

# Scaling Your App with NoSQL



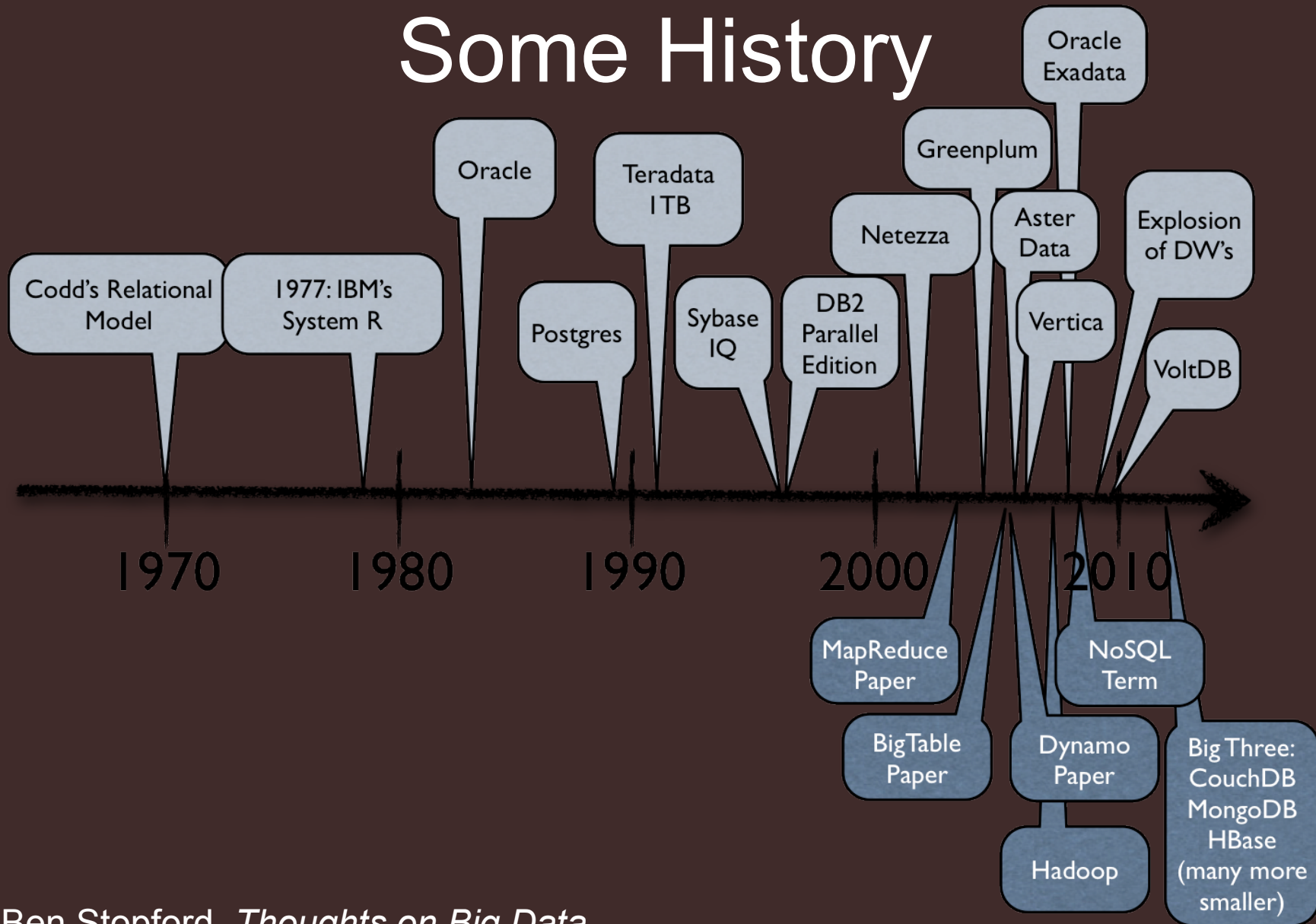
Jeremy Mikola

[jmikola.net](http://jmikola.net)

# 10gen | the MongoDB company

- Company behind MongoDB
- Provides support, training, and consulting
- Actively involved in the community
  - Mailing list, IRC, and StackOverflow
  - Conferences and local user groups
- Offices: NYC, Palo Alto, London, Dublin, Sidney
- Hiring at [10gen.com/careers](http://10gen.com/careers)

