Global Meta Cache で定義情報を共有化！ PostgreSQL の大規模システムのリソース削減

－富士通の技術者に聞く！PostgreSQL の技術－

オープンソースソフトウェアの普及が進むなか、従来は難しかったミッションクリティカル領域への導入も見られるようになってきました。
代表的なオープンソースソフトウェアデータベースである PostgreSQL も、ミッションクリティカル領域へ導入したいという要望が増えています。富士通では「ミッションクリティカル領域での運用」にも対応できるよう、PostgreSQL をベースに機能を拡張したデータベース「FUJITSU Software Enterprise Postgres（以降、Enterprise Postgres と略す）」を提供しています。2020 年 7 月には、更なる機能拡張を実現した新バージョン Enterprise Postgres 12 をリリースしました。
本特集では、Enterprise Postgres 12 に搭載された数ある新機能の中から、特に大規模データベースの運用時に課題となるメモリーの使用量を削減することで高性能を実現する「Global Meta Cache（グローバルメタキャッシュ）」機能にフォーカスをあて、開発の経緯や機能への取り組み、コミュニティーへの貢献などについて、本機能の開発担当である「冨山」とお客様対応担当である「山地」が語ります。

冨山 慶一 Yoshihide Tomiyama
富士通株式会社　ソフトウェア事業本部 データマネジメント事業部
専門分野：データベース
入社以来、SOA に基づく ESB 製品、データ連携製品、DBaaS の開発に携わり、クラウド連携、高速転送、データベース運用管理基盤を担当。現在は、PostgreSQL をベースとした「FUJITSU Software Enterprise Postgres」の開発を担当している。

山地 亮 Ryo Yamaji
富士通株式会社　ソフトウェア事業本部 データマネジメント事業部
専門分野：データベース
入社以来、PostgreSQL をベースとした「FUJITSU Software Enterprise PostgreSQL」の顧客対応を担当。システム設計の提案・課題解決に従事している。

ミッションクリティカル領域への適用拡大

ミッションクリティカル領域へ PostgreSQL が導入されるケースが増えているようですね。

冨山

そうですね。当社へご相談いただくお客様からも、ミッションクリティカルなシステムに PostgreSQL を導入したいという要望が増えてきています。

山地

データベースである PostgreSQL だけでなく、アプリケーションサーバーの GlassFish や統合監視ツールである Zabbix や Hinemos など、オープンソースソフトウェア全般的に導入が増えている印象です。これは、オープンソースソフトウェアが企業の基幹システムにまで採用されるだけの性能・信頼性が確保されてきていることの現れと、製品そのものの認知が広まってきた結果と考えています。

冨山

ただ、当社の得意とするミッションクリティカル領域においては、まだまだオープンソースソフトウェアでは機能が弱いと感じる箇所があるため、Enterprise Postgres のようなオープンソースソフトウェアの機能を拡張した製品を提供することで、それを補

っていこうと考えています。今年の 7 月には、旧バージョンから更なる機能拡張を実現した新バージョン Enterprise Postgres 12 をリリースしました。

「メタキャッシュ」機能におけるメモリー使用量の課題

今回のインタビューでは、その Enterprise Postgres 12 に搭載された新機能の一つである「Global Meta Cache」についてお伺いしたいと思います。この新機能は PostgreSQL に搭載されている「メタキャッシュ」が基になっていると思います。そこで、まず「メタキャッシュ」について教えていただけますか？

冨山

PostgreSQL は、テーブルや列の情報といったスキーマメタデータが格納されている「システムカタログ情報」のキャッシュを、コネクション単位に生成されるバックエンドプロセスごとにローカルメモリー上へ展開します。この、ローカルメモリーに展開されたシステムカタログ情報のキャッシュを「メタキャッシュ」と呼びます。

そこに改良の余地があったということは「メタキャッシュ」に課題があったのでしょうか？

冨山

そうですね。まず、メタキャッシュはスキーマメタデータが格納されているという性質上、アクセスするデータベースのテーブル数およびカラム数に比例してサイズが大きくなります。サイズが大きくなるということは、それを格納しておくのに必要なメモリー量が増えます。さらに、メタキャッシュはバックエンドプロセスごとに展開されるため、プロセス数に比例して数が増えることになります。メタキャッシュの数が増えれば、格納しておくために必要なメモリー量がさらに増えます。
このため、大規模データベースや大量のコネクションを必要とするデータベース環境では、メタキャッシュが使用するメモリー量が著しく増加してしまうという課題がありました。

山地

この「Global Meta Cache」機能の開発は、当社へ寄せられるご相談に『従来のデータベースを PostgreSQL にマイグレーションしたい』という話が増えてきたことから始まります。一口にマイグレーションと言っても多様なサイズのデータベースがあります。大規模データベースのご相談もあったのですが、資源見積もりの結果、サーバーに必要なメモリーがテラバイトオーダーになってしまうようなパターンもありました。

