Shell Programming as a Glue Language and Technology

ISSN 2187-9664

USP MAGAZINE

よりぬき版

Vol.0

400yen

for the sophisticated shell scripters

第１特集
シェルスクリプトで NoSQL

第２特集
CUI 徹底活用仕事術！
Google カレンダーはもういらない？
シェルから Evernote

対談「小さな勇気で世界を変える」
アジャイル開発と UNIX の共通項とは

Artwork created by Koji Ishizuka. ©2011 Zie's Bank Co., LTD. All rights reserved.

発刊に寄せて

「UNIX 精神への原点回帰」「洗練された広範な技術」をテーマに
プログラムの楽しさを伝える雑誌を目指します

日々進化し続ける社会環境。
私たちエンジニアは、常に技術についての選択を迫られている存在と言えます。

1994 年、マイク・ガンカーズ（X Window System 開発チームの一員）は、
以下の 9 つの至上命令に集約される「UNIX 哲学」を創出しました。

1. 小さいものは美しい。
2. 各プログラムが一つのことをうまくやるようにせよ。
3. できる限り原型（プロトタイプ）を作れ。
4. 効率よりも移植しやすさを選べ。
5. 単純なテキストファイルにデータを格納せよ。
6. ソフトウェアの効率をきみの優位さとして利用せよ。
7. 効率と移植性を高めるためにシェルスクリプトを利用せよ。
8. 束縛するインターフェースは作るな。
9. 全てのプログラムはフィルタとして振る舞うようにせよ。

これらの UNIX 哲学は、複雑化した現代社会において、物事の本質を貫き、
シンプルに強く生きていくための指標として、多くの人々の支えとなっています。

シェル（shell）は、ユーザからの指示を受けて解釈し、OS の中核部分に伝達することで、
プログラムの起動や制御などを行うプログラムです。
あらゆるプログラムは、シェルからの起動が可能であり、シェルスクリプトに処理の順番を
記述することで、複数のプログラムを操ることができます。
シェルスクリプトは、平易で短く書けるためユーザとの親和性が高いのも特徴です。
様々な言語で書かれたプログラムを自由に操るシェルスクリプトの可能性は無限大といっても過言ではありません。

現代に生きる私たちが直面する多くの課題も、この強くて柔軟な「UNIX の精神」と
「シェルプログラミングの技術」によって解決できるのではないか、と私たちは考えます。

USP MAGAZINE は UNIX 精神の原点に立ち返り、
シェルプログラミングをはじめとする広範な「洗練された」技術を紹介することで、
エンジニアの皆さんに改めてテクノロジーの楽しさ、深さを伝える雑誌を目指します。

USP MAGAZINE 編集部

Contents

USP MAGAZINE 2011 spring

シェルスクリプトで NoSQL …… 3
CUI 徹底活用仕事塾！ ……
魁！サーバ自警塾！！ ……
ログファイルに日付を付けて管理する……
シェルスクリプト事始め……
RedStar2.0 をインストールしてみた。 …
対談「小さな勇気で世界を変える」……
原色 Linux 美女図鑑…… 11

※ 薄字で記載している記事は、よりぬき版には収録されておりません。

広告募集中

USP MAGAZINE は USP 友の会会員の定期購読、IT 系イベント・勉強会での販売を予定しております。

お問合せ先
mag@usptomonokai.jp

特集１.

そんな実装で大丈夫か!?

シェルスクリプトで NoSQL

Text & Illust : リッチ・ミカン@松浦リッチ研究所
http://richlab.org/

何でもできる SQL に満足できずに NoSQL

あなたはシステムエンジニア。お店、学校、病院といったお客を相手に、在庫、成績、カルテなど、これまで様々なデータを管理するシステムを開発してきました。そんなあなたはきっと、MySQL、PostgreSQL といった有名どころの RDBMS 型 (本記事では SQL 系と称します) のデータベースソフトウェアを用意し、SQL 文を駆使したプログラムを書いてきたことでしょう。

でもあれって結構大変じゃないですか？インストールとか起動とか設定とか…。自分のプログラムを作り始める前に一苦労。納品時、お客さんのところでまた一苦労。無事納品が完了しても忘れた頃に障害発生で呼び出され、そこでもデータベースをいじらねばならず、復旧させるのにまた一苦労。

SQL 系データベースは、世の中で必要とされるデータ処理をおよそ何でもできるように作られていると思います。データベースソフトウェアにおける一つの集大成です。ゆえに複雑な要求仕様にも応えられる局面が多く、選んでおいて間違いは無いでしょう。しかし扱うのがそれなりに大変なのです。何でもできるがゆえ、個々の目的に応じてどう動かせばよいかを覚えるのが大変です。またその通りに準備するのも大変です。冒頭のような経験をお持ちの方なら言うまでも無く実感していただけるでしょうし、そうでなくてもデータベース技術者の認定試験というものが存在するところから想像がつくと思います。一言で言うと、SQL 系データベースは「何でもできるかわりに大掛かり」に思えるのです。

一方で、最近 NoSQL と呼ばれるタイプのデータベースが注目されています。これは ``Not only SQL"(SQL だけじゃない) の略とされる、SQL 系とは全く異なるデータベース。とはいっても、NoSQL という名の標準規格があるわけではなく、SQL を使わないデータベースソフトウェアの総称です。近頃 NoSQL と言うと、SQL 系データベースが苦手とする領域の一つである「超高速データ処理」や「大量のリクエストを捌く」を実現するために開発されたデータベースを指す場合が多いのですが、本来の言葉の意味からすればそれに限ることはないはずです。つまり冒頭に示したような、SQL 系データベースが大掛かり過ぎると思えてしまう領域をカバーするデータベースがあれば、それも NoSQL でしょう。

シェルスクリプト NoSQL!?

