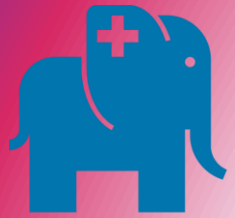


PG Day

PostgreSQL: A Reliable and Extensible Multi-Model SQL Database

Keynote by Prof. Stefan Keller, OST

Friday, 28 June 2024, 9:00 - 9:40

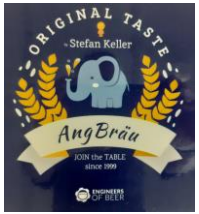
OST Campus Rapperswil (Switzerland)

About me…

Living in five bubbles

1. Family
2. …surrounded by bright staff and minds (you)
3. Databases
4. GISTech
5. OpenData (OpenStreetMap)

… and as craft mapper
and craft brewer ☺



Ntuppuppakkoranendarnnu

- "My grandfather had an elephant"
- Humorous novel in southern India in the 90ies – home to the largest Asian elephants!
- About the a girl from a poor family that once owned an elephant.
- Although she dreams of becoming rich, she marries modestly.
- TL;DR: Inner values are greater than material wealth; and true happiness lies in love and simplicity.

Speaking of "Elephants"

Michael Stonebraker used this to refer to the major traditional RDBMS vendors - arguing that these are becoming obsolete in the face of modern database technologies and workloads.

("**One Size Fits None** – Are the Database Elephants Toast?", 2015 govloop.com)

Polyglot Persistence!
(Martin Fowler)

Left: David.Monniaux CC BY-SA 4.0 | Right: govloop.com

What is a Multi-Model Database (MMDB)?

A DBMS "*… designed to support multiple data models against a single, integrated backend.*"
– Wikipedia

A DBMS that contains multiple data models in a single, integrated backend supported by a common environment.
– Own definition

MMDB - A term first introduced by Luca Garulli, founder of OrientDB, during a keynote speech at a NoSQL conference 2012.

MMDB vs. Polystores

Polystores

Using jointly multiple data storage technologies, chosen based upon the way data is being used by individual applications.

(Lu, J., Holubová, I., & Cautis, B. (2018). Multi-model databases and tightly integrated polystores. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management. "CIKM 2018 Tutorial" Univ. Helsinki)

Don't confuse Multi-model with Multi-modal (image, audio, signals)

I'd like to change the way we describe…

THURSDAY 9 AUGUST 2012

PostgreSQL: The Multi-Model Database Server

I'd like to change the way we describe PostgreSQL.

Calling PostgreSQL and Object Relational database is misleading and years out of date. Yes, PostgreSQL is Relational and the project follows the SQL Standard very closely, but that's not all it does.

PostgreSQL supports all of the following:

* Relational
* Object Relational
* Nested Relational (record types)
* Array Store
* Key-Value Store (hstore)
* Document Store (XML, JSON)

and 9.2 adds

* Range Types

So what do we call it?

Simon Riggs
(† March 2024)

"Peanut Butter and Chocolate"

Tuesday, October 17, 2023

Postgres's non-relational data type support has greatly increased its adoption in recent years. The use cases I have heard is that Postgres supports relational data, and it supports non-relational data like JSON. This ability is often called multi-model.

