RDBMS to MongoDB Migration

Considerations and Best Practices

Mat Keep
MongoDB Product Marketing

mat.keep@mongodb.com
@matkeep

mongoDB

Agenda

- Migration Roadmap
- Schema Design
- Application Integration
- Data Migration
- Operational Considerations
- Resources to Get Started

Strategic Priorities

Enabling New &
Enhancing Existing Apps

Better Customer Experience

Faster Time to Market

Lower TCO

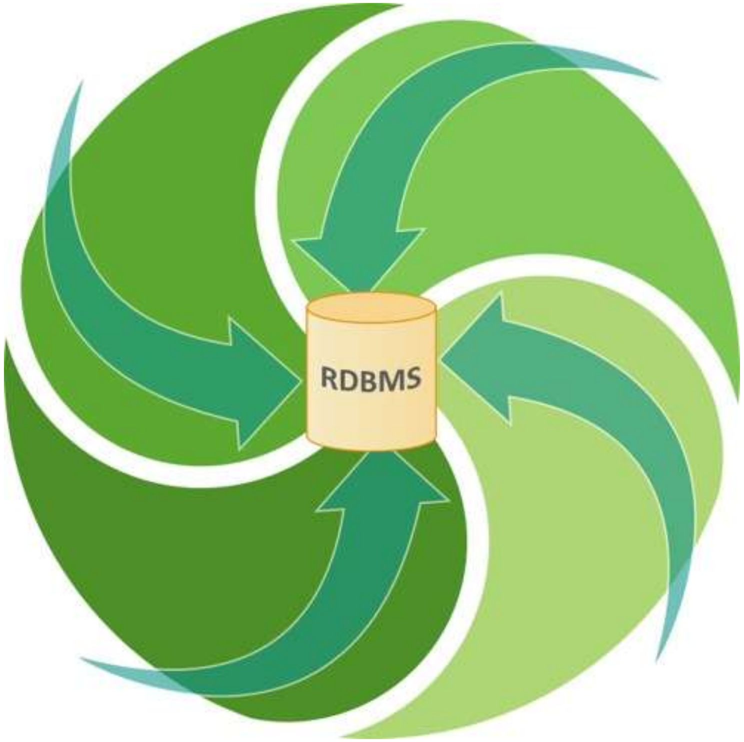
Hitting RDBMS Limits

Data Types

- Unstructured data
- Semi-structured data
- Polymorphic data

Agile Development

- Iterative
- Short development cycles
- New workloads



Volume of Data

- Petabytes of data
- Trillions of records
- Millions of queries per second

New Architectures

- Horizontal scaling
- Commodity servers
- Cloud computing

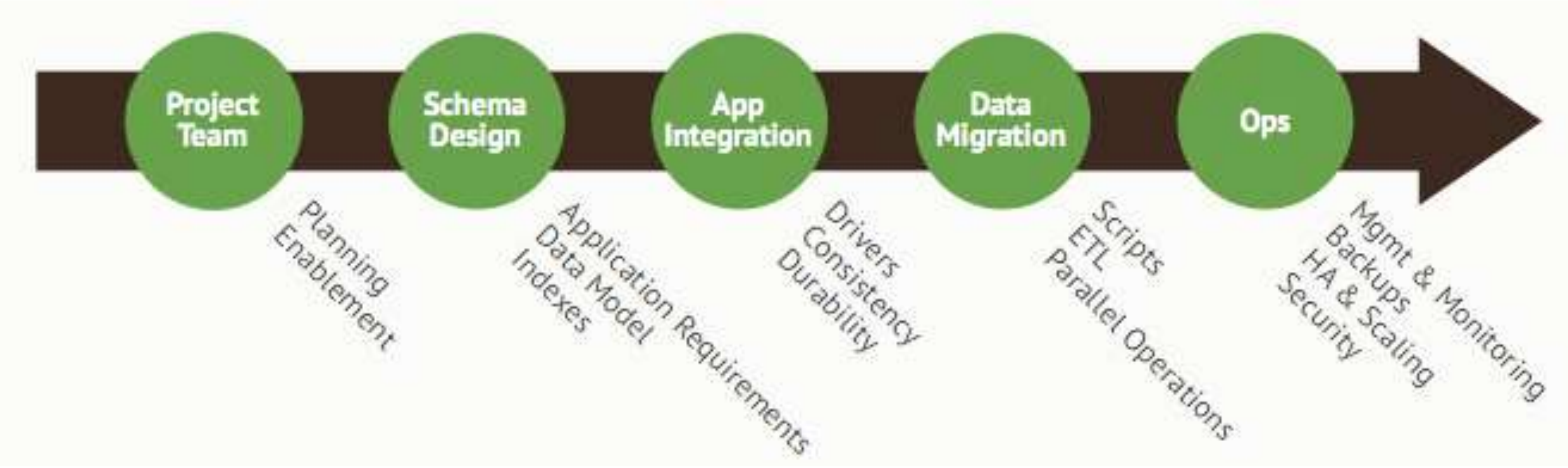
Migration: Proven Benefits

Organization	Migrated From	Application
edmunds.com	Oracle	Billing, online advertising, user data
Cisco	Multiple RDBMS	Analytics, social networking
Craigslist	MySQL	Content management
Salesforce Marketing Cloud	RDBMS	Social marketing, analytics
Foursquare	PostgreSQL	Social, mobile networking platforms
MTV Networks	Multiple RDBMS	Centralized content management
Orange Digital	MySQL	Content Management

http://www.mongodb.com/customers

Migration Steps

Migration Roadmap



- Backed by Free, Online MongoDB Training
  - 100k+ registrations to date
- Consulting and Support also available

Schema Design

On-Demand Webinar:
http://www.mongodb.com/presentations/webinar-relational-databases-mongodb-what-you-need-know-0

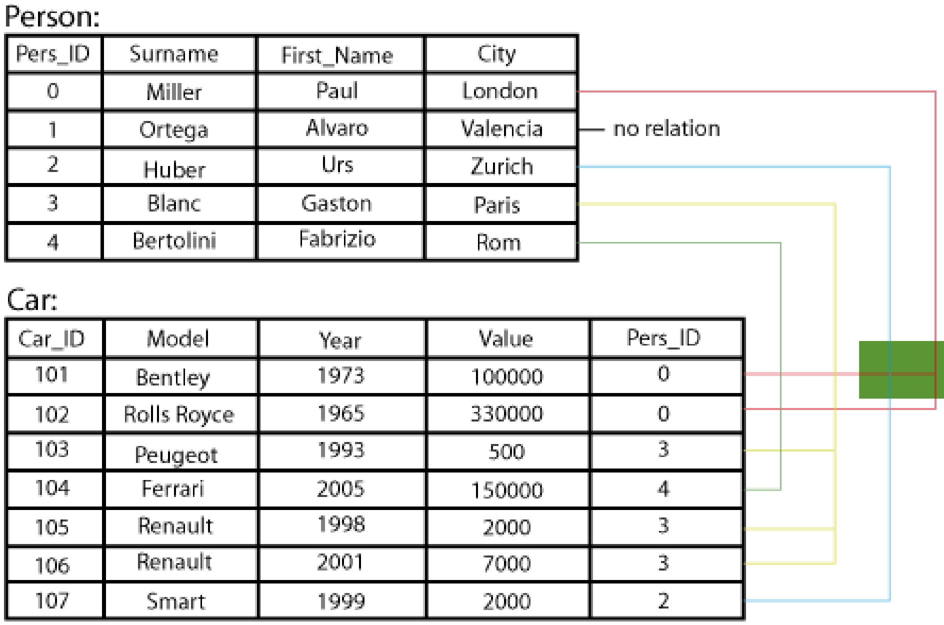
From Relational to MongoDB – What you Need to Know

