ENTITÄTENMODELL HiveGrid 2.0

Stand: 15.06.2026 · **Status:** ✅ **APPROVED & LOCKED** (UX/David — Projekt-zentrisches Modell, AD-20) · **Quelle:** Davids konsolidiertes CRM/Pipeline-Briefing (`HiveGrid_CRM_Datenlogik_UX_Entscheidungen.md`) für den Kern + Modul-Specs (`_bmad-output/specs/` , Block A) für die unveränderten Domänen.

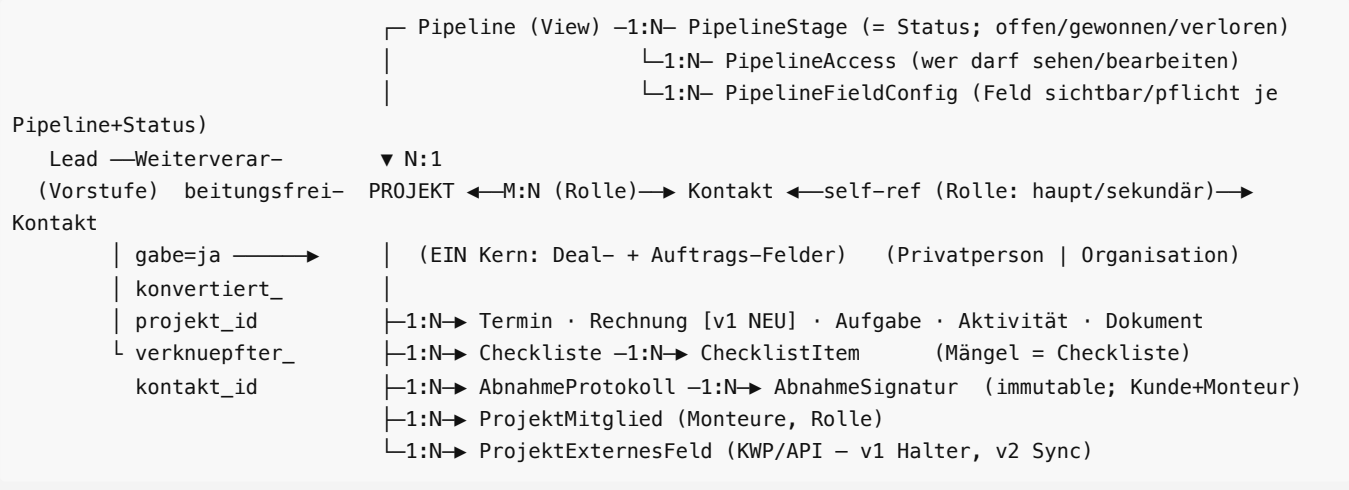
> ✅ ***Projekt-zentrisches Modell (AD-20) — verbindlich.*** *Der CRM/Pipeline-Kern: EIN* `Projekt` *-Kern + konfigurierbare* ***Pipeline-Views*** *+* ***Pipeline-Feld-Config-Layer****;* `Kontakt` *-Oberbegriff (Kontakttyp/-status/-art);* ***Lead → Projekt****;* ***Rechnung*** *als Sub-Entity (v1);* ***Abnahme*** *als Checklisten/Protokoll-Sub-Area. „Deal" und „Auftrag" sind* ***Pipeline-Typen, keine eigenen Entitäten****. Maschinenlesbar:* `entity-model.dbml` *(ER-Diagramm:* `entity-model.svg` *). Laienfassung:* `ENTITY-MODELL-ERKLAERUNG.pdf` *.* ***Hinweis zu den Domänen 6–13 unten*** *(Auth, Files, Apps, Suche, Automation/KI, Kommunikation, Kalender/Forms, HR, Admin, Billing, Onboarding): vom Pivot unberührt — wo dort* `Deal` */* `Project` *steht, ist der* `Projekt` *-Kern gemeint, wo* `Contact` */* `Company` *steht, der* `Kontakt` *-Oberbegriff.*

Lies das zuerst (3 Architektur-Grundregeln)

