

Package: qualtdict (via r-universe)

October 5, 2026

Title Generate Variable Dictionaries and Labelled Survey Data for
'Qualtrics' Surveys

Version 0.1.0

Description Generates Variable Dictionaries from 'Qualtrics'
<<https://www.qualtrics.com/about/>> survey metadata and applies
those dictionaries to create Labelled Survey Data. This package
is a metadata and labelled-export companion to 'qualtRics',
which provides access to 'Qualtrics' survey data and metadata
via the 'Qualtrics' API <<https://api.qualtrics.com/>>.

License MIT + file LICENSE

URL <https://github.com/lyh970817/qualtdict>,
<https://lyh970817.github.io/qualtdict/>

BugReports <https://github.com/lyh970817/qualtdict/issues>

Imports dplyr, glue, haven, purrr, qualtrics, rlang, sjlabelled,
stringr, tibble, xml2

Suggests covr, knitr, openNLP, rmarkdown, slowraker (>= 0.1.1),
SnowballC, stringi, testthat (>= 3.0.0), tidyr

SystemRequirements Java is needed only for POS-tag filtering via the
suggested 'openNLP' package.

VignetteBuilder knitr

Config/testthat/edition 3

Config/testthat/start-first dict_generate, dict_validate,
fetch_labelled_survey_data

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake make default-jdk libarchive-dev libicu-dev libuv1-dev libxml2-dev libssl-
dev libx11-dev zlib1g-dev

Repository <https://ropensci-staging.r-universe.dev>

Date/Publication 2026-09-05 20:00:48 UTC

RemoteUrl <https://github.com/lyh970817/qualtdict>
RemoteRef HEAD
RemoteSha 96090048a4cb1cf355f5ec8d5bd2f12e09cc5f43

Contents

assert_dict_valid	2
dict_generate	3
dict_split_blocks	5
dict_validate	6
fetch_labelled_survey_data	7
labelled_export_findings	9
survey_split_blocks	10
Index	12

assert_dict_valid	<i>Require a Variable Dictionary that can be exported</i>
-------------------	---

Description

Error when a Variable Dictionary carries Export-blocking Validation Findings: level-label codings that make Labelled Export wrong or impossible. `assert_dict_valid()` is the gate [fetch_labelled_survey_data\(\)](#) applies before downloading responses, so a defective Variable Dictionary is reported by Response Column ID instead of aborting deep inside labelling after the download is paid for.

Usage

`assert_dict_valid(dict)`

Arguments

`dict` A Variable Dictionary returned by [dict_generate](#).

Details

Export-blocking Validation Findings are the level-label findings where label and level are not a one-to-one mapping, one label is carried by several rows, or one level is carried by several rows. A level carried by several rows is the shape that aborts Labelled Export: Qualtrics keys an export column on the choice recode, so two choices sharing a recode are exported as one column carrying both meanings.

A gapped level sequence is not Export-blocking. Gapped recodes are ordinary Qualtrics survey design and label correctly.

Levels and labels are reported as the Qualtrics survey defines them. `qualtdict` neither merges the rows nor repairs the coding, so the finding stays visible as the survey-authoring error it is. [dict_validate\(\)](#) stays total: it reports every Validation Finding and never errors, so it can be used to survey the damage.

Value

dict, invisibly, when it carries no Export-blocking Validation Findings. Otherwise an error of class `qualtdict_export_blocking_findings`, whose `findings` field holds the Export-blocking Validation Findings.

See Also

`dict_validate()` for the total, never-erroring report.

Examples

```
dict <- data.frame(
  response_column_id = c("QID1", "QID1"),
  variable_name = "q1",
  label = c("Yes", "No"),
  level = c("1", "2")
)
class(dict) <- c("qualtdict", class(dict))

assert_dict_valid(dict)

# Two choices sharing one Qualtrics recode are exported as one column.
broken <- dict
broken$level <- c("1", "1")

try(assert_dict_valid(broken))
```

dict_generate

Generate a Variable Dictionary for a survey

Description

Generate a Variable Dictionary from Qualtrics survey metadata retrieved via `qualtRics`.

