CDP-TS-0001

Technical Specification

CDP — Canonical Data Platform

Version	v0.1
Issued	September 2026
Status	Reference build — exercised against synthetic data only
Publisher	VUIT Data Labs
Classification	Public
Licence of the described work	Business Source License 1.1 (converts to Apache-2.0 four years after each release)

This document specifies the CDP reference build at a level sufficient to assess, size and challenge it without a supplier in the room. It states its limitations as plainly as its design. Specification-level detail only: generator design, source-to-canonical mapping sets, the generated test suite and service implementations are released to authorised evaluators.

Contents

01	Purpose, scope and audience	2
02	Conformance and standards baseline	4
03	System context and architecture	5
04	Canonical data model	7
05	Ontology and semantic layer	10
06	Ingestion specification	13
07	Transformation specification	15
08	Data quality and reconciliation	17
09	Interfaces	19
10	Security, information governance and access control	22
11	Non-functional requirements	26
12	Operations	28
13	Exit and portability	30
14	Limitations, assumptions and roadmap	32
15	Glossary and normative references	34

01 Purpose, scope and audience

This document specifies the CDP reference build at a level sufficient for a Trust CIO, technical director, data architect or IG lead to assess it, size it and challenge it without a supplier in the room.

1.1 Purpose

CDP is a Trust-owned canonical data repository built on the published NHS Canonical Data Model (CDM). This specification defines its architecture, data model conventions, ingestion and transformation contracts, quality and reconciliation regime, interfaces, security model, non-functional envelope, operating model and exit position.

It is written to be assessable. Every claim in it is either a design commitment, a stated assumption, or an explicit limitation. Where v0.1 does not yet do something, the document says so rather than describing an intention as a feature.

1.2 In scope

- The canonical repository: schema conventions, domain inventory, key strategy, temporality and value-set binding.
- The ingestion contract between Trust source systems and the raw landing zone.
- The transformation layering, materialisation strategy and change-control workflow.
- The data-quality test taxonomy, severity model, publication gate and reconciliation evidence.
- The serving interfaces: FHIR R4 / UK Core read API, governed SQL access and bulk export.
- Authentication, role-based and row-level access control, pseudonymisation boundary and audit.
- Non-functional requirements, deployment topology, operations and support model.
- Portability and exit: the artefacts a Trust holds and how it reproduces the repository without the supplier.

1.3 Out of scope

- The schema generator, the source-to-canonical mapping tables and the generated test suite. These are build artefacts released to authorised evaluators, not published specification.
- Write-back or transactional interfaces into source systems. CDP v0.1 is read-only downstream of the source of truth.
- Clinical decision support, direct care applications and anything requiring clinical safety certification as a medical device.
- National data flows, secondary-uses submissions and commissioning returns, which remain the responsibility of existing Trust processes.
- Commercial terms. Licensing and pricing structure are covered separately under Commercials and Licence.

1.4 Audience and reading paths

READER	START AT	THEN READ
CIO / technical director	§3 Architecture	§5 Ontology, §11 Non-functional, §13 Exit and portability
Data architect	§4 Canonical data model	§5 Ontology, §6 Ingestion, §7 Transformation
Analytics lead	§5 Ontology and semantic layer	§8 Quality and reconciliation, §9 Interfaces
IG / DPO / SIRO	§10 Security, IG and access control	§14 Limitations
Infrastructure / platform team	§12 Operations	§11 Non-functional
Procurement / commercial	§2 Conformance	§13 Exit and portability

1.5 Terminology

The key words MUST, MUST NOT, SHOULD, SHOULD NOT and MAY are used as defined in RFC 2119. Requirements stated as MUST are conformance requirements on any CDP instance; SHOULD statements are strong defaults a Trust may vary with a recorded reason.

> **Status of this build**
> CDP v0.1 has been exercised against synthetic data only. It has not held live patient data and has not been deployed into a production Trust environment. Volumetric and latency figures in §11 are targets derived from synthetic load testing, not observed production performance.

02 Conformance and standards baseline

CDP defines almost nothing of its own. Its value comes from implementing standards the NHS already publishes, so that data held in it is structurally recognisable to anything else built on the same baseline.

STANDARD	ROLE IN CDP	CONFORMANCE POSITION
NHS Canonical Data Model	Normative source for canonical table and column structure.	Schema is generated from the published model; local extension is confined to a separate namespace.
HL7 FHIR R4	Base specification for the read API.	Resources, search semantics, paging and OperationOutcome errors conform to R4.
FHIR UK Core	Profiles constraining the R4 resources for UK use.	Served resources are profiled against UK Core; the capability statement declares each profile asserted.
NHS Data Dictionary	Source of value sets and code meanings.	Accepted-value tests are generated from dictionary value sets and versioned with the model release.
SNOMED CT / dm+d / ICD-10 / OPCS-4	Clinical terminologies referenced by coded columns.	Codes are stored with their code system and version; CDP does not re-map between terminologies.
ODS (Organisation Data Service)	Organisation and site identity.	Organisation codes are the basis of the row-level visibility model in §10.
DCB0129 / DCB0160	Clinical risk management for health IT.	Not yet applicable — CDP v0.1 is not deployed in support of direct care. A hazard log is maintained.
DSPT / NHS DSP Toolkit	Organisational assurance.	CDP supplies technical evidence; the toolkit submission remains the Trust's.
ISO/IEC 27001 control themes	Reference framework for the control set in §10.	Controls are mapped to themes; CDP is not itself certified.

