R Journal Entry Analysis

HHH

2021 7 26

# Prerequisite

[R | 프로그래밍 언어 R 설치하기](https://joy-accounting.netlify.app/2021-07-19-r-install-windows/) 에서와 같이 설치하고 tidyverse 패키지가 정상적으로 설치되었다는 것을 전제하여 다음에 소개하는 Code를 스크립트 창에 조금씩 써보거나 복사해서 붙여 넣는 방식으로 실습한다.

## Data 준비하기

Rstudio에는 작업 환경을 분할해서 관리할 수 있는 훌륭한 기능이 있다. New Project로 Project를 생성하고 Project를 열어서 작업을 하면 작업 환경이 지정 되고 폴더도 생성되어 그 안에서 독립된 환경을 갖추어 작업 할 수 있다. Alt + f, p 를 눌러서 Project를 생성한다.

[다운로드](https://github.com/joy-hhh/R_for_JE_test)

다운로드 링크에서 우측 상단의 ↓ Code 을 클릭하여 Download ZIP 을 통해서 파일을 다운로드 받는다.

다운로드 받은 파일은 압축을 풀어 je.csv, CYTB.xlsx, PYTB.xlsx 세 개의 파일을 생성한 프로젝트 폴더에 옮겨 담는다.

# 시작

전체 과정은 R에 파일을 불러와서 작업하고 결과물을 파일로 저장하는 것으로 마친다.

## 파일 불러오기 (Data 읽기)

코드 블럭을 설정하거나 프롬프트가 위치한 행에서 ’Ctrl + Enter’를 누르면 선택 부분이 Console로 들어가서 실행된다.

R은 이 스크립트 창에서 Console에 Code를 던져 주면서 실행하는 맛이 있다.

library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5 v purrr 0.3.4  
## v tibble 3.1.3 v dplyr 1.0.7  
## v tidyr 1.1.3 v stringr 1.4.0  
## v readr 1.4.0 v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

1. tidyverse 패키지를 불러온다. 이렇게 명시해 주면 tidyverse의 기능을 R에 붙인다고 생각하면 된다.

je <- read\_csv('je.csv', locale=locale('ko',encoding='euc-kr'))

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## .default = col\_double(),  
## ACCT\_NM = col\_character(),  
## DR = col\_number(),  
## CR = col\_number(),  
## `거래처[코드]` = col\_character(),  
## `결의사원[코드]` = col\_character(),  
## `유형번호[유형명]` = col\_character(),  
## `증빙구분[구분명]` = col\_character(),  
## INSERT\_ID = col\_character(),  
## 입력일자 = col\_character(),  
## MODIFY\_ID = col\_character(),  
## MODIFY\_DT = col\_character()  
## )  
## i Use `spec()` for the full column specifications.

print(je)

## # A tibble: 309,043 x 22  
## JEDATE `구 분[기표][번~ JENO LN\_SQ DRCR\_FG ACCTCD ACCT\_NM DR CR  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <dbl>  
## 1 2.02e7 1 2.02e8 1 3 12002 매출채권~ 132000 NA  
## 2 2.02e7 1 2.02e8 2 4 10800 외상매출~ NA 66000  
## 3 2.02e7 1 2.02e8 3 4 10800 외상매출~ NA 66000  
## 4 2.02e7 46 2.02e9 1 3 10800 외상매출~ 332090 NA  
## 5 2.02e7 46 2.02e9 2 4 40401 제품매출 NA 301900  
## 6 2.02e7 46 2.02e9 3 4 25500 부가세예~ NA 30190  
## 7 2.02e7 47 2.02e9 1 3 10800 외상매출~ 216480 NA  
## 8 2.02e7 47 2.02e9 2 4 40401 제품매출 NA 196800  
## 9 2.02e7 47 2.02e9 3 4 25500 부가세예~ NA 19680  
## 10 2.02e7 48 2.02e9 1 3 10800 외상매출~ 66000 NA  
## # ... with 309,033 more rows, and 13 more variables: 거래처[코드] <chr>,  
## # 구 분[결의][년/월/일] <dbl>, 구 분[결의][번호] <dbl>, 결의부서[코드] <dbl>,  
## # 결의사원[코드] <chr>, PJT[코드] <dbl>, 유형번호[번호] <dbl>,  
## # 유형번호[유형명] <chr>, 증빙구분[구분명] <chr>, INSERT\_ID <chr>,  
## # 입력일자 <chr>, MODIFY\_ID <chr>, MODIFY\_DT <chr>

