

Related topics to learn

- SQL Language
- Relational Database
- Non-Relational Database
- DB cluster
- MongoDB instances
- Node.js
- Neo4j
- Kubernetes
- RDBMS to NoSQL at Enterprise Scale
- MongoDB Atlas

Github: Source file and Slide

https://github.com/howtailscompany/MongoDB_Course

CONTACT

: Slack : #training-project

<https://howtailscompany.slack.com/archives/C047KPB1EP7>

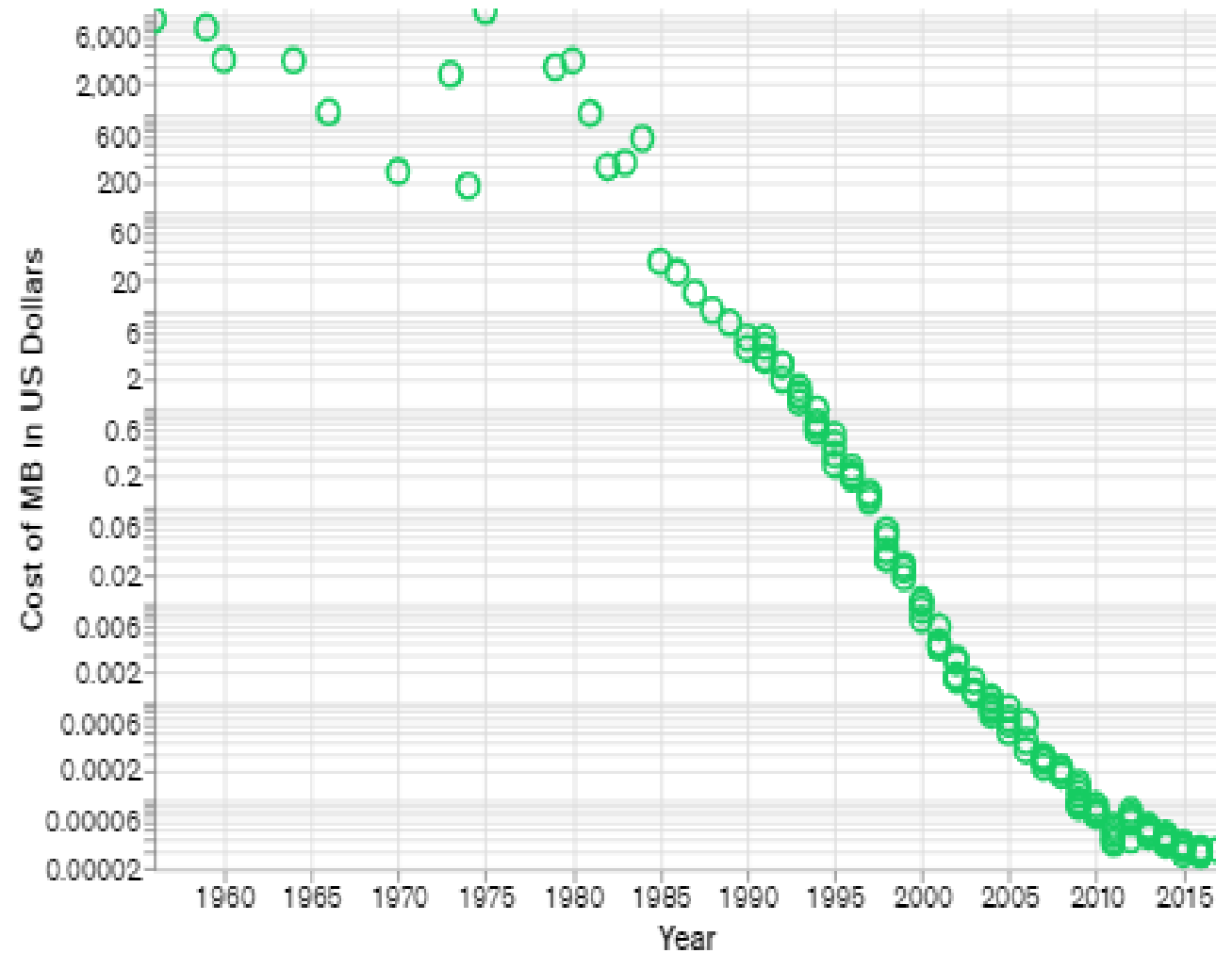
: Email : howtailscompany@gmail.com

Contents

- **Module 1:** Introduction to MongoDB
- **Module 2:** Installation and Configuration
- **Module 3:** Data Types (JSON, BSON) and Mongo Operators
- **Module 4:** Create, Read, Update and Delete operations (CRUD)
- **Module 5:** Indexing & Utilities and Storage Engines
- **Module 6:** MongoDB Database Management, Authentication and Security
- **Module 7:** Replication and Sharding
- **Module 8:** Query and Cursor
- **Module 9:** Aggregation Framework
- **Module 10:** Import Data into Cluster and Q&A

