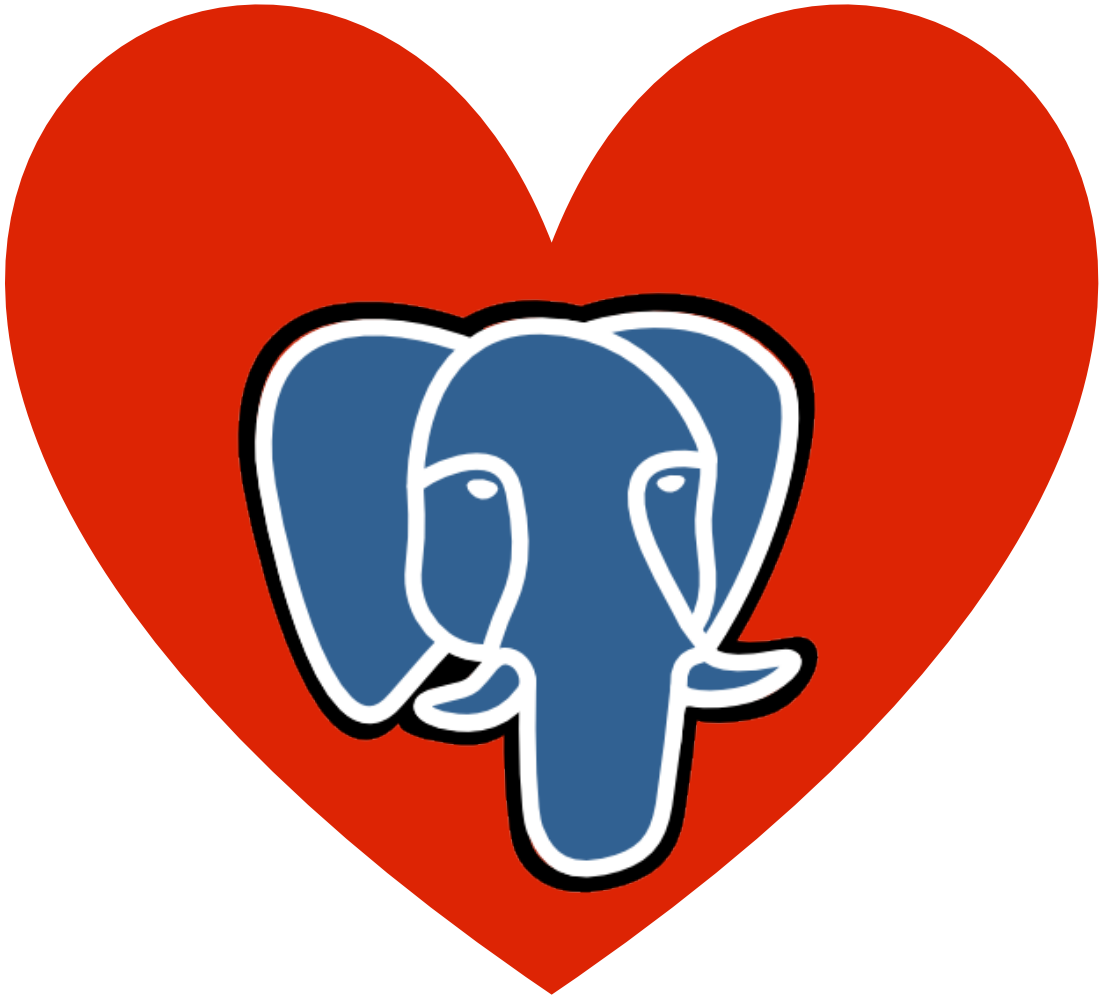
@paulmcfreely

SIMON
SOFTWARE IS EVERYWHERE



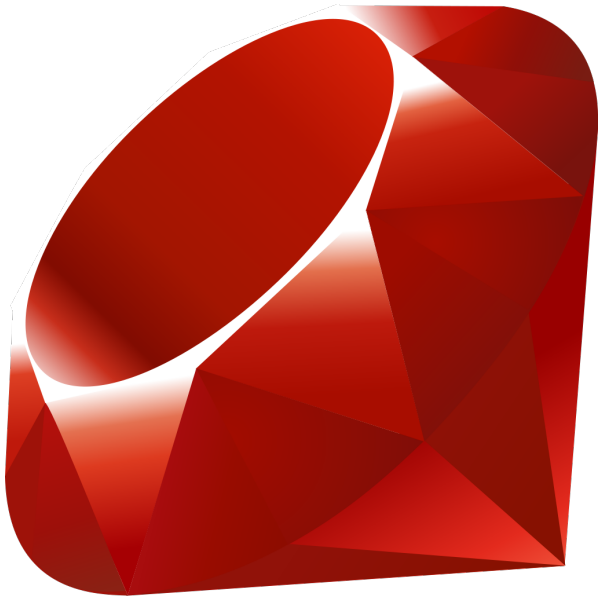
PostgresQL comme Nosql DB :

