コードレビュー

Vol.9

CODE Review

USP 研究所技術研究員
written by 大内智明

今回は、ユニケージで開発する際に必ず使用するデータ LV（レベル）について話をします。

データ整理とシェルスクリプト

ユニケージでは、取扱うデータを整理して、LV 毎に5段階に分類及びデータ管理します。LV データは、原始データ（生データ）から LV 毎のデータを作成するシェルスクリプトを用意します（図4 ／次頁）。

技術的な概要

処理性能の向上を図ります。小売業界などで行う業務システム開発は、多くの店舗を取扱う場合がよくあります。ユニケージ開発において、業務や画面操作のレスポンス（補足 1）の観点から考える業務アプリケーション用データは、店舗単位でデータ分割することがあります。店舗毎で分割するデータは、LV1 の原始データから LV 毎にデータ整理していくため、そのまま店舗単位で処理します（今回の処理では、最終的に店舗単位で使用するため、初めから LV 毎のデータも店舗単位に分割した状態で作成しています）（図1）。

店舗件数が多い場合には、複数店舗を同時に並列処理することで、処理時間を短くします（図2）。

今回は、説明しませんが、店舗をまとめて処理することで、更に処理速度が速くなる場合もあります。ユニケージには、最終的に各店舗単位にデータを高速に分割する keycut コマンドなどがありますので、一括にデータをまとめて処理を行っても、高速にデータ分割することができるため、更に処理性能が速くなります（図3）。

【補足 1】
店舗単位で操作する業務システムの場合には、予め必要最小限のデータに分割しておく方が、大量データのまま処理するよりも、レスポンスが速くなります。

図1　店舗毎に処理

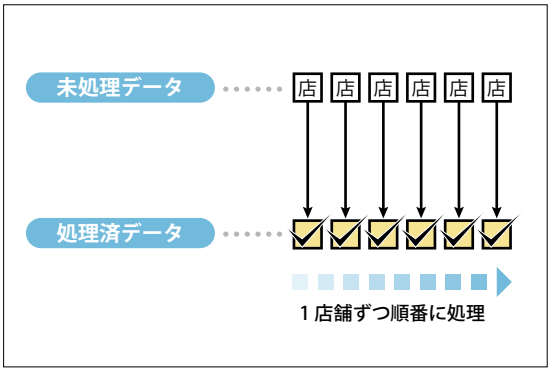


図2　数店舗毎で並列処理

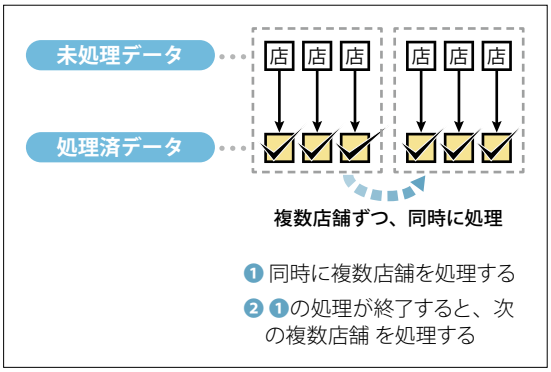


図3　店舗をまとめて処理

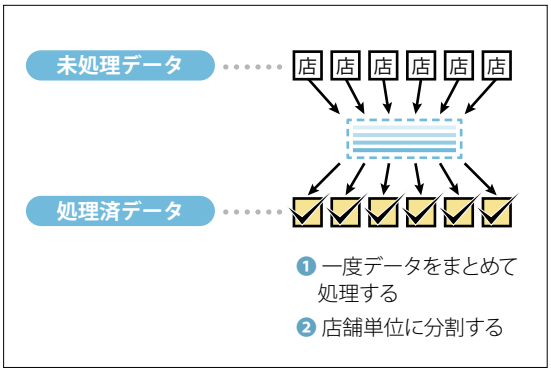
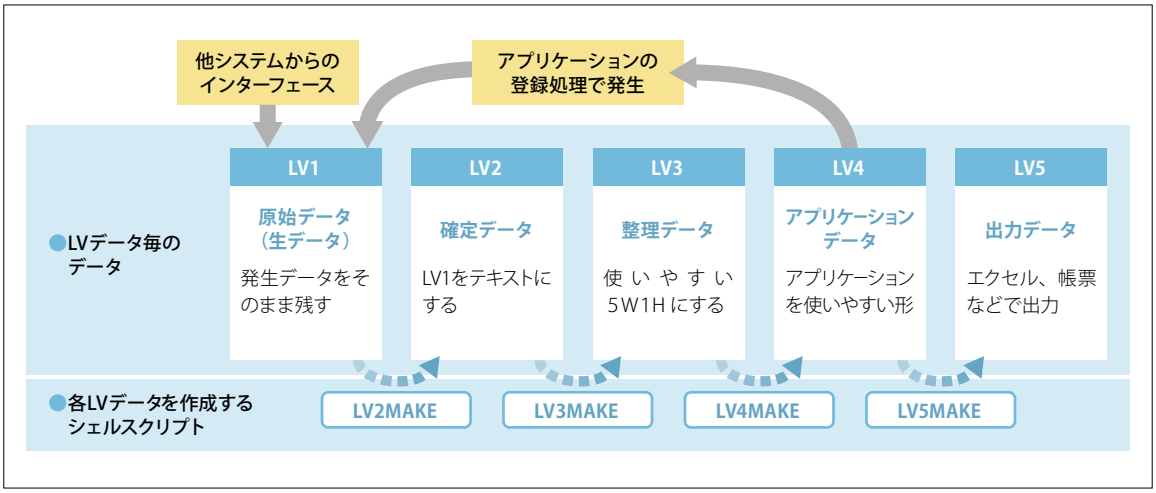