実はシェルスクリプトがそんな NoSQL になると言ったら、想像できますか？**名付けて「シェルスクリプト NoSQL」**です。シェルスクリプトといえば UNIX 系 OS でバッチ処理を行うのによく使われる簡易な言語であるわけですが、一体それのどこにデータベース管理機能があるというのでしょうか。

実は SQL 臭かったコマンドたち

厳密に言うとシェルスクリプト自体がサポートしてるのではなく、その手足となって働くコマンド群を組み合わせることで実現可能になります。例えば、一般的な UNIX 系 OS(POSIX 準拠) が持っているコマンドで join というものがあります。これの man(マニュアル) を見てみてください。

```
〉man join↵
```

man の冒頭で、Linux なら「二つのファイルを読み、フィールドが共通な行を結合する」等と書いてあるでしょうし、FreeBSD に至っては「リレーショナルデータベースの操作」とハッキリ書いてあります。つまり、join コマンドというのは SQL 文で言うところの JOIN 句 (厳密には INNER JOIN) をこなすコマンドなのです。じゃあ WHERE 句 (行の絞り込み) は？…grep コマンド、あるいは awk コマンドでできます。それなら ORDER BY 句 (行の並べ替え) は？…ほら、sort コマンド使えばいいじゃないですか。しかもパイプで uniq コマンドに繋げば GROUP BY 句 (重複行を 1 行にまとめる) ができてしまいますよ。

こんなふうに、SQL 文で表現できる処理の多くが UNIX コマンドでもできるのです。双方の表現方法を対比してみると、表 1 のようにいろいろ当てはまることがわかります。

こうしてコマンドを見回すだけでも SQL 系データベースが持っている機能に相当するものがゴロゴロあります。しかしそれぞれ単独ではあまり役に立ちません。バッチ処理とパイプラインの仕組みを使ってこれらを纏め上げ、実用的な処理単位を作る存在こそがシェルスクリプトなのです。

表 1．各種 SQL 句の働きに相当する UNIX コマンド

やりたい事	SQL 文での表現	UNIX コマンドでの表現
内部結合	INNER JOIN	join, *join1*, *join2* 等
外部結合	＊ OUTER JOIN	*join2*, *loopj* 等
グループ化	GROUP BY, DISTINCT	sort | uniq
ソート	ORDER BY	sort
行絞り込み	WHERE, HAVING	grep, awk 等
列選択	SELECT 句の直後	awk, cut, *self*, *delf*
表の行結合	UNION	cat で複数ファイルを指定
表の排他制御	LOCK TABLE(S)	lockfile(Linux の場合) 等
行追加	INSERT, SELECT INTO	“>>”(追記リダイレクト)
行更新	UPDATE	sed 等の後に “>”
行削除	DELETE	grep -v 等の後に “>”
行数数え	COUNT 関数	wc, *gyo*
各種演算・変換	(各種演算子・SQL 関数)	sed, awk, tr, *comma* などなど

※ 右列で斜体表記のコマンドは、ユニケージコマンドという独自コマンドです。(後述)

置き換えだけじゃない

ここまでの紹介内容だと単に SQL 文の置き換えのように聞こえてしまうかもしれませんが、じつはそれだけではありません。SQL 文では記述しづらいデータ処理を簡単にこなせる場合もあります。

例えばとある博物館で、1 時間毎の入場者数を記録したテーブルがあったとします。今これを元に、開館時からの累計入場者数を求めたいという要望が来ました。さて、SQL 文ではどう記述しますか？難しいですよね。ところがシェルスクリプト NoSQL なら簡単です。入場者数が記録されているファイルを awk コマンドに掛け、awk コマンド内で一行ごとに入場者数を累計していく一行プログラムを書けばよいのです。

さらにもう一つの例としてこういうのはどうでしょう。…とある学校で運動会が催されました。閉会後、白組と赤組に分かれての団体競技の結果を表にしました。まず白組から、棒倒しの得点、玉入れの得点、くす玉割りの得点、つな引きの得点、を 1 行ずつ書き、続けて赤組の得点を同様に記してあります。でも同じ競技の得点を比較したいので、得点を横に並べ、白組と赤組の 2 行にまとめたいと思いました(→図 1)。さてどうしますか？これもシェルスクリプト NoSQL なら簡単です。ユニケージコマンド (後述) というオープンソースの独自コマンドセットの一つ “yarr” コマンドを使えば一発でできます。

