U5WAJK60M : Hey guys, trying to setup a connection pool with c3p0. It seems to have trouble connecting to the production database using ssl (heroku, needs sslmode=require). I've tried```
(.setJdbcUrl my-datasource (format "jdbc:%s:%s?sslmode=require" (:subprotocol dbspec) (:subname dbspec)))
```
and
```
(.setProperties my-datasource
                (doto (java.util.Properties.)
                  [...]
                  (.setProperty "sslmode" "require")))
```

C3p0 doesn't seem to pick this up, I keep getting `SQLException: Connections could not be acquired from the underlying database!`. Getting a connection for the db-url string with the sslmode query-params does indeed work. Has anyone else experienced this?

U5WAJK60M : where `my-datasource` an instance is of `om.mchange.v2.c3p0.ComboPooledDataSource`
U5WAJK60M : Update: just tried with apache's `DBCP2`, and that seems to pick the sslmode just fine. Is there an issue with c3p0 I'm not aware of?
U0524F6MV : You might also try HikariCP for pooling, it's rock solid. <https://github.com/brettwooldridge/HikariCP>
U5WAJK60M : Thanks <@U0524F6MV>. Already looked at it, looks great indeed :slightly_smiling_face:
U28TJ0DDZ : does any one knows how should i use fast-resource or even resource to server some static files on pedestal ? how should the route be like ?
U06MTLC5R : Hey fellow Clojurians, I'm looking for a lein plugin that can deploy binaries directly to github releases (similar to <https://github.com/aktau/github-release>) any ideas?
U06MTLC5R : Done see any at <https://github.com/technomancy/leiningen/wiki/Plugins>
U06MTLC5R : Dont*
U051SS2EU : seems like something you could do with `lein shell` if it doesn't need to be cross platform <https://github.com/hyPiRion/lein-shell>
U0C3SLTHP : guys, I'm struggling to make my own flat tail call, can you guys give me some enlightenment ?
```
(defn flat
  [[h &amp; t :as list]]
  (cond
    (empty? list) nil
    (sequential? h) (concat (flat h) (flat t))
    :else (cons h (flat t))))
```

U051SS2EU : are you aware that clojure never optimizes tail calls?
U051SS2EU : generally with list processing, you don't want to use recur (which acts like an optimized tail self call) but instead make a lazy-seq, which works with your current code if you wrap the else `(lazy-seq (cons h (flat t)))`
U0C3SLTHP : &gt; are you aware that clojure never optimizes tail calls?yeah, but the loop-recur ( looks like uses go-to ) can help me with that, isn't ?

U051SS2EU : you could use recur, but that tends to be clumsy for sequences
U0C3SLTHP : yep
U051SS2EU : especially with a function that has a tree call structure
U0C3SLTHP : what you mean by tree call structure ?
U051SS2EU : every call leads to 0 or more self-calls
U051SS2EU : as opposed to 0 or 1, which is linear, 0 or more means you end up with a tree of calls
U051SS2EU : and a linear series of calls is neccesary for tail call -you can't have two tail calls
U051SS2EU : you can force it by adding a state accumulator which makes the code more complex and moves data out of the stack and into the heap
U0C3SLTHP : I see
U0C3SLTHP : yeah
U0C3SLTHP : correct me if I'm wrong, but all tail call function they have some kind of accumulator, isn't ?
U051SS2EU : not always - but it's very common
U051SS2EU : actually I think a tail call function with no accumulator would be pretty weird
U0C3SLTHP : i can't see
```

U051SS2EU : but I could see it for eg. something that repeatedly accesses a resource and eventually returns a result
U0C3SLTHP : yeah
U0C3SLTHP : clojure is not lazy by default, like haskell right, how the lazy works on clojure , `(lazy-seq (cons h (flat t)))`
U051SS2EU : right - but many functions are lazy
U5Z4ECHCM : Lazy is only there when you don't want it to be &gt;.&gt;
U051SS2EU : including concat
U5Z4ECHCM : &gt; trying to print something&gt; LAZY SEQ HELLO

U051SS2EU : that's only if you call str - just printing won't do that
U051SS2EU : ```+user=&gt; (str (map inc (range 10)))"clojure.lang.LazySeq@c5d38b66"
+user=&gt; (println (map inc (range 10)))
(1 2 3 4 5 6 7 8 9 10)
nil
```

U051SS2EU : and if you need to build up a string, `pr-str` will help ```user=&gt; (str "fixed: " (pr-str (map inc (range 10))))"fixed: (1 2 3 4 5 6 7 8 9 10)"
```

U5Z4ECHCM : Well how-about-that
U5Z4ECHCM : learn something every day
U5YHX0TQV : yada seem to have implemented something themselves <https://github.com/juxt/yada/blob/master/ext/oauth2/src/yada/oauth.clj>. Maybe its time we see a new repository appearing under your github account :wink:
U2PGHFU5U : Does anyone know how to store state in one instance of a simulation in `clj-gatling`?
/edit Answer from the docs:

<http://i.imgur.com/SISAxzT.png>

U2PGHFU5U : Hmm looks like the `context` is passed along in every step. Hopefully I can just assoc.
U2PGHFU5U : That's not it. `assoc`ing to that context is not persistent
U2PGHFU5U : over steps
U2PGHFU5U : One solution is to keep a separate database, but it is not very clean