1. **First-Class-Tabellen + Konfig-Layer, KEIN Entity-Builder (AD-2, revidiert).** Jede Entität ist eine typisierte Prisma-Tabelle mit echten Spalten. Der `Projekt` -Kern ist EINE typisierte Tabelle (Superset der Felder); Konfigurierbarkeit läuft über `CustomFieldDefinition` + `CustomFieldValue` **und** den **Pipeline-Feld-Config-Layer** (`PipelineFieldConfig` : Sichtbarkeit/Pflicht je Pipeline+Status). Der Boilerplate- `entityBuilder.prisma` (Entity/Row/RowValue) bleibt **entfernt** — kein generisches Row/RowValue-Modell.
2. **RLS auf allem Tenant-bezogenen (AD-1).** Fast jede Tabelle hat `tenant_id` + Postgres-Row-Level-Security. Ausnahmen: globale Definitions-Tabellen (App, SubscriptionProduct, FeatureFlag, Onboarding-Templates, Subprocessor) und die Plattform-Admin-Welt (eigene Trennung).
3. **Polymorphe Verknüpfung statt harter FKs, wo es modulübergreifend ist.** `FileEntityLink` , `Activity` , `SearchIndexEntry` , `Task.linked_entity` nutzen `entity_type` + `entity_id` (kein FK), damit jedes Modul andocken kann, ohne das Schema zu ändern.

Legende: `[v1]` = Launch-Scope · `[v2]` = nach Launch (markiert, nicht weggelassen) · Flags: `[RLS]` `[Soft-Delete]` (archived_at) `[polymorph]` `[verschlüsselt]` `[append-only]`

Die Kern-Kette (das Herzstück, hier zählt jede Relation)



Merksätze: **Es gibt nur einen Arbeitsdatensatz — das Projekt.** „Deal" und „Auftrag" sind Pipelines (Views), keine Entitäten — ein Projekt wechselt nur die Pipeline, es entsteht kein zweiter Datensatz und kein Won-Handoff. `Status` **ist immer pipeline-intern** (kein globaler Projektstatus). **Lead → Projekt** bei Weiterverarbeitungsfreigabe; der Kontakt wird separat verknüpft (kein Promotion-Modal mehr). **Kontakt ist Oberbegriff** (Privatperson/Organisation); Haupt-/Sekundärkontakt = Rolle einer Kontakt-Relation. **AbnahmeProtokoll bleibt rechtlich bindend** (immutable, 6 Jahre, signatur-gebunden) und bindet eine Mängel-Checkliste ein.

Cross-Cutting-Entitäten (modulübergreifend - zuerst verstehen!)

Diese werden von vielen Modulen geschrieben/referenziert. Sie sind der Grund, warum das Modell erweiterbar bleibt.

Entität	Modul	Rolle	Schlüssel
Activity [v1]	CRM	Event-Sourcing-Zentrale: jedes Modul schreibt Aktivitäten hierher, ein Stream liest sie	`subject_type` + `subject_id` (polymorph) · `actor_kind` (user/system/automation/ki_agent) · `activity_type` (~30 Typen) · `related_entity_refs[]`
AuditEntry [v1]	Audit	Append-only Compliance-Log, SHA-256-Hash-Kette	`subject_type/id` · `before_diff` / `after_diff` (JSONB) · `summary_text` (DE) · [append-only]
FileEntityLink [v1]	Files	M:N Datei ↔ beliebige Entität	`file_id` + `entity_type` + `entity_id` + `link_role` [polymorph]
CustomFieldDefinition / CustomFieldValue [v1]	CRM	Tenant-eigene Felder auf Projekt/Kontakt/Lead/Rechnung	Definition (Typ, Entity-Ziel) ← Value (entity_id + Wert) · pro Pipeline via `PipelineFieldConfig` schaltbar
Tag / TagAssignment [v1]	CRM	Farbige Tags, entity-übergreifend	Tag (name, color) ↔ Assignment (entity_type + entity_id) [polymorph]
SearchIndexEntry [v1]	Suche	Zentraler Such-Index über alle Entitäten	`entity_type/id` · `tsvector_de` (FTS) · `embedding` pgvector **[v2-Spalte]** · `permission_subject`
HiveNumberSeq [v1]	Sales	Fortlaufende Nummern (P-2026-0042, R-2026-0042)	pro `tenant_id` + `entity_type`, multi-tenant-sicher
DataConnection [v1]	Apps-Engine	Tenant ↔ externe App (AirCall, PandaDoc, Google, KWP)	`app_id` · `credentials` (verschlüsselt) · `connection_status`

Die Tenant/User-Achse (hängt an fast allem): `Tenant 1:N TenantUser N:1 User` · Rollen über `TenantRole` + `Permission` + `RoleAssignment` · `Group` orthogonal (N:M User).