1. 프로젝트 폴더 경로에 전표 파일을 두고 읽는 명령으로 읽어와서 je라고 이름 붙인다.(이름 붙인 것을 객체라고 한다. je 객체) 윈도우즈 기준의 파일의 경우 파일명 뒤의 encoding 명시를 해주면 글자가 깨지지 않고 읽어온다. 윈도우 기준의 한글 형식에 맞출 수 있다. read\_csv 명령 함수는 utf-8 encoding을 기본으로 하기 때문에 한글이 깨질 수 있다. 2기가 정도 되는 data file이 100초 정도 걸린다. (컴퓨터 성능에 따라서 다르다.) 몇십메가 정도의 data file은 몇초가 되기 전에 읽어버린다.[[1]](#footnote-24)
2. 이름 붙인 je 객체를 Console에 나타내 확인한다.

## A01 Data Integrity Test

colnames(je)

## [1] "JEDATE" "구 분[기표][번호]" "JENO"   
## [4] "LN\_SQ" "DRCR\_FG" "ACCTCD"   
## [7] "ACCT\_NM" "DR" "CR"   
## [10] "거래처[코드]" "구 분[결의][년/월/일]" "구 분[결의][번호]"   
## [13] "결의부서[코드]" "결의사원[코드]" "PJT[코드]"   
## [16] "유형번호[번호]" "유형번호[유형명]" "증빙구분[구분명]"   
## [19] "INSERT\_ID" "입력일자" "MODIFY\_ID"   
## [22] "MODIFY\_DT"

str(je)

## spec\_tbl\_df [309,043 x 22] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ JEDATE : num [1:309043] 20170101 20170101 20170101 20170101 20170101 ...  
## $ 구 분[기표][번호] : num [1:309043] 1 1 1 46 46 46 47 47 47 48 ...  
## $ JENO : num [1:309043] 2.02e+08 2.02e+08 2.02e+08 2.02e+09 2.02e+09 ...  
## $ LN\_SQ : num [1:309043] 1 2 3 1 2 3 1 2 3 1 ...  
## $ DRCR\_FG : num [1:309043] 3 4 4 3 4 4 3 4 4 3 ...  
## $ ACCTCD : num [1:309043] 12002 10800 10800 10800 40401 ...  
## $ ACCT\_NM : chr [1:309043] "매출채권미수금(카드)" "외상매출금" "외상매출금" "외상매출금" ...  
## $ DR : num [1:309043] 132000 NA NA 332090 NA ...  
## $ CR : num [1:309043] NA 66000 66000 NA 301900 ...  
## $ 거래처[코드] : chr [1:309043] "80413" "12282" "09660" "11777" ...  
## $ 구 분[결의][년/월/일]: num [1:309043] 20170101 20170101 20170101 20170101 20170101 ...  
## $ 구 분[결의][번호] : num [1:309043] 1 1 1 2 2 2 3 3 3 4 ...  
## $ 결의부서[코드] : num [1:309043] 3530 3530 3530 1400 1400 1400 1400 1400 1400 1400 ...  
## $ 결의사원[코드] : chr [1:309043] "0000000771" "0000000771" "0000000771" "0000001446" ...  
## $ PJT[코드] : num [1:309043] 3530 3530 3530 3600 3600 NA 3600 3600 NA 3530 ...  
## $ 유형번호[번호] : num [1:309043] 1 1 1 3 3 3 3 3 3 3 ...  
## $ 유형번호[유형명] : chr [1:309043] "일 반" "일 반" "일 반" "매 출" ...  
## $ 증빙구분[구분명] : chr [1:309043] NA NA NA NA ...  
## $ INSERT\_ID : chr [1:309043] "0000000771" "0000000771" "0000000771" NA ...  
## $ 입력일자 : chr [1:309043] "2017-01-04 오후 1:31:48" "2017-01-04 오후 1:31:48" "2017-01-04 오후 1:31:48" NA ...  
## $ MODIFY\_ID : chr [1:309043] "0000000260" "0000000260" "0000000260" "0000000517" ...  
## $ MODIFY\_DT : chr [1:309043] "2017-01-06 오후 3:22:38" "2017-01-06 오후 3:22:38" "2017-01-06 오후 3:22:38" "2017-02-28 오전 11:30:04" ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. JEDATE = col\_double(),  
## .. `구 분[기표][번호]` = col\_double(),  
## .. JENO = col\_double(),  
## .. LN\_SQ = col\_double(),  
## .. DRCR\_FG = col\_double(),  
## .. ACCTCD = col\_double(),  
## .. ACCT\_NM = col\_character(),  
## .. DR = col\_number(),  
## .. CR = col\_number(),  
## .. `거래처[코드]` = col\_character(),  
## .. `구 분[결의][년/월/일]` = col\_double(),  
## .. `구 분[결의][번호]` = col\_double(),  
## .. `결의부서[코드]` = col\_double(),  
## .. `결의사원[코드]` = col\_character(),  
## .. `PJT[코드]` = col\_double(),  
## .. `유형번호[번호]` = col\_double(),  
## .. `유형번호[유형명]` = col\_character(),  
## .. `증빙구분[구분명]` = col\_character(),  
## .. INSERT\_ID = col\_character(),  
## .. 입력일자 = col\_character(),  
## .. MODIFY\_ID = col\_character(),  
## .. MODIFY\_DT = col\_character()  
## .. )