2.1 Licensing baseline

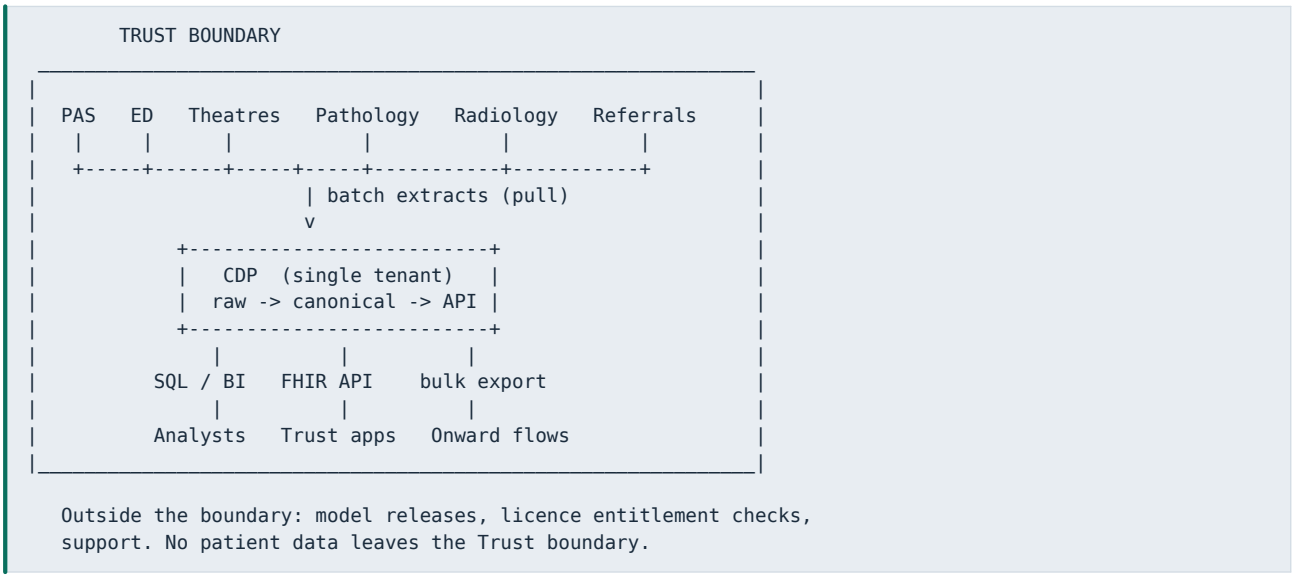
The published NHS model	Available under the Open Government Licence v3.0. CDP does not assert rights over it.
CDP itself	Business Source License 1.1. Non-production evaluation, development, testing and training use are free. A production instance holding live Trust data requires an instance licence. Each release converts to Apache-2.0 four years after publication.
Customer data	Not covered by, and never transferred under, the CDP licence. Data in a CDP instance remains the Trust's, in the Trust's tenancy.

> **Independence statement**
> CDP is developed independently by VUIT Data Labs. It is not affiliated with, endorsed by, or derived from NHS England or Palantir Technologies Inc. Third-party names are used only to identify standards and market context, and remain the trademarks of their owners.

03 System context and architecture

Six layers, each independently replaceable. The design constraint is that no layer may become the only place a piece of meaning exists — if a component is removed, its contribution must be recoverable from files in the repository.

3.1 System context



Context — CDP sits inside the Trust boundary, downstream of source systems.

3.2 The six layers

#	LAYER	RESPONSIBILITY	REPLACEABILITY TEST
1	Ingestion	Land source extracts unmodified with file, batch and timestamp lineage recorded.	Swap the extract mechanism; the landing contract in §6.2 is unchanged.
2	Transformation	Map source shapes onto the canonical model as reviewable SQL.	Replace dbt with any SQL orchestrator; the models remain readable SQL files.
3	Canonical repository	Hold the canonical copy: ~72 tables, ~1,811 columns, generated from the published CDM.	Restore from generated DDL plus data into any conformant PostgreSQL-compatible engine.
4	Quality gates	Run 392 generated tests and reconciliation on every load; gate publication.	Tests are SQL assertions; they run under any runner that can execute SQL and read exit codes.
5	Serving	FHIR UK Core read API, governed SQL access, bulk export.	Remove the API service and the repository is still fully usable over SQL.
6	Control	Authentication, RBAC, row-level security, pseudonymisation boundary, audit.	Policies are SQL objects in version control, not console configuration.

3.3 Data flow

```
source extract
     |
     v
[raw]          exact landed copy + lineage columns, append-only
     |  staging models: typing, trimming, deduplication
     v
[staging]      one model per source object, no business logic
     |  canonical models: conform to CDM structure and value sets
     v
[canonical]    the repository of record  <-- tests + reconciliation gate here
     |  marts: pathway, capacity, quality views
     v
[marts] ---> SQL / BI      [canonical] ---> FHIR API ---> clients
                                       \--> bulk export
```

Flow — every arrow is a versioned artefact, not a manual step.

3.4 Deployment topology

Tenancy	Single-tenant. One CDP instance serves one Trust (or one ICS acting as a single data controller). There is no shared multi-tenant database and no cross-Trust query surface.
Location	Deployed inside the Trust's own environment: on-premises, or in the Trust's own cloud subscription. VUIT Data Labs requires no standing access to the instance or its data.
Environments	Three by default — development (synthetic), pre-production (real structure, restricted access) and production. Promotion is by Git tag, not by manual copy.
Engine	PostgreSQL 15+ or a PostgreSQL wire-compatible analytical engine. Nothing in the repository depends on a proprietary SQL dialect extension.

3.5 Trust boundaries

BOUNDARY	CROSSES IT	CONTROL
Source system → raw	Batch extract files	Service account with read-only scope; checksum on landing
Raw → canonical	Transformation execution	Runs as a dedicated role with no access to serving roles
Identifiable → analytical	Pseudonymised keys only	Pseudonymisation applied in view definitions; salt held in secret store
Canonical → API client	FHIR resources	OAuth2 client credentials, scoped, audited
Canonical → analyst	SQL result sets	Role + row-level policy; every statement fingerprinted to the audit log
Instance → supplier	Licence entitlement heartbeat and version metadata only	No patient data; payload documented in §12.5

04 Canonical data model

The repository is not a bespoke warehouse schema with NHS-flavoured column names. It is generated from the published model, which is what makes two Trusts' repositories comparable and makes the exit argument in §13 credible.

4.1 Domain inventory

DOMAIN	TABLES	ILLUSTRATIVE CONTENT	DEFAULT VISIBILITY
Patient index	6	Patient, identifiers, address history, GP registration, death notification	Identifiable
Admitted patient care	12	Spell, episode, ward stay, transfer, discharge, delayed discharge	Pseudonymised
Outpatients	9	Appointment, clinic, attendance outcome, DNA, virtual contact	Pseudonymised
Emergency care	8	Attendance, arrival, triage, investigation, treatment, departure	Pseudonymised
Waiting lists	7	Pathway, clock start/stop, RTT status, planned procedure, cancellation	Pseudonymised
Theatres	6	Session, booking, procedure, in/out timestamps, cancellation reason	Operational
Diagnostics	9	Order, specimen, result, radiology exam, report status	Pseudonymised
Clinical coding	6	Diagnosis, procedure, coding sequence, comorbidity flags	Pseudonymised
Reference data	9	Organisation, site, specialty, treatment function, value sets, calendar	Reference
Total	~72	~1,811 generated columns	—

Indicative distribution across the v0.1 reference build. Exact table lists ship with the schema artefact.

