ULDA Enterprise Lead-360 — Design Specification

Ingestion Pipeline · PostgreSQL Schema · Complete Data Mapping · Automatic Activity Tracking

Data Architecture — InternetSP / ULDA v6.2

2026-07-16

0. Preservation Guarantee

Every one of the **36 business fields** you listed is preserved **exactly as named** and is retrievable for every lead from a single Lead-360 record. Nothing below removes, renames, or replaces any of them. This document adds recommended columns, the standardized ingestion pipeline, the physical schema, a field-by-field data map, and the automatic activity-tracking mechanism — all **additive** to the running ULDA v6.1 platform and implementable with **zero downtime**.

The core principle (established in the architecture report): **fields are stored normalized across four domains, then assembled into one flat Lead-360 row by the Serving layer.** You never lose the “single lead view” — it is produced by a serving object, not a base table.

---

1. Recommended Additional Columns (nothing removed — only added)

Your 36 fields cover identity, source, enrichment, journey, and deal. For a **telecom lead-gen / sales / AI / analytics** platform, I recommend adding the following, grouped by purpose. All are optional and additive.

1.1 Compliance & TCPA (highest priority — you run SMS + a dialer)

COLUMN	WHY
`dnc_flag` (per number)	Federal/state DNC suppression before dial/text
`litigator_flag`	Known TCPA litigators — hard suppress
`consent_status`, `consent_channel`, `consent_at`	Provable express consent per channel
`timezone`	Quiet-hours calling law (derive from state/ZIP)

COLUMN	WHY
`opt_out_sms`, `opt_out_email`	STOP / unsubscribe honoring

1.2 Contactability & phone/email intelligence

COLUMN	WHY
`phone_type` (mobile/landline/voip)	Dial strategy + mobile-only SMS
`phone_carrier`, `line_valid` (IPQS)	Deliverability, fraud, portability
`phone_tier` (T1–T4 best-number rank)	Which of many numbers to dial first
`email_status` (valid/catch-all/invalid), `email_deliverable`	Bounce avoidance

1.3 Telecom serviceability (your product edge — extends “Fiber Availability”)

COLUMN	WHY
`fiber_available_att / _frontier / _kinetic / _spectrum / _brightspeed`	Per-carrier yes/no (your named carriers)
`max_download_mbps`, `technology` (fiber/copper/FW)	Plan match
`incumbent_carrier`, `address_type` (SFU/MDU/business)	Competitive + install feasibility
`coverage_confidence`, `serviceability_checked_at`	Trust + freshness
`county`, `census_tract`, `lat`, `lng`	Coverage joins, territory, routing

1.4 Scoring, AI & pipeline

COLUMN	WHY
`lead_score`, `lead_tier`	Prioritization (live)
`intent_signals` (jsonb), `propensity_to_convert`	ML-ready features
`stage`, `status`, `disposition`	Pipeline state
`owner_agent_id`, `team_id`, `territory_id`	Assignment (distinct from “agents interacted with”)
`next_follow_up_at`, `attempts_per_channel` (jsonb), `last_attempt_at`	Cadence management

1.5 Deal / revenue (extends “Order details”)

COLUMN	WHY
`mrr`, `term_months`, `install_date`, `order_status`	Revenue + fulfilment
`sub_dealer`, `poe_id`, `commission`	FiberVolt/commission reconciliation

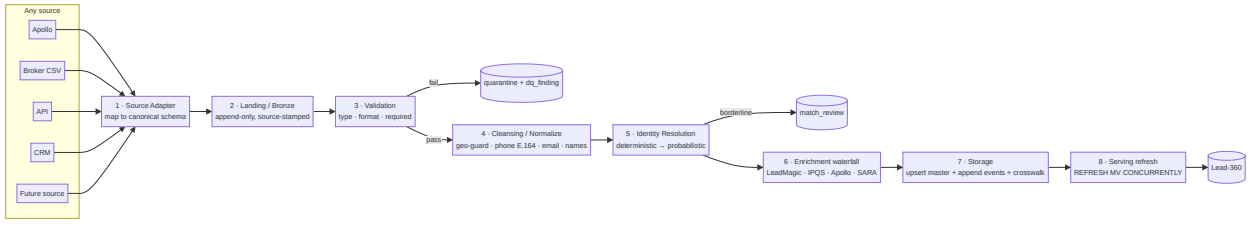
1.6 Identity, quality & engagement

COLUMN	WHY
`source_count`, `merged_from`	Dedup transparency
`first_seen_at`, `updated_at`, `data_quality_score`, `completeness`	Governance & freshness
`last_reply_at`, `total_replies`, `open_count`, `last_touched_by`	Engagement signals

