SQL

Naoko Ishibashi

2025-02-02

```
library(sqldf)
```

```
## Loading required package: gsubfn
```

```
## Loading required package: proto
```

```
## Loading required package: RSQLite
```

```
library(ggplot2)

# 職業別雇用統計データを使用します
# ファイルはかなり大きいです（N > 430,000）

# 詳細は以下をご参照ください
# http://www.bls.gov/oes/current/oes_stru.htm

# データセットの最初の数行を確認します
setwd("~/Dropbox/DATA4010/week7/Homework")
read.table("oesm.csv", sep=";", header = TRUE, fill = TRUE, nrows = 5)
```

area <int>	area_title <chr>	area_type <int>	nai… <int>	naics_title <chr>
99	U.S.	1	0	Cross-industry
99	U.S.	1	0	Cross-industry
3100003	Northeastern Nebraska nonmetropolitan area	6	0	Cross-industry
42540	Scranton--Wilkes-Barre, PA	4	0	Cross-industry
25620	Hattiesburg, MS	4	0	Cross-industry

5 rows | 1-6 of 30 columns

```
# SQL処理を開始します。まず新しいデータベースを作成し、
# 次にテーブルを追加します
```

```
# 1. 'conemp' という名前の新しいデータベースを作成してSQL処理を開始します
# その後、'oesm' というテーブルを追加します
# （ベストプラクティスとして、すでに 'oesm' テーブルが存在する場合は削除するために
# if 文を使用します）

{
  # SQLiteデータベースに接続し、'conemp'というデータベースを作成
  conemp <- dbConnect(SQLite(), dbname = "oesm.dv")

  # 'oesm' テーブルがすでに存在する場合は削除
  if (dbExistsTable(conemp, "oesm"))
    dbRemoveTable(conemp, "oesm")

  # データを 'oesm' テーブルに書き込み
  dbWriteTable(
    conemp,
    "oesm",
    "oesm.csv",
    sep = ";",
    header = TRUE,
    row.names = FALSE
  )

  # データベース接続を切断
  dbDisconnect(conemp)
}
```

```
# 2. テーブル内のすべてのデータから最初の10行を表示します

conemp1 <- dbConnect(SQLite(), dbname = "oesm.dv")
res <- dbSendQuery(conemp1, "
                   SELECT *
                   FROM oesm")
fetch(res, n = 10)  # 最初の10行を取得
```

area <int>	area_title <chr>	area_type <int>	nai… <int>	naic <chr
99	U.S.	1	0	Cros
99	U.S.	1	0	Cros
3100003	Northeastern Nebraska nonmetropolitan area	6	0	Cros
42540	Scranton--Wilkes-Barre, PA	4	0	Cros
25620	Hattiesburg, MS	4	0	Cros
3700002	Other North Carolina nonmetropolitan area	6	0	Cros
35620	New York-Northern New Jersey-Long Island, NY-NJ-PA	4	0	Cros
3700003	Western Central North Carolina nonmetropolitan area	6	0	Cros
2900002	North Missouri nonmetropolitan area	6	0	Cros
23540	Gainesville, FL	4	0	Cros

1-10 of 10 rows | 1-5 of 30 columns

```
dbClearResult(res)  # 結果セットをクリア
```

```
# 3. 列 'occtitle'、'h_mean'、'area_title' のサブセットを選択します
# グループ変数が 'major' のすべての観測値が対象です

res <- dbSendQuery(conemp1, "
                   SELECT occtitle, h_mean, area_title
                   FROM oesm
                   WHERE(grouping == 'major')
                  ")
fetch(res, n = -1)  # 全行を取得
```

```
## Warning: Column `h_mean`: mixed type, first seen values of type real, coercing
## other values of type string
```

occtitle <chr>	**h_mean** <dbl>
Architecture and Engineering Occupations	29.79
Architecture and Engineering Occupations	34.38
Architecture and Engineering Occupations	26.96
Architecture and Engineering Occupations	40.83
Architecture and Engineering Occupations	42.28
Architecture and Engineering Occupations	30.18
Architecture and Engineering Occupations	31.44
Architecture and Engineering Occupations	28.22
Architecture and Engineering Occupations	30.24
Architecture and Engineering Occupations	32.42