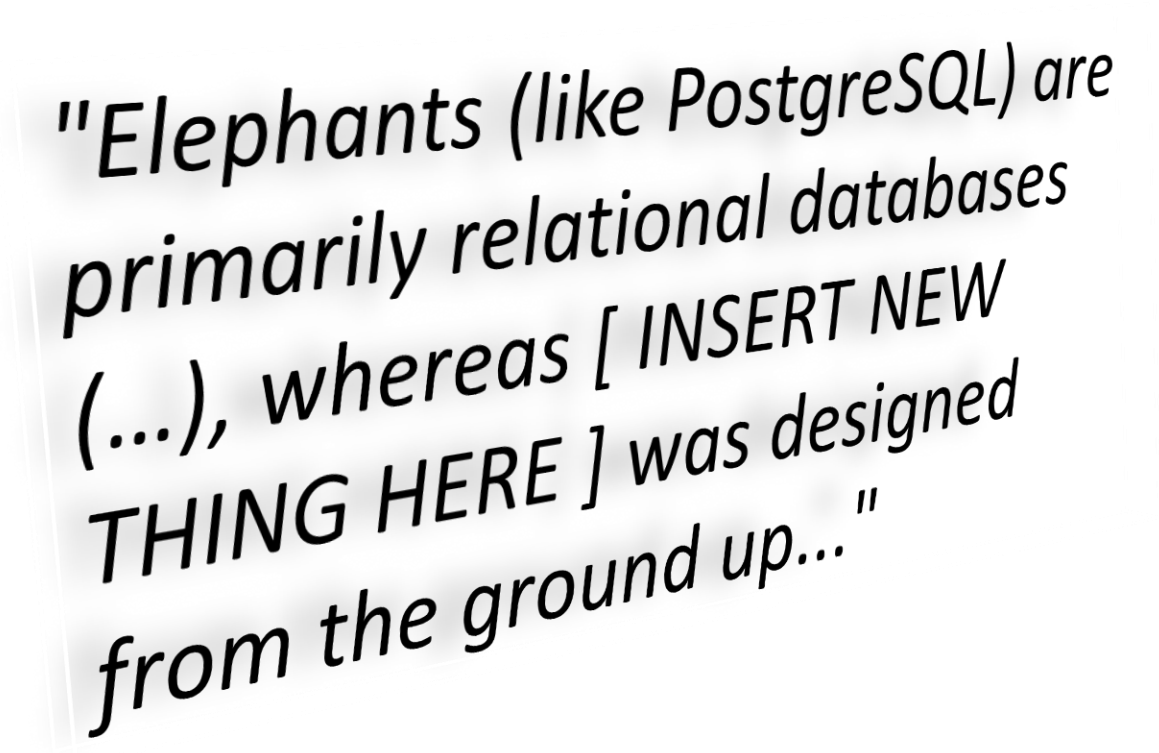
However, I am not sure if that is the right focus — it highlights Postgres's ability to store *two* types of data workloads in the same system, but it doesn't highlight the power of supporting data that can be seamlessly organized using both systems — let me explain.

People normally say, "Oh, I can store my financial data using Postgres's relational data types, and I can store my browser data in JSONB." However, what about storing most of your financial data using relational, and storing the financial data that doesn't fit cleanly in relational in JSONB, full text search, or PostGIS? And some of your browser data should probably be pulled out of JSONB and stored using relational data types for better consistency and performance.

In the 1970's, there was a marketing campaign for Reese's Peanut Butter Cups that had peanut butter and chocolate eaters colliding, creating a taste that was better than the two individual flavors. I think Postgres multi-model capability fits that *better-together* idea as well.

by Bruce Momjian:
Postgres Blog

Only Relational?



Only Relational? Only SQL?

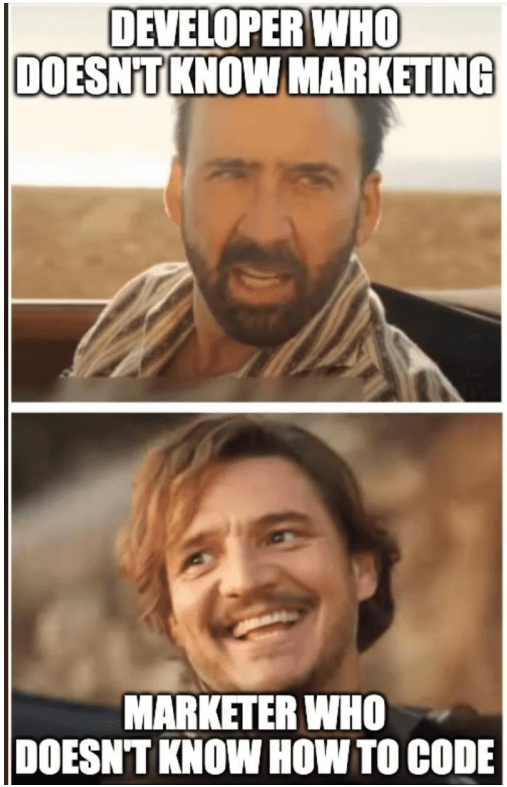
- Let's not make an artificial contrast between "general-purpose" and "purpose-built", "native", "cloud-first", "from the ground up"
- Fact is:
  - *Relational model dominates since years, growing from 70%. (DB-Engines.com)*
  - *"SQL is there to stay." (Andy Pavlo)*
  - *"Since 1999, SQL is not limited to the relational model anymore. Back then ISO/IEC 9075 (the "SQL standard") added arrays, objects and recursive queries. (…) relational SQL is only about 20% of modern SQL."* (Markus Winand, modern-sql.com)

"The great debate ..."

