U3QUAHZJ6 : hello everyone, how im supposed to create a spec where all keys are optional but at least one of the specified keys should be present?
```
(s/def ::my-spec (s/and (help-plz??)(s/keys :opt-un [::a ::b])))
(s/valid? ::my-spec {} =&gt; false
(s/valid? ::my-spec {:a 1}) =&gt; true
(s/valid? ::my-spec {:b 1}) =&gt; true
(s/valid? ::my-spec {:a 1 :b 1}) =&gt; true
(s/valid? ::my-spec {:A1 :B 1}) =&gt; true
```

U050MP39D : <@U3QUAHZJ6> (some-fn :a :b) not good enough?
U3QUAHZJ6 : some-fn is actually a function? i presumed i was supposed to fill in the gaps >.< feel so dumb right now
U050MP39D : oh I'm sorry yeah I could see how you'd read it that way. but no, some-fn is a higher order function that takes a list of predicate functions and returns a single predicate function that is the logical or of all of them
U3QUAHZJ6 : makes sense now!
U050MP39D : you could also do `#(some % [:a :b])`
U050MP39D : ^ relies on maps acting as functions
U050SC7SV : You can use :req-un [(or ::foo ::bar)]
U050MP39D : I totally forgot about that syntax
U050SC7SV : It's one thing I love in spec
U08QZ7Y5S : Say, is there a form of `(cond)` that will bind the result value of the predicate expressions? Like```
(cond
  (my-pred x)
  (do-something (my-pred x)))
```
Where `(my-pred x)` returns either `nil` or a value I want to use?

U08QZ7Y5S : ...but I'd like something like:```
(cond-let
  ([res (my-pred x)]
   (do-something res)))
```

U11BV7MTK : <https://github.com/clojure-emacs/cljs-tooling/blob/master/src/cljs_tooling/util/misc.clj#L7>
U11BV7MTK : it's a `cond-let` macro
U08QZ7Y5S : Thanks <@U11BV7MTK>, I'll check that out. Nothing built-in I'm missing then, I take it?
U11BV7MTK : not as far as i know
U08QZ7Y5S : Cool, thanks. Looks like that macro exists verbatim in a lot of libraries... <https://crossclj.info/clojure/cond-let.html>
U11BV7MTK : the exact same? I guess its just one of those clojure archetypes
U0QNQ3P3L : I am curious as to where people are storing simple jdbc queries? I have a set of ~8 or so queries, all plain queries without any parameters. Do you use a "config" type file?
U065JNAN8 : Stick 'em in the resources directory of your project then you can retrieve them by slurping the return value of `(<http://clojure.java.io/resource|clojure.java.io/resource> "foo.sql")`
U0QNQ3P3L : <@U065JNAN8> - yes, certainly in the resources directory. I was just wondering if there was a preferred file format but a .sql file makes the intent of the file very apparent, which is good.