# Salesforce Field Cleanup

A Plain-Language SOP

No code required — just know what object you want to clean up

## 1. What this is

This process helps any Salesforce admin find fields on an object that are candidates for deletion, need investigation, or should be left alone. It does not require writing any code or SOQL. You just need to know the name of the object you want to clean up.

The process uses two tools — an AI assistant connected to your Salesforce org, and VS Code with your metadata — and takes three phases. The AI handles the technical work; you make the judgment calls.

| | |
|---|---|
| **What you need** | Phase 1 & 3: An AI assistant with a Salesforce connection (e.g. Claude with SFDC MCP, or similar).<br>Phase 2: VS Code with Salesforce CLI installed and a retrieved copy of your org metadata. |
| **The core rule** | A field needs to show zero on all three checks before it is safe to delete. One non-zero finding means investigate first, not delete. |

The three checks are:

- Does the field contain any data in the org?
- Is the field referenced anywhere in code, flows, or automation?
- Is the field visible to any user — on a layout, profile, or permission set?

## 2. Phase 1 — Ask your AI assistant

Open your AI assistant (with your Salesforce org connected) and type a prompt like this:

| **Your starting prompt — just fill in the object name** |
|---|
| *I want to investigate the fields on the [OBJECT NAME] object and see if any of them can be cleaned up. Can you pull all the fields and check which ones have data in them?* |

That single prompt triggers the AI to:

- Pull the full list of fields on the object

- Check how many records have data in each field
- Group fields into tiers — zero population, near-zero, moderate, and fully populated
- Flag any fields whose labels already say things like "To Be Deleted" or "OLD"
- Note which fields are written by integrations like Workato or an ERP system

| What to watch for | Pay attention to any field that is labeled for deletion but shows high data population. That means someone flagged it without cleaning it up first — those need a conversation with a business owner before anything happens. |
|---|---|

### What the AI produces

At the end of Phase 1 the AI will produce two things:

- A tiered field list grouped by population level
- A structured handoff prompt — a block of text you copy and paste into VS Code in Phase 2

| Follow-up prompt |
|---|
| *Now generate the VS Code handoff prompt for this object so I can check for references in the metadata.* |

## 3. Phase 2 — Check references in VS Code

Open VS Code. Make sure you have a recent copy of your org metadata retrieved locally. Then open a Claude chat in VS Code and paste in the handoff prompt from Phase 1.

| What is metadata retrieval? | Retrieving metadata means downloading a copy of your org's configuration files — flows, Apex code, page layouts, profiles, etc. — to your local machine. VS Code + Salesforce CLI can do this with one command. If you've never done it, ask your AI assistant: "How do I retrieve metadata from my Salesforce org using SFDX?" |
|---|---|

Once you paste the handoff prompt, VS Code + Claude will:

- Search every flow, Apex class, validation rule, and formula for each field name
- Search every page layout to see if the field is displayed anywhere
- Search every profile and permission set to see if any user has access to the field
- Search reports and report types for the field

### What comes back

VS Code returns a one-line verdict per field, then a grouped summary:

| Field | What was found | Verdict |
|---|---|---|
| Old_Category__c | No hits anywhere | READY TO DELETE |
| Legacy_Flag__c | Found in: PricingFlow (flow) | HOLD — resolve dependency first |
| Archived_Notes__c | Found in: Admin profile (FLS) | NEEDS BUSINESS INPUT |
| BOD_Pillar__c | Found in: 3 reports | HOLD — reports will break |

Copy the entire output from VS Code and bring it back to your AI assistant for Phase 3.

## 4. Phase 3 — Triage back in the AI assistant

Paste the VS Code results back into your AI assistant chat alongside the population data from Phase 1. Then ask:

**Triage prompt**

> *Here are the VS Code results. Can you combine these with the population data we pulled earlier and give me a final triage — which fields are ready to delete, which need investigation, and which should stay?*

The AI cross-references both signals and produces the final triage:

| Bucket | Criteria | Next action |
|---|---|---|
| Delete candidate | Zero or near-zero data AND no code/flow/layout references | Get stakeholder sign-off, then delete |
| HOLD | Any code, flow, or automation reference found regardless of population | Trace and remove the dependency first, then re-evaluate |
| Needs business input | On a layout or profile but low/no data | Ask the field owner: is this intentional? |
| Keep | Actively populated or confirmed business-critical | Document and leave alone |

### The audit workbook

Ask the AI to build the workbook output:

**Workbook prompt**

> *Can you build the audit workbook for [OBJECT NAME] from this triage? I want tabs for Delete Candidates, HOLD, Needs Business Input, and Keep — with a summary dashboard at the front.*

The workbook becomes the artifact you use to get sign-off from stakeholders and track what has been cleaned up.

## 5. Before you delete anything

Even after a field clears all three checks, a few steps are required before deletion:

- Get sign-off from whoever owns the data on that object — Sales Ops, Marketing, Service, etc.
- If a flow references the field, deactivate the flow or remove the reference first. Your AI assistant can tell you exactly which flows need updating.
- If an Apex class references the field, a developer needs to update and redeploy that code first.
- If the field is on a layout, remove it from the layout before deleting.
- Test in a sandbox if the field has any data at all, even a small amount.
- Delete via Setup > Object Manager > Fields and Relationships.

| **Important** | Salesforce field deletion is permanent. Any data in the field is lost. Always confirm zero population before proceeding, and always get a stakeholder to confirm they no longer need the field. |
|---|---|

## 6. Quick reference — prompts to use

### Phase 1 — Start the audit

**Prompt 1**

*I want to investigate the fields on the [OBJECT NAME] object and see if any of them can be cleaned up. Can you pull all the custom fields and check which ones have data in them?*

**Prompt 2 (follow-up)**

*Now generate the VS Code handoff prompt for this object so I can check for references in the metadata.*

### Phase 3 — Triage

**Prompt 3**

*Here are the VS Code results. Can you combine these with the population data we pulled earlier and give me a final triage — which fields are ready to delete, which need investigation, and which should stay?*

**Prompt 4**

*Can you build the audit workbook for [OBJECT NAME] from this triage? I want tabs for Delete Candidates, HOLD, Needs Business Input, and Keep — with a summary dashboard at the front.*

### Any time — ask follow-up questions

**Examples**

*Why is BOD_Pillar__c flagged for deletion if it has data? / Which fields are being written by Workato? / What do I need to do before I can delete Old_Category__c? / Who should I ask about this field before deleting it?*

## 7. Key principles

- You don't need to write code or queries — describe what you want in plain language
- Population data alone is not enough to delete a field — you need all three signals
- Fields labeled "To Be Deleted" in Salesforce are intentions, not approvals
- ERP-written fields should always be confirmed with your integration team before touching
- Low checkbox population is usually intentional — always ask a business owner
- The AI tells you what it found; you make the call on what to do about it