# Some History



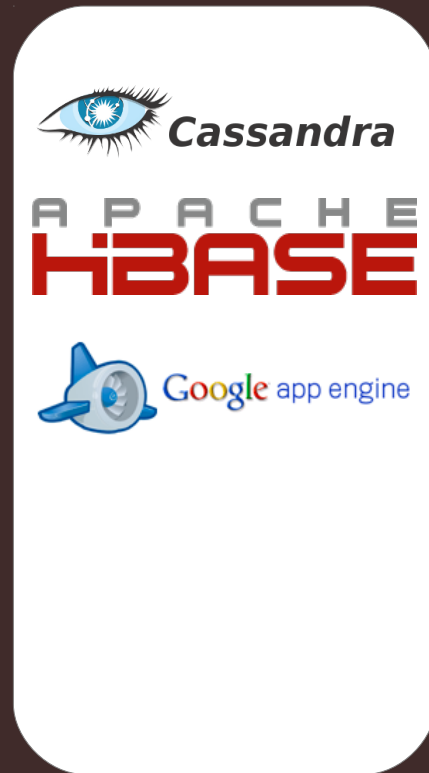
Ben Stopford, *Thoughts on Big Data*

<http://www.benstopford.com/2012/06/30/thoughts-on-big-data-technologies-part-1/>

# What is NoSQL?



Key/Value



BigTable



Document



Graph

# Key/Value Stores

- Maps arbitrary keys to values
- No knowledge of the value's format
- Completely schema-less
- Implementations
  - Eventually consistent, hierarchal, ordered, in-RAM
- Operations
  - Get, set and delete values by key

# BigTable

- Sparse, distributed data storage
- Multi-dimensional, sorted map
- Indexed by row/column keys and timestamp
- Data processing
  - MapReduce
  - Bloom filters

# Graph Stores

- Nodes are connected by edges
- Index-free adjacency
- Annotate nodes and edges with properties
- Operations
  - Create nodes and edges, assign properties
  - Lookup nodes and edges by indexable properties
  - Query by algorithmic graph traversals

# Document Stores

- Documents have a unique ID and some fields
- Organized by collections, tags, metadata, etc.
- Formats such as XML, JSON, BSON
- Structure may vary by document (schema-less)
- Operations
  - Query by namespace, ID or field values
  - Insert new documents or update existing fields



