爆速！DBチューニング超入門
～DB性能の基礎とPG-Stromによる高速化～

日本仮想化技術株式会社
宮原 徹

自己紹介

- 本名：宮原 徹
- 1972年1月 神奈川県生まれ
- 1994年3月 中央大学法学部法律学科卒業
- 1994年4月 日本オラクル株式会社入社
  - PCサーバ向けRDBMS製品マーケティングに従事
  - Linux版Oracle8の日本市場向け出荷に貢献
- 2001年1月 株式会社びぎねっと 設立
- 2006年12月 日本仮想化技術株式会社 設立
- 2008年10月 IPA「日本OSS貢献者賞」受賞
- 2009年10月 日中韓OSSアワード「特別貢献賞」受賞
- 仮想化とクラウド活用（主にインフラ寄り）に関するコンサルティングを中心に活動

日本仮想化技術株式会社 概要

- 社名：日本仮想化技術株式会社
  - 英語名：VirtualTech Japan Inc.
  - 略称：日本仮想化技術／VTJ
- 設立：2006年12月
- 資本金：3,000万円
- 売上高：1億8100万円（2022年7月期）
- 本社：東京都渋谷区渋谷1-8-1
- 取締役：宮原 徹（代表取締役社長兼CEO）
- 伊藤 宏通（取締役CTO）
- スタッフ：11名（うち8名が仮想化技術専門エンジニアです）
- URL：http://VirtualTech.jp/
- 仮想化技術に関する研究および開発
  - 仮想化技術に関する各種調査
  - 仮想化技術を導入したシステムの構築・運用サポート
  - 5G活用のためのインフラ・サービス研究開発
  - DevOps支援サービスの提供
  - GPUを活用した超高速データ分析基盤「爆速DB」の提供

ベンダーニュートラルな
独立系仮想化技術の
エキスパート集団

VirtualTech Japan

本日のアジェンダ

- ビッグデータ分析のためのDBの性能の基本
  - RDBMSの性能決定要因を確認
- DBの性能を向上させるには
  - RDBMSの検索性能を向上させるためのテクニック
- GPU活用による爆速化 ～PG-Stromの仕組み～
  - GPUを活用した検索処理の超並列処理を行う国産OSS「PG-Strom」の特長をご紹介
- PG-Stromの活用
  - 「爆速DB」とユースケース
  - 学術系ビッグデータ分析

Think ITで本内容を連載開始

https://thinkit.co.jp/series/11210

VirtualTech Japan

SQLは未経験の方のために

- PostgreSQLを使ったSQL入門教科書
  - 私が書きました
- LPI-Japanより無償ダウンロード可能
  - 印刷版は実費頒布
  - GitHubでソース公開
- クリエイティブコモンズ
  - 表示 - 非営利 - 改変禁止 4.0 国際 (CC BY-NC-ND 4.0)

https://oss-db.jp/ossdbtext

VirtualTech Japan

基礎をおさらい

ビッグデータ分析のための DBの性能の基本

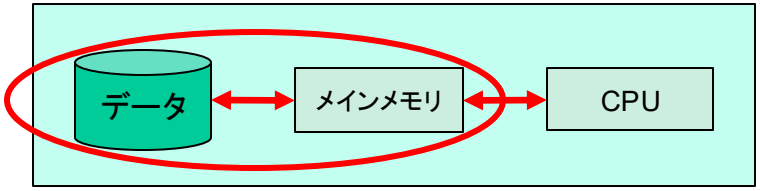
DBの検索性能を決定する要素

1. データの読み込み
2. 検索処理
3. 集計その他の演算処理

- 本資料はビッグデータ処理を想定した検索処理のみを取り上げています
- DBMS（DataBase Management System）というブラックボックスをSQLなどで操作する観点で解説しており、DBMSの実装によって詳細が異なる場合があります

