

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



The diagram illustrates a complex computational graph, likely representing a neural network architecture or a data processing pipeline. The graph is composed of several interconnected nodes and edges, with a large blue shaded region and a large orange shaded region.

Key Components and Operations:

- Input/Output Nodes:** The graph starts with an input node (green circle) and ends with an output node (green circle). There are also several intermediate nodes, including "head" (yellow rectangle), "1" (yellow rectangle), "load_table" (green rectangle), "cardinality" (green rectangle), "output" (green rectangle), and "column" (green rectangle).
- Shaded Regions:**
 - Blue Shaded Region:** Contains the "load_table" node and the "cardinality" node, along with their associated "head" and "1" nodes.
 - Orange Shaded Region:** Contains the "output" node and the "column" node, along with their associated "head" and "1" nodes.
- Operations and Data Flow:**
 - load_table:** A node that takes "patient" as input and outputs "id". It is associated with a "BlockOf x1to1" and an "EntityShape DATABASE".
 - cardinality:** A node that takes "x1to1" as input and outputs "x1to1". It is associated with a "BlockOf x1to1" and an "EntityShape DATABASE".
 - output:** A node that takes "x1to1" as input and outputs "x1to1". It is associated with a "BlockOf x1to1" and an "EntityShape DATABASE".
 - column:** A node that takes "x1to1" as input and outputs "x1to1". It is associated with a "BlockOf x1to1" and an "EntityShape DATABASE".
- Final Output:** The graph concludes with a "TupleOf" node, which is associated with a "BlockOf x1to1" and an "EntityShape DATABASE".

The diagram shows a complex flow of data and operations, with a large blue shaded region and a large orange shaded region. The right side of the image shows a detailed view of the operations, with nodes like "BlockOf", "EntityShape", "TupleOf", and "Int32".







```
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))))
```















