NoSQL with MongoDB

HOW TO CREATE A MONGODB APP USING C#

NATE BUWALDA

NEBRASKA CODE CAMP MARCH 2014

Agenda

- Intro to MongoDB
- Mongo Architectures
- Using the C# Mongo Driver

Intro to MongoDB

BIG DATA DOCUMENT STORE

Intro to MongoDB

Big Data, Big Problems

Data needs grow bigger and bigger

Traditional data store solutions are expensive

- Hard to add nodes
- Hard to keep large data durable

Mongo solves this by elastically scaling both up and out

Intro to MongoDB

What is a Mongo?

Short for HUMONGOUS

JSON based document data store

Written in C++, supports multiple OSes and plaforms

Supports indexing (normal and 2d-geospatial)

Uses a flexible multi-node configuration to support sharding and replication

Intro to MongoDB

What are Mongo's Shortcomings?

No built in support for relations

Long unit of work transactions are not supported

Does not offer fine grained user control

Not necessarily durable

All data is eventually consistent

Intro to MongoDB

Storage Basics

Each cluster can have multiple databases

- admin database controls administrative functions

Each database can have multiple collections of documents

Authentication and replication are controlled at the database level

Sharding and indexing are controlled at the collection level

Mongo architecture

Mongo Architecture

Getting started to MongoDB

Go to mongodb.org

Download the appropriate binaries for your platform

Extract the zip or tar to where you want Mongo ‘installed’

Create a directory for where the data should be saved on disk

Start the node or cluster

Mongo Architecture

Windows MongoDB Startup Script

cd c:\mongo\

start "Mongo DB 1" .\bin\mongod --port 27021 --dbpath C:\bench\mongo\data\db2 --logpath C:\bench\mongo\shard1.log --auth

Mongo Architecture

Types of Mongo Instances

Mongod

- Data storage node
- Can run 1 to many
- Can run in arbiter mode
- Can be the sole instance for development purposes

