# Package: rwevalidate (via r-universe)

July 22, 2026

**Title** Validate Patient Cohorts for Real-World Evidence Studies on the OMOP Common Data Model

**Version** 0.1.2

**Description** Validates instantiated patient cohorts on an Observational Medical Outcomes Partnership (OMOP) Common Data Model (CDM) database for real-world-evidence (RWE) studies. From a single function call it produces a structured validation report in Hypertext Markup Language (HTML) and JavaScript Object Notation (JSON) covering concept coverage, cohort attrition, temporal data density, and covariate feasibility against a comparator. The checks are aligned with the United States Food and Drug Administration (FDA) guidance on real-world data and evidence, FDA (2023) <https://www.fda.gov/media/171667/download>, the Harmonized Protocol Template to Enhance Reproducibility (HARPER), Wang and others (2022) <doi:10.1002/pds.5507>, and the Reporting of Studies Conducted Using Observational Routinely-Collected Data for Pharmacoepidemiology (RECORD-PE) statement, Langan and others (2018) <doi:10.1136/bmj.k3532>. A self-contained example database is bundled so the checks can be run without a live database connection.

**License** MIT + file LICENSE

**URL** https://github.com/tavakohr/rwevalidate

**BugReports** https://github.com/tavakohr/rwevalidate/issues

**Encoding** UTF-8

**Language** en-US

**Roxygen** list(markdown = TRUE)

**Depends** R (>= 4.1)

**Imports** DBI, RPostgres, jsonlite, rmarkdown, glue, cli, utils

**Suggests** testthat (>= 3.0.0), duckdb, ggplot2, knitr, covr

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Config/roxygen2/version** 8.0.0

**Config/pak/sysreqs** cmake make libuv1-dev libpq-dev

**Repository** https://tavakohr.r-universe.dev

**Date/Publication** 2026-07-11 17:22:43 UTC

**RemoteUrl** https://github.com/tavakohr/rwevalidate

**RemoteRef** HEAD

**RemoteSha** 80010f8512b6136b639b18d89f61bda74f77a8bc

# Contents



---

`build_check_summary` *Build the traffic-light validation summary table*

---

## Description

Collects each module's flags into one row per validation section, mapping to the regulatory framework the section addresses.

## Usage

```
build_check_summary(results)
```

## Arguments

| | |
|---|---|
| `results` | The combined results list (named module outputs, e.g. `attrition`, `density`). Modules absent from the list are skipped. |

## Value

A data.frame with columns `section`, `status`, `maps_to`, `detail`.

cdm_connect *Connect to an OMOP CDM database*

## Description

Opens a PostgreSQL connection via RPostgres and validates that the required clinical tables (person, observation_period, condition_occurrence, drug_exposure, measurement, visit_occurrence, death) exist in cdm_schema and the vocabulary tables (concept, concept_ancestor) exist in vocab_schema before returning. For split-schema CDMs (clinical and vocabulary in different schemas) pass vocab_schema; it defaults to cdm_schema for single-schema builds.

## Usage

```
cdm_connect(
  host = "localhost",
  port = 5432,
  dbname,
  user,
  password,
  cdm_schema,
  vocab_schema = cdm_schema
)
```

## Arguments

| | |
|---|---|
| host | Database host. Default "localhost". |
| port | Database port. Default 5432. |
| dbname | Database name. |
| user | Database user. |
| password | Database password. |
| cdm_schema | Schema holding the clinical CDM tables. |
| vocab_schema | Schema holding the vocabulary tables. Defaults to cdm_schema. |

## Value

A live DBI connection to the CDM database.

## Examples

```
## Not run:
# Requires a live PostgreSQL OMOP CDM server and real credentials, so this
# cannot be executed during checks. For a runnable, database-free demo of
# the package, see example_cdm() and validate_cohort().
con <- cdm_connect(
  dbname = "omop", user = "me", password = "secret",
  cdm_schema = "mimiciv_omop", vocab_schema = "vocab"
```

```
)
cdm_disconnect(con)

## End(Not run)
```

---

`cdm_disconnect` *Disconnect from a CDM database*

---

## Description

Disconnect from a CDM database

## Usage

```
cdm_disconnect(con)
```

## Arguments

| | |
|---|---|
| `con` | A live `DBI` connection returned by `cdm_connect()`. |

## Value

Invisibly returns the result of `DBI::dbDisconnect()`.