テラバイトのメモリーですか？それはマイグレーションコストに影響が出るのではないでしょうか？

山地

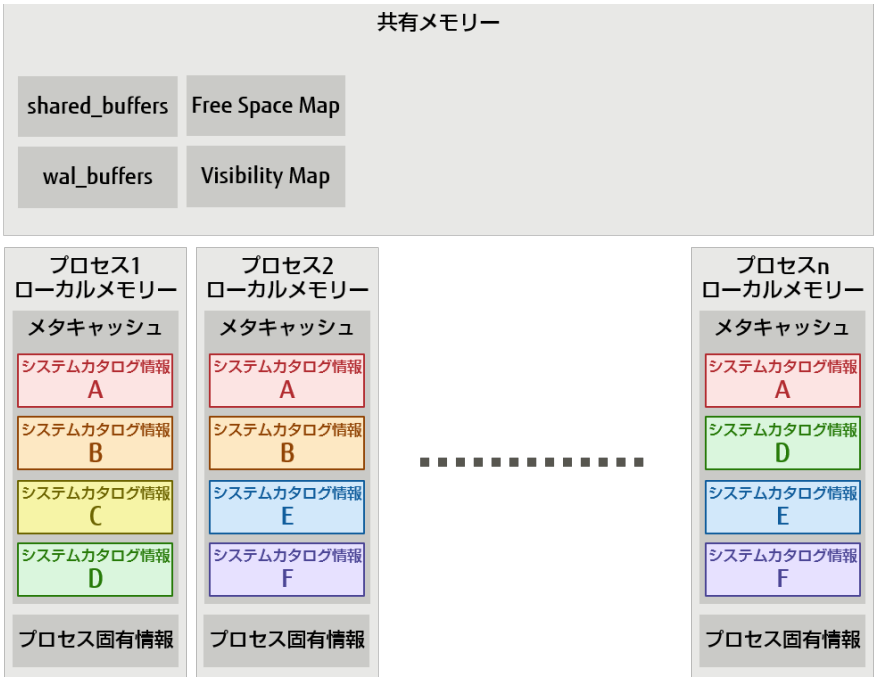
もちろん、その要求に応えられるサーバーを用意するとなると非常にコストが増えます。当社としてもこれではお客様に提案できないと考え、PostgreSQL の根本原因を解決することにしました。

メモリー使用量を削減する秘策

それがメタキャッシュで使用するメモリー量を削減するということなのですね。具体的にはどのように解決したのでしょうか？

冨山

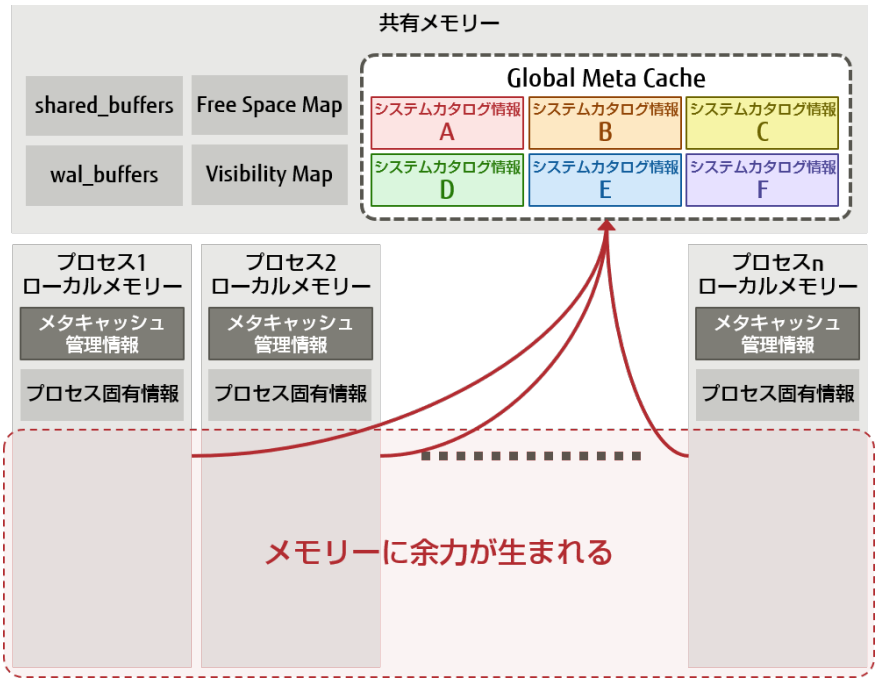
先ほど説明したとおり、PostgreSQL はコネクション単位に生成されるバックエンドプロセスごとにメタキャッシュを展開します。例えばテーブル A を参照するために必要なシステムカタログ情報のキャッシュを、各バックエンドプロセスがそれぞれのローカルメモリーに展開しているのです。つまり、同じシステムカタログ情報を含むメタキャッシュが、バックエンドプロセスの数だけ複製されているという無駄が生じています。そこで、メタキャッシュをひとまとめにして共有化し、各バックエンドプロセスから参照する形にすれば、メモリー使用量の削減に繋がるのでは？と考えました。



