コードレビュー Vol.8

CODE Review

USP 研究所技術研究員
written by 大内智明

今回は、PC画面の情報をサーバ側に送信して、LV1データ/LV4データとして登録する処理について説明します。

データの流れについての紹介

前回は、LV4 データから必要な条件で抽出して、画面出力用のレイアウトに成形する処理を行いました。今回は、画面に表示したデータを画面上で新規追加･更新･削除の操作を行い、加工した結果をサーバ側に送信します。送信した結果は、LV1、LV4 データとして登録します（図 1 朱色の部分）。

【「LV1、LV4」についての補足説明】
LV1、LV4 とはユニケージ開発手法で使用されるデータのレベルごとの名前のこと。
LV1・システムに入力された「生データ」(原始データ) でアプリケーションから入力されるデータや外部システムからインターフェースされるデータ。
・日中の更新は、アプリケーションが行います。
LV4・業務アプリケーションが登録・参照・検索などを行いやすい形のデータ。
・朝 1 回目は夜間処理で作成します。
・日中の更新は、アプリケーションが行います。

技術的な概要

ユニケージとデータベースでは、登録処理の仕組みが異なっています（図 2）。データベースは、既存ファイルを直接、更新データに書き換えて、登録処理を行います。ユニケージは、フィルタ形式（補足 1）で処理を行うため、登録処理（upl コマンド（画面 1 参照））を行うには、既存ファイルに更新データをマージして、一旦別ファイルとして保存します。保存したファイルを既存ファイルに置き換えて、登録処理を完了します（補足 2）。

【補足 1】
ユニケージのコマンドは、フィルタ形式であるため、入力ファイルと出力ファイルが別になっています。
【補足 2】
ユニケージは、登録処理の途中でエラーなどが発生しても、元の状態に戻せるように（ロールバック）、予め既存データのバックアップを取得する場合があります。データベースでは、意識しない排他処理もユニケージでは排他処理のロジックを盛り込む必要があります。

