Document Databases

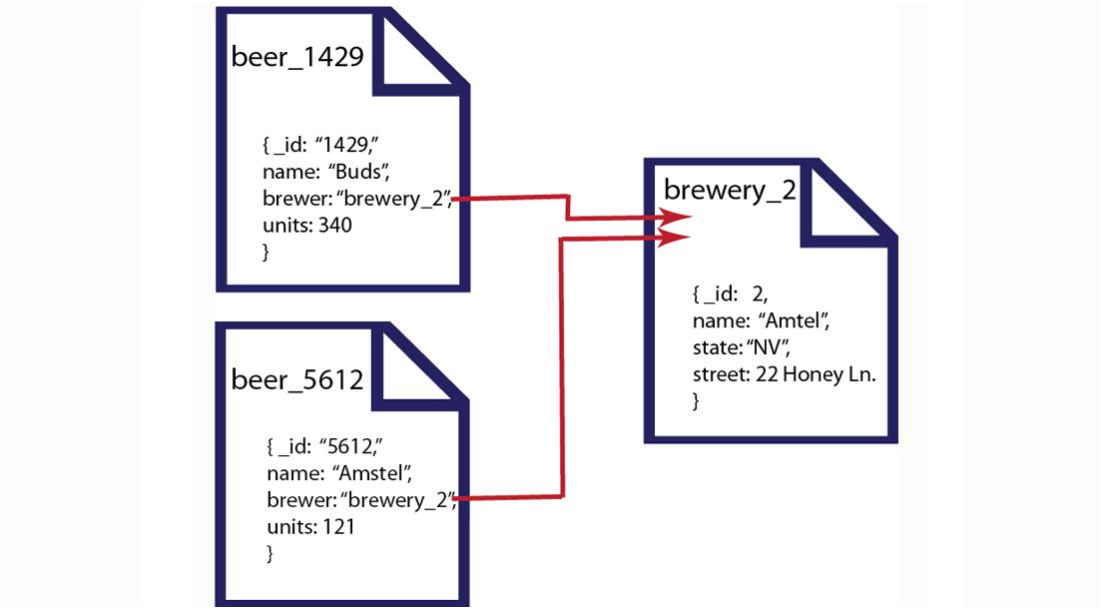
O que são e suas diferenças de bancos Relacionais

Roteiro

- O que são
- Motivação
- Quando Usar
- Replicação e Replica Sets
- Operações Básicas
- Operações de Agregação
- MongoDB em Java
- Conclusão

O que são

- São SGBD’s que utilizam documentos, em maior parte em JSON ou formatos semelhantes ao invés de Tabelas como SGBD’s relacionais.

