

NoSQL

(version 2025)

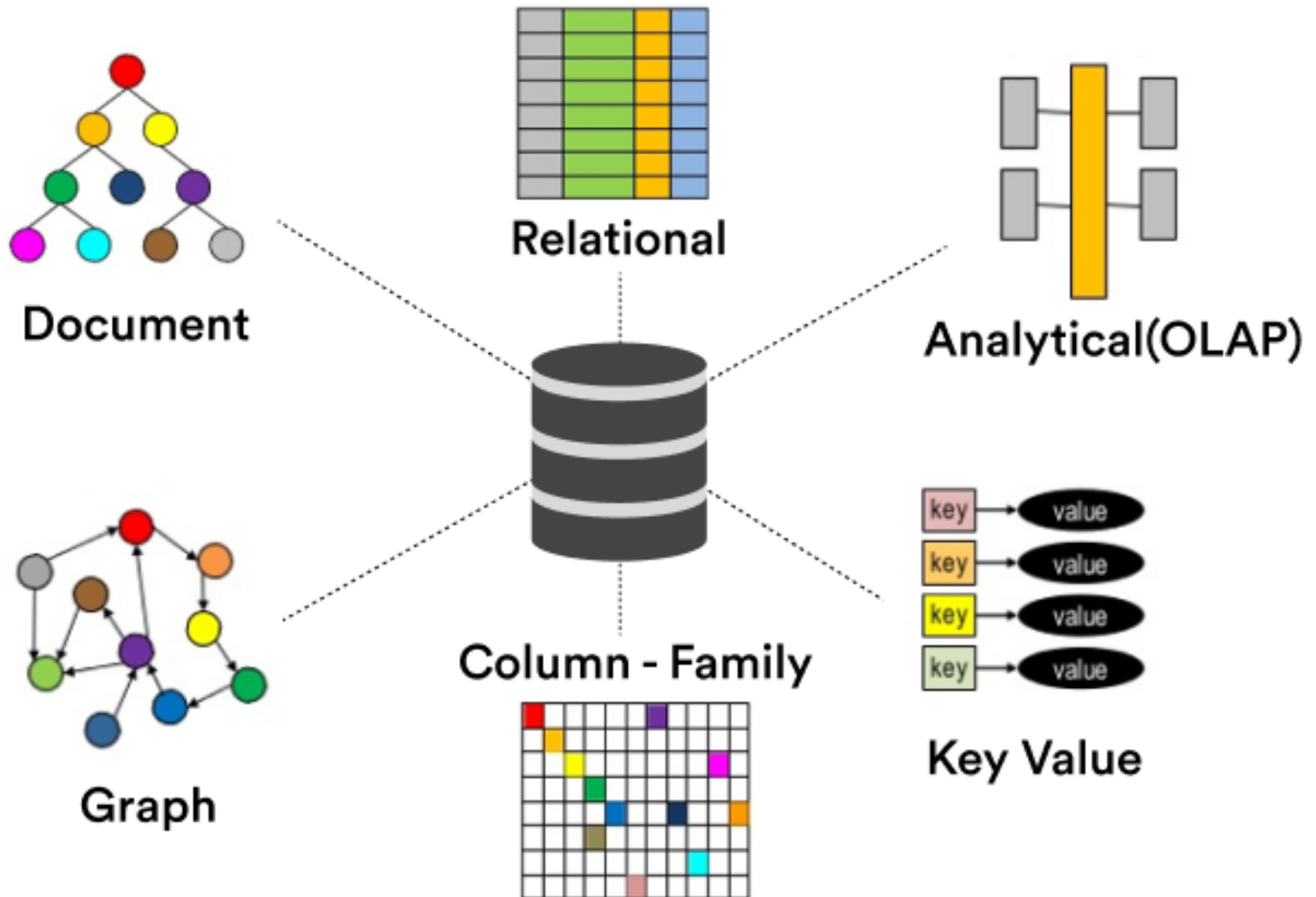
Carlos J. costa

DBMS

- Database Management System
- Software used to manage databases
- Create a database
- Modify a database
- Delete databases
- Insert data into the database
- Delete data from the database



