Overview

Contents

- MongoDB Releases
- Features
- Storage Engines

MongoDB Releases

- Naming
  - A.B.C
  - odd second numbers (B) for developer releases
  - even second numbers (B) for production releases
- Goals
  - 1 or 2 releases (A.B) per year
  - patches for some time (end of life)
  - support for
    - MongoDB: 18 months (end of support)
    - OpsManager: 12 months

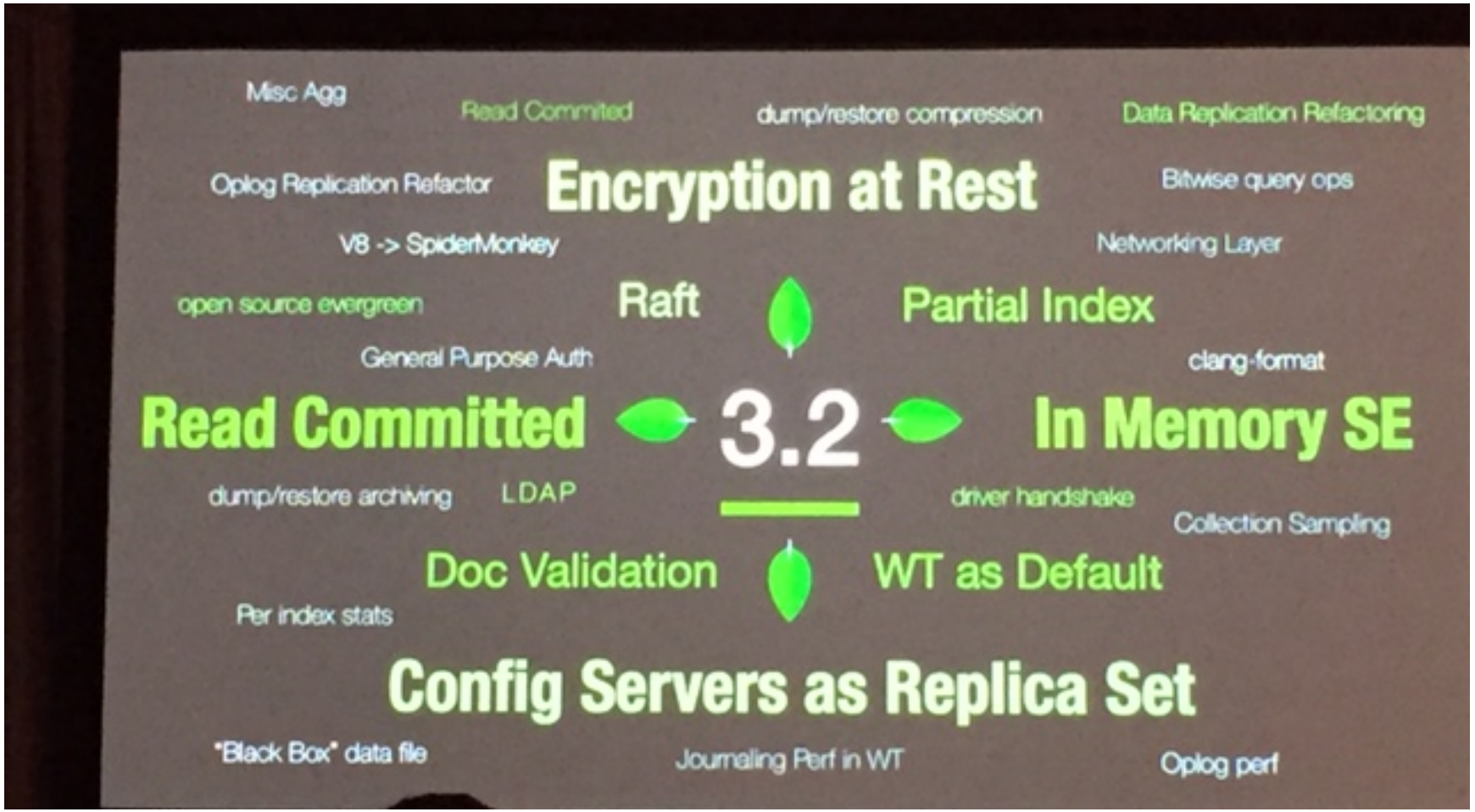
History of the MongoDB Product

- 2009 – 1.0
- 2010 – 1.2 master slave replication
- 2010 – 1.4 geo queries
- 2010 – 1.6 replica sets and sharding
- 2011 – 1.8 journaling (25 people)
- 2011 – 2.0 concurrency and fixes on everything above
- 2012 – 2.2 database level locking (100 people)
- 2013 – 2.4 text search and hashed sharding (325 people)
- 2014 – 2.6 security, auditing and new query planner (400 people)
- 2015 – 3.0 collection level locking in MMAPV1, document level locking with WiredTiger and OpsManager (MMS 1.6/OpsManager 1.8)