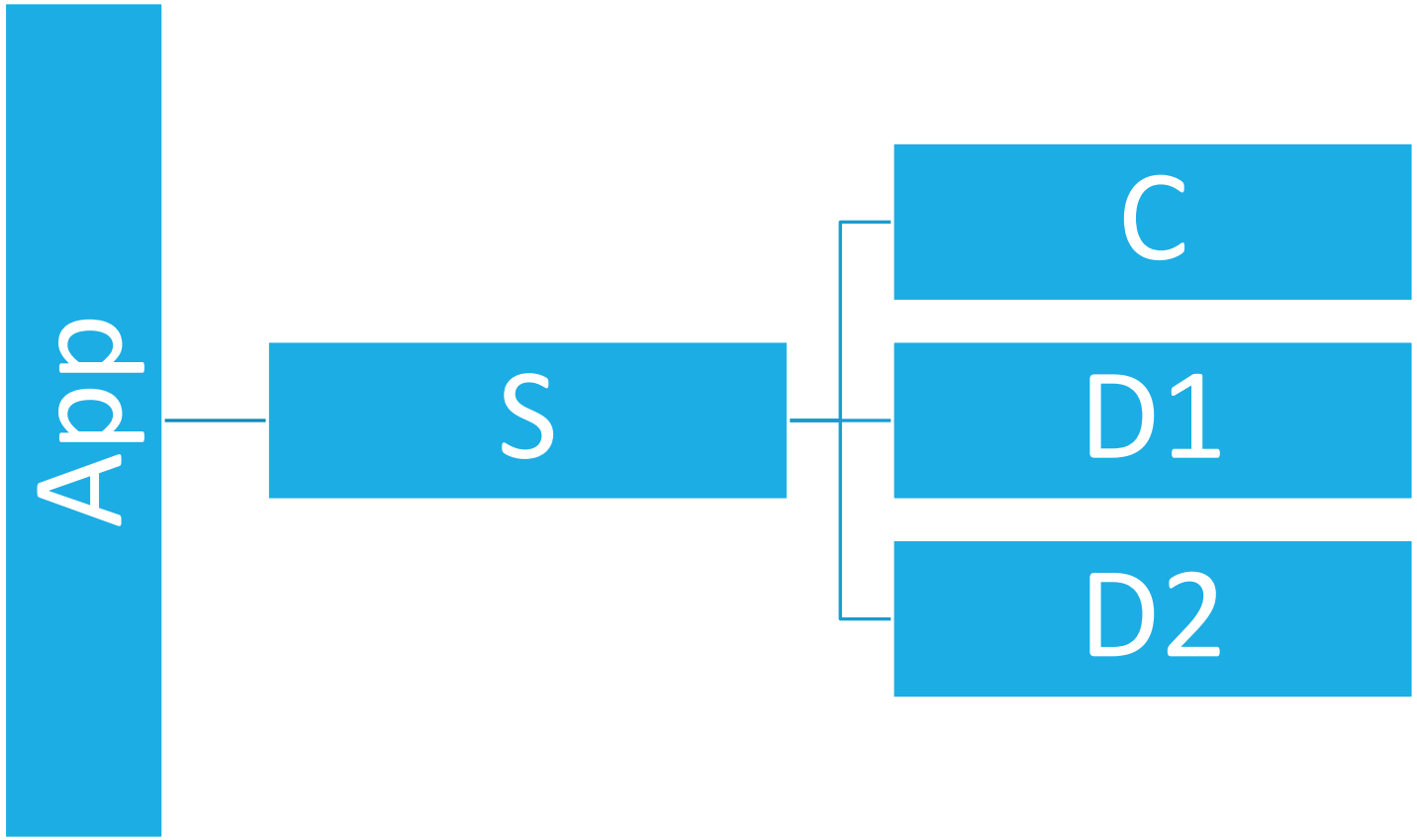
Mongos

- Shard routing node
- Need at least 1, should have 1 per application instance

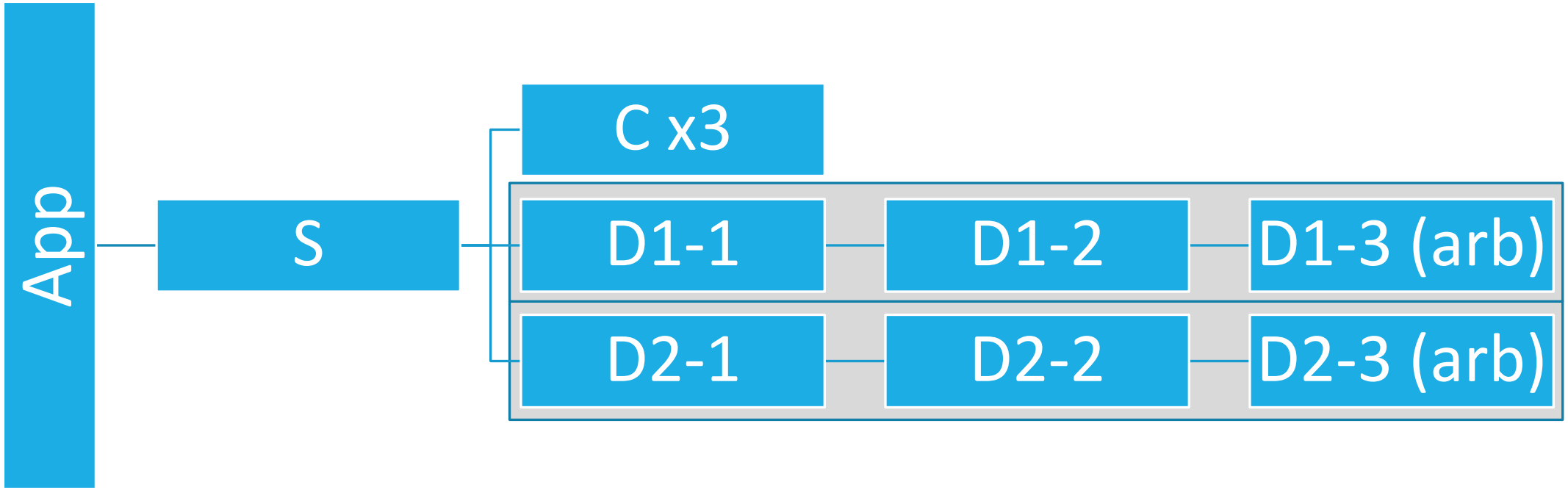
Mongoc

- Shard configuration node
- Must have exactly 1 or 3 instances

Development shard cluster



Production shard and replica cluster



Mongo with C#

USING THE OFFICIAL C# DRIVER

Mongo with C#

The Official Mongo C# Driver

Officially supported by MongoDB Inc (formerly known as 10gen)

Current version 1.8.3

Open source, available to clone on Github

Has a LINQ and non-LINQ implementation

Mongo with C#

Resources

Tutorial

http://docs.mongodb.org/ecosystem/tutorial/getting-started-with-csharp-driver/

Driver source

https://github.com/mongodb/mongo-csharp-driver

Google group

https://groups.google.com/forum/#!forum/mongodb-csharp

Mongo with C#

Creating a Connection

All functionality runs through the MongoClient class

Can use client settings or url parameters to modify the connection properties

Mongo with C#

Working With Documents

Documents can be represented in two ways

- BsonDocument
- Domain object with Mongo attributes

All documents need an id, Mongo inserts will assign one if you have not

- Sometimes it is worth assigning an ObjectId in your code prior to insertion

Bson ObjectId

- 4-byte timestamp
- 3-byte machine id
- 2-byte process id
- 3-byte counter

Mongo with C#

Sample Documents

BsonDocument class behaves like a key-value map of BsonElements

BsonElements consist of a key and a BsonValue

BsonValues are implicitly converted to and from most primative types

Mongo with C#

Document Serialization

You can create a custom serializer using the BsonClassMap

- Helpful for creating result objects that differ from the usual domain type
- Can add specific type handling

Generally easier to use the provided serializer with attributed classes

Mongo with C#

Querying for Documents

FindOne

- Returns the first document in the collection or the first document that matches the provided query