# MongoDB Philosophy

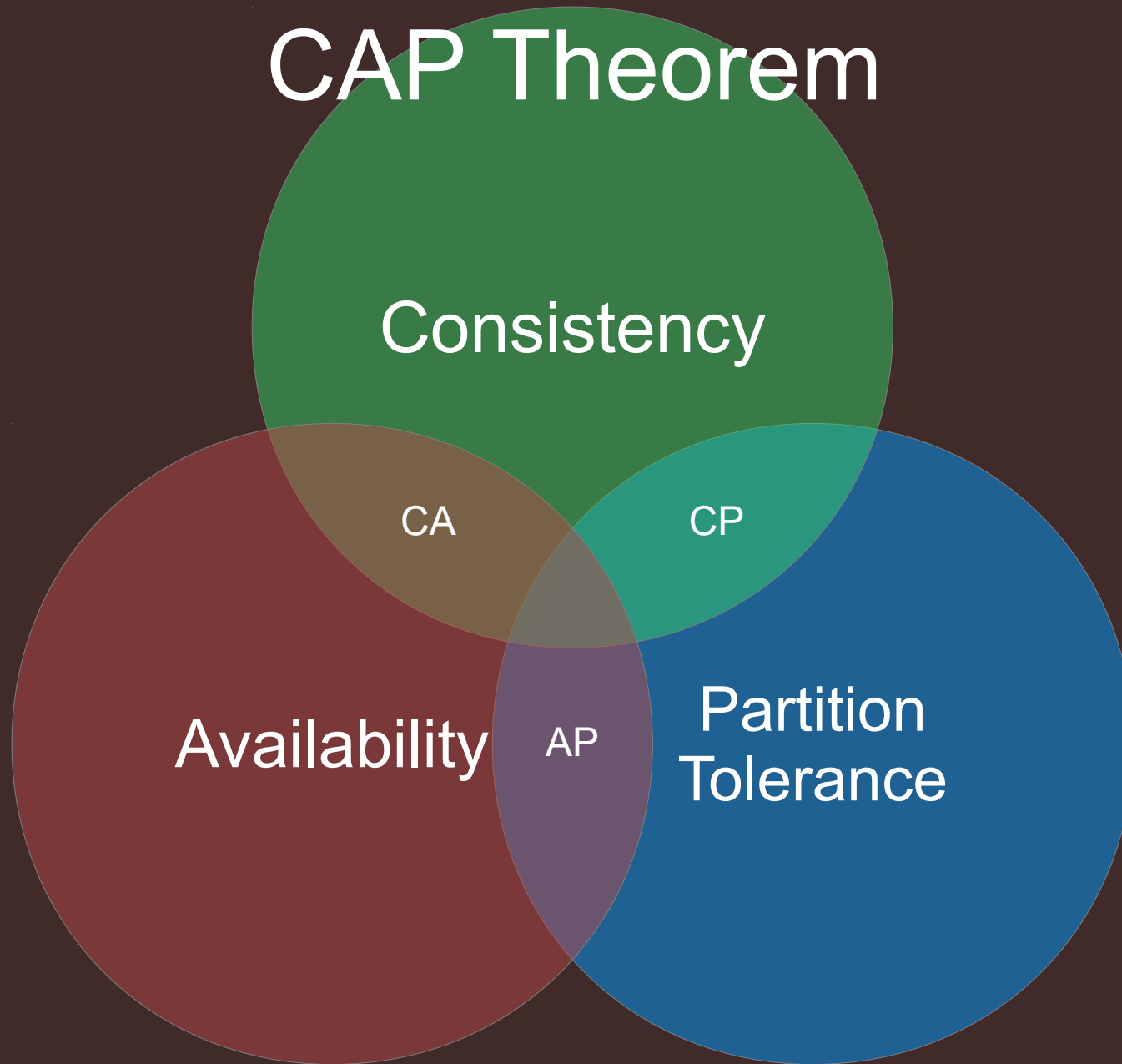
- Document data models good
- Non-relational model allows horizontal scaling
- Keep functionality whenever possible
- Minimize the learning curve
  - Easy to setup and deploy anywhere
  - JavaScript and JSON are ubiquitous
  - Automate sharding and replication

# MongoDB Under the Hood

- Server written in C++
- Server-side code execution with JavaScript
- Data storage and wire protocol use **BSON**
- Reliance on memory-mapped files
- B-tree and geospatial indexes

