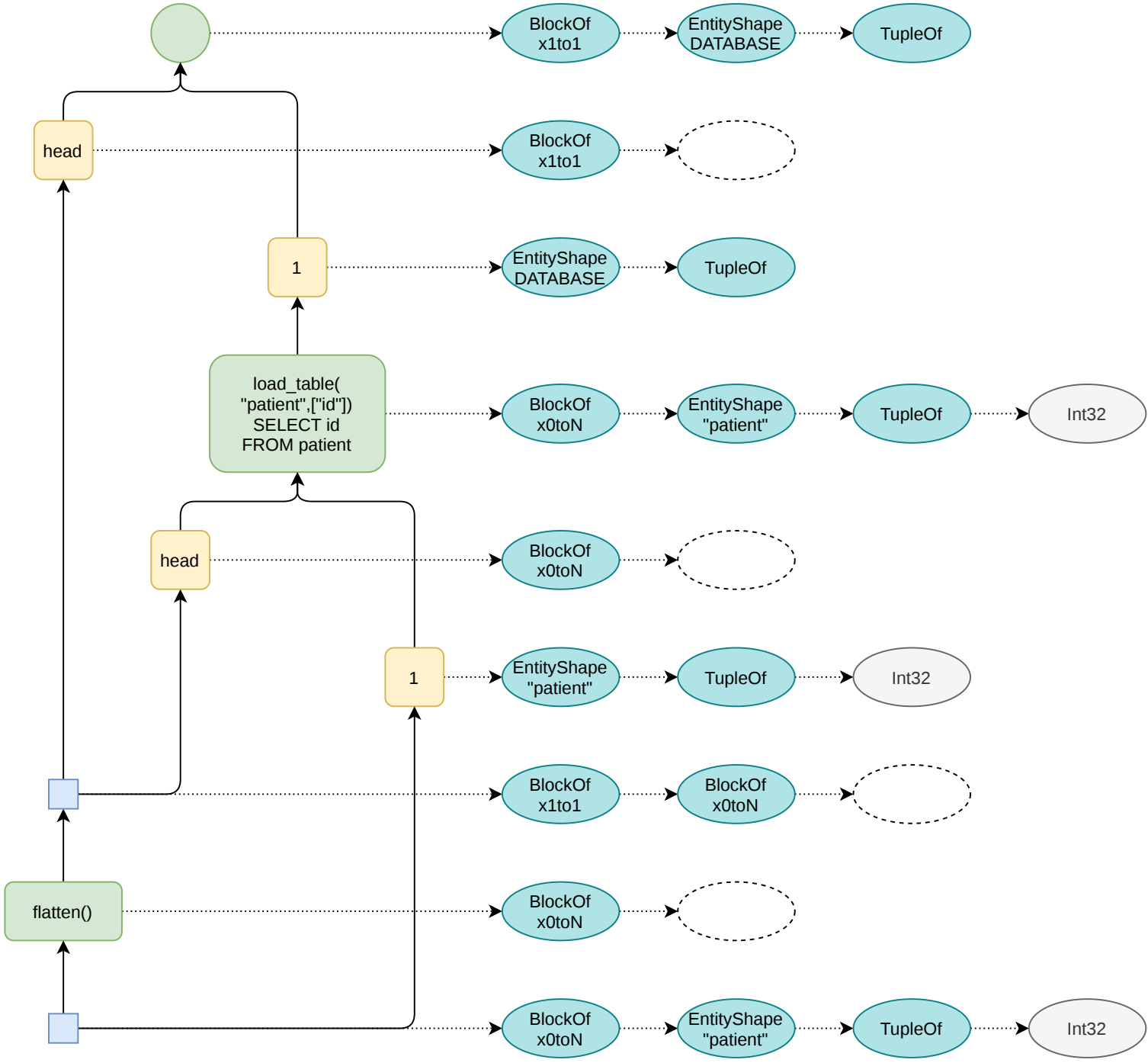


```
chain_of(
  with_elements(load_postgres_table(("public", "patient"), ["id"], [Int32])),
  flatten())
```



```
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))))
```









The diagram illustrates a neural network architecture for SQL query generation, showing a sequence of operations and their corresponding symbolic representations.

Left Side (Operations):

- Input:** A sequence of tokens (represented by blue squares) is processed by an `embed()` layer (green rectangle).
- Flattening:** The embedded sequence is flattened (`flatten()`, green rectangle).
- Cardinality:** The flattened sequence is processed by a `cardinality()` layer (green rectangle).
- Column Selection:** A `column()` layer (green rectangle) is applied to the output of the `cardinality()` layer.
- Output:** The final output is generated by an `output()` layer (green rectangle).

Right Side (Symbolic Representations):

- BlockOf:** Represented by blue ovals (e.g., `BlockOf x1to1`, `BlockOf x0toN`).
- EntityShape:** Represented by green ovals (e.g., `EntityShape DATABASE`, `EntityShape "patient"`).
- TupleOf:** Represented by blue ovals (e.g., `TupleOf`, `TupleOf "patient"`).
- Int32:** Represented by grey ovals (e.g., `Int32`).
- String:** Represented by grey ovals (e.g., `String`).

Connections:

- Dotted lines connect the operations on the left to their corresponding symbolic representations on the right.
- Solid lines show the flow of data through the network layers.













