NoSQL

Big Data Systems

Dr. Rubi Boim

NoSQL

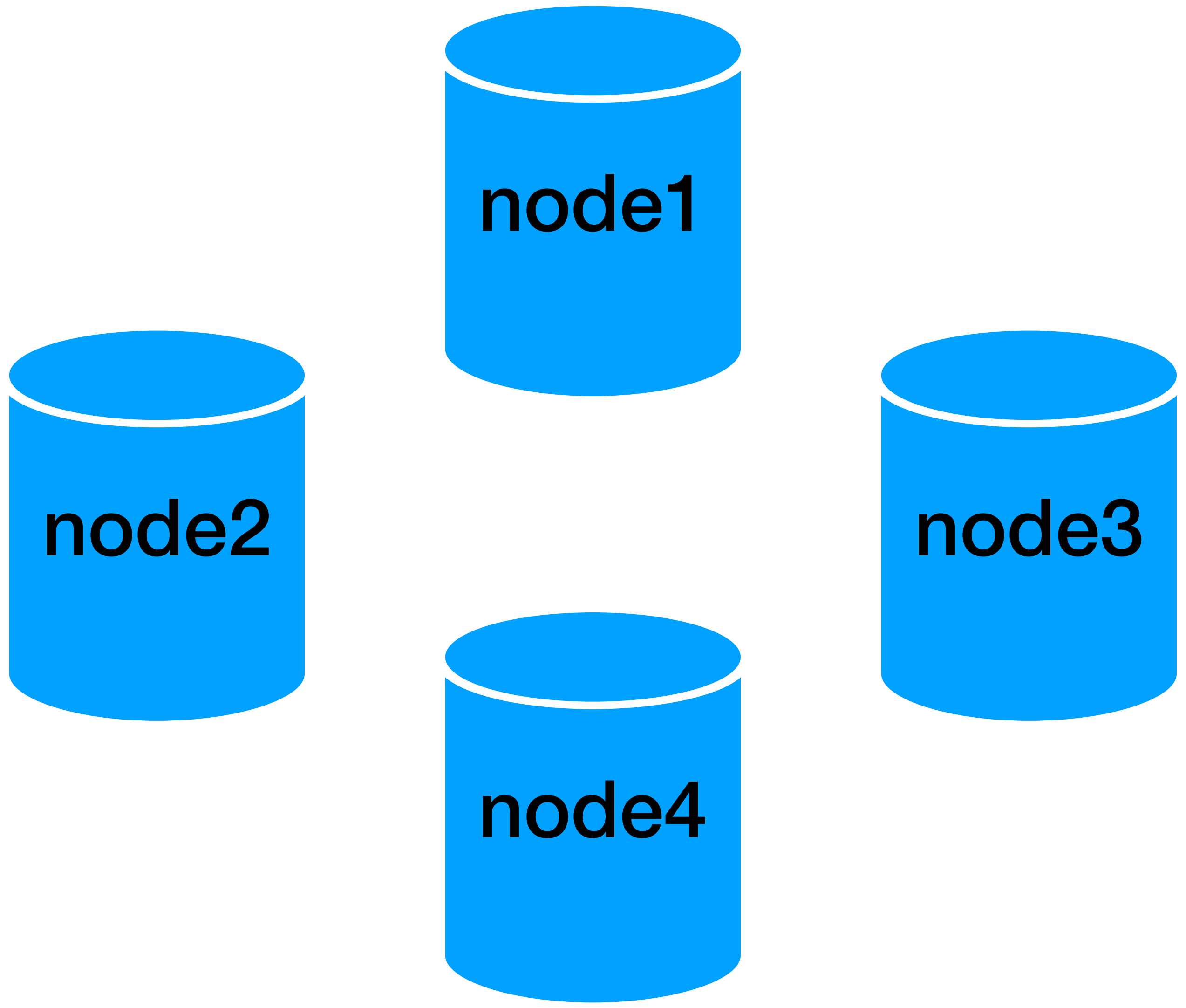
- Non SQL
- Not Only SQL
- Non schema SQL
- **Non Relational SQL**

SQL is an API

- Most (distributed) Big Data storage / database systems

NoSQL - motivation

- Scaling issues of RDBMS
- Variety of data (schema-less)
- Read / write performance



- Product needs of Google / Amazon / Facebook…

NoSQL - popular properties

- “Simpler" design / API
- Availability over Consistency
- **Performance over ACID transactions**
  extremely fast read/write at scale
- Distributed - scale wide (commodity hardware)
- Schema less
- Multi data center