図 1　処理及びデータフロー（処理を行う時間帯は 1 例です。）

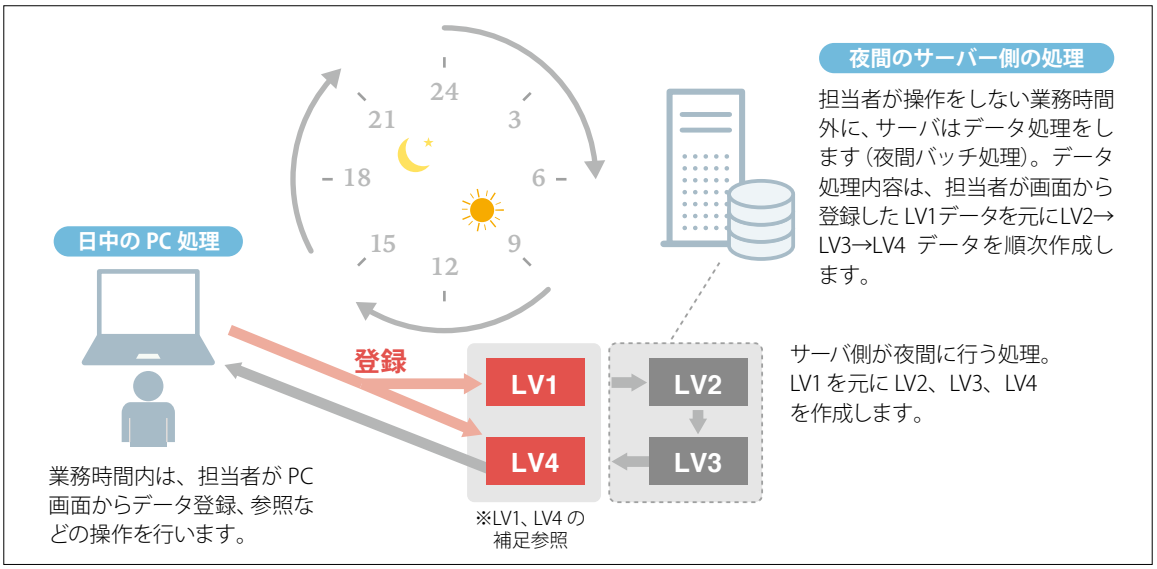
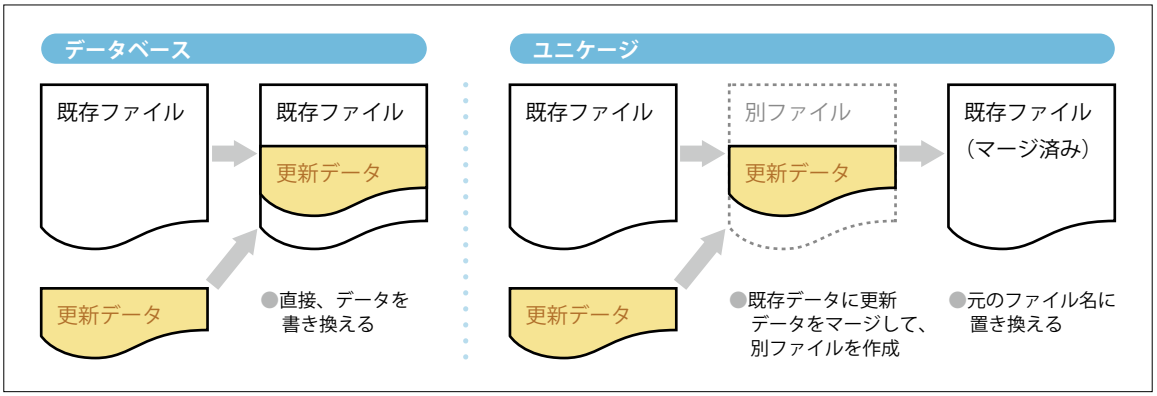


図2　データベースとユニケージの登録処理



リスト1　登録処理を行うプログラム

```
#!/bin/ush -xve
#
# 〈中略〉
#

#////////////////////////////////////////////////////////////////////////////////
# 入力パラメータチェック
#////////////////////////////////////////////////////////////////////////////////
[ $# -ne 2 ] && exit 1

flddata=$1
sdaytime=$2
sday=$(echo ${sdaytime} | self 1.1.8)
stime=$(echo ${sdaytime} | self 1.9.12)
# 入力ファイル有無チェック
[ ! -s ${flddata} ] && exit 1
ppid=$$

#////////////////////////////////////////////////////////////////////////////////
# 処理部
#////////////////////////////////////////////////////////////////////////////////
# * FLDDATA
# 1:業務日付          2:ユーザID          3:ユーザ名          4:画面ID          5:ホスト名
# 6:最終アクセス日時  7:認証用ハッシュ値  8:CRSバージョン    9:最大取得行数    10:MSCTRL最終更新日付
# 11:排他管理用日付   12:サイト区分       13:事業会社         14:削除フラグ     15:担当者
# 16:店舗コード       17:店舗コード       18:データ作成日     19:データ作成時刻 20:SEQ
# 21:削除フラグ       22:行               23:チェック結果     24:廃棄日         25:廃棄理由コード
# 26:商品/中分類コード 27:商品/中分類名   28:数量             29:原単価         30:売単価
# 31:税区分コード     32:取込区分コード   33:入力日時

# サーバに設定された業務日付を取得する
gday=$(self 1 ${flddata} | tail -n -1)
gdaytime=${gday}${sdaytime.9.14}

# 画面から取得したパラメータを変数にセット
userid=$(self 2 ${flddata} | tail -n -1)    # ユーザー
jigyoucd=$(self 13 ${flddata} |  tail -n -1)   # 事業会社
tantouid=$(self 15 ${flddata} |  tail -n -1)   # 担当者
tenpocd=$(self 16 ${flddata} |  tail -n -1)    # 店舗コード
site_kubun=$(self 12 ${flddata} |  tail -n -1)    # 本部サイト区分 1:本部、2:店舗

