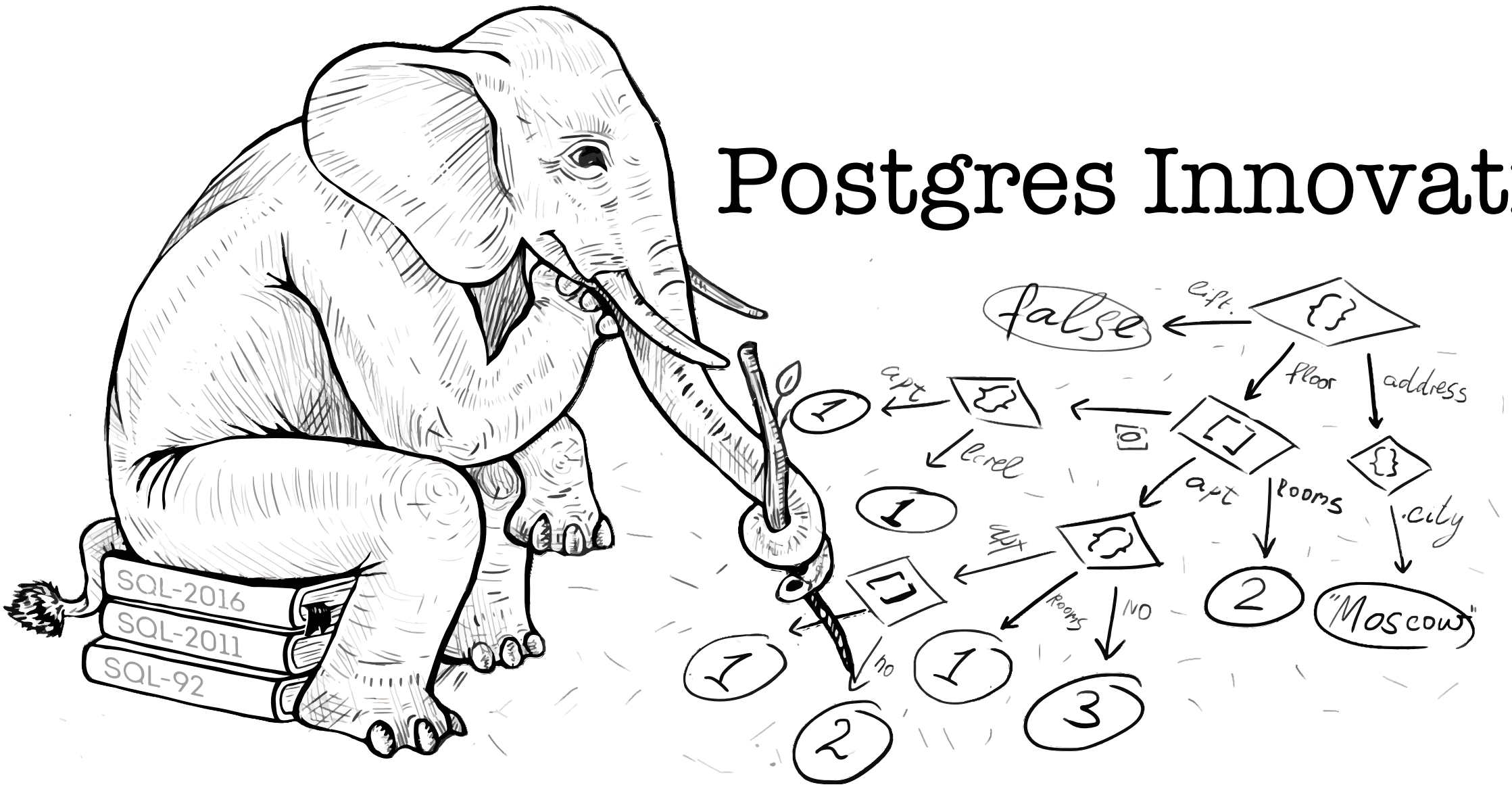
Postgres Innovations



Oleg Bartunov,
Moscow University, Postgres Professional

NP PUG PGConf.Nepal, May 11, 2023

Postgres developer/contributor since 1995

OLEG BARTUNOV

PROFESSIONAL Postgres

The first PostgreSQL talk in Nepal, May 4, 2009

wiki.fossnepal.org/index.php?title=FOSS_Ka_Kura

Search

May 4th 2009	**Introduction to PostgreSQL and talk on Full-text search capabilities in PostgreSQL**	Oleg Bartunov (/ http://www.sai.msu.su /~magera/	M.Sc. Hall, Dept. of Electronics and Computer Engineering, IOE

Kerosene was used to power my notebook and projector

I gave a talk about full text search in postgresql after trek. We made special support of nepalese locale to make it works right. There was no light in the university, so we used generator to power notebok and projector.It was really funny to speak for people sitting in the darkness.

```
commit e43bb5beb78bef14f012279a730c3d1914db7e83
Author: Teodor Sigaev <teodor@sigaev.ru>
Date:   Wed Mar 11 16:03:40 2009 +0000

    Some languages have symbols with zero display's width or/and vowels/signs which
    are not an alphabetic character although they are not word-breakers too.
    So, treat them as part of word.

    Per off-list discussion with Dibyendra Hyoju <dibyendra@gmail.com> and
    and Bal Krishna Bal <balkrishna7bal@gmail.com> about Nepali language and
    Devanagari alphabet.
```

PostgresPro

PostgreSQL has built-in Nepali support since 2018

PGConf Nepal 2018
May 04- 05, 2018

Nepali NLP Group

Nepali Support for Full Text Search in PostgreSQL

- Ingroj Shrestha

Team Lead, NLP R&D Engineer and Tech Blogger
Nepali NLP Group

PostgresPro

How to choose a right database ?

- **People usually choose a database looking on**
  - Functionality, Performance
  - Availability - License, price
  - Local expertise, Personal experience
  - Compatibility to existing environment
  - Support
- **After project started**
  - Need new functionality, Better performance
- **Project is in production, no way to change database**
  - Starting to use various ugly «solutions»
  - System works, but looks pretty strange

PostgresPro

If you chose a wrong database

System works, but looks pretty strange

PostgresPro

Database should be **Extensible** !

To adapt to

- a new types of DATA,
- new QUERIES,
- VOLUME of data growth,
- VELOCITY of data processing
- New environment

PostgresPro

Database Extensibility — Postgres Innovation !

"The main design goals of the new system are to:
2) provide user extendibility for data types, operators and access methods,"

Stonebraker M., Rowe L. A.
The design of Postgres. ACM, 1986. – Т. 15. – №. 2. – С. 340-355.

"It is imperative that a user be able to construct new access methods to provide efficient access to instances of nontraditional base types"

Michael Stonebraker, Jeff Anton, Michael Hirohama.

Extendability in POSTGRES , IEEE Data Eng. Bull. 10 (2) pp.16-23, 1987

PostgresPro

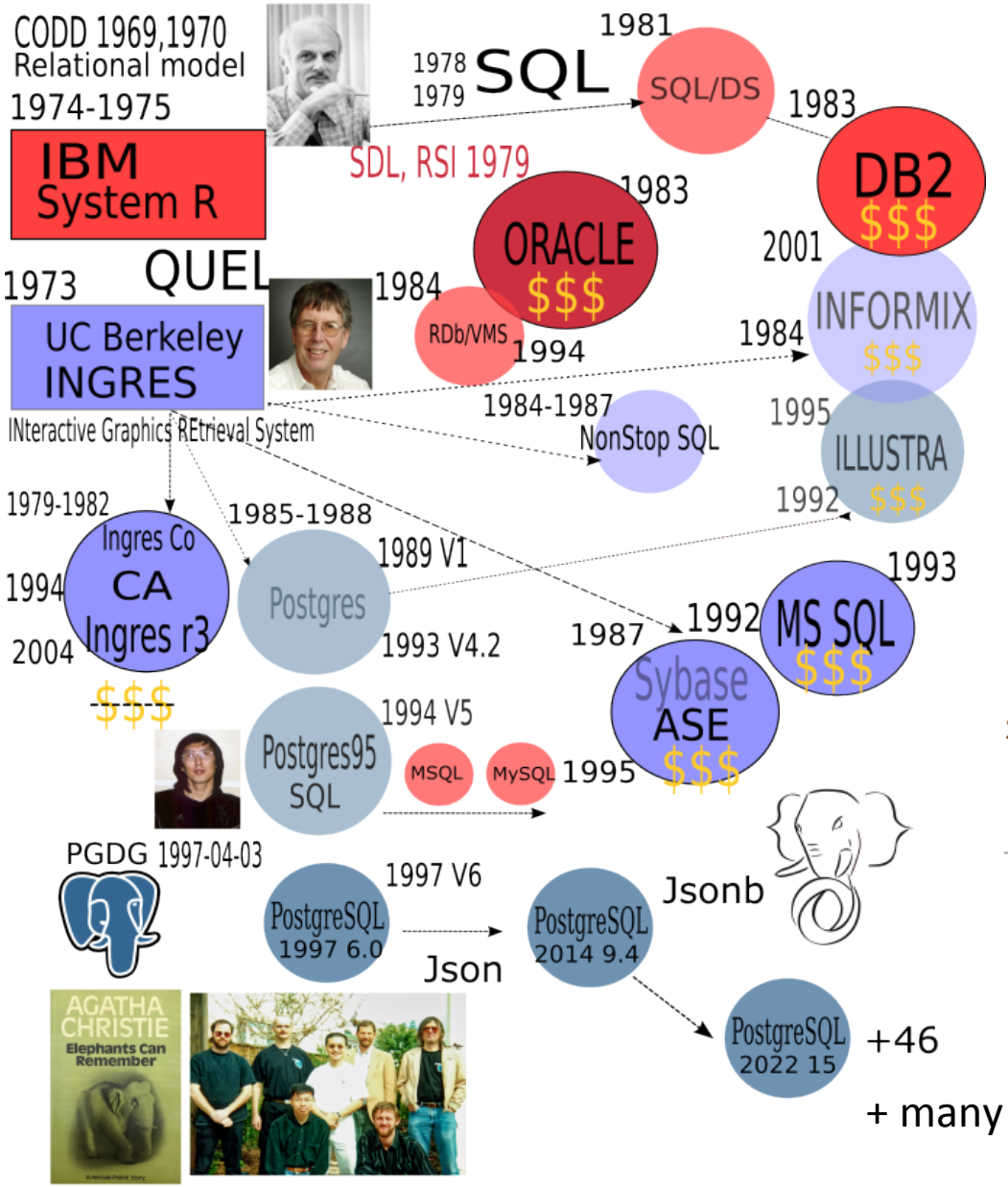
PostgreSQL Universal Database

- Any project (startup) could start with PostgreSQL
- PostgreSQL is a reliable and stable database with rich functionality and long history
- PostgreSQL has liberal BSD license, cross platform (~30)
- Developed by international community, no vendor lock
- Postgres inspire people to build new databaases, We love Forks !

