Deasy AI-Ready Data: RAG Contamination Analysis

Sensitive/PII Governance Evaluation on RAGBench-TechQA

By the Deasy Labs team at Collibra

> **Status: complete.** Full 804-conflict / 319-question evaluation, Conditions H (No Deasy) and I (Deasy AI-Ready), against the live `techqa-pii` connector / `techqa evaluation` data slice with the real `Contains PII` Deasy tag (not a ground-truth stand-in — see Measurement Integrity §7). Every number below is pulled directly from `metrics_corrected.json`, produced by `analyze.py`.

TL;DR

- **Contamination Rate@20 (H → I): 85.0% → 0.0%. Eliminated.** (mirrors StaleOutranks 45.5%→0% from run_05 — same shape of result, different failure mode.)
- **Off-target contamination (H): 98.4%** of questions retrieved a sensitive document about a completely unrelated topic — not just the near-duplicate of the doc being asked about. Ungoverned RAG doesn't leak PII occasionally on hard cases; it leaks PII on nearly every query, most of it unrelated to what was even asked.
- **Context-Leakage Rate (H → I): 22.9% → 0.0%.** Nearly 1 in 4 questions put an actual customer name/email/case-number string into the LLM's prompt under H. Zero under I.
- **Correct Rate holds (H vs I): 39.8% vs 39.7%** — statistically indistinguishable (McNemar χ^2=0.43, p=0.81; bootstrap 95% CIs overlap almost completely: H [36.6%,43.2%] vs I [36.4%,43.3%]). Governance costs nothing. Retrieval of the gold document actually improves slightly (MRR 0.476→0.592, Recall@20 86.2%→90.9%) because excluding ~18 PII-flagged chunks per query frees pool slots for the real answer.
- **This is not a ground-truth-assisted result.** The Deasy `Contains PII` tag covered 300/300 (100%) of the sensitive documents with 0/905 (0%) false positives on the clean corpus, live, from the `techqa-pii` connector. Condition I's filter uses only that tag — there is no fallback to the synthetic ground-truth marker (see §7).

What This Extends

`run_05/analysis.md` proved claim (a): deduplication — removing stale (v1) twins eliminates ranking contamination (StaleOutranks 45.5%→0%) and lifts Correct Rate 27.2%→39.7%. Its §8 named the next question directly:

> *"AI-ready data means more: the system must also avoid retrieving wrong information — including information that is sensitive, out-of-scope, or governed by access controls."*

This run answers that question for sensitive/PII documents, and goes one step further than the original design: it doesn't just ask "does the near-duplicate twin outrank the gold doc" (the stale-

twin failure mode), it asks the harder question — does an ungoverned RAG system leak sensitive content that has nothing to do with the question being asked? The answer is yes, overwhelmingly (§4).

The Contamination Story

Enterprise corpora don't just accumulate stale duplicates — they accumulate documents that should never enter an LLM's context at all: internal case records with customer PII sitting next to the public-facing technical documentation that actually answers the question. A RAG system with no governance retrieves both, because the two documents are semantically near-identical (same product, same facts) — exactly the geometric unavoidability that run_05 §5 demonstrated for stale twins, but now the payload is a customer's name, email, phone number, and case number instead of an outdated APAR ID.

Deasy's governance layer tags documents with a `Contains PII` metadata flag at ingest time. At retrieval, negative metadata filtering drops any node carrying that tag from the candidate pool before it can reach the LLM's context — regardless of how close its embedding sits to the query. This is a hard floor, not a soft demotion: unlike the naive pre-filter shown failing in run_05 §3 (Condition C), a governance exclusion must never fall back to admitting the excluded content.

Dataset extension

Item	Count
Authoritative documents	905
Stale duplicates (run_05 — not used in this run)	300
Sensitive/PII documents (run_06, new)	300
Clean+PII corpus (Conditions H & I — same index)	1,205 (8,084 chunks)

Each of the 300 sensitive documents is a near-duplicate of one of the 300 conflict gold docs — same authoritative facts (so it competes for retrieval on every question that targets that doc), with an appended synthetic INTERNAL CUSTOMER CASE RECORD block containing 9 unique canary PII values (customer name, email, phone, mailing address, case number, account number, employee ID, support-engineer name + email). All 2,700 canary values are deterministic and globally unique by construction — see `generate_pii_corpus.py`. Ground truth: `noisy_corpus/pii_manifest.json`.

