

Upstream renamed a column. You found out at 3am.

WHAT IT IS

a contract	schema + rules, versioned
owner	the producing team
enforced	← or it is a wiki page.

WHAT IT COVERS

schema	columns, types, nullable
semantics	what amount means, currency
SLA	fresh by 06:00, daily
PII	flagged, masked

WHERE IT LIVES

in the repo	yaml next to the producer
CI check	breaking change = red build
catalog	← discoverable or dead.

BREAKING OR NOT

rename / drop	breaking, new version
add nullable col	safe, notify
type change	breaking, always

ROLLOUT

v2 beside v1	consumers migrate
deprecate date	then remove
tests at the edge	dbt tests, GX

GOTCHAS

contract, no tests	good intentions
consumer-defined	producers must own it
one giant schema	per table, per product

WORTH MEMORIZING

01 A contract is a file, not a meeting

Twelve lines next to the producer code. CI reads it, consumers read it, nobody is surprised.

```
-- contracts/orders.yml, owned by team-checkout
version: 2
schema:
  order_id: {type: string, required: true}
  amount:   {type: decimal, currency: EUR}
  placed_at: {type: timestamp, tz: UTC}
sla:       {fresh_by: "06:00", cadence: daily}
pii:       [customer_email]
breaking:  rename, drop, type change -> v3
-- CI fails the producer PR when the schema drifts
```

→ The file is the enforcement point: a schema test in the producer CI, a freshness test on the consumer side, and a catalog entry generated from the same yaml.

02 Trap: the same change, with and without

Three real incidents and how a contract turns each one into a blocked pull request.

THE CHANGE	WITHOUT A CONTRACT	WITH ONE
amount → amount_cents	revenue x100 on Monday	CI blocked the PR
user_id string → int	joins returned 0 rows	v2 published, v1 kept
file lands at 09:30	dashboard empty at 08:00	SLA alert to the producer

→ Interview phrasing: a data contract moves the failure from my dashboard at 3am to their pull request at 3pm.