$\boxed{\text{derive}}$ test score

* get ready for merge

Student_ID	Subject	month	score	missing	score_derived
←	Averge		→		

① calculate averge

② replace NA

⊃ how to use assert?

* reshape ⇒ 両方

end_with

se

rowbind

```
main <- func(){                    interim
    data <- read_~~intermediate~~(          )
    data %>%
                         var_name = "score_derived"
         gen_na_assert() %>%
gen argmented  compute ~~gen~~_average() %>%
         save_interim(folder = , file = )
}.

replace_na <- func(data_input, var_name){
    data_output <- data_input %>%
         dplyr:: mutate(missing_dummy = { 1   if  score == . or NA %>%
                                          0
         dplyr:: mutate(score_derived = { score  if  missing = 0
                                          0                    1
    return(data_output)
}
                                                CHECK.
compute_average <- func(data_input){
    data_avg  <- data_input %>%
         dplyr:: group_by(studentID, ~~subject~~, month) %>%
         dplyr:: summarize(score by subject)
      reorganize => subject : avrge
      stack
    & [avrage] by studentID.
    return(data_output)
}.
```

* なぜいちいちつけるの

→ NOT vulnerable to new packages redefining the name.

· × conflicted.

derive breakfast

student_name	month	date	breakfast	duplicate

obs

pancake	dog flakes
1	0
0	1
⋮	⋮

for

① filter duplicate = 0 ! 1

② count # of pancakes / dogflakes

③ consistent names ? X

④ missing, or # days per month

dummies

(and / or operator

%in%

X pivot_wider

in Tibble, space allowed!

```
main <- func(){
    data_input <- read_intermediate(folder, file= )

    data_interim <- data_input %>%
        reshape_duplicate() %>%
        gen_dummy(varname = "pancakes",
                  var_list = c("pancake",
                               "hot cakes",
                               "pan cakes")) %>%      breakfast

        gen_dummy(varname = "dog flakes",
                  var_list = "dog flakes")      list

    data_interim %>% remove_~~irrelevant~~_date()      winter break
        collapse_date() %>%
        save_interim(file_name =        )
}

reshape_duplicate <- func(data_input)
    pivot_wider()
    return(data_output)
}
```

```
gen_breakfast_dummy <- func(data_input, varname, var list)
    data_output <- data_input %>%
        dplyr::mutate(  or ifel?
            varname = { 1
                      { 0   if  breakfast* ∈ varlist
    return(data_output)
}

remove_~~irrelevant~~_date <- func(){      list      12月20日 ~1月3日
    filter( not in varlist)
    return(output)
}

collapse_date <- func(){
    group_by(student_name, month)
    ~~max~~ sum (mean)  pancake
            max         dogflakes
```

final_merge.

score. pancake_frac dogflake_dummy

Select & merge.

Student_ID	month	

,

student_na	month		

,

Student_ID	student_na

derived derived

test_score.rDS   breakfast.Rds   student_id.Rds.

① co~~llapse breakfast~~ date

②

{ string
  numeric

Vit: translate into English

~~rename~~ month. ~~[illegible]~~ chge

```
main <- func(){
    test_score <- read_interim( folder= , file= )
    breakfast <-              "
    student_id <-       raw(     ).   ..:
    master_data <- merge_all(    ,    ,    )
    save_interim( master, folder=   )
                 prep_merge(___
}

merge_all <- func(  ,  ,  ){.
    ① merge on 1 key
    ③ merge on 2 keys
```

check_merge variable like STATA?

* inner_join
  left_join
  ↳ * check e.g.

* convert

.gen_month

student_ID	student_na	month	score	subject	pancake%	dog flake
		9				
		10				
		11				
		12				
		.				