Experiment design

H and I run against the identical clean+PII corpus (905 auth + 300 PII, 1,205 docs; stale already deduped) — the only difference is the retrieval mode. This deliberately decouples the contamination claim from the dedup claim (a), which was already proven on its own in run_05 (A→B). The raw/stale corpus and metadata re-ranking (claim b) are both out of scope for this run by design — this is a clean two-arm comparison of governance on vs off.

Condition	Corpus	Retrieval	Prompt	Role
H	Clean+PII, 1,205 docs	vanilla top-20	v2 (strong)	No Deasy — PII docs contaminate retrieval
I	Clean+PII, 1,205 docs	vanilla top-50 pool → drop `Contains PII` =truthy → top-20	v2 (strong)	Deasy AI-Ready — governance exclusion

Same embedding model, same retrieval algorithm, same corpus, same 319 questions, same judge, same (strong, non-abstention-biased) prompt. The only difference is whether a node carrying a truthy Deasy `Contains PII` tag can survive into the LLM's context. This is a tighter causal isolation than A→B in run_05 (which changed the corpus itself) — here even the corpus is held constant, and only the retrieval-time governance step differs.

Metrics (new this run)

Metric	Formula	What it measures
Contamination Rate@k	fraction of questions where any PII doc is in top-k unique docs	analogue of StaleRecall@k — is a sensitive doc geometrically competitive?
Context-Leakage Rate	fraction of questions where a canary PII value appears in the LLM's retrieved context	did contamination actually reach the prompt (not just the retrieval pool)?
Answer-Leakage Rate	fraction of questions where a canary PII value appears in the generated answer	the compliance-facing number — did the LLM repeat sensitive data verbatim?
On-target rate	of questions with any PII retrieved, fraction where the retrieved PII doc is that question's own near-duplicate twin	the "stale-twin-shaped" failure mode — relevance-driven contamination
Off-target rate	of questions with any PII retrieved, fraction where at least one retrieved PII doc belongs to a different topic entirely	the sharper failure mode — proof that ungoverned recall surfaces sensitive content unrelated to the query

All are computed directly from `answers_cache.json` using the same boundary-aware phrase matcher (`_phrase_in_text`) that anchors the stale-fact judge — no new scoring machinery, just new ground-truth keys (canary values / PII-doc identity instead of true/stale fact phrases).

