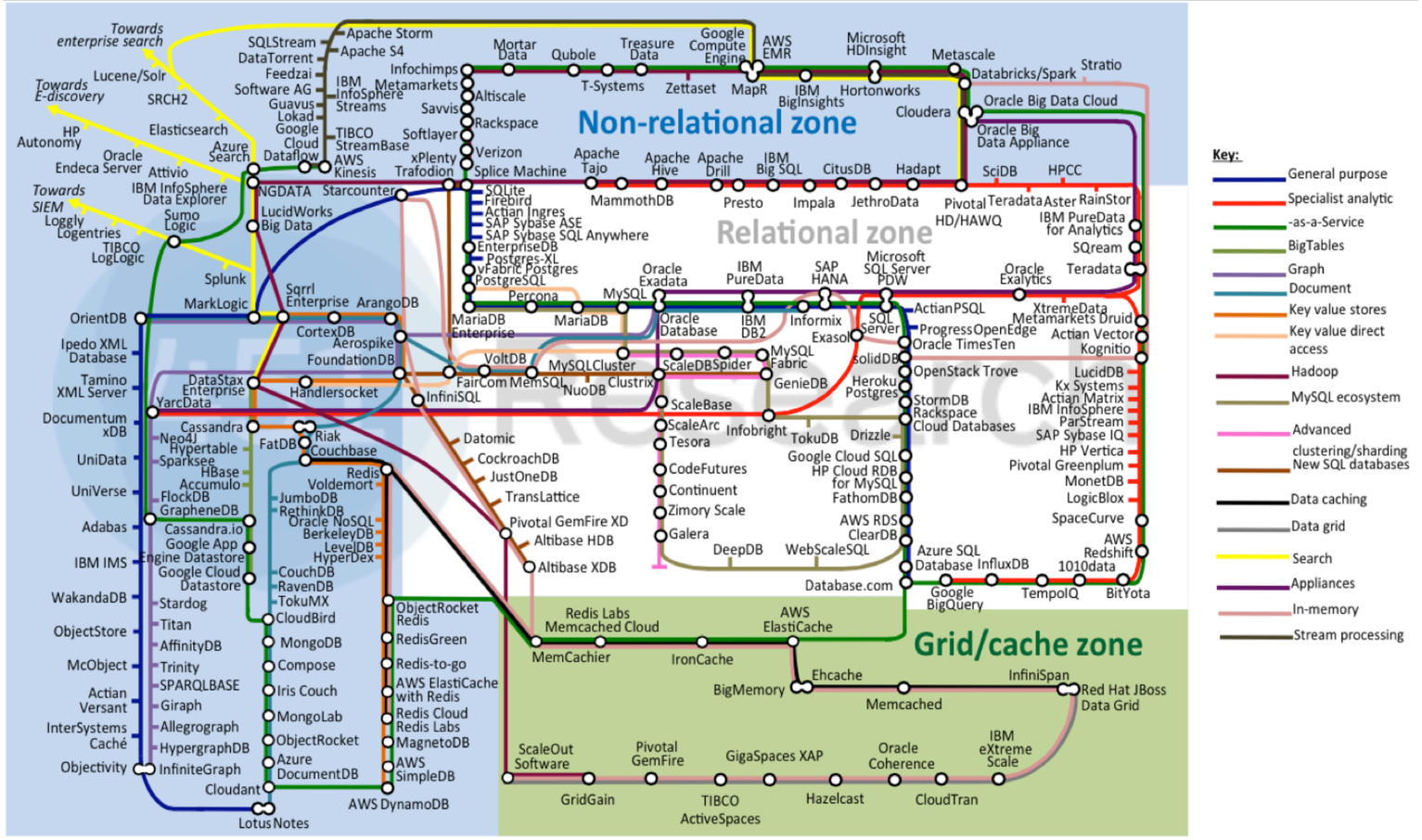
NoSQL and MongoDB:

JADS Master: Data Engineering



Part III: MongoDB

- Data model
- Querying
- Sharding
- Caveats

MongoDB is web scale

MongoDB (from humongous) is a free and open-source cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemas.

