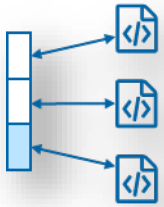
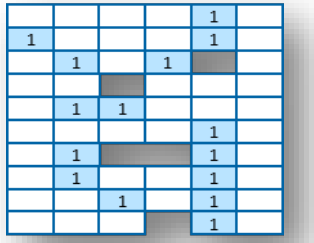
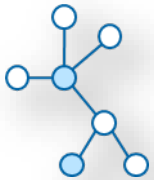
NoSQL

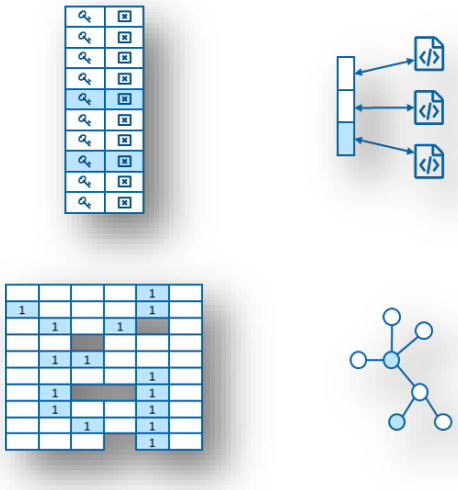






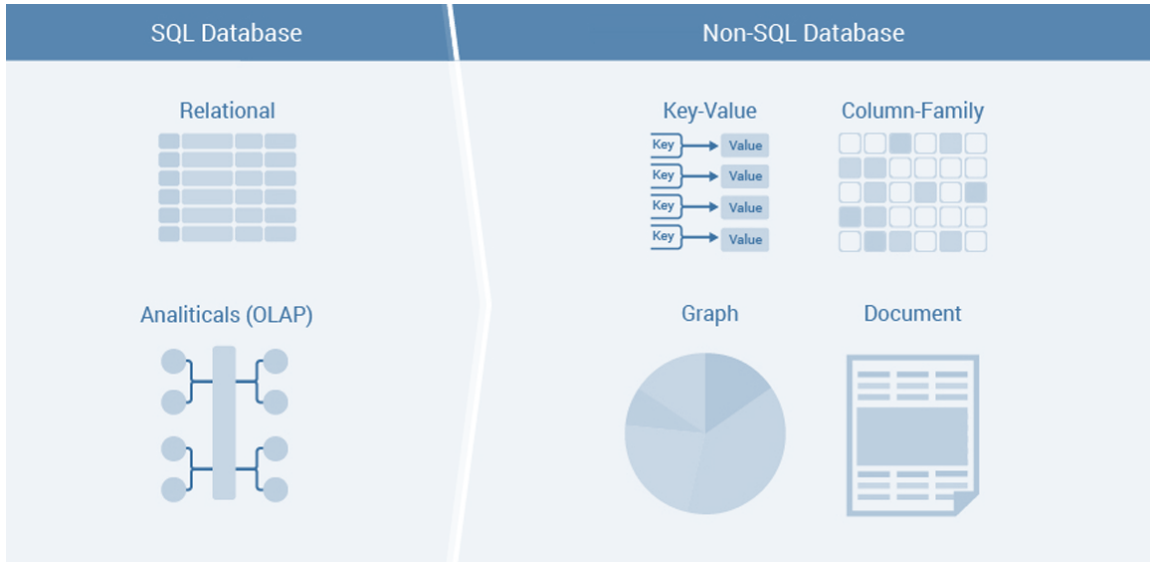
Content

- Introduction
- Key-Value
- Document-oriented
- Column-oriented
- Graph



Content

- Introduction
- Key-Value
- Document-oriented
- Column-oriented
- Graph



Introduction

NoSQL - **N**ot **o**nly **SQL**

- originally referring to non SQL or non relational
- provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases
- the term *NoSQL* was used by Carlo Strozzi in 1998 to name his lightweight, Strozzi NoSQL open-source relational database that did not expose the standard SQL interface, but was still relational
- Johan Oskarsson reintroduced the term *NoSQL* in early 2009 when he organized an event to discuss open source distributed, non relational databases

