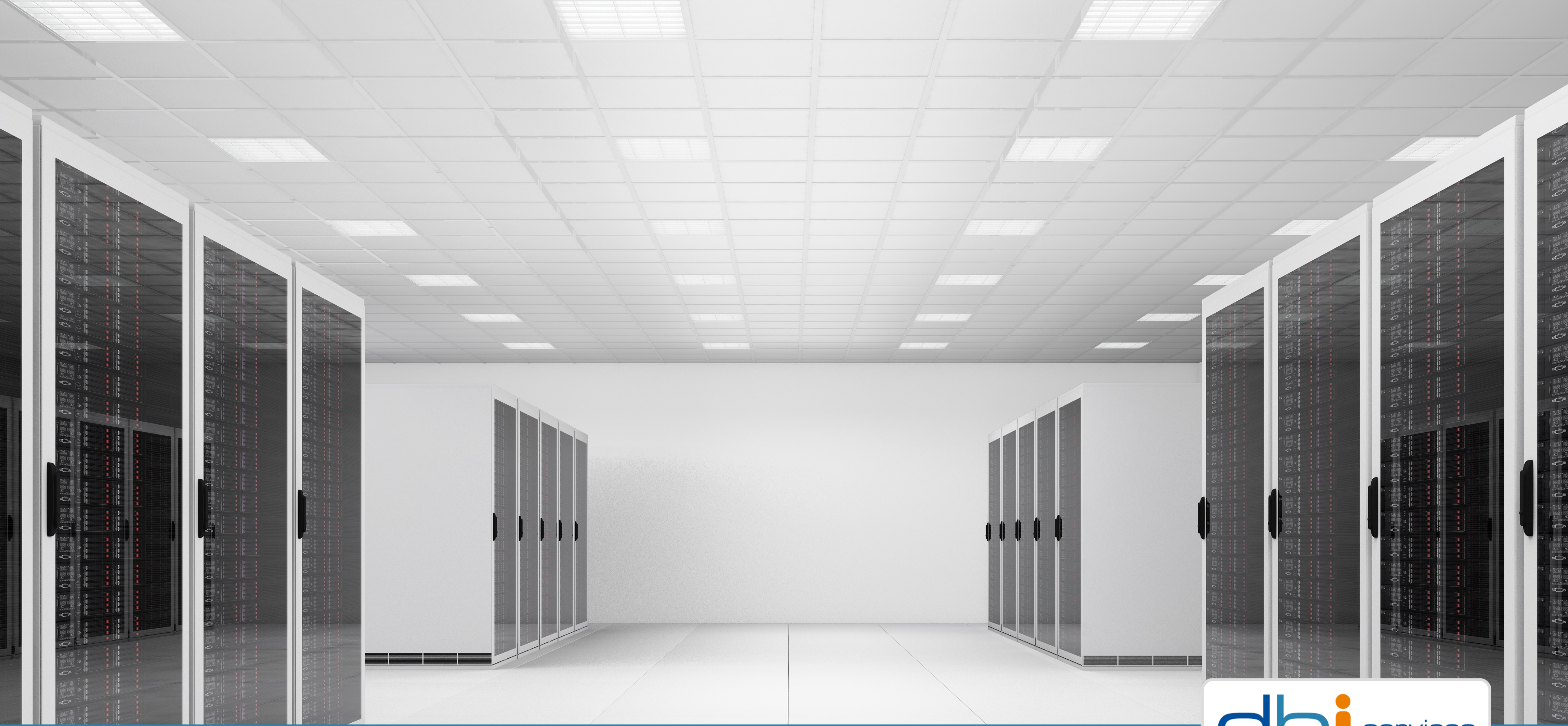


How to migrate data from MongoDB to Postgres with ToroDB

Who we are

Experts At Your Service
> Over 50 specialists in IT infrastructure
> Certified, experienced, passionate

Based In Switzerland
> 100% self-financed Swiss company
> Over CHF8 mio. Turnover

Leading In Infrastructure Services
> More than 150 customers in CH, D & F
> Over 50 SLAs dbi FlexService contracted

Certified Management System SQS ISO / IEC 20000-1

dbi services is hiring (career@dbi-services.com)

