NoSQL データベースをブラウザ上で管理するシステムの検討

岩崎　京†　　井上　潮‡

† 東京電機大学大学院 工学研究科 〒120-8551　東京都足立区千住旭町 5 番

E-mail:　†13kmc03@ms.dendai.ac.jp,　‡inoue@c.dendai.ac.jp

あらまし　SNS（twitter や Facebook 等）や検索サイト（google や yahoo 等）といった大量の情報を扱うサービスが急速に増えてきている。従来は情報を管理するのに RDBMS が使われてきた。しかしながら膨大なデータを格納及び管理の手段として RDBMS は必ずしも向いていない。そこで高速性にすぐれ、分散化の容易な NoSQL データベースが開発されている。しかしながら、一口に NoSQL データベースと言っても、それぞれのデータ構造や操作言語が全く違うため、データベースの移行や移行後の管理が容易ではない。本研究では、ユーザインタフェースを抽象化することにより、複数の NoSQL データベースを同じインタフェースで管理できるシステムについて検討を行う。

キーワード　NoSQL データベース　管理ツール　DB システム開発

1. はじめに

従来、サービスを運用する際に必要な情報は MySQL 等の RDBMS に格納をし、使用するのが通常であった。しかしながら、インターネットの普及により SNS（twitter や Facebook 等）や検索サービス（google や yahoo 等）といった多くのユーザが利用するサービスが増えてきた。ユーザの増加に伴い、サービス側で管理しなければならない情報も急速に増え続けている。いわゆるビックデータを従来通りの RDBMS で運用をすると情報の処理速度に時間が掛かり、結果としてユーザ側に不満が溜まってしまう形になってしまう。

そこで近年注目されているのが NoSQL データベースというジャンルに分類されているデータベースである。NoSQL データベースとは RDBMS 以外のデータベースを指す総称であり、それぞれのデータベースが要点特化型のデータベースとなっている。スケーラブルな構造であったり処理速度重視の構造であったりするものがある。こういった RDBMS とは違う利便性から NoSQL データベースが注目されてきている。実際、Google や Amazon といったサービスは BigTable[1]や DynamoDB[2]という独自の NoSQL データベースを作成し公開している。

NoSQL データベースはスケーラブルや処理速度という利点はあるが、実際に運用をする際は RDBMS と併用して利用をおこなったり他の NoSQL データベースと連携させたりすることが多い。そして、NoSQL データベースと言っても、それぞれのデータ構造が違うため操作言語が全く違う。そのことにより、データベースの移行や移行後に各データベースに対しての学習が強いられる。

そこで本研究では、NoSQL データベースの操作を簡易化するためにブラウザ上で管理を行うシステムの検討をする。インタフェースを抽象化することにより NoSQL データベース全般に扱えるようにし、操作の統一化を図る。そうすることにより利用者の学習を軽減させることができると考えられる。更に複数の NoSQL データベースに対して運用できるようにするので、NoSQL データベース間の連携も考慮に置いて開発を行っていく。

2. 既存の研究及び管理システム

NoSQL データベースに関する既存研究としては、菱沼らの研究[3]といった NoSQL データベースの機能を拡張することや、処理速度をいかに早くするという研究が主である。Wu-Chun らの研究[4]といったデータベースの操作の簡易化に関する研究もあるが少ない。そして NoSQL データベースの構造上の問題でもあるが一つのデータベースにのみ対応させているものが主となっている。

ブラウザ上でデータベースを管理するシステムはいくつかある。既存の管理システムは現在主流である RDBMS に関するものがほとんどであり NoSQL データベースに関するものはほとんどない。その中から知名度の高い MySQL 用の管理システムの phpMyAdmin[5]と NoSQL データベースに分類される MongoDB 用の phpMoAdmin[6]について述べる。

2.1. phpMyAdmin

MySQL サーバをウェブブラウザ上で管理するためのデータベース接続クライアントツールである。phpMyAdmin は PHP で実装されている。このツールは SQL 文を記述することなく、MySQL のデータベースに対して様々な操作が行えるような直感的な Web インタフェースとなっている。また、任意の SQL 文を記述して実行することもできる。機能制限はほとんどなく、MySQL に実装されている機能はほとんどサポートされている。

