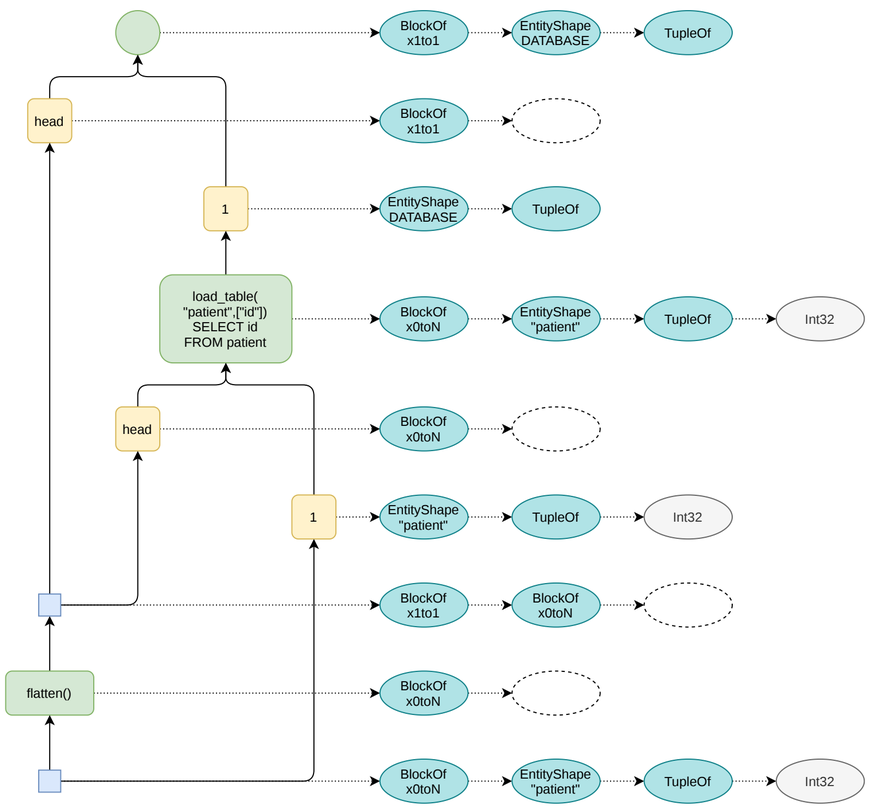
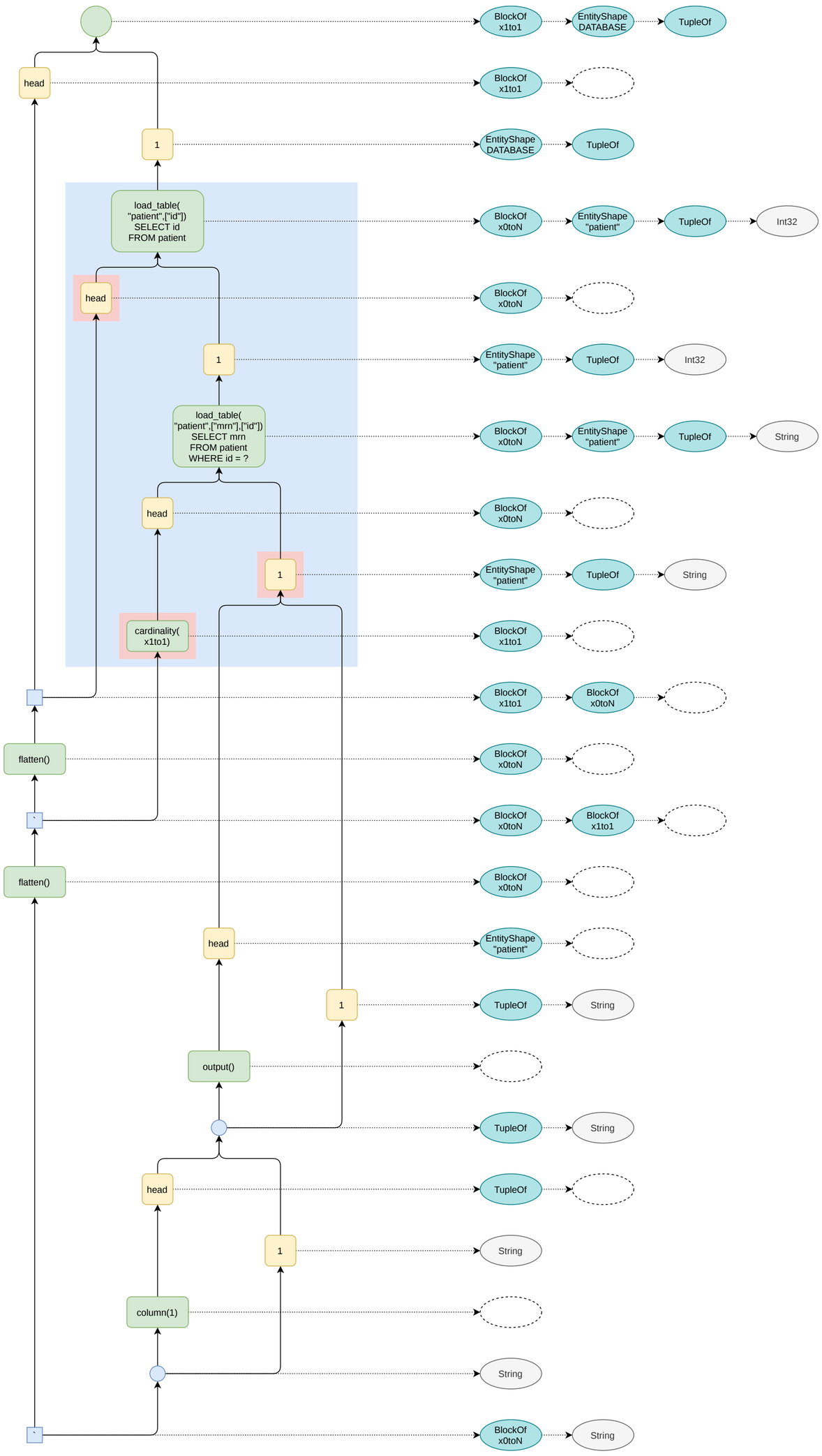
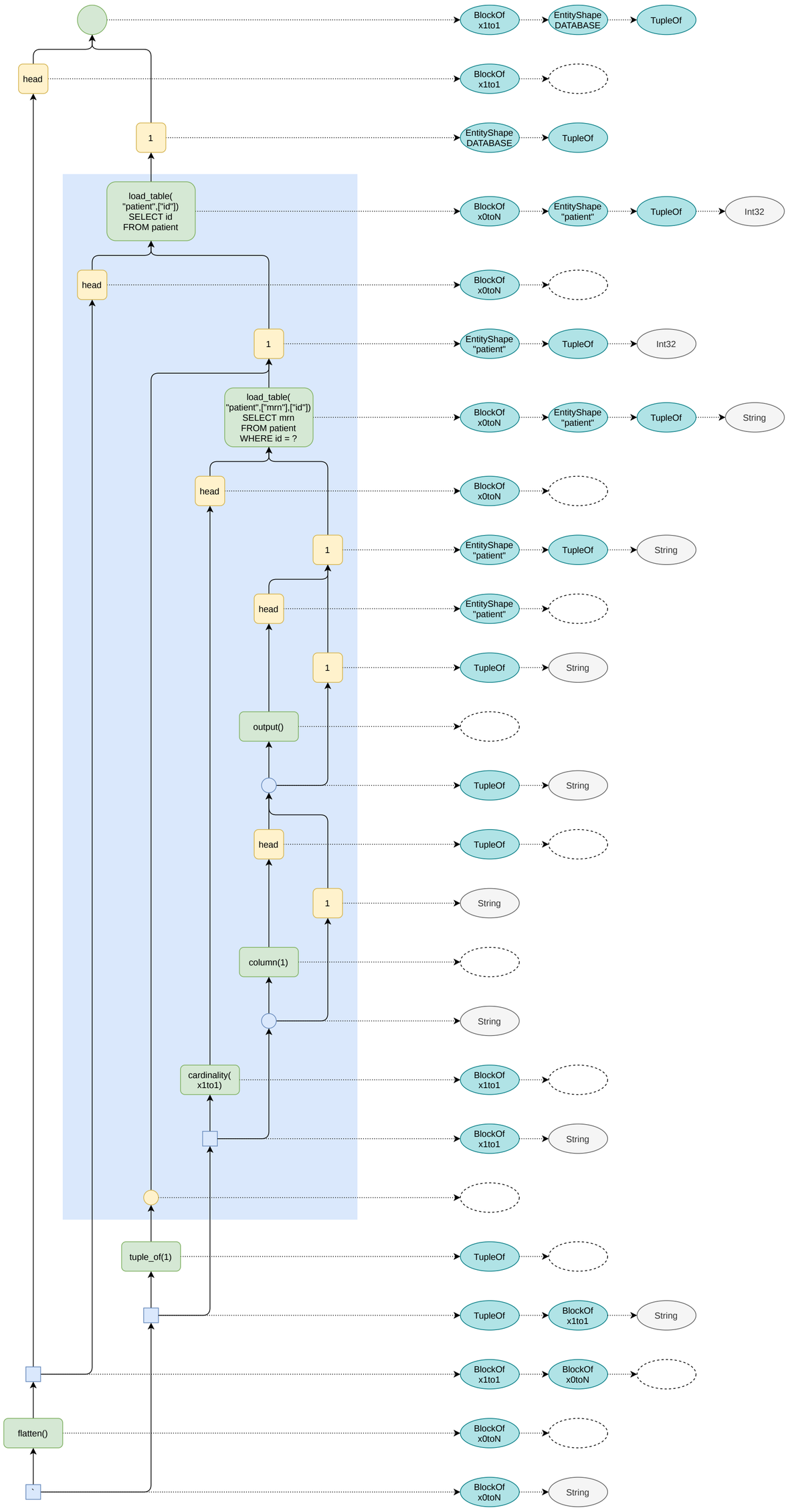
