

# NoSQL and MongoDB

## Introduction to NoSQL Databases

Eng. Hany Saad  
SD Dept.  
ITI – Assiut Branch



N★SQL

# What's NoSQL?

“Next Generation Databases mostly addressing some of the points: being non-relational, distributed, open-source and horizontal scalable. ... schema-free, easy replication support, simple API, eventually consistent ...”


- [nosql-database.org](http://nosql-database.org)

## Not Only SQL

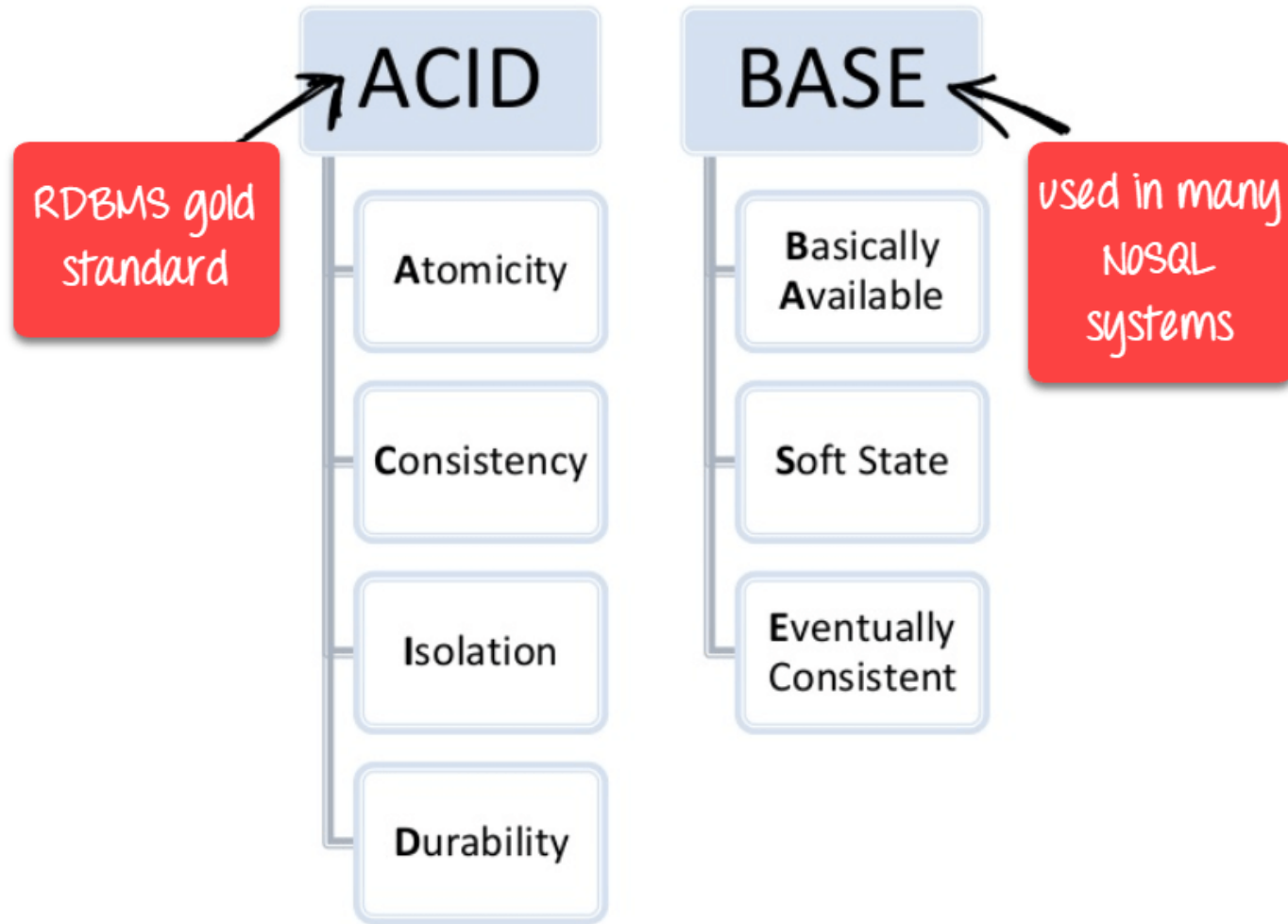
It's not about flaming SQL.  
It's about opening our minds to new technologies.

# What's NoSQL (cont.)?

---


- **NoSQL DEFINITION:** Next Generation Databases mostly addressing some of the points: being **non-relational**, **distributed**, **open-source** and **horizontally scalable**.
  - The original intention has been **modern web-scale databases**. The movement began early 2009 and is growing rapidly.
  - Often more characteristics apply such as: **schema-free**, **easy replication support**, **simple API**, **eventually consistent/BASE** (not ACID), a **huge amount of data** and more.
  - So the misleading term "*nosql*", originally referring to "**non SQL**" or "**non relational**" (the community now translates it mostly with "**not only sql**").
  - Generally, NoSQL databases are structured in a **key-value pair**, **graph database**, **document-oriented** or **column-oriented structure**.
- 