Domäne 1: Identität & Mandanten (M27)

Entität	v	Kern-Felder	Relationen / Flags
Tenant	v1	slug, status (trial/active/past_due/frozen/suspended/deletion_pending), default_language	1:N TenantUser/Role · [RLS-Wurzel]
User	v1	email, language, timezone, theme_preference, date_format	N:M Tenant (kein RLS, global)
TenantUser	v1	tenant_id, user_id, status	M:N Role via RoleAssignment · [RLS]
TenantRole	v1	name, is_template_origin	1:N Permission · [RLS] · 10 SHK-Rollen geseedet
Permission	v1	role_id, permission_key	[RLS]
Group	v1	name	N:M User · [RLS][Soft-Delete]
TenantUserInvitation	v1	email, token, expires_at, accepted_at	[RLS] · 48h-Token
Session	v1	user_id, tenant_id, last_activity_at, browser_fingerprint	Postgres-Source + Redis-Cache · [RLS]
ApiKey	**v2**	key_hash, scope, expires_at	[RLS][Soft-Delete] · wandert mit Public-API

Domäne 2: Querschnitt (Audit M21, Notifications M22, i18n M30)

Entität	v	Kern-Felder	Relationen / Flags
AuditEntry	v1	actor_kind, action, subject_type/id, before/after_diff, summary_text, ip	[RLS][append-only]
AuditExportJob	**v2**	filter_criteria, format (csv/json/pdf), status, result_file_url	[RLS]
AuditRetentionPolicy	**v2**	default_retention_months (6–120), legal_hold_active	(tenant_id PK)
Notification	v1	recipient_user_id, event_type, title, body, deeplink, priority, read_at, bundle_id	[RLS]
NotificationPreference	v1	user_id, event_type, channel_in_app/email/push, priority_threshold	[RLS]
NotificationDevice	v1	user_id, device_kind, push_token	[RLS] · Push-Kanal ruht in v1
NotificationBundle	**v2**	event_type_group, count, bundle_window_seconds	[RLS]

NotificationQuietHours	**v2**	user_id, weekday, start/end_time, urgent_override	[RLS]

i18n: keine eigene Tabelle - Strings in JSON-Files; nur Felder `User.language`, `Tenant.default_language`, `Kontakt.language`.

Domäne 3: Files & Tasks (M23, M26)

Entität	v	Kern-Felder	Relationen / Flags
File	v1	display_name, mime_type, size, storage_key, checksum, preview_status, assigned_to, source	[RLS][Soft-Delete]
FileVersion	**v2**	file_id, version_number, storage_key, change_note	[RLS][append-only]
FileChunk / FileUploadProgress	v1	upload_id, chunk_index / received_chunks, status, expires_at (24h)	chunked Upload + Resume
FileEntityLink	v1	file_id, entity_type, entity_id, link_role	[RLS][polymorph]
Task	v1	title, description, assigned_to_user **oder** _group, deadline, priority, status, source, linked_entity_type/id	[RLS][Soft-Delete][polymorph link]
TaskWatcher	v1	task_id, user_id	M:N · [RLS]

Domäne 4: Apps-Engine (M33a) - der Integrations-Vertrag

Entität	v	Kern-Felder	Relationen / Flags
App	v1	key (kwp/gmail/aircall), category, connection_type (oauth2/api_key), connection_schema, phase	global, kein RLS
DataConnection	v1	tenant_id, app_id, credentials (verschlüsselt), connection_status, last_health_check	[RLS]
AppSyncRun	v1	data_connection_id, kind, status, items_total/succeeded/failed, summary_text	[RLS]
AppSyncRunLog	v1	severity, message_text, tech_details	[RLS]
AppSyncItem	v1	entity_type/id (HiveGrid) ↔ external_id, operation, status	[RLS][polymorph]

