

Stability Study Software Should Know the Calendar on Day One

Every pull a stability study will ever need is known the day it opens. Most software discovers them one at a time, and calls the ones it missed "overdue". What the software should do instead.

Brandon Holland, Marketing & Operations Digital Solutions Architect

August 27, 2026 · 9 min read

IN BRIEF

Stability study software should create every pull the day a study opens, per timepoint, condition, batch and pack, so overdue is a count, not a calculation. It should define out-of-trend in words a reviewer can check, let only quality set a verdict, keep every amendment, and hold the data a shelf life is set from without stating one.

Key takeaways

- Every pull a study will need is known the day it opens. Software should create them all then, not find them one at a time.
- Overdue should be a list of real pulls whose window has closed, not a screen's inference from the results that arrived.
- An out-of-trend flag needs a stated rule in words a reviewer can check. A flag nobody can explain is a flag people ignore.
- Only the quality unit sets a verdict, at release, and every earlier verdict stays readable with the reason it was changed.
- Shelf life is a statistician's conclusion under published guidance. Software should hold the data, show the slope, and stop.

A stability programme is not a series of tests. It is a calendar, and the whole calendar is known on the day the first study opens. A batch put on long-term storage this month commits the laboratory to pulling samples at fixed intervals for up to five years, for every batch, in every pack, at every storage condition. Knowing what is due next month, and what was missed last month, is most of what a stability coordinator actually does.

Most software built for stability does not start from that fact. It stores results as they arrive and works out what is late by comparing dates when someone asks. This article sets out what stability study software should do instead, from the calendar through out-of-trend flags and quality verdicts to the one thing it should refuse to say. The [pharmaceutical QC lab](#) is the audience, but the same argument holds for any programme built on scheduled pulls.

Why is a stability study a calendar rather than a series of tests?

Because the design fixes the dates before any result exists. The published ICH defaults for long-term storage put pulls at zero, three, six, nine, twelve, eighteen, twenty-four, thirty-six, forty-eight and sixty months;

intermediate at zero, six, nine and twelve; accelerated at zero, three and six. A study's protocol names its conditions, its batches and its packaging configurations, and the product of those with the timepoints is the complete list of pulls the study will ever need. Nothing about that list changes when a result comes in.

A system that understands this creates every pull as a scheduled record on the day the study opens, each with its due date and due window, waiting to be taken. The coordinator's worklist is then a view of records that already exist: what is due this month, what is in testing, what is awaiting review. Nothing is worked out in the moment, and nothing can be forgotten, because a pull that nobody has taken is a record sitting in a list rather than an absence nobody noticed.

A system that does not understand this treats each pull as a new event. Someone remembers, or a spreadsheet reminds them, that batch seven is due its eighteen-month pull. The record is created when the sample is taken. If nobody remembers, no record is created and no screen shows a gap, because the system never knew the pull was due. That is how a timepoint is missed by three weeks and discovered at the next review, and a stability timepoint taken three weeks late is a finding, not a detail.

What should "overdue" mean?

Overdue should mean a real pull, created on day one, whose due window has closed without the sample being taken. It should be a list of specific records, each naming its study, batch, condition, pack and timepoint, in due-date order. The count at the top of that list is the number of things the lab has actually missed, and it should be zero on a well-run programme most of the time.

What overdue should not mean is a screen's guess. Software that calculates lateness by scanning results and inferring which timepoints ought to have arrived is guessing, and it guesses wrong in both directions: it misses pulls it never knew about, and it flags pulls that were legitimately cancelled by a reduced design. A coordinator who cannot trust the overdue count checks it against a spreadsheet, and the spreadsheet is now the system of record.

Two refinements follow. First, a pull taken outside its due window should record that fact, and by how many days, on the record itself, so a late pull is visible on the study forever rather than only on the day. Second, cancelled timepoints should read as cancelled on every screen and every printed table, never as blank, because a blank cell in a stability summary table is a question a regulator will ask.