「独自のコマンド作れば一発なのは当たり前じゃないか！」と文句を言われてしまいそうですが、シェルスクリプト NoSQL の利点の一つはそこなのです。無いものは作ればいいのです。SQL 文を使う場合、既存の句や関数 (あるいはそれの組み合わせ) でできることしかできないと思いますが、シェルスクリプト NoSQL の場合、標準コマンドに自分の

図 1. つまりこういうこと（縦並び表の横並べ）

望む処理をこなせるものが無かったとしても、C 言語なり Perl なりを使って自分で作ってしまえるのです。

「ユニケージコマンド」とは何者？

説明を後回しにしてきたユニケージコマンドについてそろそろ紹介しておきましょう。

ユニケージコマンドというのは、USP 研究所が作成している UNIX 系 OS 向け独自コマンドセットです。一番の特徴は、シェルスクリプトによるデータベース管理をより手軽に、便利に使えるようにすべく、POSIX 標準コマンドには用意されていないテキスト処理コマンドを収録していることです。現在そのうちの 26 コマンドがオープンソースとして一般に公開されており、個人の非営利利用であれば自由に利用・改変できるようになっています。

使い方はこんな感じ

まず、その 26 コマンドをさらっと紹介します（表 2）。

SQL にあって標準コマンドに無い機能を補うものや逆に SQL には無い機能を実現するもの、できなくはないものの記述すると複雑になってしまうものを 1 コマンド化したもの、さらにコンマを打ったり (comma) 桁揃え (keta) をしたりといった表示を整えるコマンドが用意されています。

今回はさらに、このコマンドを作成した USP 研究所が実際の現場で使っているコードを見せてもらいました (→次ページのコード)。その特徴が現れている箇所をほんの少し紹介しましょう。

このコードは 3 つのセクションに分かれています。最初のセクションでは、各店舗の日毎の売り上げが入っているファイルから 2007/12/03 週の全店舗商品販売数を集計

表 2．公開中のユニケージコマンド（評価版 ver. 0.1）

コマンド名	機能
comma	3 桁毎にコンマをつける
delf	指定された列を除いて表示する
divsen	指定列を 1000 で割る
getfirst	1 行目を出力する
getlast	最終行を出力する
gyo	行数をカウントする
han	仮名文字を半角にする
join0	INNER JOIN する①
join1	INNER JOIN する②
join2	RIGHT OUTER JOIN する
kasan	累計値を求める
keta	列インデント (桁揃え) をする
loopj	FULL JOIN する (3 つでも 4 つでも可)
map	縦並びのデータを横並びにする①
ratio	全行に対する構成比を求める
retu	列数をカウントする
self	指定された列を表示する
sm2	グループ化 (GROUP BY) して合計 (SUM) する①
sm4	グループ化 (GROUP BY) して合計 (SUM) する②
sm5	グループ化 (GROUP BY) して合計 (SUM) する③
tarr	横並びのデータを縦並びにする
tateyoko	表の列と行を総入れ替えする
up3	2 つ表を UNION してソートする
yarr	縦並びのデータを横並びにする②
ysum	列 (＝横方向) の総和を求める
zen	仮名文字を全角にする

ユニケージコマンドセット評価版 は USP 出版から頒布されています。詳しい情報はこちらから。
http://www.usp-lab.com/pub/

しているようです。awk を使って 2007/12/03 週の行を抜き出し、さらに self を使って商品コードと売れ数の列を抜き出し、最後に商品毎の売れ数の合計を出しています (sm2 の引数 "1 1 2 2" は 1 列目「商品コード」でグループ化して、2 列目「売数」の合計を求めるという意味)。これをテンポラリーファイルに書き出し 3 つ目のセクションで使います。2 つ目のセクションでは、在庫データを同様に集計してやはりテンポラリーファイルに書き出しています。そして 3 つ目でこれらを JOIN しています。全ての JOIN で商品コードをキーにして、最初の loopj では売り上げ数と仕入数と在庫数を FULL JOIN、次に元売数と INNER JOIN、さらに商品名と INNER JOIN しています。このコードを通せば全店舗でどの商品がどれだけ仕入れられて売れたか一目瞭然、というわけだと思います。

```
# 変数の定義
dir=$HOME/REPORT/HANBAI
datd=$dir/DATA
tmp=/tmp/$$

# 集計期間
start=20071203
end=20071209

# 販売データの集計
#####################################################
#   URE       1. 店 2. 商品 3. 日付 4. 売数 5. 売上 6. 値引
#   $tmp-ure  1. 商品 2. 期間売数合計
awk '$3>="'$start'"&&$3<="'$end'"{print}' $datd/URE  |
self 2 4                                             |
sm2 1 1 2 2                                          > $tmp-ure

# 仕入データの集計
#####################################################
# 在庫データの集計
#####################################################
#   ZAIKO. 日付 1. 店 2. 商品 3. 日付 4. 在庫
#   $tmp-zaiko 1. 商品 2. 最終日在庫数合計
self 2 4 $datd/ZAIKO.$end                            |
sm2 1 1 2 2                                          > $tmp-zaiko

# データの連結
#####################################################
# $tmp-data : 1. 商品 2. 商品名 3. 原価 4. 売価 5. 売数 6. 仕入数 7. 在庫数
loopj num=1 $tmp-ure $tmp-sire $tmp-zaiko            |
join1 key=1 $datd/GENBAI                             |
join1 key=1 $datd/NAME                               > $tmp-data
```

