

The slide features decorative curved lines in the corners. In the top right, a thick, multi-layered arc curves from the edge towards the center, with colors transitioning from light orange to white. In the bottom left, a similar multi-layered arc curves from the edge towards the center, with colors transitioning from light orange to white. The main text is centered in a bold, dark blue font.

CS489: Applied Software Development

Lesson 5b:

NoSQL Databases –

Introduction to MongoDB

Instructor: O. Kalu

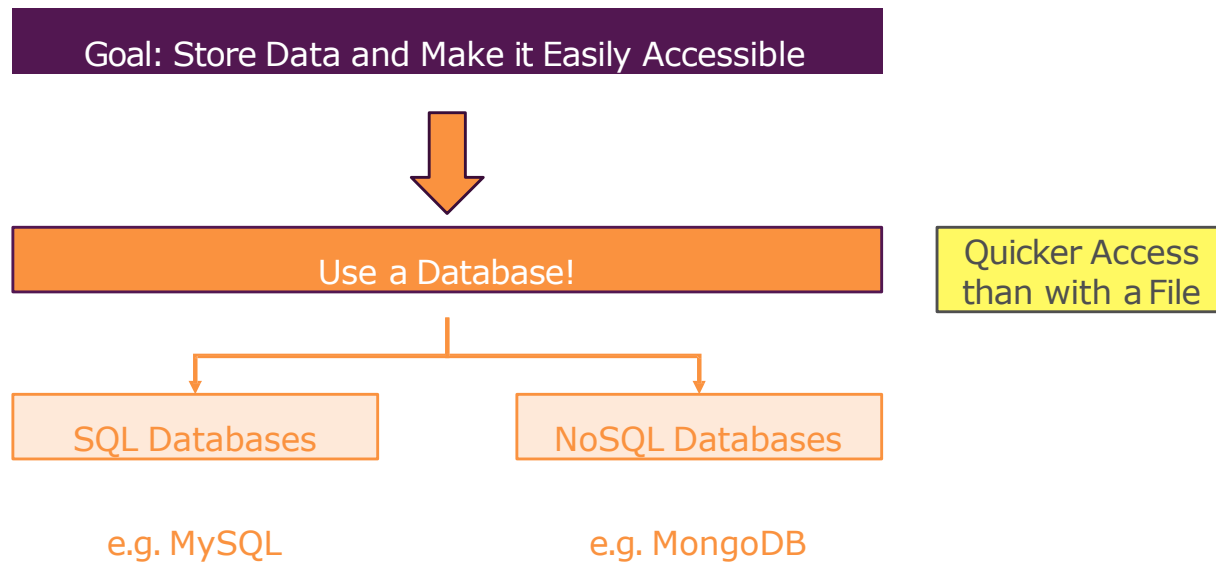
Wholeness

- This lecture aims at giving an overview and introduction to Non-relational (NoSQL) Databases and specifically focus on, a Document-oriented database named, MongoDB.
- Science of Consciousness: *Order is present everywhere; it is only our lack of understanding of the natural order of life that causes problems to arise.*

Objectives

- Know the difference between Relational (SQL) Databases and Non-relational (NoSQL) databases.
- Understand how to setup and work with MongoDB.
- Perform CRUD operations on a MongoDB database.
- Develop a MongoDB Data-driven Spring Boot CLI Application

SQL vs NoSQL



What's SQL?

