bioXplore: Database Design

Interactive exploration of biological knowledge networks

Source Open Targets Platform 26.06 (CC0 1.0)

Genes 78,691

Ontology terms 47,080 disease index · 48,321 GO

Disease–gene associations 7,554,123

Engine DuckDB 1.5.5 · single file, 1,115 MB

Tables 28

Built 2026-08-11

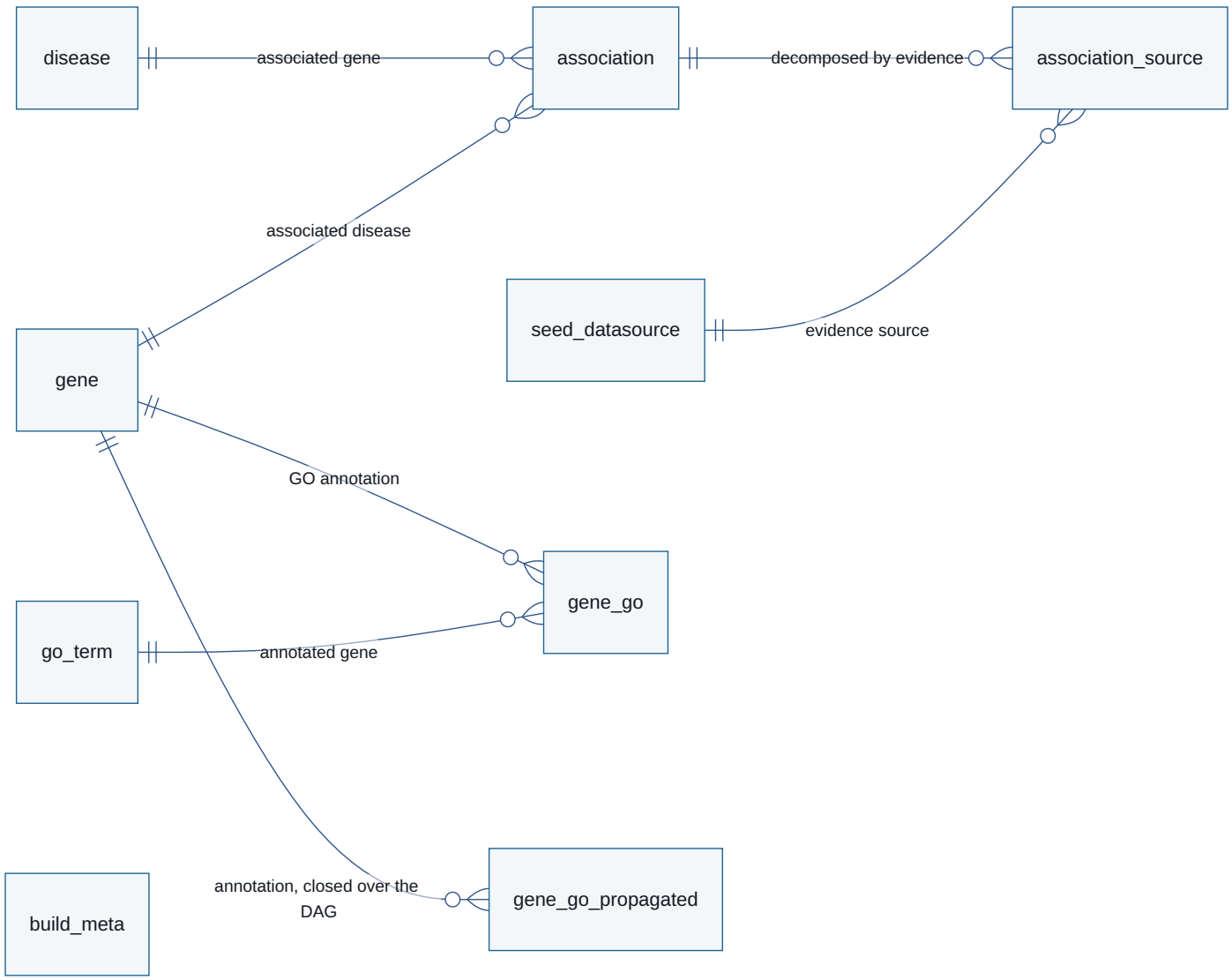
Document generated 2026-08-12

Research & development — not peer-reviewed. This schema and the knowledge graph built on it were developed with AI assistance and have not been validated by the scientific community. No claims are made as to scientific validity. Source data is credited to the Open Targets Platform and the resources it integrates; interpretation is ours.

Entity–Relationship Diagram

`||—○{` one to zero-or-many `||—○||` one to zero-or-one `○—○{` optional, self-referencing