About me

Mehdi Bada

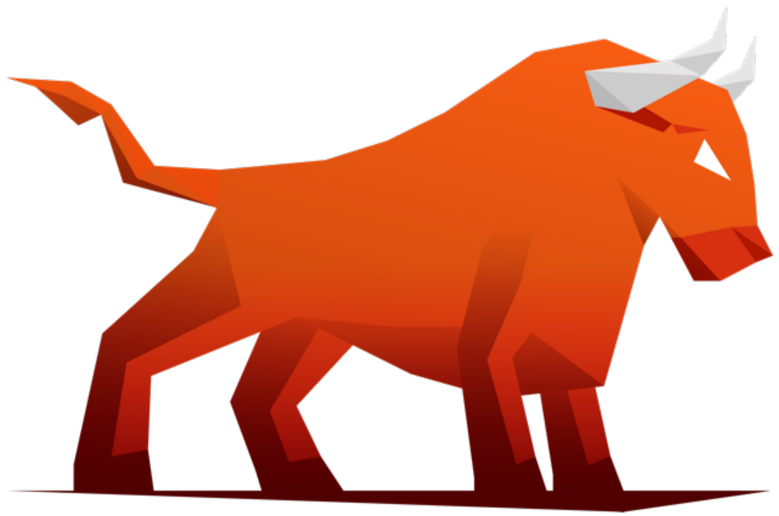
Consultant

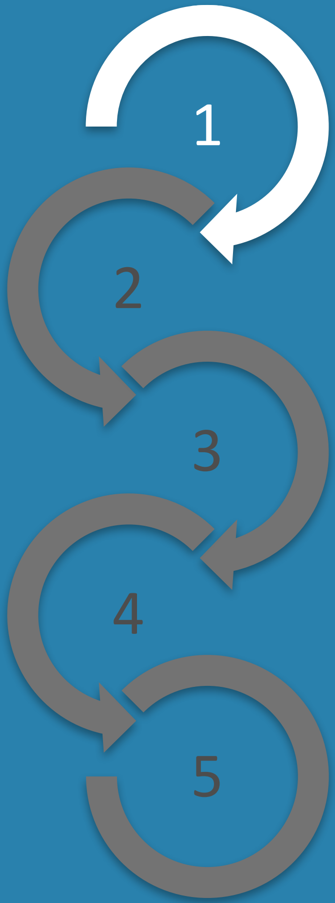
+41 79 928 75 48

mehdi.bada[at]dbi-services.com

Agenda

Introduction





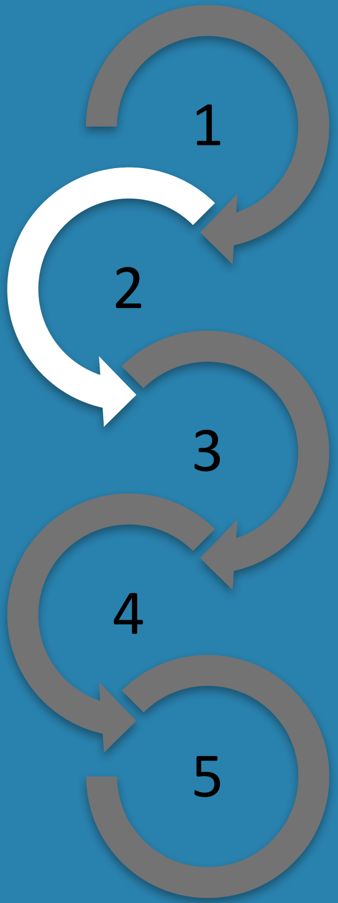
Introduction

mongoDB

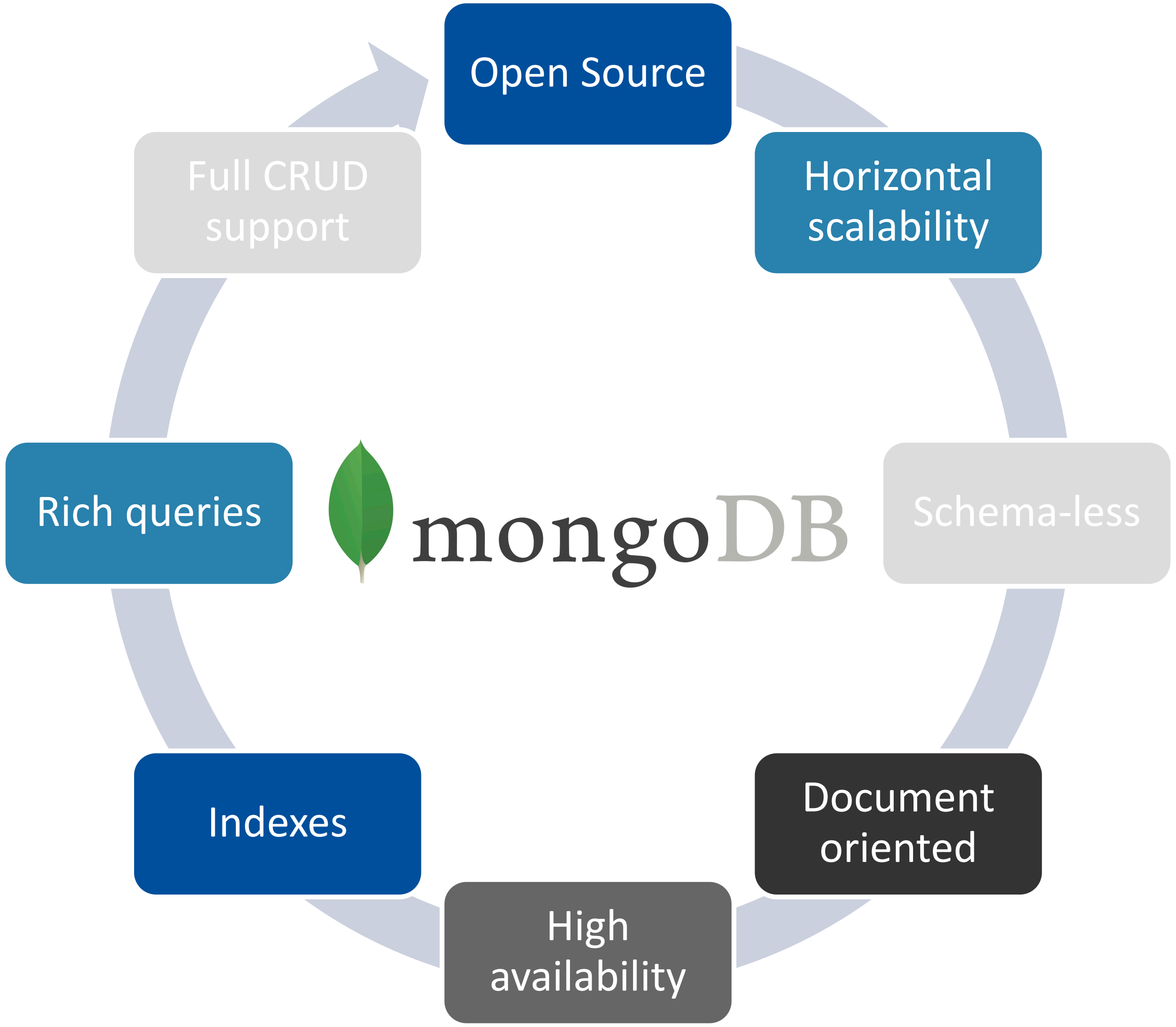
PostgreSQL

MongoDB

> Overview
> Data Model
> High availability
> Horizontal scalability
> Limitations



MongoDB
Overview



MongoDB
Overview

Concept mapping

RDBMS	MongoDB
Tables	Collections
Rows/records	Documents
Queries return rows	Queries return cursor
Join	Embedded document
Partition	Shard

MongoDB
Data Model

Data are stored as documents

MongoDB stores BSON documents (Binary JSON)

Analogous to a database row

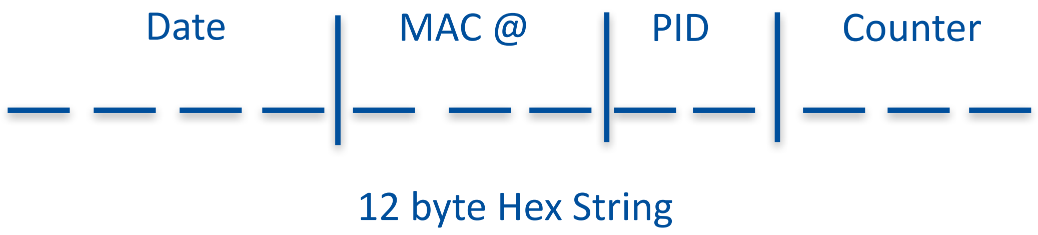
Keys and Values

> Key : String
> Values types:
>   > String, number, Boolean, null, array, object

```
{
        "_id" : ObjectId("56e92b9cfdf7bc92bbb3b51f"),
        "first_name" : "Mike",
        "surname" : "Brody",
        "city" : "New-York",
        "year" : 1987,
}
```

Special key: _id

> Unique identifier
> Object id:



MongoDB
Data Model

Relational

Pers_Id	Surname	First_name	City
0	Mike	Durand	Geneva
1	Pat	Millner	London
2	Ortega	Alvaro	New-York

Car_Id	Model	Year	Pers_Id
0	Ferrari	2013	0
1	Peugeot	2005	0
2	BMW	2016	1

MongoDB schema

```
{
      "first_name" : "Durand",
      "surname" : "Mike",
      "city" : "Geneva",
      "country" : "Switzerland",
      "cars":[
        { "model": "Ferrari",
          "year": 2013
          }
  }
```

MongoDB
High availability

Which mechanism ensure high availability of your data?

> MongoDB Replication

HA is achieved through automatic failover

MongoDB replication allows:

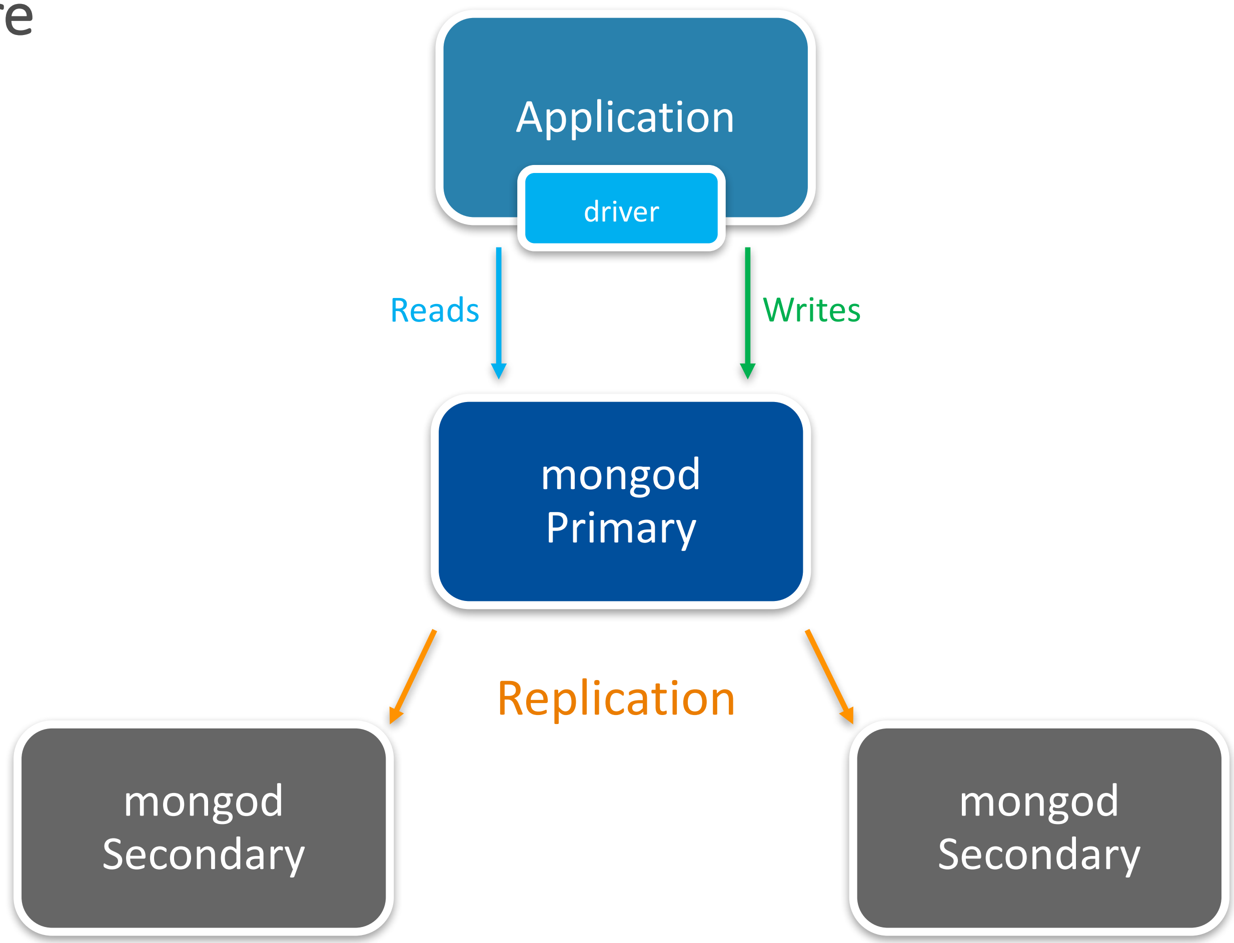
> High availability (HA)

> Disaster Recovery (DR)
>> Data duplication across multiple database servers / storages

> Functional Segregation
>> Topology of replica sets can be used for
>>> Backups, Analytics, Reporting, DR, Read operations…