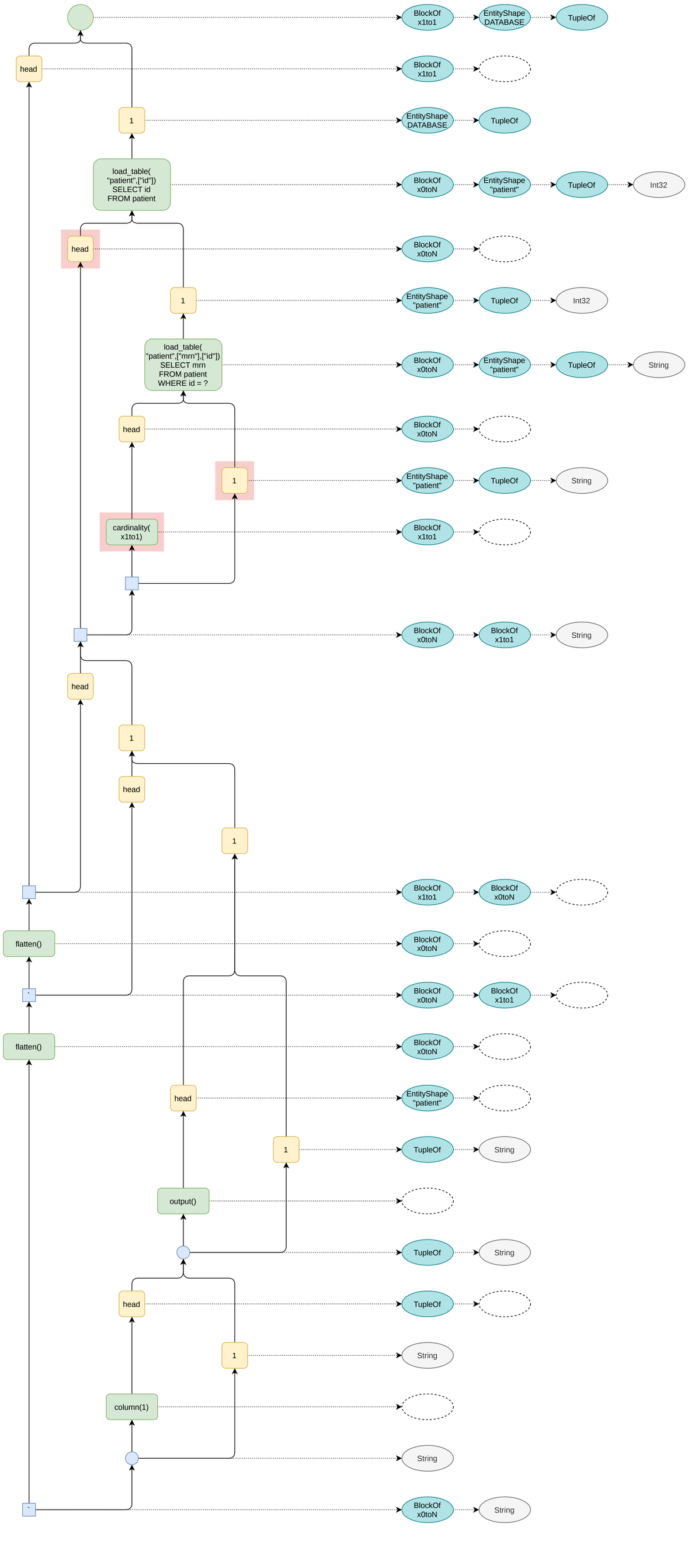
```
chain_of(
    with_elements(load_postgres_table(("public", "patient"), ["id"], [Int32])),
    flatten())
```

