# コードレビュー Vol.11

CODE Review

USP 研究所技術研究員
*written by* 大内智明

今回は、シェルスクリプトを実行して、データベースからデータを取得する方法についてご説明します。

## データベースサーバーにSQLを発行して、データを取得する

ユニケージの導入事例でもよくありますが、既存のデータベースサーバーが稼働しているところに、新規機能の追加や性能改善などのために、ユニケージを導入して処理する場合があります。具体的には、既存システムからデータを取得して、ユニケージサーバー内で加工処理を行います。加工処理されたデータは、開発要件に従いユーザPC から参照・登録することができます。また、加工処理したデータを既存システムに戻すこともできます（図1）。

データベースは、オラクルを例に説明します。SQL を発行するコマンド SQLPLUS は、前準備としてインストールが必要です（※1）。

## 技術的な概要

ユニケージサーバーは、データベースサーバーに SQL・DB・ホスト情報などのパラメータと SQL 文を発行すると、結果としてテキスト形式のデータを取得できます。

発行する SQL 文の内容には、データ出力時の形式（区切り文字／出力ファイル名など）を設定する必要があります。ユニケージ開発では、取り扱うデータ形式にルール（※2）があるため、SQL コマンドで出力した結果に対して、ユニケージ内で使いやすい形に成型します（図2）。

【補足】
※1 使用するデータベースの種類やバージョンによって、取得するコマンドが異なる場合があります。
※2 ユニケージ開発を行う上でのルールには、区切り文字に半角スペース・ヌル文字に _" 半角アンダースコアー " などがあります。

図1　データベースサーバとユニケージ

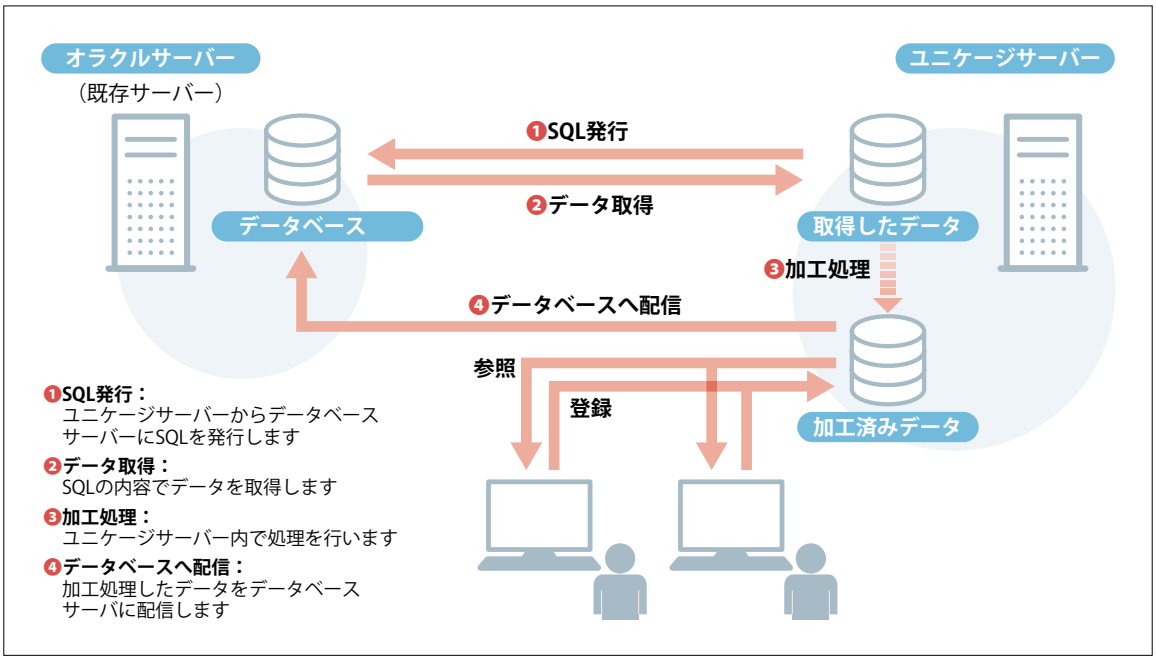
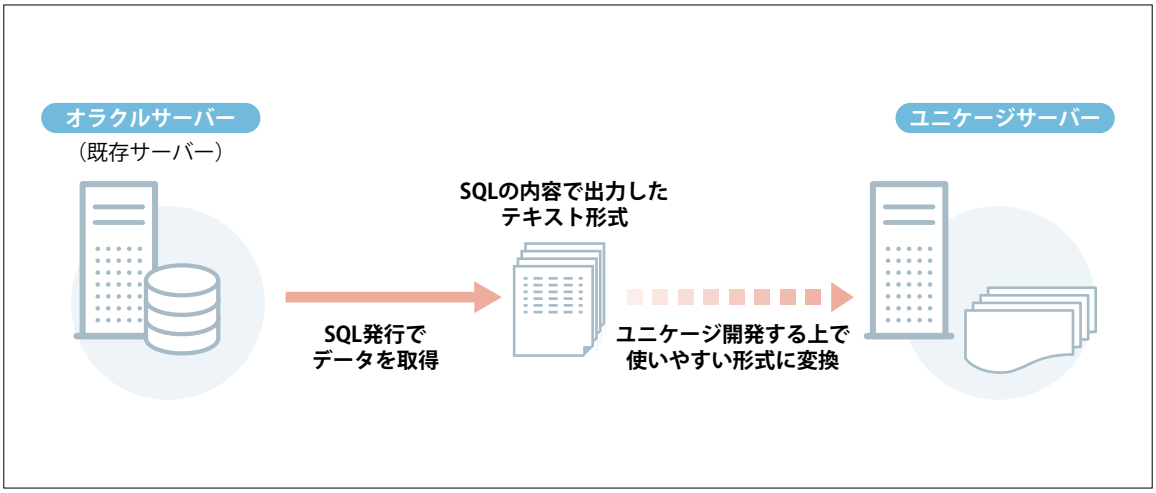


