# Full Stack III - Lecture 7.0

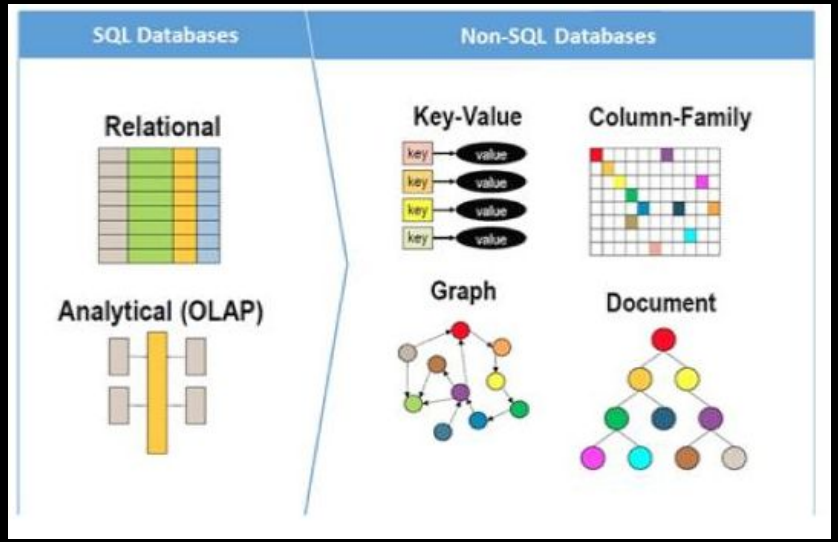
Introduction to mongoDB

# Topics

- NoSQL vs SQL
- MongoDB
- Document & Collections

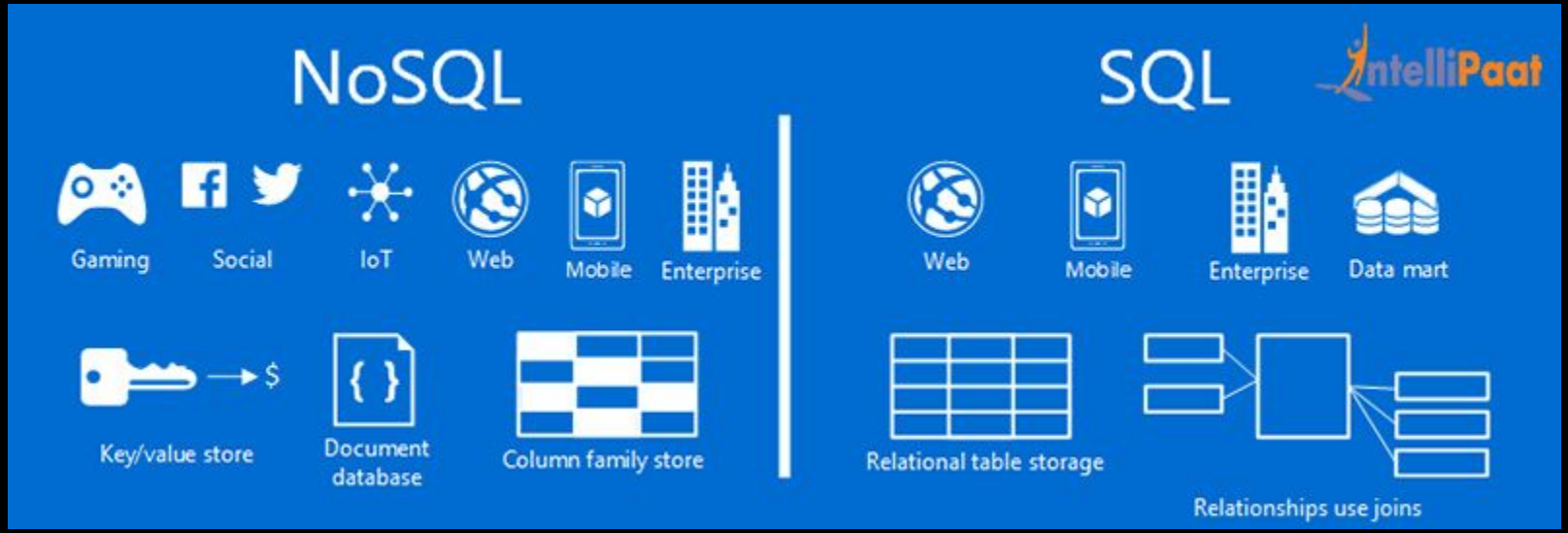
# SQL vs NoSQL


# SQL vs Non-SQL




- SQL database are primarily called Relational Databases (RDBMS). NoSQL are primarily called non-relational or distributed database
- SQL databases have predefined schema vs NoSql databases have dynamic schema for unstructured data
- SQL databases are vertically scalable vs NoSqL databases are horizontally scalable
    - *(increase hardware vs increase servers)*
- SQL databases are a good fit for the complex queries, NoSQL is not

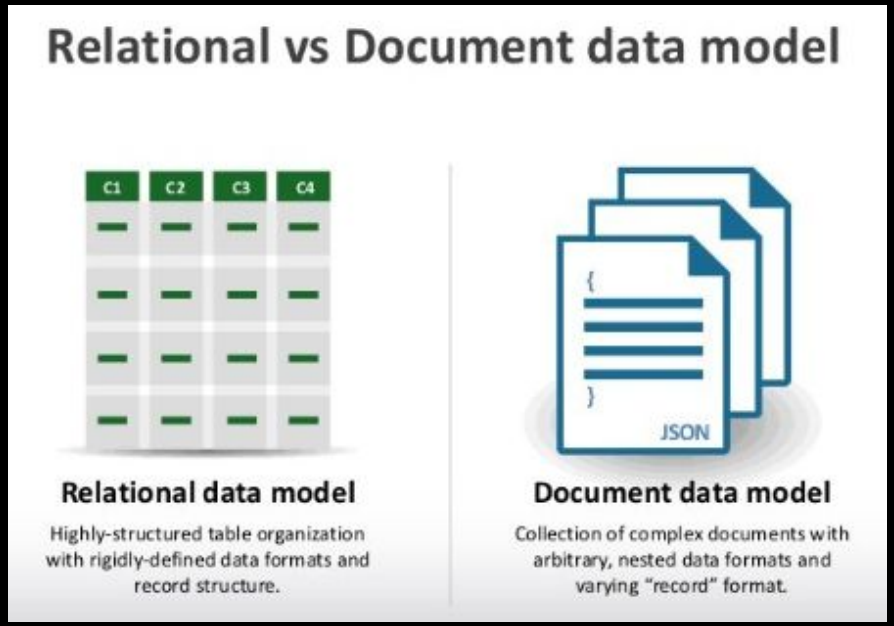
# SQL vs NOSQL




- Schema free
- Scalable
- Flexible
- Limited queries
- Emerging

- Relational Schema
- Consistent
- Rigid
- Mature
- Stable

# Relational vs Non-Relational Databases




- A non-relationship database does not incorporate the table model. Instead, data can be stored in a single document file.
- A relational database table organizes structured data fields into defined columns.

# Real World Applications

# Tinder


- The company does 1.7 billion ratings per day, which translates to 25 million matches. They have 1 million apps installed per week.
- Tinder is matchmaking via geolocation, resulting in a complex database that needs to scale and perform.
- They used MongoDB via Amazon Web Services, and Node.js.
- Mongo prototyping is easy, but scaling and maintaining is complex on distributed architectures. They had to get the best DBA consulting firms to help.