History

- ฐานข้อมูล **NoSQL** เกิดขึ้นในช่วงปลายทศวรรษ 2000 เนื่องจากค่าใช้จ่ายในการจัดเก็บลดลงอย่างมาก หมดยุคแล้วที่จะต้องสร้างแบบจำลองข้อมูลที่ซับซ้อนและจัดการได้ยากเพื่อหลีกเลี่ยงไม่ให้ข้อมูลซ้ำซ้อน (แทนที่จะเป็นพื้นที่จัดเก็บ กลายเป็นต้นทุนหลักในการพัฒนาซอฟต์แวร์) ดังนั้นฐานข้อมูล **NoSQL** จึงปรับให้เหมาะสมกับประสิทธิภาพการทำงานของนักพัฒนามากขึ้น
- แอปพลิเคชันจำเป็นต้องจัดเก็บและสืบค้นเพิ่มขึ้น ข้อมูลนี้มาในรูปแบบและขนาดทั้งหมดทั้งแบบมีโครงสร้าง กึ่งโครงสร้าง และแบบ **polymorphic** และการกำหนด **schema** ล่วงหน้าแทบจะเป็นไปไม่ได้เลย ฐานข้อมูล **NoSQL** ช่วยให้นักพัฒนาสามารถจัดเก็บข้อมูลที่ไม่มีโครงสร้างได้จำนวนมาก ทำให้มีความยืดหยุ่นสูง



Introduction to MongoDB

- MongoDB เป็น document database ประเภทหนึ่ง ที่เป็นแบบ NoSQL Database จะไม่มีการใช้คำสั่ง SQL เนื่องด้วย MongoDB ไม่รองรับการ join หรือ SQL ไม่เน้นการสร้างความสัมพันธ์ของข้อมูลแต่จะเป็นรูปแบบโครงสร้างที่เจ้าของ NoSQL สร้างขึ้นมาเองและจัดเก็บข้อมูลเป็นแบบ JSON (JavaScript Object Notation) ซึ่งจะเก็บค่าเป็น key และ value โดยจุดเด่นอยู่ที่ความเร็วในการทำงานเป็นหลัก คิวรีข้อมูลได้เร็วขึ้น Database จะเป็นทีเก็บรวบรวม collection ต่างๆ ที่มีความเกี่ยวข้องกันเอาไว้ (Collection จะคล้ายคลึงกับตารางในฐานข้อมูลเชิงสัมพันธ์) การทำงานในส่วนของ database จะลดลง โดย provides ทั้ง high performance, high availability, และ automatic scaling นั่นเอง

```
{  
  name: "sue",  
  age: 26,  
  status: "A",  
  groups: [ "news", "sports" ]  
}
```

← field: value
← field: value
← field: value
← field: value

- เร็กคอร์ดใน MongoDB เป็นเอกสาร ซึ่งเป็นโครงสร้างข้อมูลที่ประกอบด้วยคีย์ฟิลด์และค่าเอกสาร MongoDB คล้ายกับวัตถุ JSON ค่าของฟิลด์อาจรวมถึงเอกสาร อาร์เรย์ และอาร์เรย์ของเอกสารอื่นๆ

ข้อดี / ข้อเสีย ของ NoSQL

- ข้อดี : ความยืดหยุ่นสูง, การคำนวณเป็นแบบกระจาย, ค่าใช้จ่ายต่ำ, ความยืดหยุ่นทางสถาปัตยกรรม มีลักษณะเป็นข้อมูลกึ่งโครงสร้าง และ ไม่มีความสัมพันธ์ที่ซับซ้อน
- ข้อเสีย : ไม่มีมาตรฐาน, ฟังก์ชันการค้นหา จำกัด และ ไม่ได้เป็นโปรแกรมที่ใช้งานง่าย

ข้อมูลความสัมพันธ์ของการสร้างแบบจำลองในฐานข้อมูล NoSQL นั้นง่ายกว่าในฐานข้อมูลเชิงสัมพันธ์ เนื่องจากข้อมูลที่เกี่ยวข้องไม่จำเป็นต้องถูกแบ่งระหว่างตาราง โมเดลข้อมูล NoSQL อนุญาตให้ซ่อนข้อมูลที่เกี่ยวข้องภายในโครงสร้างข้อมูลเดียว

Introduction to MongoDB (Cont.)

