Harri Kauhanen

2010-03-09

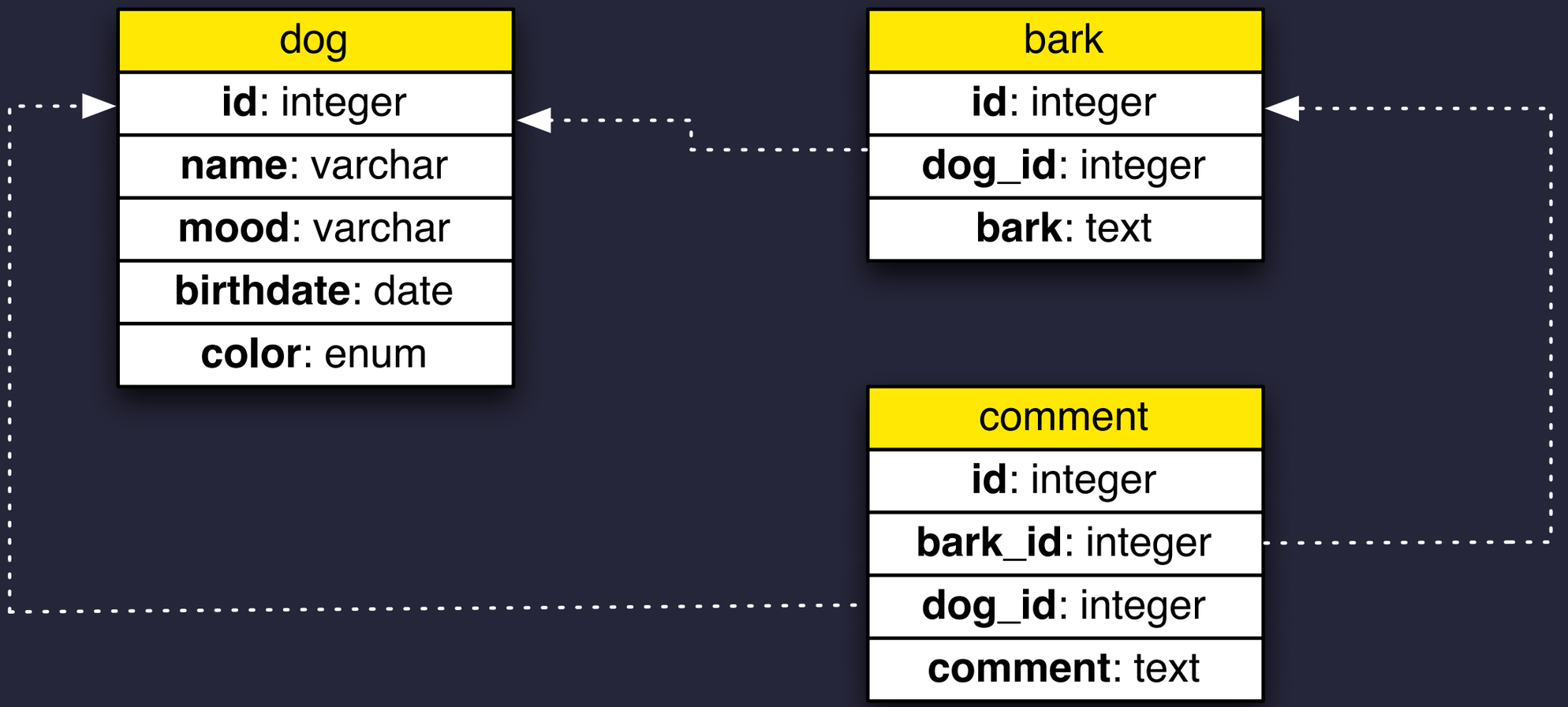
NoSQL databases

Database paradigms

- Relational (RDBMS)
- NoSQL
  - Key-value stores
  - Document databases
  - Wide column stores (BigTable and clones)
  - Graph databases
- Others

Relational databases

- ACID (Atomicity, Consistency, Isolation and Durability)
- SQL
- MySQL, PostreSQL, Oracle, ...



id	name	mood	birth_date	color
12	Stella	Happy	2007-04-01	NULL
13	Wimma	Hungry	NULL	black
9	Ninja	NULL	NULL	NULL

Key-value stores

- "One key, one value, no duplicates, and crazy fast."
- It is a Hash!
- The value is binary object aka. "blob" – the DB does not understand it and does not want to understand it.
- Amazon Dynamo, MemcacheDB, ...

