CRM DATA MODEL DESIGN TEMPLATE

Map your objects, own your fields, define your stages.
Do all of it before you touch the platform.

BUILT FOR

- RevOps & Revenue Operations leads
- CRM Admins and Architects
- Sales & Marketing Operations
- GTM Engineers
- Founders setting up their first real stack

WHAT YOU WALK AWAY WITH

- A documented object relationship map
- Field ownership rules, written down
- Lifecycle stage definitions with SLAs
- A naming convention that holds up
- HubSpot and Salesforce-specific guidance
- A pre-build checklist you'll actually use

RevOps Brief · Vol. 1 · CRM Architecture Series · HubSpot & Salesforce

Before You Open the Platform

Every broken CRM I've ever seen was broken the same way. Not a technology failure — HubSpot and Salesforce both work. The failure was always in the design phase, which usually didn't happen. Someone got access, started creating fields, and kept going. Six months later the data model is a mess of undocumented properties, lifecycle stages that mean different things to different people, and a sales team that's stopped filling things in because the system is more confusing than helpful.

That's what this document is for. Everything in here comes from real implementations — the calls that worked, the shortcuts that created problems six months later, the fights I've had and the ones I've prevented. It's opinionated on purpose. If something in here pushes back on how you've done things before, sit with that for a minute before dismissing it.

The aim isn't a flawless system from day one. It's a system that's coherent enough to improve over time without falling apart. Your CRM is never finished. Your business changes, your motion changes, your stack changes. What you need is a foundation that can absorb those changes without requiring a re-architecture every 18 months.

So: read Part I before you create anything. Do the object mapping before you touch a single field. Agree on lifecycle stages before you build a single workflow. It feels slow. It isn't. It's how you go fast without breaking things.

Now let's build something worth keeping.

How to Use This Document

Section	What You're Doing	Time Investment
Part I — Foundation	Understand why the design phase exists and what it costs to skip it	30 min read
Part II — Object Mapping	Map every object, its purpose, and how it connects to everything else	2–4 hrs workshop
Part III — Field Design & Ownership	Define every field: type, owner, who can write to it, and why it exists	3–6 hrs per object
Part IV — Lifecycle Stages	Write entry/exit definitions for each stage. Get sign-off before you build.	2–3 hrs with GTM leads
Part V — HubSpot	Apply the model to HubSpot's specific architecture and constraints	1–2 hrs review
Part VI — Salesforce	Apply the model to Salesforce's object model and relationship types	1–2 hrs review
Part VII — Data Governance	Build the rules that keep the model honest over time	1 hr setup
Part VIII — What Not to Do	Six patterns that kill CRM implementations. Check if you're building any of them.	20 min read

Section	What You're Doing	Time Investment
Part IX — Pre-Build Checklist	36 items. Don't go live until they're all checked.	30 min review

PART I

Foundation: Why the Data Model Is Everything

1.1 The Real Cost of Skipping This Step

Nobody puts "CRM re-architecture" in a vendor's ROI calculator. But it's real, and it's expensive. The teams that skip the design phase and go straight to building don't save time — they borrow it. Twelve to eighteen months later, they pay it back with interest: a re-architecture project that costs more in time, budget, and political capital than proper design ever would have.

A bad data model doesn't fail all at once. It fails slowly. Field naming gets ambiguous. Marketing and Sales start arguing about what "qualified" means in every QBR because nobody wrote it down. Two systems write to the same field and create race conditions. Validation rules are missing, so garbage enters cleanly and silently corrupts every report downstream.

By the time the CEO asks why the forecast keeps missing, the answer is buried somewhere in 200 custom fields nobody documented, a lifecycle stage that means three different things to three different teams, and a lead source field where six different values all effectively mean "website."

> ⚠ **The 18-Month Rule**
>
> A CRM built without a documented data model will need a partial or full re-architecture within 18 months. The warning signs usually show up around month nine — reporting starts feeling off, people stop trusting the numbers — but the crisis arrives at 18. Design the model now. It is dramatically cheaper.

1.2 Three Layers. Don't Confuse Them.

Your CRM operates at three distinct layers. Mixing them up — which is the most common design error I see — is how you end up with automations built on top of an unstable schema, and reports that nobody believes because the data underneath was never validated.

Layer	What It Is	Who Owns It	The Mistake Everyone Makes
Schema	Objects, fields, relationships, data types, required fields	RevOps / CRM Admin	Built reactively, one field at a time, no naming convention, no documentation
Process	Workflows, automations, lifecycle transitions, sequences, alerts	RevOps + Marketing + Sales Ops	Built before the schema is stable — so they break every time the schema changes
Reporting	Dashboards, pipeline reports, attribution, forecasting	RevOps + Leadership	Expected to be accurate when the underlying data was never cleaned or validated

1.3 Five Principles. Hold to All of Them.

These are the rules behind every recommendation in this document. When you're stuck on a design call, come back here.

One source of truth per data point

If your contract value lives in your billing system, your CPQ, and your CRM deal record — and the numbers sometimes disagree — you don't have a CRM problem. You have a data architecture problem that your CRM is exposing. Before you build anything, define the system of record for every data type. Write it down.

Explicit beats implicit, every time

A field that can be interpreted two ways will be interpreted two ways, and those interpretations will drift further apart over time. A "Lead Source" field that captures both the original acquisition channel and the last-touch campaign is not one field — it's two fields at war with each other. Name things precisely. Longer names are fine. Ambiguous names are not.

Build fields for specific reasons, not "just in case"

A field with no owner, no definition, and no data in it is not a safety net. It's debt. Only create a field when you have a specific use case, a specific owner, and a specific answer to "how does this show up in reporting?" If you can't answer all three, don't create it yet.

Automate the routine. Protect the critical.

Automation should handle the repetitive: stage transitions, score updates, task creation, alerts. But the fields that drive forecasting — close date, deal amount, ARR — should not be touched by automation unless you've deliberately designed it that way and documented the logic. When automation overwrites critical fields, you lose the ability to audit what happened.

Design for the analyst, not the rep

Your CRM is a reporting database. Salespeople happen to use it as a productivity tool. Every field you create should answer one question: "Will I ever want to filter, group, or aggregate on this?" If the answer is no, think hard about whether the field needs to exist at all.

PART II

Object Relationship Mapping

2.1 The Core Object Model

HubSpot and Salesforce use different names for the same concepts. Contact is Contact in both. Company in HubSpot is Account in Salesforce. Deal in HubSpot is Opportunity in Salesforce. The naming is different; the underlying logic is the same.

The diagram below is your starting point. Every downstream design decision — field ownership, lifecycle stages, automation logic, reporting structure — depends on getting this right. Don't skim it.