Domäne 5: Projekt-zentrischer CRM-Kern (M01–M04, AD-20) — siehe Kern-Ketten-Diagramm

> ***Ersetzt** die früher getrennten Domänen CRM-Kern / Lead-Inbox / Sales-Pipeline / Auftragspipeline. Ein `Projekt`-Kern, Pipelines als Views.*

Entität	v	Kern-Felder	Relationen / Flags
Projekt	v1	projekt_nummer (HiveNumberSeq), projektname, projekttyp, quelle, promotion, prioritaet, start/deadline, auftragsvolumen, potenzieller_auftragswert, zustaendige_person_id, projektleiter_id, projekt/rechnungs/bauher_adresse_id, kommunikation_*, stage_id (=Status), konvertiert_aus_lead_id	N:1 Pipeline/Stage · M:N Kontakt (ProjektKontakt) · 1:N Sub-Areas · [RLS][Soft-Delete]
Pipeline	v1	name, typ_hint (sales/auftrag/custom), position	1:N Stage/FieldConfig/Access · [Soft-Delete]
PipelineStage	v1	pipeline_id, name, stage_kind (offen/gewonnen/verloren), win_probability, color	N:1 Pipeline · = die Statuswerte
PipelineFieldConfig	v1	pipeline_id, stage_id (NULL=pipeline-weit), field_key (System ODER `custom:<id>`), sichtbar, pflicht, reihenfolge	Konfig-Layer · nicht sichtbare Felder bleiben erhalten
PipelineAccess	v1	pipeline_id, principal_type (rolle/user/gruppe), principal_id, kann_sehen, kann_bearbeiten	wer-sieht-Pipeline
ProjektMitglied	v1	projekt_id, user_id, rolle (projektleiter/monteur/zustaendig/beobachter)	M:N (mehrere Monteure)
Kontakt	v1	kontakt_typ (privatperson/organisation), kontakt_status (kontakt/kunde/bestandskunde), vorname/nachname, org_name, org_merkmal (gmbh/kg/öffentlich/sonstige — optional), primaer_email/-telefon, adresse_id	M:N Projekt · self-ref KontaktRelation · [RLS][Soft-Delete]
KontaktRelation	v1	von_kontakt_id, zu_kontakt_id, rolle (hauptkontakt/sekundaerkontakt), kontext	self-ref M:N — Kontaktart = Rolle der Relation
KontaktEmail / KontaktTelefon	v1	kontakt_id, wert, primaer	Multi-Werte 1:N
ProjektKontakt	v1	projekt_id, kontakt_id, rolle (auftraggeber/bauherr/rechnungsempfaenger/ansprechpartner), primaer	M:N-Brücke Projekt↔Kontakt
Lead	v1	quelle, quelle_detail, status (new/qualified/converted/lost/spam), **weiterverarbeitung_freigabe (ja/nein/unklar)**, vorname/nachname/email/telefon, rohdaten (JSONB), spam_score, konvertiert_projekt_id, verknuepfter_kontakt_id	→ Projekt, → Kontakt · [Soft-Delete] · DSGVO-Auto-Lösch
Adresse	v1	strasse, hausnummer, plz, ort, land	von Projekt (3 Slots: Projekt/Rechnung/Bauherr) + Kontakt referenziert

Sub-Areas am Projekt:

Entität	v	Kern-Felder	Flags
Rechnung	**v1 NEU**	rechnungsnummer (HiveNumberSeq), betrag, status (entwurf/gestellt/bezahlt/storniert), datum, faellig_am, datei_id, quelle (manuell/kwp)	einfaches Sub-Entity am Projekt (E3)
Termin	v1	projekt_id, kontakt_id, titel, start/ende_at, ort	Kalender-Sync (M25) dockt an
Checkliste / ChecklistItem	v1	name, typ (generisch/maengel/abnahme_doku) / text, status, maengel_klasse (kritisch/klein/kosmetisch), foto_file_id	„Mängel" = Checkliste mit typ=maengel (E2)
AbnahmeProtokoll	v1	projekt_id, maengel_checkliste_id, datum, anwesende, druckprobe, funktionspruefung, zaehlerstaende, pdf_file_id, gesperrt_at	**[immutable]** 6 Jahre · offline-fähig
AbnahmeSignatur	v1	abnahme_id, signer_name, signer_rolle (kunde/monteur), signatur_file_id, signiert_at	Touch-Unterschrift
ProjektExternesFeld	v1 Halter / **v2 Sync**	projekt_id, system (kwp), feld_key, wert, editierbar	KWP/API-Felder — v1 nur Anzeige, Sync ab v2
CustomFieldDefinition / Value	v1	entity_target (projekt/kontakt/lead/rechnung), field_type / entity_id, wert	dynamische Felder, pro Pipeline schaltbar
Activity / Tag	v1	siehe Cross-Cutting	[polymorph]