システムカタログ情報を共有するという発想ですね。もう少し詳しく教えてください。

冨山

まず、共有メモリー上に「Global Meta Cache」と呼ぶ専用の領域を作成します。そして、システムカタログ情報のキャッシュは各バックエンドプロセスのローカルメモリーには展開せず、この Global Meta Cache に展開します。各バックエンドプロセスのローカルメモリーには、共有メモリーの Global Meta Cache を参照するために必要な情報をまとめた「メタキャッシュ管理情報」と、「プロセス固有情報」のみを展開します。これにより、PostgreSQL のインスタンス全体でシステムカタログ情報が共有できるので、各バックエンドプロセスがそれぞれのローカルメモリーに展開する必要がなくなります。しかも、この方法であればプロセス数が多ければ多いほど、メモリー使用量の削減が期待できます。



なるほど！ただ、共有するとなると性能の低下などが気になりますが大丈夫でしょうか？

冨山

各バックエンドプロセスは一度アクセスした共有メモリーの Global Meta Cache に対して、二度目以降は共有メモリーのハッシュ表を検索することなくアクセスできます。このように、再アクセスの効率化を行うことで大幅な性能低下を防いでいます。

山地

開発段階における性能ベンチマークテストによると、Global Meta Cache 機能を使用することで性能は 5%程度低下しました。しかし、サーバーのメモリー使用量が減ればメモリーに余力が生まれるため、必要な処理にメモリーを多く割り当てることで、性能が低下した分を上回る性能向上が期待できるわけです。

データベースとして避けてとおれない課題の解消

それならば安心できますね。この機能を開発するにあたって、苦労した点はなんでしょうか？

冨山

システムカタログ情報のキャッシュが各バックエンドプロセスのローカルメモリーに展開されている状態であれば問題は無かったのですが、共有メモリーに作成した Global Meta Cache に展開すると、可視性の競合問題が発生します。この問題を解決することが非常に苦労しました。

「可視性の競合問題」というのは具体的にはどういうことですか？

冨山

分かりやすく言うと、プロセス 1 が更新したコミット前のキャッシュを、プロセス 2 が参照してしまうことでデータの整合性が取れなくなるという問題です。

それはデータベースの ACID 特性に関わりますよね？？

山地

はい、そこはデータベースとしては決して避けて通れない問題です。

冨山

これについては開発チームでも時間をかけて議論を重ね実装しました。具体的には、トランザクション実行中に定義情報の削除や作成が行われた場合、それに伴うキャッシュの削除や作成は該当プロセスに閉じて行われます。そのトランザクションがコミットして初めてキャッシュが削除されます。もし、キャッシュを削除する時に、他のプロセスがそのキャッシュを使用している場合は、キャッシュを参照しているプロセスの数がゼロになった時に削除されます。加えて、この古いキャッシュを他のプロセスが新規に参照することは許可していません。これにより、データの整合性が失われることはありません。

- 用語解説　「ACID 特性」とは、データベースのトランザクション処理に求められる 4 つの特性「Atomicity（原子性）」「Consistency（一貫性）」「Isolation（独立性）」「Durability（耐久性）」の頭文字をつなぎ合わせたもので、データの一貫性を保証するために必要不可欠な考え方。

コミュニティーとの関わりと将来

素晴らしいですね。新しい機能の開発にはコミュニティーとの連携も必要だと思いますが、どのように活動されてきたのですか？

山地

この Global Meta Cache 機能は、すでにコミュニティーに提案済みです。当社では次期 PostgreSQL に搭載するべく、さらにコミュニティーへのフィードバック反映を行う予定です。

将来的に PostgreSQL に正式採用されるのが楽しみですね。とても重要な機能改善だと思いますし、ミッションクリティカル領域への適用がますます進みますね。

冨山

そうですね。いままで実現が難しかった大規模データベースを PostgreSQL に移行しやすくなるので私たちも期待しています。

山地

同じ性能を得るにも従来に比べてメモリーが少なくて済みますので、サーバーのダウンサイジングも可能です。このため初期導入コストを抑えられるというメリットもありますので、お客様にとっても導入への敷居が下がるのではないかと思います。

良いことづくめですね！最後に、さらに改善したい点など、今後の取り組みについて教えてください。

冨山

さらなる機能改善を進めたいと考えています。具体的には、各バックエンドプロセスのローカルメモリーに展開している Global Meta Cache を参照するために必要な「メタキャッシュ管理情報」を制御できるようにします。
一口にキャッシュといっても、オンライン業務などアクセス頻度が高く性能が求められるものと、日次や月次のバッチ業務などアクセス頻度の低いものが混在しています。これらを綿密に管理し、アクセス頻度の低い情報をローカルメモリーから追い出すことで、よりメモリー使用量を削減して大量のコネクションを処理するためのプロセス数を増やせるようにするといった改善です。

現状に満足せず更なる高みを目指すということですね。期待しています。本日はありがとうございました。

2020 年 9 月 25 日