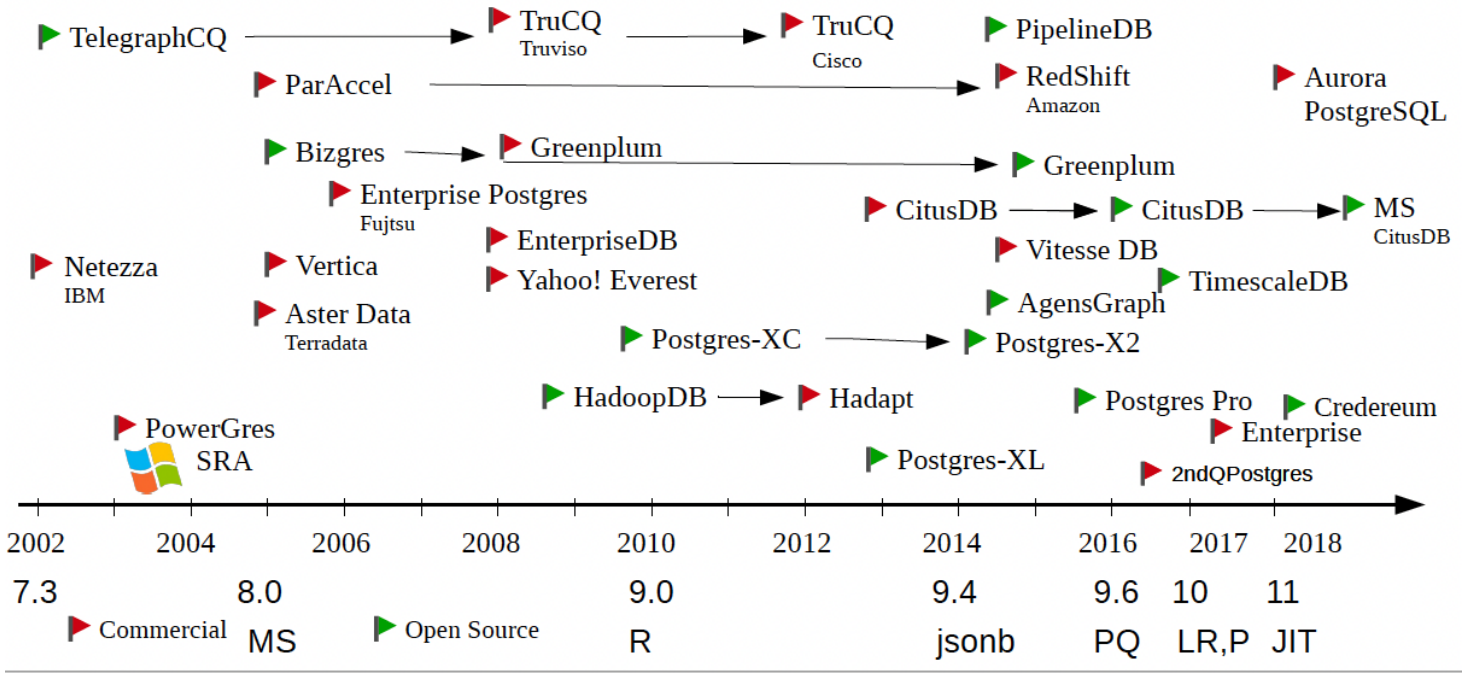
- PostgreSQL is **EXTENSIBLE**, this is the very important feature, which people miss ! It allow database to support
  - New workloads
  - New functionality
  - New environment
  - Often without restarting a server, no need core developers

PostgresPro

PostgreSQL in Database World



PostgreSQL: OLTP, MPP, OLAP, CLOUD, GIS, STREAM, TIMESERIES, GPU, BLOCKCHAIN



+ many university projects !

Professional Postgres

- Academic Postgres (x10)
- Community Postgres95 (<400)
- PostgreSQL V6

  Community develops for Community
- 200X — First Postgres-centric companies

  (GreatBridge, 2ndQuadrant, EDB...)

  +Full-time developers for Community
  - First enterprise forks

PostgresPro

Professional Postgres

- 2010 — Enterprise customers

  New features for Enterprises
- 2015 — Majority of major developers were hired by PG-companies (+Citus Data, +Postgres Professional)
  - Now the companies drive the development
  - Community: test, approve
  - Postgres became Enterprise ready (More forks)
  - Postgres became Professional

PostgresPro

PG-companies drive the development

PG-companies - proxy between Enterprise and Community

- Big enterprises require additional features "right now"
- PG-companies develop, support and test these features in their forks
- Some features returned back to community (not easy)
- Community accept (if) and support code

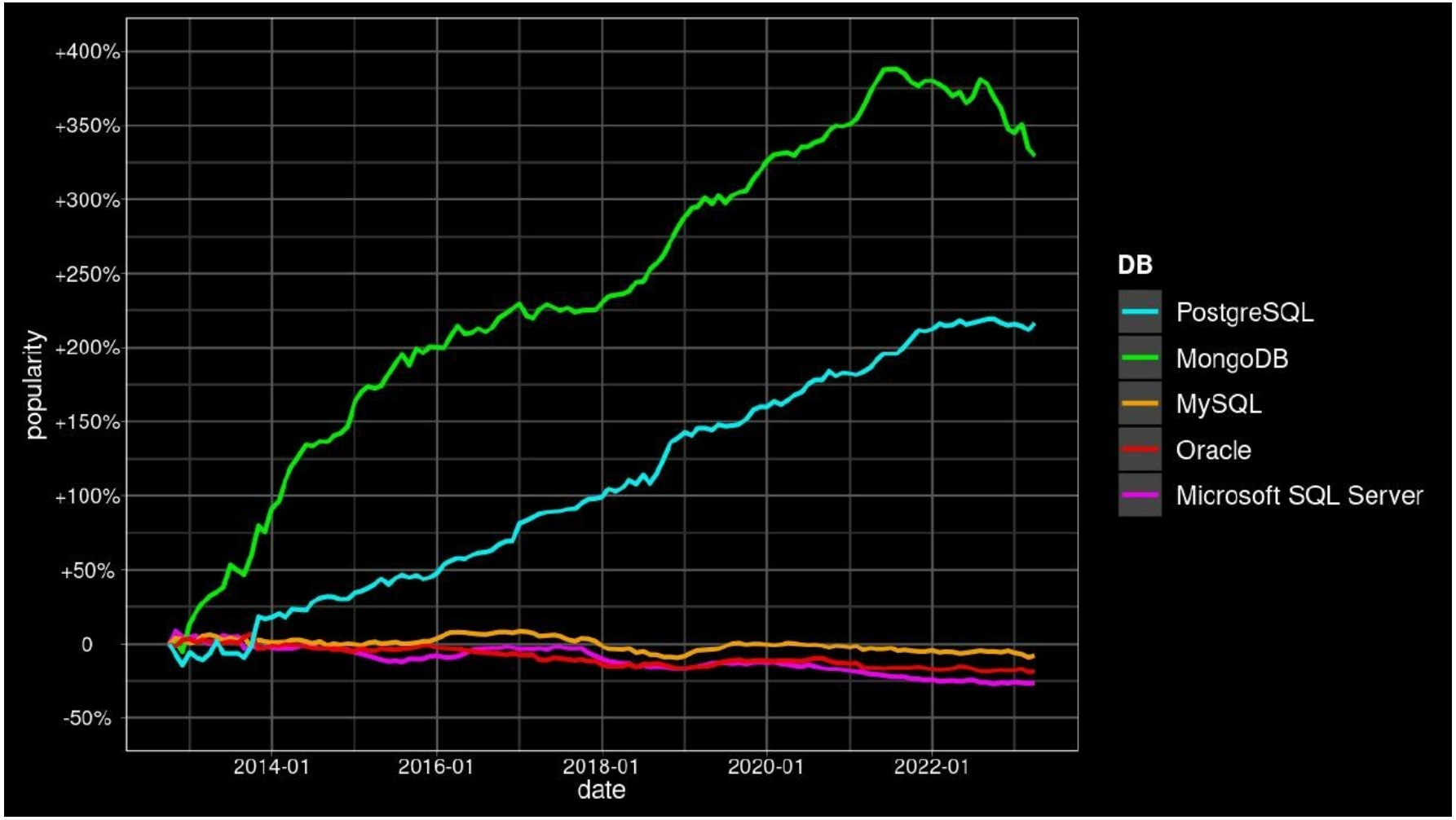
PostgresPro

Example: Postgres Pro Enterprise

- 64-bit XID (enterprise, PG17 ?)
- Adaptive Query Optimization (enterprise, opensource)
- CFS — page level compression (enterprise)
- Multi Master cluster (enterprise, opensource)
- Probackup - Incremental backup (enterprise, opensource)
- Advanced partitioning (opensource)
- Threaded Postgres (prototype)
- Built-in HA, Sharding (in development)
- SQL/JSON (PG12, PG16)
- Pluggable TOAST, Jsonb on steroids (PG17 ?)
- Check https://github.com/postgrespro

PostgresPro

Postgres Popularity is growing !

