探索的財務ビッグデータ解析と再現可能研究：
DataHub データのラングリングと Osiris データとの比較

地　道　正　行

要　旨

Moody's Analytics 社は，クラウド型のデータ配信プラットフォームである Moody's DataHub（以後，DataHub と略）を2021年から運用しており，世界の上場・非上場企業の企業情報等を提供している．本稿では，DataHub から抽出した規模の大きなデータのラングリングを行い，これまで Bureau van Dijk 社のデータベース Osiris を利用して行ってきたデータ解析の結果とを比較することによって再現性などの検証を行う．

キーワード：DataHub データ（DataHub Data），前処理（Preprocessing），データラングリング（Data Wrangling），再現可能性（Reproducibility），Osiris データ（Osiris Data）

I　はじめに

筆者らの研究グループは，Bureau van Dijk 社[1] のデータベース Osiris から抽出された世界の上場企業のデータと，同社のデータベース Orbis から抽出された上場・非上場企業のデータを用いて探索的データ解析に基づいて可視化と統計モデリングを行ってきた（cf. 地道, 2017-a, b; 地道, 2018-a, b; Jimichi *et al.*, 2018; 地道, 2020-a, b; 地道, 阪, 2021; 地道, 阪, 2022; 地道, 阪, 2023-a, b）.

1）Bureau van Dijk（ビューロー・ヴァン・ダイク）社は，現在，Moody's Analytics 社の傘下となっている．

本稿では，2021年から Moody's Analytics 社によって運用されているクラウド型のデータ配信プラットフォームである DataHub から，規模の大きなデータを抽出したものを利用し，これまで Osiris データを利用して行ってきた解析を同様に行うことが可能かどうかを検証する．

本稿の構成は以下のようなものである．まず，本研究で扱うデータファイルとディレクトリの構成を与え（II 節），企業の基本情報が収録されたデータファイルのラングリングを行う（III 節）．次に，財務情報が収録されたデータファイルのラングリングを行い，従来から利用されてきた Osiris データによる解析結果との比較を行うことによって再現性が成り立つかを検証する（IV 節）．さらに，株価情報のデータファイルのラングリングを行い，財務情報と同様に Osiris データと比較し，DataHub データの問題点を指摘する（V 節）．最後に，今後の展望と課題について述べる（VI 節）．

付録には，本研究のために利用したコンピュータ環境（付録 A）や，DataHub からのデータ抽出手順（付録 B），抽出に利用した SQL 問合せのスクリプト（付録 C）を与えている．

本研究は，データ解析環境 R[2] を利用しており，Tukey（1977）による探索的データ解析（Exploratory Data Analysis: EDA）の視点に立って行われている[3]．また，本稿の執筆には，Sweave[4] による動的文書生成によって再現可能研究を実施することを試みている[5]．

2） https://www.r-project.org/
3） R については，例えば，地道（2018-c）を参照されたい．また，財務データに関する EDA の実行については，地道（2018-a, b）も参照されたい．
4） https://stat.ethz.ch/R-manual/R-devel/library/utils/doc/Sweave.pdf
5） Sweave を利用した動的文書生成と再現可能研究については，例えば，地道（2018-b）を参照されたい．

II ファイル構成

本稿作成のために利用しているディレクトリ working の構成を図 1 に与える.

```
% tree -d working
working
├── R
├── data
│   ├── 11_tab_financials
│   ├── 15_tab_stock
│   ├── 6_tab_basic
│   └── financials
└── paper
    └── scripts
```

図 1 ：本稿作成のためのディレクトリ

今回 DataHub からダウンロードしたファイルは，図 2 のサブディレクトリ data に格納されている．ここで，（サブ）ディレクトリ data のサブディレクトは以下のようなものである：

6_tab_basic: 企業の基本情報（企業名，企業コード，業種等）のファイル群（ファイル数：160）が格納されたサブディレクトリ

11_tab_financials: 企業の財務情報（売上高，資産合計等）のファイル群（ファイル数：404）が格納されたサブディレクトリ

15_tab_stock: 企業の株価情報（時価総額等）のファイル群（ファイル数：6）が格納されたサブディレクトリ

これらのディレクトリに格納されているいずれのファイルも Parquet[6] 形式のファイルである.

6) Apache Parquet は，効果的にデータを格納し検索できるように設計されたオープンソースの列指向型データファイル形式（column-oriented data file format）である（https://parquet.apache.org 参照）．Parquet は Java，C++，Python，R 等の多くの言語で利用可能である.

```
% tree data
data
├── 11_tab_financials
│   ├── part-00000-1ea5347b-279b-4136-8410-2866bd72c453-c000.snappy.parquet
│   ├── part-00001-1ea5347b-279b-4136-8410-2866bd72c453-c000.snappy.parquet
│   ├── part-00002-1ea5347b-279b-4136-8410-2866bd72c453-c000.snappy.parquet
│   ├── part-00003-1ea5347b-279b-4136-8410-2866bd72c453-c000.snappy.parquet
│   ├── part-00004-1ea5347b-279b-4136-8410-2866bd72c453-c000.snappy.parquet
│   ├── part-00005-1ea5347b-279b-4136-8410-2866bd72c453-c000.snappy.parquet
:   :
│   └── part-00403-1ea5347b-279b-4136-8410-2866bd72c453-c000.snappy.parquet
├── 15_tab_stock
│   ├── part-00000-4a3580e5-ed12-4706-9207-c4bc92069b6e-c000.snappy.parquet
│   ├── part-00001-4a3580e5-ed12-4706-9207-c4bc92069b6e-c000.snappy.parquet
│   ├── part-00002-4a3580e5-ed12-4706-9207-c4bc92069b6e-c000.snappy.parquet
│   ├── part-00003-4a3580e5-ed12-4706-9207-c4bc92069b6e-c000.snappy.parquet
│   ├── part-00004-4a3580e5-ed12-4706-9207-c4bc92069b6e-c000.snappy.parquet
│   └── part-00005-4a3580e5-ed12-4706-9207-c4bc92069b6e-c000.snappy.parquet
└── 6_tab_basic
    ├── part-00000-5da30334-d0e7-4d05-b0de-e50aafbfc954-c000.snappy.parquet
    ├── part-00001-5da30334-d0e7-4d05-b0de-e50aafbfc954-c000.snappy.parquet
    ├── part-00002-5da30334-d0e7-4d05-b0de-e50aafbfc954-c000.snappy.parquet
    ├── part-00003-5da30334-d0e7-4d05-b0de-e50aafbfc954-c000.snappy.parquet
    ├── part-00004-5da30334-d0e7-4d05-b0de-e50aafbfc954-c000.snappy.parquet
    ├── part-00005-5da30334-d0e7-4d05-b0de-e50aafbfc954-c000.snappy.parquet
    :
    └── part-00159-5da30334-d0e7-4d05-b0de-e50aafbfc954-c000.snappy.parquet
```

図 2 ：DataHub からダウンロードしたファイルが格納されたディレクトリ data とそのサブディレクトリ

本稿では，これらのファイルをデータ解析環境 R の arrow パッケージと tidyverse パッケージ群を利用してラングリングする．なお，R とそのパッケージについての簡単な説明を付録 A に与える．

Ⅲ　基本情報ファイルのラングリング

ここでは，（サブ）ディレクトリ `6_tab_basic`（図 2 参照）に格納されているデータファイル群のラングリングについて述べる．事前の調査で，企業の基本情報である業種情報を複数もつ企業が多数含まれることが判明したため，今回のラングリングでは，企業の業種コードを除外し，データとして利用できる一意化された企業に関する情報を抽出し，CSV ファイルとして出力する処理を行った．

具体的な処理としては，DataHub からダウンロードした基本情報を含む

Parquet ファイル群を，ディレクトリ R をカレント（図 1 参照）として，スクリプト 1 を利用して R に一括して読み込み，ラングリングを行った（図 3 参照）.

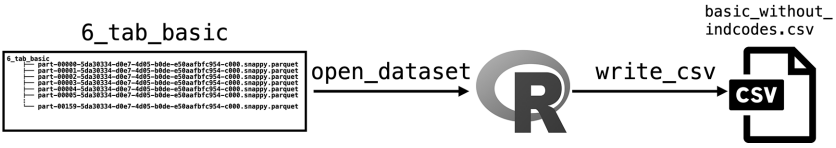

図 3 ：基本情報ファイルのラングリング

スクリプト 1 ：./scripts/1basic.R

```
1  > library(tidyverse)
2  > library(arrow)
3  > library(here)
4  #
5  > basic <- open_dataset(
6    here("../data/6_tab_basic/"),
7    format = "parquet")
8  #
9  > x.basic <- basic %>% dplyr::collect()
10 #
11 > y.basic <- x.basic %>%
12   select(-ends_with("description")) %>%
13   select(-ends_with("code_s_")) %>%
14   distinct(bvd_id_number, name, .keep_all = TRUE) %>%
15   arrange(bvd_id_number)
16 #
17 > write_csv(y.basic, file = "../data/basic_without_indcodes.csv")
```

スクリプト 1 では以下の処理を行っている：

1〜3 行目：パッケージの読み込み

5〜7 行目：関数 open_dataset[7] を利用して，ディレクトリ 6_tab_basic の Parquet ファイル群を一括して読み込んだ後，basic オブジェクトに付値

9 行目：（遅延処理を回避するために）basic オブジェクトを dplyr パッ

7） arrow パッケージに付属する関数 open_dataset は，データが存在するディレクトリを sources 引数に指定することによって，その場所に存在するファイルを全て R 6 オブジェクトとして読み込む．この仕様から，特にファイルが多数存在する場合に有効な関数といえる．

ケージの collect 関数でデータ・フレーム・オブジェクトに変換後，x.basic へ付値

11〜15行目：データを一意化するために，業種分類コードの列を select 関数で削除した後，distinct 関数で一意化したものを，arrange 関数で企業コード bvd_number で並べ替えたものを，オブジェクト y.basic へ付値

17行目：オブジェクト y.basic を CSV ファイル basic_without_indcodes.csv へ出力

以上の処理によって得られた CSV ファイルは以下のようなものである：

スクリプト 2：../data/basic_without_indcodes.csv（先頭10行）

```
 1 bvd_id_number,name,country_iso_code,main_exchange,ipo_date,delisted_date,
     date_of_incorporation,ticker_symbol,isin_number
 2 AE0000001546,BK Global General Trading (FZC),AE,Unlisted,NA,NA,2007-07-09
     T00:00:00Z,NA,NA
 3 AE0000002636,FAC FZE,AE,Unlisted,NA,NA,2011-01-01T00:00:00Z,NA,NA
 4 AE0000003589,Kingston Holdings (FZE),AE,Unlisted,NA,NA,2010-09-30T00:00:00Z
     ,NA,NA
 5 AE0000005993,Resonance Music (FZC),AE,Unlisted,NA,NA,NA,NA,NA
 6 AE0000006541,SUN Pharma Global (FZE),AE,Unlisted,NA,NA,2000-01-01T00:00:00Z
     ,NA,NA
 7 AE0000037163,Jashanmal National CO. (L.L.C.),AE,Unlisted,NA,NA,1976-02-01
     T00:00:00Z,NA,NA
 8 AE0000037226,Emirates Airline,AE,Unlisted,NA,NA,1988-04-18T00:00:00Z,NA,NA
 9 AE0000037241,Falcon Global General Trading LLC,AE,Unlisted,NA,NA,1981-01-15
     T00:00:00Z,NA,NA
10 AE0000037317,Gulf Adhesive Labels Factory LLC,AE,Unlisted,NA,NA,1989-11-02
     T00:00:00Z,NA,NA
```

ここで，各列名は以下を表す：

bvd_id_number：企業コード（ビューロー・ヴァン・ダイク社によって与えられた各企業一意のコード）

name：企業名

country_iso_code：国別コード（ISO2c コード）

main_exchange：主要株式取引所

ipo_date：新規株式公開日（Initial Public Offering: IPO）

delisted_date：上場廃止年月日

date_of_incorporation：創業年月日

ticker_symbol：ティッカーシンボル

isin_number：ISIN（International Securities Identification Number）コード

なお，CSV ファイル basic_without_indcodes.csv のサイズは以下のようなものである：

スクリプト 3 ：ファイルサイズ

```
% ls -l ../data/basic_without_indcodes.csv
-rw-r--r--@ 1 masa  staff  2481877020 11 18 10:16 basic_without_indcodes.csv
% wc -l ../data/basic_without_indcodes.csv
 29083747 basic_without_indcodes.csv
```

これらのファイル情報から，サイズは約 2.5 GB であり，行数は29,083,747であることがわかる．なお，CSV ファイルの 1 行目はヘッダー行であるので，結果的に29,083,746社の情報があることがわかる．

Ⅳ　財務情報ファイルのラングリング

1. 財務情報ファイルの読み込みと再配置

ここでは，（サブ）ディレクトリ 11_tab_financials（図 2 参照）に格納されている財務情報（information of financials）のデータファイル群のラングリングについて述べる．事前の調査で，今回のラングリングでは，1970年から2023年のデータが存在することがわかったため，一括して読み込んだデータを年毎に Parquet ファイルとして出力する処理を行った．ディレクトリ R をカレント（図 1 参照）として，R 上でスクリプト 4 を実行する（図 4 参照）．