# ACID vs. BASE



# ACID vs. BASE (Cont.)

---

- **Atomicity:** The database transaction must completely succeed or completely fail. Partial success is not allowed.
  - **Consistency:** During the database transaction, the RDBMS progresses from one valid state to another. The state is never invalid.
  - **Isolation:** The client's database transaction must occur in isolation from other clients attempting to transact with the RDBMS.
  - **Durability:** The data operation that was part of the transaction must be reflected in *nonvolatile storage* (computer memory that can retrieve stored information even when not powered — like a hard disk) and persist after the transaction successfully completes. Transaction failures cannot leave the data in a partially committed state.
- 




# ACID vs. BASE (Cont.)

---

- **Basically Available:** The system is guaranteed to be available for querying by all users. (No isolation here.)
- **Soft State:** The values stored in the system may change because of the eventual consistency model, as described in the next bullet.
- **Eventually Consistent:** As data is added to the system, the system's state is gradually replicated across all nodes. For example, in Hadoop, when a file is written to the HDFS, the replicas of the data blocks are created in different data nodes after the original data blocks have been written. For the short period before the blocks are replicated, the state of the file system isn't consistent.

# Why NoSQL?

---

- Over decades and decades of software development, we have been using databases in form of SQL (Structured Query Language) where we store our data in **relational tables**.
  - In recent years with the tremendous rise in use of internet and Web 2.0 applications, the databases have grown into thousands and thousands of terabytes. Applications such as Facebook, Google, Amazon, Whatsapp, etc. gave rise to an entire **new era of database management which follows approach of simple design, speed and faster scaling than the traditional databases**.
  - Such databases are used in **big data, massive real time applications and analytics**.
- 



# NoSQL Vs. RDBMS...

---

- **Challenges of RDBMS**
  - RDBMS assumes a well-defined structure of data and assumes that the data is largely uniform.
  - It needs the schema of your application and its properties (columns, types, etc.) to be defined up-front before building the application. This does not match well with the agile development approaches for highly dynamic applications.
  - As the data starts to grow larger, you have to scale your database vertically, i.e. adding more capacity to the existing servers.

# NoSQL Vs. RDBMS (Cont.)...

---

- **Benefits of NoSQL over RDBMS**

- **Schema Less:**

- NoSQL databases being schema-less do not define any strict data structure.

- **Dynamic and Agile:**

- NoSQL databases have good tendency to grow dynamically with changing requirements. It can handle structured, semi-structured and unstructured data.

- **Scales Horizontally:**

- In contrast to SQL databases which scale vertically, NoSQL scales horizontally by adding more servers and using concepts of sharding and replication. This behavior of NoSQL fits with the cloud computing services such as Amazon Web Services (AWS) which allows you to handle virtual servers which can be expanded horizontally on demand.

- **Better Performance:**


- All the NoSQL databases claim to deliver better and faster performance as compared to traditional RDBMS implementations.