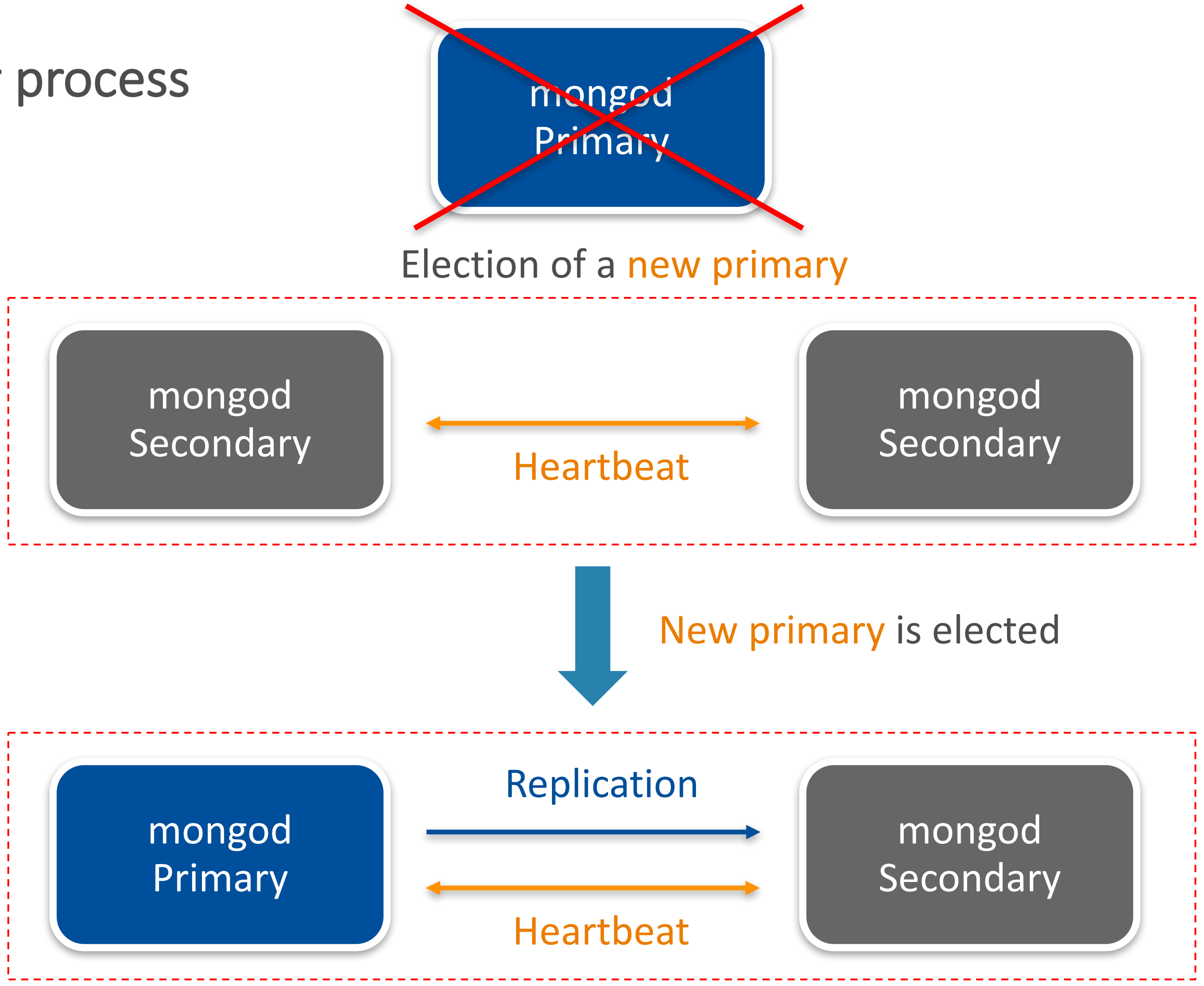
MongoDB
High availability

Replication architecture



MongoDB
High availability

Automatic failover process



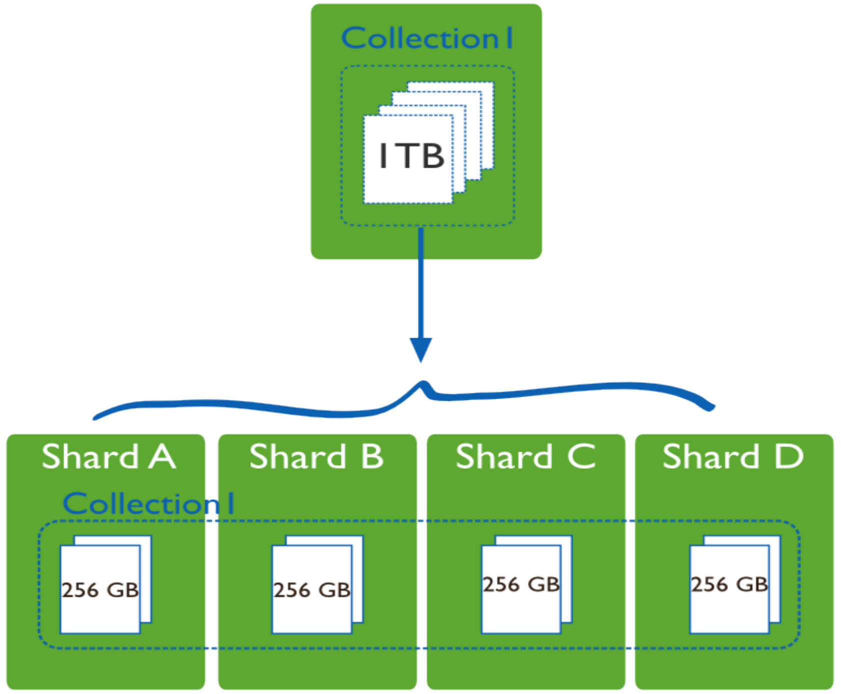
MongoDB
Horizontal scalability

Vertical scalability

Increasing CPU, RAM, I/O

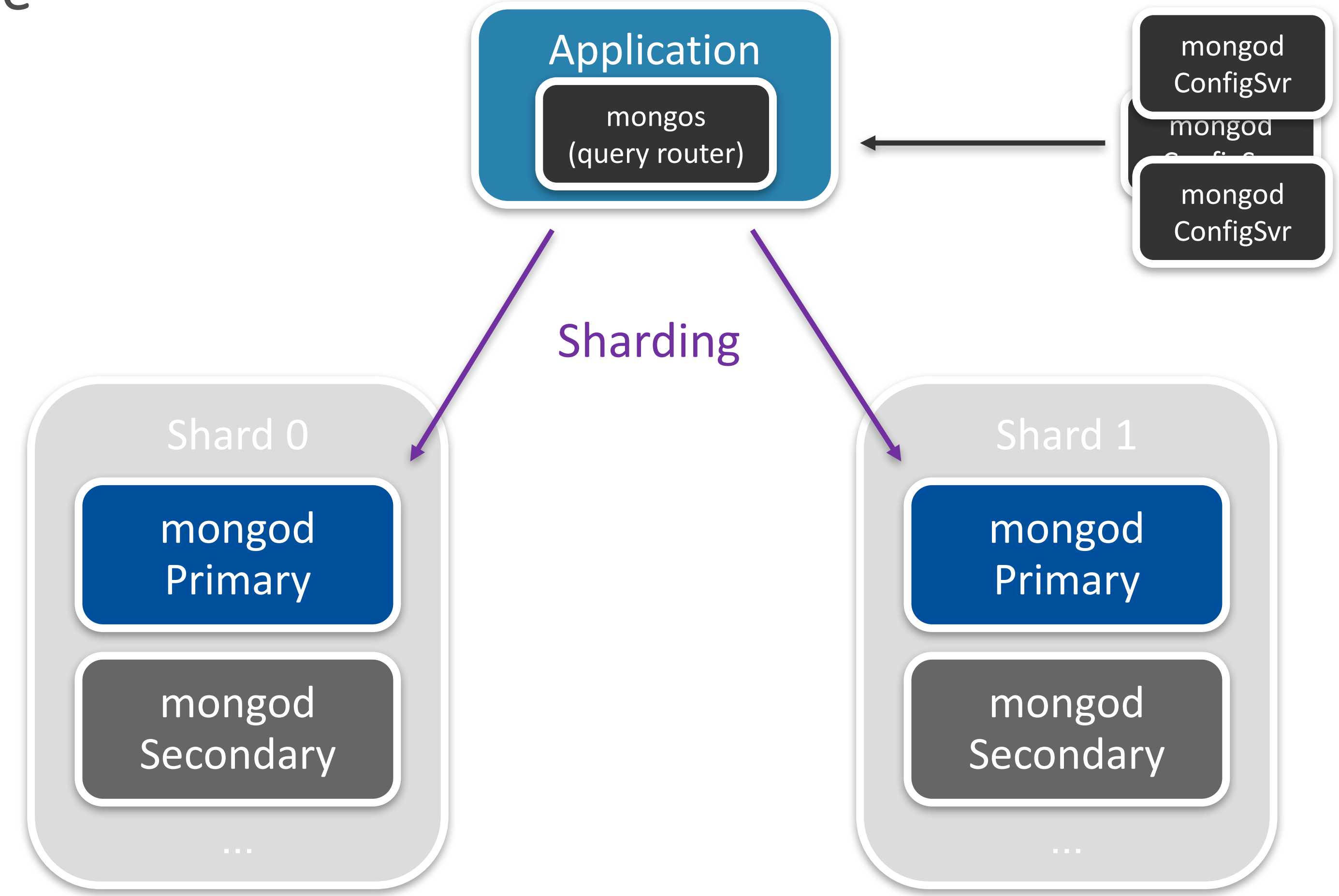
Scaling with MongoDB

> MongoDB Sharding



MongoDB
Horizontal scalability

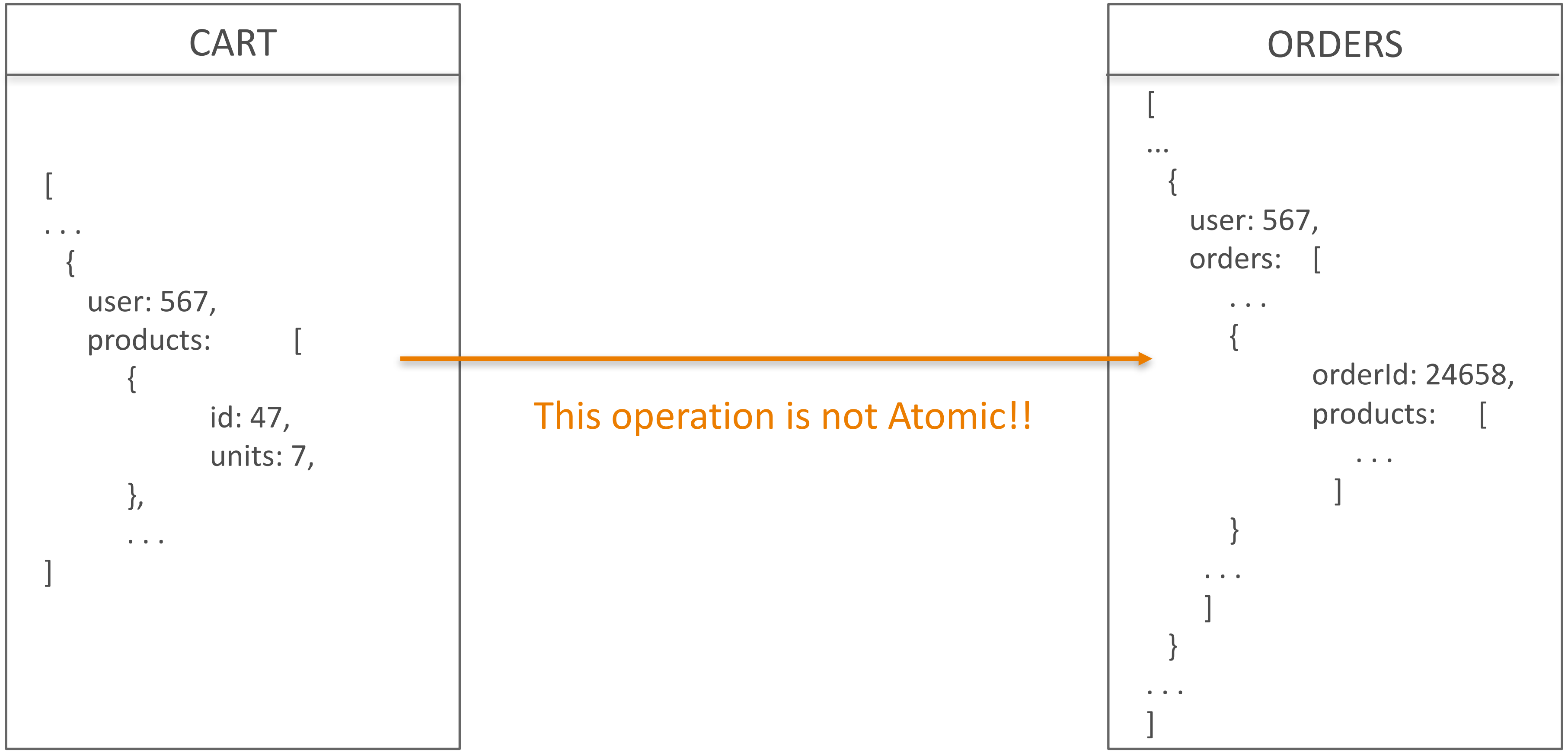
Sharding Architecture



MongoDB
Limitations

No ACID transaction

> Atomic transactions only work within the same document



```
[
. . .
  {
    user: 567,
    products:       [
      {
            id: 47,
            units: 7,
      },
      . . .
]
```

```
[
...
  {
    user: 567,
    orders:   [
        . . .
        {
              orderId: 24658,
              products:   [
                  . . .
              ]
        }