- **ฐานข้อมูล NoSQL** ถูกใช้ในเกือบทุกอุตสาหกรรม กรณีการใช้งานมีตั้งแต่กรณีที่สำคัญอย่างยิ่ง (เช่น การจัดเก็บข้อมูลทางการเงิน และบันทึกการรักษายาบาล) ไปจนถึงเรื่องทั่วไป (เช่น การจัดเก็บการอ่าน IoT จากกล่องอัจฉริยะ)
 - USE-CASES : <https://www.mongodb.com/use-cases>
 - Industries : <https://www.mongodb.com/industries>
- เมื่อไหร่เราควรเลือกใช้ NoSQL
 - การพัฒนาแบบ **Agile** ที่รวดเร็ว
 - การจัดเก็บข้อมูลที่ไม่มีโครงสร้างและกึ่งโครงสร้าง
 - ข้อมูลปริมาณมหาศาล
 - ข้อกำหนดสำหรับสถาปัตยกรรมแบบ **scale-out**
 - การทำงานกับ แอปพลิเคชันสมัยใหม่ เช่น **microservices** และ **real-time streaming**
- **Database**
 - เป็นที่เก็บรวบรวม **collection** ต่างๆ ที่มีความเกี่ยวข้องกันเอาไว้
- **Collection**
 - ถ้าเปรียบเทียบกับ **relational database** แบบเดิม **collection** ก็เปรียบได้กับ **table** หรือที่เก็บรวบรวมข้อมูล **document** ประเภทเดียวกันเอาไว้ด้วยกัน เช่น **users** ก็เป็น **collection** ที่เก็บข้อมูลของผู้ใช้แต่ละคน
- **Document**
 - **document** เป็นชื่อที่ใช้เรียกข้อมูลแต่ละชิ้นที่เก็บอยู่ใน **database** ของเรา มีลักษณะเป็น **field : value object** เช่น ตัวอย่างด้านล่าง เป็น **document** ของผู้ใช้ 1 คน ประกอบด้วยข้อมูลคือ ชื่อ, นามสกุล, และ อายุของผู้ใช้นั้น

_id ถูกสร้างให้อัตโนมัติ ตั้งแต่สร้าง **collection**

แต่ละ **document** จะมี field **_id** ให้เพื่อป้องกัน การสร้าง **document** ซ้ำๆ โดยระบบเป็นชนิด **Unique Index**

เนื่องจาก **NoSQL** ไม่มีโครงสร้างการจัดเก็บข้อมูลที่แน่นอน เพื่อที่จะสามารถปรับเปลี่ยนโครงสร้างได้ในอนาคต ซึ่งทำให้มันสามารถเก็บข้อมูลอยู่ 4 รูปแบบ ไปตามลักษณะการใช้งาน คือ แบบ **Key-Value Store**, **Document Store**, **Graph Store** และ **Wild-Column Store**

Databases and Collections

Refer to <https://www.mongodb.com/docs/manual/core/databases-and-collections/>

Introduction to MongoDB (Cont.)

SQL	MONGODB
Database	Database
Table	Collection
Row	document or BSON document
Column	Field
Index	Index
table joins	embedded documents and linking
primary key (specify any unique column or column combinations as primary key)	primary key (the primary key is automatically set to the <code>_id</code> field in MongoDB)
aggregation (e.g. by group)	aggregation pipeline



