

NoSQL Database - MongoDB

What is Mongo?

- MongoDB is an open-source **document** database.
- Provides
 - high performance
 - high availability (not 99.999% availability)
 - automatic scaling
 - non-relational
 - can be distributed / partitioned easily

Mongo

- Based on documents *think of it as a record*
- A record in MongoDB is a document, which is a data structure composed of field and value pairs
- Documents are stored in collections *think of it as tables*
- Collections do not require schema specification – dynamic schema.

Mongo – K, V pairs

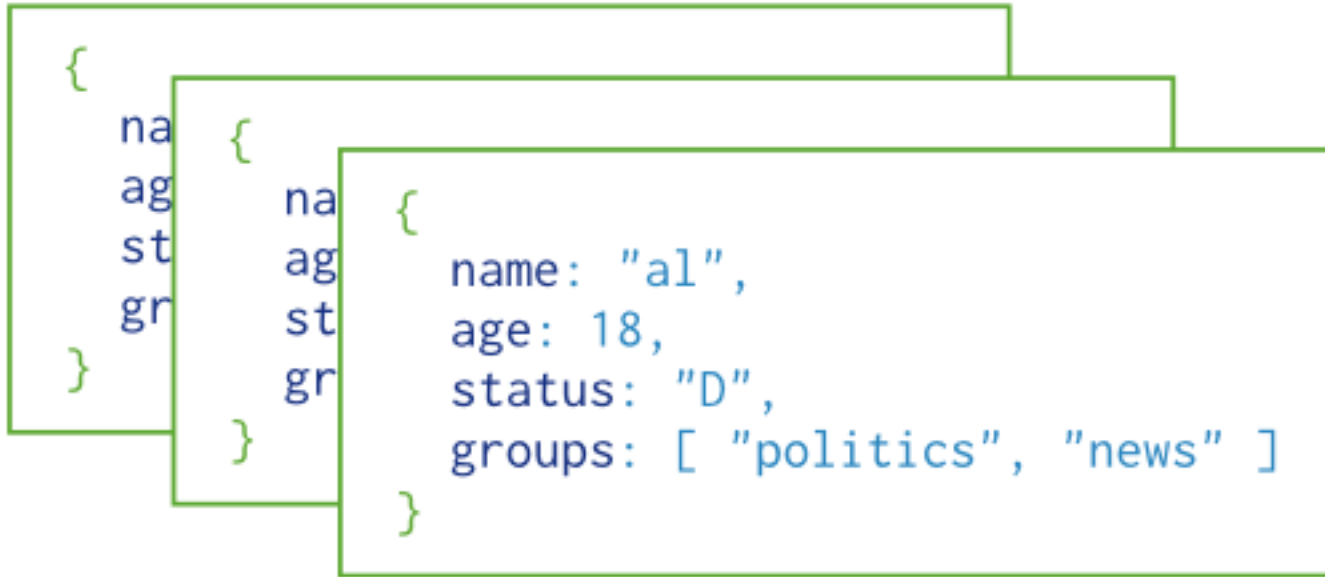
- MongoDB stores data in the form of documents, which are JSON-like field and value pairs
- Other structures that store K-V pairs: e.g. dictionaries, hashes, maps, and associative arrays

```
{  
  name: "sue",  
  age: 26,  
  status: "A",  
  groups: [ "news", "sports" ]  
}
```



← field: value
← field: value
← field: value
← field: value

Collections



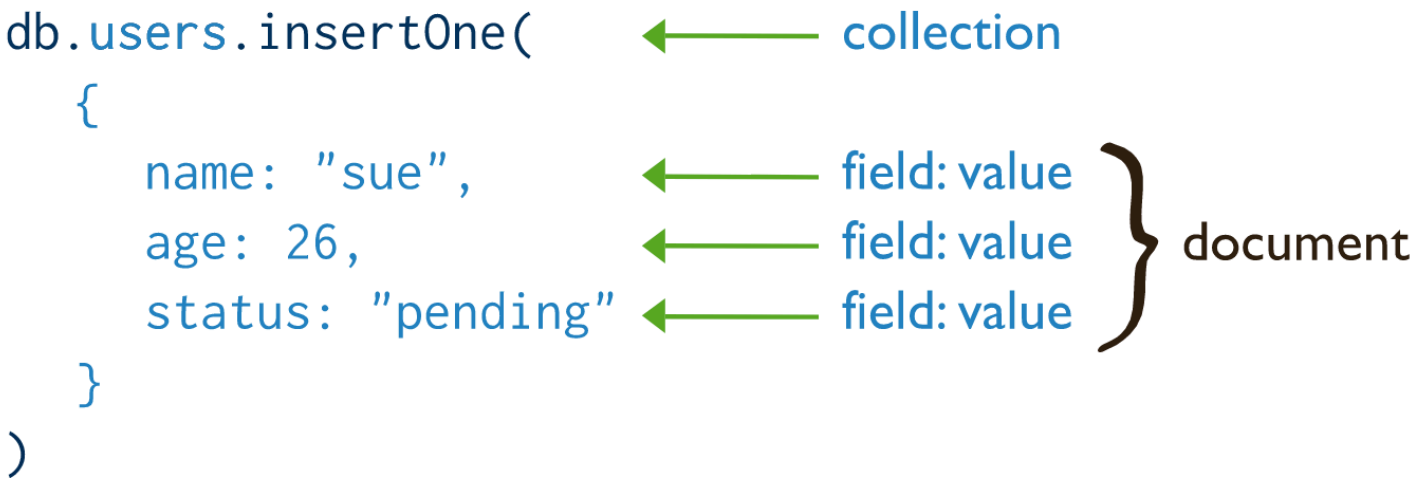
Collection

CRUD Operations

- Inserting data:

- One at a time:

```
db.users.insertOne(
  {
    name: "sue",
    age: 26,
    status: "pending"
  }
)
```



- Bulk Import: **mongoimport** command can work with JSON, CSV, and other formats

CRUD Operations

- Getting data:
- `db.collections.find({filter conditions})`

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

← collection
← query criteria
← projection
← cursor modifier

Aggregate Operations

- MongoDB's aggregation framework is modeled on the concept of **data processing pipelines**. Documents enter a **multi-stage pipeline** that transforms the documents into an aggregated result.
- Operations occur in pipelines one after the other.

Aggregate Operations

Collection

```
db.orders.aggregate( [
  $match stage → { $match: { status: "A" } },
  $group stage → { $group: { _id: "$cust_id", total: { $sum: "$amount" } } }
] )
```

{ cust_id: "A123", amount: 500, status: "A" }
{ cust_id: "A123", amount: 250, status: "A" }
{ cust_id: "B212", amount: 200, status: "A" }
{ cust_id: "A123", amount: 300, status: "D" }

orders

\$match →

{ cust_id: "A123", amount: 500, status: "A" }
{ cust_id: "A123", amount: 250, status: "A" }
{ cust_id: "B212", amount: 200, status: "A" }

\$group →

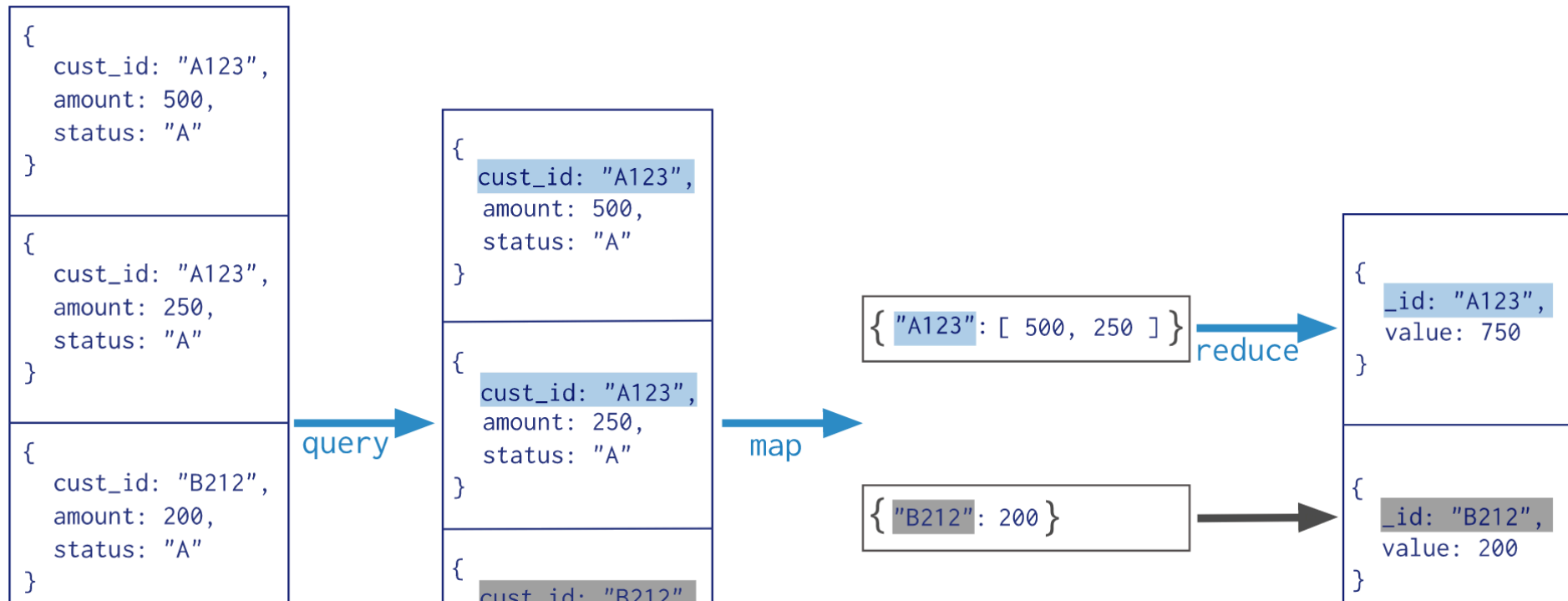
Results

```
{  
  _id: "A123",  
  total: 750  
}
```

```
{  
  _id: "B212",  
  total: 200  
}
```

MapReduce Operations

Collection
↓
`db.orders.mapReduce(`
 `map` `function() { emit(this.cust_id, this.amount); },`
 `reduce` `function(key, values) { return Array.sum(values) },`
 `{`
 `query: { status: "A" },`
 `out: "order_totals"`
 `}`
 `)`



NoSQL vs SQL databases

<https://www.mongodb.com/nosql-explained>

Summary of commands

<http://docs.mongodb.org/master/reference/mongo-shell/>

Mongo Commands

- show databases
- use test
- show tables
- `db.restaurants.count()`

More commands here:

<http://docs.mongodb.org/getting-started/shell/query/>