      . . .
    ]
  }
. . .
]
```

MongoDB
Limitations

MongoDB High availability is not safe!!

> Data loss depending the consistency level you choose

MongoDB consistency levels

> Unacknowledged: Unsafe - 42% of data loss

> Acknowledged: Unsafe

> Journaled: Unsafe

> Fsynced: Unsafe

> Replica Acknowledged: Unsafe

> Only majority is safe

https://aphyr.com/posts/322-jepsen-mongodb-stale-reads

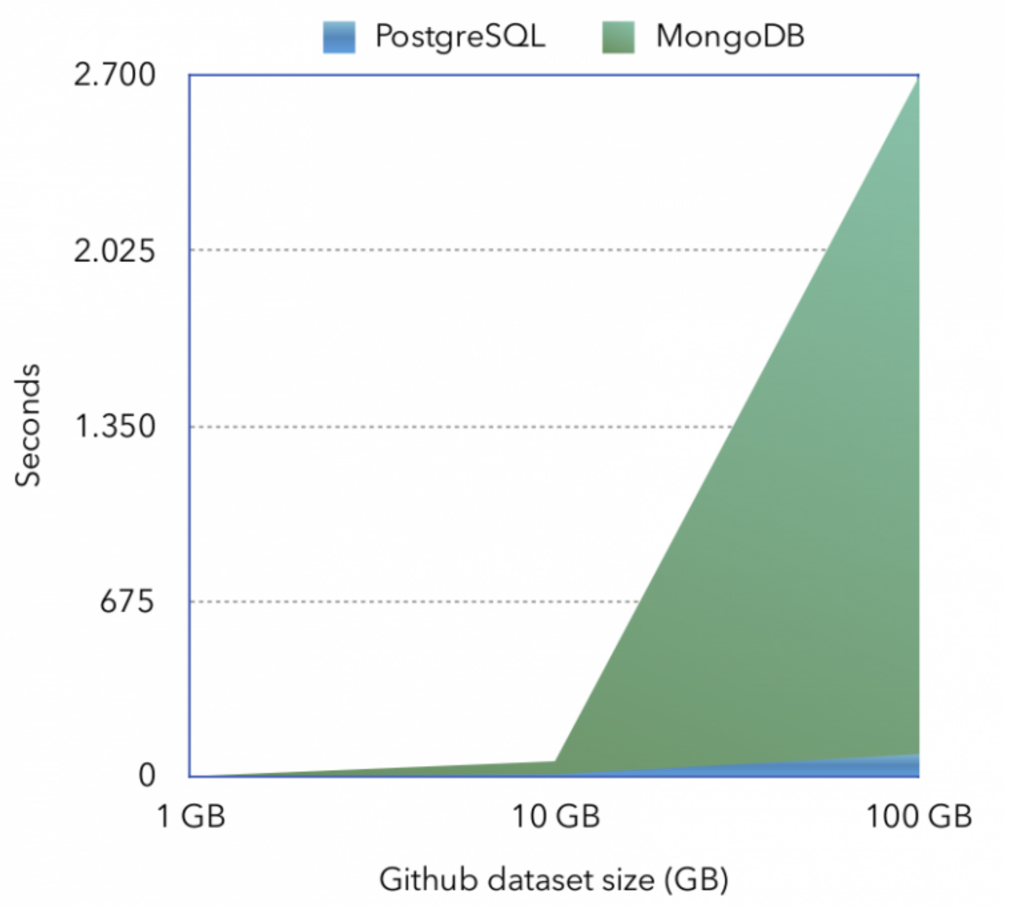
MongoDB
Limitations

BI query performances issues

MongoDB aggregation framework is very slow
> Need to scan multiple documents
> Lots of I/O required to answer the query

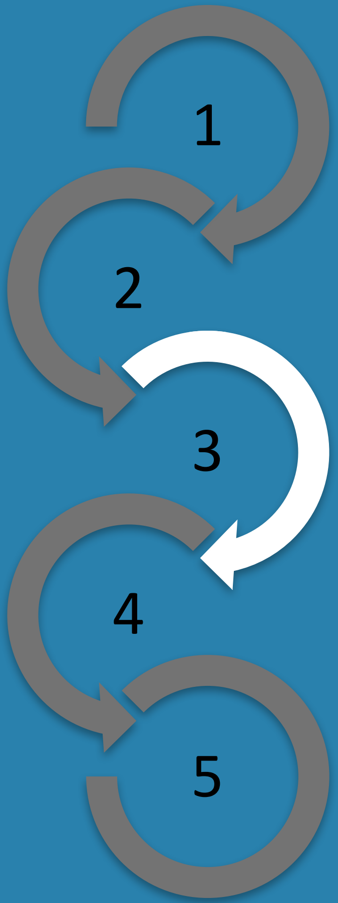
Aggregation on a relational design is “100x faster”

Solution for MongoDB BI queries?
> Implement a relational schema!!



ToroDB

> What is ToroDB?
> How it works?
> Why ToroDB?



ToroDB
What is ToroDB?

