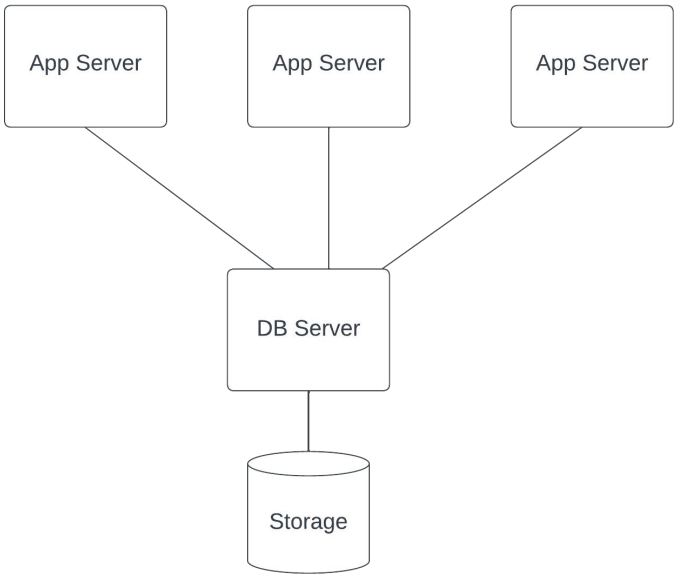
Class 8 MongoDB

Elements of Databases

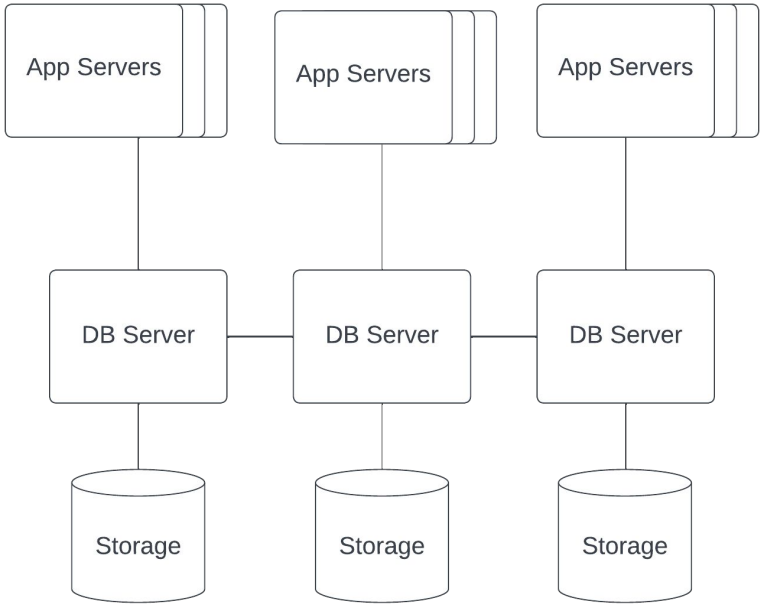
Oct 20, 2023

Database Architectures

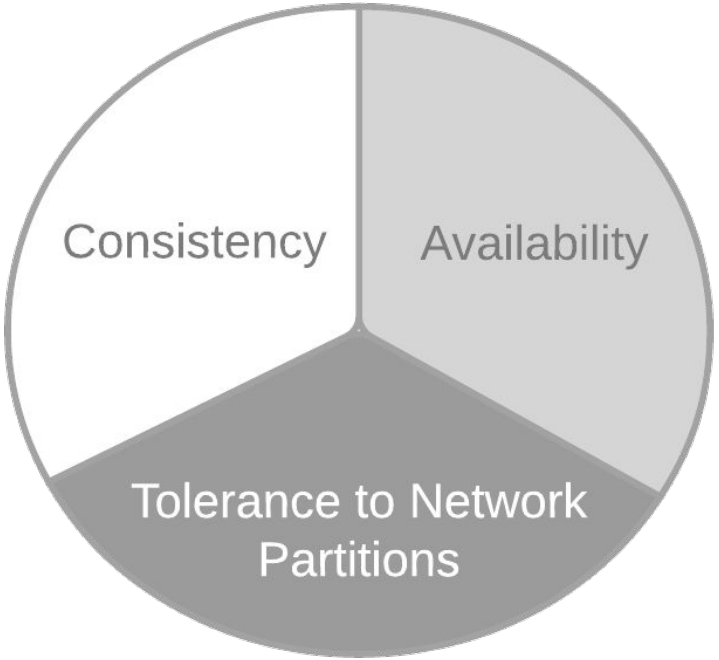
Centralized Architecture

