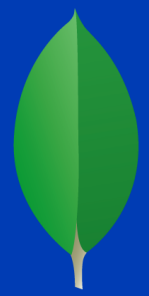


# MongoDB

"No SQL database management System"

DotNetTricks

mongoDB

# Agenda

- Introduction to NoSQL
- NoSQL Types
- Introduction to MongoDB
- Why MongoDB?
- Advantages of MongoDB
- What MongoDB doesn't have?
- Who Use MongoDB?
- RDBMS Concepts to NO SQL
- MongoDB: Hierarchical Objects

# Agenda (Contd.)

- Built-In Databases
- Data Types
- CRUD Operations
- Embed Documents
- Reference Documents

# Introduction to NoSQL

# Introduction to NoSQL

- Non-relational database management system
- Schema less
- Avoid Joins
- Easy Distribution
- Scale-out from one node to a large number of nodes
- Non-locking concurrency control mechanism
- Provide higher performance per node than RDBMS

# NoSQL Types

# NoSQL Types

- Key-value
- Graph database
- Document-oriented
- Big Table/Column families

# Introduction to MongoDB

# Introduction to MongoDB

- Introduced in 2009 by 10gen, known as MongoDB Inc.
- NoSQL database written in C, C++ and JavaScript
- Document-oriented database
- Supports dynamic schema – No DDL
- Stores data in JSON/BSON format
- Supports multiple platform like Windows, Linux, Mac etc.
- Open-source

# Why MongoDB?

# Why MongoDB?

- MongoDB stores data in objects (JSON, BSON).
- Now-a-days, programmers write code in object oriented fashion using languages like C#, Python, Php, Java etc.
- Hence, programmers need a database which can store the data in objects.
- Since, querying and manipulating data objects is easy and it reduces the time of database operations.
- Embedded documents and arrays reduce need of joins.

# Advantages of MongoDB

# Advantages of MongoDB

- Supports APIs (drivers) for many languages like C++, C#, Java, JavaScript, Python, Ruby, Perl, Java Scala etc.
- Supports Caching for speed.
- Supports Replication and Failover for High Availability.
- Supports Auto Sharding for Easy Scalability.
- Supports Indexes for high performance.
- Supports Map/Reduce for Aggregation.