table(is.na(je))

##   
## FALSE TRUE   
## 5828389 970557

sapply(je, function(x) sum(is.na(x)))

## JEDATE 구 분[기표][번호] JENO   
## 0 0 0   
## LN\_SQ DRCR\_FG ACCTCD   
## 0 0 0   
## ACCT\_NM DR CR   
## 0 207399 101644   
## 거래처[코드] 구 분[결의][년/월/일] 구 분[결의][번호]   
## 2172 0 0   
## 결의부서[코드] 결의사원[코드] PJT[코드]   
## 0 0 78366   
## 유형번호[번호] 유형번호[유형명] 증빙구분[구분명]   
## 0 0 222980   
## INSERT\_ID 입력일자 MODIFY\_ID   
## 178226 178226 772   
## MODIFY\_DT   
## 772

max(je$JEDATE)

## [1] 20171231

min(je$JEDATE)

## [1] 20170101

1. 읽어온 je의 변수명(열 이름, column명)을 확인한다.
2. je의 구조를 파악한다.
3. je에 있는 Null 값을 집계한다.
4. je의 변수명에 따른 Null값을 각각 집계하여 보여준다.
5. 전표 일자의 가장 큰 값을 나타낸다.
6. 전표 일자의 가장 작은 값을 나타낸다.

## A02 전표번호 별 차대변 일치검증

A02 <- je |>   
 select(JENO, DR, CR) |>   
 mutate\_all(~replace(., is.na(.), 0)) |>   
 group\_by(JENO) |>   
 summarise(DR\_sum=sum(DR),  
 CR\_sum=sum(CR)) |>   
 mutate(Differ= DR\_sum - CR\_sum)  
print(A02)

## # A tibble: 80,612 x 4  
## JENO DR\_sum CR\_sum Differ  
## <dbl> <dbl> <dbl> <dbl>  
## 1 201701011 132000 132000 0  
## 2 201701021 334400 334400 0  
## 3 201701022 708400 708400 0  
## 4 201701023 31900 31900 0  
## 5 201701024 145500 145500 0  
## 6 201701025 786500 786500 0  
## 7 201701026 33000 33000 0  
## 8 201701027 13838 13838 0  
## 9 201701028 148500 148500 0  
## 10 201701029 571000 571000 0  
## # ... with 80,602 more rows