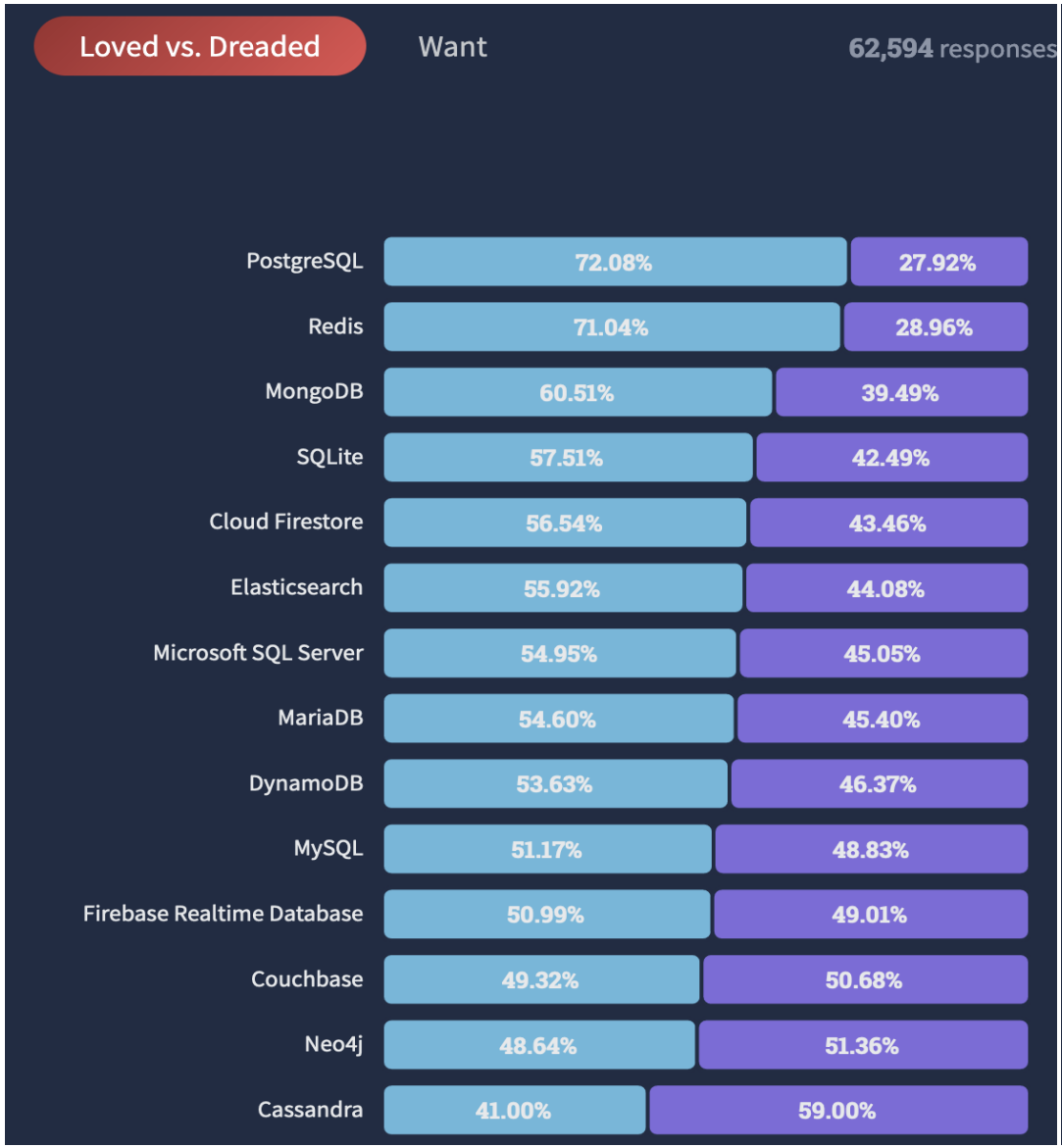


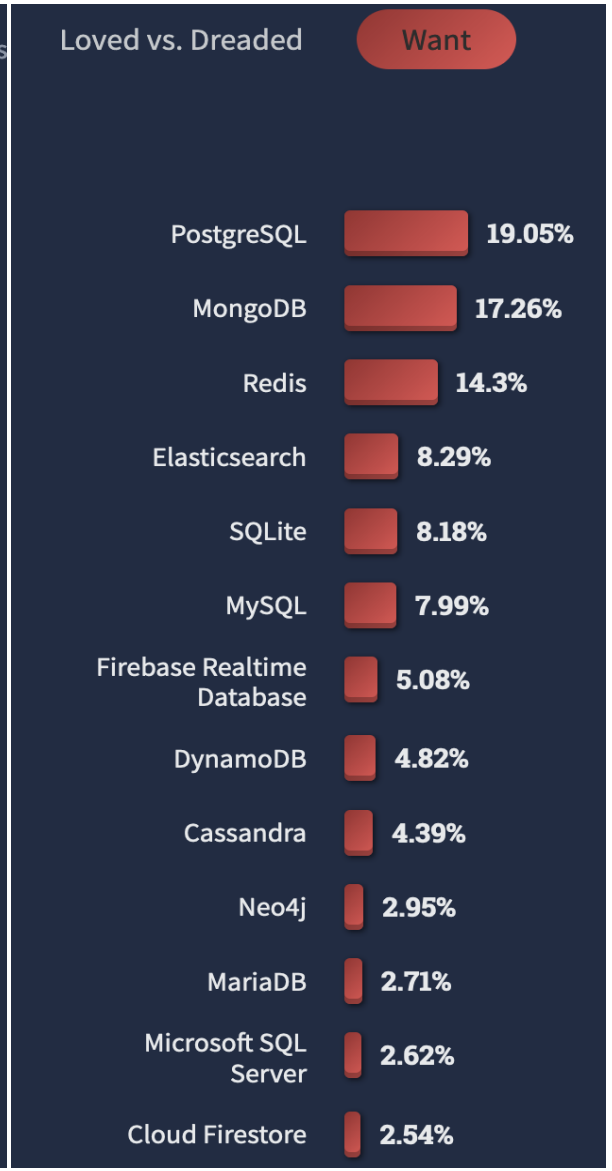
Database Popularity Trend — Db-engines.com

Postgres - database of the Year: 2017, 2018, 2020

PostgresPro

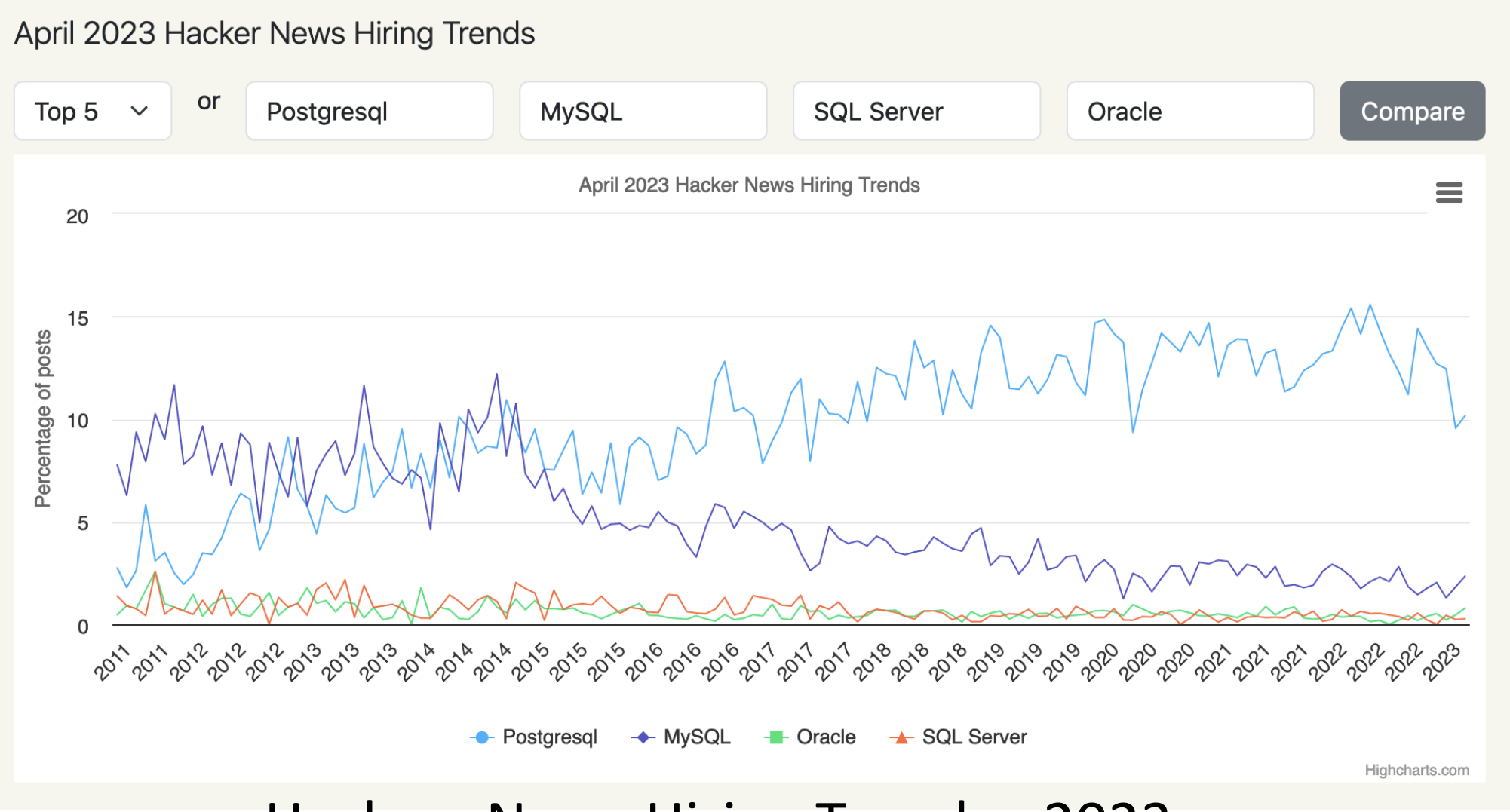
Stack Overflow Developer Survey 2022





PostgresPro

Postgres expert will always have a job !