The first database that merges the scalability of a NoSQL with the reliability of SQL

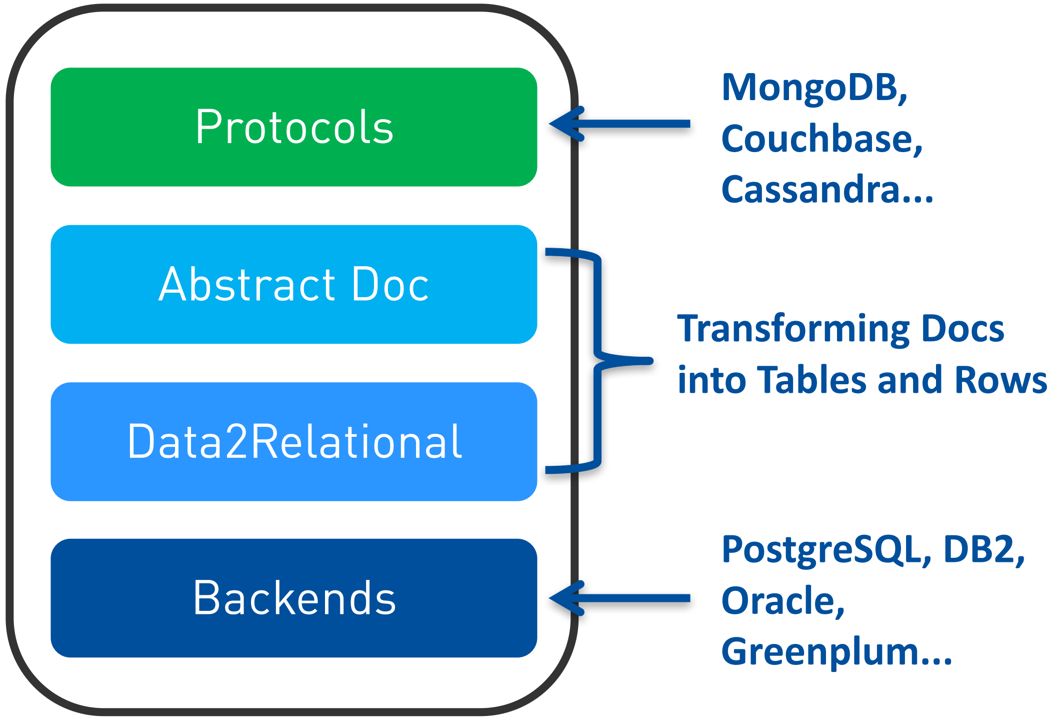
ToroDB
What is ToroDB?

Open source, document-oriented, JSON database that runs on top of PostgreSQL

BI connector for MongoDB

JSON documents are stored relationally
> Significant storage
> I/O savings

MongoDB data is persisting in tables and rows within a SQL database


Protocols
Abstract Doc
Data2Relational
Backends
MongoDB, Couchbase, Cassandra...
Transforming Docs into Tables and Rows
PostgreSQL, DB2, Oracle, Greenplum...

ToroDB
How it works?

ToroDB transforms documents to relational tables

> Data is stored in tables

> ToroDB analyzes every incoming document and separates metadata (schema) form data (tuples)

> 1+ tables per MongoDB collections

> ToroDB creates a RDBMS catalog schema per MongoDB database

> Dynamic and implicit schema generation

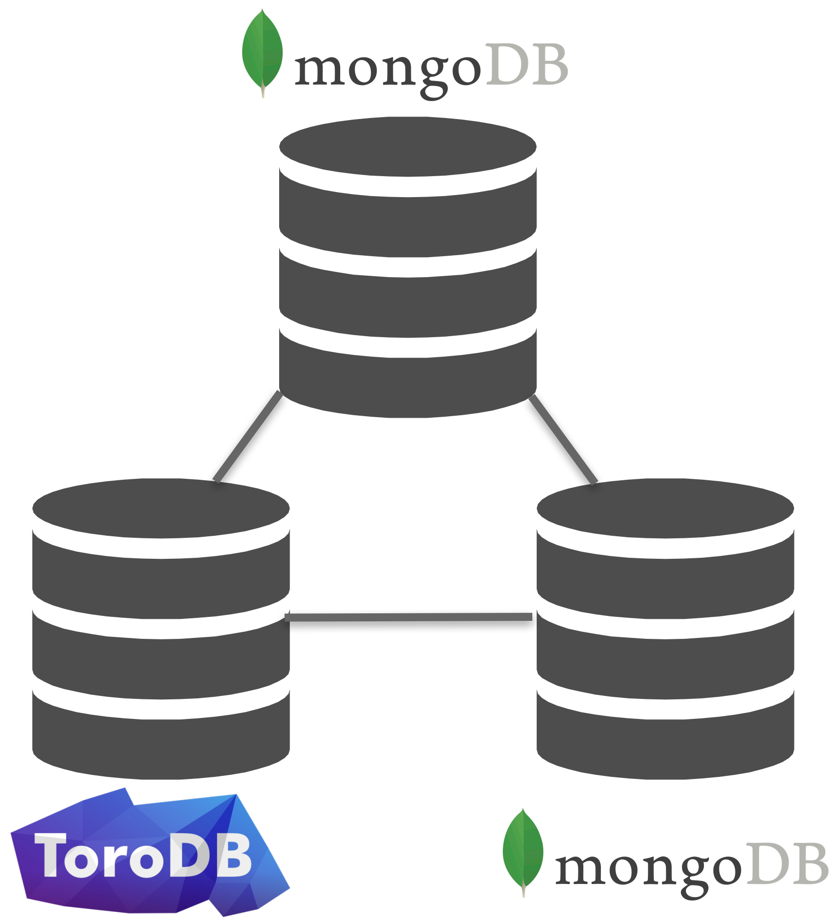
Full compatibility with MongoDB

> API programs, clients

> CRUD operations including UPDATE

ToroDB
How it works?

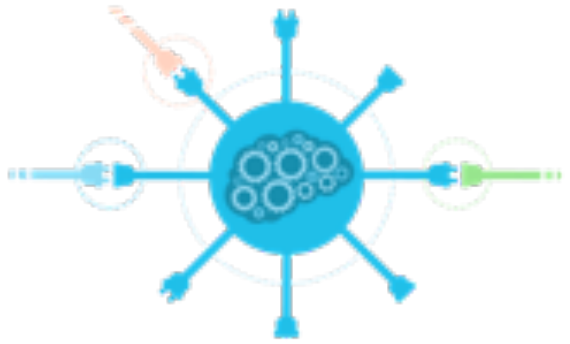
ToroDB can work **as a secondary node on a MongoDB replica set**



ToroDB
Why ToroDB?

Native SQL BI Connector

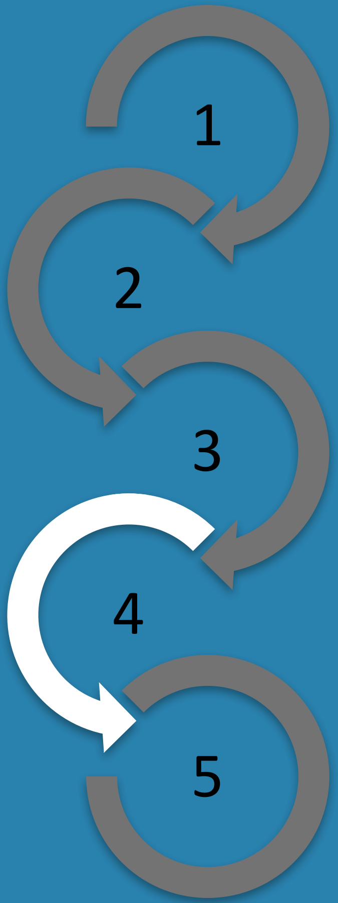
Data Integration Platform: SQL and NoSQL apps in the same RDBMS



Apps: Write data with Mongo API, query with SQL!