“key”

“value”

dog_12

```
...
name_
€#_Stella^^^
mood_€#_Happy^^^
birthdate%///135465645)
...
```

Document databases

- Key-value store, but the value is (usually) structured and "understood" by the DB.
- Querying data is possible (by other means than just a key).
- Amazon SimpleDB, CouchDB, MongoDB, Riak, ...

“key”

dog_12

“document”

```
{
  type: “Dog”,
  name: “Stella”,
  mood: “Happy”,
  birthdate: 2007-04-01
}
```

VS.

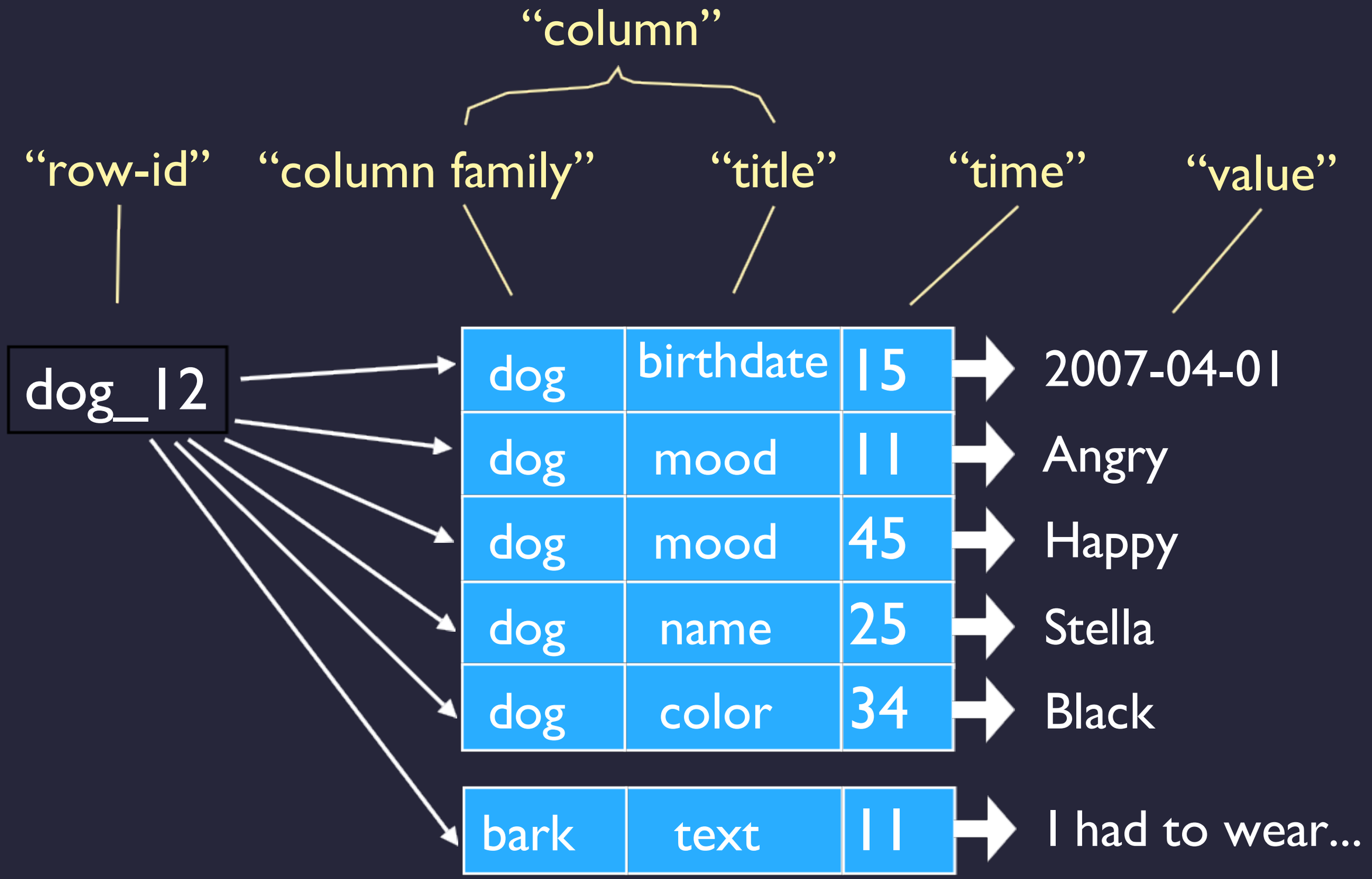
id	name	mood	birth_date	color
12	Stella	Happy	2007-04-01	NULL
13	Wimma	Hungry	NULL	black
9	Ninja	NULL	NULL	NULL