NoSQL - main types

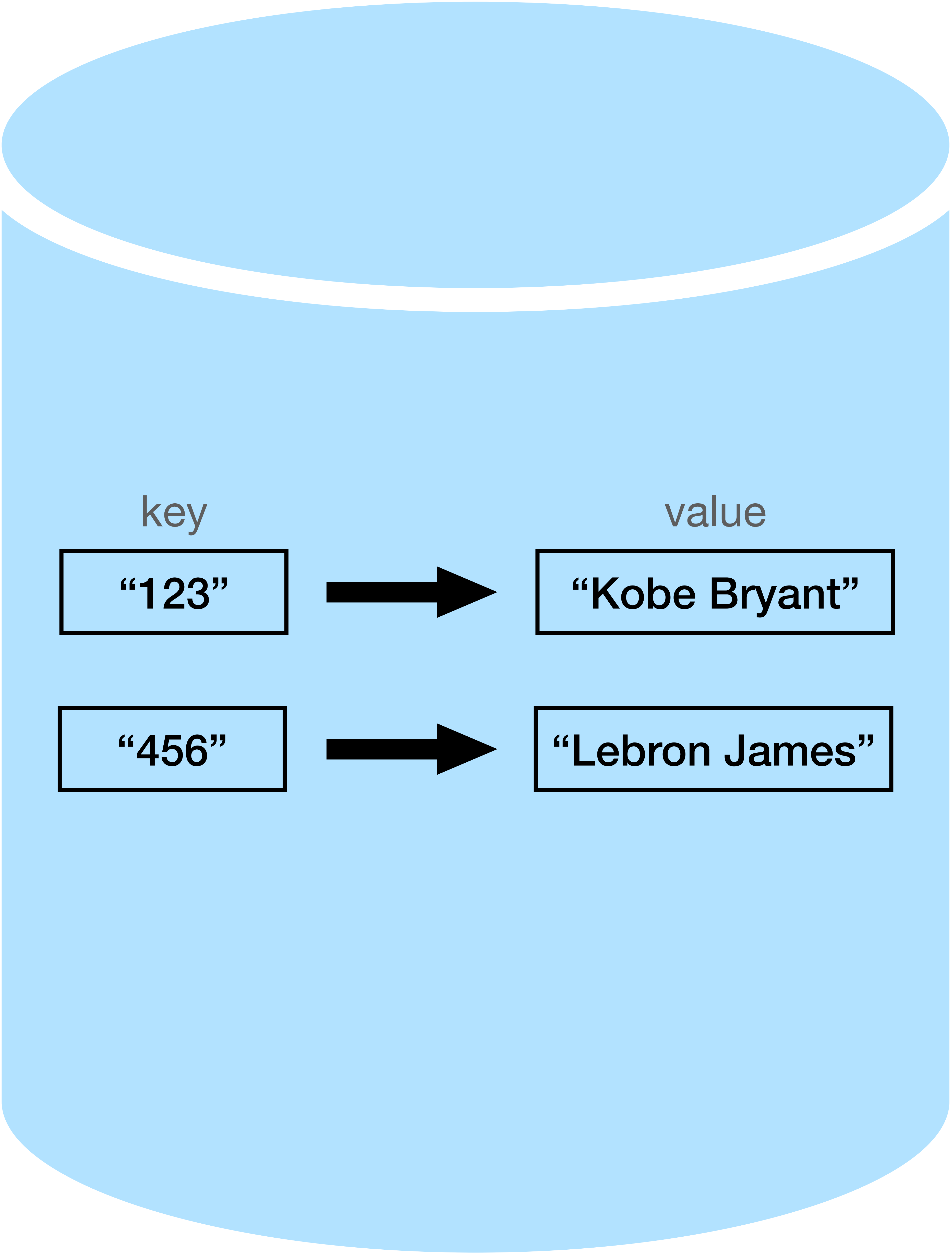
by their internal data model / API

- Key Value
- Document
- Graph
- **Wide column**

NoSQL - Key Value

- "Simplest" NoSQL database
- Hash table / dictionary
  key-value pairs
- CRUD API

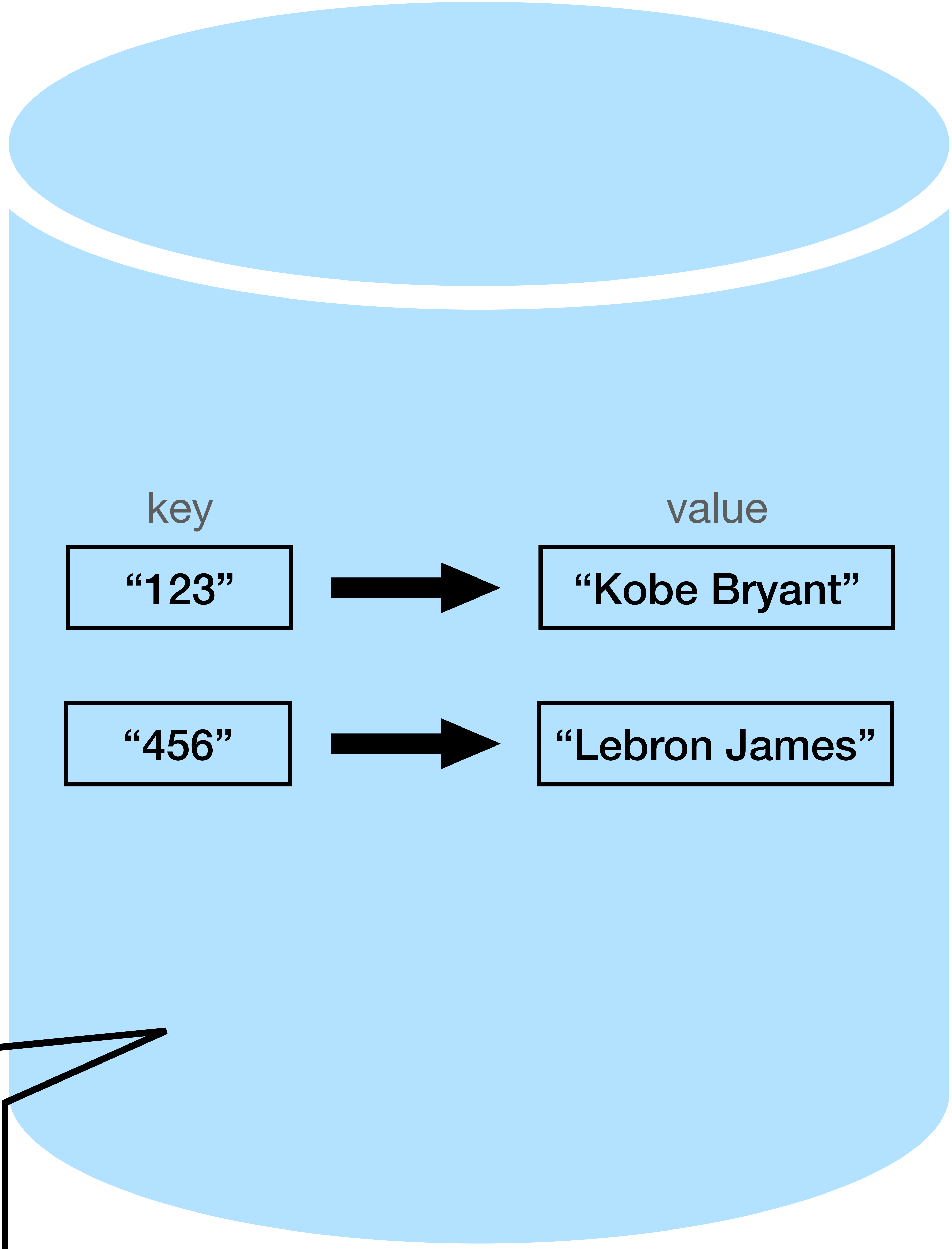
```
create(key/value)
read(key)
update(key/value)
delete(key)
```