Le meilleur des deux mondes 

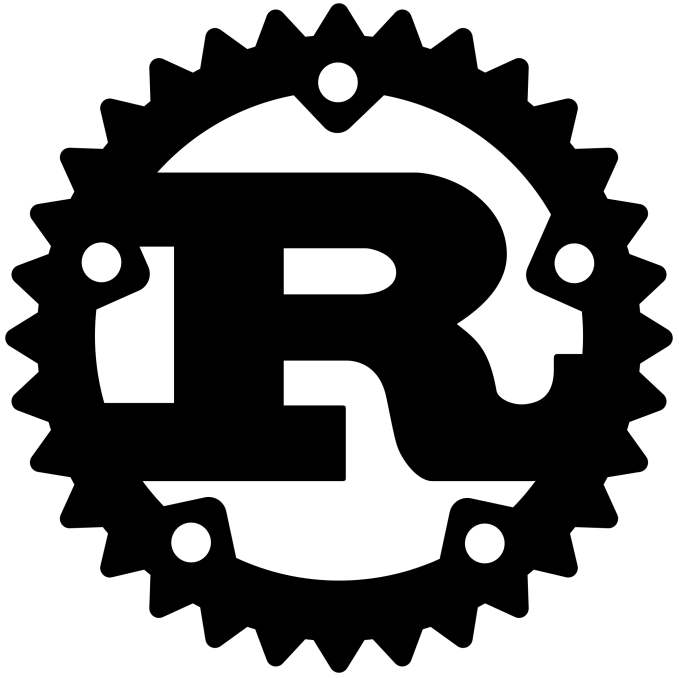
Haskell

C Lang

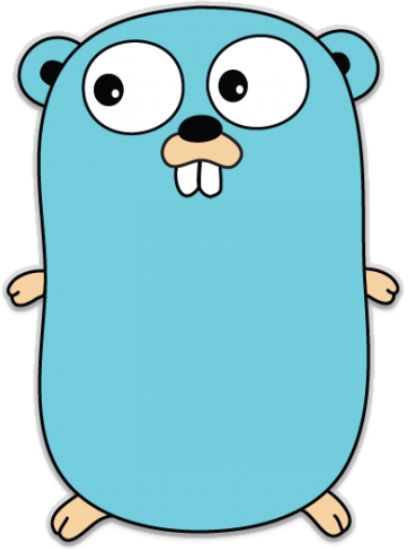
Python



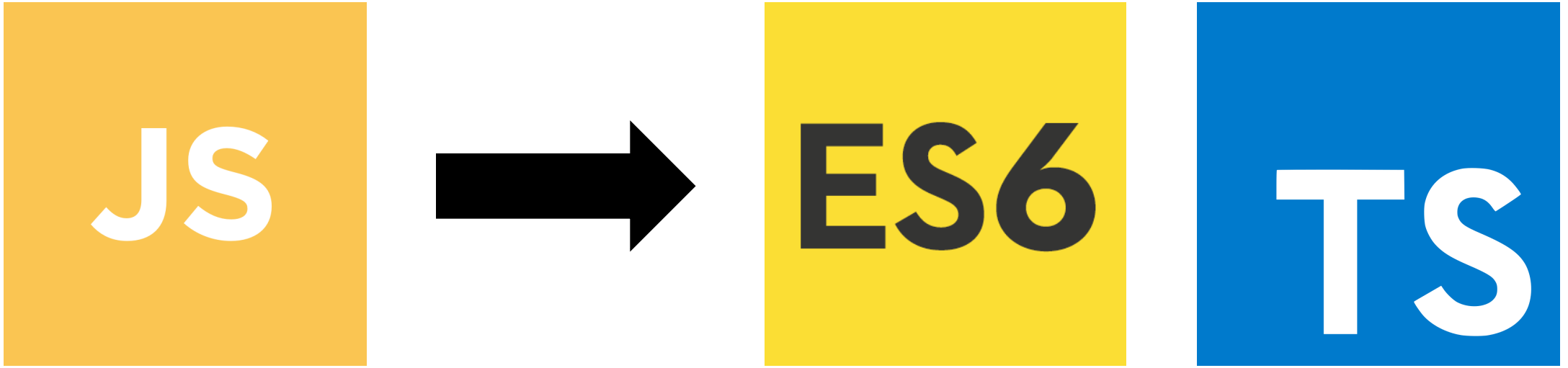
Ruby

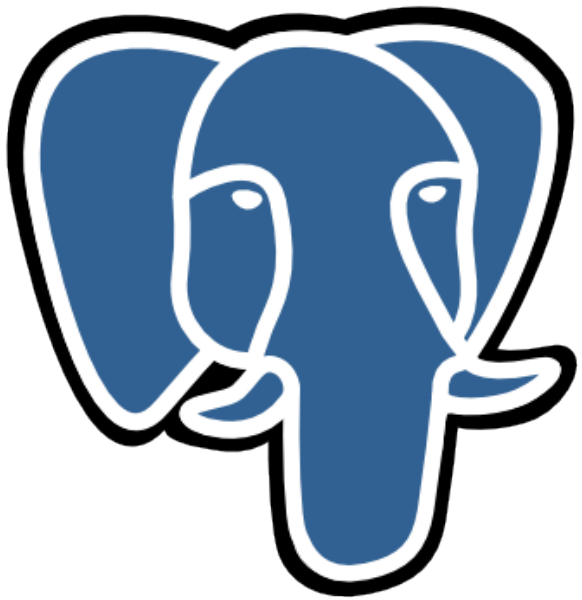


Rust

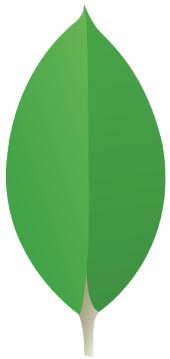


Golang


JS
ES6
TS




MySQL®



mongoDB®

Apache CouchDB
relax

? ? ?

E.15. Release 9.4

Release Date: 2014-12-18

E.15.1. Overview

Major enhancements in PostgreSQL 9.4 include:

- Add `jsonb`, a more capable and efficient data type for storing JSON data
- Add new SQL command ALTER SYSTEM for changing `postgresql.conf` configuration file entries
- Reduce lock strength for some ALTER TABLE commands
- Allow materialized views to be refreshed without blocking concurrent reads
- Add support for logical decoding of WAL data, to allow database changes to be streamed out in a customizable format
- Allow background worker processes to be dynamically registered, started and terminated

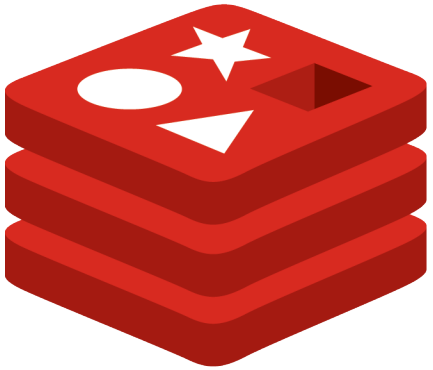
The above items are explained in more detail in the sections below.