system

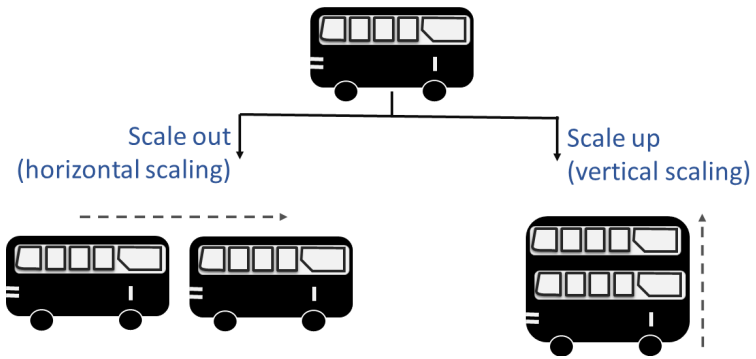
↕

movement

Introduction

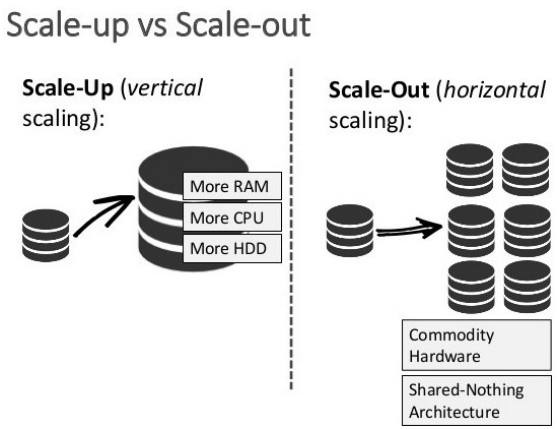
Horizontal Scaling

Why NoSQL? ➞ Relational DBs have poor horizontal scalability



Scaling

- horizontal
  add more nodes to (or remove nodes from) a system, such as adding a new computer to a distributed software application.
- vertical
  means to add resources to (or remove resources from) a single node in a system, typically involving the addition of CPUs or memory to a single computer.



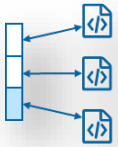
Introduction

Types

Key-Value

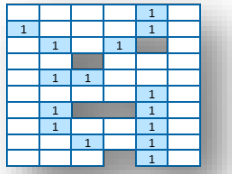
Apache CouchDB
ArangoDB
BaseX
Clusterpoint
Couchbase
Cosmos DB
IBM Domino
MarkLogic
OrientDB
Qizx
RethinkDB

Document



Aerospike
Apache Ignite
ArangoDB
Couchbase
Dynamo
FairCom c-treeACE
FoundationDB
InfinityDB
MemcacheDB
MongoDB
MUMPS
Oracle NoSQL Database
OrientDB
Redis
Riak
Berkeley DB
SDBM/Flat File dbm
ZooKeeper

Column



Accumulo
Cassandra
Druid
Hbase
Vertica

Graph

AllegroGraph
ArangoDB
InfiniteGraph
Apache Giraph
MarkLogic
Neo4J
OrientDB
Virtuoso

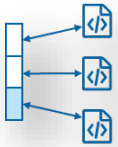
Introduction

Types

Key-Value

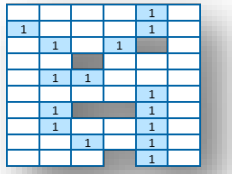
Apache CouchDB
ArangoDB
BaseX
Clusterpoint
Couchbase
Cosmos DB
IBM Domino
MarkLogic
OrientDB
Qizx
RethinkDB

Document



Aerospike
Apache Ignite
ArangoDB
Couchbase
Dynamo
FairCom c-treeACE
FoundationDB
InfinityDB
MemcacheDB
MongoDB
MUMPS
Oracle NoSQL Database
OrientDB
Redis
Riak
Berkeley DB
SDBM/Flat File dbm
ZooKeeper

Column



Accumulo
Cassandra
Druid
Hbase
Vertica

Graph

AllegroGraph
ArangoDB
InfiniteGraph
Apache Giraph
MarkLogic
Neo4J
OrientDB
Virtuoso

