ISSN 2187-9443

for the sophisticated shell scripters

特集 ついにユニケージがAWSに登場! 「AWS Tukubai」を試す

ンジャルガルの海外開拓レポ

C e

アテアエロシックション

4.1

20

浮川初子氏に訊、 技術者哲学

chLION

Artwork created by Koji Ishizuka. © 2012 Zie's Bank .Co,. LTD All rights reserved

From Editor

『UNIX という考え方』(マイク・ガンカーズ著)によれば、人間が作ることができるのは 3 つの段階から なるシステムしかありません。

その最初の段階 " 第 1 のシステム " は、1 人か、せいぜい数人の小さなグループが作った、無駄がなく俊 敏なシステムです。

最小限のコストで高い性能を実現するそのシステムのコンセプトは、人間の創造力を刺激します。 やがて、そのコンセプトに惹かれて集まってきた多くの専門家が、次の段階 " 第 2 のシステム " を作り出 すのです。

この " 第 1 のシステム " が誕生してから " 第 2 のシステム " に発展するまでは、ほとんどの場合、孤独な 長い道のりを経なければなりません。

最初は誰にも理解されない。だが、あきらめないで根気強く、そのシステムの素晴らしさを説き続けているうちに1人、2人、さらに長く続けていれば10人…100人と、指数的に理解者が増えていきます。

B.W. カーニハン教授によって評価されたことは、ユニケージ開発手法が " 第 1 のシステム " の段階を終え、 " 第 2 のシステム " に移行する時期が近いことの証しでもあるのでしょう。

しかし、さらに長い道のりを歩き続けて、"第3のシステム"を作り出せたとき、それまでの努力が"実った"と言えるのです。

USP MAGAZINE 編集部

Contents

特集 ついにユニケージが AWS に登場「AWS Tukubai」 を試す	3
うにっくすなやつら 第5回 長谷川猛	15
シルネン・ブヤンジャルガルの自立への挑戦、情報整理技術の開拓 第2回	20
スズラボ通信 第2回 すずきひろのぶ	26
漢の UNIX 第7回 後藤大地 ····································	30
浮川初子氏に訊く、技術者哲学・・・・・	34
ユニケージエンジニアの作法 第8回	40
UNIX ネイティブの電子工作塾 第3講 大野浩之 高嶋健人	44
今私たちは何を学ぶべきか 第10回 大岩元・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	48
IT 美女図鑑 —— 高坂いくこ ······	50
TechLION 再録 田中邦裕が語るエンジニアと経営と、日本と海外と	52
中小企業手作り IT 化奮戦記 第8回 菅雄	59
シェルスクリプト大喜利 第10回	62
Tech 数独 ······	66
天地概況 奈須蛍路 / 編集後記 ······	67
※ 薄字の記事はよりぬき版には収録されていません。是非正式版をお求めください。	

特集

っいにユニケージがAWSに登場 「AWS Tukubai」を試す

USP MAGAZINE 編集部

シェルスクリプトによるシステム開発。データベースアプリケーションを一切使わず、テキスト ファイルだけで業務システムを組み上げる。しかも動作は高速で、開発期間も短い……。も しそんな噂を耳にしたのなら、その正体は usp Tukubai かもしれない。

一部の人に知られるのみで、ほぼベールに包まれた存在であった usp Tukubai が、2013 年 8 月、 Amazon Web Services (AWS) に登場し、使いたい時に使いたい分だけ、誰でも気軽に利用 できるようになった。

果たして usp Tukubai とはいかなるものであるのか。本特集では、これを提供する「AWS Tukubai」の使い方を紹介しつつ、その実力の程を探っていく。



第1章—3分でわかる「AWS Tukubai」

2013 年 8 月、シェルスクリプトによる本格的システム開発環境「usp Tukubai」が、AWS上に AMI (仮 想マシン)という形で登場したという報道がなされた。これを「AWS Tukubai」と呼ぶことにするが、 それは一体何物なのだろうか。本章ではまず、AWS Tukubai とはどういうもので、どんなメリットがあ るのかを、3 分程度で理解できるよう、簡単に紹介しよう。

📘 Q1.AWS Tukubaiって何?

A1.シンプルで高速な開発環境「uspTukubai」のAWS提供版

これまで一部の企業にのみ供給され、一般ユーザー には「シェルスクリプトベースにも関わらず、動作 が高速で、コストパフォーマンスも高いらしい」と いった噂が聞こえてくるのみだった usp Tukubai。こ れを AWS 上で誰でも手軽に使えるようにしたものが AWS Tukubai である。

AWS 上で供給するメリットは誰でも使えることの みならず、初期費用ゼロかつ使用した時間分だけ料 金を払えばよい点にある。ゆえに、必要な時だけ使 いたいというオンデマンドな用途に最適である。



Q2.昨年出た「Open usp Tukubai」と何が違う?

A2.平均20倍以上の処理速度と200以上のコマンド

AWS Tukuai で提供されるビジネス版は、Open usp Tukubai (Open 版)を大幅に超える 200 個以上のコマン ドを収録。しかも**全て C 言語で書かれれており、**Open 版 に比べ、平均 20 倍以上の速さ(編集部比)で動く。

例えば、Tukuabiコマンドで定番の self (SQL では SELECT によるカラム抽出に相当)を用いた 100 万行の カラム抽出では、同じ AWS インスタンス上で実行した Open 版と速度比較してみたところ、約97倍となった。

コマンドの豊富さも見逃せない。Open 版では現在 52 個のコマンドが提供されているが、AWS Tukubai ではそ の4倍以上の 221 個もある。SQL で設計されたシステム の移植に便利な tag シリーズコマンド、Microsoft Excel ファイルを読み書きできる rexcel,wexcel コマンドの提供 など、ビジネスシーンへの対応がより強化されている。



100 万行のデータから一つの列を抽出する作業を、 同じ c1.xlarge インスタンスにて実行。

ビジネス版のみ提供される主なコマンド 名称 機能概要

bb	Bashコード美化 (beautifier) フィルタ
calsed	文字列Aを文字列 B に変換 (軽量 sed)
fsed	特定フィールドを対象にした sed
htable	HTML の 内データ抽出
pad0	指定フィールドを0パディング
tag*	SQLライクなデータ処理コマンド群
*excelx	Microsoft Excel ファイルを読み書き
	11. 今半

Q3.AWS Tukubaiの利用料金は?

A3.初期費用0で、1時間0.6^(注1)ドルから利用可能

AWS は、仮想マシン (VPS) 等を提供するサービス。VSP は何といってもハードウェア購入の初期負担 無しに始められるのがメリットだが、AWS はありがたいことに、殆ど^(注2)が従量課金制。つまり「使っ た分だけ課金。使わなければ 0 円」。一時的に Tukubai を使いたい、試してみたい、というオンデマンド 用途には最適と言えるだろう。具体的な価格については、表1、表2を参照してもらいたい。

	ikubai)汉况下	())	(1/~)	アノヘ)の主安ノ	1// 2/0	Bulling Control Control		
インスタンス		ECU*3	VT 11	ネットワーク	1時間使用料 ^{*4} (単位米ドル)			
タイプ *1	WSCPU数	ECO	メモリ	パフォーマンス	usp Tukubai	EC2	合計	
t1.mcro	64bit $ imes$ 1	1	630MiB	非常に低	0.60	0.02 ^(注1)	0.62 ^(注1)	
m1.small	64bit $ imes$ 1	1	1.7GiB	低	0.60	0.06	0.66	
m1.medium	64bit $ imes$ 1	2	3.75GiB	中	0.60	0.12	0.72	
m1.large	64bit $ imes$ 2	4	7.5GiB	中	0.60	0.24	0.84	
m1.xlarge	64bit $ imes$ 4	8	15GiB	高	0.60	0.48	1.08	
m2.xlarge	64bit $ imes$ 2	6.5	17.1GiB	中	0.60	0.41	1.01	
m2.2xlarge	64bit $ imes$ 4	13	34.2GiB	中	0.60	0.82	1.42	
m2.4xlarge	64bit $ imes$ 8	26	68.4GiB	高	0.60	1.64	2.24	
m3.xlarge	64bit $ imes$ 4	13	15GiB	中	0.60	0.50	1.10	
m3.2xlarge	64bit $ imes$ 8	26	30GiB	高	0.60	1.00	1.60	
c1.medium	64bit $ imes$ 2	5	1.7GiB	中	1.20	0.145	1.345	
c1.xlarge	64bit $ imes$ 8	20	7GiB	高	1.20	0.58	1.78	
hi1.4xlarge	64bit $ imes$ 16	35	60.5GiB	10 ギガビット	1.20	3.10	4.30	

表1.AWS Tukubai 仮想マシン (インスタンス) の主要ラインナップおよび価格

*1 デフォルト設定のディスク容量はどれも8GiB。別途追加料金により増設可能である (EBS) 他、各インスタンスタイプに応じて一定量の揮発性ディスク領域 (インスタンスストレージ:電源断時に失われる)を無料で増設することも可能。

*2 Tukubai コマンドは並列処理に向いているため、仮想 CPU 数が多いほど性能が上がり易く、お勧めである。

*3 ECU とは、CPU 性能を相対的に示すための Amazon による評価値(例: c1.medium の CPU 性能は t1.micro の 5 倍) *4 課金は、電源投入直後、及びその後 1 時間毎に行われる。従って、一瞬でも電源投入すると 1 時間分の課金が行われる。

表 2. AWS のその他主な課金条件^{*1}

項目	価格(目安)
EBS (不揮発ディスクスペース) 使用料	1GB あたり月 0.1米ドル
EBS アクセス (IOPS) 使用料	スタンダードボリュームの場合、 一時間フル稼働させたとして概ね 0.036 米ドル *2
EC2→インターネット宛データ転送料 ^{*3}	最初の 1GB までは毎月無料 以降毎月 10TB までは 1GB あたり 0.12 米ドル

*1 詳しくは、Amazon EC2 料金表のページを参照されたい。http://aws.amazon.com/jp/ec2/pricing/
 *2 表の価格は参考値。詳細な条件によって実際の価格は変化する。
 *3 インターネット→ EC2 宛のデータ転送は無料である。

注1 AWS 新規登録から1年間は、最小マシン構成のt1.microを選ぶとそのマシンの課金分(1時間 0.2 米ドル)が毎月実質無料になる。
 詳しくは「AWS 無料利用枠」で検索。

注2 ただし EBS と呼ばれるディスクスペースは、稼働させていなくてもディスクスペースを貸りているだけで料金が発生する。その他、 主な課金条件を表2にまとめてあるので参照してもらいたい。

📕 Q4.どうやって使うの?