データの読み込み

- データはストレージからメモリに読み込んで処理
- ストレージの読み込み速度とは
  - ストレージ自体の速度
  - 接続経路の速度
- ストレージ自体の速度
  - IOPSや読み書き速度（〇MB/秒）などで表される
  - HDDならプラッターサイズや回転速度が影響
  - SSDならシリコンやコントローラー速度が影響
- 接続経路の速度
  - SATAやSAS、NVMe（PCI Expressバス直結）
  - SATA(6Gbps)＜SAS(12Gbps)＜NVMe(64Gbps) ※
    - NVMeはPCIe 4.0のx4レーンを想定



※理論値であり、プロトコルオーバーヘッドなどで実速度は低下します

ストレージの接続経路と速度

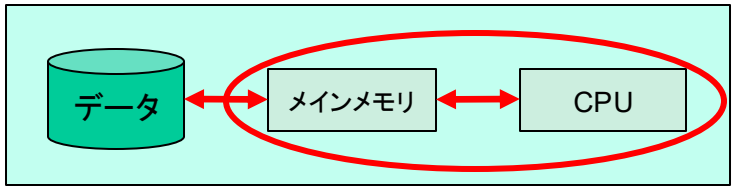
接続種類	帯域	主な用途
SATA	6Gbps（750MB/s）	一般的なPC
SAS	12Gbps（1.5GB/s）	サーバー 専用ストレージ
NVMe	64Gbps（8GB/s） （PCI-Express 4.0）	最近のPC

※経路速度なので実際の性能はディスク次第

VirtualTech Japan

検索処理

- メモリに読み込んだデータをCPUで処理
- WHERE句による条件一致処理
  - IN演算子やLIKE演算子などの処理を含む
  - インデックスが使われない場合には全件検索
  - 副問い合わせによる条件値の抽出
- JOIN句による表結合処理
- SELECT選択リストによるデータの抽出



集計その他の演算処理

- 検索処理されたデータに対する追加処理
  - CPUとメモリを使って処理
- ソート処理
- GROUP BY句による集約
- 集約関数による各種集計処理
  - COUNT関数など

演算処理を行った結果をアプリに返す

ビッグデータ分析特有のDB技術

- OLAP（Online Analytical Processing）
  - ここまでの話はデータの検索と更新処理が混在するOLTP（Online Transaction Processing）
  - OLAPはデータの多角分析を想定
  - 行指向のOLTPデータを多次元のOLAPデータに変換し分析処理

OLTPのままか、OLAPか

DBの性能を向上させるには

データベース性能向上の方法

- ストレージの読み込みを速くする
  - ハードウェアの改善など
- データの所在を明らかにする
  - インデックスの利用
  - パーティショニング
- 検索処理や演算処理を速くする
  - CPUやメモリ、ストレージを増やす
  - 単体性能を向上させるスケールアップ
  - 処理を分散させるスケールアウト