図4　LV データについて



リスト1　各 LV 毎にデータを作成するプログラム

```
#!/bin/ush -xve
# システム名        :USPシステム
# サブシステム名    :伝票管理
# 業務名            :伝票管理
# プログラム名      :伝票作成(夜間処理)
# 概要              :1日1回､夜間に実行する｡昼に発生したトランデータを集計する｡
# 備考(Usage)       :DATAMASTER.DAY.DENPYOU [YYYYMMDD] [事業会社コード]
# シェル名          :DATAMASTER.DAY.DENPYOU
# 作成日            :2014/xx/xx
# 会社名            :USP
# 作成者            :T.Ouchi

#////////////////////////////////////////////////////////////////////////////////
# 初期設定
#////////////////////////////////////////////////////////////////////////////////

# 走行ログの記録
logd="${HOME}/LOG"                                                    # ログディレクトリ
logf="${logd}/LOG.$(basename $0).$(date +%Y%m%d)_$(date +%H%M%S)_$$"  # ログファイル名
echo  "${logf}" > /dev/null 2>&1
log 2> ${logf}

# パスの定義
PATH=/home/UTL:/home/TOOL:/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin:${PATH}
LANG=ja_JP.UTF-8

#-------------------------------------------------------------------------------
# 変数の定義
#-------------------------------------------------------------------------------
tmp=/tmp/$$-$(basename $0)_$(date +%Y%m%d%H%M%S)        # 一時ファイル

hostname="${HOSTNAME}"                                  # サーバ名
semd="${HOME}/SEMAPHORE"                                # セマフォディレクトリ
shld=${HOME}/SYS
lv3d="/home/DATA/LV3"                                   # レベル3親ディレクトリ
lv4d=${HOME}/AP/DENPYOU/LV4                             # LV4MAKEシェルディレクトリ

```

ush は、ユニケージ独自シェル。
sh/bash よりも、エラー制御機能が優れています。

・走行ログのpash指定を行います。
・logコマンドで走行ログを出力します。

❶

ユニケージでは、テンポラリファイルをワークファイルとして使用します。他のプロセスもあるため、必ずユニークなファイル名にします。

```
38 sday=$(date +%Y%m%d)   # デフォルト
39
40 # エラー時の終了処理定義（ushエラーハンドラ）
41 err ERROR_EXIT(){
42     touch ${semd}/$(basename $0).${hostname}.ERROR.${gday}_${jgyk}
43     echo "${hostname} $(basename $0)_${sday}_${jgyk} ERROR $(date +%Y%m%d%H%M%S) ${logf}" >> ${logd}/UPCNT
44     exit 1
45 }
46
47 # 引き数の確認
48 [ $# -ne 2 ] && ERROR_EXIT
49 gday=$1         # 処理日
50 jgyk=$2         # 事業会社コード
51
52 # 簡易YYYYMMDD日付チェック
53 if ! isdate ${gday} ; then
54     echo "Parameter DATE error:[${gday}]"
55     ERROR_EXIT
56 fi
57
58 # 起動時刻の記録
59 echo "${hostname} $(basename $0)_${sday}_${jgyk} START $(date +%Y%m%d%H%M%S)" >> ${logd}/UPCNT
60 touch ${semd}/$(basename $0).${hostname}.START.${gday}_${jgyk}
61
62
63 ////////////////////////////////////////////////////////////////////////////////
64 # データ処理部
65 ////////////////////////////////////////////////////////////////////////////////
66
67 # 対象事業会社の全店舗分繰り返す
68 # 店舗マスタ:事業会社コード
69 #-------------------------------------------------------------------------------------------#
70 # 1:店舗コード*      2:事業会社コード
71 # NF-3:更新フラグ    NF-2:適用日      NF-1:更新者      NF:更新日時
72 cat ${lv3d}/M_TENPO/M_TENPO/TBL/TENPOCD_JGYOK.RIREKI          |
73 upmaster_valid --date ${gday} num=1 -                        |
74 selr 2 ${jgyk}                                              |
75 self 1                                                      |
76 #  1:店舗コード
77 #  全店舗を10店舗ずつのリストにして、ファイル出力する
78 gyocut -10 $tmp-tendiv-%d                    > $tmp-tenlist
79
80 # 10店舗ずつ処理
81 for file in $(cat $tmp-tenlist) ; do
82
83         # 1店舗ずつバックグラウンドで処理
84         for tencd in $(cat ${file}) ; do
85         (
86         #-------------------------------
87         # LV2データ作成
88         #-------------------------------
89             shl=LV2MAKE.DAY.DENPYOU_INPUT_TENPO
90             ${shld}/${shl} ${gday} ${tencd} 1
91
92         #-------------------------------
93         # LV3データ作成
94         #-------------------------------
95         shl=LV3MAKE.DAY.DENPYOU_DATA_INPUT_TENPO
```