Definitions

RDBMS		MongoDB
Database		Database
Table		Collection
Row		Document
Index		Index
JOIN		Embedded Document or Reference

Data Models: Relational to Document

Relational



Person:

Pers_ID	Surname	First_Name	City
0	Miller	Paul	London
1	Ortega	Alvaro	Valencia
2	Huber	Urs	Zurich
3	Blanc	Gaston	Paris
4	Bertolini	Fabrizio	Rom

Car:

Car_ID	Model	Year	Value	Pers_ID
101	Bentley	1973	100000	0
102	Rolls Royce	1965	330000	0
103	Peugeot	1993	500	3
104	Ferrari	2005	150000	4
105	Renault	1998	2000	3
106	Renault	2001	7000	3
107	Smart	1999	2000	2

MongoDB Document

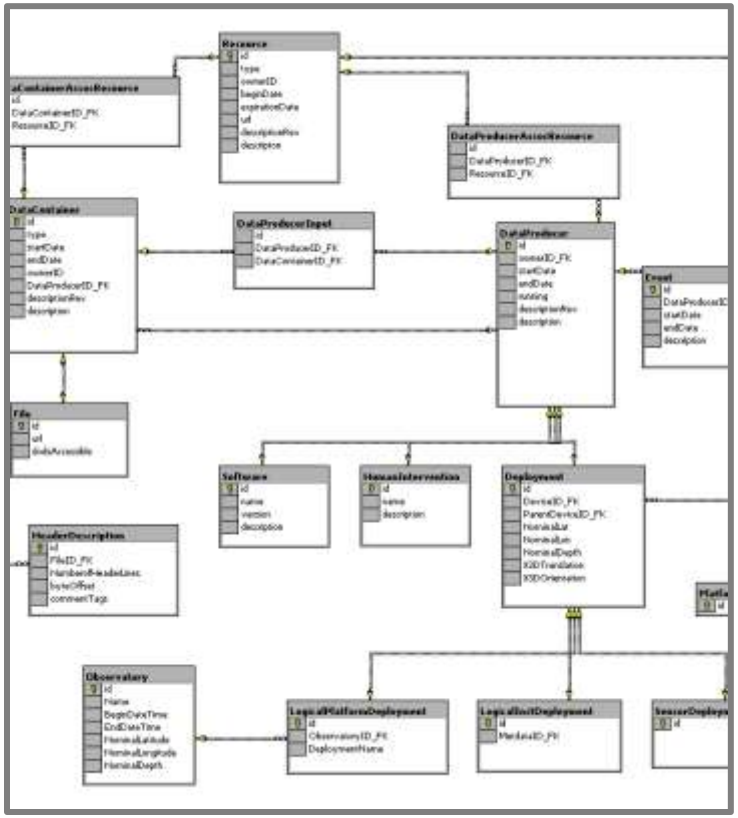
```
{
  first_name: ‘Paul’,
  surname: ‘Miller’
  city: ‘London’,
  location: [45.123,47.232],
  cars: [
   { model: ‘Bentley’,
     year: 1973,
     value: 100000, … },
   { model: ‘Rolls Royce’,
     year: 1965,
     value: 330000, … }
  ]
}
```

Document Model Benefits

- Rich data model, natural data representation
  - Embed related data in sub-documents & arrays
  - Support indexes and rich queries against any element
- Data aggregated to a single structure (pre-JOINed)
  - Programming becomes simple
  - Performance can be delivered at scale
- Dynamic schema
  - Data models can evolve easily
  - Adapt to changes quickly: agile methodology