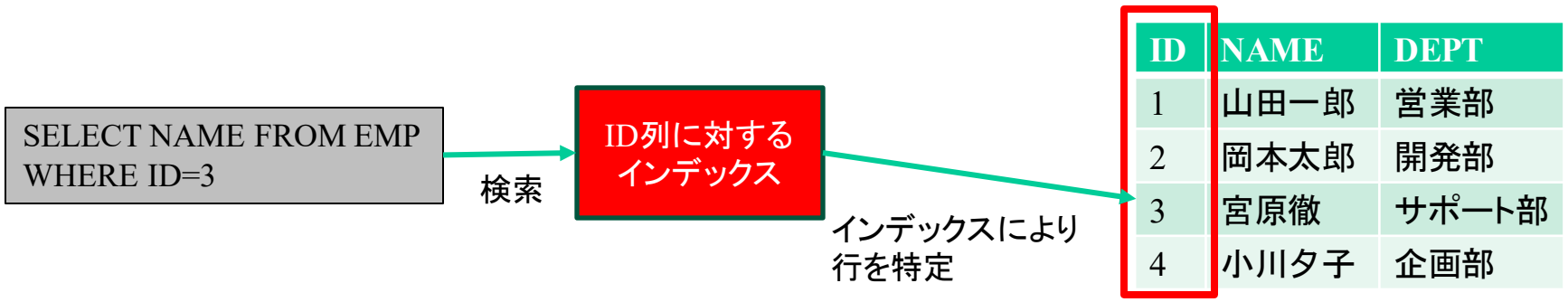
データの読み込みを速くする

- より高速なストレージデバイスを使用する
  - HDDよりSSD
  - SATA＜SAS＜＜NVMe
  - FibreChannelやiSCSIで接続経路を広帯域化
- デバイスを複数用意する
  - RAID 0（ストライピング）化
- 必要なデータだけ読み込むことで読み込み量を減らす
  - インデックスの活用
  - カラム（列）指向データベース
- 最初からメモリ（バッファ）に読み込んでおく
  - インメモリデータベース

データの所在を明らかにする

- データの在処が分からなければ全件検索するしかない
  - 読み込みに時間がかかる
  - メモリが大量に必要となる
- インデックスを利用してデータの所在を明らかにする
- インデックスも万能ではない
  - データ件数が少ない
  - カーディナリティが低い(「0か1か」など取る値の種類が少ない)
- パーティショニングでデータを分割する
- カラム(列)志向データベースの利用
  - 抽出したい列が決まっている場合