Material-Ausgang/Lagerist-Sicht, Zeiterfassung, Leistungspositionen, Baustelle, Supporttickets → v2-Backlog (Modell-Pivot E2). Material-Listen am Auftrag bleiben v1.

Domäne 6: Suche & Dashboard (M24, M20)

Entität	v	Kern-Felder	Relationen / Flags
SearchIndexEntry	v1	entity_type/id, title, body_text, tsvector_de, **embedding (pgvector → v2)**, permission_subject, is_archived	[RLS][polymorph]
RecentSearch	v1	user_id, query_text, applied_filters, result_count	[RLS]
EmbeddingProviderJob	v1	search_entry_id, provider, status	async (Embedding-Inhalt v2)
Dashboard	v1	name, is_default_for_role (gf/office/monteur/…), is_shared	[RLS][Soft-Delete]
DashboardLayout	v1	dashboard_id, user_id (NULL=shared), widgets[], layout_data	3 feste Rollen-Layouts v1
WidgetInstance	v1	widget_catalog_key, config, position, size	(Custom-Widgets + Drag&Drop → v2)

Domäne 7: Automation & KI (M17, M18)

Entität	v	Kern-Felder	Relationen / Flags
AutomationRule	v1	name, flow_data (JSON), enabled, created_via	[RLS][Soft-Delete]
AutomationRuleStep	v1	rule_id, step_kind (trigger/condition/action/flow_control), node_type, config, parent_step_ids[]	Graph-Knoten
AutomationExecution	v1	rule_id, trigger_payload, status, dry_run, idempotency_key, reverse_window_expires_at	[RLS][24h-Undo]
AutomationStepExecution	v1	execution_id, step_id, input/output, summary_text	
AutomationCredential	v1	key, value (verschlüsselt)	[verschlüsselt]
AutomationTemplate	v1	flow_data, category, is_published	SHK-Vorlagen
LlmGatewayProvider / Model	v1	key, region (EU), model_kind (chat/embedding/transcription)	Mistral primär
AiSettings	v1	enabled_use_cases[], default_models, usage_limit_tokens	pro Tenant
AiCredits	v1	current_balance_tokens, monthly_grant	→ Credit (M28)
AiTokenUsage	v1	use_case, model_id, input/output_tokens, cost	[RLS]
Conversation / ConversationMessage	v1	kind (programm_help/automation_builder), role, content	[RLS][Soft-Delete]
GeneratedObject	v1	use_case, data (Schema-validiert)	strukturierte KI-Outputs
TenantEmbedding / HiveMindRAGEmbedding	**v2**	entity_type/id, embedding (pgvector)	RAG [polymorph]
WhisperTranscript	**v2**	audio_source_type, transcript_text, confidence	Diktat

Domäne 8: Kommunikation (Email M09/M10, Telefonie M12, Chat M13)