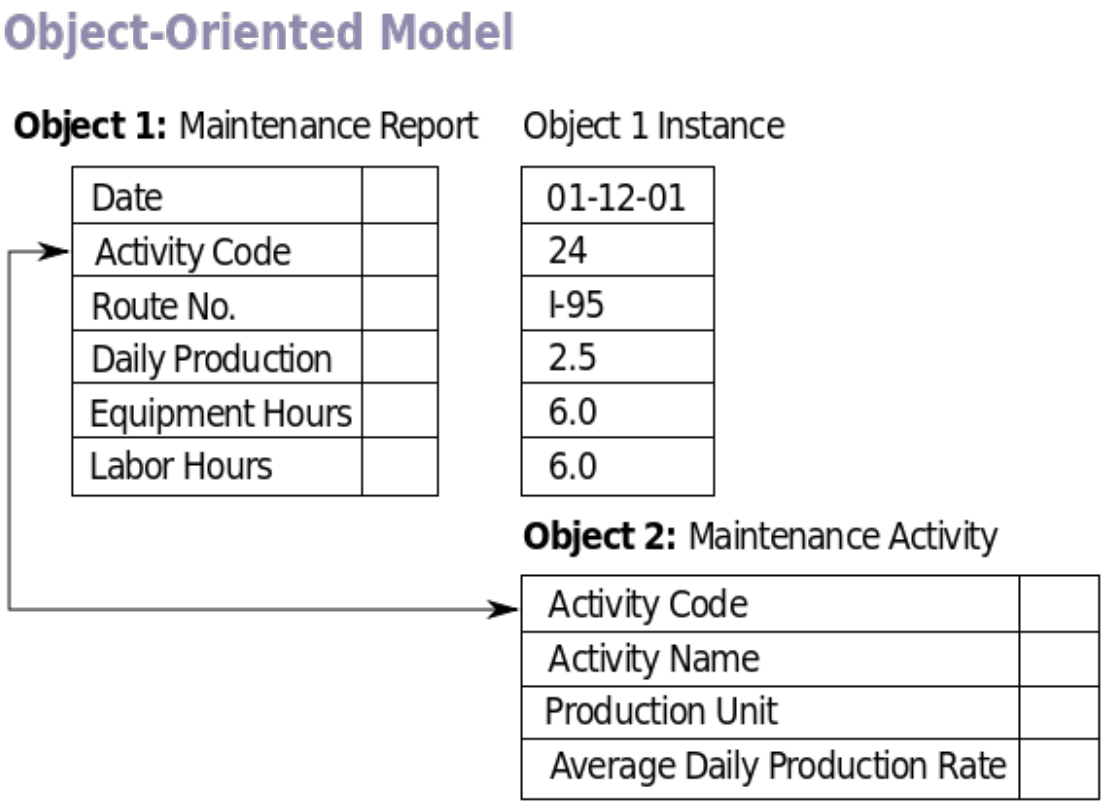
some NoSQL databases are multimodal

Introduction

Object Databases

An object database is a database management system in which information is represented in the form of objects as used in object-oriented programming.

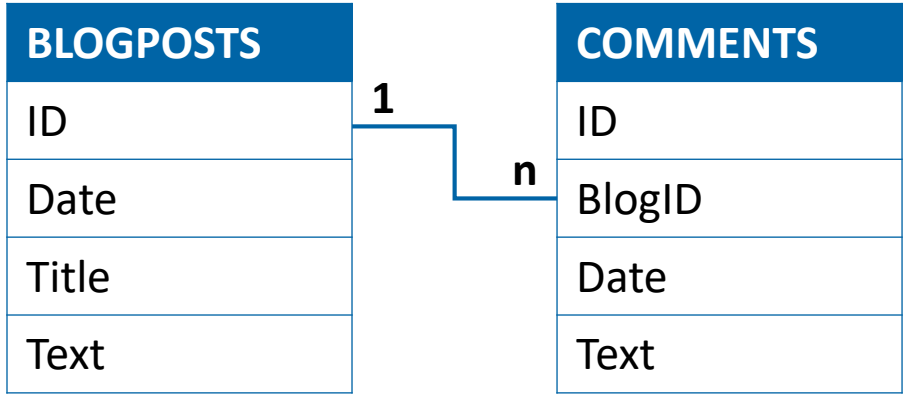
Object databases are non-relational databases and, thus, non SQL. However, since object databases have a similar horizontal scalability as relational DBs, they are not part of the NoSQL movement.