A02\_Differ <- count(A02[A02$Differ > 0, ])  
print(A02\_Differ)

## # A tibble: 1 x 1  
## n  
## <int>  
## 1 0

1. 전표번호, 차변금액, 대변금액 열을 선택(select)한다. 선택한 세 개의 열에 Null 값 (NA)이 있으면 0으로 모두 바꾼다.(mutate\_all) Null 값(NA)은 계산의 결과가 무조건 NA가 되면서 원하지 않은 불편한 결론을 내기 때문이다. 전표번호별로 묶어서(group\_by) 전표번호별 차변의 합계를 DR\_sum의 열 이름으로 전표번호별 대변의 합계를 CR\_sum의 열 이름으로 정리한다.(summarise) 전표번호별 차변 금액과 대변 금액 합계를 각각 차이를 계산하여 Differ라는 열 이름으로 추가(mutate)한다.
2. 전표번호별 차변 금액 합계와 대변 금액 합계의 차이 Differ열의 전체 합계를 계산한다.
3. 계산한 차이의 합계를 확인한다.

## A03 시산표 Reconciliation 검증(Trial Balance Rollforward Test)

시산표 검증에는 추가로 시산표 파일에서 Data를 읽어와야 한다.

시산표는 엑셀로 되어 있기 쉬우므로 엑셀 파일에서 바로 읽어온다. 전표 파일은 엑셀로 읽으면 csv 파일로 읽어 오는 것보다 훨씬 오래 걸린다. 대용량 Data는 엑셀이 담을 수 있는 행을 넘어버릴 것이므로 애초에 엑셀일 수가 없다. 시산표 정도는 프린터로 종이에 프린트 해도 두세장이면 담아 낼 수 있으므로 엑셀 파일로 바로 읽어도 상관 없다. 명령을 ’Ctrl + Enter’로 누르는 즉시 읽어들인다.

if(!require(readxl)){install.packages("readxl");library(readxl)}

## 필요한 패키지를 로딩중입니다: readxl

CYTB <- read\_excel('CYTB.xlsx')   
PYTB <- read\_excel('PYTB.xlsx')   
table(is.na(CYTB))

##   
## FALSE TRUE   
## 1948 28

colSums(is.na(CYTB))

## 차 변[잔 액] DRSUM 차 변[당월계] ACCTCD   
## 0 0 0 28   
## 계정과목 대 변[당월계] CRSUM 대 변[잔 액]   
## 0 0 0 0

table(is.na(PYTB))

##   
## FALSE TRUE   
## 1972 28

colSums(is.na(PYTB))

## 차 변[잔 액] DRSUM 차 변[당월계] ACCTCD   
## 0 0 0 28   
## 계정과목 대 변[당월계] CRSUM 대 변[잔 액]   
## 0 0 0 0

1. 엑셀을 읽기 위한 기능을 R에 부착한다. 생김새가 복잡한 이유는 readxl패키지가 설치되어 있지 않은 경우에는 설치하고 이미 설치되어 있는 경우에는 설치하지 않고 불러오라는 의미를 내포하였기 때문이다. 아직 readxl 패키지를 설치하지 않았으므로 바로 library 명령을 썼다가 오류가 나서 놀라서 충격 받지 않기를 바라는 배려이다.
2. 엑셀 파일 형태의 당기 시산표를 읽어들인다. (R이 칼질하기 좋게 R의 도마에 올린다.)
3. 엑셀 파일 형태의 전기 시산표를 읽어들인다.
4. 당기 시산표의 Null 값을 확인한다.
5. 당기 시산표의 변수별(열 이름)로 NA 갯수를 확인한다.
6. 전기 시산표의 Null 값을 확인한다.
7. 전기 시산표의 변수별(열 이름)로 NA 갯수를 확인한다.

Rstudio의 Powerful한 기능 View로 시산표를 볼 수 있다. 이 기능 때문이라도 Rstudio를 애정한다.

