エッジコンピューティングに対応した大規模分散データの高速検索フレームワーク PGSpider

"PGSpider" Distributed Data Search Framework Allowing High-Speed Retrieval of Data Stored in Edge Devices

広域に分散格納された大規模データに対して，SQLによる問い合わせ検索技術を開発し，高速なデータ検索を実現

従来のIoT（Internet of Things）システムでは，センサーなどで生成されたデータをエッジ機器でフィルタリングし，クラウドサービス環境のデータベースに格納されたデータに対して，検索を行っていました。しかし，エッジコンピューティングにより高性能化したエッジ機器にデータが格納されるようになると，クラウドサービスにデータを集めないでエッジ機器のデータを直接検索したいというニーズが生まれてきました。

そこで，東芝は，広域に分散されたエッジ機器のデータ群を，仮想的に一つのテーブルとして扱うことで高速に検索できる技術を開発し，検索フレームワークに適用しました。

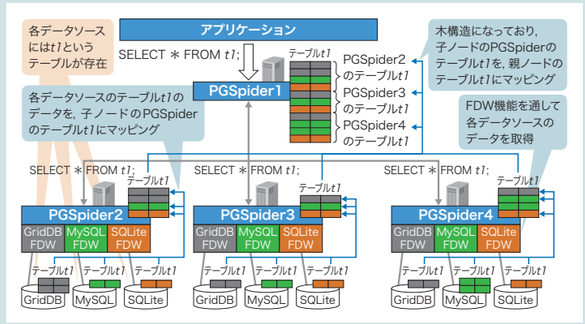

図1．PGSpiderの全体構成

PGSpiderは，テーブル名と関連付けられたデータソースのデータを取得し，結果をまとめて返します。

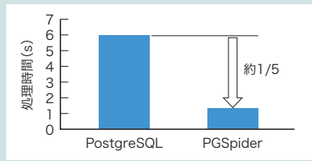

図3．検索・計算性能の評価結果

合計80万レコードのデータの平均値を取得するSQL文の実行に要する処理時間を，比較しました。

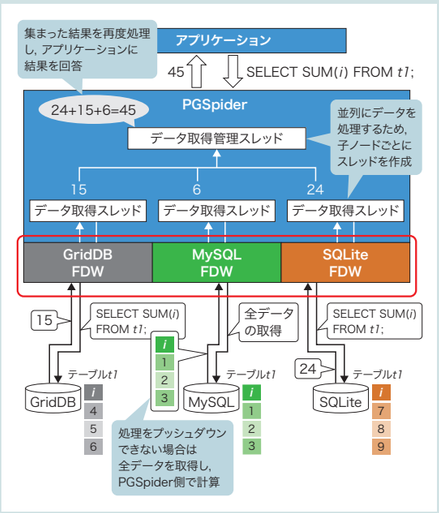

図2．PGSpiderでの合計値取得要求（SUM関数）の例

PGSpiderは，"データの並列処理" と "処理の同時プッシュダウン" で，高速化を実現しました。

大規模分散データの高速検索フレームワーク PGSpider

PGSpiderは，分散配置された複数のデータソースをそれぞれ子ノードとして接続し，データを横断的に検索するためのフレームワークです。更に，子ノードにPGSpiderを接続して中間ノード化することで，木構造の広域かつ大規模な分散環境を構築し，データ検索の高速化を実現しました。

東芝は，OSS（オープンソースソフトウェア）のRDBMS（リレーショナルデータベース管理システム）であるPostgreSQLをベースに，PGSpiderを開発しました。

仮想的なテーブルを実現するマルチテナント機能

PGSpiderに接続された子ノードのテーブル名とPGSpider内の仮想テーブル名を自動的にマッピングすることで，分散されたテーブルを一つの仮想的なテーブルとして扱うマルチテナント機能を開発しました（**図1**）。PostgreSQLでも，テーブルの関係性をマッピングすれば同様のことは可能ですが，後述するような，高速化のための手法が適用できないという欠点があります。