このツールの利点は MySQL のことを知らなくても視覚的な要素を参考にすれば管理運用をすることができる。従ってデータベース初心者もこのツールを使用すれば MySQL を使用することができるものとなっている。さらに表形式で出力することにより格納されているデータが見やすくなっている。

しかしながら、phpMyAdmin は MySQL のみに特化しているツールなので他のデータベースに対して使用することができない。つまり本研究のものとは違い 1 つのデータベースに足しての利用しやすさを追求したシステムである。

2.2 phpMoAdmin

PHP で実装がされている MongoDB を管理するデータベース接続クライアントツールである。MongoDB とはドキュメント指向型データベースで、データの形は JSON 形式に似ているものになっている。RDBMS とは違い固定的なスキーマを持たないデータベースとなっている。この MongoDB をウェブブラウザ上で管理することができるツールが phpMoAdmin である。格納されているデータが特定のスキーマを持たないので phpMyAdmin とは違い、データの閲覧時は整理されている形で表示されるのではなくのデータベースの中身をそのまま見ているような形になってしまっている。また、格納する際もあらゆる形に対応するためかテキストフィールドがひとつあるだけとなって、初めて使用する際は使い方に困る形になる。追加や編集はやりやすいとは言えないがデータの蓄積状況を確認する際は専用のクライアントで確認するよりは見やすいものとなっている。

しかしながら、こちらも phpMyAdmin と同じく MongoDB に特化しているクライアントツールのため他の NoSQL データベースには対応付けをすることが難しい。NoSQL データベース自体は運用するときに単体で使用することがほとんどないので、実際に管理をするときは他のクライアントツールを使用するなどと二度手間になってしまうことが考えられる。また、先ほど述べたようにデータを格納するときはある程度の知識がないと使用することが難しいツールとなっている。実際のクライアントツールを使用するよりは使いやすいが知識を有することや他のデータベースに運用することは難しいなどのデメリットはある。

3. 提案手法

前節で述べたが、既存のデータベース接続クライアントツールは 1 つのデータベースに特化しているものが主である。そして NoSQL データベースに対してのクライアントツールはまだ少ない。そこで本研究では、機能をフロントエンドとバックエンドに分けることによりデータベースの操作を抽象化し、複数の NoSQL データベースに運用することができるクライアントツールについて検討を行う。

NoSQL データベースは種類ごとに命令文が違う。そこで命令文の違いからおこる操作性の違いをブラウザ画面で操作をすることによりデータベースの操作統一を図る。そのためにはフロントエンド側では扱うデータベースによって作りは変えず、バックエンド側でデータベースごとに処理を行えるようにすることでユーザ側はどのデータベースに対しても同じように扱うことができると考えられる。

今回扱う NoSQL データベースは Key-Value Store（KVS）に分類されるものとする。KVS とは任意に保存したいデータ(value)に対し、対応する一位の標識(Key)を設定し保存するデータベースのことである。KVS に対応付けるためにフロントエンド側は KVS の形に対応をさせるものとする。また、データの形を合わせることでデータベース間の連携も考慮に入れてシステムの開発を行っていく。

提案手法のシステム概略図を図 1 に示す。

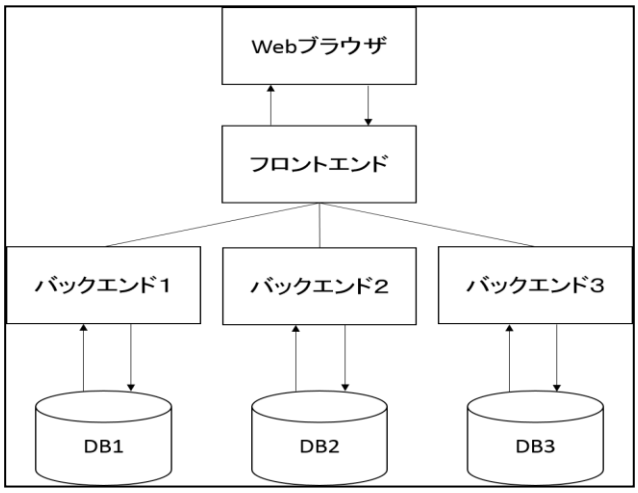

図 1．システム概略図

複数の NoSQL データベースにシステムを実装するために、はじめに使用するデータベースを選択する。選択したデータベースがどれなのかという情報はフロントエンド側で保持をし、その情報によってデータを送るバックエンド側のプログラムを決める。それぞれのバックエンド側は送られてきたデータを各データベースに対して情報を送れるようにデータを任意の形に変更を行い、データベースに対して命令を行う。またデータベース側から受け取った情報はバックエンドからデータの抽出を行い、フロントエンドに取得したデータを送り、フロントエンドで表示を行う。それによりフロントエンド側は表示の際、どのデータベースの種類に関係なくブラウザ上に同じ操作画面を出力することが可能となる。かつ格納時も統一したフォームで任意のデータを格納することが可能となる。