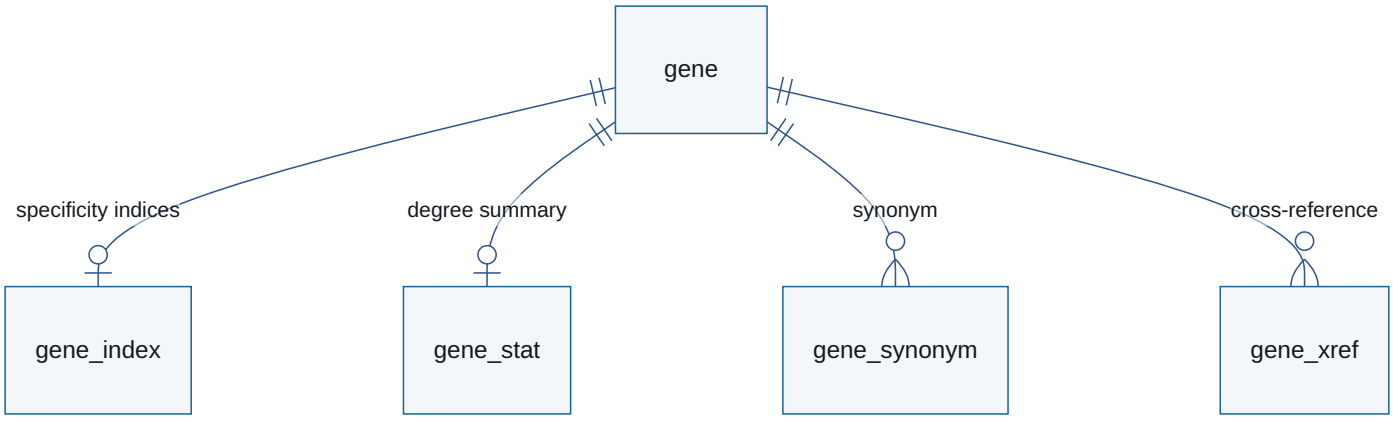
Core graph



The three entity hubs and the edges between them. `association_source` decomposes every disease-gene edge by contributing evidence source; `gene_go_propagated` is `gene_go` closed over the GO DAG. Generated from the live database; the relationship set is checked against the schema at build time. Attributes are documented separately, in the Relationships and Columns sections, rather than crowded into the boxes.

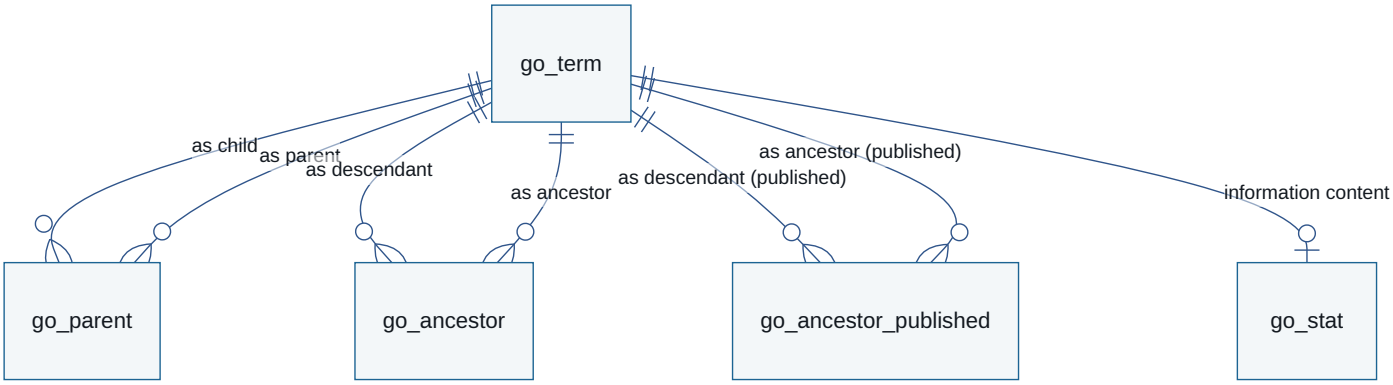
Entity–Relationship Diagram (continued)

Gene: naming and summary



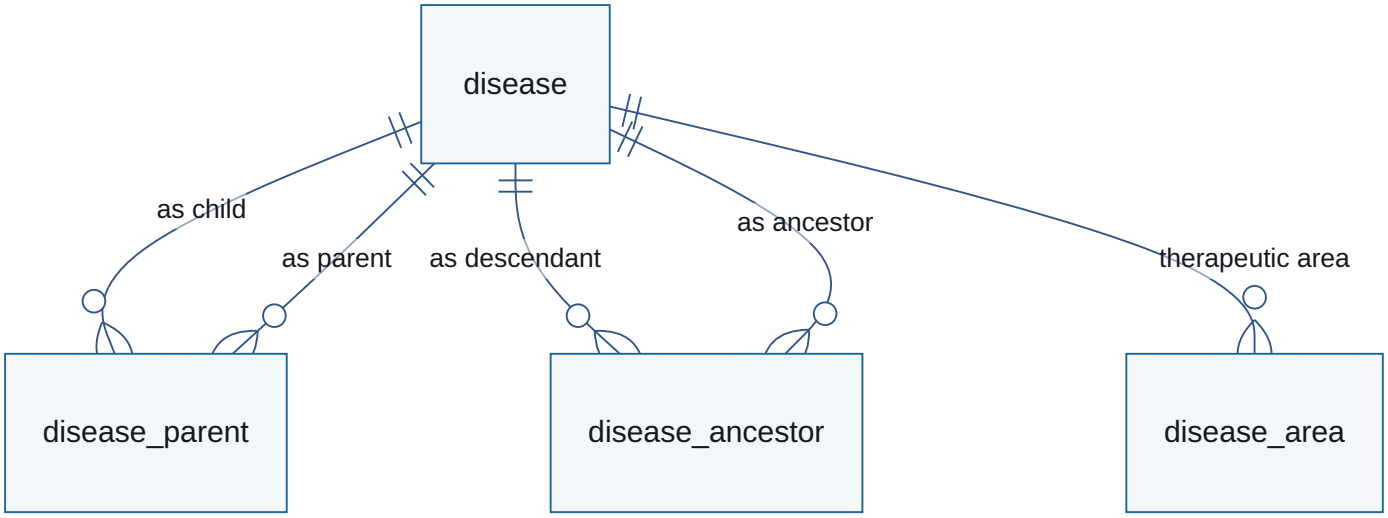
Search surfaces and precomputed degree, hanging off the gene hub.