# NoSQL Vs. RDBMS (Cont.)...

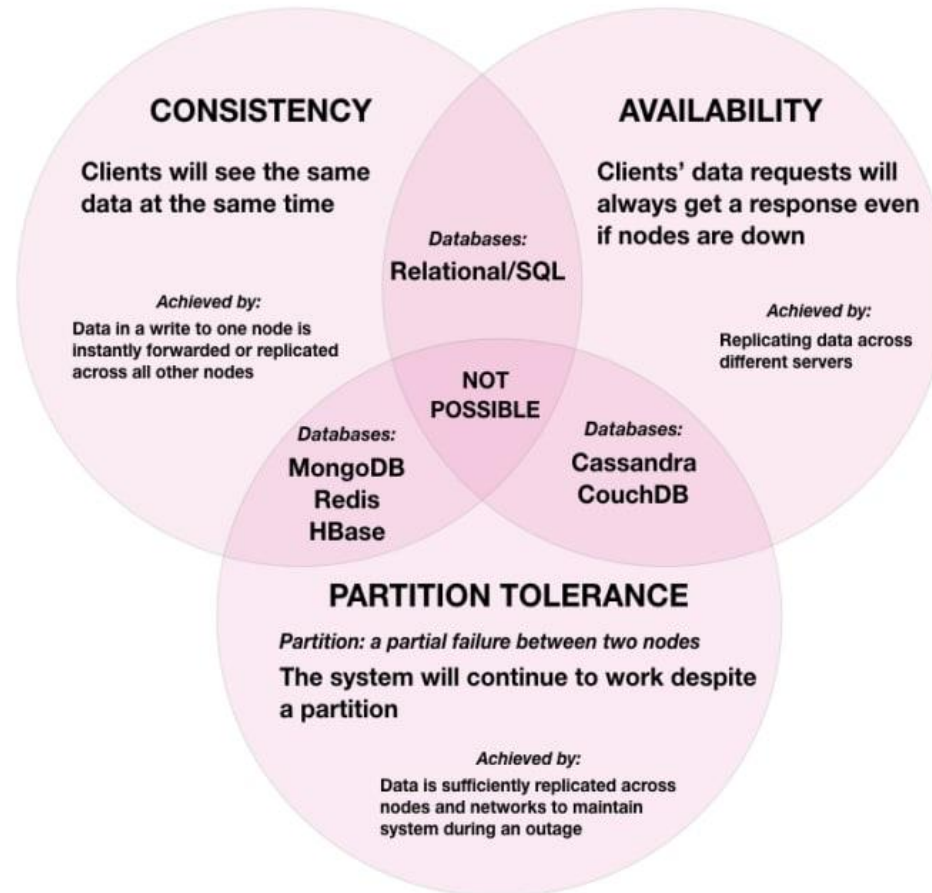
---

- **NoSQL Limitations:**

- Since NoSQL is an entire set of databases (and not a single database), the limitations differ from database to database.
  - Some of these databases do not support ACID (*Atomicity, Consistency, Isolation, Durability*) transactions while some of them might be lacking in reliability.
  - But each one of them has their own strengths due to which they are well suited for specific requirements.
  - No standardization.
  - Limited query capabilities (so far).
- 

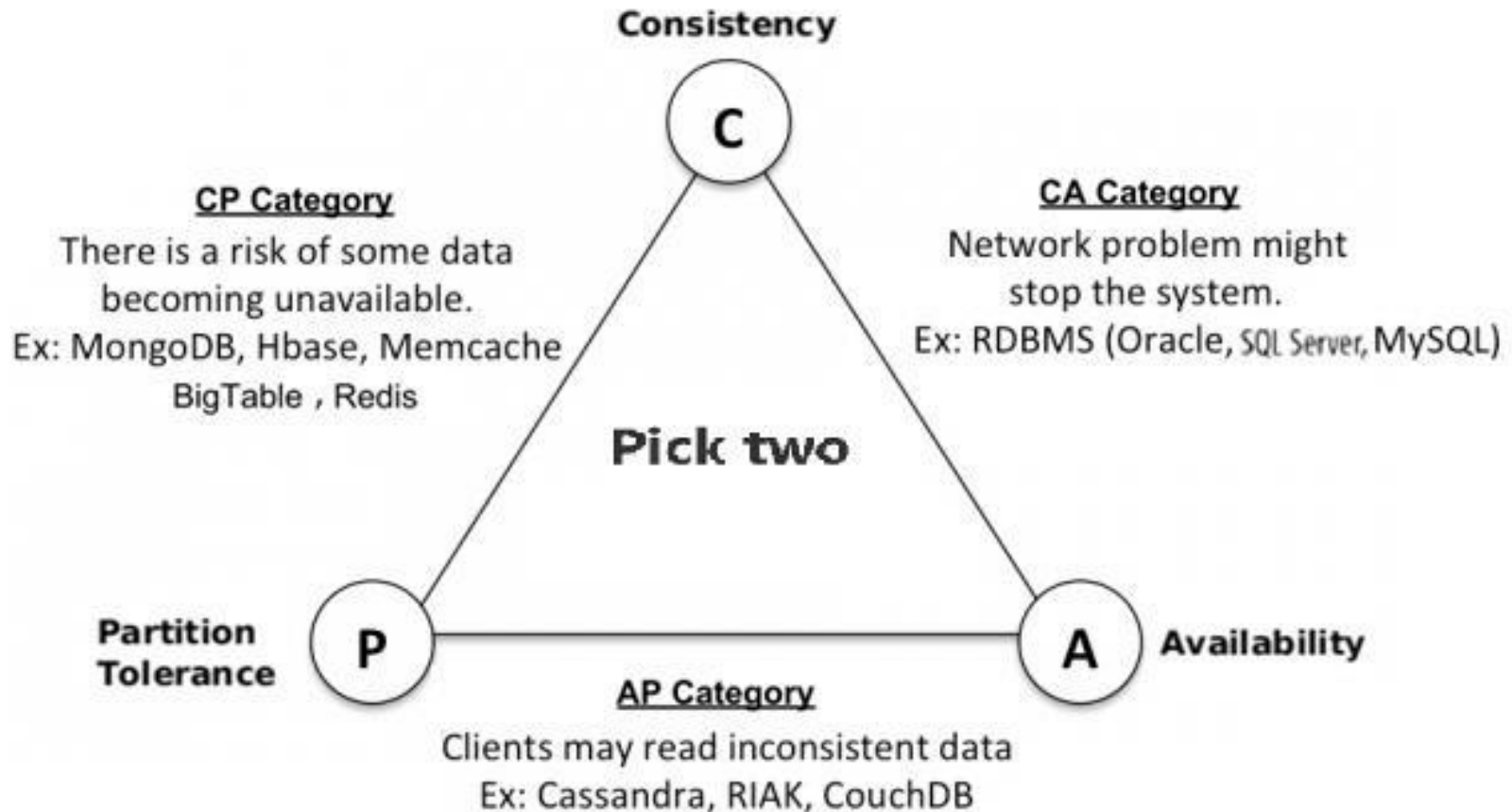
# CAP Theorem

- The CAP Theorem states that it is impossible for a distributed computer system to simultaneously provide all three of the following guarantees:



# CAP Theorem (Cont.)

---



# NoSQL Vs. RDBMS (Cont.)...









Entity	SQL Databases	NoSQL Databases
Type	One Type (SQL) with Minor Variation	Many Types (Document, Ke-Value, Tabular, Graph)
Development	1970	2000
Examples	Oracle, MSSQL, DB2 etc.	MongoDB, Cassandra, Hbase, Neo4J
Schemas	Fixed	Dynamic
Scaling	Vertical	Horizontal
Dev Model	Mix	Open Source
Consistency	Follow ACID	Follow BASE

