Package 'nordstatExtras'

April 21, 2026

Title Shared 'SQLite' Cache Backend for the 'nordstat' Package Family

Version 0.1.0

Description Provides a SQLite-backed cell-level cache that can be used as a drop-in backend by the nordstat family of packages ('rKolada', 'rTrafa', and 'pixieweb'). Designed for multi-user web applications where minimal fetch latency and asynchronous writes are required. Individual statistical values (``cells'') are stored in a gatekeeper schema with a sidecar table for arbitrary metadata dimensions, enabling deduplication across overlapping queries.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports DBI, RSQLite, rlang, tibble, jsonlite, digest

Suggests mirai, testthat (>= 3.0.0), withr

Config/testthat/edition 3

URL https://github.com/LCHansson/nordstatExtras

BugReports https://github.com/LCHansson/nordstatExtras/issues

NeedsCompilation no

Author Love Hansson [aut, cre]

Maintainer Love Hansson <love.hansson@gmail.com>

Repository CRAN

Date/Publication 2026-04-21 08:30:19 UTC

Contents

`nxt_cache_handler` *Drop-in cache handler for the nordstat family*

Description

Produces a closure matching the `cache_handler()` contract used by rKolada, rTrafa, and pixieweb: a function of `(method, df)` where `method` is one of `"discover"`, `"load"`, or `"store"`. Unlike the per-package `.rds` handlers, this one stores values in a shared SQLite file. Two kinds of cached content are supported:

Usage

```
nxt_cache_handler(
  source,
  entity,
  cache,
  cache_location,
  key_params = list(),
  kind = c("data", "metadata"),
  ttl_days = NXT_DEFAULT_TTL_DAYS,
  reconstruct_extra = list(),
  normalize_extra = list()
)
```

Arguments

`source`	One of `"kolada"`, `"trafa"`, `"pixieweb"`.
`entity`	Entity keyword (e.g. `"values"`, `"products"`, `"tables"`).
`cache`	Logical — whether caching is enabled.
`cache_location`	Either an `nxt_handle` from `nxt_open()` or a path / function-returning-a-path to a `.sqlite` file.
`key_params`	Named list of parameters that together with `source` + `entity` form a unique cache key.
`kind`	Either `"data"` (default, cell-level) or `"metadata"` (whole-BLOB). Determines the storage/TTL strategy.
`ttl_days`	Time-to-live in days. Defaults to 30.
`reconstruct_extra`	Named list of extra arguments forwarded to the reconstruct function — used when trafa/pixieweb need product/alias/etc. at load time. Ignored when `kind = "metadata"`.

normalize_extra
: Same for the normalize function, used at store time. Ignored when `kind = "metadata"` — metadata is serialized whole.

Details

- `kind = "data"` (default) — statistical data stored at cell granularity with cross-query freshness. Matches the existing rKolada `get_values()` / rTrafa `get_data()` / pixieweb `get_data()` integration.
- `kind = "metadata"` — whole serialized objects (tibbles or lists) stored as BLOBs on the `queries` row. Matches the metadata-caching integration for `get_kpi()`, `get_products()`, `get_tables()`, etc. TTL is query-level.

Value

A function with signature `function(method, df = NULL)`.

Examples

```
path <- tempfile(fileext = ".sqlite")
handle <- nxt_open(path)

# Data caching (cell-level)
ch <- nxt_cache_handler(
  source = "kolada", entity = "values", cache = TRUE,
  cache_location = handle,
  key_params = list(kpi = "N03700", period = "2024")
)
ch("discover")   # FALSE - nothing cached yet

# Metadata caching (whole-BLOB)
ch_meta <- nxt_cache_handler(
  source = "kolada", entity = "kpi", cache = TRUE,
  cache_location = handle,
  kind = "metadata",
  key_params = list(id = NULL)
)
ch_meta("discover")   # FALSE

nxt_close(handle)
unlink(path)
```

---

nxt_clear *Clear cached queries from the nordstatExtras backend*

Description

Removes queries (and any cells / metadata rows no longer referenced) from the cache. Filters narrow the scope to a single source, entity, or kind.

Usage

```
nxt_clear(handle, source = NULL, entity = NULL, kind = NULL)
```

Arguments

handle	An nxt_handle from nxt_open().
source	Optional source filter ("kolada", "trafa", "pixieweb").
entity	Optional entity filter.
kind	Optional kind filter ("data" or "metadata"). NULL (default) clears both kinds.

Value

invisible(NULL). Emits a message summarising what was removed.

Examples

```
path <- tempfile(fileext = ".sqlite")
handle <- nxt_open(path)

# Clear all cached data for a specific source
nxt_clear(handle, source = "kolada")

# Clear everything
nxt_clear(handle)

nxt_close(handle)
unlink(path)
```

---

nxt_close *Close a nordstatExtras cache handle*

Description

Flushes any pending async writes and disconnects the underlying DBI connection.

Usage

```
nxt_close(handle)
```

Arguments

handle	A handle returned by nxt_open().

Value

invisible(NULL)

Examples

```
path <- tempfile(fileext = ".sqlite")
handle <- nxt_open(path)
nxt_close(handle)
unlink(path)
```

---

nxt_flush *Flush pending async writes*

Description

Blocks until all buffered writes for a handle have been committed.

Usage

```
nxt_flush(handle)
```

Arguments

handle	An nxt_handle.

