Engineer

キャッシュ

元データのコピーを読み手の近くに置く保存領域

どのようなものか

キャッシュは、データを読む側の近くにコピーを保持する保存領域です。同じデータを再び読むとき、読み取りに時間のかかる本来の保存先ではなく、そのコピーから答えます。英語の複数形はcachesで、保存した一項目をキャッシュエントリー、その中のデータをキャッシュ済みのコピーと呼びます。

キャッシュが正本になることはありません。コピー元のファイル、行、サーバーの応答が引き続き正本であり、各エントリーは捨てて取得し直せます。ここが、複製によって意図的に全体をそろえ、同期させるレプリカや、まだ他のどこにも存在しないかもしれない転送中のデータを保持するバッファーとの違いです。Linuxのページキャッシュは、キャッシュとバッファーの両方の役割を担います。読み取ったファイルのデータを保持するとともに、書き込みを先に受け取り、ディスクに届くまで、そのページをdirty（ディスクへの書き込みが済んでいない状態）として扱います。

利用者とデータの間には、キャッシュが何層もあります。プログラムが読むすべてのファイルの下には、カーネルのページキャッシュがあります。その上で、PostgreSQLは独自のページのプール、`shared_buffers`を保持します。ブラウザーは、一人向けのプライベートキャッシュにHTTP応答を保存し、CDNやリバースプロキシは、多数の人向けの共有キャッシュを保持します。ページのスクリプトは、ブラウザーのHTTPキャッシュとは別に、サービスワーカーを通してCache Storageへ応答を保存できます。また、アプリケーションは、計算した値をRedisなどのキーバリューストアに保持できます。

どの層でも、用語は同じです。読み取りでエントリーを見つければヒット、見つからなければミスで、オリジンへ取りに行きます。読み取りのうちヒットした割合が、ヒット率です。各エントリーはキーで検索し、HTTPキャッシュでは、少なくとも要求メソッドとURLがキーに含まれます。保存領域がいっぱいになる

と、空きを作るためにエントリーを取り除きます。これを追い出しと呼び、何を取り除くかは、追い出しポリシーが決めます。一般的なのは、最後に使われた時刻が最も古いものを選ぶLRU（least recently used）です。

概念を判断に生かす

「キャッシュとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 年々重くなってきたシステムのクエリと不足するインデックスを調べ、効果のある箇所を改善します。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 30日分の電力データを180 msで処理 — 180 ms 1か月分のメーター値を集計

30日分の電力データを180 msで処理 お問い合わせ →

なぜ重要なのか

キャッシュは通常、システムを速くする最も安い方法で、新しいハードウェアや新しいスキーマより費用を抑えられます。その代償は、メモリー、手の届きにくい場所にあるコピー、元の内容と一致しなくなったコピーという形で現れます。保存領域についての三つの判断で、大半が決まります。

一つ目は、置く場所です。コピーが読む人に近いほど、節約できるものは増えます。ブラウザー内のコピーなら、サーバーだけでなく、ネットワークも使わずに済みます。一方、管理する側からは遠くなります。Redisのインスタンスは一つのコマンドで空にできますが、訪問者のスマートフォンにあるコピーは、一緒に送ったルールに従って、その場所に残ります。

二つ目は、容量です。キャッシュに渡したメモリーは、他の用途から取り上げたものだからです。PostgreSQLの文書によると、`shared_buffers` の既定値は通常128 MBです。メモリーが1 GB以上ある専用サーバーでは、25%から始めることを勧めています。また、PostgreSQLはOSのキャッシュにも依存するため、40%を超えて割り当てても、それより小さい割り当てよりよく動く可能性は低いとしています。二つのキャッシュが、一台のマシンを共有しています。

三つ目は、その保存領域を誰が共有するかです。RFC 9111は、一人の利用者向けのプライベートキャッシュと、保存した応答を複数の利用者に再利用する共有キャッシュを区別します。個人向けの応答を共有領域に置くと、一人のページが他の人へ届きます。そのため、`Authorization` ヘッダーを持つ要求への応答は、応答側が明示的に許可していない限り、共有キャッシュで再利用してはいけません。

三つの判断から、一つのルールが導かれます。キャッシュは、安全に空にできる必要があります。空にしてデータを失うなら、データベースの保証を持たないデータベースになっています。いつまでエントリーを応答に使ってよいか、何が期限前に取り除くかは、実践としてのキャッシュ処理の領域です。保存領域自体の仕事は、コピーを保持し、キーで見つけ、容量が足りなくなったら空きを作ることです。

実際の活用例

自前のフォージとウェブサーバーを動かす464 MBのホストでは、カーネルのページキャッシュが、警報で何を測るべきかを決めます。信号の途絶で警報を出すデッドマンズスイッチが、実行時刻にランダムなずれを加えながら、十五分ごとに動きます。スクリプトは、利用可能なメモリーとディスクの空き容量を測り、両方がしきい値を上回る間だけ、外部サービスへ信号を送ります。正常なLinuxでは、カーネルが未使用メモリーをキャッシュに使い、空きメモリーがゼロに近くなるため、測るべきなのは「利用可能」な量です。スクリプトは正しい列を読んでいましたが、周囲の六つの変数名は「free」となっていました。10パーセントのしきい値に対して、その名前を信じた人が464 MB中の十二メガバイトの空きを見れば、検査が壊れていると考え、一文字を変えてしまうかもしれません。そのように変えると、常にしきい値に違反する警報になり、一週間以内にミュートされることになります。変数名は、その後修正されています。