- 2015 – 3.2 WT as default, new storage engines (memory, encrypted), config servers as replica set (500 people) and OpsManager 2.0
- 2016 (Nov 1?) – 3.4, features list to be available internally in Feb.

MongoDB Upgrades

- We officially support:
  - MongoDB:
    - 2.4 (up to 03/31//2016)
    - 2.6 (up to 10/31/2016), sending notification on April 7, 2016
    - 3.0 and 3.2
  - OpsManager
    - 1.6 (up to 03/2016)
    - 1.8 and 2.0
- Some laggards on 1.8, 2.0 and 2.2
  - Product called “Extended Lifecycle Support Add-On”
- We often ask to upgrade
  - example: if running WT on 3.0, must use the latest (but not 3.0.10!)
- Complicated upgrades
  - Recommend the Consulting Package “Major Version Upgrade”

New features in 3.2



MongoDB 3.3/3.4

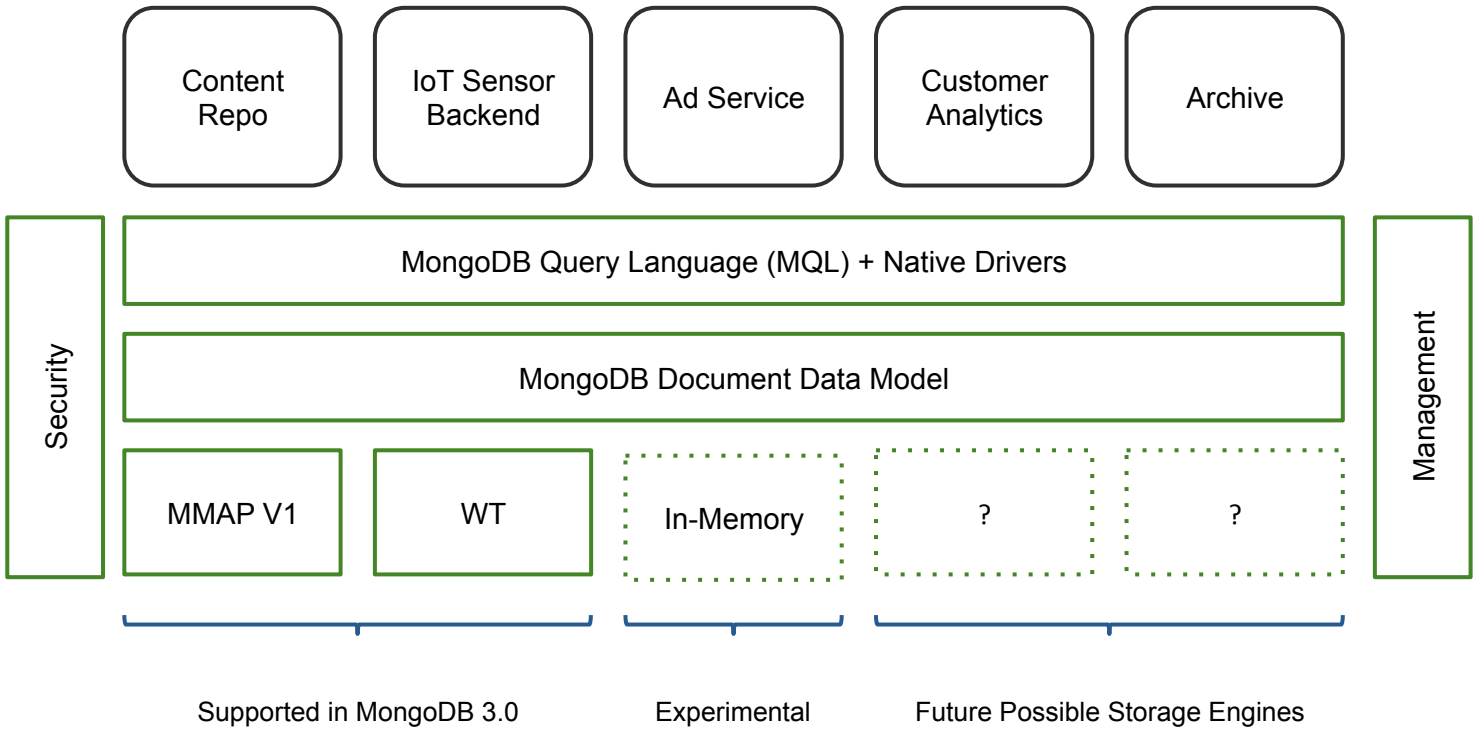
- Current plan (CONFIDENTIAL!!!) :
  - https://docs.google.com/presentation/d/1OVAKk-aZKVttoyWJ5u8zNDUWajk-21D5ob-13qF1Wg4/edit#slide=id.p
- Download the development build at:

  https://www.mongodb.com/customer-evaluation-downloads-development-versions

How our customers use MongoDB?

- Stats from Cloud Manager users (Feb 2016)
  - (usually, does not include largest deployments)
  - 15% use sharding
  - 1.5% use tag-aware sharding
  - Sharded cluster 3X the size of non-sharded ones

Architecture



Main Features

- Flexible Schema
- Storage Engines
- Replication
- Sharding
- API

Flexible Schema

BSON (bsonspec.org)

- Has everything that JSON has and some extra stuff:
  - Binary data, ObjectId, UTC datetime, Timestamp, regex…
- So what is a document, really?
  - A document is a 32 bit integer followed by an element list and a null byte
  - An element list is an element followed by another element list or zero bytes
  - An element is a single byte (type) followed by the field name and the datum, i.e. the value of the field…

BSON continued

element	::=	"\x01" e_name double	Floating point
	|	"\x02" e_name string	UTF-8 string
	|	"\x03" e_name document	Embedded document
	|	"\x04" e_name document	Array
	|	"\x05" e_name binary	Binary data
	|	"\x06" e_name	Undefined — *Deprecated*
	|	"\x07" e_name (byte*12)	ObjectId
	|	"\x08" e_name "\x00"	Boolean "false"
	|	"\x08" e_name "\x01"	Boolean "true"
	|	"\x09" e_name int64	UTC datetime
	|	"\x0A" e_name	Null value
	|	"\x0B" e_name cstring cstring	Regular expression - The f stored in alphabetical orde etc. locale dependent, 's' fo
	|	"\x0C" e_name string (byte*12)	DBPointer — *Deprecated*
	|	"\x0D" e_name string	JavaScript code
	|	"\x0E" e_name string	Deprecated
	|	"\x0F" e_name code_w_s	JavaScript code w/ scope
	|	"\x10" e_name int32	32-bit Integer
	|	"\x11" e_name int64	Timestamp
	|	"\x12" e_name int64	64-bit integer
	|	"\xFF" e_name	Min key
	|	"\x7F" e_name	Max key

BSON Notes

BSON does not include types for versioning, checksums, fixed or arbitrary decimal types… kinda important if you're a bank (SERVER-1393, SERVER-4996)!!

Strings are required to be valid UTF-8 encoded… sucks if your hostname is "普查西" (SERVER-5215)

or if you like your strings with non-English characters to sort properly (SERVER-1920)…

_id

- Each document must have an immutable _id field that is unique to the collection
- ObjectId by convention
  - 12 byte integer
    - a 4-byte value representing the seconds since the Unix epoch,
    - a 3-byte machine identifier
      - first three bytes of the (md5) hash of the machine host name, or of the mac/network address, or the virtual machine id (ex: Digest::MD5.digest(Socket.gethostname)[0, 3])
    - a 2-byte process id, and
    - a 3-byte counter, starting with a random value.
  - Monotonically increasing, contains a date at the beginning
- Can store different types, but don't do that!
  - How will they sort? Sorting order for data types, e.g. double before string…