NoSQL - Key Value

- “Simplest” NoSQL database
- Hash table / dictionary
  key-value pairs
- CRUD API

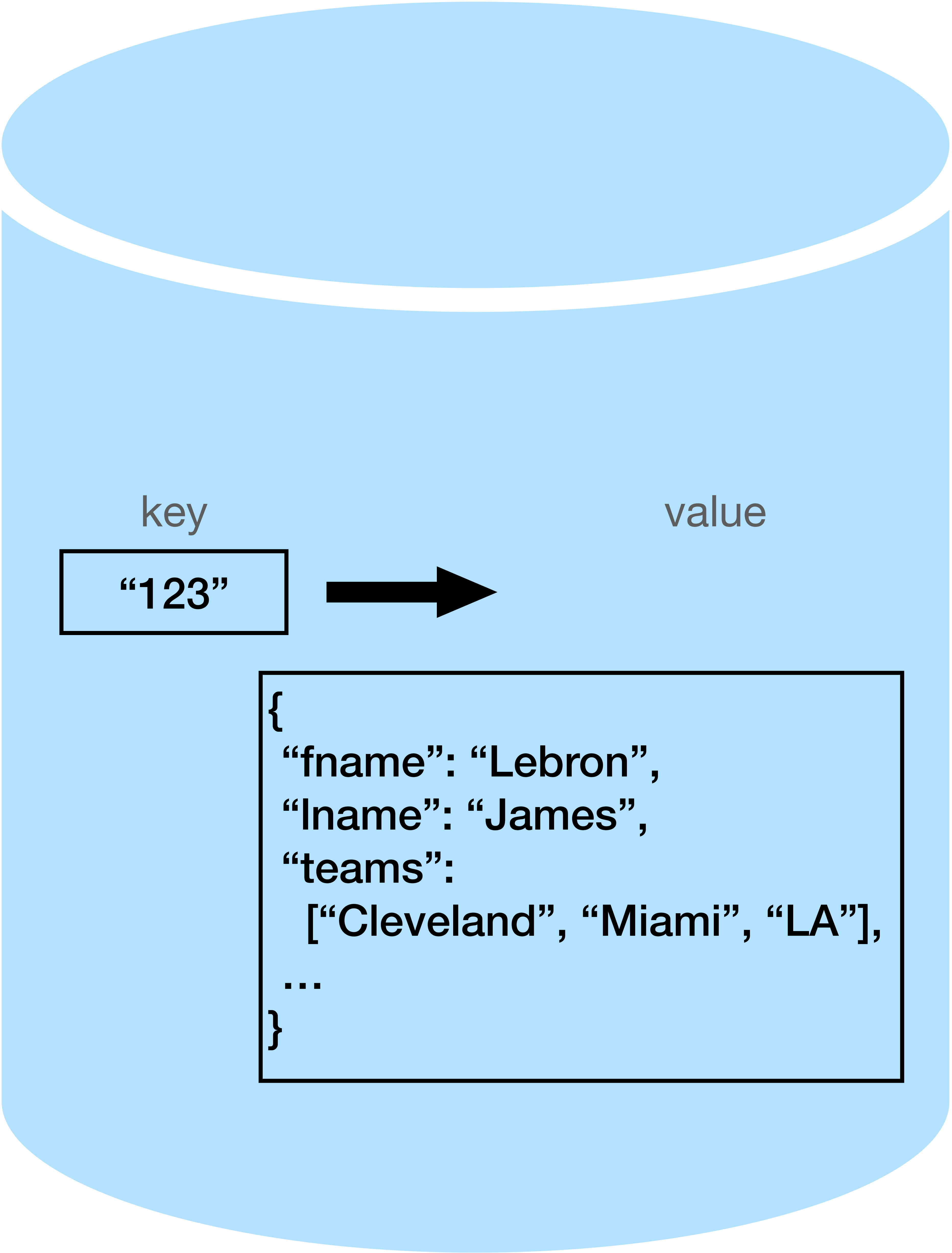
```
create(key/value)
read(key)
update(key/value)
delete(key)
```



popular systems
dynamo, levelDB, redis, memcached

NoSQL - Document

- Scheme less “document”
  accessible by unique key
- Document encoding
  json / xml / yaml / bson
- Stores all information of a given object in a single place
- CRUD API
  but not only
  (find, sort…)



key

value

“123”

```
{
 “fname”: “Lebron”,
 “lname”: “James”,
 “teams”:
   [“Cleveland”, “Miami”, “LA”],
 …
}
```