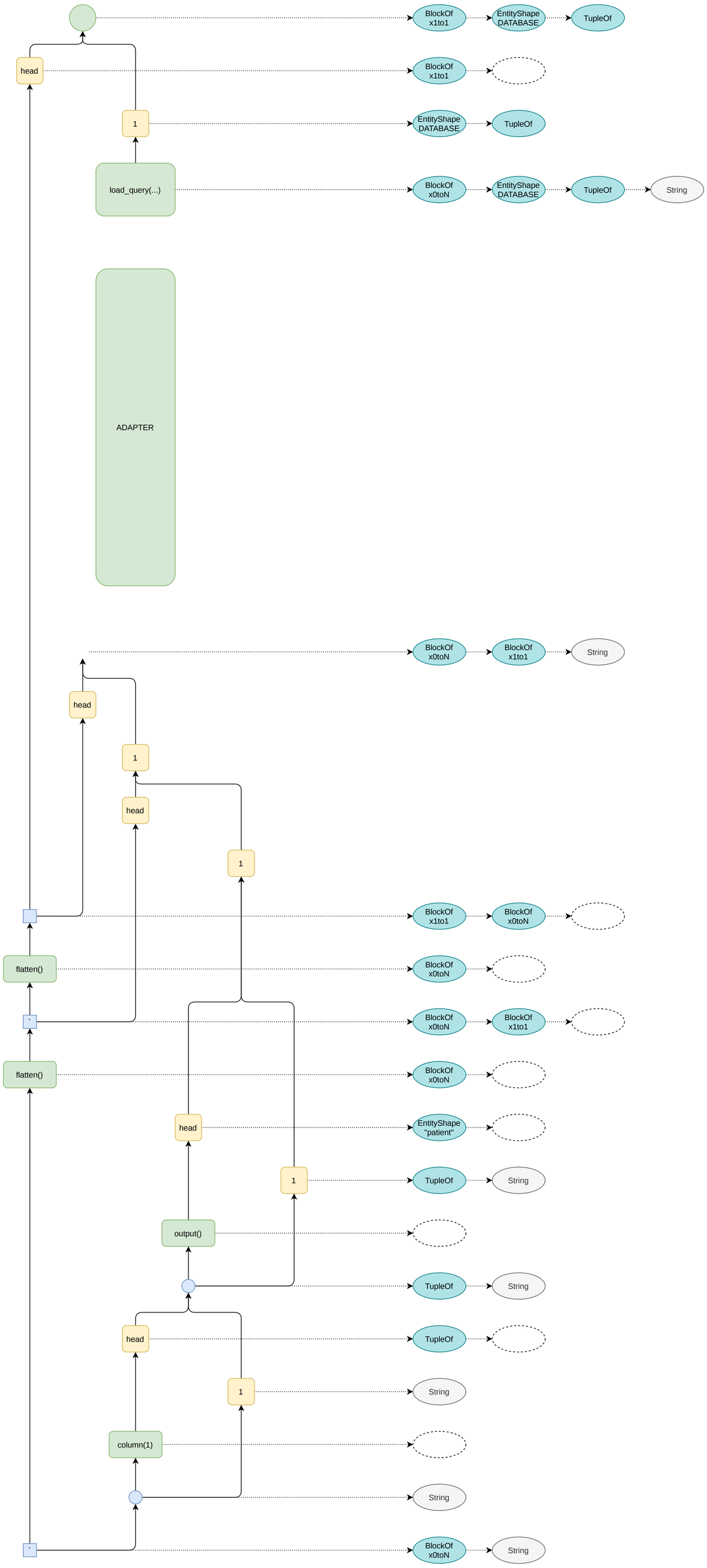


```
chain_of(with_elements(load_postgres_table(("public", "patient"), ["id"], [Int32])),
         flatten(),
         with_elements(
             chain_of(
                 load_postgres_table(("public", "patient"), ["mrn"], [String], ["id"]),
                 block_cardinality(x1to1))),
         flatten(),
         with_elements(
             chain_of(
                 output(),
                 column(1))))
```