Hackers News Hiring Trends - 2023

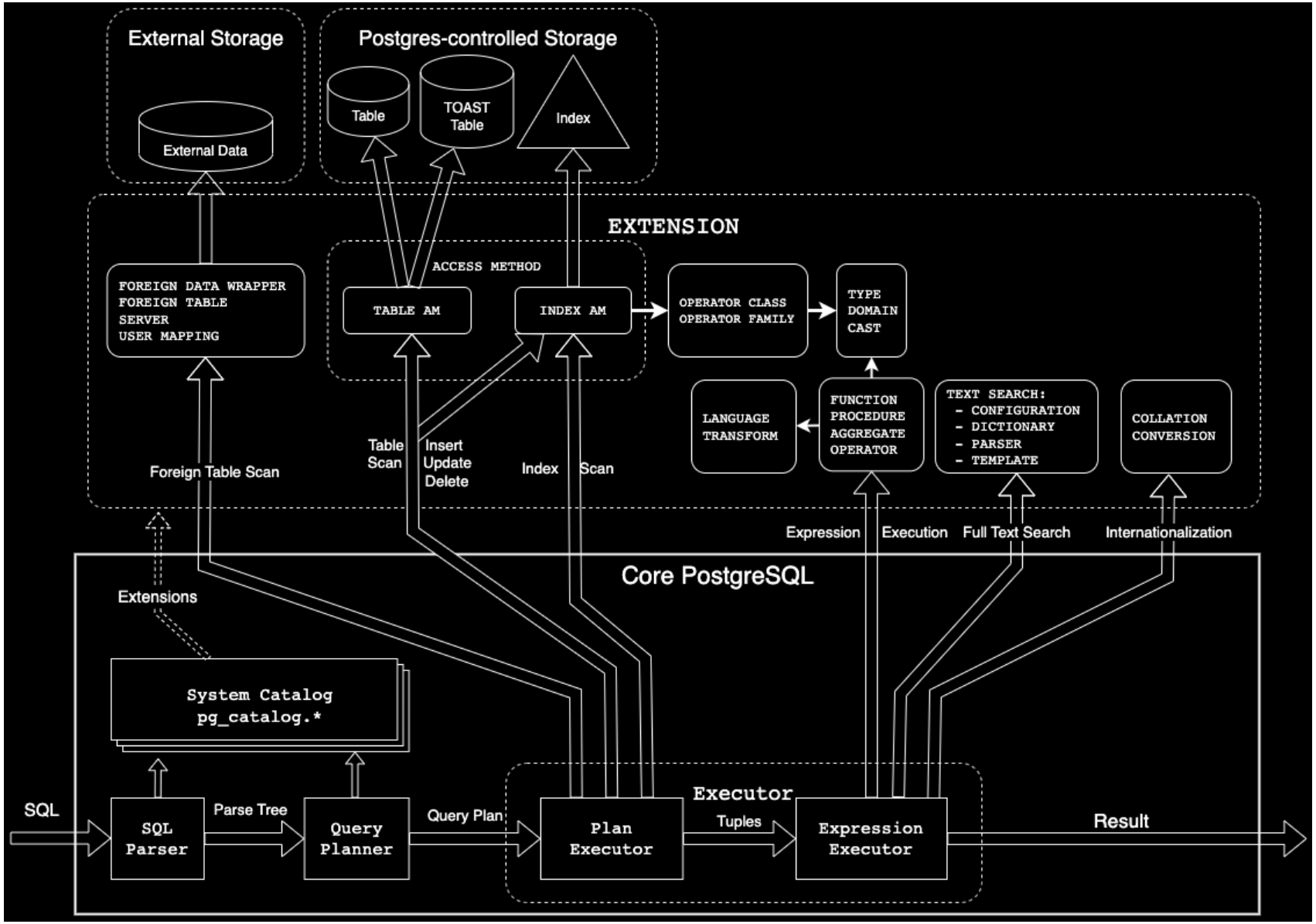
PostgresPro

My experience with Postgres Extensibility

- 1996: Start using Postgres on Web, no 8-bit support — introduced i18n (locale)
- 1999: World's top-5 portal. We start with PostgreSQL 6.5.?
  Hardware ~ my smartphone to support > 1 mln. users/day, quickly run out of resources
  - Denormalize, use arrays -> slow -> improve GiST - contrib/intarray — GiST/GIN indexes
  - Need FTS, made contrib/tsearch2 using intarray and GiST indexes
- Need fast search on hierarchical data — contrib/ltree — GiST indexes
- Need flexible schema — contrib/hstore — GiST index
- Need faster FTS — GIN index for tsearch, hstore
- Need misprint search — contrib/pg_trgm — GiST/GIN indexes
- Need Indexing the Sky — pgsphere, Q3C
- NoSQL Postgres - better/binary json - jsonb — GIN index
- Need faster FTS — RUM access methods
- SQL-2016 standard — Jsonpath, SQL/JSON
- Need faster JSONB — working in TOAST extensibility, Jsonb TOASTER
- 2023: STILL USING and Developing Postgres !

PostgresPro

Postgres Extensibility: CORE-APIs-Extensions

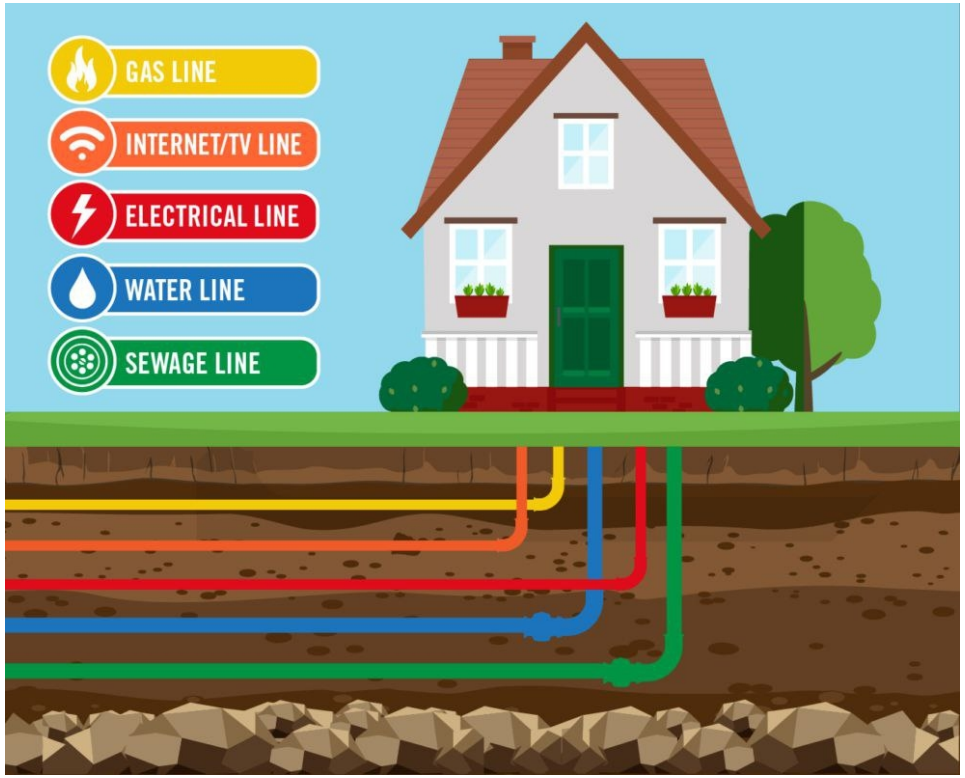


PostgresPro

Nested Extensibility

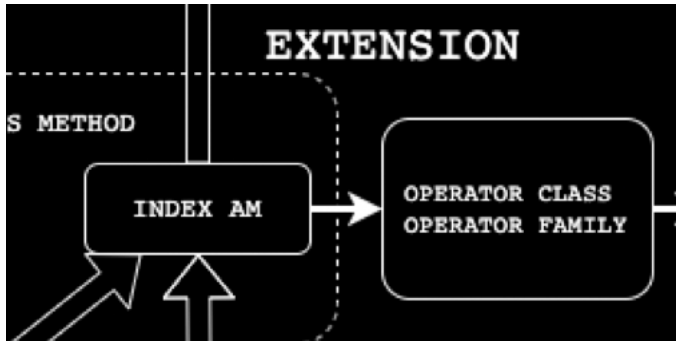
- CITY provides **CORE infrastructure** (expensive) for construction company to build an appartment building
- Construction company provides **building infrastructure** (elevator, garbage collection, cleaning ….) to apartment owner
- Apartment owner just implements his own design
- Improvements in infrastructure become available to all apartment owners

- Alternative: Build your house himself and take over all the work



PostgresPro

Postgres Extensibility: Nested API example - GIN



GIN stands for Generalized Inverted Index
(Bartunov, Sigaev, 2006)

Navigation, insert/delete,
concurrency,recovery

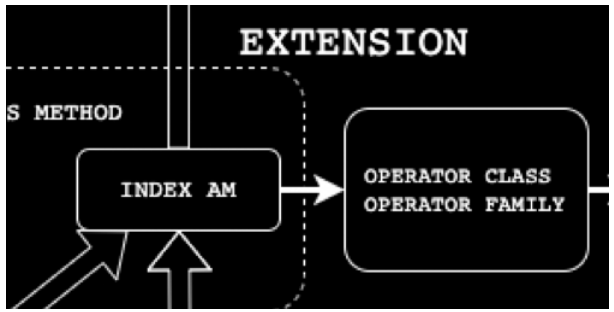
GIN

Full-Text Search	JSONB	ARRAYS (data types)	……..
tsvector_ops	jsonb_ops	array_ops	opclass
	jsonb_hash_ops		opclass
……	……	……	……

Everything could be implemented in Extension(s) !

PostgresPro

Postgres Extensibility: Nested API — GIN (FTS)



GIN for FTS: data type *tsvector*

- Words organized as B-tree
- Each word has TIDs organized as B-tree or List

Report Index

A

abrasives, 27
acceleration measurement, 58
accelerometers, 5, 10, 25, 28, 30, 36, 58, 59, 61, 73, 74
actuators, 4, 37, 46, 49
adaptive Kalman filters, 60, 61
adhesion, 63, 64
adhesive bonding, 15
assembling, 22
atomic force microscopy, 13, 27, 35
atomic layer deposition, 15
attitude control, 60, 61
attitude measurement, 59, 61
automatic test equipment, 71
automatic testing, 24

B

backward wave oscillators, 45

compensation, 30, 68
compressive strength, 54
compressors, 29
computational fluid dynamics, 23, 29
computer games, 56
concurrent engineering, 14
contact resistance, 47, 66
convertors, 22
coplanar waveguide components, 40
Couette flow, 21

dielectric relaxation, 64
dielectric thin films, 16
differential amplifiers, 28
diffraction gratings, 68
discrete wavelet transforms, 72
displacement measurement, 11
display devices, 56
distributed feedback lasers, 38