Gene Ontology: hierarchy, closure, specificity



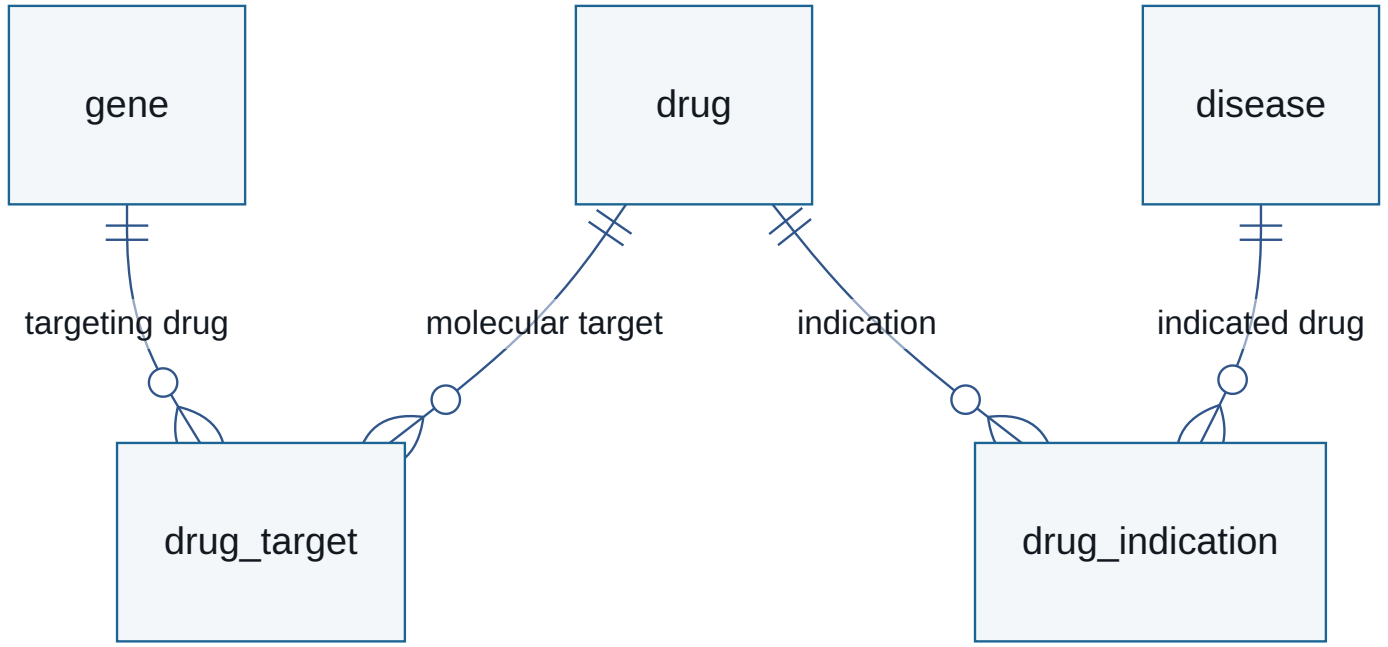
`go_parent` draws the DAG; `go_ancestor` answers questions about it; `go_stat` carries the information content that makes a GO column rankable.

Disease: hierarchy and therapeutic areas



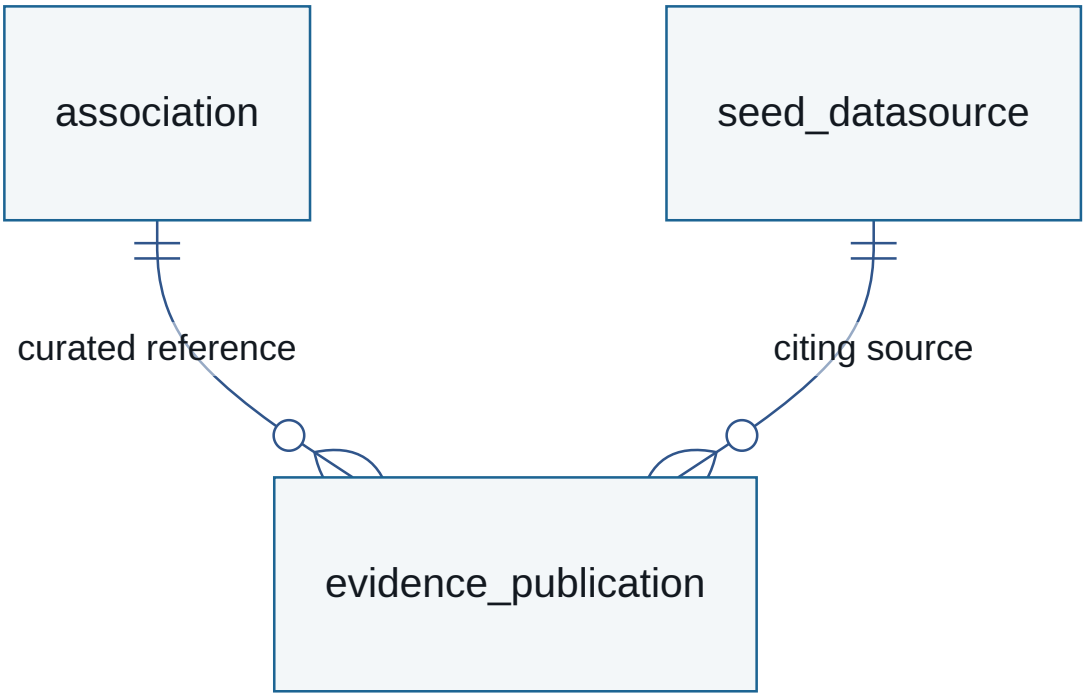
The closure is what lets a co-occurrence query exclude a disease's own ancestors and descendants before ranking.

