NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

DEPARTMENT OF CHEMICAL ENGINEERING

ADVANCED PROCESS SIMULATION

SQL vs. NoSQL

Author: Cansu Birgen
Supervisors:
Prof. Heinz Preisig
John Morud

December 17, 2014

Content

- Aim
- Motivation
- Background
- MongoDB
- Pymatgen
- Conclusions

Aim

Investigation of *NoSQL* database approach, specifically MongoDB in comparison with *SQL* approach in terms of data storage, organization and manipulation applications for chemical databases.

Motivation

Growing data volumes, data variety and complexity, and the rate at which the data needs to be analyzed require new tools for simple queries as well as complex analysis.

Background

SQL database is in which the data is organized based on the relational model of data providing a declarative method for data and query specification.

NoSQL provides a mechanism for storage and retrieval of data which is modeled in a way different than SQL approach.

Background

Common charateristics of NoSQL databases

- Data replication: redundancy and availability
- Horizontal scalability: better storage and process capacities
- No pre-defined structure: flexibility
- Open source: low costs

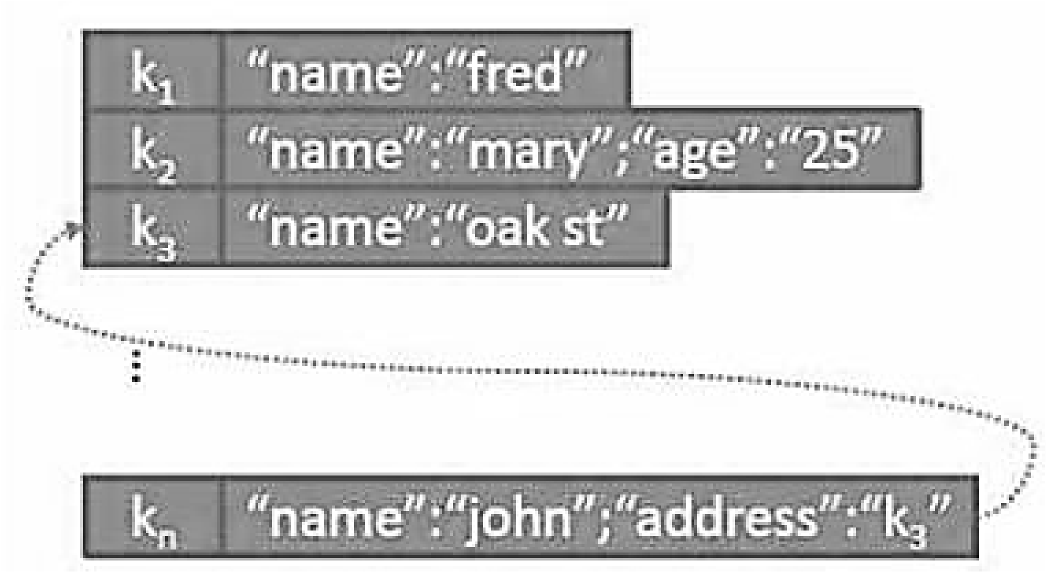
Background

Classification of NoSQL databases

- Key-value stores
- Column stores
- Graph databases
- Document stores

Background

MongoDB is a schemaless document store database written in C++ and developed in an open-source project.



k_1 "name":"fred"

k_2 "name":"mary";"age":"25"

k_3 "name":"oak st"

⋮

k_n "name":"john";"address":"k_3"

Background

Charateristics of MongoDB

- Document store
- High performance
- High availability
- Automatic scaling

Background

Terminology and concepts in SQL and MongoDB

SQL	**MongoDB**
database	database
table	collection
row	document or BSON document
column	field
index	index
table joins	embedded documents and linking
primary key (specify any unique column or column combinations as primary key)	primary key (the primary key is automatically set to the _id field in MongoDB)
aggregation (e.g. by group)	aggregation pipeline

Background

SQL statements and the corresponding MongoDB statements

SQL	**MongoDB**

SELECT *
FROM users
	db.users.find()

SELECT user_id, status
FROM users
WHERE status = "A"
	db.users.find(
	{ status: "A" },
	{ user_id: 1, status: 1, _id: 0 }
	)

UPDATE users
SET status = "C"
WHERE age > 2
	db.users.update(
	{ age: { $gt: 25 } },
	{ $set: { status: "C" } },
	{ multi: true }
	)

INSERT INTO users(user_id, age, status) VALUES("bcd001", 45, "A")	db.users.insert({ user_id: "bcd001", age: 45, status: "A" })
DELETE FROM users	db.users.remove({ })

MongoDB Example

A chemical database developed within a open chemistry project called *MongoChem* which uses *MongoDB* for data storage is queried by using *Mongo Query Language*.