4.2 Key strategy

- Every canonical table MUST carry a surrogate primary key named <entity>_sk, generated inside CDP and stable for the lifetime of the row.
- Every canonical table MUST retain the natural key it arrived with (source system code plus source identifier) so any row can be traced back to its origin.
- Foreign keys MUST reference surrogate keys, never natural keys, so a source system renumbering cannot corrupt referential integrity.

- Cross-domain joins MUST route through the patient index or the pathway entity; direct joins between clinical domains on natural keys are prohibited by convention and caught by review.
- Unresolved references MUST be recorded against an explicit unknown member rather than left null, so counts reconcile without silent row loss.

```
-- generated from the published model; edited only by regeneration
create table cdm.apc_spell (
  apc_spell_sk        bigint      primary key,
  patient_sk          bigint      not null references cdm.patient (patient_sk),
  organisation_sk     bigint      not null references cdm.organisation (organisation_sk),
  source_system_code  varchar(20) not null,
  source_spell_id     varchar(64) not null,
  admission_dttm      timestamptz not null,
  discharge_dttm      timestamptz,
  admission_method_cd varchar(10) not null,   -- value set bound
  discharge_dest_cd   varchar(10),            -- value set bound
  spell_los_days      integer,                -- derived, documented in model docs
  valid_from          timestamptz not null,
  valid_to            timestamptz,
  is_current          boolean     not null,
  batch_id            bigint      not null,
  constraint uq_apc_spell_natural
    unique (source_system_code, source_spell_id, valid_from)
);
```

Illustrative canonical shape — structure and conventions only.

4.3 Temporality

CONCERN	TREATMENT
Event time	Stored as timestamptz in UTC; local time is a presentation concern, never a storage one.
Record validity	valid_from / valid_to / is_current on every mutable entity (SCD type 2).
Load lineage	batch_id on every row, joining to a load registry with source file, checksum and run time.
Corrections	Applied as new versions, never in-place updates. History is not rewritten.
Deletions at source	Recorded as a logical closure (valid_to set) plus a deletion reason, not a physical delete.
Point-in-time query	Any as-at position is reconstructable from valid_from / valid_to without a separate snapshot store.

4.4 Value sets and coded columns

- Coded columns are bound to a named value set from the NHS Data Dictionary; the binding is recorded in the model metadata and generates an accepted-values test.
- Value sets are versioned with the model release. A code retired in a later release remains valid for historical rows.
- Terminology-coded columns store the code, the code system URI and the system version. CDP does not translate between terminologies.
- Unmapped source codes are retained verbatim in a paired _source_value column so no information is lost at the mapping boundary.

4.5 Naming conventions

OBJECT	CONVENTION	EXAMPLE
Schema	raw / stg / cdm / mart / audit / local	cdm
Canonical table	domain-prefixed singular snake_case	apc_spell
Surrogate key	<entity>_sk	apc_spell_sk
Coded column	<concept>_cd	admission_method_cd
Timestamp	<event>_dttm	admission_dttm
Date	<event>_dt	clock_start_dt
Flag	is_<condition>	is_current
Local extension	local. schema only — never inside cdm.	local.trust_ward_grouping

Local extension rule
Trust-specific tables and columns MUST live in the local schema. The cdm schema stays byte-comparable to the generated model so that a model upgrade is a regeneration, not a merge conflict. This single rule is what keeps upgrade cost bounded.

05 Ontology and semantic layer

The repository is not addressed as loose tables. An object layer — object types, typed properties and named link types — is generated from the same canonical model that generates the DDL, so applications, analysts and machine agents work against objects and relationships rather than hand-written joins.

5.1 What the ontology is

The canonical model in §4 is the physical store. The ontology is the addressable semantic layer above it: a registry of object types, each with typed properties bound to national value sets, and named link types derived from the model's declared relationships. It is generated, not curated by hand, which is what keeps it in step with the schema it describes.

ELEMENT	COUNT	DERIVED FROM
Object types	80	Canonical model object definitions
Typed properties	1,739	Canonical attribute definitions and value-set bindings
Link types	159	Declared referential relationships between objects
Visibility groups	Per object	Object-level visibility declarations, reused by row-level security
Actions	1 (v0.1)	Explicitly specified state transitions

Ontology inventory in the v0.1 reference build.

Object type	A named clinical or operational concept — patient, encounter, referral, pathway — with a stable identity and a declared property set.
Property	A typed attribute of an object type, bound where applicable to an NHS Data Dictionary value set.
Link type	A named, directional relationship between two object types, generated from the model's referential rules rather than inferred at query time.
Action	A permissioned, audited state transition on an object. Distinct from arbitrary write access; see §5.5.

5.2 Typed access — data as code

A typed access layer is generated from the object registry. Callers request an object type and traverse its link types; they do not compose joins against physical table names. Because the registry is generated from the same source as the DDL, a model change that removes or renames a property becomes an error at the access layer rather than a silently wrong query.

```
# resolve an object by identity, then traverse a declared link type
patient   = cdm.get("Patient", key)
encounters = cdm.find("Encounter", patient=patient.key, class="IMP")

# unknown object type or unknown property is rejected before any query is built
cdm.find("Encounter", wardd="X")   -> UnknownFieldError
```

Illustrative object access, not a SQL query against physical tables.

- Every object-type name and property name is validated against the generated registry before a query is constructed.
- The database boundary independently validates every table and column identifier against a strict allowlist, so correctness does not rely on the access layer having been called properly.
- All values are passed as bound parameters. Identifiers, which SQL cannot parameterise, are allowlisted instead.
- Row-level security is enforced by the database from each object's visibility groups, not by the semantic layer. A caller that fails to establish identity sees nothing rather than everything.

