Reporting rate is a denominator, not a footnote

The system computes four completeness measures and they disagree. Which one your report quotes decides whether a coverage figure is an estimate or a lower bound, and August is 29% either way.

Pierre Robentz Cassion

Lesson 4 of 8

Routine Data and DHIS2 — Periods and completeness

What this lesson covers

- Four measures, one word
- Where the system's answer differs from yours
- Reporting rate as the denominator of everything else
- The two ways to handle a silent facility
- Report the pair
- What comes next

Four measures, one word

Measure	Numerator	Denominator
Reporting rate	Datasets marked complete	Datasets expected
Actual reports	Datasets marked complete	— (a count)
Expected reports	—	Org units assigned x periods
Reporting rate on time	Completed by the deadline	Datasets expected

Four measures, one word — In Python

```
import pandas as pd

vax = pd.read_csv("vaccination-coverage-2024.v1.csv", parse_dates=["period"])
vax["reported"] = vax["report_submitted"] == True

facility_months = vax.drop_duplicates(["facility_id", "period"])
print(f"expected: {len(facility_months)}")
print(f"actual:   {facility_months['reported'].sum()}")
print(f"rate:     {facility_months['reported'].mean():.1%}")
```

Four measures, one word — In R

```
library(dplyr)

vax |>
  distinct(facility_id, period, .keep_all = TRUE) |>
  summarise(expected = n(), actual = sum(report_submitted),
            rate = mean(report_submitted))
```

Where the system's answer differs from yours

- Expected reports counts assignment, not existence — A facility that closed in March remains in the denominator for the…
- Completeness is a click, not a calculation — A facility can enter every value and never mark the form complete; the…

Where the system's answer differs from yours — In Python

```
has_data = (
    vax.groupby(["facility_id", "period"])["doses_administered"]
    .sum().gt(0).rename("has_data")
)
flags = facility_months.set_index(["facility_id", "period"])["reported"]

crosstab = pd.crosstab(flags, has_data.reindex(flags.index))
print(crosstab)
```

Where the system's answer differs from yours — In R

```
vax |>
  summarise(reported = first(report_submitted),
            has_data = sum(doses_administered) > 0,
            .by = c(facility_id, period)) |>
  count(reported, has_data)
```

Reporting rate as the denominator of everything else — In Python

```
monthly = (
    vax[vax["antigen"] == "penta3"]
    .groupby(vax["period"].dt.strftime("%Y-%m"))
    .apply(lambda g: pd.Series({
        "reporting_rate": g["reported"].mean(),
        "coverage_reporting": (g.loc[g["reported"], "doses_administered"].sum()
                               / g.loc[g["reported"], "target_population"].sum()),
    }))
)
print((monthly * 100).round(1))
```

Reporting rate as the denominator of everything else — In R

```
vax |>
  filter(antigen == "penta3") |>
  mutate(month = format(period, "%Y-%m")) |>
  summarise(
    reporting_rate = mean(report_submitted),
    coverage = sum(doses_administered[report_submitted]) /
               sum(target_population[report_submitted]),
    .by = month
  )
```

Reporting rate as the denominator of everything else

Month	Reporting rate	Coverage among reporters
July	82%	...
August	**29%**	...
September	**45%**	...
October	95%	...

Reporting rate as the denominator of everything else

- Three rules that follow, and they are the deliverable of this lesson
- **Below 90% reporting, a coverage figure is a lower bound**, not an estimate. Label it "among reporting facilities"...
- **Do not compare two periods with materially different reporting rates** without saying so. August against July is a...
- **A trend line needs the reporting rate on the same chart.** Plotted alone, a coverage series through August looks like...

The two ways to handle a silent facility

- Exclude it from both numerator and denominator — Coverage among facilities that reported
- Impute its doses — Fill from the facility's own recent months, and say so

The two ways to handle a silent facility — In Python

```
by_facility = (
    vax[(vax["antigen"] == "penta3") & vax["reported"]]
    .groupby("facility_id")["doses_administered"].median()
)
missing = (vax["antigen"] == "penta3") & ~vax["reported"]
imputed_total = (vax.loc[missing, "facility_id"].map(by_facility).sum()
                 + vax.loc[(vax["antigen"] == "penta3") & vax["reported"],
                           "doses_administered"].sum())

annual_target = (vax[vax["antigen"] == "penta3"]
                 .groupby("facility_id")["target_population"].first().sum())
print(f"imputed annual coverage: {imputed_total / annual_target:.1%}")
```

The two ways to handle a silent facility — In R

```
median_by_facility <- vax |>
  filter(antigen == "penta3", report_submitted) |>
  summarise(m = median(doses_administered), .by = facility_id)
```

The two ways to handle a silent facility

- Never impute silently — An imputed figure and a reported one in the same column with no flag is the defect the cleaning...

Report the pair — In Python

```
summary = pd.DataFrame({
    "period": ["2024-08", "2024-09", "2024-10"],
    "facilities_reporting": [11, 17, 36],
    "facilities_expected": [38, 38, 38],
    "reporting_rate": ["29%", "45%", "95%"],
    "coverage_basis": ["among reporting facilities"] * 3,
})
```

Report the pair — In R

```
tibble::tribble(
  ~period,   ~reporting, ~expected, ~basis,
  "2024-08",         11,        38, "among reporting facilities",
  "2024-09",         17,        38, "among reporting facilities",
  "2024-10",         36,        38, "among reporting facilities"
)
```

What comes next

- Everything so far has run on a CSV somebody exported by hand.

Read the full lesson, with runnable code

https://data-analysis.cassion.dev/courses/routine-data-and-dhis2/

dhis2-04-reporting-rate-as-denominator