99%

des cas,
le SQL répondra à vos besoins

elastic

Cassandra



redis

** Liste non exhaustive*

Analyse des besoins

Type d’opérations / de données

Volume des données

Performances

JSON & JSONB

JSON
JSON classique
Efficace pour la récupération de données
Opérations sont plus longue que JSONB

JSONB
Représentation JSON Binaire
Efficace pour un travail plus complexe
Plus long pour construire représentation => Couteux en écriture

```sql
CREATE TABLE movies (id INTEGER, data JSON);

INSERT INTO movies (
    1,
    '{ "title": "Sunshine", "director": { "first_name": "Danny", "last_name": "Boyle" }}'

);

INSERT INTO movies (
    2,
    '{"title": "Interstellar", "director": { "first_name": "Christopher", "last_name": "Nolan" }}'
);

```

```
13
14 SELECT id, data->'title' AS title FROM movies;
15
16 id |     title
17 ---+--------------
18   1| Sunshine
19   2| Interstellar
20
21
22 SELECT id, data->'director'->>'first_name' AS name FROM MOVIES;
23
24 id |     name
25 ---+--------------
26   1| Danny
27   2| Christoper
28
```

```
21
22 SELECT * FROM movies WHERE data->'title' = 'Sunshine';
23
24 id |     name
25 ---+--------------
26   1| '{ "title": "Sunshine", "director": { "first_name": "Danny", "last_name": "Boyle" }}'
27
28
```

Indices JSON

General inverted Index: Efficace pour de la recherche full text

```
CREATE INDEX idxgin ON users USING GIN (infos);
```

