U1WAUKQ3E : Hi! I wrote an article about PostgreSQL to Datomic migration. Hope you'll find it useful.<http://grishaev.me/en/pg-to-datomic>

U5ZAJ15P0 : igrishaev: thank you for writing this up; very useful!
U5ZAJ15P0 : there is a <#C03RZMDSH|datomic> channel, you might want to post it there too
U1WAUKQ3E : thank you, makes sense
U0CKDHF4L : does the describe of a spec always have a similar structure to the data it specifies ? Can you give a counterexample ?
U5ZAJ15P0 : <@U0CKDHF4L> I am not overly familiar with clojure specs, but map specs are represented as vectors I think
U5ZAJ15P0 : well, as a map with a `:req` vector
U0CKDHF4L : yes, ```(keys :req etc)``` -- but that has basically the same nesting structure
U0CKDHF4L : I'm trying to think of a case where you describe a nested structure of collections by a flat spec
U0GC1C09L : does clojure have a representation of infinity that can work with mathematical operators? something like `(&lt;= 1 2 Infinity)` ?
U0CKDHF4L : ```Double/POSITIVE_INFINITY```
U0CKDHF4L : or ```Number.POSITIVE_INFINITY``` in CLJS
U0GC1C09L : !! thanks :slightly_smiling_face:
U1NGX4Z6F : hey guys
U1NGX4Z6F : what's you're preferred method of checking for nils before assignation ?
U060FKQPN : <@U0GC1C09L> see also <https://dev.clojure.org/jira/browse/CLJ-1074>
U1NGX4Z6F : i just realized I am repeating `(:username respone)` when I `(if-not (nil? (:username response) (:username response) fallback-value))`
U1NGX4Z6F : isn't there a syntactic sugar around this ?
U0CKDHF4L : ```(or maybe-nil-thing default-value)```
U060FKQPN : assuming you also want to exclude `false`
U0CKDHF4L : (assuming that)