Pharmacology: drugs, targets and indications



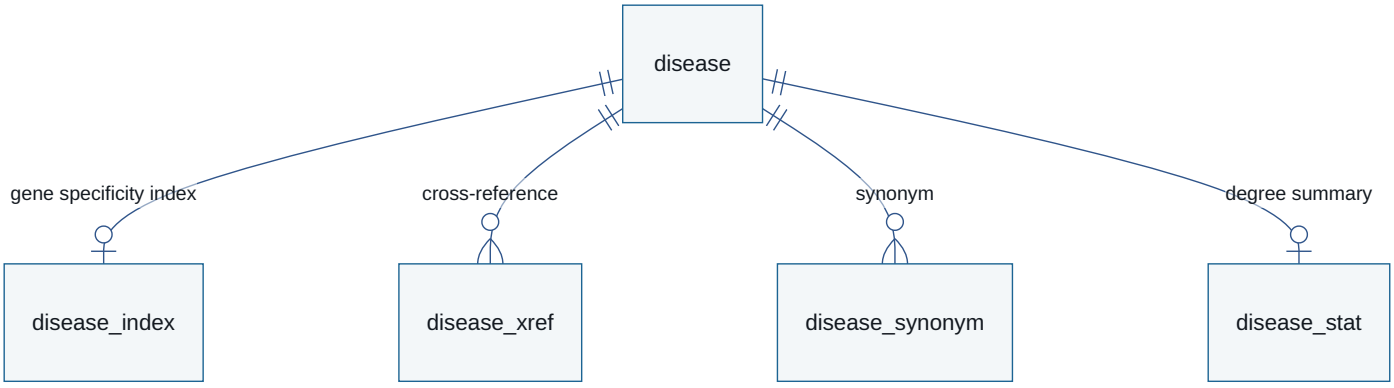
The layer behind the explorer's drug column. It reaches both hubs -- a drug points at a gene through `drug_target` and at a disease through `drug_indication` -- which is what makes gene-mediated repurposing questions a two-hop join. Note this layer overlaps the `clinical_precedence` evidence source: both derive from the same ChEMBL records, so drug evidence in a score is not independent of the drugs shown.

Literature references



Curated PubMed identifiers behind individual edges, attributed to the source that cited them. Coverage is heavily skewed toward rare and monogenic disease, because the curated sources are and the text-mining corpus is not ingested.

Disease: vocabulary and summary



Cross-references are the route to recognising the same disease arriving under a different vocabulary.

Relationships

The 33 relationships in the diagram, with the join that implements each. Read parent to child; every child side is optional, since a gene may have no disease association and a disease term may have no annotated gene.

Foreign keys are **enforced by the build rather than declared as constraints**. DuckDB is used here as an analytical store rebuilt in one atomic pass from an immutable snapshot, so there is no window in which a partial write could orphan a row; instead `validate.py` asserts referential integrity after every build and exits non-zero on any orphan. The trade is that ad-hoc writes outside the build are unguarded — acceptable because the graph is never edited in place.

Parent	Child	Relationship	Join	Notes
`disease`	`association`	associated gene	`association.disease_id = disease.disease_id`	The headline disease-gene edge, one row per pair, scored 0-1.
`gene`	`association`	associated disease	`association.gene_id = gene.gene_id`	The same edge from the gene side.
`association`	`association_source`	decomposed by evidence	`association_source.(disease_id, gene_id) = association.(disease_id, gene_id)`	One row per contributing datasource. This is what makes a score filter mean something specific rather than filtering a blended number.
`seed_datasource`	`association_source`	evidence source	`association_source.datasource_id = seed_datasource.datasource_id`	Curated labels, descriptions and Open Targets' own source weights.
`gene`	`gene_go`	GO annotation	`gene_go.gene_id = gene.gene_id`	Carries the GO evidence code and aspect, so an electronically inferred annotation is distinguishable from an experimental one.
`go_term`	`gene_go`	annotated gene	`gene_go.go_id = go_term.go_id`	The same annotation from the term side.
`gene`	`gene_go_propagated`	annotation, closed over the DAG	`gene_go_propagated.gene_id = gene.gene_id`	gene_go unioned with every ancestor of every annotated term, so enrichment and roll-up run at any granularity as a plain join.
`go_term`	`go_parent`	as child	`go_parent.go_id = go_term.go_id`	is_a and part_of edges of the GO DAG. A unary many-to-many on go_term: this is the child role.
`go_term`	`go_parent`	as parent	`go_parent.parent_id = go_term.go_id`	The parent role of the same edge. A term averages several parents, so

Parent	Child	Relationship	Join	Notes
				the structure is a DAG rather than a tree.
go_term	go_ancestor	as descendant	go_ancestor.go_id = go_term.go_id	Transitive closure over go_parent, computed here. Descendant role.
go_term	go_ancestor	as ancestor	go_ancestor.ancestor_id = go_term.go_id	Ancestor role of the same closure. Both columns are foreign keys to go_term.go_id.
go_term	go_ancestor_published	as descendant (published)	go_ancestor_published.go_id = go_term.go_id	Open Targets' own ancestors array, retained unused so the difference from our is_a/part_of closure stays inspectable.
go_term	go_ancestor_published	as ancestor (published)	go_ancestor_published.ancestor_id = go_term.go_id	Ancestor role of the published array.
disease	disease_index	gene specificity index	disease_index.disease_id = disease.disease_id	GSI: how few genes drive this disease, i.e. the monogenic-to-polygenic axis. There is deliberately no GPI companion -- see indices.py.
gene	gene_index	specificity indices	gene_index.gene_id = gene.gene_id	DSI and DPI: how narrow a gene's disease footprint is, and how far across the disease classes it reaches.
go_term	go_stat	information content	go_stat.go_id = go_term.go_id	Propagated gene count and IC per term. The table that demotes 'protein binding' without a curated stopword list.
disease	disease_parent	as child	disease_parent.disease_id = disease.disease_id	The EFO/MONDO hierarchy. A unary many-to-many on disease: this is the

Parent	Child	Relationship	Join	Notes
				child role. 12,908 diseases have more than one parent, so this is a DAG, not a tree, and 25 roots have no parent row at all.
`disease`	`disease_parent`	as parent	`disease_parent.parent_id = disease.disease_id`	The parent role of the same edge.
`disease`	`disease_ancestor`	as descendant	`disease_ancestor.disease_id = disease.disease_id`	Transitive closure over disease_parent. Descendant role: one disease has up to 65 ancestors.
`disease`	`disease_ancestor`	as ancestor	`disease_ancestor.ancestor_id = disease.disease_id`	Ancestor role. One broad term has up to 25,048 descendants, which is the asymmetry that makes excluding an anchor's own relatives worth doing before ranking co-occurrence.
`disease`	`disease_area`	therapeutic area	`disease_area.(disease_id, area_id) = disease.disease_id`	Top-level grouping. Note Open Targets counts 'measurement', 'phenotype' and 'biological_process' among its therapeutic areas.
`drug`	`drug_target`	molecular target	`drug_target.drug_id = drug.drug_id`	One row per drug-protein pair, carrying the action type.
`gene`	`drug_target`	targeting drug	`drug_target.gene_id = gene.gene_id`	The same pair from the gene side. Only ~1,550 disease-associated genes carry any drug at all, so most genes have no row here.