-- Wikipedia

Data model: Document oriented

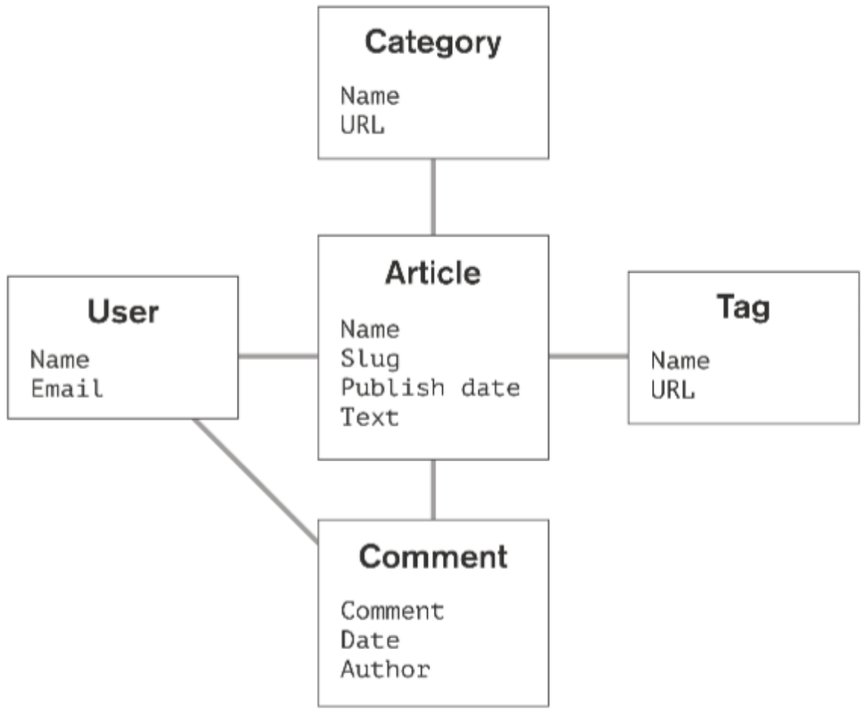
JSON Documents as data.

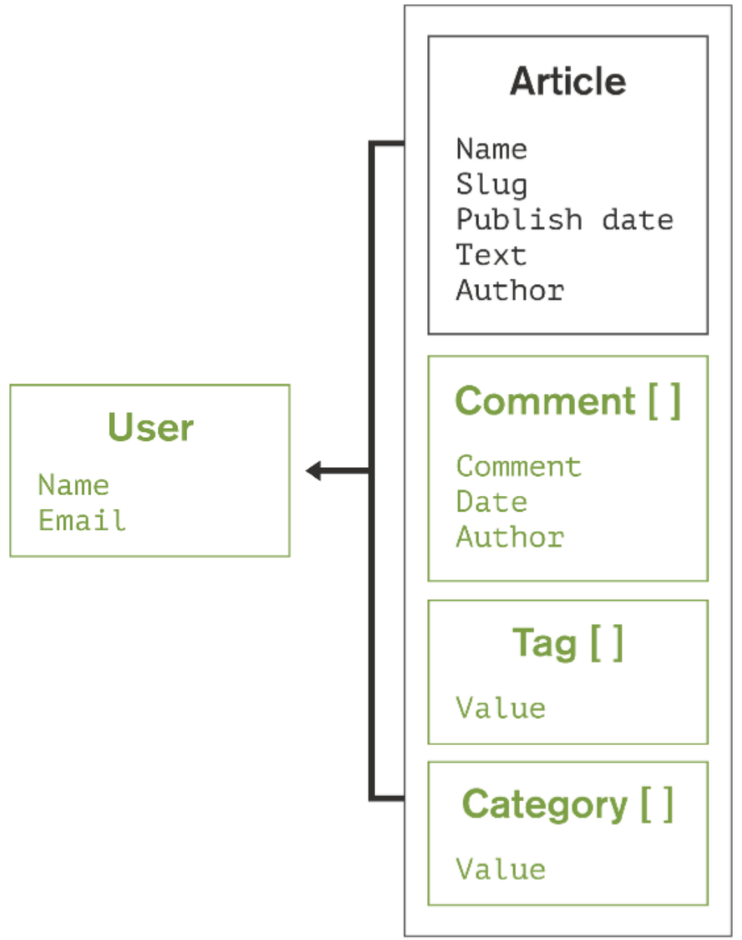
Document store:

- MongoDB

```
{
  "arguments" : { "number" : 10 },
  "url" : "http://localhost:8080/restty-tester/collection",
  "method" : "POST",
  "header" : {
    "Content-Type" : "application/json"
  },
  "body" : [
    {
      "id" : 0,
      "name" : "name 0",
      "description" : "description 0"
    },
    {
      "id" : 1,
      "name" : "name 1",
      "description" : "description 1"
    }
  ],
  "output" : "json"
}
```

Data model: Document oriented vs. Relational





Per collection:

- Create documents
- Read documents
- Update documents
- Delete documents

NB: All write operations are atomic on the document level.

MongoDB CRUD: Insert

MongoDB CRUD: Read

```
db.users.find(
    { age: { $gt: 18 } },
    { name: 1, address: 1 }
).limit(5)
```

← collection
← query criteria
← projection
← cursor modifier

MongoDB CRUD: Update

```
db.users.update(
    { age: { $gt: 18 } },
    { $set: { status: "A" } },
    { multi: true }
)
```

collection

update criteria

update action

update option

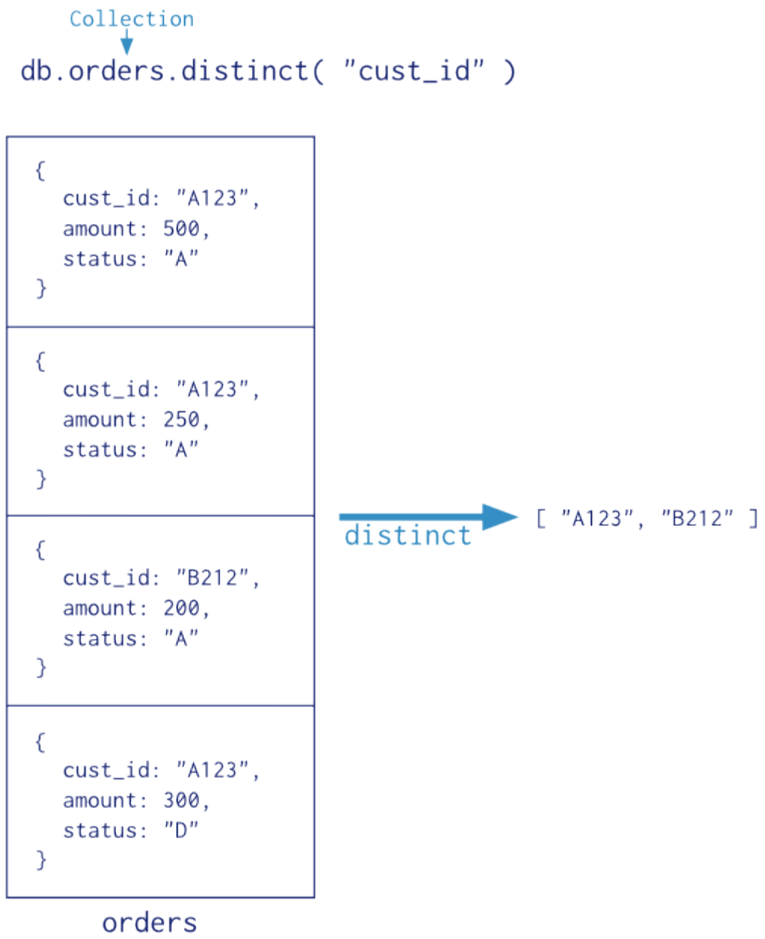
MongoDB CRUD: Delete

```
db.users.remove(
    { status: "D" }
)
```