Installation and Configuration

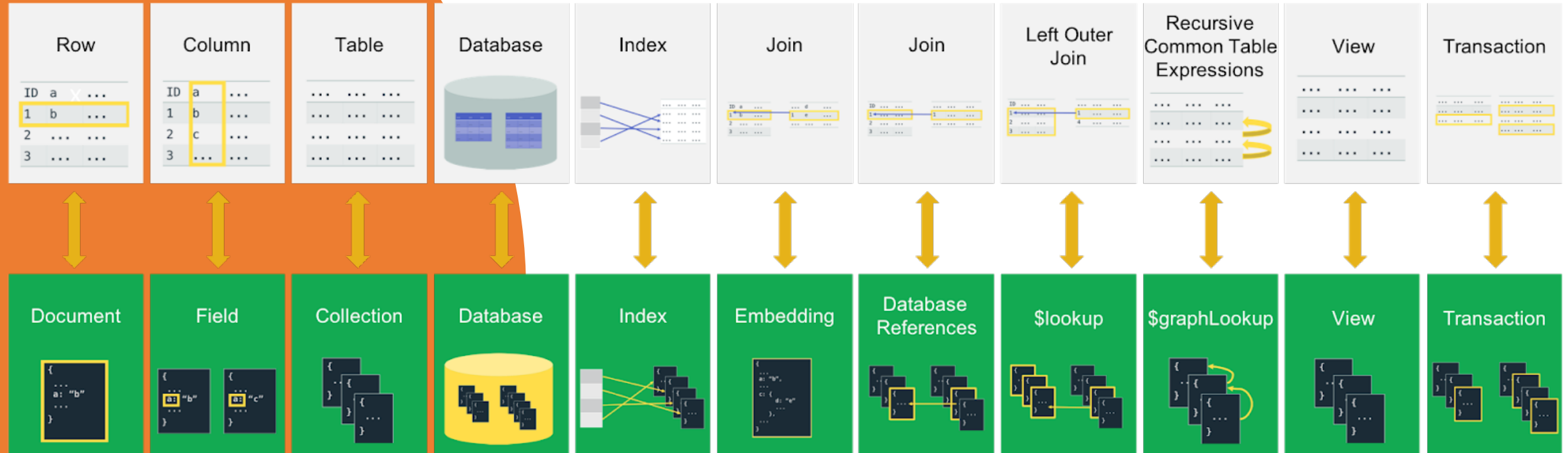
(Added Upgrade)

- DEMO > [MongoDB Community Server](#)
 - Tools
 - [MongoDB Compass](#)
 - [MongoDB Shell](#)
 - คำสั่ง Mongo Shell ที่ใช้เบื้องต้น
 - Tools - Add-ons :
 - MongoDB Command Line Database Tools
 - MongoDB BI Connector
 - MongoDB Cluster-to-Cluster Sync
 - MongoDB CLI for Cloud Manager and Ops Manager
 - Connection String URI Format
 - <https://www.mongodb.com/docs/manual/reference/connection-string/>
 - Overview : [Atlas Cloud](#)
 - [MongoDB Atlas Manual](#)
- 

Data Types (JSON, BSON)

- JSON : JavaScript Object Notation
 - Maximum document size is 1GB.
- BSON : Binary of JSON Object
 - ตัว **bson** สามารถเก็บประเภทของข้อมูลในรูปแบบของข้อมูลที่ **encode** ให้เป็น **binary** ซึ่งมีขนาดเล็ก และเวลาที่ **mongodb** นำข้อมูลที่เป็น **binary** ไปใช้งานนั้น ไม่ต้องเสีย **overhead** ในการแปลงข้อมูลมากมายนัก และที่สำคัญอีกจุดหนึ่ง คือ **bson** สามารถเก็บข้อมูลที่ **json** ไม่ **support** เช่น มีหลากหลายประเภทข้อมูลอย่าง **Double, String, Object, Array, Binary Data, ObjectId, Boolean, Date, Null, Regular Expression, JavaScript, JavaScript (with scope), 32-bit Integer, Timestamp, 64-bit Integer, Decimal128, Min Key** และ **Max** เหล่านี้ทั้งหมดมีให้ใช้งานในการสร้างแบบจำลองข้อมูลได้ตามที่มีอยู่ในโลกแห่งความเป็นจริง เป็นต้น
 - maximum BSON document size is 16MB or 16777216 bytes.
- Describe to Data Modeling
 - Schema Validation
 - Specify Validation With Query Operators
 - View Existing Validation Rules
 - Modify Schema Validation
 - Specify Validation Level for Existing Documents
 - Choose How to Handle Invalid Documents
 - Query for and Modify Valid or Invalid Documents
 - Bypass Schema Validation
- Data Modeling Relationships > One-to-One or One-to-Many
- MongoDB Limits and Thresholds >
Refer to <https://www.mongodb.com/docs/manual/reference/limits/>