Parent	Child	Relationship	Join	Notes
drug	drug_indication	indication	drug_indication.drug_id = drug.drug_id	Includes trials at any phase, not only approvals -- max_phase is what separates an approved indication from one that failed in phase 2.
disease	drug_indication	indicated drug	drug_indication.disease_id = disease.disease_id	The join that lets the explorer mark which drugs are already indicated for the anchor disease, and which are candidates from elsewhere.
association	evidence_publication	curated reference	evidence_publication.(disease_id, gene_id) = association.(disease_id, gene_id)	PubMed identifiers behind an edge, attributed to the source that cited them. Sparse by construction; see the table purpose.
seed_datasource	evidence_publication	citing source	evidence_publication.datasource_id = seed_datasource.datasource_id	Which curated resource carried the reference.
disease	disease_xref	cross-reference	disease_xref.disease_id = disease.disease_id	OMIM, UMLS, MeSH, ICD, DOID and others -- the route to recognising the same disease arriving under a different vocabulary.
disease	disease_synonym	synonym	disease_synonym.disease_id = disease.disease_id	Exact, related and narrow synonyms, for search.
disease	disease_stat	degree summary	disease_stat.disease_id = disease.disease_id	Precomputed gene counts, so the explorer can warn before drawing 4,000 edges.
gene	gene_stat	degree summary	gene_stat.gene_id = gene.gene_id	Disease and GO-term counts per gene.

Parent	Child	Relationship	Join	Notes
`gene`	`gene_synonym`	synonym	`gene_synonym.gene_id = gene.gene_id`	Symbol and name synonyms, for search.
`gene`	`gene_xref`	cross-reference	`gene_xref.gene_id = gene.gene_id`	HGNC, UniProt, ChEMBL and others.

Design Notes

1. Purpose

A local, rebuildable knowledge graph over human genes, Gene Ontology annotations and gene–disease associations, shaped so that **interesting subnetworks can be selected cheaply**. The design target is not "store the biology"; Open Targets already does that well. It is to precompute the things that let a query rank by *specificity* and *surprise* rather than by adjacency, because ranking by adjacency is what produces a hairball.

The catalog is the substrate for two products: an interactive explorer that runs in the browser over Parquet extracts, and educational content on bioxplore.org. Both redistribute the data, so licence is a hard selection criterion.

2. Source data

One source in v1: the **Open Targets Platform**, release 26.06, released under **CC0 1.0**. It ships as Parquet, so the raw layer needs no decoding — DuckDB reads the published files directly and the build stage does shaping, not parsing.

Open Targets is itself an integration, so this transitively includes EFO, MONDO, GO, Ensembl, ChEMBL, ClinVar, Orphanet, Genomics England and IMPC. Crucially their `association_by_datasource` table names the contributing source for every edge, so provenance survives into our schema instead of being flattened into one opaque number — the specific failing of the DisGeNET-based prototype this replaces.

The build started with five datasets — `target`, `disease`, `go`, `association_overall_direct`, `association_by_datasource_direct` — and has since added two more layers. **Pharmacology**: `drug_molecule`, `drug_mechanism_of_action` and `clinical_indication`, which give the explorer its drug column. **Curated literature**: the per-source `evidence_*` tables, read for their `literature` arrays only, to attach PubMed identifiers to individual edges. `evidence_europepmc` is excluded at 12 GB, which is why reference coverage is rare-disease shaped rather than uniform.

Still deferred: molecular interactions, baseline expression, variant-level credible sets, target prioritisation and mouse/human phenotype tables. See `docs/SOURCES.md` for licensing and for what was rejected.

3. Architecture: three layers

Raw — `data/raw/<release>/`, the published Parquet files byte-for-byte, plus a manifest recording every URL, size and sha256. Never edited, never queried by the application.

Core — the normalized tables in `data/bioxplore.duckdb`: entities (`gene`, `disease`, `go_term`), their relationships (`gene_go`, `association`, `association_source`), and the ontology structure (`*_parent`, `*_ancestor`). This layer is a faithful flattening of the source and contains no judgement.

Derived — tables that exist only to make selection possible: `gene_go_propagated`, `go_stat`, `disease_stat`, `gene_stat`. This is where the project's opinions live. Every one of them is a deterministic function of the core, so they can be recomputed and argued with independently.

A build is a deterministic function of (release, seeds, code). Open Targets versions its releases, so the release string alone identifies the input — two people building 26.06 a month apart get the same database.

4. The selection problem (the heart)

Joining disease → gene → GO naïvely produces a picture no one can read. Two causes, both measurable in this database rather than merely asserted.

4.1 Generic annotations dominate

`GO:0005515` (*protein binding*) is **directly** annotated to 13,295 of 20,097 protein-coding genes; the next most directly-annotated terms are `nucleus`, `cytoplasm`, `membrane`, `plasma membrane` and `cytosol`. Counted after propagation up the DAG the picture is worse still — the three ontology roots and `cellular anatomical structure` each reach more than 19,000 genes. These terms connect nearly everything to nearly everything and discriminate nothing; they are pure hairball fuel, and they are what fills the right-hand column of a naïve Disease→Gene→GO diagram.