# What are the challenges and trade-offs?

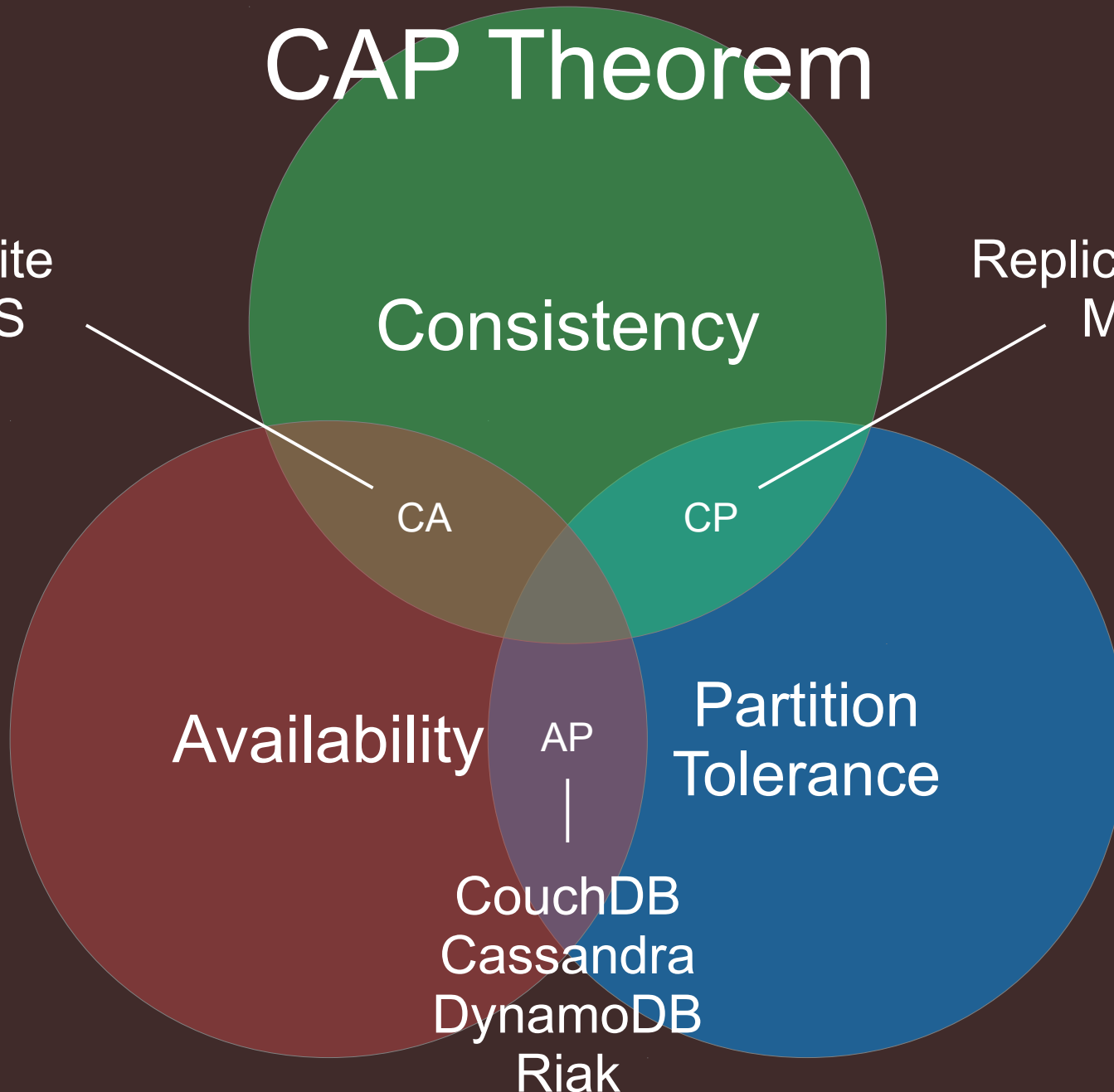
# CAP Theorem



# CAP Theorem

Single-site  
RDBMS

Replicated RDBMS  
MongoDB  
HBase  
Redis





In partitioned databases, trading some consistency for availability can lead to dramatic improvements in scalability.

Dan Pritchett, *BASE: An ACID Alternative*  
<http://queue.acm.org/detail.cfm?id=1394128>

# ACID vs. BASE

- Atomicity
- Consistency
- Isolation
- Durability
- Basically Available
- Soft state
- Eventual consistency

