

OPINION AND ANALYSIS

Editorial date: November 1, 2026

A Circular Issued Yesterday and a Model Living Last Year

> A sentence can be true, its link official, and its text quoted without error, while the whole answer is wrong—not because of what the system said, but because of the day it thought it was talking about.

A Message From September to February

On September 24, 2025, the Saudi Central Bank issued Circular 472021766. Its subject was resolutely technical: updated trade-repository reporting and risk-mitigation requirements for over-the-counter derivatives. The circular concerns the quality of data reported by banks and, among its updates, adopts the Unique Product Identifier, which names financial instruments consistently in reporting records. It expressly refers to earlier requirements connected with Circular 42056371 and replaces them. All of this is documented on the central bank's rulebook site, without need for a fictional witness.1

The decisive line describes neither the financial product nor its spreadsheet. It names **February 1, 2026** as the start date for applying the new requirements. A competent authority issued a text in September, and it was available before year-end, but its presence in the archive did not make it the rule for every event in October. Not everything born in September applies in September. This simple fact becomes difficult inside a system that equates "newest document" with "applicable authority."12

The detailed requirements add a harder surprise. From the effective date, new transactions are not accepted into the trade repository without an ISIN or UPI, as available. For transactions executed **before** application, reporting banks must update existing portfolio data through backloading by **December 31, 2026**. The transaction predates the new rule, while the duty to complete its data falls in the year the rule applies. This does not turn the past into a retrospective violation; it brings an old record into a new reporting obligation on the terms stated by the instructions.3

Suppose a compliance employee asks a retrieval system, “which identifier belongs in this report?” The assistant can answer precisely, quote the new requirements verbatim, and attach a reassuring official link. Unless it asks the report date, the contract’s execution date, and whether the request concerns a new transaction or backloading an existing portfolio, it has compressed three questions into one answer. The answer may look elegant on screen and still be unfit for operation.

We will not invent a bank that breached the rules, a dismissed employee, or a fine. The documented case is the text and its dates. The assistant dialogue is a proposed analytical exercise, not an observed event. That is enough to expose the potential defect: **legal text is not static information; its conditions of application determine when an answer drawn from it is true**.

When Faithful Quotation Becomes a Form of Error

The spread of retrieval-augmented generation taught sensible checks: did the system find the right document, support its answer with a passage, and attribute the source without invention? These are necessary, but they contain a hidden assumption: finding relevant text is a sufficient beginning for deciding what may be applied. In changing documents, it is incomplete.

For a medical definition that stays fixed, the latest reference may be the best start. For “did this disclosure duty exist when the transaction was executed?”, newest is not always best and oldest is not always wrong. The right version establishes a relationship among rule, event, date, authority, class, and exception. Correctness depends on the hierarchy and status of texts, not simply file order by last-modified date.

A system may retrieve two highly similar texts: one requires reporting within a stated period and another changes it; one exempts a company class and another narrows the exemption; a third preserves a transitional rule for entities existing before amendment. Semantic similarity may rank the temporally wrong text higher because the sentences concern the same procedure and use nearly identical terms. Updating a knowledge base can worsen the problem if old text is deleted as obsolete, though a later audit of a pre-amendment decision will need it.

The model need not hallucinate. It can be perfectly faithful to its source and incomplete in matching facts. This paradox affects how institutional knowledge systems are evaluated: they can pass **citation correctness** and fail **applicability correctness**. The difference is engineering and law, not style.

Four Dates in One Document

The picture becomes clearer when we separate clocks often mixed by search interfaces:

1. **Issue date:** when did the authority issue the document or adopt the decision?
2. **Publication or availability date:** when did it become officially published or available through the recognized process?
3. **Effective date:** when does the binding rule begin, accounting for suspension or phased application?
4. **Event time and applicable period:** when did the questioned act occur, and to which transactions, classes, or periods does the rule extend?