dog_12

```
{
 type: “Dog”,
 name: “Stella”,
 mood: “Happy”,
 birthdate: 2007-04-01,
 barks: [
   {
     bark: “I had to wear stupid..”
     comments: [
       {
         dog_id: “dog_4”,
         comment: “You look so cute!”
       }, {
         dog_id: “dog_14”,
         comment: “I hate it, too!”
       }
     ]
   }
 ]
}
```

Wide column stores

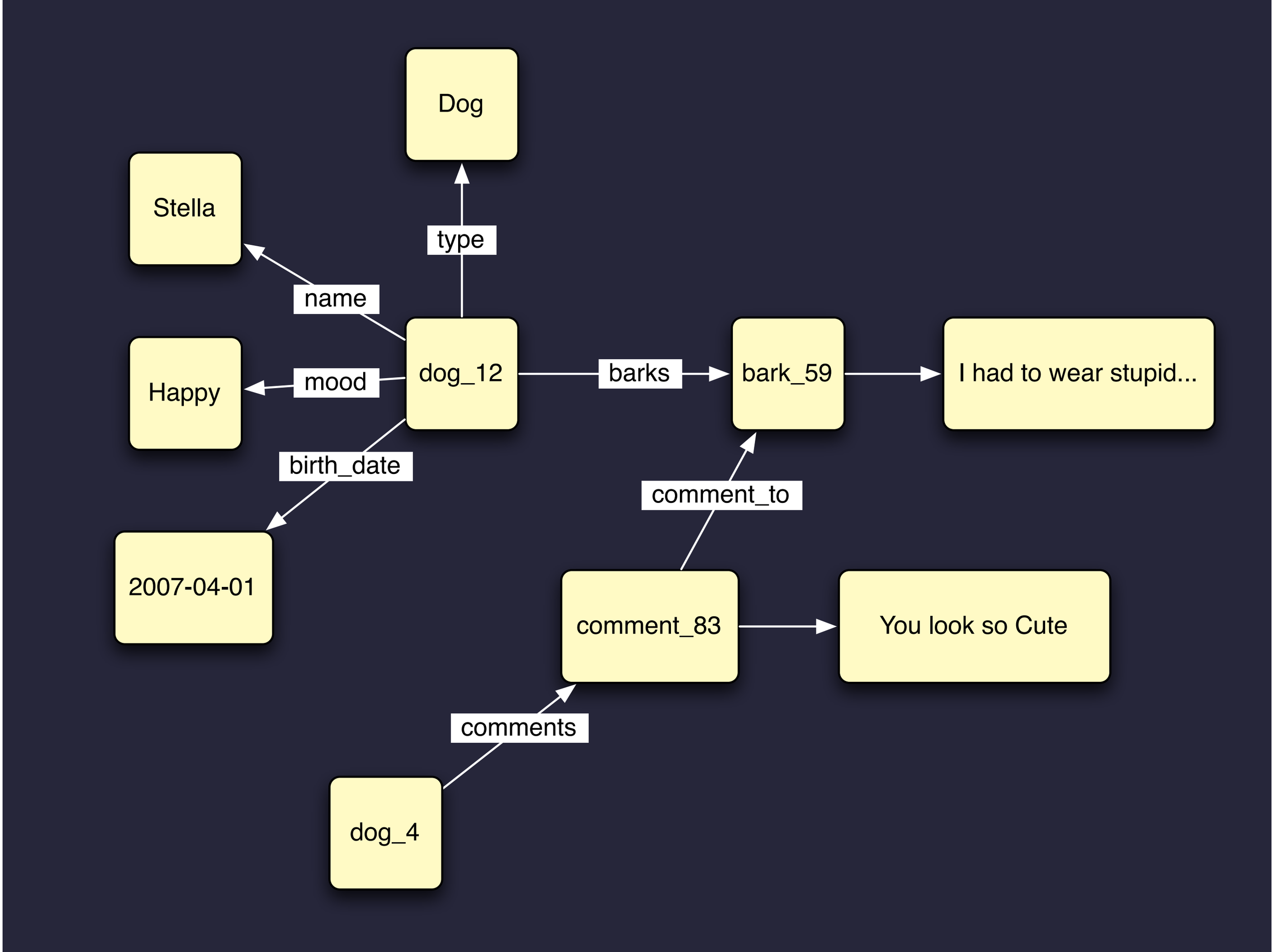
- Often referred as “BigTable clones”
- "a sparse, distributed multi-dimensional sorted map"
- Google BigTable, Cassandra (Facebook), HBase, ...