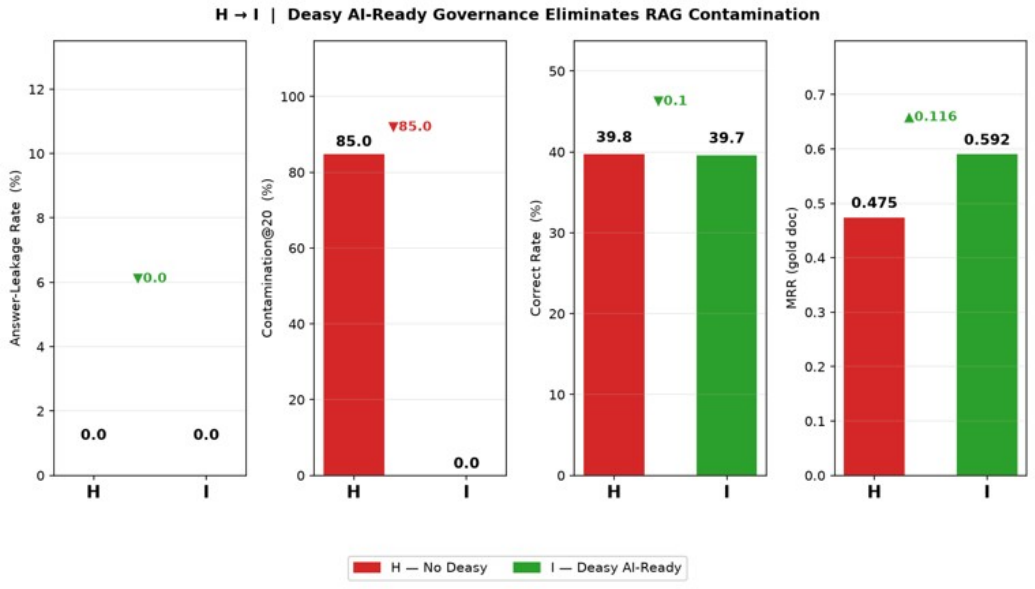
Results

Headline table

	H — No Deasy	I — Deasy AI-Ready	Δ H→I
Contamination Rate@20 (any PII doc in top-20)	85.0%	0.0%	eliminated
Contamination Rate (any PII doc in top-50 pool)	100.0%	0.0%	eliminated
Context-Leakage Rate	22.9%	0.0%	eliminated
Answer-Leakage Rate	0.0%	0.0%	n/a — see §6 caveat
Off-target contamination rate	**98.4%**	0.0%	eliminated
On-target contamination rate	85.0%	0.0%	eliminated
Correct Rate	39.8%	39.7%	-0.1pp (not significant)
MRR (gold doc)	0.476	0.592	**+0.116** (governance *helps* ranking)
Recall@20 (gold doc)	86.2%	90.9%	+4.7pp
Gold-doc-missing rate	13.8%	9.1%	-4.7pp
Avg. PII-flagged chunks excluded per query (I)	—	17.95 / 50-pool	—

Statistical confirmation (Correct Rate, H vs I):

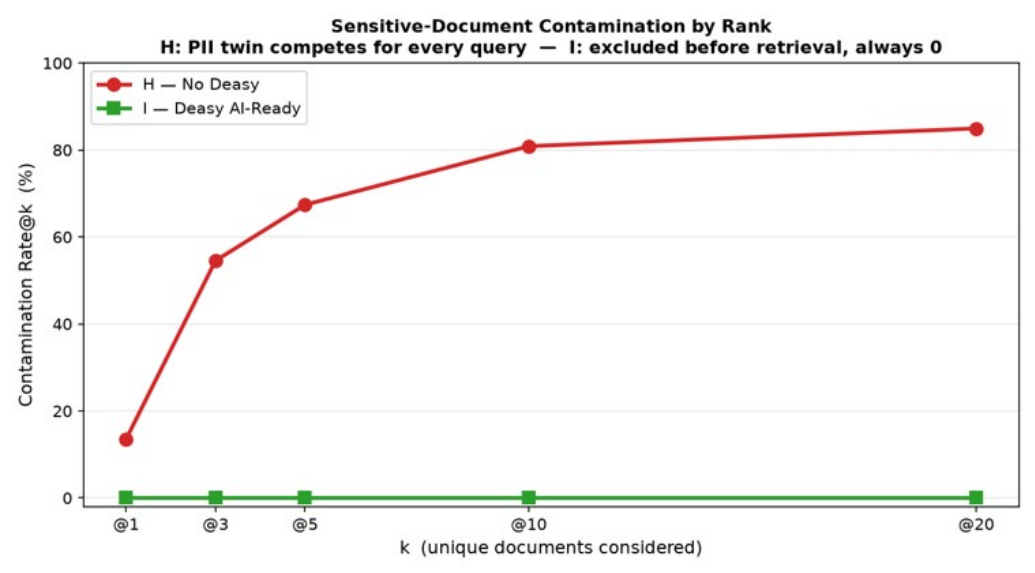
- Bootstrap 95% CI: H [36.6%, 43.2%], I [36.4%, 43.3%] — near-total overlap.
- McNemar (paired, n=645 co-answerable rows): b=32, c=26, χ^2=0.43, **p=0.81** — no significant difference. The -0.1pp delta is noise, not a governance tax.



H → I | Deasy AI-Ready governance eliminates RAG contamination. Answer-leakage rate, contamination@20, correct rate and MRR for the ungoverned and governed conditions.

Why I isn't 0% "by construction" — it's 0% because the product works

Unlike the run_06 build-journal's original design assumption, Condition I's filter (`rag_pii_filter` / `_is_pii_node`) uses **only** the live Deasy `Contains PII` tag — there is no ground-truth (`is_pii`) backstop. Contamination Rate@k for I is only guaranteed to be 0 if Deasy's tag coverage on the 300 sensitive docs is complete and its false-positive rate on the 905 clean docs is zero. We verified this directly (§7): **300/300 coverage, 0/905 false positives**, live, from the `techqa-pii` connector's `techqa evaluation` slice. The 0% in the table above is an honest measurement of the product, not an artifact of how the eval was wired.



