FLW-1232 Requirements

Oct 13, 2026 · @hi

Overview

FLW-1232 asks the Product team to document, module by module, how Utprerona (FLW) data is captured, stored, updated and linked across database tables. The output is the base input for the BIDA team's dashboard and reporting requirements.

Field	Value
Ticket	FLW-1232
Title	Utprerona (FLW) Application Data Flow & Table Mapping
Type / priority	Task / High
Status	Todo/Open
Fix version	2.14
Sprint	FLW Sprint 54
Service line	FLW Mobile App (all environments)
Reporter	Pranoti Monde
Assignee	Ishwar Chouhan
CC	Syed Saleem, Mithun James
Created	31 Aug 2026

Background and context

Dashboards and reports for Utprerona are being strengthened, but no single reference explains where each workflow's data lives. Data for one module is often spread over several tables and depends on other modules.

What the FLW-API code shows today (main branch):

- **117 JPA entities** in two databases: 105 in `db_iemr` (FLW transactions, masters, users) and 12 in `db_identity` (beneficiary and household).

- **Almost no declared relationships.** Only 9 JPA associations exist, all in dynamic forms and Sammelan attachments. Every other link is a plain id column.
- **Five shared keys tie the modules together:** beneficiary (`ben_id` → `i_beneficiarydetails_rmnch.beneficiaryId`), registration (`BeneficiaryRegID` → `m_beneficiaryregidmapping`), household (`household_id` → `i_householddetails.houseoldId`), user/ASHA (`user_id` , `asha_id` → `m_user.UserID`) and visit (`BenVisitID` → `t_benvisitdetail`).
- **Cross-database links.** Beneficiary and household keys point from `db_iemr` into `db_identity` , so no foreign keys enforce them.

A first-pass inventory with ER diagrams already exists: FLW-API Schema Map.

Scope

The ticket names five scope items: map all modules, map data flow between tables, document relationships and identifiers, identify cross-module dependencies, and identify data-flow gaps.

In scope

- All Utprerona modules backed by FLW-API: beneficiary and household, eligible couple, maternal, child, death review, disease control, TB, incentives, dynamic forms, OPD, campaigns and meetings, users and notifications.
- Tables in `db_iemr` and `db_identity` that FLW-API reads or writes.
- The path each record takes: app screen → API endpoint → service → table.
- Keys and identifiers that join tables within and across modules.

Out of scope

- Defining indicators, report logic or dashboard designs. BIDA does that next, using this output.
- Schema changes or data fixes. Gaps found here are logged as separate tickets.
- Other AMRIT services (HWC, MMU, TM), except where FLW reads their shared tables.

Functional requirements

Each requirement is one deliverable that BIDA, QA or development can check.

ID	Requirement	Ticket scope item
FR-1	List every Utprerona module and the app screens/forms in it.	Map all modules

ID	Requirement	Ticket scope item
FR-2	For each module, list every table it writes and reads, with database (`db_iemr` / `db_identity`), table name, entity class and repository.	Map all modules
FR-3	For each table, document the primary key, the business identifiers and every link column with its target table and column.	Relationships and identifiers
FR-4	Mark each link as declared (JPA association or DB foreign key) or logical (inferred from a shared id).	Relationships and identifiers
FR-5	Describe the data flow per module: app form → sync API endpoint → service → table, covering create, update and soft delete (`is_active` / `deleted`).	Data flow between tables
FR-6	Document the lifecycle chains that cross modules, for example eligible couple → pregnancy → ANC → delivery → PNC → infant → child, and screening → suspected → confirmed for TB.	Cross-module dependencies
FR-7	Document which actions write incentive records (`incentive_activity_record`) and which activity in `m_incentive_activity` each one triggers.	Cross-module dependencies
FR-8	Provide an ER diagram per module, plus one overview diagram of the shared keys.	Data flow between tables
FR-9	Record the audit and sync columns each table carries (created/modified by and date, sync status, user/ASHA id) that reports filter on.	Relationships and identifiers
FR-10	Log every data-flow gap found, with table, impact on reporting and a suggested fix.	Data-flow gaps
FR-11	Publish the output as one shared reference that Product, Development, QA and BIDA can comment on.	Objective

Data and API impact

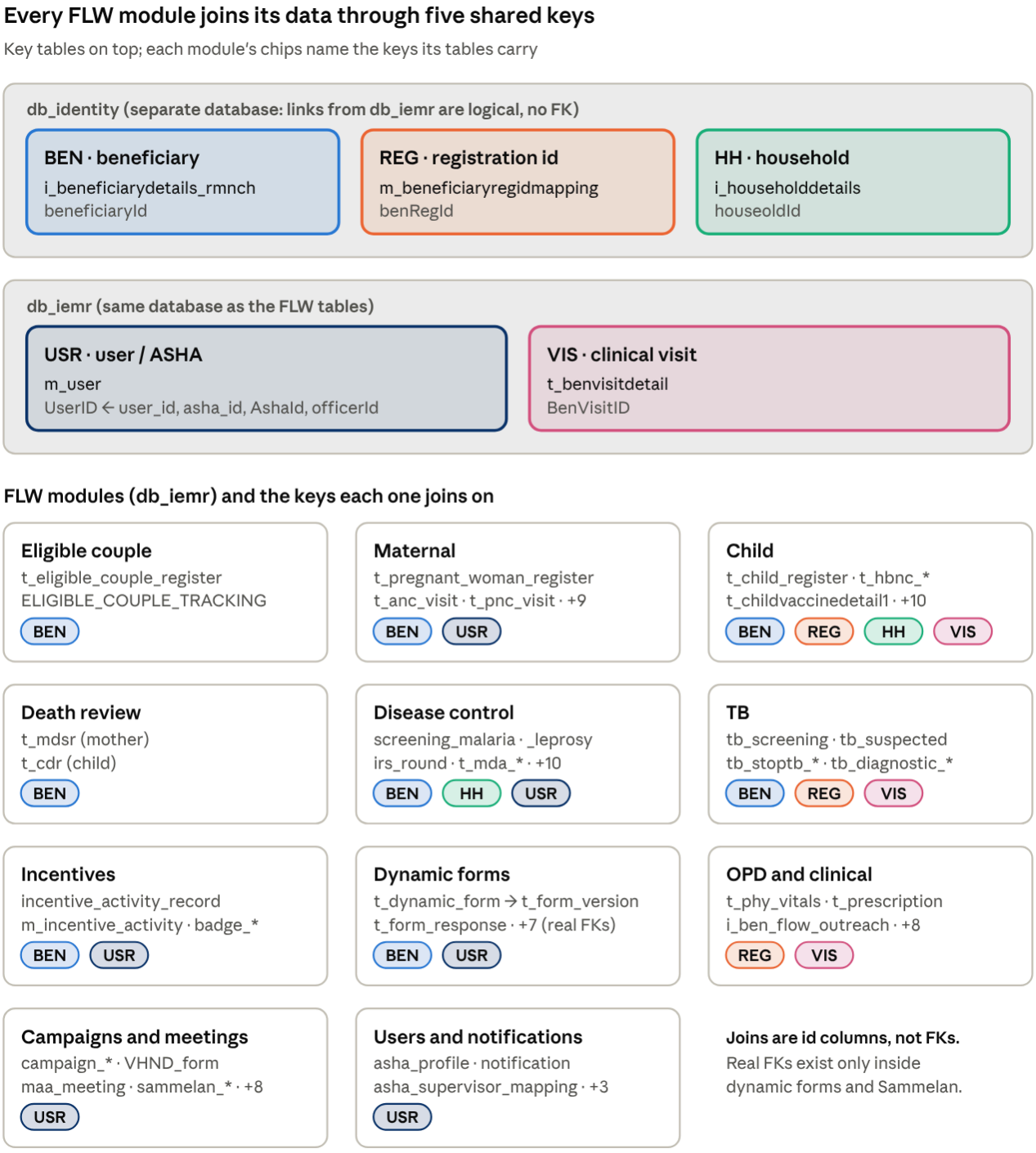
The first-pass inventory groups the 117 entities into 12 modules. This is the starting list for FR-1 to FR-3. API endpoints and app screens per module still need to be added (FR-5).

Module	Main tables	Joins on
Beneficiary and household (`db_identity`)	`i_beneficiarydetails_rmnch`, `i_householddetails`, `m_beneficiaryregidmapping`, `i_beneficiarymapping`, `i_beneficiarydetails`, `i_bornbirthdeatils`, `i_cbacdetails`	beneficiaryId, benRegId, houseoldId
Eligible couple	`t_eligible_couple_register`, `ELIGIBLE_COUPLE_TRACKING`, `non_pregnant_high_risk_assess/track`	ben_id
Maternal	`t_pregnant_woman_register`, `t_anc_visit`, `anc_counselling_care`, `t_pmsma`, `pregnant_high_risk_assess/track`, `t_micro_birth_plan`, `t_delivery_outcome`, `t_pnc_visit`	ben_id, user_id
Child	`t_infant_register`, `t_child_register`, `t_hbnc_*`, `t_hbyc`, `t_hbyc_child_visits`, `t_childvaccinedetail1`, `t_sam_visit`, `t_ors_distribution`, `adolescent_health`	ben_id, child_ben_id, household_id, BeneficiaryRegID
Death review	`t_mdsr`, `t_cdr`	ben_id
Disease control	`screening_malaria/kala_azar/aesje/filaria/leprosy`, `malaria_follow_up`, `leprosy_follow_up`, `i_mobilization_mosquito_net`, `irs_round`, `t_mda_distribution_data`, `cdtf_visit_details`, `t_eye_checkup`, `t_cbacdetails`	ben_id, household id, user_id
TB	`tb_screening`, `tb_suspected`, `tb_confirmed_cases`, `tb_stoptb_*`, `tb_diagnostic_order/result/document`	ben_id, ben_reg_id, visitCode, nikshay_id
Incentives	`m_incentive_activity`, `incentive_activity_record`, `incentive_pending_activity`, `badge_*`	activity_id, asha_id, ben_id
Dynamic forms	`t_dynamic_form`, `t_form_version`, `t_form_section`, `t_section_question`, `t_question_option`, `t_form_response`, `t_section_response`, `t_question_response`	formId, versionId, beneficiaryId

Module	Main tables	Joins on
OPD and clinical	`t_benvisitdetail`, `t_phy_vitals`, `t_phy_anthropometry`, `t_prescription`, `t_prescribeddrug`, `t_anccare`, `t_pnccare`, `i_ben_flow_outreach`, `general_opd_entry`	BeneficiaryRegID, BenVisitID
Campaigns and meetings	`campaign_ors`, `campaign_pulse_polio`, `campaign_filariasis_mda`, `deworming_form`, `ahd_form`, `VHND_form`, `vhnc_form`, `phc_review_meeting`, `maa_meeting`, `sammelan_record`, `uwin_session_record`, `t_ifa_distribution*`	user_id / asha_id
Users and notifications	`m_user`, `v_userservicerolemapping`, `asha_profile`, `asha_supervisor_mapping`, `user_fcm_tokens`, `notification`, `m_smstemplate`	UserID

Full column-level mapping and the per-module ER diagrams: FLW-API Schema Map.

ER diagram



FLW ER overview · 5 key tables, 11 modules

Every module's tables link to the five key tables through id columns. Beneficiary (BEN) is the most common join. OPD, TB and child vaccination key on the registration id (REG) and visit id (VIS) instead, so reports must bridge the two through m_beneficiaryregidmapping. Column-level diagrams for each module are in the FLW-API Schema Map.

Proposed solution

We build the mapping from the code, verify it against the live schema, and hand BIDA five artefacts, one per scope item in the ticket. The code is the source because the app, the API and the tables can drift, and only the code shows what is actually written.