インデックスを使った高速化



パーティショニング

```
SELECT QTY FROM STOCK
WHERE DATE BETWEEN '2024-02-01' AND '2024-02-29'
```

日付範囲の条件に含まれるパーティション表だけを検索
※日付型のデータ指定方法は環境によって異なります

DATE	QTY
2024-01-01	10
2024-01-02	20
……	……

DATE	QTY
2024-02-01	15
2024-02-02	8
……	……

DATE	QTY
2024-03-01	12
2024-03-02	9
……	……

検索処理や演算処理を速くする

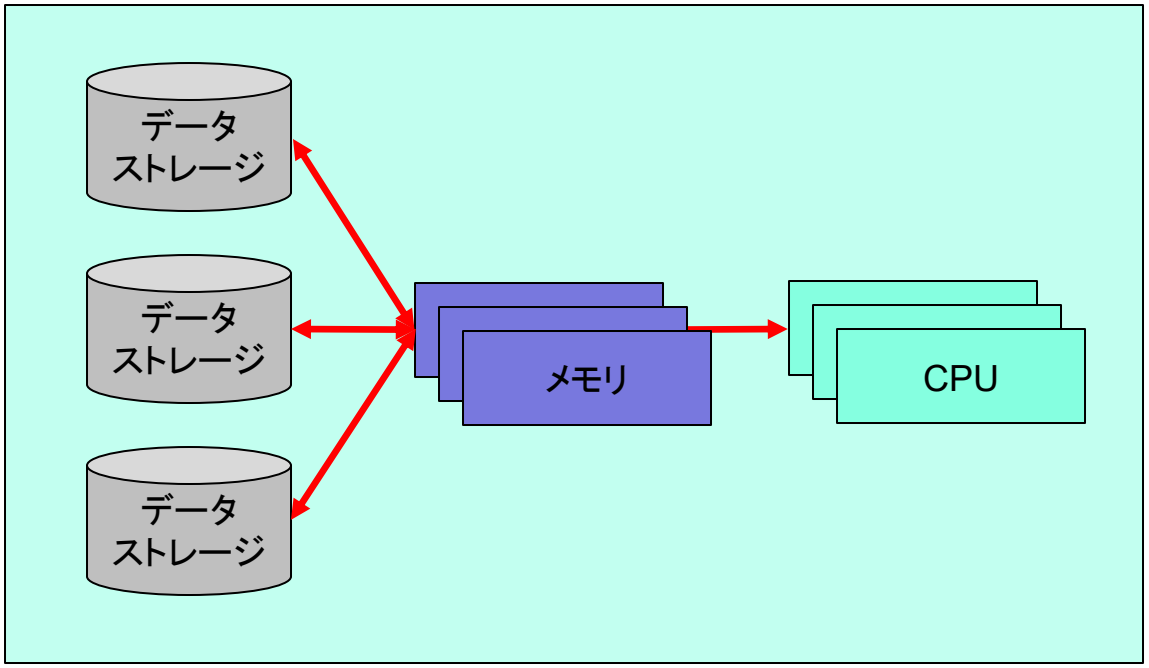
1台を高速化するスケールアップ

- CPUコアのクロック数を高速化する
  - プロセスルールの微細化の限界と発熱の制限
- CPUコア数を増やす
  - ダイサイズによる実装可能コア数の制限
  - マルチプロセスやマルチスレッドで活用

複数台で高速化するスケールアウト

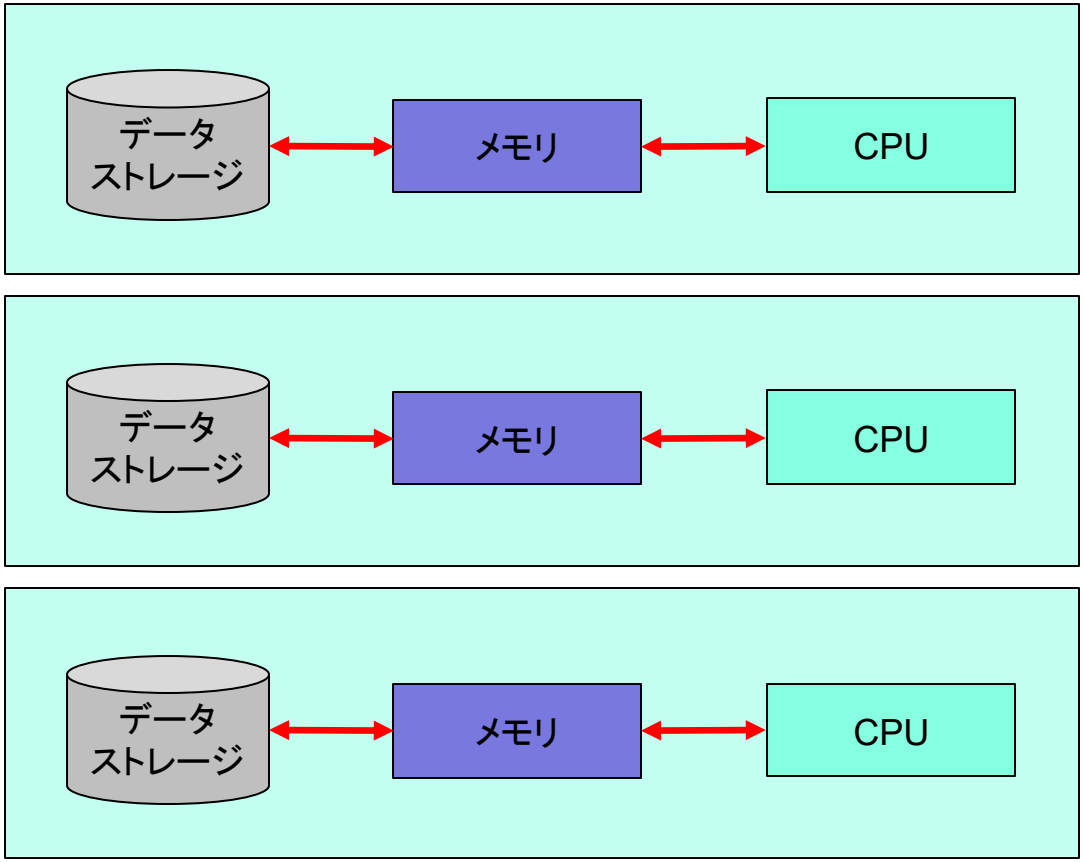
- 台数を増やしてクラスター化する
  - 複数台利用によるコストの増加
  - 管理やトラブル解決の煩雑さ