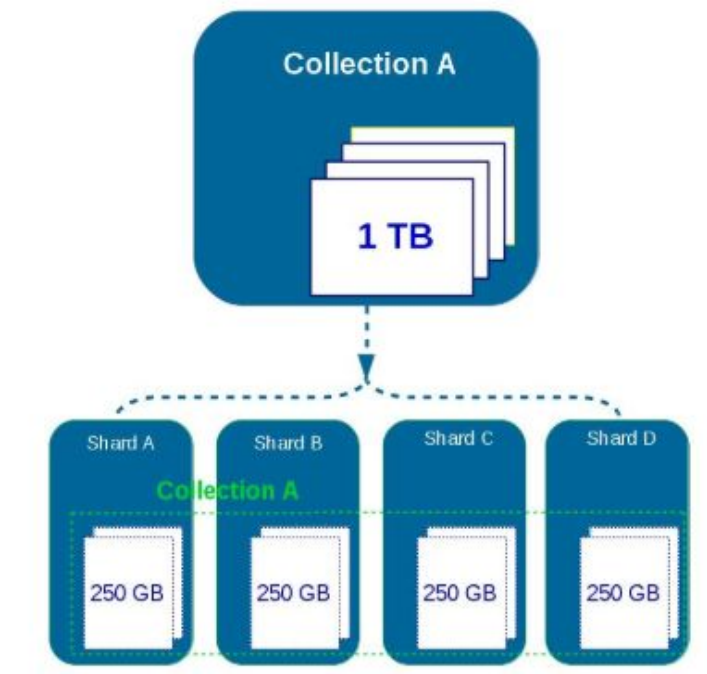
# MongoDB

# MongoDB


https://www.mongodb.com/

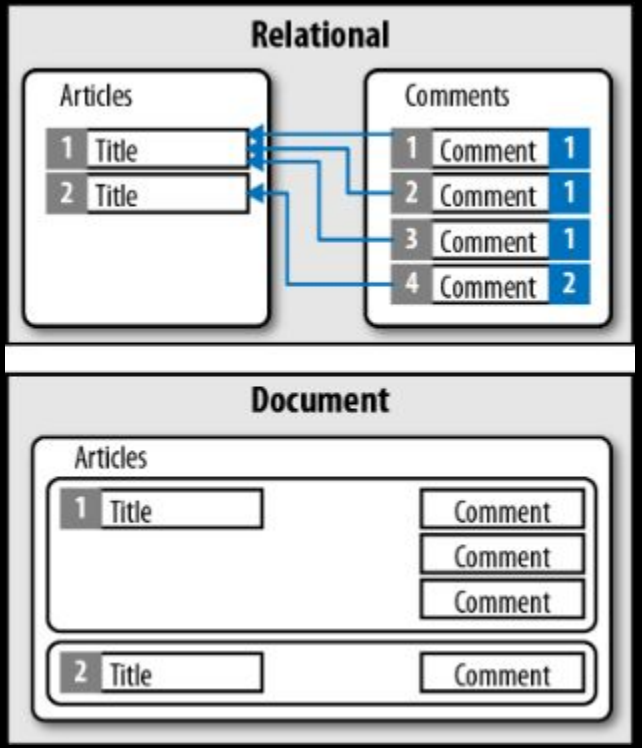
- Document-oriented NoSQL database
- JSON-like documents with dynamic schema
- Highly performant when indexing support is used
- A natural fit for Node and Express applications
- Use JSON for queries
- ** NO Table JOINS!

# MongoDB cont..




- Scalable High-Performance Open-source, Document-oriented database
- Built for Speed
- Rich Document based queries for Easy readability
- Full Index Support for High Performance
- Replication and Failover for High Availability
- Auto Sharding for Easy Scalability
    - *(type of database partitioning that separates very large databases into smaller, faster part called shards)*