Value

```
invisible(NULL)
```

Examples

```
path <- tempfile(fileext = ".sqlite")
handle <- nxt_open(path)
nxt_flush(handle)
nxt_close(handle)
unlink(path)
```

---

nxt_gc *Garbage-collect stale cache rows*

Description

Branches on queries.kind:

Usage

```
nxt_gc(handle, max_age_days = NXT_DEFAULT_TTL_DAYS)
```

Arguments

handle	An nxt_handle.
max_age_days	Maximum age in days. Default 30.

Details

- kind='data': deletes individual cells older than max_age_days, drops junction rows pointing at them, then removes queries that lost every cell. Queries that still have at least one fresh cell are kept.
- kind='metadata': deletes queries whose fetched_at is older than max_age_days (query-level TTL — metadata blobs are opaque and can't benefit from cross-query freshness).

Finally prunes orphan rows from cell_dims and meta_search.

Value

invisible(NULL).

Examples

```
path <- tempfile(fileext = ".sqlite")
handle <- nxt_open(path)

# Remove cells and metadata older than 30 days (default)
nxt_gc(handle)

# Custom TTL: remove anything older than 7 days
nxt_gc(handle, max_age_days = 7)

nxt_close(handle)
unlink(path)
```

---

nxt_is_backend *Detect whether a cache location should use the nordstatExtras backend*

Description

Returns TRUE when cache_location points at a .sqlite/.db file or an existing nxt_handle. Used by rKolada/rTrafa/pixieweb inside their cache_handler() functions to decide whether to delegate.

Usage

```
nxt_is_backend(cache_location)
```

Arguments

cache_location	A path, function returning a path, or nxt_handle.

Value

Logical scalar.

Examples

```
nxt_is_backend("my_cache.sqlite")   # TRUE
nxt_is_backend("my_cache.db")       # TRUE
nxt_is_backend("/tmp/cache_dir")    # FALSE
nxt_is_backend(tempdir())           # FALSE
```

---

nxt_open *Open or create a nordstatExtras SQLite cache*

Description

Opens a SQLite database at `path`, creating the file and applying the schema if it does not yet exist. Sets WAL mode and pragmas suitable for multi-process read/write access.

Usage

```
nxt_open(path, create = TRUE)
```

Arguments

`path`	Path to the `.sqlite` file.
`create`	Logical. If `FALSE`, `nxt_open()` fails when the file does not exist. Default `TRUE`.

Value

An object of class `nxt_handle` — a thin wrapper around the DBI connection that also carries the path and async-queue identity.

Examples

```
path <- tempfile(fileext = ".sqlite")
handle <- nxt_open(path)
print(handle)
nxt_close(handle)
unlink(path)
```

`nxt_search` *Full-text search across cached metadata*

Description

Runs a typeahead-suitable search against the `meta_search` FTS5 index. Every call to a metadata-caching function (e.g. `get_kpi()`, `get_products()`, `get_tables()`) populates this index as a side effect, so `nxt_search()` returns matches across all three source packages in a single query.

Usage

```
nxt_search(handle, query, sources = NULL, entity_types = NULL, limit = 50L)
```

Arguments

`handle`	An `nxt_handle` from `nxt_open()`.
`query`	FTS5 query string. Supports prefix match via `term*`, phrase match via `"exact phrase"`, and boolean operators `AND`/`OR`.
`sources`	Optional character vector restricting results to specific sources (e.g. `c("kolada", "pixieweb")`).
`entity_types`	Optional character vector restricting results to specific entity types (e.g. `c("kpi", "tables")`).
`limit`	Maximum number of results to return. Default 50.

Value

A tibble with columns `source`, `entity_type`, `entity_id`, `title`, `description`, and `rank` (lower = better match). Empty tibble if nothing matches.

Examples

```
path <- tempfile(fileext = ".sqlite")
handle <- nxt_open(path)

# Populate the search index by storing some metadata
ch <- nxt_cache_handler(
  source = "kolada", entity = "kpi", cache = TRUE,
  cache_location = handle, kind = "metadata",
  key_params = list()
)
ch("store", data.frame(
  id = c("N03700", "N01951"),
  title = c("Befolkning totalt", "Bruttoregionprodukt"),
  description = c("Antal invanare", "BRP per capita")
))

# Typeahead search
nxt_search(handle, "bef*")
```

```
# Filter by source
nxt_search(handle, "bef*", sources = "kolada")

nxt_close(handle)
unlink(path)
```

`nxt_write_async` *Enqueue an asynchronous cache write*

Description

Buffers a write for a specific `(source, entity, key_params)` query and returns immediately. The actual UPSERT into SQLite happens either in a background `mirai` task or synchronously when the package isn't available.

Usage

```
nxt_write_async(
  handle,
  source,
  entity,
  key_params,
  df,
  normalize_extra = list()
)
```

Arguments

`handle`	An `nxt_handle` from `nxt_open()`.
`source`, `entity`, `key_params`	Identity of the query (same as `nxt_cache_handler()`).
`df`	The tibble to store.
`normalize_extra`	Named list of extra args for the normalizer.

Value

`invisible(handle)`, for pipe-friendliness.

Examples

```
path <- tempfile(fileext = ".sqlite")
handle <- nxt_open(path)

df <- data.frame(
  kpi = "N03700", municipality_id = "0180",
  year = 2024L, value = 103.2
)
suppressWarnings(
  nxt_write_async(handle, "kolada", "values",
                  key_params = list(kpi = "N03700"), df = df)
)
nxt_flush(handle)

nxt_close(handle)
unlink(path)
```

Index