図 4 ：財務情報ファイルのラングリング

スクリプト 4 ：**./scripts/2financials.R**

```
1  # financials
2  > library(arrow)
3  > library(tidyverse)
4  > library(here)
5  #
6  > financials <- open_dataset(
7    here("../data/11_tab_financials/"),
8    format = "parquet")
9  #
10 > financials %>%
11   write_dataset(path = "../data/financials",
12                 partitioning = "fiscal_year")
```

スクリプト 4 では以下の処理を行っている：

2〜4 行目：パッケージの読み込み

6〜8 行目：関数 `open_dataset` を利用して，ディレクトリ `11_tab_financials` の Parquet ファイル群を一括して読み込んだ後，`financials` オブジェクトに付値

10〜12行目：オブジェクト `financials` を関数 `write_dataset` を利用して，会計年度（fiscal year）毎に分割（`partitioning = "fiscal_year"`）し，`path` 引数で指定したディレクトリ `financials` のサブディレクトリ `fiscal_year=xxxx%20%28USD%29` の Parquet ファイル `part-0.parquet` へ出力（ここで，`xxxx` は，`1970` から `2023` の値をとる．図 5 も参照）

以上の処理によって，DataHub から抽出された404個の財務データファイル

が再構成され，ディレクトリ financials の会計年度毎のサブディレクトリ内に Parquet ファイルとして格納される．通常，このような処理を行うためには，繰り返し処理を行う必要があるが，arrow パッケージに付属する open_dataset 関数と write_dataset 関数の利用することによって，この処理を効率的に行うことができることに留意されたい．

```
% tree financials
financials
├── fiscal_year=1970%20%28USD%29
│   └── part-0.parquet
├── fiscal_year=1979%20%28USD%29
│   └── part-0.parquet
├── fiscal_year=1980%20%28USD%29
│   └── part-0.parquet
:
└── fiscal_year=2023%20%28USD%29
    └── part-0.parquet
```

図 5 ：会計年度毎に分割し，サブディレクトリ `fiscal_year=xxxx%20%28USD%29` へ格納された Parquet ファイル `part-0.parquet`（群）

2. 2020会計年度の財務情報ファイルの再読み込みと基本情報との結合

ここでは，会計年度ごとに再配置された Parquet ファイル（2020会計年度分）と III 節で得られた企業情報に関する CSV ファイル basic_without_indcodes.csv を R へ読み込み，結合（join）する．ディレクトリ R をカレント（図 1 参照）として，R 上でスクリプト 5 を実行する（図 6 参照）．

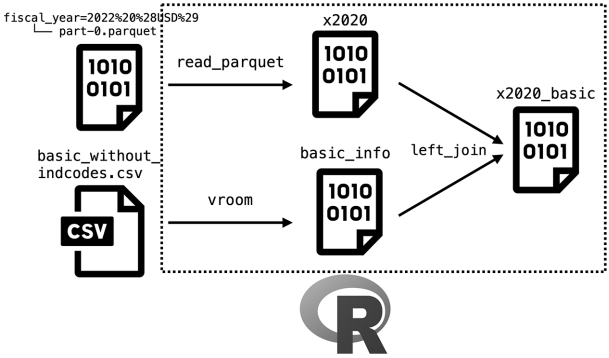

図 6 ：財務情報と基本情報の結合

スクリプト 5 ：./scripts/2financials-joinf.R（先頭10行）

```
1 > x2020 <- arrow::read_parquet(paste0("../data/financials/fiscal_year="
    ,2020,"%20%28USD%29/part-0.parquet"))
2 > basic_info <- vroom::vroom("../data/basic_without_indcodes.csv")
3 > x2020_basic <- x2020 %>% left_join(basic_info, by = "bvd_id_number")
```

スクリプト 5 では以下の処理を行っている：

1 行目：2020会計年度の財務データファイル financials/fiscal_year=2020%20%28USD%29/part-0.parquet を読み込み，オブジェクト x2020 へ付値

2 行目：企業情報ファイルを読み込み，オブジェクト basic_info へ付値

3 行目：企業コードをキーにして（by = "bvd_id_number"），オブジェクト x2020 にオブジェクト basic_info を左外部結合（left_join）したものをオブジェクト x2020_basic へ付値

以上の処理によって，2020会計年度の財務情報のオブジェクト x2020 と企業情報のオブジェクト basic_info が企業コード（bvd_id_number）によって結合され，財務情報のオブジェクト x2020 には含まれていなかった企業名等の情報がオブジェクト x2020_basic へ追加されることになる．例えば，企業コード（bvd_id_number），社名（name），本社が存在する国情報（country_iso_code）に関する情報が利用できる：

```
>  x2020_basic %>% select(bvd_id_number, name, country_iso_code)
# A tibble: 18,717,639 × 3
   bvd_id_number    name                                      country_iso_code
   <chr>            <chr>                                     <chr>
 1 UA00851519       Agroholding Avangard Private JSC AT       UA
 2 TN1249786D       STE Industrielle de Papier et Carton      TN
 3 VN0100107927     Cong TY Tnhh MOT Thanh Vien Thuong MAI Th… VN
 4 CN52605PC        Fujian nanwang environment protection sci… CN
 5 CN48529PC        Seichi technologies shenzhen limited      CN
 6 TH0107562000190 S Hotels and Resorts PCL                   TH
 7 SE5562292820     Slottsviken Fastighetsaktiebolag (Publ)   SE
 8 TH0105528018144 KCG Corporation CO LTD                     TH
 9 RS07037783       Dedinje AD Beograd                        RS
10 FR892226762      Oncodesign Precision Medicine SA          FR
# \UTF{2139} 18,717,629 more rows
```

なお，作成されたオブジェクト x2020_basic の情報は以下のようなものである：

```
> x2020_basic
# A tibble: 18,717,639 × 78
   bvd_id_number    original_currency current_assets stock debtors
   <chr>            <chr>                     <dbl> <dbl>   <dbl>
 1 UA00851519       NA                           NA    NA      NA
 2 TN1249786D       NA                           NA    NA      NA
 3 VN0100107927     NA                           NA    NA      NA
 4 CN52605PC        NA                           NA    NA      NA
 5 CN48529PC        NA                           NA    NA      NA
 6 TH0107562000190 NA                           NA    NA      NA
 7 SE5562292820     NA                           NA    NA      NA
 8 TH0105528018144 NA                           NA    NA      NA
 9 RS07037783       NA                           NA    NA      NA
10 FR892226762      NA                           NA    NA      NA
# \UTF{2139} 18,717,629 more rows
# \UTF{2139} 73 more variables: other_current_assets <dbl>,
#    cash_cash_equivalent <dbl>, fixed_assets <dbl>,
#    tangible_fixed_assets <dbl>, intangible_fixed_assets <dbl>,
#    other_fixed_assets <dbl>, total_assets <dbl>, current_liabilities <dbl>,
#    loans <dbl>, creditors <dbl>, other_current_liabilities <dbl>,
#    non_current_liabilities <dbl>, long_term_debt <dbl>, …
# \UTF{2139} Use `print(n = ...)` to see more rows
```

この結果から，企業数は，18,717,639社であり，78個の列（変数）があることがわかる．

3. Osiris データとの比較

筆者らのこれまでの研究では，世界の上場企業のデータが収録されたデー

タベース Osiris も利用しており，様々な理由[8] から，今後は DataHub から抽出したデータを利用する必要があるため，その品質を調べ Osiris データと比較することによって，データの検証を行う．本稿では，2022年版の Osiris から抽出したデータセット DS-Orbis-C-2022 を利用して比較を行う．なお，このデータセットには，世界の**上場企業**に関する**連結決算**（Consolidated accounting）に関するデータが収録されている．

前小節で，ラングリングされた DataHub のオブジェクト `x2020_basic` を利用してデータセット DS-Orbis-C-2022 と同等の仕様となるように変換する．その際，事前の調査から，以下の仕様で抽出した：

（CL1）上場情報（`main_exchange`）：非上場（`Unlisted`）ではない

（CL2）有価証券報告書の種類（`filing_type`）：年次事業報告書（`Annual report`）

（CL3）連結コード（`consolidation_code`）：連結（`C1`（連結財務諸表のみを保有している企業）または `C2`（連結財務諸表を保有しており，何らかの理由で単体財務諸表も保有している企業））

（CL4）決算月数（`number_of_months`）：12カ月（`12`）

スクリプト 6 は，仕様（CL1）〜（CL4）を実現するためのものである．

スクリプト 6 ：`./scripts/2financials-listed-C.R`（先頭10行）

```
> y2020 <- x2020_basic %>%
  filter(main_exchange != "Unlisted", filing_type == "Annual_report") %>%
  filter(consolidation_code == "C1" | consolidation_code == "C2") %>%
  filter(number_of_months == 12)
```

なお，抽出されたオブジェクト `y2020` は，以下のようなものである：

8） 本稿で DataHub からデータを抽出することを扱う理由としては，これまでは，Osiris データの抽出については，ベンダーの協力のもとで，全ての収録企業のデータを抽出していたが，今後は，大量のデータを抽出するためのサービスが DataHub に移行するという背景が最も大きなものである．

```
> y2020
# A tibble: 48,480 × 78
   bvd_id_number   original_currency current_assets     stock   debtors
   <chr>           <chr>                     <dbl>     <dbl>     <dbl>
 1 CA41751NC       CAD                      233645         0    0
 2 KR1351110045041 KRW                   149817637  15453865    2.81e7
 3 IN30447FI       INR                   383817628 195760268    1.07e8
 4 IL30663GE       ILS                   214576043 101491754    2.40e7
 5 CA31355NC       CAD                     5935055         0    0
 6 CN31019PC       CNY                   311643319 150666264    2.95e7
 7 CA41560NC       CAD                     5732732         0    1.65e6
 8 CN30617PC       CNY                   611064111  98302189    2.57e8
 9 CN47003PC       CNY                    24699382  21734685    2.52e5
10 LK30195FS       LKR                    12026312   7611101    0
# ^^e2^^84^^b9 48,470 more rows
# ^^e2^^84^^b9 73 more variables: other_current_assets <dbl>,
#   cash_cash_equivalent <dbl>, fixed_assets <dbl>,
#   tangible_fixed_assets <dbl>, intangible_fixed_assets <dbl>,
#   other_fixed_assets <dbl>, total_assets <dbl>,
#   current_liabilities <dbl>, loans <dbl>, creditors <dbl>,
#   other_current_liabilities <dbl>, non_current_liabilities <dbl>, …
```

ここで，オブジェクト y2020 には重複する企業が存在することがわかっており[9)]，そのような企業がどのようなものかを調べるには以下のように入力する：

```
> dup2020 <- y2020 %>%
+   group_by(bvd_id_number) %>%
+   count() %>%
+   filter(n >= 2) %>%
+   pull(bvd_id_number)
> length(dup2020)
[1] 91
```

この結果から，91社の企業が重複する情報をもつことがわかる．本稿では，このような企業は以下のように入力することによってデータから排除する仕様とした：

9) どのような情報ソースからデータを採集したかを表す列（変数）source_for_publicly_quoted_companies_が複数種類存在する企業がその対象となる．

```
> `%nin%` <- Negate(`%in%`)
> z2020 <- y2020 %>% filter(bvd_id_number %nin% dup2020)
> z2020
# A tibble: 48,298 × 78
   bvd_id_number   original_currency current_assets     stock  debtors
   <chr>           <chr>                     <dbl>     <dbl>    <dbl>
 1 CA41751NC       CAD                      233645         0   0
 2 KR1351110045041 KRW                   149817637  15453865   2.81e7
 3 IN30447FI       INR                   383817628 195760268   1.07e8
 4 IL30663GE       ILS                   214576043 101491754   2.40e7
 5 CA31355NC       CAD                     5935055         0   0
 6 CN31019PC       CNY                   311643319 150666264   2.95e7
 7 CN30617PC       CNY                   611064111  98302189   2.57e8
 8 CN47003PC       CNY                    24699382  21734685   2.52e5
 9 LK30195FS       LKR                    12026312   7611101   0
10 DE2150423028    EUR                  3132644622 659509427   8.25e8
# ^^e2^^84^^b9 48,288 more rows
# ^^e2^^84^^b9 73 more variables: other_current_assets <dbl>,
#   cash_cash_equivalent <dbl>, fixed_assets <dbl>,
#   tangible_fixed_assets <dbl>, intangible_fixed_assets <dbl>,
#   other_fixed_assets <dbl>, total_assets <dbl>,
#   current_liabilities <dbl>, loans <dbl>, creditors <dbl>,
#   other_current_liabilities <dbl>, non_current_liabilities <dbl>, …
```

ここで，文字列を含まない（排除）するための演算子 `%nin%` を定義しており，関数 filter を利用して，重複する企業コードのベクトルを排除している．なお，最終的に得られたオブジェクト `z2020` に含まれる企業数は，48298社である（図 7 も参照）．

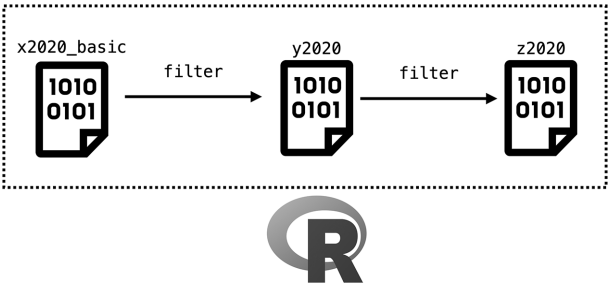

図 7 ：（基本情報結合済み）財務情報のラングリング

次に，Osiris データとの比較を行う．これまでの研究（e.g. 地道, 2018-a; 地道, 阪, 2021）では，Osiris データの効率的な前処理について検討されており，本稿で利用するデータは，2022年版 Osiris から連結決算を主体として抽出されたものを前処理したデータセット DS-Orbis-C-2022（ファイル名：