The fix is **information content**. For a term *t*, with *n(t)* genes annotated to *t* or anything below it in the ontology and *N* genes annotated to anything:

> IC(t) = −ln(n(t) / N)

`protein binding` scores near zero; a term covering eight genes scores high. Ranking a GO column by IC instead of by edge count demotes all six offenders automatically, with no hand-maintained stopword list to curate or defend. `go_stat` carries `ic`, `n_genes` (propagated) and `n_genes_direct` for every term.

4.2 Ontological neighbours masquerade as findings

Ask "which other diseases share genes with this one" and the top answers tend to be the same disease under a different label, or its own parent. The prototype's Alzheimer's view returned *Dementia*, *Memory impairment* and *Frontotemporal dementia* — a vocabulary artifact presented as a discovery.

The fix needs the disease hierarchy, which is why `disease_ancestor` is a core table: a query can exclude a term's own ancestors and descendants before ranking, leaving comorbidity claims that are actually claims.

4.3 Propagation makes granularity a choice

A gene annotated to *mitochondrion inheritance* is, by the structure of the DAG, also annotated to *biological process*. `gene_go_propagated` materializes that closure once, so enrichment and roll-up run at **any** level of the ontology as a plain join rather than a graph traversal. This is what makes semantic zoom — collapse two hundred specific terms into a dozen ancestors, then drill in — a cheap operation.

Both ontologies ship a precomputed ancestor list, so the closures are a flattening, not a transitive-closure computation.

4.4 Evidence is not uniform, and the schema says so

`gene_go` keeps the GO **evidence code** and aspect per annotation. `IEA` is electronically inferred with no curator review and is the single most common code; `IDA`, `IMP` and `EXP` are experimental. A tool that treats them alike overstates what it knows.

Likewise `association_source` keeps the contributing datasource for every disease–gene edge, and `seeds/datasource.csv` records Open Targets' own weighting — text-mined `europepmc` evidence is down-weighted fivefold relative to curated sources. A score filter over this table means something specific; a score filter over a blended number does not.

5. Tables

Core entities are `gene`, `disease` and `go_term`. Relationships are `gene_go` (gene → GO annotation, with evidence), `association` (disease → gene, headline score) and `association_source` (the same edges decomposed by evidence source). Ontology structure is carried twice per ontology: `*_parent` for the direct edges that draw a hierarchy, `*_ancestor` for the closure that answers questions about it.

The derived layer adds `gene_go_propagated`, `go_stat`, `disease_stat`, `gene_stat`, `gene_index` (DSI/DPI per gene) and `disease_index` (GSI per disease).

DSI inverts to the disease side cleanly — GSI is the monogenic-to-polygenic axis — but **DPI does not**, and the failure is instructive. Counting distinct classes only discriminates while the set is small relative to the class count. A gene sits in a handful of diseases, so its therapeutic-area count separates; a disease has hundreds of genes, so with the 18 top-level GO biological processes as the class system, 39.5% of diseases with 100–499 genes and **100%** of those with 500+ genes touch every class. Correlation with GSI is −0.85: a saturating GPI would measure gene-set size, not biology. The breadth axis for diseases instead needs a *concentration* measure (normalised entropy over a finer, information-content-selected GO slim) — the "mechanistic coherence" metric, not yet built. `seed_datasource` is curated knowledge, not source data.

Two layers hang off the hubs without belonging to either. **Pharmacology** is `drug` (molecule, modality, highest clinical phase), `drug_target` (which protein it acts on, and how) and `drug_indication` (which diseases it is approved for or trialled against). It touches both hubs, which is what makes gene-mediated repurposing a two-hop join: disease → gene → drug, with `drug_indication` marking which of the drugs found that way are already indicated for the anchor.

That layer overlaps the evidence, and the overlap is not incidental. Open Targets' `clinical_precedence` evidence source derives from the same ChEMBL records, so a gene selected partly *because* a drug targets it will

then show you that drug. The schema keeps them separate — evidence in `association_source`, pharmacology in `drug_*` — precisely so a query can switch one off and recompute without disturbing the other.

`evidence_publication` carries curated PubMed identifiers per (edge, source), built by probing each `evidence_*` table for a `literature` column — `evidence_clingen` has none, so the shape cannot be assumed. Coverage is skewed by construction: 62.8% of edges for diseases with 1–4 genes carry a reference against 1.2% for diseases with 100 or more, because the sources that cite are the rare-disease curators and the text-mining corpus is not ingested. Absence of a paper is not absence of evidence, and every view that shows this column says so.

Full column lists and row counts are generated by introspection in the Table Inventory and Columns sections of this document, so they cannot drift.

6. Source quirks worth knowing

`disease` is not a table of diseases. Open Targets ships the whole EFO index: 47,080 terms, of which 17,441 are OBA (biological attributes), 2,322 HP phenotypes, 477 GO terms, and a handful of PATO, UBERON, MP, GSSO, OBI, NCIT and OGMS entries. Roughly 26,800 are MONDO/EFO/Orphanet/OTAR disease terms. Open Targets scores associations against non-disease terms too, so filtering them out would silently discard real data — instead every row carries a `prefix` column and queries are expected to be explicit. A table named `disease` that quietly contains *"biological_process"* is a trap; a tagged one is a feature. Note also that Open Targets lists `measurement`, `phenotype` and `biological_process` among its "therapeutic areas".

Open Targets' published GO `ancestors` array is not the is_a/part_of closure. It also contains terms reachable by other relations. The clearest symptom: every apoptosis term lists `bleb assembly` (GO:0032060) among its ancestors, although bleb assembly's own `isA` / `partOf` are unrelated to apoptosis — presumably an inverse `has_part`. Propagating over it inflated bleb assembly from 7 annotated genes to 1,541 and dropped its information content from ~7 to 2.6, turning a highly specific term into one that scores as generic. **13,370 of 48,321 terms were affected**, and the distortion is worst exactly where it matters: `vacuole` 4,978 → 905, `DNA-templated transcription` 3,284 → 482, `execution phase of apoptosis` 1,539 → 49. `go_ancestor` is therefore computed here by transitive closure over `go_parent`, and the published version is retained as `go_ancestor_published` so the difference stays inspectable. `regulates` is excluded: the true path rule propagates annotations over is_a and part_of only.

