NOSQL DDBBs: Introduction to MongoDB

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Campus d'Excel·lència Internacional

Índex

KEY-VALUE ENHANCEMENTS

DOCUMENT-ORIENTED DBS

Structuring the Value

- Essentially, they are key-value stores
  - Same design and architectural features
- The value is a document
  - XML (e.g., eXist)
  - JSON (e.g., MongoDB and CouchDB)
- Tightly related to the Web
  - Normally, they provide RESTful HTTP APIs
- So… what is the benefit of having documents?
  - New data model (collections and documents)
    - **New atom: from rows to documents**
  - Indexing

Designing Document Stores

- Follow one basic rule: 1 fetch for the whole data set at hand
  - Aggregate data model: check the data needed by your application simultaneously
    - **Do not think relational-wise!**
  - Use indexes to identify finer data granularities
- Consequences:
  - Independent documents
    - Avoid pointing FKs (i.e., pointing at other docs)
  - Massive denormalization
  - A change in the application layout might be dramatic
    - It may entail a massive rearrangement of the database documents

Types of Document-Stores

- JSON-like documents
  - MongoDB
  - CouchDB
- JSON is a lightweight data interchange format
  - Brackets ([ ]) represent ordered lists
  - Curly braces ({ }) represent key-value dictionaries
    - Keys must be strings, delimited by quotes (")
    - Values can be strings, numbers, booleans, lists, or key-value dictionaries
- Natively compatible with JavaScript
  - Web browsers are natural clients for MongoDB / CouchDB

http://www.json.org/index.html

JSON Example

- Definition:

A document is an object represented with an unbounded nesting of array and object constructs

```
{
   "title": "The Social network",
   "year": "2010",
   "genre": "drama",
   "summary": "On a fall night in 2003, Harvard undergrad and computer
   programming genius Mark Zuckerberg sits down at his computer
   and heatedly begins working on a new idea. In a fury of blogging
   and programming, what begins in his dorm room soon becomes a global
   social network and a revolution in communication. A mere six years
   and 500 million friends later, Mark Zuckerberg is the youngest
   billionaire in history... but for this entrepreneur, success leads
   to both personal and legal complications.",
   "country": "USA",
 "director": {
     "last_name": "Fincher",
     "first_name": "David",
     "birth_date": "1962"
 },
 "actors": [
   {
       "first_name": "Jesse",
       "last_name": "Eisenberg",
       "birth_date": "1983",
       "role": "Mark Zuckerberg"
   },
   {
       "first_name": "Rooney",
       "last_name": "Mara",
       "birth_date": "1985",
       "role": "Erica Albright"
   },
   {
       "first_name": "Andrew",
       "last_name": "Garfield",
       "birth_date": "1983",
       "role": "  Eduardo Saverin "
   },
   {
       "first_name": "Justin",
       "last_name": "Timberlake",
       "birth_date": "1981",
       "role": "Sean Parker"
   }
 ]
}
```