E

QUERY: compensation accelerometers

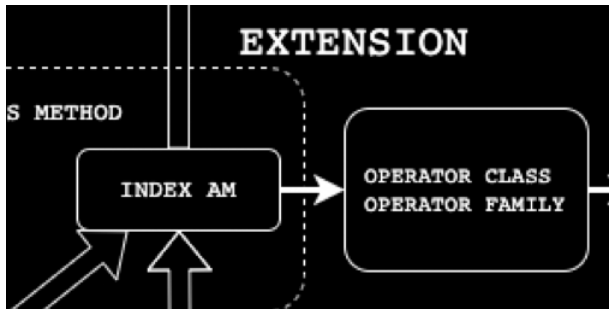
INDEX: accelerometers 5,10,25,28,**30**,36,58,59,61,73,74

compensation **30**,68

RESULT: **30**

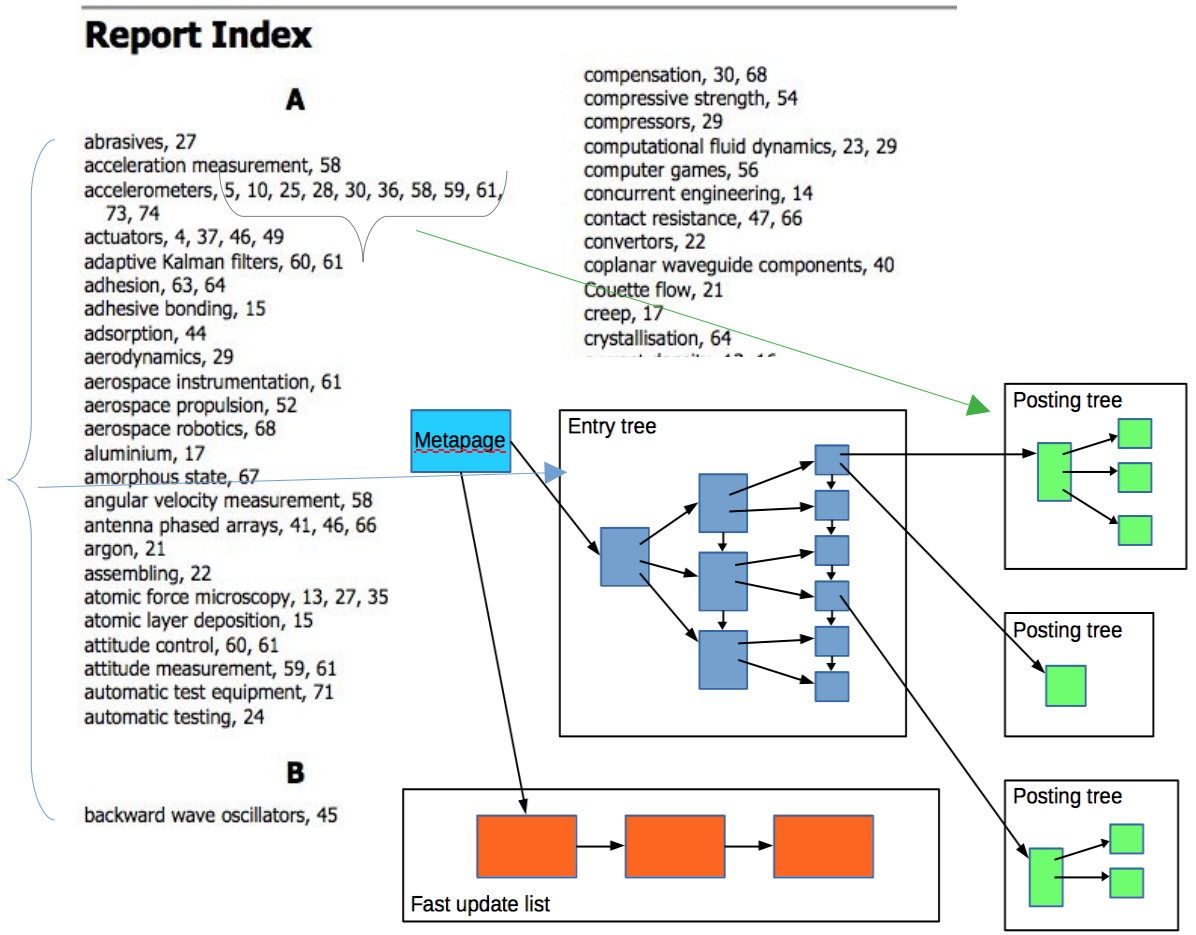
PostgresPro

Postgres Extensibility: Nested API - GIN (FTS)



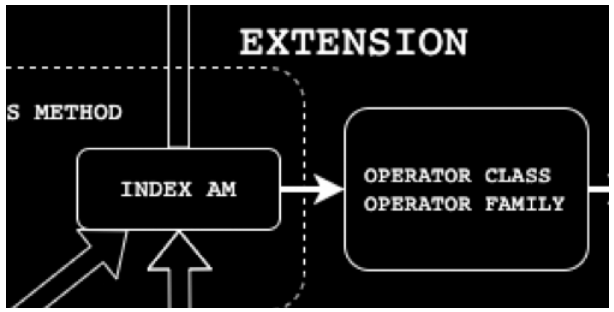
GIN for FTS: data type *tsvector*

- Words organized as B-tree
- Each word has TIDs organized as B-tree or List



PostgresPro

Postgres Extensibility: Nested API (GIN AM)



GIN for JSONB

Sample jsonb: {"k1": "v1", "k2": ["v2", "v3"]}

- **jsonb_ops** - default GIN opclass for jsonb) extracts keys,values "k1", "k2", "v1", "v2", "v3"
  Supports top-level key-exists operators ?, ?& and ?| , contains @> operator
  Overlapping of large postings might be slow
- **jsonb_hash_ops** extracts hashes of paths: hash("k1"."v1"), hash("k2".#."v2"), hash("k2".#."v3")
  Supports only contains @> operator
  Much faster and smaller than default opclass (for @>)
- Extension jsquery - **jsonb_path_value_ops, jsonb_value_path_ops, jsonb_laxpath_value_ops**

PostgresPro

Extensible (Pluggable) TOAST

TOAST allows the database to handle large column values that would not fit in a single database block. TOAST breaks up wide field values into smaller pieces, which are stored "out of line" in a TOAST table associated with the user table.

PostgresPro

The Curse of TOAST: Unpredictable performance

```
=# EXPLAIN(ANALYZE, BUFFERS) SELECT jb->'id' FROM test;
                QUERY PLAN
-------------------------------------------------
Seq Scan on test   (actual rows=10000 loops=1)
   Buffers: shared hit=2500
 Planning Time: 0.050 ms
 Execution Time: 6.147 ms
(4 rows)
```

```
CREATE TABLE test (id int, jb jsonb);
ALTER TABLE test ALTER COLUMN jb SET STORAGE EXTERNAL;
INSERT INTO test
SELECT
  i id,
  jsonb_build_object(
    'id', i,
    'foo', (SELECT jsonb_agg(0)
            FROM generate_series(1, 1960/12))
  ) jb   -- [0,0,0, ...]
FROM
  generate_series(1, 10000) i;
```

Small update cause <u>significant slowdown !</u>

```
=# UPDATE test SET jb = jb || '{"bar": "baz"}';
=# VACUUM FULL test; -- remove old versions
```

```
=# EXPLAIN (ANALYZE, BUFFERS) SELECT jb->'id' FROM test;
           QUERY PLAN
-----------------------------------------------
  Seq Scan on test (actual rows=10000 loops=1)
   Buffers: shared hit=30064
 Planning Time: 0.105 ms
 Execution Time: 38.719 ms
(4 rows)
```

```
Pageinspect: 64 pages with 157 tuples per page

WHY 30064 pages !!!!
```

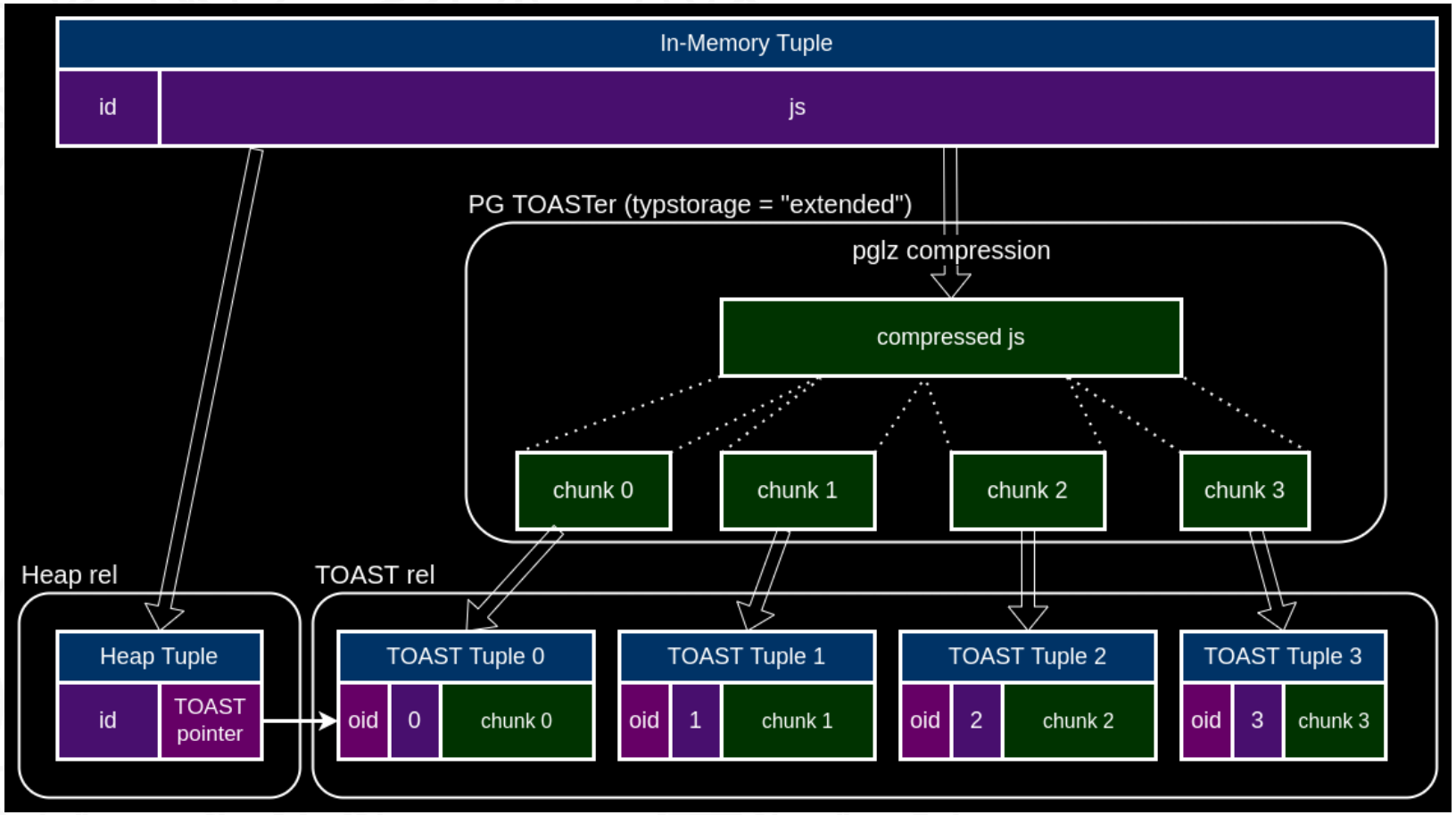
PostgresPro

TOAST Explained

The Oversized-Attribute Storage Technique

TOASTed (large field) values are compressed, then splitted into the fixed-size TOAST chunks (1996B for 8KB page)

- TOAST chunks (along with generated Oid `chunk_id` and sequence number `chunk_seq`) stored in special TOAST relation `pg_toast.pg_toast_relid`, created for each table with TOASTed attributes.
- TOASTed attribute in the original heap tuple is replaced with TOAST pointer (18 bytes) containing `chunk_id`, `toast_relid`, `raw_size`, `compressed_size`.