"column"
"row-id"
"column family"
"title"
"time"
"value"
dog_12
dog
birthdate
15
2007-04-01
dog
mood
11
Angry
dog
mood
45
Happy
dog
name
25
Stella
dog
color
34
Black
bark
text
11
I had to wear...

Graph databases

- "Relation database is a collection loosely connected tables" whereas "Graph database is a multi-relational graph".

- Neo4j, InfoGrid, ...


Dog
Stella
type
name
Happy
mood
dog_12
barks
bark_59
I had to wear stupid...
birth_date
2007-04-01
comment_to
comment_83
You look so Cute
comments
dog_4

- Relationships in RDBMS are “weak”.
  - You may “define” one by *using constraints*, documenting a relationship, writing code, using naming conventions etc.
- Relationships in graph databases are first class citizens.
- There are no relationships in key-value stores, document databases and wide column stores.
  - You may “define” one by *using validations*, documenting a relationship, writing code, using

- Relational databases have almost limitless indexing, and a very strong language for dynamic, cross-table, queries (SQL)
  - That's why they handle all kinds of relationships well and dynamically.
- NoSQL databases...
  - ...might have limited support for dynamic queries and indexing
  - ...don't support JOIN like operations of SQL
  - ...but you can store some relationships into document itself

How to query NoSQL?

- Key-Value
- Row-id/column-family:title[/time]
  - `“stella_12”/”dogs”:”name”` → `Stella`
- Graph traversal
- API
- Query-language
- Integration to indexing and search engines

Map-Reduce

- "MapReduce is a programming model and an associated implementation for processing and generating large data sets."

- Often JavaScript (NoSQL implementations)

Map-function

```
function map(doc) {
 if (doc['type'] == 'Dog') {
   emit(doc['mood'], doc['birthdate']);
 }
}
```

- Generates “indexed view” of data/documents
- This view is just another hash, but both key and value can be “anything”

Reduce-function

- Aggregate results for a “view” (after the Map-function)

```
function reduce(mood, listOfBirthdates) {
 return averageBirthDate(listOfBirthdates);
}
```

- Map-phase is easy to distribute, but you is also easy to write poor Reduce-functions

Theorems

- CAP
  - Consistency, Availability, Partition tolerance
  - “Pick two”
- N/R/W (adjusting CAP)

No consistency?

Eventual consistency

Why NoSQL?

- Schema-free
- Massive data stores

- Scalability
- Some services simpler to implement than using RDBMS
- Great fit for many “Web 2.0” services

Why NOT NoSQL

- DRBMS databases and tools are mature
- NoSQL implementations often “alpha”
- Data consistency, transactions
- “Don’t scale until you need it”

RDBMS vs. NoSQL

- Strong consistency vs. Eventual consistency
- Big dataset vs. HUGE datasets
- Scaling is possible vs. Scaling is easy
- SQL vs. Map-Reduce
- Good availability vs. Very high availability

Questions?