CS 327E Class 6

Oct 9, 2020

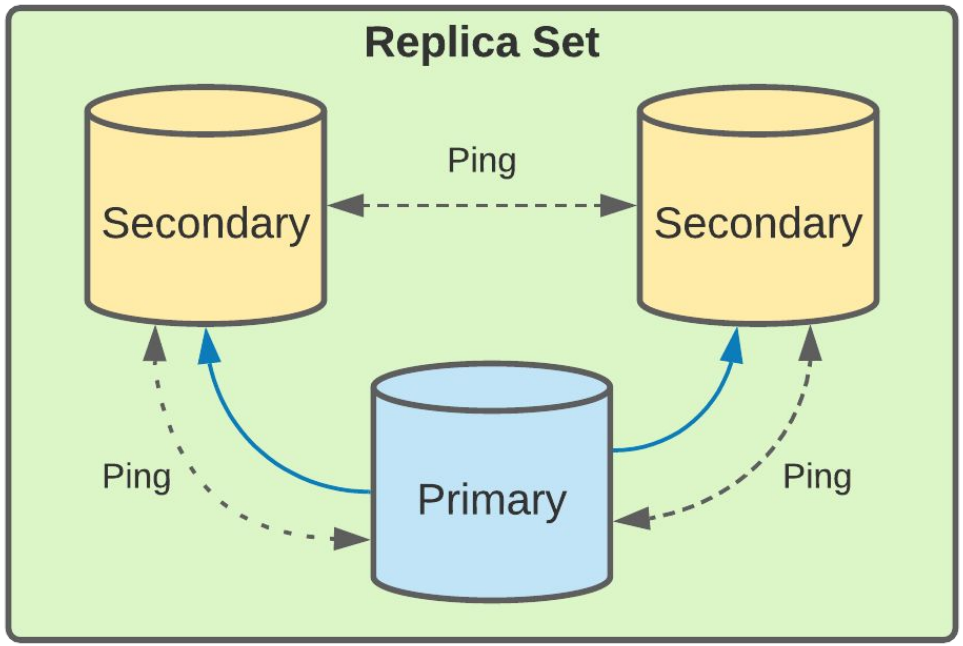
Announcements

- Review session for Test 2
- GCP coupon requests

Why MongoDB?

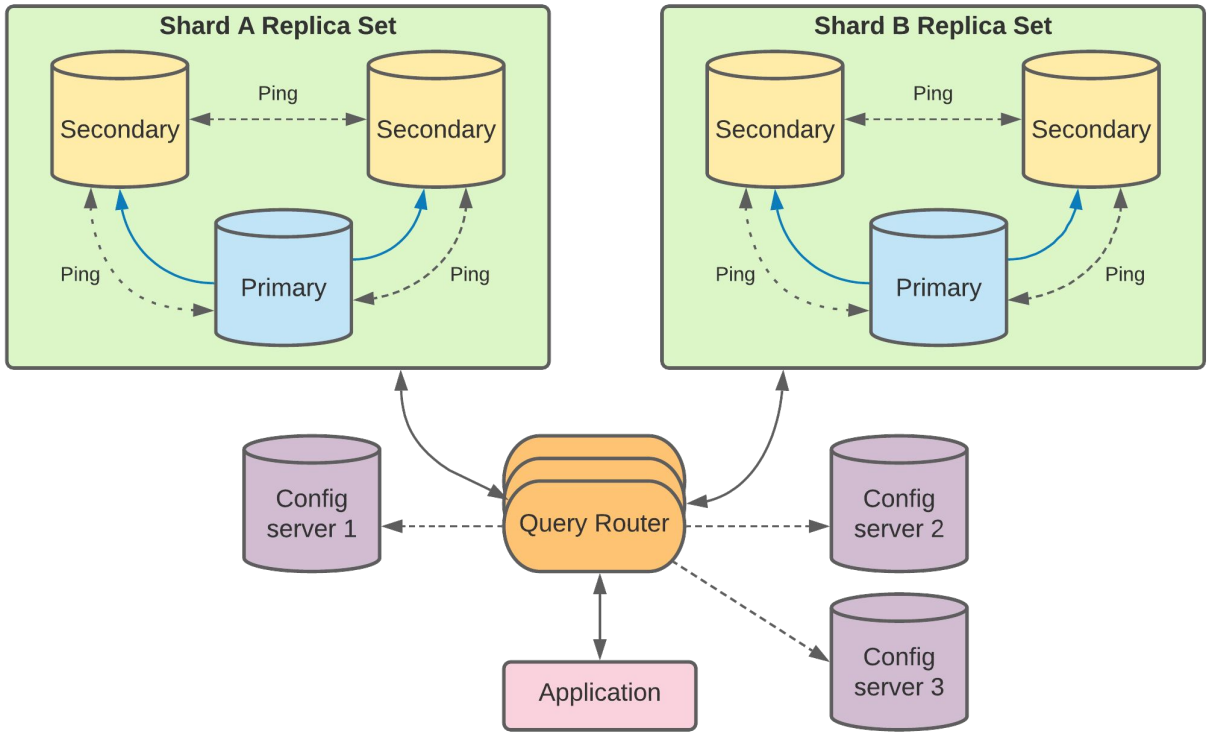
- Open-source, sponsored by MongoDB Inc.
- Designed for storing and processing web data
- Simple and flexible
- Document-oriented data model
- “Schemaless” (schema-on-reads)
- Rich query language
- Distributed database system
- Horizontal scalability through replication and sharding
- Runs on-premises and multiple cloud platforms
- Well-suited as primary datastore for web applications

