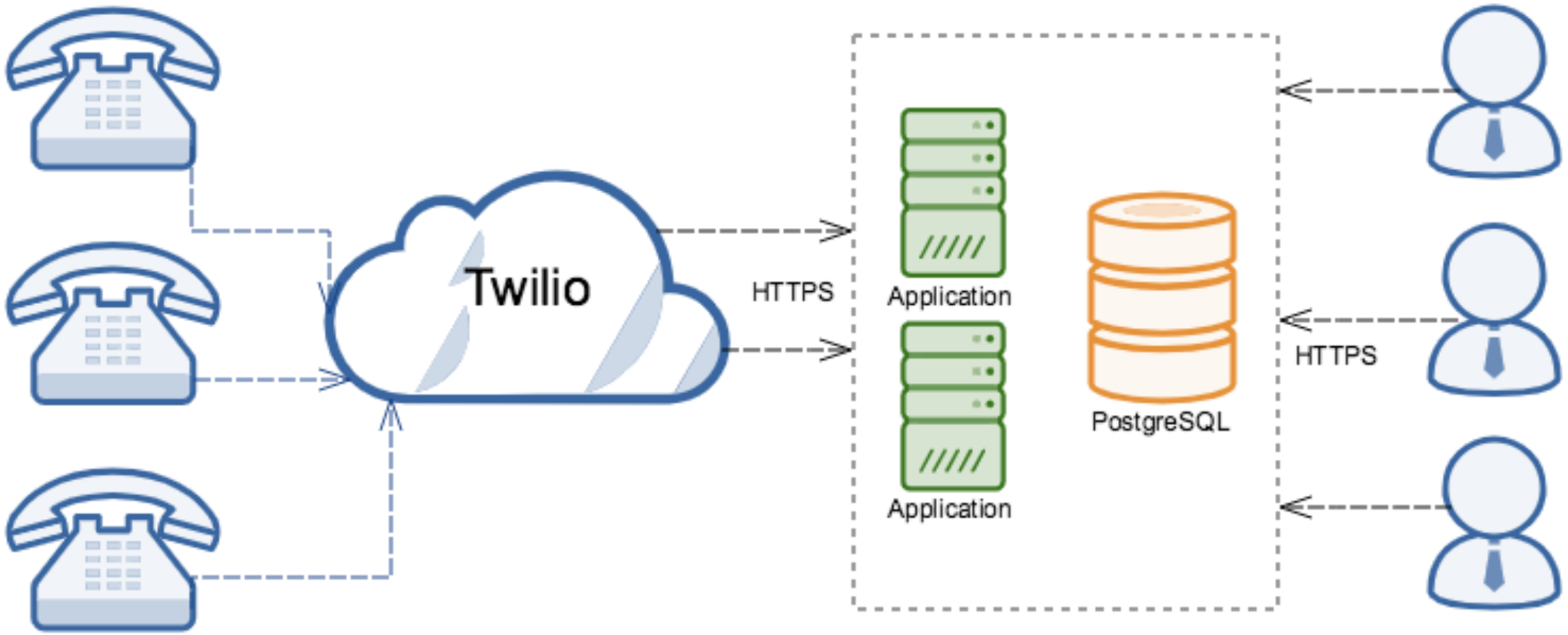
Temporal data & time travel in PostgreSQL

Alexei Mikhailov, SaleMove

Outline

- **Debugging with logs**
- **Temporal tables**
- **PostgreSQL extension temporal_tables**
- **Problems**
- **Alternative solutions**