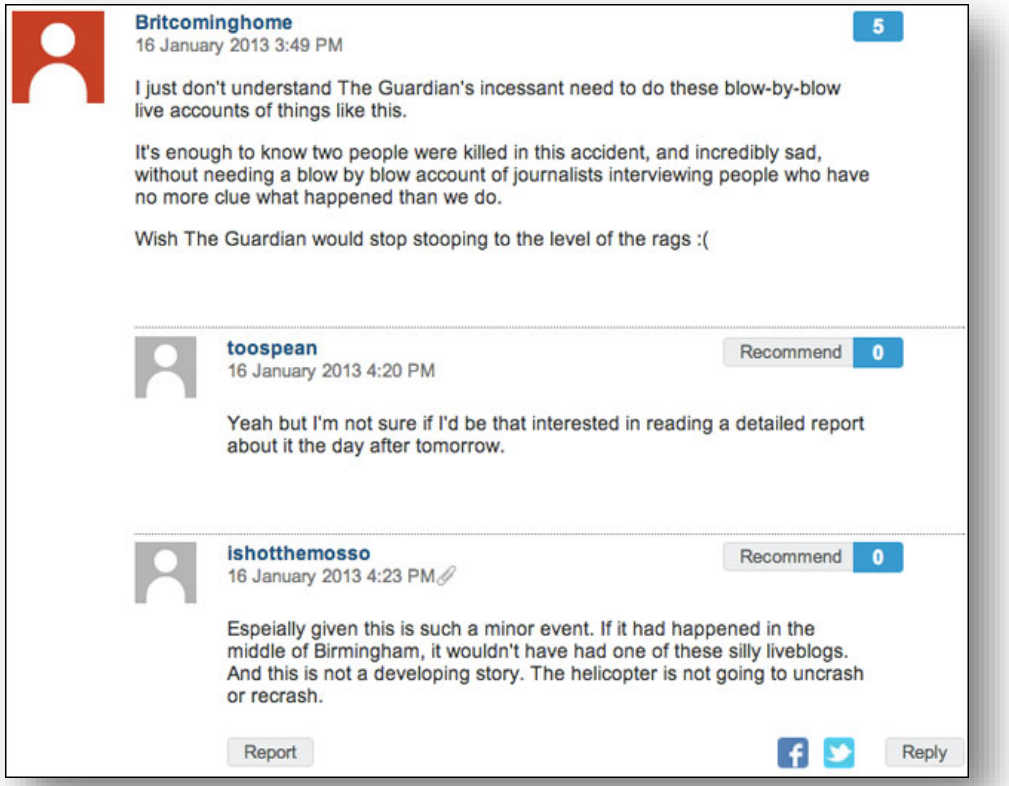


Introduction

Example Scenario

Blogging-Website
- Blogposts
- Comments





Content

- Introduction
- Key-Value
- Document-oriented
- Column-oriented
- Graph

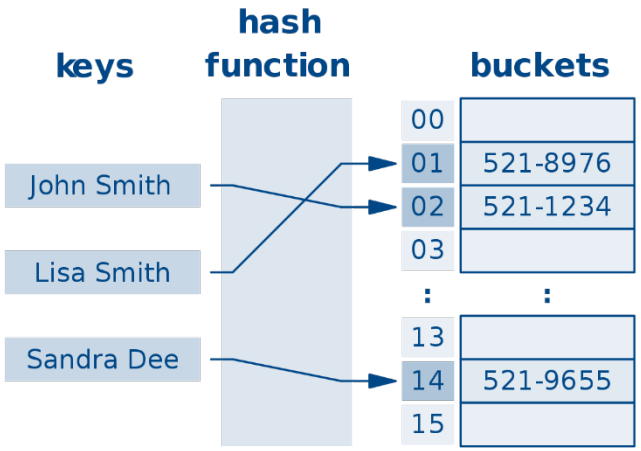
NoSQL

Key-Value

Basics

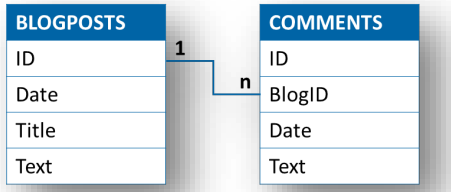
- Data is stored in a key-value pair
- Similar to a dictionary (as in Python)
- Keys must be unique, single identifiers
- Values can be anything (single value, object, record, file, etc.)
- The value's datatype is not known to the database ➞ no JOINS possible
- Use of efficient indexing methods such as Hash

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623



Key-Value

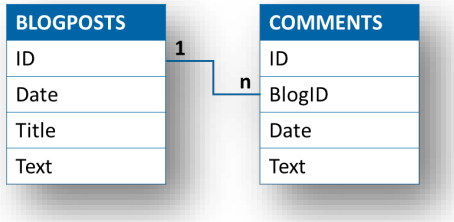
Example



0126	2017-03-05, 'Transfer Record', 'New record on the...'
0976	2015-21-11, 'Kitten', 'Who has seen my...'
3857	0126, 2017-03-09, 'Unbelievable, I thought...'
5847	0126, 2017-03-11, 'I don't believe...'
9864	...