2. End-to-End Data Ingestion Pipeline

Goal: every lead — from Apollo, brokers, CSV, APIs, CRM, or any future source — enters through **one standardized process**. No source writes to the master directly.

2.1 Pipeline diagram



2.2 The eight stages

1. **Source Adapter** — a per-source config maps arbitrary columns to a **canonical staging schema**. Adding a new source = new adapter, no pipeline change. (Extends the existing geo-guard/ `upsertLeads` chokepoint into a formal contract.)
2. **Landing / Bronze** — raw rows land append-only in `ulda_stage.lead_inbox`, stamped with `source_system, source_id, source_type, batch_id, ingested_at, row_hash`. Immutable audit of exactly what arrived.
3. **Validation** — type/format/required checks (RFC email, E.164 phone, 5-digit ZIP, USPS state). Failures → `ulda_gov.dq_finding` + quarantine (never silently dropped).
4. **Cleansing / Normalization** — the deployed **geo-normalize guard** (state/ZIP), phone → E.164, email lowercase/trim, name casing, in-batch dedup by `row_hash`.
5. **Identity Resolution** — deterministic match on normalized email/phone → existing `master_lead_id`; else probabilistic (trigram + metaphone + pgvector) → borderline to `match_review`; else mint a new immutable ID.
6. **Enrichment waterfall** — call providers in priority order only for gaps (phone via LeadMagic/IPQS, firmographics via Apollo, **fiber serviceability via SARA/Kinetic/public APIs**); each provider response is written as an **`enrichment` event**.
7. **Storage** — idempotent upsert of golden attributes (survivorship) + append events (`interaction`, `enrichment`, `order`) + write `source_crosswalk`. Dual-write trigger → `change_outbox` → drain.
8. **Serving refresh** — drain triggers `REFRESH MATERIALIZED VIEW CONCURRENTLY` on `mv_lead_360`, journey, score → Lead-360 reflects the new lead.

2.3 One entry point

All sources call **one function** — `ulda_stage.ingest_batch(source_config, rows)` — which runs stages 2–8. This is the single standardized process: consistent validation, dedup, resolution, enrichment, and storage for every lead, forever.

---

3. PostgreSQL Design (schemas, tables, indexes, constraints, MVs)

3.1 Schemas

```
leads_db
├── ulda_stage    – landing / Bronze + ingestion functions   (NEW)
├── ulda_master   – golden entities (Master Data)
├── ulda_events   – append-only events (partitioned)
├── ulda_ref      – reference (serviceability)
├── ulda_serve    – Lead-360, journey, score, embeddings (MV + views)
└── ulda_gov      – DQ, consent, lineage, match-review, audit
```

3.2 Key new/updated tables (DDL)

```
-- LANDING (Bronze) – every raw row, immutable
CREATE TABLE ulda_stage.lead_inbox (
  inbox_id      bigint GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
  source_system text NOT NULL, source_id text, source_type text,
  batch_id      uuid NOT NULL, row_hash text NOT NULL,
  raw           jsonb NOT NULL,            -- exactly what arrived
  status        text NOT NULL DEFAULT 'received',  --
received|validated|resolved|stored|quarantined
  ingested_at   timestamptz NOT NULL DEFAULT now(),
  UNIQUE (source_system, row_hash)          -- idempotent re-ingest
);

-- INTERACTION – append-only, monthly partitions (auto activity capture target)
CREATE TABLE ulda_master.interaction (
  interaction_id bigint GENERATED BY DEFAULT AS IDENTITY,
  master_lead_id uuid NOT NULL REFERENCES
ulda_master.identity_registry(master_lead_id),
```

```
  channel_id int REFERENCES ulda_master.channel(channel_id),
  agent_id uuid REFERENCES ulda_master.agent(agent_id),
  campaign_id text REFERENCES ulda_master.campaign(campaign_id),
  direction text, interaction_type text, disposition text,
  agent_login text, software_type text,      -- dialer|warroom|sms_platform
  occurred_at timestamptz NOT NULL, metadata jsonb,
  PRIMARY KEY (interaction_id, occurred_at)
) PARTITION BY RANGE (occurred_at);

-- ENRICHMENT — append-only history (last-enrichment derived from it)
CREATE TABLE ulda_master.enrichment (
  enrichment_id bigint GENERATED BY DEFAULT AS IDENTITY,
  master_lead_id uuid NOT NULL REFERENCES
ulda_master.identity_registry(master_lead_id),
  enrichment_type text NOT NULL, provider text, source_type text, status text,
  occurred_at timestamptz NOT NULL, detail jsonb,
  PRIMARY KEY (enrichment_id, occurred_at)
) PARTITION BY RANGE (occurred_at);
```