OsirisC2022.csv）を利用する．さらに，DataHub データから抽出したオブジェクト z2020 との整合性を保たせるために，ラングリングも行う．一連の処理は R 上でスクリプト 7 を実行することによって行う（図 8 も参照）．

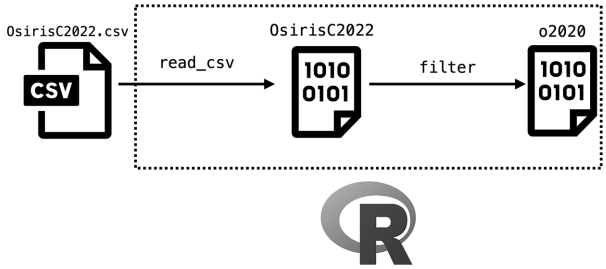

図 8 ：Osiris データのラングリング

スクリプト 7 ：./scripts/2financials-Osiris.R

```
1 > OsirisC2022 <- read_csv("../data/OsirisC2022.csv")
2 > o2020 <- OsirisC2022 %>%
3   filter(year == 2020, month == 12) %>%
4   select(sales, employees, assets_total)
```

スクリプト 7 では以下の処理を行っている：

1 行目：データセットファイル OsirisC2022.csv を読み込み，オブジェクト OsirisC2022 へ付値

2〜4 行目：オブジェクト OsirisC2022 から，2020会計年度で，資産合計と売上高が正値であり，かつ決算月数が12カ月のものを抽出（filter(year == 2020, month == 12)）し，さらに，売上高と従業員数，資産合計の列を選択（selectsales, employees, assets_total)）したものをオブジェクト o2020 へ付値

ラングリングされた結果として得られたオブジェクト o2020 は以下のようなものである：

```
> o2020
# A tibble: 60,339 × 3
       sales employees assets_total
       <dbl>     <dbl>        <dbl>
 1 559151000   2300000    252496000
 2 386064000   1298000    321195000
 3 274515000    147000    323888000
 4 322267200    384065    265314686
 5 295924334    432003    380744298
 6  40018000        NA     92007000
 7 268706000    300000    230715000
 8 273500800    662575    610008241
 9 178574000     72000    332750000
10 180543000     87000    379268000
# ^^e2^^84^^b9 60,329 more rows
```

この結果から，オブジェクト o2020 に含まれる企業数は，60339社である．

DataHub データにもとづくオブジェクト z2020 と Osiris データによるオブジェクト o2020 を利用して，可視化によって両者のデータを比較する．例として，以下のように入力することによって，2020会計年度の対数資産合計と対数売上高の散布図を描く．

```
> p.z <- z2020 %>%
+   filter(total_assets > 0, sales > 0) %>%
+   ggplot(aes(log(total_assets), log(sales))) +
+   geom_point(size = 0.1, alpha = 0.2) +
+   xlim(c(0, 30)) +
+   ylim(c(0, 30)) +
+   ggtitle("DataHub Data")
> p.o <- o2020 %>%
+   filter(assets_total > 0, sales > 0) %>%
+   ggplot(aes(log(assets_total*1000), log(sales*1000))) +
+   geom_point(size = 0.1, alpha = 0.2) +
+   xlim(c(0, 30)) +
+   ylim(c(0, 30)) +
+   ggtitle("Osiris Data (* 1000)")
```

上のスクリプトを実行することによって得られるオブジェクト p.z（DataHub データにもとづく散布図）と p.o（Osiris データにもとづく散布図）を利用して描いた散布図を図 9 に与える[10]．この図から，両方のデータセットは，ほぼ同様のものであることがわかる．なお，DataHub データは単位が

「米ドル」に対して，Osiris データは「米千ドル」であることから，単位を合わせるために，Osiris データは1000倍されていることに留意する必要がある．この事実から，Osiris データは1000ドル以下の値は丸められており，千ドル以下の値は観測されないため，散布図は「切り取られた形状」（trancated shape）が観察される[11]．

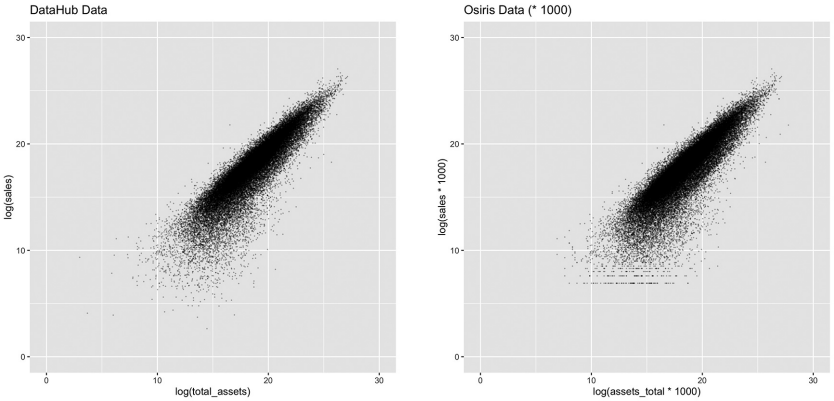

図 9 ：2020会計年度の対数資産合計と対数売上高の散布図：左図：DataHub にもとづく散布図，右図：Osiris にもとづく散布図

さらに，これらのオブジェクトを利用した統計モデリングの結果を比較しよう．地道（2017-a），Jimichi *et al.*（2018）では，Osiris データにおける売上高に対して以下のような両対数モデルが有効であることが報告されている：

$$\log(\texttt{sales}_i) = \alpha_0 + \alpha_1 \log(\texttt{employees}_i) + \alpha_2 \log(\texttt{assets\_total}_i) + \log(\epsilon_i) \tag{1}$$

ここで，対数誤差 $\log(\epsilon_i)$（$i=1, \ldots, n$）は独立に同一の非対称ティー分布 ST $(0, \omega^2, \alpha, \nu)$ に従う：

10）`plot` 関数，または `print` 関数を利用して描く．例えば，`plot(p.z)` と入力することによって描くことができる．
11）この意味で，DataHub データの方がより精確な分析が可能となる．

$$\log(\epsilon_i) \overset{\text{i.i.d.}}{\sim} \mathsf{ST}(0, \omega^2, \alpha, \nu) \tag{2}$$

(詳細については，地道, 2017-a; Jimichi *et al.*, 2018 を参照)．このことを，2020会計年度の DataHub データと Osiris データで比較検証する．

まず，非対称ティー誤差をもつ両対数モデルを DataHub データへ当てはめた結果を以下に与える：

DataHub データへの非対称ティー誤差をもつ両対数モデルの当てはめ

```
> library(sn)
> selm.ST.z2020 <- z2020 %>%
+   filter(sales > 0, number_of_employees > 0, total_assets > 0) %>%
+   selm(log(sales) ~ log(number_of_employees) + log(total_assets), family = "ST", data = .)
> selm.ST.z2020 %>% summary(param.type = "DP")
Call: selm(formula = log(sales) ~ log(number_of_employees) + log(total_assets),
    family = "ST", data = .)
Number of observations: 29788
Family: ST
Estimation method: MLE
Log-likelihood: -39515.53
Parameter type: DP

DP residuals:
       Min         1Q     Median         3Q        Max
-11.210542  -0.990975  -0.457529  -0.007665   5.736288

Regression coefficients
                          estimate   std.err   z-ratio Pr{>|z|}
(Intercept.DP)           3.438e+00 4.663e-02 7.372e+01        0
log(number_of_employees) 3.570e-01 4.280e-03 8.342e+01        0
log(total_assets)        6.926e-01 3.462e-03 2.001e+02        0

Parameters of the SEC random component
      estimate std.err
omega   0.7724   0.010
alpha  -0.9865   0.036
nu      3.1321   0.066
```

次に，Osiris データへ非対称ティー誤差をもつ両対数モデルを当てはめた結果を以下に与える：

Osiris データへの非対称ティー誤差をもつ両対数モデルの当てはめ

```
> selm.ST.o2020 <- o2020 %>%
+   filter(sales > 0, employees > 0, assets_total > 0) %>%
+   selm(log(sales*1000) ~ log(employees) + log(assets_total*1000), family = "ST", data = .)
> selm.ST.o2020 %>% summary(param.type='DP')
Call: selm(formula = log(sales * 1000) ~ log(employees) + log(assets_total *
    1000), family = "ST", data = .)
Number of observations: 37878
Family: ST
Estimation method: MLE
Log-likelihood: -49850.74
Parameter type: DP

DP residuals:
      Min        1Q    Median        3Q       Max
-9.770693 -0.955793 -0.437430  0.004824  5.749692

Regression coefficients
                          estimate   std.err   z-ratio Pr{>|z|}
(Intercept.DP)          3.537e+00 4.018e-02 8.804e+01        0
log(employees)          3.632e-01 3.810e-03 9.534e+01        0
log(assets_total *1000) 6.845e-01 3.102e-03 2.207e+02        0

Parameters of the SEC random component
      estimate std.err
omega   0.7490   0.009
alpha  -0.9488   0.031
nu      3.0351   0.056
```

以上の結果から，母数の推定値は非常に近く，検定に関する有意性は全く同じ結果を与える．

さらに，回帰診断に関するプロットを図10に与える．

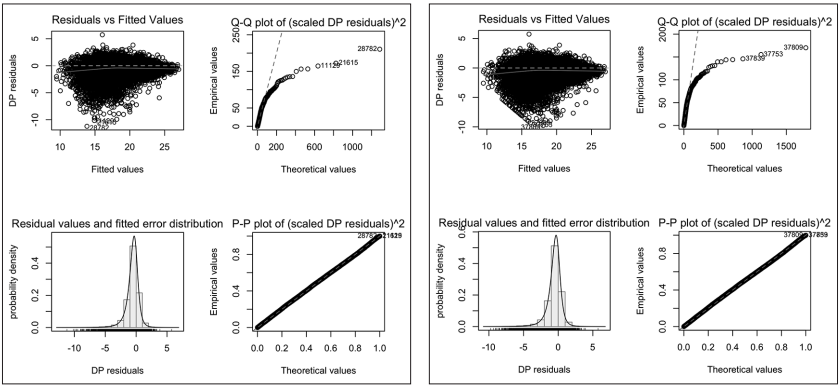

図10：回帰診断に関するプロット：左図：**DataHub** にもとづく診断プロット，右図：**Osiris** にもとづく診断プロット

回帰診断のプロットも，非常に似通った結果を与えることがわかった．また，両方の結果とも，この両対数モデルの当てはまりが良好であることを表していることもわかる[12]．

これらの結果は，異なったデータベースから抽出されたデータセットに関して，同様の結果を導くことができたことを意味し，DataHub データを利用することによって，これまで行ってきた Osiris データにもとづく上場企業のデータ解析に関する研究が DataHub データを利用しても同等の品質を与えることの根拠が得られたことになる[13]．さらに，地道（2017），Jimichi *et al.*（2018）で得られた統計モデルの有効性に関する結果が年度を経て，再度有効であることも検証できたことを意味し，この意味で再現性が成り立つことも確かめることができていることにも留意しよう．

V 株価情報ファイルのラングリング

1. 株価情報ファイルの読み込みとラングリング

ここでは，（サブ）ディレクトリ `15_tab_stock`（図 2 参照）に格納されているデータファイル群のラングリングについて述べる．具体的な処理としては，DataHub からダウンロードした株価情報の Parquet ファイル群を，ディレクトリ `R` をカレント（図 1 参照）として，スクリプト 8 を利用して R に一括して読み込み，型変換を行ったあと列を選択し，CSV ファイルへ書き出すというラングリングを行った（図11も参照）．

図11：株価情報ファイルのラングリング

12）尺度調整済み直接母数残差の 2 乗を使った Q-Q プロットは，それほど良い結果を与えていないが，このプロットはセンシティブであることが知られている．

13）ただし，ここで利用しているのは，売上高，従業員数，資産合計等の指標に限られたものであり，今後，その他の指標についても引き続き検証する必要がある．

スクリプト 8 ：./scripts/3stock.R

```
> library(arrow)
> library(tidyverse)
> library(here)
> stock <- open_dataset(
  here("../data/15_tab_stock/"),
  format = "parquet") %>%
  dplyr::collect() %>%
  select(
    bvd_id_number,
    fiscal_year,
    closing_date,
    market_price_year,
    market_capitalisation_mil_) %>%
  arrange(bvd_id_number, closing_date) %>%
  write_csv("../data/stock.csv")
```