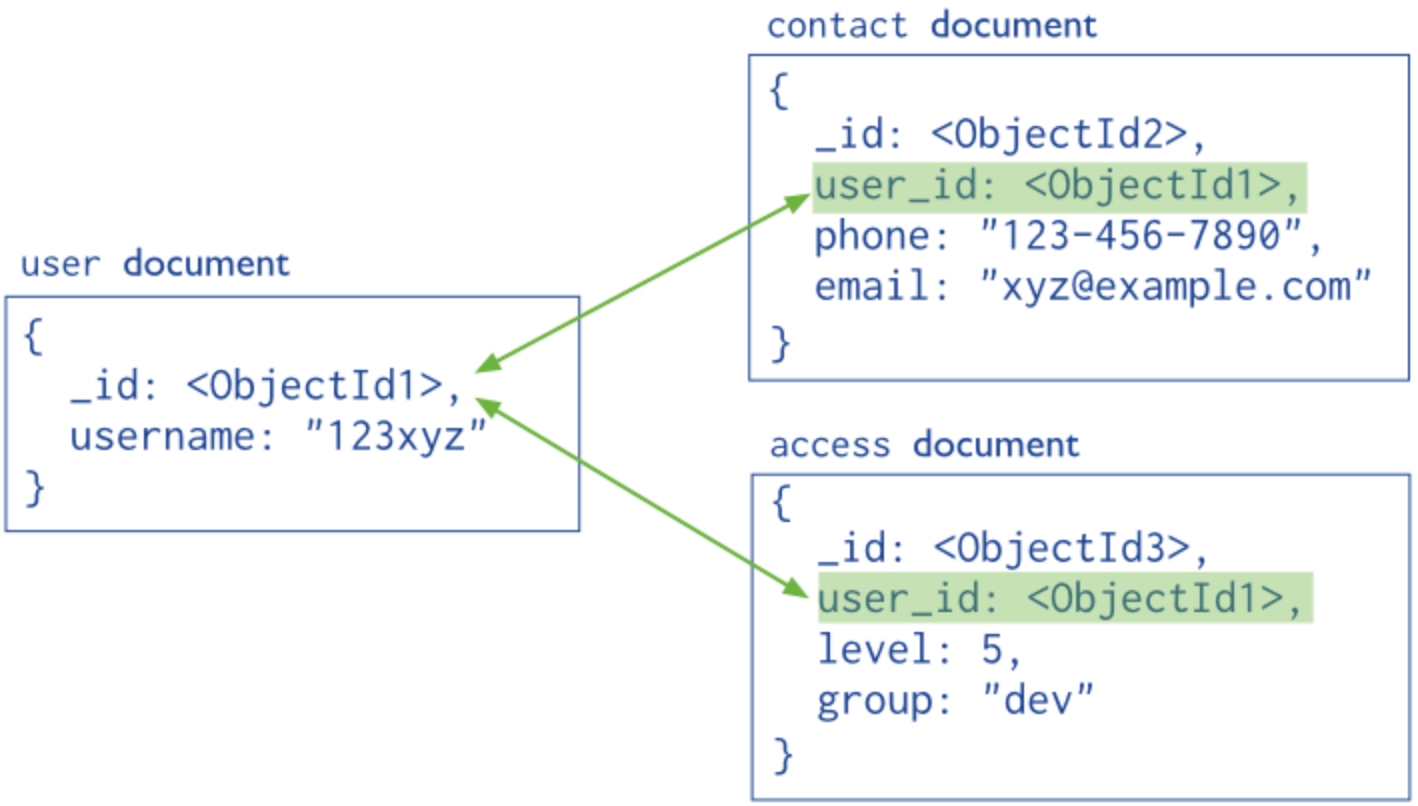
An Example of Document-Store

MONGODB

MongoDB: Data Model

- Collections
  - *Definition*: A grouping of MongoDB documents
    - A collection exists within a single database
    - Collections do not enforce a schema
  - MongoDB Namespace: *database.collection*
- Documents
  - *Definition*: JSON documents (serialized as BSON)
    - Basic atom
    - Identified by *_id* (user or system generated)
    - Aggregated view of data
    - May contain
      - References (**NOT FKs!**) and
      - Embedded documents