# What MongoDB doesn’t have?

# What MongoDB doesn't have?

- DDL
- Joins
- Transactions
- Procedures
- Functions
- Triggers

# Who Use MongoDB?

# Who Use MongoDB?











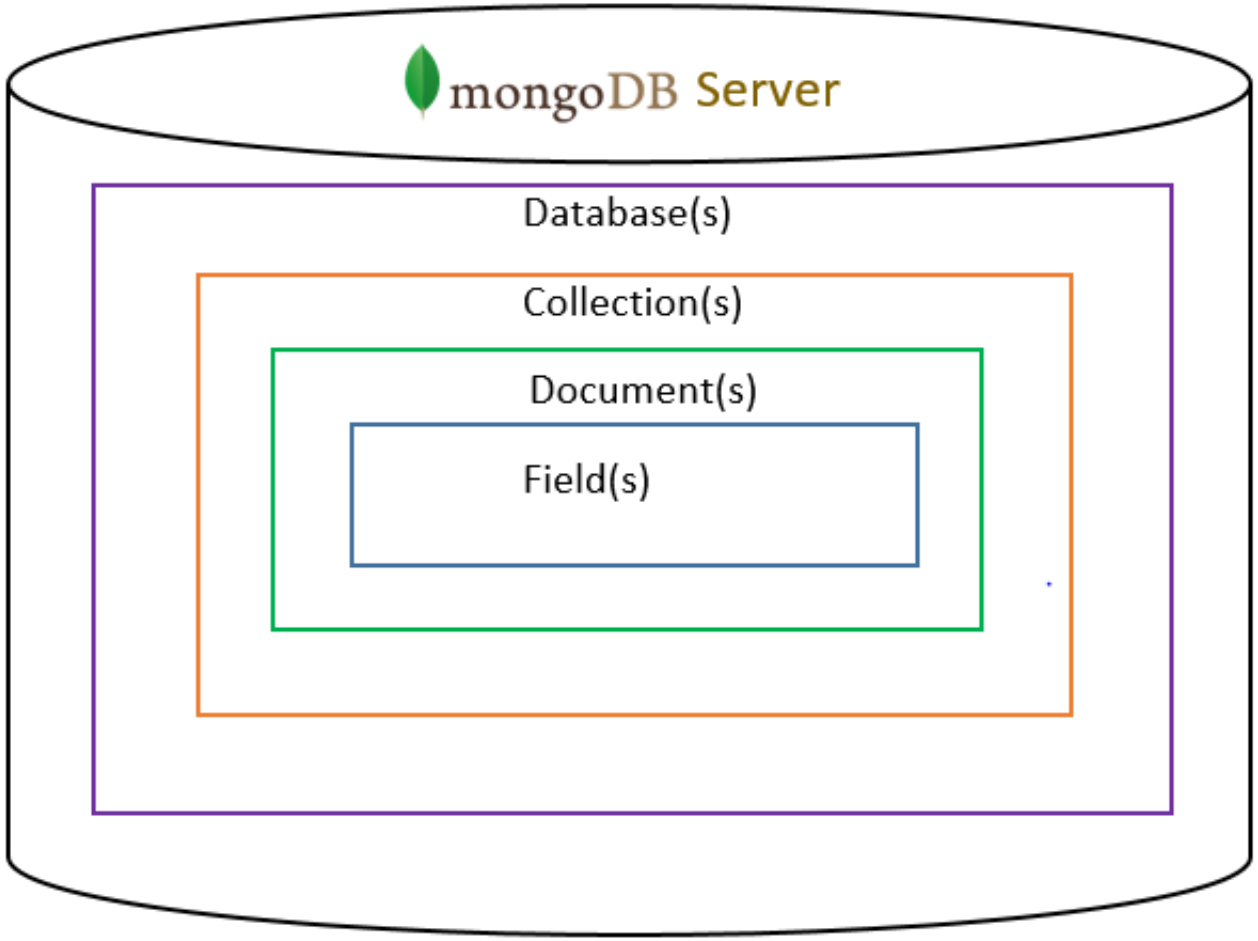

# RDBMS Concepts to NO SQL

# RDBMS Concepts to NO SQL

| RDBMS | NO SQL |
|---|---|
| Database | Database |
| Table, View | Collection |
| Row | Document |
| Column | Field |
| Primary Key | _id field |
| Foreign Key | References |
| Index | Index |
| Join | Embedded Document |
| Partition | Shard |

# MongoDB: Hierarchical Objects

# MongoDB: Hierarchical Objects



# Database

- Created on-the-fly when referenced first time.
- Contains Multiple Collections.

# Collection

- Created on-the-fly when referenced first time.
- Schema-less.
- Contains Documents.
- Supports Indexes based on one or more fields.

# Document

- Created on-the-fly when referenced first time.
- Has _id field which acts as Primary key in RDBMS.
- Supports Relationships using Embedded or References.
- Stores data in BSON (Binary form of JSON).

# Built-In Databases

# Built-In Databases

- **admin** - This is the root database. A user added to the *admin* database automatically has permissions to access all the databases. Certain server-wide commands can only be run from this database, like as listing of all the databases or shutting down the server.
- **local** - This is used to store any collections which will be local to a single server. It can not be replicated.
- **config** - This is used to store the information about the shards. To access the config database, connect to a mongos instance in a sharded cluster

# Setting up MongoDB

# Setting up MongoDB

- Set mongodb default data directory, c:/data/db
- Set user environment variable for mongodb server bin
- You can also run mongodb as a windows service by setting following path :
  - mongod --dbpath="C:\db" --logpath="C:\db\log.txt" --install
  - net start MongoDB
  - net stop MongoDB

# Data Types

# Data Types

- null
- boolean
- number
- string
- date
- regular expression
- array
- embedded document
- object id
- time stamp
- binary data
- java script code