スケールアップ型



1台のマシンのハードウェアを強化するスケールアップ型

スケールアウト型



サーバーの台数を増やして並列動作させるスケールアウト型

VirtualTech Japan

中間まとめ：DB検索が遅くなる要因

- ストレージの速度が遅い
- データの量が多い
- CPUが遅い（クロック数・コア数）
- メモリが少ない
- インデックスが適切に使われていない
- 処理が複雑（副問い合わせや集計処理）

GPU活用による爆速化
～PG-STROMの仕組み～

PG-Stromの高速化手法

- PG-StromはPostgreSQLを拡張・高速化
  - GPUによる超並列処理
  - GPUDirect Storageによるデータ高速読込
  - 列指向データ形式Apache Arrowによるデータ読込の最適化
- 通常は遅くなる処理を高速化
  - インデックスが効かないフルスキャン検索
  - ビッグデータの集計処理
  - 位置情報データの検索処理

VirtualTech Japan

GPUによる超並列処理

- CPUとGPUのコア数に大きな違い
  - 現在のサーバー用CPUがプロセッサあたり最大192コア（AMD EPYC 9005 シリーズ）
  - 現在のエンタープライズ用GPUが20,480コア（NVIDIA B200の場合）
- データの検索処理や集計処理を並列化
  - より多くのコアで超並列処理
  - 単純な処理ほど並列化に向いている
    - 計算機は条件分岐などの複雑な処理が苦手

GPUDirect Storageによる高速読込

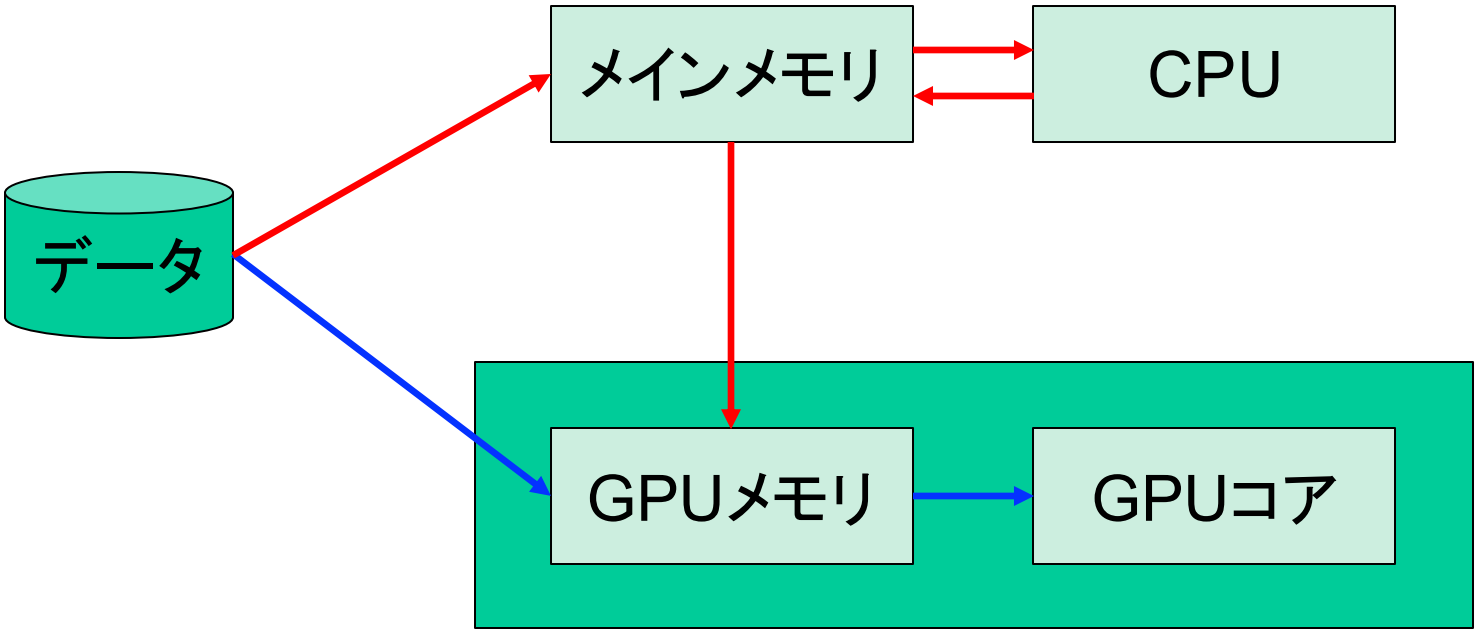
- NVMe接続されたストレージからGPUのメモリに対して直接データを読み込む技術
  - メインメモリ経由でGPUメモリに読み込むより高速
- PCIe 4.0 x4接続のSSDを4台接続して256Gbpsの帯域幅を確保
  - バイト換算で32GB/秒 ※理論値
- NVMe-oF（NVMe over Fabrics）により、外部ストレージから高速なEthernet経由で直接読み込みも可能
  - 100GbEでバイト換算で約12GB/秒 ※理論値