The Power of Dynamic Schema

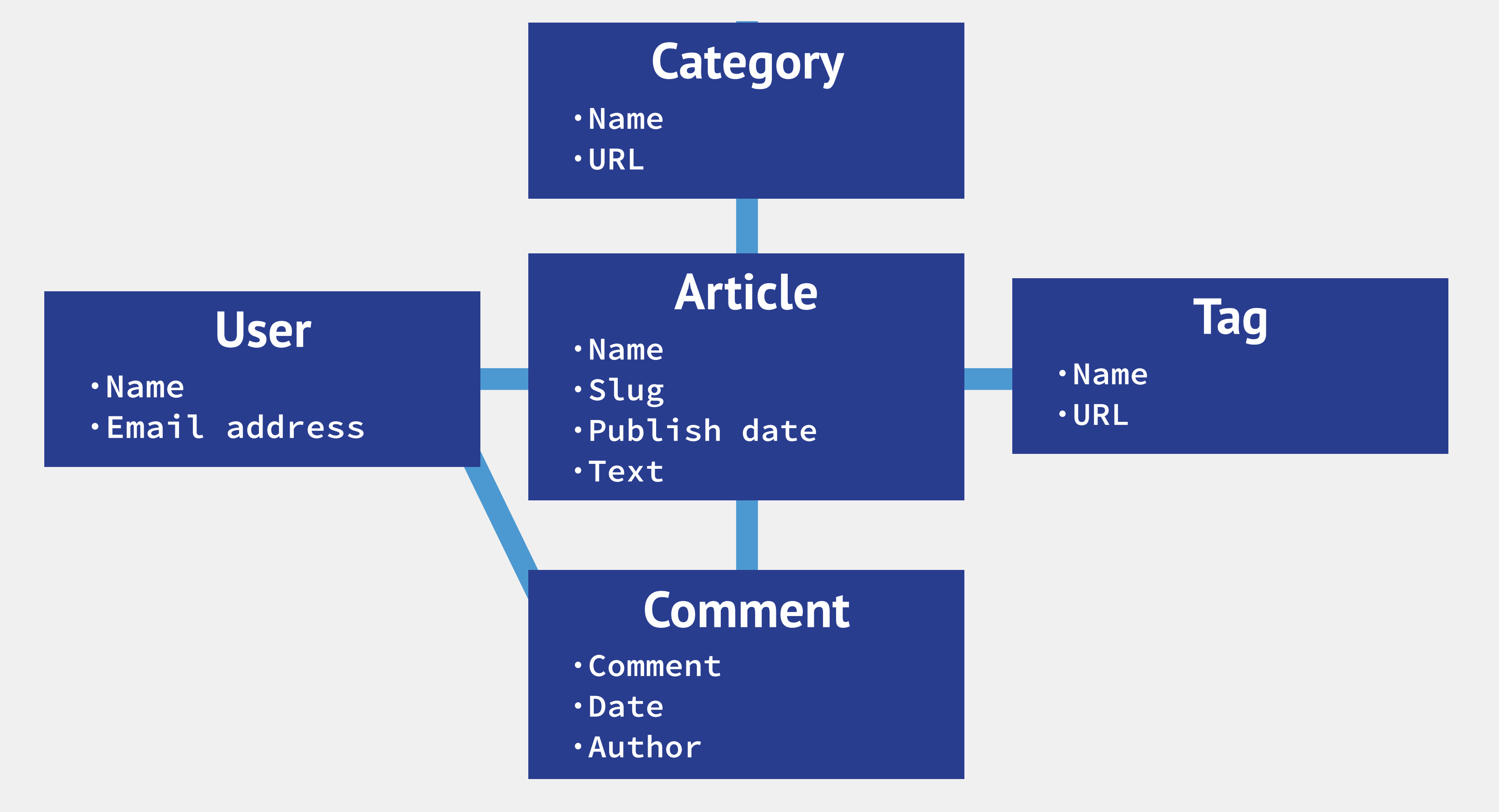
RDBMS



MongoDB

```
{
    _id : ObjectId("4c4ba5e5e8aabf3"),
    employee_name: "Dunham, Justin",
    department : "Marketing",
    title : "Product Manager, Web",
    report_up: "Neray, Graham",
    pay_band: "C",
    benefits : [
            {  type :   "Health",
               plan : "PPO Plus" },
            {  type :    "Dental",
               plan : "Standard" }
                ]
}
```

RDBMS: Blogging Platform



Category
- Name
- URL

User
- Name
- Email address

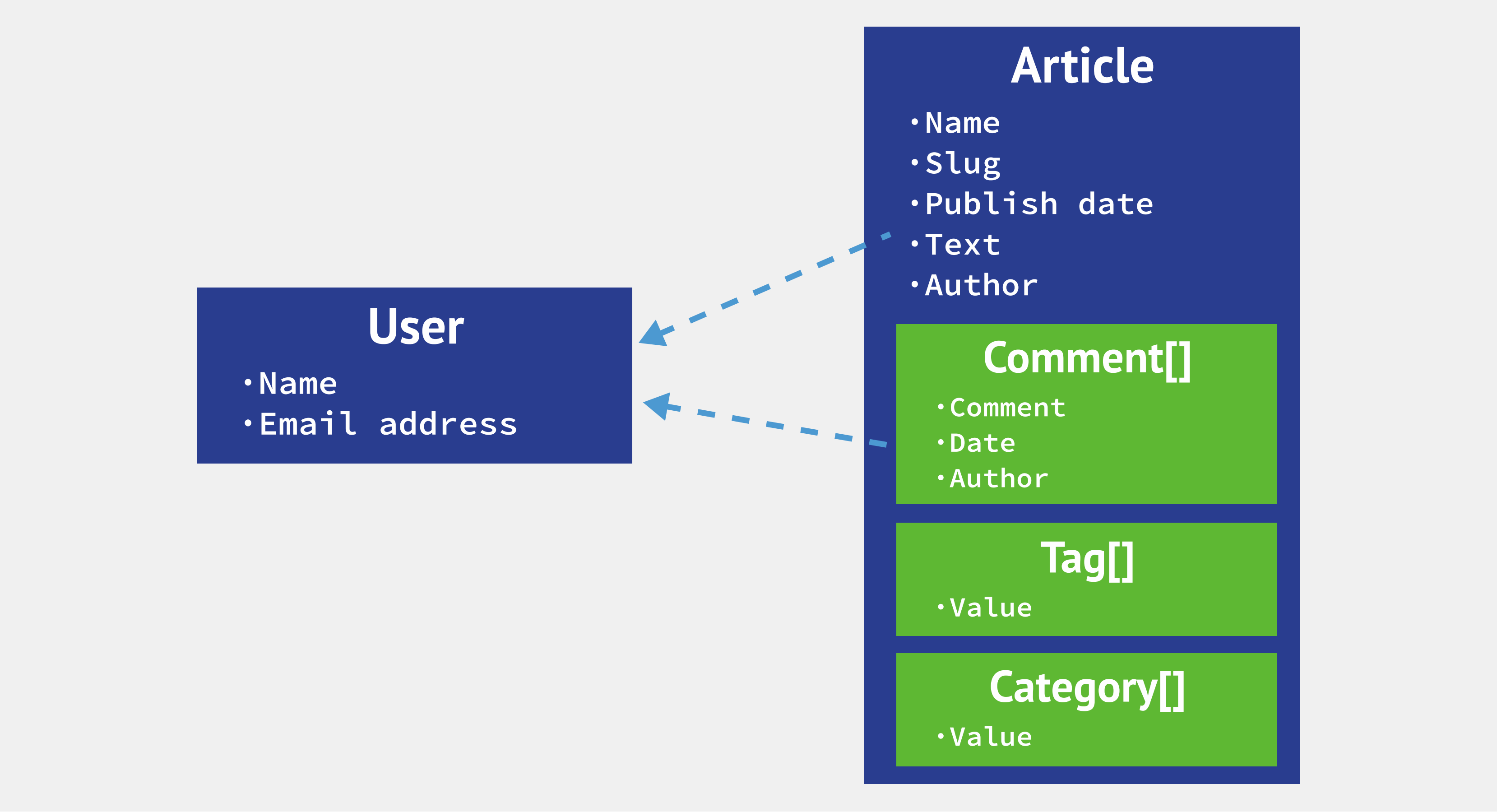
Article
- Name
- Slug
- Publish date
- Text

Tag
- Name
- URL

Comment
- Comment
- Date
- Author

JOIN 5 tables

MongoDB:
Denormalized to 2 BSON Documents



Higher Performance: Data Locality

Defining the Data Model

Application	RDBMS Action	MongoDB Action
Create Product Record	**INSERT** to **(n)** tables (product description, price, manufacturer, etc.)	**insert()** to 1 document with sub-documents, arrays
Display Product Record	**SELECT** and **JOIN** (n) product tables	**find()** aggregated document
Add Product Review	**INSERT** to "review" table, foreign key to product record	**insert() to "**review" collection, reference to product document
More actions…..	……	……

- Analyze data access patterns of the application
  - Identify data that is accessed together, model within a document
- Identify most common queries queries from logs

Modeling Relationships:
Embedding and Referencing

- Embedding
  - For 1:1 or 1:Many (where “many” viewed with the parent)
  - Ownership and containment
  - Document limit of 16MB, consider document growth
- Referencing
  - _**id** field is referenced in the related document
  - Application runs 2nd query to retrieve the data
  - Data duplication vs performance gain
  - Object referenced by many different sources
  - Models complex Many : Many & hierarchical structures