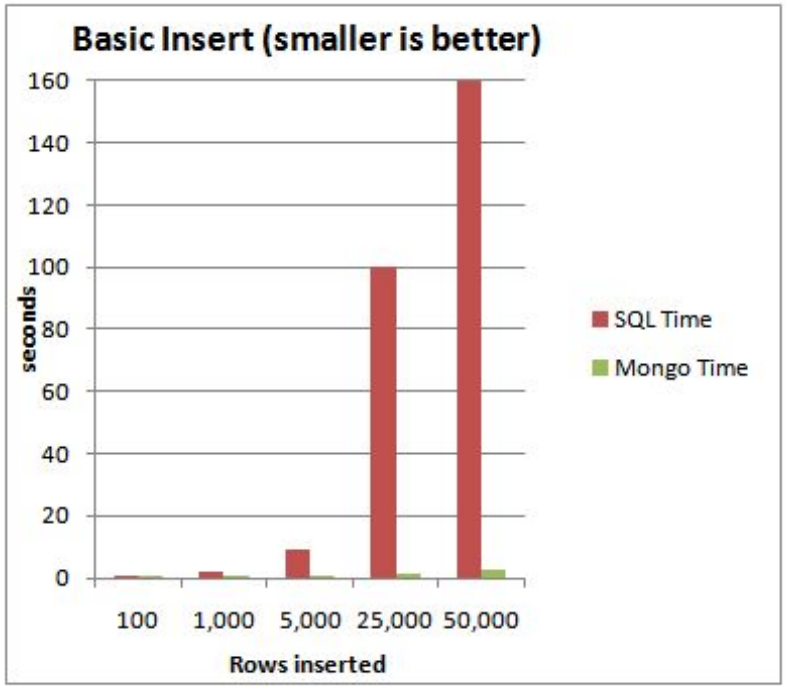


Motivação

- Bancos em SQL são muito complexos e uma modelagem correta pode demorar bastante.
- Erros na modelagem são custosos e complicados para corrigir.

“É possível exibir todos os dados de qualquer consulta única necessária em apenas um Documento?”

Vantagens

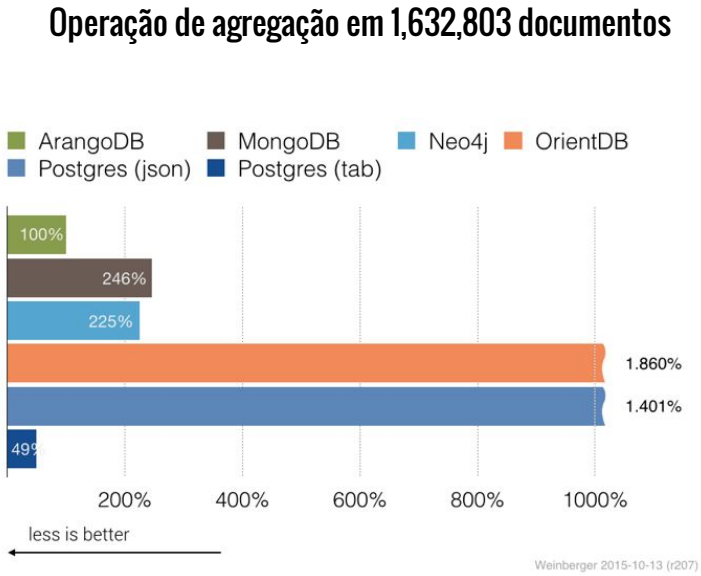


- Documentos diferentes em uma mesma coleção.
- Mais flexibilidade na alteração de tipos de documentos.
- Confecção inicial mais rápida e fácil
- Mais tolerante a erros
- Operações de inserções mais baratas

img source: http://tinyurl.com/hl7ssuo

Desvantagens

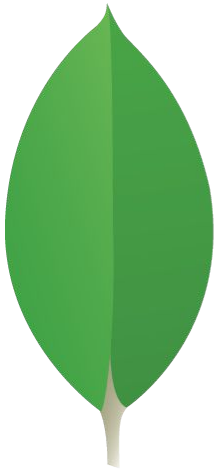
- Operações de Agregação mais custosas
- Devido a falta de relacionamento entre documentos, redundâncias se tornam necessárias
- Problema com atomicidade de operações e consistência de dados



