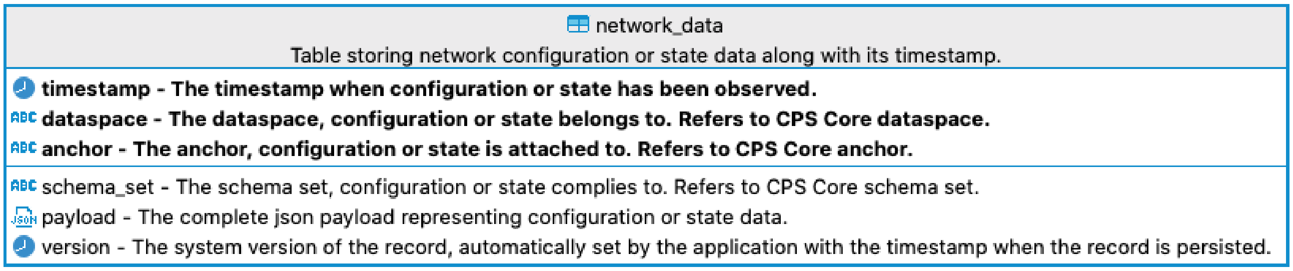
CPS-192: Design data store for Temporal Service

Data Model

Data model for CPS Temporal is limited to one entity to store configuration and state data with its timestamp.

The store keeps a new data instance each time a new data observation is made for a given dataspace, schema set and anchor.

Timestamp is used to partition time-series data.



network_data

Table storing network configuration or state data along with its timestamp.

timestamp - The timestamp when configuration or state has been observed.

dataspace - The dataspace, configuration or state belongs to. Refers to CPS Core dataspace.

anchor - The anchor, configuration or state is attached to. Refers to CPS Core anchor.

schema_set - The schema set, configuration or state complies to. Refers to CPS Core schema set.

payload - The complete json payload representing configuration or state data.

version - The system version of the record, automatically set by the application with the timestamp when the record is persisted.

Implementation

PostgreSQL database is proposed to implement CPS Temporal data store.

Timescale PostgreSQL extension is proposed for time-series data implementation.

By implementing PostgreSQL and its Timescale extension, CPS Temporal benefits from:

- Time-series data features on top of standard well know Postgres database
- All PostgreSQL relational database features
- CPS previous experience with PostgreSQL (default reference implementation for CPS Core)
- PostgreSQL JSONB data type for configuration and state data

Detailed schema information:

```
cpstemporaldb=# \d network_data
                      Table "public.network_data"
   Column   |           Type           | Collation | Nullable | Default
------------+--------------------------+-----------+----------+---------
 timestamp  | timestamp with time zone |           | not null |
 dataspace  | character varying(255)   |           | not null |
 anchor     | character varying(255)   |           | not null |
 schema_set | character varying(255)   |           | not null |
 payload    | jsonb                    |           | not null |
 version    | timestamp with time zone |           | not null |

Indexes:
    "network_data_dataspace_anchor_timestamp_idx" UNIQUE, btree (dataspace, anchor, "timestamp" DESC)
    "network_data_dataspace_schema_set_timestamp_idx" btree (dataspace, schema_set, "timestamp" DESC)
    "network_data_payload_idx" gin (payload)
    "network_data_timestamp_idx" btree ("timestamp" DESC)

cpstemporaldb=# SELECT * FROM timescaledb_information.dimensions;
 hypertable_schema | hypertable_name | dimension_number | column_name |       column_type        |
dimension_type | time_interval | integer_interval | integer_now_func | num_partitions
-------------------+-----------------+------------------+-------------+--------------------------
+----------------+---------------+------------------+------------------+----------------
 public            | network_data    |                1 | timestamp   | timestamp with time zone |
Time           | 1 day         |                  |                  |
(1 row)
```

Indexes above are designed to optimize following queries:

1. By **dataspace**, **anchor**. Order by **timestamp** desc.
2. By **dataspace**, **schema set**. Order by **timestamp** desc.
3. Adding a **payload content** condition to the 2 kinds of queries above. In this case, the gin index is used in combination with the index having the corresponding fields. See attached explain plan for an example.

The index for (timestamp) is automatically created by Timescale to support the main data partitioning by time.

References

- CPS-191: Core and Temporal Integration Design
- https://www.timescale.com/