3.3 Indexes

TABLE	INDEX	PURPOSE
interaction	`(master_lead_id, occurred_at DESC)`	per-lead journey + last
interaction	`(channel_id, occurred_at)` ; BRIN `(occurred_at)`	channel analytics; cheap time scans
enrichment	`(master_lead_id, occurred_at DESC)`	last-enrichment derive
source_crosswalk	`(source_system, source_id)` UNIQUE	dedup + source lookup
address_serviceability	`(address_id)` , `(carrier, fiber_available)`	fiber lookups
lead_inbox	`(status)` , `(batch_id)`	pipeline progress
person/company	GIN trigram on names	fuzzy match/search

3.4 Constraints

- **PK:** surrogate everywhere (`uuid` for entities, `bigint identity` +time for events).
- **FK:** every event/child → `identity_registry(master_lead_id)` ; dimensions enforced.
- **UNIQUE:** `(source_system, source_id)` crosswalk; `(address_id, carrier, checked_at)` serviceability; `(source_system, row_hash)` inbox.
- **CHECK:** email regex, E.164 phone, USPS state, 5-digit ZIP at validation.
- **Golden records never hard-delete** (`ON DELETE RESTRICT` ; merges via `merged_into`).

3.5 Materialized views

`mv_lead_360` (core profile), `mv_lead_master_flat` (**all 36 fields + additions**), `v_lead_journey_flat` (channel pivot), `mv_lead_score` , `mv_analytics_*` , `lead_embedding` (pgvector). All refreshed `CONCURRENTLY` .

---

4. Complete Data Mapping (all 36 business fields)

Legend — Domain: M=Master · E=Event · R=Reference · S=Serving-derived.

BUSINESS FIELD	DOMAIN	ORIGINATES FROM	STORED IN (PHYSICAL)
Unique Lead ID	M	Identity resolution	`identity_registry.master_l`
First and Last Name	M	source / enrichment	`person.first_name/last_nam`
Email	M	source / enrichment	`contact_point` (email) → gold
Direct Number	M	source / enrichment	`contact_point` (phone, person
Company	M	source / enrichment	`company.legal_name` (via employment)
Street	M	source / geo-recovery	`address.line1`
City	M	source	`address.city`

BUSINESS FIELD	DOMAIN	ORIGINATES FROM	STORED IN (PHYSICAL)
State	M	source	`address.state` (USPS)
ZIP	M	source / recovery	`address.postal_code`
Fiber Availability (AT&T, Kinetic, etc)	R	SARA/Kinetic/public API	`address_serviceability(car fiber_available)`
Company Mobile / Phone / Landline Number	M	source / enrichment	`contact_point` (typed phones)
Source ID	M	source	`source_crosswalk.source_id`
Source Name (Apollo, Tanveer)	M	source	`source_crosswalk.source_sy`
Source Type (Database, Private Broker)	M	source adapter	`source_crosswalk.source_ty`
Date Added	M	ingest	`identity_registry.created_`
Last Enrichment Date	E→S	enrichment events	`enrichment.occurred_at`
Last Enrichment Type	E→S	enrichment events	`enrichment.enrichment_type`
Last Enrichment Source	E→S	enrichment events	`enrichment.provider`
First Channel Interaction	E→S	interaction	`interaction` (rn=1)
First Channel initial activity date	E→S	interaction	`interaction.occurred_at` (rn
First Channel Campaign Name	E→S	interaction⋈campaign	`campaign.name` (rn=1)
Second Channel Interaction/date/campaign	E→S	interaction (rn=2)	`interaction` / `campaign`
Third Channel Interaction/date/campaign	E→S	interaction (rn=3)	`interaction` / `campaign`
Fourth Channel Interaction/date/campaign	E→S	interaction (rn=4)	`interaction` / `campaign`