5.3 Machine-navigable surface

The object registry is also exposed as a tool surface: an agent can enumerate object types, inspect their properties and link types, and issue typed reads without being given free-form SQL. A natural-language query endpoint is built on that surface, so a model chooses among declared object types and properties rather than generating raw SQL.

- The agent surface is read-only in v0.1; actions are not exposed to it.
- Registry validation and identifier allowlisting apply identically to agent-initiated calls.
- Every agent-initiated read is subject to the same row-level policy and the same audit record as a human query.

5.4 Relationship to the physical model

Single source	Object types, DDL, row-level policies, quality tests and pathway views are all generated from one canonical model definition. A model update touches one file by hand.
No shadow model	The ontology adds no facts. Every property resolves to a canonical column or a declared derivation; there is no separate store to reconcile.
Cross-checking	Generator outputs are cross-checked against each other, which is how structural defects — a missing primary key on an object, for instance — surface before release rather than in production.
Exit	Because the ontology is generated, an exiting Trust holds both the physical data and the definitions that produced the semantic layer. Nothing about it is locked in a vendor console.

5.5 Actions and write-back

CDP is read-mostly by design: source systems remain authoritative and the repository is a consumer and re-projector of their data, not a second place clinical facts are authored. v0.1 specifies exactly one action — recording that an admission has been discharged — to demonstrate that permissioned, audited write-back through the object layer is architecturally present rather than theoretical.

PROPERTY	POSITION
Scope	One encounter state transition; no clinical content is authored
Authorisation	Role-scoped, with database-enforced write policy on the target object
Audit	Append-only record of principal, object identity, prior and new state, and time
Reconciliation	The transition is reflected back to the source system's own record of truth; CDP does not diverge from it
Everything else	Read-only. Extending the action set is a specification change, not a configuration change.

What the ontology is not

It is not a clinical authoring system, not an EPR replacement, and not a general write interface. It carries one specified action in v0.1. Application-building surfaces on top of the object layer are roadmap, and are stated as such in §14.

06 Ingestion specification

Ingestion is deliberately dull. Nothing is interpreted, corrected or enriched before it is captured, so the raw zone is always a defensible record of what the source system actually said.

6.1 Source system classes

CLASS	TYPICAL SYSTEMS	EXTRACT MODE	DEFAULT CADENCE
Administrative	PAS, referral management	Full or watermarked delta	Nightly
Emergency	ED system	Watermarked delta	Nightly, hourly optional
Theatres	Theatre management	Full	Nightly
Diagnostics	LIMS, RIS	Watermarked delta	Nightly
Coding	Clinical coding system	Full	Nightly
Reference	ODS, dictionary value sets	Full replace	On release

6.2 Landing contract

Any mechanism may deliver extracts — scheduled SQL extract, file drop, SFTP, message consumer — provided it satisfies this contract. CDP does not mandate an ETL tool.

- Each delivery MUST be a discrete, immutable unit with a manifest declaring source system code, source object, extract window, row count and checksum.
- Column names and values MUST be landed as received. No type coercion, trimming, case folding or null substitution occurs before landing.
- Every landed row MUST receive batch_id, landed_at, source_file and row_ordinal.
- The raw zone MUST be append-only. Re-delivery creates a new batch; it never overwrites a prior one.
- A delivery whose declared row count or checksum does not match its content MUST be rejected whole and MUST NOT be partially loaded.

```
{
  "source_system_code": "PAS",
  "source_object": "spell",
  "extract_mode": "delta",
  "watermark_column": "last_modified_dttm",
  "window_from": "2026-09-14T00:00:00Z",
  "window_to":   "2026-09-15T00:00:00Z",
  "row_count": 4182,
  "sha256": "9f2c...",
  "extracted_at": "2026-09-15T01:12:04Z",
  "extract_version": "pas-spell/3"
}
```

Manifest shape accompanying each delivery.

6.3 Idempotency and replay

- Loading the same delivery twice MUST produce the same canonical state. Deduplication is by natural key plus source modification timestamp.
- Any batch, or any contiguous set of batches, MUST be replayable from the raw zone without re-extraction from the source system.
- A full rebuild of the canonical layer from raw MUST be possible as a single command, and is exercised as part of release testing.

6.4 Late and out-of-order data

CASE	BEHAVIOUR
Row arrives after its period was published	New version created; the affected period is flagged as restated in the load registry.
Row arrives with an older modification timestamp than the current version	Retained in raw, not promoted; recorded as a suppressed-regression event.
Referenced parent has not yet arrived	Child row is loaded against the unknown member and re-resolved on the next run.
Source system replays historic data wholesale	Treated as a normal batch; SCD versioning absorbs it without duplicate counting.

6.5 Error quarantine

- Rows failing structural validation are written to a quarantine table with the failing rule, the batch and the full original row.
- Quarantine is visible to the Trust, counted in the load report, and never silently discarded.
- A quarantine rate above a configured threshold for a source object raises a blocking condition on publication (§8.4).

07 Transformation specification

All transformation logic is version-controlled SQL that a Trust analyst can read without a licence, a viewer or a supplier. This is the substance behind the phrase data as code.

7.1 Layering rules

LAYER	PERMITTED OPERATIONS	PROHIBITED
raw	Landing only	Any transformation whatsoever
stg (staging)	Typing, renaming, trimming, deduplication, one model per source object	Joins across sources, business rules, aggregation, filtering out records
cdm (canonical)	Conforming to canonical structure, key resolution, value-set mapping, SCD versioning	Trust-specific business definitions, reporting aggregations, presentation formatting
mart	Business definitions, aggregation, pathway and capacity views	Redefining canonical entities, writing back into cdm
local	Trust extensions and local groupings	Modifying generated cdm objects

7.2 Project structure

```
cdp/
  models/
    staging/      stg_pas__spell.sql            one per source object
    canonical/    cdm_apc_spell.sql             one per canonical table
    marts/        mart_rtt_pathway.sql
  generated/
    ddl/          cdm_schema.sql                regenerated from the model
    tests/        cdm_tests.yml                 generated assertions
    valuesets/    nhs_dd_valuesets_2026-08.yml  versioned with the release
  policies/       rls_visibility.sql  roles.sql  audit.sql
  recon/          recon_spec.yml  recon_queries/
  infra/          docker-compose.yml  terraform/
  docs/           model docs, lineage, upgrade notes
```

Repository layout held in the Trust's own Git.