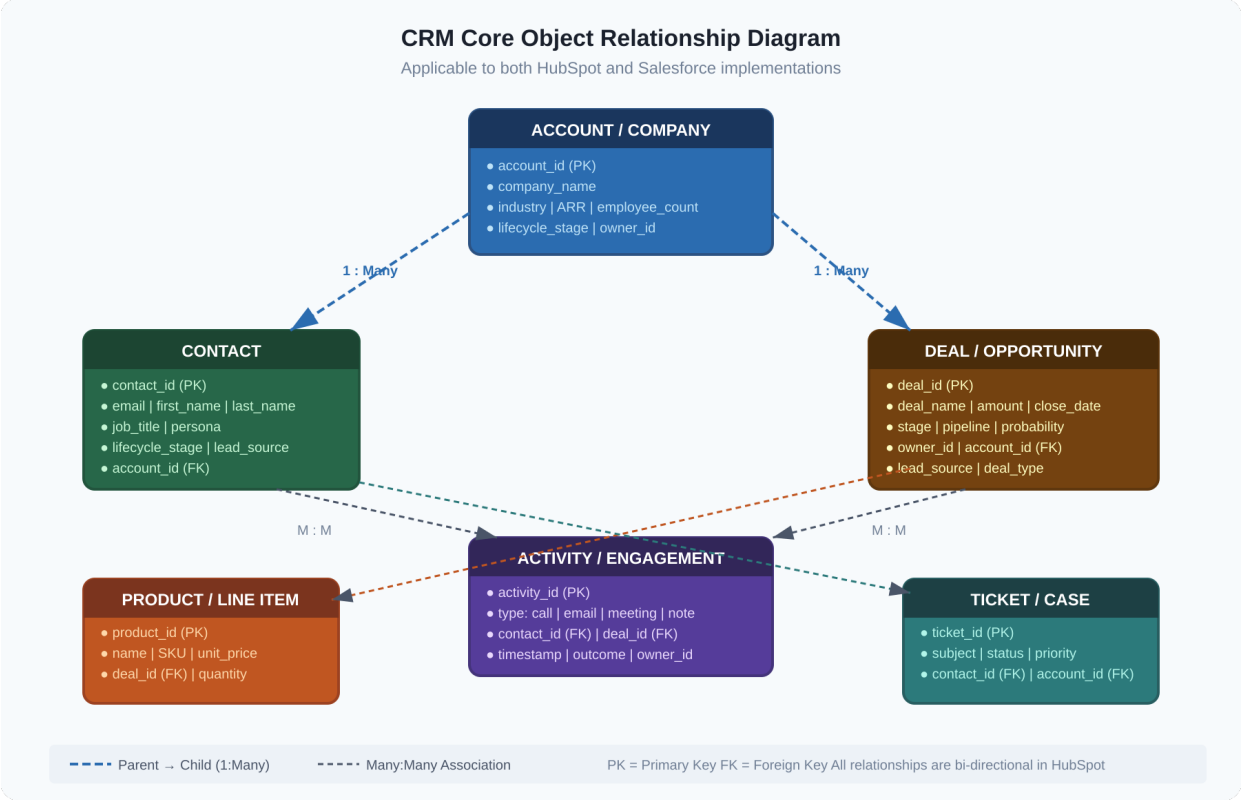

Figure 1 — Core CRM Object Relationship Diagram. Applies to both HubSpot and Salesforce. Relationships are bi-directional; labels denote cardinality.

2.2 Relationship Types — Pick the Right One

The relationship type you choose is hard to change after data exists. Here's what each option actually means and when to reach for it.

Relationship Type	What It Means	HubSpot	Salesforce	Use It When
One-to-Many (1:M)	One parent record, many children	Associations	Master-Detail or Lookup	Account → Contacts, Account → Deals

Relationship Type	What It Means	HubSpot	Salesforce	Use It When
Many-to-Many (M:M)	Records on both sides relate to many on the other	Associations (native)	Junction Object	Contacts ↔ Deals, Contacts ↔ Campaigns
Lookup (Weak)	A reference to another object; parent not required	Associations	Lookup Relationship	Deal → an optional second Contact
Master-Detail (Strong)	Child requires a parent; inherits parent's sharing rules	Not available	Master-Detail	Line Items → Opportunity
Self-Referential	An object that relates to itself	Workaround required	Lookup to same object	Account parent hierarchy

2.3 Object Inventory Worksheet

Fill this out before you open the platform. Seriously. This exercise surfaces 80% of design conflicts before they become schema problems. If two stakeholders describe the same object differently, that's a conversation you need to have now — not after you've built automations on top of the wrong assumption.

Object	Its Job in Your GTM	Key Relationships	Custom Object Needed?	Estimated Volume
Contact / Lead	An individual — prospect, customer, or partner	Lives under Account; connects to Deals and Activities	Rarely	Under 500K for most SaaS
Account / Company	An organisation — target, prospect, or customer	Has Contacts, Deals, and Tickets	Rarely	Under 100K for most SaaS
Deal / Opportunity	An active or closed revenue event	Under Account; has Contacts and Line Items	Sometimes (e.g. renewal as separate type)	10K–50K active is common
Activity / Engagement	Every touchpoint: call, email, meeting, note	Tied to Contact AND Deal	No	Can hit millions — plan your storage
Product / Line Item	What was sold, at what price, in what quantity	Lives on a Deal	Sometimes, for complex pricing	Mirrors deal volume
Ticket / Case	Post-sale support interactions	Connects to Contact and Account	Sometimes	Driven by ticket volume
Campaign	The marketing programme behind attribution	Contacts via Campaign Member	Rarely	Hundreds to low thousands

Object	Its Job in Your GTM	Key Relationships	Custom Object Needed?	Estimated Volume
Custom Object	Partner record, contract, subscription, etc.	Define all associations before creating	Yes — define it first	Depends entirely on use case

> **Run a Workshop Before You Build**
>
> Get Marketing, Sales, CS, and Finance in a room for two hours before anything is created in the platform. Put the object diagram on a whiteboard. Ask each team: "What do you need to know about each object to do your job?" The answers tell you what fields you actually need. The disagreements tell you where your governance gaps are.

2.4 Cardinality Reference — Know Your Limits

One Contact associated to 47 Deals might be correct if you're tracking full history. One Contact associated to 47 Companies is almost certainly a data quality problem. Set thresholds before you go live and build alerts when they're breached.

Association	Normal Range	Flag At	What to Check
Contact → Companies	1 primary, a few historical	More than 3	Likely a merge or deduplication issue
Company → Contacts	Scales with org size	Over 500 for an SMB	Check if the company record is a catchall
Company → Deals	Unlimited	Over 50 open simultaneously	Check for automation-created duplicates
Deal → Contacts	1 primary plus 2–8 stakeholders	Over 15	Usually a cleanup problem
Contact → Activities	Unlimited	Monitor storage costs	Archive activities older than 3 years
Deal → Line Items	Depends on catalogue size	Over 100	Check if product data is being used correctly

PART III

Field Design & Ownership

3.1 The Field Ownership Matrix

The field ownership matrix is the most underrated document in CRM design. One simple question: who is allowed to write to this field? That question has huge consequences. When two teams — or two systems — write to the same field without coordination, you get overwrites, race conditions, and data that slowly drifts from the truth.

The matrix below shows a standard model. Adapt it to your org. But never leave it undefined.

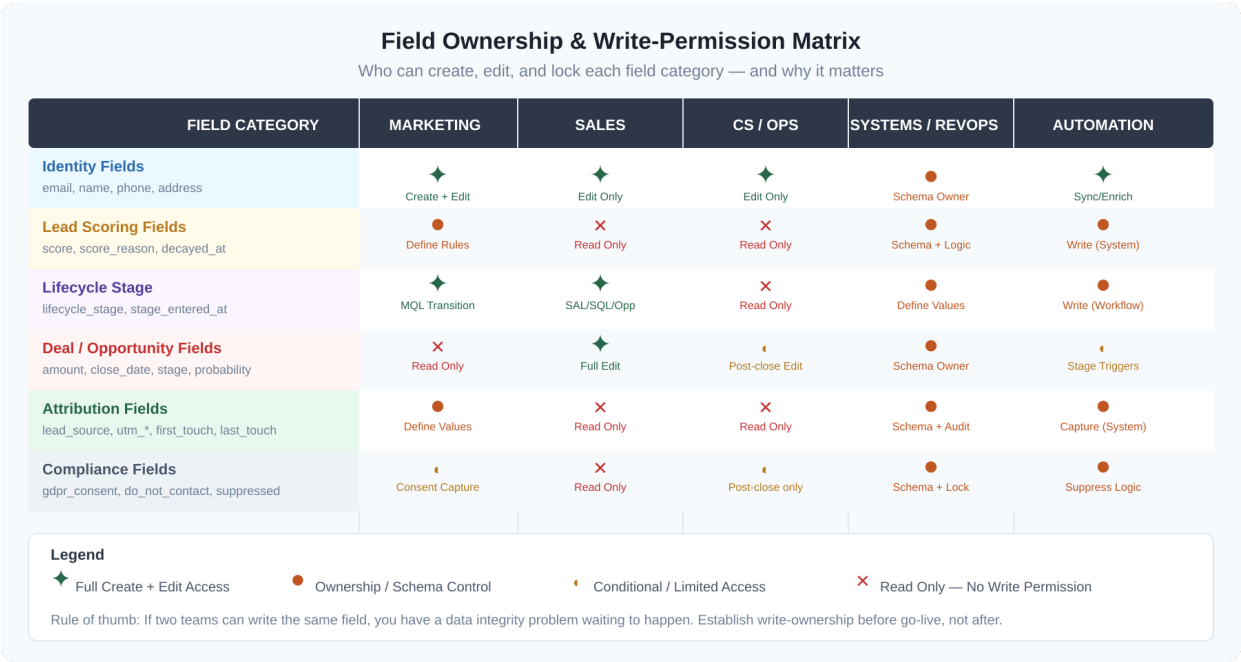

Figure 2 — Field Ownership & Write-Permission Matrix. Get this agreed before go-live. Revisit every quarter.