Top10 Databases que usam Documentos

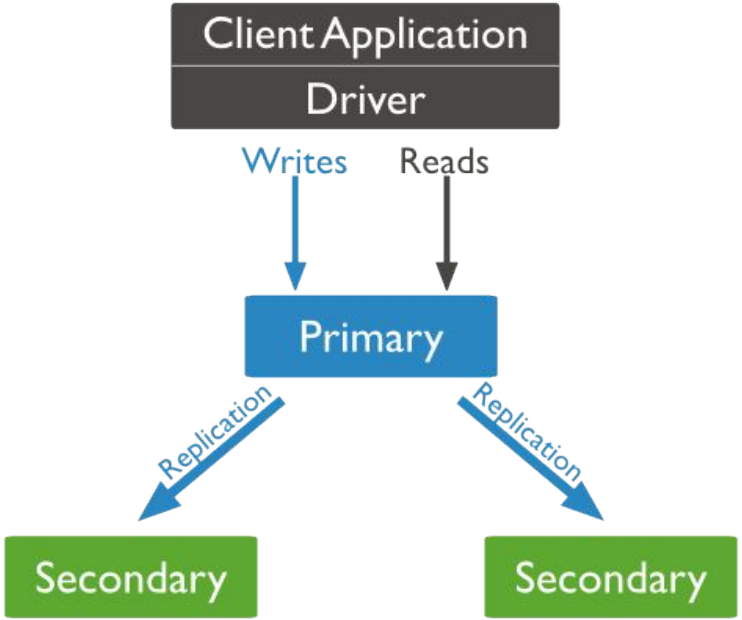
44 systems in ranking, November 2016

Rank			DBMS	Database Model	Score		
Nov 2016	Oct 2016	Nov 2015			Nov 2016	Oct 2016	Nov 2015
1.	1.	1.	MongoDB	Document store	325.48	+6.67	+20.87
2.	↑ 3.	↑ 4.	Amazon DynamoDB	Document store	29.78	+0.80	+8.04
3.	↓ 2.	3.	Couchbase	Document store	29.05	-0.25	+3.23
4.	4.	↓ 2.	CouchDB	Document store	22.66	+0.48	-3.72
5.	5.	5.	MarkLogic	Multi-model	10.22	-0.08	-0.78
6.	6.	↑ 7.	OrientDB	Multi-model	6.07	-0.17	+0.57
7.	7.	↑ 10.	RethinkDB	Document store	5.90	+0.66	+2.22
8.	8.	8.	Cloudant	Document store	4.61	+0.09	-0.75
9.	9.	↓ 6.	RavenDB	Document store	4.18	+0.12	-1.63
10.	↑ 11.	↑ 13.	Microsoft Azure DocumentDB	Document store	3.25	+0.36	+1.37