User

Id	Email	Name
1	josh@miu.edu	Josh Edward
2	emma@miu.edu	Emma Smith
3

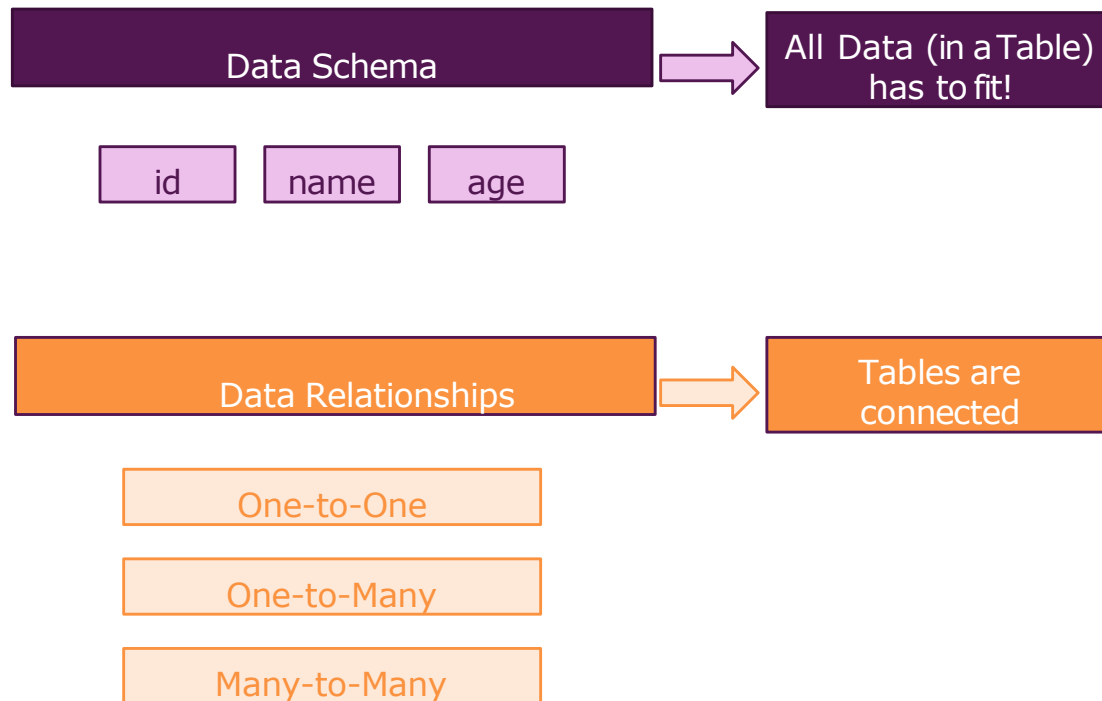
Product

Id	Title	Price	Description
1	Node.js	10	Good
2	Angular	20	Great
3	React.js	20	Great