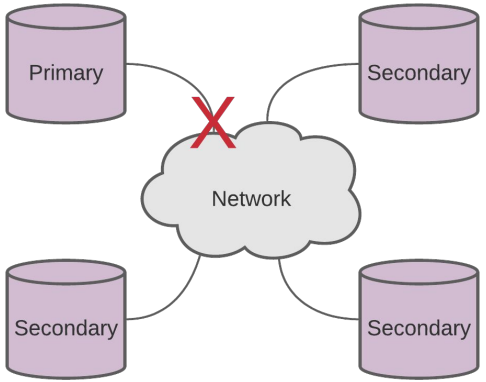


Distributed Architecture



The CAP Theorem





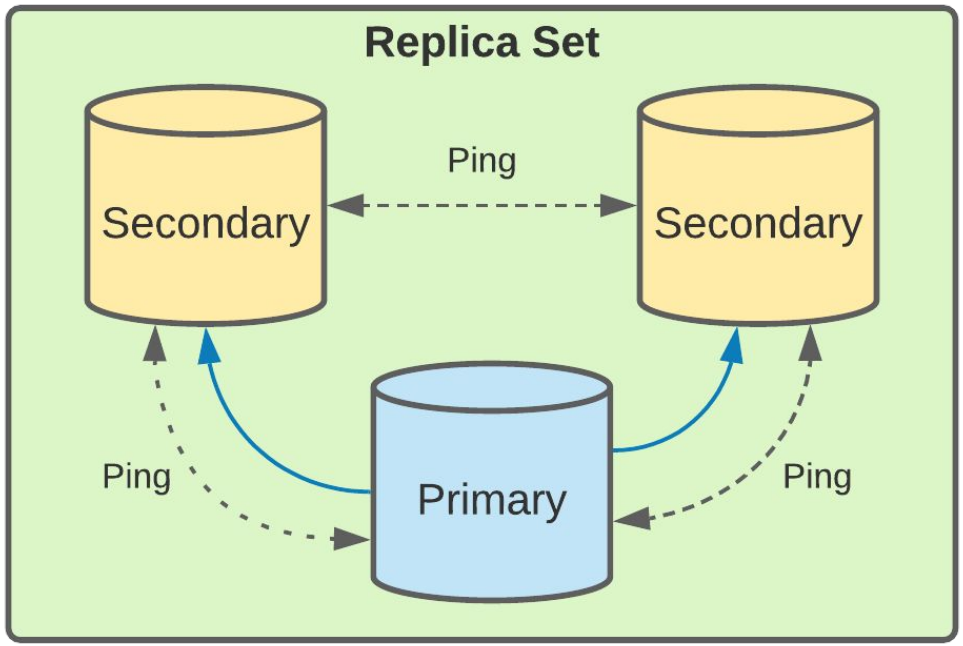
Theorem: You can have **at most two** of these properties for any database system.

Eric Brewer, PODC keynote, July 2000.

