



Moving MongoDB Workloads to PostgreSQL with FerretDB

Alexey Palazhchenko, Co-Founder and CTO &
Peter Farkas, Co-Founder and CEO, FerretDB Inc.
PGConf India 2024, Bangalore

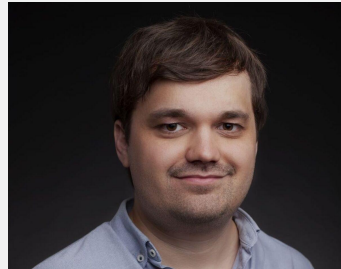


Founders of FerretDB



Alexey Palazhchenko, CTO

Ex-Percona, Talos



Peter Farkas, CEO

Ex-Percona, Cloudera



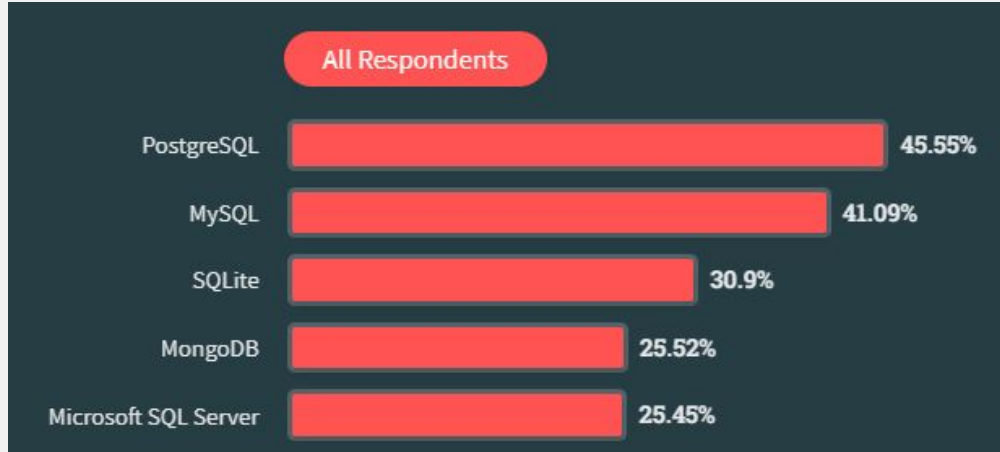
Peter Zaitsev

**Founder and former CEO of
Percona**

Agenda

- Why move MongoDB to Postgres? MongoDB's licensing
- History of SQL as an Open Standard
- MongoDB Query Language as an Open Standard?
- Architecture of FerretDB
- Conclusion

MongoDB's popularity



“Which database environments have you done extensive development work in over the past year, and which do you want to work in over the next year?” 76k responses

[Source: StackExchange Developer Survey, 2023 \(excerpt\)](#)

“Which database environments have you done extensive development work in over the past year, and which do you want to work in over the next year?”

Wait, isn't MongoDB open source?



SSPL

MongoDB - since 2018, released under the Server Side Public Licence (SSPL).

If MongoDB is used as part of a Cloud Service...

... source code of everything you use to provide that service needs to be licensed under SSPL.

Or you pay a licence fee.

MongoDB's approval (and license) is needed. This **kills innovation and creates vendor lock-in.**

More info:

www.ssplisbad.com

Also: Peter Zaitsev and Matt Yonkovic's articles on Percona Blog



Why not just use Postgres and JSON?

MongoDB has a large ecosystem. Many JS frameworks, stacks (MERN, MEAN, MEVN...) and other tools include or depend on MongoDB compatibility to work.

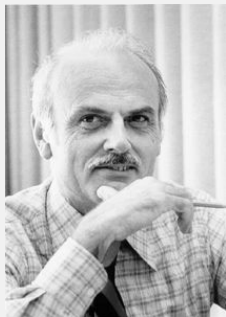
The Meteor logo consists of the word "METEOR" in a bold, black, sans-serif font. The letter "O" is replaced by a stylized red and white graphic of a comet or meteor streak.The Next.js logo features the word "NEXT" in a black, sans-serif font, followed by ".js" in a smaller font. A thin black line is drawn diagonally across the letters "E" and "X".The React logo includes the blue "Atom" symbol (a central blue dot with three elliptical orbits) and the word "React" in a blue, sans-serif font.The Express logo is the word "Express" in a thin, black, sans-serif font.The JS logo is a yellow square with the letters "JS" in black, bold, sans-serif font.The Node.js logo features the word "node" in a black, lowercase, sans-serif font. The letter "o" is a green 3D cube. Below "node" is a green hexagon with "JS" inside, and a registered trademark symbol.The FerretDB logo consists of a black circular icon with a white stylized animal head (a ferret) and the text "FerretDB" in a bold, black, sans-serif font.