Key-Value

Example



0126	['date':'2017-03-05';'title':'Transfer Record';'text':'New record on the...']
0976	['date':'2015-21-11';'title':'Kitten';'text':'Who has seen my...']
3857	['blogid':'0126';'date':'2017-03-09';'text':'Unbelievable, I thought...']
5847	['blogid':'0126';'date':'2017-03-11';'text':'I don’t believe...']
9864	...

Key-Value

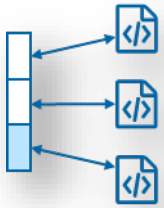
Notes

- The database knows nothing about the values (no record/data «types», no relationships) ➞ almost no functionality
- No schema ➞ no complexity, high flexibility
- Key-Values pairs can only be update as a whole. Partially updating a value is not possible
- Perfectly suited to retrieve a single dataset from a huge pool (web store, blog, user profiles, posts, etc.)

Data model	Performance	Scalability	Flexibility	Complexity	Functionality
Relational	variable	variable	low	moderate	relational algebra
Key–Value	high	high	high	none	variable (none)
Document	high	variable (high)	high	low	variable (low)
Column	high	high	moderate	low	minimal
Graph	variable	variable	high	high	graph theory

Content

- Introduction
- Key-Value
- Document-oriented
- Column-oriented
- Graph

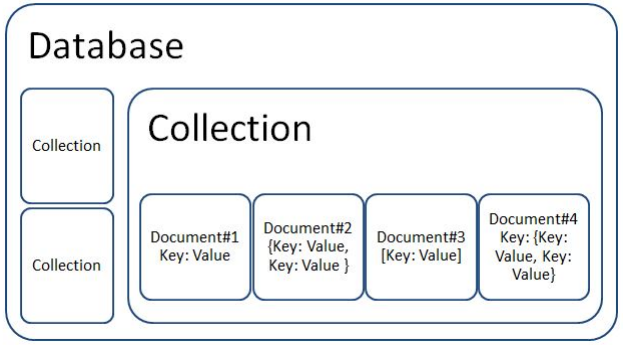


NoSQL

Document-oriented

Basics

- Data is stored in documents with a **given data format**
- Typical data formats are JSON and XML
- Indexing based on document properties (filenames are irrelevant)
- Documents containing similar content are grouped in **collections**
- Document structure is not fixed
- Allows defining rules based on the content

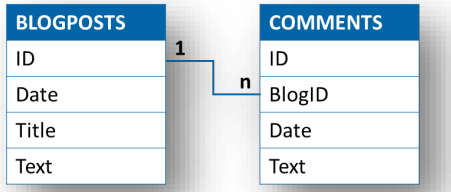


Customer Document

```
"customer" =
{
   "id": "Customer:1",
   "firstName": "John",
   "lastName": "Wick",
   "age: 25,
   "address": {
        "country": "US",
        "city": "New York",
        "state": "NY"
        "street": "21 2nd Street",
      },
   "hobbies": [ Football, Hiking ],
   "phoneNumbers": [
      {
        "type": "Home",
        "number": "212 555-1234"
      },
      {
        "type": "Office",
        "number": "616 565-6789"
      }
    ]
}
```