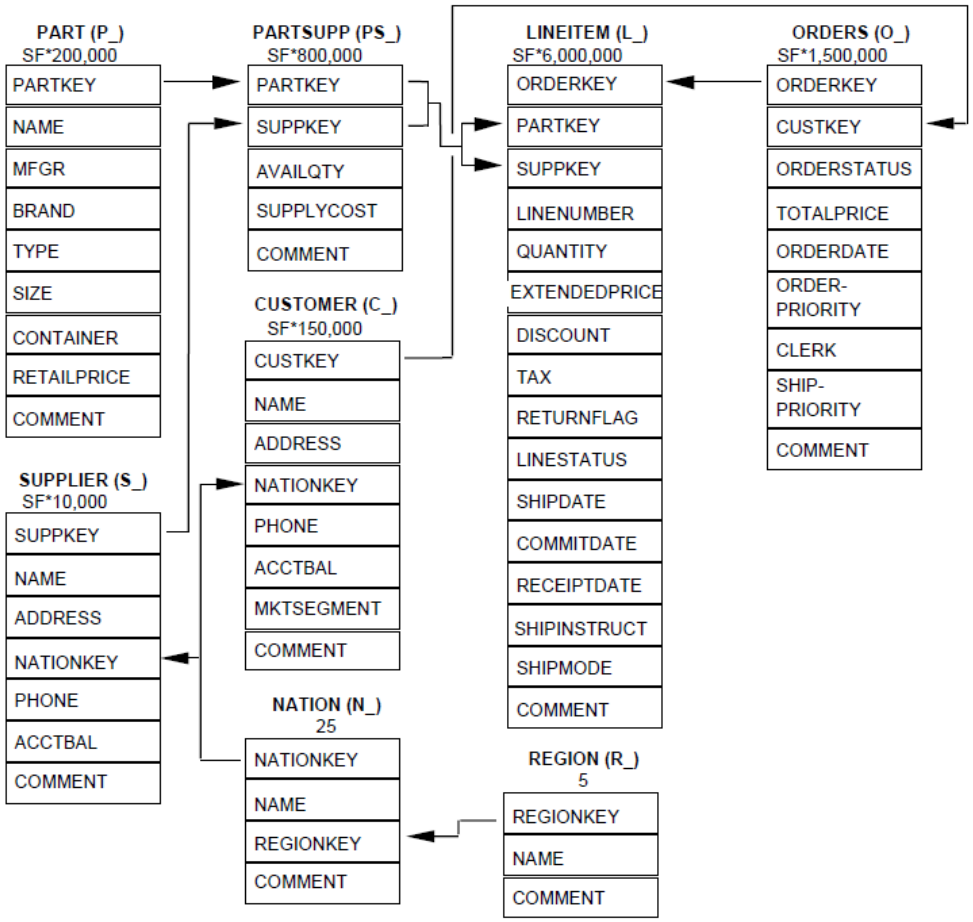
MongoDB: Document Example



MongoDB: Document Example

Activity: Modeling in MongoDB (I)

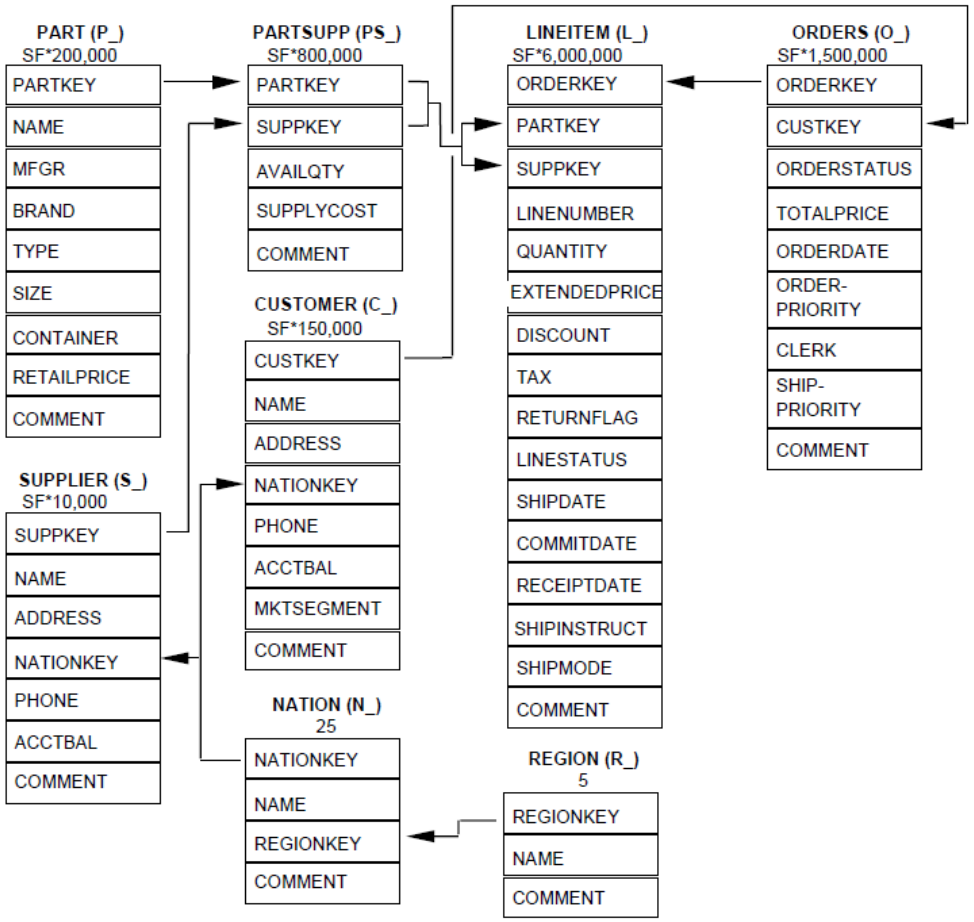
- *Objective: Learn how to model documents*
- *Tasks:*
  1. *(15') Model the TPC-H database in a normalized way*
     1. *A table is a collection*
     2. *An instance is a document*
  2. *(5') Discussion*