Two GO identifier conventions coexist. The `go` dataset and `target.go[].id` use colons (`GO:0005515`); the ids imported into the EFO disease index use underscores (`GO_0008150`). They are not joinable without normalization, and they mean different things in context. Do not cross them.

Direct vs indirect associations. Open Targets publishes both. A *direct* association attaches evidence to the exact disease term it was reported against; an *indirect* one also inherits evidence from that term's descendants. We ingest **direct** and keep the full ancestry separately, so roll-up remains a query-time choice rather than being baked into the numbers. `association_overall_indirect` can be added later without disturbing anything.

`timeseries` is not ingested. Both association tables carry a large nested per-year novelty curve. It is never selected, so Parquet's columnar layout means it is not read off disk — that accounts for most of the 2.4 GB of raw data that does not appear in the database. `currentNovelty` is kept as a scalar.

Most genes are not protein-coding. `target` has 78,691 rows but only 20,097 are protein-coding; 34,880 are lncRNA and 9,487 processed pseudogenes. `is_protein_coding` is materialized because almost every biological question wants it and almost no default should silently apply it.

7. Rebuilding

```
PYTHONPATH=src .venv/bin/python src/fetch.py       # -> data/raw/<release>/
PYTHONPATH=src .venv/bin/python src/build.py       # -> data/bioxplore.duckdb
PYTHONPATH=src .venv/bin/python src/validate.py    # -> docs/build_report.md
PYTHONPATH=src .venv/bin/python src/design_doc.py  # -> this document
```

`build.py` drops and recreates the database in one pass; `--stages` rebuilds named stages into an existing file. Foreign keys are enforced by the build rather than declared as constraints — `validate.py` asserts referential integrity after every build and exits non-zero on any orphan.

Table Inventory

Every table in the built graph, with live row counts and the complete column list. Generated by introspection, so this section cannot drift from the database.

Table	Rows	Columns	Purpose
association_source	7,842,921	5	The same edges decomposed by contributing evidence source -- provenance kept rather than blended away.
association	7,554,123	5	Disease to gene, the headline Open Targets score and evidence count per pair.
gene_go_propagated	1,539,446	2	gene_go closed over the GO DAG: every gene paired with every ancestor of every term it is annotated to.
go_ancestor_published	646,714	2	Open Targets' published ancestors array. Retained for comparison only; it is not the is_a/part_of closure.
gene_go	641,624	5	The Gene to GO layer, one row per annotation, carrying the GO evidence code and aspect.
go_ancestor	474,733	2	Transitive ancestor closure of the GO DAG, computed here from the direct is_a/part_of edges.
disease_ancestor	385,020	2	Transitive ancestor closure of the disease hierarchy.
gene_xref	361,572	3	External identifiers (HGNC, UniProt, ChEMBL, and others).
gene_synonym	274,815	3	Alternative symbols and names, for search and for resolving user input.
evidence_publication	158,723	4	Curated literature references behind individual disease-gene edges, one row per (edge, source, PMID). Not text-mined: Europe PMC's own evidence is 12 GB and is not ingested, so coverage is rare-disease shaped.
disease_xref	132,647	3	External vocabulary identifiers (OMIM, UMLS, MeSH, ICD, DOID).
disease_synonym	113,039	3	Exact, related and narrow synonyms, for search.
drug_indication	86,468	3	Diseases a drug is approved for or has been trialled against, at any phase. The disease side.
gene	78,691	11	Identity spine for genes: Ensembl id, approved symbol and name, biotype, genomic location.
gene_stat	78,691	5	Precomputed per-gene degree: associated diseases and GO terms.
disease_area	70,170	2	Membership in Open Targets' top-level therapeutic areas.
go_parent	64,862	3	Direct is_a and part_of edges of the GO DAG.
disease_parent	63,582	2	Direct parent edges of the EFO/MONDO hierarchy.
go_stat	48,321	7	Per-term gene counts and information content -- the ranking signal that makes a GO column readable.
go_term	48,321	4	GO term definitions: label, namespace, obsolescence flag.
disease	47,080	6	The Open Targets EFO index. Not only diseases: every row carries a prefix so a query can say which branches it means.
disease_stat	47,080	6	Precomputed per-disease degree, including counts above a score threshold.
drug	22,407	5	Drug molecules from ChEMBL by way of Open Targets: name, modality and the highest clinical phase the molecule has reached.
gene_index	17,247	9	Per-gene disease specificity (DSI) and pleiotropy (DPI) indices, after DisGeNET's measures of the same names.

Table	Rows	Columns	Purpose
disease_index	11,876	12	Per-disease gene specificity index (GSI), the inverse framing of DSI. No pleiotropy counterpart: it saturates.
drug_target	8,991	4	Which protein a drug acts on, and how -- inhibitor, agonist, antagonist. The gene side of the pharmacology layer.
seed_datasource	20	7	Curated labels, descriptions and source weights for the evidence datasources. Knowledge, not source data.
build_meta	1	3	Build provenance: Open Targets release, build timestamp, DuckDB version.