MongoDB Overview

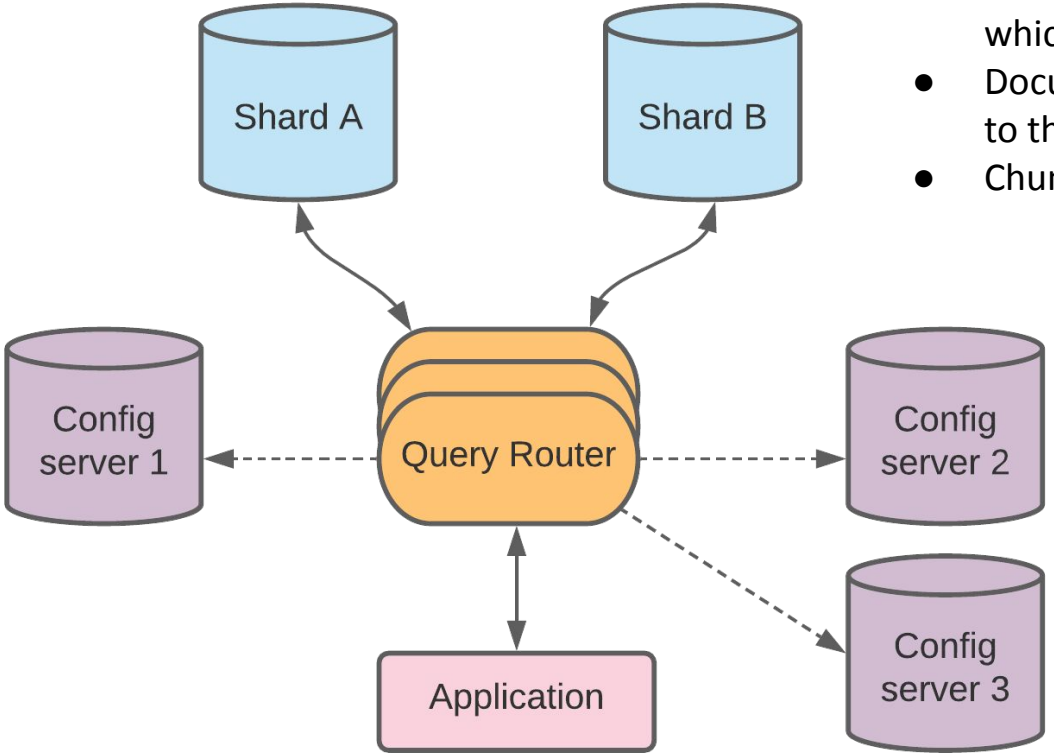
+ Distributed database system
+ Open-source software (sponsored by MongoDB Inc.)
+ Designed for storing and processing web data
+ Document-oriented data model
+ "Schemaless" (schema-on-reads)
+ Rich query language
+ Secondary indexes
+ Horizontal scaling through replication and sharding
+ Runs on-premise and in cloud (Atlas offering)
+ Primary datastore for many web applications
+ Multi-document transaction support
- Sharding is not automatic