	SQL	NoSQL
Database Type	Relational Databases	Non-relational Databases / Distributed Databases
Structure	Table-based	<ul style="list-style-type: none"> <li>• Key-value pairs</li> <li>• Document-based</li> <li>• Graph databases</li> <li>• Wide-column stores</li> </ul>
Scalability	Designed for scaling up vertically by upgrading one expensive custom-built hardware	Designed for scaling out horizontally by using shards to distribute load across multiple commodity (inexpensive) hardware
Strength	<ul style="list-style-type: none"> <li>• Great for highly structured data and don't anticipate changes to the database structure</li> <li>• Working with complex queries and</li> </ul>	<ul style="list-style-type: none"> <li>• Pairs well with fast paced, agile development teams</li> <li>• Data consistency and integrity is not top priority</li> </ul>



# NoSQL Types

← Increasing Data Complexity

Type	Examples
Key-Value Store	 
Wide Column Store	 
Document Store	 
Graph Store	 

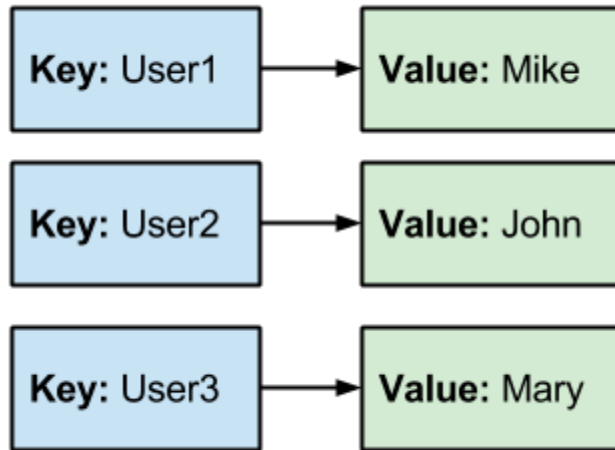
# NoSQL Types - Key Value Databases

---

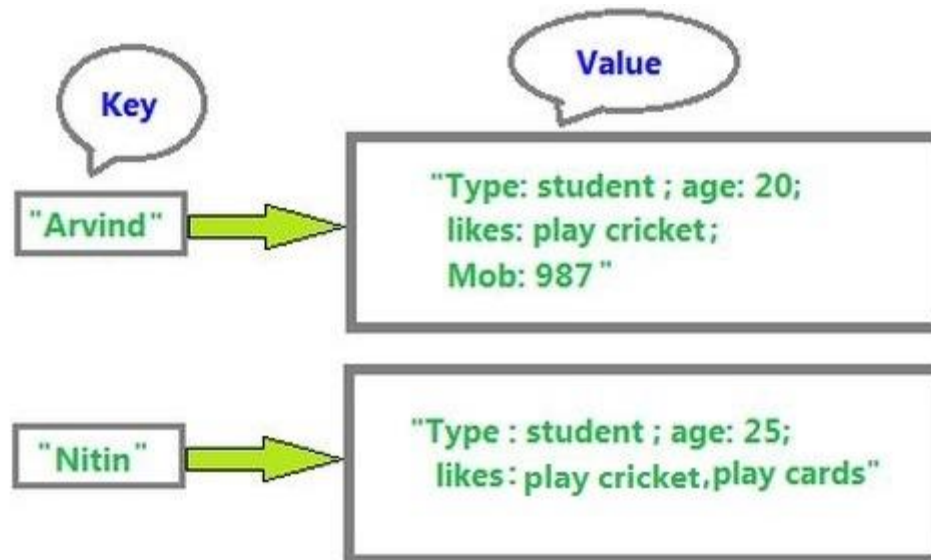
- **Key Value Databases**

- The key of a key/value pair is a unique value in the set and can be easily looked up to access the data.
- Key/value pairs are of varied types: some keep the data **in memory** and some provide the capability to **persist the data to disk**.
- Key-value databases can be applied to many scenarios. For example, key-value stores can be useful for storing things such as the following: Shopping cart contents, Product reviews, Product details, Session information, user profiles.
- Examples: Tokyo Cabinet/Tyrant, Redis, Voldemort, Oracle BDB, Amazon SimpleDB, Riak.

# NoSQL Types - Key Value Databases (Cont.)



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



# NoSQL Types - Column Based Databases

---

- **Column Based Databases**

- The column-oriented storage allows data to be stored effectively.
- Were created to store and process very large amounts of data distributed over many machines.
- There are still keys but they point to multiple columns. The columns are arranged by column family.
- Businesses Use Wide Column Databases to Handle:
  - High volume of data
  - Extreme write speeds with relatively less velocity reads
  - Data extraction by columns using row keys.
- Like: Sensor Logs [[Internet of Things \(IOT\)](#)], Geographic information, etc.
- Examples: [Cassandra](#), [HBase](#).