したがってバックエンド側のプログラムを使用する NoSQL データベース毎に作成をすることになるが、データベース操作時は命令文の違いを考慮することなくどのデータベースに対しても同じ方法で操作を行うことが可能となる。

4. 実験

4.1 実装したプロトタイプ

複数の NoSQL データベースに実装することができる新たなクライアントツールの提案を行うためにどの機能を付加するべきかを考察するためにプロトタイプの実装を行った。今回、プロトタイプの実装を行う際に使用したサーバ環境と実際に対応付けを行って使用した NoSQL データベースを表 1 に提示する。使用した NoSQL データベースは前述でも述べたとおり KVS に分類されているものとなっている。

表 1 サーバ環境

OS	Windows7 64bit
Web サーバ	Apache 2.2.21
言語	PHP 5.3.8
NoSQL DB	Memcached 2.2.6 Redis 2.6.12

プロトタイプには取得、格納の機能を持たせた。一連の機能の流れは以下に示す。

データ取得時

① ユーザに使用する NoSQL データベースを選択させる。
② 選択された NoSQL データベースに格納されているデータを取得するためにバックエンドでデータベースに命令を送る。
③ 取得してきたデータをフロントエンド側に送り、データを全件出力させる。
④ データを出力する際にデータ毎に削除、変更を行えるボタンを描画させる。

データ格納時

① ユーザに格納したいデータを key と value に分けて入力をさせる。
② 入力された key と value をバックエンドに送る。
③ バックエンド側で key と value をセットにしてそれぞれのデータベースに対応した形に変更を行いデータベースに送信をする。
④ 送信が完了次第、確認をするためにデータの表示画面に移行させる。

上の機能を複数のデータベースに実装するためにプロトタイプでは、システム起動時の画面で使用する NoSQL データベースの選択を行うように描画を行う（図２）。選ばれたデータベースの情報を保持したままこれ以降に動作させるバックエンド側のプログラムを決めるようにした。次に NoSQL データベースの選択が完了したあと選択をしたデータベースに格納されているデータの表示を行う画面に遷移するようにする。遷移をする際にバックエンドでデータベースに格納されているデータを取得してきてフロントエンドに送り表示をおこなう。その際、今回扱うのは KVS のデータベースなので Key と Value が一目でわかるように表示をさせるようにする（図３）。

次に、格納時は Key と Value をわけてフォームに入力してもらい１件ずつ格納する形とした（図４）。格納完了時は続けて入力フォームを表示するのではなく、確認をするためにデータ取得画面に移行を行う。入力した Key と Value はフロントエンドからバックエンドに送り、そこで各データベースに対応した形に変えてデータベースに送信を行い、格納をするようにした。