Entität	v	Kern-Felder	Relationen / Flags
TenantSendingDomain	v1	domain, dkim/spf/dmarc-Records, verification_status	[RLS][verschlüsselt][Soft-Delete]
TenantInboundAddress	v1	local_part, full_address, purpose (lead_inbox/bcc_logging/reply_routing)	[RLS][Soft-Delete]
Email	v1	purpose, from_sender_id, to, delivery_status, related_kontakt/projekt_id	System-/Transaktions-Mails · [RLS] [Soft-Delete]
EmailIntegrationMailbox / Message / SyncCursor / WebhookEvent	**v2**	provider (gmail/microsoft), oauth_tokens, matched_kontakt/projekt_id	Postfach-Sync [RLS][polymorph]
Campaign	v1	name, subject, template_id, sender_id, recipient_segment_id, status, delivered/opened/clicked_count	[RLS][Soft-Delete]
OutboundEmail	v1	campaign_id, recipient_kontakt_id, tracking_token, unsubscribe_token, delivery_status	pro Empfänger [RLS]
OutboundEmailOpen / Click	v1	opened_at (IP anonymisiert) / clicked_at, link_url	Tracking

EmailTemplate / EmailSender	v1	body_html / from_name/email, sending_domain_id, is_default	[RLS]
RecipientSegment	v1	entity_type, filter_definition (JSONB), is_static_snapshot	[RLS][Soft-Delete]
Unsubscribe / EmailSuppressionList	v1	kontakt_id, scope, reason / email_address (unique), reason	[RLS]
TelephonyConnection	v1	provider (aircall), oauth_token (verschl.), webhook_secret	max. 1 aktiv/Tenant · [RLS]
TelephonyUserBinding	v1	tenant_user_id ↔ provider_user_id	AirCall-User-Mapping
TelephonyCallSession	v1	direction, status, from/to_number (E.164), matched_kontakt/lead/projekt_id, note, recording_url	[RLS][Soft-Delete][polymorph match]
TelephonyWebhookEvent	v1	event_type, signature_valid, processing_status	[append-only][30d]
HalloPetra-Modelle	**v2**	(M105)	zweiter Telefonie-Adapter
TeamChat	v1	kind (project/group/dm), projekt_id, archived_reason, last_message_at	[RLS][Soft-Delete]
TeamChatMember	v1	chat_id, user_id, role_in_chat, last_read_message_id, is_pinned	[RLS]
TeamChatMessage	v1	chat_id, author, body (Markdown), parent_message_id (Thread), mentions[], attachments[]	[RLS][Soft-Delete]
TeamChatThread / PinnedMessage / Reaction / Read	v1	parent_message_id / message_id / emoji / read_at	Threads, Pins (max 50), Reactions, Read-Receipts

Domäne 9: Kalender, Booking, Forms, Signatur (M25, M06, M05, M07)

Entität	v	Kern-Felder	Relationen / Flags
CalendarConnection	v1 (Google) / **v2 (MS)**	provider, oauth_refresh_token (verschl.), status	[RLS][verschlüsselt]
CalendarEvent	v1	source (internal/external_synced), starts/ends_at, is_busy, internal_entity_link	[RLS][Soft-Delete]
CalendarEventMapping / SyncCursor / WebhookEvent	v1	internal↔external_id / cursor_token / payload	Sync-Infra
BookingPage	v1	slug, duration_minutes, assignment_strategy (eine Strategie v1), is_published, intake_form_id	[RLS][Soft-Delete]
BookingPageProvider	v1	booking_page_id, tenant_user_id	M:N User
AvailabilityRule / Exception	v1	day_of_week, start/end_time / date, kind	Verfügbarkeit
Appointment	v1	booking_page_id, provider_user_id, start/end, status (scheduled/held/cancelled/no_show/rescheduled), kontakt_idlIead_id, form_submission_id	→ Kontakt/Lead/Form · [RLS]
Form	v1	slug, subtype (standard/acceptance_form/onboarding/survey), status, is_public	1:N Field/Action · [RLS] [Soft-Delete]
FormField	v1	field_key, field_type (12 Typen), required, validation_rules	N:1 Form
FormSubmission	v1	form_id, submission_data (JSON), linked_entity_type/id	[RLS][Soft-Delete] [polymorph]
FormFile	v1	submission_id, field_key, file_id	→ File
FormAction	v1	action_type (send_email/create_lead/create_projekt/webhook/trigger_automation), action_config	Post-Submit
FormLink	**v2**	token, expires_at, usage_limit	personalisierte Links
SignatureProviderConnection	v1	data_connection_id (Apps-Engine!), provider (PandaDoc), is_default	[RLS]
ProjektSignatureDocument	v1	projekt_id, provider_document_id, source_pdf_file_id, status (draft/sent/viewed/signed/declined), signed_pdf_file_id	→ Projekt (Angebot/Pipeline) · [RLS] [Soft-Delete]
ProjektSignatureSigner	v1	document_id, signer_email/name, linked_kontakt_id, status, signed_at	→ Kontakt