# NoSQL Types - Column Based Databases (Cont.)

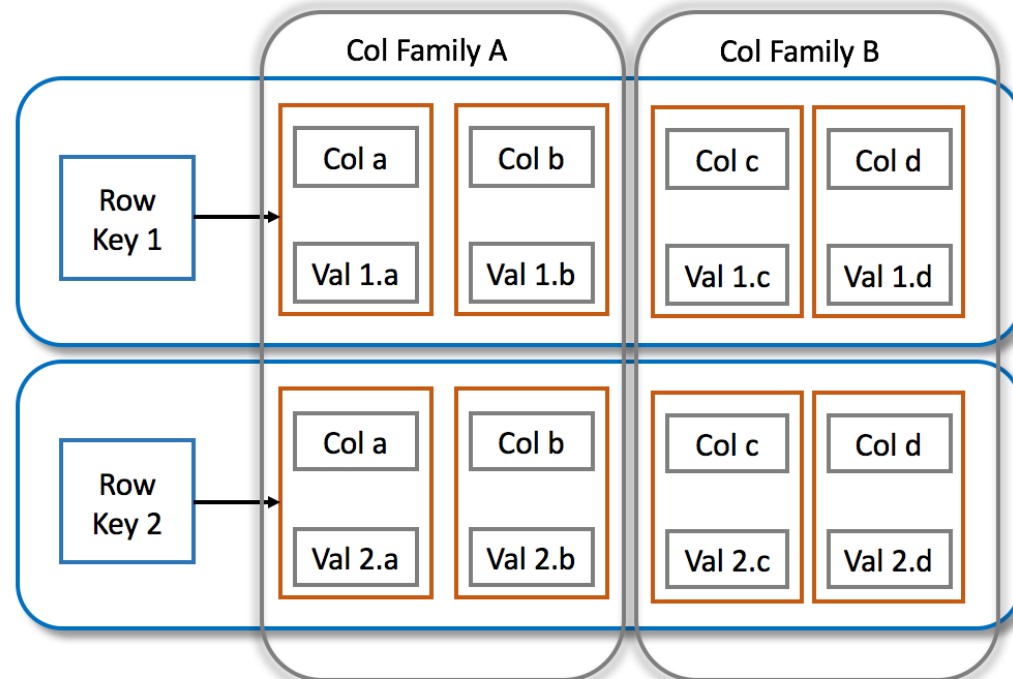
## Row-Oriented vs Column-Oriented

Row-oriented: rows stored sequentially in a file

Key	Fname	Lname	State	Zip	Phone	Age	Sales
1	Bugs	Bunny	NY	11217	(123) 938-3235	34	100
2	Yosemite	Sam	CA	95389	(234) 375-6572	52	500
3	Daffy	Duck	NY	10013	(345) 227-1810	35	200
4	Elmer	Fudd	CA	04578	(456) 882-7323	43	10
5	Witch	Hazel	CA	01970	(567) 744-0991	57	250

Column-oriented: each column is stored in a separate file  
Each column for a given row is at the same offset.

Key	Fname	Lname	State	Zip	Phone
1	Bugs	Bunny	NY	11217	(123) 938-3235
2	Yosemite	Sam	CA	95389	(234) 375-6572
3	Daffy	Duck	NY	10013	(345) 227-1810
4	Elmer	Fudd	CA	04578	(456) 882-7323
5	Witch	Hazel	CA	01970	(567) 744-0991