※理論値は概算値であり、プロトコルオーバーヘッドなどで実速度は低下します

VirtualTech Japan

GPUDirect Storage

- データをメインメモリ経由ではなく直接GPUメモリに読み込み

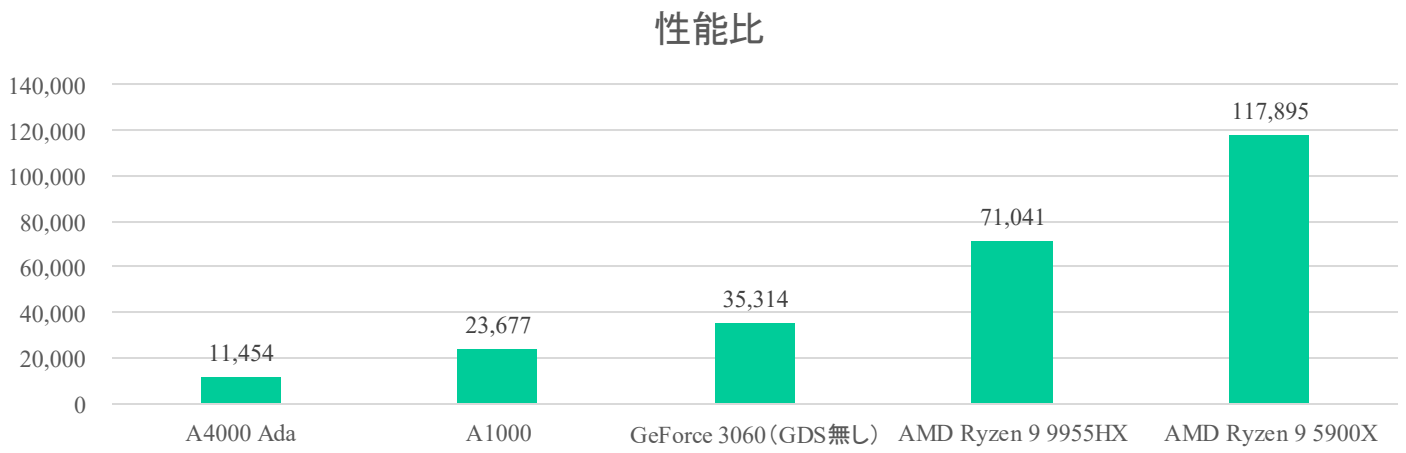


簡単なベンチマーク例

- RTX A4000搭載ノートPC
- 2つのテーブルをJOINするSELECT文
  - SELECT count(*) FROM t_test AS a, t_join AS b WHERE a.id = b.id GROUP BY a.ten;
  - t_test表：2億5千万行・121GB
  - t_join表：10万件・5096KB
- 結果はPG-Stromが約3倍高速
  - PG-Strom ON：38.3秒
  - PG-Strom OFF：114.7秒