These are not merely columns. They may coincide in a simple system or diverge by months and years. Institutions add a fifth date: **when did the system or employee actually receive the information?** If an amendment is published today and enters the knowledge repository next week, the institution has two records: what the official source says was applicable, and what its systems held when the decision was made. The questions do not automatically coincide, and ignorance does not itself create a legal exemption, but it matters when investigating cause.

Temporal-database theory has long distinguished **valid time** in the represented world from **transaction or system time** in the database. The first asks when a description was true in the reality modeled; the second asks when the database carried it. Keeping both creates bitemporal data capable of reconstructing what the legal record says was true and what the database knew at a past point. IBM documented the idea in temporal-data-management literature years before language models entered offices.[4]

Transferring it to law requires care. A rule's "valid time" is not a raw date whose meaning a computer decides; it may arise from an original text, regulation, transition, judicial order, and territorial jurisdiction. A database can record the conclusion, source, and objections, but does not create a binding legal interpretation. Design must separate **established fact**, **interpreted applicability rule**, and **executive decision**.

From an Old Regime to a New One, Without Treating Everyone Alike

A second Saudi example is less specialized than derivatives and reveals what date-sorting misses. The Council of Ministers approved the new Commercial Register Law on **September 17, 2024**. It was issued on **September 22**, published in the Official Gazette on **October 4, 2024**, and specified application 180 days after publication. The Ministry of Commerce announced actual entry into force on **April 3, 2025**.[56]

The law changed familiar details: one establishment register across the Kingdom instead of issuing new branch registers, and annual confirmation of register data instead of the former renewal concept. Yet the decree gave holders of existing branch registers **five years from the effective date to regularize their position**. A rule against new branch registers does not allow an automated system to declare every old branch register expired or unlawfully existing on day one. "No new issuance" differs from "treatment of an existing case under a transition period."[56]

Imagine an institution using an internal assistant to assess a supplier in April 2025. An employee asks, "the company has an old branch register; should the file be rejected automatically?" If the system extracts the single-register provision and ignores the transition, it may sound decisive where decision is not. If it retrieves the pre-2025 version and says branch registers continue to be issued as before, it errs in the opposite direction. The question requires distinguishing an **existing pre-effective-date register** from **an application for a new branch register**, while checking the rest of the law, regulation, and decision-maker's requirements. This is not a report of a real rejection but an engineering consequence possible from two official published provisions.

"Effective date" alone is no magic key. A general regime may enter force while containing a rule extending to earlier classes. If the institution stores the whole document as valid from April 3, 2025 without decomposing transitional provisions into conditions, it owns the correct date and produces the wrong conclusion. *The unit of understanding is not the file, but a specified provision with scope, class, and time.*

The Rule Officially Published but Never Effective

If the Saudi examples separate issue and application, the United States supplies a harder case: an authentic final document has an announced effective date, then a court prevents its application before that date arrives.

In April 2024, the US Federal Trade Commission announced a federal rule restricting worker noncompete agreements. The final rule was published in the *Federal Register* on **May 7, 2024**, with a planned effective date of **September 4, 2024**. A language model can find the full government text, including detailed provisions for existing and new contracts, and offer a coherent, flawlessly copied report.7

On **August 20, 2024**, a court order stopped enforcement before the planned date. On **September 5, 2025**, the commission took steps to dismiss its appeal. The FTC's official page states directly that the rule **is not in effect and is not enforceable**. That does not yield one conclusion for every US noncompete agreement; state laws and other existing rules are separate.8

Adding an "effective date" field to a PDF does not fix this. Retrieval can read September 4 and assume application began, although an earlier judicial event changed the status. Asked what applies today, it may quote the original text from the most official drafting source and miss the regulatory-status page. **Published text is evidence of what an authority decided at one moment; alone, it is not evidence that the decision became enforceable at every later moment.**

In the first Saudi example, the system had to wait for a future date. In the second, it had to respect a transition inside an effective law. In the US case, it had to understand that the scheduled date never took effect. Three forms of legal time and three ways a non-lying model can be wrong.