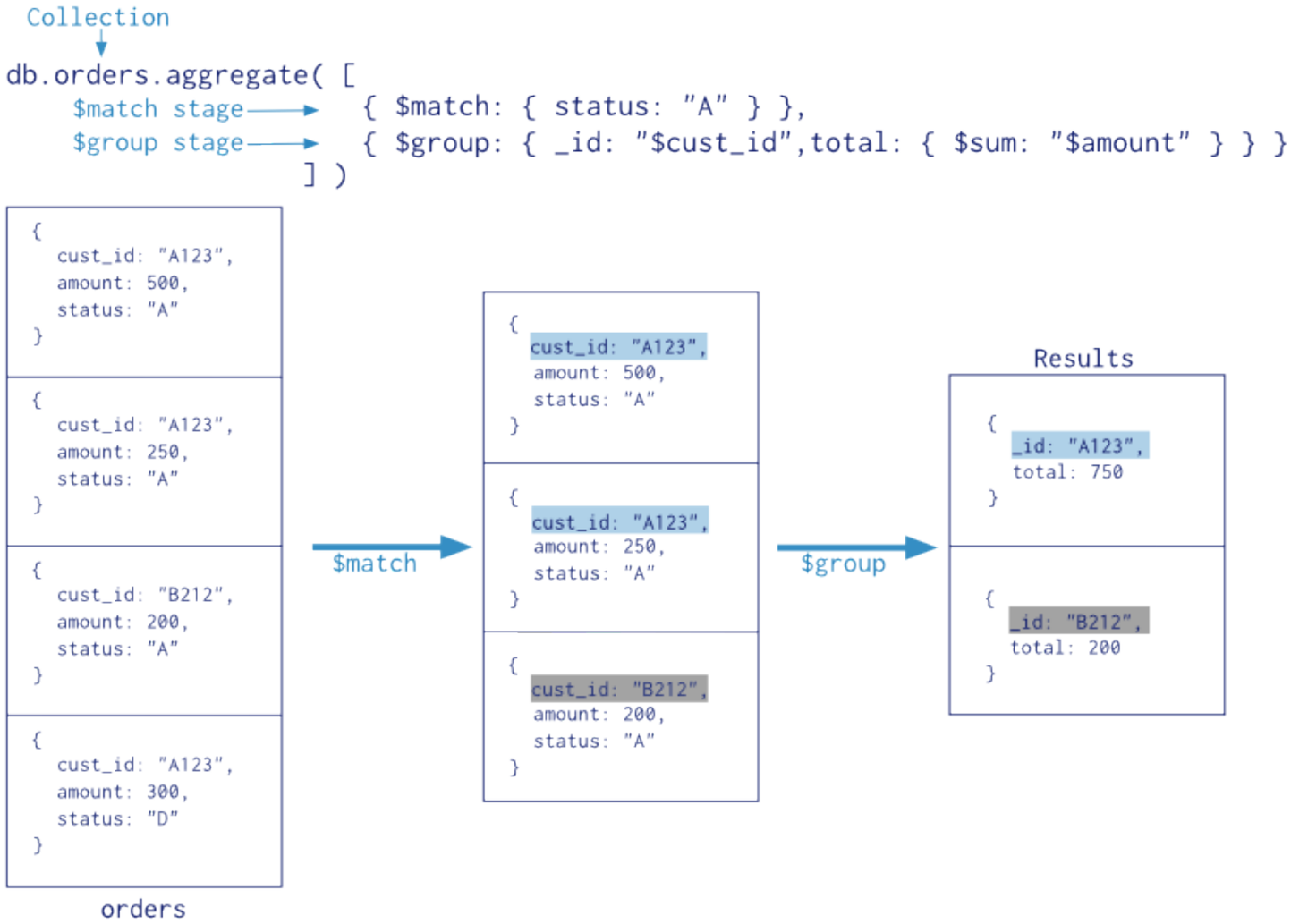
collection

remove criteria