Database Management System



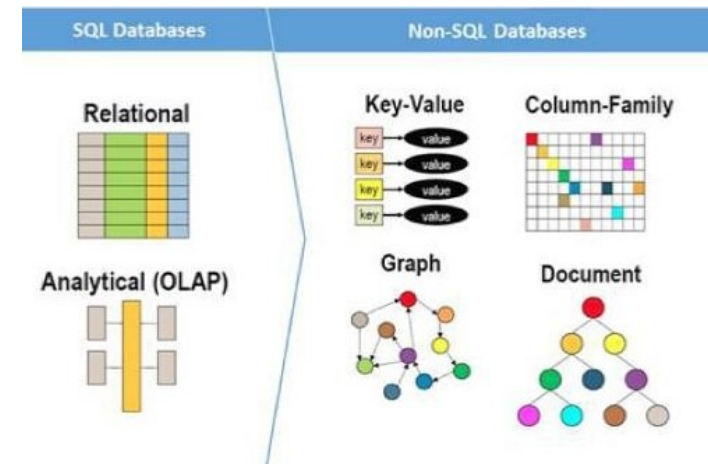
All Database Rating

Top 40 Results

Submission ID	Description	Database Rating
⌵ ID000573	(MariaDB) Amazon Web Services - db.m5.xlarge	998
⌵ ID000581	(MySQL) Amazon Web Services - db.m5.xlarge	978
⌵ ID000571	(MariaDB) Amazon Web Services - db.m5.xlarge	974
⌵ ID000582	(MySQL) Amazon Web Services - db.m5.xlarge	972
⌵ ID000561	(MySQL) Amazon Web Services - db.m5.xlarge	959
⌵ ID000598	(SQL Server) Amazon Web Services - db.m5d.xlarge	954
⌵ ID000560	(MySQL) Amazon Web Services - db.m5.xlarge	924
⌵ ID000597	(SQL Server) Amazon Web Services - db.m5d.xlarge	906
⌵ ID000562	(MySQL) Amazon Web Services - db.m5.large	902
⌵ ID000577	(PostgreSQL) Amazon Web Services - db.m5.xlarge	871
⌵ ID000593	(SQL Server) Amazon Web Services - db.m5d.large	839

NoSQL

- Next Generation Databases mostly addressing some of the points:
 - being non-relational,
 - distributed,
 - open-source and
 - horizontal scalable.
- The original intention has been modern web-scale databases.



NoSQL

- The movement began early 2009 and is growing rapidly.
- Often more characteristics apply as:
 - schema-free,
 - easy replication support,
 - simple API,
 - eventually consistent / BASE (not ACID),
 - a huge data amount, and more.

Relational Databases: ACID Properties

- **Atomic**
 - All of the work in a transaction completes (commit) or none of it completes
- **Consistent**
 - A transaction transforms the database from one consistent state to another consistent state.
 - Consistency is defined in terms of constraints.
- **Isolated**
 - The results of any changes made during a transaction are not visible until the transaction has committed.
- **Durable**
 - The results of a committed transaction survive failures



NoSQL: BASE Transactions

- Acronym opposite of ACID
 - **B**asically **A**vailable,
 - **S**oft state (State of the system may change over time)
 - **E**ventually Consistent (asynchronous propagation)

Brewer's CAP Theorem

A distributed system can support only two of the following characteristics:

- Consistency

- All replicas contain the same version of data
- Client always has the same view of the data (no matter what node)

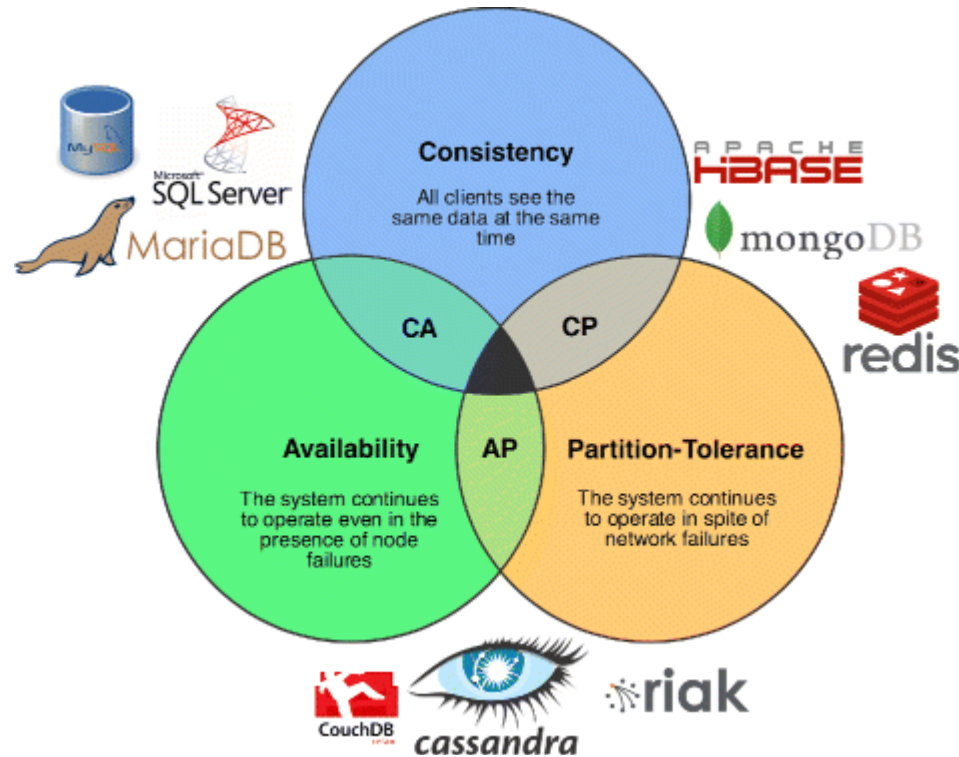
- Availability

- Systems remains operational on failing nodes
- All clients can always read and write