# NoSQL Types - Document Oriented Databases

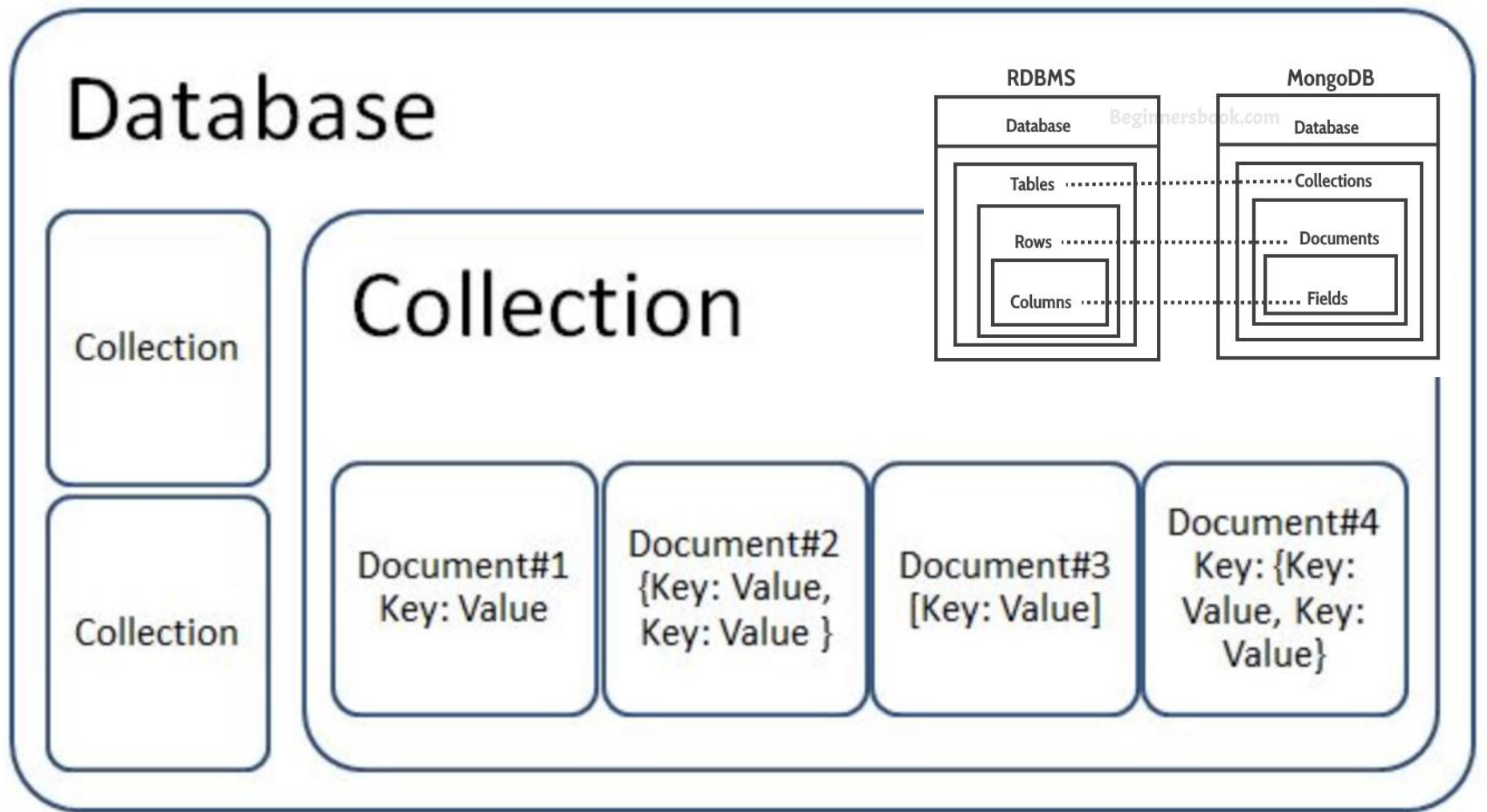
---

- **Document Oriented Databases**

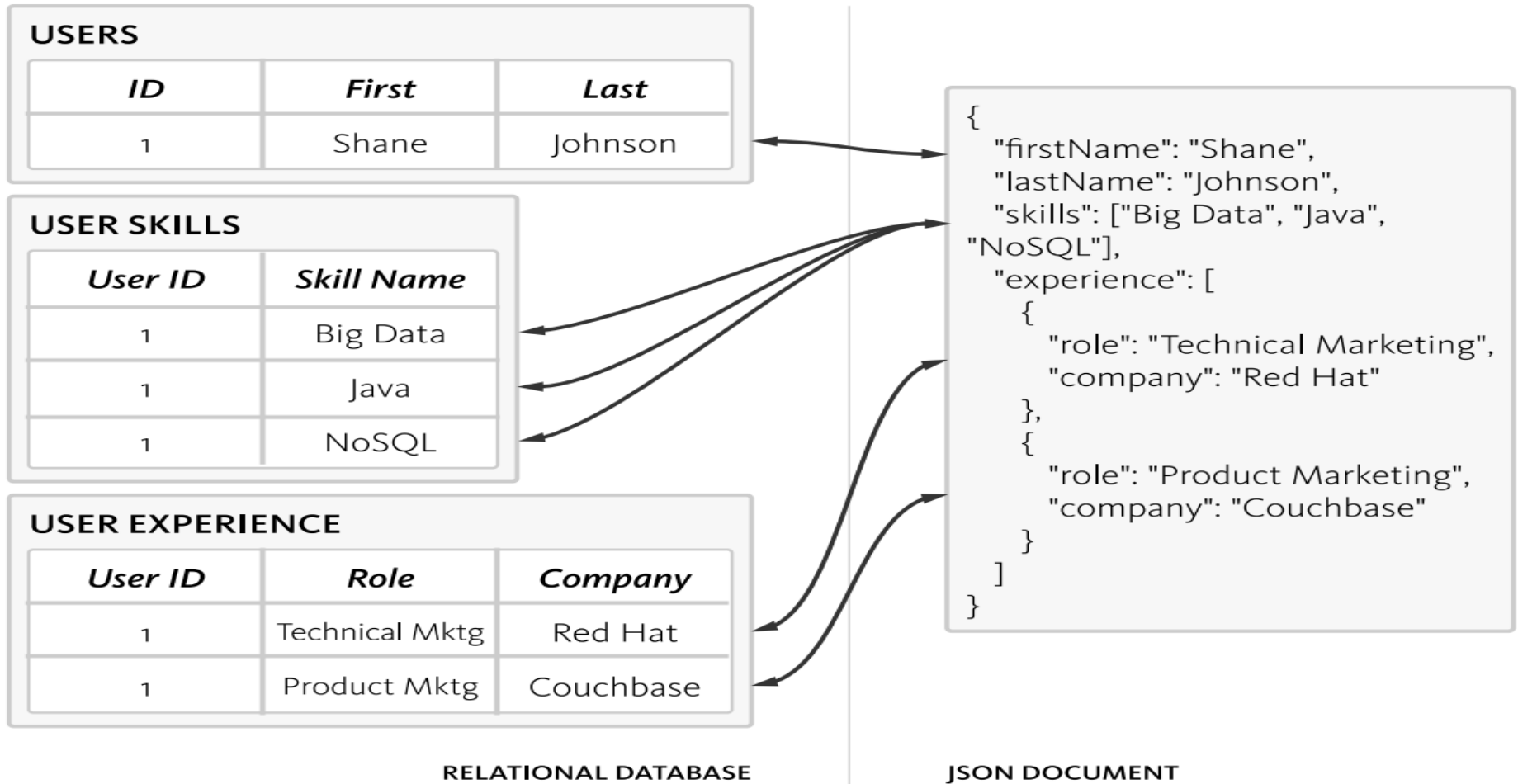
- These were inspired by Lotus Notes and are similar to key-value stores.
- The model is basically versioned documents that are collections of other key-value collections.
- The semi-structured documents are stored in formats like JSON.
- Document databases are essentially the next level of Key/value, allowing nested values associated with each key.
- Document databases support querying more efficiently.
- Examples: CouchDB, MongoDb.
- In fact, **MongoDB** has become one of the most popular NoSQL databases.