The short story of SQL

The short history of SQL

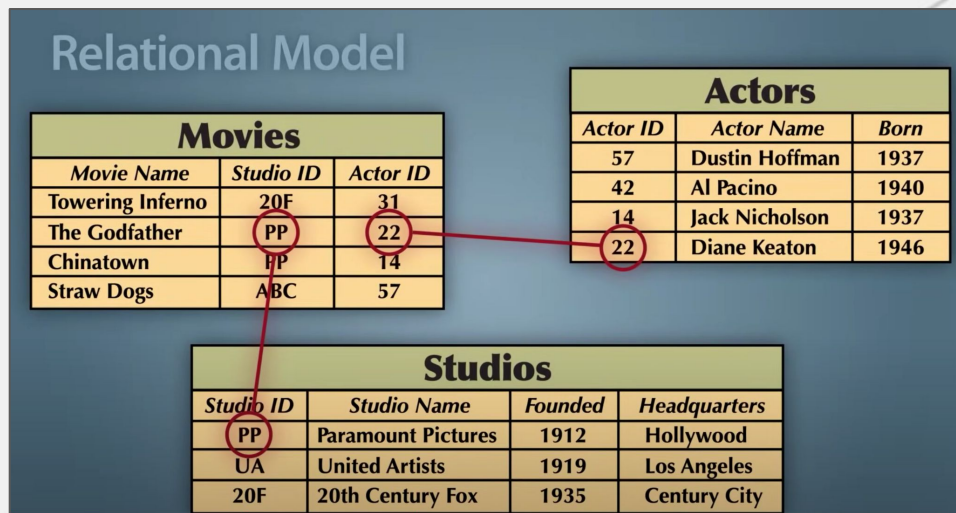
ACM paper: A relational model of data for large shared data banks, Jun/1970

Author: Edgar F "Ted" Codd, IBM Research Labs

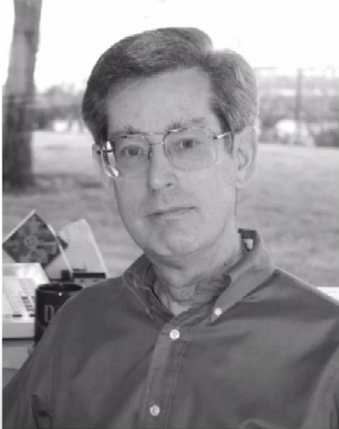


1.2.3. *Access Path Dependence.* Many of the existing formatted data systems provide users with tree-structured files or slightly more general network models of the data. Application programs developed to work with these systems tend to be logically impaired if the trees or networks are changed in structure. A simple example follows.

Muthukkaruppan, K. (2023) "Evolution of transactional databases," *Postgres Conf. SV. Postgres Conf. SV*, San Jose: Hilton, 20 April.



The short history of SQL



Don Chamberlin



Ray Boyce († 1974)



At IBM, in the early 70s, Don Chamberlin and Ray Boyce laid down the foundations of the SQL query language for relational databases, which even non-developers could use.