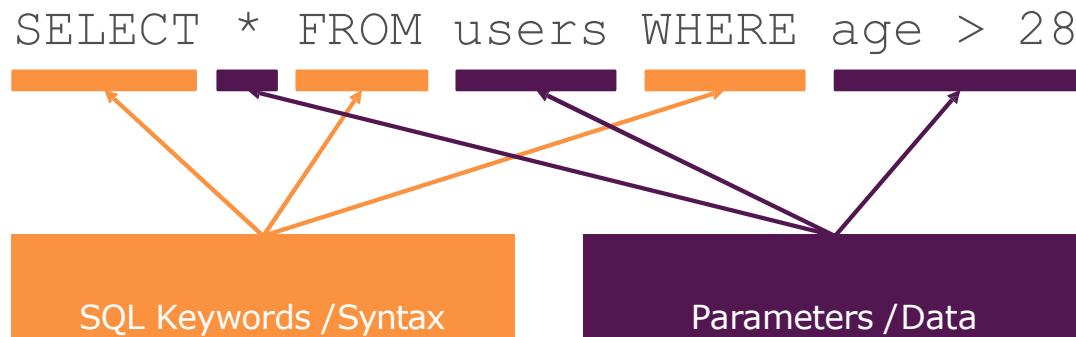
Order

Id	user_id	product_id
1	1	2
2	1	1
3	2	2

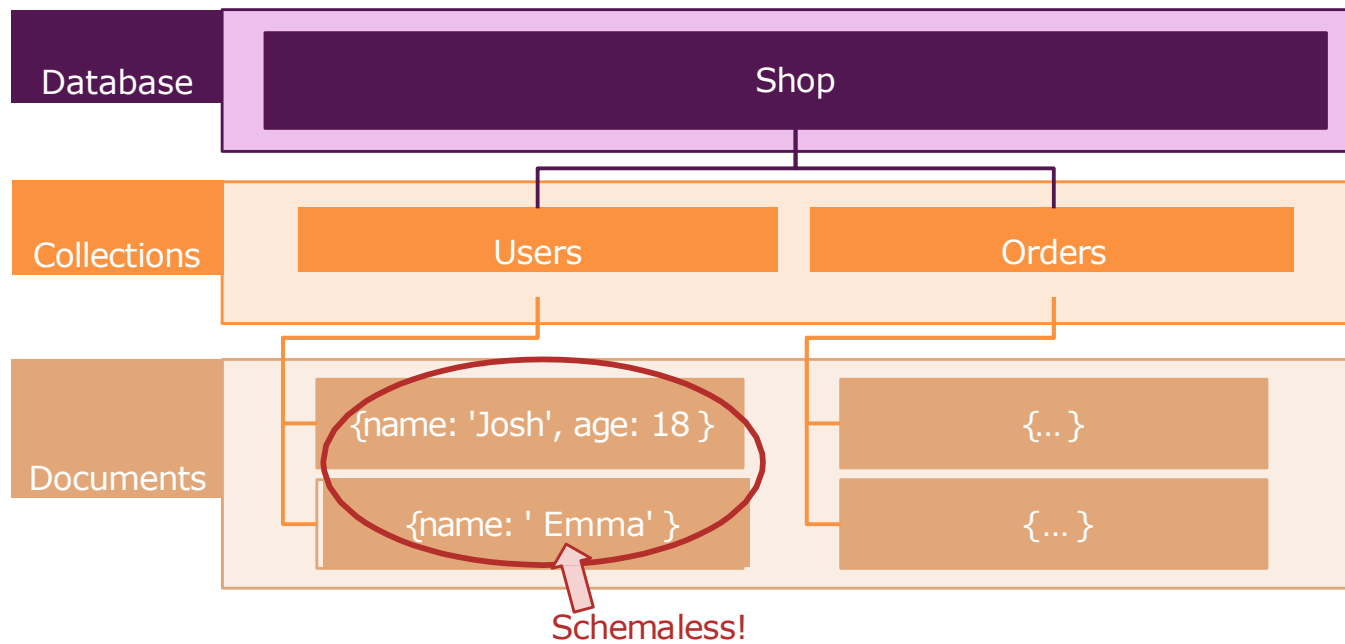
Core SQL Database Characteristics



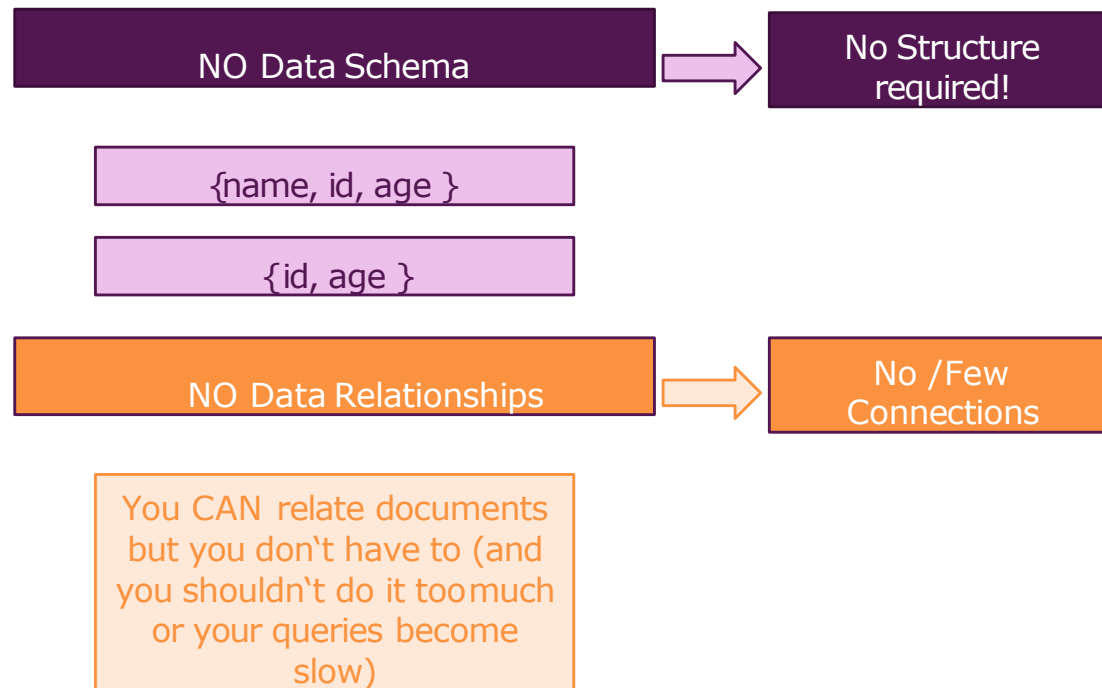
SQL Queries



NoSQL

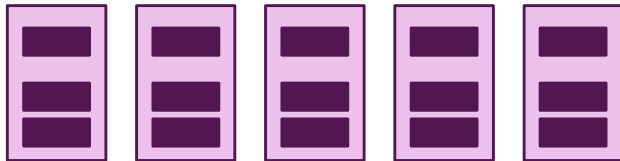


NoSQL Characteristics



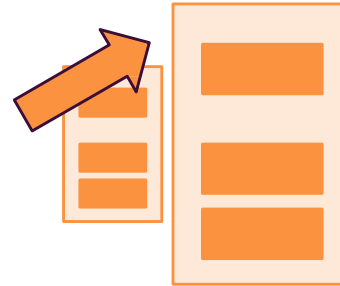
Horizontal vs Vertical Scaling

Horizontal Scaling



Add More Servers (and mergeData into one Database)

Vertical Scaling



Improve Server Capacity / Hardware

SQL vs NoSQL

SQL

Data uses Schemas

Relationships (normalized with fk)!

Data is distributed across multiple tables (normalized)

Horizontal scaling is difficult / impossible; Vertical scaling is possible