Concepts from SQL to MongoDB





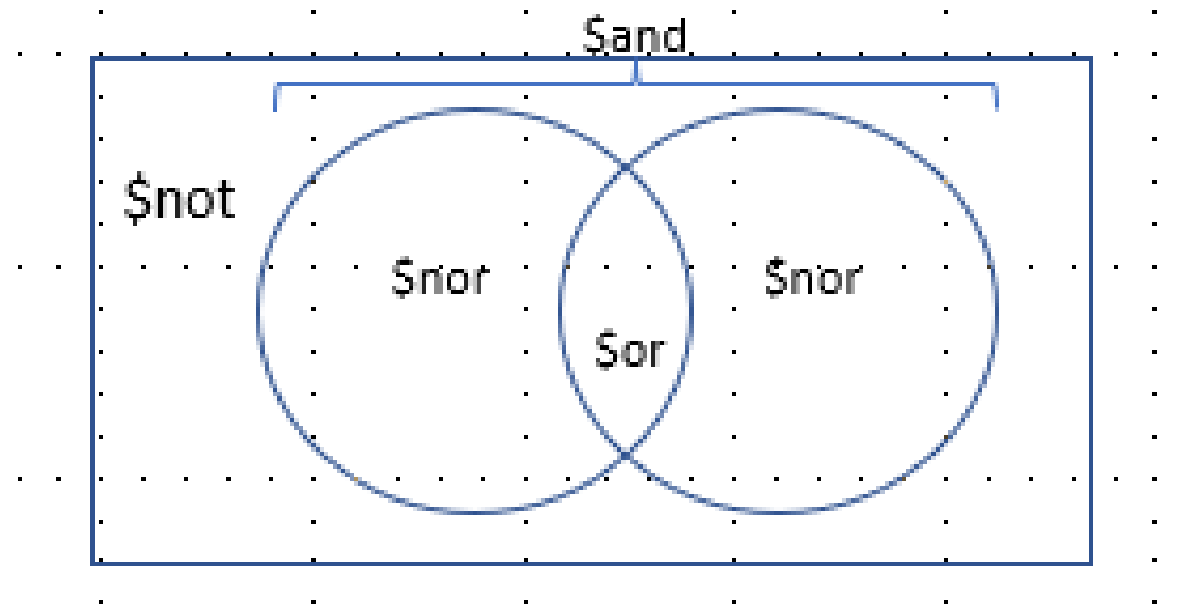
Mongo Operators


Overview

- Query and Projection Operators
 - Refer to <https://www.mongodb.com/docs/manual/reference/operator/query/>
 - Comparison Query Operators, Logical Query Operators, Element Query Operators, Evaluation Query Operators, Geospatial Query Operators, Array Query Operators, Bitwise Query Operators, Projection Operators and Miscellaneous Query Operators
 - Update Operators
 - Field Update Operators, Array Update Operators and Bitwise Update Operator
 - Refer to <https://www.mongodb.com/docs/manual/reference/operator/update/>
 - Aggregation Pipeline Stages
 - Refer to <https://www.mongodb.com/docs/manual/reference/operator/aggregation-pipeline/>
 - Aggregation Pipeline Operators
 - Refer to <https://www.mongodb.com/docs/manual/reference/operator/aggregation/>
- 
- 
- 
- 

Mongo Operators (cont.)

- Query and Projection Operators





Create, Read, Update and Delete operations (CRUD)

Demo

- Create Operations

- insert

<https://www.mongodb.com/docs/v4.2/reference/insert-methods/>

- Read Operations

- find

<https://www.mongodb.com/docs/v4.2/tutorial/query-documents/>

- Update Operations

- update

<https://www.mongodb.com/docs/v4.2/tutorial/update-documents/>

- Delete Operations

- delete

<https://www.mongodb.com/docs/v4.2/tutorial/remove-documents/>

- Bulk Write Operations

Refer to <https://www.mongodb.com/docs/v4.2/core/bulk-write-operations/>

Additional:

<https://www.mongodb.com/docs/v4.2/reference/method/db.collection.bulkWrite/#db.collection.bulkWrite>



MongoDB CRUD Concepts

Overview & Demo

- Exit Codes and Statuses

Refer to <https://www.mongodb.com/docs/manual/reference/exit-codes/>

- Atomicity, consistency, and distributed operations

- Atomicity and Transactions
- Read Isolation, Consistency, and Recency
- Distributed Queries

- Query Plan, Performance, and Analysis

- (Query Plans, Query Optimization, Analyze Query Performance, Write Operation Performance)

- Miscellaneous

- Tailable Cursors
 - Field Names with Periods (.) and Dollar Signs (\$)
- 

Create, Read, Update and Delete operations (cont.)