```

```
# 本部店舗区分
selr 1 "0148" ${lv3d}/M_SONOTA/M_KUBUN/TBL/M_KUBUN |
self 2 3 > $tmp-honten
site_name=$(selr 1 ${site_kubun} $tmp-honten | self 2)
# MD業務G
mdg=$(gawk '$1=="'"${jigyoucd}"'"' ${lv3d}/M_SONOTA/M_JGYK/TBL/JGYKCD_MDGYOMUGRP | self 2)

self 17/33 ${flddata}  > ${tmp}-data
#  1:店舗コード        2:データ作成日     3:データ作成時刻   4:SEQ              5:削除フラグ
#  6:行                7:チェック結果     8:廃棄日           9:廃棄理由コード   10:商品/中分類コード
#  11:商品/中分類名    12:数量            13:原単価          14:売単価          15:税区分コード
#  16:取込区分コード   17:入力日時

#-------------------------------------------------------------
# 入力データの取得
#-------------------------------------------------------------
cat ${tmp}-data     |
#  1:店舗コード       2:データ作成日       3:データ作成時刻       4:SEQ
#  5:削除フラグ       6:行                 7:チェック結果         8:廃棄日
#  9:廃棄理由コード   10:商品/中分類コード 11:商品/中分類名       12:数量
#  13:原単価          14:売単価            15:税区分コード        16:取込区分コード
#  17:入力日時
self   1  2  3  4  5    ¥
       7  8  9  10 12   ¥
       13 14 15 16 17 s  |
# 1:店舗コード    2:データ作成日  3:データ作成時刻  4:SEQ                5:レコード状態フラグ
# 6:チェック結果  7:廃棄日        8:廃棄理由コード  9:商品/中分類コード  10:数量
# 11:原単価       12:売単価       13:税区分コード   14:取込区分コード    15:入力日時
#
# 「コード:名称」から「コード」抜き出す
gawk '{$8=index($8,":")!=0?substr($8,1,index($8,":")-1):$8;print }' |
cat        > ${tmp}-all_data

# 更新フラグ(削除フラグ)が1:新規の場合
cat ${tmp}-all_data    |
gawk '$5=="1"{print $0}'  > ${tmp}-shinki
# 更新フラグ(削除フラグ)が4:更新の場合
cat ${tmp}-all_data    |
gawk '$5=="4"{print $0}'  > ${tmp}-koushin
# 更新フラグ(削除フラグ)が8:削除の場合
cat ${tmp}-all_data    |
gawk '$5=="8"{print $0}'  > ${tmp}-sakujyo

# LV4データの用意
lineup 1 ${tmp}-all_data  > $tmp-input.ten
for ten in $(cat $tmp-input.ten) ; do
  [ ! -s ${lv4d}/${ten}/DENPYO_DATA_HAIKI/DENPYO_DATA_HAIKI ] && continue

  cat ${lv4d}/${ten}/DENPYO_DATA_HAIKI/DENPYO_DATA_HAIKI
done            |
msort key=1/4   |
cat     > ${tmp}-lv4_data

# 更新フラグ(削除フラグ)が1:新規のものがあったら
if [ -s ${tmp}-shinki ] ; then
  #〈中略〉
  msort key=1/4  > ${tmp}-shinki_lv4
else
```