1-10 of 10,000 rows | 1-2 of 3 columns Previous **1** 2 3 4 5 6 … 1000 Next

```
dbClearResult(res)  # 結果セットをクリア
```

```
# 4. 変数 'naics' のユニークな値をすべて表示します

status <- dbSendQuery(conemp1, "
                      SELECT DISTINCT naics
                      FROM oesm")
fetch(status, n = -1)  # 全てのユニークな値を取得
```

```
## Warning: Column `naics`: mixed type, first seen values of type integer,
## coercing other values of type string
```

naics <int>
0
1
11
113000
113300
115000
115100
115200
21
211000

1-10 of 470 rows Previous **1** 2 3 4 5 6 … 47 Next

```
dbClearResult(status)  # 結果セットをクリア
```

```
# 5. データセット内の各年における観測値の数をカウントします

res2 <- dbSendQuery(conemp1, "
                    SELECT year, COUNT(*)
                    FROM oesm
                    GROUP BY year")
fetch(res2, n = -1)  # 全行を取得
```

year <int>	COUNT(*) <int>
2014	435004
2015	439942

2 rows

```
dbClearResult(res2)  # 結果セットをクリア
```

```
# 6. 各年ごとの年収中央値 (a_mean) の最小値と最大値を求めます

res <- dbSendQuery(conemp1, "
                   SELECT MIN(a_mean), MAX(a_mean)
                   FROM oesm
                   WHERE (a_mean IS NOT 'NA')
                     AND (a_mean IS NOT '#')
                     AND (a_mean != '')
                   GROUP BY year
                   ")
fetch(res, -1)  # 全行を取得
```

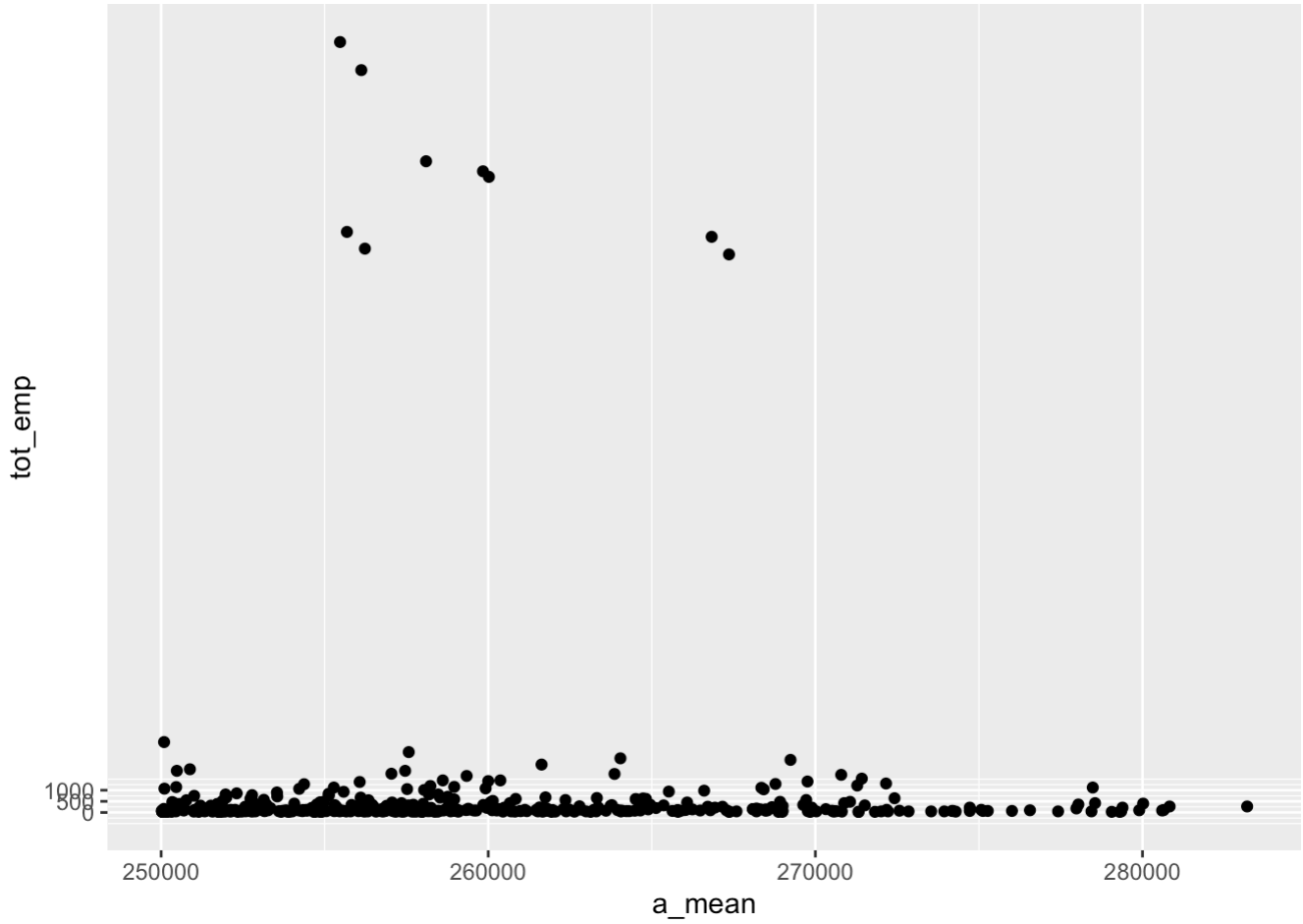
MIN(a_mean) <int>	MAX(a_mean) <int>
16600	277420
16740	286460