Replication in MongoDB



- High-availability
- Redundancy
- Automatic failovers
- Load balancing reads

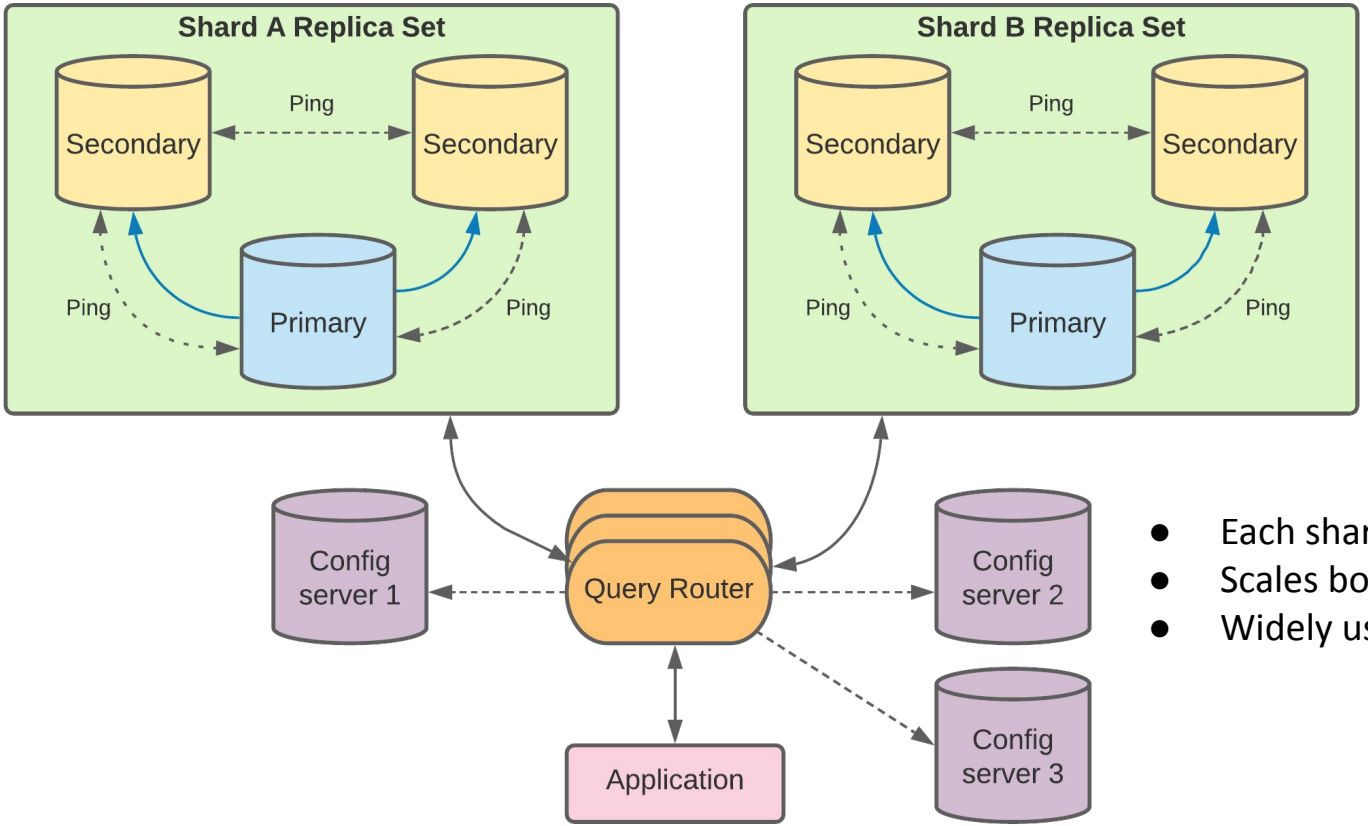
Sharding in MongoDB



- `shard key` = one or more fields of a document which determine how documents get sliced
- Documents with the same shard key are assigned to the same `chunk`
- Chunks are assigned to a `shard`

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

Sharding + Replication



- Each shard is deployed as a replica set
- Scales both reads and writes
- Widely used in prod environments

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
- Documents are nested via Object type
- Max document size: 16 MB (including nested objects)

- 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"
    ]
}
```

Reads

```
db.coll.findOne(selection, projection)
db.coll.find(selection, projection)
```

Legend:
Selection, Projection, Output

```
> selection = {"company name": "Google Inc.", "symbol": "GOOG"}
{ "company name" : "Google Inc.", "symbol" : "GOOG" }
> projection = {"company name": 1, exchange: 1, symbol: 1, _id:0}
{ "company name" : 1, "exchange" : 1, "symbol" : 1, "_id" : 0 }
>
> db.market.find(selection, projection).pretty()
{ "company name" : "Google Inc.", "exchange" : "NASDAQ", "symbol" : "GOOG" }
{ "company name" : "Google Inc.", "exchange" : "NASDAQ", "symbol" : "GOOG" }
{ "company name" : "Google Inc.", "symbol" : "GOOG", "exchange" : "NASDAQ" }
>
```

Nested Queries

Legend:
Selection, Projection, Output

```
> selection = {"summary.date": 20211022}
{ "summary.date" : 20211022 }
> projection = {"summary.date": 1, "summary.open": 1, "summary.high": 1, _id:0}
{ "summary.date" : 1, "summary.open" : 1, "summary.high" : 1, "_id" : 0 }
>
> db.market.find(selection, projection)
{ "summary" : { "date" : 20211022, "open" : 2807.02, "high" : 2831.17 } }
```