Limitations for lots of (thousands) read & write queries per second

NoSQL

Schema-less

No (or very few) Relationships

Data is typically merged / nested in a few collections (denormalized)

Both horizontal and vertical scaling is possible

Great performance for mass read & write requests

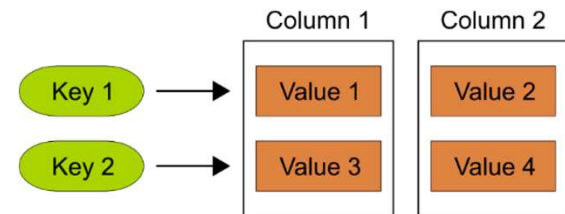
NOSQL Database Types

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

Key-Value Stores

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

Document Databases



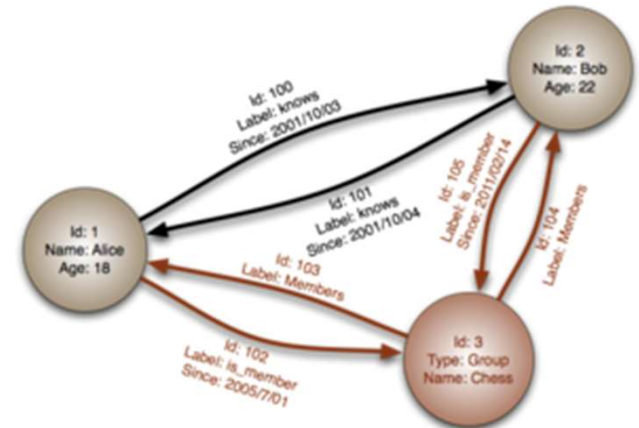
Column Family Stores

Key-Value pairs in hash table, always unique key. Logical group of keys are called: buckets