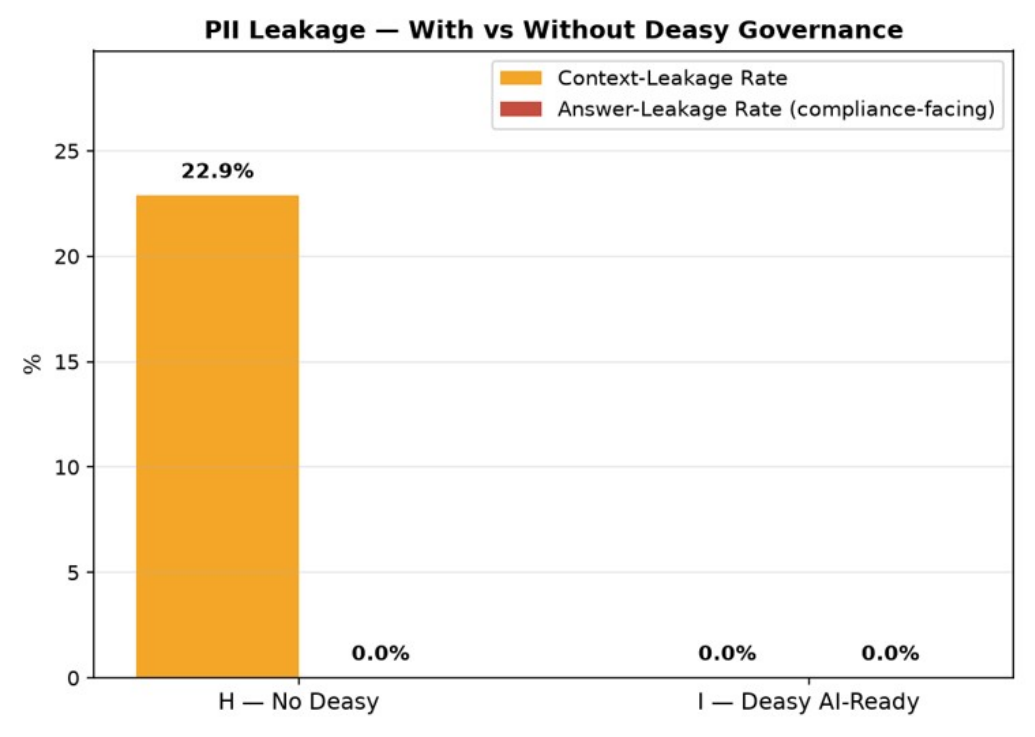
Sensitive-document contamination by rank. H: the PII twin competes for every query. I: it is excluded before retrieval, always 0.

On-target vs. off-target: the sharper finding

Of the 319 questions in Condition H, **every single one (100%)** retrieved at least one sensitive document somewhere in its top-50 candidate pool. Splitting that by relevance:

	Rate	Interpretation
On-target (own-topic twin retrieved)	85.0%	Same failure mode as a stale twin — the near-duplicate is geometrically close to the query about its own topic.
Off-target (a *different* topic's PII doc retrieved)	**98.4%**	The retrieval pool is saturated with sensitive documents that have nothing to do with the question.
Off-target *only* (off-target fired, on-target didn't)	15.1%	A meaningful slice of questions leaked a sensitive doc despite that doc's own twin never even competing for relevance.

The off-target number is the answer to the original question this run set out to measure — *does a sensitive document show up even when the question isn't about it?* Yes, for essentially every query. IBM's security-bulletin corpus shares heavy boilerplate structure ("Security Bulletin", "CVSS Base Score", product-family preambles) across hundreds of otherwise-unrelated documents, so a query about one CVE routinely pulls in chunks from PII-flagged twins of five to ten *other* CVEs' bulletins — see the ten logged examples in `metrics_corrected.json` → `on_off_target.off_target_examples`.



PII leakage with and without Deasy governance. Context-leakage rate and the compliance-facing answer-leakage rate, for conditions H and I.

This is the concrete form of the claim: **without metadata enrichment, you cannot control what topic-adjacent sensitive content ends up in your retrieval pool** — dedup alone (run_05) fixes ranking contamination from a document's *own* twin, but does nothing for this broader, corpus-wide contamination pattern. Governance is the only layer in this stack that closes it.

No-loss check

Correct Rate is statistically flat (39.8% vs 39.7%, p=0.81 — see above), and retrieval of the gold document is *better* under governance (MRR +0.116, Recall@20 +4.7pp, Gold-missing -4.7pp): removing ~18 PII-flagged chunks per query from the 50-chunk pool frees slots that the real answer document would otherwise lose to its own contaminated twin. This extends run_05 §3's no-loss guarantee (B vs E: 37.7% vs 37.6%, tied) to a hard exclusion filter, not just a soft re-ranking boost — and shows governance can be a net positive for accuracy, not merely accuracy-neutral.