Referencing Publisher ID in Book

```
publisher = {
    _id: "oreilly",
    name: "O’Reilly Media",
    founded: "1980",
    location: "CA"
}

book = {
    title: "MongoDB: The Definitive Guide",
    authors: [ "Kristina Chodorow", "Mike Dirolf" ],
    published_date: ISODate("2010-09-24"),
    pages: 216,
    language: "English",
    publisher_id: "oreilly"
}
```

Indexing in MongoDB

- MongoDB indexing will be familiar to DBAs
  - B-Tree Indexes, Secondary Indexes
- Single biggest tunable performance factor
  - Define indexes by identifying common queries
  - Use MongoDB `explain` to ensure index coverage
  - MongoDB profiler logs all slow queries

- Compound
- Unique
- Array
- TTL
- Geospatial
- Hash
- Sparse
- Text Search

Application Integration

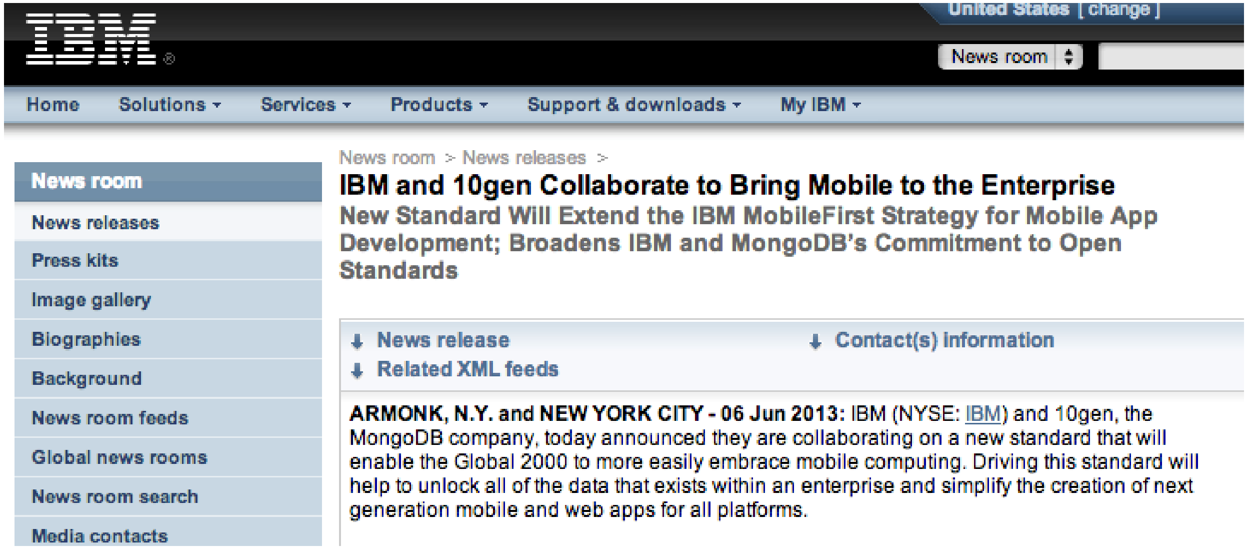
MongoDB Drivers and API

Drivers

Drivers for most popular programming languages and frameworks

Implemented as methods within API of the language, not a separate language like SQL

Java
Microsoft .NET
php
Ruby
C++
Js
C
Scala
JavaScript
Perl
Python
node JS
ERLANG
Haskell



IBM and 10gen Collaborate to Bring Mobile to the Enterprise

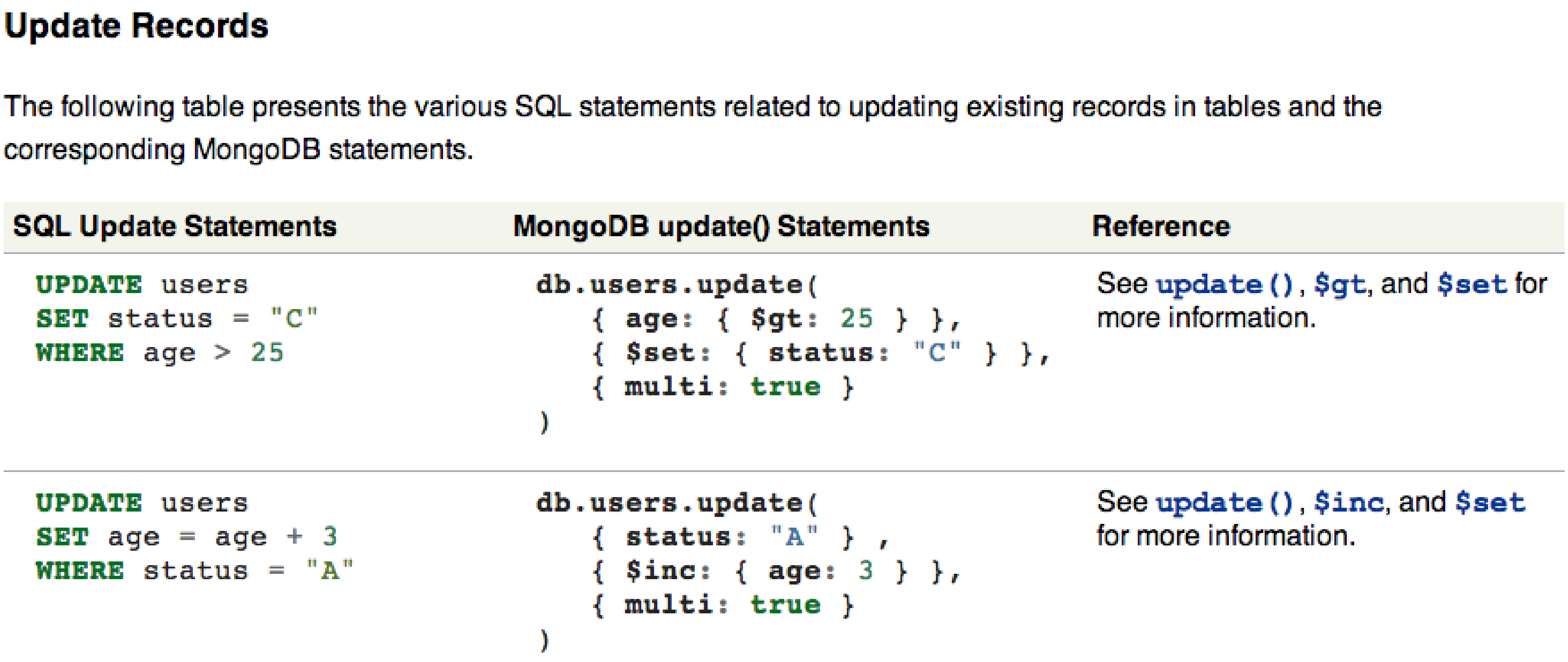
New Standard Will Extend the IBM MobileFirst Strategy for Mobile App Development; Broadens IBM and MongoDB's Commitment to Open Standards

ARMONK, N.Y. and NEW YORK CITY - 06 Jun 2013: IBM (NYSE: IBM) and 10gen, the MongoDB company, today announced they are collaborating on a new standard that will enable the Global 2000 to more easily embrace mobile computing. Driving this standard will help to unlock all of the data that exists within an enterprise and simplify the creation of next generation mobile and web apps for all platforms.