MongoDB Example

Collection **Query criteria**

```
db.molecules.find( { atomCount: { $gt: 5, $lte:
10 } }, { name:1, atomCount:1, _id:
0} ).limit(5).sort( {name: 1 } )
```

Modifier

```
{ "atomCount" : 8 }
{ "atomCount" : 6 }
{ "name" : "", "atomCount" : 6 }
{ "atomCount" : 9, "name" :
"(methylthio)methane" }
{ "atomCount" : 8, "name" : "1,2-dichloroethane" }
```

MongoDB Example

Collection **Embeded sub-documents**

```
db.molecules.find( { "descriptors.tpsa" : 20.2,
"descriptors.xlogp3" : 0 }, { name: 1, descriptors: 1,
_id: 0 } ).limit(2)
```

Modifier

```
{ "name" : "2-chloroethanol", "descriptors" :
{ "tpsa" : 20.2, "new boiling_point" : 127, "new
boiling point" : 127, "vabc" : 67.14972242352783,
"boiling_point" : 127, "new melting_point" : -63,
"boiling" : 127, "mass" : 80.5135, "melting_point" :
-63, "xlogp3" : 0, "rotatable-bonds" : 1, "melting jlk"
: -63 } }
{ "descriptors" : { "tpsa" : 20.2, "xlogp3" : 0,
"mass" : 639.0474, "rotatable-bonds" : 31, "vabc" :
760.7044737816262 }, "name" : "2-
```

MongoDB Example

Collection **Embeded sub-documents** **Query criteria**

```
db.molecules.findOne( { "atoms.elements.number" : { $gt: 5 } }, {name: 1,
_id: 0, atoms: 1 } )
```

```
{ "atoms" : { "elements" : { "number" : [      8,    8,    8,    8,    7, 6,       6,    6,
     6,    6,    6,    6,    6,    6,    1,    1, 1,       1,    1,    1,    1,    1,    1,    1,    1,
     1,    1, 1,       1,    1,    1,    1 ] }, "coords" : { "3d" : [   -0.1902, 1.3409,
0.3005,  2.448,    -2.3678, 1.071,    2.2391,  -1.9287, -1.1496, 1.2775,  2.3629,
     -1.1618, -2.1546, -0.6965, -0.0209, -0.7847, -0.9897, 0.4738,  0.1977,
0.0264,  -0.1011, -2.2003, -0.7415, -1.5539, -3.1585, -1.7242, 0.5181,  -
2.6089,  0.698,    0.4303,  1.6026,  -0.2265, 0.4447,  2.1184,  -1.5795, 0.0175,
     0.6202,  2.3512,  -0.1299, 0.5936,  3.4751,  0.8612,  -0.8149, -0.9382,
1.5693,  -0.5333, -2.0143, 0.177,    0.2374,  -0.0302, -1.1943, -1.6769, 0.1203,
     -1.9745, -1.7507, -1.6822, -1.8849, -3.25,     -0.6979, -1.8617, -3.155,   -
1.6611,  1.6102,  -4.1514, -1.493,   0.1213,  -2.838,   -2.717,   0.1888,  -
2.2788,  0.8768,  1.458,    -2.2939, 1.4619,  -0.2844, -3.7052, 0.7061,  0.4169,
     1.598,    -0.16,     1.5404,  2.3437,  0.507,    0.1121,  0.9681,  3.1259,
1.8267,  -0.4251, 3.8589,  0.9587,  1.2389,  4.2848,  0.5088,  2.7859,  -
3.2422,  0.7823 ] } }, "name" : "(2-acetoxy-3-carboxy-propyl)-trimethyl-
ammonium" }
```

Pymatgen

Pymatgen (Python Materials Genomics) is a robust, open-source Python library for materials analysis contributing to Materials Project which is an initiative to make calculated properties of all known inorganic materials available to materials researchers.

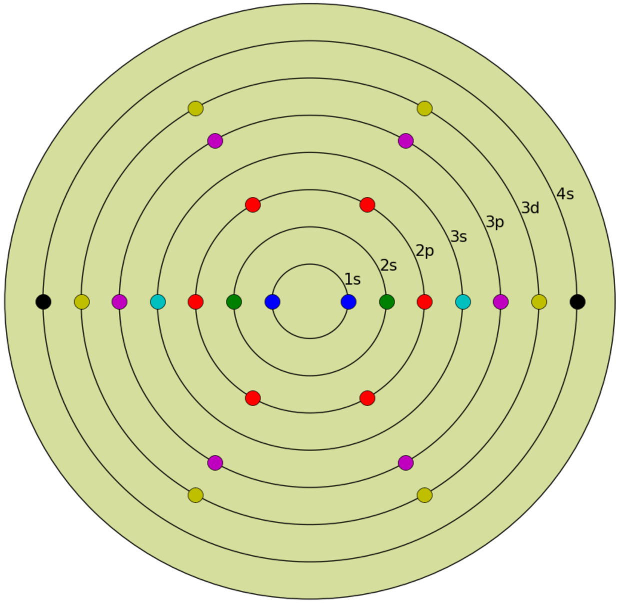
Pymatgen

MongoDB acts as

- Workflow manager for managing the state of high-throughput calculations.
- Execution engine for storage and analytics for the calculation results.
- Back-end repository as a searchable back-end for data dissemination.

Pymatgen

Full electronic structure information is available in Pytmatgen library. Matplotlib which is a Python 2D plotting library is used for drawing electron structure of Fe atom.



Conclusions

Drivers

- Flexible and schemaless data model: storage of any type of data, create and handle complex data models.
- Sub-document creation, insertion and storage without any limitation or imposed structure.
- Explicit array storage.
- Horizontal scaling: no limit in scaling up.
- Open-source and growing community: low costs and high accessibility.
- Mapping methods from SQL to MongoDB

Conclusions

Obstacles

- Favor availability over consistency: returning any value before converging.
- Security concerns: fine-grained permissions or access control yet to be provided.
- Immaturity: tested less by less users.

Final word

Both SQL and NoSQL approaches have limitations and advantages which strongly depend on the types of the application, its requirements and priorities.

Thank You