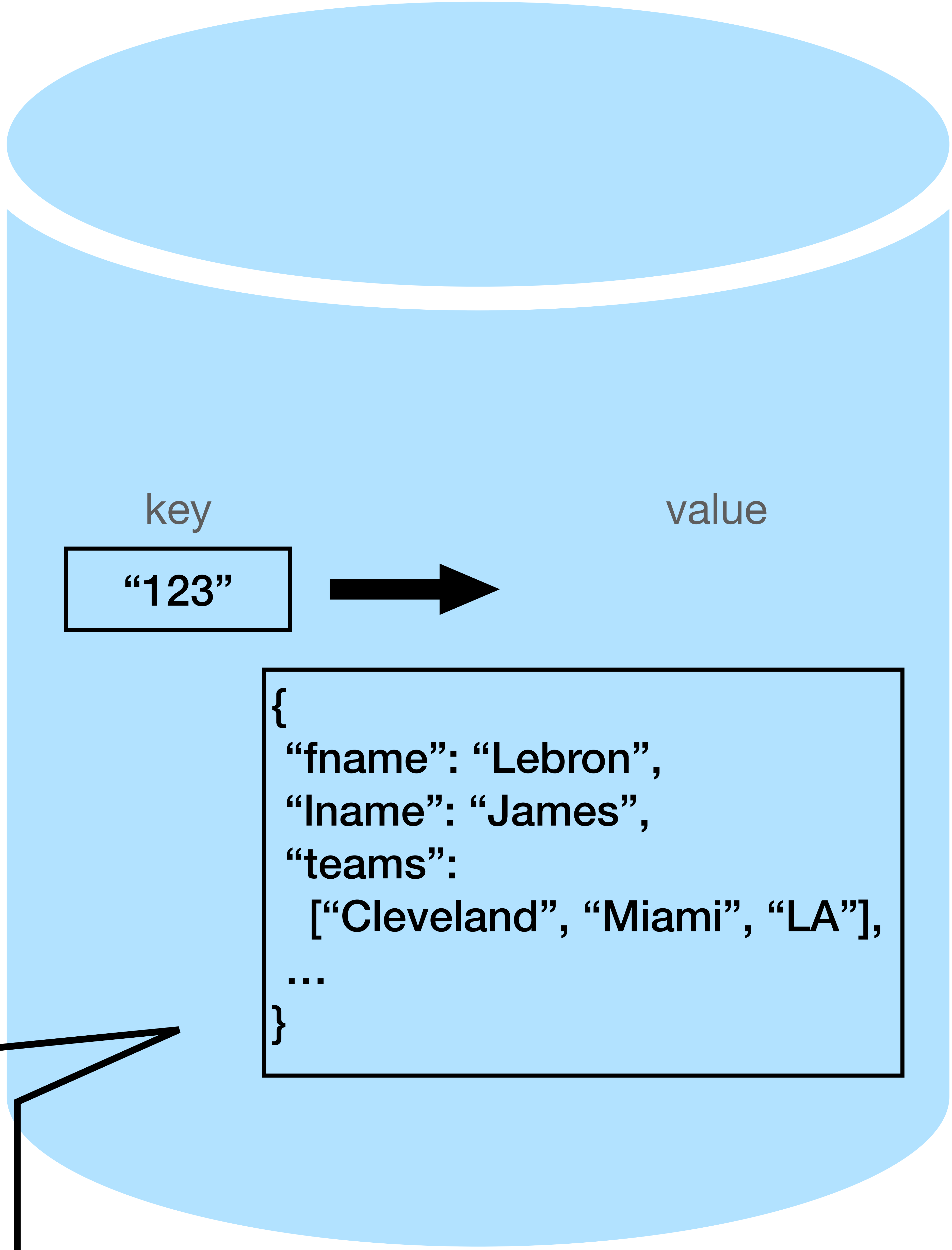
NoSQL - Document

- Scheme less “document”
  accessible by unique key

- Document encoding
  json / xml / yaml / bson

- Stores all information of a given object in a single place

- CRUD API
  but not only
  (find, sort…)

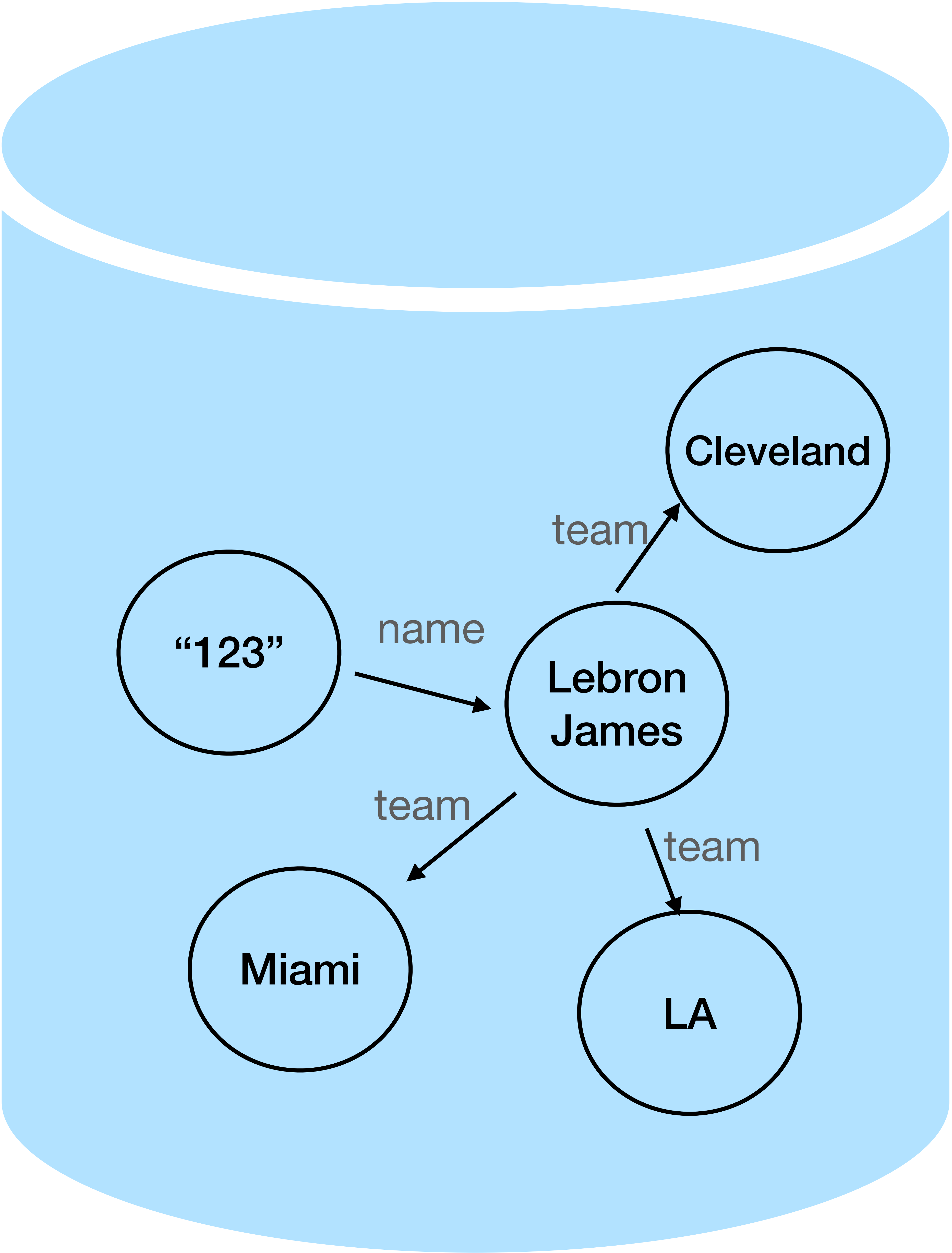


```
{
 “fname”: “Lebron”,
 “lname”: “James”,
 “teams”:
   [“Cleveland”, “Miami”, “LA”],
 …
}
```