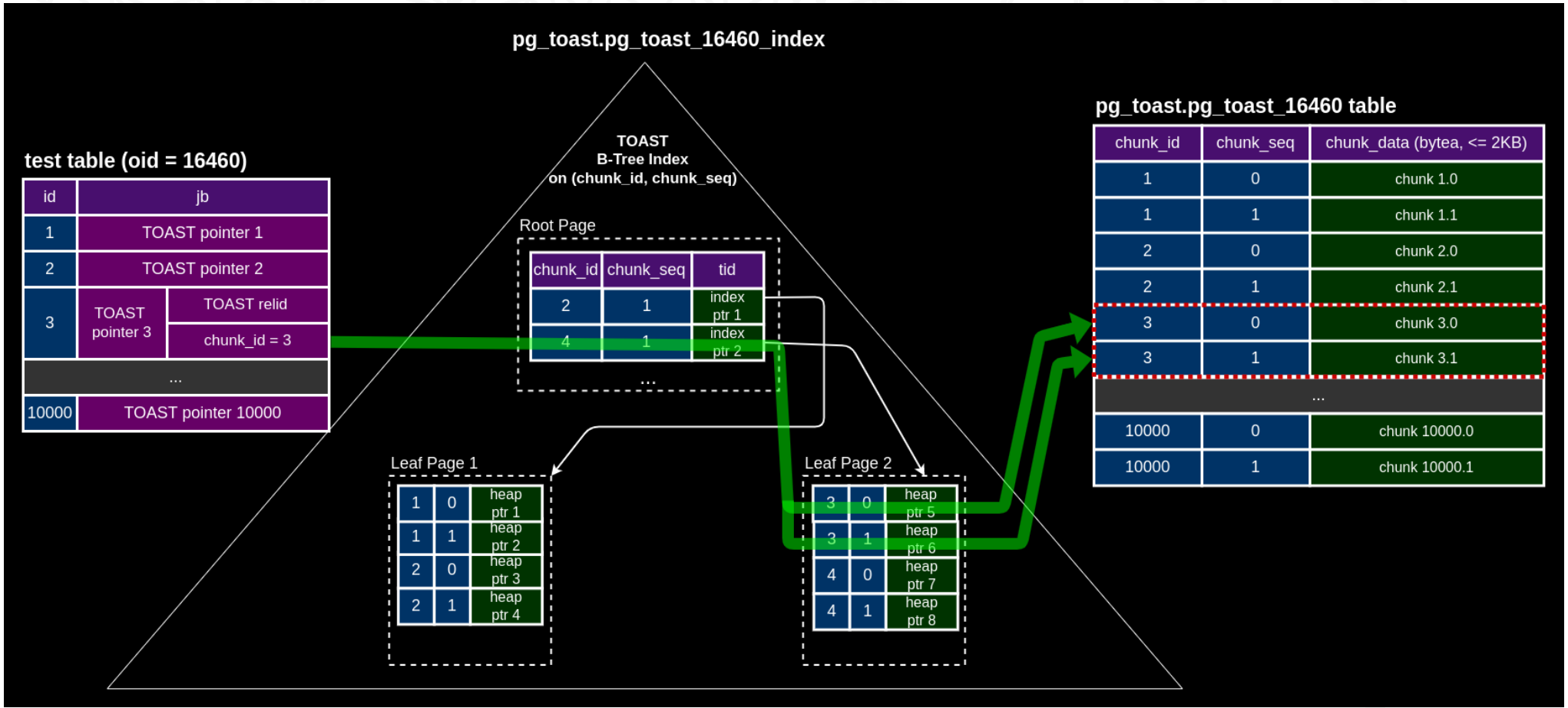


https://www.postgresql.org/docs/current/storage-toast.html

PostgresPro

TOAST access

TOAST pointer refers (by Oid `chunk_id`) to heap tuples with chunks using B-tree index (`chunk_id`, `chunk_seq`). Overhead to read only a few bytes from the first chunk can be 3,4 or even 5 index blocks.



PostgresPro

The Curse of TOAST

Access to TOASTed JSONB requires reading at least 3 additional buffers:

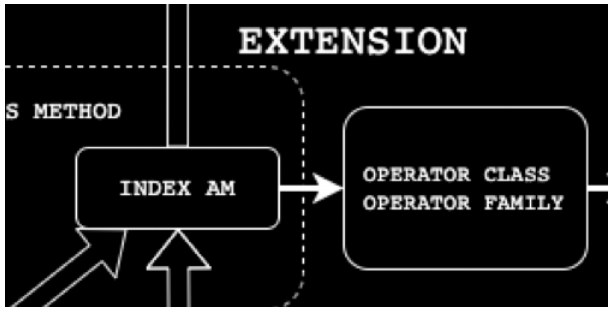
- 2 TOAST index buffers (B-tree height is 2)
- 1 TOAST heap buffer
  - 2 chunks can be read from the same page, but if JSONB size > Page size (8Kb), then more TOAST heap buffers

```
=# EXPLAIN (ANALYZE, BUFFERS) SELECT jb->'id' FROM test;
            QUERY PLAN
 ------------------------------------------------
   Seq Scan on test (actual rows=10000 loops=1)
    Buffers: shared hit=30064
  Planning Time: 0.105 ms
  Execution Time: 38.719 ms
 (4 rows)
```

Table	64
TOAST index	2 * 10000
TOAST table	1 * 10000
Total	**30064**

PostgresPro

Extensible (Pluggable) TOAST

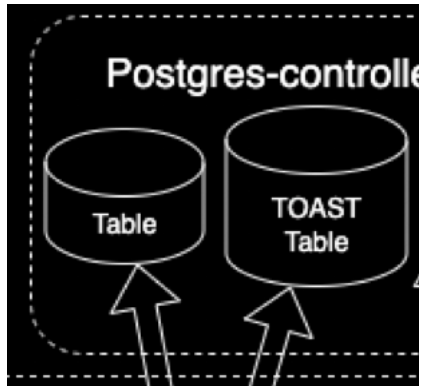


- TOAST is a very stable technology, which just works !
- TOAST has several hard-coded strategies to work with different data types, but it is "ancient" and knows nothing about jsonb, arrays, and other non-atomic data types.
- TOAST makes no attempt to take into account a workload, for example, append-only data.
- TOAST works only with binary BLOBs, when the TOASTed attribute is being updated, its chunks are simply fully copied. The consequences are:
  - TOAST storage is duplicated
  - WAL traffic is increased in comparison with updates of non-TOASTED attributes, since the whole TOASTed values is logged
  - As a result, performance is too low

- It’s time to improve it !

PostgresPro

Extensible (Pluggable) TOAST

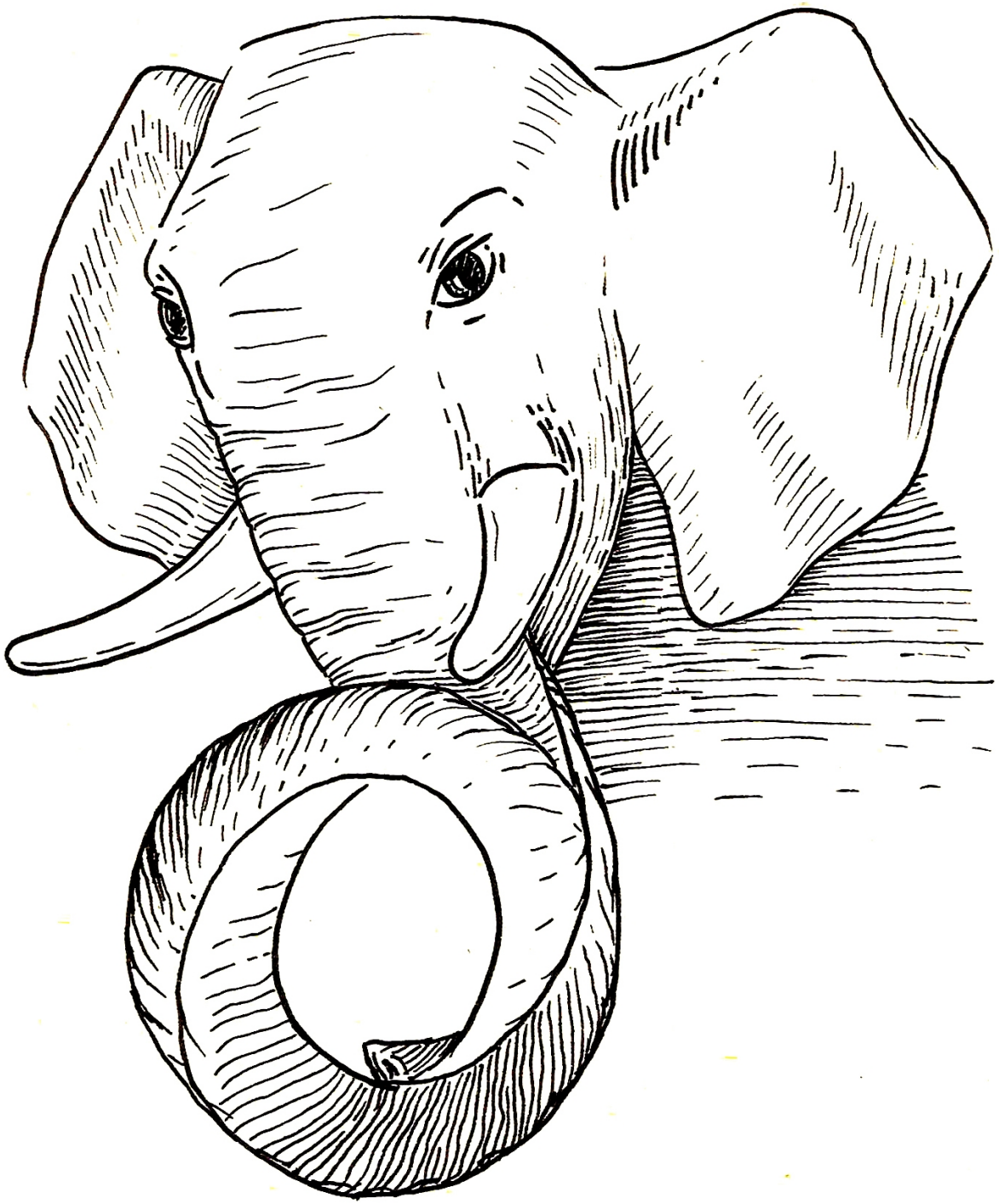


TOAST API (WIP):

- Data type aware TOAST
  - Blazing fast JSONB for SELECT and UPDATE
  - BINARY BLOBS storage
- Workload aware TOAST — appendable Bytea (streaming binary data into Postgres !)