Usage

```
dict_generate(
  surveyID,
  variable_name = c("question_name", "semantic_name"),
  block_pattern = NULL,
  block_sep = ".",
  semantic_name_preprocess = NULL,
  embedded_data_block_assignment = c("none", "previous", "next"),
  quiet = TRUE
)
```

Arguments

surveyID	String. Unique Qualtrics survey ID. Live metadata retrieval requires Qualtrics credentials configured for qualtrics.
variable_name	String. Source for the final variable_name column in the Variable Dictionary. Use question_name for the raw Qualtrics Question Name or semantic_name for a generated Semantic Name based on question text and response metadata. Semantic Name generation needs the optional packages slowraker , SnowballC , stringi , and tidyr ; the default question_name path never uses them.
block_pattern	Function. A function that given the name of a Survey Block, returns a Block Prefix to prepend to Semantic Names in that block. Used only when variable_name = "semantic_name". Defaults to NULL.
block_sep	String. Separator between variable names and block prefixes returned by block_pattern. Used only when variable_name = "semantic_name". Defaults to ".".
semantic_name_preprocess	Function. An optional function that receives the full post-normalisation dictionary and returns a modified dictionary for Semantic Name generation. It runs only when variable_name = "semantic_name". Temporary helper columns added by this function are not included in the returned Variable Dictionary.
embedded_data_block_assignment	String. Survey Flow adjacency policy for assigning Embedded Data Fields to Survey Blocks. Use "none" to leave Embedded Data Fields unassigned, "previous" to assign fields with an unambiguous Survey Flow location to the nearest previous Survey Block, or "next" to assign them to the nearest next Survey Block.
quiet	Boolean. If TRUE, suppress routine progress messages and progress bars. Defaults to TRUE.

Details

The returned Variable Dictionary preserves response_column_id as the downloaded Response Column ID, qid as the bare Qualtrics question identifier, row_source as the Dictionary Row Source, question_name as the raw Qualtrics naming reference, and variable_name as the final export-safe Dictionary Variable Name used by Labelled Survey Data. Question-backed rows use row_source = "question". Flat Embedded Data Fields defined by Qualtrics metadata use row_source = "embedded_data". Scoring Variables defined by survey description metadata use row_source = "scoring". Text-analysis Sidecars discovered from Response Column Map Classification use row_source = "text_analysis" and inherit parent qid, question_name, and block when a clear parent QID can be determined. Embedded Data Fields remain unassigned by Survey Block unless embedded_data_block_assignment explicitly requests Survey Flow adjacency assignment. That assignment is based on Survey Flow returned by fetch_description(), where Embedded Data nodes carry field payloads; the type-only Survey Flow returned by metadata() is not used for Embedded Data Field locations. Scoring Variables remain unassigned by Survey Block.

When variable_name = "semantic_name", the Variable Dictionary also includes semantic_name. Semantic Names are readable best-effort conveniences generated from survey text and metadata; they are not stable guarantees across package versions or survey text changes. For long text, Semantic Names select important words from ranked keywords and preserve those selected words

in the order they appear in the naming text. Semantic Name generation needs the optional packages **slowraker**, **SnowballC**, **stringi**, and **tidyr**. Java (via **openNLP**) is needed only for POS-tag filtering, which the Semantic Name path does not use.

Value

A Variable Dictionary: a qualtdict data frame. The as-downloaded survey definition (the parsed `fetch_description()` payload used for generation) is attached as the `survey_definition_raw` attribute for callers that persist it; it is NULL when the survey definition is unavailable (for example on synthetic or offline fixture paths).

Examples

```
if (interactive()) {
  survey_id <- "SV_XXXXXXXXXXXXXXXXX"

  # Create a function for \code{block_pattern}
  # that returns the first three letters of a string
  block_pattern <- function(x) {
    substring(x, 1, 3)
  }

  mydict <- dict_generate(survey_id,
    variable_name = "semantic_name",
    block_pattern = block_pattern,
    block_sep = "."
  )
}
```

dict_split_blocks	<i>Split a Variable Dictionary by Survey Block</i>
-------------------	--

Description

Create block-specific views of a Variable Dictionary without changing the return type of [dict_generate](#).

Usage

```
dict_split_blocks(dict)
```

Arguments

`dict` A Variable Dictionary returned by [dict_generate](#).

Value

A named list of Variable Dictionaries, one per Survey Block. When rows have no Survey Block assignment, the list includes `". .unassigned"`.