3.2 What Every Field Needs Before It Gets Created

Every custom field should be documented before it exists in the platform. These are the minimum requirements. If you can't answer all of them, the field isn't ready to be built.

Attribute	What You're Defining	Example
API Name	The machine-readable name used by integrations and automations	hs_contact_lead_source_first_touch
Display Label	What users actually see in the CRM	Lead Source — First Touch
Object	Which object this field lives on	Contact
Data Type	Text, Number, Dropdown, Date, Boolean, etc.	Dropdown (Single Select)

Attribute	What You're Defining	Example
Allowed Values	The complete picklist — and who controls adding new values	Paid Search, Organic, Direct, Referral…
Required?	Does saving a record require this field to be filled?	No — populated by automation at form submit
Field Owner	The authoritative writer for this field	Marketing / Automation
Write Permissions	Who else can edit this field, if anyone	RevOps only; locked to Sales and CS
Populated By	How does this field get its value in practice?	UTM parameter captured at first form submission
Used In Reporting?	Which dashboards or reports depend on this field?	Attribution Dashboard, Demand Gen Weekly
Integration Sync?	Does this field talk to another system?	Syncs to Marketo; incoming from Salesforce
Notes	Edge cases, conditional logic, or anything that'll confuse the next person	Set at first touch only. Never overwritten after.

3.3 Contact Field Reference

This is a starting point. Your business needs different fields than mine did. What matters is that every field you add has the spec above before it's created.

Identity & Profile Fields

Field Label	API / SF Field Name	Type	Owner	Notes
First Name	firstname	Text	System	Standard field. Do not rename it.
Last Name	lastname	Text	System / Sales	Recommend making this required
Email	email	Email	System	Primary unique identifier. Validate format on entry.
Phone	phone	Phone	Sales	E.164 format if you're integrating a dialler
Job Title	jobtitle	Text	Contact / Sales	Use for persona mapping, not segmentation on its own
Persona	hs_contact_persona	Dropdown	Automation	Derived from job title + seniority via workflow
Seniority Level	hs_contact_seniority	Dropdown	Automation	C-Suite, VP, Director, Manager, IC, Unknown

Attribution Fields — Get These Right or Pay for It Later

Field Label	API / SF Field Name	Type	Owner	Notes
Lead Source (First Touch)	hs_contact_lead_source_orig	Dropdown	Automation	Captured at first conversion. Never overwritten. This is your acquisition data.
Lead Source (Latest)	hs_contact_lead_source_latest	Dropdown	Automation	Updates on each new touch. Pair with a date stamp field.
UTM Source	hs_contact_utm_source	Text	Automation	Raw UTM value. Preserve exactly as captured — no cleanup here.
UTM Medium	hs_contact_utm_medium	Text	Automation	Raw UTM value. Same rule.
UTM Campaign	hs_contact_utm_campaign	Text	Automation	Raw UTM value. Same rule.
First Touch Date	hs_contact_first_touch_date	Date/Time	Automation	Timestamp of first known conversion event
First Touch Page	hs_contact_first_touch_page	Text	Automation	Page URL at first conversion — useful for content attribution
Referral Source	hs_contact_referral_source	Dropdown	Automation / SDR	For non-digital referrals: Partner, Customer, Event

Qualification & Scoring Fields

Field Label	API / SF Field Name	Type	Owner	Notes
Lead Score	hs_lead_score	Number	Automation (System)	Read-only for all teams. Managed entirely by the scoring model.
Score Bucket	hs_contact_score_bucket	Dropdown	Automation	Hot (>80), Warm (40–79), Cold (<40) — keeps reporting readable
ICP Fit Score	hs_contact_icp_fit	Number (1–5)	Automation / RevOps	Firmographic match score. Define the dimensions before building.
ICP Tier	hs_contact_icp_tier	Dropdown	Automation	Tier 1 / 2 / 3 / Out of ICP

Field Label	API / SF Field Name	Type	Owner	Notes
SDR Qualification Notes	hs_contact_sdr_qual_notes	Long Text	SDR	Freetext context only. Not used in any report.
BANT: Budget Confirmed	hs_contact_bant_budget	Boolean	SDR / AE	Yes / No / Unknown
BANT: Authority	hs_contact_bant_authority	Dropdown	SDR / AE	Decision Maker, Influencer, Champion, Blocker
BANT: Need Confirmed	hs_contact_bant_need	Boolean	SDR / AE	Yes / No / Unknown
BANT: Timeline	hs_contact_bant_timeline	Dropdown	SDR / AE	<30 days, 30–90 days, 90+ days, No timeline defined

3.4 Deal / Opportunity Field Reference

The Fields That Drive Forecasting

Field Label	API / SF Field Name	Type	Owner	Notes
Deal Name	dealname / Name	Text	AE	Enforce a naming convention: [Company] — [Product] — [Qtr]. Without this, pipeline reporting is a mess.
Amount	amount / Amount	Currency	AE	Must equal sum of line items if you're using Products. If not, you have two numbers that disagree.
Close Date	closedate / CloseDate	Date	AE	Required. Do not auto-populate this with end-of-quarter. That's not a forecast; that's a guess.
Deal Stage	dealstage / StageName	Dropdown	AE / Automation	Tied to pipeline. Each stage should have defined entry and exit criteria.
Pipeline	pipeline	Dropdown	AE / RevOps	New Business, Expansion, Renewal — separate pipelines from day one.
Deal Type	hs_deal_type	Dropdown	AE	New Logo, Expansion, Renewal, Reactivation

Field Label	API / SF Field Name	Type	Owner	Notes
Probability	hs_deal_probability	Percentage	Automation / AE	Drive from stage by default. Allow AE override for forecast exceptions.
Forecast Category	hs_deal_forecast_cat	Dropdown	AE / RevOps	Omit, Pipeline, Best Case, Commit, Closed
Lead Source (Deal)	hs_deal_lead_source	Dropdown	Automation	Inherited from the originating Contact at opportunity creation

Win/Loss Fields — Most Teams Under-Build These

Field Label	API / SF Field Name	Type	Owner	Notes
Close Reason (Won)	hs_deal_close_reason_won	Dropdown	AE	Champion strength, POC success, Pricing fit, Urgency created, Incumbent displacement
Close Reason (Lost)	hs_deal_close_reason_lost	Dropdown	AE	Price, Competitor, No budget, No urgency, Champion left, Status quo won
Primary Competitor	hs_deal_lost_competitor	Dropdown	AE	Update the picklist quarterly based on what you're actually seeing
Decision Maker Engaged?	hs_deal_dm_engaged	Boolean	AE	One of the strongest predictors of forecast accuracy. Require this before Commit stage.
POC Completed?	hs_deal_poc_completed	Boolean	AE / SE	Flag for proof of concept or trial completion
POC Outcome	hs_deal_poc_outcome	Dropdown	AE / SE	Success, Partial, Failed, N/A
Proposal Sent Date	hs_deal_proposal_sent_date	Date	Automation	Stamp when the proposal is sent via CRM. Measure proposal-to-close velocity.
Days in Current Stage	hs_deal_days_in_stage	Number	Automation	Auto-calculated. Critical for velocity analysis and stall detection.

3.5 Naming Conventions

Naming conventions aren't bureaucracy. They're infrastructure. A CRM with consistent naming is one where a new admin can get oriented in days instead of weeks. It's one where automations are readable. It's one where integrations don't silently break because someone changed a display label and didn't realise the API name didn't follow.