筆者はコードの意味について一行一行解説を受けたわけではないのですが、ユニケージコマンドリファレンスを見ながらちょっと眺めただけでわかってしまいました。そして、やっている事が実に SQL 臭いといいますか…。いや、そうではなくて、今まで SQL 文で表現していたものがこういうふうにしてシェルスクリプト NoSQL でも容易に表現できてしまうということですね。

「紙書類の置き換え」が発想の原点

ところでなぜこのような 26 コマンドが公開されるに至ったのでしょうか。コマンドセット公開元である USP 研究所に話を聞いてみると興味深い内容でしたので紹介します。これはデータベース NoSQL がどういう用途に向いているのかにも繋がる話でもあります。

今回伺った話によれば、まず、シェルスクリプトでデータベースを構築しようという発想の原点は「紙書類の置き換え」であるといいます。書類というのは基本的に文書 (つまりテキスト) であり、従って紙の書類はテキストファイルで十分置き換えられるのだそうです。そして、事務作業はそういった文章を扱う仕事であるから、事務作業をシェルスクリプトによるデータベースに置き換えることはごく自然にできるとのことです。

シェルスクリプトで数々のテキスト処理コマンドを駆使してデータベース相当の処理をするなら、データはテキストファイルで持たせなければならないわけですが、これは紙の書類の置き換えという発想と相性がよく、互いに都合がよいと言えます。従ってテキスト処理コマンドを自作していく方針になったそうですが、ここでコストという企業ならではの発想が色濃く反映していくことになります。

独特のラインナップになった理由

先に紹介したユニケージコマンドを見て、筆者は二つ程気になったことがあります。一つは名前です。日本語が名前になっているコマンドが目立ちませんか。筆者は、オープンソースとして広く公開するなら英語に統一すべきじゃ

ないかと感じましたので理由を質問してみました。すると「日本人であるお客様や自分たちにとってわかり易くすることが大切。その方がプログラミングやメンテナンスが早く簡単に行える」という回答でした。本記事の序盤で「SQLは大掛かりな…」と述べましたが、大掛かりさを取り除く上で一つ合理的な判断だと思います。ちなみに、逆に日本語ではないコマンドも混在している理由も尋ねてみましたが、それは「日本語にするとかえって長ったらしくなるから」ということで、実用性とのバランスでそのようになったのでしょう。

もう一つは機能です。ユニケージコマンドを使わずとも、標準コマンドで書ける処理がいろいろあります。ratio(累計を求める) は awk でできることを既に示しましたし、self(指定列を取り出す) や delf(指定列を除く) 等は awk や cut コマンドでできます。最たる例は gyo(行数を数える) で、こんなもの wc コマンドでできます。この点についても質問してみましたが、理由はやはりコストから来ているそうです。頻繁に行われるデータ処理で、標準コマンドでの記述が面倒臭いものに対してはあえて自作コマンドを作成するのだそうです。冷静に考えると、最後に例示した wc も行数以外に余計なスペースやファイル名が併記されます。これらを取り除くのにいちいち awk '{print $1}' をなどと後述しなければならない手間を考えると…、うーん納得です。

このようにして、開発や保守に掛かるコストを強く意識し、業務の現場で求められるシステムという命題に応えるべく作り出したものが、ユニケージコマンドであり、あのような独特のナリやラインナップになったのだそうです。大掛かりな SQL 系データベースの代わりを担うための選択肢としてシェルスクリプト NoSQL を利用しようと考えるなら、実用性本位で生まれたこのコマンドセットはとても有用だと思います。

そんな実装で大丈夫か？— パフォーマンス

ここまでシェルスクリプト NoSQL の機能的な話をしてきましたが、パフォーマンスは大丈夫なのでしょうか。シェルスクリプトは (ついでに言うと中で頻繁に利用している awk コマンドも) コンパイル要らずのインタプリタ型言語で、しかもデータの実態はテキストファイル。いくら手軽な手法とはいえ、使い物になるスピードが出るのかと疑いたくなります。どこからともなく言われそうです。「そんな実装で大丈夫か？」と。そこで、簡単ではあるもののベンチマークテストをしてみました。