7.3 Materialisation strategy

LAYER	MATERIALISATION	RATIONALE
stg	View, or ephemeral where volume demands	Cheap to change; no duplicated storage
cdm (high volume)	Incremental table on natural key + modification timestamp	Bounded nightly runtime as history grows
cdm (low volume / reference)	Full table refresh	Simplicity beats micro-optimisation
mart	Table, refreshed after the canonical gate passes	Consumers never read a partially rebuilt mart
Serving views	View over cdm with pseudonymisation and policy applied	One definition of what a role may see

7.4 Change control

01 Change is proposed as a branch in the Trust's repository, with the reason recorded in the commit.

02 Continuous integration runs the full test suite plus reconciliation against a synthetic or masked dataset.

03 A named reviewer approves. Canonical-layer changes require a second reviewer.

04 Merge to main tags a release; deployment to pre-production is automatic.

05 Promotion to production is an explicit, recorded action against a tagged release — never a direct edit.

06 Every deployed state is reproducible from a tag, and every canonical row's logic is traceable to a commit.

7.5 Lineage

- Model-to-model lineage is derived from the SQL itself and published as a browsable graph with the documentation build.
- Row-level lineage is available through batch_id and the retained natural key: any canonical row resolves to a source file, a delivery window and a checksum.
- Column-level provenance is recorded in the model metadata for every canonical column, including whether it is landed, mapped or derived.

08 Data quality and reconciliation

Quality is not a dashboard published after the fact. It is a gate: 392 generated assertions and a line-by-line reconciliation run on every load, and a blocking failure stops publication.

8.1 Test taxonomy

CLASS	ASSERTS	TYPICAL SEVERITY	INDICATIVE COUNT
Uniqueness	Surrogate and natural keys are unique in scope	Blocking	~90
Not null	Mandatory canonical columns are populated	Blocking	~110
Referential	Every foreign key resolves to a parent row	Blocking	~70
Accepted values	Coded columns hold only bound value-set members	Blocking on core, warning on peripheral	~60
Range and plausibility	Dates, ages and durations fall inside defensible bounds	Warning	~30
Temporal integrity	Discharge follows admission; no overlapping current versions	Blocking	~20
Freshness	Each source object has landed within its expected window	Blocking	~10
Volumetric	Row counts sit within tolerance of trailing baseline	Warning, blocking beyond hard tolerance	~10
Business rule	Domain-specific invariants, e.g. RTT clock consistency	Configurable	~20

8.2 Severity model

Blocking	Publication halts. Marts and serving views continue to expose the last good state. An alert is raised with the failing assertion and sample failing keys.
Warning	Publication proceeds. The failure is recorded, counted, attributed to a named owner and trended over time.
Informational	Recorded for monitoring only; no owner, no alert.

Severity is declared in configuration, in version control. A Trust may downgrade a test, but the downgrade is a reviewed commit with a reason — never an undocumented console toggle.

8.3 Reconciliation

Reconciliation answers the only question that matters at cutover: does the canonical repository agree with the source system, and with the platform being replaced? It is a declared specification, not an ad-hoc query.

```
- name: apc_spells_by_month_specialty
  grain: [discharge_month, treatment_function_cd]
  measures: [spell_count, bed_days]
  left:  { system: PAS,  query: recon_queries/pas_spells.sql }
  right: { system: CDP,  query: recon_queries/cdp_spells.sql }
  tolerance: { absolute: 0, relative: 0.000 }
  on_breach: block
  evidence: recon/output/apc_spells_by_month_specialty.csv
```

Reconciliation specification shape.

- Reconciliation runs at a declared grain — typically period by specialty, or period by site — not on totals alone, so offsetting errors cannot cancel out.
- Default tolerance for counted activity is zero. Any variance is itemised to the individual key rather than reported as a percentage.
- Where a variance is expected and explicable (a known source defect, a definitional difference), it is recorded as an accepted variance with a reason and an owner, and continues to be reported.
- Where the Trust is migrating from an existing platform, the same specification is run three-way: source system, incumbent platform, CDP.

8.4 Publication gate

```
load batch -> structural validation -> canonical build
                                          |
                                tests + reconciliation
                                  /            \
                            all pass          any blocking failure
                               |                     |
                    publish marts,            hold publication,
                    advance the               retain last good state,
                    published watermark       alert with failing keys
```

8.5 Evidence outputs

- A per-run quality report: assertions executed, passed, warned, failed, with failing sample keys.
- A reconciliation pack: per-grain variance tables, exportable as CSV for audit.
- A load registry entry: batch, source file, checksum, row counts landed, quarantined and published.
- A trend series suitable for a Board data-quality report, generated from the same runs — not compiled separately.

09 Interfaces

Two first-class interfaces — FHIR UK Core for interoperability and governed SQL for analysts — plus bulk export. Anything conformant can read the repository without bespoke integration work.

9.1 FHIR read API

PROPERTY	POSITION
Base specification	HL7 FHIR R4 (4.0.1)
Profiles	FHIR UK Core, asserted per resource in the capability statement
Operations	read, vread, search-type, and a capability statement at /metadata
Write operations	Read-mostly by design. One narrow Encounter state transition is supported, as specified in §5.5; no other resource accepts writes in v0.1.
Formats	application/fhir+json; XML not supported in v0.1
Paging	Bundle.link next / self, cursor-based, default 50 and maximum 500 entries per page
Errors	OperationOutcome with a machine-readable issue code on every 4xx and 5xx
Versioning	Resource versionId and lastUpdated derived from canonical SCD metadata
Authorisation	OAuth2 client credentials, scoped per resource type

RESOURCE	BACKED BY	KEY SEARCH PARAMETERS
Patient	patient, patient_identifier	identifier, birthdate, _lastUpdated
Encounter	apc_spell, apc_episode, op_appointment, ec_attendance	patient, class, date, status, service-provider
Condition	coded_diagnosis	patient, code, encounter, recorded-date
Procedure	coded_procedure, theatre_procedure	patient, code, encounter, date
Observation	diagnostic_result	patient, code, date, category
ServiceRequest	diagnostic_order, referral	patient, category, authored, status
Organization	organisation, site	identifier, name
Practitioner	practitioner (pseudonymised by default)	identifier

Resource coverage in v0.1 and the canonical entities behind each.

```
GET /fhir/Encounter?patient=Patient/8f31c2&class=IMP&date=ge2026-04-01
Accept: application/fhir+json
Authorization: Bearer <token>

200 OK  -> Bundle (searchset) with link[next]

403 Forbidden -> OperationOutcome
{ "resourceType": "OperationOutcome",
  "issue": [{ "severity": "error", "code": "forbidden",
              "diagnostics": "Scope does not permit Encounter.read for this organisation" }] }
```

Illustrative request and error shape.