- About every ten years or so, there is a “**great debate**” between, on the one hand, those who see the problem of data modelling through a more or less relational lens, and on the other hand, a noisier set who have a hot new thing to promote. The debate usually goes like this:
- **Refuseniks**: *"You relational people with your flat tables and silly SQL. You are so unhip! You simply cannot deal with the problem of [INSERT NEW THING HERE]".*
- **R-people**: *You make some good points. But there is an enormous amount of money (effort) invested in building scalable, efficient and reliable database management products and no one is going to drop all of that on the floor. We plan to incorporate the best of these ideas into our products.* (Source: Paul Brown via Akmal Chaudhri)

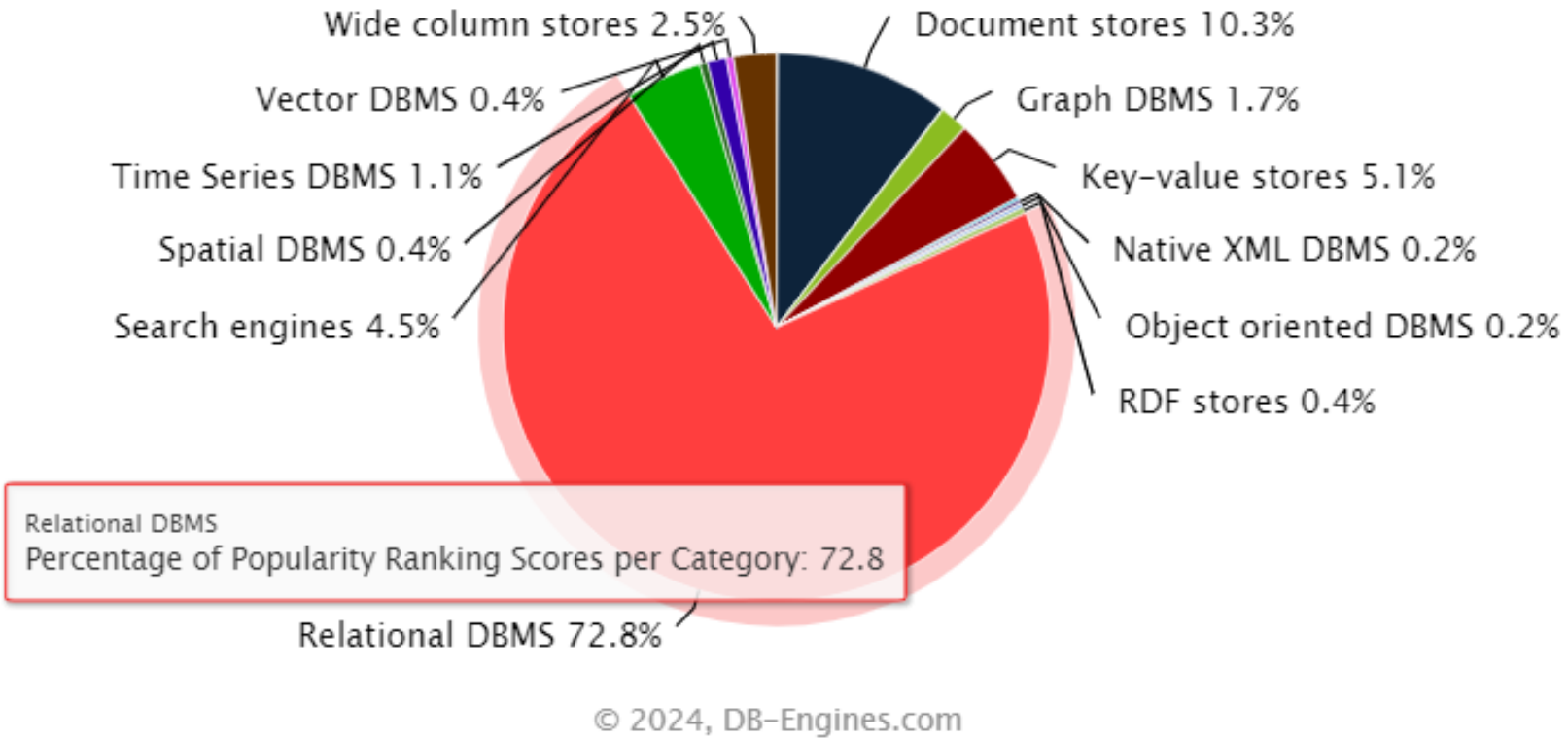
Why?

