5B-05

PostgreSQL の NoSQL データベース GridDB との連携

片山 大河† 廣瀬 繁雄† 望月 翔平† 金松 基孝†
株式会社 東芝 ソフトウェア技術センター†

1. はじめに

PostgreSQL[1]は外部データベース（DB）にアクセス可能にする仕組み Foreign Data Wrapper（FDW）がある。この機能は DB のマイグレーションや複数の異種 DB を併用するシステムなどで利用され、近年その機能強化が盛んに行われている。ユーザは、この仕組みに基づいて各 DB 向けに実装されたモジュールをインストールすることで、任意の DB にアクセス可能になる。

この度、この機能を用いて PostgreSQL からインメモリ DB GridDB[2]にアクセス可能にした。GridDB は IoT データ向けの NoSQL 型インメモリ DB で、その特徴的な仕様や利用方法に従ってモジュールを実装する必要がある。本稿ではその実現方法について説明する。

2. FDW 機能とは

FDW 機能は DB の違いを吸収する仕組みで、PostgreSQL から外部 DB へのアクセスを可能にする機能である。複数の外部テーブルに跨がる問い合わせも PostgreSQL エンジンが個々の外部 DB 向けの問い合わせに分解する。I/F と動作仕様があり、それに基づいて開発者が実装した各 DB 向け FDW を介して外部 DB にアクセスする（図 1）。FDW の I/F は主にプラン構築、データ参照処理、データ更新処理の 3 つに分けられる。

2.1. プラン構築処理

外部 DB へのプッシュダウンの可不可判定、問い合わせ内容の構築、コスト計算などを行う。プッシュダウンとは、WHERE 条件、集約関数、ジョインなどの処理を外部 DB 上で実行させることである。プッシュダウンできないと判定した場合、処理はエンジン上で実行される。今回の GridDB-FDW は WHERE 条件のみプッシュダウンできるようにした。

2.2. データ参照処理

外部 DB で参照クエリを実行して、結果をエンジンに渡す。FDW-API の呼び出しの流れは、まず BeginForeignScan が呼ばれる。次に IterateForeignScan が呼ばれ、ここで外部 DB からフェッチした結果セットから 1 行分をエンジンに渡す。NULL を返すまで繰り返され、最後に EndForeignScan が呼ばれる。

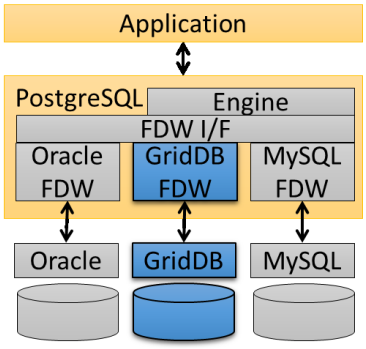

図1 FDW と外部データベース

2.3. データ更新処理

FDW-API の呼び出しの流れは、開始時に BeginForeignModify、終了時に EndForeignModify が呼ばれる。その間で、INSERT の場合 ExecForeignInsert が呼ばれる。UPDATE あるいは DELETE の場合、まず、対象データを参照するために 2.2 節で説明した処理が実行される。つまり対象となり得る行を一旦外部 DB からフェッチし、エンジンが対象かどうかを判断する。その後、データ更新を実行する。これには 2 通りの実行形態がある。外部 DB に対して複数行を一括して更新処理を実行する方法と、更新対象の結果を参照してから 1 件ずつ更新命令を実行する方法である。GridDB には一括更新方法が提供されていないため、後者の手段を用いる。この場合、繰り返し ExecForeignUpdate あるいは ExecForeignDelete が呼ばれる。

3. GridDB-FDW

GridDB はリレーショナル DB とは異なる特徴を持つため、それに応じて GridDB-FDW を設計する必要がある。この章では GridDB の特徴に基づいて GridDB-FDW の参照処理と更新処理の実現方法について説明する。

3.1. GridDB とは

GridDB は NoSQL 型のインメモリ DB で、スケーラビリティを備えた分散 DB である。キー・バリュー型を拡張したデータモデルを採用している。レコードの集合体はコンテナと呼ばれ、ユーザは任意のデータ型を複数個持つコンテナを定義できる。このコンテナをテーブルとみなし GridDB-FDW を実装した。

Accessing to NoSQL Database GridDB from PostgreSQL
†Taiga Katayama, Shigeo Hirose, Shohei Mochizuki, Mototaka Kanematsu
†Toshiba Corporation, Corporate Software Engineering & Technology Center