Document-oriented

Example



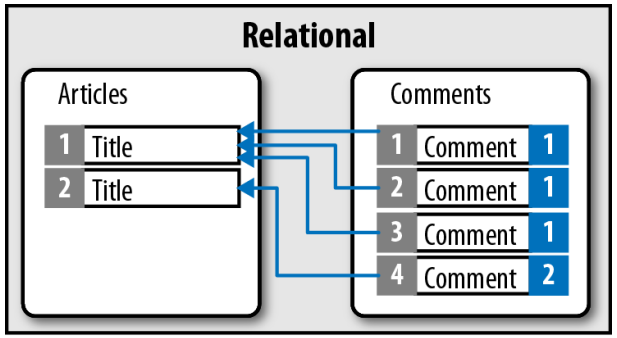
file 1

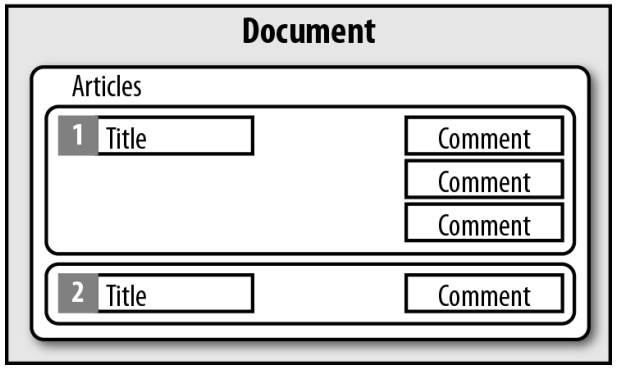
```
"_id":"0126",
"date":"2017-03-05",
"title":"Transfer Record",
"text":"New record on the... "
"comments":[
{"date":"2017-03-09",
 "text":"Unbelievable, I thought... "},
{"date":"2017-03-11",
 "text":"I don't believe... "}]
```

file 2

```
"_id":"0976",
"date":"2015-21-11",
"title":"Kitten",
"text":"Who has seen my..."
"comments":[]
```

Blogpost Collection





Document-oriented

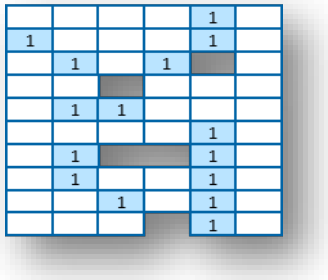
Notes

- The database knows the data format of the document ➞ functionality possible
- No fixed schema ➞ low complexity, high flexibility
- Making use of the data format, documents can also be updated partially
- Used for large amount of structured or semi-structured data where schema-flexibility is needed (e.g. CMS)

Data model	Performance	Scalability	Flexibility	Complexity	Functionality
Relational	variable	variable	low	moderate	relational algebra
Key–Value	high	high	high	none	variable (none)
Document	high	variable (high)	high	low	variable (low)
Column	high	high	moderate	low	minimal
Graph	variable	variable	high	high	graph theory

Content

- Introduction
- Key-Value
- Document-oriented
- Column-oriented
- Graph



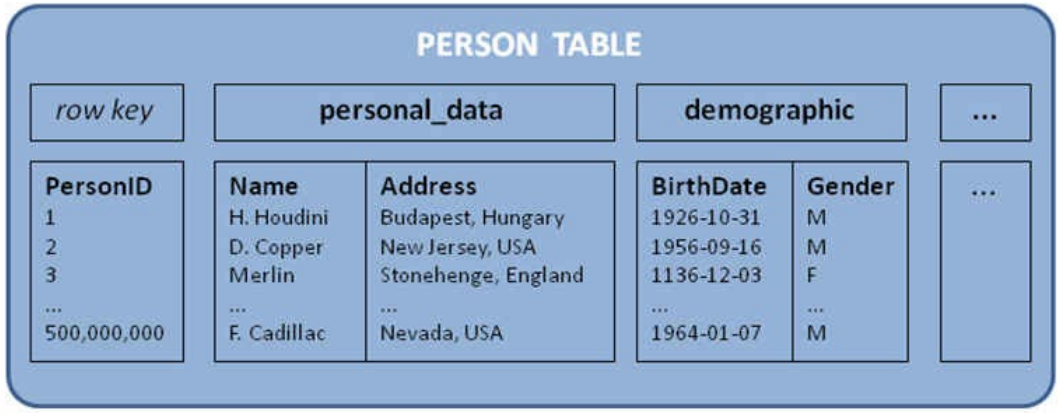
Column-oriented

Basics

- Data is stored in **tables**
- Records are modeled as **rows**, attributes are stored in **columns**

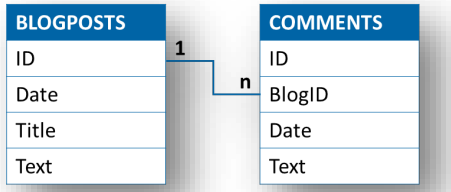
like in a relational database, but...

- Columns are grouped in **column families**
- Only column families have to be defined on creation
- **New columns can be added** on-the-fly on a record-basis
- Data is stored column-wise
- There are no database managed relationships



Column-oriented

Example



BLOGPOSTS			
row_key	blogpost_data (column family)		
id	**title**	**date**	**text**
0126	Transfer Record	2017-03-05	New record on the...
0976	Kitten	2015-21-11	Who has seen my...