2 rows

```
dbClearResult(res)  # 結果セットをクリア
```

```
# 7. 同じデータベースから新しいテーブルを作成します
# このテーブルには 'occcode' と 'occtitle' のユニークな値を含めます

# すでに "New_oesm" テーブルが存在する場合は削除
if(dbExistsTable(conemp1, "New_oesm")) dbRemoveTable(conemp1, "New_oesm")

# ユニークな値を含む新しいテーブル "New_oesm" を作成
distinct <- dbSendQuery(conemp1, "
                        CREATE TABLE New_oesm AS
                        SELECT DISTINCT occcode, occtitle
                        FROM oesm
                        ")

dbClearResult(distinct)  # 結果セットをクリア
```

```
# 8. 新しいSQLスキルを活用して、いずれかのテーブルから情報を抽出し、
# 抽出データを用いて任意の視覚化を作成します。
# また、可視化に関するコメントを1〜2文で記述します。

# データの内容を確認するためのクエリ作成

# 従業員総数と年収中央値（$250,000以上）の関係を調査
res <- dbSendQuery(conemp1, "
                    SELECT tot_emp, a_mean
                    FROM oesm
                    WHERE (a_mean IS NOT 'NA')
                      AND (a_mean IS NOT '#')
                      AND (a_mean != '')
                      AND (a_mean > 250000)
                      AND (tot_emp IS NOT 'NA')
                    ")

# fetch関数でデータを抽出
htemp <- fetch(res, n = -1)

htemp$tot_emp <- as.numeric(htemp$tot_emp)  # 数値型に変換

# ggplotで散布図を作成
ggplot(data=htemp) +
  geom_point(mapping=aes(y=tot_emp, x=a_mean)) +
  scale_y_continuous(breaks = c(0,500,1000))
```




```
# 結果セットをクリア
dbClearResult(res)

##------------------------------
# # 8. 分析
##------------------------------
# 調査結果から、中央値年収が$250,000から$260,000の範囲である企業の多くは
# およそ1000人の従業員を抱えていることがわかりました。
# 一方で、巨大企業では年収中央値が$275,000から$268,000の範囲に収まる傾向が見られます。
```