lineup コマンド
ユニケージ独自コマンド。指定したフィールドのデータのラインナップを取り出す

[1]

```
100   cat ${tmp}-shinki > ${tmp}-shinki_lv4
101 fi
102   # 更新フラグ(削除フラグ)が4:更新の場合
103   if [ -s ${tmp}-koushin ] ; then
104     cat ${tmp}-lv4_data                |
105     # 1:店舗コード      2:データ作成日   3:データ作成時刻   4:SEQ              5:レコード状態フラグ
106     # 6:チェック結果    7:廃棄日         8:廃棄理由コード   9:商品/中分類コード 10:数量
107     # 11:原単価         12:売単価        13:税区分コード    14:取込区分コード   15:登録日時
108     # 16:登録ユーザID   17: 更新フラグ   18:更新日          19:担当者           20:更新日時
109     #
110     # 更新フラグが8:削除のものは排除
111     gawk '$17!="8"{print $0}'          |
112     # キーと必要項目のみにする
113     self  1  2  3  4  5   ¥
114           6  7  8  9  10  ¥
115           11 12 13 14 15  ¥
116           16 20                        |
117     #  1:店舗コード           2:データ作成日      3:データ作成時刻      4:SEQ
118     #  5:レコード状態フラグ   6:チェック結果      7:廃棄日              8:廃棄理由コード
119     #  9:商品/中分類コード    10:数量             11:原単価             12:売単価
120     #  13:税区分コード        14:取込区分コード   15:登録日時           16:登録ユーザID
121     #  17:更新日時
122     cjoin2 key=1/4 - ${tmp}-koushin    |
123     #〈サーバ内データ〉
124     #  1:店舗コード           2:データ作成日      3:データ作成時刻      4:SEQ
125     #  5:レコード状態フラグ   6:チェック結果      7:廃棄日              8:廃棄理由コード
126     #  9:商品/中分類コード    10:数量             11:原単価             12:売単価
127     #  13:税区分コード        14:取込区分コード   15:登録日時           16:登録ユーザID
128     #  17:更新日時
129     #〈入力データ〉
130     #  18:レコード状態フラグ  19:チェック結果
131     #  20:廃棄日              21:廃棄理由コード   22:商品/中分類コード  23:数量
132     #  24:原単価              25:売単価           26:税区分コード       27:取込区分コード
133     #  28:入力日時
134     gawk '{print          $1, $2, $3, $4, $18,
135                 $19, $20, $21, $22, $23,
136                 $24, $25, $26, $27, $15,
137                 $16, "4", "'${gday}'", "'${userid}'", "'${sdaytime}'",
138                 $17, $28
139     }'                                 |
140     #  1:店舗コード           2:データ作成日      3:データ作成時刻      4:SEQ
141     #  5:レコード状態フラグ   6:チェック結果      7:廃棄日              8:廃棄理由コード
142     #  9:商品/中分類コード    10:数量             11:原単価             12:売単価
143     #  13:税区分コード        14:取込区分コード   15:登録日時           16:登録ユーザID
144     #  17:更新フラグ          18:更新日           19:担当者             20:更新日時
145     #  21:現在更新日時        22:検索時更新日時
146     cat       > ${tmp}-koushin_lv4.updchk
147     self 1/NF-2 ${tmp}-koushin_lv4.updchk > ${tmp}-koushin_lv4
148 else
149     cat ${tmp}-koushin     > ${tmp}-koushin_lv4
150 fi
151 # 更新フラグ(削除フラグ)が8:更新の場合
152 if [ -s ${tmp}-sakujyo ] ; then
153   #〈中略〉
154   self 1/NF-2 ${tmp}-sakujyo_lv4.updchk > ${tmp}-sakujyo_lv4
155 else
156   cat ${tmp}-sakujyo     > ${tmp}-sakujyo_lv4
157 fi
```