Approach

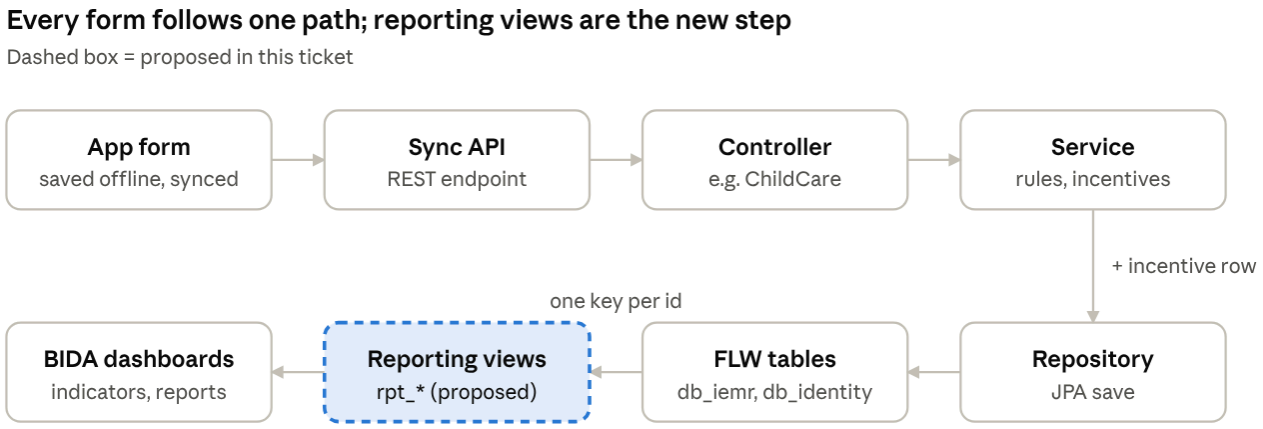
1. **Extract from code.** Scan FLW-API for `@Entity` / `@Table` classes, repositories, services and `@RestController` endpoints. The first pass is done: 117 entities in 12 modules (see the Schema Map).
2. **Verify against the database.** Compare the extract with `information_schema` for `db_iemr` and `db_identity` on a production replica. This confirms physical column names, indexes, row counts and orphan rates.
3. **Trace the flow.** For each module, follow one record from the app form to the sync endpoint, controller, service, repository and table. Use the graph of FLW-API calls, plus one test submission per form on UAT.
4. **Document and review.** Publish the artefacts below in one shared reference. Review them with Development, QA and BIDA.

Deliverables by scope item

Scope item (ticket)	Deliverable	Format
Map all relevant Utprerona modules	Module catalogue: module → app screens → endpoints → tables	Table in this doc + data dictionary sheet
Map data flow between tables	Data-flow diagram for each module, showing create, update and soft delete	Diagrams (below and in the Schema Map)
Document relationships and identifiers	Data dictionary: every table, PK, link column, target and audit/sync columns, plus the key register	Spreadsheet for BIDA, ER diagrams
Identify cross-module dependencies	Lifecycle chains and a module dependency matrix	Diagram + matrix
Identify data-flow gaps	Gap register with reporting impact and fix, plus a Jira ticket per code fix	Table + linked tickets

The controllers already identified in the graph give the starting endpoint list, for example `ChildCareController` (HBNC, HBYC, vaccination, SAM, ORS, IFA), `DeathReportsController` (CDR, MDSR), `DiseaseControlController` (malaria, kala-azar, AES/JE, filaria, leprosy, mosquito net), `CampaignController` (ORS, polio, filariasis MDA), `IncentiveController` and `AshaProfileController`.

Data flow



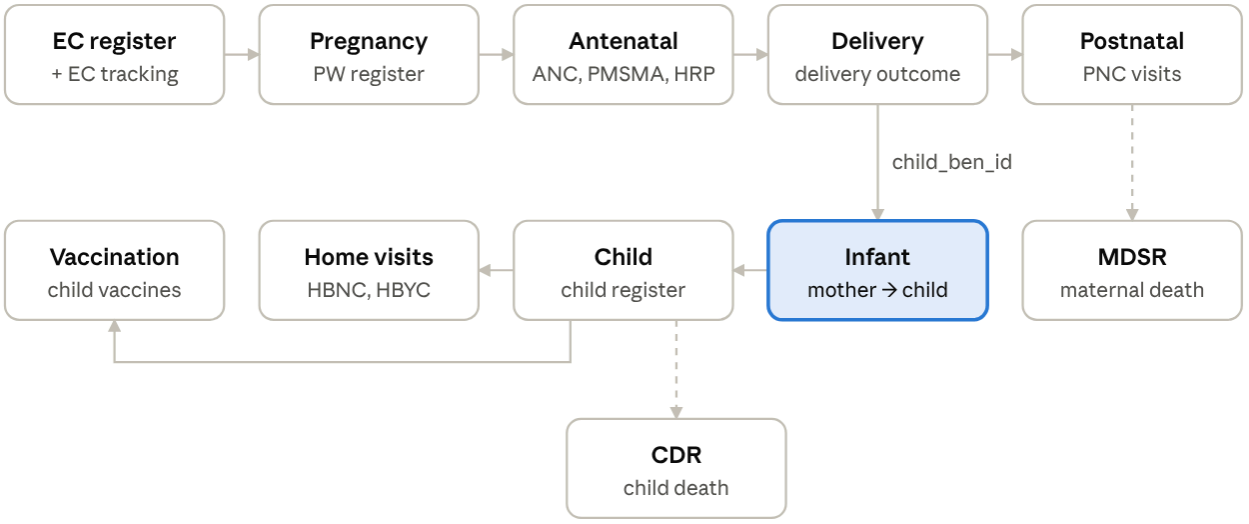
Data flow · app form to BIDA dashboard

Every module follows the same path. The service layer is where cross-module writes happen, such as an incentive row when an ANC visit is saved. The per-module flow (FR-5) fills in the actual endpoint, service and tables for each form.

Cross-module dependencies

The infant register hands the mother's ben_id over to the child's

Top row: mother (ben_id). Lower rows: child (child_ben_id). Dashed: only if a death occurs



Every arrow is a logical link through a beneficiary id; none is a foreign key.

Mother-to-child lifecycle · 10 steps

The maternal and child modules are one chain. A report that follows a mother into her child's records must go through `t_infant_register`, the only table that holds both `ben_id` (mother) and `child_ben_id` (child).

Module	Depends on	Through
Maternal	Eligible couple	ben_id of the woman
Child	Maternal (delivery outcome)	t_infant_register.child_ben_id
Death review	Maternal (MDSR), Child (CDR)	ben_id
TB	Beneficiary core, OPD visit	ben_reg_id, visitCode
Incentives	Every module that triggers a payment	ben_id, asha_id, activity_id
All modules	Beneficiary and household, Users	beneficiary, household and user ids

Gap fixes and reporting layer

Instead of changing the transactional schema now, we propose a read-only reporting layer of SQL views. It gives BIDA one name per key and closes most gaps without app changes. Code fixes are raised as separate tickets.

Gap	Proposed fix	Change type
No FKs across `db_iemr` and `db_identity`	Nightly orphan-check queries per table (rows whose `ben_id` is not in `i_beneficiarydetails_rmnch`), with counts on a data-quality dashboard	Reporting job
Beneficiary key has several names	Views expose one column, `beneficiary_id`, in every module view	Reporting views
Household key has seven spellings	Views expose one column, `household_id`, mapped to `i_householddetails.houseoldId`	Reporting views
Some modules use RegID, others beneficiary id	A bridge view `rpt_beneficiary_key` (beneficiary_id, ben_reg_id, household_id, asha_id) that every module view joins	Reporting views
Two entities write `i_ben_flow_outreach`	Confirm the owning flow with Development; document which columns each one writes	Clarification ticket
`t_ifa_distribution` vs `t_ifa_distribution_data`	Confirm which table is current; views read only that one	Clarification ticket
Implicit column names	Fix the physical names from the `information_schema` check (approach step 2)	Documentation
Missing indexes on join keys (to verify)	Add indexes on `ben_id`, `household_id`, `user_id` and created date where missing	DB change ticket

Proposed view set

- `rpt_beneficiary_key`: one row per beneficiary with all of its identifiers and the ASHA it belongs to.
- One `rpt_<module>` view per module, such as `rpt_maternal_anc` and `rpt_child_hbnc`. Each carries `beneficiary_id`, `household_id`, `asha_id`, the event date, `is_active` and the module's fields.
- `rpt_incentive_claims`: incentive records joined to the activity master and the source module.

- New code should use the same standard names (`beneficiary_id`, `household_id`, `user_id`) so the views stay thin.

Delivery plan

The work fits in Sprint 54 as six steps. Steps 1 and 2 are already done in first-pass form.

Step	Work	Owner	Output	State
1	Extract entities, tables and keys from FLW-API code	Ishwar Chouhan	Schema Map (117 entities, 12 modules)	First pass done
2	Draw ER diagrams per module and the overview	Ishwar Chouhan	Diagrams in this doc and the Schema Map	First pass done
3	Verify against the live schema (columns, indexes, row counts, orphans)	Ishwar Chouhan + DB admin	Corrected data dictionary	To do
4	Trace each module's flow: form → endpoint → table, one UAT submission each	Ishwar Chouhan + QA	Module catalogue, flow diagrams	To do
5	Write the gap register and raise fix tickets	Ishwar Chouhan	Gap register, Jira tickets	To do
6	Review with Development, QA and BIDA, then sign off	Product (Pranoti Monde)	Signed-off reference	To do

The reporting views are proposed here but are built under a separate ticket once BIDA confirms the indicator list.

Acceptance criteria

- [ ] **AC-1 (coverage):** Given the FLW-API `main` branch, when the entity list is compared with the mapping, then every `@Entity` class (117 today) appears in exactly one module.
- [ ] **AC-2 (tables):** Given any mapped table, when a reviewer checks the entity source, then its database, table name and primary key match the `@Table` and `@Id` annotations.
- [ ] **AC-3 (links):** Given any link column in the mapping, then its target table and column are named, and it is marked declared or logical.

- [ ] **AC-4 (data flow):** Given any module, when a tester submits its form in the app, then the rows created or updated match the tables and flow listed for that module.
- [ ] **AC-5 (cross-module):** Given a beneficiary who moves from eligible couple to pregnancy, delivery and child, then each step's table and joining key is documented and can be joined in a SQL query.
- [ ] **AC-6 (gaps):** Every gap in the log has a table, a reporting impact and a suggested fix, and each fix needing code has its own Jira ticket.
- [ ] **AC-7 (sign-off):** Product, Development, QA and BIDA have reviewed the document, and BIDA confirms it is enough to start defining indicators.

Gaps, open questions and dependencies

Data-flow gaps already visible (seed for FR-10)

Gap	Reporting impact
No foreign keys between `db_iemr` and `db_identity`	Orphan rows possible; every report join must be validated
The beneficiary key has several names (`ben_id`, `beneficiary_id`, `BeneficiaryId`, `beneficiaryId`)	Easy to join on the wrong column
The household key has seven spellings (`household_id`, `house_hold_id`, `house_hold_details_Id`, `HouseHoldId`, `householdId`, `houseoldId` and more)	Household-level indicators need a mapping layer
Some modules key on RegID, others on beneficiary id	Joins must go through `m_beneficiaryregidmapping`
`BenFlowStatus` and `GeneralOpdData` both map `i_ben_flow_outreach`	Unclear which flow owns the rows
`t_ifa_distribution` and `t_ifa_distribution_data` overlap	Possible double counting of IFA
Some link fields have no `@Column`, so names depend on Hibernate naming	Physical column names need a check against the live DB

Open questions

- [ ] Which modules are live in Utprerona today, and which are behind feature flags?
- [ ] Should the mapping cover the mobile app's local Room database, or only the server tables?

- [ ] Which environment is the source of truth for the live schema check: UAT or production replica?
- [ ] Does BIDA need a set format, such as a data dictionary in a spreadsheet?

Assumptions

- The FLW-API `main` branch matches release 2.14.
- Utprerona uses the same FLW-API and schemas as the generic FLW app.

Dependencies

- Read access to the `db_iemr` and `db_identity` schemas to verify columns and row counts.
- App team input for the screen → endpoint mapping (FR-5).
- BIDA review slot before closing the ticket (AC-7).

Appendix: FLW-API Schema Map

Every JPA entity in FLW-API (main branch) mapped to its physical table, with primary key, link columns and repository, followed by an ER diagram for each module. Tables and columns come from the @Table/@Column annotations; the class list was cross-checked with the graphify code graph. Link targets: BEN = i_beneficiarydetails_rmnch.beneficiaryId, REG = m_beneficiaryregidmapping.benRegId, HH = i_householddetails.houseoldId, USR = m_user.UserID, VIS = t_benvisitdetail.BenVisitID.

