Dates, ages and reporting periods

Age in completed months computed correctly rather than by dividing days, reporting periods that survive a partial month, and the boundary arithmetic a growth standard depends on.

Pierre Robentz Cassion

Lesson 7 of 8

Python for Programme Data — Producing and handing over

What this lesson covers

- Why this gets its own lesson
- Parsing, once, explicitly
- Age in completed months
- Reporting periods
- Resampling a time series
- Durations
- Time zones, briefly
- What comes next

Why this gets its own lesson

- Date arithmetic looks like the easy part and produces some of the most expensive errors in this sector.

Parsing, once, explicitly — In Python

```python
import pandas as pd

muac = pd.read_csv(path)
muac["screening_date"] = pd.to_datetime(
    muac["screening_date"], format="%Y-%m-%d", errors="raise"
)
```

Parsing, once, explicitly

- **State the format.** An inferred format can change between files as the mix of values changes — the same column parsed...
- **Use `errors="raise"`.** The alternative, `errors="coerce"`, turns every unparseable date into missing, so a file in...

Age in completed months — In Python

```
# Wrong often enough to matter.
age_months = ((visit_date - date_of_birth).dt.days / 30.4375).astype(int)
```

Age in completed months — In Python

```
def age_in_months(dob: pd.Series, visit: pd.Series) -> pd.Series:
    months = (visit.dt.year - dob.dt.year) * 12 + (visit.dt.month - dob.dt.month)
    # Not yet reached the day-of-month, so the last month is not complete.
    return months - (visit.dt.day < dob.dt.day).astype(int)
```

Age in completed months — In Python

```
dob   = pd.to_datetime(["2022-03-15", "2022-03-16", "2022-02-28"])
visit = pd.to_datetime(["2024-03-15", "2024-03-15", "2024-03-01"])

age_in_months(pd.Series(dob), pd.Series(visit)).tolist()   # [24, 23, 24]
```

Age in completed months

- Why one month matters here
- WHO growth standards switch measurement at exactly 24 months — Below 24 months a child is measured lying down and...
- Age bands are the other place — `0-5`, `6-23`, `24-59` months are the standard nutrition bands, and a child at exactly...

Age in completed months — In Python

```
BANDS = [0, 6, 24, 60]
LABELS = ["0-5m", "6-23m", "24-59m"]

muac["age_band"] = pd.cut(
    muac["age_months"], bins=BANDS, labels=LABELS, right=False, include_lowest=True
)
```

Age in completed months — In Python

```
check = muac.loc[muac["age_months"].isin([5, 6, 23, 24, 59, 60]),
                 ["age_months", "age_band"]].drop_duplicates().sort_values("age_months")
print(check)
```

Reporting periods — In Python

```
muac["month"] = muac["screening_date"].dt.to_period("M")

by_month = muac.groupby("month").size()
```

Reporting periods — Example

```
2024-01     207
2024-02     391
...
2024-11     375
2024-12     156
```

Reporting periods — In Python

```
current = pd.Period("2024-06", freq="M")
print(current - 1)                  # 2024-05
print(current.start_time, current.end_time)
```

Reporting periods

- Partial periods are the trap

Reporting periods — In Python

```
first, last = muac["screening_date"].min(), muac["screening_date"].max()
print(first.date(), last.date())

muac["partial_period"] = muac["month"].isin(
    [first.to_period("M"), last.to_period("M")]
)
```

Reporting periods

- Label them rather than dropping them — A reader who sees December fall off a chart concludes the programme collapsed; a…
- Aligning two sources on periods

Reporting periods — In Python

```
epi = pd.read_csv(path, parse_dates=["period"])
epi["month"] = epi["period"].dt.to_period("M")
print(epi["month"].nunique())      # 12
```

Resampling a time series — In Python

```
daily = muac.set_index("screening_date")
weekly = daily.resample("W")["child_id"].size()
monthly = daily.resample("MS")["child_id"].size()       # MS = month start
```

Durations — In Python

```
referrals["days_to_service"] = (
    referrals["service_date"] - referrals["referral_date"]
).dt.days
```

Durations — In Python

```
hours = (referrals["service_date"] - referrals["referral_date"]) / pd.Timedelta(hours=1)
referrals["same_day"] = hours < 24
```

Durations — In Python

```
impossible = referrals["days_to_service"] < 0
print(f"service recorded before referral: {impossible.sum()}")
```

Time zones, briefly — In Python

```
submissions["submitted_at"] = pd.to_datetime(
    submissions["submitted_at"], utc=True
).dt.tz_convert("America/Port-au-Prince")

submissions["submission_date"] = submissions["submitted_at"].dt.date
```

What comes next

- Every column now means what it says, including the ones made of dates.

Read the full lesson, with runnable code

https://data-analysis.cassion.dev/courses/python-for-programme-data/python-07-dates-ages-periods