คำสั่ง

[create](#)

[createIndexes\[1 \]](#)

[aggregate](#)

[distinct](#)

[findAndModify](#)

[find](#)

[mapReduce](#)

[delete](#)

[update](#)

[shardCollection](#)

[count](#)

Mongosh Methods

[db.createCollection\(\)](#)
[db.createView\(\)](#)

[db.collection.createIndex\(\)\[1 \]](#)

[db.collection.aggregate\(\)](#)

[db.collection.distinct\(\)](#)

[db.collection.findAndModify\(\)](#)
[db.collection.findOneAndDelete\(\)](#)
[db.collection.findOneAndReplace\(\)](#)
[db.collection.findOneAndUpdate\(\)](#)

[cursor.collation\(\)](#)เพื่อระบุการจัดเรียงสำหรับ [db.collection.find\(\)](#)

[db.collection.mapReduce\(\)](#)

[db.collection.deleteOne\(\)](#)
[db.collection.deleteMany\(\)](#)
[db.collection.remove\(\)](#)

[db.collection.updateOne\(\)](#),
[db.collection.updateMany\(\)](#),
[db.collection.replaceOne\(\)](#)

[sh.shardCollection\(\)](#)

[db.collection.count\(\)](#)

การดำเนินการอัปเดต แทนที่ และลบแต่ละรายการใน [db.collection.bulkWrite\(\)](#)

Indexing

Overview & Demo

- When to use Index

- เราสามารถเพิ่ม Query Performance ด้วยการทำ Index ใน MongoDB เปรียบเสมือน การเขียนสารบัญที่ทำให้เราเข้าถึงข้อมูลได้เร็วขึ้น และยังประหยัดเวลาในการค้นหาได้เร็วขึ้น ข้อควรระวังคือไม่ควรทำ Index ให้กับทุกๆ field ทั้งหมดเพราะจะทำให้เมื่อต้องการ Create, Update, Insert และ Delete จะเกิด Performance ลดลงได้ ตัว Index เมื่อสร้างจำนวนที่มากขึ้นจะไปใช้พื้นที่เพิ่มขึ้นในตัว collection และ document ที่เรามีอยู่ จึงไม่ควรสร้าง Index ลงไปที่ document ที่เรามีการ Update, Insert และ Delete อยู่บ่อยๆ

- Create and Manage Index

- สำหรับ field ที่นำมาใช้ทำ Index นั้นไม่จำเป็นต้องเป็น unique field โดยสามารถนำ field อะไรมาทำก็ได้ Index มี 4 แบบ Single Field, Compound, Multikey และ Text รูปแบบ command การสร้าง index คือ

`db.collection.createIndex(<key and index type specification>, <options>)`

- Dropping Indexes

- รูปแบบ command คือ `db.Collection.dropIndex("Index_name")` และ `db.Collection.dropIndex({ "name" : 1 })` หรือ
- `db. Collection.dropIndexes(["Indexname1_1", "Indexname2_1"])`

- About Query Plan

- จะมีการเรียกใช้ method `explain()` เพื่อใช้สำหรับแสดงข้อมูลในการ query ข้อมูลของคำสั่งต่างๆใน MongoDB เช่นการ `find()`, `count()` หรือ `aggregate()` เป็นต้น
- Explain Results <https://www.mongodb.com/docs/manual/reference/explain-results/>

Utilities

Overview & Demo

- The MongoDB Database Tools

- เป็นเครื่องมือที่ใช้กับฐานข้อมูล MongoDB คือชุดของยูทิลิตี้บรรทัดคำสั่งสำหรับการทำงานกับ MongoDB ประกอบด้วยในารี่ต่อไปนี้ Import / Export, mongodump, mongotop, mongostat เป็นต้น

Refer to <https://www.mongodb.com/docs/database-tools/>

- Utilities

- ส่งคืนค่าสถิติเกี่ยวกับประสิทธิภาพและกิจกรรมของ instances อย่างรวดเร็ว โดยทั่วไปแล้ว สิ่งเหล่านี้จะมีประโยชน์มากที่สุดสำหรับการวินิจฉัยปัญหาและการประเมินการทำงานของปกติทั่วไป
- เมื่อทำการ Installation แล้วสำหรับ Window OS ตัว tools จะไปอยู่ที่ path > C:\Program Files\MongoDB\Tools\100\bin