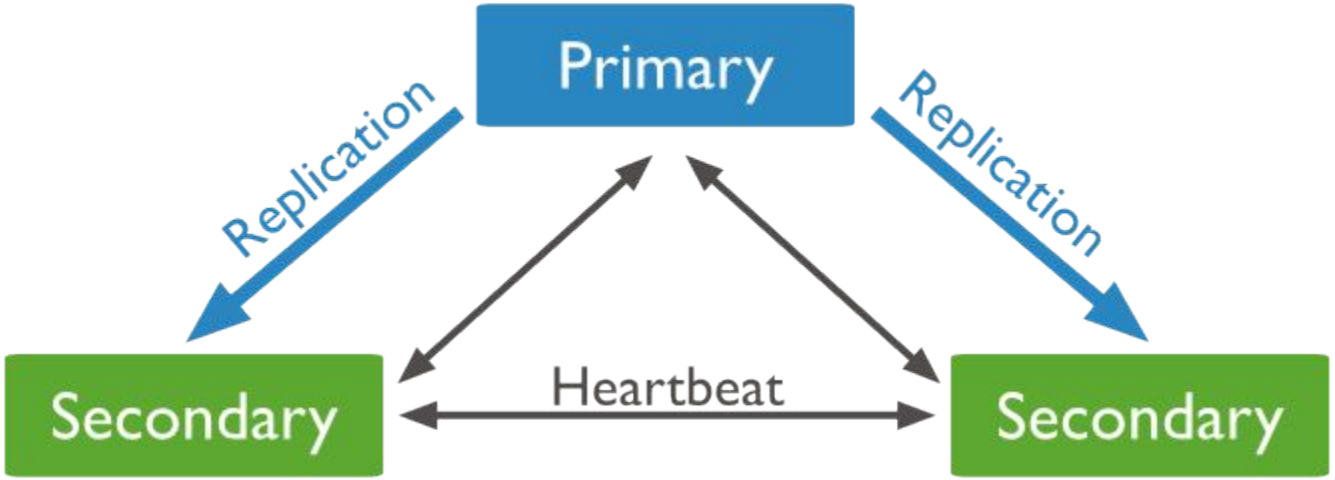


mongoDB®

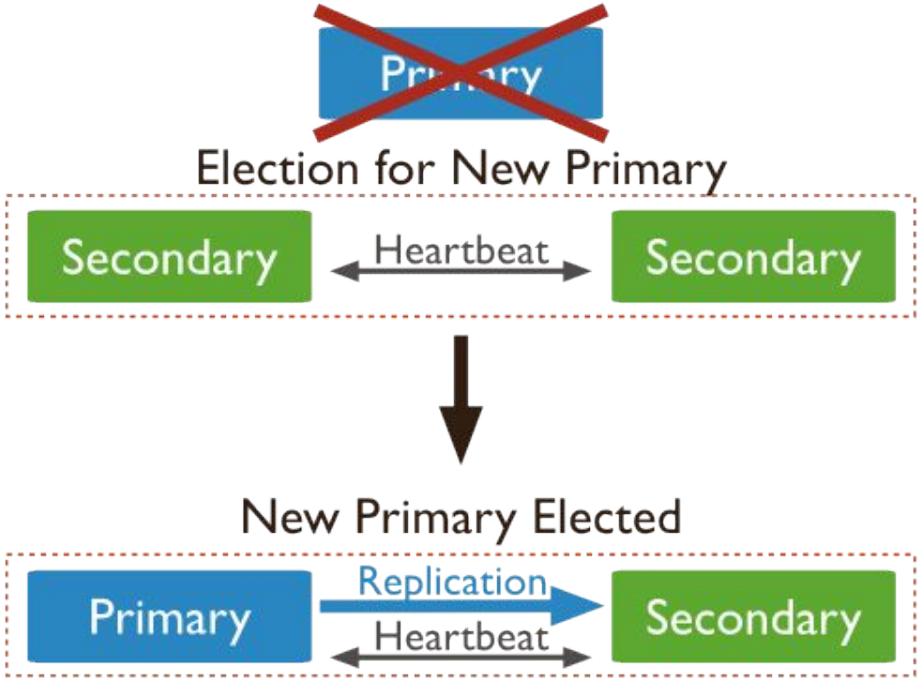
Replicação e Datasets



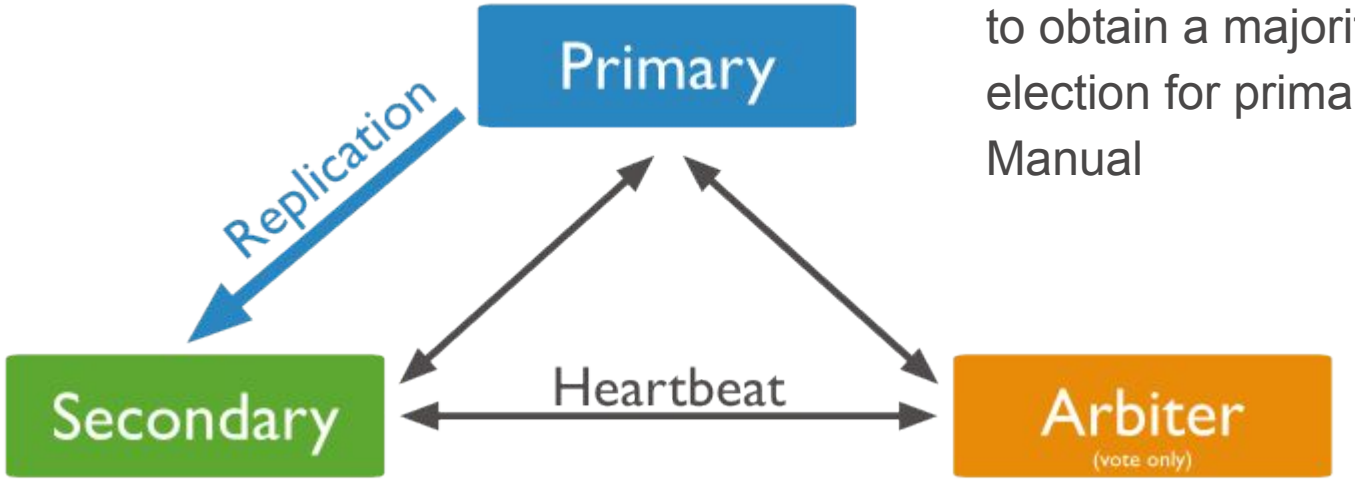
Sincronização



Automatic Failover



Arbiters



“ If your replica set has an even number of members, add an arbiter to obtain a majority of votes in an election for primary.” - MongoDB Manual