A4.次章でじっくり解説!

AWS なら、自分でマシンを構築するより遥かに簡単に始められる。だが、一部に英 語表記の手続き画面があるなど、AWSを使った事のない人には少々難しいかもしれな い。そこで AWS 初心者向けの始め方マニュアルを用意した。早速、次ページを開こう!



第2章—AWS Tukubai をはじめてみる。

昨年の Open usp Tukubai に続き、AWS 上でビジネス版 usp Tukubai が提供されたことで、より身近に なった Tukubai。ただ、使い始めるまでの手続きは、AWS 初心者には少々難しいかもしれない。そこで AWS Tukubai を使い始めるまでを手取り足取り紹介しよう。また、説明ビデオ(約8分)も公開されてい るので、そちらを参考にするものよいだろう。→ http://www.youtube.com/watch?v=Yxop1Wrjc-M

- 1.AWS を始める

 AWSサインアップのため、ブラウザーからhttp:// aws.amazon.com/jp/free/ ヘアクセス。新規サイ ンアップ者は1年間「無料利用枠」を使えるので、参 考に読んでおこう。読んだら「サインアップ」開始だ。



指示に従い、名前や利用料金支払いのためのクレジットカード番号等を入力していく。イタズラサインアップ防止の為、途中で電話による認証がある。(こちらの声を音声認識するぞ。スゴい!)そしてサインアップ完了後、下の画面に。そうしたら「AWS Management Console の起動」を選択。



マーケットプレイスで 2. Tukubai を取得

ここは AWS にサインインした直後の画面だ。
 (AWS を以前から利用していて、サインアップ済の人はここから始める)

この画面の右上の「アカウント / コンソール」内 にある「AWS Management Console」を選ぶ。



 一番左の列にある「EC2」を選ぶ。(Tukubai は EC2 の中で提供されているのだ)



特集―ついにユニケージが AWS に登場 「AWS Tukubai」を試す

5) EC2 ダッシュボード画面に到着。ここから Tukubai 入り仮想マシン (= インスタンス)を取得 し、作る。そのため「Launch Instance」をクリック。



6) Tukubai入り仮想マシンはAWSマーケットプレ イスで提供されている。よって、左サイドの「AWS Marketplace」を選択した後、右上に「Tukubai」 という検索キーワードを入れて、検索(Go)する。



 見つかった! そこに表示されている品名 "Unicage (usp Tukubai) Data Analytics Platform" をクリック。



8) 商品詳細画面。左下にマシン構成、右下に価格表 (仮想マシンが置かれる地域によって若干価格差 あり)がある。確認したら「Continue」をクリック。



3. マシンスペックを決める

 9) この画面で、仮想マシンのスペックを決める。だ が AWS 用の公開鍵 / 秘密鍵を持っていない場合 は、画面左下にある「Key Pair」項目内の「1.Visit the Amazon EC2 Console」にて、予め鍵を作ら ねばならない。(作ってある人は手順 12 へ)



10) こんなふうにして鍵を作る。途中で入力を求められるのは鍵の名前であり、パスフレーズではない。だから分かりやすい名前を付けておこう。

Requests	Click the crea	ou do not have any key pairs defined. te cay Pair button to download a new p	rivate key.
rved Instances	Greate Key Pair	Carcal 💌	
	Vie Dei Dans	W. Talaka	
e Taska	Ney For Name:	Sectores .	
		Create _	
85		4	

第2章—AWS Tukubai をはじめてみる

- 21) 鍵が発行されると秘密鍵がダウンロードされるの で適宜保管しておくこと。ダウンロード後、ブラ ウザーを手順9の画面に切り替え、再読み込み (F5) させる。これで AWS 用の鍵の登録は完了だ。
- 12) さて、仮想マシンのスペックを決める。左上に「1-Click Launch」と「Launch with EC2 Console」の2つの決め方が用意されているのだが、AWS 初心者は前者が絶対オススメ。ということで、「1-Click Launch」からマシンを選択する。すると右の価格表が自動的に選択されるので確認しよう。(ここで仮想マシン「Standard Micro(t1.micro)」を選ぶと、利用1年未満なら毎月最初の750時間は、EC2の分(\$0.02/hr)が無料になるぞ)決めたら右上の「Accept Terms & Launch with 1-Click」をクリック。



13) この画面に来たら、裏では仮想マシンが生成され、起動が行われている。つまり課金が始まっているということ。さあ、「AWS Management Console」へ戻って使い始めよう。

awsmarke	tolace	Amazen Wati Services H
••	Tele, Tonus Al Missure, Citige end)	Your Account
Shop All Categories +		CO (). Your Selfs
Satiware and AWS	howly usage fees apply when the instance is running. These fees wi	appear on your monthly bill. ココをクリック
Thank You! An inst	ance of this software will be deployed on EC2 soon	n after your subscription competes. Related Links
Once you are subcoribed. Once the cethware is ready	an instance of this ophysics will be deployed on BC2. (at loast 2-3 minuted), you can accous the ophysics He Your Sphysics	AVVE Mension ant Conscients Your Settings
Usage Instructions		 Continue shopping on AWS Marketplace
Once the instance is easily Unicoge plottern provided y by writing shell acripts. All Show more	g, soli into the instance using over rest and your Amazon private key. The su with ever 200 new commands that you can use at the command line or if the Unicage commands are located in AwsfaceMagTawabeNew. You c	
Software Installation	Details	
Product	Unicogo (usp Tukubal) Data Analytics Platform	
Version	2013/07/02, roleased 06/07/2013	
	US East (Magina)	
Region		

4. Tukubaiマシンを使う

14) 再び EC2 ダッシュボード画面。今度は左サイド



15) するともう動き出して (running) いる。ここでその仮想マシンをクリックすると、やがて画面底部に、アクセス用のアドレス (Public DNS) 等が表示される。このアドレスはメモしておこう。

termine Statistics Statisti	RC2 Dashboard	Launch Instance Actions	*				C	
Type Process Type Process Type Process Type Process Type T	Events 4							
PUTNESS PUTNESS	Tags	Weening: All Instances	· All instance	Types 💌 🕑) K	< 101013	 NUMBER
Mittanie Spartingt, mit Altz <	B INSTANCES	P Name to Instance	AMI ID	Reat Device	Type	State	Status Checks	Nam Stat
apid starts Altonomotions Altono	Instances	K overy ilicot27ani	ami-decfa/h7	rbs	c1 x longe	a runeing	C 2/2 checks o	none
Forces Alts Auto Aut	Spot Requests Reserved Instances							
Atta Reads 7:198 W 60-775 1986 W 60-775 1987 W 60-775 1987 W 70 1987 1987 1987 W 70 19	H INVES		oading, please	wait				
Read 10 104 チェックを入れて、************************************	4441			*****				
	Bundle Talks	チェックを入	h7	and all				
Guided Gu	B BLASTIC BLOCK STORE	Key Bale Name:	AND TALKS			Ken	will the	45.00
Designed Sector Protein Sector Protein Se	Volumes	Magilarian	hasti			4041	Louisth Index:	0
al Landon (1982)(21) Back Po Back P	Snapshoos	Flastic 12:				Read	Desire:	
Secret 2 for back Back PA Reserved Onces Lad Safe Parks Lad Safe Parks	WARMAN ASSAULT	Root Device Type:	abs			Ten		defail
Back Spanner His Caphalered His Caphalered	Servery Centers	Table Bulley	***			1.00	and a	
The service and the constraints and the cons	Datic Ps	ERS Continuinger	fains				and the second sec	11001110
Last Barkers 19/2793 19/279 19/279 19/279 19/279 19/279 19/279 19/2	Placement Groups	Black Devices:	のアドレ	Z (Puł	lic DI	vs)を:	メチする	781
Approximate Andre DANS Andr	Load Balancers	National Interfaces	0) 1 0	× (1 ui	JIC DI	(J) (C)		
Network Netrices Private DOS: 0-10-27-20-0-0-0-000 Product Coden: marks Private DOS: 0-10-27-20-0-0-0-0-0-0-0-0-0-0-0-0-0-0-0-0-	Kety Paera	Public DAS:	ar2.104.71.61.61	Consultant amag	Industry Com			
Private IPs: 20.231.258.278 Secondary Information Infor	Network Interfaces	Private DNS:	e-10-000-000-0			Pres	duct Codes:	marke
Consultant Melanta Hite -		Extensio 18x1	10.114.158.178					
		Korondary Brigata Ide-						
		the reason where the memory with the	CARACTERIA CONTRACT	ATTEND TOTAL	1.04 6.041			econom.