# CRUD Operations

# CRUD Operations

- Create
  - db.collection.insert( <document> )
  - db.collection.save( <document> ) - do insert and update (with _id)
- Read
  - db.collection.find( <query>)
  - db.collection.findOne( <query> )
- Update
  - db.collection.update( <query>, <update> )
- Delete
  - db.collection.remove( <query> )

# Embedded Document

# Embedded Document

- Embed the related data in a single or document
- This schema is known as *denormalized* models

```
{
  _id: <ObjectId1>,
  username: "123xyz",
  contact: {
            phone: "123-456-7890",
            email: "xyz@example.com"
          },
  access: {
            level: 5,
            group: "dev"
          }
}
```

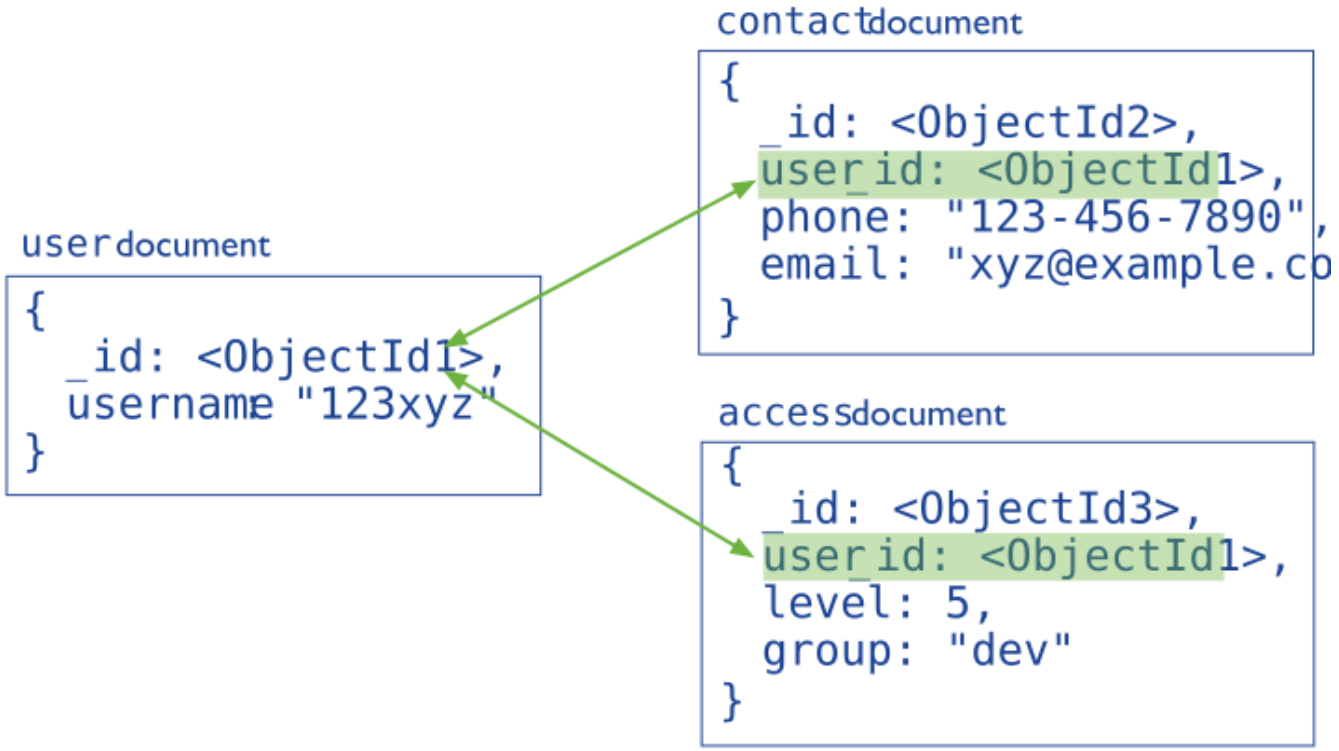
Embedded sub-document

Embedded sub-document

# Reference Document

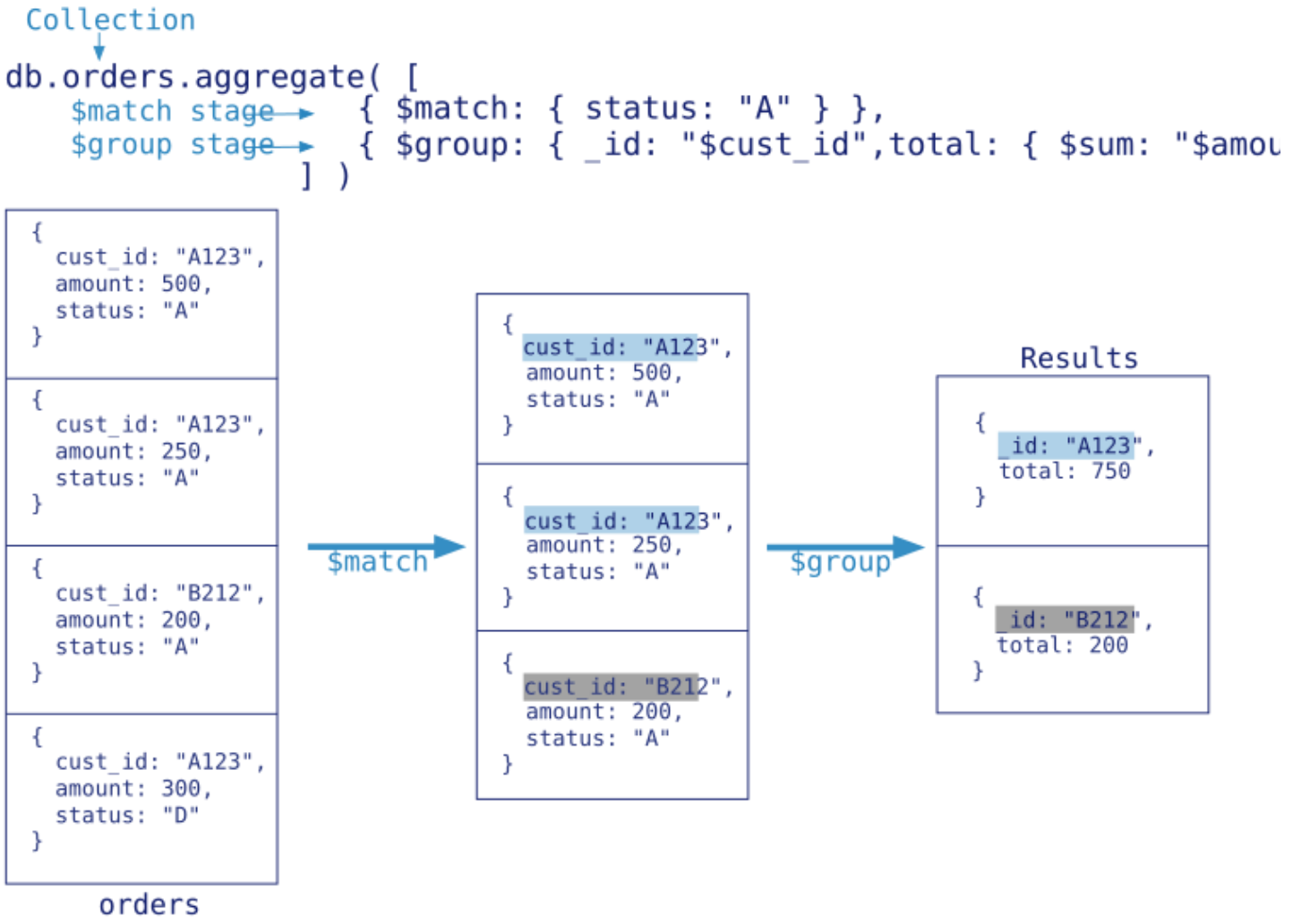
# Reference Document

- Separate the related data into more than one document
- This schema is known as normalized models