Operações Básicas - Shell

```
wrf@wrf13:~$ sudo mongod
[sudo] password for wrf:
2016-09-16T08:44:49.746-0300 I CONTROL  [initandlisten] MongoDB starting : pid=3886 port=27017 dbpath=/data/db 64-bit host=wrf13
2016-09-16T08:44:49.746-0300 I CONTROL  [initandlisten] db version v3.0.12
2016-09-16T08:44:49.746-0300 I CONTROL  [initandlisten] git version: 33934938e0e95d534cebbaff656cde916b9c3573
2016-09-16T08:44:49.746-0300 I CONTROL  [initandlisten] build info: Linux ip-10-167-97-24 3.2.0-4-amd64 #1 SMP Debian 3.2.46-1 x86_64 BOOST_LIB_VERSION=1_49
2016-09-16T08:44:49.746-0300 I CONTROL  [initandlisten] allocator: tcmalloc
2016-09-16T08:44:49.746-0300 I CONTROL  [initandlisten] options: {}
2016-09-16T08:44:49.806-0300 I JOURNAL  [initandlisten] journal dir=/data/db/journal
2016-09-16T08:44:49.806-0300 I JOURNAL  [initandlisten] recover : no journal files present, no recovery needed
2016-09-16T08:44:49.884-0300 I JOURNAL  [durability] Durability thread started
2016-09-16T08:44:49.884-0300 I JOURNAL  [journal writer] Journal writer thread started
2016-09-16T08:44:49.927-0300 I CONTROL  [initandlisten] ** WARNING: You are running this process as the root user, which is not recommended.
2016-09-16T08:44:49.927-0300 I CONTROL  [initandlisten]
2016-09-16T08:44:49.927-0300 I CONTROL  [initandlisten]
2016-09-16T08:44:49.927-0300 I CONTROL  [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.
2016-09-16T08:44:49.927-0300 I CONTROL  [initandlisten] **        We suggest setting it to 'never'
2016-09-16T08:44:49.927-0300 I CONTROL  [initandlisten]
2016-09-16T08:44:49.927-0300 I CONTROL  [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/defrag is 'always'.
2016-09-16T08:44:49.927-0300 I CONTROL  [initandlisten] **        We suggest setting it to 'never'
2016-09-16T08:44:49.927-0300 I CONTROL  [initandlisten]
2016-09-16T08:44:50.884-0300 I NETWORK  [initandlisten] waiting for connections on port 27017
```

Operações Básicas - Shell

```
> show dbs
                  0.078GB
local             0.078GB
                  0.203GB
test              0.078GB
                  0.203GB
> use test
switched to db test
> db.getCollectionNames();
[ "map_ex", "restaurantes", "system.indexes" ]
> db.createCollection("exemplo");
{ "ok" : 1 }
> db.getCollectionNames();
[ "exemplo", "map_ex", "restaurantes", "system.indexes" ]
```

Operações Básicas - Java

```
MongoClient mongo = new MongoClient();
MongoDatabase db = mongo.getDatabase("test");

//db.createCollection("exemplo");
MongoCursor<String> dbs = mongo.listDatabaseNames().iterator();
System.out.println("Banco de dados:");
while ( dbs.hasNext() ) {
    System.out.println( " - " + dbs.next());
}

System.out.println("Coleções");
MongoIterable<String> collections = db.listCollectionNames();
for ( String c : collections ){
    System.out.println(" -- " + c);
}
```