Activity: Modeling in MongoDB (II)

- *Objective: Learn how to model documents*
- *Tasks:*
  1. *(15') Model the TPC-H database according to the query below*
  2. *(5') Discussion*

```
SELECT l_orderkey,
sum(l_extendedprice*(1-l_discount))
as revenue, o_orderdate,
o_shippriority
FROM customer, orders, lineitem
WHERE c_mktsegment = '[SEGMENT]' AND
c_custkey = o_custkey AND l_orderkey
= o_orderkey AND o_orderdate <
'[DATE]' AND l_shipdate > '[DATE]'
GROUP BY l_orderkey, o_orderdate,
o_shippriority
ORDER BY revenue desc, o_orderdate;
```



Activity: Modeling in MongoDB (III)

- *Objective: Learn how to model documents*
- *Tasks:*
  1. *(30') Model the TPC-H database according to the following queries*
  2. *(5') Discussion*

Q1

```
SELECT l_orderkey,
sum(l_extendedprice*(1-
l_discount)) as revenue,
o_orderdate, o_shippriority

FROM customer, orders, lineitem

WHERE c_mktsegment = '[SEGMENT]'
AND c_custkey = o_custkey AND
l_orderkey = o_orderkey AND
o_orderdate < '[DATE]' AND
l_shipdate > '[DATE]'

GROUP BY l_orderkey, o_orderdate,
o_shippriority

ORDER BY revenue desc,
o_orderdate;
```

Q2

```
SELECT l_returnflag, l_linestatus,
sum(l_quantity) as sum_qty,
sum(l_extendedprice) as
sum_base_price,
sum(l_extendedprice*(1-l_discount))
as sum_disc_price,
sum(l_extendedprice*(1-
l_discount)*(1+l_tax)) as sum_charge,
avg(l_quantity) as avg_qty,
avg(l_extendedprice) as avg_price,
avg(l_discount) as avg_disc, count(*)
as count_order

FROM lineitem

WHERE l_shipdate <= '[date]'

GROUP BY l_returnflag, l_linestatus

ORDER BY l_returnflag, l_linestatus;
```

Activity: Modeling in MongoDB (III)

- *Objective: Learn how to model documents*
- *Tasks:*
  1. *(30') Model the TPC-H database according to the following queries*
  2. *(5') Discussion*

Q3

```
SELECT n_name, sum(l_extendedprice * (1 -
l_discount)) as revenue

FROM customer, orders, lineitem, supplier,
nation, region

WHERE c_custkey = o_custkey AND l_orderkey =
o_orderkey AND l_suppkey = s_suppkey AND
c_nationkey = s_nationkey AND s_nationkey =
n_nationkey AND n_regionkey = r_regionkey AND
r_name = '[REGION]' AND o_orderdate >= date
'[DATE]' AND o_orderdate < date '[DATE]' +
interval '1' year

GROUP BY n_name

ORDER BY revenue desc;
```