VC backed marketing:



And we're techies:

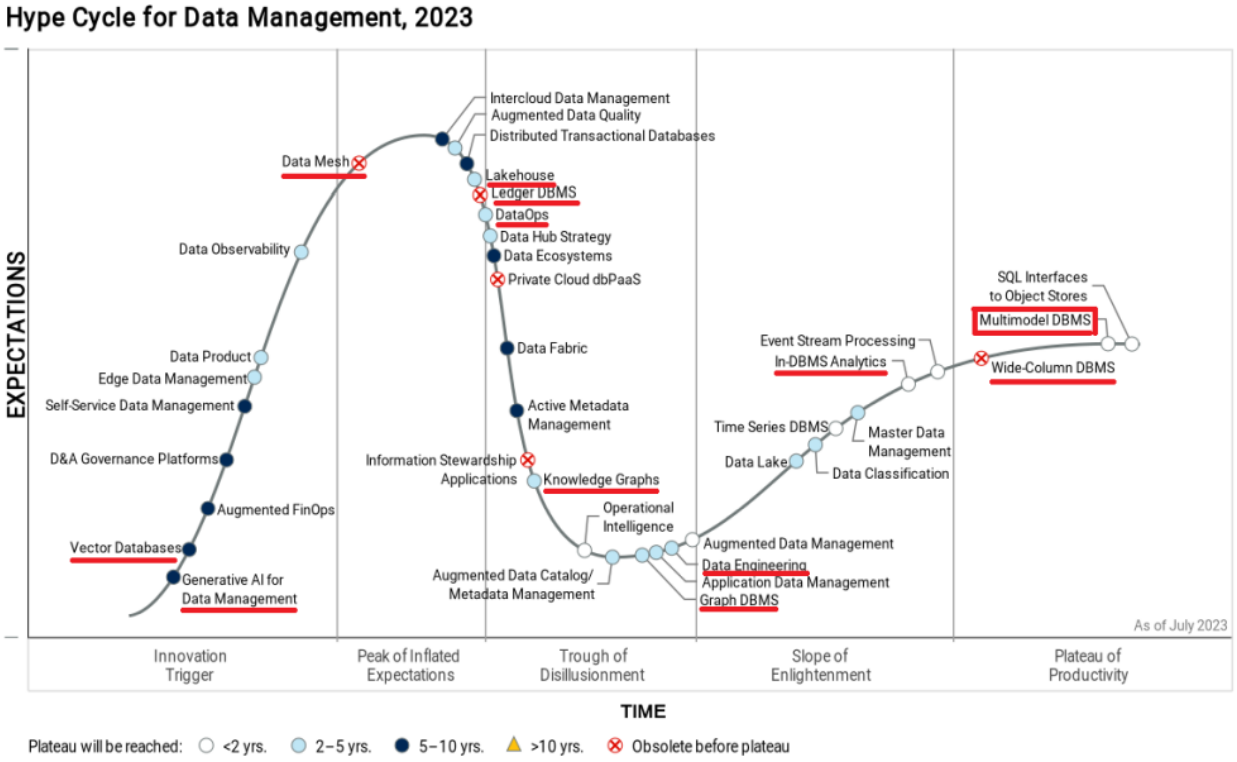
DB-Engines Ranking



Gartner

Plateau of Productivity:

- SQL for ...
- Multimodel, also once a specialized niche within the database market, has become the standard for almost all databases. While different databases continue to have different levels of functionality and performance for different models, “relational model only” is now an exception rather than the majority of the market.