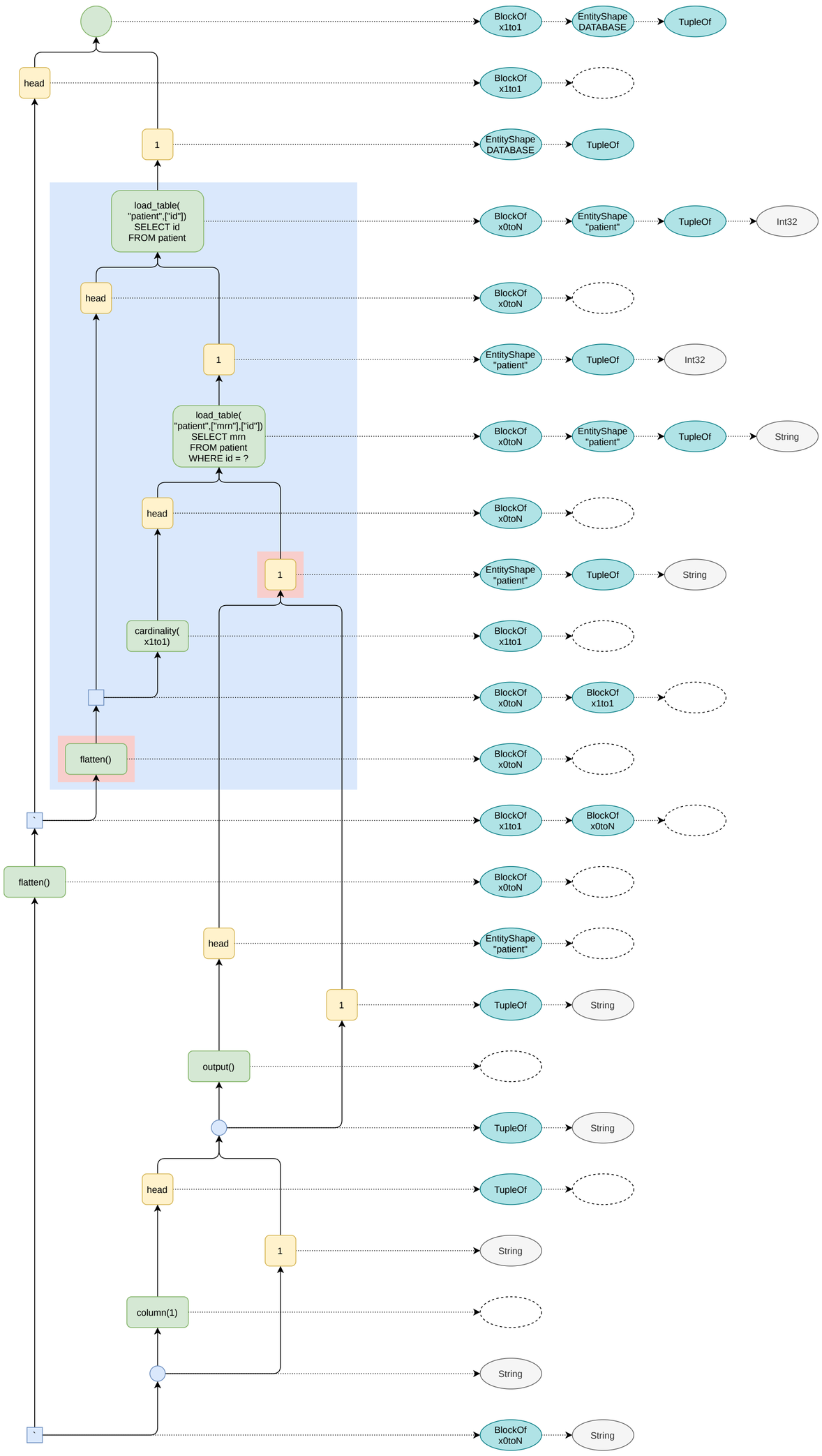


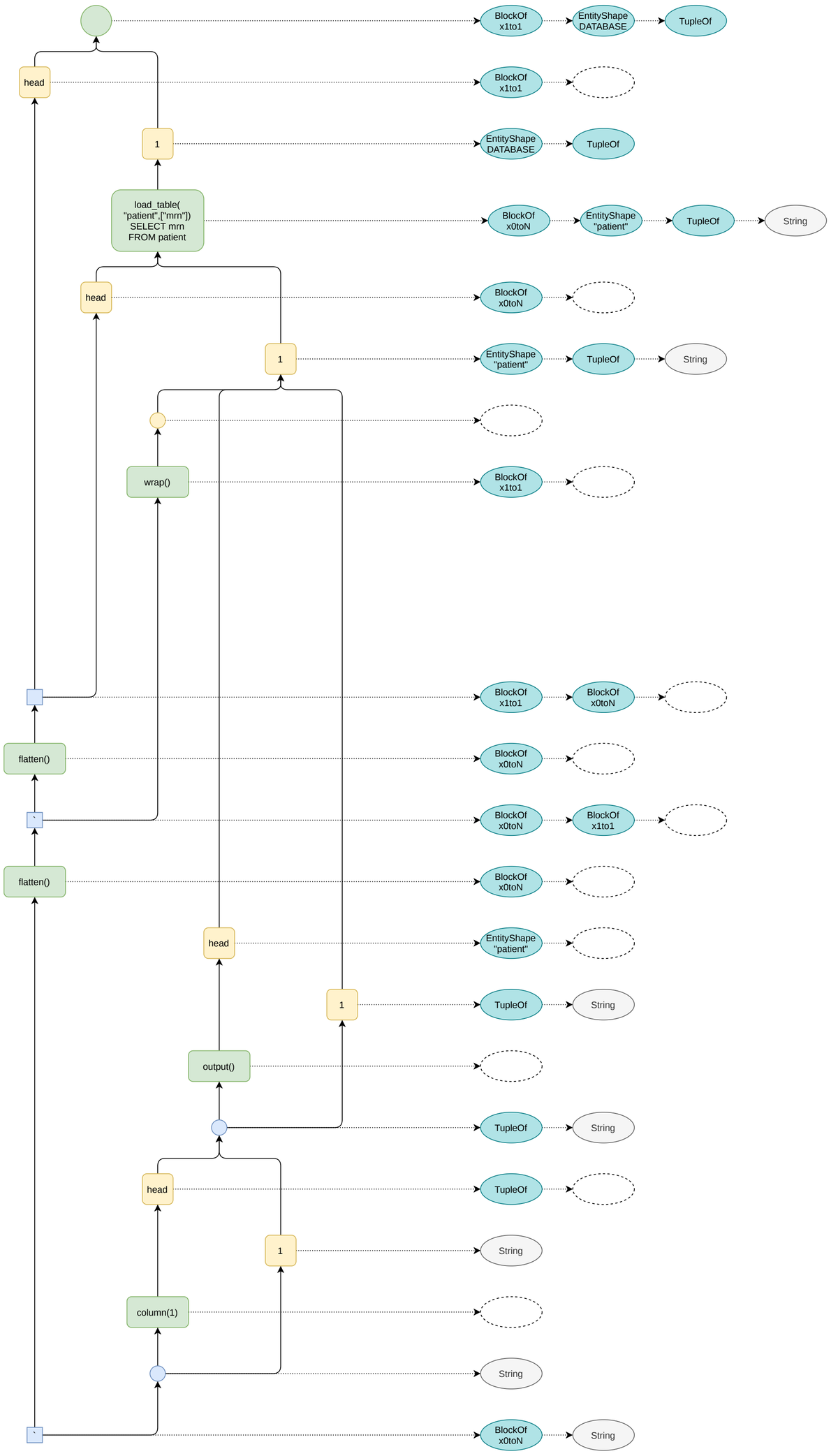

BlockOf x1to1
EntityShape DATABASE
TupleOf
head
BlockOf x1to1
1
EntityShape DATABASE
TupleOf
load_table("patient",["id"]) SELECT id FROM patient
BlockOf x0toN
EntityShape "patient"
TupleOf
Int32
head
BlockOf x0toN
1
EntityShape "patient"
TupleOf
Int32
load_table("patient",["mrn"],["id"]) SELECT mrn FROM patient WHERE id = ?
BlockOf x0toN
EntityShape "patient"
TupleOf
String
head
BlockOf x0toN
1
EntityShape "patient"
TupleOf
String
head
EntityShape "patient"
1
TupleOf
String
output()
TupleOf
String
head
TupleOf
1
String
column(1)
String
cardinality(x1to1)
BlockOf x1to1
BlockOf x1to1
String
tuple_of(1)
TupleOf
TupleOf
BlockOf x1to1
String
BlockOf x1to1
BlockOf x0toN
flatten()
BlockOf x0toN
`
BlockOf x0toN
String

