def-sqlvec-fns

HugSQL generates a format internally known as **sqlvec**. The sqlvec format is a vector with an SQL string in the first position containing any `?` placeholders, followed by any number of parameter values to be applied to the SQL in positional order. For example:

```
["select * from characters where id = ?", 2]
```

The sqlvec format is a convention used by `clojure.java.jdbc`, `clojure.jdbc`, and `next.jdbc` for value parameter replacement. Because of the underlying support in these libraries and the JDBC-driver-specific issues for data type handling, HugSQL also uses the sqlvec format by default for value parameters.

HugSQL provides `hugsql.core/def-sqlvec-fns` to create functions returning the sqlvec format. The created functions have an -sqlvec suffix by default, though this is configurable with the `:fn-suffix` option. These functions are helpful during development/debugging and for the purpose of using the parameter-replacing functionality of HugSQL without using the built-in adapter database functions to execute queries.

`hugsql.core/def-sqlvec-fns` doc:

```
=> (doc hugsql.core/def-sqlvec-fns)
-------------------------
hugsql.core/def-sqlvec-fns
([file] [file options])
Macro
  Given a HugSQL SQL file, define the <name>-sqlvec functions in the
  current namespace.  Returns sqlvec format: a vector of SQL and
  parameter values. (e.g., ["select * from test where id = ?" 42])

  Usage:

   (def-sqlvec-fns file options?)
```

```
where:
 - file is a string file path in your classpath,
   a resource object (java.net.URL),
   or a file object (java.io.File)
 - options (optional) hashmap:
   {:quoting :off(default) | :ansi | :mysql | :mssql
    :fn-suffix "-sqlvec" (default)

:quoting options for identifiers are:
  :ansi double-quotes: "identifier"
  :mysql backticks: `identifier`
  :mssql square brackets: [identifier]
  :off no quoting (default)

Identifiers containing a period/dot . are split, quoted separately,
and then rejoined. This supports myschema.mytable conventions.

:quoting can be overridden as an option in the calls to functions
created by def-db-fns.

:fn-suffix is appended to the defined function names to
differentiate them from the functions defined by def-db-fns.
```

Edit this page