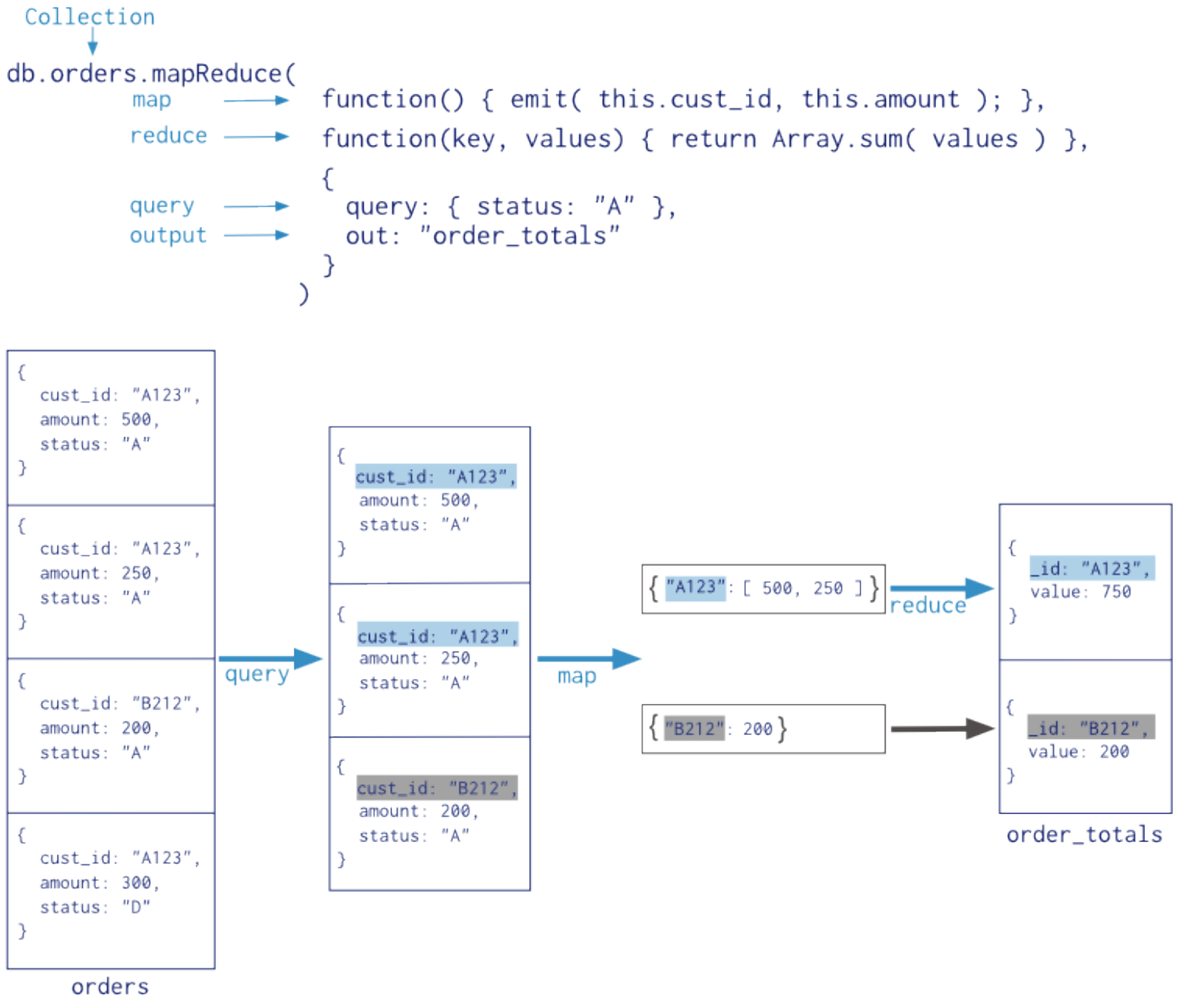
Aggregations: Single purpose, pipelineing, mr