16) ログインしよう。まず AWS が用意しているターミ ナルによるログイン方法の解説から。これを使う場 合は上部メニューの「Action」から「Connect」を選ぶ。

1 Services - I	fdt v		10	mayalii Matau	ura • N. Vincinia	- Holp +
BC2 Dashboard Events Tags	Launch Instance	Actions - Instance Management Correct	r sch		C	• •
Instances Spot Requests	I Name 10 Inda I onpy 10 -	Get System Log Create Image (IBS AMI) Add(Ext Tags	Type c1 viorge	State strateg	Status Checks	Alarm Stat
Reserved Initiaties H Bindes AMs Bunde Talks H EUASTIC BLOCK STORE Volumes Snapsnocs		Charpe Scanagline's seed Bundle Instance Initiation state APRO Learnh March III Ma Charge Terminitian Production Wav/Charge User Data Charge Terminitian Spatial Charge Initiation Spatial Charge Initiation Spatial Charge Initiation Instalan Databa Nature Instalan Databa Initiation B D Addresses				
B NETWORK'S SECURTY Securty Orbups Bastic Ps Placement Groups Light References	Black Devices: Network Interfec Public DNS:	Actions Torminuto Reboot Silon Silon	0.420F14#5.00			
Ney Pars Private DN Network Interfaces Private In Secondary Launch Tie	Private DNS: Private IPS: Secondary Privat Lounch Time:	Cloudivatils Incentering Enable Detailed Monitoring Deable Detailed Monitoring Addition Alarms	s than an hou	Pro	duct Cedes:	

特集──ついにユニケージが AWS に登場 「AWS Tukubai」を試す

17) 先程ダウンロードした秘密鍵のパスを指定して 「Launch SSH Clinet」を押す。 すると Java アプレッ ト版のターミナルが起動する。(要 Java Runtime **Environment**)



18) ターミナルが起動し、ログイン成功。はじめまして!



19)「Setting」内の「Terminal」の設定をいじれば、 日本語表示にだって対応させることができる。

- 🗆 X

Soot€ip=10=244=113=244:[↑] [80x24] File Edit Settings Plugins Tunnels Help New Ser Connection. 🔝 Terminal Settings X Terminal... General Colors Misc VT 1 VT 2 Reset To D Terminal type xterm Ŧ Auto Save S Columns 80 Auto Load S Rows 24 Save Pass Encoding utf-8 -Font MS 明朝 -Size FA 独科書M . MindTerm hone: C:¥Users¥A Scrollback buffer FA 瑞筆行書M Initializing random sener Scrollbar position HGI'3-9-9E-PRO Connected to server runni MS ゴシック MS明朝 Server's hostkey (ssh-rs MS Pゴシック openash md5; cd;ca;d9;92 MS P明朝 bubblebabble: xunif-hefuv OK Cancel Last login: Sun Aug 10 1 [root@ip-10-244-118-244 ~____

これで、Linuxマシンとして普通に使えるぞ!

- 他の SSH クライアントからログインする場合 -16')手順 16 から……もちろん SSH やいつも自分 で使っているターミナルソフトでログインす ることもできる。先程の公開鍵、および先程 メモしておいた Public DNS のアドレスに対 し、root としてログインすればよい。



20) 最後に、仮想マシンの止め方。EC2 ダッシュボー ド画面にて、対象の仮想マシンを選択(チェック) している状態で、上部メニューの「Action」の中 の「Stop」を選べば止まる。止めないと1時間 毎に課金され続けるので注意。ただし、ディスク (EBS) 使用料は Stop しても課金され続ける。まあ 1GB あたり月 0.1 ドルではあるのだが……。これ も止めたい場合は、Stop ではなく Terminate を して、その後 EBS を Delete する。もちろんその 場合、ディスクの内容は破棄されるので注意!

🧊 Services v 🛛	dt v		Tar	royaki Matow	ra • N. Viroinia	т нарт
EC2 Dashboard	Launch Instance	Actions ~			C	• 6
Events Tags	Vering: All Instances	Brotance Management Connect	e tech.) R	((161/12	canas >
B INSTANCES	D Name to Insta	Get Windows Admin Password	Type	State	Status Checks	Nam Sta
Instances	R ampty Chief	Add/Eldit Tags	c1.xlorge	a running	😋 2/2 checks p	none
Reserved Instances		Change Source/Dest, Change Source/Dest, Change				
B INVOES		Laurch More Live This				
AMs		Clearsedute IP Address Change Termination Protection				
Bundle Talks		View/Change User Data				
RASTORIOCK STORE		Charge Shittante Type Charge Shittante Type				
volumes		Altach Natwork Interface				
Snapshots		Manada Envida 10 Addresses Sto	op なら	マシン	/停止、	
E NEWYORK & SECURITY		Actors To	minate	+	マシンの時	南
Security Orbups	Black Davider	Terminate	minau	14-0	< / / WX	禾
Batte Ps	Network Interfact	Rep				
Placement oroups	Public DNS:	and B	14201445.00R			
Kity Para	Private DNS:	Cloud/vatch Brontennig		Pro	fuct Codes:	
Network Interfaces	Private IPs:	Enable Detailed Monitoring				
	Secondary Privat	Addition Alarmo				
	Lautch time:		"S fear to you?			

AWS Tukubai を使い始める方法はおわかりいただけ ただろうか? 次章では、様々なデモを紹介しよう。

第3章—実力がよくわかる、Tukubaiデモ5選

「AWS に usp Tukubai が登場した」とはいうものの、その usp Tukubai (Tukubai)は一体、どんなこと ができて、どれくらい速いのか?そこで編集部では、今回独自に組んだベンチマークプや過去に収録し た Tukubai レシピ、更にはリリース元の USP 研究所が提供しているデモプログラムや教材を動作させ、 Tukubaiの実力の程を検証した。これらのプログラムは公開しているので皆さんも是非試してもらいたい。

🍍 準備. デモプログラムのインストール

AWS Tukubai の実力を体験するためにより抜いた デモプログラムを UEC サイトに用意した。まずは こちら(https://uec.usp-lab.com/201309uspmag-sample.tgz)をダウンロードしてもらい たい。(ただし、作成したばかりの AWS Tukubai 仮想 マシンにはダウンロードを行うための wget コマンド が入っていないため、先にインストールしておく)

\$ yum -y install wget + \$ wget https://uec.usp-lab.com/201309-uspmag-sampl e.tgz +

デモ1.帳票を作成する

まずは小手調べ……、というわけでもないが Tukubai コマンドの基本が短く凝縮された例を紹介したい。

解凍したディレクトリの中にある SALES_REPORT というディレクトリに入ってもらいたい。入ったらそ こに、report というファイルがあるのでこれを実行し てみよう。

\$ cd *TukubaiDemoディレクトリ*/SALES_REPORT ↓ \$./report ↓

これは同じディレクトリにある、商品原価・売価マ スタ (PRICE)、売上 (SALES)、部門マスタ (CATEGORY)、 部門名マスタ (CATEGORY_NAME)の各ファイル (テー ブル) に基づいて右のような帳票を出力するものだ。 SQLの世界で言うならビューに相当するし、その場合 は SELECT 文を使えば恐らく同じことができる。

実行を終えたら、reportのソースコードを眺めてみ てもたらいたい。15個のコマンドがパイプで繋がれ、 工場の製造ラインのごとく逐次実行されるようになっ ていることがわかる。やるべきことは上から下へと素 直に書かれ、各行も人間にとって理解し易い処理単位 これを解凍し、生成されたディレクトリの直下にある INSTALL.sh を実行すれば準備完了だが、一つだけ注意!

\$ tar jxf 201309-uspmag-sample.tgz + \$ cd TukubaiDemo + \$./INSTALL.sh +

リスト1.売上データに基づく帳票出力結果

Web アプリのデモを含んでいるため、前述の INSTALL.sh は Apache の設定 (httpd.conf) とファイア ウォール (iptables) の設定を書き換える。それらを既 に弄っている場合は、INSTALL.sh の冒頭に記されて いる説明に基づいて、手動でインストールを行う。

売上レポート 3 部門 種別 数量 売上 粗利 利益率% 4 ================ ======== ======= _____ **5** 001 野菜 327, 537 175, 521 40.401 23.0 6 002 果物 255, 235 144, 118 33, 202 23.0 7 003 339, 523 195, 023 41,661 21.4 魚類 8 004 肉類 114, 795 72, 395 12, 906 17 8 9 005 調味 217, 198 81, 404 20, 373 25 0 10 006 雑貨 115, 552 46, 375 10, 695 23.1 11 007 パン 49, 472 38, 238 6,614 17.3 12 008 冷凍 72,660 28, 241 7, 173 25.4 13 010 飲料 125, 012 72,688 17,021 23.4 14 011 酒類 24, 259 12, 075 3, 165 26.2 15 ==== 16 @@@ 22.3 **** 1, 641, 242 866, 078 193, 210

になっている。このスタイルこそが Tukubai プログ ラミングの真髄なのである。

また、その**実行に要した時間にも注目**してもらいた い。売上データは約300万行あるが、c1.xlarge インス タンスなら約3秒で処理が終わるのである。同じデー タの処理をリレーショナルデータベースでやったとし たら、果たして同程度の時間で終えられるだろうか。

10 USP MAGAZINE 2013 autumn

特集──ついにユニケージが AWS に登場 「AWS Tukubai」を試す

Fデモ2.1000 万件データ処理ベンチマーク

次は Tukubai コマンド1つ1つの処理能力を評価 するためのベンチマークをしてみよう。このデモは、 解凍したディレクトリ直下の BENCHMARK という ディレクトリの中に用意されている。