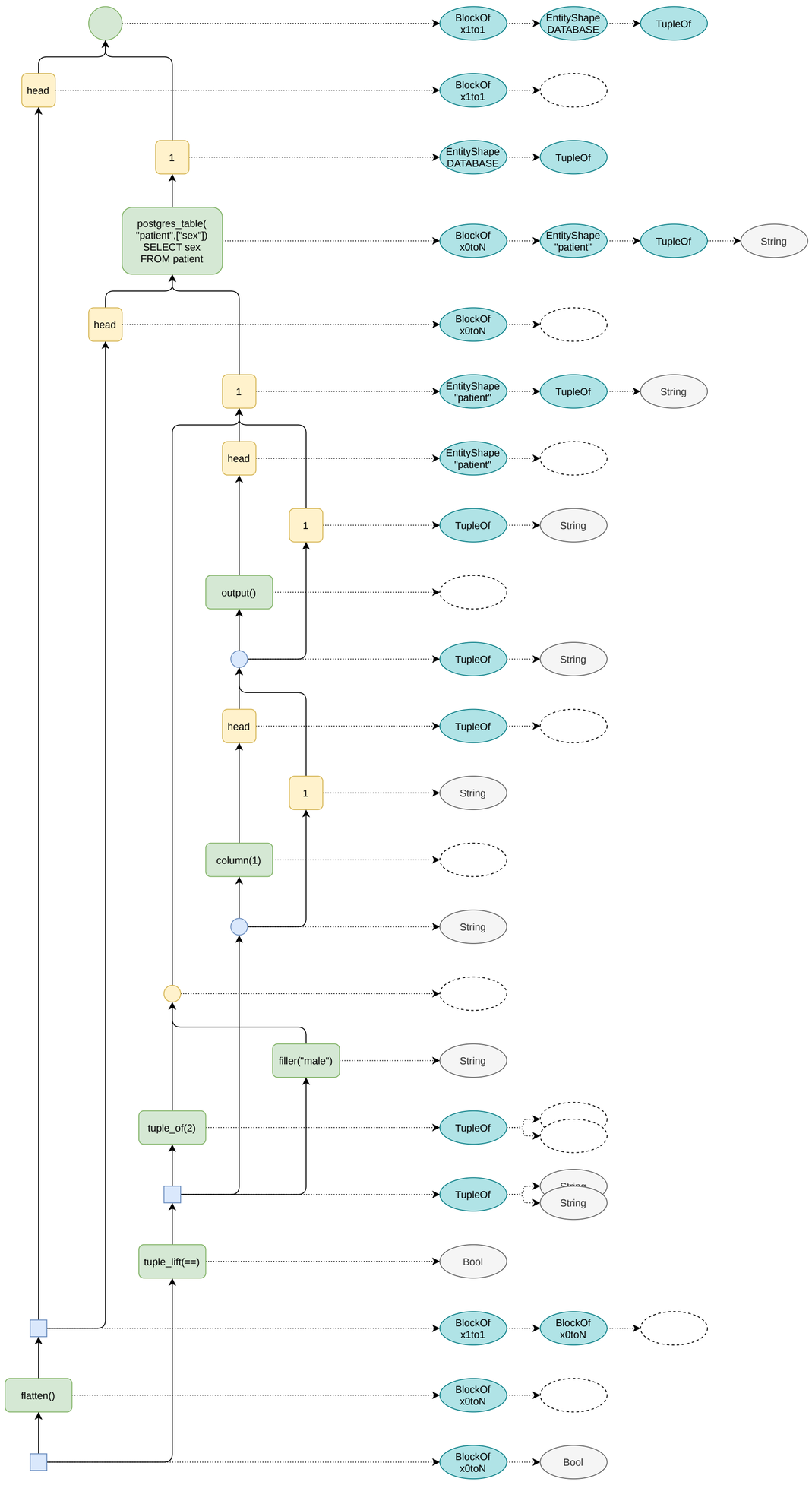

BlockOf x1to1
EntityShape DATABASE
TupleOf
head
BlockOf x1to1
1
EntityShape DATABASE
TupleOf
load_table("patient",["id"]) SELECT id FROM patient
BlockOf x0toN
EntityShape "patient"
TupleOf
Int32
head
BlockOf x0toN
1
EntityShape "patient"
TupleOf
Int32
load_table("patient",["mrn"],["id"]) SELECT mrn FROM patient WHERE id = ?
BlockOf x0toN
EntityShape "patient"
TupleOf
String
head
BlockOf x0toN
1
EntityShape "patient"
TupleOf
String
cardinality(x1to1)
BlockOf x1to1
BlockOf x0toN
BlockOf x0toN
String
head
1
head
1
BlockOf x1to1
BlockOf x0toN
flatten()
BlockOf x0toN
`
BlockOf x0toN
BlockOf x1to1
flatten()
BlockOf x0toN
head
EntityShape "patient"
1
TupleOf
String
output()
TupleOf
String
head
TupleOf
1
String
column(1)
String
`
BlockOf x0toN
String

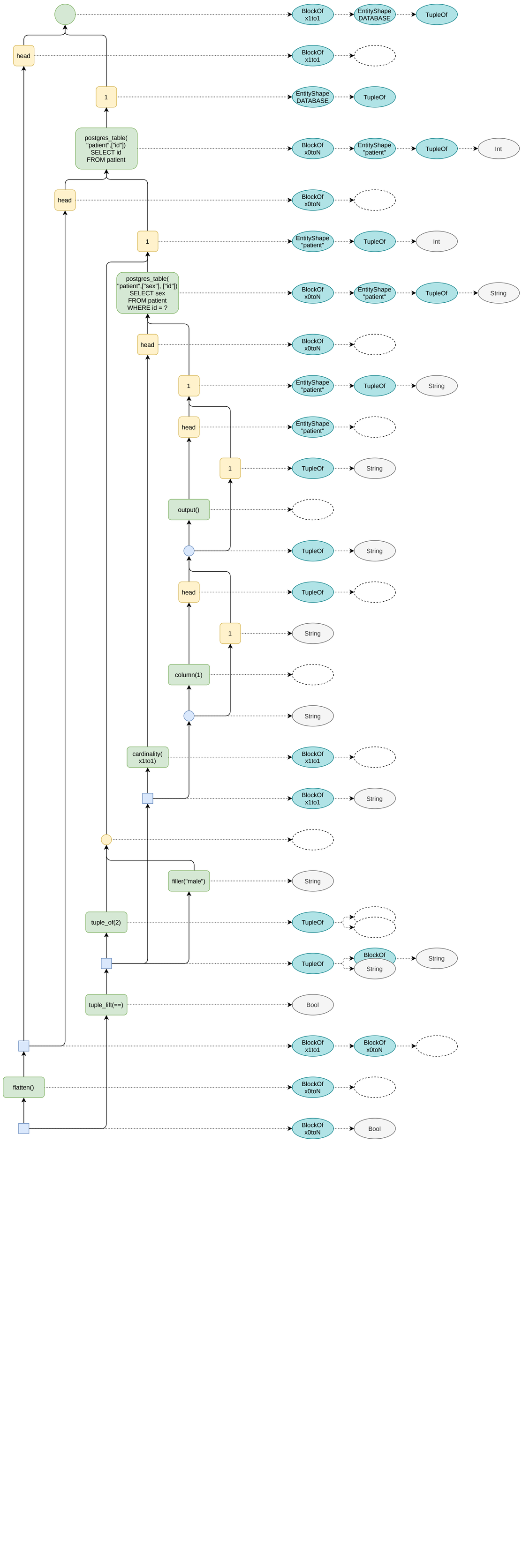

BlockOf x1to1
EntityShape DATABASE
TupleOf
head
BlockOf x1to1
1
EntityShape DATABASE
TupleOf
load_query(...)
BlockOf x0toN
EntityShape DATABASE
TupleOf
String
ADAPTER
BlockOf x0toN
BlockOf x1to1
String
head
1
head
1
BlockOf x1to1
BlockOf x0toN
flatten()
BlockOf x0toN
`
BlockOf x0toN
BlockOf x1to1
flatten()
BlockOf x0toN
head
EntityShape "patient"
1
TupleOf
String
output()
TupleOf
String
head
TupleOf
1
String
column(1)
String
`
BlockOf x0toN
String