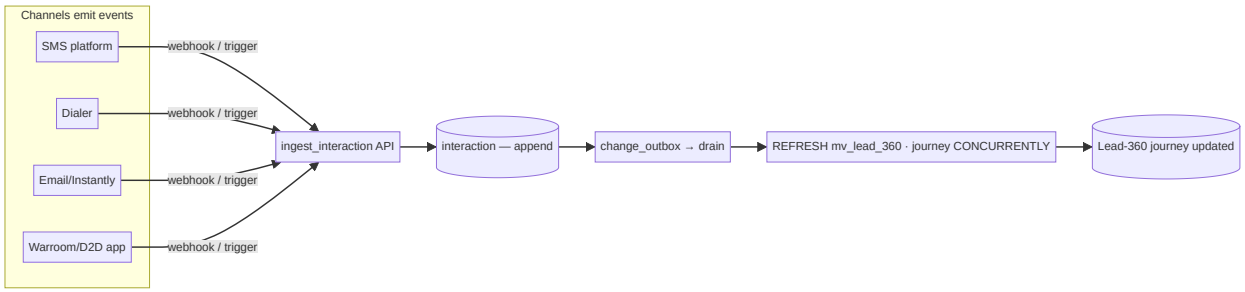
BUSINESS FIELD	DOMAIN	ORIGINATES FROM	STORED IN (PHYSICAL)
Last interaction date	E→S	interaction	`interaction.occurred_at`
Last interaction channel	E→S	interaction⋈channel	`channel.name` (latest)
Agents interacted with (chrono, login/software)	E→S	interaction	`interaction.agent_id, agent_login, software_type`
Last Agent interaction	E→S	interaction	`interaction.agent_id` (latest
Product offered	E→S	opportunity/order	`product.name` via `opportunity` / `order_line`
Order details	E→S	order	`order` + `order_line`

Reading: M fields return immediately (identity/geo); R fills after a serviceability check; E→S fills as activity is captured. Every field lands in one row: `ulda_serve.mv_lead_master_flat`.

5. Automatic Activity Tracking

Goal: every interaction (channel, campaign, software, agent, timestamp, disposition) is captured **automatically** and reflected in the journey + Lead-360 — no manual logging.

5.1 Mechanism



5.2 Capture points (one contract, every channel)

Each platform calls `ulda_stage.ingest_interaction(payload)` on every event: - **SMS platform** → on send/reply: `channel=SMS, software_type=sms_platform, campaign, disposition` - **Dialer** → on call/disposition: `channel=Voice, software_type=dialer, agent_login, disposition` - **Email/Instantly** → on send/open/reply: `channel=Email, software_type=instantly, campaign` - **Warroom / D2D** → on knock/booking: `channel=Door/Web, software_type=warroom, agent_login`

The function resolves the lead to its `master_lead_id`, appends one `interaction` row (with agent login + software type — satisfying “agents interacted with, chronological via agent login / software type”), and the drain refreshes the journey. **First/Second/Third/Fourth/Last channel and agent columns update themselves.**