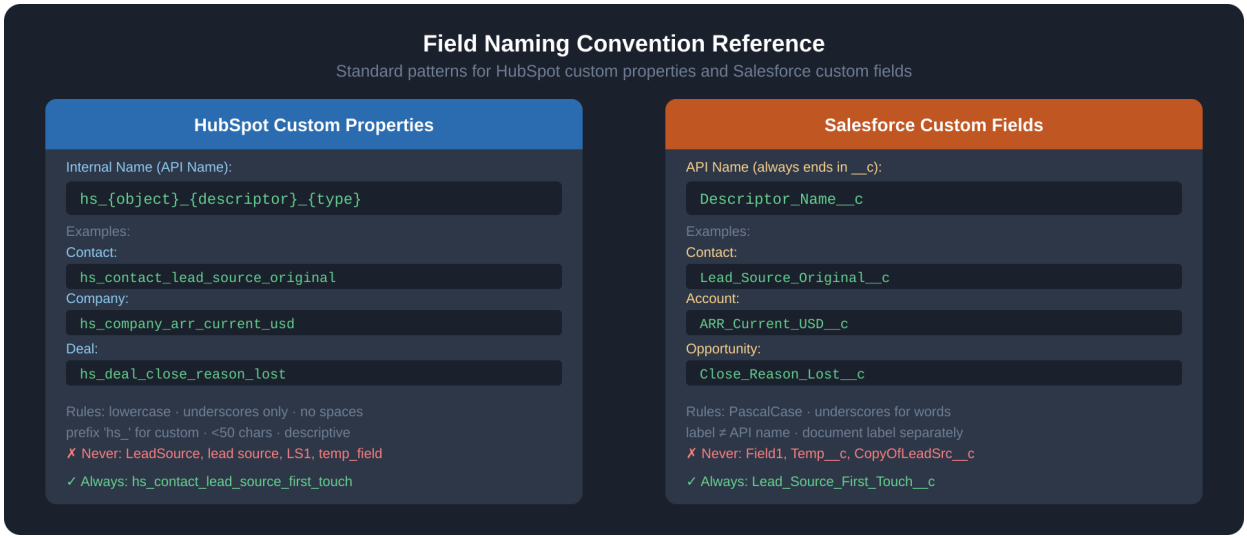

Figure 3 — Field naming conventions for HubSpot custom properties and Salesforce custom fields. These are non-negotiable.

Four Rules That Aren't Negotiable

Rule 1: No field gets created without being documented in the field registry first. Rule 2: No abbreviations that won't be obvious to someone who joins the team in a year. Rule 3: In Salesforce, the API name is permanent after creation — get it right before saving. Rule 4: Never put the word "new" in a field name. New_Deal__c becomes Old_Deal__c and you cannot rename it.

PART IV
Lifecycle Stage Design

4.1 Why Lifecycle Stages Break

Lifecycle stages fail for one reason: nobody wrote down what they mean before the CRM went live. Marketing calls something an MQL. Sales looks at the same record and calls it garbage. They're both right by their own definition — because there is no shared definition. This is not a technology problem. It's an alignment problem, and the technology will faithfully execute whichever misaligned version you build into it.

The fix is painful because it requires getting people in a room and making actual decisions. But it's a one-time pain. Undefined lifecycle stages are an ongoing one — they come up in every pipeline review, every QBR, every conversation about what the funnel numbers mean.

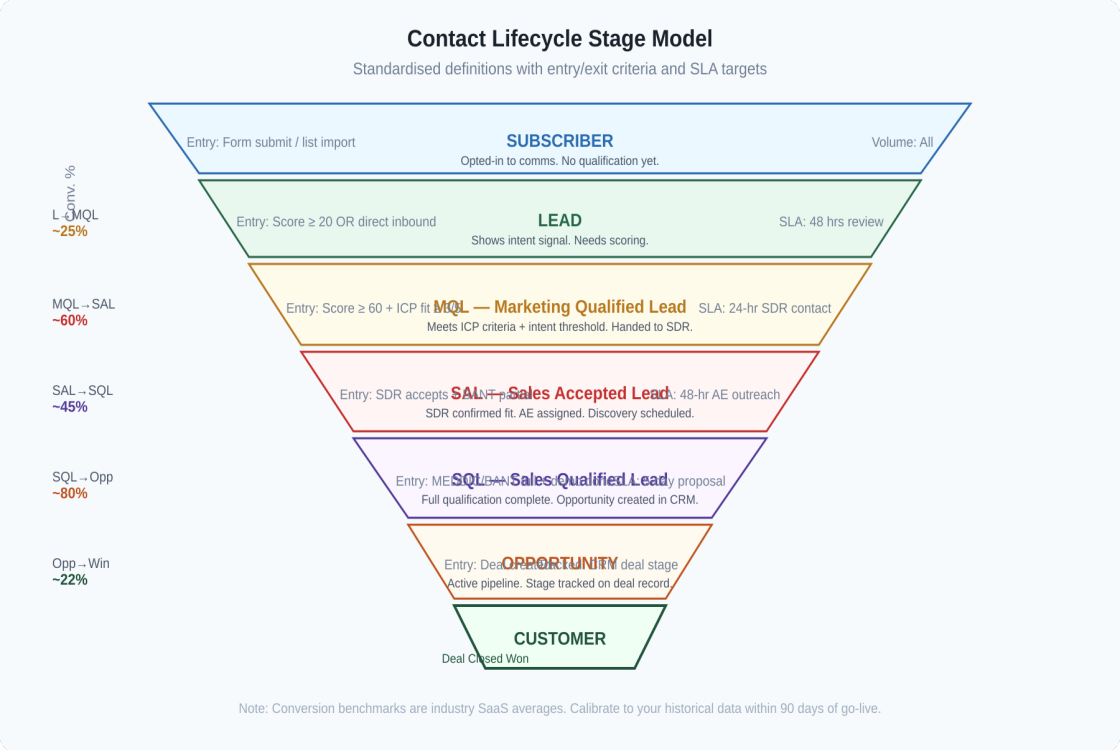

Figure 4 — Contact Lifecycle Stage Model with entry criteria, SLA targets, and benchmark conversion rates for B2B SaaS.

4.2 Lifecycle Stage Definitions

Stage 1: Subscriber

A Subscriber has raised their hand in some way — downloaded something, signed up for a newsletter, registered for a webinar. They're in your database, they've opted in to hear from you, and they've shown zero purchase intent. They are not leads. Do not route them to your SDR queue.

Attribute	Definition
Entry	Opted into marketing comms via form, event registration, or list import
Exit → Lead	Reaches lead score threshold, or takes a high-intent action (pricing page, contact sales)
Owner	Marketing. Passively nurtured until an intent signal fires.
SLA	None. Nurtured automatically until something changes.
CRM Action	Contact created; lifecycle_stage = Subscriber; lead_source_original stamped immediately
Don't	Count these as leads in your funnel reports. They're not leads yet.

Stage 2: Lead

A Lead has moved beyond passive interest. They've taken enough actions — repeated visits, content downloads, a threshold crossed in your scoring model — to suggest they're aware of a problem and looking for solutions. Not qualified. Not sales-ready. But worth watching.

Attribute	Definition
Entry	Score ≥ 20, OR a high-intent page view, OR a direct inbound action (contact form, live chat)
Exit → MQL	Score hits MQL threshold AND meets minimum ICP criteria (≥ 3 of 5 dimensions)
Exit → Disqualified	Confirmed as student, competitor, wrong geo, spam, or a duplicate record
Owner	Marketing. SDR may pick up if the inbound action was high-intent.
SLA	SDR review within 48 hours for any Lead scoring above 40
CRM Action	lifecycle_stage updated; lead_entered_date stamped; alert if high-intent action taken

Stage 3: MQL — Marketing Qualified Lead

The MQL is the handoff. Marketing is saying: "We've done our job — this person is worth a conversation." The definition of that needs to be written down, agreed on, and enforced in the system. An MQL that doesn't get contacted within 24 hours doesn't just miss an opportunity. It decays. Intent is perishable.