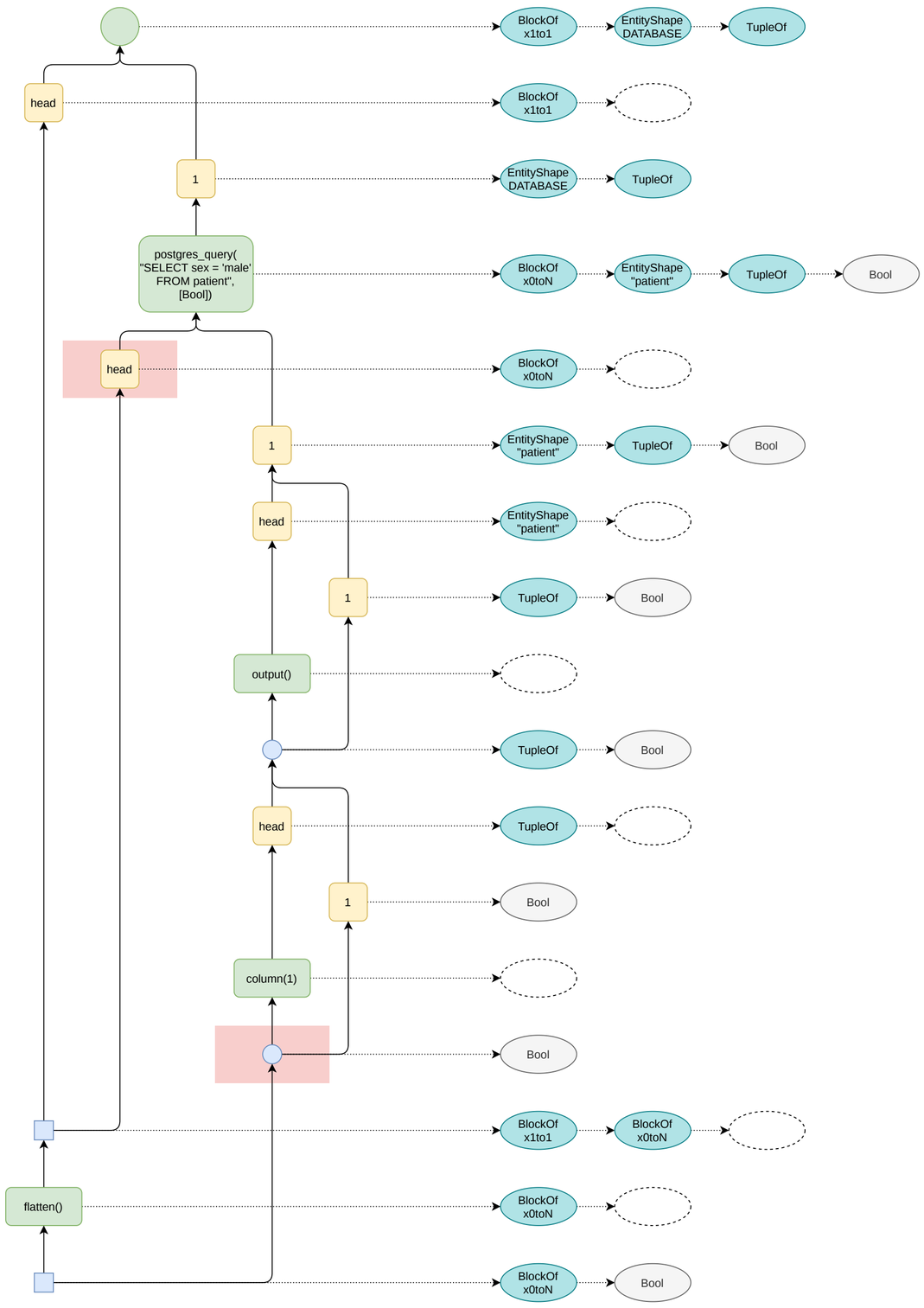
```
chain_of(with_elements(load_postgres_table(("public", "patient"), ["id"], [Int32])),
        flatten(),
        with_elements(
            chain_of(
                load_postgres_table(("public", "patient"), ["mrn"], [String], ["id"]),
                block_cardinality(x1to1))),
        flatten(),
        with_elements(
            chain_of(
                output(),
                column(1))))
```

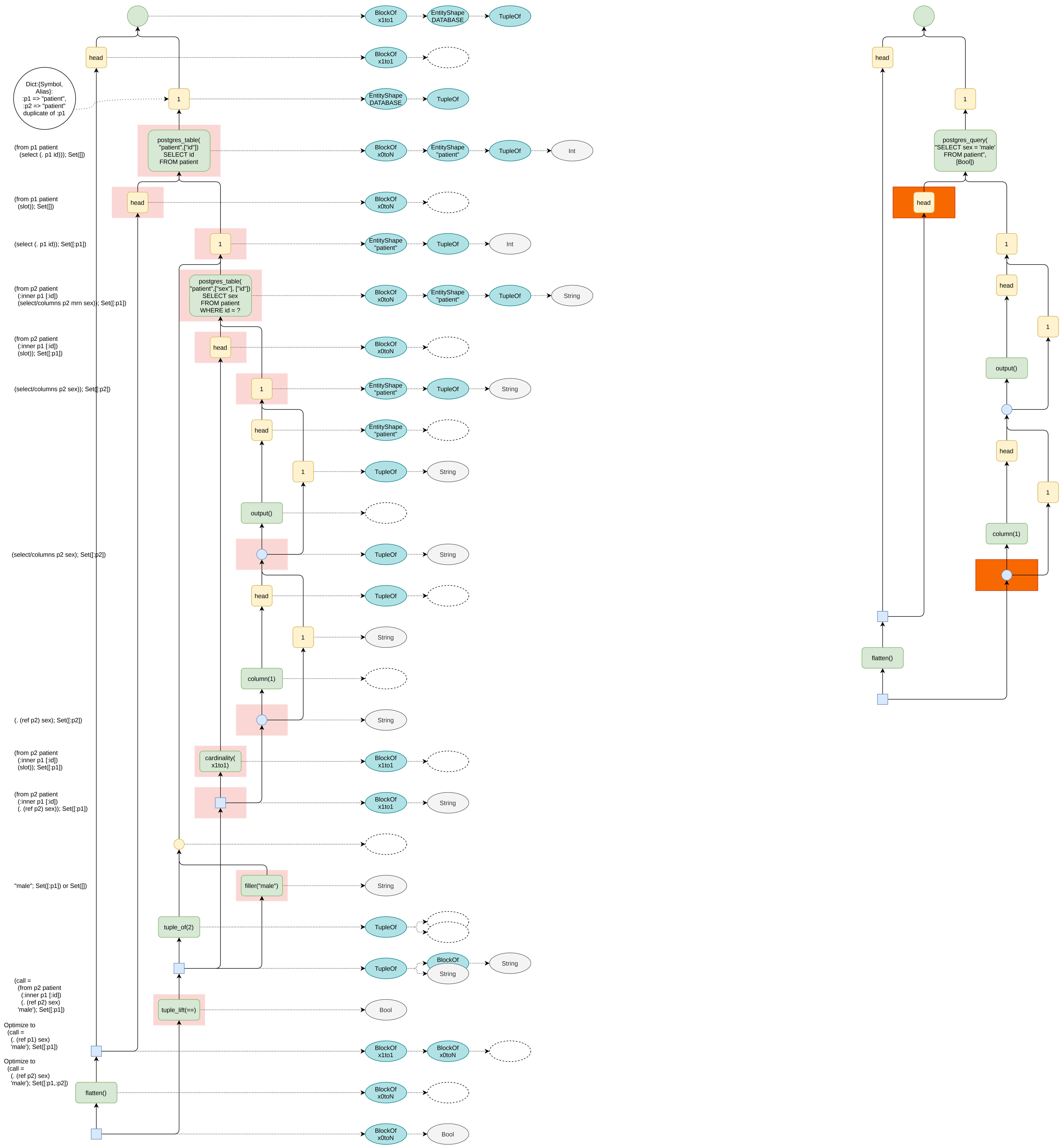



BlockOf x1to1
EntityShape DATABASE
TupleOf
head
BlockOf x1to1
1
EntityShape DATABASE
TupleOf
load_table("patient",["mrn"]) SELECT mrn FROM patient
BlockOf x0toN
EntityShape "patient"
TupleOf
String
head
BlockOf x0toN
1
EntityShape "patient"
TupleOf
String
wrap()
BlockOf x1to1
BlockOf x1to1
BlockOf x0toN
flatten()
BlockOf x0toN
`
BlockOf x0toN
BlockOf x1to1
flatten()
BlockOf x0toN
head
EntityShape "patient"
1
TupleOf
String
output()
TupleOf
String
head
TupleOf
1
String
column(1)
String
`
BlockOf x0toN
String

