For Luís — the P1-0 design was reviewed against your data, and six things need your word

From: Miguel (GOTT.Sibyla) · **Date:** 2026-08-05 · **Read time:** ~10 minutes **Reports on:** the P1-0 review and corrections of 2026-08-04. **Supersedes:** `FDR-to-Sibyla-decisions-for-Luis.md` (2026-08-03). Read this one; the earlier note's closing line — "*nothing on the .NET side is waiting on a decision*" — did not survive the day.

Why you're getting a second note in two days

On 2026-08-04 the .NET side produced the **P1-0 design freeze**: six normative documents that map your FDR model field by field onto PostgreSQL, ~1,700 lines, 19 captured decisions. It was written by reading your prototype **read-only at a pinned SHA** (`9359c67`, re-confirmed at `6146004`). No financial or personal value was copied out of it, and nothing was written to it.

Then it was reviewed — **against your live `Editor/Data/*.json` and `Scripts/`, not against the freeze's own description of them.** That distinction is the whole story, because the two turned out to disagree in a lot of places.

You were meant to do that review. Three items in the freeze were assigned to you: verify the schema/key mapping against the prototype, receive the Discard/Purge design, and run a policy pre-flight. You were unavailable, and the .NET side could not sit still for a week, so an assisted review was commissioned to stand in for you. It is not a replacement and this note does not pretend otherwise — see *What was substituted, and what wasn't*, below.

The outcome: do not sign off yet.

Review result	Count
Blocking defects **proven on counted rows**	10
Blocking design / governance defects	9
Decisions frozen in name only	4
Checklist rows the freeze over-claimed about itself	5
Claims tested and verified clean	~40

All nineteen have a decided correction, written up as **C1–C19** in `docs/p1-0-corrections.md`. Six of them were decided *on your behalf* and are the reason for this note.

First, the good news, and it is the part that mattered most

Your identity work is correct and the .NET side is inheriting it unchanged. This was the expensive part, it was re-verified against real data, and all three closures stand:

Claim	Result
LGCode is a per-row identity	**Confirmed** — 1,475 rows, 1,475 distinct LGCodes, 1,475 distinct `(Filename, EntryCode)`, zero duplicate pairs. The pre-migration collision is fully repaired.
Bank `SourceKey` excludes `SourceFile`	**Confirmed** — present on 426/426 generated rows.
PAYCODE / RCVCODE derivation	**Confirmed**, and our company-scoping is a *safe superset* of your global key — zero cross-company FiscalDocumentID overlap today.
`ItemClass` is exactly three values	**Confirmed** — no fourth. The "Information" worry was a false alarm: that is a DOCRQE **Priority**; DOCEFL/DOCFLG use **ReviewPriority = Informational**. Three distinct fields.
The occurrence tiebreaker	**Confirmed load-bearing** — exactly one pair of byte-identical lines exists, same account and date. Without the tiebreaker those two merge and the pipeline *under*-generates.
~30 further parity checks	Clean — RECREV 21 columns in the exact order, DOCLOG 16, DOCEFL 20, BNKMOV 23, DOCFLG 19 with `PriorRelatedRecordID` correctly excluded, SNCACC parent chain fully resolvable, all 47 USEROBS review targets resolve.

What did not hold is the .NET mapping of your data, not your data. Ten defects would have failed on import — not "might fail", *would* fail, on rows we counted. Nine of the ten are our drafts describing your tables wrongly. Which is exactly what your review was supposed to catch, and why we are not signing anything until you have seen the six below.

A. The six that need your word

Each is decided and written into the corrections, so the .NET side is not blocked waiting. But each was decided **against pinned data rather than against your judgement**, and they stay reversible until you say otherwise. Short answers are fine — "yes", "no, because...", "re-measure after the next round".

A.1 · C1 — the seven aggregate payroll documents

Seven documents aggregate a whole pay run (component counts 2, 2, 2, 3, 3, 11, 15). Our mapping gave FDCHDR one scalar `SourceKey` with a foreign key to BNKMOV. Join test: **419 of 419 single movements resolve, 0 of 7 aggregates.** `SourceKey` works fine as an *identity* — deterministic, built from components in official movement order, which is what makes re-ingestion idempotent — and fails only as a *foreign key*. One column doing two jobs.

Our fix keeps your key and adds a junction table for the actual links, with `UNIQUE (Company, BMCode)` — a movement is consumed by at most one generated document. That holds on all 457 referenced BMCodes today.