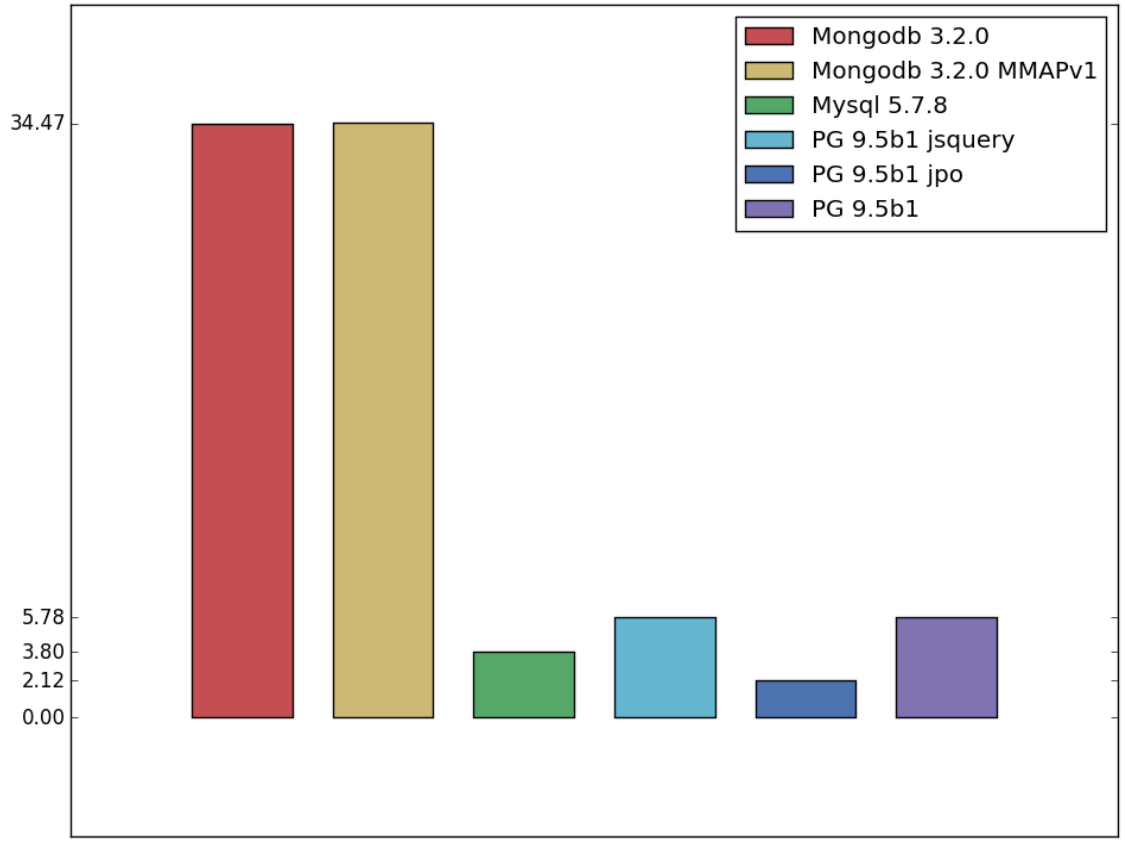
Indices JSONB

Hash et Btree Indices

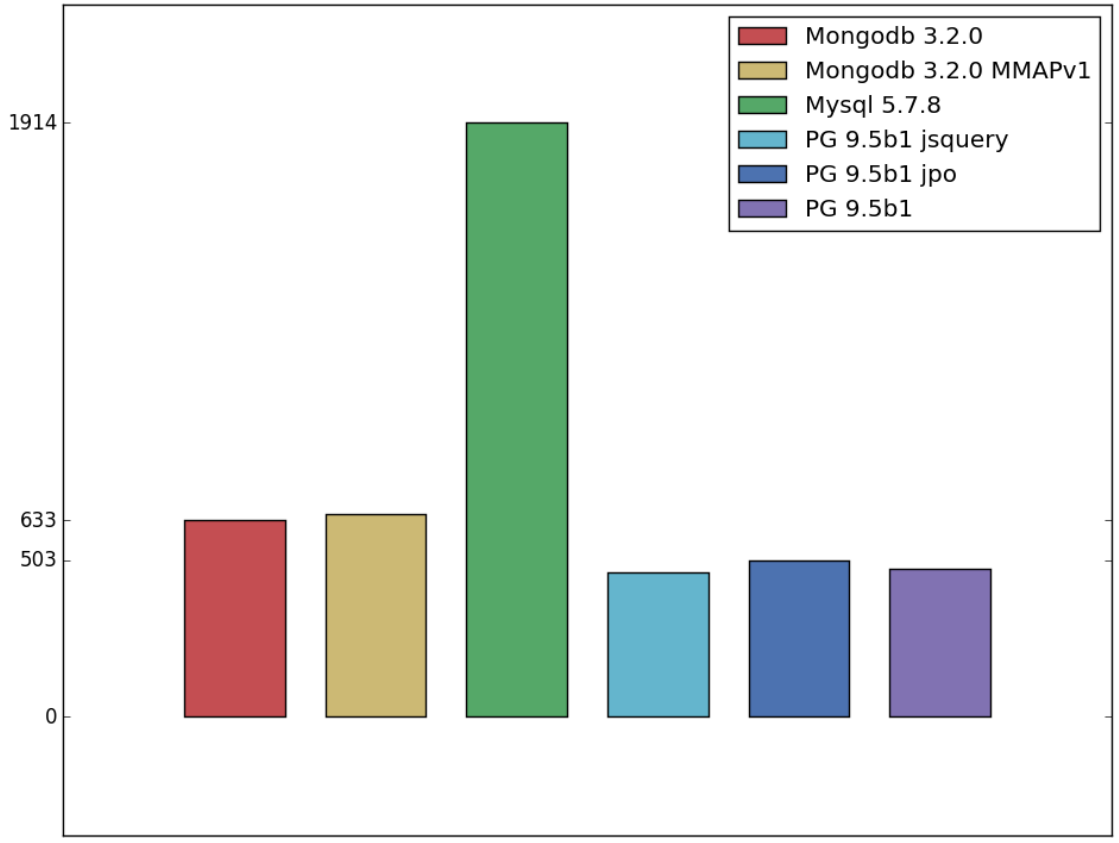
Important pour vérifier l’égalité entre deux documents JSON