- Partition tolerance

- Multiple entry points
- System remains operational on system communication malfunction
- System works well across physical network partitions

Brewer's CAP Theorem

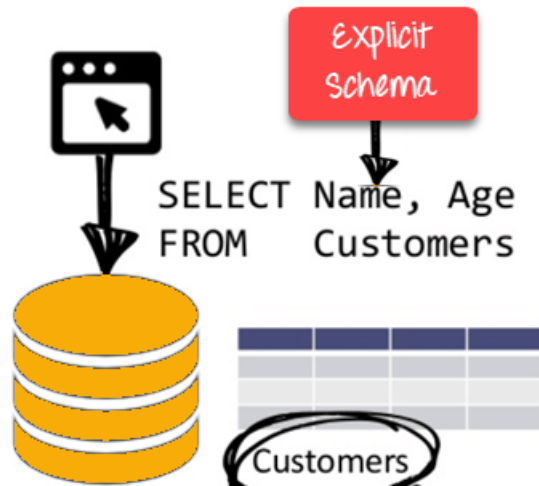


Brewer's CAP Theorem

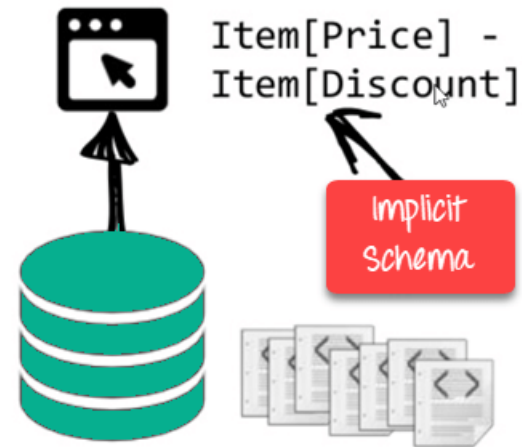
- What the CAP theorem really says:
 - If you cannot limit the number of faults and requests can be directed to any server and you insist on serving every request you receive then you cannot possibly be consistent
- How it is interpreted:
 - You must always give something up: consistency, availability or tolerance to failure and reconfiguration

RDBMS vs NoSQL

RDBMS:



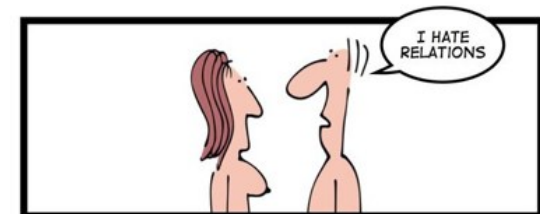
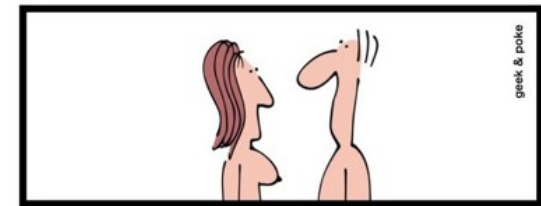
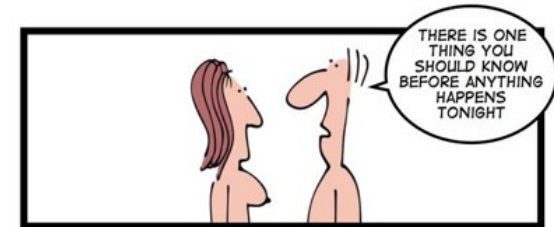
NoSQL DB:



Taxonomy of NoSQL

- Key-Value
- Graph Database
- Document-oriented
- Column Family

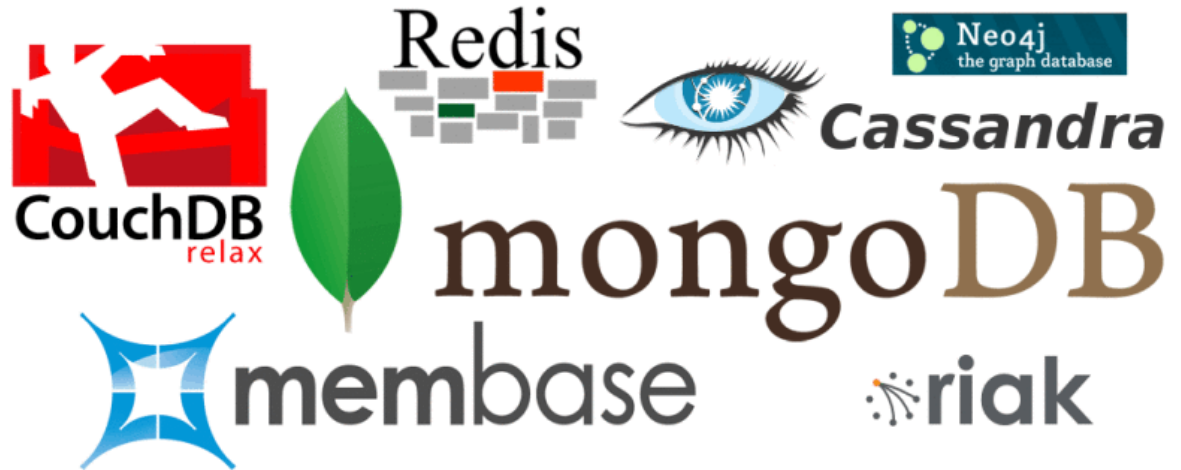
The Hard Life of a NoSQL Coder



Part 1: The Outing

<http://nosql-database.org/>

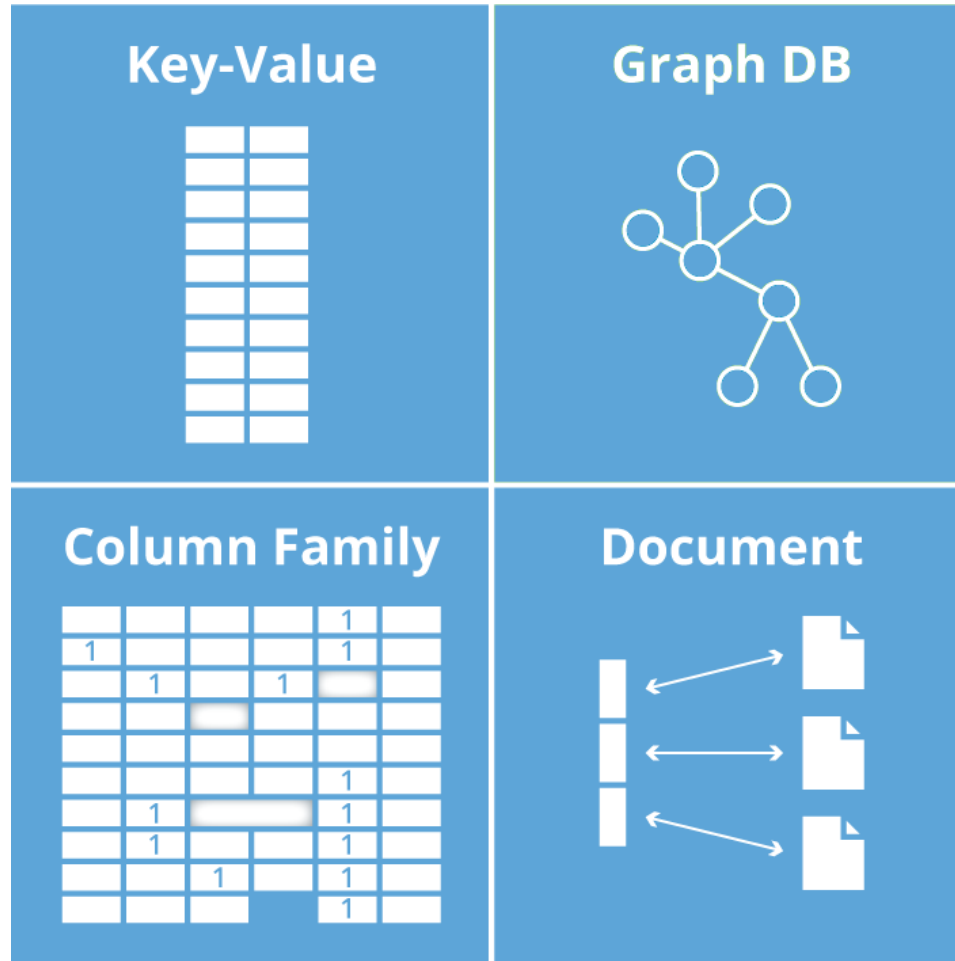
Taxonomy of NoSQL



- Key-Value
- Graph Database
- Document-oriented
- Column Family

<http://nosql-database.org/>