\$ cd *TukubaiDemoディレクトリ*/BENCHMARK 🛹

テストデータを生成

最初に、ターゲットとなるテストデータを生成しよう。まずは次のプログラムを実行してしばし待とう。

\$./make_TESTDATA_ja.sh 10000000 > TESTDATA 🚽

引数の0の数は7つ、つまり1000万だ。これで TESTDATAというファイルに1000万行からなるラン ダムなデータが生成される。

データは、4列で構成されており、1行目から順に、 id(受付番号)、pref(都道府県名)、balance(収支)、 date(日付)である。ベンチマーク用のデータなので、 あまり深い意味は無いことをご了承いただきたい。

基本中の基本「列の抽出」

全データの中から東京都の経費列のみを抜き出し てみよう。SQL 的に言えば SELECT balance FROM TESTDATA WHERE pref = '東京都'; 的な作業だ。

\$ time	cat TESTDATA	selr 2 ′東京都′	self 1 > 0
UTPUT 🗸]		
real	Om1.182s		
user	Om0.694s		
sys	OmO. 490s		

編集部で試したマシンは c1.xlarge (Xeon E5410 2.33GHz、8 コア)であったが、結果はご覧のとおり 2 秒足らずであった。

抽出するだけでなく、もちろん値の加工もできる。 例えば上記の値の合計を求めてみよう。これも SQL 的に言えば SELECT SUM(balance) FROM TESTDATA WHERE pref = ' 東京都'; 的な作業だ。

\$ time	cat TESTDATA selr 2 ´東京都´ self 1	t
r −d ,	sm2 0 0 1 1 comma 1 > OUTPUT 🚽	
real	Om1.188s	
user	Om0.724s	
sys	Om0. 492s	

CPU コアが8つのマシンであるため、trやsm2コ

マンドといった CPU に相応の負荷をもたらすコマン ドが2個増えても並列実行で吸収できるということ であろう。コマンドを実行するのに要した CPU 時間 の合計値(2行目)は増えているので、仕事量はきち んと増えていることがわかる。

「ソート」の威力が凄まじい

ソートもまたビジネス版 Tukubai の威力が発揮される部分だ。msort という強力なソートコマンドが用意されており、これはソートをオンメモリで行う上に、 マルチコア対応。早速試してみよう。

サンプルデータを、第一に4列目(日付)の昇順、 第二に2列目(都道府県名)の昇順でソートしてみ よう。SQL的に言えば SELECT * FROM TESTDATA ORDER BY DATE ASC, PREF ASC; である。

\$ time msort key=4@2 TESTDATA > OUTPUT -

-p8というオプションを付ければ8コアで処理する。

\$ time msort −p8 key=4@2 TESTDATA > OUTPUT 🚽

結果を表3にまとめた。1000万行ソートが13秒 で終わるというのは驚異的だと思う。ちなみに標準で 入っている GNU版 sort コマンドにも全く同じ処理を させてみたが、その速度差93倍!!…雲泥の差だった。

C 言語の鬼は伊達じゃない

Tukubai コマンドは、C 言語熟練者の手で徹底的に チューニングされているといい、ご覧のとおり速い。

コマンドのヘルプが man2 コマンドで見られるの で、他のコマンドもベンチマークテストしてみるとよ いだろう。ヘルプは、環境変数 LANG に ja_JP.UTF-8 を設定しておけば日本語版になる。例えば sm2 とい うコマンドの日本語 man は次のようにして見られる。

\$ export LANG=ja_JP.UTF-8

表3.1000万行ソート対決!

GNU 版 sort	msort	msort(8パラ)
20分16秒98	22秒75	13秒15

c1.xlarge (Xeon 2.33GHz、論理 CPU 数 8) 上での比較

■ デモ3. ルービックキューブと帳票、宝くじ

次に紹介するデモは、本誌 Vol.5 と 2012 autumn で 紹介した、Open usp Tukubai 用レシピ 3 本だ。AWS Tukubai は、Open 版のものも当然動く。しかも速い!

ルービックキューブシミュレーター

1つ目は色鮮やかなルービックキューブシミュレー ターの紹介だ。解凍したディレクトリの中の RUBIK_ CUBE というディレクトリに入り、rubik_cube という シェルスクリプトを実行してみよう。

\$ cd *TukubaiDemoディレクトリ*/RUBIK_CUBE ↓ \$./rubik_cube ↓

すると**画面1**のように、正面、上面、右面、背面、 底面、左面の現在の色配置が表示される。ここでそれ ぞれの頭文字(F,U,R,B,D,L)をタイプするとその面が 90°回転する。小文字ならば反対方向に回転する。プ ロンプトにある英字の羅列は答えを示しており、一手 目からその通りにタイプすると色が揃う。(終了は"q")

このデモは Tukubai が**業務システムのみならず、** 数理操作にも活用できることを示している。是非ソー スコードも見てその様子を実感してもらいたい。

部門トレンド(帳票デモその2)

2つ目は帳票デモだ。本章の最初にデモした帳票よ りもさらに複雑である。解凍したディレクトリの中の BUMONTRENDというディレクトリに入り、同名の シェルスクリプトを、とにかく実行してもらいたい。 この時、ターミナル画面の文字数は縦横共に大きめに しておくのがおすすめである。

```
$ cd TukubaiDemoディレクトリ/BUMONTREND <del>(</del>
$ . /BUMONTREND <del>(</del>
```

画面に出力されるのは商品部門・地域別の売数推移 の帳票だ(写真1)。ここまで綺麗に表示できるなら、 大掛かりな帳票エディター等使わずにプレーンテキス トで出して印刷するので十分に思えてこないだろうか。

宝くじ番号照会 Web アプリ

3 つ目は、ちょっとイジワルなジャンボ宝くじ当選 番号照会 Web アプリだ。まずは、次のように打ち込 んで、LOTTERY ディレクトリの中の LOTTERY.DB と いうシェルスクリプトを実行してもらいたい。





写真1. 画面に出力された帳票を紙に印刷



画面2. 宝くじ当せん番号照会アプリ

\$ cd TukubaiDemoディレクトリ/LOTTERY/SHELL →
\$./LOTTERY.DB →

これは宝くじの公式ページ HTML から、当せん番号の抽出を行うものである。終わったら次に、Web ブラウザーから次の URL を開いてもらいたい。

http://*第2章の手順15で得られたアドレス*/TukubaiDemo /LOTTERY/CGI/LOTTERY.CGI

すると画面2のように当せん番号照会画面が表示される。ここで、実際のジャンボ宝くじの番号や組番号、 抽せん回を入力すればよい。Ajax でリアルタイムに、 現在までの入力で当たっている可能性のある金額が表 示される。大抵は、最後の桁まで入力し終えたところ でドボン(0円)になるという、イジワル仕様だ。

特集—ついにユニケージが AWS に登場 「AWS Tukubai」を試す

Final Field Fie

AWS Tukubai には 200 種類以上ものコマンドがあ るが、ここではページの許す限りそれらを紹介しよう。 まずは、解凍したディレクトリの中の SHORTDEMO ディレクトリに移動しよう。準備はいいかな?

\$ cd *TukubaiDemoディレクトリ*/SHORTDEMO 🚽

コマンド 59 連発!

★ CSV ファイルを読みたい!…sed,AWK でも頑張れ ばできるが、fromcsv なら超簡単。値として改行やカ ンマ、引用符("")が含まれてても大丈夫。サンプル の sample.csv を生で見てから、fromcsv に掛けてみよ。

```
$ cat sample.csv → テまずは元の中身を確認
$ fromcsv sample.csv → ←そしたら使ってみよう!
```

★ Excel ファイルを読み書きしたい!…確かに Excel ファイルがシェルスクリプトで一気に弄れたらさぞ便 利だろう。それなら rexcelx と wexcelx。第1引数か ら順に、シート番号、セル範囲(左上)、セル範囲(右下)、 ターゲット Excel ファイルと指定すれば OK。試しに、 AWS Tukuabi 価格表を記したサンプル pricelist.xlsx を 読んでみよ。

```
$ rexcelx 1 B5 F20 pricelist.xlsx | keta
見難いので桁揃えしている↑
```

1番目のシートの、B5~F20の範囲にある値を抜 き出した。「本当に合ってる?」と思うなら、元の pricelist.xlsx を Excel で開いてみればいい。

★バイナリコード混じりファイルに困っている?…そ うそう、HTTP POST で届くデータや破損したテキス トファイルでたまにそういうことがあって、表示が乱 れて厄介だ。そういう時は utf8nude。これに掛ければ、 UTF-8 に準拠しないバイナリデータを除去する。

試しにビープ音を鳴らすコード(0x07)を混ぜたサ ンプルテキスト beep.txt を utf8nude に掛けてみよ。

\$ cat beep.txt → ←まず元の中身を確認(音が鳴る) \$ cat beep.txt | utf8nude → ←これは鳴らない!

★余計なことしない sed…例えば、テキストファイル NAME_TMPL.txt 内の "##NAME##" の部分をシェル変 数 a に代入された文字列に置換したいと思った場合、

\$ cat NAME_TMPL.txt | sed "s/##NAME##/\$a/g" 🚽

と書くのはアウト!ナゼなら、aに "/"や "&" や "\"が

含まれていたら誤動作するから。もちろん予めエス ケープしてやればいいんだけど、なんて面倒臭い!そ んな時には calsed が超便利。grep に対する fgrep み たいなもので、文字に対して一切特殊な反応をしない。 先の例ならこう書けば解決。あぁ便利、しかも速い!