NOTE

“Overdue” is only meaningful when the calendar was created in advance. If your current software cannot show you every pull a study will need before any of them has been taken, its overdue count is an inference, and the spreadsheet your coordinator keeps beside it is not a failure of discipline. It is the system doing what the software should.

How should software decide a result is out of trend?

A result can be inside every specification limit and still be a problem, because stability is about direction. An assay that reads 99.1, 98.4, 97.2 and 95.8 at successive timepoints has not failed, but a reviewer who does not see the slope will be surprised at thirty-six months. That is what an out-of-trend flag is for, and there are two honest ways to raise one.

The first is a step past an allowance. The specification gives each test an allowance for how much a result may move between pulls, expressed per unit of time, and the software scales it to the real gap between the two timepoints so that the same allowance means the same thing over three months as over twelve. The second is a straight-line projection: a line drawn from the first result through the latest, carried to the end of the study, reaches a limit before the study ends. Both are arithmetic a reviewer can check by hand.

What matters more than the rule is the wording. A flag should say, in words, which rule fired and against what: “moved 3.4 from the 12-month result, and the allowance over that 6-month gap is 3”. A flag nobody can trace is a flag people learn to ignore, and a test with no stated allowance should be able to fail specification but never trend, because a permanently flagged test teaches people to ignore flags.

HOW A TREND FLAG SHOULD BE RAISED

Step past the scaled allowance since the previous pull

1 of the two rules

Straight-line projection reaching a limit before the study ends

1 of the two rules

A statistical model, confidence bound or regression

0 of the two rules

The two out-of-trend rules described in this article

Who sets the verdict, and what happens when it changes?

The software computes. The quality unit decides. That distinction should run through every screen: until a qualified reviewer releases a timepoint, its status should say that nobody has signed it off, and the verdict on screen should be labelled as what the system computes rather than as what the record says. An analyst entering results should have no release control to press. The verdict is only ever set by the quality unit, at release, every time.

Release should write the reviewer's name, the date, and the specification version the result was judged against onto the record. Those three facts are what an inspector reconstructs when a released result is questioned years later, and they should not have to be reconstructed. A released timepoint that was out of specification or out of trend should be marked as such wherever the study appears, so the finding cannot be lost in a table of passes.

Verdicts change. A result is re-tested, a specification is corrected, an investigation concludes the original entry was a transcription error. Amending a released timepoint should demand a stated reason and keep the earlier release readable: who released it, when, to what verdict, and why it was changed. Nothing in a stability programme should ever be deleted. Timepoints are cancelled, specifications are versioned, releases are superseded, and the history stays. A programme that can quietly overwrite a verdict is a programme

whose summary table an inspector cannot trust.

Why must a specification keep its versions?

Specifications change during a five-year study. A limit is tightened after a process improvement, a test is added, an allowance is revised once real variability is understood. Each change is legitimate. What is not legitimate is re-judging results that were released under the old specification against the new one, silently, because the software holds only the current limits.

So a specification should be versioned, and every released timepoint should keep the version it was judged against. A summary table should print the limits that applied to each result, and a reviewer opening a two-year-old timepoint should see the specification as it was on the day, not as it is today. New results are judged against the current version. Old ones keep theirs. The alternative is a study whose early results appear to fail a limit that did not exist when they were released, and an investigation into a problem the software invented.

Many products also carry two sets of limits: tighter release limits applied when a batch goes to market, and wider shelf-life limits it must still meet at expiry. Stability results should be judged against the shelf-life set, so that "out of specification" in a stability report always means what it says. The one place the release set matters is the zero-month result, which is effectively what batch release looked at, and a zero-month value inside shelf-life limits but outside release limits should carry an advisory rather than a failure. It is a batch release question, and recording it as a stability failure puts it in the wrong report in front of the wrong reader.

What should stability software refuse to say?

A shelf life. The software should hold the data a statistician sets a shelf life from, present it clearly, and stop there. No regression, no confidence bound, no batch poolability test, no proposed expiry date. Every trend flag should be a straight line and labelled as one on every page it appears on.