What Does a Computer Know When It Knows a Date?

Bitemporal databases help make this precise. Imagine a record containing document and provision identifiers, issuing authority, scope, effective start and end, the date knowledge entered the repository, the date it was replaced or corrected there, documented status, and the original link. One version may contain clauses effective on different dates. Storing one file and one date makes reconstruction impossible without additional interpretation.

For "what was required on January 31, 2026?", we seek not only what the database held when asked, but the provision applicable **to the event type** that day. For "what did our employee rely on on January 31?", we need a second archive view: what the system actually displayed. Legal evaluation of a decision and reconstruction of the decision-maker's information must not be confused. They can diverge through update delay, incomplete indexing, copying error, or a later ruling affecting legal status.

Let a provision's start be b and end n. A preliminary engine might test whether a transaction date lies in the half-open interval **[b, n)**, including the start and excluding the end. This is a practical computing convention to avoid boundary collision, not a general legal rule for

counting days or deadlines. Law may specify calendars, business days, exceptions, and precise times. Time fields need the correct zone, calendar, and day-or-hour precision.

Not every provision active in the same period applies to every person. Predicates must describe entity type, formation date, party capacity, location, contract kind, event execution date, reporting date, exemptions, and authority. For commercial registers, “is this a branch register?” is insufficient without “did it exist before effectiveness?” For derivatives, “when was the contract made?” is insufficient without “is this a new report or backloading an existing portfolio?”

If a later correction arrives, history must not be erased. Keep what appeared and what was corrected, explicitly distinguishing “knowledge record replaced on this date” from “legal effect changed from another date.” This is the heart of bitemporal design. It can answer what provision we now attribute to an old event and what belief our systems held when making the earlier recommendation—questions a current-row database cannot answer.[49]

What Did Temporal Retrieval Researchers Actually Do?

In October 2026, days before this article was prepared, **TimelyRAG** appeared in *Findings of EMNLP 2026*. It studied a directly related problem: regulatory documents, policies, and terms of service evolving through amendments that leave much content unchanged and alter limited sections. The researchers called this an environment of **overlapping-evolving documents**. A wrong candidate is not topically remote from the right one, but often so similar that semantic search judges it excellent.[10]

They built **TimelyQABench**, covering amendments, exceptions, validity periods, and event-based conditions. It is a **controlled synthetic benchmark**, not real litigation or Saudi bank decisions. A language model generated Korean questions in four domains: legal regulations, university rules, corporate policies, and terms of service; verification used language models and limited human review. The researchers stated the limitation.[10][11]

Their method reranks results by semantic and temporal relevance, considering whether a question names a date. Under the paper’s tested comparisons, improvement reached **28.6 percent on nDCG@10** in the maximum reported cases. This is a **relative improvement in retrieval ranking quality on a research benchmark**, not a 28.6-percentage-point increase in legal correctness or proof that the system can issue regulatory opinions or legal advice. The paper says the method depends on initial retrieval of candidate versions and does not solve every multi-event logical complexity.[10][11]

The paper isolates a problem distinct from answering Arabic well or finding relevant text: can an engine rank two very similar provisions when their validity differs for the time in the question? When the answer belongs in law rather than a laboratory, scope, status, and source

authority must still be interpreted and the conclusion checked by specialists. This distinguishes **retrieving applicable evidence** from **determining the governing rule**.

Earlier LegalBench-RAG research studied precise retrieval of supporting legal passages, while another benchmark focused on legal retrieval requiring reasoning. Both help a system locate relevant material, but neither removes the need to test whether it was effective on the event date. Each metric captures one slice of correctness. Mixing slices into one score is how an incomplete system acquires the appearance of complete confidence.[12][13]

An Operating Architecture That Starts With the Question

If building a system for a bank or public body to review regulatory procedure, I would not begin with a huge language model asked to "remember" updates. I would turn the open question into an **inspectable applicability case**. Depending on domain, it needs event date, date of the decision under review, entity or contract type, jurisdiction, requested action, and whether the user asks about current status or a past act's validity. If a decisive element is missing, the model asks for it rather than guessing politely.