View(CYTB)

아주 자연스럽게 느껴지는 엑셀 같은 화면을 만날 수 있다. 물론 읽기 전용이다. 각 변수(열)에 필터 값으로 함축해 볼 수도 있다. 다른 언어에서 찾아 볼 수 없는 독보적인 기능이다. Rstudio는 시원하게 작업중인 중간에 들여다 볼 수가 있다. 무조건 말로만 해야 하는 것이 아니다. 보면서 해야 한다. 말로 하고 듣기도 하지만 볼수도 있어야 당연하다. 우리는 엑셀에 익숙하므로 즉각적으로 표를 보고 싶다. View(je) 로 전표 데이터를 볼 수도 있고 환경 창에 있는 변수를 마우스 클릭해도 View() 기능이 작동한다.

CYTB <- drop\_na(CYTB, ACCTCD)  
PYTB <- drop\_na(PYTB, ACCTCD)  
CYTB\_FP <- CYTB[1:99,]  
CYTB\_PL <- CYTB[100:length(CYTB$ACCTCD),]  
PYTB\_FP <- PYTB[1:103,]  
  
tail(CYTB\_FP)

## # A tibble: 6 x 8  
## `차 변[잔 액]` DRSUM `차 변[당월계~ ACCTCD 계정과목 `대 변[당월~  
## <dbl> <dbl> <dbl> <chr> <chr> <dbl>  
## 1 0 0 0 38300 자 기 ~ 0  
## 2 56201942 3.09e7 72053772 38900 매도가능증~ 15851830  
## 3 0 0 0 35100 이 익 준~ 0  
## 4 0 0 0 37200 부의지분법~ 0  
## 5 0 3.12e7 31216627 37400 보험수리적~ 36377018  
## 6 0 1.32e9 0 37500 전기이월이~ 0  
## # ... with 2 more variables: CRSUM <dbl>, 대 변[잔 액] <dbl>

head(CYTB\_PL)

## # A tibble: 6 x 8  
## `차 변[잔 액]` DRSUM `차 변[당월계~ ACCTCD 계정과목 `대 변[당월~  
## <dbl> <dbl> <dbl> <chr> <chr> <dbl>  
## 1 0 0 0 40100 상 품 ~ 468556458  
## 2 0 0 0 40401 제 품 ~ 3765154556  
## 3 0 0 0 40700 교 육 ~ 671429789  
## 4 0 0 0 41100 임 대 ~ 70591818  
## 5 0 0 0 41200 기 타 ~ 90910  
## 6 2790687018 2.79e9 626758061 45100 상 품 매 ~ 0  
## # ... with 2 more variables: CRSUM <dbl>, 대 변[잔 액] <dbl>

tail(PYTB\_FP)

## # A tibble: 6 x 8  
## `차 변[잔 액]` DRSUM `차 변[당월계~ ACCTCD 계정과목 `대 변[당월~  
## <dbl> <dbl> <dbl> <chr> <chr> <dbl>  
## 1 0 -7.14e6 0 38800 매도가능~ 0  
## 2 145999555 1.87e8 187178917 38900 매도가능~ 41179362  
## 3 0 0 0 35100 이 익 ~ 0  
## 4 0 0 0 37200 부의지분~ 0  
## 5 0 1.50e7 15037512 37400 보험수리~ 3308252  
## 6 0 0 0 37500 전기이월~ 0  
## # ... with 2 more variables: CRSUM <dbl>, 대 변[잔 액] <dbl>

CYTB\_FP <- full\_join(CYTB\_FP, PYTB\_FP, by='ACCTCD') |>   
 mutate\_all(~replace(., is.na(.), 0)) |>   
 mutate(move = (DRSUM.x - CRSUM.x) -(DRSUM.y - CRSUM.y)) |>   
 select(ACCTCD, move)  
  
CYTB\_PL <- CYTB\_PL |>   
 mutate(move = (DRSUM - CRSUM)) |>   
 select(ACCTCD, move)  
  
CYTB\_move <- bind\_rows(CYTB\_FP, CYTB\_PL)