Attribute	Definition
Entry	Score ≥ [your threshold — set it based on your historical data] AND ICP fit ≥ 3/5 AND not disqualified
ICP Dimensions (example)	Industry + Company size + Geography + Tech stack + Buying role — define all five before launch
Exit → SAL	SDR contacts within SLA, validates fit, and formally accepts the lead in CRM

Attribute	Definition
Exit → Recycled	SDR cannot make contact after 5 attempts over 5 days. Returns to nurture — not lost.
Exit → Disqualified	SDR confirms the lead is outside ICP, a competitor, or an invalid contact
Owner	SDR / BDR. Named SDR assigned within 4 hours of MQL trigger.
SLA	First contact attempt within 24 hours. Five attempts over five business days before recycling.
CRM Action	lifecycle_stage = MQL; mql_date stamped; SDR task created; owner assigned automatically
Reporting note	MQL volume and MQL-to-SAL conversion are your primary marketing performance metrics

Stage 4: SAL — Sales Accepted Lead

SAL is the accountability checkpoint that most teams skip and then regret. When the SDR accepts a lead, they're confirming: this person exists, they work somewhere in ICP, and there's a plausible reason to pursue them. Without SAL, you cannot measure whether Marketing is sending quality — or whether Sales is rejecting leads they shouldn't be.

Attribute	Definition
Entry	SDR has reviewed, made contact or done research, and confirmed minimum fit. Logged as accepted in CRM.
Minimum to Accept	Company is real and in ICP + contact is reachable + there is a plausible use case
Rejection Rule	SDR must select a rejection reason from a defined picklist. No freetext rejections. Ever.
Exit → SQL	Discovery call done. BANT partially confirmed. AE assigned. Opportunity created.
Owner	SDR — actively working toward a booked discovery call
SLA	AE contact or discovery call booked within 48 hours of SAL acceptance
CRM Action	lifecycle_stage = SAL; sal_date stamped; AE assigned; discovery call task created

Stage 5: SQL — Sales Qualified Lead

SQL is the moment the deal becomes real. Discovery has happened, the AE has confirmed there's a genuine opportunity here, and a deal record gets created. From this point, lifecycle is tracked at the deal level — not the contact level. The contact moves to SQL; the deal tells the rest of the story.

Attribute	Definition
Entry	Discovery done + AE confirms: real need, engaged authority, defined timeline, budget exists or is findable

Attribute	Definition
Qualification Framework	MEDDIC, BANT, or SPICED. Pick one. A hybrid is just no framework with extra steps.
Deal Created	Opportunity record opened in CRM and associated to both Contact and Account
Exit → Opportunity	Automatic on deal creation. SQL and Opportunity stages overlap intentionally.
Owner	AE. Full ownership of deal and contact from this point forward.
SLA	Next step defined and dated on the deal record within 24 hours of deal creation
CRM Action	lifecycle_stage = SQL; sql_date stamped; deal created and associated

4.3 Deal Pipeline Stage Design

Pipeline stages live on the deal record, separate from contact lifecycle stages. Here is a standard seven-stage new business pipeline. You can rename the stages to fit your language — but don't compress below five stages without losing visibility into where deals actually stall.

#	Stage	Default %	To Enter	To Exit	Avg. Days (SaaS)
1	Discovery	10%	SQL created; first call done	AE confirms qualification criteria are met	1–5
2	Qualification	20%	Qualification framework partially confirmed	Full qualification done; solution approach agreed	5–10
3	Demo / Evaluation	35%	Product demo scheduled or delivered	Prospect engaged; next step clearly defined	7–14
4	POC / Pilot	50%	Trial or POC formally started	Success criteria agreed and met	14–30
5	Proposal	65%	Commercial proposal sent	Proposal reviewed; procurement engaged	5–10
6	Negotiation	80%	Legal or commercial negotiation underway	Terms agreed; contract ready to sign	5–15
7	Verbal Commit	90%	Prospect has verbally committed	Contract signed	1–5
—	Closed Won	100%	Contract executed	— Terminal stage	—
—	Closed Lost	0%	Prospect has declined	— Terminal stage	—

4.4 The Lead-to-Revenue Flow

This diagram shows how lifecycle stages map to team ownership, system events, and CRM record actions across the full buyer journey. If you can only hang one thing on the wall during your go-live planning, make it this.

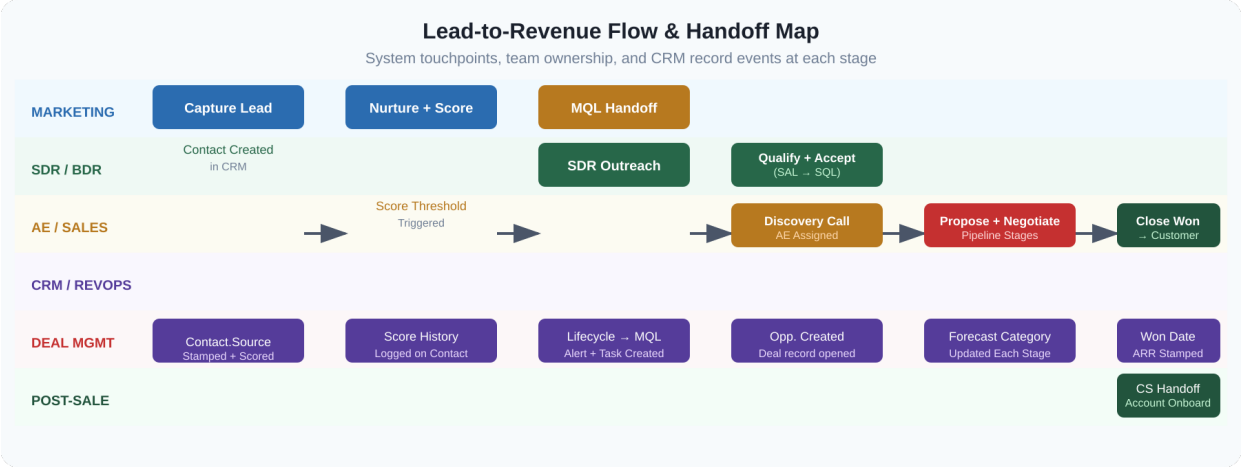

Figure 5 — Lead-to-Revenue handoff flow. System events, team swim lanes, and CRM record actions at each stage.

PART V

HubSpot-Specific Implementation

5.1 HubSpot Object Architecture

HubSpot supports Contacts, Companies, Deals, Tickets, Calls, Emails, Meetings, Notes, Tasks, Products, and custom objects (Enterprise tier). The tier you're on determines what you can build natively versus what you'll need to work around.

HubSpot Object	Salesforce Equivalent	What You Need to Know
Contact	Contact / Lead (pre-converted)	HubSpot has no separate Lead object. Everyone is a Contact. Design your lifecycle tracking around this from the start.
Company	Account	A Contact can associate to multiple Companies. The Primary Company is what most reports use.
Deal	Opportunity	Deals live in a Pipeline. You can — and should — run multiple pipelines for different sales motions.
Ticket	Case	Post-sale CS and support. Link to both Contact and Company.
Custom Object	Custom Object	Enterprise only. Define the object, its properties, and every association before you create it.
Product / Line Item	Product2 / OpportunityLineItem	Products live in a catalogue. Line Items live on the Deal. The association is automatic when you add a product.
Association	Relationship (Lookup / Master-Detail)	HubSpot uses Associations for everything. Cardinality and labels can be customised in Enterprise.

5.2 Property Groups

HubSpot organises custom properties into Groups, which appear as sections in the record sidebar and form editors. Get the group structure right upfront. Reorganising it later is annoying and easy to avoid.