Q4

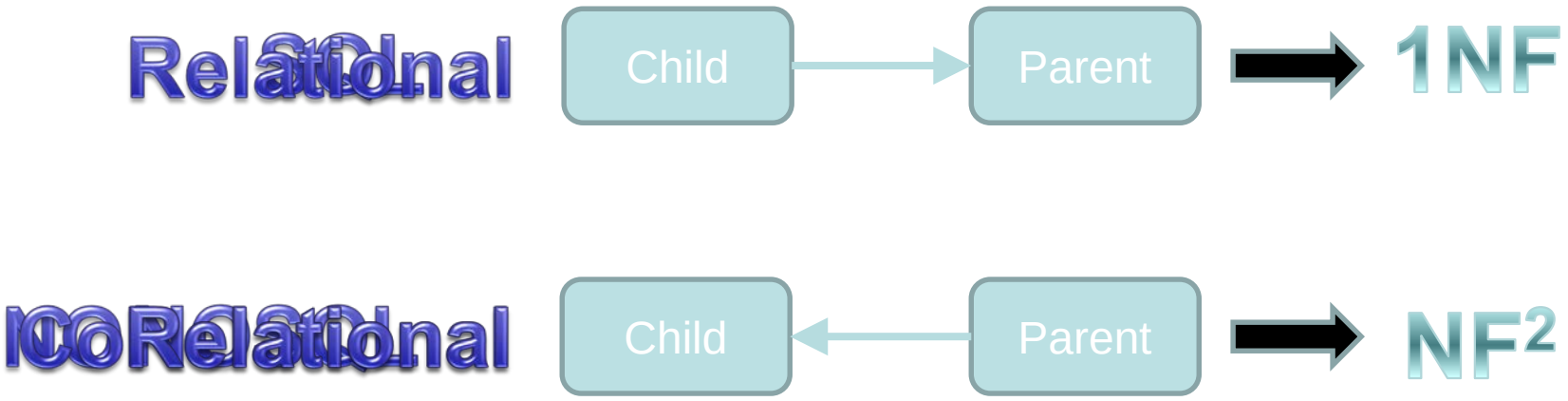
```
SELECT s_acctbal, s_name, n_name,
p_partkey, p_mfgr, s_address, s_phone,
s_comment

FROM part, supplier, partsupp, nation,
region

WHERE p_partkey = ps_partkey AND s_suppkey
= ps_suppkey AND p_size = [SIZE] AND p_type
like '%[TYPE]' AND s_nationkey =
n_nationkey AND n_regionkey = r_regionkey
AND r_name = '[REGION]' AND ps_supplycost =
(SELECT min(ps_supplycost) FROM partsupp,
supplier, nation, region WHERE p_partkey =
ps_partkey AND s_suppkey = ps_suppkey AND
s_nationkey = n_nationkey AND n_regionkey =
r_regionkey AND r_name = '[REGION]')

ORDER BY s_acctbal desc, n_name, s_name,
p_partkey;
```