# NoSQL Types - Document Oriented Databases (Cont.)



# NoSQL Types - Document Oriented Databases (Cont.)



# NoSQL Types - Graph Based Databases

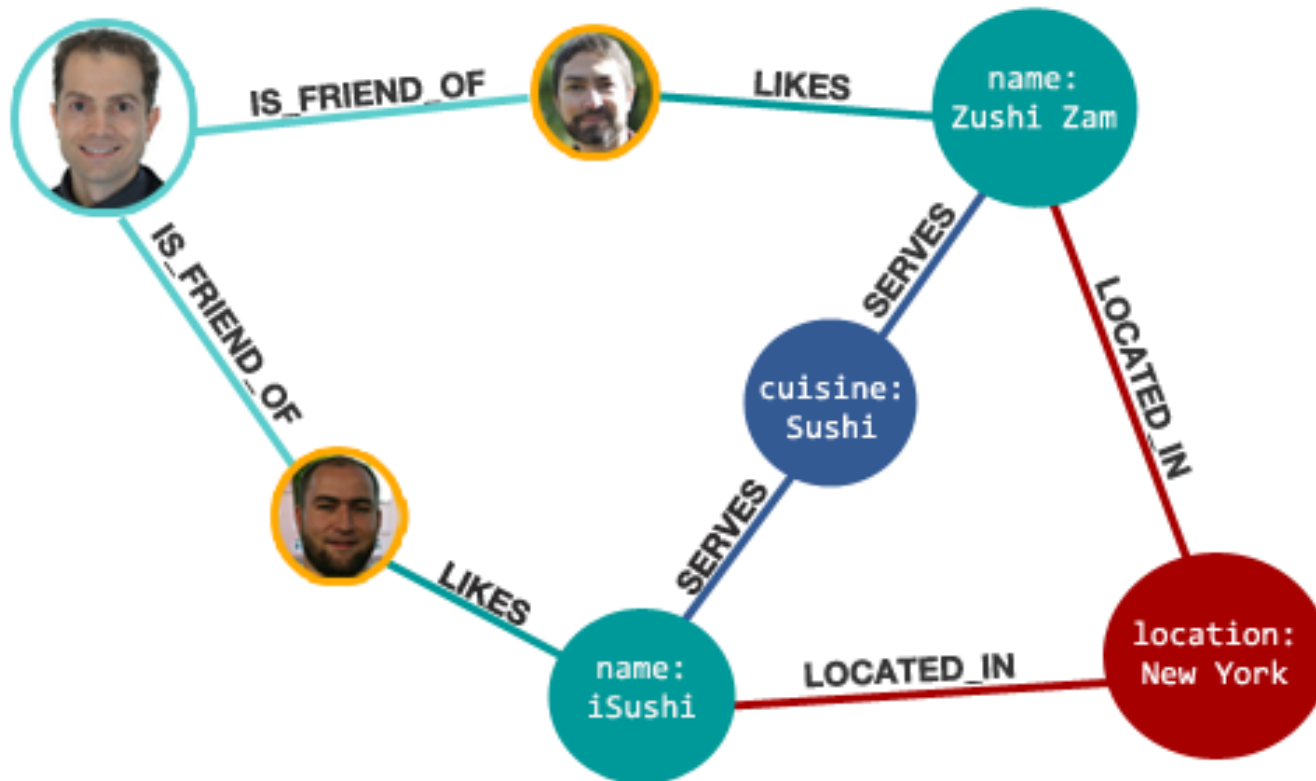
---

- **Graph Based Databases**

- Instead of tables of rows and columns and the rigid structure of SQL, a flexible graph model is used which, again, can scale across multiple machines.
- A graph database uses graph structures with nodes, edges, and properties to represent and store data.
- By definition, a graph database is any storage system that provides index-free adjacency.
- This means that every element contains a direct pointer to its adjacent element and no index lookups are necessary.
- Examples: Neo4J, InfoGrid, Infinite Graph.