スクリプト 8 では以下の処理を行っている：

1〜3 行目：パッケージの読み込み

5〜6 行目：関数 open_dataset を利用して，ディレクトリ 15_tab_stock の Parquet ファイル群を一括して読み込み

7 目：dplyr パッケージの collect 関数でデータ・フレーム・オブジェクトに変換

8〜14行目：select 関数で列を選択した後，arrange 関数で企業コード bvd_number と決算年月日 closing_date で並べ替え

15行目：CSV ファイル stock.csv へ出力

以上の処理によって得られた CSV ファイルは以下のようなものである：

スクリプト 9 ：../data/stock.csv（先頭10行）

```
bvd_id_number ,fiscal_year,closing_date,market_price_year,market_
    capitalisation_mil_
AE115877GU,2021 (USD),2021-12-31T00:00:00Z,2021,272999.99
AE115877GU,2022 (USD),2022-12-31T00:00:00Z,2022,270800
AE115877GU,2023 (USD),NA,2023,NA
AE140001GU,2018 (USD),2018-12-31T00:00:00Z,2018,5319.28
AE140001GU,2019 (USD),2019-12-31T00:00:00Z,2019,5619.93
AE140001GU,2020 (USD),2020-12-31T00:00:00Z,2020,6706.91
AE140001GU,2021 (USD),2021-12-31T00:00:00Z,2021,6799.42
AE140001GU,2023 (USD),NA,2023,NA
AE140007GU,2020 (USD),2020-12-31T00:00:00Z,2020,334.71
```

ここで，各列は以下を表している：

bvd_id_number：企業コード（ビューロー・ヴァン・ダイク社によって与えられた各企業一意のコード）

fiscal_year：会計年度＋通過単位（USD）

closing_date：決算年月日

market_price_year：期末株価

market_capitalisation_mil_：株式時価総額（単位：百万米ドル）

なお，CSV ファイル stock.csv のサイズは以下のようなものである：

スクリプト10：ファイルサイズ

```
% ls -l ../data/stock.csv
-rw-r--r--@ 1 masa  staff  37554327 11 30 11:06 ../data/stock.csv
% wc -l ../data/stock.csv
  727559 ../data/stock.csv
```

これらのファイル情報から，サイズは約 37.5 MB であり，行数は727,559であることがわかる．

注意 V. 1．DataHub に収録されている株式時価総額（market_capitalisation_mil_）の単位は，百万米ドルではなく本社が存在する国の現地通貨となっており，各国間の比較を行うことができない．

2. 2020会計年度の株価情報ファイルの再読み込みと基本情報及び財務情報との結合

ここでは，前小節で出力された株価情報に関する CSV ファイル stock.csv を R へ読み込み，2022会計年度の連結決算の上場企業に関する財務情報をもつオブジェクト z2020 と結合する．ディレクトリ R をカレント（図１参照）として，R 上でスクリプト11を実行する（図12も参照）．

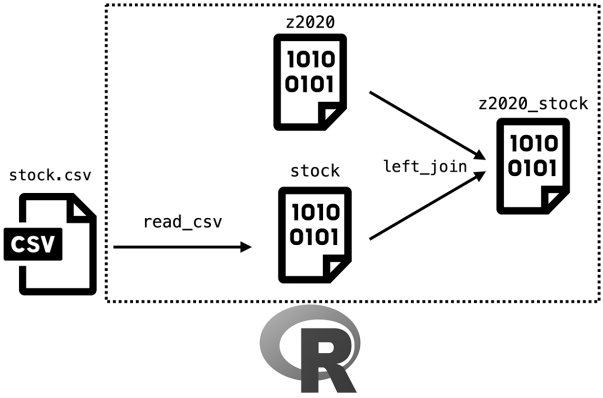

図12：株価情報と財務・基本情報の結合

スクリプト11：./scripts/3stock-joinf.R（先頭10行）

```
1 > stock <- read_csv("../data/stock.csv")
2 > z2020_stock <- z2020 %>%
3   left_join(stock, by = c("bvd_id_number", "closing_date"))
```

スクリプト5では以下の処理を行っている：

1行目：株価情報に関するCSVファイル`stock.csv`を読み込み，オブジェクト`stock`へ付値

2〜3行目：企業コードと決算年月日をキーにしてオブジェクト`z2020`にオブジェクト`stock`を左外部結合（`left_join`）したものをオブジェクト`z2020_stock`へ付値

以上の処理によって，2022会計年度の連結決算の上場企業に関する財務情報をもつオブジェクト`z2020`と株価情報をもつオブジェクト`stock`が結合され，財務情報のオブジェクト`z2020`には含まれていなかった時価総額等の情報が追加されることになる．例えば，`z2020`オブジェクトに由来する企業コード（`bvd_id_number`），社名（`name`），当期純利益（`p_l_for_period_net_income_`），に加えて，`stock`オブジェクトに由来する株式時価総額（`market_capitalisation_mil_`）の情報が利用できる：

```
> z2020_stock %>%
+ select(bvd_id_number, name, p_l_for_period_net_income_, market_capitalisation_mil_)
# A tibble: 48,298 × 4
   bvd_id_number   name   p_l_for_period_net_i…^^c2^^b9market_capitalisatio…^^c2^^b2
   <chr>           <chr>                  <dbl>                  <dbl>
 1 CA41751NC       EMP …                  -80518                     NA
 2 KR1351110045041 Seow…                15400995                 143220
 3 IN30447FI       Amar…                87998453                 145788.
 4 IL30663GE       Mayt…                47987557                   5226.
 5 CA31355NC       Clea…                 -996088                     56
 6 CN31019PC       Zhuz…                24309629                   3840.
 7 CN30617PC       Shan…                26243859                  12406.
 8 CN47003PC       Shen…                -2031095                     NA
 9 LK30195FS       Regn…                  938984                   1106.
10 DE2150423028    Jung…               160304571                   1818.
# ^^e2^^84^^b9 48,288 more rows
# ^^e2^^84^^b9abbreviated names: ^^c2^^b9^^e2^^80^^8bp_l_for_period_net_income_,
#   ^^c2^^b2^^e2^^80^^8bmarket_capitalisation_mil_
```

なお，作成されたオブジェクト z2020_stock の情報は以下のようなものである：

```
> z2020_stock
# A tibble: 48,298 × 81
   bvd_id_number   original_currency current_assets     stock  debtors
   <chr>           <chr>                     <dbl>     <dbl>    <dbl>
 1 CA41751NC       CAD                      233645         0   0
 2 KR1351110045041 KRW                   149817637  15453865   2.81e7
 3 IN30447FI       INR                   383817628 195760268   1.07e8
 4 IL30663GE       ILS                   214576043 101491754   2.40e7
 5 CA31355NC       CAD                     5935055         0   0
 6 CN31019PC       CNY                   311643319 150666264   2.95e7
 7 CN30617PC       CNY                   611064111  98302189   2.57e8
 8 CN47003PC       CNY                    24699382  21734685   2.52e5
 9 LK30195FS       LKR                    12026312   7611101   0
10 DE2150423028    EUR                  3132644622 659509427   8.25e8
# ^^e2^^84^^b9 48,288 more rows
# ^^e2^^84^^b9 76 more variables : other_current_assets <dbl>,
# cash_cash_equivalent <dbl>,fixed_assets <dbl>,
# tangible_fixed_assets <dbl>,intangible_fixed_assets <dbl>,
# other_fixed_assets <dbl>,total_assets <dbl>,
# current_liabilities <dbl>,loans <dbl>,creditors <dbl>,
# other_current_liabilities <dbl>,non_current_liabilities <dbl>,…
```

この結果から，企業数は，48298社であり，81個の列（変数）があることがわかる．

3. Osiris データの比較

財務データに関して行った DataHub データと Osiris データを利用した場合の結果の比較を，株価データに関しても行う．スクリプト12は，Osiris

データに関して，DataHub データの2020会計年度の連結決算の上場企業に関する財務データと株価データを結合したオブジェクト z2020_stock（決算月数12カ月）と同様のオブジェクトを得るためのものである．（図13も参照）．

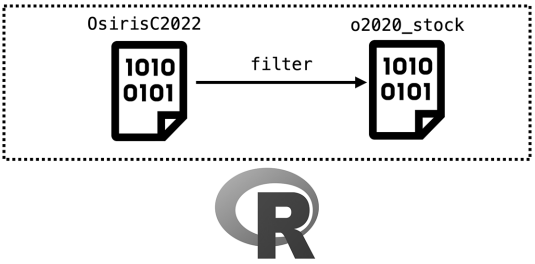

図13：Osiris データにおける株価情報のラングリング

スクリプト12：./scripts/3stock-Osiris.R

```
> o2020_stock <- OsirisC2022 %>%
  filter(year == 2020, month == 12) %>%
  select(net_income, market_cap)
```

スクリプト12では，Osiris データのオブジェクト OsirisC2022 から，2020会計年度で決算月数12カ月のものを抽出し，さらに当期純利益と株式時価総額の列を抽出している．抽出されたオブジェクト o2020_stock は以下のようなものである：

```
> o2020_stock
# A tibble: 60,339 × 2
   net_income market_cap
        <dbl>      <dbl>
 1   13510000    397486.
 2   21331000   1634168.
 3   57411000   2007837.
 4    5064500     58929.
 5    2908384      6531.
 6    4881000     47629
 7    7179000     89399.
 8    8996898     62065.
 9  -22440000    174288.
10  -21680000         NA
# ^^e2^^84^^b9 60,329 more rows
```

では，DataHub データにもとづくオブジェクト z2020_stock と Osiris データによるオブジェクト o2020_stock を利用して，可視化によって両者のデータを比較する．2020会計年度の対数当期純利益と対数時価総額の散布図を描く．

```
> p.z.s <- z2020_stock %>%
+   filter(p_l_for_period_net_income_ > 0, market_capitalisation_mil_ > 0) %>%
+   ggplot(aes(log(p_l_for_period_net_income_),
+              log(market_capitalisation_mil_))) +
+            geom_point(size = 0.8, alpha = 0.2) +
+   xlim(c(0, 30)) + ylim(c(-5, 25))
> p.o.s <- o2020_stock %>%
+   filter(net_income > 0, market_cap > 0) %>%
+   ggplot(aes(log(net_income*1000), log(market_cap))) +
+   geom_point(size = 0.8, alpha = 0.2) +
+   xlim(c(0, 30)) + ylim(c(-5, 25))
```

上のスクリプトを実行することによって得られるオブジェクト p.z.s（DataHub データにもとづく散布図）と p.o.s（Osiris データにもとづく散布図）を利用して描いた散布図を図14に与える．この図から，両データは異なったものであることがわかる．この理由は，注意 V. 1 にも述べたように，DataHub の株価は現地通貨にもとづくためであり，このことから，複数の国の

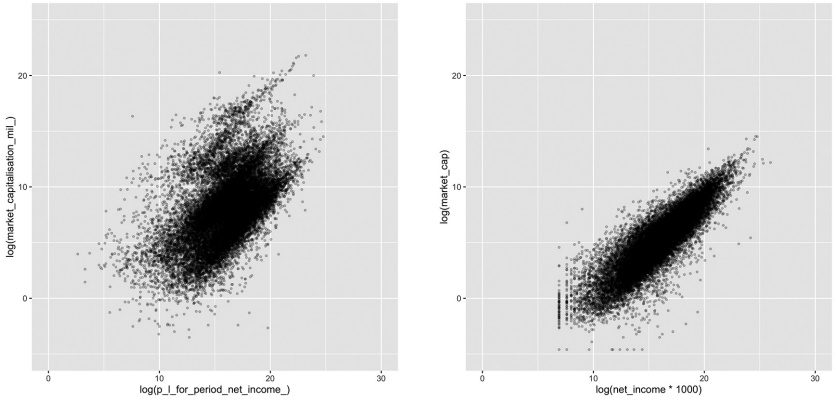

図14：2020会計年度の対数当期純利益と対数時価総額の散布図：左図：DataHub にもとづく散布図，右図：Osiris にもとづく散布図

企業を同時に扱うためには，通貨に関する何らかの換算が必要となるため，本稿ではこれ以上の議論は行わず，別の機会に譲ることにする．

VI おわりに

本稿では，DataHub から抽出されたデータのラングリングと，Osiris データによる解析等の結果を比較することによって，再現性の検証を行った．結果として，2020会計年度の連結決算にもとづく世界の上場企業の財務データに関しては，再現することを確認することができた．ただし，株価に関するデータに関しては，DataHub の仕様から，再現性を確認することが難しいことがわかった．

今後は，DataHub データに関して以下のことのことを考察する必要がある：

- 株価に関連するデータ解析
- Orbis データにもとづく解析結果の再現性の検証
- mdx 環境[14] でのラングリングと解析

これらの事項ついては，今後の課題としたい．

（筆者は関西学院大学商学部教授）

参考文献

［1］ 地道正行（2017-a）『R による対数非対称正規線形モデルによる財務データの統計モデリング』，商学論究，第64巻，第 5 号，pp. 159-185，関西学院大学商学研究会.
［2］ 地道正行（2017-b）『R を利用した非対称分布族にもとづく財務データの統計モデリング』，経済学論究，第71巻，第 2 号，pp. 141-174，関西学院大学経済学部研究会.
［3］ 地道正行（2018-a）『探索的財務ビッグデータ解析：前処理，データラングリング，再現可能性』，商学論究，第66巻，第 1 号，pp. 1-32，関西学院大学商学研究会.
［4］ 地道正行（2018-b）『探索的財務ビッグデータ解析：データ可視化，統計モデリング，モデル選択，モデル評価，動的文書生成，再現可能研究』，商学論究，第66巻，第 2 号，pp. 1-41，関西学院大学商学研究会.
［5］ 地道正行（2018-c）『データサイエンスの基礎：R による統計学独習』，裳華房.
［6］ 地道正行（2020-a）『探索的財務ビッグデータ解析：前処理の並列化』，商学論究，

14）https://mdx.jp/

第67巻，第 3 号，pp. 1-19，関西学院大学商学研究会.
［7］ 地道正行（2020-b）『探索的財務ビッグデータ解析：PG-Strom によるデータラングリングの並列化』，商学論究，第68巻，第 1 号，pp. 1-34，関西学院大学商学研究会.
［8］ Jimichi, M., D. Miyamoto, C. Saka, and S. Nagata（2018）Visualization and statistical model-ing of financial big data: Double-log modeling with skew-symmetric error distributions, *Japanese Journal of Statistics and Data Science*, Vol. 1, No. 2, pp. 347-371, https://doi.org/10.1007/s42081-018-0019-1
［9］ 地道正行，阪智香（2021）『財務データと ESG レーティングデータの前処理と結合』，商学論究，第68巻，第 3 号，pp. 79-116，関西学院大学商学研究会.
［10］ 地道正行，阪智香（2022）『探索的財務ビッグデータ解析と再現可能研究：非上場企業のデータラングリング』，商学論究，第69巻，第 3・4 合併号，pp. 83-120，関西学院大学商学研究会.
［11］ 地道正行，阪智香（2023-a）『探索的財務ビッグデータ解析と再現可能研究：mdx 環境とローカル環境の協調による非上場企業データのラングリングと可視化の自動化』，商学論究，第70巻，第 3 号，pp. 123-173，関西学院大学商学研究会.
［12］ 地道正行，阪智香（2023-b）『探索的財務ビッグデータ解析と再現可能研究：非上場企業のデータ可視化と考察』，商学論究，第71巻，第 1 号，pp. 1-122，関西学院大学商学研究会.
［13］ Moody's Analytics（2023）*Moody's DataHub User Guide ver. 2*, 2023.09.08.
［14］ Tukey, J. W.（1977）*Exploratory Data Analysis*, Addison-Wesley Publishing Co.