popular systems
mongoDB, CouchDB

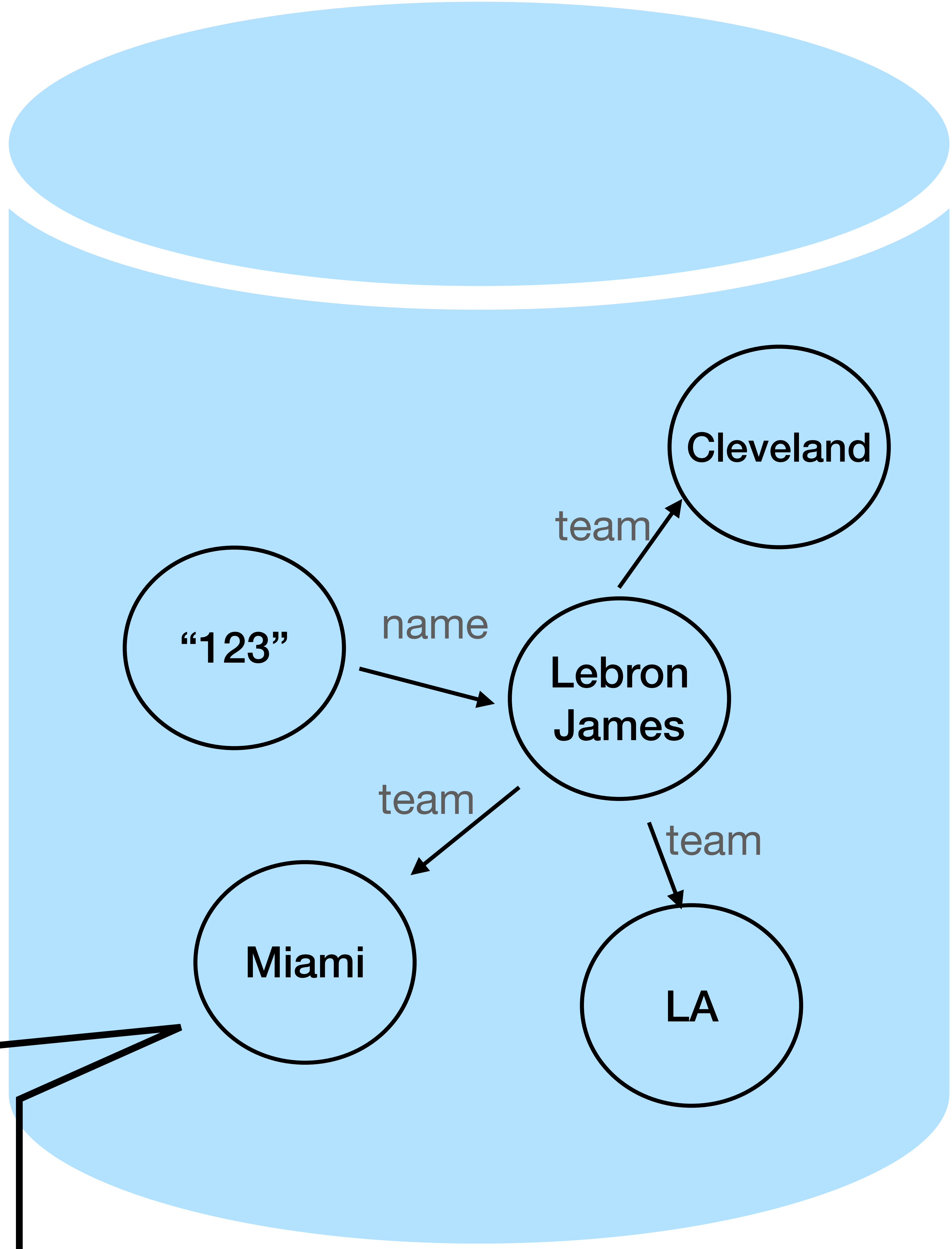
NoSQL - Graph

- Based on graph theory
  nodes, edges, properties, weights
- ACID / Transactions
  not always
- Built in graph algorithms