Call center



Call center DB

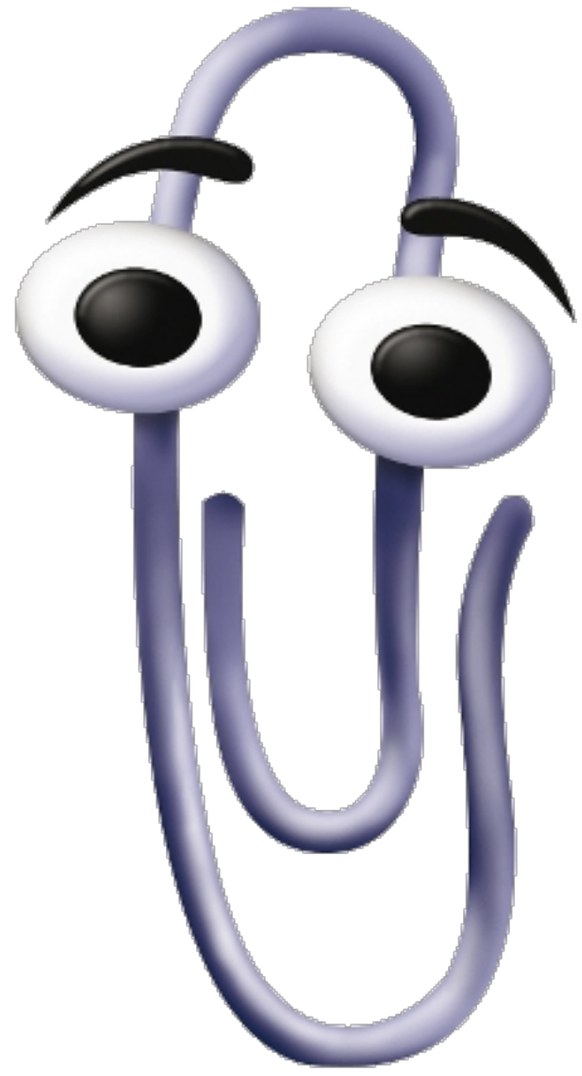
Table “**agents**”

Column	Type
id	integer
status	character varying

Call center DB

Table “**agents**”

Column	Type
id	integer
status	character varying



Logging

```
logger.info("Call received", call.id)
...
logger.info("Agent is online", agent.id)
...
logger.info("Agent received call",
    call.id, agent.id)
...
logger.info("Agent is offline", agent.id)
```

Logging

```
logger.info("Call received", call.id)
...
logger.info("Agent is online", agent.id)
...
logger.info("Agent received call",
    call.id, agent.id)
...
logger.info("Agent is offline", agent.id)
```

Call center DB

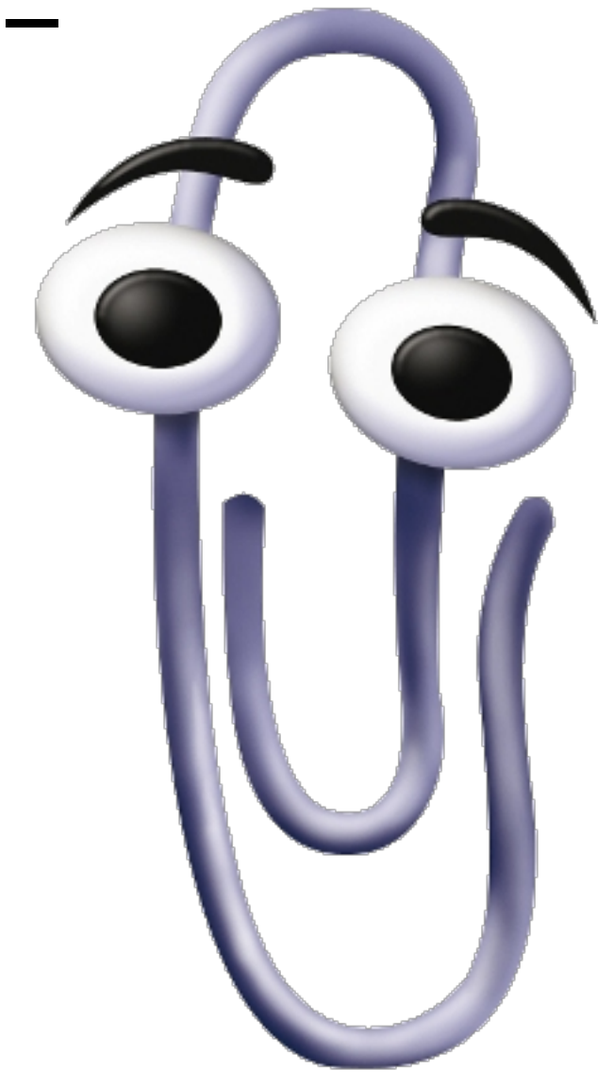
Table “**agents**”