ベンチマーク結果は意外なものに

【比較対象】

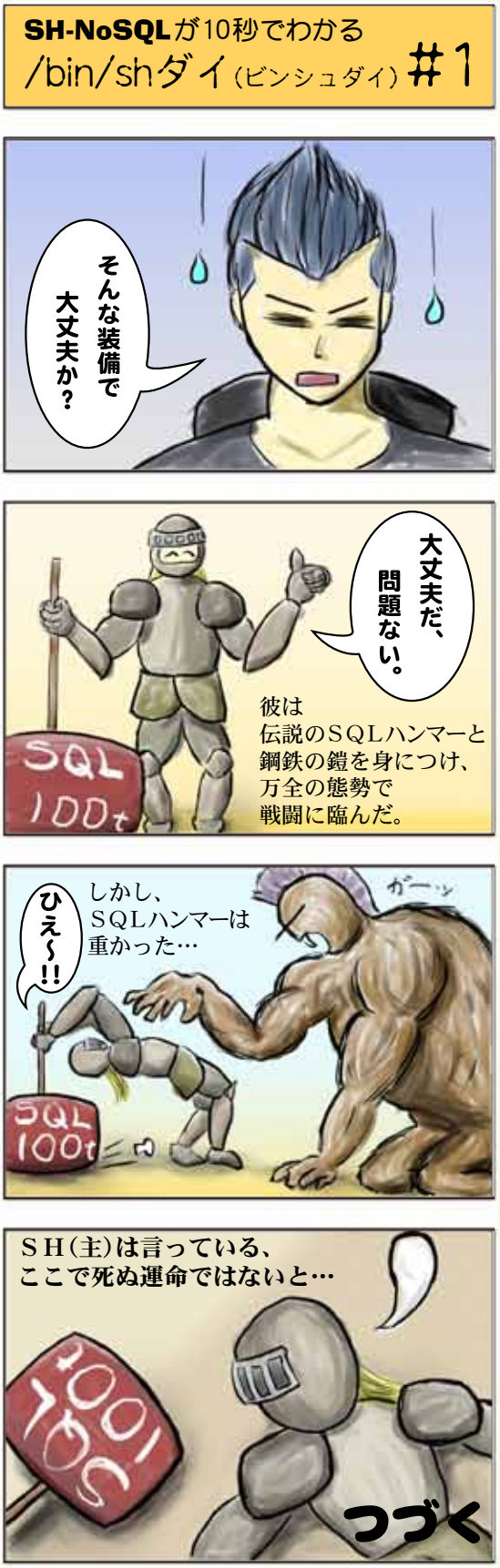
広く利用されている SQL 系データベースと比較するのがわかりやすいと思いますので、MySQL および PostgreSQL と比較することにしました。バージョンはそれぞれ 5.5.8、9.0.2、どちらも執筆時の最新版です。それぞれ同じホストにデフォルトインストールしました。対するシェスクリプト NoSQL は、そのホストに入っている /bin/sh をはじめとするコマンド群とします。もう少し具体的に記しますと、ホスト OS は FreeBSD 8.1-RELEASE で、sort コマンドも GNU 版ではなく BSD 版にしてあります。これは筆者が FreeBSD 使いであり、たまたま手近にこのような環境があっただけです。

【比較内容】

本来なら様々なメニューを用意して徹底的に比較したいところですが、誌面の都合もありますので 1 競技 2 種目としました。テストは次のようにして実施しました。

●準備●

1. SQL の INTEGER 型で表現可能範囲 (-2147483648 ～ 2147483647) で乱数を awk 等で 100 万個作ります。この時、値の重複があるようにします。これはグループ化作業もベンチマーク内容に含ませるためです。どの程度重複させるのがよいかについては様々意見があるでしょうが、今回のテストでは 2% ほどを重複させてみました。
2. SQL 系データベースでは、INTEGER 型カラム 1 つからなるテーブルと、VARCHAR(11) 型カラム 1 つからなるテーブルを用意し、それぞれに対してこの乱数列 100 万個をインポートします。そしてインポート後にはどちらのカラムにもインデックスを作成し、PostgreSQL では念のため VACUUM も実施しておきます。一方シェルスクリプト NoSQL では、1 行 1 乱数のテキストファイルとして置いておきます。これで準備完了です。



●テスト●

3. グループ化して昇順ソートさせ、画面に結果が表示され始めるまで（され終わるまでではない）の時間を計測します。具体的には各々のコンソールで次のコマンドを実施します。

◆MySQL および PostgreSQL

```
SELECT num FROM tbl GROUP BY num ORDER BY num ASC;
```

◆シェルスクリプト NoSQL（数値扱いの場合）

```
cat tlb.txt | sort -k 1n,1 | uniq > /dev/null
```

◆シェルスクリプト NoSQL（文字列扱いの場合）

```
cat tlb.txt | sort | uniq > /dev/null
```

計測の仕方を詳しく示しておきます。MySQL の場合は mysql コマンドが標準で表示する結果、PostgreSQL の場合は psql コマンド内で \timing という設定を行うことで表示される結果、シェルスクリプトの場合は結果を /dev/null に捨てるまでの動作を time コマンドで計測した結果としました。

