**Business Case Solution**

# **Task 1 — Build the “Delivery Facts” base (one row per order × seller)**

## **Problem**

Right now, information about a delivery is scattered across many tables (orders, items, customers, sellers). If we keep joining them again and again, we’ll waste time and make mistakes. We need **one clean base** that shows, for each shipment:

* who shipped (seller) and to which state (customer),
* when we **approved**, **handed to carrier**, **delivered**, and what we **promised**,
* how many days it took in total (**lead time**), how long the **seller took before shipping** (**handling**), how long the **courier took** (**transit**),
* and whether it was **on time** or **late**.

## **What this solves**

* Gives us **one trusted table** so we don’t repeat messy joins.
* Splits delay into **seller side (handling)** vs **courier side (transit)** — critical for root-cause.
* Everything else (lanes, slow sellers, hot spots) will reuse this base.

## **SQL (create a reusable view)**

CREATE OR REPLACE VIEW v\_delivery\_facts\_plus AS

WITH order\_seller AS (

-- Ensure one row per order × seller (orders can have multiple items per seller)

SELECT DISTINCT order\_id, seller\_id

FROM olist\_order\_items

)

SELECT

os.order\_id,

os.seller\_id,

o.customer\_id,

c.customer\_state,

s.seller\_state,

-- Key milestones

o.order\_approved\_at,

o.order\_delivered\_carrier\_date, -- when seller handed off to carrier

o.order\_delivered\_customer\_date, -- when customer received it

o.order\_estimated\_delivery\_date, -- promised latest date

-- KPIs (in days; rounded)

ROUND(EXTRACT(EPOCH FROM (o.order\_delivered\_customer\_date - o.order\_approved\_at))/86400.0, 2)

AS lead\_time\_days,

ROUND(EXTRACT(EPOCH FROM (o.order\_delivered\_carrier\_date - o.order\_approved\_at))/86400.0, 2)

AS handling\_days,

ROUND(EXTRACT(EPOCH FROM (o.order\_delivered\_customer\_date - o.order\_delivered\_carrier\_date))/86400.0, 2)

AS transit\_days,

ROUND(EXTRACT(EPOCH FROM (o.order\_estimated\_delivery\_date - o.order\_approved\_at))/86400.0, 2)

AS promised\_days,

GREATEST(

ROUND(EXTRACT(EPOCH FROM (o.order\_delivered\_customer\_date - o.order\_estimated\_delivery\_date))/86400.0, 2),

0

) AS lateness\_days,

CASE WHEN o.order\_delivered\_customer\_date <= o.order\_estimated\_delivery\_date THEN 1 ELSE 0 END

AS is\_on\_time

FROM olist\_orders o

JOIN order\_seller os ON os.order\_id = o.order\_id

JOIN olist\_customers c ON c.customer\_id = o.customer\_id

JOIN olist\_sellers s ON s.seller\_id = os.seller\_id

WHERE o.order\_status = 'delivered'

AND o.order\_approved\_at IS NOT NULL

AND o.order\_delivered\_customer\_date IS NOT NULL

AND o.order\_estimated\_delivery\_date IS NOT NULL;

**Quick sanity check**

SELECT COUNT(\*) AS rows,

ROUND(AVG(is\_on\_time)::numeric,3) AS on\_time\_rate,

MIN(lead\_time\_days) AS min\_lead, MAX(lead\_time\_days) AS max\_lead

FROM v\_delivery\_facts\_plus;

# **Task 2 — Lane performance (which routes are slow?)**

## **Problem**

A “lane” is the route **seller\_state → customer\_state**. Ops needs to know **which routes fail most** so they can talk to the right courier partners or adjust routing. We also want to see if slowness comes more from **handling** (seller) or **transit** (courier) by looking at typical times.

## **What this solves**

* Turns millions of rows into a **short list of problem routes**.
* Adds **volume guards** so we don’t chase tiny lanes.
* Gives typical **handling vs transit medians** to hint at the cause.