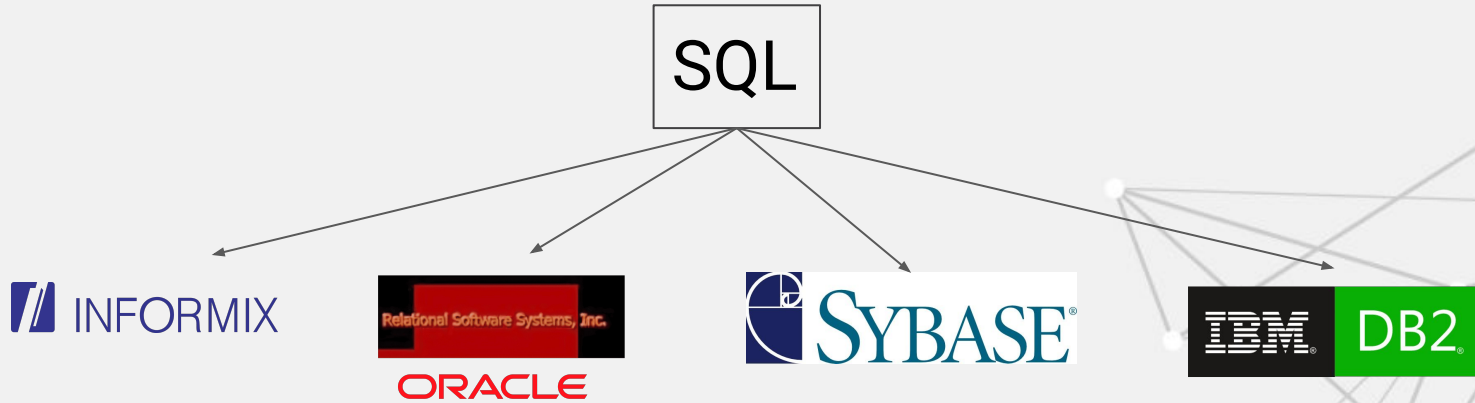
The short history of SQL - late 70s

- IBM releases multiple relational database products speaking SQL
- Vendor lock-in, no other commercially available SQL database
- Like SQL? Use IBM.



The evolution of SQL as an Open Standard - 80s

SQL is great, let's implement it in dozens of different ways!



SQL?

SQL?

SQL?

SQL?

Many different dialects, no conformity between products.



The evolution of SQL as an Open Standard - 86-87

- SQL Becomes an ANSI and later ISO Open Standard called SQL86.
- Anyone can implement them
- Features can be added on top (standard extension)

All vendors were proprietary, this still meant vendor lock-in.



SQL, an Open Standard. Here comes Open Source!

Mid 90s, early 2000s: Open Source projects started adopting SQL, partial implementations of the standard.

SQL is available to be used by anyone.



Since then: hundreds of derivatives



History repeats itself...sort of

Is this just the thing of the past?

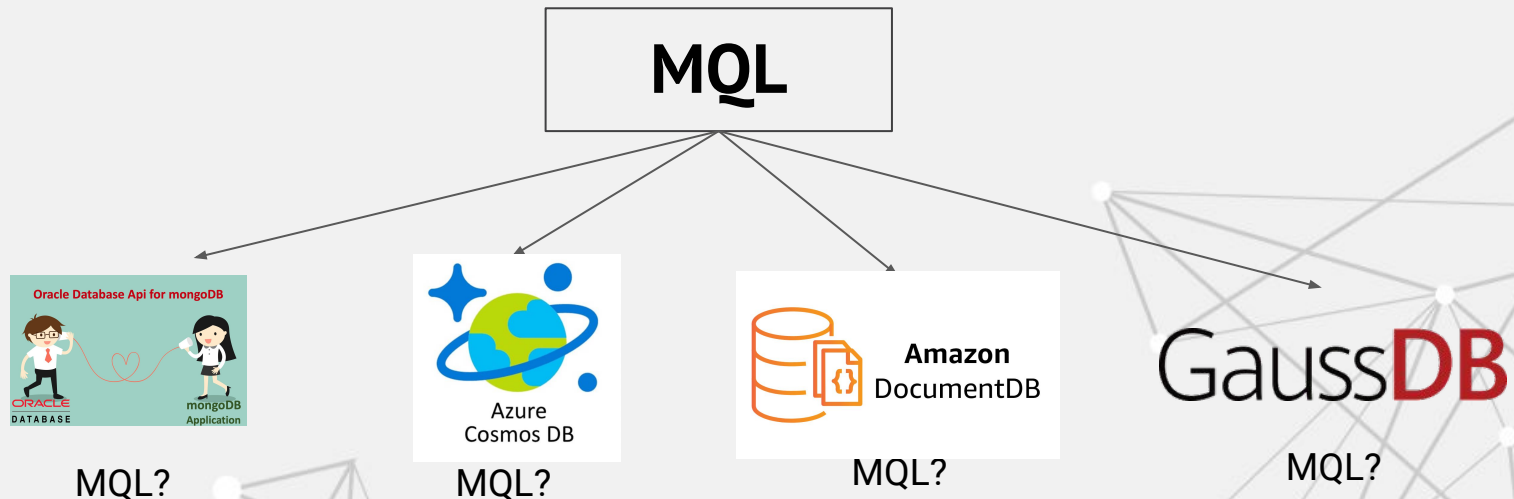
No. Look at Document Databases!