IBM

MongoDB API selected as standard for mobile app development

Mapping the MongoDB API to SQL



Update Records

The following table presents the various SQL statements related to updating existing records in tables and the corresponding MongoDB statements.

SQL Update Statements	MongoDB update() Statements	Reference

`UPDATE users`
`SET status = "C"`
`WHERE age > 25`
	`db.users.update(`
	`   { age: { $gt: 25 } },`
	`   { $set: { status: "C" } },`
	`   { multi: true }`
	`)`
		See `update()`, `$gt`, and `$set` for more information.

`UPDATE users`
`SET age = age + 3`
`WHERE status = "A"`
	`db.users.update(`
	`   { status: "A" } ,`
	`   { $inc: { age: 3 } },`
	`   { multi: true }`
	`)`
		See `update()`, `$inc`, and `$set` for more information.

Mapping Chart:

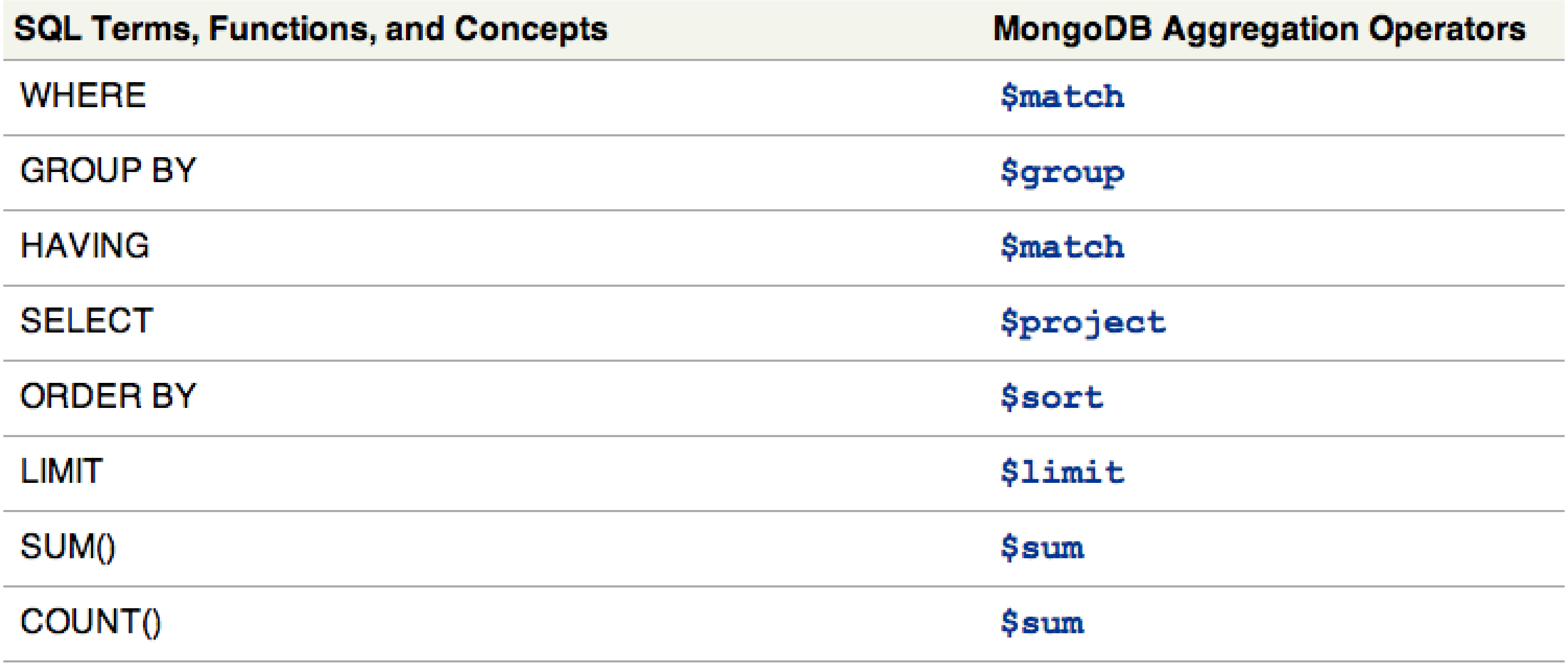
http://docs.mongodb.org/manual/reference/sql-comparison/

mongoDB

Application Integration
MongoDB Aggregation Framework

- Ad-hoc reporting, grouping and aggregations, without the complexity of MapReduce
  - Max, Min, Averages, Sum
- Similar functionality to SQL GROUP_BY
- Processes a stream of documents
  - Original input is a collection
  - Final output is a result document
- Series of operators
  - Filter or transform data
  - Input/output chain
- Supports single servers & shards

SQL to Aggregation Mapping



SQL Terms, Functions, and Concepts	MongoDB Aggregation Operators
WHERE	`$match`
GROUP BY	`$group`
HAVING	`$match`
SELECT	`$project`
ORDER BY	`$sort`
LIMIT	`$limit`
SUM()	`$sum`
COUNT()	`$sum`

Mapping Chart:

http://docs.mongodb.org/manual/reference/sql-aggregation-comparison/

Application Integration
Advanced Analytics

- Native MapReduce in MongoDB
  - Enables more complex analysis than Aggregation Framework
- MongoDB Connector for Hadoop
  - Integrates real time data from MongoDB with Hadoop
  - Reads and writes directly from MongoDB, avoiding copying TBs of data across the network
  - Support for SQL-like queries from Apache Hive
  - Support for MapReduce, Pig, Hadoop Streaming, Flume