## **SQL (lane KPIs with medians + “worst 10” list)**

WITH lane AS (

SELECT

seller\_state,

customer\_state,

COUNT(\*) AS orders,

ROUND(AVG(is\_on\_time)::numeric, 3) AS on\_time\_rate,

ROUND(AVG(lead\_time\_days)::numeric, 2) AS avg\_lead\_time\_days,

-- Medians reduce outlier noise for diagnosis

percentile\_cont(0.5) WITHIN GROUP (ORDER BY handling\_days) AS p50\_handling\_days,

percentile\_cont(0.5) WITHIN GROUP (ORDER BY transit\_days) AS p50\_transit\_days

FROM v\_delivery\_facts\_plus

GROUP BY seller\_state, customer\_state

)

-- Worst 10 lanes to fix first (real traffic + poor on-time)

SELECT seller\_state, customer\_state, orders, on\_time\_rate, avg\_lead\_time\_days,

ROUND(p50\_handling\_days::numeric,2) AS p50\_handling\_days,

ROUND(p50\_transit\_days::numeric,2) AS p50\_transit\_days

FROM lane

WHERE orders >= 300 -- volume guard; tweak if you like

AND on\_time\_rate < 0.85 -- “bad” threshold; tweak to your SLA

ORDER BY on\_time\_rate ASC, orders DESC

LIMIT 10;

**If you also want a full lane table for reference**, save it as a view:

CREATE OR REPLACE VIEW v\_lane\_performance AS

SELECT \* FROM (

SELECT

seller\_state,

customer\_state,

COUNT(\*) AS orders,

ROUND(AVG(is\_on\_time)::numeric, 3) AS on\_time\_rate,

ROUND(AVG(lead\_time\_days)::numeric, 2) AS avg\_lead\_time\_days,

percentile\_cont(0.5) WITHIN GROUP (ORDER BY handling\_days) AS p50\_handling\_days,

percentile\_cont(0.5) WITHIN GROUP (ORDER BY transit\_days) AS p50\_transit\_days

FROM v\_delivery\_facts\_plus

GROUP BY seller\_state, customer\_state

) t;

# **Task 3 — Find slow-handling sellers (who delays before shipping?)**

## **Problem**

Some delays happen **before** the package even reaches the courier. We need to see which sellers take too long to process/pack and hand off.

## **What this solves**

* Surfaces sellers with **high median handling time** so category/ops can **coach, set SLAs, or adjust exposure**.
* Keeps it fair with a **volume guard** (ignore tiny sellers).
* Also shows **on-time rate** and **median transit** so you can tell seller vs courier issues.

## **SQL — slow-handling sellers (ranked by median handling)**

WITH seller\_kpis AS (

SELECT seller\_id, COUNT(\*) AS delivered\_orders,

ROUND(AVG(is\_on\_time)::numeric, 3) AS on\_time\_rate,

percentile\_cont(0.5) WITHIN GROUP (ORDER BY handling\_days) AS p50\_handling\_days,

percentile\_cont(0.5) WITHIN GROUP (ORDER BY transit\_days) AS p50\_transit\_days

FROM v\_delivery\_facts\_plus

GROUP BY seller\_id

)

SELECT seller\_id, delivered\_orders, ROUND(on\_time\_rate, 3) AS on\_time\_rate,

ROUND(p50\_handling\_days::numeric, 2) AS p50\_handling\_days,

ROUND(p50\_transit\_days::numeric, 2) AS p50\_transit\_days

FROM seller\_kpis

WHERE delivered\_orders >= 200 -- fairness guard

ORDER BY p50\_handling\_days DESC, delivered\_orders DESC

LIMIT 20;

*Readout you can say:* “These sellers are slow **before ship** (median handling high). Let’s coach them first.”

# **Task 4 — Diagnose each lane: is lateness handling or transit?**

## **Problem**

Knowing a lane is “bad” isn’t enough. We must tell ops **what to fix** on that route: the seller’s **handling** process or the courier’s **transit**.

## **What this solves**

* For each lane, we label **late orders** as handling-driven or transit-driven by comparing each order’s times to that lane’s own **p75 thresholds** (top 25% is “unusually high”).
* Produces a lane table with **counts and %** by driver and a **primary\_driver** column.
* Gives ops a **clear instruction** per lane (coach sellers vs push 3PL).