GridDB には 4 種類の製品があり、今回はオープンソースとして公開されている Community Edition を対象とした。

3.2. GridDB-FDW のデータ参照処理

GridDB へのデータ参照の命令は TQL を使用する。TQL は簡易版 SQL で GridDB 専用の問い合わせ言語である。特定の列を選択することはできず、全列選択となる。WHERE 条件は使用可能である。GridDB-FDW を次のように実装した。

プランの生成では全列選択する TQL 文を構築し、BeginForeignScan で TQL を実行する。更に、結果セットを GridDB-FDW 内にフェッチする。その後、繰り返し呼ばれる IterateForeignScan では、その結果セットから 1 行ずつ取得し、必要な列のみをエンジンに返す（図 2）。最後に EndForeignScan で結果セットを解放する。

3.3. GridDB-FDW のデータ更新処理

GridDB に対するデータ更新は API で行う。INSERT の場合は、追加する行データを作成してから追加 API を実行する。UPDATE と DELETE の場合は、まず SELECT で結果セットを取得する。結果セットはカーソルで管理されていて、それが指す行を更新あるいは削除する API がある。UPDATE する場合、結果セット内の値を変更しておく（図 2）。この仕様に基づいて、更新処理に必要なデータの構築方法を説明する。

INSERT の場合、エンジンから渡される情報は追加する行データをすべて含んでいるため特に問題なく実装できる。

UPDATE の場合、更新列に対する更新後の情報のみが与えられる。2.3節で説明したとおり、参照処理と同様の処理を実行するため、結果セットは構築できている。ここで課題が 2 つある。

1 つ目は、結果セット情報を更新処理

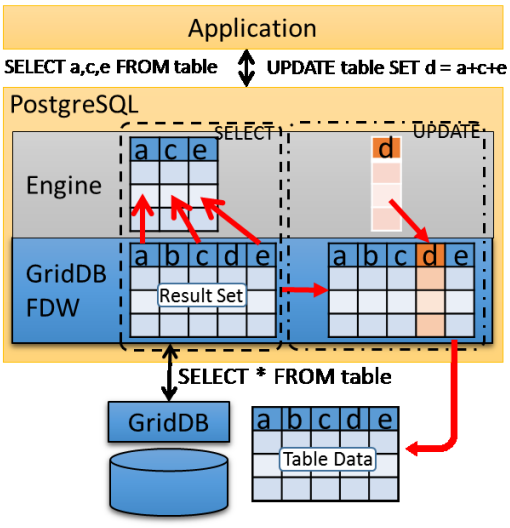

図2 参照と更新のデータの流れ

ExecForeignUpdate に渡す方法である。FDW にはそのような機能が存在しない。解決手段として、データの受け渡し用にグローバル変数を用意した。参照処理と更新処理の間に他のテーブルに対する参照処理が割り込むことはないため、この実装で問題ないと考えた。

2 つ目の課題は更新行の特定方法である。FDW 機能には、いずれかの列を行識別子として設定する機能がある。しかし、この機能は PRIMARY KEY 制約を持たないテーブルには使用できない。そこで、この問題は FDW-API の実行順序の特徴を利用して解決した。FDW-API は次の順序で実行される。

1. BeginForeignScan
2. BeginForeignModify
3. IterateForeignScan
4. ExecForeignUpdate または ExecForeignDelete
5. 3 と 4 の繰り返し
6. EndForeignModify
7. EndForeignScan

3.2節で説明したように、BeginForeignScan で結果セットを構築する。以降、1 行ずつ行データをエンジンに返す処理と 1 行を更新する処理が交互に実行される。この特徴を利用して、結果セットのカーソルが指し示す位置が更新対象行とみなし更新処理を実装した。

4. GridDB-FDW のトランザクション管理

GridDB はコンテナ単位でトランザクションが管理されていて、初めてそのコンテナにアクセスした際に暗黙的にトランザクションが開始される。TQL や更新 API の実行時に対象のコンテナのハンドルを覚えておき、コミット時に参照できるようにした。

5. 動作確認

PostgreSQL には SQL の実行をベースとしたテストフレームワークがある。PostgreSQL をソースコードからビルドする際に、make check を実行するとテストプログラムを実行できる。この仕組みを利用して動作確認した。

6. おわりに

本稿では GridDB-FDW の仕組みを説明した。これにより動作仕様が特殊な DB である GridDB に PostgreSQL からアクセスできるようになった。FDW 機能は盛んに強化されている。それに追従するように、GridDB-FDW も対応を進めていく。ソースコードは[3]にて公開予定である。

参考文献

[1] PostgreSQL, https://www.postgresql.org/
[2] GridDB, https://griddb.net/
[3] GridDB-FDW, https://github.com/t-kataym/griddb_fdw/