Property Group	What Lives Here	Apply To
Contact Information	Name, email, phone, address, job title	All implementations
Company Information	Mirrored company name, website, industry	Contact object in B2B
Persona & Qualification	Persona, seniority, ICP score and tier, BANT fields	B2B implementations
Lead Scoring & Intent	Lead score, score bucket, score last updated, intent signals	All implementations

Property Group	What Lives Here	Apply To
Acquisition & Attribution	Lead source (original and latest), UTM fields, first/last touch dates	All implementations
Lifecycle Dates	Lifecycle stage, MQL date, SAL date, SQL date, Customer date	All implementations
Compliance & Consent	GDPR consent, do not contact, suppression reason and date	Required for EU / GDPR
Integration Fields	External IDs, sync timestamps, source system flags	Any integration scenario

5.3 Workflow Design — How to Avoid the Sprawl

Workflow sprawl is real and it sneaks up on you. One day you have 12 workflows, all clearly named and well-documented. Eighteen months later you have 200, nobody knows which ones are still active, and nobody wants to touch any of them in case something breaks. Here's how to prevent it:

- **One workflow per lifecycle transition. MQL logic and SQL logic do not belong in the same workflow.**
- Name every workflow with a prefix: [LCS] for lifecycle, [SCR] for scoring, [ALT] for alerts, [INT] for integration. This is not optional.
- Every workflow needs a description that explains what it does and why it exists. Write it before you activate.
- Maintain a suppression list: competitors, internal contacts, invalid emails. Enroll it in every workflow.
- Audit active workflows every six months. Kill anything that isn't needed. Zombie workflows cause real, hard-to-diagnose problems.
- Never have two workflows write to the same field unless you've explicitly designed the order of operations.

5.4 Association Labels — Use Them

Association labels (Enterprise) let you define the relationship type, not just the link. Not just "this Contact is on this Deal" — but "this Contact is the Economic Buyer on this Deal." This is one of the most underused features in HubSpot and one of the most valuable for enterprise sales motions.

Association	Labels to Define	Why It Matters
Contact → Deal	Decision Maker, Champion, Technical Evaluator, Economic Buyer, Blocker	Multi-threading visibility. You can't see deal risk without it.
Contact → Company	Primary Contact, Secondary, Historical	Avoids confusion when a contact is associated to multiple companies over time
Deal → Company	Primary Account, Partner Account	Useful for channel and partner sales motions
Contact → Ticket	Reporter, Affected User, CC	CS use case — who submitted vs. who is impacted

PART VI
Salesforce-Specific Implementation

6.1 The Lead vs. Contact Decision

This is the most consequential architecture call in any Salesforce implementation. It affects your entire funnel reporting, integration design, and workflow complexity. Make it consciously, document it, and don't revisit it every six months.

Approach	What It Means	Pros	Cons	Best Fit
Lead → Convert	Prospects start as Leads, converted to Contact + Account + Opportunity at SQL	Standard SFDC pattern; native lead assignment rules; clean separation of unqualified vs. qualified	Conversion is one-way; duplicates are common; reporting across Leads and Contacts is genuinely painful	Mid-market with high inbound lead volume and an SDR team
Contact Only	Everyone enters as a Contact; Account created at MQL or first association	Single person object; simpler reporting; matches HubSpot model if you're running both	Loses native Lead features; requires custom-built qualification tracking	Orgs migrating from HubSpot; enterprise with named accounts
Hybrid	Leads for inbound volume; direct Contact creation for outbound and named accounts	Flexible; mirrors real-world sourcing patterns	Most complex to administer; requires very clear routing rules on when each path applies	Large, mature GTM running multiple distinct motion types

6.2 Relationship Design in Salesforce

Salesforce supports Lookup and Master-Detail relationships. The distinction matters more than most new admins expect, and it's very hard to change after the fact.

Type	Child Needs Parent?	Delete Cascade?	Roll-Up Summary?	Sharing Inherited?	Use It For
Lookup	No	No	No	No	Contact → Campaign, Deal → optional secondary Contact
Master-Detail	Yes — required	Yes	Yes	From parent	Line Item → Opportunity, Activity → Case
Many-to-Many (Junction)	Both parents required	Cascades from either	Via junction	From both parents	Campaign Members, multi-contact deals

Type	Child Needs Parent?	Delete Cascade?	Roll-Up Summary?	Sharing Inherited?	Use It For
Hierarchical Lookup	No	No	No	No	Account parent hierarchy, User manager hierarchy

6.3 Record Types vs. Page Layouts — Know the Difference

Using one when you need the other is one of the most common Salesforce architecture mistakes. The confusion adds unnecessary complexity and makes the org harder to maintain.

Record Type = A Different Business Process

Use a Record Type when a category of record needs different picklist values or represents a genuinely different business process. "New Business Opportunity" vs. "Renewal Opportunity" is a valid Record Type distinction — different stages, different probabilities, different fields required. Record Types are heavyweight. Use them deliberately, not by default.

Page Layout = What Different Users See

Use a Page Layout when you just want to show different fields to different user profiles. Sales Reps don't need compliance fields. Finance doesn't need SDR qualification notes. Page Layouts handle this without Record Type complexity. Most "we need a Record Type" requests are actually Page Layout requests in disguise.

6.4 Validation Rules to Activate Before Day One

Validation rules are your last line of defence against bad data. These are the ones that should be live before a single real user touches the system:

Rule	Object	Logic (simplified)	Why It's There
Close Date in Future	Opportunity	If Stage ≠ Closed, CloseDate must be >= TODAY()	Stops the pipeline filling with overdue deals that inflate your forecast
Amount Required at Demo+	Opportunity	If Stage >= Demo, Amount cannot be null or zero	Forces AE to enter a value before advancing. You can't forecast what you can't see.
Lost Reason Required	Opportunity	If Stage = Closed Lost AND reason field is blank, block save	Win/loss analysis is worthless without consistent data here
Email Format Check	Contact / Lead	Email must match valid format regex	Stops invalid emails from entering scoring, sending, and reporting
Deal Name Convention	Opportunity	Name must include the Company Name	Enforces your naming convention without manual policing

Rule	Object	Logic (simplified)	Why It's There
DM Engaged at Commit	Opportunity	If Forecast = Commit AND DM_Engaged = false, warn	Surfaces the most common reason Commit-stage deals don't close on time

PART VII

Data Governance Framework

7.1 The Five-Layer Model

Data governance is not a one-time project. It's an ongoing practice. The model below organises it into five layers, from schema design at the bottom to reporting at the top. Each layer depends on the integrity of the layer below it. You cannot have reliable reporting on top of broken validation rules. You cannot have stable automation on top of an unstable schema. Build bottom-up.

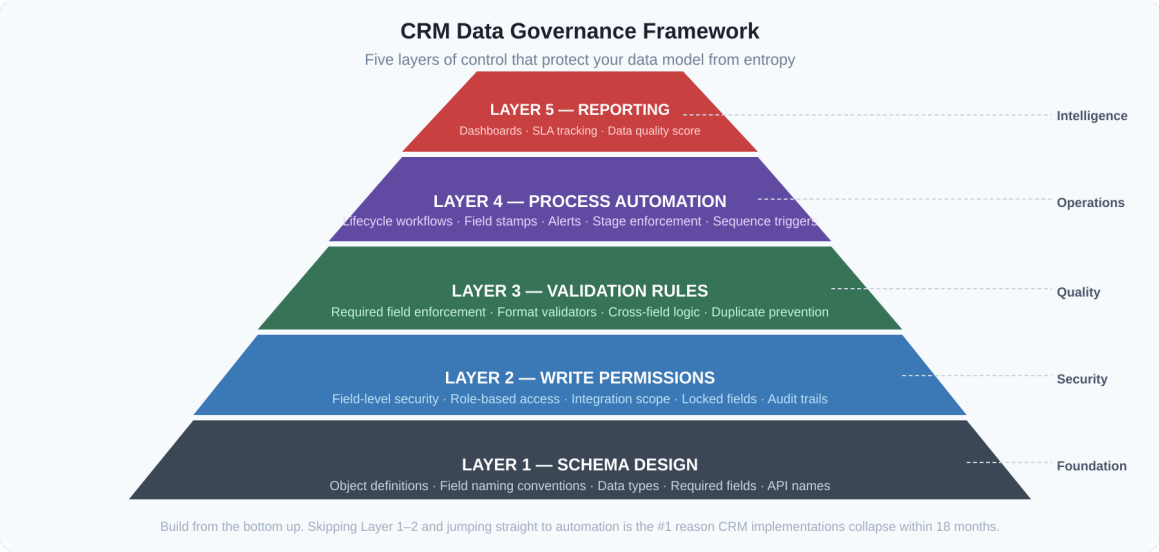

Figure 6 — Five-Layer CRM Data Governance Framework. Start at the foundation. Every layer above it is only as strong as what's below.