図2　ユニケージ用データ形式の生成




リスト1　データベースより必要データを取得

```
 1 #!/bin/bash -xv
 2 #
 3 # BATCH_SQLPLUS host user pass dbname table
 4 # Usage   : BATCH_SQLPLUS
 5 # 説明    :sqlplusコマンドを実行して、データベースより必要データを取得
 6 #
 7 # Written by xxxx / Date : 20xx/xx/xx
 8
 9 # 起動サーバが、有効なサーバー(正)かどうかを確認する。
10 msctrl -job hac_sys -ctrl C print host | grep -q "^${HOSTNAME}" || exit 0
11
12 # 走行ログ
13 logfile=/home/usp/LOG/LOG.$(basename $0).$(date +%Y%m%d)_$(date +%H%M%S)
14 exec 2> $logfile
15
16 # 変数の定義
17 export LANG=ja_JP.UTF-8
18 export PATH=/home/UTL:/home/TOOL:$PATH
19 export LD_LIBRARY_PATH=/usr/lib/oracle/xx.x/client64/lib/
20 export NLS_LANG=JAPANESE_JAPAN.UTF8
21 tmp=/tmp/$$-$(basename $0)_ $(date +%Y%m%d%H%M%S)
22 today=$(date +%Y%m%d)
23 todayhms=$(date +%Y%m%d%H%M%S)
24
25 home=/home/usp
26 logd=$home/LOG
27 semd=$home/SEMAPHORE
28
29 # エラーチェックと終了処理の定義
30 ERROR_CHECK() {
31   [ $(plus ${PIPESTATUS[@]}) -eq 0 ] && return
32   touch $semd/$(basename $0).$HOSTNAME.ERROR.$today
33   exit 1
34 }
35
36 # 起動時刻
37 echo "$HOSTNAME $(basename $0) START $(date +%Y%m%d%H%M%S)" >> $logd/UPCNT
```