Domäne 10: HR-Hub (M14) - viel Verschlüsselung

Entität	v	Kern-Felder	Flags
HrEmployeeProfile	v1	user_id (1:1), personal_number, hire_date, position, team, location_id, manager_user_id, emergency_contact_*	[RLS]
HrLocation	v1	name, address_*, geo, image_file_id	[RLS][Soft-Delete]
HrAnnouncement (+ Read/Reaction/Comment)	v1	title, body, audience_kind, requires_acknowledgement	[RLS][Soft-Delete]
HrEmployeeDocument	v1	employee_user_id, document_kind, file_id, visibility	[RLS][Soft-Delete]
HrResource / HrQuickLink	v1	kind, url/file_id, category, audience	[RLS][Soft-Delete]
HrSurvey (+ Item/Submission)	v1	is_anonymous, result_visibility, item_type (text/rating_5/nps/choice/boolean)	[RLS] · anonym: submitter_hash
HrRequest (+ Template)	v1	request_kind (expense/hardware/training/reference), form_submission_id, status, assigned_approver	[RLS][Soft-Delete]
EmployeePayslip	v1	period_year/month, file_id (PDF), is_read	[RLS][Soft-Delete]
EmployeeUploadReminder (+ Dispatch)	v1	document_kind, cadence_kind, due_at, status	[RLS]
EmployeeVacationRequest	v1	start/end_date, request_kind, status, assigned_approver, calendar_event_id	→ Calendar · [RLS]
EmployeeSickLeaveReport	v1	start/return_date, certificate_file_id, status	[RLS][**verschlüsselt** AU-Schein]

EmployeePasswordSafeVault (+ Entry/Access)	v1	username/password/notes (**AES-256-GCM**), permission_level (read/write/admin)	[RLS][**verschlüsselt**] · Reveal audit-geloggt

Domäne 11: Plattform-Admin (M37) - eigene Welt, KEIN tenant_id

Entität	v	Kern-Felder	Flags
AdminUser	**v2**	email, password_hash, totp_secret (Pflicht), status	v1 = Boilerplate-Modell mit MFA-Step-up; getrennte Identität → v2
AdminRole / AdminPermission / AdminSession	**v2**	name (super_admin/support/sales) / permission_id / expires_at (8h), ip	parallele RBAC → v2
AdminAuditLog	v1	admin_user_id, action_type, target_kind/id, summary_text, metadata (Diff)	cross-tenant (tenant_id=null)
AdminTenantNote	v1	tenant_id, note_text, category, pinned	Markdown-Notizen
AdminSupportTicket (+ Message)	v1	source_kind (in_app_feedback_widget/inbound_email), tenant_id, status, priority	speist sich aus Feedback-Widget
AdminLead / AdminDeal / AdminLeadActivity	**v2**	(M040 Plattform-Lead-Pipeline)	unser eigenes Sales-Tool

Domäne 12: Billing & Abo-Baukasten (M28) - wichtig: das Recut-Modell

Struktur (keine Seat-Typen mehr - Basis-Preis pro Nutzer + Feature-Module als Positionen):

```
SubscriptionProduct ─1:N─▶ SubscriptionPrice (stripe_price_id, interval month|year|one_time)
        │  (Plan: Core/Swarm/Colony)
        ├─M:N─▶ SubscriptionFeature (via ProductFeature, +quota_value JSON)  ◀── Konnektor-Gate!
        └─1:N─▶ SubscriptionUsageBasedPrice ─1:N─▶ UsageBasedTier (Staffel-Overage)
                                                          │
TenantSubscription (1:1 Tenant) ─N:1─▶ Product            │
        ├─1:N─▶ TenantSubscriptionProductPrice (quantity=Seats, is_metered)
        ├─1:N─▶ TenantSubscriptionUsageRecord (metric, usage_amount, reported_to_stripe)
        └─ status (trial/active/past_due/cancelled/frozen) + Stripe-IDs + Dunning-Felder
Credit (ai_token/bonus, balance_after, valid_until) ─N:1─▶ Tenant
CheckoutSessionStatus (stripe_checkout_session, intent) ─N:1─▶ Tenant
```

`SubscriptionFeature.code` (z. B. `kwp_integration`) ist die Brücke zur Konnektor-Pflicht: `tenantHasModule()` prüft, ob das Feature im Abo des Tenants enthalten ist. Quotas (max_users, ai_tokens) hängen an `SubscriptionProductFeature.quota_value`.