Examples

```
dict <- data.frame(
  response_column_id = c("QID1", "QID2"),
  variable_name = c("q1", "q2"),
  block = c("Block A", "Block B")
)
class(dict) <- c("qualtdict", class(dict))

dict_split_blocks(dict)
```

dict_validate	<i>Return Validation Findings for a Variable Dictionary</i>
---------------	---

Description

Validate Variable Dictionary consistency and return a stable validation object. Validation Findings report repaired variable_name values, non-unique or unsafe Dictionary Variable Names, and level-label coding issues. The validation result is a consistency screen, not proof that the source Qualtrics metadata is correct.

Usage

```
dict_validate(dict, quiet = TRUE)
```

Arguments

dict	A Variable Dictionary returned by dict_generate .
quiet	Boolean. If TRUE, suppress routine validation messages. Defaults to TRUE.

Details

dict_validate() is total: it reports every Validation Finding and never errors, so it can be used to survey the damage in a defective Variable Dictionary. Use [assert_dict_valid\(\)](#) when you want the Export-blocking subset to error instead.

Value

A list with stable components:

- `validation_findings` - A data frame of Validation Findings for Variable Dictionary consistency issues, including repaired variable names, non-unique or unsafe final variable_name values, and level-label coding findings. Each row carries a severity: "definite" when the finding makes the affected export column uninterpretable or its identity unreliable (the Export-blocking level-label codings, and inconsistent, duplicate, or unsafe final variable_name values), "suggestive" when the finding only reports something worth review while the column's data and identity stay sound (a repaired variable_name; a level-label finding, such as a gapped level run, none of whose tripped tests is Export-blocking).
- `level_label_pairs` - A data frame containing the unique level-label pairings.

Examples

```
dict <- data.frame(
  response_column_id = c("QID1", "QID2"),
  variable_name = c("q1", "q2"),
  label = c("Yes", "No"),
  level = c("1", "2")
)
class(dict) <- c("qualtdict", class(dict))

validation <- dict_validate(dict)
```

fetch_labelled_survey_data

Fetch Labelled Survey Data

Description

Download Qualtrics responses through qualtrics and apply a Variable Dictionary generated by [dict_generate](#). Export Variables are matched by response_column_id, renamed to variable_name, and labelled with question, item, level, and label metadata using sjlabelled.

Usage

```
fetch_labelled_survey_data(
  dict,
  extra_columns = c("externalDataReference", "startDate", "endDate"),
  exclude_findings = c("none", "definite", "validation"),
  ...,
  require_valid_dict = TRUE,
  quiet = TRUE
)
```

Arguments

dict	A Variable Dictionary returned by dict_generate .
extra_columns	A character vector of downloaded survey data columns to retain in the Labelled Survey Data in addition to Export Variables. Defaults to c("externalDataReference", "startDate", "endDate"). Columns absent from the downloaded data warn (naming the missing columns) and are skipped, so several candidate names can be supplied when only one is expected to be present (for example ordered id-column fallbacks). Use NULL to retain no extra columns (this also suppresses the response metadata columns described below).
exclude_findings	Which Validation Finding classes cost their rows after survey download. "none" keeps all downloaded variables represented in the Variable Dictionary. "definite"

excludes only the rows whose Validation Findings are Definite Validation Findings (`severity == "definite"` in `dict_validate` output): the Export-blocking level-label codings, and the variable-name findings that make the renamed export's column identity unreliable. Rows carrying only Suggestive Validation Findings (a repaired `variable_name`; a gapped level run) keep their columns and stay reported by `dict_validate()`. `"validation"` excludes every row with any Validation Finding, suggestive ones included.

... Other arguments passed to `fetch_survey`. Filtering arguments such as `include_questions` may be passed through. `qualtdict` owns `surveyID`, `import_id`, `convert`, `label`, and `breakout_sets`; passing those arguments errors because they are needed for reliable Variable Dictionary to Response Column ID matching.

`require_valid_dict` Boolean. If `TRUE` (the default), the Variable Dictionary is checked with `assert_dict_valid` before any responses are downloaded, and Export-blocking Validation Findings error. Use `FALSE` to download from a survey known to carry them; the affected Export Variables then keep unreliable value labels. Because `require_valid_dict` follows ..., supply it by name.

`quiet` Boolean. If `TRUE`, suppress routine validation messages. Defaults to `TRUE`.

Details

Live calls require Qualtrics credentials configured for `qualtrics` because response data are downloaded from the Qualtrics API. `qualtdict` owns the `fetch_survey()` settings needed for reliable Variable Dictionary to Response Column ID matching: `import_id = TRUE`, `label = FALSE`, `convert = FALSE`, and `breakout_sets = TRUE`. User-supplied passthrough arguments cannot override those settings.