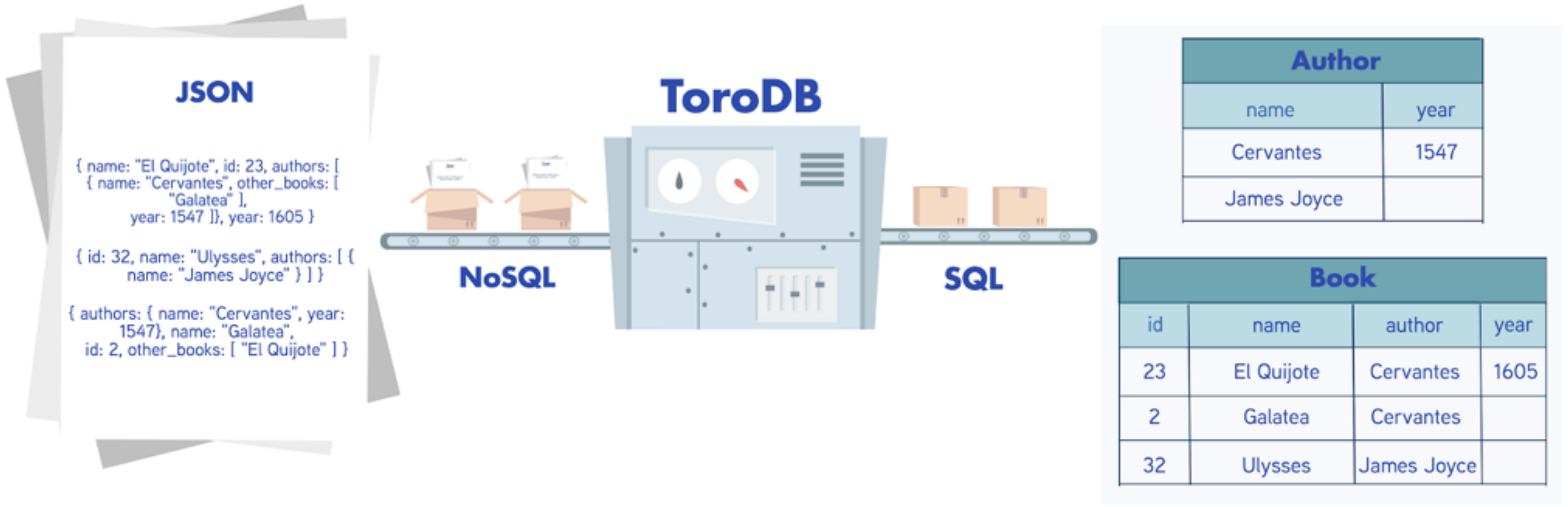
Migration: from MongoDB to PostgreSQL

> Overview
> Prerequisites
> Configuration
> Demo



Migration: from MongoDB to PostgreSQL

Overview



```
{ name: "El Quijote", id: 23, authors: [
  { name: "Cervantes", other_books: [
     "Galatea" ],
     year: 1547 ]}, year: 1605 }

{ id: 32, name: "Ulysses", authors: [ {
     name: "James Joyce" } ] }

{ authors: { name: "Cervantes", year:
     1547}, name: "Galatea",
  id: 2, other_books: [ "El Quijote" ] }
```

Author

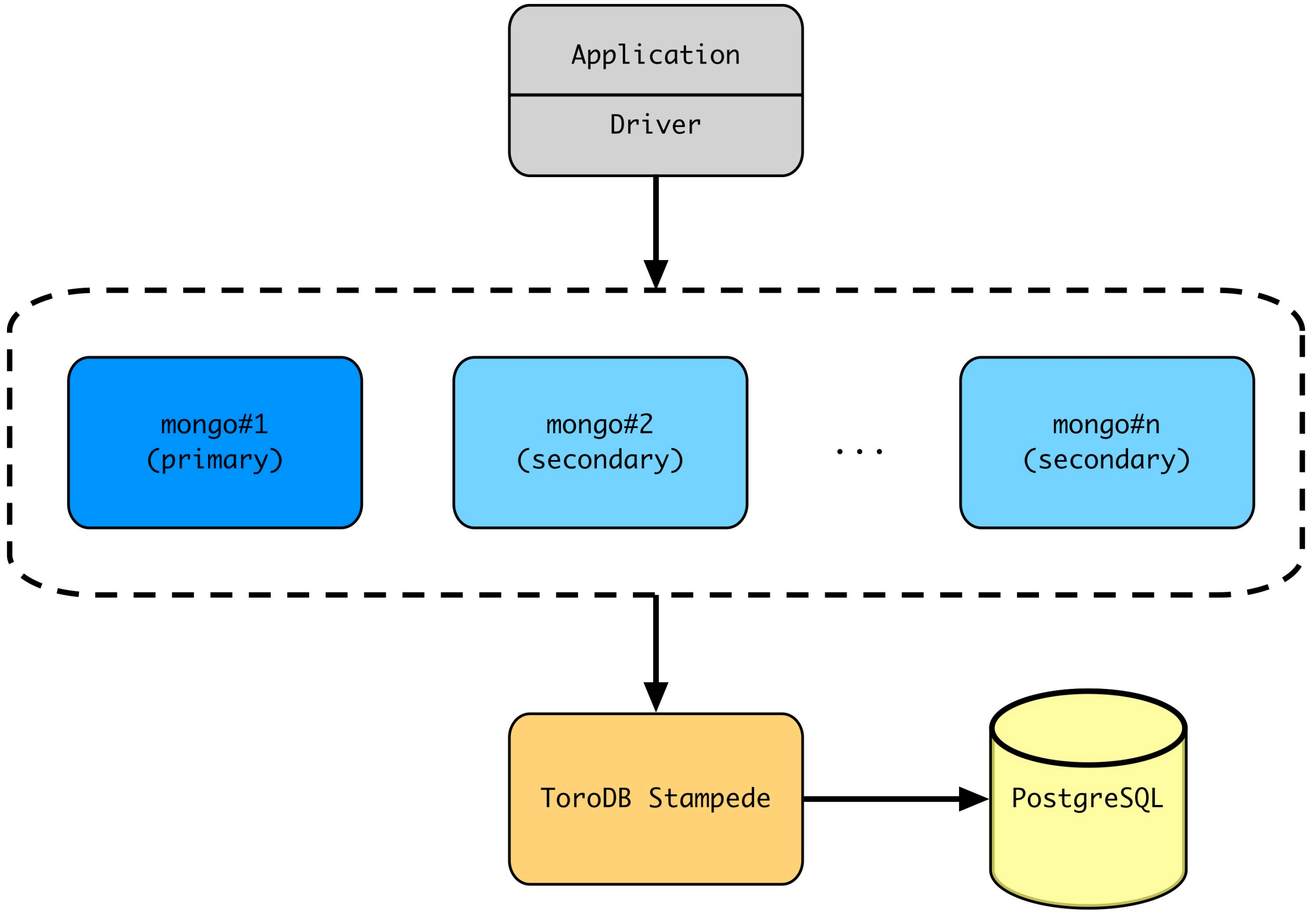
name	year
Cervantes	1547
James Joyce	

Book

id	name	author	year
23	El Quijote	Cervantes	1605
2	Galatea	Cervantes	
32	Ulysses	James Joyce	

Migration: from MongoDB to PostgreSQL
Overview

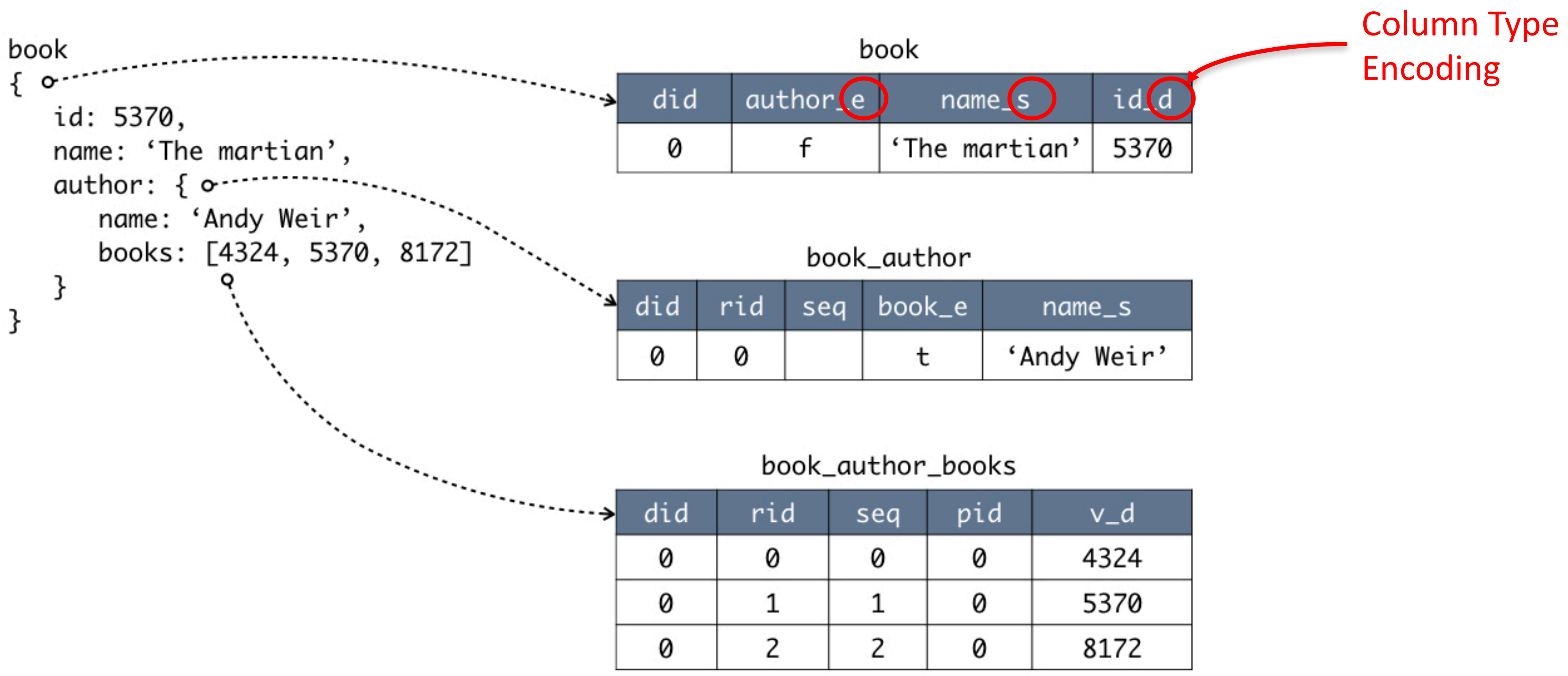
ToroDB Stampeded uses MongoDB replica set oplog to keep track of the modifications in MongoDB