Storage Engines

- เป็น **component** หลักของ **Mongo DB** ทำหน้าที่ในการจัดเก็บข้อมูล รับผิดชอบในการจัดการวิธีการจัดเก็บข้อมูล ทั้งในหน่วยความจำและบนดิสก์ **MongoDB** รองรับเอ็นจินการจัดเก็บข้อมูลหลายตัว
- ส่วน **Journal** คือ บันทึกที่ช่วยให้ฐานข้อมูลกู้คืนในกรณีที่เกิด **hard shutdown** มีตัวเลือกที่กำหนดค่าได้หลายแบบ เพื่อให้มีความทนทานในกรณีที่เกิดความล้มเหลว **MongoDB** ใช้การเขียนบันทึกล่วงหน้า ไปยัง ไฟล์ **journal** นี้บน **disk**
 - จะอยู่ที่ `\data\db\journal`
- ส่วน **GridFS** เป็นระบบจัดเก็บข้อมูลเอกสารที่เหมาะกับการจัดการไฟล์ขนาดใหญ่ เช่น ไฟล์ที่เกินขีดจำกัดขนาดเอกสาร 16 MB ลงในในฐานข้อมูล **MongoDB**

Refer to mongofiles > <https://www.mongodb.com/docs/database-tools/mongofiles/#mongodb-binary-bin.mongofiles>
- **WiredTiger Storage Engine** : เป็นเอ็นจินการจัดเก็บข้อมูลเริ่มต้น ที่เริ่มต้นใช้มาตั้งแต่ใน **MongoDB 3.2** ที่ใช้กับ **workload** ที่เป็น **new deployment** มีส่วนฟังก์ชันการทำงานของ **document-level, concurrency model, checkpointing** และ **compression**
 - **WiredTiger** ลบไฟล์เจอร์นัลเก่าโดยอัตโนมัติเพื่อรักษาเฉพาะไฟล์ที่จำเป็นในการกู้คืนจากจุดตรวจสอบล่าสุด
 - Refer to <https://www.mongodb.com/docs/manual/core/journaling/>
- **In-Memory Storage Engine** : พร้อมใช้งานใน **MongoDB Enterprise** แทนที่จะจัดเก็บเอกสารบนดิสก์ เอกสารจะเก็บไว้ในหน่วยความจำสำหรับ **predictable data**



MongoDB Database Management

Overview & Demo

- Managing Users : Create a User, Authenticate a User และ List Users
 - การใช้ Kerberos Authentication, LDAP Authentication
 - การ list all users ด้วย use admin > db.system.users.find()
- Perform Backup and Restore
 - Back Up, Restore, and Archive Data perform at Atlas
<https://www.mongodb.com/docs/atlas/backup-restore-cluster/>
 - Back Up and Restore with MongoDB Tools
<https://www.mongodb.com/docs/manual/tutorial/backup-and-restore-tools/>
 - Recover a Standalone after an Unexpected Shutdown
`mongod --dbpath /data/db --repair`
- Using Export and Import
 - Within MongoDB Compass
- Monitoring MongoDB :
`db.getFreeMonitoringStatus()` , `db.enableFreeMonitoring()` , `db.disableFreeMonitoring()`
 - <https://www.mongodb.com/docs/manual/administration/monitoring/>
- ACID Properties in Database Management Systems