Performances

Source: **pg_nosql_benchmark** via http://erthalion.info/

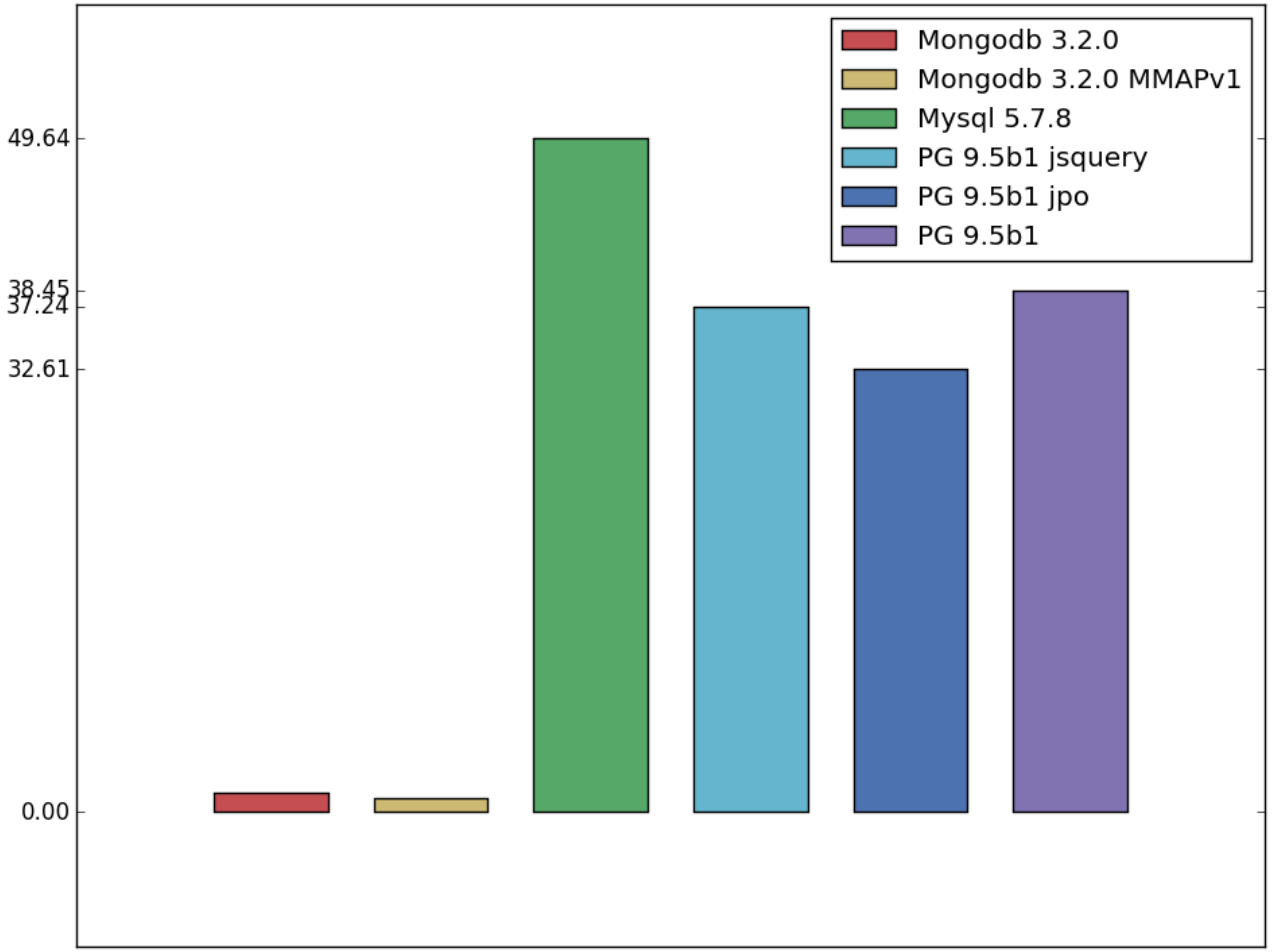


SELECT



INSERT

Update


Mongodb 3.2.0
Mongodb 3.2.0 MMAPv1
Mysql 5.7.8
PG 9.5b1 jsquery
PG 9.5b1 jpo
PG 9.5b1
49.64
38.45
37.24
32.61
0.00

ORMs

Il est souvent nécessaire de réécrire une couche pour gérer le JSON

- Sérialization / Déserialization
- Manipulation

Le futur ?