A1. Entity to table mapping (117 entities)

Entity	DB	Table	PK	Link columns → target	Repository	Module
EligibleCoupleRegister	db_iemr	t_eligible_couple_register	id	ben_id→BEN	EligibleCoupleRegisterRepo	Eligible couple
EligibleCoupleTracking	db_iemr	ELIGIBLE_COUPLE_TRACKING	id	ben_id→BEN	EligibleCoupleTrackingRepo	Eligible couple
PregnantWomanRegister	db_iemr	t_pregnant_woman_register	id	ben_id→BEN	PregnantWomanRegisterRepo	Maternal
ANCVisit	db_iemr	t_anc_visit	id	ben_id→BEN	ANCVisitRepo	Maternal
AncCounsellingCare	db_iemr	anc_counselling_care	id	beneficiary_id→BEN, anc_visit_id→t_anc_visit, user_id→USR	AncCounsellingCareRepo	Maternal
PMSMA	db_iemr	t_pmsma	id	ben_id→BEN	PmsmaRepo	Maternal
PNCVisit	db_iemr	t_pnc_visit	id	ben_id→BEN	PNCVisitRepo	Maternal
DeliveryOutcome	db_iemr	t_delivery_outcome	id	ben_id→BEN	DeliveryOutcomeRepo	Maternal
MicroBirthPlan	db_iemr	t_micro_birth_plan	id	ben_id→BEN, user_id→USR	MicroBirthPlanRepository	Maternal
PregnantWomanHighRiskAssess	db_iemr	pregnant_high_risk_assess	id	ben_id→BEN, user_id→USR	HRPregnantAssessRepo	Maternal
PregnantWomanHighRiskTrack	db_iemr	pregnant_high_risk_track	id	ben_id→BEN, user_id→USR	HRPregnantTrackRepo	Maternal
NonPregnantWomanHighRiskAssess	db_iemr	non_pregnant_high_risk_assess	id	ben_id→BEN, user_id→USR	HRNonPregnantAssessRepo	Maternal
NonPregnantWomanHighRiskTrack	db_iemr	non_pregnant_high_risk_track	id	ben_id→BEN, user_id→USR	HRNonPregnantTrackRepo	Maternal
HighRiskAssess	db_iemr	high_risk_assess	id	ben_id→BEN, user_id→USR	HighRiskAssessRepo	Maternal
AncCare	db_iemr	t_anccare	ID	BeneficiaryRegID→REG, BenVisitID→VIS	AncCareRepo	OPD & clinical
PNCCare	db_iemr	t_pnccare	ID	BeneficiaryRegID→REG, BenVisitID→VIS	PNCCareRepo	OPD & clinical
InfantRegister	db_iemr	t_infant_register	id	ben_id→BEN (mother), child_ben_id→BEN (child)	InfantRegisterRepo	Child
ChildRegister	db_iemr	t_child_register	id	ben_id→BEN	ChildRegisterRepo	Child
HbncVisit	db_iemr	t_hbnc_visit	id	ben_id→BEN, household_id→HH, asha_id→USR	HbncVisitRepo	Child
HbncPart1	db_iemr	t_hbnc_part1	id	ben_id→BEN, visit_no	HbncPart1Repo	Child
HbncPart2	db_iemr	t_hbnc_part2	id	ben_id→BEN, visit_no	HbncPart2repo	Child
HbncVisitCard	db_iemr	t_hbnc_visit_card	id	ben_id→BEN, visit_no	HbncVisitCardRepo	Child
HBYC	db_iemr	t_hbyc	id	ben_id→BEN	—	Child

Entity	DB	Table	PK	Link columns → target	Repository	Module
HbycChildVisit	db_iemr	t_hbyc_child_visits	id	beneficiary_id→BEN, household_id→HH, user_id→USR	HbycRepo	Child
ChildVaccination	db_iemr	t_childvaccinedetail1	ID	BeneficiaryRegID→REG, BenVisitID→VIS, vaccineId→m_immunizationservicevaccination	ChildVaccinationRepo	Child
Vaccine	db_iemr	m_immunizationservicevaccination	VaccinationID	dependantVaccineId→self	VaccineRepo	Child
SamVisit	db_iemr	t_sam_visit	id	beneficiary_id→BEN, household_id→HH, user_id→USR	SamVisitRepository	Child
OrsDistribution	db_iemr	t_ors_distribution	id	beneficiary_id→BEN, household_id→HH, user_id→USR	OrsDistributionRepo	Child
AdolescentHealth	db_iemr	adolescent_health	id	ben_id→BEN, userID→USR	AdolescentHealthRepo	Child
CDR	db_iemr	t_cdr	id	ben_id→BEN	CdrRepo	Death review
MDSR	db_iemr	t_mdsr	id	ben_id→BEN	MdsrRepo	Death review
ScreeningMalaria	db_iemr	screening_malaria	id	ben_id→BEN, house_hold_details_Id→HH, user_id→USR, visit_Id	DiseaseMalariaRepository	Disease control
MalariaFollowUp	db_iemr	malaria_follow_up	id	ben_id→BEN, house_hold_details_id→HH, user_Id→USR, disease_Id	MalariaFollowUpRepository	Disease control
ScreeningKalaAzar	db_iemr	screening_kala_azar	id	ben_id→BEN, house_hold_details_Id→HH, user_id→USR	DiseaseKalaAzarRepository	Disease control
ScreeningAesje	db_iemr	screening_aesje	id	ben_id→BEN, house_hold_details_Id→HH, user_id→USR	DiseaseAESJERepository	Disease control
ScreeningFilariasis	db_iemr	screening_filaria	id	ben_id→BEN, house_hold_details_Id→HH, user_id→USR	DiseaseFilariasisRepository	Disease control
ScreeningLeprosy	db_iemr	screening_leprosy	id	ben_id→BEN, house_hold_details_Id→HH, user_id→USR	DiseaseLeprosyRepository	Disease control
LeprosyFollowUp	db_iemr	leprosy_follow_up	id	ben_id→BEN, visit_number	LeprosyFollowUpRepository	Disease control
MosquitoNetEntity	db_iemr	i_mobilization_mosquito_net	id	beneficiary_id→BEN, household_id→HH, user_id→USR	MosquitoNetRepository	Disease control
IRSRound	db_iemr	irs_round	id	householdId→HH, user_id→USR	IRSRoundRepo	Disease control
MdaDistributionData	db_iemr	t_mda_distribution_data	ID	BeneficiaryId→BEN, HouseHoldId→HH, AshaId→USR, FormId	MdaFormSubmissionRepository	Disease control
ChronicDiseaseVisitEntity	db_iemr	cdtf_visit_details	id	beneficiary_id→BEN, household_id→HH, form_id, user_id→USR	ChronicDiseaseVisitRepository	Disease control
EyeCheckupVisit	db_iemr	t_eye_checkup	id	beneficiary_id→BEN, household_id→HH, user_id→USR	EyeCheckUpVisitRepo	Disease control
CbacDetailsImer	db_iemr	t_cbacdetails	Id	BeneficiaryRegId→REG, Visitcode	CbaclemrDetailsRepo	Disease control
TBScreening	db_iemr	tb_screening	id	ben_id→BEN, ben_reg_id→REG, user_id→USR, visitCode	TBScreeningRepo	TB
TBSuspected	db_iemr	tb_suspected	id	ben_id→BEN, benRegID→REG, user_id→USR, nikshay_id	TBSuspectedRepo	TB
TBConfirmedCase	db_iemr	tb_confirmed_cases	id	ben_id→BEN, benRegID→REG, user_id→USR	TBConfirmedTreatmentRepository	TB
StopTBRegistration	db_iemr	tb_stoptb_registration	id	benRegID→REG, healthFacilityId, villageId	StopTBRegistrationRepo	TB
StopTBGeneralOpd	db_iemr	tb_stoptb_general_opd	id	ben_reg_id→REG, benVisitID→VIS, visitCode	StopTBGeneralOpdRepo	TB
StopTBGeneralExamination	db_iemr	tb_stoptb_general_examination	id	beneficiary_reg_id→REG, ben_id→BEN, benVisitID→VIS	StopTBGeneralExaminationRepo	TB
StopTBDiagnostics	db_iemr	tb_stoptb_diagnostics	id	ben_reg_id→REG, visitCode, nikshay_id	StopTBDiagnosticsRepo	TB
DiagnosticOrder	db_iemr	tb_diagnostic_order	id	beneficiary_id→BEN, external_order_id (UK)	DiagnosticOrderRepo	TB
DiagnosticResult	db_iemr	tb_diagnostic_result	id	diagnostic_order_id→tb_diagnostic_order, external_order_id (UK), beneficiary_id→BEN	DiagnosticResultRepo	TB

Entity	DB	Table	PK	Link columns → target	Repository	Module
DiagnosticDocument	db_iemr	tb_diagnostic_document	id	diagnostic_order_id→tb_diagnostic_order, beneficiary_id→BEN	DiagnosticDocumentRepo	TB
DiagnosticProviderToken	db_iemr	tb_diagnostic_provider_token	id	— (UK provider_code+token_type)	DiagnosticProviderTokenRepo	TB
IfaDistribution	db_iemr	t_ifa_distribution	id	beneficiary_id→BEN, house_hold_id→HH, user_id→USR, form_id	IfaDistributionRepository	Campaigns & meetings
IFAFormSubmissionData	db_iemr	t_ifa_distribution_data	id	beneficiary_id→BEN, household_id→HH, user_id→USR, form_id	IFAFormSubmissionRepository	Campaigns & meetings
CampaignOrs	db_iemr	campaign_ors	id	user_id→USR	OrsCampaignRepo	Campaigns & meetings
PulsePolioCampaign	db_iemr	campaign_pulse_polio	id	user_id→USR	PulsePolioCampaignRepo	Campaigns & meetings
FilariasisCampaign	db_iemr	campaign_filariasis_mda	id	user_id→USR	FilariasisCampaignRepo	Campaigns & meetings
DewormingForm	db_iemr	deworming_form	id	user_id→USR	DewormingFormRepo	Campaigns & meetings
AHDForm	db_iemr	ahd_form	id	user_id→USR	AHDFormRepo	Campaigns & meetings
VHNDForm	db_iemr	VHND_form	id	user_id→USR, vhnd_place_id	VhndRepo	Campaigns & meetings
VhncForm	db_iemr	vhnc_form	id	user_id→USR	VhncFormRepo	Campaigns & meetings
PHCReviewForm	db_iemr	phc_review_meeting	id	user_id→USR, place_Id	PHCReviewFormRepo	Campaigns & meetings
MaaMeeting	db_iemr	maa_meeting	id	asha_id→USR	MaaMeetingRepository	Campaigns & meetings
SammelanRecord	db_iemr	sammelan_record	id	asha_id→USR (UK asha_id+meeting_date)	SammelanRecordRepository	Campaigns & meetings
SammelanAttachment	db_iemr	sammelan_attachment	id	sammelan_id→sammelan_record (@ManyToOne)	SammelanAttachmentRepository	Campaigns & meetings
UwinSession	db_iemr	uwin_session_record	id	asha_id→USR	UwinSessionRepository	Campaigns & meetings
IncentiveActivity	db_iemr	m_incentive_activity	id	— (master)	IncentivesRepo	Incentives
IncentiveActivityLangMapping	db_iemr	m_incentive_activity_lang_mapping	id	activity_name→m_incentive_activity (inferred)	IncentiveActivityLangMappingRepo	Incentives
IncentiveActivityRecord	db_iemr	incentive_activity_record	id	activity_id→m_incentive_activity, asha_id→USR, ben_id→BEN, verifiedBy_userId→USR	IncentiveRecordRepo, SupervisorDashboardRepo	Incentives
IncentivePendingActivity	db_iemr	incentive_pending_activity	id	activity_id, record_id→incentive_activity_record, m_incentive_id→m_incentive_activity, user_id→USR	IncentivePendingActivityRepository	Incentives
BadgeConfig	db_iemr	badge_config	config_key	—	BadgeConfigRepo	Incentives