9.2 Governed SQL access

- Analysts connect with ordinary PostgreSQL drivers, from Power BI, Tableau, R, Python or psql. No proprietary client is required.
- Access is granted to serving views, not base tables, so pseudonymisation and row-level policy are applied in one place.
- Read roles are defined in SQL under policies/ and version-controlled; grants are code, not console state.
- Every statement is recorded to the audit log with principal, role, visibility group, object, statement fingerprint and rows returned.

9.3 openEHR projection

For Trusts holding an openEHR position, already-validated FHIR resources are projected onto published openEHR archetypes rather than mapped a second time from canonical columns. Archetype identifiers are taken as published from the openEHR Clinical Knowledge Manager; none are invented for CDP.

CONCEPT	ARCHETYPE CLASS	SOURCE
Encounter	COMPOSITION	openEHR CKM, used as published
Admission	ADMIN_ENTRY	openEHR CKM, used as published
Diagnosis	EVALUATION	openEHR CKM, used as published
Procedure	ACTION	openEHR CKM, used as published

Archetype coverage in v0.1.

> **Projection scope**
> The projection is a read-side representation covering four concepts. It is not a full openEHR clinical data repository and makes no template-conformance claim beyond the archetypes listed.

9.4 Bulk export

PROPERTY	POSITION
Purpose	Onward flows, archival, and exit
Formats	CSV and Parquet, plus the generated DDL required to reload them
Scope	Any canonical table or view the requesting role may read
Integrity	Manifest per export with row counts and checksums, matching the ingestion manifest shape
Control	Export capability is a distinct, separately grantable privilege and is always audited

9.5 Interface stability

Canonical schema	Changes only with a model release. Additive changes are minor; structural changes are major and documented in upgrade notes.
FHIR API	Semantic versioning on the service. Breaking changes require a major version and a parallel-running period.
Serving views	Treated as the published contract to analysts. Column removal requires notice and a deprecation period.

10 Security, information governance and access control

The control set is stated with its gaps. A Trust IG team should be able to read this section and know exactly what it is being asked to accept.

10.1 Roles and responsibilities

PARTY	ROLE	NOTES
The Trust	Data controller	Determines purposes; owns the DPIA and the lawful basis
VUIT Data Labs	Software supplier	Not a processor in a self-hosted deployment; holds no access to Trust data by default
VUIT Data Labs (support engagement only)	Processor, if and only if the Trust grants time-bounded access	Governed by a separate data processing agreement

10.2 Authentication

- Human access integrates with the Trust identity provider over OIDC or SAML. CDP does not maintain its own password store for interactive users.
- Multi-factor authentication is enforced by the Trust identity provider; CDP defers to it and records the authentication context in the audit log.
- Machine access uses OAuth2 client credentials with short-lived tokens and per-client scopes.
- Database service accounts are non-interactive, least-privileged and rotated on a defined schedule.

10.3 Authorisation model

LAYER	MECHANISM	DEFINED IN
Role	Named database roles with explicit grants (analyst, clinical_audit, ig_officer, engineer, api_service)	policies/roles.sql
Object	Grants on serving views only; no direct grants on base canonical tables	policies/roles.sql
Row	Row-level security policies filtering by organisation and visibility group	policies/rls_visibility.sql
Column	Identifiable columns exposed only through views a role is granted	serving view definitions
Audit	Append-only, insert-only for all roles; readable by ig_officer	policies/audit.sql

```
alter table cdm.apc_spell enable row level security;

create policy apc_spell_org_visibility on cdm.apc_spell
  for select to analyst
  using (
    organisation_sk in (
      select organisation_sk
      from   cdm.role_organisation_visibility
      where  role_name = current_setting('cdp.role_name', true)
    )
  );
```

Illustrative row-level policy — policy as code, reviewable in a pull request.

10.4 Pseudonymisation boundary

- Identifiable attributes are confined to the patient index domain. Clinical and operational domains carry pseudonymised keys.
- Pseudonymisation is deterministic within an instance so longitudinal analysis works, and instance-specific so identifiers are not comparable across Trusts.
- The salt is held in the platform secret store, never in the repository, and is not recoverable from an export.
- Re-identification requires an explicit, separately granted role and every use is audited as a distinct event class.

10.5 Audit log

COLUMN	CONTENT
occurred_at	UTC timestamp of the access
principal	Authenticated identity or service client
role_name	Effective role at execution
visibility_group	Row-level scope applied
object_name	View, table or FHIR resource accessed
statement_fingerprint	Stable hash of the normalised statement, so patterns are analysable without storing parameters
rows_returned	Volume returned, for exfiltration monitoring

- The log is append-only: no role, including engineering roles, holds update or delete privileges on it.
- Default retention is defined by the Trust, with a recommended minimum of 12 months online and archival thereafter.
- Access to the audit log is itself audited.

10.6 Cryptography and secrets

CONTROL	POSITION
In transit	TLS 1.2 minimum, TLS 1.3 preferred, on every interface including database connections
At rest	Volume or tablespace encryption provided by the Trust platform; CDP does not implement its own at-rest layer
Backups	Encrypted with keys held by the Trust, tested by restore, never held by the supplier
Secrets	Held in the platform secret store or Trust vault; never in the repository, never in environment files committed to Git
Key custody	Entirely the Trust's. VUIT Data Labs holds no key material for a self-hosted instance

10.7 IG position and gaps

Available now	Draft DPIA, data flow map, retention position, control mapping, audit specification and the reconciliation evidence pack.
Trust responsibility	Lawful basis, DPIA sign-off, national opt-out handling, records-management schedule and DSPT submission.
Outstanding	No independent penetration test yet published. No production deployment history. No DCB0129 clinical safety case, as v0.1 is not deployed in support of direct care.