謝辞

本研究を行うにあたり，Moody's Analytics 社の増田歩氏，三浦泰介氏，Jiahui Lum 氏，Chungmin Yi 氏には，DataHub から大規模のデータを抽出するための方法を解説いただいたり，SQL 問合せのスクリプトを提供いただいた．ここに感謝の意を表する．また，本稿のドラフトに対して詳細なコメントを頂いた関西学院大学商学部の阪智香教授に感謝の意を表する．

本研究の一部は以下の助成を得ている．

科研費 KAKENHI 科学研究費 基盤研究 C：「企業価値報告とサステナビリティ報告の統合的フレームワークの構築」（2023年～2025年），課題番号：23K01689

JHPCN mdx 2017年度～2023年度 学際大規模情報基盤共同利用・共同研究点（JHPCN）課題：「財務ビッグデータの可視化と統計モデリング」，課題番号：jh171002-NWJ, jh181001-NWJ，jh191002-NWJ，jh201003-NWJ，jh211001-NWJ，jh221001，jh231001

統計数理研究所 統計思考院 2023年度 統計思考院公募型人材育成事業ワークショップ課題：「探索的ビッグデータ解析と再現可能研究」，課題番号：2023-思考院-7004

統計数理研究所 2023年度 統計数理研究所公募型共同利用一般研究 2 課題：「財務ビッグデータの統計モデリングと可視化に関する研究」，課題番号：2023-ISMCRP-2017

関西学院大学図書館図書費 B，研究設備費（III），個人研究費

付録 A　コンピュータ環境

本稿の執筆に際して主に利用したコンピュータ環境の情報を与える.

ハードウェア環境

- MacBook Pro 2021:
  Processor: Apple M1 Max
  Cores: 10
  Main Memory: 64 GB
  OS: macOS Ventura（12.6）
- Mac Studio 2022:
  Processor: Apple M1 Ultra
  Cores: 20
  Main Memory: 128 GB
  OS: macOS Ventura（12.6）

ソフトウェア環境

- R（R. Ihaka, R. Gentleman, R Core Team, `https://www.r-project.org/`）
- R Packages
  - arrow（Neal Richardson, `https://arrow.apache.org/docs/r/`）
  - dplyr（H. Wickham, `http://dplyr.tidyverse.org/`）
  - GGally（B. Schloerke, `http://ggobi.github.io/ggally/`）
  - ggplots2（H. Wickham, `https://ggplot2.tidyverse.org`）
  - here（Kirill Müller, `https://here.r-lib.org`）
  - magrittr（S. M. Bache, H. Wickham, and L. Henry, `https://magrittr.tidyverse.org`）

－plotly（C. Sievert, https://plotly.com/r/）

－sn（A. Azzalini, http://azzalini.stat.unipd.it/SN/）

－vroom（J. Hester, https://vroom.r-lib.org）

－xtable（D. B. Dahl, http://xtable.r-forge.r-project.org/）

- RStudio（posit, https://posit.co）
- Sweave（F. Leisch, https://stat.ethz.ch/R-manual/R-devel/library/utils/doc/Sweave.pdf）

R 関数 sessionInfo を実行することによって，本稿を執筆することに利用した R に関する環境情報を以下に与える：

sessionInfo による情報

- R version 4.3.2 (2023-10-31), aarch64-apple-darwin20
- Locale: ja_JP.UTF-8/ja_JP.UTF-8/ja_JP.UTF-8/C/ja_JP.UTF-8/ja_JP.UTF-8
- Time zone: Asia/Tokyo
- TZcode source: internal
- Running under: macOS Ventura 13.5.2
- Matrix products: default
- BLAS:
  /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
- LAPACK:
  /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib ;
  LAPACK version3.11.0
- Base packages: base, datasets, graphics, grDevices, methods, stats, stats4, utils
- Other packages: dplyr 1.1.2, forcats 1.0.0, ggplot2 3.4.4, lubridate 1.9.2, purrr 1.0.1, readr 2.1.4, sn 2.1.1, stringr 1.5.0, tibble 3.2.1, tidyr 1.3.0, tidyverse 2.0.0
- Loaded via a namespace (and not attached): cli 3.6.1, colorspace 2.1-0, compiler 4.3.2, fansi 1.0.4, farver 2.1.1, generics 0.1.3, glue 1.6.2, grid 4.3.2, gtable 0.3.3, hms 1.1.3, labeling 0.4.2, lattice 0.21-9, lifecycle 1.0.3, magrittr 2.0.3, MASS 7.3-60, Matrix 1.6-1.1, MatrixModels 0.5-1, mnormt 2.1.1, munsell 0.5.0, numDeriv 2016.8-1.1, pillar 1.9.0, pkgconfig 2.0.3, quantreg 5.95, R6 2.5.1, rlang 1.1.1, scales 1.3.0, SparseM 1.81, splines 4.3.2, stringi 1.7.12, survival 3.5-7, tidyselect 1.2.0, timechange 0.2.0, tools 4.3.2, tzdb 0.4.0, utf8 1.2.3, vctrs 0.6.3, withr 2.5.0

付録 B　DataHub からのデータ取得

ここでは，DataHub を構成する 4 つのページ（page）Exchage，Products，Spaces，Exports を利用して，データを取得するための手順を紹介する．特に，Amazon WorkSpaces[15] を利用してデータ抽出する．なお，詳細は，

15）Amazon WorkSpaces は，Amazon Web Service（AWS）上のサービスの一つであり，クラウド型の仮想デスクトップサービスのことである．（https://aws.amazon.com/

Moody's Analytics（2023）を参照されたい．大まかな手順は以下のようなものである：

（DH1）DataHub サイト https://datahub.moodys.com/ にアクセス

（DH2）Spaces ページを利用した Space（作業空間）の作成

（DH3）WorkSpaces を起動し，仮想デスクトップ上の Spaces アイコンをダブルクリック

（DH4）Space をアクティベート（Activate）

（DH5）バッチ処理（Batch Analytics）環境 Hue[16] を利用して SQL 問合せを行いデータを抽出

（DH6）Hue で抽出したデータを，DataHub サイト上の Products ページを利用してプロダクト化

（DH7）SFTP プロトコルを利用した Endpoint を作成後，Exports ページを利用し，プロダクトをエクスポート（Export）

（DH8）ローカルの SFTP クライアント[17] を利用して，Endpoint にアクセス後，データファイル（ここでは，Parquet 形式を選択）をダウンロード

以上の手順において，（DH1），（DH2），（DH6），（DH7）は DataHub のサイト上で行うが，（DH3）〜（DH5）は WorkSpaces を起動することによって利用できる仮想環境（Windows）上のサービスを利用することによって実行することに留意されたい．なお，最後の（DH8）はローカル環境から DataHub に構築された Endpoint（SFTP サイト）に接続することによって実行される．

また，本稿では，バッチ処理環境では最もプリミティブと思われる Hue

---

jp/workspaces/）ここでは，AWS と区別するために Amazon WorkSpaces を WorkSpaces と表す．

16）Hue は，データベースおよびデータウェアハウス向け SQL アシスタントである．オープンソースとして開発されている．https://jp.gethue.com//

17）macOS の「ターミナル」では sftp コマンドや，FileZilla（https://filezilla-project.org）等の専用ソフトウェアを利用すればよい．

を利用し，SQL 問合せを用いてデータを抽出したが，その他にも Spark や Python，Hadoop，RStudio[18] 等を利用して抽出することも可能である．また，Endpoint の作成において，Amazon S3，Google Cloud Storage 等のクラウドサービスを選択するオプションも用意されている．これらのサービスの利用についても，Moody's Analytics（2023）を参照されたい．

付録 C　SQL 問合せ

本稿で扱うデータを抽出するために，WorkSpaces の Hue 上で利用した SQL 問合せを以下に与える：

スクリプト13：1_tab_temp_bvdid.txt

```
CREATE TABLE IF NOT EXISTS publish_db.1_tab_temp_bvdid
AS
(SELECT
      tab_bvdid_name.bvd_id_number
FROM db_firmographics__semi_annual_6d25ce86.bvd_id_and_name tab_bvdid_name
INNER JOIN
      (SELECT
            bvd_id_number
  FROM db_firmographics__semi_annual_6d25ce86.legal_info
      WHERE
            status = "Active"
            AND
            type_of_entity = "Corporate"
      GROUP BY bvd_id_number
      ) tab_legal
ON tab_bvdid_name.bvd_id_number = tab_legal.bvd_id_number
INNER JOIN
      (SELECT
            bvd_id_number
      FROM db_financials_history__semi_annual_d8c6aaa5.industry_global_
         financials_and_ratios_usd
      WHERE
            consolidation_code in ("C1", "C2", "U1", "U2")
      GROUP BY bvd_id_number
      ) tab_ind_fin
ON tab_bvdid_name.bvd_id_number = tab_ind_fin.bvd_id_number
);
```

18）筆者は利用したことがないが，RStudio 上で R を利用したデータ抽出には，SparkR パッケージを利用する仕様となっているため，本質的には Spark を利用することになる．

スクリプト14：2_tab_temp_contact.txt

```
CREATE TABLE IF NOT EXISTS publish_db.2_tab_temp_contact
AS
(SELECT
      contact.name_internat AS name,
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
      contact.country_iso_code AS country_iso_code
FROM publish_db.1_tab_temp_bvdid 1_tab_temp_bvdid
INNER JOIN
      (SELECT
          bvd_id_number,
          name_internat,
        country_iso_code
      FROM db_firmographics__semi_annual_6d25ce86.contact_info
      WHERE
          name_internat IS NOT NULL
          AND country_iso_code IS NOT NULL) contact
ON 1_tab_temp_bvdid.bvd_id_number = contact.bvd_id_number
);
```

スクリプト15：3_tab_temp_ind.txt

```
CREATE TABLE IF NOT EXISTS publish_db.3_tab_temp_ind
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
      industry.us_sic_primary_code_s_ AS us_sic_primary_code_s_,
    industry.us_sic_primary_code_text_description AS us_sic_primary_code_
        text_description,
    industry.nace_rev_2_primary_code_s_ AS nace_rev_2_primary_code_s_,
    industry.nace_rev_2_primary_code_text_description AS nace_rev_2_primary
        _code_text_description,
    industry.naics_primary_code_s_ AS naics_primary_code_s_,
    industry.naics_primary_code_s_text_description AS naics_primary_code_s_
        text_description
FROM publish_db.1_tab_temp_bvdid 1_tab_temp_bvdid
INNER JOIN
      (SELECT
           bvd_id_number,
        us_sic_primary_code_s_,
        us_sic_primary_code_text_description,
        nace_rev_2_primary_code_s_,
        nace_rev_2_primary_code_text_description,
        naics_primary_code_s_,
        naics_primary_code_s_text_description
    FROM db_firmographics__semi_annual_6d25ce86.industry_classifications
      WHERE
           us_sic_primary_code_s_ IS NOT NULL
           OR us_sic_primary_code_text_description IS NOT NULL
```

```
            OR nace_rev_2_primary_code_s_ IS NOT NULL
            OR nace_rev_2_primary_code_text_description IS NOT NULL
            OR naics_primary_code_s_ IS NOT NULL
            OR naics_primary_code_s_text_description IS NOT NULL) industry
ON 1_tab_temp_bvdid.bvd_id_number = industry.bvd_id_number
);
```