Domäne 13: Onboarding, Branding, DSGVO (M29, M31, M35)

Entität	v	Kern-Felder	Flags
Onboarding (+ Filter/Step)	v1	code (shk_standard), industry, template_directory	global · Wizard-UI → v2
OnboardingSession (+ Action)	v1	onboarding_id, current_step, status, payload	[RLS]
OnboardingDemoProvisioning	v1	template_code, entities_created (JSON), removable_until (+30d)	is_demo_data-Flag in Listen
OnboardingInvitationDispatch	v1	tenant_user_invitation_id	→ M27
(User.theme_preference)	v1	system/light/dark - **v1 nur light/dark**	Feld in User; TenantBranding → v2
GdprExportJob	v1	scope (user_self/tenant_full), status, result_file_id (ZIP), result_expires_at	[RLS]
TenantDeletionRequest	v1	reason, grace_period_ends_at (+30d), status	[RLS] · Slug-Confirm
AvvAcceptance	v1	avv_version, accepted_by, pdf_file_id, signer_name	[RLS]
Subprocessor	v1	name, country, is_eu_eea, data_categories, dpa_url	global

Zähl-Übersicht & offene Modellierungs-Fragen

~135 Entitäten gesamt (Kern konsolidiert: Deal+Project → Projekt), davon ~25 mit `[v2]` markiert. Cross-Cutting (polymorph): Activity, AuditEntry, FileEntityLink, SearchIndexEntry, Tag, CustomField, AppSyncItem, Task-Link, Telephony/Email-Match, ProjektKontakt.

Punkte, die du (UX) und Raffael früh klären solltet:

1. **Polymorphe Refs ohne FK** (Activity/FileEntityLink/SearchIndexEntry): keine referenzielle Integrität auf DB-Ebene - App-seitige Konsistenz + Cleanup-Jobs nötig. Bewusst so (Erweiterbarkeit).
2. **Adressen:** gelöst über eigene `Adresse`-Tabelle (Projekt mit 3 Slots: Projekt/Rechnung/Bauherr; Kontakt 1 Slot). HrLocation behält embedded address_*.
3. **Sichtbarkeit & Rechte pro Pipeline** (`PipelineAccess` + `PipelineFieldConfig`): der zentrale neue Mechanismus — Field-Level-Sichtbarkeit (z.B. Volumen nur für Vertrieb) läuft jetzt über die Pipeline-Konfiguration statt über harte Permission-Bundles. Früh durchdenken.
4. **HiveNumberSeq** ist der einzige bewusst nicht-RLS-isolierte Schreibpfad mit Race-Potenzial: braucht Row-Lock/Sequence-Strategie (für Projekt- + Rechnungs-Nummern).
5. **`is_demo_data`-Flag** muss auf allen seedbaren Entitäten existieren (für Bulk-Lösch) - quer durchs Modell, nicht vergessen.
6. **Werte final definieren:** `kontakt_status` (Kontakt/Kunde/Bestandskunde …?), Label „Unternehmen" vs. „Organisationen", je KWP-Feld editierbar vs. nur-synced.

Begleitdateien: `entity-model.dbml` *(DBML für dbdiagram.io / ER-Diagramm* `entity-model.svg` *)* · `ENTITY-MODELL-ERKLAERUNG.pdf` *(laienverständliche Fassung). Approved-Quelle:* `HiveGrid_CRM_Datenlogik_UX_Entscheidungen.md` *.*
Architektur-Entscheidungen: `docs/ARCHITECTURE-DECISIONS.md` *(AD-1 RLS, AD-2 First-Class revidiert, AD-20 Projekt-zentrisch).*