## **SQL — attribute late orders by lane (p75 thresholds)**

-- 1) Lane-specific thresholds to define "unusually high"

WITH lane\_thresh AS (

SELECT seller\_state, customer\_state,

percentile\_cont(0.75) WITHIN GROUP (ORDER BY handling\_days) AS p75\_handling,

percentile\_cont(0.75) WITHIN GROUP (ORDER BY transit\_days) AS p75\_transit

FROM v\_delivery\_facts\_plus

GROUP BY seller\_state, customer\_state),

-- 2) Label each late order against its lane's thresholds

late\_labeled AS (

SELECT f.seller\_state, f.customer\_state,

CASE WHEN f.handling\_days IS NOT NULL AND f.handling\_days >= lt.p75\_handling THEN 1 ELSE 0 END AS handling\_flag,

CASE WHEN f.transit\_days IS NOT NULL AND f.transit\_days >= lt.p75\_transit THEN 1 ELSE 0 END AS transit\_flag

FROM v\_delivery\_facts\_plus f

JOIN lane\_thresh lt USING (seller\_state, customer\_state)

WHERE f.is\_on\_time = 0 )

-- 3) Lane diagnosis summary

SELECT seller\_state, Customer\_state, COUNT(\*) AS late\_orders, SUM(handling\_flag) AS late\_due\_handling, SUM(transit\_flag) AS late\_due\_transit,

ROUND(100.0 \* SUM(handling\_flag) / NULLIF(COUNT(\*),0), 1) AS handling\_share\_pct,

ROUND(100.0 \* SUM(transit\_flag) / NULLIF(COUNT(\*),0), 1) AS transit\_share\_pct,

CASE WHEN SUM(handling\_flag) > SUM(transit\_flag) THEN 'handling'

WHEN SUM(transit\_flag) > SUM(handling\_flag) THEN 'transit' ELSE 'mixed'

END AS primary\_driver

FROM late\_labeled

GROUP BY seller\_state, customer\_state

HAVING COUNT(\*) >= 100 -- only lanes with enough late orders

ORDER BY primary\_driver, late\_orders DESC;

*Readout you can say:* “Lane **SP→RJ** is **handling-driven** (62% of late orders due to handling). Lane **MG→BA** is **transit-driven** (70%).”

# **Task 5 — Hot spots: the worst seller × lane pairs**

## **Problem**

The pain is usually concentrated in a **few seller × lane combos** (a particular seller struggling on a specific route). We need that **action list**.

## **What this solves**

* Focuses attention on **where to call first tomorrow**.
* Uses **late\_rate** with a **volume guard** to avoid chasing noise.
* Adds median handling/transit so you can tell whether to coach the seller or push the courier.

## **SQL — top problem pairs (min volume, high late rate)**

WITH base AS ( SELECT seller\_id, seller\_state, customer\_state,

COUNT(\*) AS orders, AVG(1 - is\_on\_time)::numeric AS late\_rate,

percentile\_cont(0.5) WITHIN GROUP (ORDER BY handling\_days) AS p50\_handling\_days,

percentile\_cont(0.5) WITHIN GROUP (ORDER BY transit\_days) AS p50\_transit\_days

FROM v\_delivery\_facts\_plus

GROUP BY seller\_id, seller\_state, customer\_state )

SELECT seller\_id, seller\_state, customer\_state, orders,

ROUND(late\_rate, 3) AS late\_rate,

ROUND(p50\_handling\_days::numeric, 2) AS p50\_handling\_days,

ROUND(p50\_transit\_days::numeric, 2) AS p50\_transit\_days

FROM base

WHERE orders >= 50 -- enough traffic to matter

AND late\_rate >= 0.20 -- tweak threshold to your SLA

ORDER BY late\_rate DESC, orders DESC

LIMIT 20;

*Readout you can say:* “**Seller S123 on SP→BA** has **28%** late rate; median handling **2.6d** (vs transit **1.1d**) → seller-side fix.”