【テスト結果と考察】

OS による実行時間のゆらぎの影響を抑えるため、それぞれ 5 回実施して平均値で比較したのですが、結果はこうなりました（表 3）。

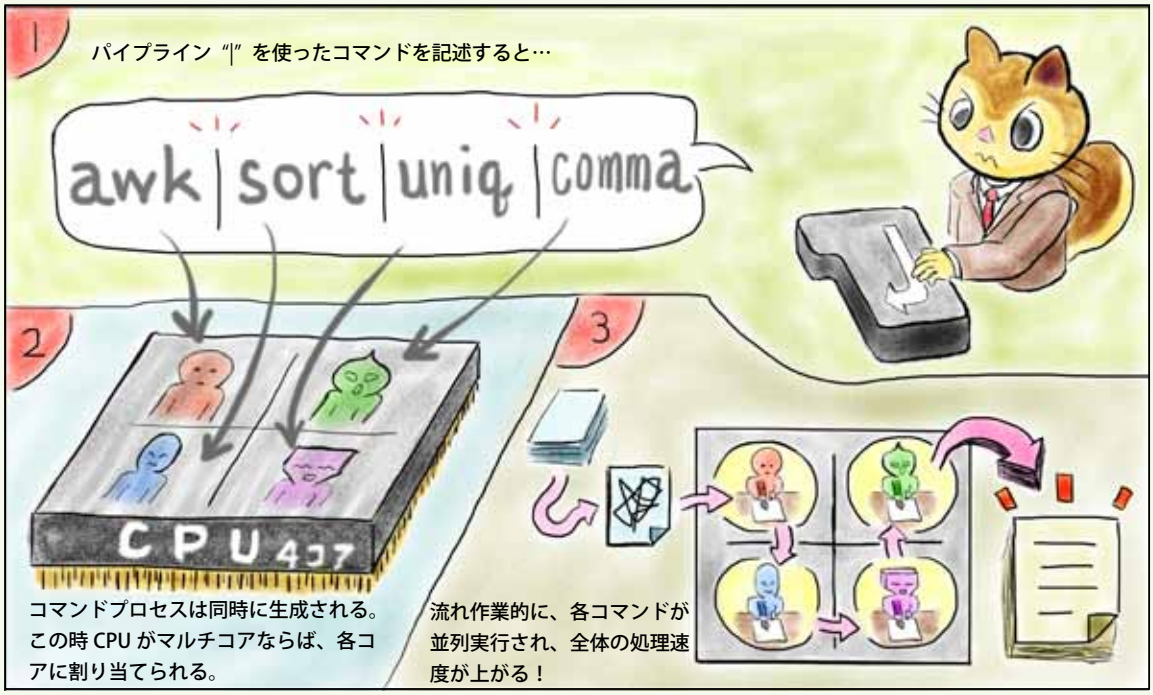
どうですか、意外にもシェルスクリプト NoSQL の圧勝です。なかなかやりますね。今回の比較では、乱数列を数値扱いする種目、乱数列を文字列扱いする種目、という 2 種目を用意したわけですが、SQL 系は数値扱いの方が速く、逆にシェルスクリプト NoSQL は文字列扱いの方が速いという結果が出ました。SQL 系は、数値をコンピューター的により扱い易い数値として格納できるから数値扱いの方が速いのでしょう。一方シェルスクリプト NoSQL では文字列のまま格納していますから、数値ソート時は内部で数値に変換するという処理を挟むこととなり、結果として数値扱いの方が遅くなってしまったのでしょう。それでも 2 位に 2 倍以上の差をつけていることには目を見張るものがあります。

それにしてもシェルスクリプト NoSQL が 1 位になった

表 3．ベンチマーク結果

実装	100万行グループ化ソート所要時間（数値扱い）	100万行グループ化ソート所要時間（文字扱い）
MySQL	22.98 秒	24.98 秒
PostgreSQL	9.56 秒	14.49 秒
シェルスクリプト NoSQL	**4.39 秒**	**3.97 秒**

※ 5 回の平均値。また、コマンド入力から表示開始直前までの所要時間です。



理由はどこにあるのでしょうか。それはシェススクリプトの低水準性にあると思います。この記事の冒頭で SQL 系データベースは「何でもできる代わりに大掛かり」と述べましたが、「何でも」を実現するために何重ものレイヤーの上で動いていると考えられます。つまり、関数やライブラリー・モジュールを何重にも呼ぶことで「何でも」を実現しているのです。これに対しシェルスクリプトを使ってデータベースを実現する場合、SQL 系ほど完璧な「何でも」を追求せずに awk や sort いった原始的なコマンドで実現しており、レイヤーが薄いおかげで高速性がもたらされているのでしょう。

マルチコア CPU で鬼に金棒

シェルスクリプト NoSQL の強みは実はもう一つあります。それは、マルチプロセッシングによる恩恵です。

CPU は 4,5 年くらい前から、クロック数を上げるのではなくコア数を増やす方法で性能を上げるようになってきました。とはいってもこの恩恵を受けるには、スレッドやプロセス数を増やして並列処理するようなプログラミングでなければなりません。そのようなプログラムを作るのは少々手間です。ところがシェルスクリプト NoSQL はこの恩恵を簡単に受けられてしまいます。なぜなら、awk して sort して uniq して…というように、既に細かな粒度でプロセスが分かれているからです。