# Consistency Models

## Eventual Consistency

### Monotonic Read Consistency

#### MRC + RYOW

#### Immediate Consistency

#### Strong Consistency (single-entity)

#### Transactions (multi-entity)

Read-your-own Writes

<http://blog.mongodb.org/post/475279604/on-distributed-consistency-part-1>

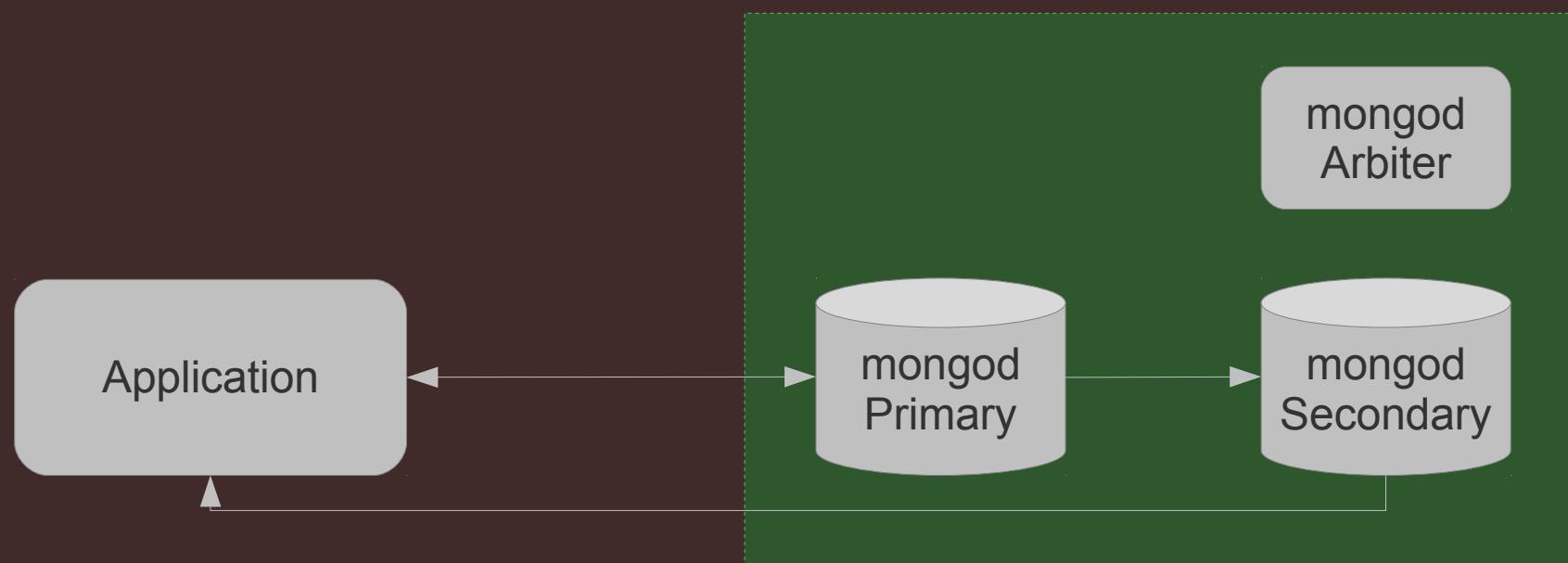


# Strong Consistency with MongoDB

- Writes occur in order
- Read-your-own writes
- Replication via idempotent operations
- Control replication per write if desired
- Atomic operations within a single document
- Durability with journaling