Document Databases uses Key-Value pairs in a document (JSON, BSON)

Column Stores data is stored in cells that are grouped in columns of data rather than rows (unlimited columns)

Graph Databases, uses flexible graphical representation (edges and nodes) instead of k/v pairs. Index free. Very fast for associative data sets and maps.



Graph Databases

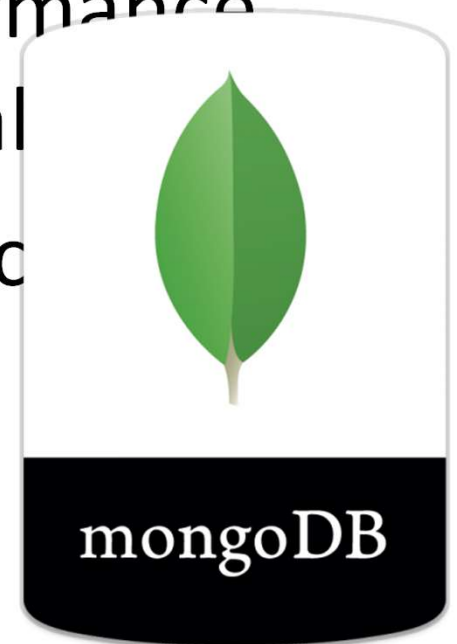
NoSQL Revolution

- NoSQL (originally referring to "non SQL" or "non relational") databases were created for "Big Data" and Real-Time Web Applications, it that can and volume

Name	Year	Type	Developer
MongoDB	2008	Document	10Gen
CouchDB	2005	Document	Apache
Cassandra	2008	Column Store	Apache
CouchBase	2011	Document	Couchbase
Riak	2009	Key-Value	Basho Technologies
SimpleDB	2007	Document	Amazon
BigTable	2015	Column Store	Google
Azure Cosmos DB	2017	Multi-Model	Microsoft

What is MongoDB?

- MongoDB is an open-source document database that provides high performance, high availability, and automatic scaling.
- Non relational DB, stores BSON documents.
- Schemaless: Two documents don't need the same schema.



Document Data Model

- A record in MongoDB is a Document
- Structure of key/value pairs
- Values may contain other documents, arrays and arrays of documents.

```
{  
  "id": 1,  
  "firstname": "Josh",  
  "lastname": "Edward",  
  "email": "test@mim.edu",  
  "phones": ["6414511111", "6414512222"]  
}
```


BSON

- BSON, short for Binary JSON, is a binary-encoded serialization of JSON-like documents.
- Both JSON and BSON support Rich Documents (embedding documents and arrays within other documents and arrays).
- BSON also contains extensions that allow representation of data types that are not part of the JSON spec. (For example, BSON has a BinData ObjectId, 64 bits Integers and Date type...etc)

BSON characteristics

- Lightweight
 - Keeping spatial overhead to a minimum is important for any data representation format, especially when used over the network.
- Traversable
 - BSON is designed to be traversed easily. This is a vital property in its role as the primary data representation for MongoDB.
- Efficient
 - Encoding data to BSON and decoding from BSON can be performed very quickly in most languages. For example, integers are stored as 32 (or 64) bit integers and they don't need to be parsed to and from text.