Storage Engines at a glance

MMAPv1

- MongoDB >= 1.0
- OS manages memory
- No compression
- All collections and indexes within one set of db files
- Padding/PowerOf2Sizes, in-place updates
- Supports 32-bit and 64-bit

WiredTiger

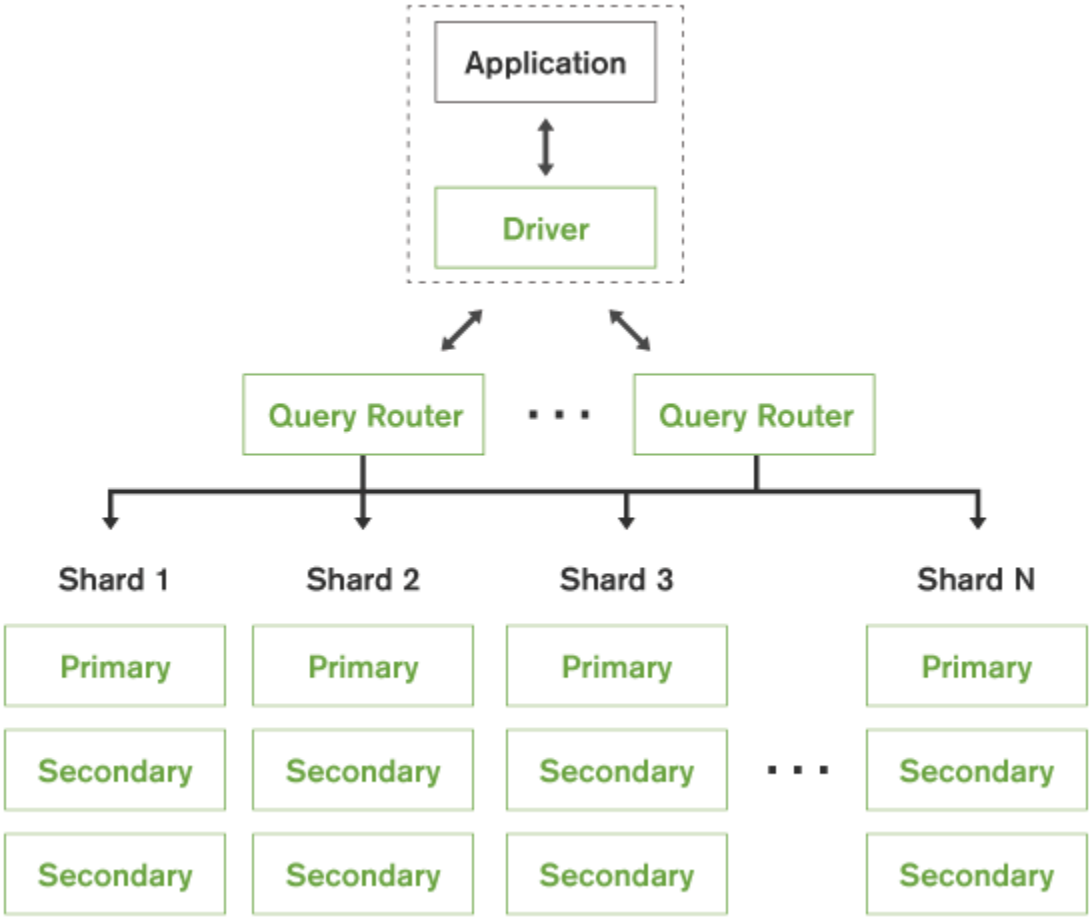
- MongoDB >= 3.0
- Server manages memory
- Compression (snappy, zlib)
- One collection or index per file
- No padding, always reallocate for writes
- Supports 64-bit only

Other storage engines

- inMemory
  - Beta in 3.2.0
- encrypted
- ephemeralForTest
  - Experiment in 3.0, kept as an example for the API

Replication

Sharding


Application
Driver
Query Router
Query Router
Shard 1
Shard 2
Shard 3
Shard N
Primary
Primary
Primary
Primary
Secondary
Secondary
Secondary
Secondary
Secondary
Secondary
Secondary
Secondary

References

- https://www.mongodb.com/support-policy
- https://www.mongodb.com/products/consulting
- https://docs.mongodb.org/manual/release-notes/3.2
- https://www.mongodb.com/customer-evaluation-downloads-development-versions
- http://bsonspec.org/spec.html