```
> selection = {"summary.date": 20211022, "symbol": "GOOG"}
{ "summary.date" : 20211022, "symbol" : "GOOG" }
> projection = {"summary.date": 1, "summary.open": 1, "summary.high": 1, _id:0}
{ "summary.date" : 1, "summary.open" : 1, "summary.high" : 1, "_id" : 0 }
>
> db.market.find(selection, projection).pretty()
{
        "summary" : {
                "date" : 20211022,
                "open" : 2807.02,
                "high" : 2831.17
        }
}
```

Or Queries

Legend:
Selection, Projection, Output

```
> selection = {"$or": [{"summary.date": 20211022}, {"summary.date": 20201007}]}
{
        "$or" : [
                {
                        "summary.date" : 20211022
                },
                {
                        "summary.date" : 20201007
                }
        ]
}
> projection = {"summary.date": 1, "summary.open": 1, "summary.high": 1, _id:0}
{ "summary.date" : 1, "summary.open" : 1, "summary.high" : 1, "_id" : 0 }
>
> db.market.find(selection, projection)
{ "summary" : { "date" : 20211022, "open" : 2807.02, "high" : 2831.17 } }
{ "summary" : [ { "date" : 20201007, "open" : 1464.29, "high" : 1468.96 }, { "date" : 20201006, "open" : 1476.89, "high" : 1480.93 } ] }
```

Boolean Operators:
$or
$and

Range Queries

```
> selection = {"summary.low": {"$gte": 1450, "$lte": 1455}}
{ "summary.low" : { "$gte" : 1450, "$lte" : 1455 } }
>
```

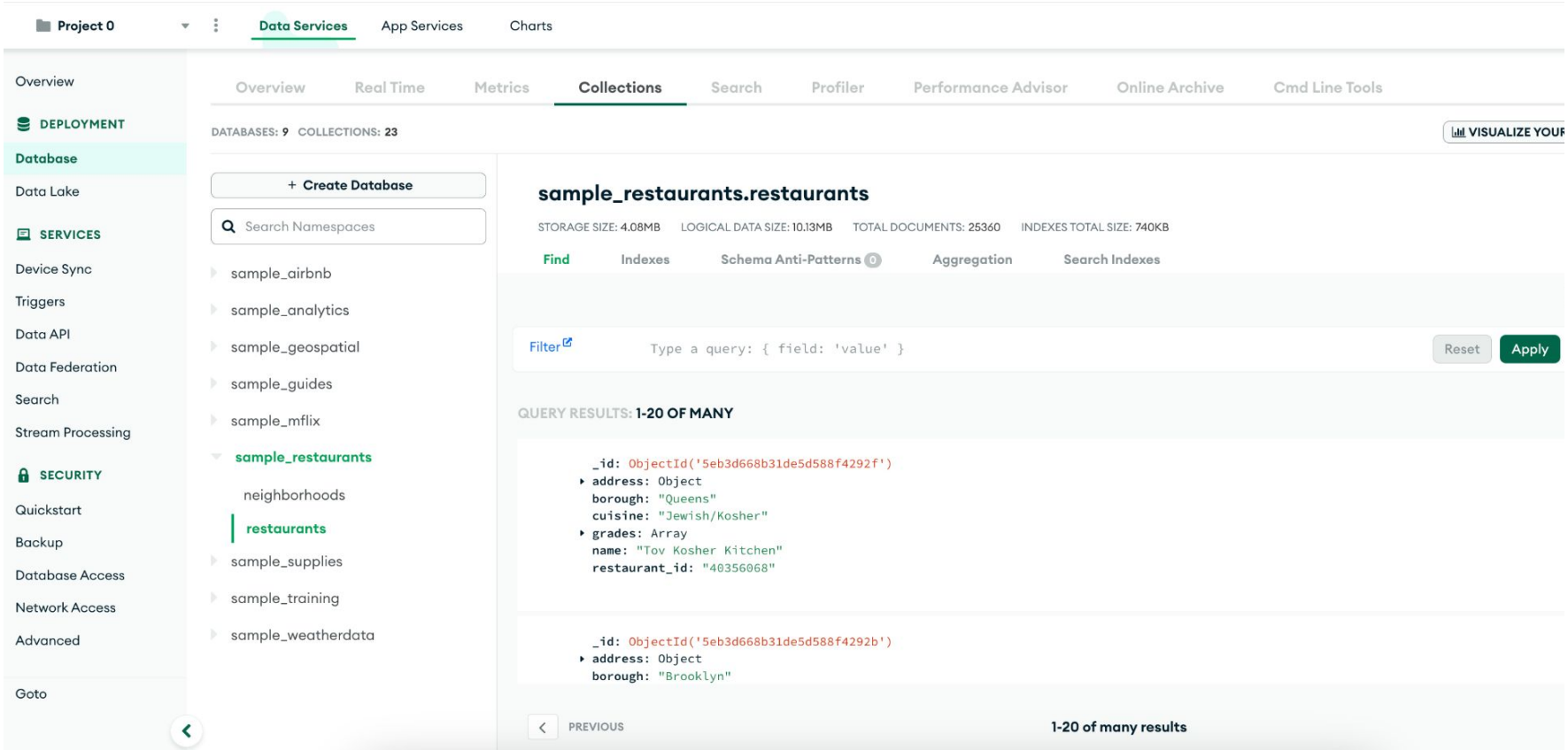
```
> projection = {"summary.date": 1, "summary.low": 1, _id:0}
{ "summary.date" : 1, "summary.low" : 1, "_id" : 0 }
>
```

