

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



The diagram illustrates the execution of a SQL query, showing the relationship between the query plan (left) and the resulting data structures (right).

**Query Plan (Left):**

- The query starts with a `load_table("patient", ["id"]) SELECT id FROM patient` operation.
- This is followed by a `cardinality(x1to1)` operation.
- The result is then flattened (`flatten()`).
- The flattened result is used in a `column(1)` operation.
- The final result is output (`output()`).

**Data Structures (Right):**

- The query plan nodes are mapped to specific data structures and values:
- `load_table("patient", ["id"]) SELECT id FROM patient` maps to `BlockOf x1to1`, `EntityShape DATABASE`, and `TupleOf`.
- `cardinality(x1to1)` maps to `BlockOf x1to1`, `EntityShape DATABASE`, and `TupleOf`.
- `flatten()` maps to `BlockOf x1to1`, `EntityShape DATABASE`, and `TupleOf`.
- `column(1)` maps to `BlockOf x1to1`, `EntityShape DATABASE`, and `TupleOf`.
- `output()` maps to `BlockOf x1to1`, `EntityShape DATABASE`, and `TupleOf`.
- The final result is mapped to `BlockOf x1to1`, `EntityShape DATABASE`, and `TupleOf`.

The diagram shows the flow of data from the query plan to the final output, with various intermediate data structures and values.







```
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))))
```



