- MongoDB, a then open source database, develops MQL - the MongoDB Query Language.
- “Don’t need to be a DBA to run a database”
- Achieves market dominance in certain segments
- Goes proprietary in 2018
- Attempts to redefine the meaning of open source - SSPL license



A familiar situation

MQL is great, let's implement it in dozens of different ways!



All proprietary. Products look similar, but incompatible with each other. Once you choose one, you may be stuck with that.



The issue with most MongoDB Alternatives

All great products!

Similar query language, but

- vastly different feature set
- different degree of compatibility
- no chance of migration between them
- all proprietary
- Most tied to cloud vendors

For a “MongoDB alternative” - MongoDB sets the pace.



We are in need of a new Open Standard



MQL, an Open Standard?

- A standardized, core feature set based on MongoDB
- A JSON query language
- Can be extended at the expense of portability

Overwhelming interest from vendors and developers in the industry.

It will:

- Ensure portability between products
- Can be extended, just like SQL
- Stimulates innovation, increase competition
- Be very good for users

Architecture of FerretDB



FerretDB

- A MongoDB compatibility layer (proxy)
- Written in Go
- PostgreSQL as backend
- Usable on-prem or in the cloud
- Released under Apache 2.0



MongoDB compatibility means that existing elements of the MongoDB Ecosystem (tools, frameworks and applications) are possible to use with FerretDB



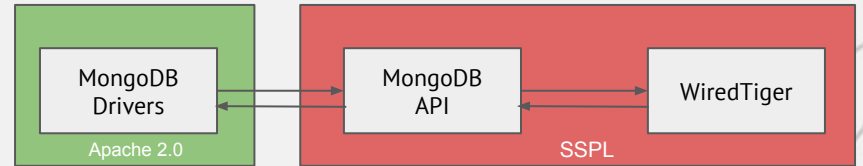
FerretDB vs. MongoDB architecture and licensing

MongoDB Drivers

- Main reason behind high adoption of MongoDB
- Provides unmatched Developer Experience
- Free to use under Apache 2.0

MongoDB Backend

- Licensed under SSPL
- Proprietary vendors (Amazon, IBM, Oracle, etc.) replaced it in their own implementations of a compatible product



FerretDB replaces the MongoDB Backend with PostgreSQL



Why PostgreSQL?

- FOSS
- Huge, supportive community
- Existing JSON compatibilities
- High number of PostgreSQL users run MongoDB
- Strong interest from users with extensive operational experience

Runs with a lot of derivatives and providers of PostgreSQL...



FerretDB 1.x

- Stores documents in (indexed) jsonb columns
- Fetches a superset of requested documents with SQL
- Filters fetched documents itself

FerretDB 1.x: Storage

- Field order is significant in BSON documents
- Field order is not preserved by jsonb objects

```
{
  "$s": {
    "$k": [
      "platform",
      "Application"
    ]
  },
  "platform": "Node.js v14.17.3",
  "application": "mongosh 1.0.1"
}
```

FerretDB 1.x: Storage

- BSON has more data types than JSON
 - int32, int64, double
 - Timestamp
 - JavaScript code

```
{
  "$s": {
    "p": {
      "v": {
        "t": "int"
      },
    },
  },
  "v": 42.0
}
```

FerretDB 1.x: Querying

- We need to compare BSON values for filtering and sorting
- Including values of different types
- And problematic values like nulls

WHERE $v < 42$
ORDER BY v

int32, int64, double
string
object, array

FerretDB 1.x: Querying

`NaN < 1`

`null < NaN`

`[] < null`

`[] < [null]`

`null </> [null]`

(depends on sorting order)

FerretDB 1.x: Testing

- Our own compatibility tests
- Tests of existing applications
- «MongoDB API Tester» tests
- Users

FerretDB 1.x: What we did so far

- Did a comparison in Go
- Fetched all data
- Added great integration tests
- Removed support for edge-cases
- Pushed down simple queries
 - `WHERE _jsonb->'v' = 42`
- Made indexes work

FerretDB 2.x: What should we do?