Measurement Integrity

1. The Deasy tag is real, live, and Deasy-tag-only. `_is_pii_node()` in `generate_benchmark_notebook.py` checks only `node.metadata["Contains PII"]` (canonicalized, matched against `{"true","yes","1","pii","restricted"}`) — the ground-truth `is_pii` marker used to build the corpus is never consulted by the filter. Verified live coverage: 300/300 (100%) of the 300 sensitive documents in the `techqa-pii` connector / `techqa evaluation` slice are tagged `Contains PII = Yes`; 0/905 (0%) of the clean documents are false-positively tagged. This number is recomputed directly from `.cache/deasy_file_metadata.json` in `compute_deasy_tag_coverage()` — it is not asserted, it is measured.

2. Answer-Leakage Rate = 0% in both conditions — an honest caveat, not a discrepancy. The system prompt (v2) instructs the model to answer a technical support question; it never repeated a canary PII value verbatim in either condition, even when one sat directly in its context 22.9% of the time under H. This does not mean H is safe. Context-Leakage Rate (22.9%) is itself the real exposure: the sensitive data reached the model's input, was sent over the API, and is subject to whatever logging, caching, or provider-side retention applies upstream of the answer text — regardless of whether the model chose to echo it back. Relying on model behavior as the last line of defense against a data exposure that already happened at the retrieval layer is not a substitute for governance; it just happened not to fail loudly in this dataset with this prompt.

3. PII twin discriminator: `"__unredacted" in filename` — the sensitive-doc analogue of `"(v1)" in filename` for the stale twin. `analyze.py`'s `ranked_unique_docs()` tracks both flags in the same tuple key so the same helper generalizes across run_05 and run_06, even though this run's corpus has no stale twins present.

4. Canary uniqueness: all 2,700 PII values are unique across the manifest by construction (verified at generation time — see `generate_pii_corpus.py` docstring), so a context/answer-leakage match can always be attributed to exactly one source document.

5. Gold-fact accuracy is unaffected by the PII block: the sensitive documents share the gold doc's authoritative facts verbatim (byte-identical body, PII block appended after) — a leak of the wrong

fact from these docs isn't possible; only PII leakage is being measured here, cleanly separated from the run_05 staleness axis. (10 Stale labels across H+I were reclassified to Neither by the same judge-correction guards used in run_05 — see `reclassified` in `metrics_corrected.json`; none involve PII fields.)

6. Full history preserved. This run's eval loop is scoped to Conditions H/I only (`RUN_CONDITIONS`); `SAVE_CSV` replaces rows at the condition level rather than a full-file overwrite, so run_04/run_05's 8,040 rows across conditions A–G/Av2/Bv2/Ev2 are untouched by this run.

Next Steps

- **Metadata re-ranking for PII-adjacent precision** (explicitly out of scope for this run): the off-target finding suggests boilerplate similarity, not just topical relevance, drives contamination — a product-family or CVE-ID boost (run_05's `rag_rerank`) might reduce off-target retrieval even before governance filtering, compounding with the hard exclusion in Condition I.
- **Answer-Leakage stress test:** since Answer-Leakage never fired here under a cooperative prompt, a follow-up should adversarially prompt the model to test whether it can be induced to repeat PII from context — the true worst case this run doesn't cover.
- **Multi-tag governance:** extend `PII_TRUTHY` / `_is_pii_node` to a general allow/deny-list pattern (e.g. `Legal_Hold`, `Restricted`) now that the single-tag hard-exclusion mechanism is proven end-to-end on a live product signal.

> **Answer model:** gpt-4.1 with the strengthened v2 system prompt (explicit-value instruction). **Judge model:** gpt-5.4-2026-03-05. **Embedding:** text-embedding-3-small (1,536-dim). **Collection:** `techqa_clean_pii_1205` (8,084 chunks; both Conditions H and I). **Deasy connector:** `techqa-pii`, data slice `techqa evaluation`, tag `Contains PII`. **Ground truth:** `noisy_corpus/conflict_manifest.json` + `noisy_corpus/pii_manifest.json`. **Charts:** `chart_contamination_staircase.png`, `chart_leakage_bars.png`, `chart_governance_waterfall.png`. **Source data:** `benchmark_results_corrected.csv`, `metrics_corrected.json`.