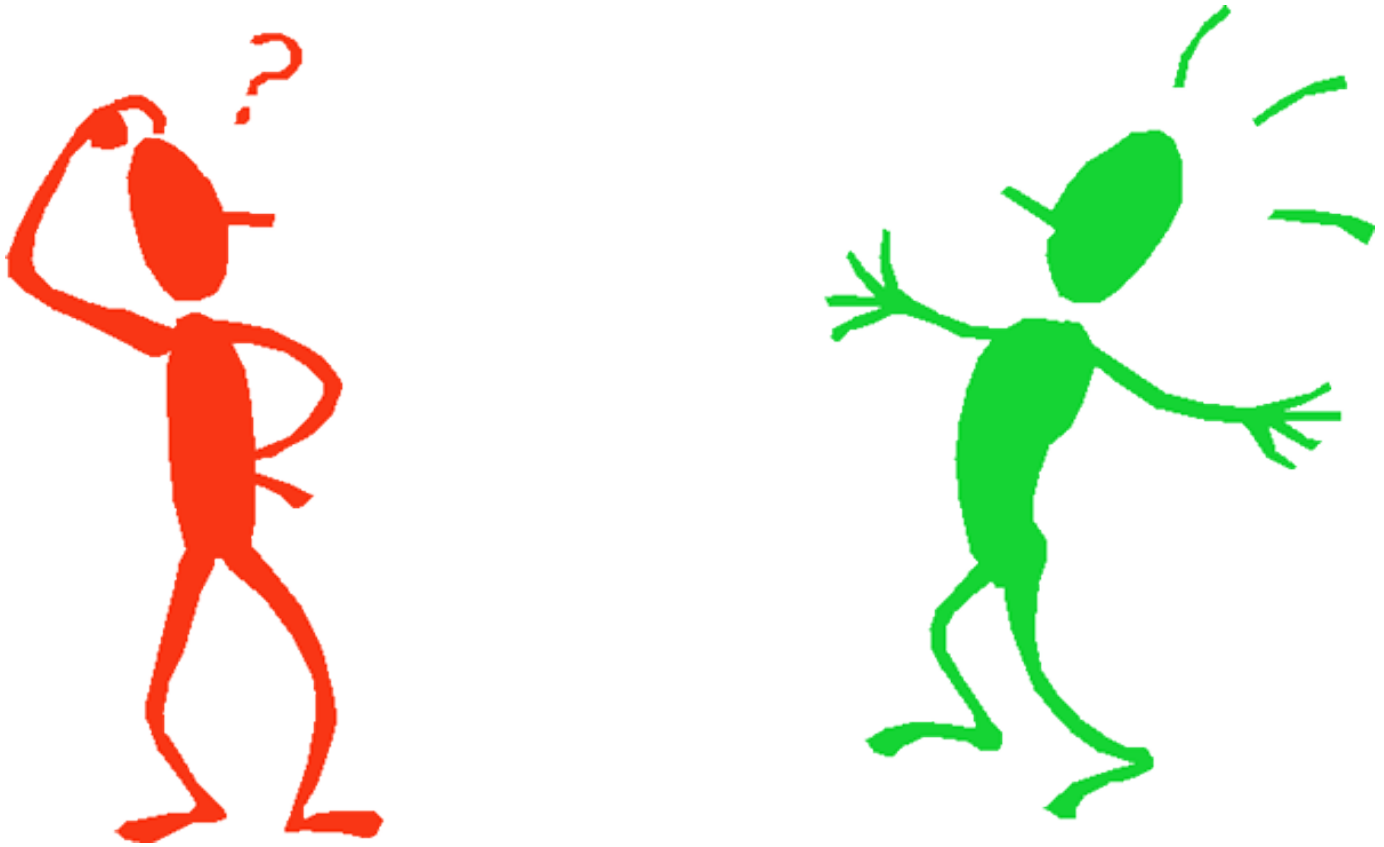


# Aggregation

# Aggregation



# Q&A



# It’s the beginning…


DotNetTricks

mongoDB