Salesforce Validation Rules Best Practices (Examples + Patterns)

CHEATSHEET

Master clean logic, actionable error messages, and bypass patterns for Salesforce admins.

Stop writing NOT() wrappers around validation rule logic. Negating valid states is how you get false positives and silent failures.

1 Catch the Invalid State Directly

Always test for what is WRONG, not what is RIGHT. Avoid wrapping everything in complex NOT() statements that are impossible to debug.

- Poor: NOT(AND(NOT(ISPICKVAL(StageName, 'Closed Won')), ISBLANK(CloseDate)))
- Better: AND(ISPICKVAL(StageName, 'Closed Won'), ISBLANK(CloseDate))
- Target the invalid state directly to align formula logic with plain-English business rules
- Clean formulas are faster to read, test, and maintain for future admins

2 Write Micro Help Articles, Not 'Error'

Your validation rule error is the only guide your user gets when a save fails. Treat it like micro-documentation to avoid administrative support tickets.

- Identify exactly what went wrong instead of writing generic 'Error' messages
- Tell the user which field has the issue and what they need to do to fix it
- Point the error message directly to the affected field instead of the top of the page

3 Know Your Rule Scope & Limitations

Validation rules evaluate on every single record save. Understanding this lifecycle prevents broken background automations and night-shift API crashes.

- Triggers on both insert and update operations by default
- Formula returns TRUE = blocks the save; formula returns FALSE = save proceeds
- Applies across all entry points: UI, API, Flow, Apex, and Bulk Data Loads
- Keep compile size within the hard limit of 3,900 compiled characters per rule