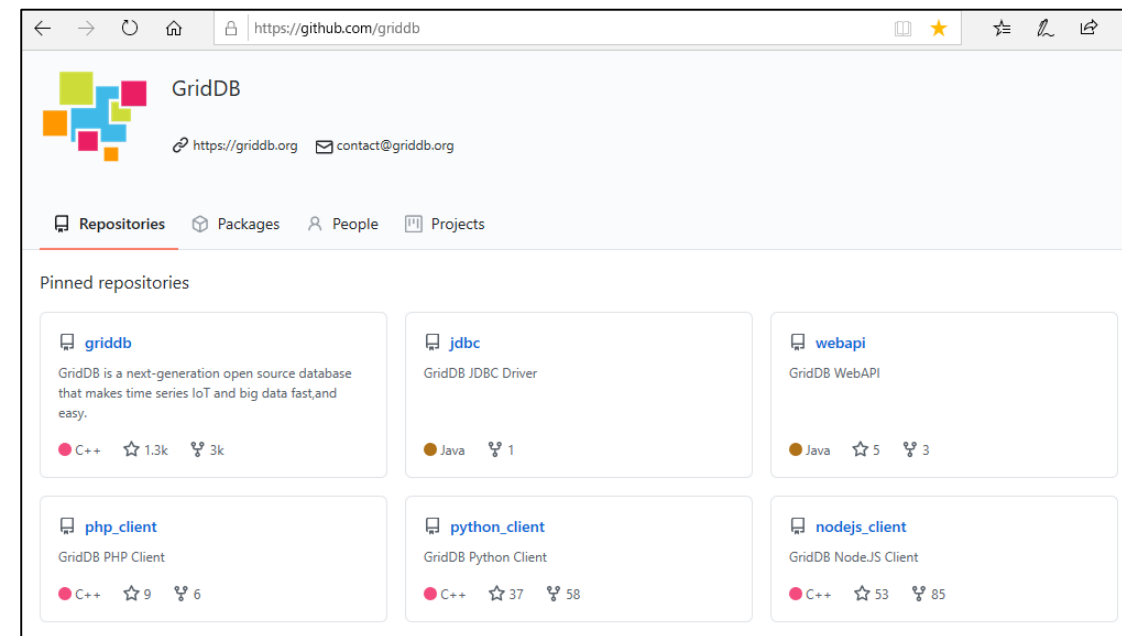
<https://github.com/griddb/griddb>

- **目的**

- ビッグデータ技術の普及促進
 - 多くの人に知ってもらいたい、使ってもらいたい。
 - いろんなニーズをつかみたい。
- 他のオープンソースソフトウェア、システムとの連携強化

- **ライセンス**

- サーバはAGPL-3.0
- 各種開発言語のクライアント、OSSとのコネクタはApache-2.0



主なOSS活動

- ① **GridDB本体の機能強化**
- ② **主要OSSとの連携強化**
- ③ **APIの拡充**
- ④ **GitHub以外のサイトからの情報発信**
 - パッケージ
 - デベロッパーズサイト（WP、ブログなど）・・・フィックスターズ社
 - SNS・・・フィックスターズ社
- ⑤ **主要OSSリポジトリへのコントリビュート**
- ⑥ **プラットフォームの拡充**
- ⑦ **その他**
 - OSCなどカンファレンス参加
 - ハンズオン無料セミナー・・・（株）アイ・ティ・イノベーション

OSS活動の全体イメージ

性能測定

収集

分散処理

可視化

Webアプリ

分析

AI/機械学習 ...

④ GitHub以外のサイトからの情報発信

PyPI/npm/Maven/Packagist/...

② 主要OSSとの連携強化

Spark
コネクタ

Fluentd/Grafana/Redash
プラグイン

YCSB
コネクタ

Kafka
コネクタ

Hadoop
MapReduce
コネクタ

WebAPI

Python/Node.JS/Go/PHP/Ruby/Perlクライアント

③ APIの拡充

Javaクライアント

JDBCドライバ

Cクライアント

① GridDB本体の機能強化

GridDB V4.6 CE(Community Edition)

⑤ 主要OSSリポジトリへのコントリビュート

⑥ プラットフォームの拡充

CentOS +
Ubuntu、openSUSE
Windows、MacOS
Docker

<https://github.com/griddb/>
GitHub



最近の主な活動（2021年度）

2021年

- 5月 Node API (Node.JS V16対応)
- 7月 Redash Plugin (SQL対応)
- 8月 V4.6.1CEのソース公開
Node API (バッチ処理対応)
- 9月 Pythonクライアント (Python3.9対応)
- 11月 V4.6向けDockerイメージ

2022年

- 1月 Node API (worker threads対応)
Goクライアント(Go 1.16対応)
- 3月 PHPクライアント(PHP 8対応)

- アプリケーション開発者向けのサイト
 - 様々なコンテンツを公開
 - ホワイトペーパー
 - ブログ
- など



- GridDBに関するリリース、イベント、などをお知らせします。
(日本国内向け)



05

まとめ

まとめ

- GridDBはビッグデータ・IoT向けのデータベースです。
- GridDBコマンドライン・インターフェースとオープンソース活動をご紹介しました。
 - 今後も様々な拡張、拡充を進めて参ります。

GridDBのオープンソース版(GridDB CE)を是非とも使ってみてください。

<https://github.com/griddb/>

ご参考 : GridDBに関する情報

- **GridDB GitHubサイト**

- <https://github.com/griddb/griddb/>

griddb github

検索

- **GridDB デベロッパーズサイト**

- <https://griddb.net/>

griddb net

検索

- **Twitter: GridDB (日本)**

- https://twitter.com/griddb_jp

griddb jp

検索

- **Twitter: GridDB Community**

- <https://twitter.com/GridDBCommunity>

- **Facebook: GridDB Community**

- <https://www.facebook.com/griddbcommunity/>

- **Wiki**

- <https://ja.wikipedia.org/wiki/GridDB>

- **GridDB お問い合わせ**

- OSS版のプログラミング関連 : Stackoverflow(<https://ja.stackoverflow.com/search?q=griddb>)もしくはGitHubサイトの各リポジトリのIssueをご利用ください

- プログラミング関連以外 : contact@griddb.netもしくはcontact@griddb.orgをご利用ください



TOSHIBA