More NoSQL Databases

Gregory S. DeLozier, Ph.D.
Kent State University

Document Stores

```
{
     first_name: “Paul”,
     surname: “Miller”,
     city: “London”,
     location: [45.123,47.232],
     cars: [
        { model: “Bentley”,
          year: 1973,
          value: 100000, ….},
        { model: “Rolls Royce”,
          year: 1965,
          value: 330000, ….},
     ]
}
```

Redis

Redis

* In-memory store
  Key-value store
  Messaging database, pub-sub, synchronization

* Backed up to disk
* Eventual consistency

* http://www.slideshare.net/itamarhaber/redis-use-patterns-devcontlv-june-2014

* http://redis.io/
* http://try.redis.io/

Redis - Demo

* Create a Codio machine
* Get the Redis source
* Compile the source
* Try it out

* https://redislabs.com

Redis - Python

* https://pypi.python.org/pypi/redis

* pip install redis

```
>>> import redis
>>> r = redis.StrictRedis(host='localhost', port=6379, db=0)
>>> r.set('foo', 'bar')
True
>>> r.get('foo')
'bar'
```

Redis - Python / Lab Demo

...try this out

CouchDB

CouchDB

* http://couchdb.apache.org/

* General purpose database
* Fairly easy to use

* Complex to build and install

* Built-in replication
* http://docs.couchdb.org/en/2.0.0/intro/why.html

Eventual Consistency - CAP

* http://guide.couchdb.org/draft/consistency.html

Consistency

All database clients see the same data, even with concurrent updates.

Availability

All database clients are able to access some version of the data.

Partition tolerance

The database can be split over multiple servers.

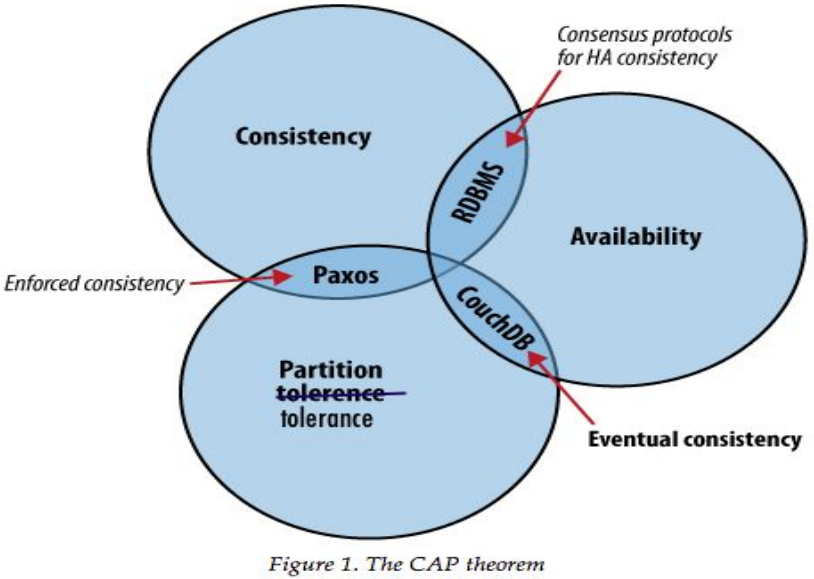

Figure 1. The CAP theorem

CAP Theorem

* https://en.wikipedia.org/wiki/CAP_theorem

It is impossible to provide all three of:
- Consistency
- Availability
- Partition tolerance

Partitioned system must choose "consistent" or "available."
ACID => prioritize consistency
BASE (basically available, soft state, eventually)
=> prioritize availability

CouchBase Queries

'SQL for documents'

```
{
  "email":"testme@gmail.com",
  "friends":[
            {"name":"rick"},
            {"name":"cate"}
           ]
}
```

Like Query

```
SELECT * FROM `bucket` WHERE LIKE "%@gmail.com";
```

Array Query

```
SELECT * FROM `bucket` WHERE ANY x IN friends SATISFIES x.name = "cate" END;
```

CouchDB vs Cloudbase

	Couchbase Server	Apache CouchDB
Data models	Document, Key-Value	Document
Storage	Append-only B-Tree	Append-only B-Tree
Consistency	Strong	Eventual
Topology	Distributed	Replicated
Replication	Master-Master	Master-Master
Automatic failover	Yes	No
Integrated cache	Yes	No
Memcached compatible	Yes	No
Locking	Optimistic & Pessimistic	Optimistic with MVCC
MapReduce (Views)	Yes	Yes
Query language	Yes, N1QL (SQL for JSON)	No
Secondary indexes	Yes	Yes
Notifications	Yes, Database Change Protocol	Yes, Changes Feeds

http://developer.couchbase.com/documentation/server/current/introduction/intro.html

CouchBase Python SDK

https://github.com/couchbase-guides/python-sdk

Demo time...

Codernity

* Pure Python, NoSQL Document Store, fast
* http://labs.codernity.com/codernitydb/

PickleDB

* Pure Python, simple, easy to work with.
* https://pythonhosted.org/pickleDB/