Stated plainly

CDP v0.1 has never held live patient data. A Trust adopting it is adopting a reference build with a documented control set and no production track record. That is the honest position, and it is the reason the first engagement is structured as a design partnership rather than a purchase.

11 Non-functional requirements

Targets, with the assumptions they rest on. These are derived from synthetic load testing on the reference build, not from observed production operation.

11.1 Volumetric assumptions

DIMENSION	REFERENCE (MEDIUM ACUTE TRUST)	TESTED TO
Patient index	1.2 million patients	3 million
Admitted spells per year	120,000	400,000
Outpatient appointments per year	900,000	2.5 million
ED attendances per year	200,000	600,000
Diagnostic results per year	6 million	15 million
History retained	10 years	10 years
Canonical row count	~250 million	~700 million
Storage (compressed, incl. history)	1.5 – 3 TB	8 TB

11.2 Performance targets

MEASURE	TARGET	CONDITION
Nightly load window	≤ 3 hours end to end, including tests and reconciliation	Reference volumes, delta extracts
Full rebuild from raw	≤ 12 hours	10 years of history
Test suite execution	≤ 25 minutes for 392 assertions	Reference volumes
Reconciliation pack	≤ 20 minutes	Declared grains, 24 months
Interactive SQL on marts (p95)	< 3 seconds	Indexed mart access, typical BI query
FHIR read by id (p95)	< 300 ms	Warm cache, single resource
FHIR search, 50 entries (p95)	< 1.5 seconds	Indexed search parameters
Bulk export throughput	≥ 20 million rows per hour, Parquet	Single export worker

11.3 Availability, backup and recovery

PROPERTY	POSITION
Service class	Analytical, not direct care. Availability targets are set accordingly.
Target availability (serving)	99.5% in business hours, Trust-defined maintenance windows
Recovery point objective	24 hours from backup; effectively the last completed batch, since raw is replayable
Recovery time objective	≤ 8 hours to restore serving from backup; ≤ 24 hours to rebuild fully from raw
Backup regime	Nightly full plus continuous WAL archiving, retention set by the Trust
Restore testing	Quarterly restore rehearsal, evidenced
Disaster recovery	Rebuild from Git plus raw zone. No supplier dependency in the recovery path.

11.4 Scaling

- Vertical first: the reference build runs comfortably on 16 vCPU / 128 GB with fast local NVMe or equivalent provisioned IOPS.
- Read scaling by read replicas for serving views and the FHIR service; the transformation role always runs against the primary.
- Partitioning on event date for the highest-volume canonical tables, so nightly work touches recent partitions only.
- The transformation layer is engine-portable SQL, so a Trust already running a large analytical engine can target it instead of a dedicated PostgreSQL instance.

11.5 Accessibility and localisation

- Operational interfaces target WCAG 2.2 AA. The primary consumption surfaces are the Trust's own BI tools, which retain their own accessibility posture.
- All storage is UTC; presentation in local time is a consumer concern.
- English (United Kingdom) only in v0.1.

12 Operations

A Trust platform team should be able to run this instance with the same tooling it already uses for anything else. Nothing here requires a supplier console.

12.1 Deployment

TARGET	ARTEFACT	USE
Local / evaluation	docker-compose.yml	Full stack on one machine with synthetic data
Trust cloud subscription	Terraform module	Managed PostgreSQL, object storage, secret store, container runtime
On-premises	Container images plus documented PostgreSQL prerequisites	Existing Trust virtualisation and storage

- All infrastructure is declared. There is no documented click-path deployment, because a click-path is not reproducible.
- Container images are versioned and pinned to a release tag; no image floats on latest.
- The deployment declares its own database roles and policies from policies/, so environments cannot drift apart in their access model.

12.2 Environment promotion

```
dev (synthetic)  --tag-->  pre-prod (restricted)  --approved-->  prod
      |                          |                                |
  full test suite       full suite + recon        full suite + recon + gate
      |                          |                                |
  fast iteration        representative volumes         published watermark
```

12.3 Observability

SIGNAL	EMITTED	TYPICAL ALERT
Batch outcome	Per batch: landed, quarantined, published row counts	Any blocking failure; quarantine above threshold
Load duration	Per model and per run	Run exceeds the load window
Test results	Per assertion, per run	New blocking failure; warning trend deteriorating
Reconciliation variance	Per grain, per run	Any variance outside declared tolerance
Freshness	Per source object	Expected delivery not received
Audit volume	Rows returned per principal	Unusual volume returned by a single principal
Platform	CPU, memory, storage headroom, replication lag	Standard Trust infrastructure thresholds

Metrics are exposed in Prometheus format and logs as structured JSON, so they land in whatever monitoring stack the Trust already runs. CDP does not require its own monitoring product.

12.4 Model currency and upgrades

01 A new published model version is tracked and a regenerated schema plus upgrade notes are issued as a CDP release.

02 The release states every structural change, every value-set change and every test change, with a migration script.

03 The Trust applies it in development, runs the suite and reconciliation, and reviews the diff.

04 Because Trust-specific objects live in local., the upgrade is a regeneration rather than a merge of hand-edited schema.

05 Promotion follows §12.2. The previous release remains restorable from its tag.

12.5 Supplier interaction and support

INTERACTION	DATA INVOLVED	FREQUENCY
Licence entitlement check	Instance identifier, release version, environment class. No patient data.	Periodic heartbeat, or offline token where egress is not permitted
Release delivery	Signed artefacts pulled by the Trust	On release
Support	Diagnostics the Trust chooses to share; access only if explicitly granted and time-bounded	On request

- An air-gapped deployment is supported: entitlement is satisfied by an offline token and releases are delivered as signed bundles.
- Support response commitments, hours of cover and named contacts are set in the instance licence, not in this document.

12.6 Incident handling

- Data-quality incidents follow the publication gate: the last good state stays served while the failure is investigated.
- Security incidents follow the Trust's own incident process. CDP supplies the audit trail; the Trust owns notification and reporting duties.
- Software defects are reported to VUIT Data Labs with a reproducible case; severity and response are defined in the licence.

13 Exit and portability

The test of this design is what a Trust holds on the day it stops paying anyone. The answer must be: everything needed to keep running, without permission.