Just Another Point of View

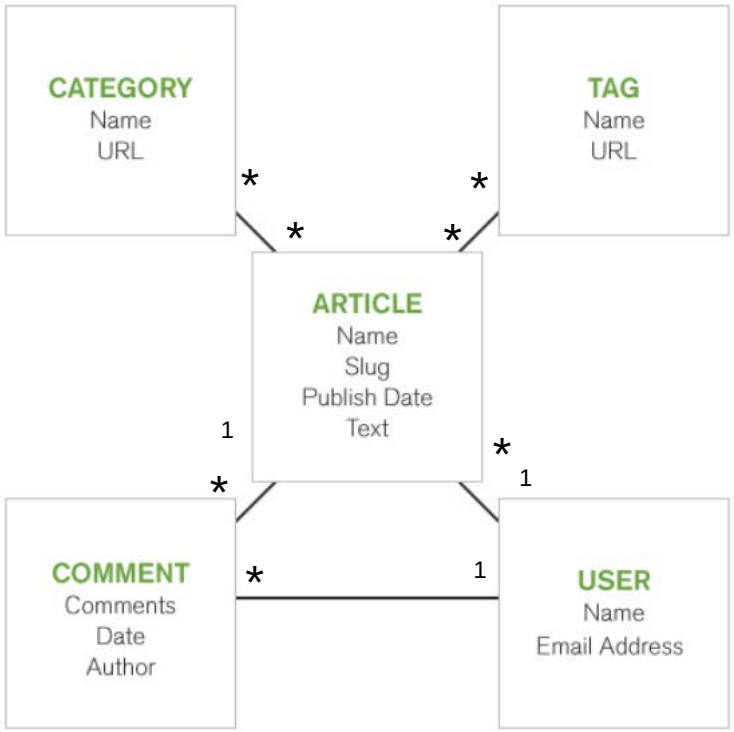


Yet Another Modeling Activity

- *Objective: Learn how to model documents*
- *Tasks:*
  1. *(10') Model the provided schema in MongoDB*
  2. *(5') Discussion*

Example:
Consider the example of a blogging platform. In this example, a relational application relies on five separate tables (implemented in MySQL) in order to build the blog entry.

The webpage always renders an article in the main frame. On the top, it shows the categories it belongs to, the data about the user who created it and the article metadata (e.g., publish date). Then, at the bottom of the page all comments are shown one after another.



MongoDB Shell

- Show dbs
- Use <*database*>
- Show collections
- Show users
- coll = db.<*collection*>
- Find(*criteria, projection*)
- Insert(*document*)
- Update(*query, update, options*)
- Save(*document*)
- Remove(*query, justOne*)
- Drop()
- EnsureIndex(*keys, options*)

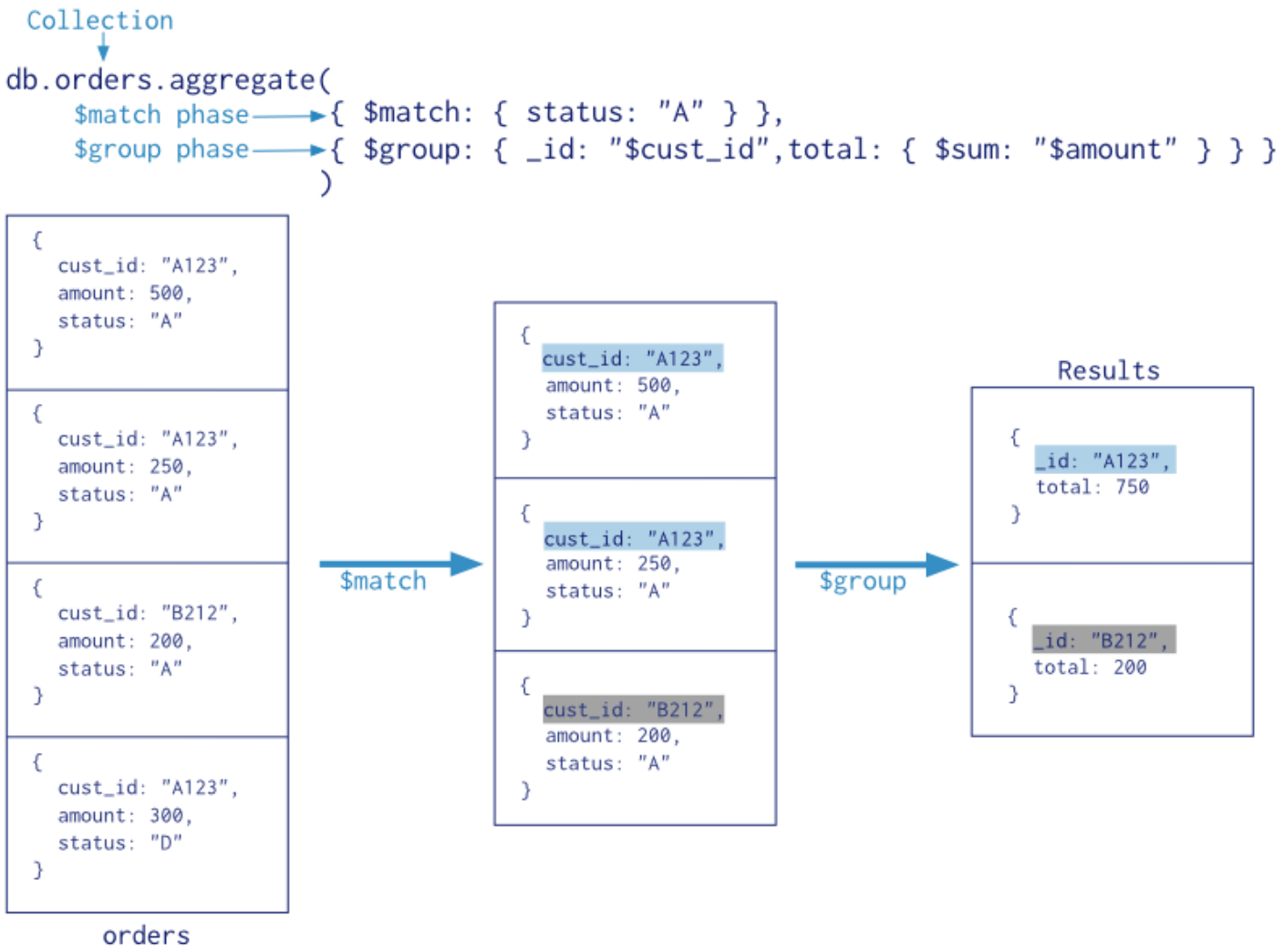
- Notes:
  - *db* refers to the current database
  - *query* is a document (query-by-example)