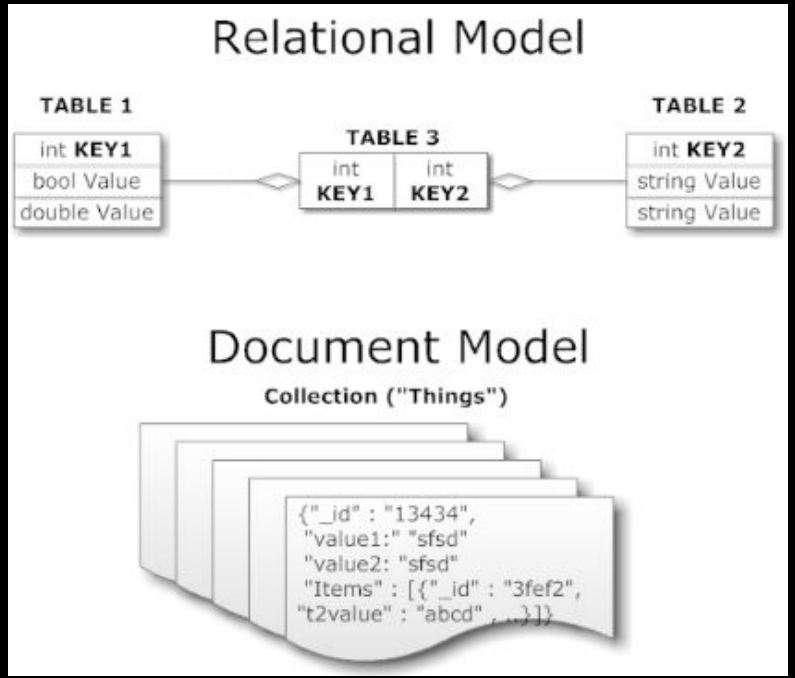
# MongoDB Recap..




- SQL was invented in the 70;s to store data.
- MongoDB stores documents (or) objects.
- Today, everyone works with objects (Python/Java/etc.)
- We need a Database to persist our objects. Why not store the objects directly?
- Embedded documents and arrays reduce the need for join.  MongoDB has no Joins and no transactions!
- MongoDB is great for Real-time data ie. Gaming.