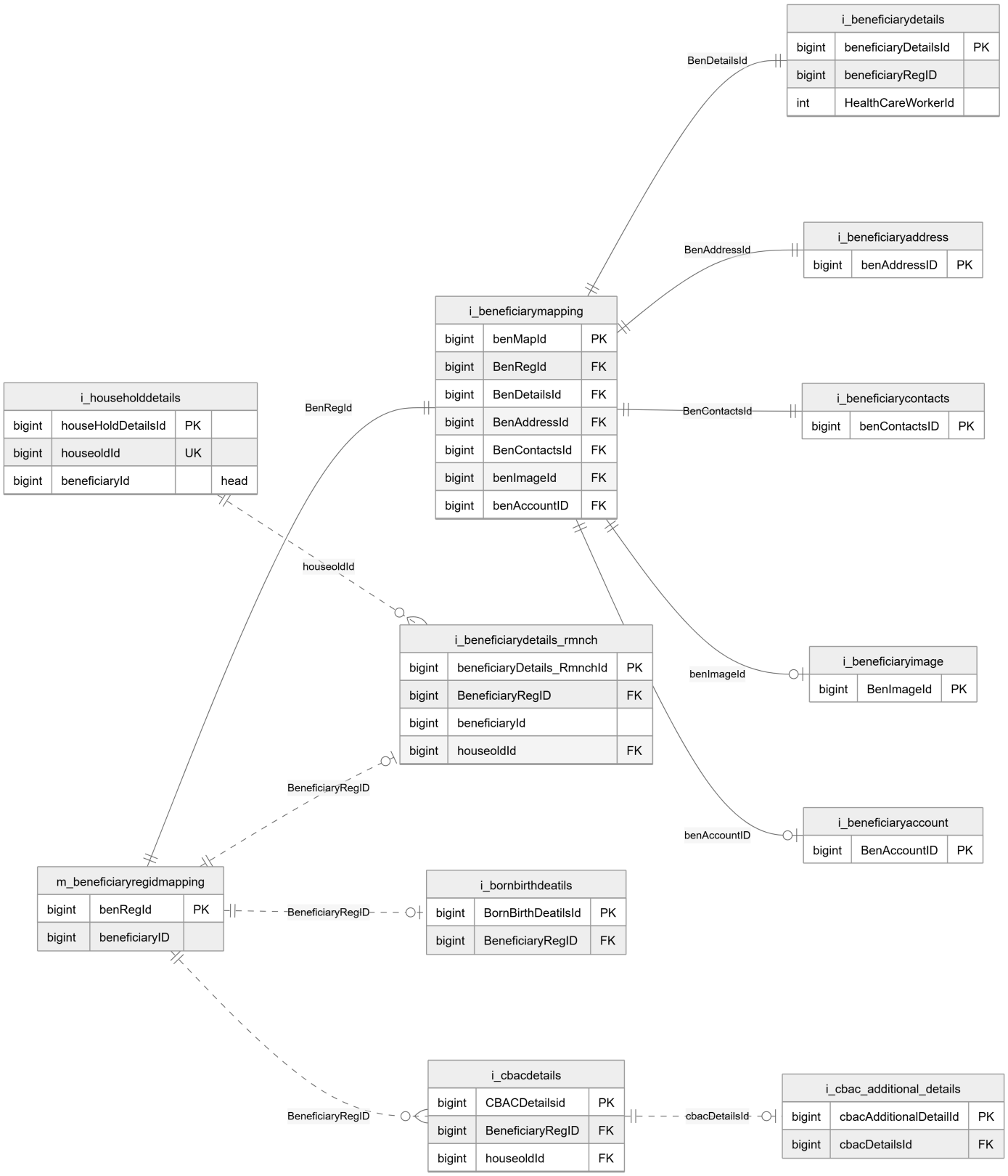
Entity	DB	Table	PK	Link columns → target	Repository	Module
BadgeEarned	db_iemr	badge_earned	id	user_id→USR, badge_id (UK user_id+badge_id+level+award_key)	BadgeEarnedRepo	Incentives
BadgeStreakFreeze	db_iemr	badge_streak_freeze	id	user_id→USR, badge_id	BadgeStreakFreezeRepo	Incentives
DynamicForm	db_iemr	t_dynamic_form	formId	—	DynamicFormRepo	Dynamic forms
FormVersion	db_iemr	t_form_version	versionId	formId→t_dynamic_form (@ManyToOne)	FormVersionRepo	Dynamic forms
FormSection	db_iemr	t_form_section	sectionId	version_id→t_form_version (@ManyToOne)	FormSectionRepo	Dynamic forms
SectionQuestion	db_iemr	t_section_question	questionId	sectionId→t_form_section (@ManyToOne)	SectionQuestionRepo	Dynamic forms
QuestionOption	db_iemr	t_question_option	optionId	questionId→t_section_question (@ManyToOne)	QuestionOptionRepo	Dynamic forms
QuestionValidation	db_iemr	t_question_validation	validationId	questionId→t_section_question (@ManyToOne)	QuestionValidationRepo	Dynamic forms
OptionCondition	db_iemr	t_option_condition	conditionId	optionId→t_question_option, targetQuestionId→t_section_question, targetSectionId→t_form_section (@ManyToOne)	OptionConditionRepo	Dynamic forms
FormResponse	db_iemr	t_form_response	responseId	beneficiaryId→BEN, formId, versionId, officerId→USR	FormResponseRepo	Dynamic forms
SectionResponse	db_iemr	t_section_response	sectionResponseId	responseId→t_form_response, sectionId→t_form_section	SectionResponseRepo	Dynamic forms
QuestionResponse	db_iemr	t_question_response	questionResponseId	sectionResponseId, questionId, optionId	QuestionResponseRepo	Dynamic forms
BenVisitDetail	db_iemr	t_benvisitdetail	BenVisitID	BeneficiaryRegID→REG, VisitCode	BenVisitDetailsRepo	OPD & clinical
BenChiefComplaint	db_iemr	t_benchiefcomplaint	ID	BeneficiaryRegID→REG, BenVisitID→VIS	BenChiefComplaintRepo	OPD & clinical
BenPhysicalVitalDetail	db_iemr	t_phy_vitals	ID	BeneficiaryRegID→REG, BenVisitID→VIS	BenPhysicalVitalRepo	OPD & clinical
BenAnthropometryDetail	db_iemr	t_phy_anthropometry	ID	BeneficiaryRegID→REG, BenVisitID→VIS	BenAnthropometryRepo	OPD & clinical
PhyGeneralExamination	db_iemr	t_Phy_GeneralExam	ID	BeneficiaryRegID→REG, BenVisitID→VIS	PhyGeneralExaminationRepo	OPD & clinical
BenReferDetails	db_iemr	t_benreferdetails	benReferID	BeneficiaryRegID→REG, BenVisitID→VIS, referredToInstituteID	BenReferDetailsRepo	OPD & clinical
PrescriptionDetail	db_iemr	t_prescription	PrescriptionID	BeneficiaryRegID→REG, BenVisitID→VIS	PrescriptionDetailRepo	OPD & clinical
PrescribedDrugDetail	db_iemr	t_prescribeddrug	PrescribedDrugID	PrescriptionID→t_prescription, BeneficiaryRegID→REG, BenVisitID→VIS	PrescribedDrugDetailRepo	OPD & clinical
BenFlowStatus	db_iemr	i_ben_flow_outreach	ben_flow_id	beneficiary_reg_id→REG, beneficiary_id→BEN	BenFlowStatusRepo	OPD & clinical
GeneralOpdData	db_iemr	i_ben_flow_outreach	ben_flow_id	beneficiary_reg_id→REG, beneficiary_visit_id, beneficiary_id→BEN	GeneralOpdRepo	OPD & clinical
GeneralOpdEntry	db_iemr	general_opd_entry	id	asha_id→USR, beneficiary_id→BEN	BeneficiaryRepository	OPD & clinical
User	db_iemr	m_user	UserID	DesignationID, ServiceProviderID	EmployeeMasterRepo	Users & notifications
UserServiceRole	db_iemr	v_userservicerolemapping (view)	USRMappingID	UserID→m_user, RoleID, ProviderServiceMapID, WorkingLocationID	UserServiceRoleRepo	Users & notifications
AshaWorker	db_iemr	asha_profile	id	employee_id→m_user (inferred), ProviderServiceMapID	AshaProfileRepo	Users & notifications
AshaSupervisorMapping	db_iemr	asha_supervisor_mapping	id	supervisorUserID→USR, ashaUserID→USR, facilityID	AshaSupervisorLoginRepo, FacilityLoginRepo	Users & notifications
UserFcmTokenData	db_iemr	user_fcm_tokens	id	user_id→USR	UserFcmTokenRepo	Users & notifications
Notification	db_iemr	notification	id	sender_user_id→USR, receiver_user_id→USR	NotificationRepository	Users & notifications

Entity	DB	Table	PK	Link columns → target	Repository	Module
SMSTemplate	db_iemr	m_smstemplate	SMSTemplateID	SMSTypeID, ProviderServiceMapID	SMSTemplateRepository	Users & notifications
OtpBeneficiary	db_iemr	m_otp_beneficiary	id	beneficiaryId→BEN	—	Users & notifications
RMNCHBeneficiaryDetailsRmnch	db_identity	i_beneficiarydetails_rmnch	beneficiaryDetails_RmnchId	BeneficiaryRegID→REG, beneficiaryId, houseoldId→HH, RelatedBeneficiaryIds	BeneficiaryRepo	Beneficiary & household
RMNCHHouseHoldDetails	db_identity	i_householddetails	houseHoldDetailsId	houseoldId (business id), BeneficiaryRegID→REG, beneficiaryId (head)	HouseHoldRepo	Beneficiary & household
RMNCHBornBirthDetails	db_identity	i_bornbirthdeatils	BornBirthDeatilsId	BeneficiaryRegID→REG, beneficiaryId	—	Beneficiary & household
CbacDetails	db_identity	i_cbacdetails	CBACDetailsid	BeneficiaryRegID→REG, beneficiaryId, houseoldId→HH	CbacRepo	Beneficiary & household
CbacAdditionalDetails	db_identity	i_cbac_additional_details	cbacAdditionalDetailId	cbacDetailsId→i_cbacdetails	CbacAdditionalDetailRepo	Beneficiary & household
RMNCHMBeneficiaryregidmapping	db_identity	m_beneficiaryregidmapping	benRegId	beneficiaryID	—	Beneficiary & household
RMNCHMBeneficiarymapping	db_identity	i_beneficiarymapping	benMapId	BenRegId→REG, BenDetailsId, BenAddressId, BenContactsId, benImageId, benAccountID, BenConsentId	—	Beneficiary & household
RMNCHMBeneficiarydetail	db_identity	i_beneficiarydetails	beneficiaryDetailsId	beneficiaryRegID→REG, HealthCareWorkerId	—	Beneficiary & household
RMNCHMBeneficiaryaddress	db_identity	i_beneficiaryaddress	benAddressID	Perm/Curr/Emer State, District, Village IDs	—	Beneficiary & household
RMNCHMBeneficiarycontact	db_identity	i_beneficiarycontacts	benContactsID	—	—	Beneficiary & household
RMNCHMBeneficiaryImage	db_identity	i_beneficiaryimage	BenImageId	—	—	Beneficiary & household
RMNCHMBeneficiaryAccount	db_identity	i_beneficiaryaccount	BenAccountID	—	—	Beneficiary & household

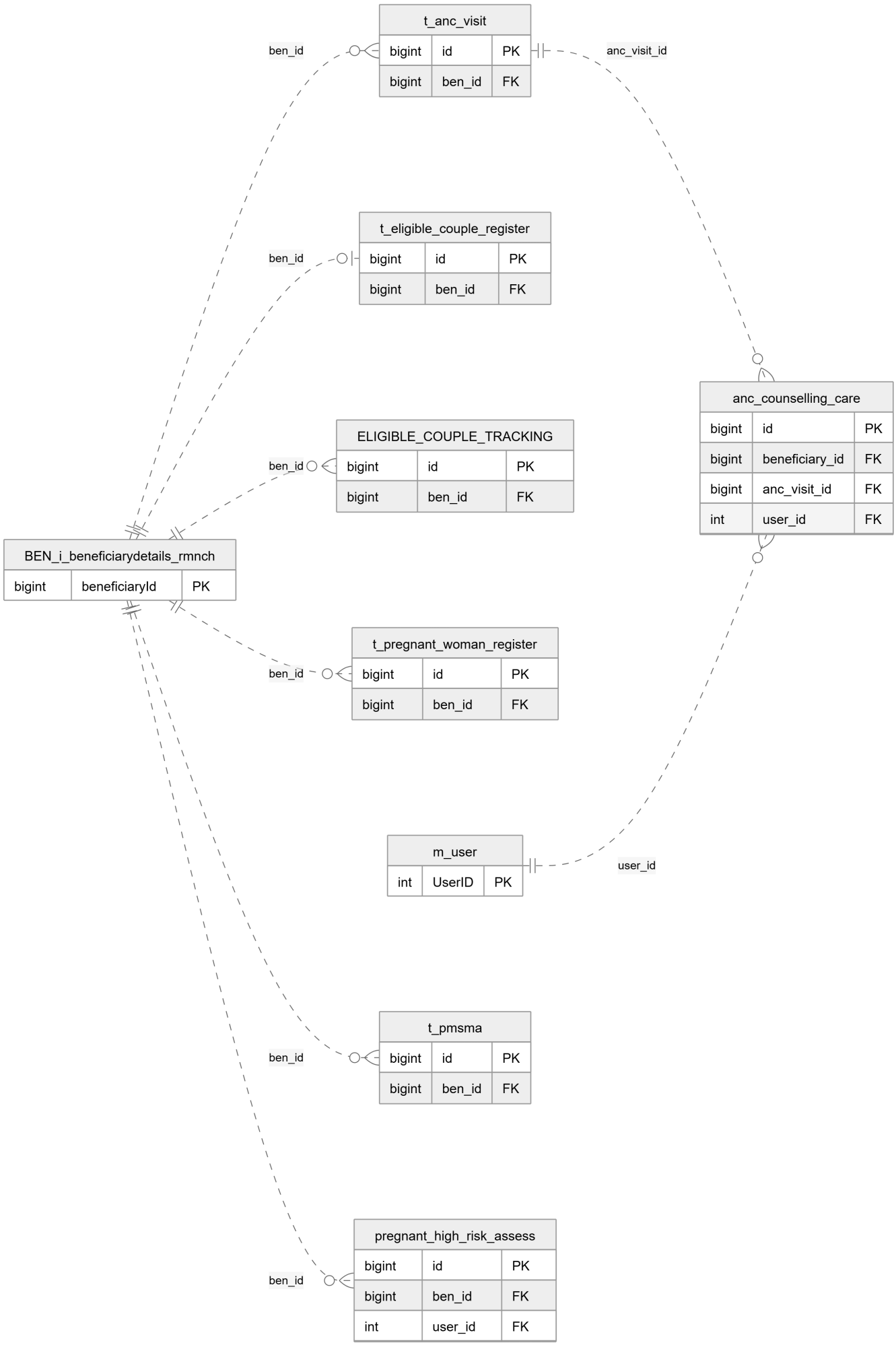
A2. ER diagrams by module

Solid lines are JPA associations. Dashed lines are logical links through shared id columns (no database foreign key). Large modules are split into parts; each part repeats the hub table.