BI Integration

mongoDB

ACTUATE
The BIRT Company™

informatica

JASPERSOFT

pentaho™
open source business intelligence™

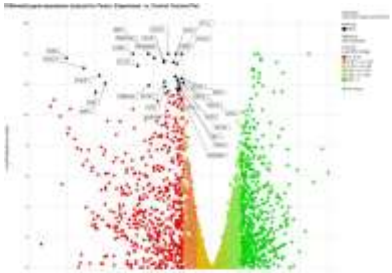
QlikView

talend*

MicroStrategy

Dashboards

Reports



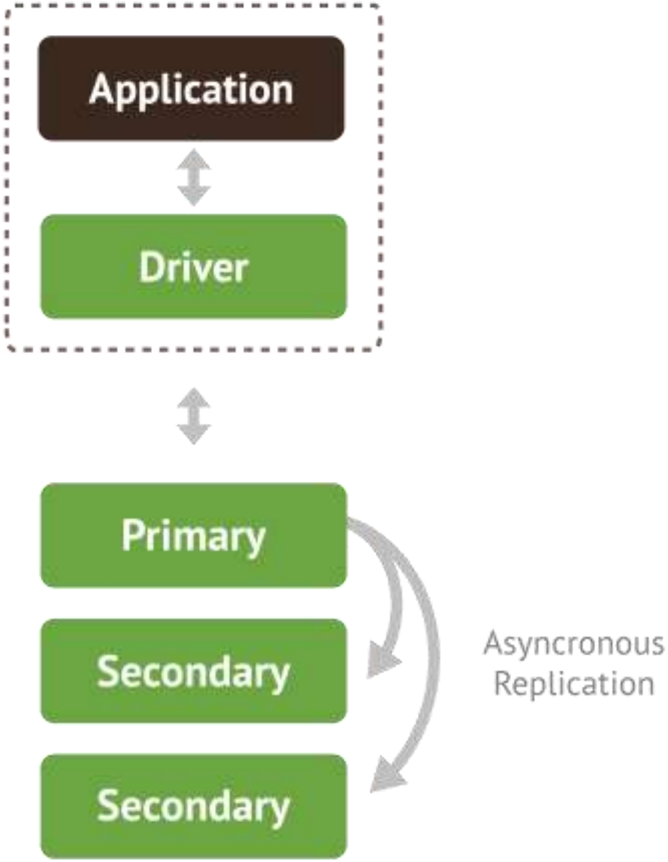
Visualizations

Document Level Atomicity

```
{
 first_name: ‘Paul’,
 surname: ‘Miller’
 city: ‘London’,
 location: [45.123,47.232],
 cars: [
   { model: ‘Bently’,
     year: 1973,
     value: 100000, … },
   { model: ‘Rolls Royce’,
     year: 1965,
     value: 330000, … }
 }
}
```

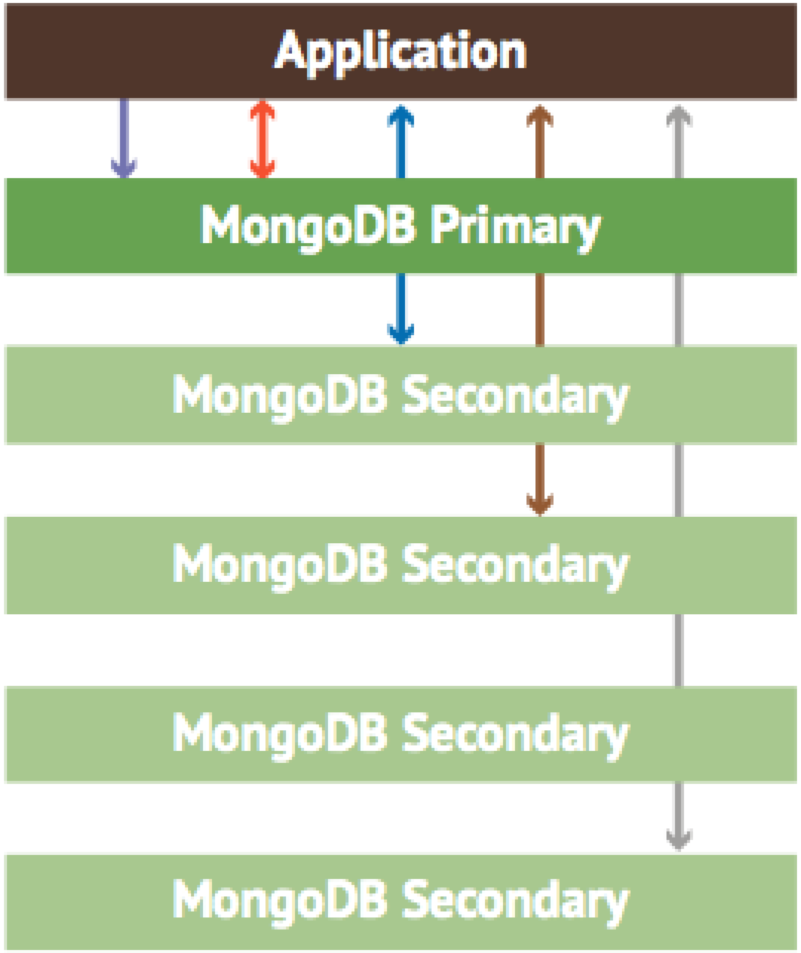
- “All or Nothing” updates
- Extends to embedded documents and arrays
- Consistent view to application
- Transaction-like semantics for multi-doc updates with **findandmodify()** or 2PC

Maintaining Strong Consistency



- By default, all reads and writes sent to Primary
  - Reads to secondary replicas will be eventually consistent
  - Scale by sharding
- Read Preferences control how reads are routed