ベンチマーク比較

- CPUおよびGPU、GDS無しで測定
  - GPUはミドルクラスでも十分性能が出ている
  - GDSが無いと性能は3割ダウン
  - CPUはメモリ帯域が性能に直結



VirtualTech Japan

Apache Arrowによる読込の最適化

- Apache Arrow形式はカラム（列）指向のデータフォーマット
  - インメモリデータベースに向いている
- あらかじめ集計などを行う列を抽出してデータファイル化
  - 読込量を減らして高速処理
- 更新はできないので検索処理のみに使用
  - OLTP系DBならテーブルからArrow形式に変換
- Fluentdの出力をArrow形式で保存
  - IoTなどのシステム

GPUキャッシュ

- GPUメモリ上にデータをキャッシュ
  - ストレージからの読込不要に
  - GPUメモリに乗りきるデータサイズに有効
    - Tesla A100で80GBのGPUメモリ
- メインメモリでOLTP処理されているテーブルデータを差分同期可能

PostGIS関数のGPU対応

- 地理空間情報を扱うPostGIS関数をGPU対応
  - 対応している関数は一部の関数のみ
- PostGISでは点や線分、区画（ポリゴン）などをジオメトリ型として扱う
  - 例：緯度経度からジオメトリ型（点）に変換できる
- 関数の例
  - st_contains()：ジオメトリa（ポリゴン等）にジオメトリb（点など）が包含されるかを判定
  - st_distance()：ジオメトリ間の距離を返す
- GiSTインデックス利用で更に高速化可能

VirtualTech Japan

現在の開発状況

- 新版バージョンVer6系が正式リリース
  - 内部アーキテクチャの改善
  - DPU（NICなどのプロセッサ）対応
  - Apache Arrowの疑似パーティション化
- 様々な機能拡張も順次実行中
  - 純国産です

https://github.com/heterodb/pg-strom

OSS版とサブスクリプションの違い

- OSS版とサブスクリプション購入には以下の違いがあります

機能	OSS版	サブスクリプション
GPU数	1基のみ	複数可能
GPUDirect Storage	1台のみ※	複数台
GiSTインデックス結合	×	○
HyperLogLog	×	○
技術サポート	×	メール
アップデートのサポート	×	○

※GPUDirect StorageはDatacenter（旧Tesla）またはRTX（旧Quadro）が必要です。GeForceでは動きません

VirtualTech Japan

OSS版PG-Strom導入

- OSS版PG-StromはCUDA対応GPUがあれば動作可能
  - GPUDirect Storageは対応GPUが必要
  - NVIDIA GeForceはGDS非対応で性能は3割減
  - NVMe SSDは1台に限定（それでも十分高速）
- 対応Linuxディストリビューションは CUDAがサポートされているもの
- インストールガイドを提供
  - 鋭意アップデート中
- Think ITの記事もあります

https://thinkit.co.jp/article/23090

VirtualTech Japan

PG-Stormの活用

「爆速DB」とユースケース
学術系ビッグデータ分析

爆速DB

- 「爆速DB」はPG-Stromをベースに導入から運用までをワンストップでサポートするデータ分析基盤ソリューションです
- 推奨ハードウェア構成をベースにしたハードウェアアプライアンスを提供しています
  - サブスクリプションのみ購入も可能
- 仮想マシンやコンテナでの動作もサポートします
- GPUが扱える各種クラウドサービスにも対応します
  - mdx、さくらの高火力サーバーなど