図 2.データベース選択画面

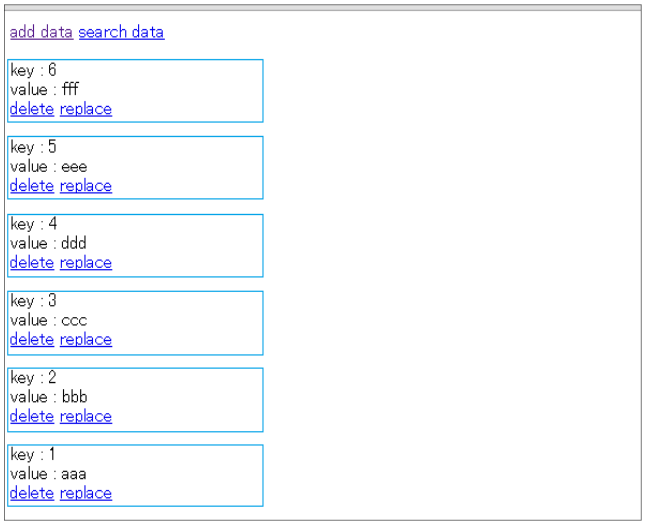

図 3.取得時

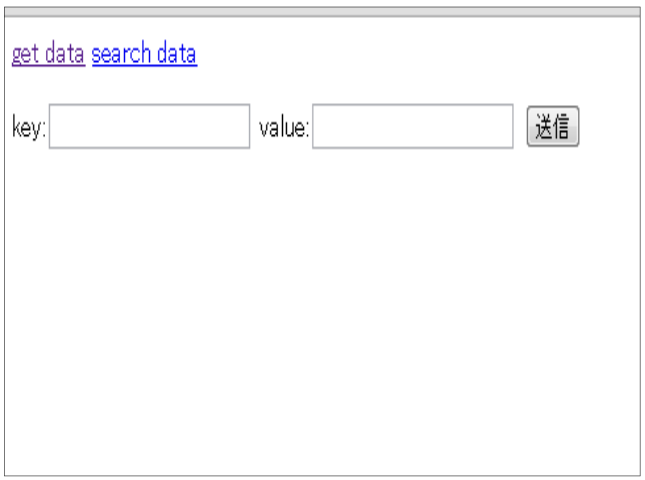

図 4.格納時

4.2 結果

実装したプロトタイプは、データの取得、格納に対しては問題なく行えるものとなった。実際のコマンドプロンプト等でデータベース内に格納されているものを確認するよりもデータベース内にある全ての情報を確認しやすいものとなった。データ取得時にかかる時間も遅くなることなく、データベースの性能をほとんど損なうことなく動作することができた。（表２）

表２　取得時間（秒）

	1 万件	10 万件
Memcached	1.8	9.5
Redis	0.7	6.5

また、格納する際も命令文を記述することなくできることによりデータベースを使うときには使いやすいものとなっていると考えられる。しかしながらデータを格納する際には 1 件ずつしか格納することができないのでビックデータを一度に入れることができない。データを確認する際には便利だが格納することに関してはまだ改良の余地があると考えられる。

5. おわりに

本論文では、NoSQL データベースを運用する際により効率よく作業できるためのシステムの検討を行った。NoSQL データベース同士の命令文の違いを解消するためにブラウザでデータベースの管理が行えるシステムを組んだ。命令文が異なる複数の NoSQL データベースに対して同じように操作をすることができるようにするため、システムを表示するフロントエンドとデータベースごとに処理を行うバックエンドに分けることによりインタフェースを統一し、操作の仕方を抽象化した。そうすることによりユーザ側は違うデータベースを扱っていても同じ操作でデータベースの運用をすることができるようにすることを目的とした。実際にプロトタイプを構築し 2 つの NoSQL データベース、memcached と Redis に対して実装を行った。この２つのデータベースは同じ KVS という分類にされているデータベースながら命令文は違うデータベストなっている。しかし今回作成したプロトタイプを

使用することにより操作の仕方は両者に対して同じように行うことができた。操作性に関しては、実際に提供されているコマンドプロンプトのようなクライアントよりデータの確認がしやすいものとなった。そして、このシステムを使用した上でのデータベースの処理速度は特に遅くなることはなかった。

今後の課題は KVS 以外の NoSQL データベースに対する実装である。KVS 以外の NoSQL データベースも注目をされてきているので、データの形が違う中でも操作性の統一をすることができるのかが必要になってくると思われる。

参考文献

[1] BigTable(Google App Engine)
https://developers.google.com/appengine

[2] DynamoDB
http://aws.amazon.com/jp/dynamodb/

[3] 菱沼直子,竹房 あつ子,中田 秀基,小口正人, Cassandra による KVS データ処理におけるデータ容量と処理性能に関する考察, deim2012, 2012-03-03

[4] Wu-Chun Chung, Hung-Pin Lin, Shih-Chang Chen Mon-Fong Jiang ,Yeh-Ching Chung , JackHare: a framework for SQL to NoSQL translation using MapReduce , Autom Softw Eng, 15 December 2012

[5] phpMyAdmin
http://www.phpmyadmin.net/home_page/

[6] phpmMoAdmin
http://phpmoadmin.com/