Four layers follow, each with a distinct function:

Layer one, verified source registry: preserved originals of laws, circulars, regulations, and rulings affecting applicability, with links, digital identity, authority, issue and publication dates, and retrieval logs. A file retains its original status; tomorrow's page update does not become a new "issue date." Source changes are reviewed under defined rules, and previous versions retained.

Layer two, provisions and transitions registry: source-linked passages stating commencement, expiry, amendment, exception, delay, suspension, phase, and class-specific application. A "version 1 then version 2" chain is insufficient: one clause may be repealed while another remains; a court may suspend a provision without rewriting the original; an amendment may be regional or sectoral. Relations among provisions, not merely file order, must be represented.

Layer three, case-linked retrieval: where confirmed data permit, eliminate inapplicable candidates; rank remaining passages by subject and time; actively seek transition rules, exceptions, stays, and later interpretations with effect. An "old" passage may explain former status or still govern a class. Aggressive exclusion can erase the evidence needed to prove history.

Layer four, constrained and reviewable execution: tested, explicit policy-as-code handles operational duties that can be stated clearly; the model proposes text-supported reasoning rather than authority to create it. Conflicting sources or specialist interpretation go to legal or

compliance review. Programming cannot settle every interpretive dispute, but can expose assumptions to challenge.

These layers require **separating the answer environment from execution**. An assistant may explain why an ISIN or UPI field is requested, but changing portfolio data, closing an application, or refusing a registration requires authority, investigation, and institutional conditions. The point is not prolonged philosophizing before every click. It is a clear boundary between preliminary explanation and decisions affecting rights or money.

Applicability Proof: What Will We Show an Auditor Two Years Later?

I propose an editorial and engineering device called a **temporal applicability proof**. It is not a published international standard or an existing product claimed to be in use. It is a testable design: a compact trace attached to a regulatory answer, proving not only **where the sentence came from**, but **why the text applied to this person on this date**.

It has seven parts: event identity and dates; source authority, version link, and passage; provision-validity boundaries; class, jurisdiction, and exception conditions; relation to provisions it superseded, suspended, or amended; when the knowledge existed in the institution's database; and limits of confidence, interpretation, and review. It records an unambiguous policy-rule version identifier and a signature or content hash preserving evidence integrity. The "proof" is not a seal declaring correctness by definition, but an evidence package an auditor can reject or correct.

Audit then changes. Instead of asking an employee two years later why the assistant said something, the institution asks separately: **what regulatory evidence made the answer applicable at event time?** and **what knowledge version was before the employee at decision time?** If a previously omitted official document is later found, specified provisions can be re-examined without retrying the whole archive from employee memory.

The danger is implying that measurement settles everything. A proof may store a correct date and miss an exception. Application to a party may itself be disputed. Absence of conflict in the database is not proof of absence in law. After every successful test, "insufficient evidence" must remain available. That is not AI failure but an honest description of the boundary of automated inference.

The Test Laboratory: Five Ways a Non-Hallucinating System Can Fail

There is little value in launching a regulatory assistant on easy questions and announcing accuracy above ninety percent. Prefer cases where **wrong answers are highly plausible** and their texts genuinely exist in the archive. Specialists build test cases with stated facts, dates, and reasoned reference outcomes, then systems are compared on a held-out set:

1. **Issued before effectiveness:** a circular appears months earlier, then a question concerns an act before application. Does the engine distinguish knowledge of change from an obligation to follow it?
2. **Repealed version formerly effective:** review of an old decision. Does the system find the former rule while preventing its use for a new decision?
3. **Effectiveness that never occurred:** final text followed by a judicial stay. Does it search rule status, not only the text? The FTC case is a reference pattern.
4. **Transition for existing classes:** a new law stops a register type while giving existing ones a regularization period. Does it distinguish an entity formed before effectiveness from a later application?
5. **A new duty concerning records of old events:** reporting updates for an existing derivatives portfolio. Does it distinguish transaction execution from the date reporting data must be amended?