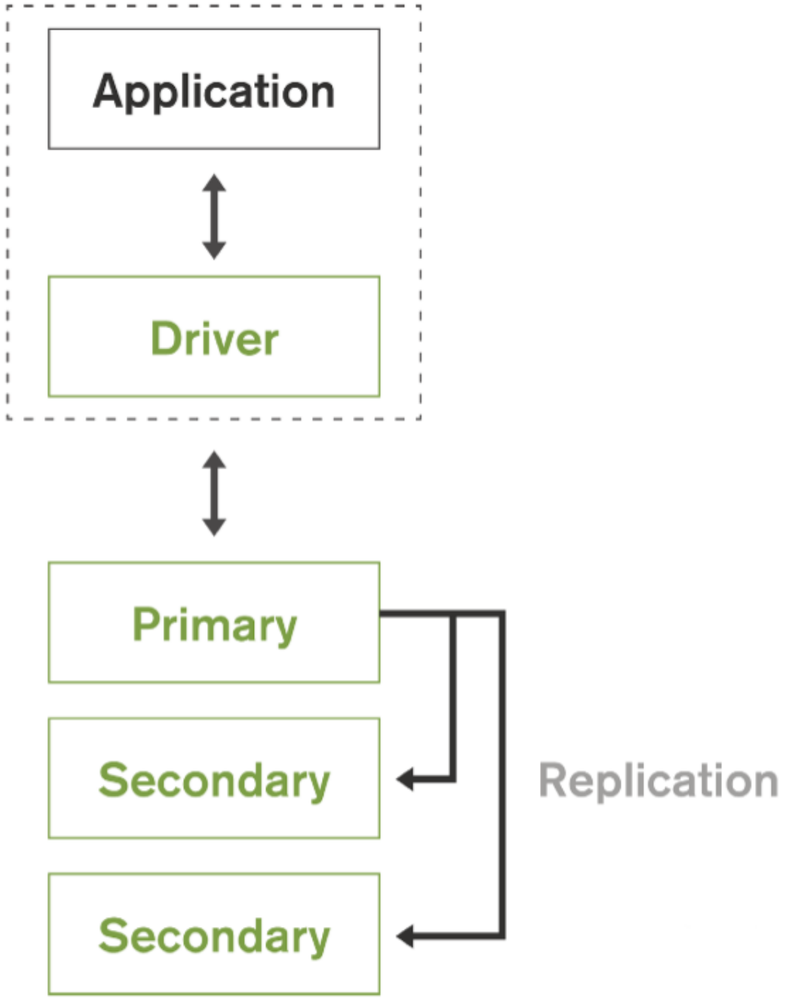
Aggregations: Single purpose, pipelineing, mr