- More details in http://www.sai.msu.su/~megera/postgres/talks/toast-nizhny-2022.pdf

PostgresPro

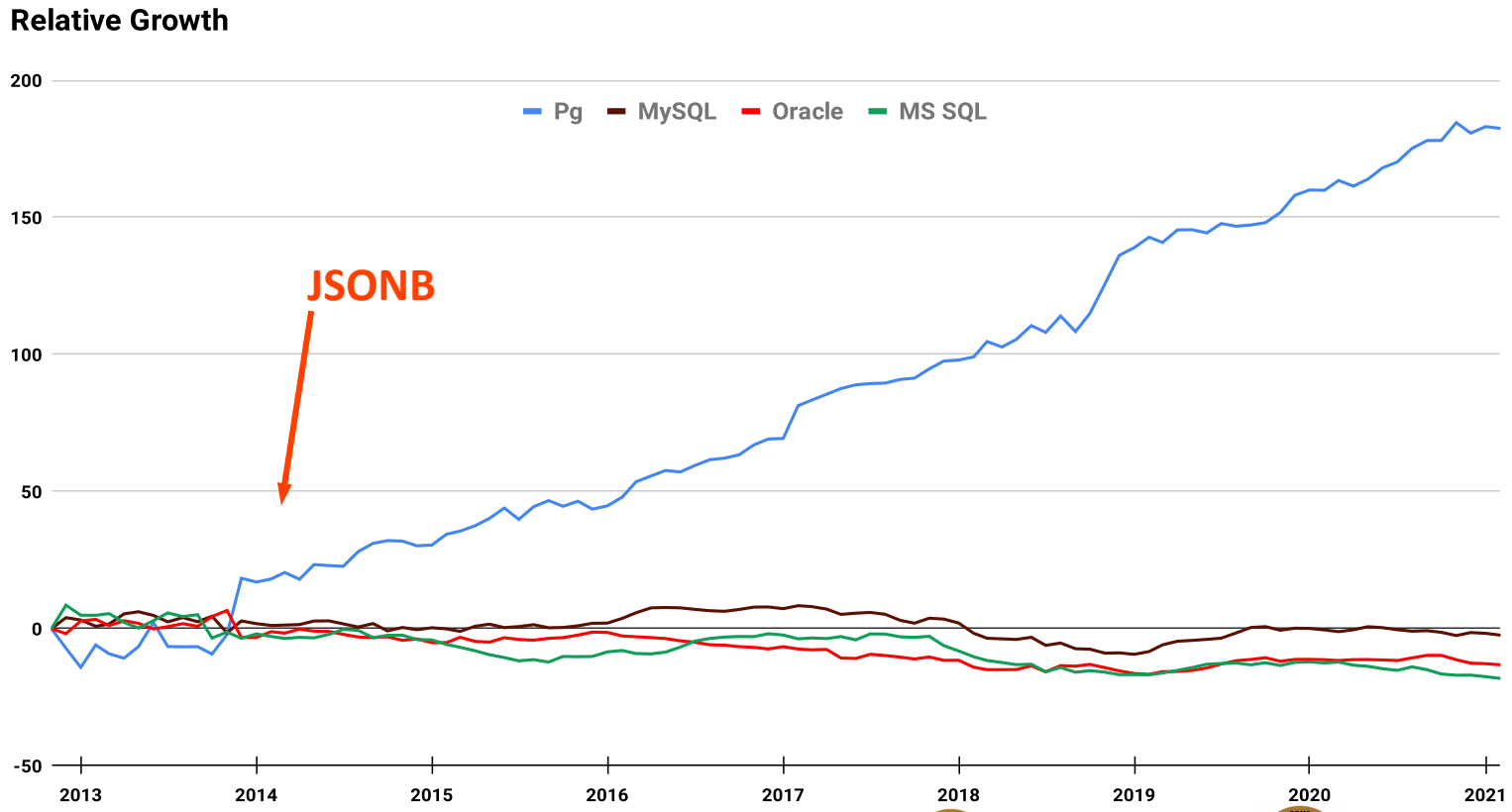


NOSQL Postgres

PostgresPro

Postgres breathed a second life into relational databases

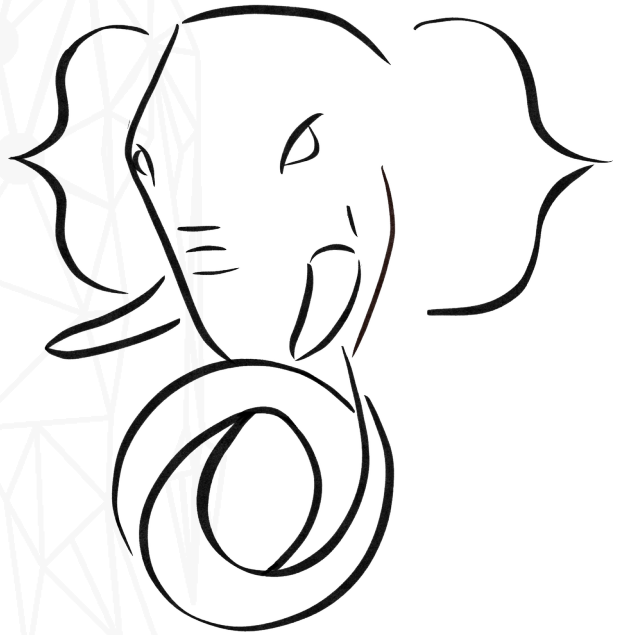
- **Postgres innovation** - the first relational database with NoSQL support
- NoSQL Postgres attracts the NoSQL users
- JSON became a part of SQL Standard 2016



db-engines.com/en/ranking

PG16: SQL/JSON/TABLE (#postgrespro)

Tnanks, Alvaro Herera for committing !



PostgresPro

SQL/Foundation recognized JSON after the success of Postgres

SQL:2016 — 22 JSON features out of 44 new optional. December of 2016

PostgresPro

Why we love JSON[B] ?

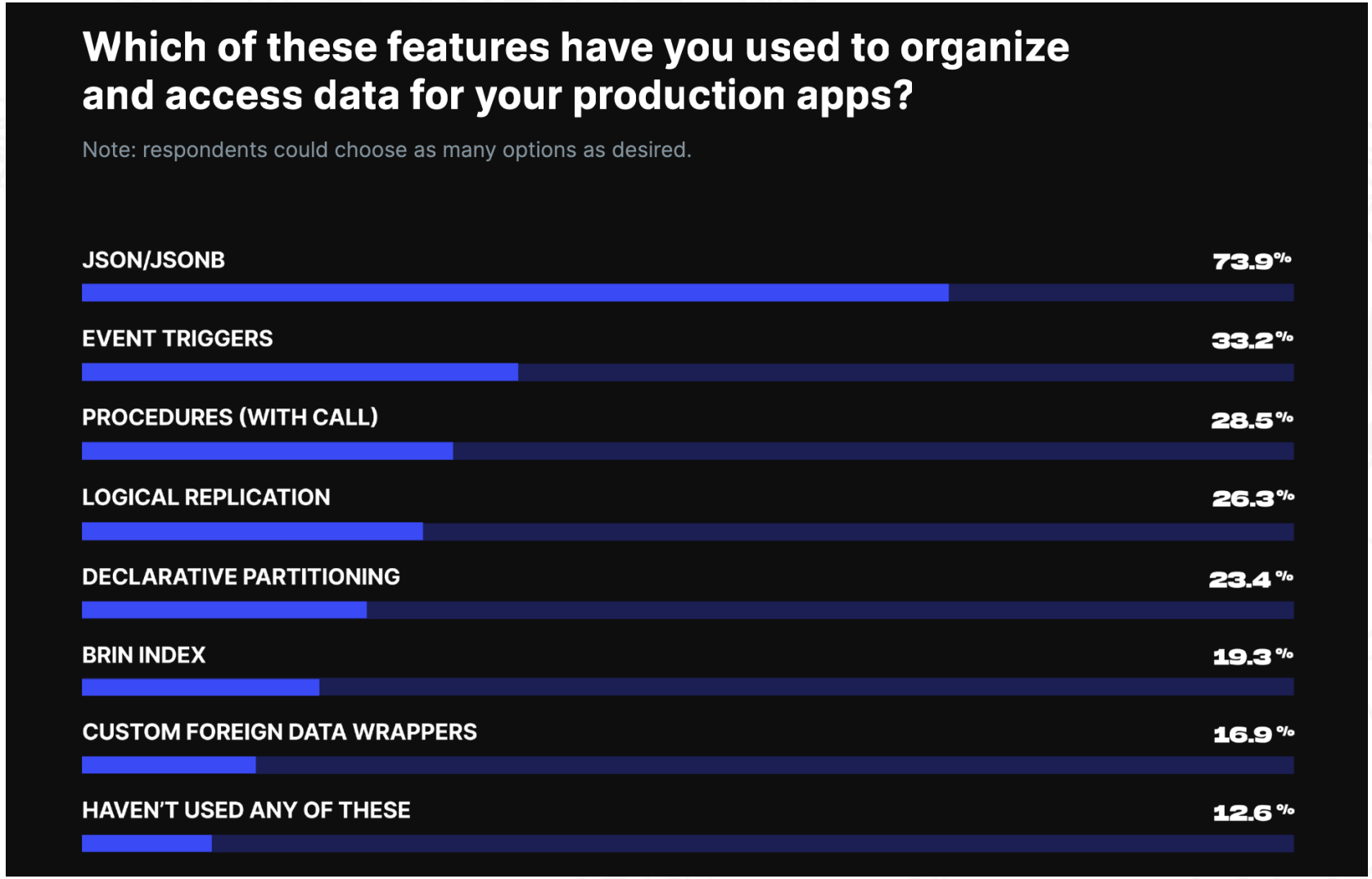
Startups want/need JSON[B]

```
CREATE JSON products (
  ……..
);
```