NoSQL - Graph

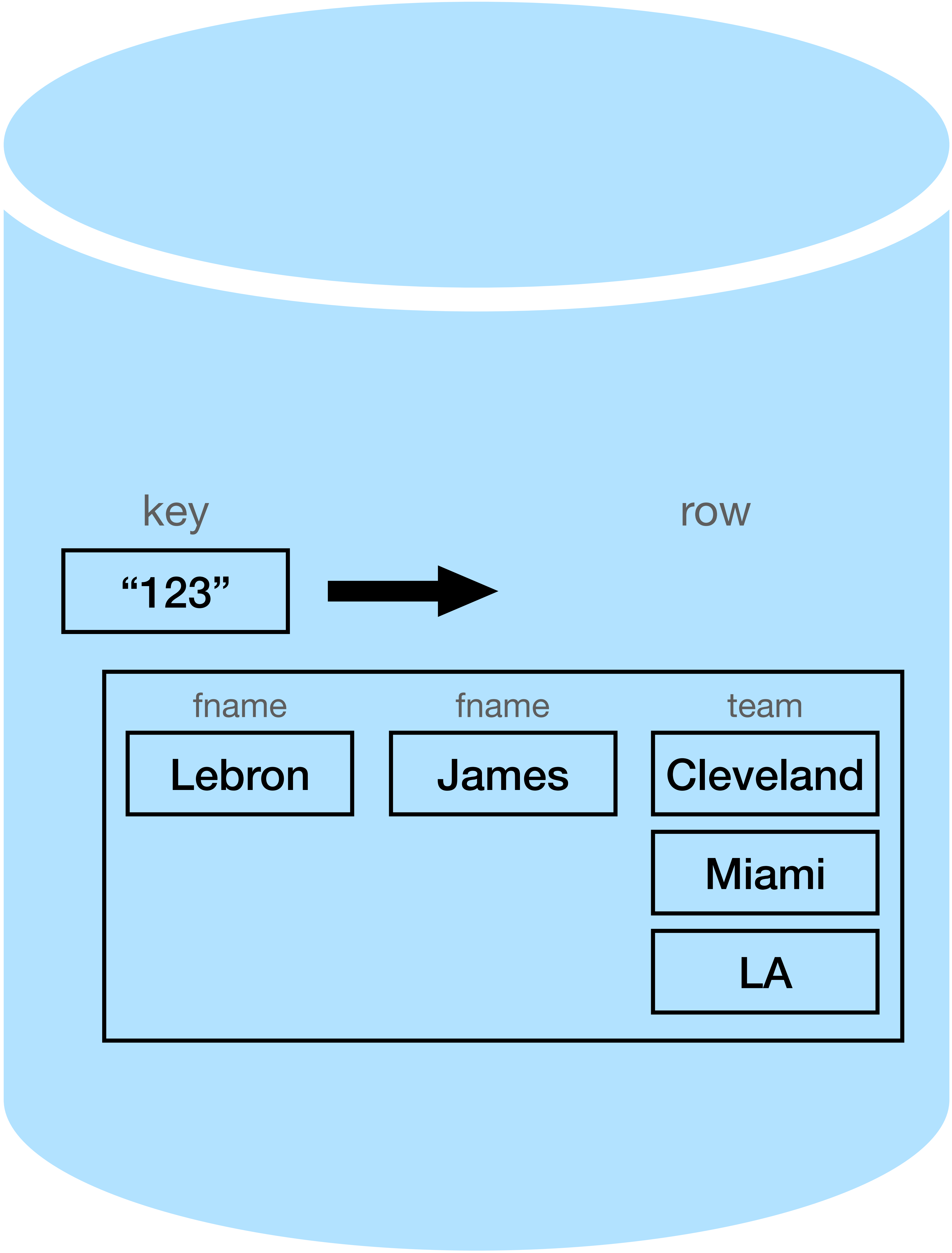
- Based on graph theory
  nodes, edges, properties, weights
- ACID / Transactions
  not always
- Built in graph algorithms