また，PGSpiderで木構造を構築した場合，中間ノードのPGSpiderにあるテーブルを上位のPGSpiderでマッピングすることで，複数のPGSpiderのテーブルを一つのテーブルとして扱うことができます。

異なるデータソースに対するアクセス機構

PostgreSQLには，SQL2003規格に規定されているSQL/MED（Management of External Data）に，ほかのデータソースへアクセスする仕組みとしてFDW（Foreign Data Wrapper）機能があります。この機能により，様々なデータソースへのアクセスが可能になります。現在では，OracleやSQL Serverなどの商業RDBMS，SQLiteやMySQLなどのOSSのRDBMS，及びApache CassandraやGridDBなどのNoSQL（Not Only SQL）データベースといったデータソースへアクセスするためのFDWがユーザーによって作成・公開されており，これらをPGSpiderのデータソースとして使用可能です。

データ検索を高速化するための手法

大規模データが広域に分散格納されたケースでデータ検索を高速化するために，二つの手法を実現しました。

一つはデータの並列処理による高速化です。複数のデータソースから高速にデータを取得するには，データを並列に取得して処理する必要があります。しかし，PostgreSQLでは，FDWを利用した場合にデータの並列処理ができません。PGSpiderでは，子ノードの数だけスレッドを作成することで並列処理を実現し，FDWからのデータ取得を高速化しました（**図2**）。

もう一つは，処理を可能な限り子ノードへ移譲するプッシュダウンによる高速化です。例えば，WHERE句のようなフィルタリング関数やSUM関数などの集計関数を，子ノードへプッシュダウンすれば，親ノードでのデータ処理を最小限に抑えることができ，データの転送と処理の両方で効率的となります。処理のプッシュダウン機能自体はPostgreSQLに存在しますが，複数のデータソースに対する同時プッシュダウンには対応していません。そこで，プッシュダウンできないデータソースはPGSpider側で処理し，更に集まった結果を再度処理することで，同時プッシュダウンを実現しました。

性能評価

既存のPostgreSQLと，PGSpiderの性能比較を実施しました。MySQLとTinyBrace（当社が開発したRDBMS）の2種類のデータソースに対し，八つに分散させたテーブルから横断的に検索を行いました。PGSpiderは，中間ノードのPGSpiderが四つで，各テーブルに10万レコードのデータが保存されています。PostgreSQLは，中間ノードを持たず，八つのデータソースを直接持っています。

PGSpiderの性能は，テーブル*t1*のカラム*i*の合計値を計算する，"SELECT SUM(*i*) FROM *t1*;" というSQL文を実行して測定しました。ここで，PostgreSQLにはマルチテナント機能がないため，このSQL文は使用できません。そこで，集合演算UNION ALLで全ての結果を統合し，その上でSUM関数を実行することで，性能を測定しました。

この結果，PGSpiderは，既存のPostgreSQLに比べ処理時間が約1/5になりました（**図3**）。ノード数，レコード数，ネットワーク速度やそのほかの環境で処理時間は変化しますが，大規模になるほどその差が大きくなります。

今後の展望

テーブルとテーブルを結合する演算のJOINについて，プッシュダウンを検討しています。2017年には，PostgreSQLにJOINのプッシュダウン機能が追加されましたが，PGSpiderでは更に複数のテーブルにまたがるJOINの設計を構築する必要があります。

また，現在の構成では，ある中間ノードが何らかの原因で消失した場合，その中間ノードに接続する子ノードからデータが取得できなくなります。消失発生時には，子ノードを自動的に別の中間ノードに接続してデータ取得ができるといった，高い可用性を持たせる仕組みの開発にも取り組みます。

このように，PGSpiderを更に高速化させ，可用性を高めることで，適用範囲の拡大を目指します。

廣瀬　繁雄
研究開発本部 ソフトウェア技術センター
オープンソース技術部