VirtualTech Japan

活用ユースケース

- 大容量ログの解析に
  - Webサービス等のアクセスログ
  - 通信ログ（通信サービス企業様）
  - IoTのセンサー等のログ
- 位置情報分析
  - 移動体通信デバイスの位置情報分析

学術系ビッグデータ分析

- 学術研究分野でのデータ分析においてPG-Stromの需要が増えている
  - 有意性のある統計分析を行おうとするとデータ量が多くなる
  - 多次元データ分析の必要性が高まっている
- RやPythonは手軽だがビッグデータ分析には不向き
  - 処理がシングルスレッドでスケールしない
  - 処理時間が長くなるので試行錯誤しづらい

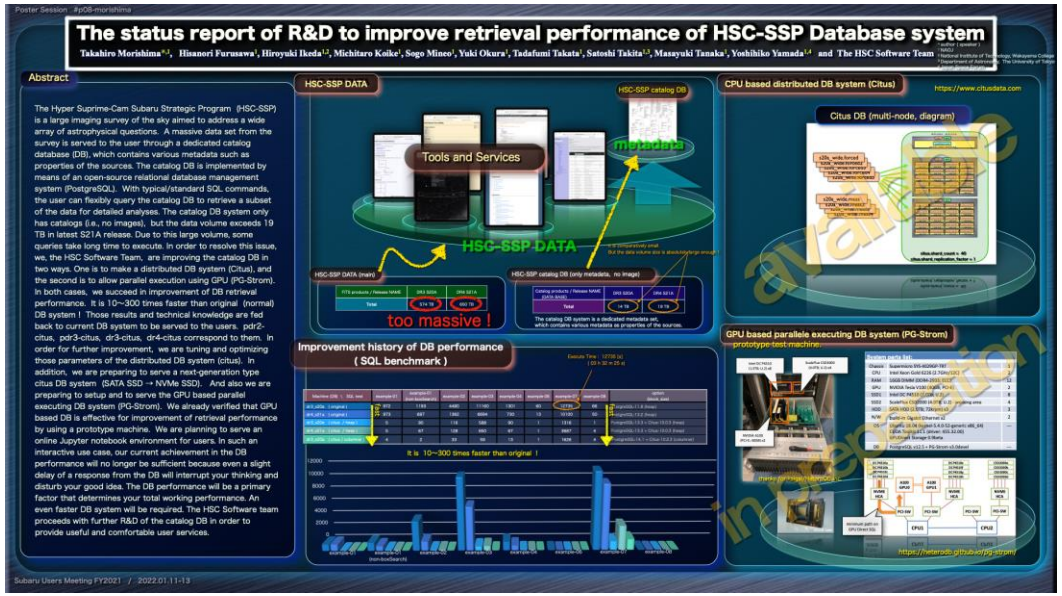
学術系活用事例

- mdxのGPUノードでPG-Stromを実行
- 「世界中の企業の財務諸表データを集めて解析・可視化するという世界初の研究に取り組み、企業が生み出した付加価値の分配の国ごとの違いなど、これまで知られていなかった企業活動の実態をグローバルな視点で明らかにしつつあります。」

https://mdx.jp/use-case/case2/

学術系活用事例

- すばる望遠鏡の観測した大量のデータの分析におけるPG-Stromの活用



https://www.naoj.org/Archive/Science/SubaruUM/SubaruUM2021/_src/p08_morishima.pdf

学術系活用が増えています

- 乱気流シミュレーションデータの分析
  - 3次元化されたデータを高速に処理
  - 参考 : https://turbulence.idies.jhu.edu/home

- 活用のご支援
  - 無償のOSS版でもRやPythonより高速
  - 基本的な導入から活用開始まで無償支援
  - 共同研究形式も可能

お問い合わせ先

メールにて

sales@VirtualTech.jp

評価したい等々、
お気軽にお問い合わせください

VirtualTech Japan

ありがとうございました