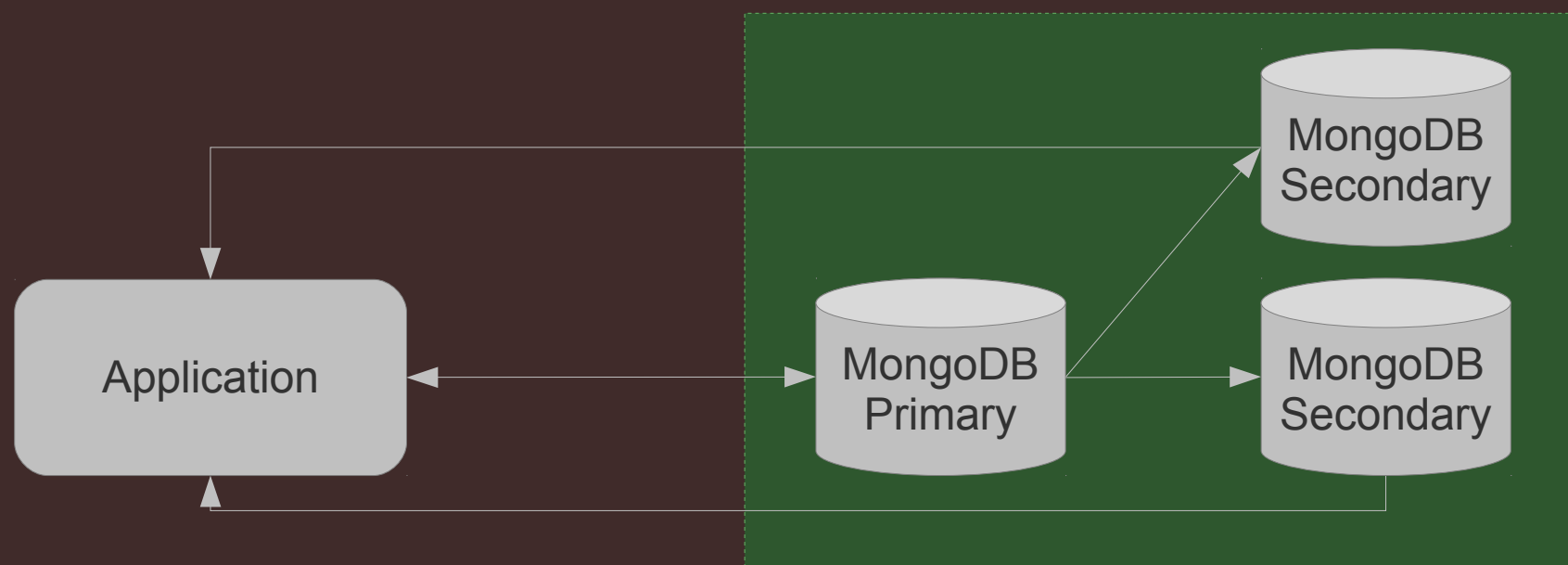
# Replica Sets

- Primary, secondary and arbiter
- Optionally direct read queries to secondary
- Automatic failover