1. 결측값(Null, NA)이 ACCTCD 열에 있었으며 결측값 때문에 어려움이 발생할 수 있으므로 결측을 없애고 시작한다.
2. 전기 시산표에서도 결측을 없앤다.
3. 결측을 없애고 나서 View(CYTB) 기능으로 읽어들인 시산표를 보니 99행이 시산표 중 재무상태표의 마지막 행이다. 시산표의 재무상태표 부분을 나누기 위하여 1행 부터 99행까지를 썰어낸다.(slicing)
4. 시산표의 손익계산서 부분을 뜯어내기 위하여 100행부터 끝까지 썰어낸다.
5. 시산표에서 썰어낸 당기재무상태표 하단을 확인한다. 재무상태표 끝단이 썰렸는지 확인.
6. 시산표에서 썰어낸 당기손익계산서 상단을 확인한다. 손익계산서 시작점부터 썰렸는지 확인.
7. 전기시산표에서 썰어낸 전기재무상태표 하단을 확인한다. 재무상태표 끝단이 썰렸는지 확인.
8. 전기 시산표의 재무상태표 부분을 썰어낸다. 전기 시산표가 필요한 이유는 시산표의 재무상태표 부분에는 기초잔액이 포함되어 있다. 전표에는 기초잔액이 포함되어 있지 않고 당기의 변동분만을 포함하고 있다. 만약 전표가 기초 잔액을 포함하고 있다면 이 Code는 수정되어야 할 수도 있다. 일반적으로 전표는 당기 변동을 분개로 포함하고 시산표의 재무상태표 부분은 기초잔액과 당기 변동을 포함하여 잔액이 기말 잔액이 되도록 한다. 기초잔액이 없이는 기말 잔액이 존재할 수 없다.
9. 당기시산표에 전기 시산표의 재무상태표 부분을 계정과목별로 맞춰서 붙인다. 이 때 당기시산표에 있는 계정과 전기시산표에 있는 계정이 모두 포함되도록 붙여야 한다. 전기에 변동이 있었으나 당기에 변동이 없었던 항목이 누락되지 않아야 하기 때문이다.
10. 시산표의 손익계산서 부분은 당기에 발생한 잔액만 있으면 된다. 재무상태표처럼 기초잔액이라는 개념이 없다.
11. 재무상태표의 잔액에서 기초잔액을 제거한 결과와 시산표의 손익계산서 부분의 잔액을 합하여 전표에서 당기에 변동이 있었던 금액을 하나로 붙인다.

A03 <- je |>   
 select(ACCTCD, DR, CR) |>   
 mutate\_all(~replace(., is.na(.), 0)) |>   
 group\_by(ACCTCD) |>   
 summarise(DR\_sum=sum(DR),  
 CR\_sum=sum(CR))  
  
A03$ACCTCD <- as.character(A03$ACCTCD)  
A03 <- left\_join(A03, CYTB\_move, by = 'ACCTCD')  
  
A03 <- A03 |> mutate\_all(~replace(.,is.na(.), 0)) |>   
 mutate(Differ = (DR\_sum - CR\_sum - move))  
A03\_Differ <- count(A03[A03$Differ > 0, ])  
A03[A03$Differ > 0, ]

## # A tibble: 1 x 5  
## ACCTCD DR\_sum CR\_sum move Differ  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 37500 1320667762 0 -2957850338 4278518100

1. 전표에서 계정과목별 차변금액과 대변금액 합계를 계산하여 준비한다.
2. 시산표에서 준비한 당기 변동금액(전표에서 계산한 계정과목별 당기 변동 금액)을 전표에서 계산한 차변금액과 대변 금액의 합계에 붙인다.
3. 결측값의 갯수를 확인한다.
4. 변수별 결측값의 갯수를 확인한다.
5. 결측값을 0으로 바꾸고 전표의 차변합계에서 대변합계를 빼서 당기변동분을 계산하고 시산표에서 계산한 당기변동분을 빼서 차이를 낸다.
6. 차이가 0보다 큰 항목 갯수를 세어 A03\_Differ 변수로 지정한다.
7. A03 변수에서 차이가 0보다 큰 항목을 찾아 나타낸다. 이익잉여금 항목 하나만 나타나면 이상이 없다고 생각할 수 있다.