Authentication and Security

Overview

- Default MongoDB Port

Refer to <https://www.mongodb.com/docs/manual/reference/default-mongodb-port/>

- MongoDB Server Parameters

Refer to <https://www.mongodb.com/docs/manual/reference/parameters/>

- Log Messages

Refer to <https://www.mongodb.com/docs/manual/reference/log-messages/>

- Administration

Refer to <https://www.mongodb.com/docs/manual/administration/>

- Security

Refer to <https://www.mongodb.com/docs/manual/security/>

Replication and Sharding

- **Replication** คือ การกระจายข้อมูลท่ามกลาง **MongoDB servers** หลายๆ ตัว (หรือ หลายๆ **node**) โดย **MongoDB** สามารถกระจายข้อมูลไปยัง 1 หรือมากกว่านั้นและข้อมูลจะ **sync** กันตลอดเวลาเมื่อมีข้อมูลเปลี่ยนแปลง ซึ่ง **replication** แบบนี้จะทำงานผ่านสิ่งที่เรียกว่า **replica set** โดยที่ **replica sets** ก็คือกลุ่มของ **nodes** ที่ถูกตั้งค่าให้ **sync** กันอัตโนมัติ และนอกจากการ **sync** ข้อมูลแล้ว หาก **node** หลัก (**primary node**) เสียหายหรือไม่สามารถเข้าถึงได้ มันก็ยังทำ **automatic failover** ให้
 - Restore a Replica Set from MongoDB Backups
<https://www.mongodb.com/docs/manual/tutorial/restore-replica-set-from-backup/>
- **Replica Set Status**
 - มักมีปัญหากจากการเชื่อมต่อเครือข่ายระหว่างสมาชิก ในการตรวจสอบสถานะของเรพลิกา ให้ใช้ตัว **replSetGetStatus**
>> rs.status
- **Sharding** คือการกระจายข้อมูลไปเก็บยัง **MongoDB** หลายๆ เครื่อง เพื่อเพิ่มขนาด **storage** ในการเก็บข้อมูล เพิ่มประสิทธิภาพในการทำงานและรองรับ **Horizontal Scaling** หรือ **Scale Out** การแบ่งข้อมูลสามารถแบ่งได้หลายแบบไม่ว่าจะเป็น **Rank Based** และ **Hash Based** ตามที่ต้องการ การทำ **Sharding** ใน **MongoDB** ต้องมีส่วนประกอบ 3 ส่วนได้แก่
 - Query Router – เป็นตัวเชื่อมต่อกับ **Client** หรือ **Application**
 - Config Server – เก็บข้อมูล **Meta Data** ของ **Shard Cluster**
 - Shard – เก็บข้อมูล
 - Refer to <https://www.mongodb.com/docs/v5.0/sharding/#std-label-sharding-introduction>
- **Backup and Restore Sharded Clusters**
<https://www.mongodb.com/docs/manual/administration/backup-sharded-clusters/>
- **Sharding and Monitoring**
 - ช่วยตรวจสอบเพิ่มเติมเพื่อให้แน่ใจว่าข้อมูลมีการกระจายอย่างมีประสิทธิภาพระหว่างโหนดและการดำเนินการแบ่งส่วนข้อมูลทำงานอย่างเหมาะสม



Query and Cursor

Overview & Demo

- Query
 - Query on Embedded/Nested Documents
 - Query an Array
 - Query an Array of Embedded Documents
 - Project Fields to Return from Query
 - Query for Null or Missing Fields
- Cursor
 - Iterate a Cursor in mongosh : วนซ้ำเคอร์เซอร์ใน mongosh
 - กำหนดเคอร์เซอร์ที่ส่งคืนจาก `find()` เมธอดให้กับตัวแปรโดยใช้ `var` คีย์เวิร์ด สามารถเรียกตัวแปรเคอร์เซอร์ในเชลล์เพื่อวนซ้ำได้ถึง 20 ครั้ง และพิมพ์เอกสารที่ตรงกันออกทาง `std output`
 - Refer to <https://www.mongodb.com/docs/manual/tutorial/iterate-a-cursor/>
 - Cursor Batches : MongoDB ส่งคืนผลลัพธ์การสืบค้นเป็นกลุ่ม จำนวนข้อมูลในชุดงานจะไม่เกินขนาดเอกสาร BSON สูงสุด
 - ขณะที่วนซ้ำเคอร์เซอร์และถึงจุดสิ้นสุดของแบทช์ที่ส่งคืน หากมีผลลัพธ์เพิ่มเติม `cursor.next()` จะดำเนินการ `getMore operation` เพื่อดึงแบทช์ถัดไป หากต้องการดูจำนวนเอกสารที่เหลืออยู่ในชุดงาน ขณะที่วนซ้ำเคอร์เซอร์ สามารถใช้ `objsLeftInBatch()`
- Cursor Information : เป็นเซตข้อมูล `metrics` ฟิลด์ประกอบด้วยฟิลด์ที่มี ข้อมูล `metrics.cursor` สามารถดูด้วยคำสั่ง `> db.serverStatus().metrics.cursor`



Aggregation Framework

Overview & Demo

- Views

Refer to <https://www.mongodb.com/docs/manual/core/views/>


- Create and Query a View
- Use a View to Join Two Collections
- Create a View with Default Collation
- Modify a View
- Remove a View

- System Collections

Refer to <https://www.mongodb.com/docs/manual/reference/system-collections/>

- Collection Methods

Refer to <https://www.mongodb.com/docs/manual/reference/method/js-collection/>

- Aggregation Commands
 - Aggregation Commands Comparison
 - Variables in Aggregation Expressions
 - SQL to Aggregation Mapping Chart
- 



Import Data into Cluster

- Overview & Demo





Q&A