Data Durability – Write Concerns



- Configurable per operation
  - Default is ACK by primary

⟶ Unacknowledged

⟷ Acknowledged by Primary

⟷ Acknowledged by Primary and 1 Secondary

⟷ Acknowledged by Replica Set Majority

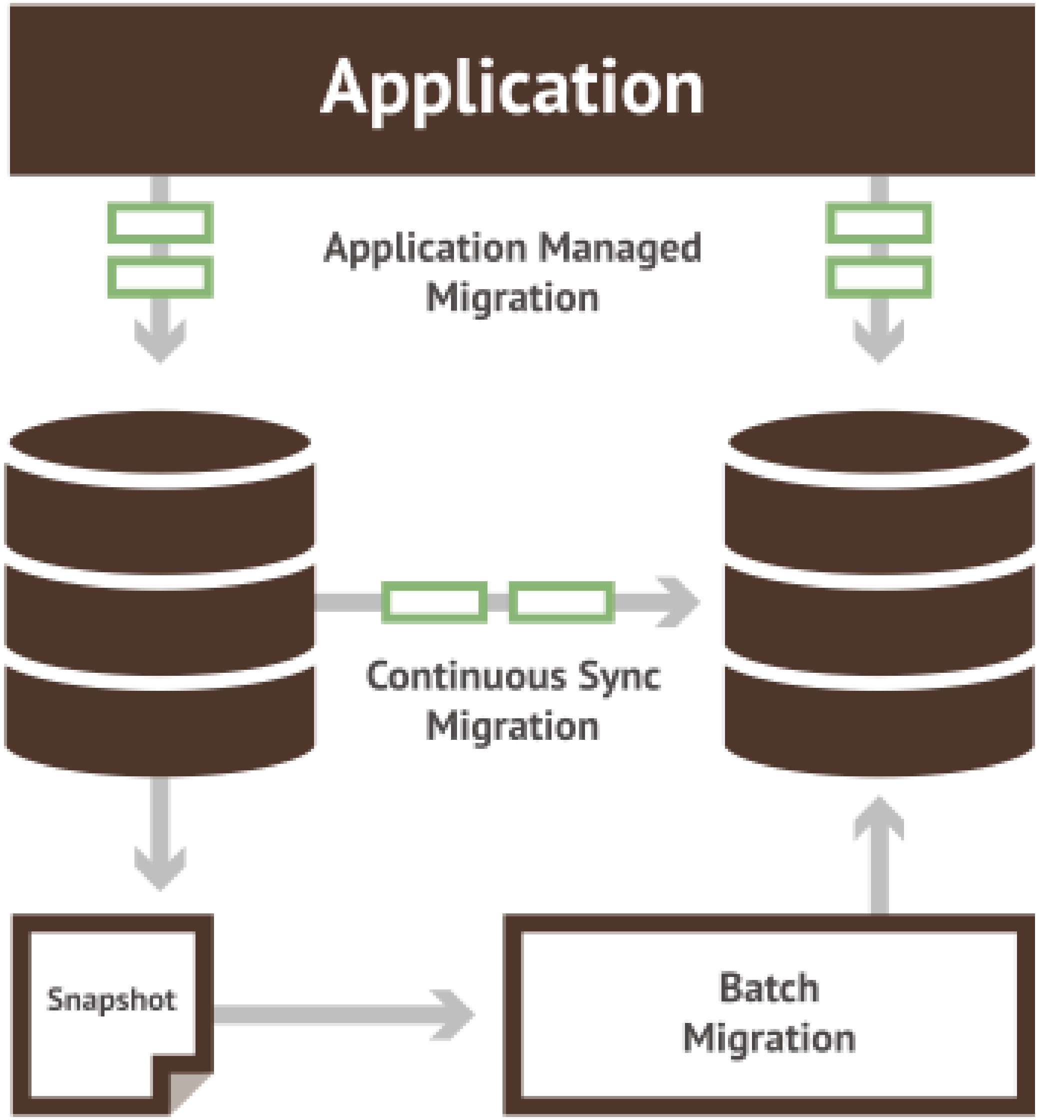
⟷ Acknowledged by Replica Set Members

Data Durability – Journaling

- Guarantees write durability & crash resistance
  - All operations written to journal before being applied to the database (WAL)
  - Configure writes to wait until committed to journal before ACK to application
  - Replay journal after a server crash
- Operations committed in groups, at configurable intervals
  - 2ms – 300ms

Migration and Operations

Data Migration



Operations

- Monitoring, Management and Backup
- High Availability
- Scalability
- Hardware selection
  - Commodity Servers: Prioritize RAM, Fast CPUs & SSD
- Security
  - Access Control, Authentication, Encryption

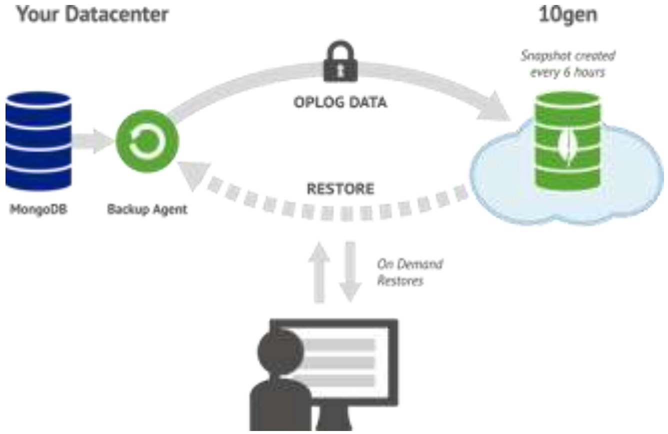
Download the Whitepaper

MongoDB Operations Best Practices

MongoDB Management Service

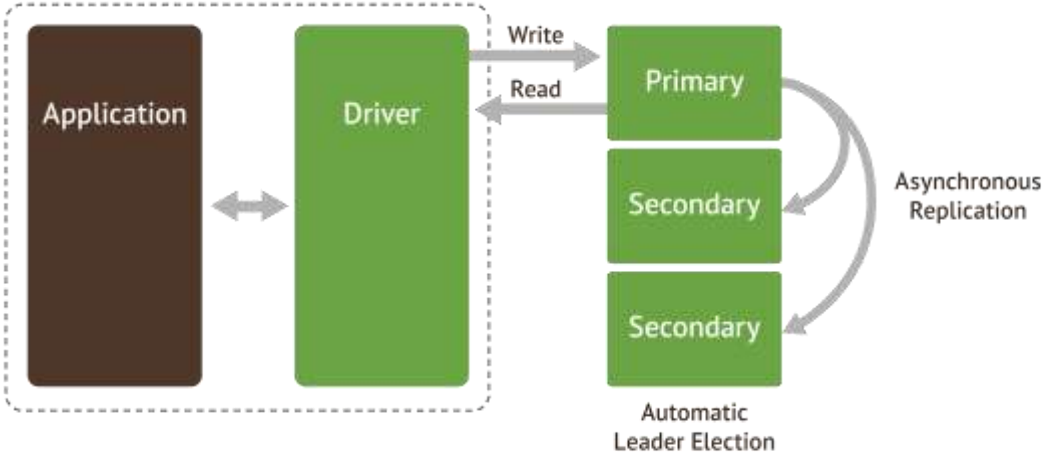
Cloud-based suite of services for managing MongoDB deployments

- Monitoring, with charts, dashboards and alerts on 100+ metrics
- Backup and restore, with point-in-time recovery, support for sharded clusters



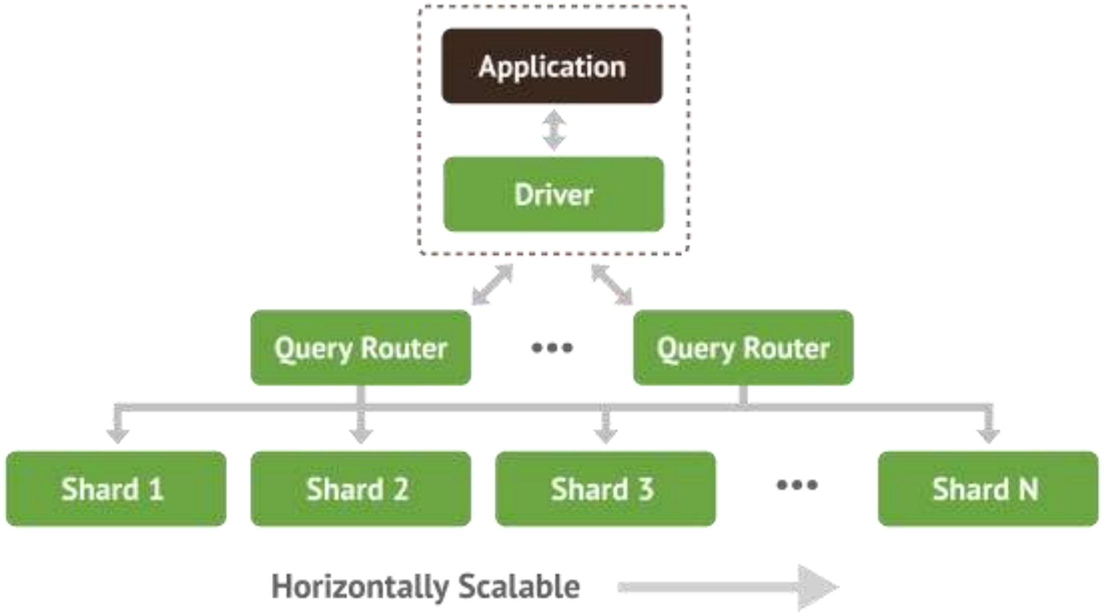
- MMS On-Prem included with MongoDB Enterprise (backup coming soon)