Columns

association

5 columns · 7,554,123 rows

Column	Type	Key
EDGE		
disease_id	VARCHAR	PK, FK
gene_id	VARCHAR	PK, FK
STRENGTH		
score	DOUBLE	
evidence_count	BIGINT	
novelty	DOUBLE	

association_source

5 columns · 7,842,921 rows

Column	Type	Key
disease_id	VARCHAR	PK, FK
gene_id	VARCHAR	PK, FK
datasource_id	VARCHAR	PK, FK
score	DOUBLE	
evidence_count	BIGINT	

build_meta

3 columns · 1 rows

Column	Type	Key
ot_release	VARCHAR	
built_at	TIMESTAMP	
duckdb_version	VARCHAR	

disease

6 columns · 47,080 rows

Column	Type	Key
disease_id	VARCHAR	PK
name	VARCHAR	
description	VARCHAR	
prefix	VARCHAR	
is_therapeutic_area	BOOLEAN	
is_leaf	BOOLEAN	

disease_ancestor

2 columns · 385,020 rows

Column	Type	Key
disease_id	VARCHAR	PK, FK
ancestor_id	VARCHAR	PK, FK

disease_area

2 columns · 70,170 rows

Column	Type	Key
disease_id	VARCHAR	PK, FK
area_id	VARCHAR	PK, FK

disease_index

12 columns · 11,876 rows

Column	Type	Key
disease_id	VARCHAR	PK, FK
name	VARCHAR	
prefix	VARCHAR	
n_genes_lo	BIGINT	
n_genes_hi	BIGINT	
gsi_lo	DOUBLE	
gsi_hi	DOUBLE	
retention	DOUBLE	
score_lo	DECIMAL(2,1)	
score_hi	DECIMAL(2,1)	
n_genes_total_lo	BIGINT	
n_genes_total_hi	BIGINT	

disease_parent

2 columns · 63,582 rows

Column	Type	Key
disease_id	VARCHAR	PK, FK
parent_id	VARCHAR	PK, FK

disease_stat

6 columns · 47,080 rows

Column	Type	Key
disease_id	VARCHAR	PK, FK
name	VARCHAR	
prefix	VARCHAR	
n_genes	BIGINT	
n_genes_score_ge_0_5	BIGINT	
max_score	DOUBLE	

disease_synonym

3 columns · 113,039 rows

Column	Type	Key
disease_id	VARCHAR	FK
synonym	VARCHAR	
kind	VARCHAR	

disease_xref

3 columns · 132,647 rows

Column	Type	Key
disease_id	VARCHAR	FK
source	VARCHAR	
xref_id	VARCHAR	

drug 5 columns · 22,407 rows

Column	Type	Key
IDENTITY		
drug_id	VARCHAR	PK
name	VARCHAR	
MODALITY		
drug_type	VARCHAR	
CLINICAL STAGE		
max_phase	VARCHAR	
description	VARCHAR	

drug_indication 3 columns · 86,468 rows

Column	Type	Key
drug_id	VARCHAR	PK, FK
disease_id	VARCHAR	PK, FK
max_phase	VARCHAR	

drug_target 4 columns · 8,991 rows

Column	Type	Key
drug_id	VARCHAR	PK, FK
gene_id	VARCHAR	PK, FK
action_type	VARCHAR	
mechanism	VARCHAR	

evidence_publication 4 columns · 158,723 rows

Column	Type	Key
disease_id	VARCHAR	FK
gene_id	VARCHAR	FK
datasource_id	VARCHAR	FK
pmid	VARCHAR	

gene 11 columns · 78,691 rows

Column	Type	Key
IDENTITY		
gene_id	VARCHAR	PK
symbol	VARCHAR	
name	VARCHAR	
CLASSIFICATION		
biotype	VARCHAR	
is_protein_coding	BOOLEAN	
GENOMIC LOCATION		
chromosome	VARCHAR	
start_pos	BIGINT	
end_pos	BIGINT	
strand	INTEGER	
tss	BIGINT	
ANNOTATION		
function_description	VARCHAR	

gene_go 5 columns · 641,624 rows

Column	Type	Key
gene_id	VARCHAR	PK, FK
go_id	VARCHAR	PK, FK
aspect	VARCHAR	
evidence	VARCHAR	
source	VARCHAR	

gene_go_propagated 2 columns · 1,539,446 rows

Column	Type	Key
gene_id	VARCHAR	PK, FK
go_id	VARCHAR	PK, FK

gene_index 9 columns · 17,247 rows

Column	Type	Key
gene_id	VARCHAR	PK, FK
symbol	VARCHAR	
biotype	VARCHAR	
n_diseases	BIGINT	
n_classes	BIGINT	
dsi	DOUBLE	
dpi	DOUBLE	
n_diseases_total	BIGINT	
n_classes_total	BIGINT	

gene_stat 5 columns · 78,691 rows

Column	Type	Key
gene_id	VARCHAR	PK, FK
symbol	VARCHAR	
biotype	VARCHAR	
n_diseases	BIGINT	
n_go_terms	BIGINT	

gene_synonym 3 columns · 274,815 rows

Column	Type	Key
gene_id	VARCHAR	FK
synonym	VARCHAR	
kind	VARCHAR	

gene_xref 3 columns · 361,572 rows

Column	Type	Key
gene_id	VARCHAR	FK
source	VARCHAR	
xref_id	VARCHAR	

go_ancestor 2 columns · 474,733 rows

Column	Type	Key
go_id	VARCHAR	PK, FK
ancestor_id	VARCHAR	PK, FK

go_ancestor_published 2 columns · 646,714 rows

Column	Type	Key
go_id	VARCHAR	PK, FK
ancestor_id	VARCHAR	PK, FK

go_parent 3 columns · 64,862 rows

Column	Type	Key
go_id	VARCHAR	PK, FK
parent_id	VARCHAR	PK, FK
relation	VARCHAR	

go_stat 7 columns · 48,321 rows

Column	Type	Key
IDENTITY		
go_id	VARCHAR	PK, FK
name	VARCHAR	
namespace	VARCHAR	
COVERAGE		
n_genes_direct	BIGINT	
n_genes	BIGINT	
SPECIFICITY		
ic	DOUBLE	
n_genes_universe	BIGINT	

go_term 4 columns · 48,321 rows

Column	Type	Key
go_id	VARCHAR	PK
name	VARCHAR	
namespace	VARCHAR	
is_obsolete	BOOLEAN	

seed_datasource

7 columns · 20 rows

Column	Type	Key
datasource_id	VARCHAR	PK
label	VARCHAR	
datatype	VARCHAR	
organisation	VARCHAR	
weight	DOUBLE	
homepage	VARCHAR	
description	VARCHAR	