\$ cat NAME_TMPL.txt | calsed "##NAME##" "\$a"]

★ SQL ライクに列を指定したい…リレーショナル DB から乗り換える時、気になるのは列(カラム)の指定 を基本的に列番号で行う点だろう。「列名で指定でき ないの?」実はできる! Tukubai には tag で始まるコ マンド群(現在 51 個)があり、それらを使えば OK。 対象となるデータの1 行目を列名と見なし、その名前 で指定できるのだ。例えば self コマンドに対応するの は tagself コマンド。試しに ID、姓、名の3 列からな るサンプル名簿 members.txt から姓だけ取り出そう。

\$ tagself "姓" members.txtᆗ

★このマシンの CPU は何コア?…並列処理が得意な Tukubai を使いこなすなら知っておきたいところ。で も、どうやって知ればいいのか。知っててたとして も調べ方は OS によってバラバラで面倒。それなら corenum コマンド。実行すれば素直にコア数を返す。

💲 corenum 🚽

★**巨大ファイルの分割**…ローテーションをさせ忘れて 巨大なログファルイができてしまった! とりあえず 10000 行毎に分割したいと思った時、さあどうやる? while 文!? いやいや多分遅くて日が暮れる。そんな時 は gyocut コマンド。下記のように書け、動作も速い!

\$ seq 1 100000 > many. txt → ←10万行のfile作成 \$ gyocut -10000 few. %02d many. txt → ↑ few. 01~10という10個のファイルに分割される

★ファイル末尾の改行コード…これが有ったり無かっ たりすると煩わしいので無ければ付けたいと思うこと がある。そんな時は kaigyo コマンド。たったそれだ けのためのコマンド?…そう。でも結構便利よ。

\$ echo -n hoge → ←これは鬱陶しい!
\$ echo hoge | kaigyo → ←kaigyoコマンド付ければ
\$ echo -n hoge | kaigyo → ←これも鬱陶しくない。

デモ 5.サンプル Web 帳票システム「USP ストア」

																	現在時間)) 2013章	¥9月1	888	ロダインID: us 3 16:40:34 UT/
PLU⊐-	ード検索	5																			
19.4 J	プレミアムミ	チーズケ・	-+		化水田	i (a		500 ED	PI II-	to Dat	169769	369636				D (*				-	7
100-0					1 VBC/T			00011	FLOUR	-1-: [043	153702.	333133				019	- L			_	_
5品コード <mark>0</mark>	49537623	53135			代表表	i 🗇		780円	比較日	前日	-					1	検索	牛蛇	セル		
lét-	2010年0	28110		·表:	示胤습:清	いた	na ini	。 吊子は個	[計. 和利	⊯· 個3	æ•浦伯	と率は96									-
191	2010年0	3月10日		・折	子(折退于	趣願)J. 200	9年6月	7月音E	仮蔵	の用す	38P%	の折返時	劇の平り	均値です						
				·																	
												時間	-301 LS:	を情報							
			実績 累計	ELOXEI	Ħ	09 10	10 1	11 	12 3	13	14 15	15 16	16 17	17 18	18 9	19 20	20 21	21 22	22 23	23 24	
		売散	1086	1359	-273	101	148	158	150	172	154	203	-	-	-	-	-	-	-	-	
20	王	在庫設	24001	38272	-14271	4430	6154	9236	12434	16150	19020	24001	-	-		-	-			-	
頭	現在: 17:00 折子: 17:02	外知道	22/02	254	-24	2130	28.9	28.1	23.0	22.8	28.9	22.6	-			-	-				
31		储引率	1.8	1.6	-0.3	1.0	1.1	1.2	1.3	1.4	1.5	1.6	-	-	-	-	-	-	-	-	
		清化率	4.3	3.4	0.9	2.2	3.9	4.2	4.3	4.3	4.4	4.3	-	-	-	-	-	-	-	-	
	<u>北海</u> 現在: 17:00 折子: 16:23	売載	30	39	-9	2	4	4	2	6	- 5	- 7	-	-	-	-	-	-	-	-	1
463		在庫設	60	618	-558	88	84	80	78	72	67	60	-	-	-	-	-	-	-	-	
锁		Price T	15.0	24.1	-648	25.0	15.0	0.0	25.0	15.0	0.20	25.0	-			-	-			-	
折		(43)14	23.0	24.1	-0.2	20.0	20.0	20.0	23.0	20.8	25.6	25.0	-			-	-			-	1
		清化中	33.3	5.9	27.4	2.2	6.7	11.1	13.3	20.0	25.6	33.3	-	-	-	-	-	-	-	-	1
		売赦	17	21	-4	0	3	5	2	2	2	3	-	-	-	-	-	-	-	-	
		在庫設	-4	272	-276	13	10	5	3	1	-1	-4	-	-	-	-	-	-	-	-	
<u>-</u>	<u>査査</u> 現在: 17:00 折子: 16:22	納品予	0	288	-288	0	0	0	0	0	0	0	-	-	-	-	-	-	-	-	1
		荒利率	25.8	24.4	1,4	-	25.8	25.8	25.8	25.8	25.8	25.8	-	-	-	-	-	-	-	-	1
	TS 10/22	The second second						100 March 100 Ma													4

最後に紹介するデモは、Web ブラウザーから帳票 を閲覧する業務システム「USP ストア」である。

このアプリケーションは、シェルスクリプトによる システム開発方法を学ぶために制作するサンプルアプ リケーションであり、**USP 研究所の「教育講座」の K3 クラスのカリキュラム**になっている。(興味のある 方は、本誌 2013 summer 号を参照されたい) 今回は 特別にそのプログラムを収録させたもらった。

デモの使い方

Web ブラウザーから次の URL を開いてもらいたい。

http://*第2章の手順15で得られたアドレス*/TukubaiDemo /USPSTORE/CGI/JIKANTAI.CGI

すると本ページ冒頭のような画面が表示される。 この画面は、PLU(プライスルックアップ)と呼ば

Name	Last modified	Size Description	
Parent Directory		-	
20090301.DATA	07-Dec-2011 00:45	5.7M	
20090301 TIME	07-Dec-2011 00:48	600	
20090302 DATA	07-Dec-2011 00:45	5.6.1M	主 し つ ち か の
20090302.TIME	07-Dec-2011 00:46	3 600	売上ナータ寺の
PLATAD ENERGY P	02-0-2011-00-4		生データファイル
PLU GENBAI	07-Dec-2011 00:48	3 12K	
PLU NAME	07-Dec-2011 00.46	6 8.6M	
TEN BUMON OHOL	IR 07-Dec-2011 00:45	5 21K	
TEN NAME	07-Dec-2011 00:46	3 576	
TODAY.DATA	07-Dec-2011 00:4	5 6.9M	PLU マスタテーク
TODAY.TIME	07-Dec-2011 00:46	3 600	ファイル

画面3.各種生データを見られる

れる商品コードを入力し、時間帯別販売状況に関する 帳票を閲覧するものである。PLUを指定する他、閲覧 した日付の指定、及び販売状況を比較したい過去の日 付を設定するようになっている。

従ってこのシステムは、まず各商品の PLU コード を知らなければ検索できないのだが、何も入力しなけ れば「PLU コード=04953762353135、プレミアムチー ズケーキ」が自動的に入力されるようになっている。

また用意されている売上データもサンプルであるため、ここには 2009 年、2010 年の 3 月頃のものしかない。そこで、次の URL ヘアクセスすることで生データを見られるようにしてある。(画面 3)

http://*第2章の手順15で得られたアドレス*/TukubaiDemo /USPSTORE/POMPA/

これを見ながら、様々な PLU コードや日付を入力 した場合の動作を見てみるとよいだろう。

個性が強いが万能な Tukubai

このように Tukubai は、データベースアプリを凌 ぐデータ管理、帳票作成、数理計算、そして Web ア プリケーションまで何でもこなす。

また、ソースコードや格納データを覗けばそれがい かに実現されているかもわかるだろう。個性は強いが、 何でもこなす Tukubai に、是非親しみを感じてもら えればと思う。



アプリの開発の目的は「情報の収集 と共有化」だ。例えば、日頃の商品販 売情報を収集する事で、売れ筋商品や 季節変動がわかる。経費削減や業務改 善案を共有できれば、社内全体に利点 がある。さらに、営業マンが営業ノウ ハウを共有すれば、営業力の強化にな る。「情報」を資産と見て、情報資産 を運用する道具としてアプリがある。

ところで Linux には、無料のプログ ラム言語やアプリ開発環境がある。こ れらを使わないのはもったいない。そ こで今回はアプリ開発の話を書く事に した。

グループウェアと Web 掲示板

2000 年、Linux でサーバーを構築 した。丁度その頃、「ナレッジマネー ジメント」という言葉が流行っていた。 情報共有化の事で、それを実現させる のはグループウェアだという。IT を活 用して、業務改善や企業業績が向上す れば、IT 先進企業として雑誌に掲載さ れる。そうなれば、私は有名人になれ るかもしれない。

そんな欲求も働き、まずはサイボー ズのお試し版を Linux サーバーにイン ストールしてみたが、ほとんど利用さ れなかった。私のいる本社では、社員 全員が同じ部屋にいるため、連絡事項 や会議室予約機能などは、ホワイト ボードやメールで事足りる。敢えてグ ループウェアを使う利点がなかった。 利点が示せない状態では稟議を挙げて も却下されるし、実際、却下された。

最初からグループウェアでは敷居が 高い。そこで予算をかけず、かつ、気 軽に使ってもらえる物から導入する事 にした。まずは Web 掲示板を思いつ いた。掲示板を使って、本社や営業所 にいる社員同士が、経費削減や業務改 善を提案し合う事で、情報共有ができ る。そして便利さを知ってもらう事で、 利用する人が増えるかもしれない。す ぐに無料の Web 掲示板の CGI プログ ラムをダウンロードして、使用するこ とにした。最初は書き込み自由で行 なったので、仕事以外の世間話や、面 白い内輪ネタを書いてくれる人が何人 かいた。それが功を奏したのか、読み 書きする人も増えてきた。