> **Question:** is one-movement-one-document a property of the design, or of the data we happen to have? **Can a future pay-run shape break it?** If it can, the constraint is wrong and we want to know now rather than at import.

(Also: `SourceBMCode` appears zero times in either of our drafts. We are now naming it as the human-readable pointer derivable from the junction. Tell us if it is more than that.)

A.2 · C2 — ENTITM has two ingestion surfaces and we only saw one

`entity_products.json` stores **`CodeName`** and has no `EntityCode` field at all. `build_workbook.py` then resolves `codename_to_entitycode` and writes **`EC######`** into the rendered column. Our draft wrote a rule saying "import must reject an EC code" — which would **reject 100% of rendered rows**.

We now store `CodeName`, treat the rendered `EntityCode` as a derived export projection, and accept both surfaces on import, rejecting only an *unresolvable* value.

> **Question:** is that the right split — **the render authoritative for export parity, the JSON authoritative for storage**? We are assuming yes because P1-12 has to reproduce your sheets byte-comparably.

A.3 · C3 — grandfathering, and the column we were about to use

This is the highest-consequence single defect, and it is one we would not have caught without your data.

DOCEFL already carries **`EffectiveFrom`**, populated on all 45 rules (`2026-07-31` ×33, `08-01` ×8, `08-02` ×2, `08-03` ×2). Our draft invented a separate `EnforcementStartsAt` and left it unseeded — and the obvious repair, seed it from `EffectiveFrom`, **grandfathers nothing**: every one of the 29 open blocking instances was detected on or after its own rule's date. All 29 would block on day one. That is precisely the go-live queue D7 exists to prevent, and it would break our acceptance criterion that actively-blocking flags reproduce your zero.

The reason is benign: you wrote the rules in the same working week as the detections. `EffectiveFrom` records **when you wrote the rule in FDR** — it is real history and we are keeping it as provenance — but it is not an enforcement start date for a different system.

Our fix: `EnforcementStartsAt` is a Sibyla concept, seeded for all 45 rules to the Sibyla go-live timestamp. All 2,711 instances grandfathered, 0 actively blocking, matching your state. **Grandfathered does not mean hidden**: the 29 stay open, visible, audited and reviewable — they just do not gate execution.

> **Question: is any of those 29 something you intend to block on?** If one of them is a genuine stop-the-line condition rather than a housekeeping flag, tell us which, and it gets an earlier enforcement start of its own.

A.4 · C4 — we wrote your vocabularies from memory, and it showed

One hyphen. Every pinned value in `docrqe.json`, `recrev.json`, `flag_instances.json` and `flag_evaluation.json` is **`Non-Blocking`**; every CHECK constraint and blocking predicate in both our drafts said `NonBlocking`. **Every gating predicate would have been false against your data.** One character, total enforcement failure.

That turned out to be a pattern rather than an incident — five domains disagreed, two "boolean" columns hold conditional free text, and one whole state was missing. So the fix is procedural: `docs/p1-0-vocabularies.md` is now **generated from the pin by script**, is normative over every draft, and imports fail closed on an unlisted value.

Two things in it are yours to confirm:

> **Question 1: `DOCTYP.DocClass` is `Payables` (12 rows) and `Receivable` (7 rows)** — one plural, one singular. Typo, or deliberate? We are adopting your spelling verbatim to protect render parity, so if it is a typo we would rather fix it at source than translate it on import.
>
> **Question 2: `Waived` is on 861 DOCFLG rows — 32% of the table.** We are treating it as a terminal state in its own right, alongside `Resolved` and `Superseded`, with its own waiver evidence and authority. Folding it into `Resolved` would erase an audit distinction across a third of the table. **Is `Waived` semantically "closed without acting" — and who is entitled to set it?**

A.5 · C9 — DOCTYP coverage, and a scope boundary we may have drawn wrong

640 DOCLOG rows have no matching DOCTYP rule: NoDocMov/External 366, Financing/External 125, Duplicate/External 93, Payroll/External 56. On reading, most of that is a **scope boundary** rather than a gap — bank-generated rows are not classified documents — so our acceptance check is now scoped to captured documents (`Source <> 'BNK'`), and one real gap gets a rule.

> **Question:** is that the right boundary? Concretely — **should a `Duplicate/External` row have a DOCTYP rule of its own**, or is it correct that it never reaches classification?