一層上には、データベースの前段に置くアプリケーションキャッシュがあります。シャーディングの前にデータベースを最適化するノートでは、売り切れ時の処理の前段に置いています。商品の全在庫が確保されると、データベースは、最も早く確保が失効する時刻を答えられます。確保から10分後です。ノートでは、その時刻を期限として、「fully reserved」（全数確保済み）という状態をキャッシュします。それまでは、データベースではなくキャッシュが訪問者を断ります。カードの拒否や買い物かごの放棄で、確保が早く解放された場合は、エントリーを削除し、直ちにその一個を再び販売可能にします。キャッシュは拒否できますが、売るのは、条件付きの `UPDATE` を実行するデータベースだけです。古いエントリーのために数秒間、販売機会を失うことはあっても、同じ一個を二度売ることはありません。

最も外側の層は、訪問者自身の機器です。このサイトは、そこにスクリプトが管理する保存領域を作りません。Cookieのページでは、IndexedDBもCache Storageも使わず、サービスワーカーも登録しないと説明しています。ブラウザーは、サイトが管理するオフライン用コピーとは別の通常のHTTPキャッシュに、応答を保持することがあります。

問題が起きるところ

再起動後、すべての読み取りが一度にオリジンへ届きます。 空のキャッシュでは、すべてがミスになるため、デプロイ、キャッシュの再起動、フェイルオーバーによって、読み取りの全負荷がデータベースへ届きます。大半がヒットすることで持ちこたえていたシステムは、キャッシュが再び埋まる前に停止しかねません。オリジンをミス時の処理負荷に耐える規模にし、トラフィックを受ける前にキャッシュへデータを読み込み、段階的にトラフィックを戻します。PostgreSQLの `pg_prewarm` は、バッファーキャッシュに対してこれを行い、共有バッファーに何があったかを記録して、再起動後に読み戻すこともできます。

キャッシュが増え、マシンのメモリーを使い切ります。 64ビットのシステムでは、Redisは既定でメモリー上限を設けないため、キャッシュ用のインスタンスは、保持するキーとデータが増えるにつれて、OSが介入するまで大きくなります。上限とポリシーを一緒に設定します。 `noeviction` では、満杯のインスタンスは、新しいデータをキャッシュするコマンドにエラーを返します。すべてを保持する保存先には合いますが、忘れることを期待するキャッシュには意外な動作です。Redisの文書は、別のものを選ぶ理由がなければ、 `allkeys-lru` を妥当な初期選択肢としています。

```
maxmemory 256mb
maxmemory-policy allkeys-lru
```

訪問者に、別の言語のページが届きます。 値が依存する入力を、キーから省いています。デンマーク語で描画したページをURLだけで保存すると、次の読者が何を求めても、デンマーク語が届きます。言語、通貨、ロールなど、出力を変えるすべての入力からキーを作ります。HTTPキャッシュでは、応答ヘッダーの `Vary` に、キーへ含める要求ヘッダーを指定します。RFC 9111は、そのヘッダーが一致するか、再検証した場合にだけ、保存済みの応答を再利用できるとしています。

インストールしたアプリが、古いコードで動き続けます。 Cache Storageのエントリーには、独自の有効期限がなく、サービスワーカーのスクリプトを更新しても残ります。同じキャッシュ名のままファイルを変更すると、古いコピーを配信し続けます。また、公開のたびにキャッシュを改名しても、古いキャッシュを削除しなければ、その都度、機器に一つずつ増えます。Service Workers仕様の勧めに従って、キャッシュ名にバージョンを入れ、新しいワーカーが引き継いだら、古いものをすべて削除します。

参考資料

- RFC 9111：HTTP Caching：プライベート・共有HTTPキャッシュ、キー、共有キャッシュが扱ってはいけない応答に関するIETF標準です。
- Linuxカーネル：メモリー管理の概念：ページキャッシュがファイルのデータをメモリーに保持する仕組みと、そのページを回収可能として数える理由を説明します。
- PostgreSQL：リソース消費：`shared_buffers`、通常の既定値、OSのキャッシュも考慮した容量の決め方を説明します。
- Service Workers：Cache Storageと、ブラウザーのHTTPキャッシュとの違いを定めるW3C仕様です。
- Redis：キーの追い出し：`maxmemory`と、各追い出しポリシーでキャッシュが満杯になった際の動作を説明します。

用語集

実際の制約に合わせて選ぶ

代表的なデータセット、そこから答えを得たい問い、現在のボトルネックをお知らせください。品質、処理量、復旧の測り方を合意できます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「キャッシュとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/cache/