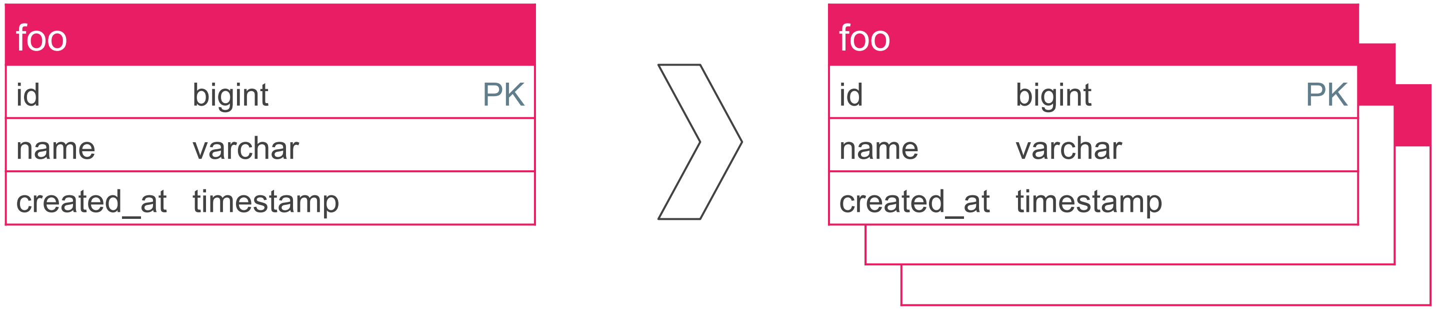
Partitionnement par UUID v7

Florent Jardin – PG Day France 2025

DALIBO
L'expertise PostgreSQL

Besoin

- Une table doit être partitionnée sur une colonne TIMESTAMP pour faciliter les purges mensuelles

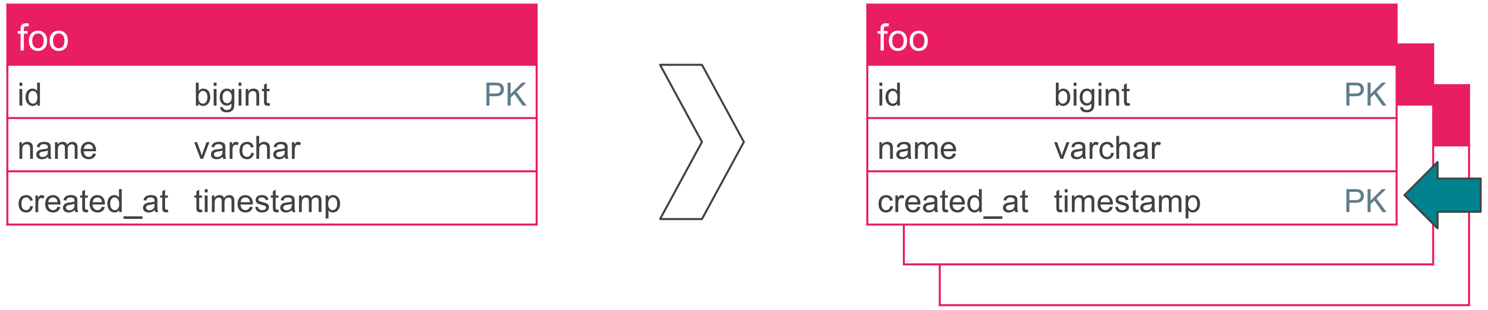


Limitations

- PostgreSQL impose d'inclure les clés de partitionnement dans les contraintes de clés primaires ou contraintes uniques

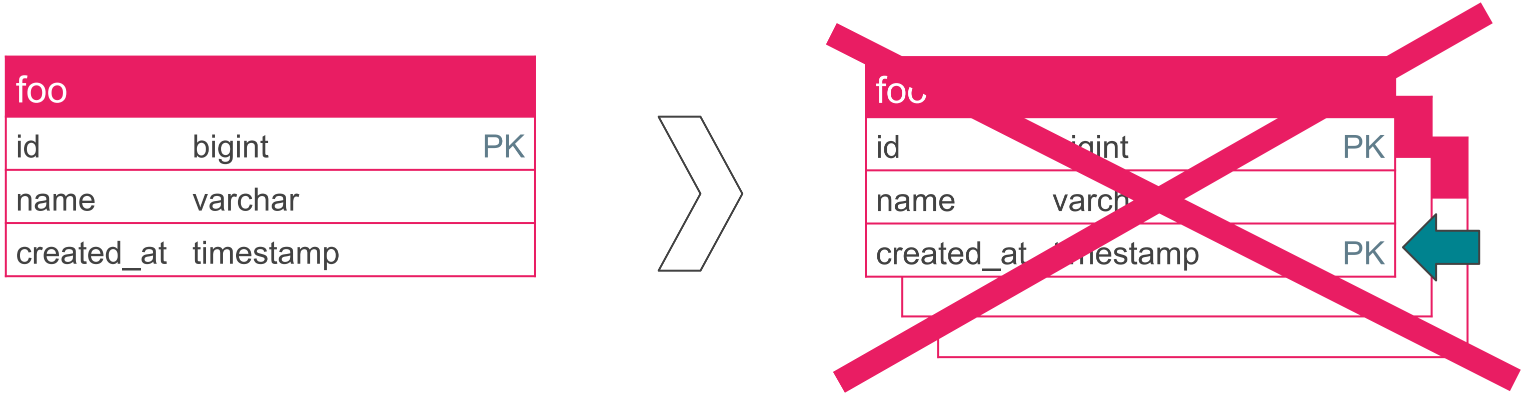
Limitations

- PostgreSQL impose d'inclure les clés de partitionnement dans les contraintes de clés primaires ou contraintes uniques