By default, Labelled Survey Data includes variables with Validation Findings. Use `exclude_findings` when you want to explicitly remove them after download. Because `quiet` follows ..., supply it by name.

Export-blocking Validation Findings are checked before the download, so a defective Variable Dictionary costs no responses. The check is skipped when `exclude_findings` is `"definite"` or `"validation"`, because every Export-blocking Response Column ID is then dropped from the Variable Dictionary after download anyway (every Export-blocking Validation Finding is a Definite Validation Finding). The pre-download check never aborts on Suggestive Validation Findings under any setting.

Unless `extra_columns = NULL`, Labelled Survey Data additionally carries the per-response metadata columns `startDate`, `endDate`, `recordedDate`, `ResponseId`, and `Progress`. These are emitted under exactly those names regardless of the source export casing: `import_id = TRUE` downloads them as `startDate`, `endDate`, `recordedDate`, `_recordId`, and `progress`, and each is normalised to its canonical name. Inclusion is best effort: any metadata column absent from the download (for example in sanitized offline fixtures) is omitted without warning rather than erroring.

Value

Labelled Survey Data: a data frame whose Export Variables are renamed to `variable_name` and annotated with labels from the Variable Dictionary.

Examples

```

if (interactive()) {
  survey_id <- "SV_XXXXXXXXXXXXXXX"

  # Generate a dictionary
  mydict <- dict_generate(survey_id, variable_name = "question_name")
  survey_dat <- fetch_labelled_survey_data(mydict,
    unanswer_recode = -77,
    unanswer_recode_multi = 0
  )
}

```

labelled_export_findings

Return Labelled Export Findings

Description

Labelled Export Findings describe issues detected while matching a Variable Dictionary to downloaded survey data.

Usage

```
labelled_export_findings(x)
```

Arguments

x Labelled Survey Data returned by [fetch_labelled_survey_data\(\)](#).

Value

A data frame of Labelled Export Findings.

Examples

```

dat <- data.frame(q1 = "1")
attr(dat, "labelled_export_findings") <- data.frame(
  finding = "missing_response_column_id",
  response_column_id = "QID2",
  qid = "QID2",
  variable_name = "q2",
  reason = "not_found_in_downloaded_survey_data"
)

labelled_export_findings(dat)

```

survey_split_blocks	<i>Split Labelled Survey Data by Survey Block</i>
---------------------	---

Description

Create block-specific views of Labelled Survey Data without changing the return type of [fetch_labelled_survey_data](#).

Usage

```
survey_split_blocks(
  dat,
  dict = attr(dat, "dict", exact = TRUE),
  extra_columns = c("externalDataReference", "startDate", "endDate")
)
```

Arguments

dat	Labelled Survey Data returned by fetch_labelled_survey_data .
dict	A Variable Dictionary returned by dict_generate .
extra_columns	A character vector of raw Labelled Survey Data columns to retain in each block data set. Defaults to <code>c("externalDataReference", "startDate", "endDate")</code> . Columns absent from the data warn (naming the missing columns) and are skipped. Use NULL to retain no extra columns.

Details

When dict is not supplied, `survey_split_blocks()` uses the dict attribute attached by [fetch_labelled_survey_data](#).

Value

A named list of data frames, one per Survey Block. When dictionary rows have no Survey Block assignment, the list includes `"..unassigned"`.

Examples

```
dict <- data.frame(
  response_column_id = c("QID1", "QID2"),
  variable_name = c("q1", "q2"),
  block = c("Block A", "Block B")
)
class(dict) <- c("qualtdict", class(dict))

dat <- data.frame(
  externalDataReference = "R_1",
  startDate = "2026-06-01",
  endDate = "2026-06-01",
  q1 = "1",
  q2 = "2"
```

```
)  
attr(dat, "dict") <- dict  
  
survey_split_blocks(dat)
```

Index

`assert_dict_valid`, [2](#), [8](#)
`assert_dict_valid()`, [6](#)

`dict_generate`, [2](#), [3](#), [5–7](#), [10](#)
`dict_split_blocks`, [5](#)
`dict_validate`, [6](#), [8](#)
`dict_validate()`, [2](#), [3](#)

`fetch_labelled_survey_data`, [7](#), [10](#)
`fetch_labelled_survey_data()`, [2](#), [9](#)
`fetch_survey`, [8](#)

`labelled_export_findings`, [9](#)

`survey_split_blocks`, [10](#)