

### Hypertables

Create Table- 2 steps

1. create a standard postgresql table :

```
CREATE TABLE conditions ( time TIMESTAMPTZ
```

2. use create\_hypertable command

```
SELECT create_hypertable('conditions', 'time');
```

Convert table **conditions** to hypertable with just time partitioning on column

```
SELECT create_hypertable('conditions', 'time', chunk_time_interval => INTERVAL '1 day');
```

```
SELECT create_hypertable('conditions', 'time', 'location', 4, partitioning_func => 'location');
```

```
SELECT create_hypertable('conditions', 'time', if_not_exists => TRUE);
```

Create Index: CREATE INDEX ... WITH (timescaledb.transaction\_per\_chunk,...)

```
CREATE INDEX ON conditions(time, device_id) WITH (timescaledb.transaction_per_chunk);
```

show\_chunks()

```
SELECT show_chunks('conditions');
```

```
SELECT show_chunks(older_than => INTERVAL '3 months');
```

### Compression

Use Alter table

```
ALTER TABLE <table_name> SET (timescaledb.compress, timescaledb.compress_lun_name => [ , ... ]');
```

```
ALTER TABLE metrics SET (timescaledb.compress, timescaledb.compress_orderby = 'time DESC', timescaledb.compress_lun_name => [ , ... ]');
```

add\_compress\_chunks\_policy()

```
SELECT add_compress_chunks_policy('conditions', INTERVAL '60d');
```

### Automation

add\_reorder\_policy()

```
SELECT add_reorder_policy('conditions', 'conditions_device_id_time_idx');
```

alter\_job\_schedule()

```
SELECT alter_job_schedule(job_id, schedule_interval => INTERVAL '5 minutes') FROM timescaledb_info;
-- conditions_agg ':: regclass;
```

### Automation (cont)

reschedules the continuous aggregate job for the conditions\_agg view so that it runs every five minutes.

### Utilities

Get information about hypertables

```
SELECT * FROM timescaledb_info.hypertables;
```

Get statistics about chunk compression

```
SELECT * FROM timescaledb_info.chunk_compression;
```

Get statistics about hypertable compression

```
SELECT * FROM timescaledb_info.hypertable_compression;
```

Get metadata and settings information for continuous aggregates

```
SELECT * FROM timescaledb_info.continuous_aggregates;
```

Get information about background jobs and statistics related to continuous aggregates

```
SELECT * FROM timescaledb_info.background_jobs;
```

Get relation size of the chunks of an hypertable

```
SELECT chunk_table, table_size, index_size, total_size FROM timescaledb_info.chunk_sizes;
```

Get approximate row count for hypertable(s) based on catalog estimate

```
SELECT * FROM hypertable_approximate_row_count;
```

Get relation size of hypertable

```
SELECT table_size, index_size, toast_size, total_size FROM timescaledb_info.hypertable_sizes;
```

### Continuous Aggregate

Create View

```
CREATE VIEW continuous_aggregate_view( time, time_bucket('30 day', time));
```

Alter View

```
ALTER VIEW contagg_view SET (timescaledb.refresh_view_interval => INTERVAL '1 day');
```

Refresh View

```
REFRESH MATERIALIZED VIEW contagg_view;
```

Drop View

```
DROP VIEW contagg_view CASCADE;
```