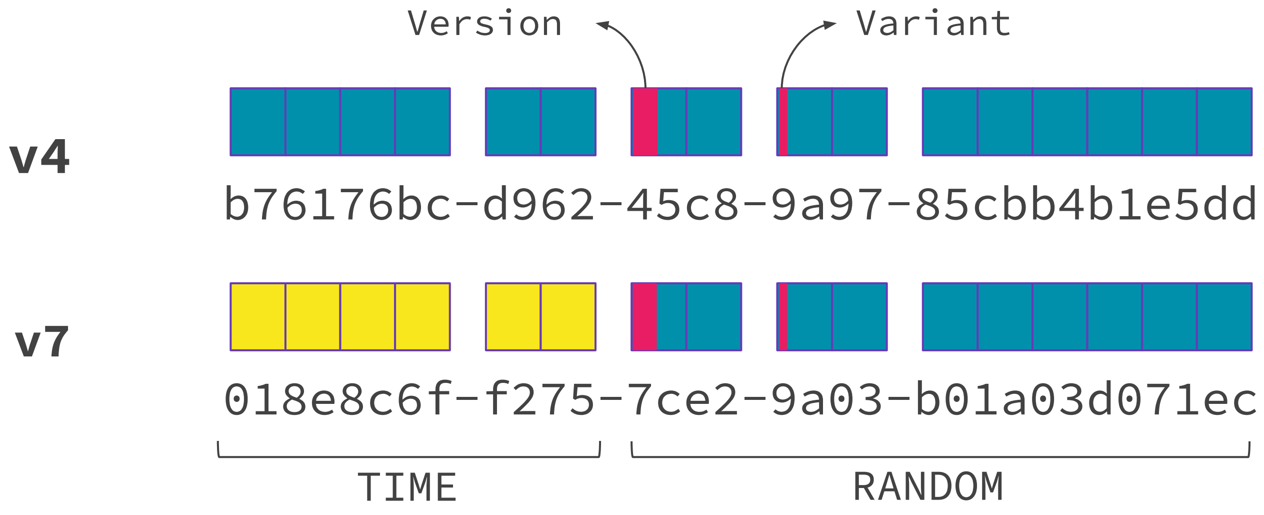
Limitations

- PostgreSQL impose d'inclure les clés de partitionnement dans les contraintes de clés primaires ou contraintes uniques



Mai 2024

La RFC 9562 décrit la version 7 d'une donnée UUID pour encoder un timestamp dans une valeur unique et lui permettre ainsi d'être triée.


Version
Variant
v4
b76176bc-d962-45c8-9a97-85cbb4b1e5dd
v7
018e8c6f-f275-7ce2-9a03-b01a03d071ec
TIME
RANDOM

Commit 78c5e14

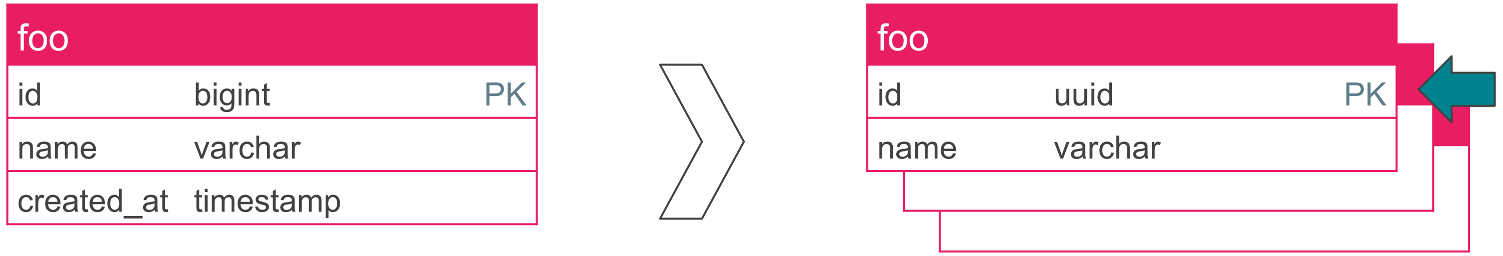
```
author    Masahiko Sawada <msawada@postgresql.org>
          Wed, 11 Dec 2024 23:54:41 +0000 (15:54 -0800)
committer Masahiko Sawada <msawada@postgresql.org>
          Wed, 11 Dec 2024 23:54:41 +0000 (15:54 -0800)
Add UUID version 7 generation function.

This commit introduces the uuidv7() SQL function, which generates UUID
version 7 as specified in RFC 9652. UUIDv7 combines a Unix timestamp
in milliseconds and random bits, offering both uniqueness and
sortability.
This commit also expands the uuid_extract_timestamp() function to
support UUID version 7.

Additionally, an alias uuidv4() is added for the existing
gen_random_uuid() SQL function to maintain consistency.
```

Solution !