A.6 · C12 — OFDGAP

`document_gaps.json` — 27 rows, `GPCode`, fed by `build_document_gaps.py` and by ENTMST's `InvoiceFrequency` / `ExpectedInvVal`. It appears **zero times** in either of our drafts, in a roster we reported as complete. We

are not importing it as a table: gap detection becomes what every other v5.0 detection is — a rule that emits into the flag and queue machinery with a declared `ItemClass`.

> **Question:** does OFDGAP carry state you need to *keep* (a gap someone has already looked at and dismissed), or is it purely derived output that can be recomputed? If it carries decisions, it needs a queue row, not a detector.

B. One ask, and it is nearly free

Ingest the 19 documents queued at the pin — 18 supplier invoices and the July bank statement sitting in `Inputs/` — and re-run the aggregate join test.

An overlapping statement regenerates P/F/O documents, which is **exactly the scenario your re-key was built to survive**. Running it gives us a free acceptance test of SourceKey anchoring on live data before we commit the .NET mapping — and it may add aggregate payroll documents, which is blocker A.1 above. If a new pay-run shape breaks one-movement-one-document, we would much rather learn it from your run than from a failed import.

What we would want back: all generated documents re-anchor, zero duplicates, all aggregates resolve, `UNIQUE (Company, BMCode)` holds.

C. The reconciliation percentage — a question, not a challenge

Our P1-11 acceptance criterion said "reproduce 94.6% ±0.1%". When we went to implement it, we found **no definition of numerator or denominator anywhere**, so we computed six plausible ones over BNKREC at the pin:

Definition	Result
matched / (matched + unmatched), by row	54.3%
(matched + internal) / all rows	58.3%
distinct movements with any match / BNKMOV	55.9%
excluding ground-truth matches	53.4%
by bank amount	47.8%
by row, excluding Internal	54.3%

None comes close. The 865 `Unmatched` rows are genuinely unresolved — each carries a `BMCode`, a non-zero amount, and your own "*Unresolved: no bank/document link yet*" flag.

We are not saying 94.6% is wrong. We are saying we could not reconstruct it from BNKREC, so we could not write an acceptance test against it. The .NET side has adopted an explicit definition — *distinct*

BNKMOV movements carrying at least one non-`Unmatched` match, over all movements, internal transfers counting as reconciled — and re-baselined to **55.9%**.

> **Question: what does your 94.6% measure?** If it is a narrower population — say, movements in scope for matching rather than all movements — then both numbers are right and we want them reconciled rather than one quietly replaced.

A warning we owe you either way: anyone who sees 55.9% next to your 94.6% will conclude the port collapsed. It did not; they measure different populations. Neither of us should quote either figure without its definition.

D. Two things that came *off* your plate

D.1 · The Roles and Responsibilities policy no longer needs an FDR-side pre-flight.

The 3 Aug note asked you to run Policy Change Validation before adopting it. We performed the pre-flight instead — all thirteen binding documents, one by one — and found two things.

First, the policy names five superseded documents and **conflicts materially with eight more it does not claim**, which leaves those conflicts unresolved rather than resolved. Second, the supersession authority was **added during adaptation**: the source draft says this should be a cross-cutting policy *above* four procedure families; the adapted text says *supersedes … where they conflict*, with no stated basis.

All eight conflicts share one cause — the policy asserts "*only authenticated deterministic .NET code executes*" and was then applied to a Python prototype that has no .NET in it. Under that sentence, your Code Change Governance becomes "the AI copying a script into live `Scripts/`", your Document Archiving Policy is entirely "the AI running `mkdir` and `mv`", and `apply_review_decisions.py` is an AI executing decisions.

Decision: the policy is Sibyla-scoped. Two systems, two legitimate role models — in Sibyla the AI proposes and .NET executes; in FDR the AI executes under your thirteen rules. **Your documents stand as they are.** That dissolves all eight conflicts without amending anything, and restores the "cross-cutting, above" reading the source draft intended.

D.2 · The `_to_delete/` contradiction is resolved.

Your Document Archiving Policy says a byte-identical re-appearance is swept to `_to_delete/` and **gets no new DOCLOG row**. Our Discard/Purge design said the same event **registers a new capture event**, auto-`Discarded`, with an Annotation finding. Two binding rules, one case, opposite outcomes — we had handed you a governance

answer that contradicted your own procedure. C5 settles it: register the capture event, not a DOCLOG row. The Discard/Purge design is still yours to take as the never-delete mechanism; it just no longer fights your archiving rule.