## B01 매출의 상대계정분석

Corr\_Acc = '40401'  
B09\_main <- je |> filter(ACCTCD == Corr\_Acc) |>   
 select(JENO, ACCTCD)  
B09\_Corr <- je |>   
 select(JENO, ACCTCD)  
B09 <- semi\_join(B09\_Corr, B09\_main, by = 'JENO')  
B09 <- B09 |> filter(!is.na(ACCTCD)) |>   
 count(ACCTCD)

1. 상대계정분석에서 기준이 되는 본계정코드를 설정한다.
2. 전표 데이터에서 본계정이 포함된 전표번호를 필터링(filter)한다. (행 선택)
3. 전표 데이터에서 전표번호와 계정코드 열을 선택(select)한다. (열 선택)
4. 선택한 전표번호와 계정코드에서 본계정이 포함된 전표번호만 남겨서 B09 변수로 지정한다.
5. 본계정과 상대계정의 전표 갯수를 세어 B09 변수로 지정한다.

View(B09) 명령으로 확인해 보거나 B09를 Console에 쳐서 확인해 볼 수도 있다. 계정코드를 기준으로 본계정과 상대계정들이 모여 있다. 계정과목명이 보이지 않아 보기에 불편하므로 계정과목명을 붙인다.

B09\_name <- je |> select(ACCTCD, ACCT\_NM) |> distinct()   
B09 <- left\_join(B09, B09\_name, by = 'ACCTCD')   
print(B09)

## # A tibble: 8 x 3  
## ACCTCD n ACCT\_NM   
## <dbl> <int> <chr>   
## 1 10800 57348 외상매출금   
## 2 11300 66 진행률미수금   
## 3 25500 57216 부가세예수금   
## 4 25900 75 선수금   
## 5 26900 2 프로젝트손실충당부채   
## 6 40100 44 상품매출   
## 7 40401 57419 제품매출   
## 8 54100 1 프로젝트손실전입액(제)

1. 계정코드와 계정과목명 열을 선택한다. 중복되는 계정과목명을 삭제하고 하나씩만 남긴다.
2. 계정코드에 맞는 계정과목명을 B09 에 붙인다.

## write file

R에 파일의 Data를 읽어와서 요리가 되었는데 이 요리의 결과물을 다시 파일로 갖고 싶다. 과정을 거쳐 오면서 결정적 증거물은 메모리에 객체로 꼬리표를 남겨 두었다. 이 객체를 파일로 저장하고 싶다면 다음 명령어들로 할 수 있다.

A02 |> write\_csv('A02.csv')  
A03 |> write\_csv('A03.csv')  
B09 |> write\_csv('B09.csv')

“나는 csv 싫다. 엑셀로 저장할거다.” 하는 마음이라면 다음과 같이 하면 된다.

if(!require(writexl)){install.packages("writexl");library(writexl)}   
  
A02 |> write\_xlsx('A02.xlsx')  
A03 |> write\_xlsx('A03.xlsx')  
B09 |> write\_xlsx('B09.xlsx')

Coding의 매력은 강력한 재사용성이다. Coding을 하느라 끙끙 앓을 수도 있지만 한번 앓고 나면 그 과정은 내가 아니라 컴퓨터가 한다. 엄청난 속도로 일처리에 부스터를 달고 날아가는 기분이다.

반드시 한번 시도해 볼만 하다. 개발자가 되고자 하는 것이 아니다. 컴퓨터를 말로 부려먹는 느낌을 가져보고 싶은 것이다.

|  |
| --- |
| 조이회계 Joy Accounting |
| <https://joy-accounting.netlify.com/> |

1. 원래 utf-8 encoding 형식으로 되어 있는 파일을 , locale=locale(‘ko’,encoding=‘euc-kr)’ 없이 읽으면 훨씬 더 빠르게 읽는다. 파일 형식에 대한 전환 과정이 필요 없기 때문이다. 대용량의 전표를 받아서 test 할 때는 utf-8 encoding 형식으로 요청하는 것도 방법이다. [↑](#footnote-ref-24)