cjoin2 コマンドは
ユニケージ独自コマンド(画面1参照)。

[1]

```
158
159 #-------------------------------------------------------------
160 # 画面からのデータ(新規追加、更新、削除)を合わせる
161 #-------------------------------------------------------------
162 cat ${tmp}-koushin_lv4 ${tmp}-sakujyo_lv4 ${tmp}-shinki_lv4 |
163 msort key=1/4           |                                                    [1]
164 gawk ´{$19="´${tantouid}´";print}´     > ${tmp}-new_data                    ❷
165 #  1:店舗コード          2:データ作成日       3:データ作成時刻   4:SEQ
166 #  5:レコード状態フラグ  6:チェック結果       7:廃棄日           8:廃棄理由コード
167 #  9:商品/中分類コード   10:数量              11:原単価          12:売単価
168 #  13:税区分コード       14:取込区分コード    15:登録日時        16:登録ユーザID
169 #  17:更新フラグ         18:更新日 19:担当者  20:更新日時
170
171 #-------------------------------------------------------------
172 # マスタ情報取得
173 #-------------------------------------------------------------
174 #〈中略〉
175 # ${tmp}-torikomi_kubun
176 # 1:取込区分  2:取込区分名称
177 # ${tmp}-SHOHN_INFO
178 # 1:商品コード  2:基本マスタ有無  3:商品名  4:店舗マスタ有無  5:原価税抜
179 # 6:売価税抜  7:中分類コード  8:税区分  9:用度品区分  10:取り扱い不可区分
180 #
181 # ${tmp}-TENPO_INFO                                                        ❸
182 # 1:店舗コード  2:店舗名  3:事業会社コード  4:事業会社名
183 #
184 # ${tmp}-JGYKCD_LOCKBI
185 # 1:事業会社コード  2:締日有無  3:伝票訂正方法
186 #
187 # ${tmp}-zei_kubun
188 # 1:税区分  2:税区分名称
189 #
190 # ${tmp}-HaikiRiyuKbn_NAME
191 # 1:廃棄理由  2:廃棄理由名称
192 #
193 #-------------------------------------------------------------
194 # LV4にコード名称を付与                                                    [2]
195 #-------------------------------------------------------------
196 cat ${tmp}-new_data                          |
197 cjoin2 +_ key=14 ${tmp}-torikomi_kubun       |
198 cjoin2 +_ key=9 ${tmp}-SHOHN_INFO -          |
199 cjoin2 +_ key=1 ${tmp}-TENPO_INFO -          |
200 cjoin2 +_ key=3 ${tmp}-JGYKCD_LOCKBIi       |
201 cjoin2 +_ key=21 ${tmp}-zei_kubun           |
202 cjoin2 +_ key=13 ${tmp}-HaikiRiyuKbn_NAME   |
203 #                                                                          ❹
204 #〈中略〉
205 #
206 # 1:店舗コード      2:店舗名          3:事業会社コード  4:締日有無      5:(本部/店舗)伝票訂正方法
207 # 6:事業会社名      7:データ作成日    8:データ作成時刻  9:SEQ          10:レコード状態フラグ(新規・変更・削除)
208 # 11:チェック結果   12:廃棄日         13:廃棄理由コード 14:廃棄理由名  15:商品/中分類コード
209 # 16:基本マスタ有無 17:商品/中分類名  18:店舗マスタ有無 19:原価税抜    20:売価税抜
210 # 21:中分類コード   22:税区分        23:税区分名       24:用度品区分  25:取り扱い不可区分
211 # 26:数量           27:原単価         28:売単価         29:税区分コード 30:取込区分コード
212 # 31:取込区分名
213 #〈中略〉
214 cat              > ${tmp}-check-out
```

```
215 ############################################################
216 # 日付・商品コード・原価・売価などの各項目チェックを行う
217 ############################################################
218
219
220 ################################################
221 # データ整形
222 ################################################
223 cat ${tmp}-check-out    |
224 fsed ´s/_|//41´         |
225 fsed ´s/|_//41´         |
226 gawk ´{print $3,$6,$1,$2,$7,
227               $8,$9,$12,$15,$17,
228               $13,$14,$26,$27,$28,
229               $22,$23,$30,$31,$40,
230               $41,"0","´${site_kubun}´","´${site_name}´",$36,
231               $39,$32,$33,$34,$35,
232               "´${userid}´",$37}´                |
233 gawk ´$7!="_"{$7=sprintf("%05d",$7)}{print}´    |
234 cat            > ${tmp}-result
235 # 1:事業会社コード   2:事業会社名        3:店舗コード          4:店舗名              5:データ作成日付
236 # 6:データ作成時刻   7:SEQ              8:廃棄日              9:商品/中分類コード    10:商品/中分類名
237 # 11:廃棄理由区分    12:廃棄理由区分名  13:数量               14:原単価             15:売単価
238 # 16:税区分          17:税区分名        18:伝票データ取込区分  19:伝票データ取込区分名
239 # ＜中略＞
240
241 #-------------------------------------------------------------
242 # lv1データの更新
243 #-------------------------------------------------------------
244 lv1d=${apld}/INPUT/${gday}/${tenpocd}/DENPYO_DATA_HAIKI
245 # LV1ディレクトリ作成
246 mkdir -p ${lv1d}
247
248 cat ${tmp}-result     |
249 # LV1に必要なデータのみにする
250 delf 2 4 10 12 17 19 24 |
251 # 1:事業会社コード 2:店舗コード   3:データ作成日付    4:データ作成時刻 5:SEQ
252 # 6:廃棄日       7:商品/中分類コード 8:廃棄理由    9:数量    10:原単価
253 # 11:売単価   12:税区分    13:伝票データ取込区分
254 # ＜中略＞
255 msort key=1/5      |
256 cat       > ${tmp}-haiki_data_lv1
257
258 #-------------------------------------------------------------
259 # LV4データの更新
260 #-------------------------------------------------------------
261 # ロック
262 lockfile=${lv4d}/DENPYO_DATA_HAIKI.LOCK
263 if ulock ${lockfile}
264 then
265   # 正常にロックファイルが生成
266
267   # LV1へファイルとして保存
268   cp ${tmp}-haiki_data_lv1 ${lv1d}/${tenpocd}.${userid}.$$.${gdaytime}
269   # LV4ディレクトリ作成及びファイル作成
270   mkdir -p ${lv4d}/${tenpocd}/DENPYO_DATA_HAIKI
271   touch ${lv4d}/${tenpocd}/DENPYO_DATA_HAIKI/DENPYO_DATA_HAIKI
272
```

