The pipeline ran twice. So did the charges.

D8A.ACADEMY/RESOURCES/IDEMPOTENCY

01 The crime scene

Nothing here is broken. That is the problem: it works twice.

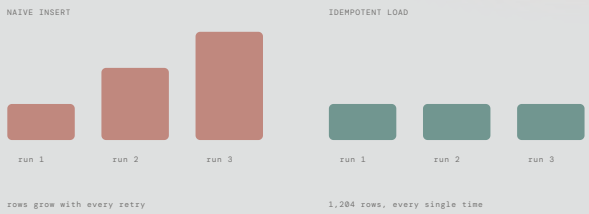
```
-- runs every night at 03:00
INSERT INTO orders
SELECT * FROM staging_orders;

-- timeout after commit, before ack.
-- the scheduler does what
-- schedulers do: it retries.
```

→ The retry is not the bug. Retries are normal. A pipeline that cannot survive one is the bug.

02 Idempotent, in one picture

Run it once, twice, or five times: the table ends up identical.



→ Formally: f(f(x)) = f(x). Practically: reruns are free, and 3am pages stop.

03 Fix 1: MERGE on a key

Let the natural key decide: update if it exists, insert if it does not.

```
MERGE INTO orders o
USING staging_orders s
  ON o.order_id = s.order_id
WHEN MATCHED THEN UPDATE
  SET amount = s.amount
WHEN NOT MATCHED THEN
  INSERT VALUES (s.*);
```

→ Postgres spells it INSERT … ON CONFLICT DO UPDATE. Same idea, same safety.

04 Fix 2: overwrite the slice

Delete the day, reload the day. The rerun rewrites the same slice.

```
BEGIN;
DELETE FROM orders
 WHERE order_date = '2026-08-18';
INSERT INTO orders
SELECT * FROM staging_orders
 WHERE order_date = '2026-08-18';
COMMIT;
```

→ The warehouse version is INSERT OVERWRITE PARTITION. One transaction, or you traded one bug for another.

⚠ COMMON TRAPS

05 The rerun checklist

Before you ship a pipeline, ask it these four questions.

`natural key?`	Something stable to merge on. Not the auto-increment id.
`scoped runs?`	A run owns a slice (a date, a batch id) it can rewrite.
`safe retries?`	Run it twice in staging. Diff the row counts.
`late data?`	Yesterday reruns when stragglers arrive. Same property saves you.

→ If you cannot rerun yesterday without fear, you do not have a pipeline. You have a ritual.