## Examples

```
# Works on any DBI connection, demonstrated here on the bundled in-memory
# DuckDB example CDM (no database server needed).
if (requireNamespace("duckdb", quietly = TRUE)) {
  con <- example_cdm()
  cdm_disconnect(con)
}
```

---

`classify_status` *Classify a vector of module flags into a traffic-light status*

---

## Description

Flags are strings prefixed `"FAIL:"` or `"WARN:"` (see the module functions). Any `FAIL` -> `"fail"` (red); else any `WARN` -> `"warn"` (amber); else `"pass"` (green).

## Usage

```
classify_status(flags)
```

### Arguments

| | |
|---|---|
| flags | Character vector of flag messages (may be empty). |

### Value

A single string: "pass", "warn", or "fail".

---

## example_cdm *Create a small in-memory OMOP CDM for examples and demos*

---

### Description

Builds a tiny synthetic OMOP CDM (20 persons) in an in-memory DuckDB database and returns a live connection. Every table rwevalidate queries for the concept, attrition, density, and covariate modules is populated, together with a cohort table that has the standard subject_id, cohort_definition_id, cohort_start_date, and cohort_end_date columns. The cohort table carries two arms so all four modules can be demonstrated: persons 1-10 are the target cohort (cohort_definition_id = 1) and persons 11-20 are a comparator (cohort_definition_id = 2). The clinical tables and a minimal vocabulary both live in the default main schema.

### Usage

```
example_cdm()
```

### Details

This exists so the package can be tried, and its examples can run, without a live database connection. It is a synthetic demo fixture, not a substitute for a real CDM. The duckdb package (a soft dependency) must be installed. The caller owns the returned connection and should close it with cdm_disconnect().

### Value

A live DBI connection to an in-memory DuckDB OMOP CDM.

### Examples

```
if (requireNamespace("duckdb", quietly = TRUE)) {
  con <- example_cdm()
  attrition <- run_attrition(con, cdm_schema = "main",
                             cohort_table = "cohort", cohort_id = 1,
                             vocab_schema = "main")
  print(attrition$cohort_size)
  cdm_disconnect(con)
}
```

`render_validation_report` *Render the validation report (HTML) and write the JSON sidecar*

## Description

Renders the bundled R Markdown template to a self-contained HTML report and writes a machine-readable `validation_results.json` alongside it.

## Usage

```
render_validation_report(
  results,
  output_dir,
  cdm_schema,
  cohort_id,
  package_version = as.character(utils::packageVersion("rwevalidate")),
  run_date = Sys.time(),
  quiet = TRUE
)
```

## Arguments

| | |
|---|---|
| `results` | Combined results list (e.g. `list(attrition = ..., density = ...)`, optionally `data_source = ...`). |
| `output_dir` | Directory to write the report into (created if needed). |
| `cdm_schema` | CDM schema name (for the report header / JSON). |
| `cohort_id` | Cohort definition id (for the report header / JSON). |
| `package_version` | Package version string. Defaults to the installed `rwevalidate` version. |
| `run_date` | Timestamp for the run. Defaults to `base::Sys.time()`. |
| `quiet` | Passed to `rmarkdown::render()`. Default `TRUE`. |

## Value

Invisibly, a list with `html`, `json`, and `check_summary`.

`run_attrition` *Module 2 - Cohort Attrition Audit*

## Description

Profiles an instantiated cohort: size, index-date distribution, prior observation coverage, and demographics at index. Maps to HARPER section 5 and RECORD-PE item 6. All thresholds are arguments.

## Usage

```
run_attrition(
  con,
  cdm_schema,
  cohort_table,
  cohort_id,
  obs_window = c(-365, 0),
  vocab_schema = "vocab",
  obs_warn_pct = 70,
  obs_fail_pct = 50,
  attrition_warn_threshold = 0.5
)
```