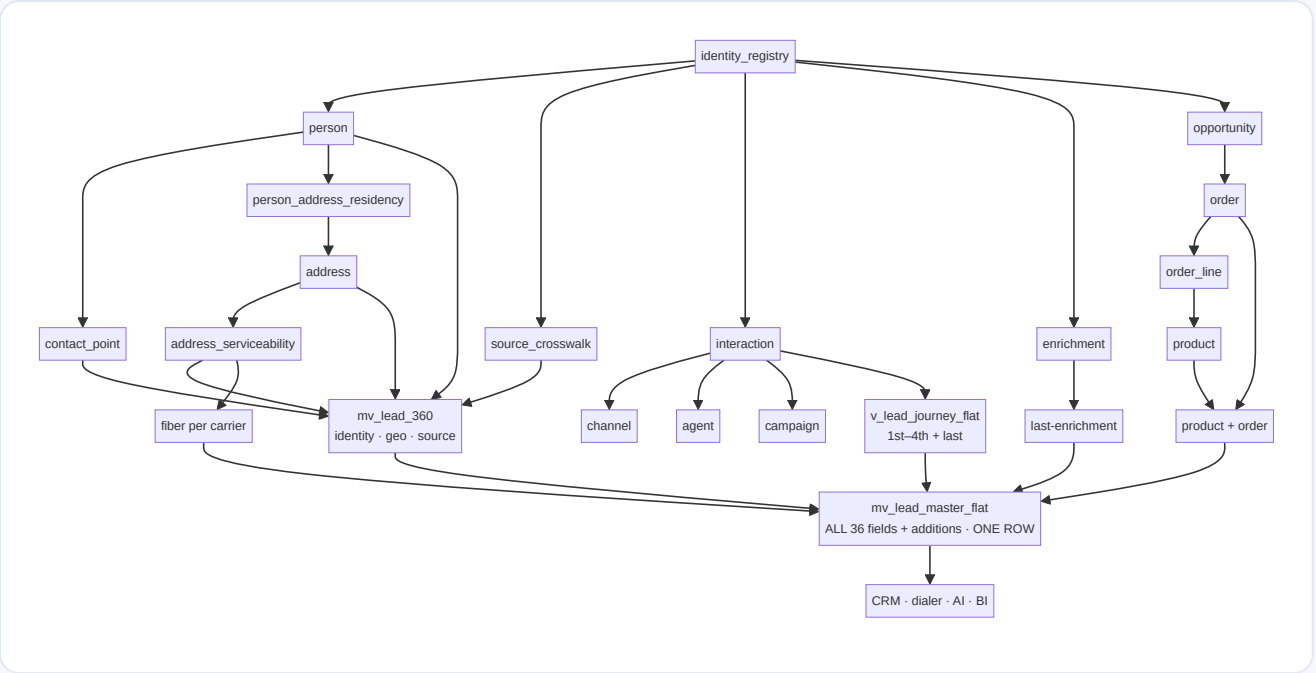
5.3 Why automatic

Because the journey columns are **derived** from `interaction`, a single append instantly changes “last interaction”, “last agent”, and — once it’s the 1st–4th touch — the corresponding channel slot. No column to update, no app logic to maintain: capture the event once, the Lead-360 view recomputes.

---

6. Enterprise Architecture, ERD & Lead-360 Assembly

6.1 How the normalized tables produce ONE Lead-360 record



`mv_lead_master_flat` LEFT JOINs the core profile to the derived journey/enrichment/order/fiber pieces on `master_lead_id`. Because every domain shares that one key, the assembly is a clean set of joins — defined once. Applications run `SELECT * FROM ulda_serve.mv_lead_master_flat WHERE master_lead_id = $1` and receive the complete, single Lead-360 record with all business fields intact.

6.2 ERD & full schema

See the companion Enterprise Architecture Report (`enterprise-architecture.html`) for the full 7-diagram set (System, Physical, Logical, Domain, ERD, Data Flow, Lead-360) and per-domain/per-component detail.

---

7. Implementation Plan (additive · zero-downtime)

STEP	ACTION	RISK
1	Create `ulda_stage` + `lead_inbox` + `ingest_batch()` / `ingest_interaction()`	new objects — none

STEP	ACTION	RISK
2	Convert `interaction` / `enrichment` to partitioned append-only (empty now)	pure DDL — none
3	Add recommended columns (§1) to the appropriate domain tables	`ADD COLUMN` — instant
4	Build `v_lead_journey_flat` + `mv_lead_master_flat` (all 36 + additions)	derived views — none
5	Wire channel webhooks/triggers → `ingest_interaction()` (auto tracking)	app config — reversible
6	Backfill `interaction` from `disposition_event` + warroom activity	read-only source; batched

Steps 1–4 are pure additive DDL/views (zero-downtime, reversible by `DROP`). Steps 5–6 populate the timeline. Every business field remains present and correctly named throughout.