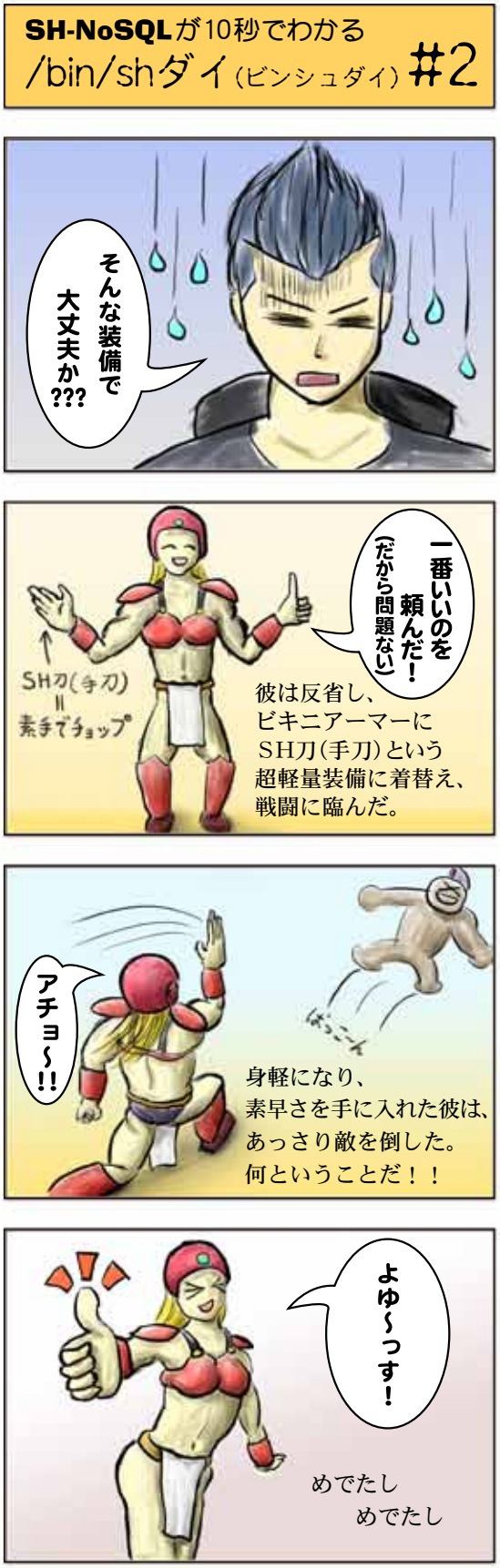
awk して sort して uniq して…というように多段の処理をさせる際は "|"(パイプ , パイプライン) で繋ぎますね。これは、最初のプロセスが生まれて、処理終わったら次のプロセスに生まれ変わって…、という動作ではありません。パイプで繋がれた全てのプロセスが同時に生まれ、各自データが流れて来次第処理をするのです。まさに工場の流れ作業、つまりパイプラインなのです。だからコア数 (もしくはプロセッサ数) が増えれば、これらの各プロセスが綺麗に分散して配置され易くなるというわけです。元々細かい粒度でプロセスが分かれていることにより得られる恩恵なのです。

先の比較はシングルプロセッサー PC 上で行いましたが、最近の 6 コア CPU を搭載した PC 上で比較したらどういうことになるのか、興味深いところです。

賢く使おう、シェルスクリプト NoSQL

しかしシェルスクリプト NoSQL がこの比較で最速だったことについて、「単に有利な種目を選んだだけでしょ」という指摘もあることでしょう。一応 SQL 組にはインデックスを持たせたり、またシェルスクリプトの強みが発揮されないようシングルプロセッサー PC で実施したりと配慮はしたつもりですが、確かに苦手とする種目が思いつかないわけでもありません。

例えば、SQL で言う UPDATE(レコードのサイズが変わ



る場合）や DELETE は苦手なはずです。シェルスクリプト NoSQL はデータをテキストファイルで持っていますから、UPDATE や DELETE が行われるとファイルを途中から以降全部書き直さなければならないことになります。よって、ファイルが大きかったりそれらの要求が頻繁であったりするほど不利になるであろうことは想像に難しくありません。

UPDATE と DELETE 操作の考え方

UPDATE と DELETE、データベースにとってどちらも重要な機能です。では、前述の指摘に基づいて、シェルスクリプト NoSQL は大して使えない（= スケーラビリティーの低い）データベースという評価になってしまうのでしょうか。いえいえ、そんなことはありません。前述の指摘では、UPDATE と DELETE の表面的な操作を真似て、本質的な働きを見ていないことに問題があるだけです。

この問題を解決するためのアプローチは色々考えられますが、例えばシェルスクリプト NoSQL の原点が「紙書類の置き換え」にあることに基づいて考えてみましょう。

もし UPDATE や DELETE に相当する作業が紙の上で正直に行われていたとすれば、修正が必要な箇所から後ろを一旦全部消して書き直さなければならないことになります。でも実際はそんなこと殆どしません。せいぜい書き直したい箇所を含むページをバインダーから取り除き、直したページを挿し込むくらいでしょう。一冊作り直すという非効率な作業は通常求められないはずです。そう考えれば、データを多くのファイルに分散して持たせればよいことに気づきます。あとはファイルシステムという階層管理が得意なシステム（これも一種の NoSQL データベースと言える？）をうまく組み合わせれば、上手に管理できるはずです。

このように考えれば、シェルスクリプト NoSQL が苦手とする処理は実際それほど存在しないはずなのです。もし動作が重いと感じるようであれば、紙書類でやる場合を思い浮かべながらデータの持たせ方を考え直すべきなのです。

そもそも NoSQL は適材適所に…

万能な SQL 系が既にある中で、あえて次々生まれてきた NoSQL。ある意味 SQL 系の万能さを犠牲にして特性を持たせています。ですからその特性を理解し、的確な使い方、かつ適材適所な使い道を考えることは必然と言えます。シェルスクリプト NoSQL とて例外ではないでしょう。