7.2 The Field Registry

A Field Registry is a spreadsheet with one row for every custom field in your CRM. It is the single source of truth for your schema. Update it every time a field is created, changed, or deleted.

This sounds like admin overhead. It isn't. It's the first document that disappears when your RevOps team turns over, and the first thing a new admin wishes had existed. Build it once. Keep it current.

At minimum, your Field Registry should contain:

- API name and display label
- Object the field lives on
- Data type and allowed values (full picklist for dropdown fields)
- Field owner and write permissions
- How and when the field is populated
- Which reports, dashboards, or workflows depend on it
- Date created and who created it
- Last modified date and the reason it was changed
- Status: Active, Deprecated, or Scheduled for Removal

7.3 Data Quality Scoring

You can't fix what you don't track. A Data Quality Score is a calculated metric — updated weekly — that measures the health of your CRM data. Build a dashboard for it. Make it visible. When it drops, investigate before the effects reach your pipeline reporting and automation.

Dimension	What to Measure	Target	Red Flag
Completeness	% of required fields populated on active records	≥ 95%	< 85%
Lead Source Coverage	% of contacts with a valid, non-null lead source	≥ 98%	< 90%
Lifecycle Stage Coverage	% of contacts with a defined stage (not Unknown or blank)	≥ 99%	< 95%
Deal Stagnation	% of open deals with no stage movement in 30+ days	< 10%	> 20%
Activity Coverage	% of open deals with an activity logged in the last 14 days	≥ 80%	< 60%
Duplicate Rate	% of contacts matching another contact on email or name+company	< 1%	> 3%
Missing Loss Reasons	% of Closed Lost deals with no close reason entered	< 2%	> 10%
Overdue Close Dates	% of open deals where close date is in the past	< 5%	> 15%

7.4 Change Management

The model will change. New fields will be requested. Stages will be redefined. Pipelines will be restructured. How you manage those changes determines whether the system stays coherent or slowly turns into one nobody trusts.

The Change Request Process

1. Request submitted in writing — including the proposed field name, its purpose, the field owner, and which reports will use it.
2. RevOps reviews within 5 business days — checking for duplication, naming compliance, and downstream impact on automations and integrations.
3. Impact assessment documented — which automations, reports, and integrations are affected?
4. Approved or rejected in writing. Rejections include the reason and any alternatives.
5. If approved: field registry updated first, then the field is created in the platform.
6. Post-creation: all affected automations and reports validated within 3 business days.

> ⚠ **"Can You Just Add a Field?"**

"Can you just add a field for this?" Every RevOps person has heard it. Every field added without a process creates future maintenance, potential automation conflicts, and reporting ambiguity. The answer to "can you just add a field" should always be: "Let's document what it's for first." If that slows someone down, good. It's supposed to.

PART VIII

What Not to Build

Six patterns that kill CRM implementations. They're all common. They all feel reasonable when you're in the middle of them. They all cause pain later. Read this with a critical eye toward what you're currently building.

Anti-Pattern 1: The Vague Lead Source

"Website" as the lead source for 60% of your contacts. It happens because nobody defined what website means before go-live. Organic search? Direct traffic? A demo request? A content download? All of those are different channels with different economics — and you can't differentiate them after the fact.

Fix it before launch: define your lead source picklist exhaustively, map every entry point to a specific value, and build automations that capture UTM parameters at form submission and stamp the right value at that moment. Force specificity. "Website" is not a value.

Anti-Pattern 2: Lifecycle Stages Nobody Agreed On

You launch. Marketing calls something an MQL. Sales calls the same record garbage. Both are right by their own definition, because there is no shared definition. This happens in every organisation that doesn't run an alignment workshop before go-live.

The fix is a two-hour meeting — Marketing, Sales leadership, and RevOps — where you go stage by stage and write down the entry criteria for each one. Not discuss them. Write them down. Get sign-off. Put them in this document. Enforce them with automation. No launch until this is done.

Anti-Pattern 3: All the Data Living in Notes

The sales leader who, when asked about pipeline, says "just look at the notes." Every qualification detail, every BANT answer, every next step is in a text block on the contact or deal record. You cannot report on notes. You cannot build automations on notes. You cannot forecast from notes.

Notes are for context that doesn't fit in a structured field. Everything that needs to show up in a report, drive an automation, or inform a forecast needs to be in a structured field. Identify the 10–15 qualification data points that matter. Build fields for them. Enforce that they're populated at each stage gate.

Anti-Pattern 4: One Pipeline for Everything

New business, renewals, expansions, and partner deals all in the same pipeline. This is not a pipeline — it's a spreadsheet with a probability column. These motions have different stages, different average cycle lengths, different win rates, and different forecasting models. Mixing them means every pipeline metric is a meaningless average.

Separate pipelines from day one: New Business, Renewal / Expansion, Partner / Channel. The extra overhead is worth it. The reporting clarity alone pays for itself in the first quarter review.

Anti-Pattern 5: The Integration That Writes to Everything

You connect your CRM to marketing automation. The integration is configured as "bidirectional sync, all fields." Two weeks later, a lead score update from marketing overwrites the close date on an opportunity because the field mapping was wrong. This happens more often than you'd expect.

The rule: every integration needs a documented field-level sync configuration. For each field, define the sync direction and the conflict resolution rule. Start with "sync nothing" and add fields deliberately. Never default to "sync everything."

Anti-Pattern 6: Automating Before Testing

The workflow that sends a congratulations email when a deal closes Closed Won is live. Except the deal stage picklist has "Closed - Won" (with a hyphen) because someone added it manually six months ago, and the workflow checks for "Closed Won." So the email never fires, nobody notices, and twelve deals close without triggering the onboarding sequence.

Test every automation in a sandbox with real-world edge cases before it goes live. Maintain a test contact and deal you can push through every stage transition manually, once a month. Automation fails silently. You have to go looking for the failures.

Anti-Pattern	The Real Cost	How to Prevent It
Vague lead source values	Attribution is useless. Budget decisions are made blind.	Define the picklist before go-live. Map every entry point to a specific value.
Undefined lifecycle stages	Marketing vs. Sales conflict at every QBR.	Written definitions. Sign-off from both teams. No launch without it.
Everything in notes	Can't report, automate, or forecast on it.	Structured fields for structured data. Notes for context only.
Single mega-pipeline	Pipeline metrics are meaningless averages.	Separate pipelines by sales motion from day one.
Unscoped integrations	Data corruption, overwrites, lost auditability.	Document every field sync direction and conflict rule before activating.
Untested automations	Silent failures, missed handoffs, broken sequences.	Sandbox testing with edge cases. Monthly validation of critical workflows.

PART IX

Pre-Build Checklist

Don't touch the platform until everything on this list is done. Print it. Write initials and dates next to each item. The ten hours this takes will save you a hundred hours of rework later — and that's a conservative estimate.