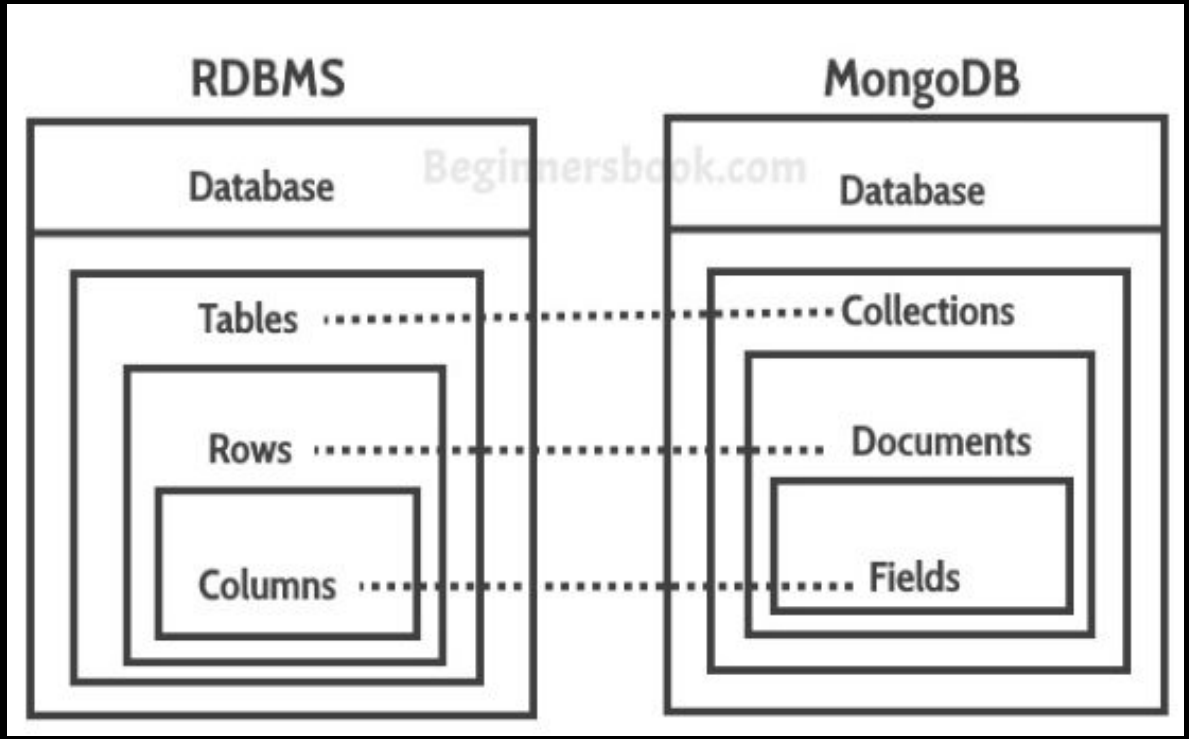
# MongoDB vs SQL

# Mongo Database



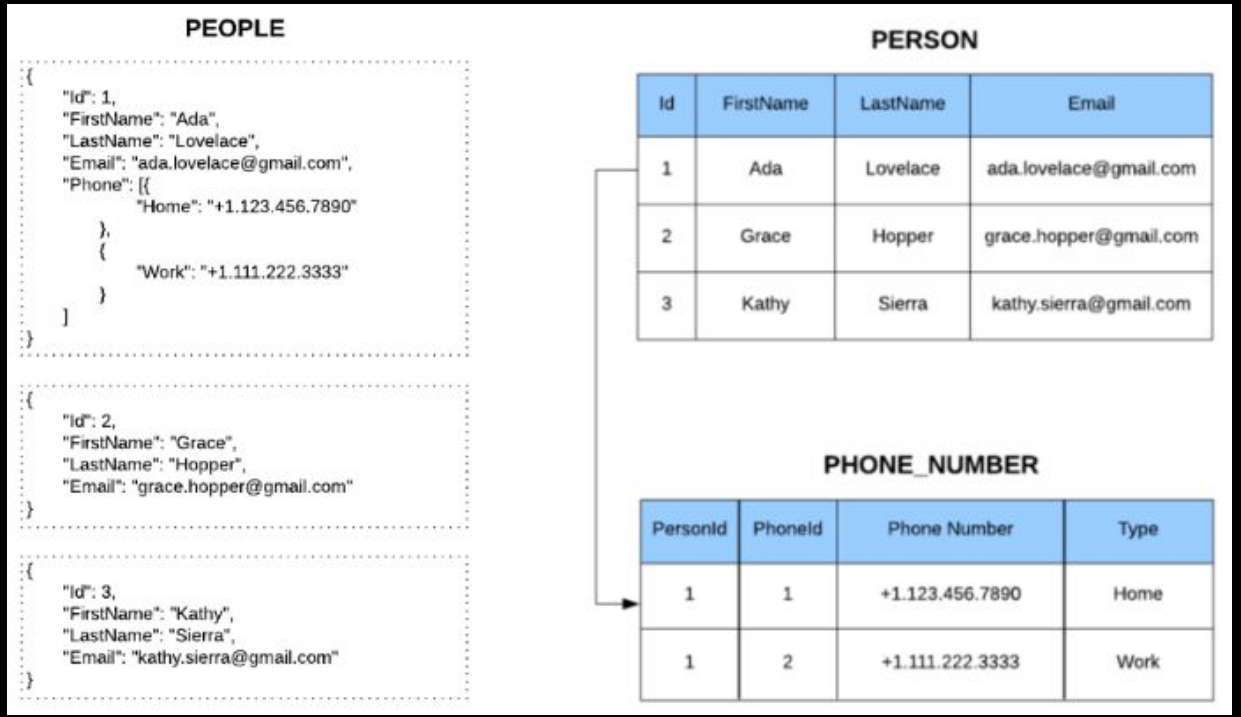

- Database in MongoDB is the same concept as Database in SQL
- Made up of Multiple Collections
- Created on-the-fly when referenced for the first time

# MongoDB mappings to RDBMS



# Documents vs Table

## NoSQL Documents vs. Relational Tables in SQL



# MongoDB Queries Are Expressive

Find all contacts with at least one home phone or hired after 2014-02-02

SQL

```
select A.did, A.lname, A.hiredate, B.type,
B.number from contact A left outer join phones B
on (B.did = A.did) where b.type = 'home' or
A.hiredate > '2014-02-02'::date
```