- Convertir la colonne primaire en UUID v7 !



Partitionnement manuel

```
CREATE TABLE foo (
  id uuid NOT NULL DEFAULT uuidv7(),
  name varchar
) PARTITION BY RANGE (id);

CREATE TABLE foo_202506 PARTITION OF foo
  FOR VALUES FROM (uuidv7_boundary('2025-06-01'))
             TO (uuidv7_boundary('2025-07-01'));

CREATE TABLE foo_default PARTITION OF foo DEFAULT;
```

dverite/postgres-uuidv7-sql

```sql
CREATE FUNCTION uuidv7_boundary(timestamptz) RETURNS uuid
AS $$
  /* uuid fields: version=0b0111, variant=0b10 */
  select encode(
    overlay('\x00000000000070008000000000000000'::bytea
      placing substring(
        int8send(
          floor(extract(epoch from $1) * 1000)::bigint) from 3)
        from 1 for 6),
    'hex')::uuid;
$$ LANGUAGE sql stable strict parallel safe;
```

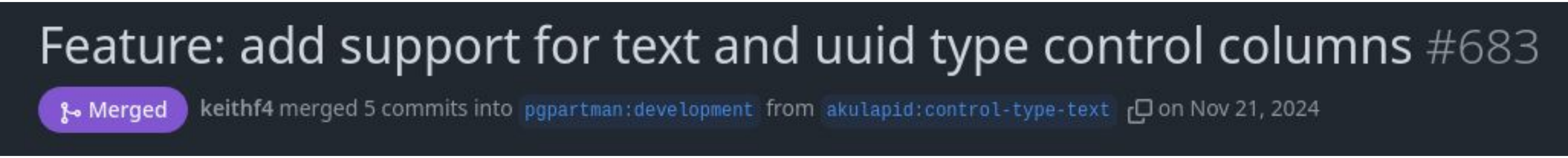
```
pgday=# \d+ foo
                Partitioned table "public.foo"
 Column |       Type        | Collation | Nullable | Default
--------+-------------------+-----------+----------+----------
 id     | uuid              |           | not null | uuidv7()
 name   | character varying |           |          |

Partition key: RANGE (id)
Partitions: foo_202506 FOR VALUES
              FROM ('0197285b-e300-7000-8000-000000000000')
                TO ('0197c2da-ab00-7000-8000-000000000000'),
            foo_default DEFAULT
```

```
pgday=# INSERT INTO foo (name) VALUES ('PG Day France 2025');
INSERT 0 1

pgday=# SELECT tableoid::regclass partname,
               *,
               uuid_extract_timestamp(id) created_at,
          FROM foo \gx

-[ RECORD 1 ]-----------------------------------
partname   | foo_202506
id         | 019738fe-c01c-7b18-b14f-48285240a19e
name       | PG Day France 2025
created_at | 2025-06-04 15:31:48.892+02
```



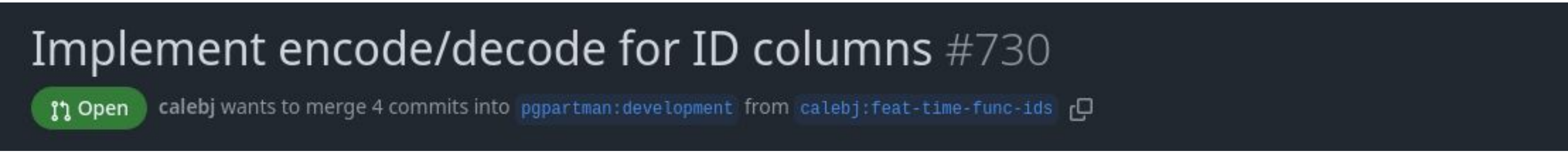
Feature: add support for text and uuid type control columns #683

Merged keithf4 merged 5 commits into pgpartman:development from akulapid:control-type-text on Nov 21, 2024

```
---

CREATE TABLE foo (
  id uuid NOT NULL DEFAULT uuidv7(),
  name varchar
) PARTITION BY RANGE (id);

SELECT partman.create_parent(
  p_parent_table := 'public.foo',
  p_control := 'id',
  p_interval := '1 month',
  p_time_encoder := 'partman.uuid7_time_encoder',
  p_time_decoder := 'partman.uuid7_time_decoder'
);
```


Implement encode/decode for ID columns #730

Open calebj wants to merge 4 commits into `pgpartman:development` from `calebj:feat-time-func-ids`

```sql
---

SELECT * FROM partman.partition_data_time(
  p_parent_table => 'public.foo'
);

-- ERROR:  Cannot run on partition set without time
           based control column or epoch flag set with
           an id column. Found control: uuid, epoch: none
-- CONTEXT:  PL/pgSQL function partman.partition_data_time()
             line 63 at RAISE
```