Column	Type
id	integer
status	character varying
priority	integer default 0

Call center DB

Table “**agents**”

Column	Type
id	integer
status	character varying
priority	integer default 0



Logs

```
2018-11-11 11:00 Agent received call, ID=6038
2018-11-11 11:01 Agent is offline, ID=8594
2018-11-11 11:02 Call finished, ID=4757
2018-11-11 11:03 Agent is ready, ID=8022
2018-11-11 11:04 Agent is online, ID=3538
2018-11-11 11:05 Call received, ID=3133
2018-11-11 11:06 Agent is ready, ID=9886
2018-11-11 11:07 Agent received call, ID=6061
2018-11-11 11:08 Agent is offline, ID=7769
2018-11-11 11:09 Agent is ready, ID=9012
2018-11-11 11:10 Agent received call, ID=6698
2018-11-11 11:11 Agent is ready, ID=5740
2018-11-11 11:12 Call received, ID=5376
2018-11-11 11:13 Agent is online, ID=8493
2018-11-11 11:14 Agent is offline, ID=8832
2018-11-11 11:15 Agent is online, ID=6997
2018-11-11 11:16 Call finished, ID=8042
2018-11-11 11:17 Agent received call, ID=3666
2018-11-11 11:18 Agent received call, ID=5452
2018-11-11 11:19 Call finished, ID=7292
2018-11-11 11:20 Agent received call, ID=9690
2018-11-11 11:21 Call finished, ID=8523
2018-11-11 11:22 Call finished, ID=7981
2018-11-11 11:23 Agent received call, ID=3991
2018-11-11 11:24 Agent is ready, ID=3091
2018-11-11 11:25 Call received, ID=9600
2018-11-11 11:26 Agent is offline, ID=7565
2018-11-11 11:27 Agent is online, ID=3256
2018-11-11 11:28 Agent is ready, ID=3060
2018-11-11 11:29 Agent is offline, ID=3569
```

Logs

2018-11-11 11:00 Agent received call, ID=6038, **priority=2**
2018-11-11 11:01 Agent is offline, ID=8594, **priority=1**
2018-11-11 11:02 Call finished, ID=4757
2018-11-11 11:03 Agent is ready, ID=8022, **priority=4**
2018-11-11 11:04 Agent is online, ID=3538, **priority=3**
2018-11-11 11:05 Call received, ID=3133
2018-11-11 11:06 Agent is ready, ID=9886, **priority=3**
2018-11-11 11:07 Agent received call, ID=6061, **priority=5**
2018-11-11 11:08 Agent is offline, ID=7769, **priority=2**
2018-11-11 11:09 Agent is ready, ID=9012, **priority=1**
2018-11-11 11:10 Agent received call, ID=6698, **priority=3**
2018-11-11 11:11 Agent is ready, ID=5740, **priority=2**
2018-11-11 11:12 Call received, ID=5376
2018-11-11 11:13 Agent is online, ID=8493, **priority=4**
2018-11-11 11:14 Agent is offline, ID=8832, **priority=4**
2018-11-11 11:15 Agent is online, ID=6997, **priority=1**
2018-11-11 11:16 Call finished, ID=8042
2018-11-11 11:17 Agent received call, ID=3666, **priority=1**
2018-11-11 11:18 Agent received call, ID=5452, **priority=4**
2018-11-11 11:19 Call finished, ID=7292
2018-11-11 11:20 Agent received call, ID=9690, **priority=3**
2018-11-11 11:21 Call finished, ID=8523
2018-11-11 11:22 Call finished, ID=7981
2018-11-11 11:23 Agent received call, ID=3991, **priority=2**
2018-11-11 11:24 Agent is ready, ID=3091, **priority=5**
2018-11-11 11:25 Call received, ID=9600
2018-11-11 11:26 Agent is offline, ID=7565, **priority=1**
2018-11-11 11:27 Agent is online, ID=3256, **priority=1**
2018-11-11 11:28 Agent is ready, ID=3060, **priority=3**
2018-11-11 11:29 Agent is offline, ID=3569, **priority=2**