しかし、あまりにも度が過ぎる書き 込みが出てきたため、業務以外の内容 が禁止になり、結局、掲示板は使われ なくなってしまった。

全員が節度を保ちながら、利点を感 じつつ、利用しやすい環境を整えなく てはならないという、難しい課題にぶ つかってしまった。

役員予定表

営業所からの電話で「〇〇本部長の 予定は?」という問い合わせがある。

本社の壁の張り紙に役員予定表が あったが、社内向け Web で閲覧でき たら便利だと思った。

しかし、そんな物はネットで無料配 布していないので、自作アプリの挑戦 になった。CGI のサンプル集を見ると Perl が多かったので、Perl の本を片手 に見よう見まねで、自作 CGI を作成 した。

だが、実際に使いはじめると、予定 表を作成している担当者から「使い勝 手が悪い」とか「張り紙も作っている ので、二度手間」と言った意見が出て きて、長続きしなかった。現在でも張 り紙が続いている。

PostgreSQL と PHP に 出会う

2001 年、オープンソース・データ ベースの PostgreSQL と出会う。

データ検索システムが構築できると 思った。しかし、PostgreSQLとWeb と連動させるためには、仲介するプロ グラム言語を学ぶ必要があった。

ところが、シェルスクリプトや Perl のサンプル集がなかった上、Java は サンプル集を見た瞬間、難しいと思い、 及び腰になってしまった。

そこで、メーリングリストでお勧め の言語を聞いたところ、「PHP が手っ 取り早い」と勧められた。

PHP はオブジェクト指向言語だ。 だが実際には、オブジェクトの概念を 知らなくても問題はなく、手続き型言 語として扱えるため、初心者でも触り やすい言語だ。特に、一度でもC言 語を勉強した人なら、覚えやすい。

まずは商品名検索システムから作る 事にした。基幹業務システムは AS400 で運用している。

以前から商品コードブック代わり に、AS400 上や Web 上で商品検索で きれば便利だと思っていたが、当時の AS400 では高価なミドルウェアが必 要であり、予算面で不可能だった。

そこで、AS400の商品マスターを CSVファイルに落とし込み、それを PostgreSQLに格納した。

Web上で手軽に、商品名の一部か ら商品コードを検索したり、反対に商 品コードから商品名の検索が可能に なった。コードブックから探す手間を 省けるようになり、社内での評判は良 かった。

Web データ検索システム 構築

勤務先では、あるデータを顧客に提 供していた。顧客から電話で問い合わ せがある度に、データー覧表からデー タを探して、FAX で送っていた。

1日40~50件の問い合わせがあ り、処理に手間がかかっていた上、 365日対応を謳っていたため、休日は 持ち回りで電話当番をしていた。私も 当番に入れられていたので、休みたい のに休日出勤をした事が何度もあっ た。そこで Web 上でのデータ検索シ ステムを作る案件が浮上した。

当初、外注する予定だったが、私は 「折角、PostgreSQL や PHP があるの にもったいない」と思った。そこで外 注依頼する前に、検索システムの原型 を作れば良いと考え、突貫工事で作成 した。

自社開発すれば経費削減になるの で、褒めてくれるかと思いきや、逆に 上層部から「勝手な事をするな」と言 われてしまった。しかし、既に原型が できてしまったため、私が作った物が 採用された。

検索に使うデータは、データベース 化されていなかった。そのため、デー タベースに格納するための入力作業が 必要だった。担当部署の同僚が、目を 赤くしながらデータ入力や確認作業に 追われていたが、データベース化する 事で、管理しやすくなった。

検索方法、データの規格などの関係 で、担当部署の意見を聞きながら検索 システムの改良を繰り返し、完成させた。

ところが、Web 検索サービスを開 始したものの、利用者は伸び悩んだ。

考えられる原因は、当時パソコンを 使っている顧客が少なかったこと、さ らに電話だと「〇〇のデータをくれ」 で済むが、検索システムだと、パソコ ンを起動させ、Webに接続し、デー タ検索するという手間がかかること だった。

顧客は利点がないと使ってくれない。この当たり前な事に気づき、携帯でも閲覧できるようにしたり、電話でのデータ問い合わせがある度に、ネッ

ト検索可能の宣伝チラシを FAX で配 布したりした。

その結果、ネット検索する顧客が増 え、当番制で行なっていた休日出勤を 廃止する事ができた。

カレンダー式掲示板の導入

外出予定や来客予定などの連絡事項 は、メールが主流になると、メールの 整理を行なっても、見落とす問題が出 てきた。他にも、複数人にメールを送 る際に、送り漏れも発生する。

そこで、各人がカレンダー式の掲示 板に予定を書き込めるようになれば便 利だと考えた。

カレンダー式の掲示板を探してみる と、PHP で作られたプログラムを発 見した。しかし、そのままでは使えな いので大幅に改造した。

導入した結果、連絡事項や社員の予 定がカレンダーに書き込まれるように なり、各人の予定が把握しやすくなっ た上、メールの見落としなどが減った。 まさに情報の一元管理と共有化の効果 だった。

AS400 との連動

2006 年、基幹業務で使っている AS400 を買い換えた。その時、AS400 に接続する Client Accessの Linux 版が IBM のサイトで無償配布され ているのを知った。ODBC 接続で AS400 と Linux が連動可能になるた め、ODBC を活用すると、Web 上で AS400 のデータ検索ができる。(図 1)



図1 web からの AS400 のデータ検索





図 2 PDF 帳票生成

これによって高価なミドルウェアの 購入の必要性がなくなった。しかし、 ClientAccess について日本語の資料が ない。英語と格闘になった上、設定の 方法は手探りだった。七転八倒の末、 連動が可能になり、わざわざ CSV ファ イルに落とし込み、PostgreSQL へ格 納する手間もなくなった上、更新し忘 れによるデータの不整合の問題が解消 された。

現時点でのデータ閲覧が可能になる 事で正確な販売履歴などが検索・閲覧 できる上、伝票検索を行なう事で伝票 漏れや記入間違いを発見しやすくな り、在庫管理にも役立った。

AS400内のデータを補完する形で PostgreSQLを使う事で、顧客情報の 書き込みなどを行なえるようになった。

また、PHP の無料 PDF 生成ライブ ラリの FPDF を使う事で、AS400 の実 績データを基に PDF 帳票作成も可能 になった。AS400 で帳票を作成した 場合、ファイル化するためのツールは 高価だったため、紙で印刷する必要が あった。だが、PDF 帳票生成が可能 になった事で、メールでの実績表の配 布が可能になった。(図 2)

Web アプリの注意点

Web アプリの場合、外部の不特定 多数の人が接続できるため、セキュ リティー対策は必須になる。私自身、 最初は Web アプリの脆弱性を知らな かったため、何ら対策を行なっていな かったが、Web アプリの脆弱性のセ ミナーや本を読んで以来、可能な限り、 対策を取る事にしている。

主な対策法を2つ挙げてみた。1つ 目はクエリーの無害化だ。無害化対策 をしていない掲示板で、書き込み中に、 HTML 文の終了を意味する </hrml> タグを記入し、それをサーバーに送る と、掲示板が尻切れトンボになる場合 がある。これは、掲示板に書き込んだ </html> を、ブラウザが HTML 文の 一部と判断するからだ。掲示板が途中 で表示終了になる程度なら笑い話で済 むが、場合によってはタグ記号を悪用 したクロスサイトスクリプティングと いう問題が発生するおそれがある。対 策方法は、タグで使われる「<」「>」 などの文字を、「<」や「&qt;」といっ た特殊文字に変換する処置を施す事 だ。幸い、PHP には変換関数がある ので、それの活用をお勧めする。

2つ目はデータベースと連動させる 場合の、SQL インジェクト対策だ。

もし、何の対策も行なっていなけれ ば、SQL文を含んだクエリーを送り込 まれ、データベースの中身を覗き見さ れる危険性がある。「」の文字を「"」 に変換するなどして、SQLインジェク トを防止する必要がある。

活用されなかったアプリの話

アプリ開発をしたものの活用されな かった例はいくつもある。共通した原 因は、使う側が利点を感じなかった事 だ。上層部の指示で、簡単な Web 営 業日報や、Web 目標管理表などを作っ たのだが、あまり活用されなかった。 業務命令で、営業マンが日報への書 き込みを行なっても、部門長といった 閲覧・評価する側が、それに対して何 らかのコメントをしない限り、営業マ ンは「読んでくれていない」と思うし、 Web 日報以外にも、紙で営業日報を 書いていたら、二度手間になるため、 活用する気が失せてしまうという結果 になった。つまり、閲覧・評価側の立 場の人が、営業マンと対話するという姿 勢でコメントをする必要がある上、既 存の手書き日報の廃止という事をしな ければ、使われないシステムになって しまう。

シェルスクリプトについて

私の勤務先では基幹業務をAS400 で稼動させている。データをLinux上 のCSVファイルへ容易に落とし込む 仕組みがないため、残念ながらシェル スクリプトで業務アプリを作る機会が ない。だが、シェルスクリプトは手軽 で便利な道具だ。ファイルそのものを データベースとして扱い、コマンドで 必要なデータの取り出しや集計ができ る。Linuxの勉強会で、シェルスクリ プトを使って家計簿を作っている主婦 に会った事がある。