Taxonomy of NoSQL



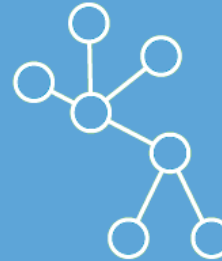
Taxonomy of NoSQL

Key-Value – is
a hash table of
keys

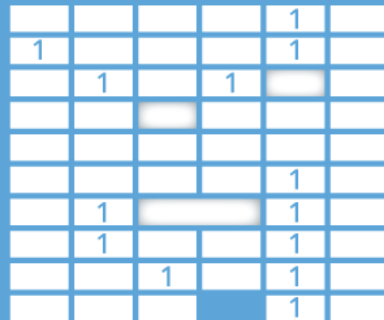
Key-Value



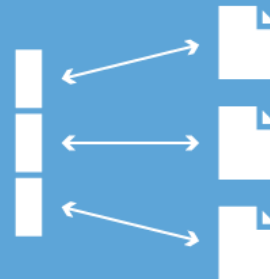
Graph DB



Column Family



Document



Taxonomy of NoSQL

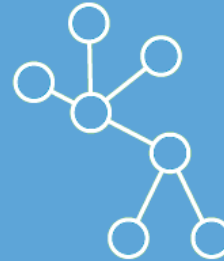
Key-Value



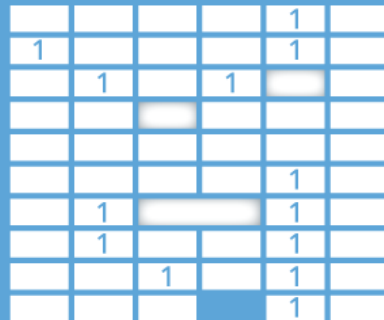