http://docs.mongodb.org/manual/reference/mongo-shell/

MongoDB: Querying

- Find and findOne methods (based on the *query by example* paradigm)
  - database.collection.find()
  - database.collection.find(*{ qty: { $gt: 25 } }*)
  - database.collection.find(*{ field: { $gt: value1, $lt: value2 } }*);
- The Aggregation Framework
  - An aggregation pipeline
  - Documents enter a multi-stage pipeline that transforms the documents into an aggregated results
    - *Filters* that operate like queries
    - *Document transformations* that modify the form of the output
    - Grouping
    - Sorting
    - Other operations
- MapReduce

The Aggregation Framework



Query Optimization

- The aggregation framework creates a left-deep tree access plan and applies pipelining
  - Note that the first operation is executed in parallel in all nodes containing data (exploiting **data locality**)
  - From there on, a node takes care of the query and data is shipped to it to execute the rest of the pipeline
- MongoDB barely applies any optimization technique in its querying flow
  - First versions: Nothing!
  - From version 2.6: Primitive rule-based optimization approach http://docs.mongodb.org/manual/core/aggregation-pipeline-optimization/
- Be careful when creating your pipes (you are the most important optimizer!)
  - Push selections and projections to the beginning of the pipeline
  - A cost-based approach badly needed…

  http://docs.mongodb.org/manual/core/query-optimization/

Activity: Querying in MongoDB

- *Objective: Learn how to query documents*
- *Tasks:*
  - Follow the instructions to set up MongoDB in your computer:
    http://docs.mongodb.org/manual/tutorial/install-mongodb-on-windows/
  - Take any of the schemas previously discussed and insert some documents:
    http://docs.mongodb.org/manual/core/write-operations/
  - Query the documents created by the API find method (i.e., by means of query by example):
    http://docs.mongodb.org/manual/core/read-operations/
    http://docs.mongodb.org/manual/reference/method/db.collection.find/#db.collection.find
  - Query the documents but now using the aggregation framework
    - Most usual operators:
    - $match (selection)
    - $project (projection)
    - $sort (sorting)
    - $group (group by)
      - $avg, $max, $min, $sum, etc. (computed per group)
    - Many more: boolean, operations, etc.

    http://docs.mongodb.org/manual/aggregation/
    http://docs.mongodb.org/v3.0/reference/operator/aggregation/#aggregation-pipeline-operator-reference
  - What kind of queries cannot be performed by means of the find method that the aggregation framework can answer?

MODELING

HOMEWORK

Modeling Document-Stores

- This is **not mandatory**, but read the "*migrating RDBM to MongoDB*" document uploaded to Atenea to learn more about modeling documents
  - When doing so, please, be critic. There will be some claims that might be rather controversial
  - In general, however, this white paper is sound and it is aligned with what we have discussed in the session

Thanks for your attention!