Logs have flaws

- **Incomplete**
  `> 2018-11-11 11:11 Agent is ready`
  What agent? What is his priority?

- **Personally identifiable information (PII)**

- **Reactive**
  Didn't have logs back then? GL HF

- **Hard to use**
  `> 2018-11-10 23:00 Agent is ready, id=007`
  `> 2018-11-12 01:00 Agent is offline, id=007`

  *Search logs for date 2018-11-**11**: no results for agent 007*

- **Not durable** *

Temporal tables

- **Audit and security**
- **Debugging**
- **Undo functionality**
- **Statistics, prediction**

Temporal tables

- Big new feature of SQL 2011
  https://cs.ulb.ac.be/public/_media/teaching/infoh415/tempfeaturessql2011.pdf

- Introduces new syntax
  `... FROM table FOR SYSTEM_TIME AS OF '<timestamp>'`

$$$	**Oracle**	Yes	Since 10g+ (2005)
$$$	**IBM DB2**	Yes	Since version 10 (2010)
$$$	**MS SQL Server**	Yes	Since SQL Server 2016
	PostgreSQL	No	3rd party extension temporal_tables
	MySQL	No	

Temporal tables

- Tables have additional time-period (interval) information
- Time period is closed-open (inclusive from the left, exclusive from the right)
    - PostgreSQL: **`tstzrange('[2018-01-01, 2018-12-01)')`**
- **System time-period**: what was the ACID state of my data on <time>?
    - Maintained by system
    - **UPDATE** / **DELETE** / **INSERT** automatically maintain this info
    - Only supports history, must never be future
    - Used by **`...`**`FROM table FOR SYSTEM_TIME AS OF`
    - **Usually separate table**

Temporal tables

- **Business time-period**
    - Maintained by application / user
    - Represent business reality, e.g. "This contract will have price X for next 3 months"

```
contract_id | price |       valid_period
------------+-------+--------------------------
     1      |  100  | [2018-01-01, 2018-12-01)
     1      |  50   | [2018-12-01, 2019-02-01)
```

    - Can be in the future
    - No special syntax for SELECT
- **Bi-temporal tables** = tables with system period and business period

History table

```
ALTER TABLE agents
ADD COLUMN system_period tstzrange NOT NULL;

CREATE TABLE agents_history (LIKE agents);
```

History table

```
CREATE FUNCTION save_previous_version() RETURNS TRIGGER AS $body$
BEGIN
  IF (TG_OP = 'INSERT') THEN
    NEW.system_period := tstzrange(current_timestamp, NULL, '[)');
    RETURN NEW;
  ELSIF (TG_OP = 'UPDATE') THEN
    NEW.system_period := tstzrange(current_timestamp, NULL, '[)');
    INSERT INTO agents_history (id, status, priority,
system_period) VALUES (OLD.id, OLD.status, OLD.priority,
tstzrange(lower(OLD.system_period), current_timestamp, '[)'));
    RETURN NEW;
  ELSE
    INSERT INTO agents_history (id, status, priority,
system_period) VALUES (OLD.id, OLD.status, OLD.priority,
tstzrange(lower(OLD.system_period), current_timestamp, '[)'));
    RETURN OLD;
  END IF;
END;
$body$
LANGUAGE plpgsql;

CREATE TRIGGER history_trigger
BEFORE INSERT OR UPDATE OR DELETE ON agents
FOR EACH ROW EXECUTE PROCEDURE save_previous_version();
```