Migration: from MongoDB to PostgreSQL

Overview

During the replication ToroDB Stampeded transforms JSON documents into a relational schema



```
book
{
    id: 5370,
    name: 'The martian',
    author: {
        name: 'Andy Weir',
        books: [4324, 5370, 8172]
    }
}
```

book

did	author_e	name_s	id_d
0	f	'The martian'	5370

book_author

did	rid	seq	book_e	name_s
0	0		t	'Andy Weir'

book_author_books

did	rid	seq	pid	v_d
0	0	0	0	4324
0	1	1	0	5370
0	2	2	0	8172

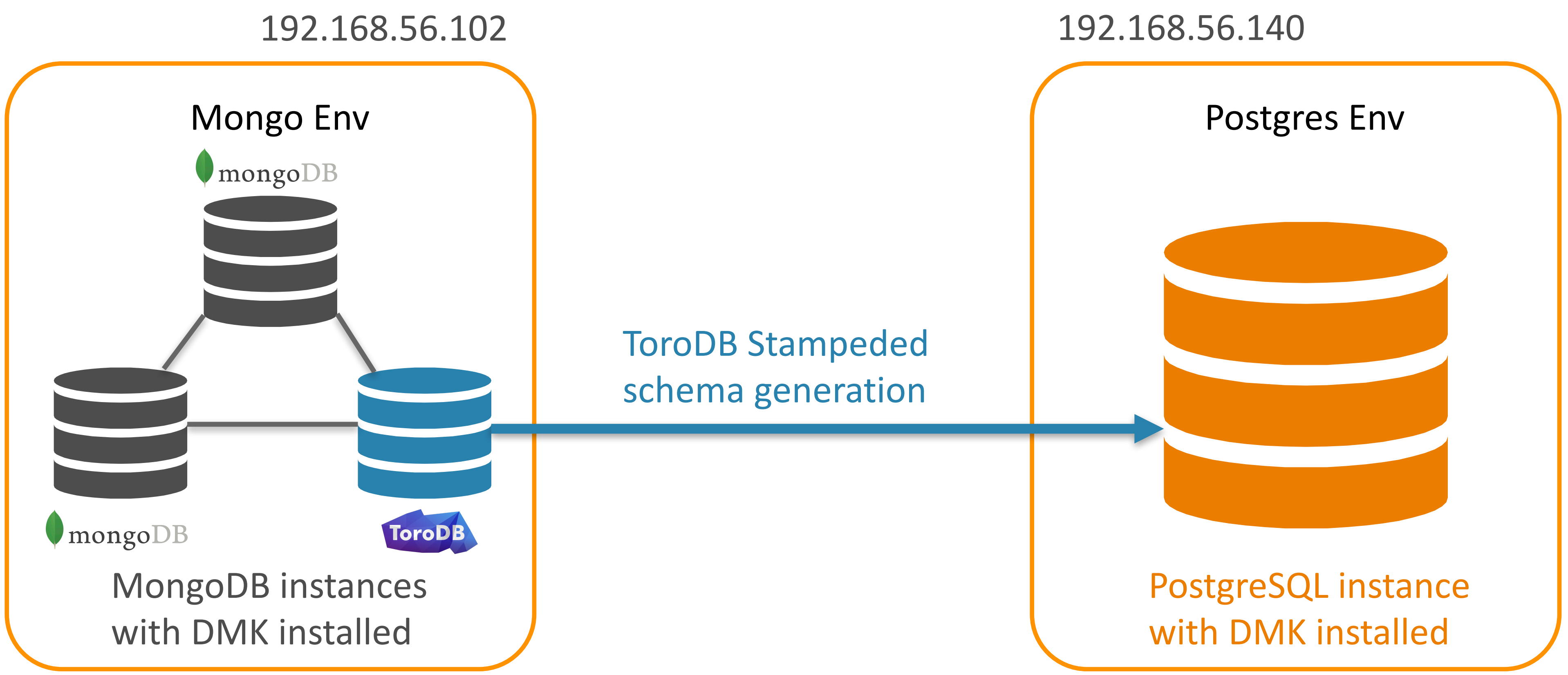
Migration: from MongoDB to PostgreSQL
Prerequisites

Runtime dependencies

Technology	Description
MongoDB	Install and start MongoDB instances with the replication features (dbi services best practices installation)
Replica sets configuration	ToroDB Stampede receives data from a MongoDB replica set. A single-node replica set is sufficient.
PostgreSQL	Install and start a PostgreSQL instance. Create a dedicated user and database. (dbi services best practices)
Java	ToroDB Stampede is written in Java so a Java Runtime Environment (JRE) required to run it. Java 8 is recommended.

Migration: from MongoDB to PostgreSQL
Configuration

Architecture


192.168.56.102
192.168.56.140
Mongo Env
mongoDB
mongoDB
ToroDB
MongoDB instances with DMK installed
ToroDB Stampeded schema generation
Postgres Env
PostgreSQL instance with DMK installed

Migration: from MongoDB to PostgreSQL
Configuration

After installation, export $TOROHOME variable

```
mehdi@MacBook-Pro:/u00/app/torodb/ export TOROHOME /u00/app/torodb/torodb-stampede-1.0.0-
beta2"

mehdi@MacBook-Pro:/u00/app/torodb/ echo $TOROHOME
/u00/app/torodb/torodb-stampede-1.0.0-beta2
```

Create and adapt the ToroDB configuration file (YAML)

```
mehdi@MacBook-Pro:/u00/app/torodb/ /u00/app/torodb/torodb-stampede-1.0.0-beta2/bin/torodb-
stampede -l > ../conf/torodb.yaml

mehdi@MacBook-Pro:/u00/app/torodb/ vi ../conf/torodb.yaml
```

Migration: from MongoDB to PostgreSQL
Configuration

ToroDB Stampeded reads databases credentials from the .toropass file

Create the .toropass file in the home directory

```
mehdi@MacBook-Pro:/u00/app/torodb/ echo ”<host>:<port>:<database>:<user>:<PASSWD>" >
"$HOME/.toropass"

mehdi@MacBook-Pro:/u00/app/torodb/ chmod 0400 ”~/.toropass”
```

Migration: from MongoDB to PostgreSQL
Configuration

Custom configuration for ToroDB Stampeded on startup

> Command line options
> Configuration file

Recommended to use a configuration file

```
mehdi@MacBook-Pro:/u00/app/torodb/ ./torodb-stampede -c myconfiguration.yml
```

Print the current configuration (YAML)

```
mehdi@MacBook-Pro:/u00/app/torodb/ ./torodb-stampede -l
```