13.1 What the Trust holds

ARTEFACT	FORM	DEPENDS ON THE SUPPLIER?
Canonical data	Rows in the Trust's own database	No
Schema definition	Generated SQL DDL in the Trust's Git	No
Transformation logic	Readable, version-controlled SQL	No
Quality assertions	Generated SQL tests in the repository	No
Reconciliation specification and evidence	YAML plus CSV output	No
Access policies and roles	SQL under policies/	No
Infrastructure definition	Docker Compose and Terraform	No
Change history	Git history in the Trust's repository	No
Future model releases and support	Supplier release stream	Yes — this is what the instance licence buys

13.2 Reproducing the repository without the supplier

01 Provision a PostgreSQL-compatible engine.

02 Apply generated/ddl/cdm_schema.sql to create the canonical model.

03 Run the staging and canonical models against the raw zone with any SQL orchestrator.

04 Execute generated/tests and the reconciliation specification to prove the result.

05 Apply policies/ to reinstate roles, row-level security and audit.

No step in that sequence requires a licence key, a supplier endpoint or a proprietary runtime. The dependency the licence creates is on future model currency and support — never on the ability to read your own data.

13.3 Exit from an incumbent platform into CDP

STAGE	ACTIVITY	GATE TO PASS
1. Establish	Stand up the instance; land one source system	Structural tests pass on landed data
2. Conform	Build the canonical layer for the first domains	Blocking assertions clear; lineage documented
3. Reconcile	Three-way reconciliation: source, incumbent, CDP	Zero unexplained variance at declared grain
4. Parallel run	Both platforms serve the same period	Agreement sustained across a full reporting cycle
5. Rebase	Repoint reports and consumers to CDP	Consumer sign-off per report
6. Decommission	Retire the incumbent dependency	Evidence pack retained for audit

13.4 Licence conversion

Under the Business Source License 1.1, each CDP release carries a Change Date four years after its publication, on which that release becomes available under Apache-2.0. This is contractual, automatic, and independent of the commercial relationship or the continued existence of VUIT Data Labs.

> **Why this matters more than any feature**
> The reason to build a Trust-owned canonical repository is not that it does something a commercial platform cannot. It is that the canonical copy of your data, and the logic that produced it, stop being contingent on a contract you do not control.

14 Limitations, assumptions and roadmap

The section a supplier usually omits. Read it before the architecture diagrams if you are deciding whether to spend time on this.

14.1 Limitations of v0.1

- Exercised against synthetic data only. No live patient data has been held.
- No production deployment, and therefore no observed production performance or operational history.
- Read-mostly: one specified encounter state transition (§5.5). No general write interface and no clinical authoring.
- The ontology carries no application-building surface in v0.1: object types, links and typed access only.
- FHIR coverage is limited to the resources listed in §9 and to JSON only.
- No independent penetration test has been published; no clinical safety case exists because CDP is not deployed in support of direct care.
- Domain coverage is acute-centred. Community, mental health and primary care domains are not implemented in v0.1.
- The reference build has not been exercised against a real export from an existing federated platform instance, only against synthetic data shaped to that structure.

14.2 Assumptions

- The Trust can obtain read-only batch extracts from its source systems.
- The Trust holds or can provision a PostgreSQL-compatible engine and a container runtime.
- The Trust has, or will establish, a Git repository and a review culture for SQL change.
- SQL capability exists in the Trust's analytics or data engineering team. CDP does not remove the need for it — it makes it sufficient.
- The published NHS Canonical Data Model continues to be published and maintained.

14.3 Roadmap direction

THEME	DIRECTION	TRIGGER
Design-partner deployment	First instance holding real Trust data, with reconciliation evidence published	A design-partner Trust
Domain breadth	Community, mental health and primary care domains	Partner requirement
Assurance	Independent penetration test and published control evidence	Ahead of first production go-live
FHIR breadth	Additional resources, XML support, subscription notifications	Consumer demand
Migration tooling	Validated loaders for exports from an existing federated platform instance	Access to a real export
Model currency automation	Automated diff and migration generation on model release	Ongoing

What would falsify the approach

If a Trust cannot obtain reliable extracts from its source systems, or has no capacity to review SQL, CDP is the wrong answer for it. A repository defined as code needs someone willing to read the code.

15 Glossary and normative references

Terms used with a specific meaning in this document, and the sources it depends on.

15.1 Glossary

Canonical repository	The cdm schema: the Trust's copy of record, generated from the published model.
CDM	NHS Canonical Data Model — the published structural standard CDP generates from.
Data as code	Every artefact defining the repository — schema, logic, tests, policies, infrastructure — is a reviewable file under version control.
Blocking failure	A test failure that halts publication, leaving the last good state served.
Publication gate	The point at which tests and reconciliation must pass before marts and serving views advance.
Published watermark	The batch up to which consumers are permitted to see data.
Reconciliation	Declared, grain-level comparison between CDP and another system, with itemised variance.
Serving view	The governed view a role reads, with pseudonymisation and row-level policy applied.
Visibility group	The organisational scope a role may see rows for, enforced by row-level security.
SCD type 2	Slowly changing dimension pattern: change creates a new version rather than overwriting.
Quarantine	Structurally invalid rows retained, counted and visible rather than discarded.
Change Date	The date under BSL 1.1 on which a given release becomes available under Apache-2.0.
Instance licence	The commercial licence required for a production instance holding live Trust data.

15.2 Normative references

- NHS Canonical Data Model, NHS England (Open Government Licence v3.0).
- HL7 FHIR R4 (4.0.1), Health Level Seven International.
- FHIR UK Core, NHS England.
- NHS Data Model and Dictionary, NHS England.
- SNOMED CT UK Edition; dm+d; ICD-10; OPCS-4.
- Organisation Data Service reference data, NHS England.
- RFC 2119, Key words for use in RFCs to Indicate Requirement Levels.
- Business Source License 1.1, MariaDB Corporation Ab.
- Apache License, Version 2.0, The Apache Software Foundation.

15.3 Document control

FIELD	VALUE
Reference	CDP-TS-0001
Version	v0.1
Issued	September 2026
Classification	Public
Publisher	VUIT Data Labs
Licence of the described work	Business Source License 1.1
Review cycle	On each model release, or at least annually

Deeper artefacts

The full generated DDL, the source-to-canonical mapping set, the generated test suite, the FHIR service implementation, the IG starter pack and the exit runbook are released to named evaluators on approval of an access request.