From Data Types to Models/DB Categories

Data Types

- Basic data types
  - Numbers, Range, String / Text
  - Date/Time/Interval
- Aggregate (flat) data types
  - Array, Dictionary, "Union"
- Composite data types
  - Nested (record types)
  - Tree (incl. JSON, XML formats)
  - Graph
- Binary
- Geometry data types
- User Defined
- "Anything" data type

Models / DB Categories

- Basic data types
  - **Relational DBMS, Search engines**
  - **Relational DBMS, Time series DBMS**
- Aggregate (flat) data types
  - **Array stores, Key-Value stores**
- Composite data types
  - **Object-relational**
  - **Tree stores, Document stores?**
  - **Graph DBMS**
- **Relational**
- **Spatial DBMS**
- **Object-relational**
- **Document stores?**

DB-Engines.com (DB-E) enhanced

- The "usual NoSQL babbling":
  - Relational DBMS (DB-E)
  - Key-value storesc
  - Document stores (DB-E)
  - Graph DBMS (property graph, DB-E)
  - Wide column stores (DB-E)

- The "lesser known":
  - Array Stores (Keller), Multivalue DBMS (DB-E)
  - Tree stores (Keller), Navigational DBMS (DB-E)
  - Spatial DBMS (DB-E)

- The "other":
  - Search engines / Fulltext (DB-E)
  - RDF stores (DB-E)
  - Object oriented DBMS (DB-E)

- The "implementation oriented":
  - *(Wide column stores (DB-E) again)*
  - Vector DBMS (DB-E)
  - Time series DBMS (DB-E)
  - Native XML DBMS (DB-E)
  - Event stores (DB-E)
  - Content stores (DB-E)
  - In-Memory / Cache (Keller)
  - Immutable stores (Keller)

What is a database anyway?

- Database / DBMS vs. Store?

- DBMS → RDBMS → Codd's 12 rules (actually 13) to define what is required from a database management system to be a relational database management system (RDBMS)
  - SQLite?

The reliable and extensible database…

- The reliability
  - What to say here?

- The extensibility
  - UDF
  - Extensions like hstore and PostGIS
  - PGXN
  - …

My beloved PostgreSQL

Operators, e.g. contains @> – Ascii art of 🫵

- Range: SELECT int4range(10, 20) @> 11; → t
- Arrays: SELECT ARRAY[1,4,3] @> ARRAY[3,1,3]; → t
- hstore: SELECT 'a=>b, b=>1, c=>NULL'::hstore @> 'b=>1'; → t
- JSONB: SELECT '{"a":1, "b":2}'::jsonb @> '{"b":2}'::jsonb; → t
- Graph: SELECT …?

My small wishes for PostgreSQL...

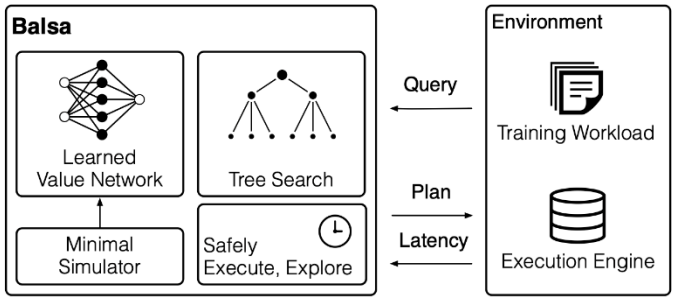
- sort arrays
- Pivot
- temporal tables
- more time series functionality
- …

My 2 big wishes for PostgreSQL...

- Columnar storage and parallelization
  - Read CSV and Parquet files (in parallel like DuckDB)
  - Built-in Columnar storage
  - Vectorized query execution (vs. "Volcano" strategy)

My 2 big wishes for PostgreSQL...

- Autotuning
  - "Human-in-the loop"
  - then automated database tuning
- Useful, promising and hype
- Academic examples (Open Source):
  - DB-BERT - Extracts hints for database parameter settings from text via natural language analysis (Univ. Cornell, Python)
  - BALSA - Learns to optimize SQL queries by trial-and-error using deep reinforcement learning and sim-to-real learning (Univ. Berkeley, Python)
  - LEON - An ML-based framework that aids the an expert cost-based query optimizer (Huawei, C)



https://db-engines.com/de/blog_post/106, 2. January 2024

We are MMDB!

And "No", the database Elephants aren't toast:

"One Size Fits Most"!

Contact: Prof. Stefan Keller, stefan.keller@ost.ch, www.ost.ch/ifs, Linkedin