U050KSS8M : <@U11BV7MTK> yes that's correct
U050KSS8M : see my follow up
U11BV7MTK : ah ok. i thought you were still attributing it to the `:keys` destructuring
U11BV7MTK : my bad
U0564EGNY : <@U3L6TFEJF> - I ended up using postwalk with core.match
U3L6TFEJF : <@U0564EGNY> nice! mind sharing the solution?
U0564EGNY : I suppose the map should be mapv
U0564EGNY : hmm, not sure how this is actually working without that :wink:
U0564EGNY : oh, and the outer seq needs to be there because our tree may have multiple paths, duh
U0564EGNY : ok, I have a bug when adding another path in the tree. I'm flattening at the wrong place
U3L6TFEJF : here's my attempt with Specter:```
(def TreeValues
  (s/recursive-path [] p
    (s/if-path map?
      [(s/collect-one s/FIRST s/FIRST) s/MAP-VALS p]
      s/STAY)))
```

U3L6TFEJF : ```user =&gt; (s/select TreeValues m)
[[:a :b :c 1] [:a :b 2]]
user =&gt; (s/select TreeValues {:a {:fields {:b {:options [{:label "foo" :value "bar"}]}}}})
[[:a :fields :b :options [{:label "foo" :value "bar"}]]]
```

U3L6TFEJF : I haven't figured out yet how to not include the leaf itself
U0564EGNY : map over the results with `butlast`
U3L6TFEJF : just noticed that mine is not correct either, I get `[:a :b 2]` instead of `[:a :d 2]`
U0J9LVB6G : <@U0564EGNY> Here's one implementation.
```
(defn all-paths [m]
  (letfn [(step [acc current-path x]
            (reduce-kv (fn [acc k v]
                         (let [path (conj current-path k)]
                           (if (map? v)
                             (step acc path v)
                             (conj acc path))))
                       acc, x))]
    (step [] [] m)))

(all-paths {:a {:b {:c 1} :d 2}}) ;=&gt; [[:a :b :c] [:a :d]]
```

U0564EGNY : thanks, I'll try that on my data set
U071CG4QY : <https://stackoverflow.com/questions/21768802/how-can-i-get-the-nested-keys-of-a-map-in-clojure>
U3L6TFEJF : sweet, there you have a working version of what I was trying to do:```(def TreeValues
  (s/recursive-path [] p
    (s/if-path map?
       [s/ALL (s/collect-one s/FIRST) s/LAST p]
       s/STAY)))

user=&gt; (map butlast (s/select TreeValues {:a {:b {:c 1} :d 2}}))
((:a :b :c) (:a :d))
```

U3JURM9B6 : what is the best data format for exchanging data between clojure and python?
U064X3EF3 : depends on what your needs are, but you should look at <https://github.com/cognitect/transit-format>
```