History table

INSERT INTO agents (id, status, priority) VALUES (1, 'ready', 0);

"agents"

```
 id | status | priority | system_period
----+--------+----------+----------------
  1 | ready  |        0 | ["18:59:17",)
(1 row)
```

"agents_history"

```
 id | status | priority | system_period
----+--------+----------+----------------
(0 rows)
```

History table

```
UPDATE agents SET status = 'offline' WHERE id = 1;
```

"agents"

```
 id | status  | priority | system_period
----+---------+----------+---------------
  1 | offline |        0 | ["19:10:47",)
(1 row)
```

"agents_history"

```
 id | status | priority |          system_period
----+--------+----------+-------------------------
  1 | ready  |        0 | ["18:59:17","19:10:47")
(1 row)
```

History table

```
UPDATE agents SET priority = 2 WHERE id = 1;
```

"agents"

```
 id | status  | priority | system_period
----+---------+----------+---------------
  1 | offline |        2 | ["19:15:06",)
(1 row)
```

"agents_history"

```
 id | status  | priority |          system_period
----+---------+----------+--------------------------------
  1 | ready   |        0 | ["18:59:17","19:10:47")
  1 | offline |        0 | ["19:10:47","19:15:06")
(2 rows)
```

History table

```
DELETE FROM agents WHERE id = 1;
```

"agents"

```
 id | status  | priority | valid_from | valid_to
----+---------+----------+------------+----------
(0 rows)
```

"agents_history"

```
 id | status  | priority |                system_period
----+---------+----------+----------------------------------------
  1 | ready   |        0 | ["18:59:17","19:10:47")
  1 | offline |        0 | ["19:10:47","19:15:06")
  1 | offline |        2 | ["19:15:06","19:17:47")
(3 rows)
```

Queries

```
CREATE VIEW agents_with_history AS
SELECT * FROM agents
UNION ALL
SELECT * FROM agents_history
```

Queries

How did the data look like at point in time T?

```
SELECT * FROM agents_with_history
WHERE id = 1
  AND '19:12'::timestamptz <@ system_period

 id | status  | priority |       system_period
----+---------+----------+---------------------------
  1 | offline |        0 | ["19:10:47","19:15:06")
(1 row)

SELECT * FROM agents_with_history
WHERE id = 1
  AND '19:09'::timestamptz <@ system_period

 id | status | priority |       system_period
----+--------+----------+---------------------------
  1 | ready  |        0 | ["18:59:17","19:10:47")
(1 row)
```

Queries

How did the data change in interval (T1, T2)

```
SELECT * FROM agents_with_history
WHERE id = 1
  AND tstzrange('[18:55,19:12]') <@ system_period

 id | status  | priority |      system_period
----+---------+----------+--------------------------
  1 | ready   |        0 | ["18:59:17","19:10:47")
  1 | offline |        0 | ["19:10:47","19:15:06")
(2 rows)

SELECT * FROM agents_with_history
WHERE id = 1
  AND tstzrange('[19:05,)') <@ system_period

 id | status  | priority |      system_period
----+---------+----------+--------------------------
  1 | offline |        0 | ["19:10:47","19:15:06")
  1 | offline |        2 | ["19:15:06","19:15:47")
(2 rows)
```

Extension

- https://github.com/arkhipov/temporal_tables
- **pgxnclient install temporal_tables**
- **CREATE EXTENSION temporal_tables**

  Requires superuser

- ```
  CREATE TRIGGER versioning_trigger
  BEFORE INSERT OR UPDATE OR DELETE ON agents
  FOR EACH ROW EXECUTE PROCEDURE
  versioning ('system_period', 'agents_history', true);
  ```