## Arguments

| | |
|---|---|
| `con` | A live `DBI` connection (see `cdm_connect()`). |
| `cdm_schema` | Schema holding the clinical CDM tables (e.g. `"mimic_cdm"`). |
| `cohort_table` | Cohort table, schema-qualified (e.g. `"results.rwevalidate_test_cohort"`). Must have columns `subject_id`, `cohort_definition_id`, `cohort_start_date`, `cohort_end_date`. |
| `cohort_id` | Integer cohort definition id to profile. |
| `obs_window` | Length-2 numeric `c(pre, post)` in days relative to index. The required prior-observation window is `abs(obs_window[1])` days (default 365). |
| `vocab_schema` | Schema holding the vocabulary tables (`concept`). In this split-schema environment the clinical schema's `concept` table is empty, so the gender label is resolved from `vocab_schema`. Default `"vocab"`. |
| `obs_warn_pct, obs_fail_pct` | Prior-observation coverage thresholds (percent of cohort with sufficient prior observation). Below `obs_warn_pct` flags `WARN`, below `obs_fail_pct` flags `FAIL`. Defaults 70 and 50. |
| `attrition_warn_threshold` | Reserved for stepwise inclusion-criteria attrition (single-criterion drop fraction). Not exercised in v0.1 because criteria lists are not yet a supported input. Default 0.5. |

## Details

This profiles the cohort as it was handed in. It reports the cohort-level counts that item 6 asks for, but it does not yet rebuild a step-by-step attrition flowchart (how many subjects each inclusion or exclusion criterion removed), because the cohort table does not carry the criteria that built it. Per-criterion attrition is planned for a later version.

## Value

A named list:

**cohort_size** integer - distinct subjects in the cohort

**index_date_summary** data.frame of `index_year`, `n`

**obs_coverage** data.frame: `cohort_n`, `n_sufficient`, `pct_sufficient`, `min_prior_days`. `cohort_n` counts every cohort subject. A subject whose index date falls outside any observation period counts as insufficient prior observation rather than being dropped, so `cohort_n` matches `cohort_size`.

**demographics** data.frame: `sex`, `mean_age`, `min_age`, `max_age`, `n`

**flags** character vector of `WARN:/FAIL:` messages (empty if all pass)

## Examples

```
if (requireNamespace("duckdb", quietly = TRUE)) {
  con <- example_cdm()
  res <- run_attrition(con, cdm_schema = "main", cohort_table = "cohort",
                       cohort_id = 1, vocab_schema = "main")
  res$cohort_size
  res$demographics
  cdm_disconnect(con)
}
```

---

# run_concepts *Module 1 - Concept Coverage*

---

## Description

Profiles whether the cohort-defining concepts are present, well-mapped, and richly represented in the CDM. Maps to FDA RWE Relevance (are the data fit for the question?). This is the first module to read the vocabulary schema.

## Usage

```
run_concepts(
  con,
  cdm_schema,
  cohort_table = NULL,
  cohort_id = NULL,
```

```
  concept_ids,
  domain = "condition",
  vocab_schema = "vocab",
  prevalence_warn_pct = 10,
  prevalence_fail_pct = 1,
  unmapped_warn_pct = 20
)
```

## Arguments

| | |
|---|---|
| con | A live DBI connection (see cdm_connect()). |
| cdm_schema | Schema holding the clinical CDM tables. |
| cohort_table | Cohort table, schema-qualified. Optional and currently unused; reserved for cohort-restricted prevalence in a later version. |
| cohort_id | Integer cohort definition id. Optional and reserved, see cohort_table. |
| concept_ids | Numeric vector of cohort-defining seed concept id(s). Their descendants are expanded via {vocab_schema}.concept_ancestor. |
| domain | Domain the seed concepts live in; one of the names of the internal domain map ("condition", "drug", "measurement", "procedure"). Default "condition". |
| vocab_schema | Schema holding the vocabulary tables. Default "vocab". |
| prevalence_warn_pct, prevalence_fail_pct | Concept-prevalence thresholds (percent of CDM persons with at least one descendant record). Below warn flags WARN, below fail flags FAIL. Defaults 10 and 1. |
| unmapped_warn_pct | Per-domain unmapped-record threshold (percent mapped to concept_id = 0) above which a WARN is flagged. Default 20. |

## Value

A named list:

**prevalence** data.frame: n_with, n_total, pct

**ancestor_coverage** data.frame: total_descendants, present_descendants, pct_present

**mapping_by_domain** data.frame: domain, n_records, n_unmapped, pct_mapped

**flags** character vector of WARN:/FAIL: messages

## Examples

```
if (requireNamespace("duckdb", quietly = TRUE)) {
  con <- example_cdm()
  # 316139 = "Heart failure" (SNOMED), the seed concept in the demo CDM.
  res <- run_concepts(con, cdm_schema = "main", concept_ids = 316139,
                      domain = "condition", vocab_schema = "main")
  res$prevalence
  res$mapping_by_domain
  cdm_disconnect(con)
}
```

`run_covariates` *Module 4 - Covariate Feasibility*

## Description

Compares a target cohort against a comparator on baseline covariates: standardized mean differences (SMD love-plot data), a per-arm prevalence table, and a simple power calculation. Maps to HARPER Sec.4 (comparability). Runs only when a `comparator_id` is supplied.