Key-Value



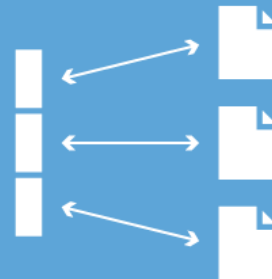
Graph DB



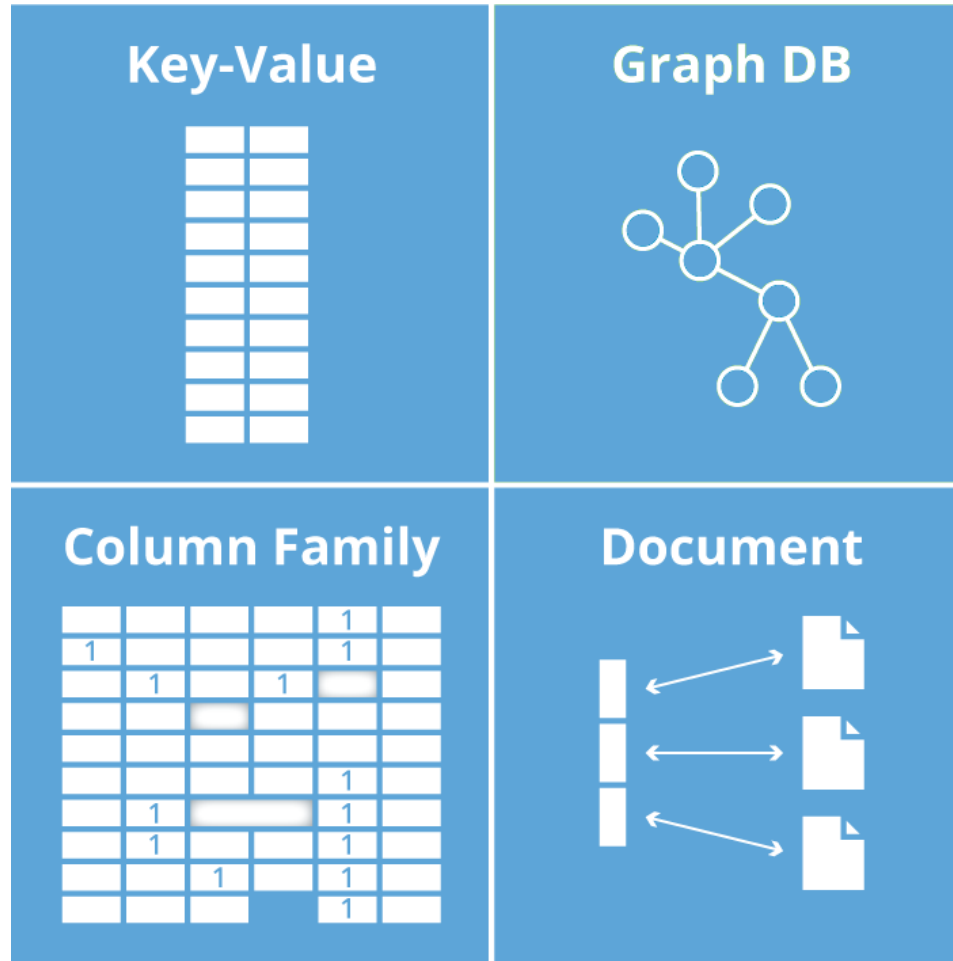
Column Family



Document

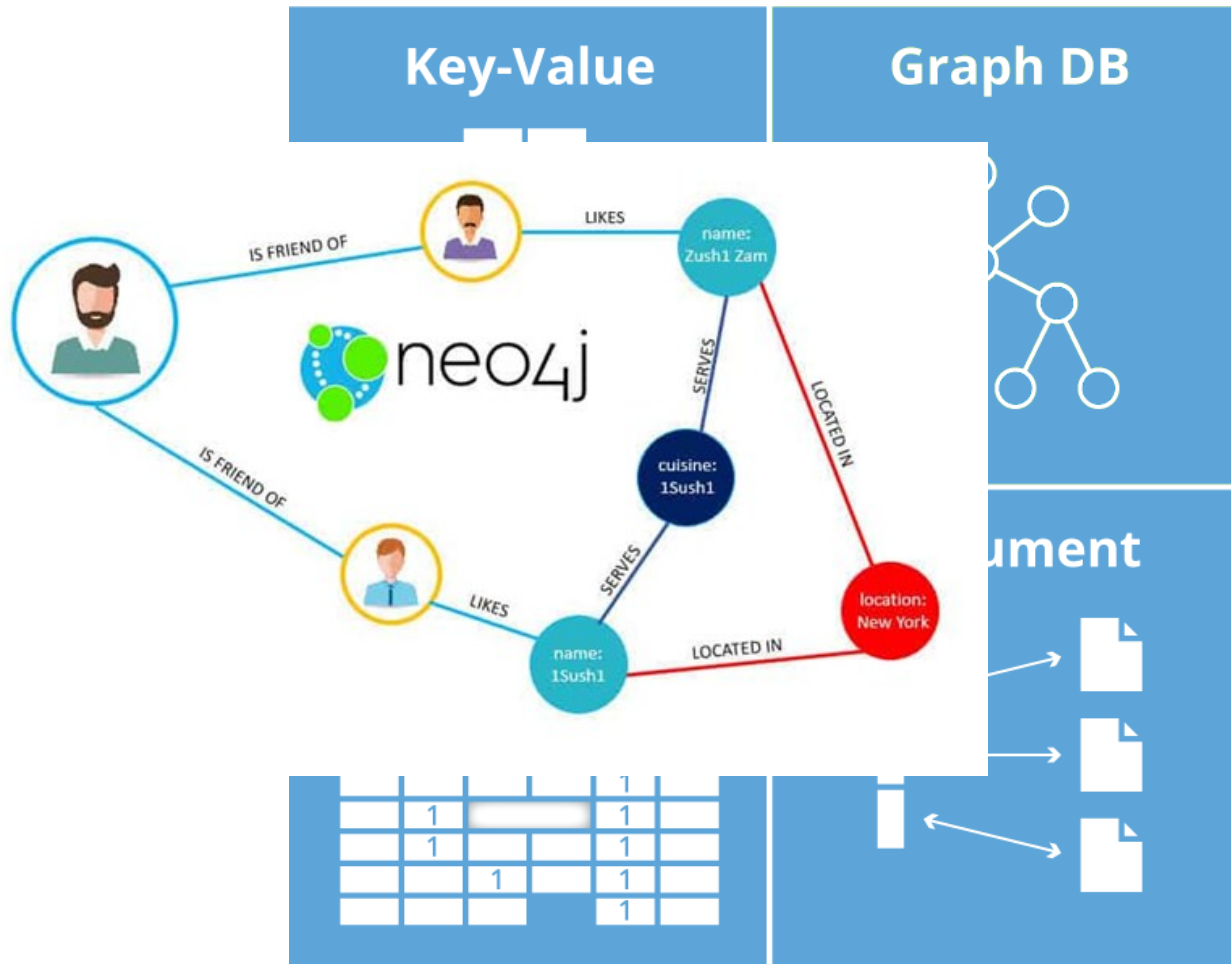


Taxonomy of NoSQL



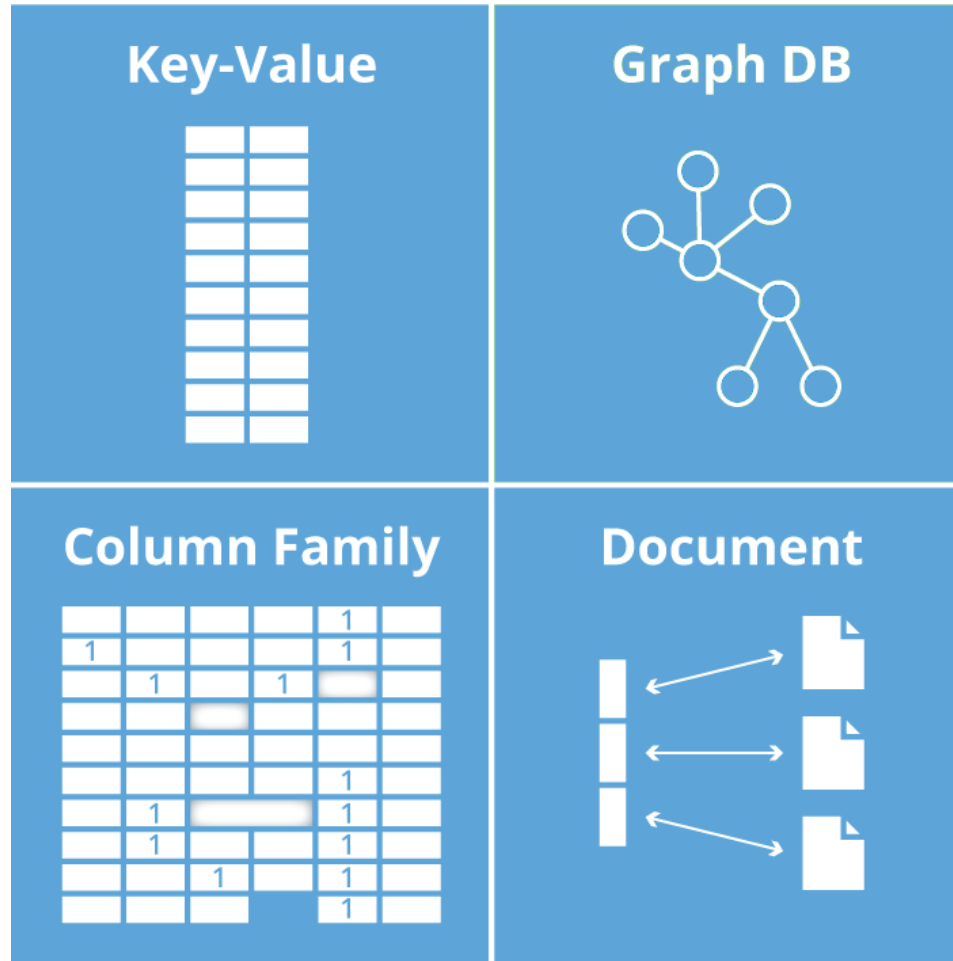
Graph Database
- uses graph structures for queries with nodes, edges and properties to represent and store data.