Replication



- Redundancy
- Failovers
- Balance reads
- Scaling eventually consistent reads
- Widely used in production

Replication + Sharding



- `Shard key`: one or more fields of a document
- Documents split into `chunks` based on shard key
- Chunks are assigned to `shards`

Key Range	Chunk	Shard
0...20	1	A
21...40	2	B
41...60	3	A
61...80	4	B
81...100	5	A

Data Model

- MongoDB Document == BSON object
- Unordered key/value pairs with nesting
- Documents have unique identifiers (_id)

- Data types: String, Int, Double, Boolean,
- Date, Timestamp, Array, Object, ObjectId

- Max document size: 16 MB

- Documents grouped into collections
- Collections grouped into databases

```
{
    "_id" : ObjectId("5f807ab092ea454d1100d13a"),
    "name" : {
        "first" : "Jim",
        "last" : "Gray"
    },
    "nationality" : "American",
    "born" : Date("1944-01-12"),
    "employers" : [
        "Microsoft",
        "DEC",
        "Tandem",
        "IBM"
    ],
    "contributions" : [
        "database transactions",
        "OLAP cube"
    ]
}
```

Writing to Mongo

```
db.foo.insertOne(document)
db.foo.insert([document1, document2, documentn])
```

```
> doc = {"company name": "Google Inc.", "exchange": "NASDAQ", "symbol": "GOOG"}
{ "company name" : "Google Inc.", "exchange" : "NASDAQ", "symbol" : "GOOG" }
>
> db.market.insertOne(doc)
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5f7e2215801f0b72e50f3fd8")
}
> _
```

Writing to Mongo

```
> doc = {"company name": "Google Inc.", "exchange": "NASDAQ", "symbol": "GOOG", "summary": {"date": 20201007, "open": 1464.29
, "high": 1468.96, "low": 1461.47}}
{
        "company name" : "Google Inc.",
        "exchange" : "NASDAQ",
        "symbol" : "GOOG",
        "summary" : {
                "date" : 20201007,
                "open" : 1464.29,
                "high" : 1468.96,
                "low" : 1461.47
        }
}
>
> db.market.insertOne(doc)
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5f7e3fd38cacf89fdc68b264")
}
```

Writing to Mongo

```
> doc = {"company name": "Google Inc.", "symbol": "GOOG", "exchange": "NASDAQ", "summary": [{"date": 20201007, "open": 1464.2
9, "high": 1468.96, "low":  1461.47}, {"date": 20201006, "open": 1476.89, "high": 1480.93, "low": 1453.44}]}
{
        "company name" : "Google Inc.",
        "symbol" : "GOOG",
        "exchange" : "NASDAQ",
        "summary" : [
                {
                        "date" : 20201007,
                        "open" : 1464.29,
                        "high" : 1468.96,
                        "low" : 1461.47
                },
                {
                        "date" : 20201006,
                        "open" : 1476.89,
                        "high" : 1480.93,
                        "low" : 1453.44
                }
        ]
}
> db.market.insertOne(doc)
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5f7f368dbebbf1322c4b1ae4")
}
```

Reading from Mongo

```
db.foo.findOne(query_expression, projection)
db.foo.find(query_expression, projection)
```

```
> db.market.find({"company name": "Google Inc.", "symbol": "GOOG"}, {"summary": 1}).sort({"_id": 1}).limit(3).pretty()
{ "_id" : ObjectId("5f7f58fddd0243ebf5582a24") }
{
        "_id" : ObjectId("5f7f5906dd0243ebf5582a25"),
        "summary" : {
                "date" : 20201007,
                "open" : 1464.29,
                "high" : 1468.96,
                "low" : 1461.47
        }
}
{
        "_id" : ObjectId("5f7f5911dd0243ebf5582a26"),
        "summary" : [
                {
                        "date" : 20201007,
                        "open" : 1464.29,
                        "high" : 1468.96,
                        "low" : 1461.47
                },
                {
                        "date" : 20201006,
                        "open" : 1476.89,
                        "high" : 1480.93,
                        "low" : 1453.44
                }
        ]
}
```