# NoSQL Types - Graph Based Databases (Cont.)

---



# NoSQL Types - Summery

## Flavors of NoSQL

Key/Value  
Volatile

key	value
123	123 Main St.
126	(805) 477-3900

Key/Value  
Persistent

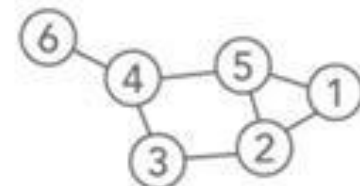
Wide-Column

1	{Things: {A: foo, B: bar, C: baz}}
2	{Things: {C: Apple, E: Banana}, People: {A: Pat}}
3	{Languages: {A: C, B: Java, C: Python}}

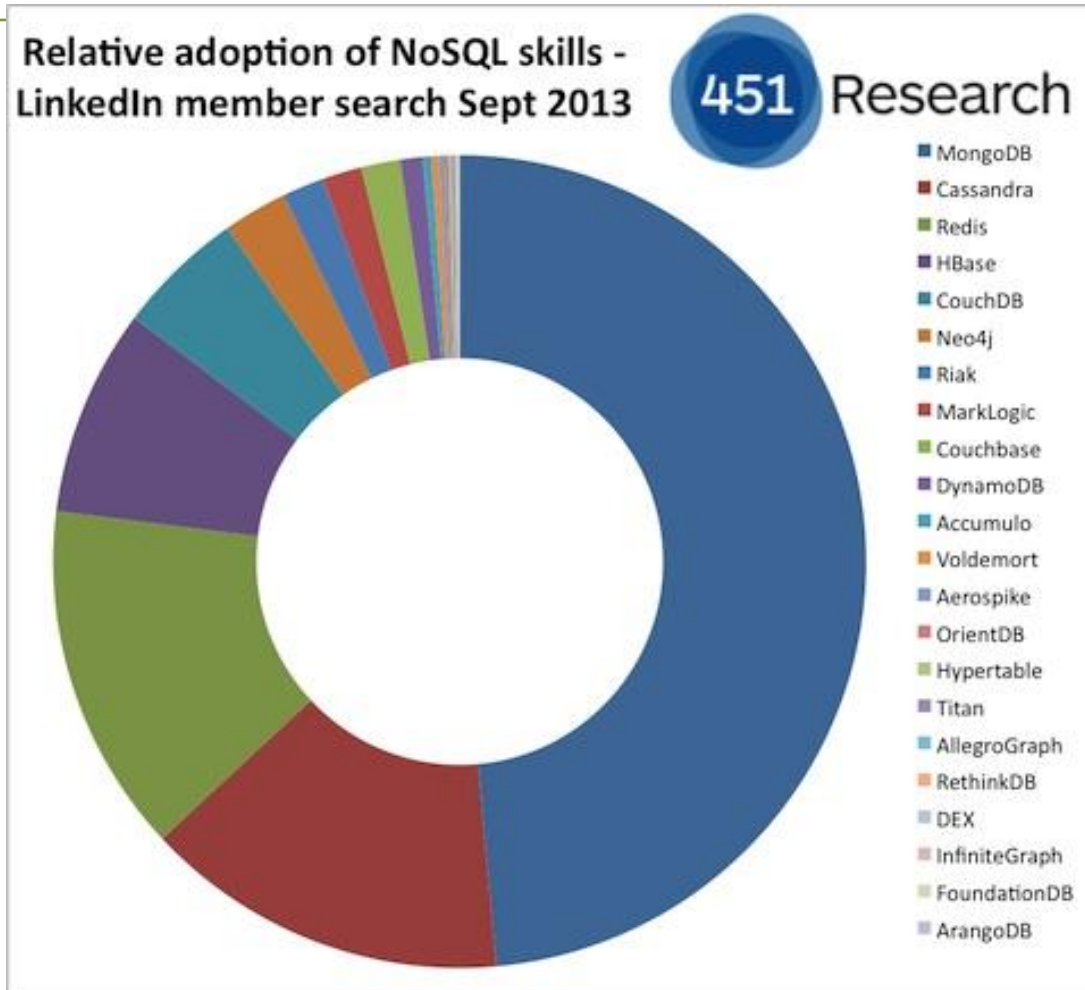
Document

```
{ "user": {  
  "id": "123",  
  "name": "Benjamin",  
  "addresses": [  
    { "city": "Paris", "country": "France" },  
    { "city": "Atlanta", "country": "USA" },  
  ]  
}
```

Graph




# Top NoSQL DBs





# Conclusion

---

- A lot of organizations today are adapting to such databases for their huge datasets and high-scale applications.
  - This shows that NoSQL is definitely going to be the next big thing in web and database technologies which has the potential to break the years long legacy of RDBMS.
- 

A decorative horizontal band at the top of the slide featuring a complex, overlapping geometric pattern of various shades of green, creating a sense of depth and movement.

# Thanks...

A decorative horizontal band at the bottom of the slide, similar to the one at the top, featuring a green geometric pattern.

**Eng. Hany Saad**  
**SD Dept.**  
**ITI – Assiut Branch**