Taxonomy of NoSQL



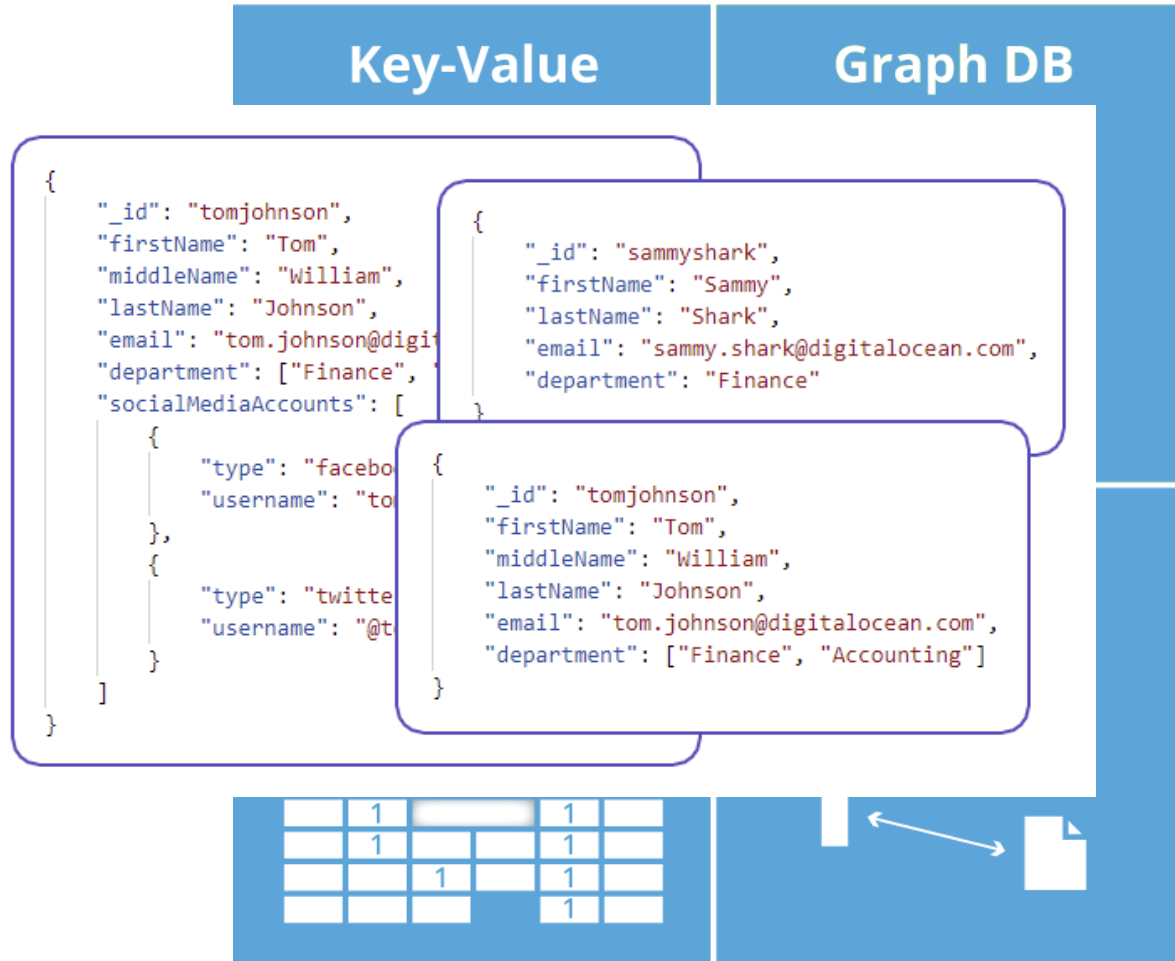
Graph Database
- uses graph structures for queries with nodes, edges and properties to represent and store data.