- Generate complex queries?
- Use stored procedures?
- Use custom operators?
- Use custom data type?
- Use custom extension?

FerretDB 2.x: What should we do?

- Use custom extension
- Technology preview will be available in about one month from now

FerretDB 2.x: What should we do?

- Roadmap: <https://github.com/orgs/FerretDB/projects/2>

Conclusion

- There is a need for an Open Standard for MQL
- MongoDB will become a commodity, like SQL

Best outcome for developers and the industry.

- FerretDB leads the way opening up the Document Database market
- We build FerretDB with the community
- We are looking for Postgres experts and providers to work with us on making this happen

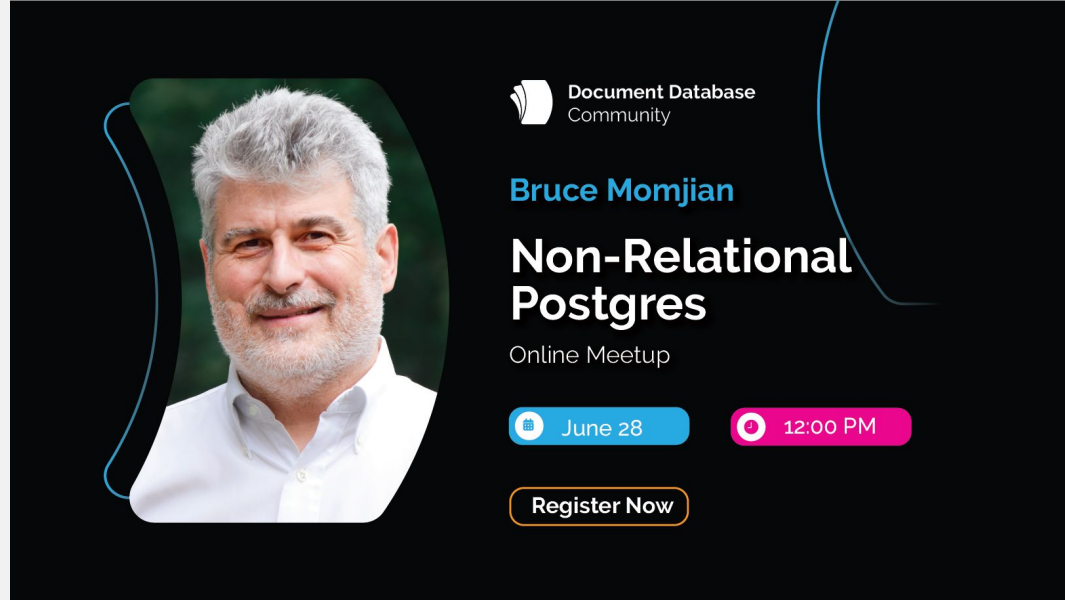
The Document Database Community

Founded by FerretDB,
open to all.

Monthly, soon bi-weekly talks.

Aim is to facilitate discussion
between vendors on industry trends,
standardizing
document databases.

www.documentdatabase.org



The image shows a promotional card for an event. On the left is a portrait of Bruce Momjian, a man with grey hair and a beard, wearing a white shirt. To the right of the portrait, the text reads: 'Document Database Community' with a logo, 'Bruce Momjian' in blue, 'Non-Relational Postgres' in large white font, 'Online Meetup', 'June 28' in a blue pill, '12:00 PM' in a pink pill, and a 'Register Now' button in a yellow pill. The background is black with blue and white decorative lines.

Document Database
Community

Bruce Momjian

**Non-Relational
Postgres**

Online Meetup

June 28

12:00 PM

Register Now

Questions

Try FerretDB:

try.ferretdb.io, scaleway.com, vultr.com

Run it on prem:

docs.ferretdb.io

Star us on GitHub:

github.com/FerretDB/FerretDB

www.ferretdb.io