Phase 1 — Alignment & Decisions

#	Item	Owner	✓
1	Platform confirmed: HubSpot, Salesforce, or both	Leadership + RevOps	☐
2	CRM Admin / RevOps lead named and given authority to make architecture decisions	Leadership	☐
3	GTM motion documented: inbound, outbound, PLG, channel, or a specific hybrid	GTM Lead	☐
4	ICP definition agreed in writing: industry, size, geography, buying role	Sales + Marketing	☐
5	Lifecycle stage definitions written, reviewed, and signed off by Sales and Marketing	RevOps	☐
6	Deal pipeline stages defined with entry/exit criteria and default probabilities	Sales Leadership + RevOps	☐
7	Lead routing rules defined: territory, round-robin, or account-based	RevOps + Sales Ops	☐
8	Lead source picklist values defined and mapped to every inbound entry point	RevOps + Marketing	☐
9	SLA targets agreed for each lifecycle handoff: MQL→SAL and SAL→SQL	Sales + Marketing Leadership	☐
10	Integration scope confirmed: which systems connect to the CRM and in which direction	RevOps + Engineering	☐

Phase 2 — Data Model Design

#	Item	Owner	✓
11	Object inventory complete: all objects listed with their purpose and relationships	RevOps	☐
12	Object relationship diagram drawn and reviewed with key stakeholders	RevOps	☐

#	Item	Owner	✓
13	Field naming convention documented, agreed, and shared with all admins	RevOps	☐
14	Custom field list compiled per object with full spec: type, owner, validation, use case	RevOps	☐
15	Field ownership matrix complete — write permissions defined per field per team	RevOps	☐
16	Field registry document created (a spreadsheet is fine, but it must exist)	RevOps	☐
17	Picklist values documented for every dropdown and select field	RevOps + relevant teams	☐
18	Required fields identified per object and per stage gate	RevOps + Sales + Marketing	☐
19	Validation rules designed — logic written down before anything is built	RevOps	☐
20	Duplicate management strategy defined: merge rules and match criteria	RevOps	☐

Phase 3 — Platform Build

#	Item	Owner	✓
21	User roles and profiles configured before any users are added to the system	CRM Admin	☐
22	Field-level security applied per the ownership matrix — write restrictions enforced	CRM Admin	☐
23	All objects configured per the object inventory	CRM Admin	☐
24	All custom fields created per spec, named per convention, documented in registry	CRM Admin	☐
25	Validation rules activated and tested with real-world edge cases	CRM Admin	☐
26	Lifecycle stage workflow built and tested end-to-end in sandbox	CRM Admin	☐
27	Lead scoring model configured and validated against historical data where available	RevOps + Marketing	☐
28	All pipelines configured with correct stages — one pipeline per sales motion	CRM Admin	☐
29	Lead assignment and routing rules activated and tested with representative data	CRM Admin + Sales Ops	☐
30	Integration connections established with scope-limited, documented field mappings	RevOps + Engineering	☐

Phase 4 — Go-Live Readiness

#	Item	Owner	✓
31	User training done — lifecycle stage definitions communicated to every GTM team	RevOps + Enablement	☐
32	Data import validated: no duplicate contacts, correct lifecycle stages, clean lead sources	CRM Admin	☐
33	Critical automations tested with real-world records in staging environment	CRM Admin	☐
34	Core dashboards built: pipeline, funnel conversion, activity coverage, data quality	RevOps	☐
35	Rollback plan documented for critical issues in the first 30 days	RevOps + IT	☐
36	30-60-90 day post-launch review schedule set with leadership	RevOps	☐

Appendix A — Lead Source Picklist

Use this as your starting point. Adjust for your channels, but maintain this level of specificity. Never accept "Other" as a permanent value — if it's being populated frequently, you have unmapped channels that deserve their own entries.

Value	Definition	Attribution Signal
Paid Search — Google	Google Ads (Search or PMax) click-through	utm_medium=cpc + utm_source=google
Paid Search — Bing	Microsoft Ads click-through	utm_medium=cpc + utm_source=bing
Paid Social — LinkedIn	LinkedIn Ads, any format	utm_source=linkedin + utm_medium=paid-social
Paid Social — Meta	Facebook or Instagram Ads	utm_source=facebook OR instagram
Organic Search	Unpaid search engine click-through	utm_medium=organic OR referrer contains search engine with no utm_medium
Organic Social	Unpaid social post click-through	utm_medium=social + no paid campaign attached
Direct / Dark Social	Typed URL, bookmarked, or dark social (Slack, newsletters, etc.)	No utm, no referrer, or utm_medium=direct
Content / SEO Asset	Blog post or organic content download specifically	Referrer = blog subdomain + organic medium
Email — Outbound	SDR / AE cold email sequence click-through	utm_medium=email + utm_source=outbound-sequence
Email — Marketing	Marketing campaign email click-through	utm_medium=email + utm_source=hubspot or marketo
Webinar / Virtual Event	Registration or attendance at a virtual event	utm_medium=webinar OR event registration form
Event — In Person	Trade show, conference, or field event	Manually set by SDR or via event upload
Partner / Channel	Referred by a technology or channel partner	Partner referral form or manually set by SDR
Customer Referral	Referred directly by an existing customer	Referral form or manually set by AE or CS
Product-Led (PLG)	Self-serve trial or freemium sign-up	utm_medium=product OR product signup form
Review Site	G2, Capterra, Trustpilot, or similar click-through	utm_source=g2 OR capterra, etc.

Value	Definition	Attribution Signal
Outbound — Cold Call	SDR cold call that resulted in a qualified conversation	Manually set by SDR at time of call
Web Chat	Drift, Intercom, or similar real-time chat conversation	Chat integration stamp or utm_medium=chat

Appendix B — Dashboard Framework

Six dashboards every RevOps team should have active from the day they go live. Each one answers a specific question for a specific audience.

Dashboard	Audience	Key Metrics	Cadence
Pipeline & Forecast	Sales Leadership, CEO, CFO	Pipeline by stage, weighted forecast, forecast vs. target, deal velocity, pipeline coverage ratio	Daily
Funnel & Conversion	Marketing, SDR Leadership, RevOps	Volume by stage, MQL→SAL→SQL rates, time-in-stage by cohort, funnel velocity trends	Weekly
Attribution & Channel	Marketing Leadership, CMO	Revenue by lead source, CAC by channel, pipeline created by campaign, ROI by channel	Weekly
Sales Activity	Sales Managers, SDR Managers	Activities per rep, deal activity coverage, meetings booked vs. target, connect rates	Daily
Data Quality	RevOps, CRM Admin	Completeness by field, lifecycle coverage, duplicate rate, overdue deals, missing loss reasons	Weekly
Customer Health	CS Leadership, RevOps	NPS/CSAT trends, product adoption signals, renewal pipeline, churn risk flags, expansion pipeline	Weekly

Appendix C — Qualification Frameworks

Choose one and use it consistently. A hybrid framework that mixes two methodologies isn't a framework. It's inconsistency with a brand name on it.

Framework	Stands For	Best For	Watch Out For
BANT	Budget, Authority, Need, Timeline	SMB, transactional, short-cycle deals under 30 days	Too simple for enterprise. Leads to checkbox qualification rather than genuine discovery.
MEDDIC	Metrics, Economic Buyer, Decision Criteria, Decision Process, Identify Pain, Champion	Enterprise, complex, multi-stakeholder deals	High overhead to fill out. Requires disciplined coaching to execute well in practice.
MEDDPICC	MEDDIC + Paper Process + Competition	Enterprise with long procurement cycles and real competitive dynamics	The most complete framework — but genuinely hard to execute without a strong sales culture behind it.
SPICED	Situation, Pain, Impact, Critical Event, Decision	Mid-market, consultative selling, CS-led expansion	Newer. Less tooling support. Requires enablement investment to land well.
ANUM	Authority, Need, Urgency, Money	Inbound-led, product-led, shorter-cycle deals	Misses process complexity entirely. Poor fit for anything involving multiple stakeholders.

Appendix D — Integration Field Sync Template

Complete this for every field that syncs between your CRM and a connected system. Doing this before activating any integration has saved more implementations than any other single practice in this document.

Field (CRM)	System	Field (External)	Direction	Conflict Rule	Null Handling	Notes
email	Marketo	Email Address	Bidirectional	CRM wins	Never overwrite with null	Primary identifier — treat as read-only in Marketo
lifecycle_stage	Marketo	Lead Status	CRM → Marketo only	CRM always wins	Map to equivalent Marketo status	Never let Marketo write lifecycle stage back to CRM
lead_score	Marketo	Lead Score	Marketo → CRM only	External always wins	Allow null — score not yet set	RevOps owns the scoring logic in Marketo
deal_amount	NetSuite / Billing	Order Value	Billing → CRM (post-close only)	Billing wins after close	Do not sync if null	Only sync on Closed Won. Never overwrite open deals.
company_arr	Chargebee / Stripe	MRR × 12	Billing → CRM	Billing wins	Sync only if > 0	Recalculate nightly via scheduled job
[Your field]	[Your system]	[Map it here]	[Define the direction]	[Define the rule]	[Define the handling]	[Document the logic]

Questions, feedback, or want to talk through your CRM build?
Contact us: **signals@revopsbrief.com** · www.revopsbrief.com