私自身、業務アプリではないが、 Linuxのログを見たり、ログの集計を 行なうのにはシェルスクリプトを使っ ている。

機会があれば、シェルスクリプトで 業務アプリを組んでみたいと思う。





司会:『もっと吹く』編集長・みかん

皆様こんにちは。今月もシェルスクリプト大喜利(略 して sh 大喜利)のコーナーがやってまいりました。

秋といえば季語的にも虫の声。時代と共に自然が減っ てあまり色々な虫の声が聴けなくなったと嘆くことな かれ。今はインターネットで、実に様々な虫の声が聴 ける時代でもあります。「コオロギ鳴き声」で検索した らコオロギ類の鳴き声をまとめているサイトを見つけ たんですが、コオロギと名の付く虫の声が28 種類もあ りました。どれも聞き覚えがあって「アレとかソレも、 みんなコオロギだったのか」と驚かされました。

さて、これまた驚きの連続、シェルスクリプト大喜利。 まずは本コーナーのご説明からいきましょう。

シェルスクリプト大喜利とは

シェルスクリプト大喜利特有のルール

- 、sh 大喜利はクイズやテストではありませぬ。なの で決まった答えというものはないのです。あえて 言うなら面白いスクリプトが正解!
- ニ、面白いスクリプトとは、例えばこんなもの。
 - イ、人が考えつかない意外性がある
 - **ロ**、<u>美しい、芸術的、記述がシンプル、高速、など</u> **ハ**、アイデア・こだわりが光る
 - =、ネタになるバカバカしさ、くだらなさがある
 - など。でも最後のは段位強制返還の恐れありよ。:-)
- ミ、スクリプト動作環境はLinuxとし、基本的に Linux JM(http://linuxjm.sourceforge.jp/)に記載されているコマンド及び機能のみ使用可能とします。これは多くの人が楽しめるようにするためなのです。(JMにあるので、Cシェル系での解答もOK! あとksh、2shもOKにひまひた)
- 四、sh 大喜利はシェルスクリプトを披露する場なの
 で、Perl や Ruby、Python などは使っちゃダメ

です。そもそも JM にも載っていません。逆にシェ ルスクリプトにとって不可欠な awk や sed 等は OK です。JM にもありますし。でも、よっぽど面 白ければ、Perl とかもなきにしもあらず??

五、Open usp Tukubai(http://uec.usp-lab.com/) も 使用 OK ! 但し、使う意義がそれなりに感じら れないと採用はキビシいですよ〜。

ルールもおさらいしたところで、さあ始めましょう!

本番開始

<第一問>

7 ~ n (nは引数で指定)の自然数の中に存在する『ス ミス数』を全て求めるシェルスクリプトを書いてください。

すっかり恒例になったした第一問の整数数列問題。 えぇ~、今回で5回目でございます。

スミス数ってのは、元の数の各桁の数字の和と、素 因数分解してできた数字の各桁の数字の和が同じにな るものを言うんだそうです。例えば、666。各桁の和 は 6+6+6 で 18。一方素因数分解すると 2*3*3*37 な ので、2+3+3+3+7 でやはり 18、というわけです。

さて今回はまず、非常にオシい解答からご紹介。

◎おととさんの解答

1	#!/bin/sh
2	for i in \$(seq 1 \$1); do
3	n=\$(factor \$i sed ´s/¥([0-9]¥)/¥1 /g´)
4	[\$(plus \${n%:*}) -eq \$(plus \${n#*:})] && ec
	ho \$i
5	done

Tukubaiには、plusという、与えられた引数の総和 を返すコマンドがあるんですけど、それを使ってくれ てます。factorの返す元数と素因数のアラビア数字を 1文字ずつ分割し、元数と素因数の区切りである":" の両端で分けて plus コマンドにそれぞれ流し込んで 和が等しいかどうか見てるのです。

いやぁ、plus なんてコマンドがあるのによくぞ気

が付いてくれた! エラい、……だけどこの解答、不 正解なんだなぁ。**スミス数は素数を含めない**のだ。理 由は素数はそれ以上素因数分解できないから。なので、 factor コマンドの後に素数を除外する1行が必要なの だ。Tukubai コマンド使ってくれてるのでオマケした いところではあるけど、残念ながら**路位なるす**。

◎熱ゃ!お茶!さんの解答

1 #!/bin/sh

2 for n in \$(seq 1 \$1); do f=\$(factor \$n); echo \${f
 #* } | fgrep ´ ` >/dev/null && [\$((\$(echo \$f | t
 r : `¥n´ | sed ´1s/¥([0-9]¥)/¥1+/g´ | sed ´2s/ *¥
 ([0-9]¥)/-¥1/g´ | tr -d ´¥n´ | sed ´s/+-/-/´))) =
 `0´] && echo \$n; done

投稿者コメント「ワンライナーを目指してみました。 for 文は消せませんでしたが」ということです。コー ドを読んでみると、factor が返す文字列の":"の前後 で行を分け、数字を1桁ずつ分解すると共に、上の 行には "+"を付け、下の行には "-"を付け、元に戻し て全部足し合わせて0になるかどうかを見てますね。

うーん、これは面白いやり方だね。ワンライナーら しいごちゃごちゃ感。そしてちゃんと事前に素数を候 補から除く処理もあるし、申し分ナシだ。よし、**一段 掻ち**。これで二段だ。

◎Kさんの解答

```
1 #!/bin/sh
2 g () { echo $(( $(echo $* | sed -e 's/¥([0-9]¥)/
  ¥1+/g′ -e ′s/$/0/′) )); }
3 seg 1 $1
4 while read i; do
    factor $i
6 done
7 while read j f; do
    i=${j%:}
8
    if [ -n "$(echo $f | grep ´ ´)" ]; then
9
     if [ $(g $i) -eq $(g $f) ]; then
        echo $i
12
      fi
13
    fi
14 done
```

続いては常連の K さん。いつもありがとうござい ます。さて、求め方は factor の返す文字列を ":" で分 けて、両側の数字の総和が等しいかどうか見るという のはやはり同様ですね。ただし、総和を求めるのがシェ ル関数になっているのが特徴的。

これは、熱や!お茶!さんの解答と比較して速かった。 前者は処理全体を一つのループで囲んでいるけど、こちらはループが小分けになっていて効率的に動くのかな? いいねぇ、**一路役与**。これで八段だ。 そして最後に今回の最優秀解答。

◎tnazuka さんの解答

```
1 #!/bin/sh
```

2 seq 1 \$1 | xargs -n 1 factor | tr -d : | awk 'NF> 2 {print \$1;print "X", \$1;\$1="";print "Y" \$0}' | se d '/^[XY]/s/¥([0-9]¥)/¥1 /g' | awk 'NR%3==1 {print ;next} {for(i=2;i<=NF;i++) {t+=\$i};print t;t=0}' | awk 'NR%3==1 {n=\$1;next} NR%3==2 {x=\$1;next} {if(\$1 ==x) {print n}}'

これは素晴らしい!! for、while 等のループ文ナシに 見事にワンライナー。しかも他を圧倒して速いです! コメントによると「元の数値を、各桁に分解されるも のとは別にストリーム内に保持することで、シェル変 数とループ文から解放されました」とのことです。

いやぁ、それにしてもこれは速い! 最初の factor の実行のところも xargs でやってしまってるし、そう やって**ループ文を無くすどこんなにも遠くなるのか!** これ は**三段接与**。それくらいの衝撃はあるね。これで六段!

\diamond \diamond \diamond

さて、つづいて第二問!問題文がちと長いですが。



3D プリンターも珍しくなくなって、時代は 3D ! UNIX コマンドも 3D 対応しなきゃ、というわけでこ のお題です。要素は 3³ 個だし回転方向は x-z 平面だけ、 と限定的なお題にしたんですが、それらを一般化した 解答まで届いちゃったりして、恐れ入りました。

◎tomi さんの解答

```
1 #!/bin/bash
2 col=' ¥7 ¥4 ¥1 ¥8 ¥5 ¥2 ¥9 ¥6 ¥3' # 90° 用
3 #col=' ¥9 ¥8 ¥7 ¥6 ¥5 ¥4 ¥3 ¥2 ¥1' # 180° 用
4 #col=' ¥3 ¥6 ¥9 ¥2 ¥5 ¥8 ¥1 ¥4 ¥7' # 270° 用
5 paste -d' hoge.* | sed 's/¥([^]*¥) *¥([[]*¥) *¥([^]*¥) *¥([^]*¥) *¥([^]*¥) *¥([^]*¥) *¥([^]*¥) *¥([^]*¥) *¥([^]**) *¥([^]**) *¥([^]**) *¥([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **([^]**) **(
```

なんとAWK版、sed版、Tukubai版の3つを送っ てくれました。どれも3ファイルを横(y方向)に 連結したうえで列を入れ替えるということをやっ ているわけですが、列を入れ替える操作は AWK や Tukubai(self コマンド) だと何の面白味も無くあっさ りできてしまうのであえて sed 版を採用しました。実 行するとカレントディレクトリーにある hoge.{1,2,3} を元に newhoge.{1,2,3} に回転後の値を書きだします。

うーん、アタクシの気まぐれなのかもしれないけど、 列の並べ替え後、新しい3つのファイルになる区切 り位置に2連続,3連続の半角スペースを入れてるあ たり、この sed 版はいいね。AWK のように列番号を ダイレクトに指定できない sed でも、これでウマい こと左、中、右で三分割してる。よし、**初路燈与**。