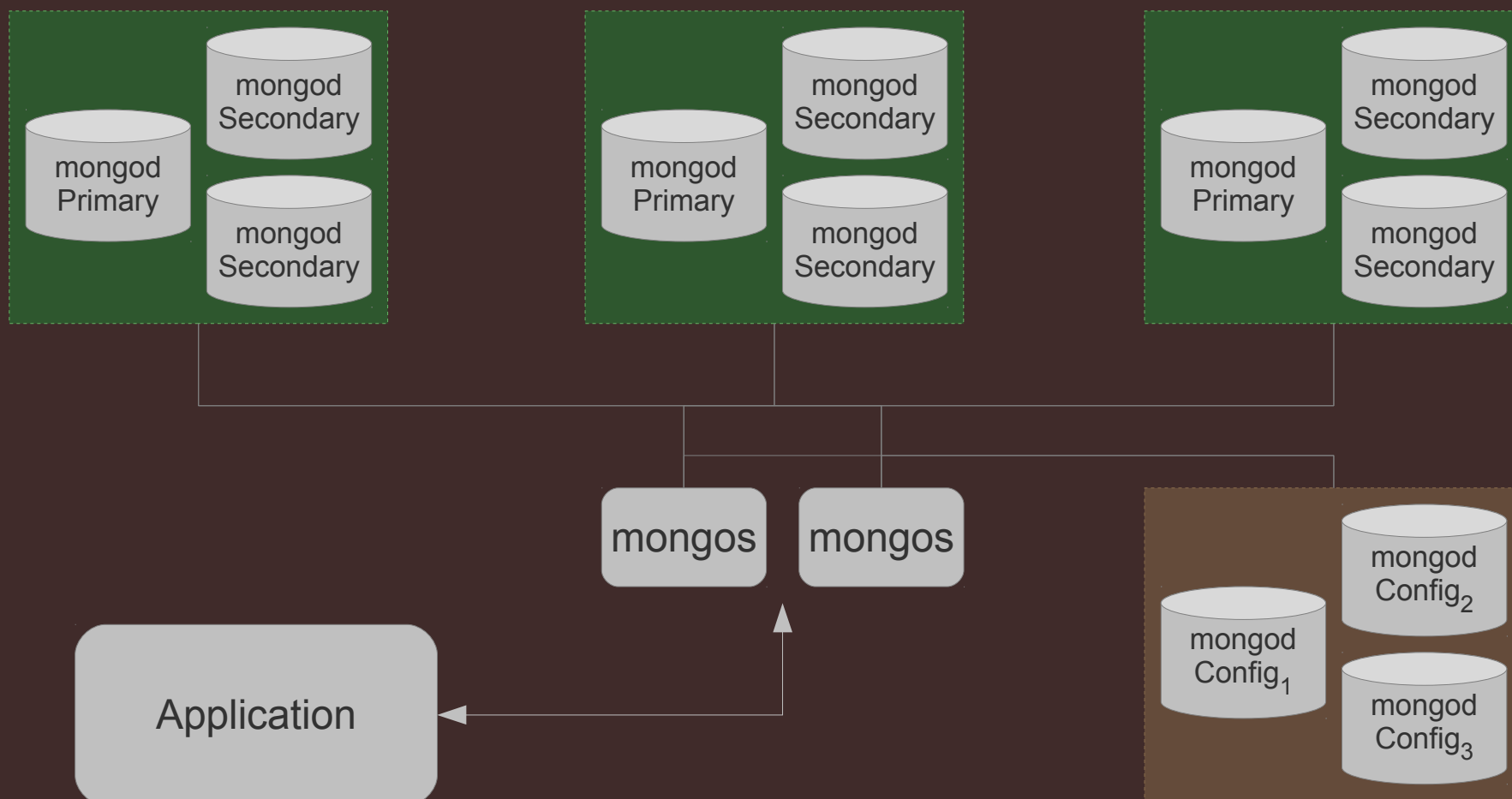


# Replica Sets

- Primary with two secondaries
- Arbiter unnecessary for odd number of nodes



# Sharding



# Sharding

- **mongos** processes
  - Route queries to shards and merges results
  - Coordinates balancing amongst shards
  - Lightweight with no persistent state
- Config servers
  - Launched with **mongod --configsvr**
  - Store cluster metadata (shard/chunk locations)
  - Proprietary replication model

Sharding is the tool for scaling a system.

Replication is the tool for data safety, high availability, and disaster recovery.

<http://www.mongodb.org/display/DOCS/Sharding+Introduction>

It's not just about servers.

# Scaling Development

- Data format analogous to our domain model
  - Embedded documents
  - Arrays (of scalars, documents, other arrays)
- Schema agility for ever-changing requirements
- Useful features
  - Aggregation framework
  - Built-in MapReduce, Hadoop integration
  - Geo, GridFS, capped and TTL collections



# Working with Data

```
$ mongo
MongoDB shell version: 2.2.0-rc0
connecting to: test


> db.events.insert({name:"CloudCamp", tags: ["unconference", "tech"]})
> db.events.findOne()
{
  "_id" : ObjectId("50199154647dc9a55063bd3f"),
  "name" : "CloudCamp",
  "tags" : [
    "unconference",
    "tech"
  ]
}
> db.events.update({name:"CloudCamp"}, {$set: {name: "CloudCamp Newark"}})
> db.events.findOne({tags: "unconference"}, {name: 1})
{
  "_id" : ObjectId("50199154647dc9a55063bd3f"),
  "name" : "CloudCamp Newark"
}
```

# Case Study: Craigslist

- 1.5 million new classified ads posted per day
- MySQL clusters
  - 100 million posts in live database
  - 2 billion posts in archive database
- Schema changes
  - Migrating the archive DB could take months
  - Meanwhile, live DB fills with archive-ready data

# Case Study: Craigslist

- Utilize MongoDB for archive storage
- Average document size is 2KB
- Designed for 5 billion posts (10TB of data)
- High scalability and availability
  - New shards added without downtime
  - Automatic failover with replica sets



We can put data into MongoDB faster than we can get it out of MySQL during the migration.

Jeremy Zawodny, software engineer at Craigslist and author of *High Performance MySQL*

<http://blog.mongodb.org/post/5545198613/mongodb-live-at-craigslist>

# Case Study: Shutterfly

- 20TB of photo metadata in Oracle
- Complex legacy infrastructure
  - Vertically partitioned data by function
  - Home-grown key/value store
- High licensing and hardware costs

# Case Study: Shutterfly

- MongoDB offered a more natural data model
- Performance improvement of 900%
- Replica sets met demand for high uptime
- Costs cut by 500% (commodity hardware)

# Case Study: OpenSky

- E-commerce app built atop Magento platform
- Multiple verticals (clothing, food, home, etc.)
- MySQL data model was highly normalized
- Product attributes were not performant

# Case Study: OpenSky

- Integrated MongoDB alongside MySQL
- Documents greatly simplified data modeling
  - Product attributes
  - Configurable products, bundles
  - Customer address book
- Purchases utilized MySQL transactions
- Denormalized order history kept in MySQL



# Additional Case Studies



[10gen.com/customers](http://10gen.com/customers)

# Try It Out

- Binaries for Linux, OS X, Windows and Solaris
- Supported drivers for over a dozen languages
- Community-supported drivers for many more
- Browser-based demo at [mongodb.org](http://mongodb.org)

# Questions?