Aggregations: Single purpose, pipelineing, mr

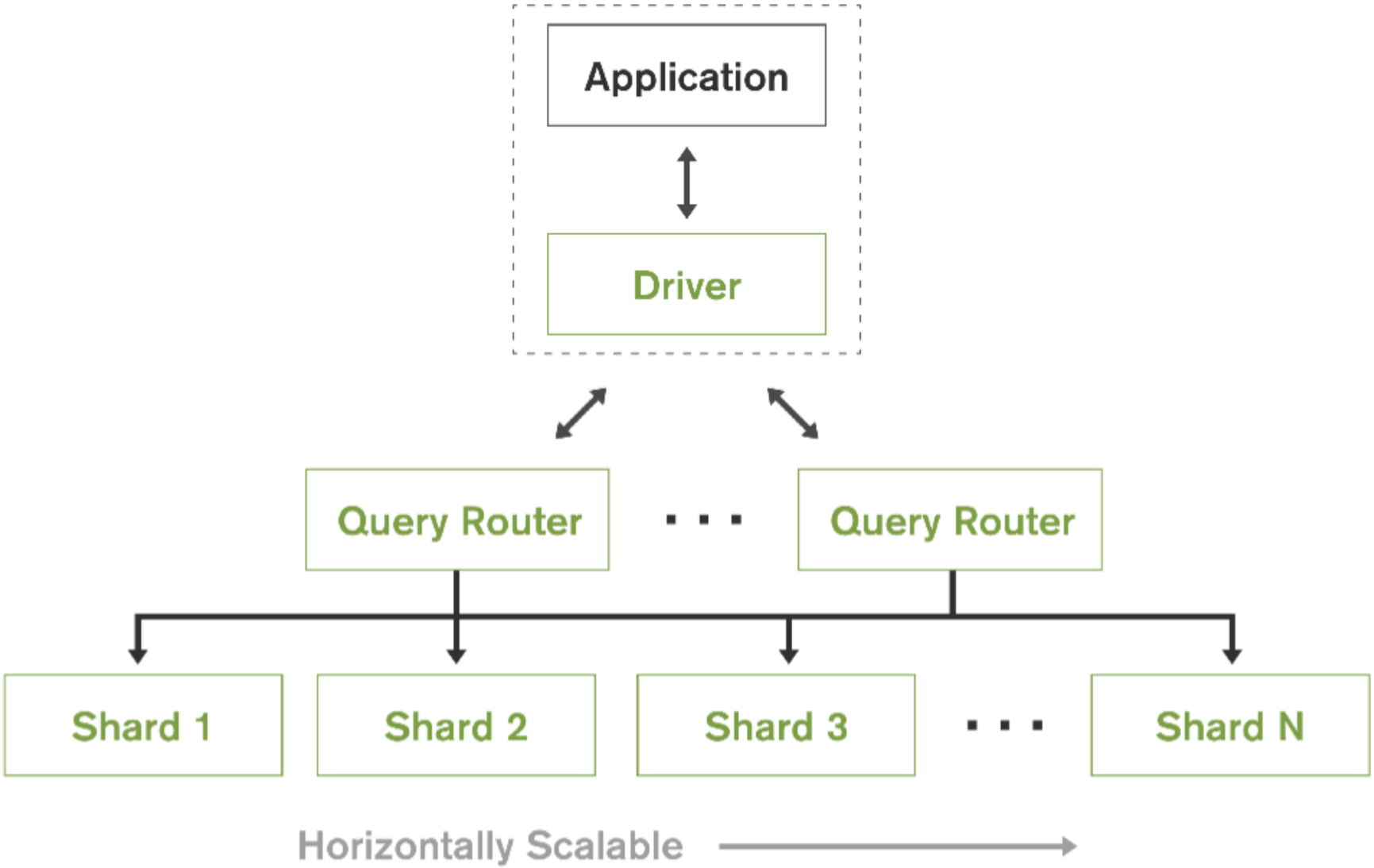


High-availability



Scalability

Scalability



Critics


YouTube NL
Search
0:36 / 5:35
Episode 1 - Mongo DB Is Web Scale
g
garlt
Subscribe 2,537
358,042 views
Add to
Share
More
2,996
84

Caveats

- Durability: Mongo will ack write succesful even though data might not be on disk
- Failure scenarios: stale reads or rollback of writes when an application can access two partitioned processes
- Concurrency control depends on storage engine: collection level/document level
- Queries against an index are not atomic and might miss documents being updated during the query

Questions?