◎Kさんの解答

```
1 #!/bin/sh
2 for z in 1 2 3; do
   awk '{for(x=1; x<=3; x++) print x, NR, '$z', $x}'
   $1. $z
4 done
5 case $3 in
    "xz90" | "zx270") sort -k 1n, 1 -k 2n, 2 -k 3nr, 3 ;;
6
    "xz180"|"zx180") sort -k 3nr, 3 -k 2n, 2 -k 1nr, 1 ;;
    "xz270"|"zx90") sort -k 1nr, 1 -k 2n, 2 -k 3n, 3 ;;
8
    "yz90" | "zy270") sort -k 2n, 2 -k 3nr, 3 -k 1n, 1 ;;
9
10
    "yz180"|"zy180") sort -k 3nr, 3 -k 2nr, 2 -k 1n, 1 ;;
    "yz270"|"zy90") sort -k 2nr, 2 -k 3n, 3 -k 1n, 1 ;;
    "xy90" |"yx270") sort -k 3n, 3 -k 1n, 1 -k 2nr, 2 ;;
12
   "xy180"|"yx180") sort -k 3n, 3 -k 2nr, 2 -k 1nr, 1 ;;
13
14 "xy270"|"yx90") sort -k 3n, 3 -k 1nr, 1 -k 2n, 2 ;;
15 esac
16 awk 'BEGIN{x=1; y=1; z=1; out="'$2'. "z}
        {if (x != 3) {
18
          printf("%s ", $4) > out
19
          x++
        } else {
          printf("%s¥n", $4) > out
           x=1
           if (y != 3) {
24
            v++
           } else {
26
             v=1
             z++
             out="'$2'."z
28
29
          }
30
         }
       }′
```

来ました、一般化。これは回転軸を一般化してくれ てます。引数 1,3 で入出力ファイル名(拡張子無)を、 引数 2 で回転方向と量を指定するようです。コメン トによれば、更に X,Y,Z のサイズも一般化したかった そうですが、さすがにそれは難しかったそうです。

いやぁ、なかなかやるね! 回転を実現するやり方 も、3³個の要素を全て縦に並べ、回転の都合に合わせ てソートしてて面白い。よし、**一段積号**。おっと九段だ! そして、一般化した解答がもう一つ来ました。

◎C。H8O7 (クエン) さんの解答

```
1 #!/bin/sh
 2 f=${1%, *}
 3 n=$(ls $f.* | wc −l | tr −Cd 0−9)
 4 files=$(yes $f. | head -n $n | awk 'BEGIN{ORS="
   "}{print $0 NR}')
 5 case $2 in
     90) a=-1; b=-$n; c=$((n*n+1)) ;;
     180) a=$n; b=-1 ; c=0
                                      ;;
     270) a=1 ; b=$n ; c=0
 8
 9
    *)
          exit 1
10 esac
11 fields=$(awk —v n=$n —v a=$a —v b=$b —v c=$c ´
            BEGIN {
12
13
                 for (x=n; x>0; x--) {
                     for (y=0; y<n; y++) {
14
15
                         printf("$%d, ", a*x+b*y+c);
16
17
18
            }')
19 paste $files | awk '{print '${fields%,}'}' > $f.
   Λ
20 for file in $files; do
       fields=$(awk -v x=${file##*.} -v n=$n '
21
                BEGIN {
23
                     for (y=1; y<=n; y++) {
                         printf("$%d, ", n*(x-1)+y);
24
25
                }')
26
27
       awk ' {print ' $ { fields %, } ' } ' $ f. 0 > $ file
28 done
29 rm $f.0
```

こちらは K さんが難しいと言っていた X,Y,Z のサイ ズを一般化してます。ただし、回転方向は固定なよう で、第二引数で "90","180","270" を指定するみたいで す。また、こちらは第一引数で指定したファイルに上 書きするので 90 で 4 回実行すれば元に戻ります。

これまたいいねぇ!どうやってサイズを一般化して るのかと思えば、AWK で AWK のソースコードを作っ てるのか。よし、**一段複与**! これで三段だ。

この 2 人が協力したら完全な一般化バージョンも完 成するかもしれないな。いずれにせよ、これで UNIX コマンドの 3D 化も安泰だ。(ホント?)

 \diamond \diamond \diamond

さて最後、変り種の三問目ですよ。

<第三問>

rev コマンドって知ってますか?各行の文字位置を左右逆に するものなんですけど……。これの使い道を教えてください。

これ、皆さん使ったことありますか? アタクシ、 何に使ったらよいのかさっぱりわからなかったのでお 題にしてみたのですが、お蔭様で色々来ました。 **◎イタローさんの解答**

1 sl rev



コメント「rev を使えば忌々しい汽車を破壊できま す」……って、おいおい! **こっちの画面まで破壊され** るじゃないか!! それだったら、"sl | head -c 1" とかで瞬 殺する方がいいような……。いや〜けしからんなー。 これは久しぶりに**一段強制返還**だ! はい、初段ね。 ②**tomi さんの解答**

1 #!/bin/bash

2 expand \$1 | ycat -0 - <(sed 's/.*//g' \$1) | rev

こちらは、引数で与えられたファイルの文字の並び を左右入れ替えるものだそうです。単なる rev と違っ て、全行がちゃんと右揃えになります。ちなみに第二 問の Tukubai 版の応用でできてしまったそうです。

なるほど Tukubai の ycat を使うとこんなに簡単に できてしまうのか。ファイルの文字位置が反転して気 持ちーなー。しかも、この後更に rev に通せば元のテ キストを右揃えしたことになるじゃないか! よし、 **一段積与**。これで二段だ。

◎ゆきとさんの解答

1 echo "山本山" | awk ´ {print;print}´ | sed ´2s/¥(. *¥)/echo "¥1" | rev/e´ | uniq | wc -l | awk ´\$1>1 {printf "non-"} {print "palindrome"}´ これは回文 (palindrome) か否かを判定するワンラ

イナーだそうです。但し sed は GNU 版専用とのこと。 はい、回文判定への応用は、アタクシも来ると思っ てたんだよね。AWK の対応次第だけどマルチバイト にも対応してるし、いいね。**一路検与**。これで三段だ。

◎猿ニ号さんの解答

1	cat < <l< th=""><th>IST rev sort -k 2,2 rev</th></l<>	IST rev sort -k 2,2 rev
2	three, 3	This_is_the_third.
3	one, 1	This_is_the_first.
4	two,2	This_is_the_second.
5	LIST	

コメント「列の文字列を右から左にソートしたい場 合に rev は重宝しますよ。わりと有名では?」という ことで、恥ずかしながら「なるほど!」と思いました。

この解答は今回一番実用的なものかのしれないな。 もしかしたら rev コマンドの作者もこういう用途を想 定してたのかもしれない。まぁ、この解答の例文から いくと数字が複数桁になったらダメになっちゃうけど ね。でも教えてくれてありがとう、**一段授与**。これで 四段。

$\diamond \diamond \diamond$

といったろころで本日の大喜利はこれにてお開き! 読者の皆さん、投稿してくれた皆さん、今回もありが とうございました。



於回のお題

- ←、1 ~ n(nは引数で指定)の自然数の中に存在する 「ズッカーマン数」を列挙するシェルスクリプトを 書いてください。
- 二、今回の rev コマンドの解答を見ていて、このコマ ンドはテキストファイルの全行を右揃えするのに 応用できることを気付かされました。そこで問題。 与えられたテキストファイルをセンタリングする シェルスクリプトを書いてください。

センタリング(中央寄せ)とは、 こうやって 各行内の中央に文字列を寄せる編集操作ですよ

ミ、このシェルスクリプト大喜利で過去に、head に ムチャクチャ巨大な数の行数指定を与えたり、あ るいは mkdir でムチャクチャ深いディレクトリを 掘るなどした珍解答がありました。

そこで、そんなふうにして何かムチャクチャ巨大 な事やらせるシェルスクリプトと、結果何が起こ るかを教えてください。(バージョン番号や、GNU 版か BSD 版か等、再現に必要な環境も教えてくだ さい)でも、**致命的な損害を与える解答はダメふ!**

と投稿のじかた

お題への解答は、お名前(ペンネーム)、解答したい お題番号、解答スクリプト、簡単な補足の四点セット で下記の宛先へ。一人何問でも何個でも解答可! 尚、 次回締め切りは **11 月 25 日 (月) 午前 0 時**とします。 しかもその間は何度でも解答の修正を受け付けます。

お題もどしどし送ってくださーい

お題の投稿も大募集。こっちは締め切りなしで ずーっと募集中。そして、考えてくれた方にも段位を 授与します。自分で出題して解答するのも、OK !

<u> 投稿先</u>

どちらも投稿先は、mag@usp-lab.comです。面 白ければ(ワリと)何でもあり! じゃんじゃん投稿 待ってます!

〒105-0003 東京都港区西新橋 3-4-2 SS ビル 3 階

発行人 當仲寛哲

発行・発売

有限会社ユニバーサル・シェル・プログラミング研究所 電話 03-3432-1174

発行日 2013 年 9 月 20 日



Doorkeeperは、イベントを通じてコミュニティを運営するイベント主催者のためのプラットフォームです。 イベントのお申込みやイベント参加者の管理など、運営に関する様々なことをサポートしています。

www.doorkeeper.jp





USP MAGAZINE バックナンバー 好評発売中!

vol.0 2011 spring~vol.5 2012 summer







USP MAGAZINEは、世界初のシェルスクリプト技術情報誌。でもご存知のとおり、シェルスクリプトはグルー言語。 OS 深層から様々な言語・アプリの話まで、さらには技術の先にいるエンジニア達にもスポットを当てます。 目指すは、シェルスクリプトカとエンジニア達の地位向上!

自炊不要 定期購読又はバックナンバーをお申込みいただくと、PDF 版が無料で手に入ります。

> ご購入、お申込みはこちらから⇒ USP 出版

検索