COMMENTS			
row_key	comments_data (column family)		
id	**date**	**text**	**blog_id**
3857	2017-03-09	Unbelievable, I thought...	0126
5847	2017-03-11	I don’t believe...	0126

Column-oriented

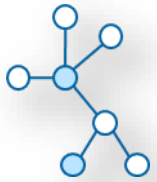
Notes

- Storing column-wise allows to distribute the attributes of a record ➞ high scalability
- Column families must be defined on creation ➞ limited flexibility
- No database-wise management of relationships ➞ minimal functionality
- Records can be updated partially
- Used for large amount of structured data where some schema-flexibility is needed (e.g. Logging)

Data model	Performance	Scalability	Flexibility	Complexity	Functionality
Relational	variable	variable	low	moderate	relational algebra
Key–Value	high	high	high	none	variable (none)
Document	high	variable (high)	high	low	variable (low)
Column	high	high	moderate	low	minimal
Graph	variable	variable	high	high	graph theory

Content

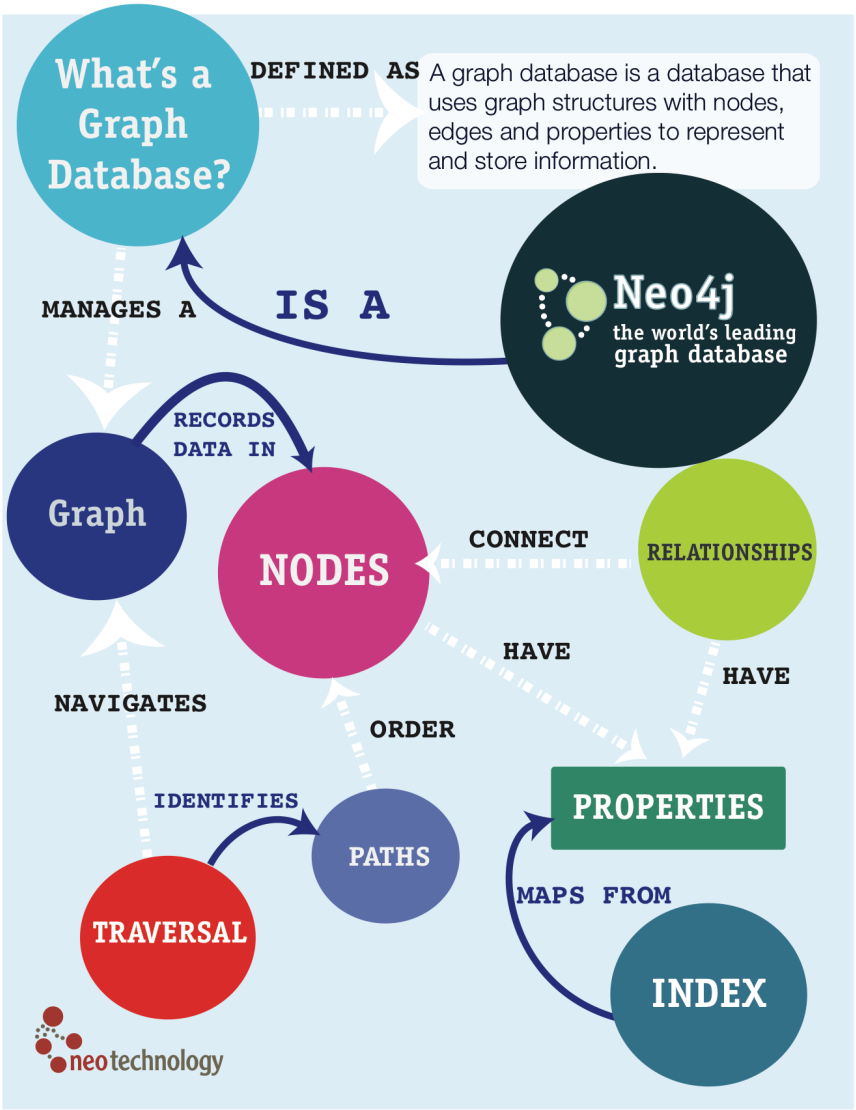
- Introduction
- Key-Value
- Document-oriented
- Column-oriented
- Graph