Non-Relational

- Scalability and Performance *(embedded data models reduces I/O activity on database system)*
- Depth of Functionality *(Aggregation framework, Text Search, Geospatial Queries)*
- To retains scalability
 - MongoDB **does not support** favor Joins between two collections *(\$lookup)*
 - **No relational algebra:** tables/columns/rows *(SQL)*
 - **No Transactions** across multiple collections *(Do it programmatically, documents can be accessed atomically) – Newer versions of MongoDB now support ACID transaction to an extent*

Schema

- By default, a collection does not require its documents to have the same schema, the documents in a single collection do not need to have the same set of fields and the data type for a field can differ across documents within a collection.
- Starting of MongoDB 3.2, you can enforce document validation rules for a collection during update and insert operations

Document Structure

- The value of a field can be any of the BSON data types, including other documents, arrays, and arrays of documents.

```
const doc = {  
  _id: new ObjectID('5e44ab7638d4f738f05c57a8'),  
  name: { first: "Josh", last: "Edward" },  
  birth: new Date('Oct 31, 1979'),  
  email: "test@mim.edu",  
  phones: ["6414511111", "6414512222"]  
}
```

Setup

- Follow the link to install MongoDB
 - <https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows/>
- Two ways to start your MongoDB
 - as a Windows Service
 1. From the Services console, locate the MongoDB service.
 2. Right-click on the MongoDB service and click **Stop** (or **Pause**).
 - from the Command Interpreter
 1. Create database directory - C:/data/db
 2. Start your MongoDB database.
 - "C:\Program Files\MongoDB\Server\4.2\bin\mongod.exe" --dbpath="c:\data\db"
 3. Connect to MongoDB
 - "C:\Program Files\MongoDB\Server\4.2\bin\mongo.exe"

Create new Database and Collections

- MongoDB stores documents in collections. (Collections are similar to tables in relational databases)

use myDB

- If a database/collection does not exist, MongoDB creates the db/collection when you first store data for that collection

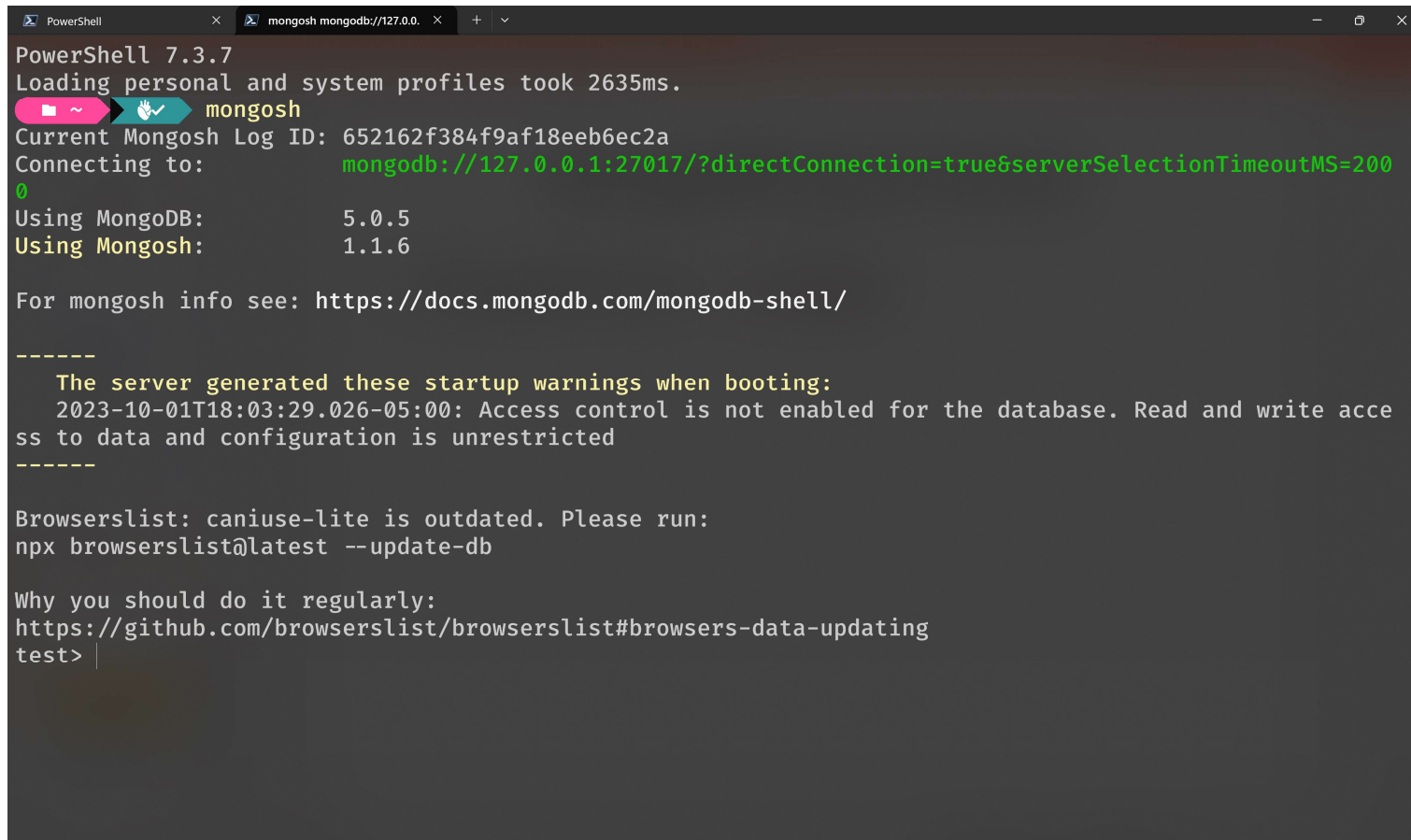
use myNewDB

```
db.myNewCollection.insertOne( { x: 1 } )
```