E. Figures we owe you a correction on

The 3 Aug note quoted your 19:30 numbers back at you. Three were wrong, and the errors were ours, not yours:

Figure	3 Aug note	Pinned at `9359c67`
DOCEFL rules	43	**45**
DOCFLG instances	2,574	**2,711** (Open 149 · Resolved 1,497 · Waived 861 · Superseded 204)
DOCRQE Decision-class	"164 real decisions of 180"	**64 Decision-class, of which 41 open**
FDCHDR	1,158	reported as both 1,158 and 1,153 — **needs one measurement**
Orphan DOCLOG	129	129 / 182 / 134 exist across sources — **three measurements of three different things; needs one definition**

All counts re-measure identically at `6146004`. They will all move once the 19 queued documents land, which is another reason to run them.

F. What was substituted, and what wasn't

Being straight about the limits of a review you did not perform:

- **Your structural verification was performed** — the schema and key mapping was checked against the pinned data, and the results are section A.
- **Your domain judgement was not, and could not be.** Whether a given prototype quirk is a defect to fix or behaviour to preserve is yours. Where our drafts and your data disagreed, **the review assumed your data is right and our draft is wrong.** That is the correct default and it is still an assumption — confirming it is what section A is asking.
- **One artefact was written to your prototype**, unintentionally and harmlessly: reading `entity_utils.py` produced `Scripts/__pycache__/entity_utils.cpython-310.pyc`. `__pycache__/` is gitignored, so it is not a repository change. No data file, script or spec was touched.
- **One earlier finding was withdrawn.** An automated pass reported our v14 flow diagram as missing. It exists; the reviewer was reading a partial listing. Mentioned only because it is the same class of error as the ones above, on our side this time.

G. Still yours, carried forward from 3 Aug

Unchanged, nothing on the .NET side depends on them:

- **Hydra iT — Fatura FV2502560.PDF** (2025-11-13) — capture or formally exclude. *(Roadmap 34)*
- **Doc_GBA_FA_48_2026_Signed.pdf** — differs from the archived copy at the same size, almost certainly the signed variant. Restore from `_to_delete` if the signed copy is the one to retain.
- **Diego Berlitz** — the remaining duplicate-identity CONFLICT.
- **SNCACC against LMD's chart of accounts**, and the 6 ITMMST rows with blank SNCACC. *(Roadmap 26)*
- **202601–202606 cross-check** against the accounting records. *(Roadmap 27)*
- **Toorist / Itoorer intercompany treatment** — shareholder loan, current account, or capital contribution. The ignore-for-matching rule is a holding position. *(Related: C15 now adds the RelatedParty funding classification on the .NET side, which our freeze had declared decided and had not actually built.)*
- **Missing reference data** — 75 providers without `InvoiceFrequency`, COCACC still one placeholder row, 27 FDCHDR rows at `FEX = NO RATE`.
- **Company dimension on capture** — still your call whether the FDR side needs its own copy of the `counterparty_not_internal_company` gate.
- **Is the 2026 Jan–Jun payroll rebuild stable enough to import?** — still open from 3 Aug, and now doubly relevant: those are the rows C1's junction table has to carry.

The short version

#	Item	Needs
A.1	Can a future pay-run break one-movement-one-document?	**Your word**
A.2	Render authoritative for export, JSON for storage?	**Your word**
A.3	Is any of the 29 blocking instances a genuine stop-the-line?	**Your word**
A.4	`Receivable` singular — typo or deliberate? `Waived` semantics and authority?	**Your word**
A.5	Should `Duplicate/External` have a DOCTYP rule?	**Your word**
A.6	Does OFDGAP carry decisions, or is it purely derived?	**Your word**
B	Ingest the 19 queued documents, re-run the join test	**Nearly free, high value**
C	What does 94.6% measure?	**Your word, not blocking**
D.1	Roles policy — Sibyla-scoped, your 13 documents stand	**Nothing needed**
D.2	`_to_delete/` contradiction resolved by C5	**Nothing needed**
E	FDCHDR and orphan-DOCLOG each need one definition	**Both of us**
G	Hydra iT, GBA signed variant, Diego Berlitz, SNCACC, 202601–06, intercompany, reference data	Yours / accountant / business

What is blocked on you: nothing, strictly. The corrections are applied so the .NET side can proceed. **What is reversible until you answer:** all six in section A — and A.1 and A.3 are the two where being wrong is expensive after a migration exists rather than before.

The full evidence is in `docs/p1-0-review-findings.md`; the decisions with their measurements are in `docs/p1-0-corrections.md`; the generated vocabulary annex is `docs/p1-0-vocabularies.md`. If you read only one,

read the corrections file — it is written so each entry stands alone.