Engineer

Time-ordered keys for a growing data model

Adopted UUID v7 time-ordered identifiers (PostgreSQL 18) as entity keys to reduce B-tree index fragmentation and speed up queries.

2025

time-ordered keys that keep the indexes tight

Situation. Every entity needs a unique id, and the reflex choice is a random UUID. The trouble is that random ids land all over a B-tree index. Inserts scatter, the index fragments, and as tables grow both writes and range scans pay for it. On a platform meant to keep growing, that's a slow leak you'd rather not build in.

Task. Keep the global uniqueness of a UUID but drop the fragmentation that comes with the randomness.

Action. The standard became UUID v7, which is time-ordered — the leading bits are a timestamp, so new rows sort into the index instead of peppering it. PostgreSQL 18 has this natively as uuidv7(), wrapped in a small uuid_generate_v7() function so the same call behaves cleanly on Azure's Flexible Server, then made the default for entity primary keys across the schema. Nothing exotic about it; it's the kind of decision that's cheap if you make it early and a pain to retrofit later.

Result. UUID v7 became the configured key convention, giving new identifiers time ordering. Reduced fragmentation and faster queries were design aims; no before-and-after workload benchmark is recorded in this case.

Bring this experience to your project

Use "Time-ordered keys for a growing data model" as a starting point for a similar challenge in your organisation. The services behind this work are available for your project. A structure that matches how the business actually works, so the questions you need answered later are still possible to ask.

Explore our services Get in touch →

Look behind the result

Read the situation, decisions and result in this example. We can discuss which parts transfer to your constraints and which need another approach. Time-ordered keys for a growing data model — time-ordered keys that keep the indexes tight

Explore our services Get in touch →

Chasing the last bit of database performance?

Talk through your project with the engineer responsible for delivery. We agree the scope, price basis and handover in writing before work begins.

Role context: Database Engineer. It draws on Backend Engineering, Databases, PostgreSQL, Data Engineering and Performance Tuning. And while you are here, we are rather good at Database Design & Modeling and Database Performance Tuning.

Get in touch Explore our services

One piece of **a longer record** — and it is all on this site.

Every entry is written the same way: the situation, the task, what we did and what changed.

See the whole record

Define the improvement you need

Bring a representative dataset, the questions it must answer and the current bottleneck. We can agree how to measure quality, throughput and recovery.

See what it costs Get in touch →

Chasing the last bit of database performance?

Ask about the decisions behind “Time-ordered keys for a growing data model”. You will speak with the people who can explain the work and assess your own project.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/portfolio/quicker-answers-from-a-better-choice-of-key/