From a defect to the thing that caused it

Five categories of cause, a stopping rule for the five whys, and why "the staff need training" is the answer that gets written when nobody looked. Test the cause against the data before you put it in the report.

Pierre Robentz Cassion

Lesson 7 of 8

Data Quality Assessment — Turning findings into change

What this lesson covers

- A finding is not a cause, and only a cause can be fixed
- Five categories, and they need different money
- Let the data narrow it before you ask anyone
- Then ask the form
- The five whys, with a stopping rule
- Test the cause against the data
- Put the cause at the level the fix lives at
- What comes next

A finding is not a cause, and only a cause can be fixed

Recommendation. Train staff on data quality.

Five categories, and they need different money

Category	Looks like	Fixed by
Definition	Two sites counting different things under one indicator name	A written indicator reference sheet
Tool	A form that cannot express what happened, or a system that drops values	A form or configuration change
Capacity	The right tool used wrongly, consistently, by someone who was never shown	Supervision and job aids, occasionally training
Incentive	Reporting that improves when it is looked at, or numbers that meet a target exactly	Changing what the number is used for
System	Everyone fails at once — no forms, no fuel, no network, no supervisor	Logistics, budget, staffing

Five categories, and they need different money

- Definition is the largest category and the least suspected — A verification factor far from 1 is more often two people. . .
- Capacity is the smallest category and the most frequently blamed — because it is the only one where the fix is a. . .

Let the data narrow it before you ask anyone — In Python

```
by_month = vax.groupby(vax["period"].dt.strftime("%Y-%m"))["reported"].mean()
by_facility = vax.groupby("facility_id")["reported"].mean()

print("spread across months:  ", round(by_month.max() - by_month.min(), 2))
print("spread across facilities:", round(by_facility.max() - by_facility.min(), 2))
```

Let the data narrow it before you ask anyone — In R

```r
by_month <- vax |> mutate(m = format(period, "%Y-%m")) |>
  summarise(r = mean(report_submitted), .by = m)
by_facility <- vax |> summarise(r = mean(report_submitted), .by = facility_id)

c(months = diff(range(by_month$r)), facilities = diff(range(by_facility$r)))
```

Let the data narrow it before you ask anyone

- **Concentrated in time, spread across units.** August, everywhere. That is a system cause — something failed for the...
- **Concentrated in one unit, spread across time.** SCH09's `Y/N` coding, every month. That is a tool or definition...
- **Spread across both.** Age heaping, every team, all year. That is a method cause, inherent to how the measurement is...
- Write the shape down before you propose a cause — It rules out three of the five categories for free, and it is the...

Then ask the form

- What values can the form actually accept? SCH09's registers offered `Y/N` boxes; the extract's boolean cast...
- Is the field required? An optional field with a 40% missing rate is a form design decision, not a compliance problem.
- What does the field label say? A column headed "cases" collects whatever the person filling it believes a case is.

The five whys, with a stopping rule

- Stop when you reach something a named person can change with a budget you can estimate — Anything past that point is...

The five whys, with a stopping rule

Attendance was under-reported by 42% in SCH09. — *Why?* The extract read `Y` as missing. — *Why?* The register used `Y/N` and the form expected `true/false`. — *Why?* The paper register printed in 2019 has yes/no boxes; the digital form was designed later, separately. — **Stop.** The education office can reprint the register or the HMIS team can accept both codings. Both are costed in an afternoon.

Test the cause against the data — In Python

```
# Cause: one school's register uses Y/N. Prediction: Y/N appears in that
# school only, and in most of its months.
yn = attendance[attendance["present"].isin(["Y", "N"])].merge(
    roster[["student_id", "school_id"]], on="student_id", how="left"
)
print(yn["school_id"].value_counts())
print(yn.groupby("school_id")["attendance_date"].nunique())
```

Test the cause against the data — In R

```
attendance |>
  filter(present %in% c("Y", "N")) |>
  left_join(select(roster, student_id, school_id), by = "student_id") |>
  count(school_id)
```

Put the cause at the level the fix lives at

- A **facility-level** cause needs a facility-level action. "FAC020's tally sheet is missing" is fixed by delivering a...
- A **district-level** cause needs a district-level action. "No facility received forms in August" is not thirty-eight...
- A **system-level** cause needs a system-level action, and naming it is often the most valuable thing in the report even...

Put the cause at the level the fix lives at

The purpose of naming causes is that the next round is better. A report that assigns blame accurately and changes nothing has failed at the only thing it was for.

What comes next

- You have findings, and now each has a cause with a level and a test behind it.

Read the full lesson, with runnable code

https://data-analysis.cassion.dev/courses/data-quality-assessment/dqa-07-root-cause