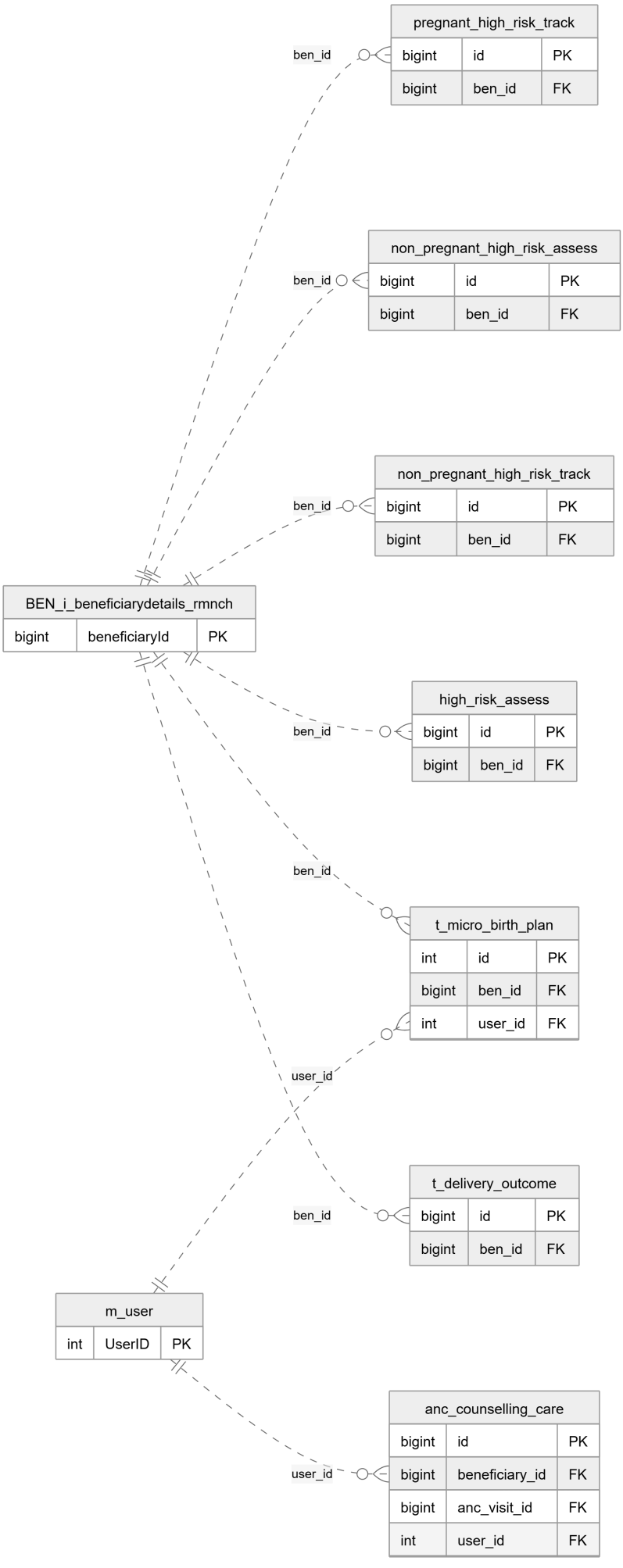
Beneficiary & household (db_identity)



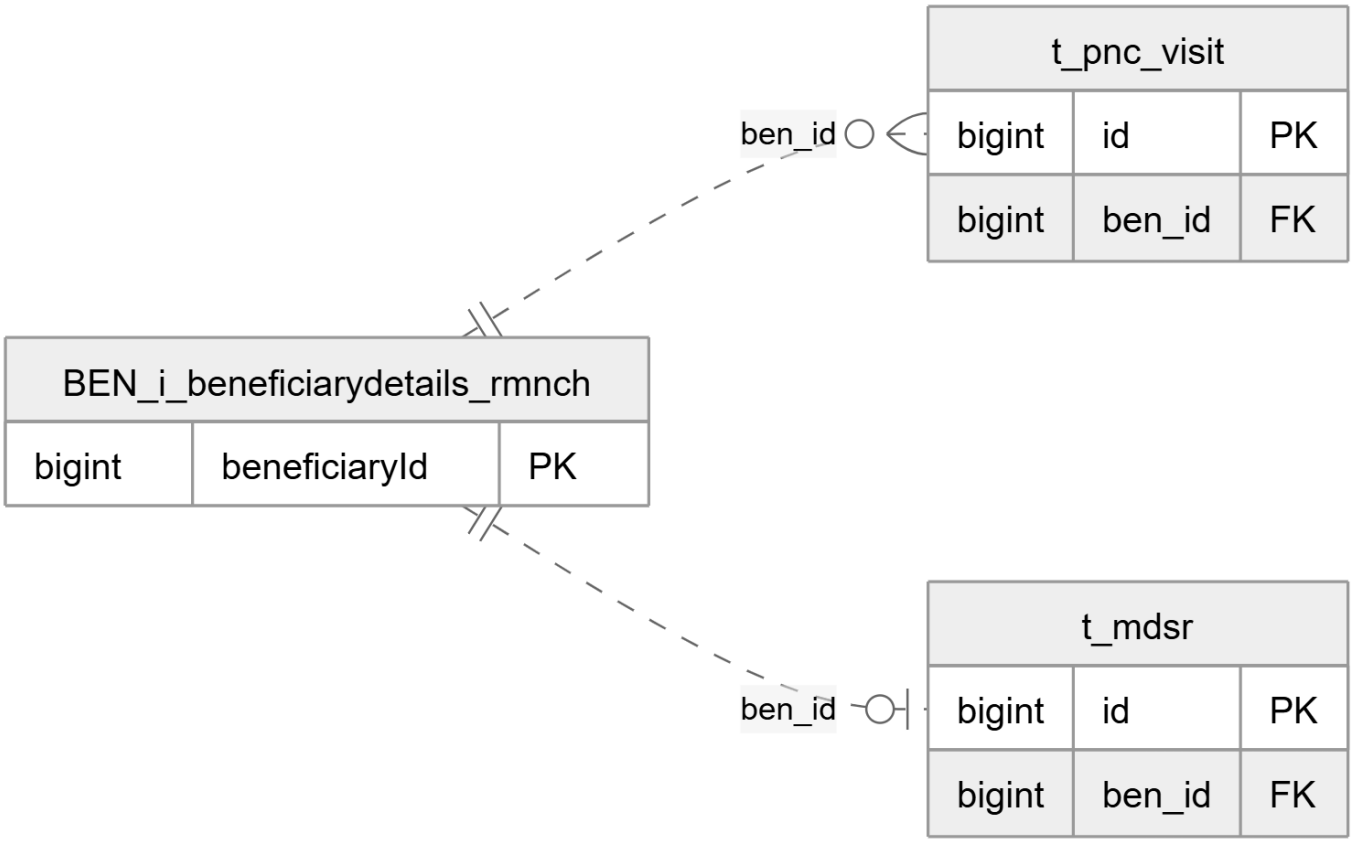
Eligible couple & maternal (db_iemr), part 1 of 3



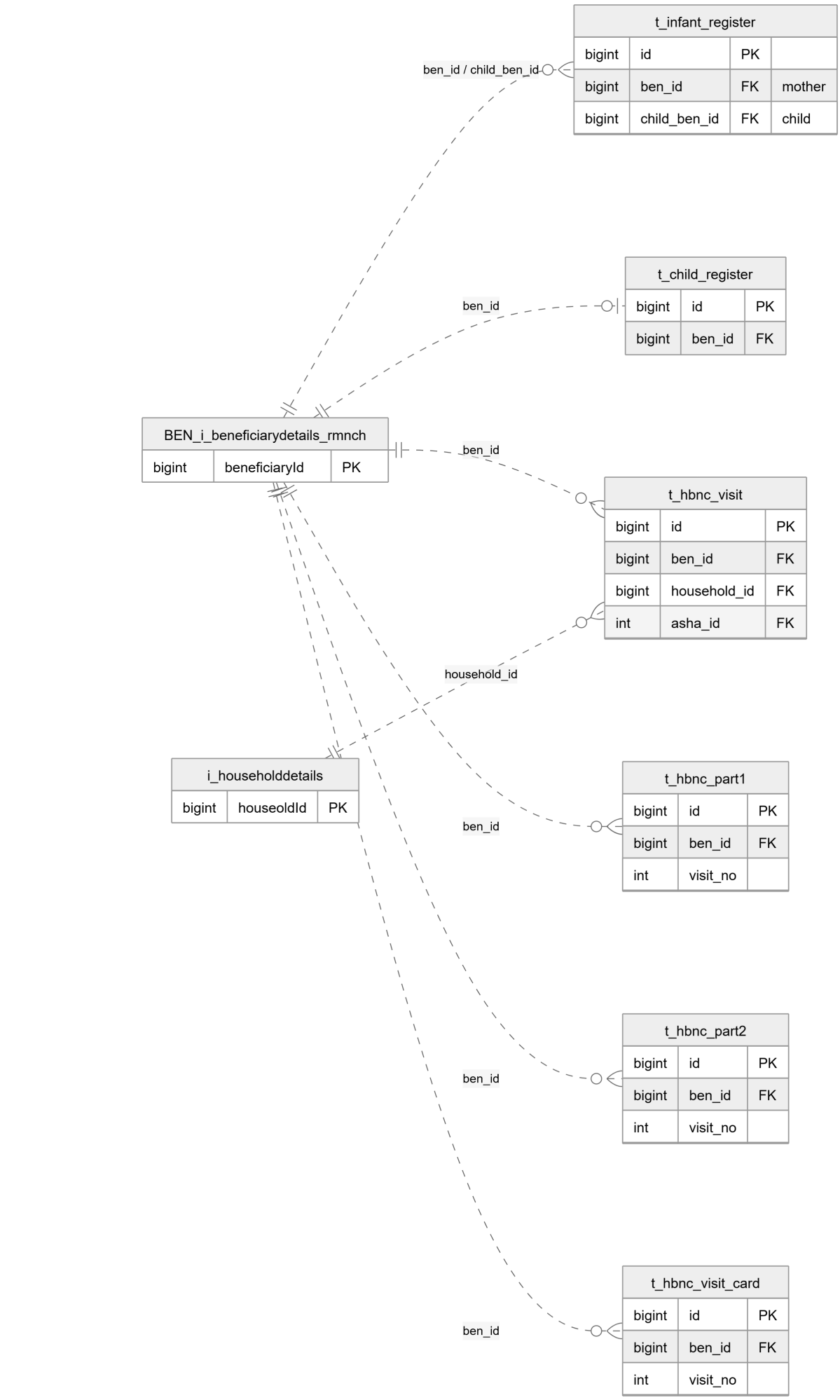
Eligible couple & maternal (db_iemr), part 2 of 3