- Queries are constructed using a QueryDocument or the Query Builder

QueryBuilder

- Builds a query from composed together parts
- Preferred method of querying over writing QueryDocuments by hand

Find

- Returns a MongoCursor that implements an IEnumerable of documents that match the provided query

Mongo with C#

Inserting Documents

Insert is used to only insert a document that may not have an id key defined

Can insert multiple documents at once by using InsertBatch

Save can be used to insert a document with an id key defined

- If a document with the specified id exists, the existing document will be updated

WriteConcern controls how long the driver waits for a response from the insert command

- By default, the driver sets WriteConcern to 1

Mongo with C#

Querying with LINQ

Can perform LINQ queries by calling the AsQueryable method on the Mongo collection object

Many LINQ methods are supported

- Where
- Any
- Count
- First
- OrderBy

Some important ones are not

- GroupBy

Mongo with C#

Modifying Documents

Update can be used to modify existing keys or add new ones

FindAndModify is more atomic, but only works for one document at a time

Save is a more generic method than Update, but has a high overhead

- It will replace everything in document with the keys and values provided in the object passed to it

Documents are deleted by using Remove or RemoveAll

Mongo with C#

Complex Queries

MapReduce can be used to perform group by operations

- Results stored in a temporary collection
- Performed by sending JavaScript to the server to be executed on the server

Aggregation framework can accomplish the same task, but has less flexibility

- The aggregation query itself is composed of BsonDocuments

Mongo with C#

Tips and Tricks

Wrap the driver classes in interfaces to help with unit testing

Create a generic repository class for CRUD operations

Avoid working with more data than you need

The driver is not a good ORM

Closing remarks

QUESTIONS?

Contact me

natebuwalda@gmail.com

http://github.com/natebuwalda

@scotchnate

Thanks!