fsed コマンドは、ユニケージ独自コマンド（画面 2 参照）

delf コマンドは、ユニケージ独自コマンド（画面 3 参照）

ulock コマンドは、ユニケージ独自コマンド（画面 4 参照）

[2]

[3]

❺ ❻ ❼

```
273   # LV4に既存のデータがあれば追記
274   cat ${lv4d}/${tenpocd}/DENPYO_DATA_HAIKI/DENPYO_DATA_HAIKI |
275   self 3 5 6 7 0       |
276   msort key=1/4     > ${tmp}-lv4_old
277
278   selr 3 ${tenpocd} ${tmp}-result  |
279   self 3 5 6 7 0       |
280   msort key=1/4     |                                    ❼ [3]
281   upl key=1/4 ${tmp}-lv4_old -  |   ←upl コマンドは、ユニケージ独自コマンド（画面 5 参照）
282   delf 1/4       > ${tmp}-lv4_result
283   mv ${tmp}-lv4_result ${lv4d}/${tenpocd}/DENPYO_DATA_HAIKI/DENPYO_DATA_HAIKI
284
285   # ロックファイル削除
286   rm -f ${lockfile}
287 fi
288
289 #/////////////////////////////////////////////////////////////////////////////
290 # 終了
291 #/////////////////////////////////////////////////////////////////////////////
292 set +xv
293 rm -f ${tmp}-* 2>/dev/null
294 set -xv
295 exit 0
```