## Usage

```
run_covariates(
  con,
  cdm_schema,
  cohort_table,
  cohort_id,
  comparator_id,
  vocab_schema = "vocab",
  covariate_window = c(-365, 0),
  top_n = 20,
  smd_warn_threshold = 0.1,
  power_baseline_rate = 0.15,
  power_effect_rate = 0.3,
  power_sig_level = 0.05,
  power_warn_threshold = 0.8
)
```

## Arguments

| | |
|---|---|
| `con` | A live `DBI` connection (see `cdm_connect()`). |
| `cdm_schema` | Schema holding the clinical CDM tables. |
| `cohort_table` | Cohort table, schema-qualified, containing both arms. |
| `cohort_id` | Integer cohort definition id of the target arm. |
| `comparator_id` | Integer cohort definition id of the comparator arm. |
| `vocab_schema` | Schema holding the vocabulary tables. Default `"vocab"`. |
| `covariate_window` | Length-2 numeric `c(pre, post)` days defining the baseline window relative to index for comorbidity/drug covariates. Default `c(-365, 0)`. |
| `top_n` | Number of top conditions/drugs to profile. Default 20. |
| `smd_warn_threshold` | Absolute SMD above which a covariate is flagged imbalanced. Default 0.1 (standard). |
| `power_baseline_rate`, `power_effect_rate` | Assumed comparator and target event rates for the two-proportion power calculation. Defaults 0.15, 0.30. |

power_sig_level
: Significance level for the power calculation. Default 0.05.

power_warn_threshold
: Power below this value flags a WARN. Default 0.80.

## Value

A named list: smd_table, prevalence_table, power, flags.

## Examples

```
if (requireNamespace("duckdb", quietly = TRUE)) {
  con <- example_cdm()
  # The demo CDM ships two arms: target = 1, comparator = 2.
  res <- run_covariates(con, cdm_schema = "main", cohort_table = "cohort",
                        cohort_id = 1, comparator_id = 2, vocab_schema = "main")
  res$smd_table
  res$power
  cdm_disconnect(con)
}
```

---

# run_density *Module 3 - Temporal Data Density*

---

## Description

Profiles longitudinal data accrual around the index date: record density per patient per month across clinical domains, and follow-up completeness with a censoring-reason breakdown. Maps to FDA RWE Reliability (data accrual).

## Usage

```
run_density(
  con,
  cdm_schema,
  cohort_table,
  cohort_id,
  obs_window = c(-365, 365),
  sparse_warn_min = 1
)
```

## Arguments

| | |
|---|---|
| con | A live DBI connection (see cdm_connect()). |
| cdm_schema | Schema holding the clinical CDM tables (e.g. "mimic_cdm"). |
| cohort_table | Cohort table, schema-qualified. |

| | |
|---|---|
| cohort_id | Integer cohort definition id. |
| obs_window | Length-2 numeric c(pre, post) in days relative to index. Density is computed for records inside [index + pre, index + post]. Default c(-365, 365). |
| sparse_warn_min | Minimum total records a domain must have within the window before it is considered present; domains below this flag WARN (likely missing/sparse domain). Default 1 (flag only truly empty domains). |

## Value

A named list:

**density_by_domain** data.frame: domain, months_from_index, n_patients, n_records, records_per_patient

**followup_summary** data.frame: one row per censoring_reason with n and median_followup_days (median over that group)

**followup_detail** per-subject data.frame: subject_id, effective_end_date, followup_days, censoring_reason

**flags** character vector of WARN:/FAIL: messages

## Examples

```
if (requireNamespace("duckdb", quietly = TRUE)) {
  con <- example_cdm()
  res <- run_density(con, cdm_schema = "main", cohort_table = "cohort",
                     cohort_id = 1)
  head(res$density_by_domain)
  res$followup_summary
  cdm_disconnect(con)
}
```

---

# validate_cohort *Validate a patient cohort for real-world-evidence use*

---

## Description

One-call entry point: connects to the OMOP CDM (or uses a supplied connection), runs the attrition (Module 2) and temporal-density (Module 3) checks, and produces a self-contained HTML report plus a JSON sidecar aligned with FDA RWE and HARPER frameworks.