popular systems
neo4j, DataStax enterprise graph

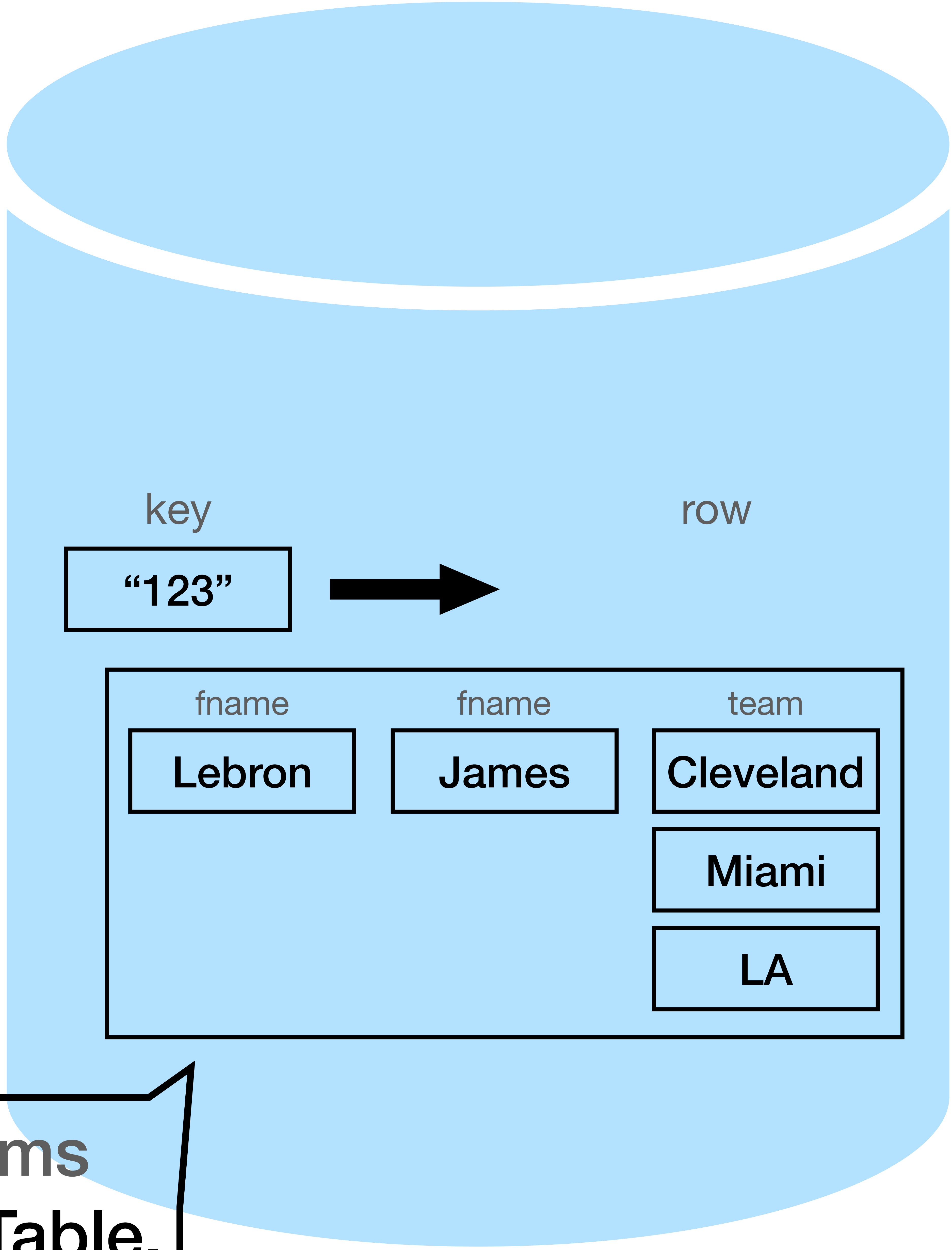
NoSQL - Wide column

- Tables, rows, columns
  can vary from row to row within a table
- Can be viewed as
  2-dimensional key-value store
- Custom API
  with some SQL sometimes (CQL)
- NOT a columnar database
  but can be sometimes



NoSQL - Wide column

- Tables, rows, columns
  can vary from row to row within a table
- Can be viewed as
  2-dimensional key-value store
- Custom API
  with some SQL sometimes (CQL)
- NOT a columnar database
  but can be sometimes



key

row

“123”

fname

fname

team

Lebron

James

Cleveland

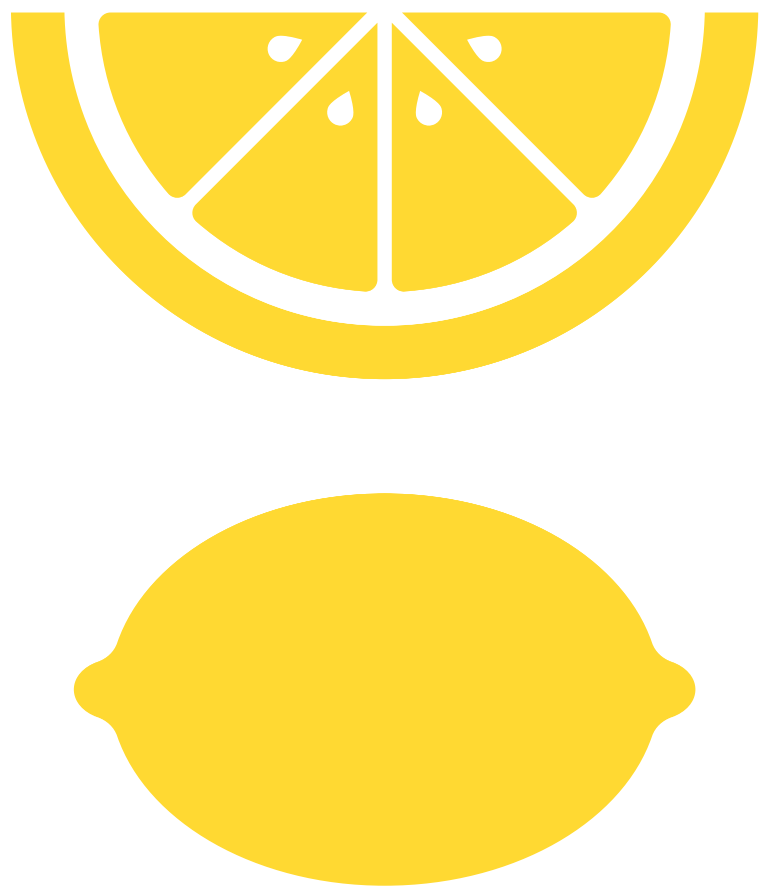
Miami

LA

popular systems
Cassandra, BigTable, DynamoDB, HBase

NoSQL - Wide column

- A lot more on these systems
- Stay tuned :)



ACID vs BASE

Basically **A**vailable
Soft state
Eventual consistency

ACID (reminder)