Operações Básicas - Java

```
out 28, 2016 1:54:53 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Cluster created with settings {hosts=[127.0.0.1:27017], mode=SINGLE, requiredClusterType=UNKNOWN
out 28, 2016 1:54:53 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: No server chosen by ReadPreferenceServerSelector{readPreference=primary} from cluster descriptio
out 28, 2016 1:54:53 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:1, serverValue:44}] to 127.0.0.1:27017
out 28, 2016 1:54:53 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=127.0
out 28, 2016 1:54:53 PM com.mongodb.diagnostics.logging.JULLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:45}] to 127.0.0.1:27017
```

Operações Básicas - Java

```
Banco de dados:
 - wiidas
 - local
 - test
 - aeroespacial
 - mapasbd
Coleções
 -- exemplo
 -- map_ex
 -- restaurantes
 -- system.indexes
```

Operações Básicas - Shell

```
> db.exemplo.find().pretty();
{
        "_id" : ObjectId("57dbe94f0507a4d8710ac5b2"),
        "name" : "Caio",
        "age" : 23
}
{
        "_id" : ObjectId("57dbe9750507a4d8710ac5b3"),
        "name" : "Ezequiel",
        "age" : "Melhor Nao Comentar"
}
{
        "_id" : ObjectId("57dbee630507a4d8710ac5b5"),
        "name" : "Camila",
        "age" : 24
}
> db.exemplo.find({"age": {$type: 1}});
{ "_id" : ObjectId("57dbe94f0507a4d8710ac5b2"), "name" : "Caio", "age" : 23 }
{ "_id" : ObjectId("57dbee630507a4d8710ac5b5"), "name" : "Camila", "age" : 24 }
```

Operações Básicas - Java

```
System.out.println("\nConsulta find(): \n");
FindIterable<org.bson.Document> documentos = db.getCollection("exemplo").find();
for ( org.bson.Document document : documentos ) {
    System.out.println(document);
}

System.out.println("\nConsulta find $type: \n");
FindIterable<org.bson.Document> iterable = db.getCollection("exemplo").find( new org.bson.Document("age",
        new org.bson.Document("$type", 1) ));
for( org.bson.Document d : iterable ){
    System.out.println(d.toString());
}
```

Operações Básicas - Java

```
Consulta find():

Document{{_id=57dbe94f0507a4d8710ac5b2, name=Caio, age=23.0}}
Document{{_id=57dbe9750507a4d8710ac5b3, name=Ezequiel, age=Melhor Nao Comentar}}
Document{{_id=57dbee630507a4d8710ac5b5, name=Camila, age=24.0}}

Consulta find $type:

Document{{_id=57dbe94f0507a4d8710ac5b2, name=Caio, age=23.0}}
Document{{_id=57dbee630507a4d8710ac5b5, name=Camila, age=24.0}}
```

Operações de Agregação

- Operações que são realizadas em todos os documentos. Funções como $count, $avg, entre outras que já foram implementadas e outras que podemos implementar nós mesmos.

```
db.exemplo.aggregate([{$group: {_id: null, mediaIdade: {$sum:"$age"}}}]);
"_id" : null, "mediaIdade" : 47 }
db.exemplo.aggregate([{$group: {_id: null, mediaIdade: {$avg:"$age"}}}]);
"_id" : null, "mediaIdade" : 23.5 }
```

Operações de Agregação - Java

```
BsonDocument doc3 = new BsonDocument("$avg", new BsonString("$age"));
BsonField doc2 = new BsonField("mediaIdade", doc3);

AggregateIterable<org.bson.Document> agg =
        db.getCollection("exemplo").aggregate(Arrays.asList(Aggregates.group("_id", doc2)));
org.bson.Document result2 = agg.first();
double age2 = result2.getDouble("mediaIdade");

System.out.println(age2);

mongo.close();
```

```
Aggregate:

23.5
```

Conclusão

- Alternativa para SGBD’s tradicionais.
- Não são Objetivamente melhores ou piores. Depende da Aplicação
- Amigável para novos empreendedores e novatos na área pela facilidade de manutenção e correção do banco.
- Suporta operações simples a operações complexas e até customizáveis, no caso de operações de agregação
- Já tem sistemas funcionais para replicação de dados e uso de vários data-sets.

Dúvidas?