```
> db.market.find(selection, projection).pretty()
{
        "summary" : [
                {
                        "date" : 20201007,
                        "low" : 1461.47
                },
                {
                        "date" : 20201006,
                        "low" : 1453.44
                }
        ]
}
>
```

Legend:
Selection, Projection, Output

Range operators:
$lt
$gt
$lte
$gte

Switching to MongoDB Atlas



MongoDB code lab

- Clone snippets repo
- Open mongodb notebook and js script
- Run through the queries
- Answer the two prompts

Inserts

Legend:
Document, Output

```
db.coll.insertOne(document)
db.coll.insert([document1, document2, documentn])
db.coll.insertMany([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("6247434f7dbfdcf5f6767219")
}
>
```

Inserts

Legend:
Document, Output

```
> doc2 = {"company name": "Google Inc.", "exchange": "NASDAQ", "symbol": "GOOG", "summary": {"date": 20211022, "open":
 2807.02, "high": 2831.17, "low": 2743.41}}
{
        "company name" : "Google Inc.",
        "exchange" : "NASDAQ",
        "symbol" : "GOOG",
        "summary" : {
                "date" : 20211022,
                "open" : 2807.02,
                "high" : 2831.17,
                "low" : 2743.41
        }
}
> doc3 = {"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}]}
{
        "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.insertMany([doc2, doc3])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("624744a17dbfdcf5f676721c"),
                ObjectId("624744a17dbfdcf5f676721d")
        ]
}
>
```

Simple Updates

Legend:
Selection, Update

```
db.coll.update(selection, update)
db.coll.updateMany(selection, update)
```

```
> field = {"address": "2000 Charleston Road"}
{ "address" : "2000 Charleston Road" }
>
```

```
> db.market.update({}, {"$addToSet": field})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

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

Deletes

Legend:
Selection, Output

```
db.coll.deleteOne(selection)
db.coll.deleteMany(selection)
```

```
> selection = {"_id": ObjectId("624747197dbfdcf5f6767222")}
{ "_id" : ObjectId("624747197dbfdcf5f6767222") }
>
```

```
> db.market.deleteOne(selection)
{ "acknowledged" : true, "deletedCount" : 1 }
>
```

```
> selection = {"exchange": "NASDAQ"}
{ "exchange" : "NASDAQ" }
>
```

```
> db.market.deleteMany(selection)
{ "acknowledged" : true, "deletedCount" : 2 }
>
```

Project 6

http://www.cs.utexas.edu/~scohen/projects/project-6.pdf