High Availability: Replica Sets



- Automated replication and failover
- Multi-data center support
- Improved operational simplicity (e.g., HW swaps)
- Maintenance & Disaster Recovery

Scalability: Auto-Sharding



- Three types of sharding: hash-based, range-based, tag-aware. Application transparent
- Increase or decrease capacity as you go
- Automatic balancing

Summary and Getting Started

Summary

- Benefits of migration are well understood
- Many successful projects
- Largest differences in data model and query language
  - MongoDB is much more suited to the way applications are built and run today
- Many principles of RDBMS apply to MongoDB

Download the Whitepaper

http://www.mongodb.com/dl/migrate-rdbms-nosql

For More Information

Resource	Location
MongoDB Downloads	mongodb.com/download
Free Online Training	education.mongodb.com
Webinars and Events	mongodb.com/events
White Papers	mongodb.com/white-papers
Case Studies	mongodb.com/customers
Presentations	mongodb.com/presentations
Documentation	docs.mongodb.org
Additional Info	info@mongodb.com

mongoDB

BACKUP

Enable Success:

MongoDB University

Public

- 2-3 day courses
- Dev, Admin and Essentials courses
- Worldwide

Private

- Customized to your needs
- On-Site

Online

- Free, runs over 7 weeks
- Lectures, homework, final exam
- 100k+ enrollments
- Private online for Enterprise users

Enable Success:
MongoDB Support & Consulting

Community Resource

- Google Groups & StackOverflow Forums
- MUGs, Office Hours
- IRC Channels
- Docs

Commercial Support

- Access to MongoDB engineers
- Up to 24 x 7, 30 minute response
- Unlimited incidents & hot fixes

Consulting

- Lightning consults
- Healthchecks
- Custom consults
- Dedicated TAM

Case Study

orange™

Serves variety of content and user services on multiple platforms to 7M web and mobile users

Problem

- MySQL reached scale ceiling – could not cope with performance and scalability demands
- Metadata management too challenging with relational model
- Hard to integrate external data sources

Why MongoDB

- Unrivaled performance
- Simple scalability and high availability
- Intuitive mapping
- Eliminated 6B+ rows of attributes – instead creates single document per user / piece of content

Results

- Supports 115,000+ queries per second
- Saved £2M+ over 3 yrs.
- “Lead time for new implementations is cut massively”
- MongoDB is default choice for all new projects

Case Study

salesforce marketing cloud™

Runs social marketing suite with real-time analytics on MongoDB

Problem

- RDBMS could not meet speed and scale requirements of measuring massive online activity
- Inability to provide real-time analytics and aggregations
- Unpredictable peak loads

Why MongoDB

- Ease of use, developer ramp-up
- Solution maturity – depth of functionality, failover
- High-performance with write-heavy system
- Queuing and logging for easy search at app layer

Results

- Decreased app development from months to weeks
- 30M social events per day stored in MongoDB
- 6x increase in customers supported over one year

mongoDB

Case Study

shutterfly

Uses MongoDB to safeguard over **6 billion** images served to millions of customers

Problem

- 6B images, 20TB of data
- Brittle code base on top of Oracle database – hard to scale, add features
- High SW and HW costs

Why MongoDB

- JSON-based data model
- Agile, high performance, scalable
- Alignment with Shutterfly's services-based architecture

Results

- 80% cost reduction
- 900% performance improvement
- Faster time-to-market
- Dev. cycles in weeks vs. tens of months

Case Study

Uses MongoDB to power enterprise social networking platform

Problem

- Complex SQL queries, highly normalized schema not aligned with new data types
- Poor performance
- Lack of horizontal scalability

Why MongoDB

- Dynamic schemas using JSON
- Ability to handle complex data while maintaining high performance
- Social network analytics with lightweight MapReduce

Results

- Flexibility to roll out new social features quickly
- Sped up reads from 30 seconds to tens of milliseconds
- Dramatically increased write performance

Case Study

craigslist

Stores billions of posts in myriad formats with MongoDB

Problem

- 1.5M posts per day, different structures
- Inflexible MySQL, lengthy delays for making changes
- Data piling up in production database
- Poor performance

Why MongoDB

- Flexible document-based model
- Horizontal scalability built in
- Easy to use
- Interface in familiar language

Results

- Initial deployment held over 5B documents and 10TB of data
- Automated failover provides high availability
- Schema changes are quick and easy

Case Study

Uses MongoDB as go-to database for all new projects

Problem

- RDBMS had poor performance and could not scale
- Too much operational overhead
- Needed more developer control

Why MongoDB

- Ease of use and integration with systems
- Small operational footprint
- Document model supports continuous development
- Flexible licensing model

Results

- Time from release to production reduced to <30 minutes
- Easy to add new features
- Developers can focus on apps instead of ops

Case Study

foursquare

Stores user and location-based data in MongoDB for social networking mobile app

Problem

- Relational architecture could not scale
- Check-in data growth hit single-node capacity ceiling
- Significant work to build custom sharding layer

Why MongoDB

- Auto-sharding to scale high-traffic and fast-growing application
- Geo-indexing for easy querying of location-based data
- Simple data model

Results

- Focus engineering on building mobile app vs. back-end
- Scale efficiently with limited resources
- Increased developer productivity

Case Study

GILT

MongoDB enables Gilt to roll out new revenue-generating features faster and cheaper

Problem

- Monolithic Postgres architecture expensive to scale
- Limited ability to add new features for different business silos
- Spiky server loads

Why MongoDB

- Dynamic schema makes it easy to build new features
- Alignment with SOA
- Cost-effective, horizontal scaling
- Easy to use and maintain

Results

- Developers can launch new services faster, e.g., customized upsell emails
- Stable, sub-ms performance on commodity hardware
- Reduced complexity yields lower overhead

Case Study

UNDER ARMOUR®

Built custom ecommerce platform on MongoDB in 8 Months

Problem	Why MongoDB	Results
• Dated e-commerce site with limited capabilities • Usability issues • SQL database did not scale	• Multi-data center replication and sharding for DR and scalability • Dynamic schema • Fast performance (reads and writes)	• Developers, users are empowered • Fast time to market • Database can meet evolving business needs • Superior user experience

MongoDB Features

- JSON Document Model with Dynamic Schemas
- Auto-Sharding for Horizontal Scalability
- Text Search, Geospatial queries
- Aggregation Framework and MapReduce
- Full, Flexible Index Support and Rich Queries
- Built-In Replication for High Availability
- Advanced Security
- Large Media Storage with GridFS