画面 1　cjoin2 コマンド

2ファイルをキーで連結する。マッチングしない場合には、ダミーデータを出力する。

```
$ cat mst
0000003 杉山_____ 26 F
0000005 崎村_____ 50 F
0000007 梶川_____ 42 F
$ cat trn
0000005 82 79 16 21 80
0000001 46 39 8  5  21
0000004 58 71 20 10 6
0000009 60 89 33 18 6
0000003 30 50 71 36 30
0000007 50 2  33 15 62
$ cjoin2 key=1 mst trn
0000005 崎村_____ 50 F 82 79 16 21 80
0000001 _ _ _ 46 39 8  5  21
0000004 _ _ _ 58 71 20 10 6
0000009 _ _ _ 60 89 33 18 6
0000003 杉山_____ 26 F 30 50 71 36 30
0000007 梶川_____ 42 F 50 2  33 15 62
```

マッチしない場合には、デフォルト"_"でダミー出力する。

画面 2　fsed コマンド

指定フィールドの文字置換を行う。

```
$ cat data
tokyo 1234 tokyo 5678
osaka 1234 osaka 5678

$ fsed 's/tokyo/TOKYO/1' 's/osaka/OSAKA/3' data
TOKYO 1234 tokyo 5678
oksaka1234 OSAKA 5678
```

画面 3　delf コマンド

指定したフィールドを除いて出力する。

```
$ cat data
0000000 浜地_____ 50 F
0000001 鈴田_____ 50 F
0000003 杉山_____ 26 F
0000004 白土_____ 40 M
$ delf 2 data  <- 第 2 フィールドを除いて出力します。
0000000 50 F
0000001 50 F
0000003 26 F
0000004 40 M
```

画面 4　ulock コマンド

排他制御を行う。ulock は排他的にロックファイルを作成して、完全排他区間を実現する。

```
$ cat lock.sh
#!/bin/bash

if ulock lock; then
  #
  # 読み書きなどの処理
  #
  rm -f lock
fi
```

画面 5　upl コマンド

二つのファイルを同一キーフィールドでマージして
キーフィールドの値が同一の最終行を抽出する。

```
(マスター:master)
a店 103 62
b店 157 94
c店 62 30
d店 210 113
e店 237 121

(トランザクション:tran)
a店 131 84
c店 198 105
e店 81 48

$ upl key=1 master tran > data
(data)          <- tranをマージして同一店の最終行を抽出
a店 131 84
b店 157 94
c店 198 105
d店 210 113
e店 81 48
```

既存ファイル

更新ファイル(更新データ)

更新データがマージされたファイル

コードの見どころ

今回のコードについて、3 つの観点から説明していきます。

[1] 画面から取得した入力データを元に更新用データを作成します。55 ～169 行目です。まずは、入力データを新規・更新・削除データに分割して、それぞれを処理します(❶55～157 行目)。次に分割したデータを結合します（❷159 ～169 行目)。

[2] [1] で作成した更新データに、PC 画面で表示用に使用する各項目の名称を付与します。171 ～239 行目で行っています。まずは、付与するマスタ項目（名称付き）を作成します（❸171 ～191 行目)。次に更新データにマスタ（名称付き）を付与します（❹193 ～214 行目)。今回は省略していますが、入力項目について整合性のチェック（コードや日付などの整合性）を行います。最後に LV1、LV4 登録用にデータを成形します（❺220 ～239 行目)。

[3] LV1 と LV4 の登録処理を行います。241 ～ 287 行目で行います。まずは、LV1 データを作成します（❻241 ～256 行目)。LV4 は [2] で既に作成済み。次に LV1、LV4 をそれぞれ登録処理します（❼258 ～287 行目)。

まとめ

ユニケージの登録処理では、LV1 (夜間処理) と LV4 (PC 画面出力用) の 2 種類のファイルを更新します (補足 3)。排他制御についてもユニケージの場合は、データベースと異なり盛り込む必要があります。

LV1 は、夜間処理にて、LV1 → LV2 → LV3 → LV4 を生成して、日中更新した LV4 を定期的に再作成します（整合性を保つ)。

【補足 3】
LV4 データは、データの整合性を保つため、定期的に LV3 から全件作成し直します。