Reading from Mongo

```
> db.market.find({"summary.date": 20201007}, {"summary": 1}).pretty()
{
        "_id" : ObjectId("5f7f3677bebbf1322c4b1ae3"),
        "summary" : {
                "date" : 20201007,
                "open" : 1464.29,
                "high" : 1468.96,
                "low" : 1461.47
        }
}
{
        "_id" : ObjectId("5f7f368dbebbf1322c4b1ae4"),
        "summary" : [
                {
                        "date" : 20201007,
                        "open" : 1464.29,
                        "high" : 1468.96,
                        "low" : 1461.47
                },
                {
                        "date" : 20201006,
                        "open" : 1476.89,
                        "high" : 1480.93,
                        "low" : 1453.44
                }
        ]
}
```

Reading from Mongo

```
> db.market.find({"summary.date": 20201007, "summary.date": 20201006}, {"summary": 1}).pretty()
{
        "_id" : ObjectId("5f7f8a1ad400cb46a62c861a"),
        "summary" : [
                {
                        "date" : 20201007,
                        "open" : 1464.29,
                        "high" : 1468.96,
                        "low" : 1461.47
                },
                {
                        "date" : 20201006,
                        "open" : 1476.89,
                        "high" : 1480.93,
                        "low" : 1453.44
                }
        ]
}
```

Reading from Mongo

```
> db.market.find({"$or": [{"summary.date": 20201007}, {"summary.date": 20201006}]}, {"summary.date": 1}).pretty()
{
        "_id" : ObjectId("5f7f8a0fd400cb46a62c8619"),
        "summary" : {
                "date" : 20201007
        }
}
{
        "_id" : ObjectId("5f7f8a1ad400cb46a62c861a"),
        "summary" : [
                {
                        "date" : 20201007
                },
                {
                        "date" : 20201006
                }
        ]
}
```

Boolean Operators:

$ne
$not
$or
$nor
$and
$exists

Reading from Mongo

```
> db.market.find({"summary.low": {"$gte": 1450, "$lte": 1455}}, {"summary":1}).pretty()
{
        "_id" : ObjectId("5f7f368dbebbf1322c4b1ae4"),
        "summary" : [
                {
                        "date" : 20201007,
                        "open" : 1464.29,
                        "high" : 1468.96,
                        "low" : 1461.47
                },
                {
                        "date" : 20201006,
                        "open" : 1476.89,
                        "high" : 1480.93,
                        "low" : 1453.44
                }
        ]
}
```

Ranges:
$lt
$gt
$lte
$gte

Updates in Mongo

```
db.foo.update(query_expression, update)
db.foo.updateMany(query_expression, update)
```

```
> doc = {"company name": "Alphabet, Inc."}
{ "company name" : "Alphabet, Inc." }
>
> db.market.updateMany({}, {"$set": doc})
{ "acknowledged" : true, "matchedCount" : 3, "modifiedCount" : 3 }
>
```

```
> doc = {"summary": {"date": 20201008, "open": 1465.09, "high": 1485.45, "low": 1465.09}}
{
        "summary" : {
                "date" : 20201008,
                "open" : 1465.09,
                "high" : 1485.45,
                "low" : 1465.09
        }
}
> db.market.update({"_id" : ObjectId("5f7f8a1ad400cb46a62c861a")}, {"$addToSet": doc})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

Deletes in Mongo

```
db.foo.remove(query_expression)
```

```
> doc = {"_id": ObjectId("5f7f8a1ad400cb46a62c861a")}
{ "_id" : ObjectId("5f7f8a1ad400cb46a62c861a") }
>
> db.market.remove(doc)
WriteResult({ "nRemoved" : 1 })
>
```

```
> doc = {"company name": "Alphabet, Inc."}
{ "company name" : "Alphabet, Inc." }
>
> db.market.remove(doc)
WriteResult({ "nRemoved" : 2 })
>
```

Set up Mongo

https://github.com/cs327e-fall2020/snippets/wiki/MongoDB-Setup-Guide

Practice Problem

Translate the following SQL query to MongoDB's query language:

```
SELECT Title, Artist, Date, 'Height (cm)', 'Width (cm)'
FROM Artworks
WHERE Nationality = 'Swedish'
AND Classification = 'Sculpture'
ORDER BY 'Height (cm)' DESC, 'Width (cm)' DESC
LIMIT 1;
```

Project 5

http://www.cs.utexas.edu/~scohen/projects/Project5.pdf