ushは、エラーハンドラーが発生すると、err関数を呼び出して、エラー処理を行います。
本スクリプトでエラーが発生すると、異常終了します。

本スクリプトが、開始したことをログに出力しています。

upmaster_valid は、ユニケージ独自コマンド。
履歴データから、指定日付時点の有効データを抽出します。

gyocut はユニケージ独自コマンド。
画面 1 参照

同じ処理を多数の店舗で繰り返す際には、
1度に複数店舗ずつを並列処理することで、
処理性能が向上します。

各店舗毎にLV1→LV2→LV3→LV4とLVデータを生成します。

```
 96         ${shld}/${shl} ${gday} ${tencd} 1
 97
 98         #--------------------------------
 99         # LV4データ作成
100         #--------------------------------
101         shl=${lv4d}/LV4MAKE.DAY.TEGAKI_SIRE_DATA
102         ${shld}/${shl} ${gday} ${tencd}
103
104         # 終了した店舗CD
105         echo ${tencd} END
106         )&
107     done
108     # 10件が終了するまで待つ
109     wait
110     sleep 1
111 done
112
113 #////////////////////////////////////////////////////////////////////////////
114 # 終了処理
115 #////////////////////////////////////////////////////////////////////////////
116  # 終了時刻の記録
117 echo "${hostname} $(basename $0)_${sday}_${jgyk} END $(date +%Y%m%d%H%M%S)" >> ${logd}/UPCNT
118 touch ${semd}/$(basename $0).${hostname}.END.${gday}_${jgyk}
119
120 # 終了
121 rm -f ${tmp}-*
122 echo "$(basename $0) exit 0"
123 exit 0
```

❹
❺
各店舗毎にLV1→LV2→LV3→LV4とLVデータを生成します。
❷
❻
本スクリプトが、終了したことをログに出力しています。

画面1　gyocut コマンドの使用方法

```
[usp@demo1 ~]$ seq 1 5000 | gyocut -1000 /tmp/
gyocut.%02d
/tmp/gyocut.01
/tmp/gyocut.02
/tmp/gyocut.03
/tmp/gyocut.04
/tmp/gyocut.05
[usp@demo1 ~]$ gyo -f /tmp/gyocut.0*
/tmp/gyocut.01 1000
/tmp/gyocut.02 1000
/tmp/gyocut.03 1000
/tmp/gyocut.04 1000
```

1. 5000件データを1000件ずつのファイルに分割します。
2. 分割したファイル名を出力します。

コードの見どころ

ユニケージのシェルスクリプトは、定型文の書き方があります。

[1] ヘッダ部では、シェルスクリプトの説明・各種定義・起動ログの記録などを行います。❶1〜60行目で行っています。

[2] 次にボディ部で、プログラム処理を記述します。❷63〜111行目で行っています。LV1データからLV2データの作成を行います（❸87〜90行目）。LV2データからLV3データの作成を行います（❹92〜96行目）。LV3データからLV4データの作成を行います（❺98〜102行目）。

[3] 最後に、終了ログの記録やworkファイルの削除を行います。❻113〜123行目です。

ユニケージ開発では、作成するLVデータ毎に分けてシェルスクリプトを製造することで、データ整理及びシェルスクリプトもLV毎に分けることができるので、メンテナンスの向上を図ることができます。

まとめ

ユニケージ開発では、LV毎のシェルスクリプトを用意することで、各シェルスクリプト自体の機能をシンプルにできます。また、シェルスクリプトに障害や仕様変更が発生しても、最小限のシェルスクリプトだけの改修で済ませることができるため、メンテナンスも比較的行い易くなります。

次回は、LVレジ毎にデータを作成するシェルスクリプトについて紹介します。