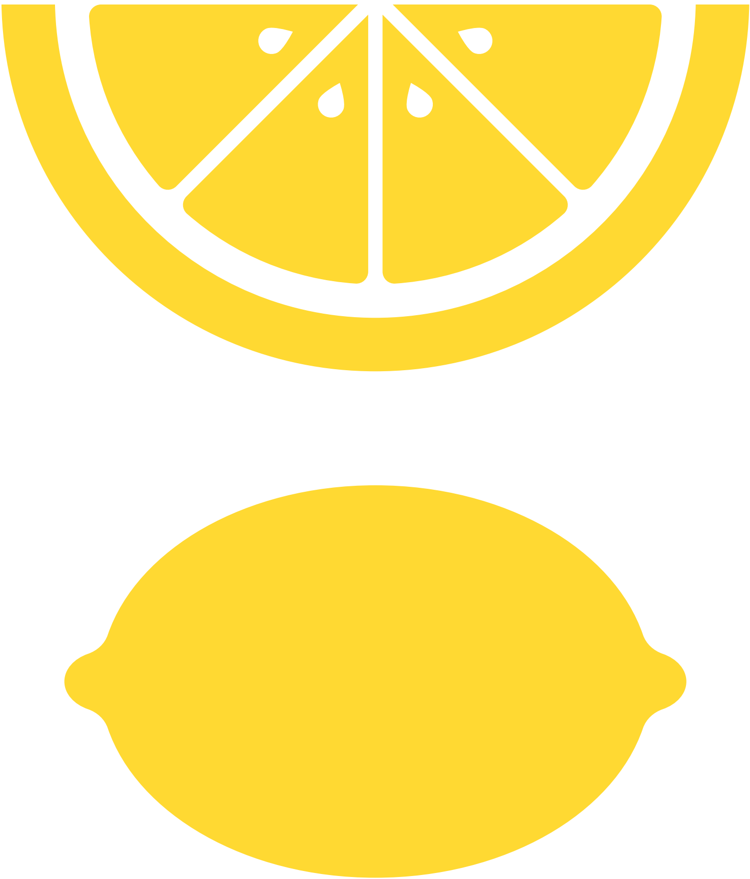
- **A**tomicity
- **C**onsistency
  correctness / referential integrity (foreign key) - NOT like in CAP
- **I**solation
- **D**urability

BASE

- **B**asically **A**vailable
  There will be a response even if node fails (response=fail)

- **S**oft state
  State can change even if no read/write are performed
  (the system is aiming towards consistent)

- **E**ventual consistency
  reads may be inconsistent, but over time will be consistent

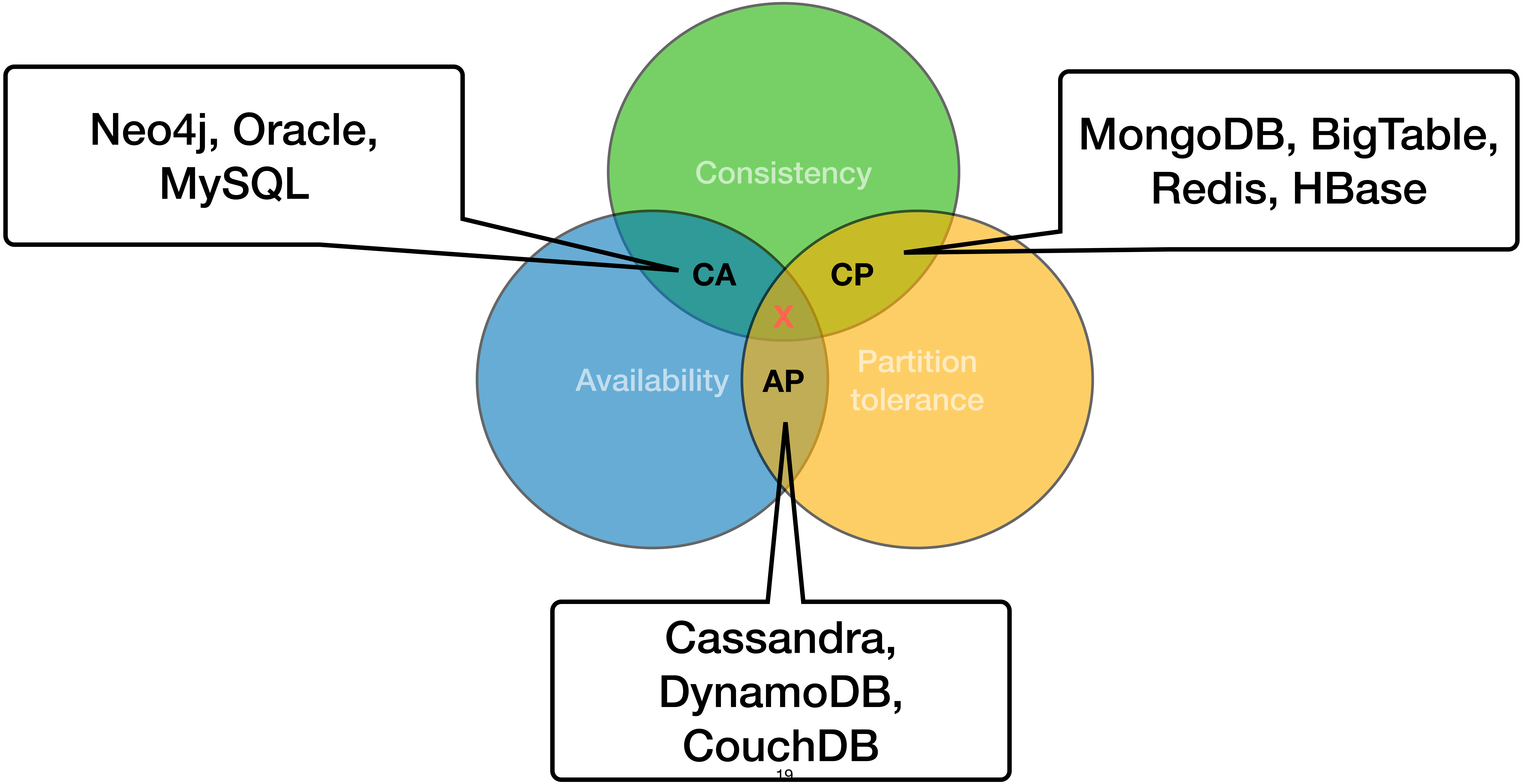


ACID vs BASE

Most relational DBs
(Oracle, MySQL…)

Most NoSQL systems
(C*, BigTable, MongoDB,…)

NoSQL and CAP



Neo4j, Oracle, MySQL

MongoDB, BigTable, Redis, HBase

Cassandra, DynamoDB, CouchDB