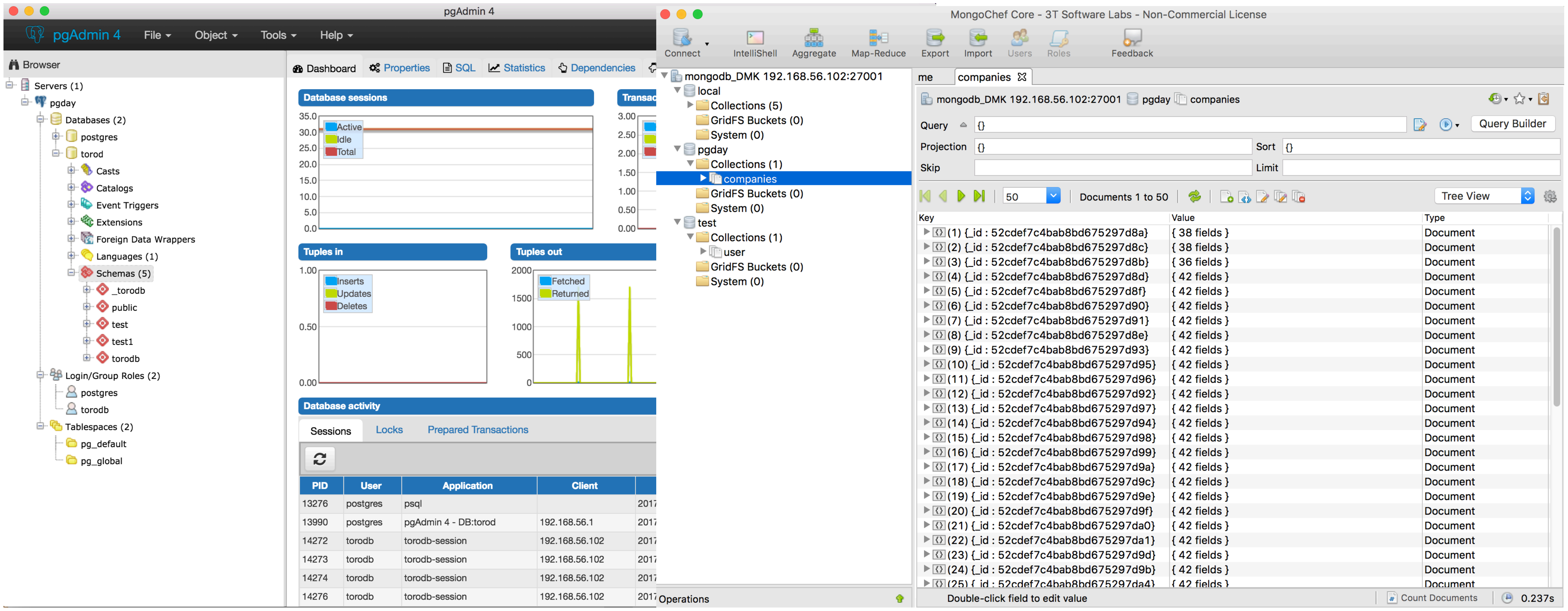
Migration: from MongoDB to PostgreSQL

Demo

Steps	Description
1	Java 8 (JRE) Installation and Configuration: http://download.oracle.com/otn-pub/java/jdk/8u131-b11/d54c1d3a095b4ff2b6607d096fa80163/jre-8u131-linux-x64.tar.gz
2	PostgreSQL Installation and Configuration: - Modify, if needed, PostgreSQL instance configuration: /u02/pgdata/PG1/postgresql.conf https://www.torodb.com/stampede/docs/1.0.0-beta2/configuration/postgresql-configuration-tips/ - Create user torodb and database torod (with password) - Test the connection with new user and databases - Adapt pg_hba.conf for new connections
3	MongoDB Installation and Configuration: - Configure and initialize MongoDB replication: replset = torodb - Check replication config. (primary and secondary) - Import data into MongoDB
4	ToroDB Stampede Installation and Configuration: - torodb.yaml configuration file creation - .toropass file creation for creation - Start ToroDB Stampeded

Migration: from MongoDB to PostgreSQL

Demo



pgAdmin 4

pgAdmin 4 File Object Tools Help

Browser

- Servers (1)
  - pgday
    - Databases (2)
      - postgres
      - torod
        - Casts
        - Catalogs
        - Event Triggers
        - Extensions
        - Foreign Data Wrappers
        - Languages (1)
        - Schemas (5)
          - _torodb
          - public
          - test
          - test1
          - torodb
    - Login/Group Roles (2)
      - postgres
      - torodb
    - Tablespaces (2)
      - pg_default
      - pg_global

Dashboard | Properties | SQL | Statistics | Dependencies

Database activity

Sessions | Locks | Prepared Transactions

PID	User	Application	Client	
13276	postgres	psql		2017
13990	postgres	pgAdmin 4 - DB:torod	192.168.56.1	2017
14272	torodb	torodb-session	192.168.56.102	2017
14273	torodb	torodb-session	192.168.56.102	2017
14274	torodb	torodb-session	192.168.56.102	2017
14276	torodb	torodb-session	192.168.56.102	2017

MongoChef Core - 3T Software Labs - Non-Commercial License

Connect IntelliShell Aggregate Map-Reduce Export Import Users Roles Feedback

- mongodb_DMK 192.168.56.102:27001
  - local
    - Collections (5)
    - GridFS Buckets (0)
    - System (0)
  - pgday
    - Collections (1)
      - companies
    - GridFS Buckets (0)
    - System (0)
  - test
    - Collections (1)
      - user
    - GridFS Buckets (0)
    - System (0)

Operations

me | companies

mongodb_DMK 192.168.56.102:27001 pgday companies

Query {} Query Builder

Projection {} Sort {}

Skip Limit

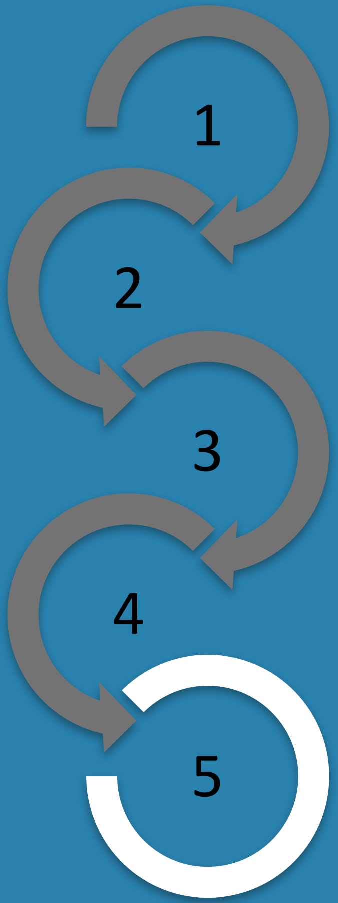
50 Documents 1 to 50 Tree View

Key	Value	Type
(1) {_id : 52cdef7c4bab8bd675297d8a}	{ 38 fields }	Document
(2) {_id : 52cdef7c4bab8bd675297d8c}	{ 38 fields }	Document
(3) {_id : 52cdef7c4bab8bd675297d8b}	{ 36 fields }	Document
(4) {_id : 52cdef7c4bab8bd675297d8d}	{ 42 fields }	Document
(5) {_id : 52cdef7c4bab8bd675297d8f}	{ 42 fields }	Document
(6) {_id : 52cdef7c4bab8bd675297d90}	{ 42 fields }	Document
(7) {_id : 52cdef7c4bab8bd675297d91}	{ 42 fields }	Document
(8) {_id : 52cdef7c4bab8bd675297d8e}	{ 42 fields }	Document
(9) {_id : 52cdef7c4bab8bd675297d93}	{ 42 fields }	Document
(10) {_id : 52cdef7c4bab8bd675297d95}	{ 42 fields }	Document
(11) {_id : 52cdef7c4bab8bd675297d96}	{ 42 fields }	Document
(12) {_id : 52cdef7c4bab8bd675297d92}	{ 42 fields }	Document
(13) {_id : 52cdef7c4bab8bd675297d97}	{ 42 fields }	Document
(14) {_id : 52cdef7c4bab8bd675297d94}	{ 42 fields }	Document
(15) {_id : 52cdef7c4bab8bd675297d98}	{ 42 fields }	Document
(16) {_id : 52cdef7c4bab8bd675297d99}	{ 42 fields }	Document
(17) {_id : 52cdef7c4bab8bd675297d9a}	{ 42 fields }	Document
(18) {_id : 52cdef7c4bab8bd675297d9c}	{ 42 fields }	Document
(19) {_id : 52cdef7c4bab8bd675297d9e}	{ 42 fields }	Document
(20) {_id : 52cdef7c4bab8bd675297d9f}	{ 42 fields }	Document
(21) {_id : 52cdef7c4bab8bd675297da0}	{ 42 fields }	Document
(22) {_id : 52cdef7c4bab8bd675297da1}	{ 42 fields }	Document
(23) {_id : 52cdef7c4bab8bd675297d9d}	{ 42 fields }	Document
(24) {_id : 52cdef7c4bab8bd675297d9b}	{ 42 fields }	Document
(25) {_id : 52cdef7c4bab8bd675297da4}	{ 42 fields }	Document

Double-click field to edit value Count Documents 0.237s

Conclusion

> Advantages vs Drawbacks



Conclusion
Advantages vs Drawbacks

✓

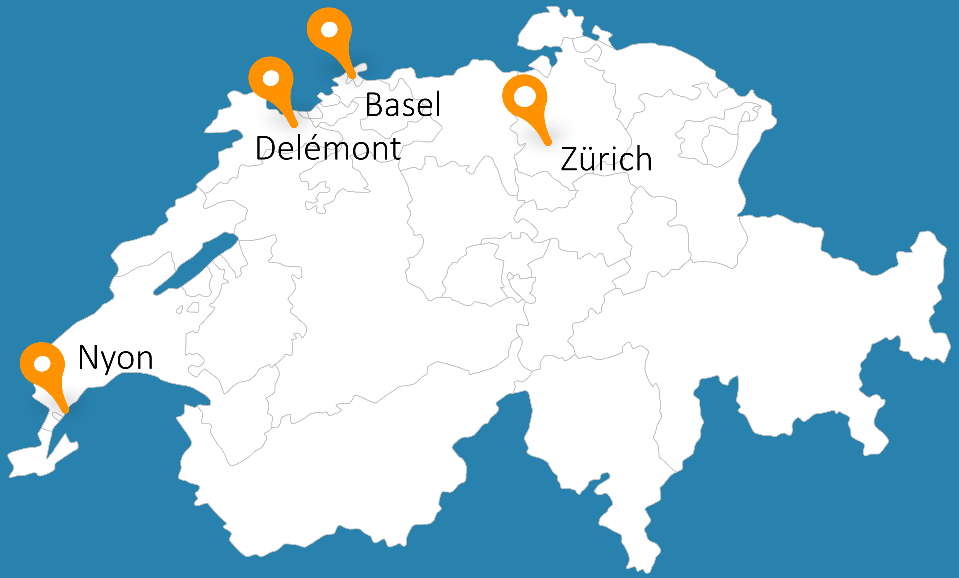
- ✓ Fast and Powerful
- ✓ Dynamic schema generation
- ✓ Dynamic changes in the schema
- ✓ Open source and free!!
- ✓ Cross-platform: Linux, Windows

✗

- ✗ Only supports PostgreSQL as RDBMS backend
- ✗ Need improvements for different schema in the same mongo collection

Any questions?

Please do ask!



We would love to boost your IT-Infrastructure

How about you?