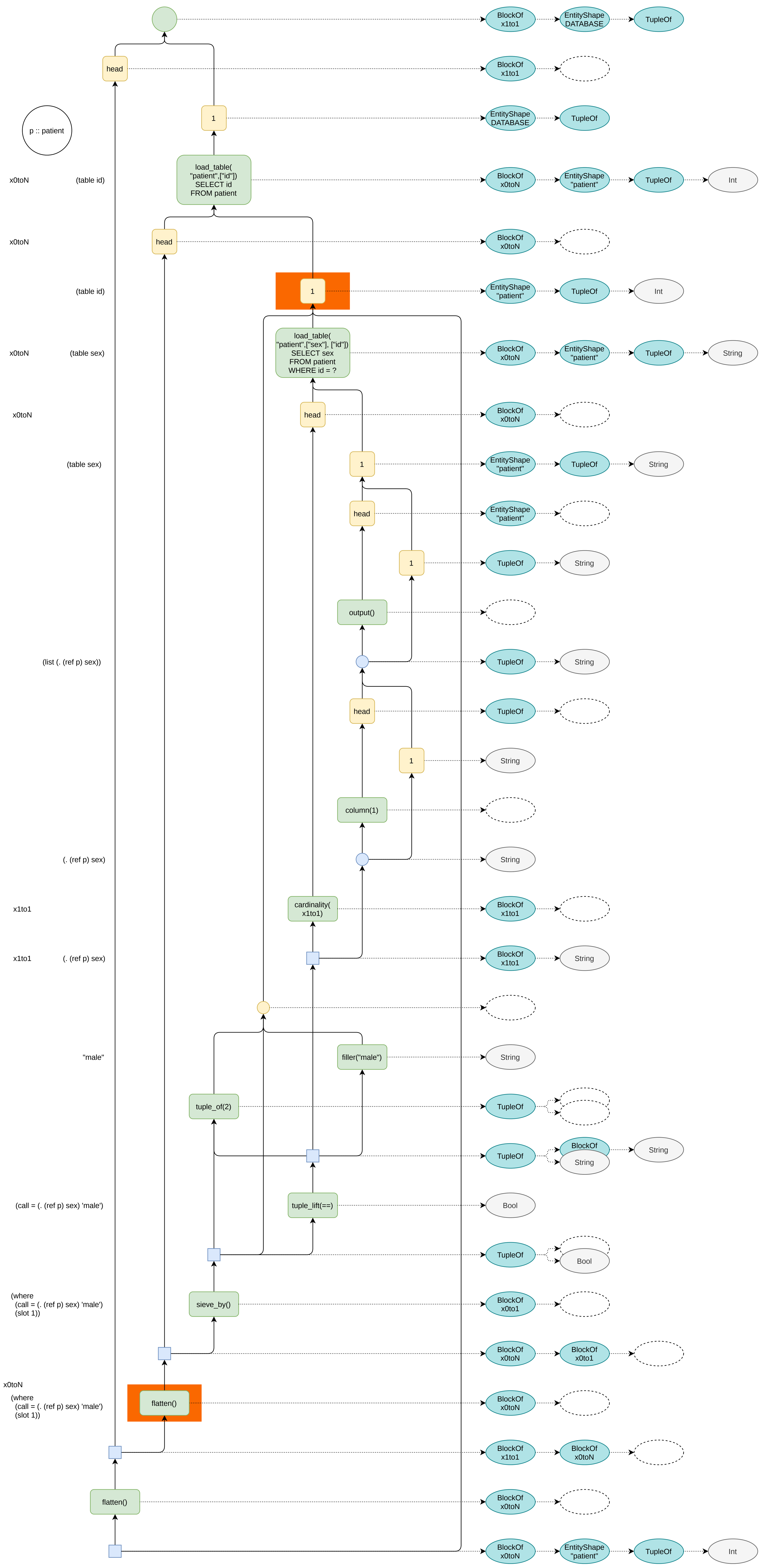

BlockOf x1to1
EntityShape DATABASE
TupleOf
head
BlockOf x1to1
1
EntityShape DATABASE
TupleOf
postgres_table("patient",["sex"]) SELECT sex FROM patient
BlockOf x0toN
EntityShape "patient"
TupleOf
String
head
BlockOf x0toN
1
EntityShape "patient"
TupleOf
String
head
EntityShape "patient"
1
TupleOf
String
output()
TupleOf
String
head
TupleOf
1
String
column(1)
String
filler("male")
String
tuple_of(2)
TupleOf
TupleOf
String
tuple_lift(==)
Bool
BlockOf x1to1
BlockOf x0toN
flatten()
BlockOf x0toN
BlockOf x0toN
Bool


BlockOf x1to1
EntityShape DATABASE
TupleOf
head
BlockOf x1to1
1
EntityShape DATABASE
TupleOf
postgres_table("patient",["id"]) SELECT id FROM patient
BlockOf x0toN
EntityShape "patient"
TupleOf
Int
head
BlockOf x0toN
1
EntityShape "patient"
TupleOf
Int
postgres_table("patient",["sex"], {"id"}) SELECT sex FROM patient WHERE id = ?
BlockOf x0toN
EntityShape "patient"
TupleOf
String
head
BlockOf x0toN
1
EntityShape "patient"
TupleOf
String
head
EntityShape "patient"
1
TupleOf
String
output()
TupleOf
String
head
TupleOf
1
String
column(1)
String
cardinality(x1to1)
BlockOf x1to1
BlockOf x1to1
String
filler("male")
String
tuple_of(2)
TupleOf
TupleOf
BlockOf
String
String
tuple_lift(==)
Bool
BlockOf x1to1
BlockOf x0toN
flatten()
BlockOf x0toN
BlockOf x0toN
Bool