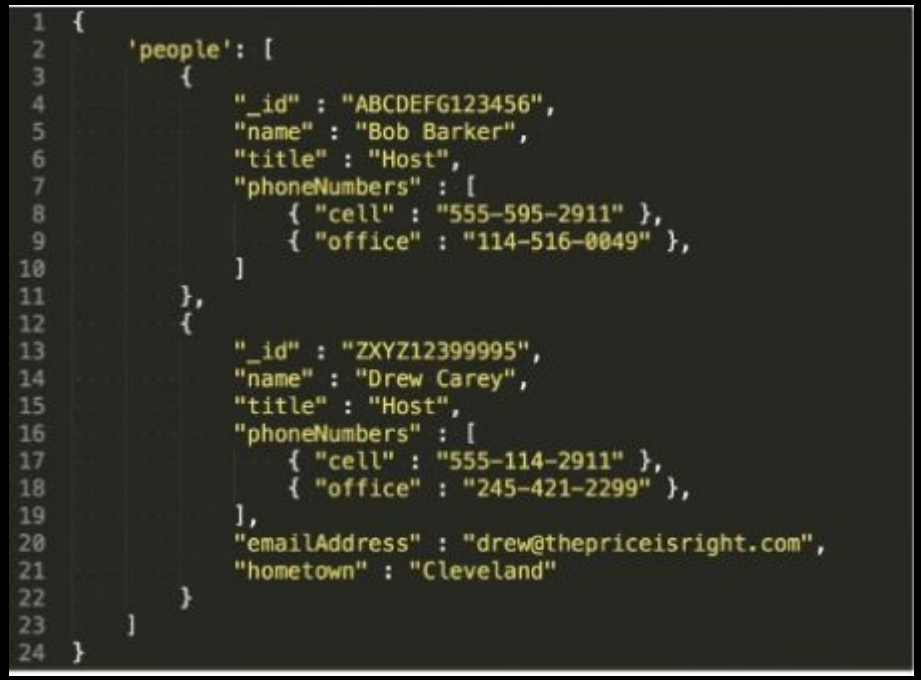
MongoDB CLI

```
db.contacts.find({"$or": [
{"phones.type":"home"},
        {"hiredate": {"$gt": new ISODate("2014-
02-02")}}
]});
```

# Documents & Collections

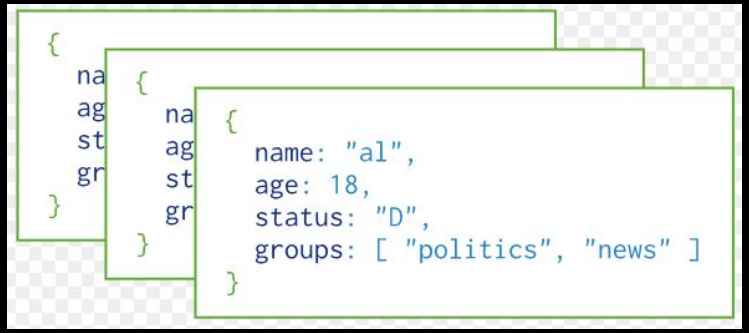
# MongoDB Document based

- A document in MongoDB, notice the JSON schema
- Treat your data more like objects.



```
{
    'people': [
        {
            "_id" : "ABCDEFG123456",
            "name" : "Bob Barker",
            "title" : "Host",
            "phoneNumbers" : [
                { "cell" : "555-595-2911" },
                { "office" : "114-516-0049" },
            ]
        },
        {
            "_id" : "ZXYZ12399995",
            "name" : "Drew Carey",
            "title" : "Host",
            "phoneNumbers" : [
                { "cell" : "555-114-2911" },
                { "office" : "245-421-2299" },
            ],
            "emailAddress" : "drew@thepriceisright.com",
            "hometown" : "Cleveland"
        }
    ]
}
```

# Mongo Collection




- A MongoDB Collection is the equivalent to a SQL Table
- Schema-less and contains Documents.
- Indexable by one/more keys.
- Created on-the-fly when referenced for the first time.
- Capped Collections: Fixed size, older records get dropped after reaching the limit.

# Mongo Document

```
{
  person: {
    first_name: "Peter",
    last_name: "Peterson",
    addresses: [
      {street: "123 Peter St"},
      {street: "504 Not Peter St"}
    ],
  }
}
```

- A MongoDB Document is the equivalent to a SQL Record/Row
- Stored in a Collection
- Can have _id key - works like Primary keys in SQL
- Document storage in BSON (Binary form of JSON)