スクリプト16：4_tab_temp_legal.txt

```
CREATE TABLE IF NOT EXISTS publish_db.4_tab_temp_legal
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
    legal.main_exchange AS main_exchange,
    legal.ipo_date AS ipo_date,
    legal.delisted_date AS delisted_date,
    legal.date_of_incorporation AS date_of_incorporation
FROM publish_db.1_tab_temp_bvdid 1_tab_temp_bvdid
INNER JOIN
     (SELECT
           bvd_id_number,
           main_exchange,
       ipo_date,
       delisted_date,
       date_of_incorporation
     FROM db_firmographics__semi_annual_6d25ce86.legal_info
     WHERE
           main_exchange IS NOT NULL
           OR ipo_date IS NOT NULL
           OR delisted_date IS NOT NULL
           OR date_of_incorporation IS NOT NULL) legal
ON 1_tab_temp_bvdid.bvd_id_number = legal.bvd_id_number
);
```

スクリプト17：5_tab_temp_id.txt

```
CREATE TABLE IF NOT EXISTS publish_db.5_tab_temp_id
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
    id.ticker_symbol AS ticker_symbol,
    id.isin_number AS isin_number
FROM publish_db.1_tab_temp_bvdid 1_tab_temp_bvdid
INNER JOIN
      (SELECT
            bvd_id_number,
        ticker_symbol,
        isin_number
      FROM db_identifiers__monthly_4de4c350.identifiers
```

```
      WHERE
            ticker_symbol IS NOT NULL
            OR isin_number IS NOT NULL) id
ON 1_tab_temp_bvdid.bvd_id_number = id.bvd_id_number
);
```

スクリプト18：6_tab_basic.txt

```
CREATE TABLE IF NOT EXISTS publish_db.6_tab_basic
AS
(SELECT
      CASE
            WHEN 2_tab_temp_contact.bvd_id_number IS NOT NULL THEN 2_tab_
               temp_contact.bvd_id_number
            WHEN 3_tab_temp_ind.bvd_id_number IS NOT NULL THEN 3_tab_temp_
               ind.bvd_id_number
            WHEN 4_tab_temp_legal.bvd_id_number IS NOT NULL THEN 4_tab_temp
               _legal.bvd_id_number
      END AS bvd_id_number,
      2_tab_temp_contact.name AS name,
      2_tab_temp_contact.country_iso_code AS country_iso_code,
      3_tab_temp_ind.us_sic_primary_code_s_ AS us_sic_primary_code_s_,
    3_tab_temp_ind.us_sic_primary_code_text_description AS us_sic_primary_
       code_text_description,
    3_tab_temp_ind.nace_rev_2_primary_code_s_ AS nace_rev_2_primary_code_s_
       ,
    3_tab_temp_ind.nace_rev_2_primary_code_text_description AS nace_rev_2_
       primary_code_text_description,
    3_tab_temp_ind.naics_primary_code_s_ AS naics_primary_code_s_,
    3_tab_temp_ind.naics_primary_code_s_text_description AS naics_primary_
       code_s_text_description,
    4_tab_temp_legal.main_exchange AS main_exchange,
    4_tab_temp_legal.ipo_date AS ipo_date,
    4_tab_temp_legal.delisted_date AS delisted_date,
    4_tab_temp_legal.date_of_incorporation AS date_of_incorporation,
    5_tab_temp_id.ticker_symbol AS ticker_symbol,
    5_tab_temp_id.isin_number AS isin_number
FROM publish_db.2_tab_temp_contact 2_tab_temp_contact
FULL OUTER JOIN
      (SELECT
            bvd_id_number,
       us_sic_primary_code_s_,
       us_sic_primary_code_text_description,
       nace_rev_2_primary_code_s_,
       nace_rev_2_primary_code_text_description,
       naics_primary_code_s_,
       naics_primary_code_s_text_description
   FROM publish_db.3_tab_temp_ind) 3_tab_temp_ind
ON 2_tab_temp_contact.bvd_id_number = 3_tab_temp_ind.bvd_id_number
FULL OUTER JOIN
      (SELECT
```

```
            bvd_id_number,
            main_exchange,
        ipo_date,
        delisted_date,
        date_of_incorporation
      FROM publish_db.4_tab_temp_legal) 4_tab_temp_legal
ON 2_tab_temp_contact.bvd_id_number = 4_tab_temp_legal.bvd_id_number
FULL OUTER JOIN
      (SELECT
            bvd_id_number,
        ticker_symbol,
        isin_number
      FROM publish_db.5_tab_temp_id) 5_tab_temp_id
ON 2_tab_temp_contact.bvd_id_number = 5_tab_temp_id.bvd_id_number
);
```

スクリプト19：7_tab_temp_ind_fin.txt

```
CREATE TABLE IF NOT EXISTS publish_db.7_tab_temp_ind_fin
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
      tab_ind_fin.original_currency AS original_currency,
      concat(
            IF( MONTH(tab_ind_fin.closing_date) + DAY(tab_ind_fin.closing_
                date) <> 2 AND MONTH(tab_ind_fin.closing_date) BETWEEN 1 and
                 3,
                  CAST(YEAR(tab_ind_fin.closing_date)-1 AS string),
                  CAST(YEAR(tab_ind_fin.closing_date) AS string) ),
            "␣(USD)") AS fiscal_year,
    tab_ind_fin.current_assets AS current_assets,
    tab_ind_fin.stock AS stock,
    tab_ind_fin.debtors AS debtors,
    tab_ind_fin.other_current_assets AS other_current_assets,
    tab_ind_fin.cash_cash_equivalent AS cash_cash_equivalent,
    tab_ind_fin.fixed_assets AS fixed_assets,
    tab_ind_fin.tangible_fixed_assets AS tangible_fixed_assets,
    tab_ind_fin.intangible_fixed_assets AS intangible_fixed_assets,
    tab_ind_fin.other_fixed_assets AS other_fixed_assets,
    tab_ind_fin.total_assets AS total_assets,
    tab_ind_fin.current_liabilities AS current_liabilities,
    tab_ind_fin.loans AS loans,
    tab_ind_fin.creditors AS creditors,
    tab_ind_fin.other_current_liabilities AS other_current_liabilities,
    tab_ind_fin.non_current_liabilities AS non_current_liabilities,
    tab_ind_fin.long_term_debt AS long_term_debt,
    tab_ind_fin.other_non_current_liabilities AS other_non_current_
        liabilities,
    tab_ind_fin.provisions AS provisions,
    tab_ind_fin.shareholders_funds AS shareholders_funds,
```

```
30      tab_ind_fin.capital AS capital,
31      tab_ind_fin.other_shareholders_funds AS other_shareholders_funds,
32      tab_ind_fin.total_shareh_funds_liab_ AS total_shareh_funds_liab_,
33      tab_ind_fin.enterprise_value AS enterprise_value,
34      tab_ind_fin.working_capital AS working_capital,
35      tab_ind_fin.net_current_assets AS net_current_assets,
36      tab_ind_fin.number_of_employees AS number_of_employees,
37      tab_ind_fin.operating_revenue_turnover_ AS operating_revenue_turnover_,
38      tab_ind_fin.sales AS sales,
39      tab_ind_fin.costs_of_goods_sold AS costs_of_goods_sold,
40        tab_ind_fin.gross_profit AS gross_profit,
41      tab_ind_fin.other_operating_expenses AS other_operating_expenses,
42      tab_ind_fin.depreciation_amortization AS depreciation_amortization,
43      tab_ind_fin.operating_p_l_ebit_ AS operating_p_l_ebit_,
44      tab_ind_fin.financial_revenue AS financial_revenue,
45      tab_ind_fin.financial_expenses AS financial_expenses,
46      tab_ind_fin.financial_p_l AS financial_p_l,
47      tab_ind_fin.p_l_before_tax AS p_l_before_tax,
48      tab_ind_fin.taxation AS taxation,
49      tab_ind_fin.p_l_after_tax AS p_l_after_tax,
50      tab_ind_fin.p_l_for_period_net_income_ AS p_l_for_period_net_income_,
51      tab_ind_fin.material_costs AS material_costs,
52      tab_ind_fin.costs_of_employees costs_of_employees,
53      tab_ind_fin.research_development_expenses AS research_development_
          expenses,
54      tab_ind_fin.cash_flow cash_flow,
55      tab_ind_fin.added_value AS added_value,
56      tab_ind_fin.ebitda AS ebitda,
57      tab_ind_fin.consolidation_code AS consolidation_code,
58        tab_ind_fin.filing_type AS filing_type,
59      tab_ind_fin.closing_date AS closing_date,
60      tab_ind_fin.number_of_months AS number_of_months,
61      tab_ind_fin.audit_status AS audit_status,
62      tab_ind_fin.accounting_practice AS accounting_practice,
63      tab_ind_fin.source_for_publicly_quoted_companies_ AS source_for_
          publicly_quoted_companies_,
64      tab_ind_fin.original_units AS original_units,
65      tab_ind_fin.exchange_rate_from_original_currency AS exchange_rate_from_
          original_currency
66  FROM publish_db.1_tab_temp_bvdid 1_tab_temp_bvdid
67  INNER JOIN
68        (SELECT
69             bvd_id_number,
70          original_currency,
71          current_assets,
72          stock,
73          debtors,
74          other_current_assets,
75          cash_cash_equivalent,
76          fixed_assets,
77          tangible_fixed_assets,
78          intangible_fixed_assets,
```

```
        other_fixed_assets,
        total_assets,
        current_liabilities,
        loans,
        creditors,
        other_current_liabilities,
        non_current_liabilities,
        long_term_debt,
        other_non_current_liabilities,
        provisions,
        shareholders_funds,
        capital,
        other_shareholders_funds,
        total_shareh_funds_liab_,
        enterprise_value,
        working_capital,
        net_current_assets,
        number_of_employees,
        operating_revenue_turnover_,
        sales,
        costs_of_goods_sold,
        gross_profit,
        other_operating_expenses,
        depreciation_amortization,
        operating_p_l_ebit_,
        financial_revenue,
        financial_expenses,
        financial_p_l,
        p_l_before_tax,
        taxation,
        p_l_after_tax,
        p_l_for_period_net_income_,
        material_costs,
        costs_of_employees,
        research_development_expenses,
        cash_flow,
        added_value,
        ebitda,
        consolidation_code,
            filing_type,
        closing_date,
        number_of_months,
        audit_status,
        accounting_practice,
        source_for_publicly_quoted_companies_,
        original_units,
        exchange_rate_from_original_currency
    FROM db_financials_history__semi_annual_d8c6aaa5.industry_global_
        financials_and_ratios_usd
    WHERE
          (consolidation_code IS NOT NULL
          AND filing_type IS NOT NULL
```

```
130         AND closing_date IS NOT NULL)
131             AND
132             (original_currency IS NOT NULL
133         OR current_assets IS NOT NULL
134         OR stock IS NOT NULL
135         OR debtors IS NOT NULL
136         OR other_current_assets IS NOT NULL
137         OR cash_cash_equivalent IS NOT NULL
138         OR fixed_assets IS NOT NULL
139         OR tangible_fixed_assets IS NOT NULL
140         OR intangible_fixed_assets IS NOT NULL
141         OR other_fixed_assets IS NOT NULL
142         OR total_assets IS NOT NULL
143         OR current_liabilities IS NOT NULL
144         OR loans IS NOT NULL
145         OR creditors IS NOT NULL
146         OR other_current_liabilities IS NOT NULL
147         OR non_current_liabilities IS NOT NULL
148         OR long_term_debt IS NOT NULL
149         OR other_non_current_liabilities IS NOT NULL
150         OR provisions IS NOT NULL
151         OR shareholders_funds IS NOT NULL
152         OR capital IS NOT NULL
153         OR other_shareholders_funds IS NOT NULL
154         OR total_shareh_funds_liab_ IS NOT NULL
155         OR enterprise_value IS NOT NULL
156         OR working_capital IS NOT NULL
157         OR net_current_assets IS NOT NULL
158         OR number_of_employees IS NOT NULL
159         OR operating_revenue_turnover_ IS NOT NULL
160         OR sales IS NOT NULL
161         OR costs_of_goods_sold IS NOT NULL
162         OR gross_profit IS NOT NULL
163         OR other_operating_expenses IS NOT NULL
164         OR depreciation_amortization IS NOT NULL
165         OR operating_p_l_ebit_ IS NOT NULL
166         OR financial_revenue IS NOT NULL
167         OR financial_expenses IS NOT NULL
168         OR financial_p_l IS NOT NULL
169         OR p_l_before_tax IS NOT NULL
170         OR taxation IS NOT NULL
171         OR p_l_after_tax IS NOT NULL
172         OR p_l_for_period_net_income_ IS NOT NULL
173         OR material_costs IS NOT NULL
174         OR costs_of_employees IS NOT NULL
175         OR research_development_expenses IS NOT NULL
176         OR cash_flow IS NOT NULL
177         OR added_value IS NOT NULL
178         OR ebitda IS NOT NULL
179         OR consolidation_code IS NOT NULL
180         OR filing_type IS NOT NULL
181         OR closing_date IS NOT NULL
```

```
        OR number_of_months IS NOT NULL
        OR audit_status IS NOT NULL
        OR accounting_practice IS NOT NULL
        OR source_for_publicly_quoted_companies_ IS NOT NULL
        OR original_units IS NOT NULL
        OR exchange_rate_from_original_currency IS NOT NULL)) tab_ind_fin
ON 1_tab_temp_bvdid.bvd_id_number = tab_ind_fin.bvd_id_number
);
```