しかし「シェルスクリプト NoSQL の特性は一体何なのか？」これを理解すれば、適材適所な用途が当初の想像を遥かに超えて次から次へと見つかることに、きっとあなたは驚くことでしょう。

原色 Linux 美女図鑑

第 1 回 Manami さん（Linux 女子部）

Linux 好きの女子と Linux の素敵な関係を紹介するこのコーナー。第 1 回目となる今回は、Linux 女子部を主催する Manami さんが登場です。カメラや着物が好きで活動的な Manami さんですが、実は LPIC レベル 3 試験の認定者！ Linux の実力も折り紙付きです。そんな Manami さんに Linux の出会いと、これからの目標について伺いました！（編集部）

Profile
TwitterID : @mana_cat
Blog http://mana-cat.blogspot.com/
Facebook http://facebook.com/manami.fukunaga

Q）Linux との出会いを教えてください
A）19 歳のとき、個人 Web サイトを立ち上げたのがきっかけです。

19 歳の頃、個人 Web サイトを立ち上げたのがきっかけで、Linux と出会いました。当時は Blog という言葉もなかった時代で、CMS をいじって試行錯誤しながらケータイ向けのサイトを作っていましたね。因みに、なんでケータイ向けサイトかというと、周りの友だちが PC を持っていなくて、ケータイからじゃないと見れなかったからです（笑）

Q) 普段は Linux とどう付き合っていますか？
A) 仕事で EC サイトの運用を担当！

大学卒業を機に熊本から上京し、お仕事でご縁があって大手 EC サイトの運用を担当することになりました。そのとき、Linux のほか、Solaris、FreeBSD とも出会うことができました。なので、いまはお仕事で Linux と深く関わってるかなと思います。自宅では、おうちサーバとして Fedora を使っています。あと Linux じゃないけど、Solaris (SPARC) のワークステーションもあって、検証機として使ってます。

Q)Linux のどこが好き？
A) 初心者でも使いやすい。仲間が多いのも魅力。

Linux の良いところは、どんな人でもとっつきやすいところ。初心者が挫折しにくく、馴染み易いことは大事ですよね。あと、周りに Linux 使っている人が多くて仲間もできやすいし、関連書籍も多く、セミナーや勉強会も充実していて、なおかつ LPIC などの認定資格がありスキルアップしやすい環境だと思います。こういった Linux を取り巻く活発的な環境が、私は好きです！

Q) これから、Linux とどう付き合っていきたい？
A) みんなの「やる気」を応援したい！

Linux を使っている女性は、とても勉強熱心なんです！ Linux 女子部を立ち上げたときも、向上心の高い Linux 女子が多いことから、自然とミュニティ設立にまで至りました。私はみんなの「やる気」を応援していきたいと思います。これからいろんなイベントをコミュニティで提案していき、Linux 女子が楽しくお仕事できるようなキッカケになればいいな、と思っています。今後の Linux 女子部の活動にご期待ください！！

有楽町のカフェでお会いした Manami さんは、とてもやわらかい印象で、Linux 女子部に、男女問わずたくさんの人が集まる理由が分かる気がしました。これからも活発な活動を予定している Linux 女子部公式ホームページは、**http://sites.google.com/site/linuxjoshi/** です。

USP MAGAZINE vol.0
編集・発行人　鹿野恵子　発行・発売 USP 友の会　〒105-0003 東京都港区西新橋 3-4-2 SS ビル 3 階
有限会社ユニバーサル・シェル・プログラミング研究所内　電話 03-3432-1174
定価 400 円（本体価格 381 円）

シェルでつながる技術の輪！
なんでもありのグルーコミュニティ

USP友の会 会員募集中

USP 友の会
マスコットキャラクター
「ちんじゅうちゃん」

UNIX/Linux/ シェルスクリプトの 可能性を極限まで追求します！

【入会特典】

・入会時「ちんじゅうちゃん」グッズプレゼント。

・エンジニアの好奇心を刺激する内容満載
「USP MAGAZINE」が年４回お手元に届きます。
＜今後の特集予定＞
シェルスクリプトで Hadoop、言語対抗コマンドバトル・ロワイヤル
Android で CUI!?、運用監視もシェルで実装
お願い！最新技術ランキング……等々

・各種イベント開催時に会員向けサービス（割引）予定。

【入会希望の方は】

・USP 友の会ホームページの申込フォームより入会をお申込いただいた上、
年会費 2,000 円を所定の銀行口座にお振込みください。

詳しくは http://usptomonokai.jp にアクセス！

＜イベント予告＞

TechLION

2011/03/31 Thursday
OPEN18:30/START19:30
@Shinjyuku Naked Loft
charge 1,500yen(adv)

本物のコンピュータ技術とは何か？
TechLION は「本物」とエンジニアを結ぶトークライブ。
IT という名の荒野を生き残る秘訣がここにある。

出演：宮原徹、法林浩之、上田隆一、他
ローソンチケットまたはネイキッドロフトの
ウェブサイトにてチケット好評発売中！！