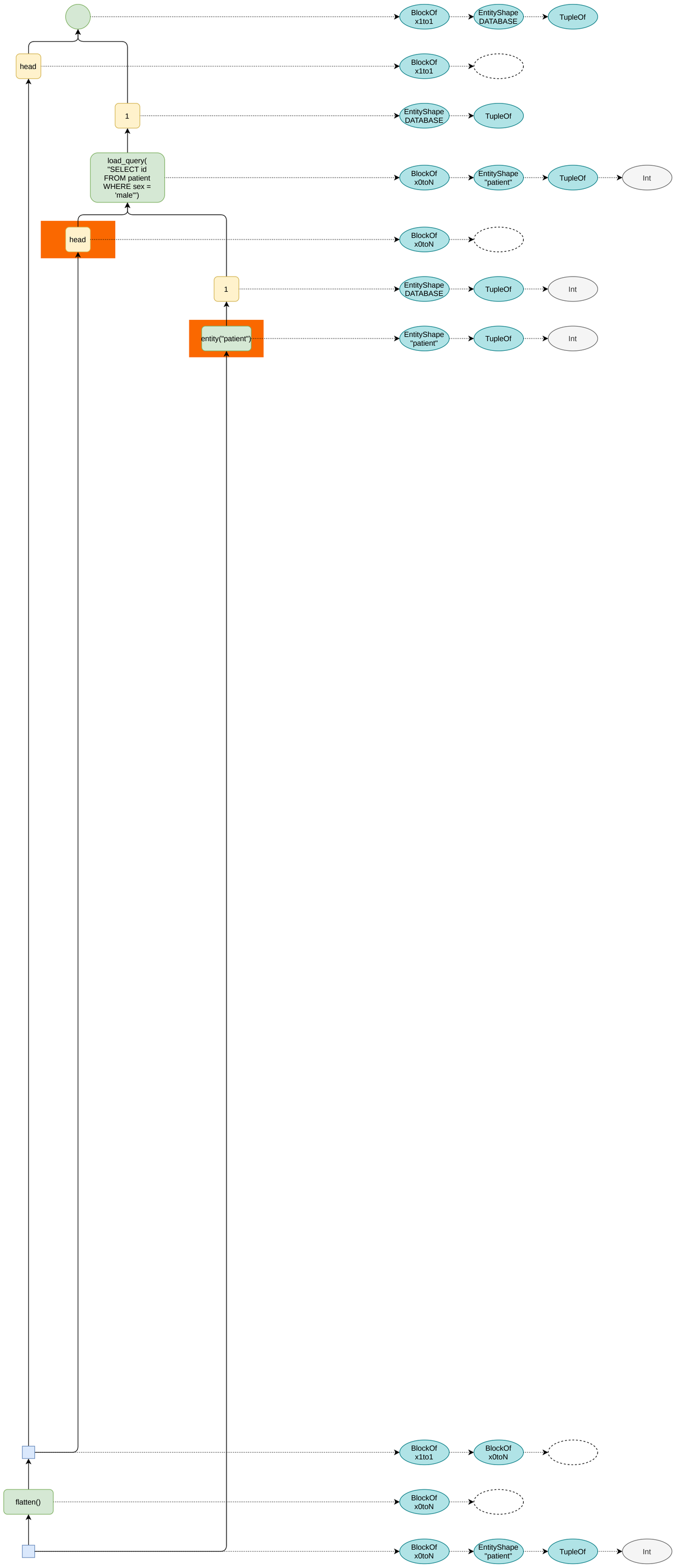

BlockOf x1to1
EntityShape DATABASE
TupleOf
head
BlockOf x1to1
1
EntityShape DATABASE
TupleOf
postgres_query("SELECT sex = 'male' FROM patient", [Bool])
BlockOf x0toN
EntityShape "patient"
TupleOf
Bool
head
BlockOf x0toN
1
EntityShape "patient"
TupleOf
Bool
head
EntityShape "patient"
1
TupleOf
Bool
output()
TupleOf
Bool
head
TupleOf
1
Bool
column(1)
Bool
BlockOf x1to1
BlockOf x0toN
flatten()
BlockOf x0toN
BlockOf x0toN
Bool



BlockOf x1to1
EntityShape DATABASE
TupleOf
head
Dict:{Symbol, Alias}: :p1 => "patient", :p2 => "patient" duplicate of :p1
1
(from p1 patient (select (. p1 id))); Set([])
postgres_table("patient",["id"]) SELECT id FROM patient
BlockOf x0toN
EntityShape "patient"
Int
(from p1 patient (slot)); Set([])
(select (. p1 id)); Set([:p1])
(from p2 patient (:inner p1 [:id]) (select/columns p2 mrn sex)); Set([:p1])
postgres_table("patient",["sex"], ["id"]) SELECT sex FROM patient WHERE id = ?
String
(from p2 patient (:inner p1 [:id]) (slot)); Set([:p1])
(select/columns p2 sex)); Set([:p2])
output()
(select/columns p2 sex); Set([:p2])
column(1)
(. (ref p2) sex); Set([:p2])
(from p2 patient (:inner p1 [:id]) (slot)); Set([:p1])
cardinality(x1to1)
(from p2 patient (:inner p1 [:id]) (. (ref p2) sex)); Set([:p1])
"male"; Set([:p1]) or Set([])
filler("male")
tuple_of(2)
BlockOf
(call = (from p2 patient (:inner p1 [:id]) (. (ref p2) sex) 'male'); Set([:p1])
tuple_lift(==)
Bool
Optimize to (call = (. (ref p1) sex) 'male'); Set([:p1])
Optimize to (call = (. (ref p2) sex) 'male'); Set([:p1,:p2])
flatten()
postgres_query("SELECT sex = 'male' FROM patient", [Bool])



BlockOf x1to1
EntityShape DATABASE
TupleOf
head
BlockOf x1to1
p :: patient
1
EntityShape DATABASE
TupleOf
x0toN
(table id)
load_table("patient",["id"]) SELECT id FROM patient
BlockOf x0toN
EntityShape "patient"
TupleOf
Int
x0toN
head
BlockOf x0toN
(table id)
1
EntityShape "patient"
TupleOf
Int
x0toN
(table sex)
load_table("patient",["sex"], ["id"]) SELECT sex FROM patient WHERE id = ?
BlockOf x0toN
EntityShape "patient"
TupleOf
String
x0toN
head
BlockOf x0toN
(table sex)
1
EntityShape "patient"
TupleOf
String
head
EntityShape "patient"
1
TupleOf
String
output()
(list (. (ref p) sex))
TupleOf
String
head
TupleOf
1
String
column(1)
(. (ref p) sex)
String
x1to1
cardinality(x1to1)
BlockOf x1to1
x1to1
(. (ref p) sex)
BlockOf x1to1
String
"male"
filler("male")
String
tuple_of(2)
TupleOf
TupleOf
BlockOf
String
String
(call = (. (ref p) sex) 'male')
tuple_lift(==)
Bool
TupleOf
Bool
(where (call = (. (ref p) sex) 'male') (slot 1))
sieve_by()
BlockOf x0to1
BlockOf x0toN
BlockOf x0to1
x0toN
(where (call = (. (ref p) sex) 'male') (slot 1))
flatten()
BlockOf x0toN
BlockOf x1to1
BlockOf x0toN
flatten()
BlockOf x0toN
BlockOf x0toN
EntityShape "patient"
TupleOf
Int


BlockOf x1to1
EntityShape DATABASE
TupleOf
head
BlockOf x1to1
1
EntityShape DATABASE
TupleOf
load_query("SELECT id FROM patient WHERE sex = 'male'")
BlockOf x0toN
EntityShape "patient"
TupleOf
Int
head
BlockOf x0toN
1
EntityShape DATABASE
TupleOf
Int
entity("patient")
EntityShape "patient"
TupleOf
Int
BlockOf x1to1
BlockOf x0toN
flatten()
BlockOf x0toN
BlockOf x0toN
EntityShape "patient"
TupleOf
Int