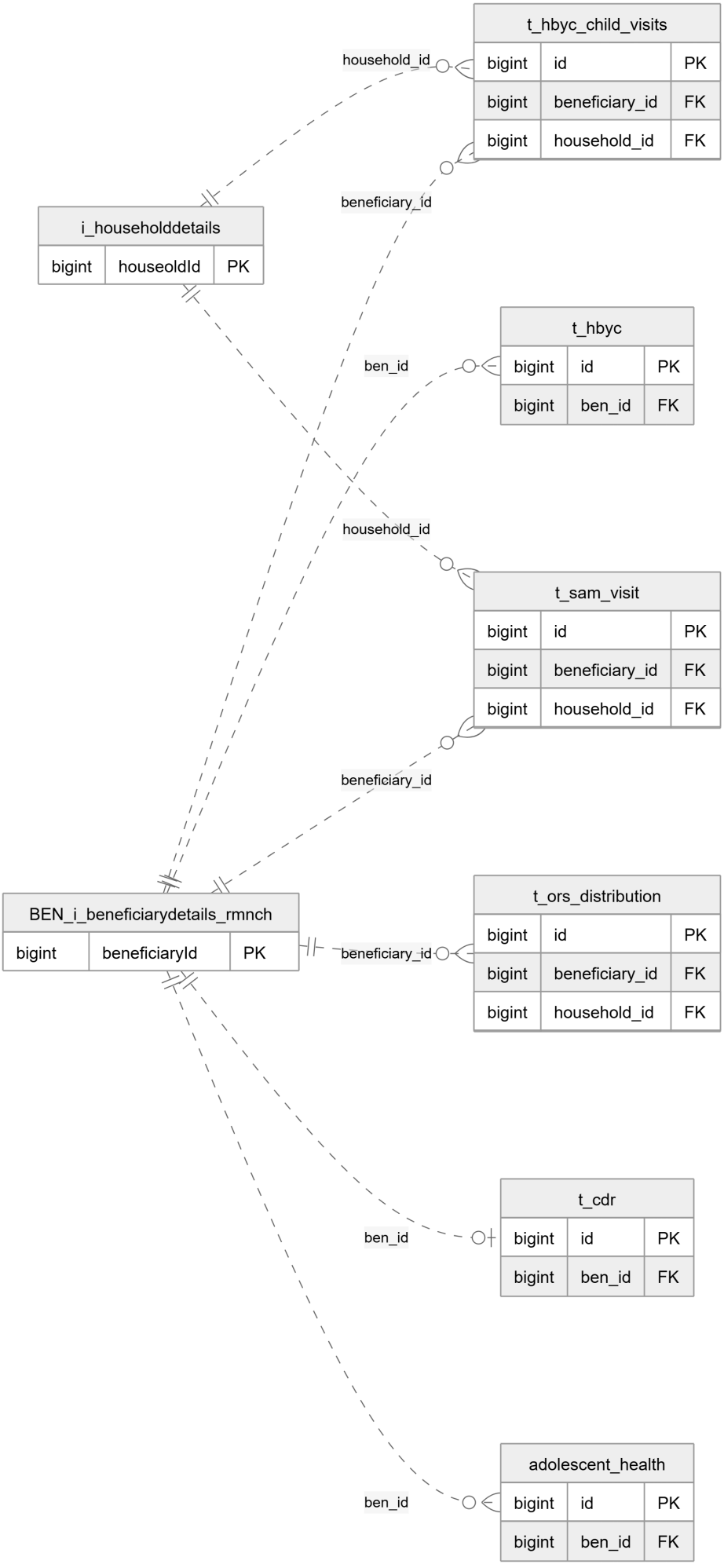
Eligible couple & maternal (db_iemr), part 3 of 3



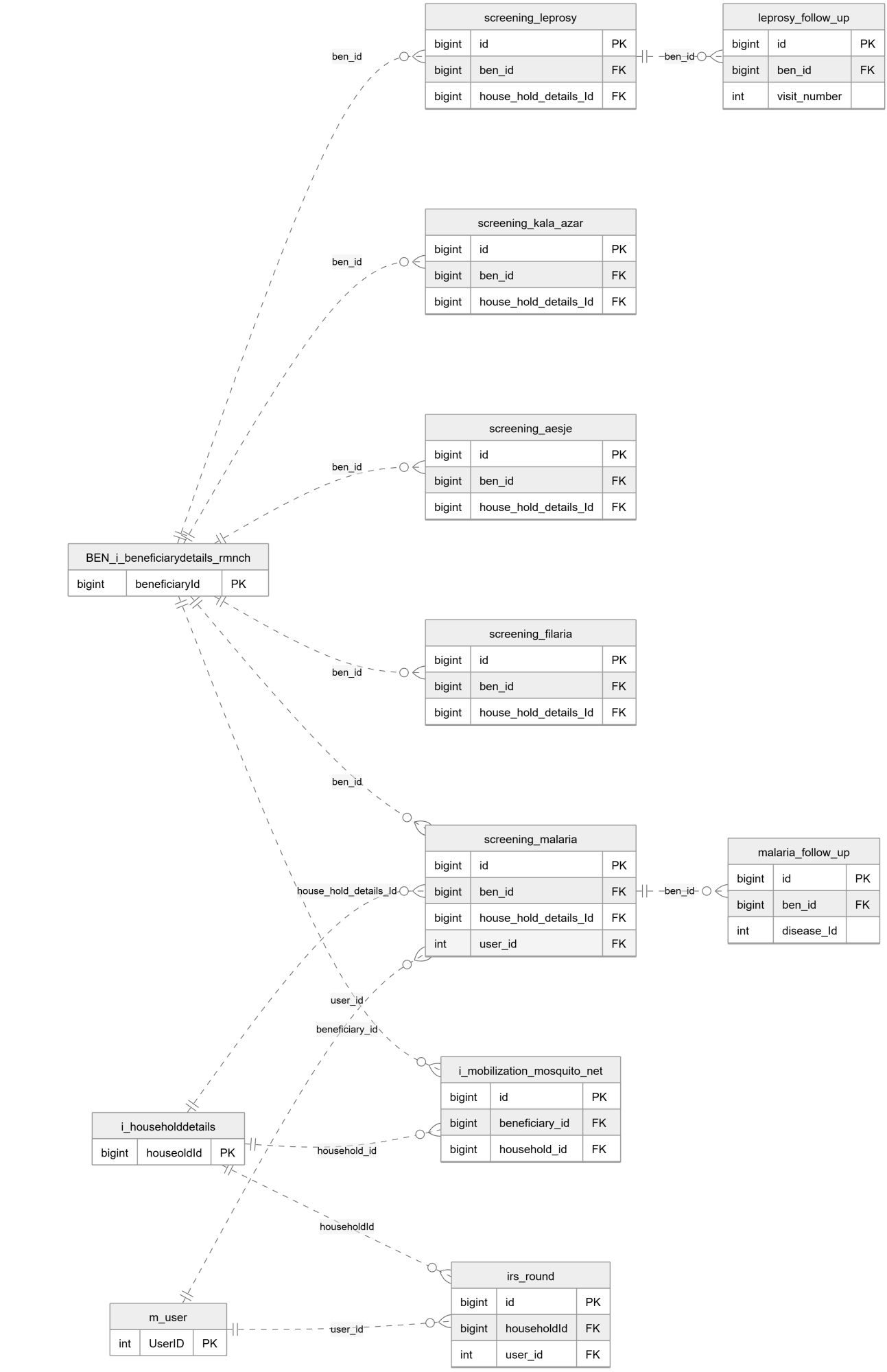
Child: infant, HBNC, HBYC, immunization, SAM (db_iemr), part 1 of 2



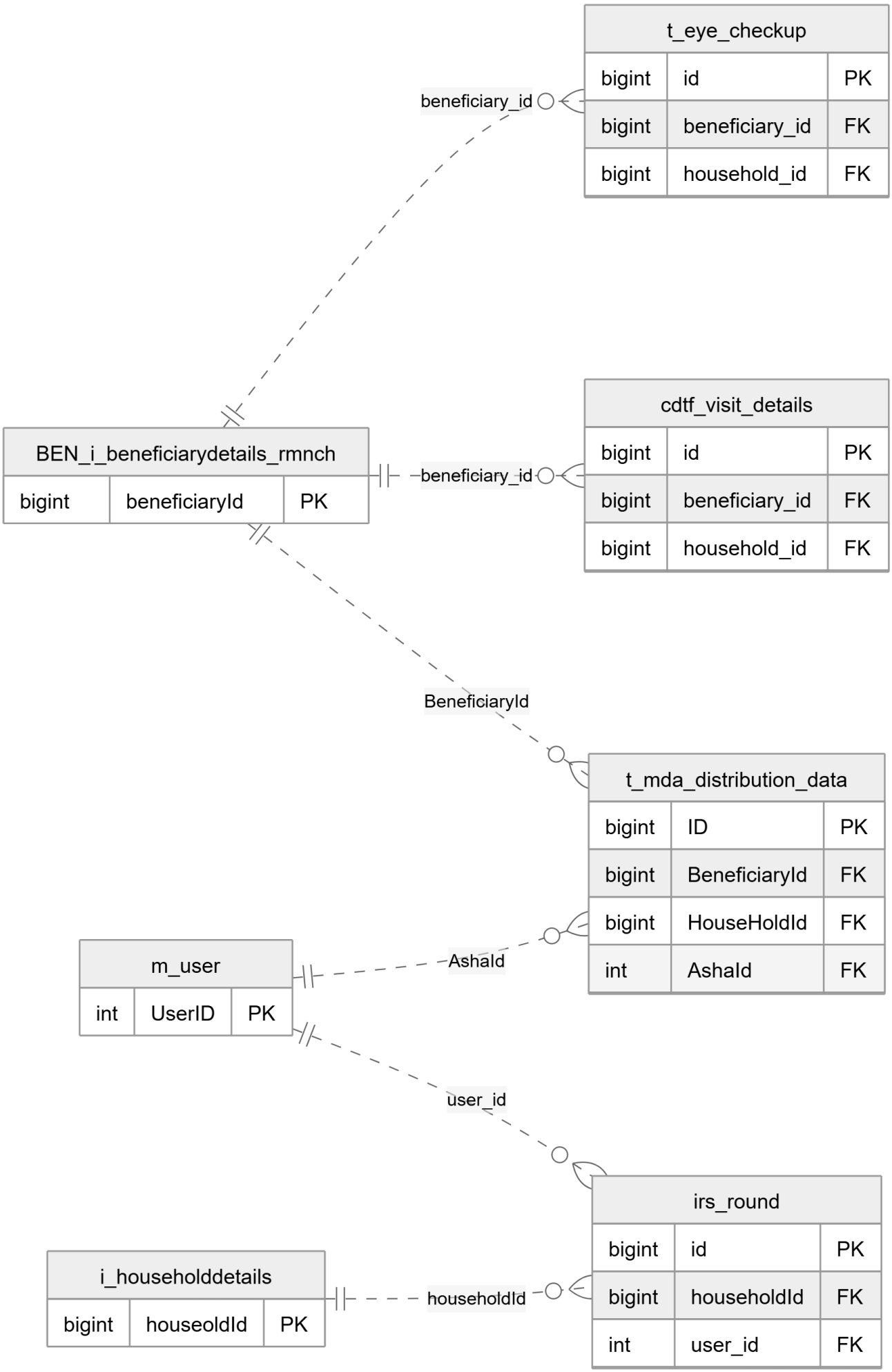
Child: infant, HBNC, HBYC, immunization, SAM (db_iemr), part 2 of 2