ユニケージでは、サーバー毎に処理業務の役割（ROLE）をもっています。今回の処理に対して、自身が有効なサーバーかどうかを判定します。

環境変数を設定

エラーチェック処理関数
本関数は、各処理の後に実行することで、エラーが発生したかどうかをチェックできます。
使い方は、98行目のように複数コマンドをパイプ“｜”で繋いで、各コマンドが正常終了／異常終了しているかをチェックできます。

```
38 touch $semd/$(basename $0).$HOSTNAME.START.$today
39
40 ##########################################################################################
41 # 今回の処理は副サーバで処理を行わないため、ここで抜ける
42 if [ $(msctrl -job hac_sys -host ${HOSTNAME} print msflg) = "S" ] ; then
43   # 副サーバでは処理を行わない
44   echo "$HOSTNAME $(basename $0) END $(date +%Y%m%d%H%M%S)" >> $logd/UPCNT
45   touch $semd/$(basename $0).$HOSTNAME.END.$today
46
47   rm -f $tmp-*
48   exit 0
49 fi
50
51 ##########################################
52 # オプション設定
53 host=$1     # host
54 user=$2     # user
55 pass=$3     # pass
56 dbname=$4   # DB
57 table=$5    # Table
58 [ -z $table ] && exit 0
59
60 ##########################################
61 # 取得開始
62 echo Downloading... $host $table
63
64 # sql文を生成
65 cat << FIN | sed 's|###FILE###|'$tmp-$host.$table.$today'|' | sed 's|###FROM###|'$table'|' > $tmp-sqlbun.sql
66 set heading off
67 set feedback off
68 set trimspool on
69 set termout off
70 set echo off
71 set pages 0
72 set colsep '    '
73 set line 9999
74 whenever sqlerror exit 1
75 spool ###FILE###
76 select
77 *
78 from ###FROM###;
79 spool off
80 ;
81 quit
82 FIN
83
84 # 全部一気に取り込み
85 sqlplus64 $user/$pass@$host:xxxx/$dbname @$tmp-sqlbun.sql
86 ERROR_CHECK
87 cat $tmp-$host.$table.$today      |
88 # この時点ではタブ(colsepで指定した文字列)区切り
89 # フィールド頭の空白を消す
90 sed 's/\t */\t/g' | sed 's/^ *//' |
91 # フィールド末尾の空白を消す
92 sed 's/ *\t/\t/g' | sed 's/ *$//' |
93 tr " " "_"                        |
94 # ヌルは_に変換
95 sed 's/\t\t/\t_\t/g' | sed 's/\t\t/\t_\t/g' | sed 's/^\t/_\t/' | sed 's/\t$/\t_/' |
```

ユニケージ開発では、正副のサーバー構成でシステム構築することを推奨しています。データベースサーバーからデータを集積するサーバーは、必ず正サーバーで行います。副サーバーは、正サーバーから配信します。
**msctrl コマンド**は、各サーバーの正副・業務役割の情報を出力できます。

❶ SQL 文は、スクリプト内で引数のパラメータをセットします。ひな形のテンプレートをヒアドキュメントで作成して、動的なパラメータは、都度設定します。

❷ SQLPLUS コマンドで、SQL 文を実行します。

❸ SQLPLUS コマンドで出力したファイルは、ユニケージで処理しにくい形のため、一旦、ユニケージで操作しやすい形に変換します。

```
 96 # 最後、タブ区切りを空白区切りにする
 97 tr "\t" " "                        > $tmp-$host.$table.$today.new
 98 ERROR_CHECK
 99 cat $tmp-$host.$table.$today.new
100 ERROR_CHECK
101
102 ##########################################################################################
103 # 終了処理
104 echo "$HOSTNAME $(basename $0) END $(date +%Y%m%d%H%M%S)" >> $logd/UPCNT
105 touch $semd/$(basename $0).$HOSTNAME.END.$today
106
107 rm -rf $tmp-*
108 exit 0
```

❸

## コードの見どころ

コードについて、3つのセクションに分けてご説明します。

[1] 本シェルスクリプトに引数として設定した値をそれぞれ変数にセットします。次にセットした変数は、ヒアドキュメントの中でデータベースから取得したい形式・ファイル名などの形に値を埋め込み、SQL 文を作成します（❶52 ～ 82行目まで）。

[2] 予めインストールして用意して置いた SQL 取得コマンドと [1] で用意した SQL 文を組み合わせて、指定範囲のデータを取得します（❷84 ～ 86行目まで）。

[3] データベースから取得した直後のデータが、そのままユニケージ内で使用できない場合には、加工してユニケージ内で操作しやすいように形を成型します（❸87 ～ 100行目まで）。

## まとめ

既存システムに新規機能を追加したい場合には、既存システムのデータベースから店舗や商品などのマスタ情報や取引情報をユニケージサーバー内に取り込むことで、ユニケージサーバー内で処理を行うことができます。

今回は、相手サーバーからデータを取得する集信型（※3）のパターンについて説明しました。他にも相手サーバーから自サーバーにデータを送信してもらう配信型（※3）のパターンもあります。

また、取り扱うデータ種類については、CSV ／固定長データなどの様々なパターンに対応することもできます。

【補足】

※3 サーバー間のデータ通信方法は、配信型及び集信型の方法があります。配信型は、配信元から配信先にデータを配布するため、配信タイミングも配信元で管理できます。また、順次配布できるため、サーバー負荷も制御できます。集信型は、配信元サーバーからデータ取得を行うため、既に配信元でデータが更新済みかどうかの判断が必要になります。また、集信するタイミングによっては、負荷が配信元サーバーに集中する場合があります。基本的にユニケージ開発では、メリットの多い配信型でのデータ通信を推奨しています。