- *The insertOne() operation creates both the database myNewDB and the collection myNewCollection if they do not already exist.*

MongoDB Tools Demo

- MongoDB Shell – mongosh



```
PowerShell 7.3.7
Loading personal and system profiles took 2635ms.
~ mongosh
Current Mongosh Log ID: 652162f384f9af18eeb6ec2a
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Using MongoDB:      5.0.5
Using Mongosh:       1.1.6

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

-----
The server generated these startup warnings when booting:
2023-10-01T18:03:29.026-05:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

Browserslist: caniuse-lite is outdated. Please run:
npx browserslist@latest --update-db

Why you should do it regularly:
https://github.com/browserslist/browserslist#browsers-data-updating
test> |
```


MongoDB Tools Demo

- MongoDB Compass

MongoDBApplication Development

- Demo/Exercise:
- Using Spring Boot, create a new CLI Application project, adding as dependencies – Spring Data MongoDB
- Implement Code to perform Data Access operations with MongoDB database

Connecting the Parts of Knowledge With the Wholeness of Knowledge

Overview of NoSQL Databases and MongoDB

1. NoSQL Databases offer relatively better performance than Relational databases, but they trade-off some Data integrity features found in relational (SQL) databases.
2. MongoDB is arguably the predominant document-oriented NoSQL database currently in use in many real-world, high-performant applications.

-
3. **Transcendental consciousness** is the underlying basis of all levels of creation.
 4. **Impulses within the Transcendental Field:** The performance benefit of NoSQL databases such as MongoDB, forms the basis of several high-throughput software applications, and this arises as an impulse of the Transcendental Field.
 5. **Wholeness moving within itself:** In Unity Consciousness, one directly perceives that all expressions and levels of creation are nothing more than one's own Self – pure consciousness.

The slide features decorative curved lines in the corners. In the top right, a thick, multi-layered arc curves from the edge towards the center, with colors transitioning from light orange to white. In the bottom left, a similar multi-layered arc curves from the edge towards the center, with colors transitioning from light orange to white. The main text is centered in a bold, dark blue font.

CS489: Applied Software Development