- **Doesn't require same set of columns**
  ```

Advantages

- **ACID**
- **Backups**
- **Easy to integrate with existing database**
- **Low performance overhead**

Data duplication

- **Normalization**
  Move heavy columns out of the main table

  ***Before**: users (id, name, picture)*
  ***After**: users(id, name, picture_id), pictures(id, data)*

- **Remove heavy columns from historical table**
  Data will be lost

- **Prune historical tables**

- **Separate tablespace on cheap HD**

Caveats

- **Structure is not in sync with main table**
    - Can use inheritance, but with caution
        - Event triggers (i.e. *do something when schema changes*)
- **No primary key, no foreign keys**
- **You MUST use tstzrange for system_period column**
  If you're using temporal_tables extension. Only "gist" index is supported.

Alternative solutions

- **RDBMS with support for temporal tables**
  Oracle, IBM DB2, MS SQL

- **Event sourcing**

- **Immutable data**

```
| agents |    | statuses   |
+--------+    +------------+
| id     |<---| agent_id   |
| name   |    | status     |
                | timestamp  |
```

- **Application-managed audit logs**

Demo

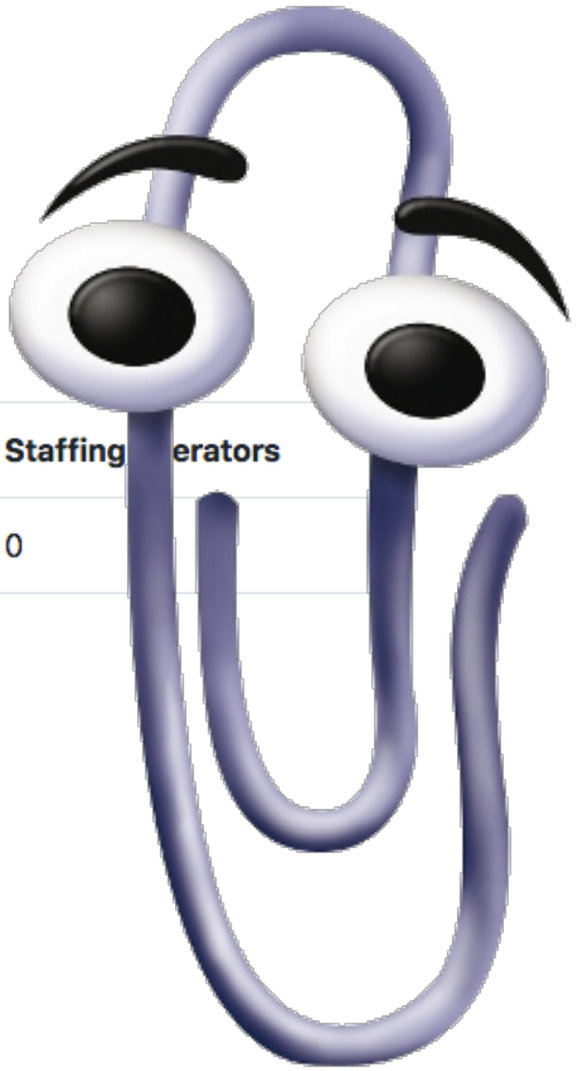
Site ID(s) cc0a1ccc-e00a-4b28-9ea5-277548995fec ×

Queue ID(s)

Point in time dd/mm/yyyy, --:--:-- (GMT-06:00) Central Time (US

Show

Queue Name	ID	Status	Enqueued visitors	Staffing erators
General	8788b207-edf8-4a66-a27f-ccd73b565e60	unstaffed	0	0



Demo

Site ID(s) cc0a1ccc-e00a-4b28-9ea5-277548995fec ×

Queue ID(s)

Point in time dd/mm/yyyy, --:--:-- (GMT-06:00) Central Time (US

Show

Queue Name	ID	Status	Enqueued visitors	Staffing operators
General	8788b207-edf8-4a66-a27f-ccd73b565e60	unstaffed	0	0

Recap

- Logs are not sufficient
- Temporal tables are possible in any DB
- Use with caution
- Know alternatives

Questions?

Slides:
https://github.com/take-five/temporal-tables-presentation