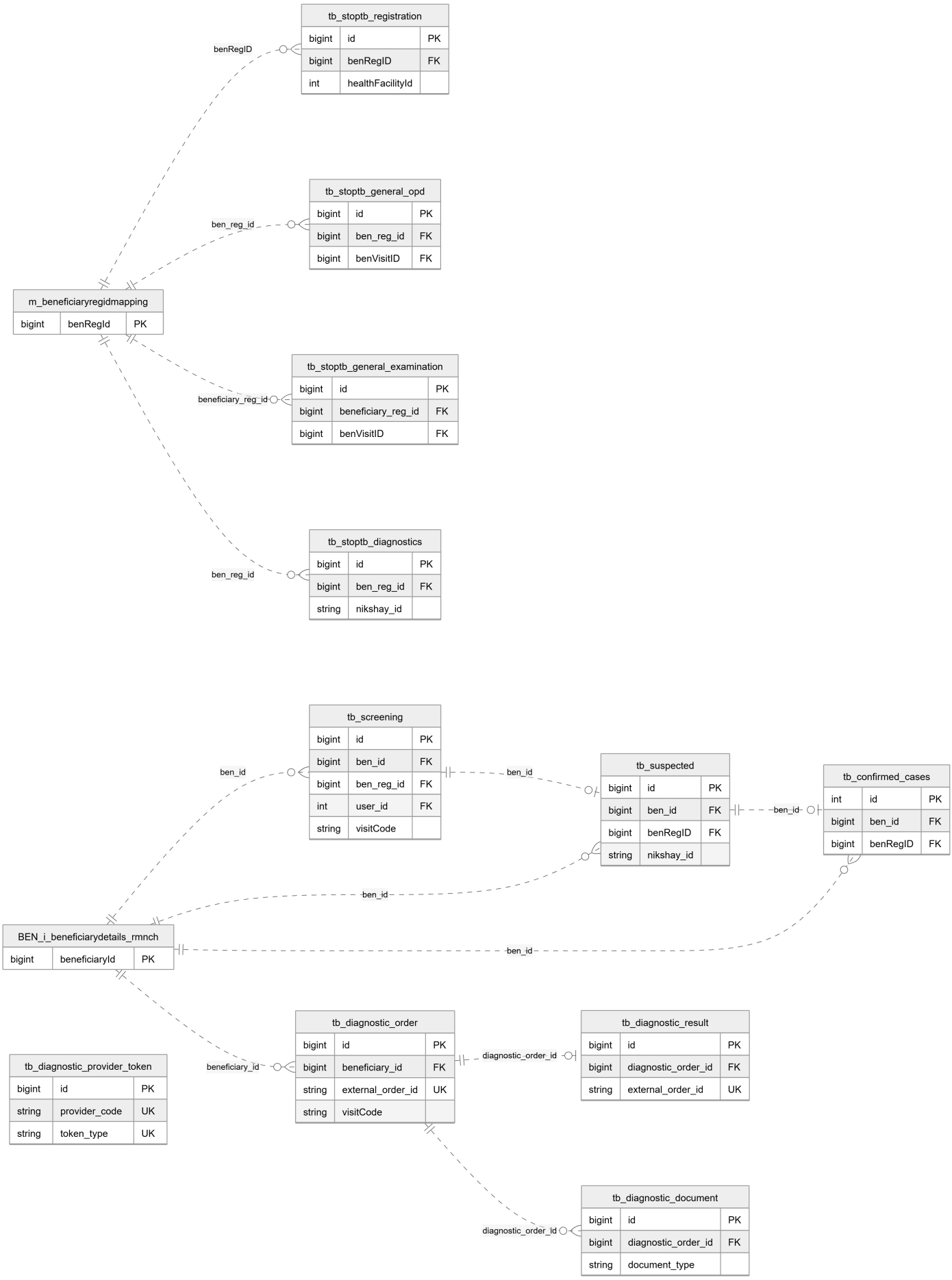
Disease control and vector control (db_iemr), part 1 of 2



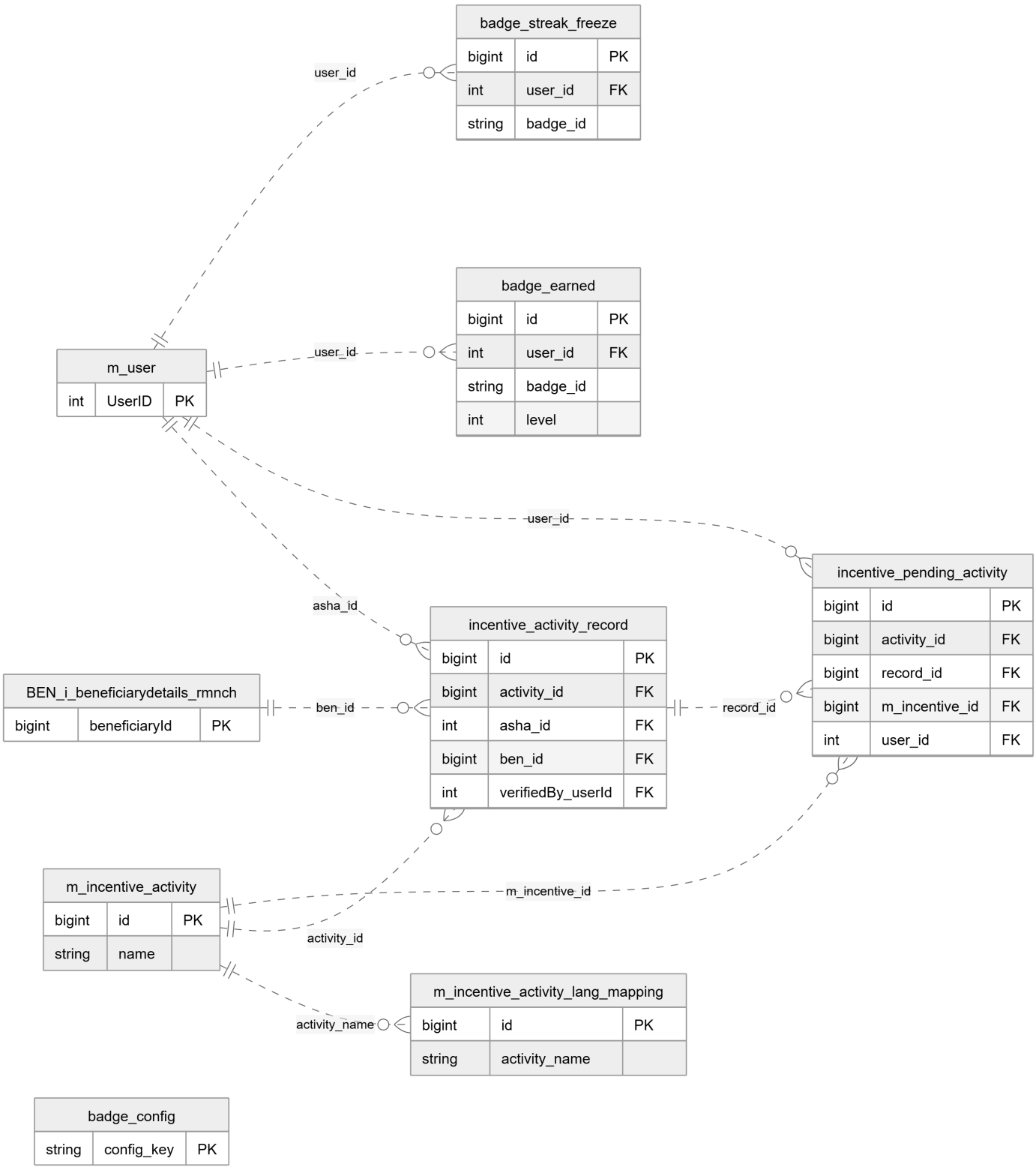
Disease control and vector control (db_iemr), part 2 of 2



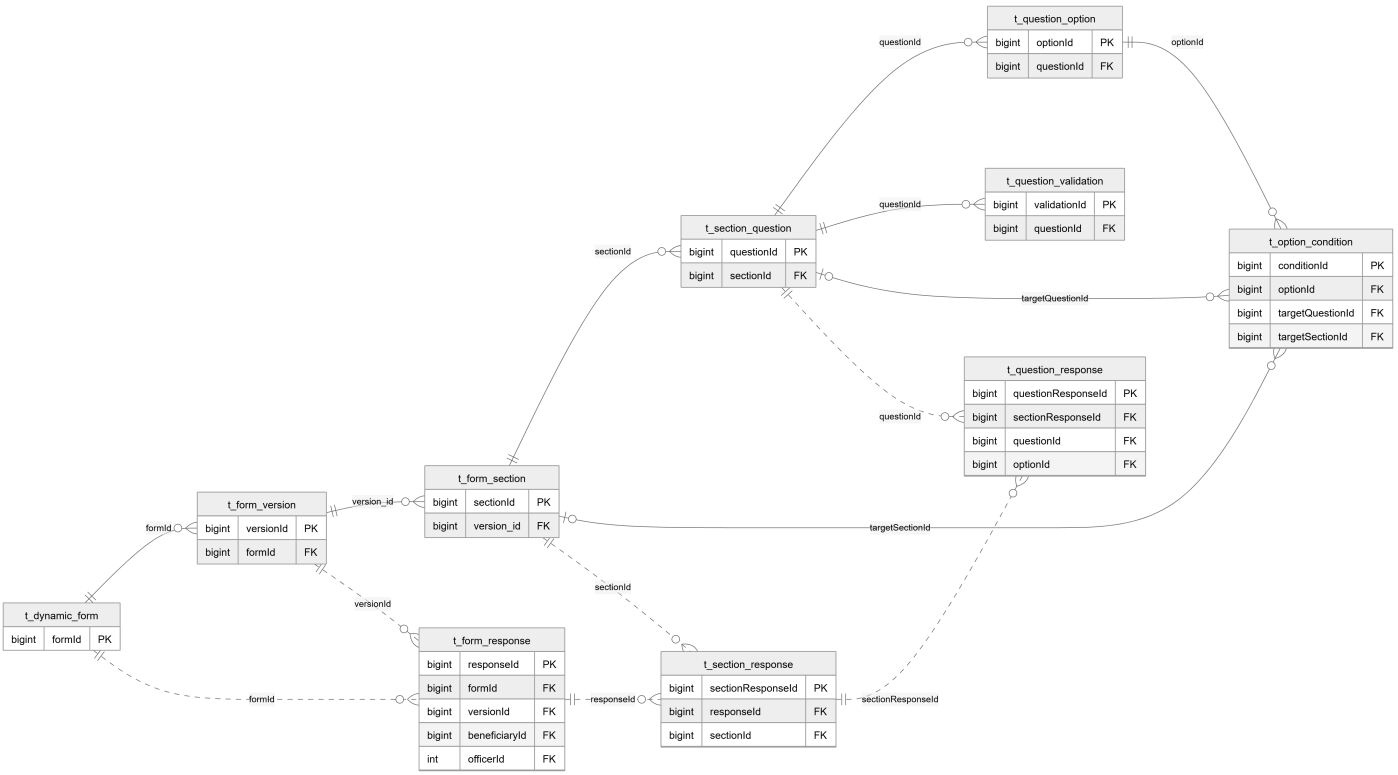
TB, Stop-TB and diagnostics (db_iemr)



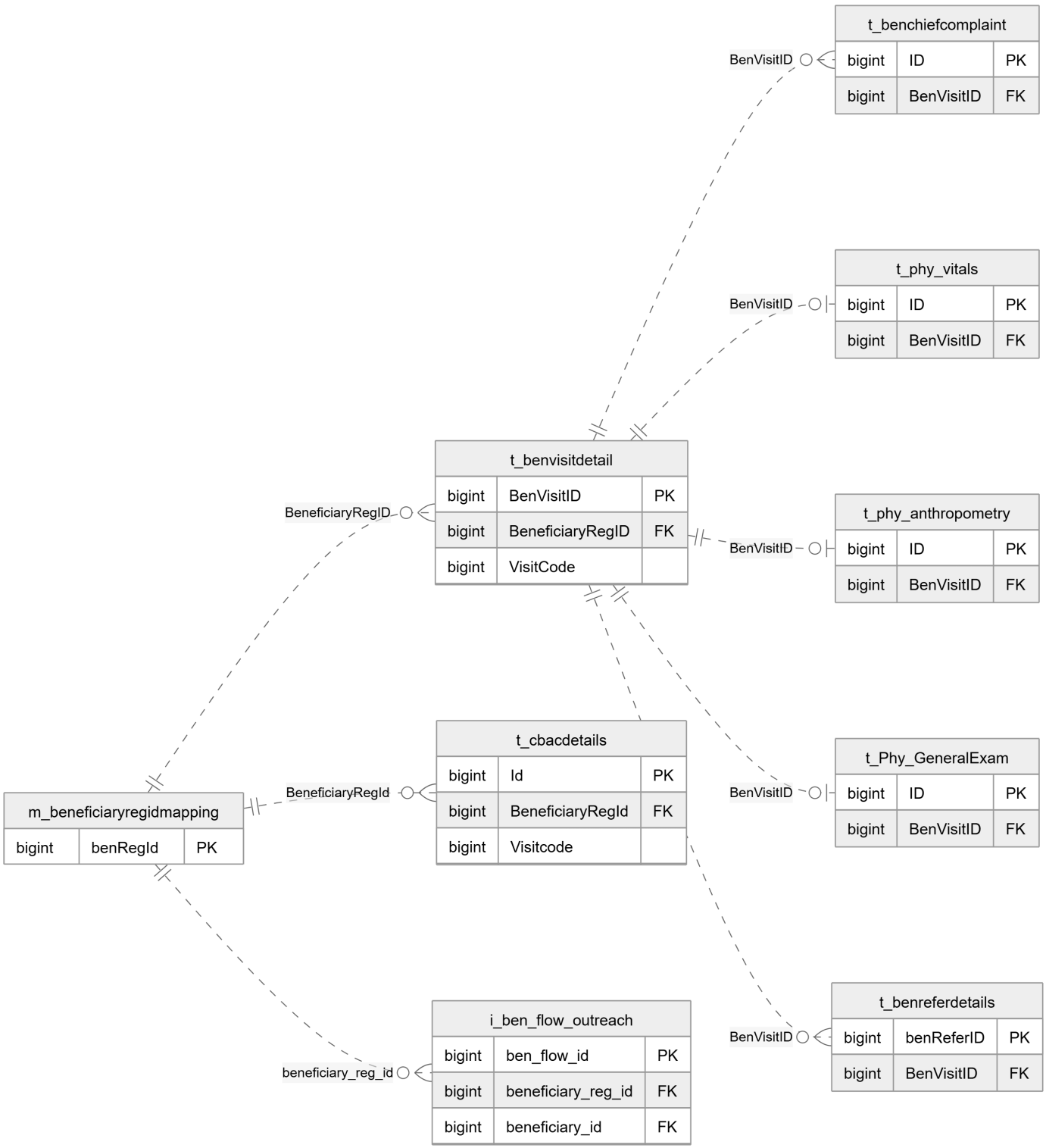
Incentives & gamification (db_iemr)