- Popular — microservices, clouds, startups
- Ubiquitous format for data interchange, storing API messages (XML is too much)
- Simple database design (simple queries) , support of Agile development
- Data migration (schema evolution). Old applications can easy accept new data.
- Compact storage of metadata — one column for all
- One-Type-Fits-All: Client app, backend, database — one format, all server side languages support JSON, now SQL support JSON
- JSON relaxed code-centric vs data-centric

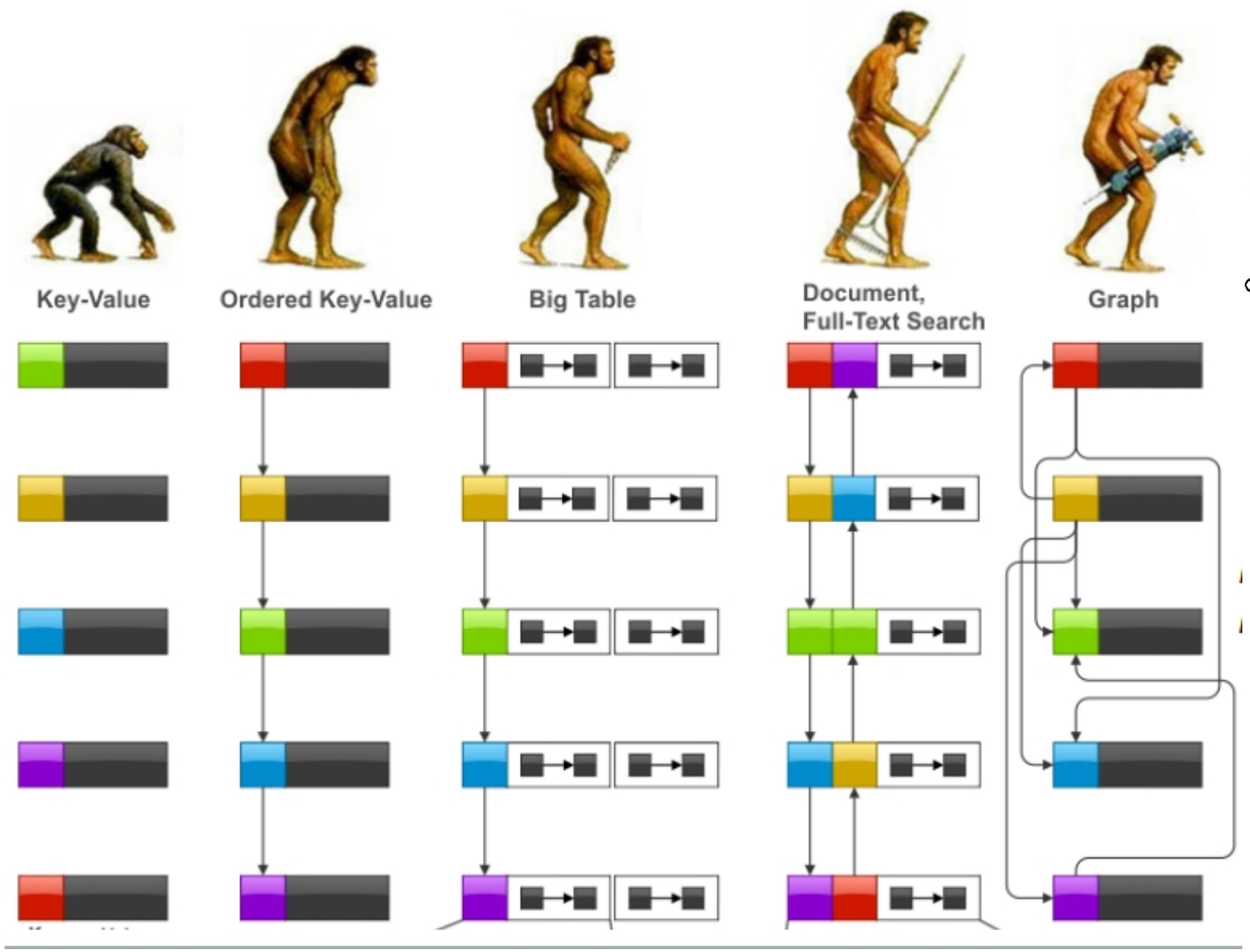
PostgresPro

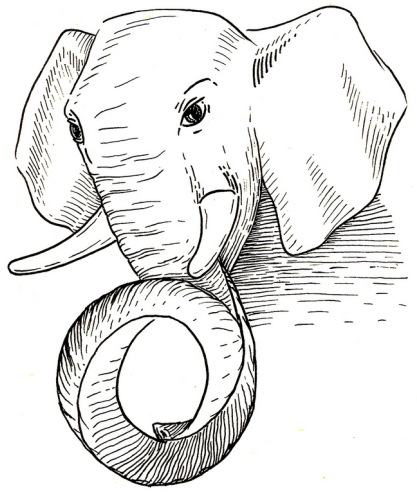
Why we love JSON[B] ?



PostgresPro

NOSQL POSTGRES STORY





SQL/JSON — PG16(2023)
- Complete SQL/JSON
- Better indexing, syntax

JSONPATH - 2019
- SQL/JSON - 2016
- Functions & operators
- Indexing

JSONB - 2014
- Binary storage
- Nesting objects & arrays
- Indexing

JSON - 2012
- Textual storage
- JSON verification

HSTORE - 2003
- Perl-like hash storage
- No nesting, no arrays
- Indexing

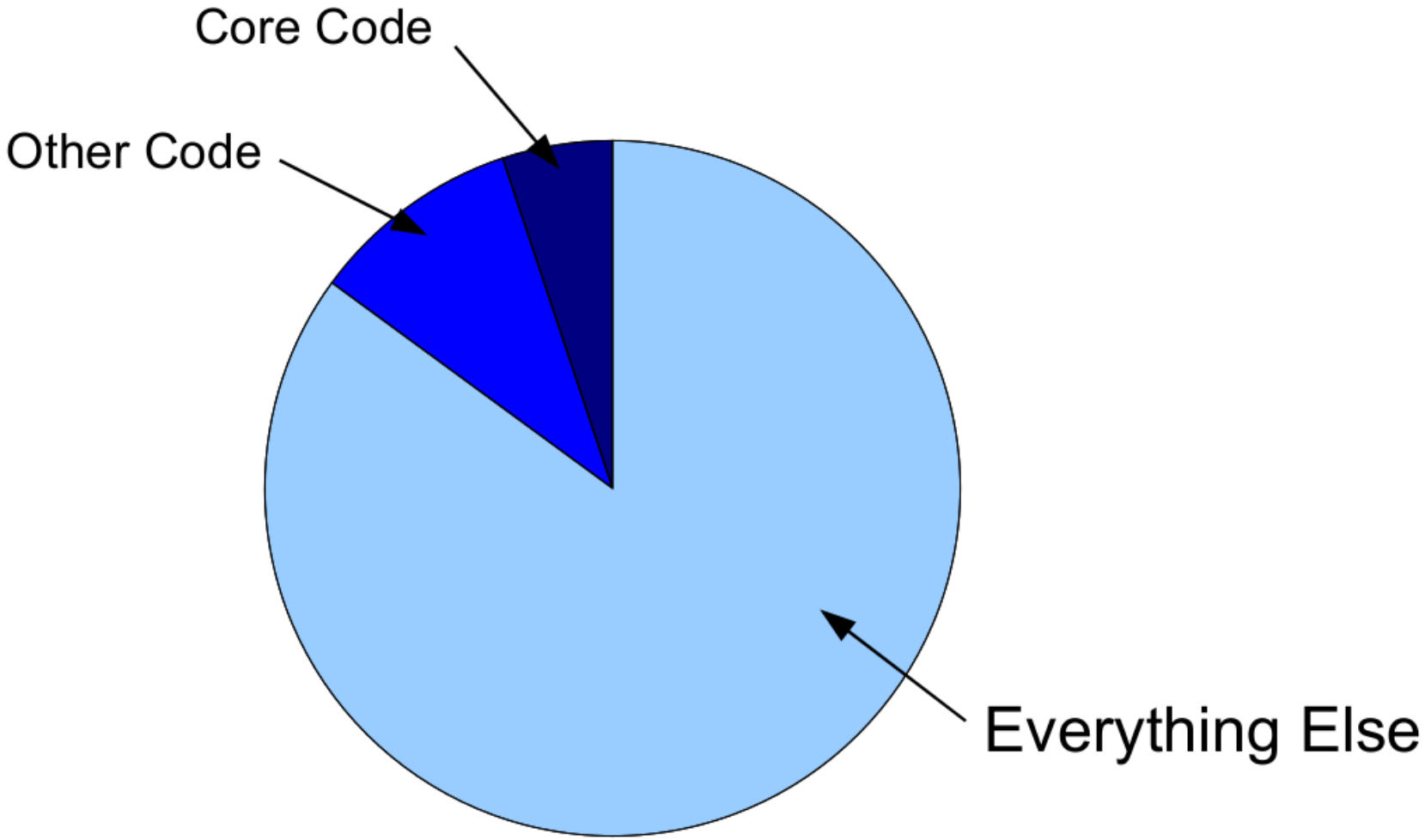
PostgresPro

NoSQL Postgres Future

- JSONB - 1st-class citizen in Postgres:
  Efficient storage,select, update, API
  - Extend further Postgres Extensibility - TOAST API
    - JSONB TOASTER - blazing performance!
- Dot notation for JSONB, Jsonpath syntax extension
- JSONB executor for efficient intra-operations
- Projective indexing for JSONB — index what you want
- COPY with FORMAT JSONPATH - copy what you want
- Unification of JSON and JSONB - choose what you want

PostgresPro

Contribute to Postgres, Build you career !



PostgresPro

Contribute to Postgres, Build your career !

Core development

Development, review, testing, reporting bugs.
Google Summer of Code (GSoC) — good start for students, we love students.

Ecosystem

Extensions, drivers, ORM, monitoring tools…
Postgres support in applications
Distributions, packages

Documentation

Improvement, translations, writing books, papers, ...blogging!

Meetings, Education

Creating of local communities, conference, meetups, seminars, hackatons, educational and training courses. Teach Postgres !

Use PostgreSQL!

Use Postgres in your company !

Sponsorship

Help development, sponsor community events.

PostgresPro


ALL
YOU
NEED
POSTGRES
IS