## Usage

```
validate_cohort(
  cdm_schema,
  cohort_table,
  cohort_id = 1,
  con = NULL,
  vocab_schema = "vocab",
```

```
  concept_ids = NULL,
  concept_domain = "condition",
  comparator_id = NULL,
  obs_window = c(-365, 0),
  density_window = c(-365, 365),
  output_dir = "./validation_report",
  host = "localhost",
  port = 5432,
  dbname = NULL,
  user = NULL,
  password = NULL,
  render_html = TRUE,
  quiet = TRUE
)
```

## Arguments

| | |
|---|---|
| cdm_schema | Schema holding the clinical CDM tables (e.g. "mimic_cdm"). |
| cohort_table | Cohort table, schema-qualified (e.g. "results.rwevalidate_test_cohort"). |
| cohort_id | Integer cohort definition id. Default 1. |
| con | Optional open DBI connection. If NULL (default) one is opened from the connection arguments and closed on exit. |
| vocab_schema | Schema holding the vocabulary tables. Default "vocab". Required because the clinical schema's concept table may be empty in split-schema builds. |
| concept_ids | Optional numeric vector of cohort-defining seed concept id(s). When supplied, Module 1 (concept coverage) runs and populates report Section 2. When NULL (default) Module 1 is skipped. |
| concept_domain | Domain the seed concepts live in ("condition", "drug", "measurement", "procedure"). Default "condition". |
| comparator_id | Optional integer cohort definition id of a comparator arm. When supplied, Module 4 (covariate feasibility) runs and populates report Section 5. When NULL (default) Module 4 is skipped. |
| obs_window | Length-2 numeric c(pre, post) days for attrition prior observation. Default c(-365, 0). |
| density_window | Length-2 numeric c(pre, post) days for density. Default c(-365, 365). |
| output_dir | Directory for the report + JSON. Default "./validation_report". |
| host, port, dbname, user, password | PostgreSQL connection arguments, used only when con is NULL. |
| render_html | Render the HTML report. Default TRUE. If Pandoc is unavailable, only the JSON sidecar is written (with a warning). |
| quiet | Passed to rmarkdown::render(). Default TRUE. |

## Details

Supply either an open con (e.g. for DuckDB testing) or PostgreSQL connection arguments (dbname, user, password, ...). Connections opened internally are closed on exit; a connection passed in via con is left open.

Against a live PostgreSQL OMOP CDM, a typical call looks like:

```
validate_cohort(
  cdm_schema   = "mimic_cdm",
  cohort_table = "results.rwevalidate_test_cohort",
  cohort_id    = 1,
  dbname = "FHIR", user = "me", password = "secret",
  vocab_schema = "vocab",
  output_dir   = "./validation_report"
)
```

## Value

Invisibly, a list with results (module outputs + data_source), report (paths + check summary), and flags (all collected flags).

## Examples

```
# Runnable demo on a small in-memory DuckDB CDM (no database needed).
# render_html = FALSE writes only the JSON sidecar, so Pandoc is not required.
if (requireNamespace("duckdb", quietly = TRUE)) {
  con <- example_cdm()
  out <- validate_cohort(
    cdm_schema   = "main",
    cohort_table = "cohort",
    cohort_id    = 1,
    con          = con,
    vocab_schema = "main",
    output_dir   = tempfile("rwe_demo_"),
    render_html  = FALSE
  )
  print(out$report$check_summary)
  cdm_disconnect(con)
}
```

---

# write_results_json *Write the JSON sidecar (no rendering)*

---

## Description

Exposed separately so results can be persisted even when Pandoc is unavailable for HTML rendering.

## Usage

```
write_results_json(
  results,
  output_dir,
  cdm_schema,
  cohort_id,
  package_version = as.character(utils::packageVersion("rwevalidate")),
  run_date = Sys.time(),
  check_summary = NULL
)
```

## Arguments

| | |
|---|---|
| results | Combined results list (e.g. list(attrition = ..., density = ...), optionally data_source = ...). |
| output_dir | Directory to write the report into (created if needed). |
| cdm_schema | CDM schema name (for the report header / JSON). |
| cohort_id | Cohort definition id (for the report header / JSON). |
| package_version | Package version string. Defaults to the installed rwevalidate version. |
| run_date | Timestamp for the run. Defaults to base::Sys.time(). |
| check_summary | Optional pre-built summary; rebuilt if NULL. |

## Value

The path to the written JSON file (invisibly).

# Index