One proposed measure is **temporal applicability accuracy**: the share of cases selecting the correct provision by effective period and scope against documented, reviewed reference conclusions, with abstention for missing decisive data counted separately. A system asking for a missing date should not be punished as though it invented one, nor rewarded for refusing every question. Also measure answer coverage, correctness of clarification requests, citation accuracy for the effective passage, confidently wrong answer rate, and lag from official publication to production availability.

The benchmark is invalid if it uses only model-generated questions and calls the result proof of safe banking procedure. It needs framed human samples, boundary cases on transition days, intersecting circulars, independent interpretation review, and retrospective tests across dates. Where real data are confidential, analogous cases can be built from public rules without treating them as evidence of live operational performance. TimelyRAG’s published figures are a useful research start, not a ready compliance score for Saudi management.

The Objection We Should Take Seriously

An experienced technology director might say: we have few circulars, monthly legal review, and versioned document management. Why temporal layers, provision management, and an exception engine? In many uses they are partly right. If an assistant only explains non-consequential terms, documents are stable, and a team updates them continually, an updated official source, good index, visible review date, and warning against using the text alone for decisions may suffice.

Complexity is not virtue. A badly implemented bitemporal registry can create false confidence and uncompensated operating cost. Preserving every change is insufficient when specialists cannot interpret or connect it to the right class. Architecture should scale with possible harm, changing-source count, decision frequency, and the need to defend past decisions.

The objection weakens at one point: **when the answer enters an execution chain with financial, regulatory, or human effect, and the institution may later have to explain why it applied this rule rather than another**. A digital archive then ceases to resemble a reading repository and becomes decision-process memory. Memory without clocks cannot reconstruct what happened even if it keeps every file.

AI advocates cannot answer by inflating model capability. Models may read long text skillfully, but cannot know whether every court order, supplementary circular, and exempt class entered the repository. They can read the record, not testify to the completeness of the unrecorded world. Stylistic confidence cannot repair that epistemic limit.

One Date, Two People, Two Answers

Ultimately a real person matters even without an invented story: an employee completing a bank report, an owner trying to understand an old commercial register, or an official reviewing a decision from months ago. Temporal error is not merely conflict between file versions. It can make a correct act appear defective under a later text, a stayed rule appear binding, or a transition designed to avoid harm disappear.

In software, the newest version is said to fix the last one. In rules and rights, the newest can be the wrong answer for an old event and even for today if it is not yet effective or has been suspended. Because evidence remains in the same language, the error is seductive: everything looks documented. This may be the hardest institutional-model failure—excellent proof of a **text** never proved applicable to the **case**.

To test institutional AI maturity, stop asking only “what is the newest text?” Ask something heavier: **on the day we made this decision, which rule applied to this event, what evidence established that, and what did our system know at that moment?** If it cannot separate these

questions, its answer is not “institutional truth,” however beautiful the prose.

References

1. SAMA Circular, September 24, 2025.
2. SAMA Rulebook, effective date February 1, 2026.
3. Registration, disclosure, and risk-mitigation requirements, item 10, Saudi Central Bank.
4. IBM, Best Practices for DB2 Temporal Data Management, April 2012.
5. Commercial Register Law, Saudi Bureau of Experts.
6. Saudi Ministry of Commerce, Commercial Register and Trade Names laws enter into force, April 3, 2025.
7. Federal Register, Non-Compete Clause Rule, May 7, 2024.
8. FTC, Noncompete Rule: status and court order.
9. Microsoft Learn, Query data in a system-versioned temporal table.
10. Nam et al., TimelyRAG, *Findings of EMNLP 2026*.
11. Nam et al., TimelyRAG full paper and stated limitations, 2026.
12. Pipitone and Alami, LegalBench-RAG, 2024.
13. Zheng et al., A Reasoning-Focused Legal Retrieval Benchmark, 2025.