スクリプト20：8_tab_temp_det_ind.txt

```
CREATE TABLE IF NOT EXISTS publish_db.8_tab_temp_det_ind
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
      concat(
            IF( MONTH(tab_det_ind.closing_date) + DAY(tab_det_ind.closing_
               date) <> 2 AND MONTH(tab_det_ind.closing_date) BETWEEN 1 and
                3,
                  CAST(YEAR(tab_det_ind.closing_date)-1 AS string),
                  CAST(YEAR(tab_det_ind.closing_date) AS string) ),
            "_(USD)") AS fiscal_year,
    tab_det_ind.closing_date AS closing_date,
    tab_det_ind.consolidation_code AS consolidation_code,
      tab_det_ind.income_taxes AS income_taxes,
      tab_det_ind.income_tax_payable AS income_tax_payable,
      tab_det_ind.deferred_taxes AS deferred_taxes,
      tab_det_ind.dividends_payable AS dividends_payable
FROM publish_db.1_tab_temp_bvdid1_tab_temp_bvdid
INNER JOIN
      (SELECT
            bvd_id_number,
            closing_date,
            consolidation_code,
        income_taxes,
        income_tax_payable,
        deferred_taxes,
        dividends_payable
      FROM db_listed_financials__monthly_6f8c084e.detailed_format_
         industries_usd
      WHERE
            (closing_date IS NOT NULL
            AND consolidation_code IS NOT NULL)
            AND
            (income_taxes IS NOT NULL
            OR income_tax_payable IS NOT NULL
            OR deferred_taxes IS NOT NULL
            OR dividends_payable IS NOT NULL)) tab_det_ind
ON 1_tab_temp_bvdid.bvd_id_number = tab_det_ind.bvd_id_number
);
```

スクリプト21：9_tab_temp_cash_nonus_ind.txt

```
CREATE TABLE IF NOT EXISTS publish_db.9_tab_temp_cash_nonus_ind
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
      concat(
            IF( MONTH(tab_cash_nonus_ind.closing_date) + DAY(tab_cash_nonus_ind.closing_date) <> 2 AND MONTH(tab_cash_nonus_ind.closing_date) BETWEEN 1 and 3,
               CAST(YEAR(tab_cash_nonus_ind.closing_date)-1 AS string),
               CAST(YEAR(tab_cash_nonus_ind.closing_date) AS string) ),
            "_(USD)") AS fiscal_year,
    tab_cash_nonus_ind.closing_date AS closing_date,
    tab_cash_nonus_ind.consolidation_code AS consolidation_code,
      tab_cash_nonus_ind.def_inc_taxes_invest_tax_credit AS def_inc_taxes_invest_tax_credit,
      tab_cash_nonus_ind.net_cash_from_operating_activities AS net_cash_from_operating_activities,
      tab_cash_nonus_ind.net_cash_used_by_investing_activities AS net_cash_used_by_investing_activities,
      tab_cash_nonus_ind.net_cash_provided_by_used_in_financing_activities AS net_cash_provided_by_used_in_financing_activities,
      tab_cash_nonus_ind.cash_equivalents_at_end_of_year AS cash_equivalents_at_end_of_year
FROM publish_db.1_tab_temp_bvdid1_tab_temp_bvdid
INNER JOIN
      (SELECT
            bvd_id_number,
            closing_date,
            consolidation_code,
            def_inc_taxes_invest_tax_credit,
            net_cash_from_operating_activities,
            net_cash_used_by_investing_activities,
            net_cash_provided_by_used_in_financing_activities,
            cash_equivalents_at_end_of_year
      FROM db_listed_financials__monthly_6f8c084e.cash_flow_non_us_industries_usd
      WHERE
            (closing_date IS NOT NULL
            AND consolidation_code IS NOT NULL)
            AND
            (def_inc_taxes_invest_tax_credit IS NOT NULL
            OR net_cash_from_operating_activities IS NOT NULL
            OR net_cash_used_by_investing_activities IS NOT NULL
            OR net_cash_provided_by_used_in_financing_activities IS NOT NULL
            OR cash_equivalents_at_end_of_year IS NOT NULL)) tab_cash_nonus_ind
ON 1_tab_temp_bvdid.bvd_id_number = tab_cash_nonus_ind.bvd_id_number
);
```

スクリプト22：10_tab_temp_cash_us_ind.txt

```
CREATE TABLE IF NOT EXISTS publish_db.10_tab_temp_cash_us_ind
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
      concat(
            IF( MONTH(tab_cash_us_ind.closing_date) + DAY(tab_cash_us_ind.
               closing_date) <> 2 AND MONTH(tab_cash_us_ind.closing_date)
               BETWEEN 1 and 3,
                  CAST(YEAR(tab_cash_us_ind.closing_date)-1 AS string),
                  CAST(YEAR(tab_cash_us_ind.closing_date) AS string) ),
            "␣(USD)") AS fiscal_year,
    tab_cash_us_ind.closing_date AS closing_date,
    tab_cash_us_ind.consolidation_code AS consolidation_code,
      tab_cash_us_ind.total_cash_from_operating_activities AS total_cash_
         from_operating_activities,
      tab_cash_us_ind.total_cash_from_investing_activities AS total_cash_
         from_investing_activities,
      tab_cash_us_ind.total_cash_from_financing_activities AS total_cash_
         from_financing_activities,
      tab_cash_us_ind.net_cash_ending_balance AS net_cash_ending_balance
FROM publish_db.1_tab_temp_bvdid 1_tab_temp_bvdid
INNER JOIN
      (SELECT
            bvd_id_number,
            closing_date,
            consolidation_code,
            total_cash_from_operating_activities,
            total_cash_from_investing_activities,
            total_cash_from_financing_activities,
            net_cash_ending_balance
      FROM db_listed_financials__monthly_6f8c084e.cash_flow_us_industries_
         usd
      WHERE
            (closing_date IS NOT NULL
            AND consolidation_code IS NOT NULL)
            AND
            (total_cash_from_operating_activities IS NOT NULL
            OR total_cash_from_investing_activities IS NOT NULL
            OR total_cash_from_financing_activities IS NOT NULL
            OR net_cash_ending_balance IS NOT NULL)) tab_cash_us_ind
ON 1_tab_temp_bvdid.bvd_id_number = tab_cash_us_ind.bvd_id_number
);
```

スクリプト23：11_tab_financials.txt

```
CREATE TABLE IF NOT EXISTS publish_db.11_tab_financials
AS
(SELECT
      CASE
            WHEN 7_tab_temp_ind_fin.bvd_id_number IS NOT NULL THEN 7_tab_
               temp_ind_fin.bvd_id_number
            WHEN 8_tab_temp_det_ind.bvd_id_number IS NOT NULL THEN 8_tab_
               temp_det_ind.bvd_id_number
            WHEN 9_tab_temp_cash_nonus_ind.bvd_id_number IS NOT NULL THEN 9
               _tab_temp_cash_nonus_ind.bvd_id_number
             WHEN 10_tab_temp_cash_us_ind.bvd_id_number IS NOT NULL THEN 10_
               tab_temp_cash_us_ind.bvd_id_number
      END AS bvd_id_number,
      7_tab_temp_ind_fin.original_currency AS original_currency,
      CASE
             WHEN 7_tab_temp_ind_fin.fiscal_year IS NOT NULL THEN 7_tab_temp
               _ind_fin.fiscal_year
             WHEN 8_tab_temp_det_ind.fiscal_year IS NOT NULL THEN 8_tab_temp
               _det_ind.fiscal_year
            WHEN 9_tab_temp_cash_nonus_ind.fiscal_year IS NOT NULL THEN 9_
               tab_temp_cash_nonus_ind.fiscal_year
            WHEN 10_tab_temp_cash_us_ind.fiscal_year IS NOT NULL THEN 10_
               tab_temp_cash_us_ind.fiscal_year
      END AS fiscal_year,
     7_tab_temp_ind_fin.current_assets AS current_assets,
     7_tab_temp_ind_fin.stock AS stock,
     7_tab_temp_ind_fin.debtors AS debtors,
     7_tab_temp_ind_fin.other_current_assets AS other_current_assets,
     7_tab_temp_ind_fin.cash_cash_equivalent AS cash_cash_equivalent,
     7_tab_temp_ind_fin.fixed_assets AS fixed_assets,
     7_tab_temp_ind_fin.tangible_fixed_assets AS tangible_fixed_assets,
     7_tab_temp_ind_fin.intangible_fixed_assets AS intangible_fixed_assets,
     7_tab_temp_ind_fin.other_fixed_assets AS other_fixed_assets,
     7_tab_temp_ind_fin.total_assets AS total_assets,
     7_tab_temp_ind_fin.current_liabilities AS current_liabilities,
     7_tab_temp_ind_fin.loans AS loans,
     7_tab_temp_ind_fin.creditors AS creditors,
     7_tab_temp_ind_fin.other_current_liabilities AS other_current_
        liabilities,
     7_tab_temp_ind_fin.non_current_liabilities AS non_current_liabilities,
     7_tab_temp_ind_fin.long_term_debt AS long_term_debt,
     7_tab_temp_ind_fin.other_non_current_liabilities AS other_non_current_
        liabilities,
     7_tab_temp_ind_fin.provisions AS provisions,
     7_tab_temp_ind_fin.shareholders_funds AS shareholders_funds,
     7_tab_temp_ind_fin.capital AS capital,
     7_tab_temp_ind_fin.other_shareholders_funds AS other_shareholders_funds
        ,
     7_tab_temp_ind_fin.total_shareh_funds_liab_ AS total_shareh_funds_liab_
        ,
```

```
39      7_tab_temp_ind_fin.enterprise_value AS enterprise_value,
40      7_tab_temp_ind_fin.working_capital AS working_capital,
41      7_tab_temp_ind_fin.net_current_assets AS net_current_assets,
42      7_tab_temp_ind_fin.number_of_employees AS number_of_employees,
43      7_tab_temp_ind_fin.operating_revenue_turnover_ AS operating_revenue_
           turnover_,
44      7_tab_temp_ind_fin.sales AS sales,
45      7_tab_temp_ind_fin.costs_of_goods_sold AS costs_of_goods_sold,
46        7_tab_temp_ind_fin.gross_profit AS gross_profit,
47      7_tab_temp_ind_fin.other_operating_expenses AS other_operating_expenses
           ,
48      7_tab_temp_ind_fin.depreciation_amortization AS depreciation_
           amortization,
49      7_tab_temp_ind_fin.operating_p_l_ebit_ AS operating_p_l_ebit_,
50      7_tab_temp_ind_fin.financial_revenue AS financial_revenue,
51      7_tab_temp_ind_fin.financial_expenses AS financial_expenses,
52      7_tab_temp_ind_fin.financial_p_l AS financial_p_l,
53      7_tab_temp_ind_fin.p_l_before_tax AS p_l_before_tax,
54      7_tab_temp_ind_fin.taxation AS taxation,
55      7_tab_temp_ind_fin.p_l_after_tax AS p_l_after_tax,
56      7_tab_temp_ind_fin.p_l_for_period_net_income_ AS p_l_for_period_net_
           income_,
57      7_tab_temp_ind_fin.material_costs AS material_costs,
58      7_tab_temp_ind_fin.costs_of_employees costs_of_employees,
59      7_tab_temp_ind_fin.research_development_expenses AS research_
           development_expenses,
60      7_tab_temp_ind_fin.cash_flow cash_flow,
61      7_tab_temp_ind_fin.added_value AS added_value,
62      7_tab_temp_ind_fin.ebitda AS ebitda,
63      CASE
64              WHEN 7_tab_temp_ind_fin.consolidation_code IS NOT NULL THEN 7_
                   tab_temp_ind_fin.consolidation_code
65              WHEN 8_tab_temp_det_ind.consolidation_code IS NOT NULL THEN 8_
                   tab_temp_det_ind.consolidation_code
66              WHEN 9_tab_temp_cash_nonus_ind.consolidation_code IS NOT NULL
                   THEN 9_tab_temp_cash_nonus_ind.consolidation_code
67              WHEN 10_tab_temp_cash_us_ind.consolidation_code IS NOT NULL
                   THEN 10_tab_temp_cash_us_ind.consolidation_code
68        END AS consolidation_code,
69        7_tab_temp_ind_fin.filing_type AS filing_type,
70        CASE
71              WHEN 7_tab_temp_ind_fin.closing_date IS NOT NULL THEN 7_tab_
                   temp_ind_fin.closing_date
72              WHEN 8_tab_temp_det_ind.closing_date IS NOT NULL THEN 8_tab_
                   temp_det_ind.closing_date
73              WHEN 9_tab_temp_cash_nonus_ind.closing_date IS NOT NULL THEN 9_
                   tab_temp_cash_nonus_ind.closing_date
74              WHEN 10_tab_temp_cash_us_ind.closing_date IS NOT NULL THEN 10_
                   tab_temp_cash_us_ind.closing_date
75          END AS closing_date,
76      7_tab_temp_ind_fin.number_of_months AS number_of_months,
77      7_tab_temp_ind_fin.audit_status AS audit_status,
```

```
78      7_tab_temp_ind_fin.accounting_practice AS accounting_practice,
79      7_tab_temp_ind_fin.source_for_publicly_quoted_companies_ AS source_for_
           publicly_quoted_companies_,
80      7_tab_temp_ind_fin.original_units AS original_units,
81      7_tab_temp_ind_fin.exchange_rate_from_original_currency AS exchange_
           rate_from_original_currency,
82        8_tab_temp_det_ind.income_taxes AS income_taxes,
83        8_tab_temp_det_ind.income_tax_payable AS income_tax_payable,
84        8_tab_temp_det_ind.deferred_taxes AS deferred_taxes,
85        9_tab_temp_cash_nonus_ind.def_inc_taxes_invest_tax_credit AS def_inc_
             taxes_invest_tax_credit,
86        8_tab_temp_det_ind.dividends_payable AS dividends_payable,
87        9_tab_temp_cash_nonus_ind.net_cash_from_operating_activities AS net_
             cash_from_operating_activities,
88        9_tab_temp_cash_nonus_ind.net_cash_used_by_investing_activities AS
             net_cash_used_by_investing_activities,
89        9_tab_temp_cash_nonus_ind.net_cash_provided_by_used_in_financing_
             activities AS net_cash_provided_by_used_in_financing_activities,
90        9_tab_temp_cash_nonus_ind.cash_equivalents_at_end_of_year AS cash_
             equivalents_at_end_of_year,
91       10_tab_temp_cash_us_ind.total_cash_from_operating_activities AS total
             _cash_from_operating_activities,
92       10_tab_temp_cash_us_ind.total_cash_from_investing_activities AS total
             _cash_from_investing_activities,
93       10_tab_temp_cash_us_ind.total_cash_from_financing_activities AS total
             _cash_from_financing_activities,
94       10_tab_temp_cash_us_ind.net_cash_ending_balance AS net_cash_ending_
             balance
95  FROM publish_db.7_tab_temp_ind_fin 7_tab_temp_ind_fin
96  FULL OUTER JOIN
97        (SELECT
98              bvd_id_number,
99              fiscal_year,
100             closing_date,
101             consolidation_code,
102          income_taxes,
103          income_tax_payable,
104          deferred_taxes,
105          dividends_payable
106       FROM publish_db.8_tab_temp_det_ind)8_tab_temp_det_ind
107 ON 7_tab_temp_ind_fin.bvd_id_number = 8_tab_temp_det_ind.bvd_id_number
108       AND 7_tab_temp_ind_fin.closing_date = 8_tab_temp_det_ind.closing_date
109       AND   7_tab_temp_ind_fin.consolidation_code = 8_tab_temp_det_ind.
             consolidation_code
110 FULL OUTER JOIN
111       (SELECT
112             bvd_id_number,
113             fiscal_year,
114             closing_date,
115             consolidation_code,
116             def_inc_taxes_invest_tax_credit,
117             net_cash_from_operating_activities,
```

```
            net_cash_used_by_investing_activities,
            net_cash_provided_by_used_in_financing_activities,
            cash_equivalents_at_end_of_year
      FROM publish_db.9_tab_temp_cash_nonus_ind) 9_tab_temp_cash_nonus_ind
ON 7_tab_temp_ind_fin.bvd_id_number = 9_tab_temp_cash_nonus_ind.bvd_id_
   number
      AND   7_tab_temp_ind_fin.closing_date = 9_tab_temp_cash_nonus_ind.
         closing_date
      AND 7_tab_temp_ind_fin.consolidation_code = 9_tab_temp_cash_nonus_ind
         .consolidation_code
FULL OUTER JOIN
      (SELECT
            bvd_id_number,
            fiscal_year,
            closing_date,
            consolidation_code,
            total_cash_from_operating_activities,
            total_cash_from_investing_activities,
            total_cash_from_financing_activities,
            net_cash_ending_balance
      FROM publish_db.10_tab_temp_cash_us_ind) 10_tab_temp_cash_us_ind
ON 7_tab_temp_ind_fin.bvd_id_number = 10_tab_temp_cash_us_ind.bvd_id_number
      AND   7_tab_temp_ind_fin.closing_date = 10_tab_temp_cash_us_ind.
         closing_date
      AND 7_tab_temp_ind_fin.consolidation_code = 10_tab_temp_cash_us_ind.
         consolidation_code
);
```