This is not modesty. Shelf-life determination is a statistical judgement made under published guidance, by a person qualified to make it, taking into account batch poolability, the shape of the degradation and the regulatory context of the submission. A number produced by software and printed on a report acquires an authority it has not earned, and a reviewer who disagrees with it has to argue with a machine in front of an inspector. The software's job is to make the reviewer's data complete and legible. The conclusion is the reviewer's.

The same restraint applies elsewhere. Photostability, where a product is exposed to a defined light dose and tested once against a dark control, has no timepoints and no trend, and the published guidance defines significant change per product. Software should show the difference between exposed and control for every test and invent no threshold. A person concludes. That rule, repeated wherever a judgement is required, is what separates stability software a quality unit can defend from software that quietly makes decisions on its behalf.

What a stability programme carries in this work

LabLynx has supported stability study testing in pharmaceutical and food and beverage laboratories for long enough to know where the spreadsheet beside the system comes from. The argument above is the shape of what we build, and the boundary is stated plainly below.

How LabLynx handles this

LabLynx One should create every timepoint a study will ever need on the day it opens, with its due date and window, so overdue is a list of real pulls rather than a calculation; judge each result against the shelf-life limits of a versioned specification with the two trend rules above, stated in words; let only the quality unit release a verdict and keep every amendment with its reason; and state no shelf life of any kind. It records excursions and results entered by people and does not read chamber sensors. If your programme needs a validated statistical shelf-life package, that remains a separate tool, and we will say so.

Frequently asked questions

What is a stability study LIMS?

Software that manages a pharmaceutical or product stability programme: the studies, their batches, packaging configurations and storage conditions, the scheduled pulls at each timepoint, the results entered against a versioned specification, the trend and specification flags, quality release of each timepoint, and the printed documents. A good one creates the whole calendar of pulls when a study opens.

What are the ICH stability timepoints?

The published ICH defaults for long-term storage are zero, three, six, nine, twelve, eighteen, twenty-four, thirty-six, forty-eight and sixty months. Intermediate storage uses zero, six, nine and twelve months, and accelerated storage uses zero, three and six. A protocol may vary these, and reduced designs may cancel some timepoints, but the schedule is fixed in the protocol before the study starts.

What does out of trend mean in stability testing?

A result inside its specification limits but moving in a way that predicts a failure. Two rules commonly raise the flag: the result moved more than the specification's allowance since the previous pull, scaled to the time between them, or a straight line through the results reaches a limit before the study ends. The flag should say which rule fired and against what figures.

Should stability software calculate shelf life?

No. Shelf life is a statistical determination made under published guidance by a qualified person, taking into account batch poolability, the shape of the degradation and the regulatory context. Software should hold the data that determination is made from, show trends clearly and label every trend line as a straight line. A proposed expiry date printed by software acquires authority it has not earned.

How is photostability different from a stability series?

A photostability study is a single exposure rather than a series over time. The product is placed under light until it has received a defined dose and tested once, beside an identical sample kept in the dark for the same period. Any change is judged against that dark control. There are no timepoints and no trend, and the published guidance defines significant change per product.

Sources and references

1. [ICH Q1A\(R2\) Stability Testing of New Drug Substances and Products](#)

International Council for Harmonisation

2. [ICH Q1B Photostability Testing of New Drug Substances and Products](#)

International Council for Harmonisation

3. [21 CFR 211.166 Stability testing](#)

Electronic Code of Federal Regulations

Brandon Holland

Marketing & Operations Digital Solutions Architect

Brandon Holland is LabLynx's Marketing & Operations Digital Solutions Architect. He builds and runs the digital systems behind the company's public surface, from the website and content architecture to the tooling that keeps product information accurate everywhere it appears. He writes about laboratory informatics from the systems side: how lab records get structured, searched, and kept defensible as a lab scales.

Read the current version online

<https://www.lablynx.com/resources/articles/stability-study-software-calendar/>

Downloaded September 7, 2026. LabLynx builds laboratory informatics software: LIMS, ELN, LIS and lab automation.