Graph

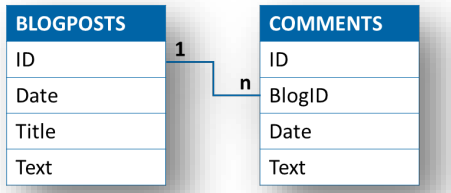
Basics

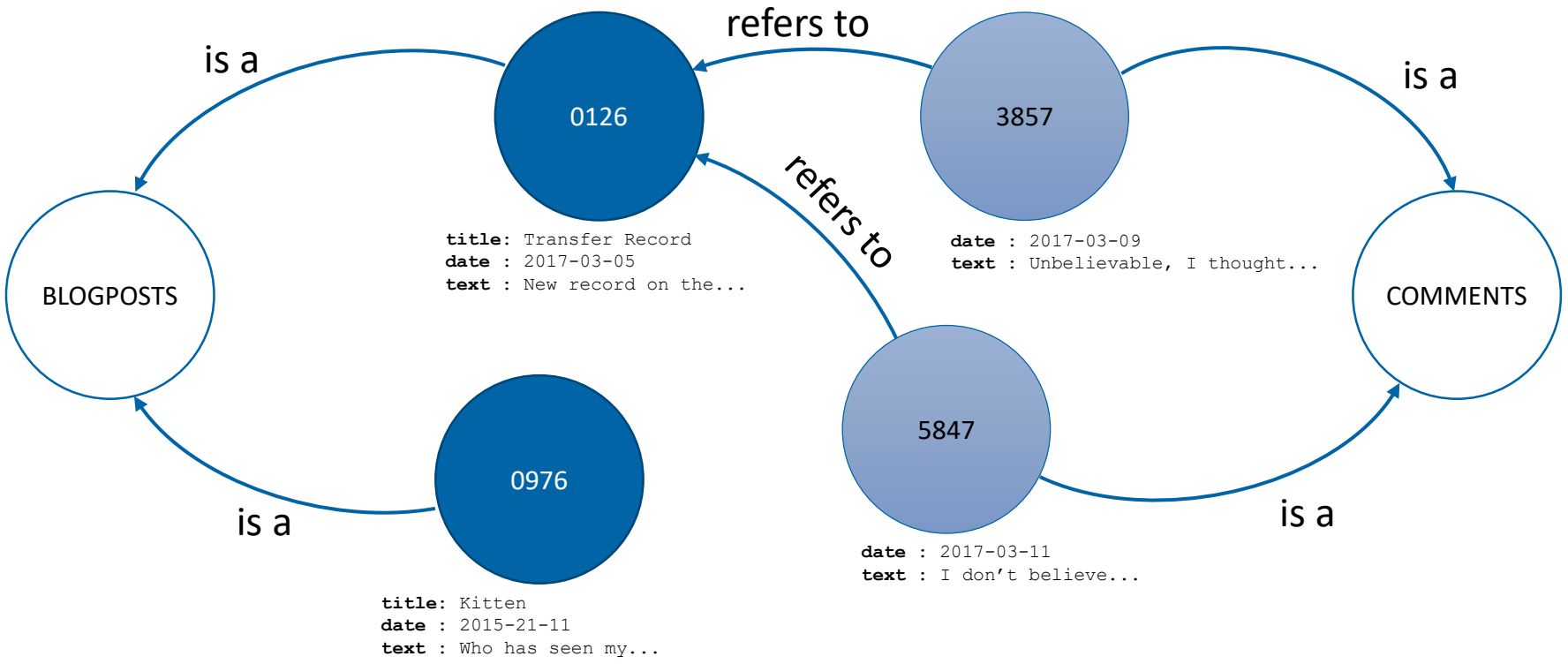
- Makes use of **graph theory**
- Stores data in **nodes**
- Models relationships by using **edges**
- Nodes and edges have properties



Graph

Example





Graph

Notes

- Uses the power of almost 300 years of graph theory ➞ functionality
- High complexity, very difficult to visualize
- Nodes and edges can be added on-the-fly
- Nodes and edges can be updated partially

Data model	Performance	Scalability	Flexibility	Complexity	Functionality
Relational	variable	variable	low	moderate	relational algebra
Key–Value	high	high	high	none	variable (none)
Document	high	variable (high)	high	low	variable (low)
Column	high	high	moderate	low	minimal
Graph	variable	variable	high	high	graph theory