スクリプト24：12_tab_temp_sec_stock.txt

```
CREATE TABLE IF NOT EXISTS publish_db.12_tab_temp_sec_stock
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
      tab_sec_stock.date_of_current_market_capitalisation AS date_of_
         current_market_capitalisation
FROM publish_db.1_tab_temp_bvdid1_tab_temp_bvdid
INNER JOIN
      (SELECT
            bvd_id_number,
            date_of_current_market_capitalisation
     FROM db_stock_data__monthly_7d23c2d6.security_stock_information_and_
        current_stock_information
     WHERE
            date_of_current_market_capitalisation IS NOT NULL) tab_sec_
               stock
ON 1_tab_temp_bvdid.bvd_id_number = tab_sec_stock.bvd_id_number
GROUP BY 1_tab_temp_bvdid.bvd_id_number, tab_sec_stock.date_of_current_
   market_capitalisation
);
```

スクリプト25：13_tab_temp_ann_stock.txt

```
CREATE TABLE IF NOT EXISTS publish_db.13_tab_temp_ann_stock
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
      concat(
            IF( MONTH(tab_ann_stock.closing_date) + DAY(tab_ann_stock.
               closing_date) <> 2 AND MONTH(tab_ann_stock.closing_date)
               BETWEEN 1 and 3,
                  CAST(YEAR(tab_ann_stock.closing_date)-1 AS string),
                  CAST(YEAR(tab_ann_stock.closing_date) AS string) ),
            "␣(USD)") AS fiscal_year,
      tab_ann_stock.closing_date AS closing_date,
      tab_ann_stock.market_price_year_end AS market_price_year_end,
      tab_ann_stock.market_capitalisation_mil_ AS market_capitalisation_mil
         _
FROM publish_db.1_tab_temp_bvdid 1_tab_temp_bvdid
INNER JOIN
      (SELECT
            bvd_id_number,
            closing_date,
            market_price_year_end,
            market_capitalisation_mil_
      FROM db_stock_data__monthly_7d23c2d6.annual_stock_valuation_annual_
         stock_data
      WHERE
            closing_date IS NOT NULL
            AND
            (market_price_year_end IS NOT NULL
            OR market_capitalisation_mil_ IS NOT NULL)) tab_ann_stock
ON 1_tab_temp_bvdid.bvd_id_number = tab_ann_stock.bvd_id_number
);
```

スクリプト26：14_tab_temp_mon_info.txt

```
CREATE TABLE IF NOT EXISTS publish_db.14_tab_temp_mon_info
AS
(SELECT
      1_tab_temp_bvdid.bvd_id_number AS bvd_id_number,
      concat(
             IF (MONTH(tab_mon_info.market_price_year) + DAY(tab_mon_info.
                market_price_year) = 2,
                   CAST(YEAR(tab_mon_info.market_price_year) AS string),
                   NULL),
             "␣(USD)") AS fiscal_year,
      IF (MONTH(tab_mon_info.market_price_year) + DAY(tab_mon_info.market_
         price_year) = 2,
                   CAST(YEAR(tab_mon_info.market_price_year) AS string),
                   NULL) AS market_price_year,
```

```
13        tab_mon_info.monthly_closing_price_january AS monthly_closing_price_
              january,
14        tab_mon_info.monthly_closing_price_february AS monthly_closing_price_
              february,
15        tab_mon_info.monthly_closing_price_march AS monthly_closing_price_
              march,
16        tab_mon_info.monthly_closing_price_april AS monthly_closing_price_
              april,
17        tab_mon_info.monthly_closing_price_may AS monthly_closing_price_may,
18        tab_mon_info.monthly_closing_price_june AS monthly_closing_price_june
              ,
19        tab_mon_info.monthly_closing_price_july AS monthly_closing_price_july
              ,
20        tab_mon_info.monthly_closing_price_august AS monthly_closing_price_
              august,
21        tab_mon_info.monthly_closing_price_september AS monthly_closing_price
              _september,
22        tab_mon_info.monthly_closing_price_october AS monthly_closing_price_
              october,
23        tab_mon_info.monthly_closing_price_november AS monthly_closing_price_
              november,
24        tab_mon_info.monthly_closing_price_december AS monthly_closing_price_
              december
25  FROM publish_db.1_tab_temp_bvdid 1_tab_temp_bvdid
26  INNER JOIN
27        (SELECT
28              bvd_id_number,
29              market_price_year,
30              monthly_closing_price_january,
31              monthly_closing_price_february,
32              monthly_closing_price_march,
33              monthly_closing_price_april,
34              monthly_closing_price_may,
35              monthly_closing_price_june,
36              monthly_closing_price_july,
37              monthly_closing_price_august,
38              monthly_closing_price_september,
39              monthly_closing_price_october,
40              monthly_closing_price_november,
41              monthly_closing_price_december
42        FROM db_stock_data__monthly_7d23c2d6.monthly_information
43        WHERE
44              market_price_year IS NOT NULL
45              AND
46              (monthly_closing_price_january IS NOT NULL
47              OR monthly_closing_price_february IS NOT NULL
48              OR monthly_closing_price_march IS NOT NULL
49              OR monthly_closing_price_april IS NOT NULL
50              OR monthly_closing_price_may IS NOT NULL
51              OR monthly_closing_price_june IS NOT NULL
52              OR monthly_closing_price_july IS NOT NULL
53              OR monthly_closing_price_august IS NOT NULL
```

```
            OR monthly_closing_price_september IS NOT NULL
            OR monthly_closing_price_october IS NOT NULL
            OR monthly_closing_price_november IS NOT NULL
            OR monthly_closing_price_december IS NOT NULL)) tab_mon_info
ON 1_tab_temp_bvdid.bvd_id_number = tab_mon_info.bvd_id_number
);
```

スクリプト27：15_tab_stock.txt

```
CREATE TABLE IF NOT EXISTS publish_db.15_tab_stock
AS
(SELECT
      CASE
            WHEN 12_tab_temp_sec_stock.bvd_id_number IS NOT NULL THEN 12_
               tab_temp_sec_stock.bvd_id_number
            WHEN 13_tab_temp_ann_stock.bvd_id_number IS NOT NULL THEN 13_
               tab_temp_ann_stock.bvd_id_number
            WHEN 14_tab_temp_mon_info.bvd_id_number IS NOT NULL THEN 14_tab
               _temp_mon_info.bvd_id_number
      END AS bvd_id_number,
      CASE
            WHEN 13_tab_temp_ann_stock.fiscal_year IS NOT NULL THEN 13_tab_
               temp_ann_stock.fiscal_year
            WHEN 14_tab_temp_mon_info.fiscal_year IS NOT NULL THEN 14_tab_
               temp_mon_info.fiscal_year
      END AS fiscal_year,
      13_tab_temp_ann_stock.closing_date AS closing_date,
      12_tab_temp_sec_stock.date_of_current_market_capitalisation AS date_
         of_current_market_capitalisation,
      13_tab_temp_ann_stock.market_price_year_end AS market_price_year_end,
      13_tab_temp_ann_stock.market_capitalisation_mil_ AS market_
         capitalisation_mil_,
      14_tab_temp_mon_info.market_price_year AS market_price_year,
      14_tab_temp_mon_info.monthly_closing_price_january AS monthly_closing
         _price_january,
      14_tab_temp_mon_info.monthly_closing_price_february AS monthly_
         closing_price_february,
      14_tab_temp_mon_info.monthly_closing_price_march AS monthly_closing_
         price_march,
      14_tab_temp_mon_info.monthly_closing_price_april AS monthly_closing_
         price_april,
      14_tab_temp_mon_info.monthly_closing_price_may AS monthly_closing_
         price_may,
      14_tab_temp_mon_info.monthly_closing_price_june AS monthly_closing_
         price_june,
      14_tab_temp_mon_info.monthly_closing_price_july AS monthly_closing_
         price_july,
      14_tab_temp_mon_info.monthly_closing_price_august AS monthly_closing_
         price_august,
```

```
      14_tab_temp_mon_info.monthly_closing_price_september AS monthly_
         closing_price_september,
      14_tab_temp_mon_info.monthly_closing_price_october AS monthly_closing
         _price_october,
      14_tab_temp_mon_info.monthly_closing_price_november AS monthly_
         closing_price_november,
      14_tab_temp_mon_info.monthly_closing_price_december AS monthly_
         closing_price_december
FROM publish_db.12_tab_temp_sec_stock 12_tab_temp_sec_stock
FULL OUTER JOIN
      (SELECT
            bvd_id_number,
            fiscal_year,
            closing_date,
            market_price_year_end,
            market_capitalisation_mil_
      FROM publish_db.13_tab_temp_ann_stock) 13_tab_temp_ann_stock
ON 12_tab_temp_sec_stock.bvd_id_number = 13_tab_temp_ann_stock.bvd_id_
   number
FULL OUTER JOIN
      (SELECT
            bvd_id_number,
            fiscal_year,
            market_price_year,
            monthly_closing_price_january,
            monthly_closing_price_february,
            monthly_closing_price_march,
            monthly_closing_price_april,
            monthly_closing_price_may,
            monthly_closing_price_june,
            monthly_closing_price_july,
            monthly_closing_price_august,
            monthly_closing_price_september,
            monthly_closing_price_october,
            monthly_closing_price_november,
            monthly_closing_price_december
      FROM publish_db.14_tab_temp_mon_info) 14_tab_temp_mon_info
ON
12_tab_temp_sec_stock.bvd_id_number = 14_tab_temp_mon_info.bvd_id_number
AND 13_tab_temp_ann_stock.fiscal_year = 14_tab_temp_mon_info.fiscal_year
);
```