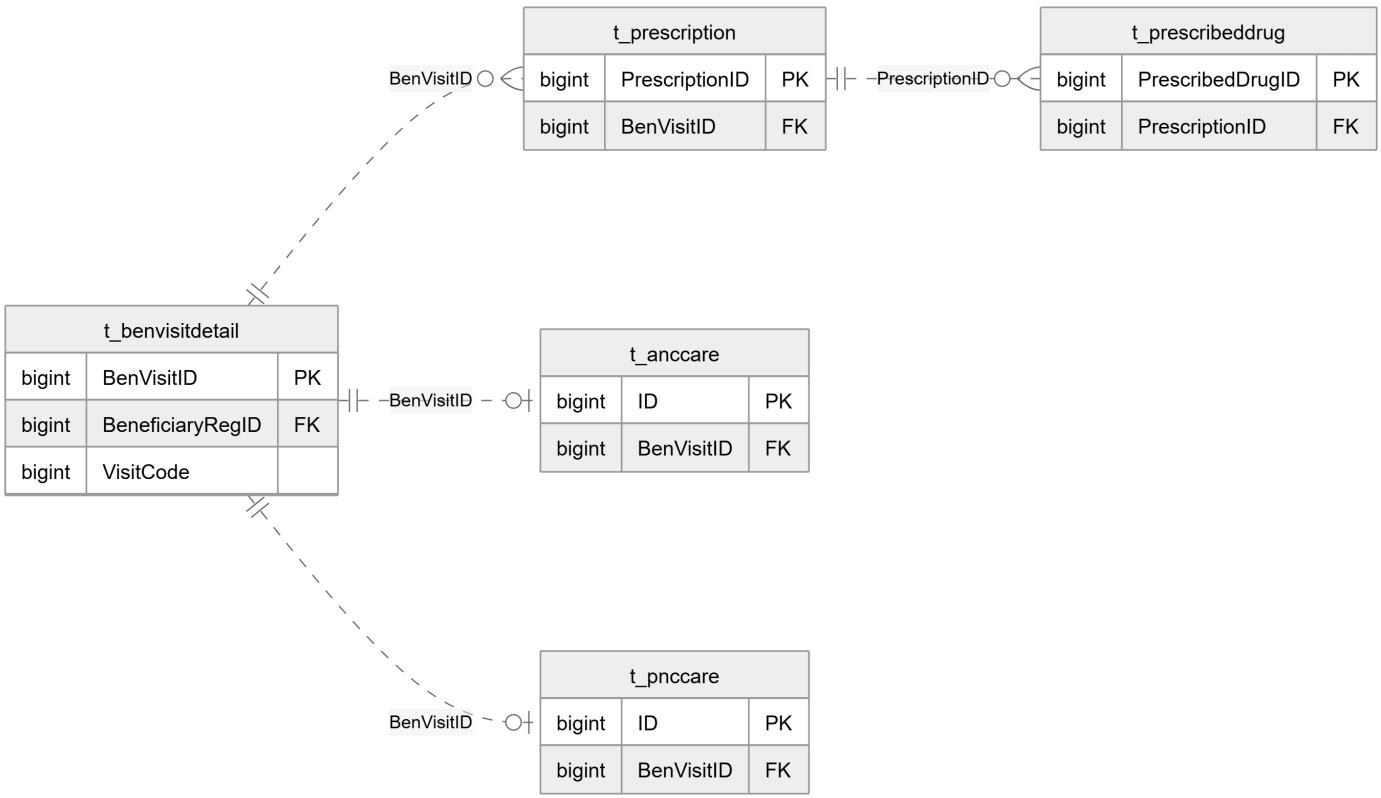
Dynamic forms (db_iemr)



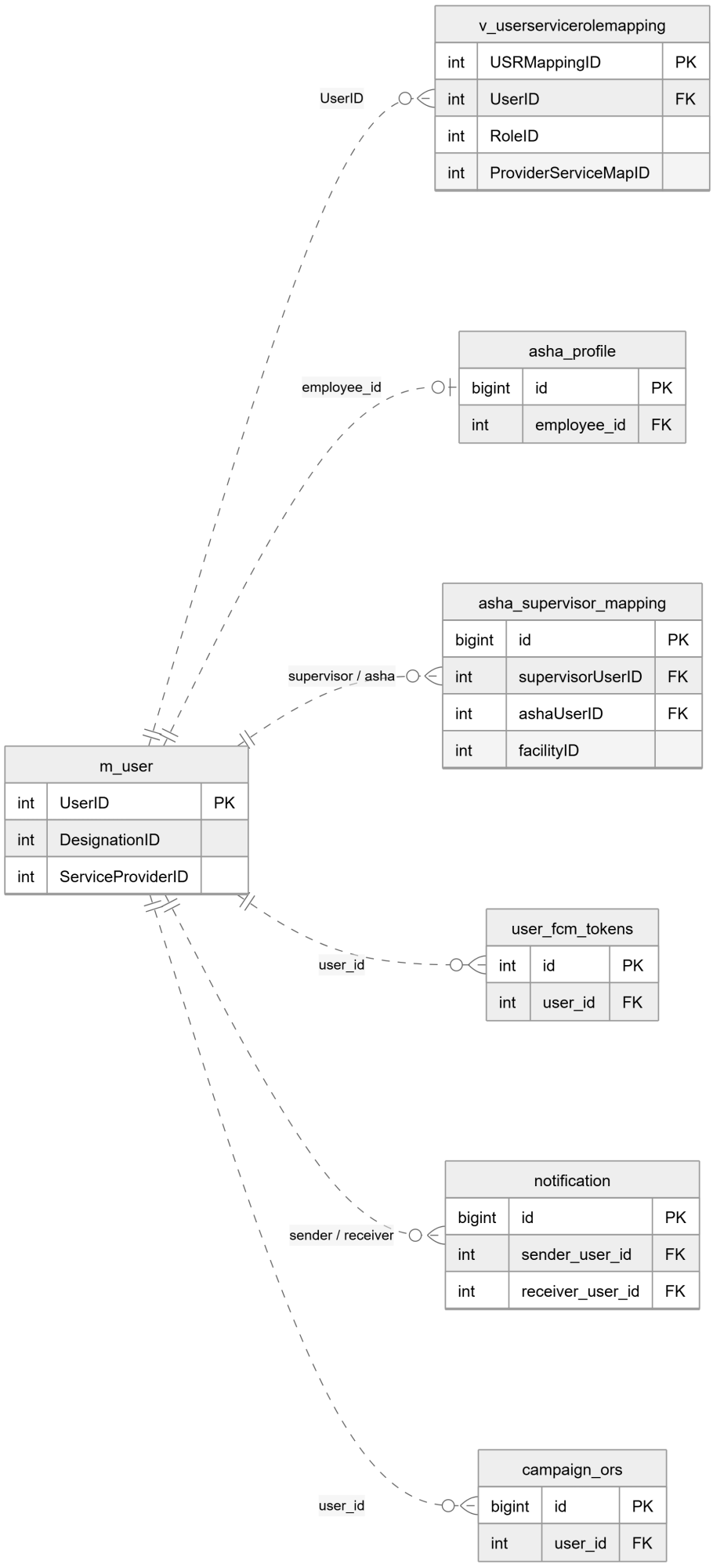
General OPD & clinical visit (db_iemr), part 1 of 2



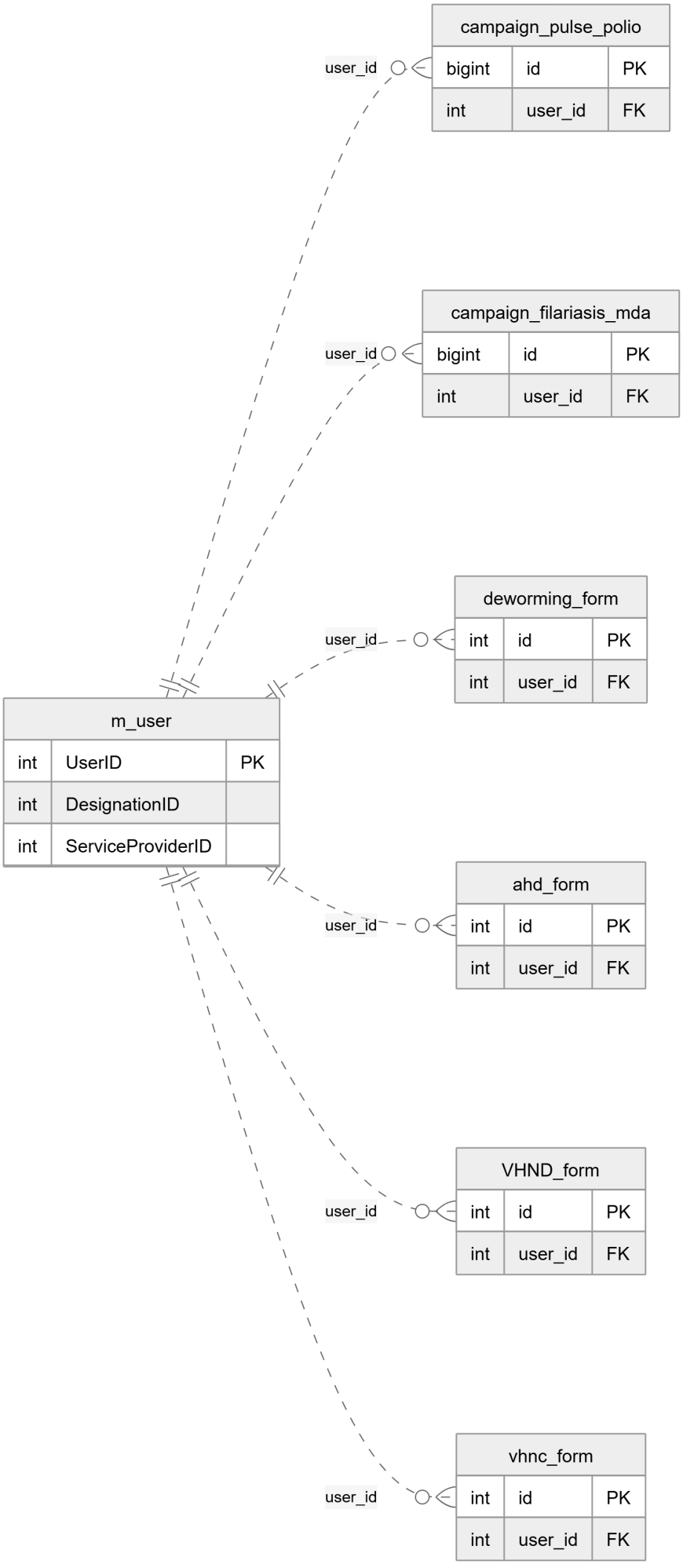
General OPD & clinical visit (db_iemr), part 2 of 2



Campaigns, meetings, users & notifications (db_iemr), part 1 of 3



Campaigns, meetings, users & notifications (db_iemr), part 2 of 3



Campaigns, meetings, users & notifications (db_iemr), part 3 of 3