Taxonomy of NoSQL



Document-oriented – stores data in flexible hierarchical data structures

Taxonomy of NoSQL



Document-oriented



Taxonomy of NoSQL

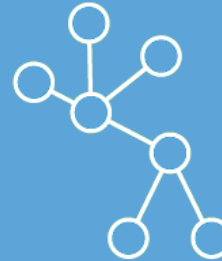
Column

Family — Each storage block contains data from only one column
A wide-column store can be interpreted as a two-dimensional key-value store

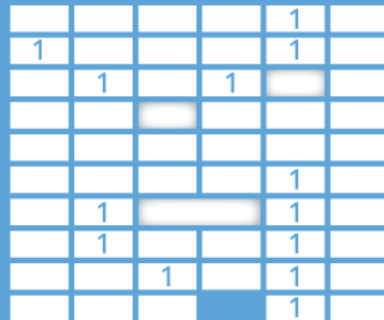
Key-Value



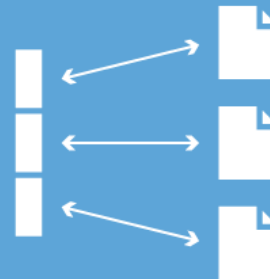
Graph DB



Column Family

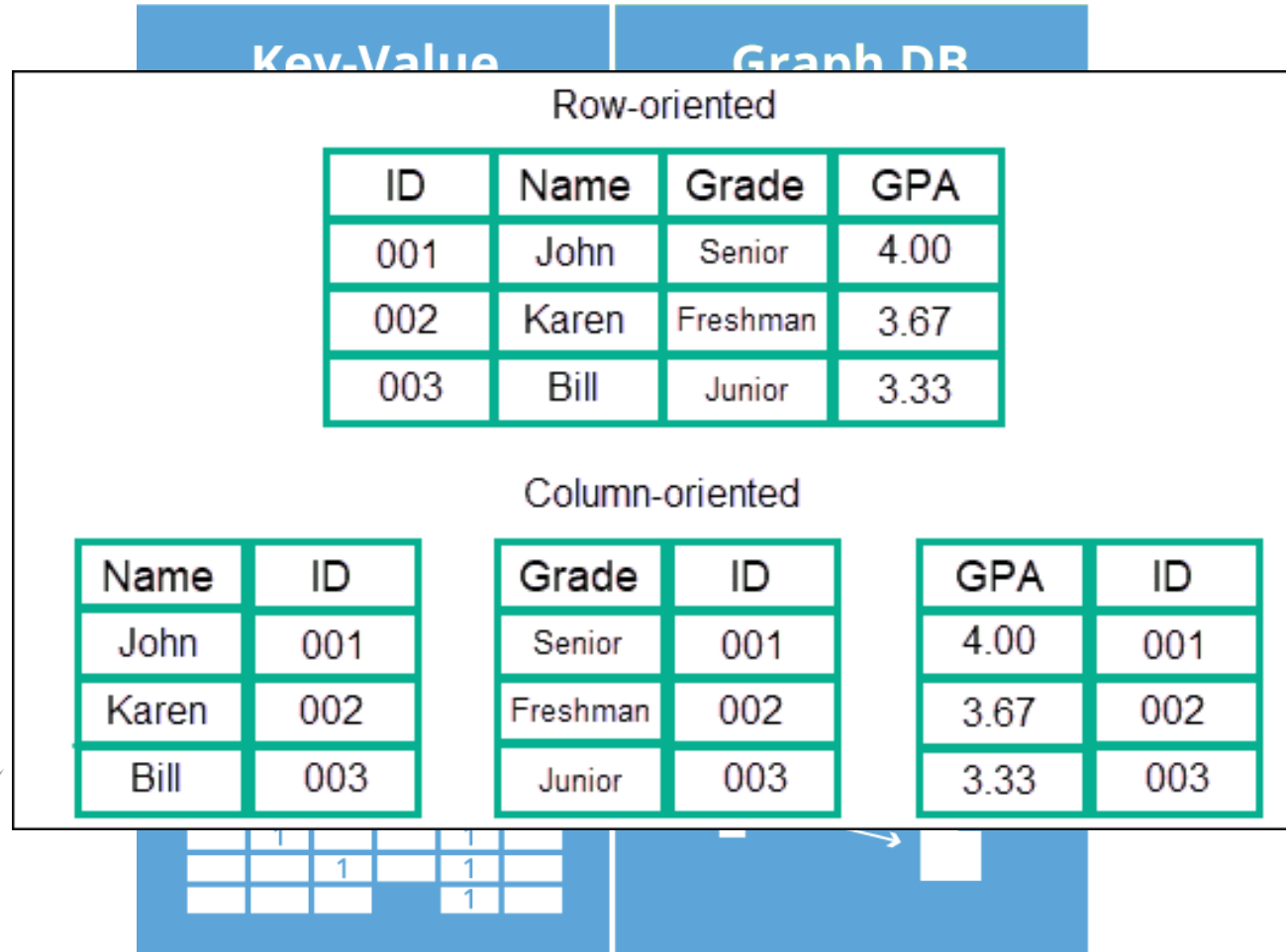


Document



Taxonomy of NoSQL

Column Family





- **Is a document database**
- **Stores data in flexible, JSON-like documents**
 - meaning fields can vary from document to document and data structure can be changed over time
- **Is a distributed database at its core**
 - high availability, horizontal scaling, and geographic distribution are built in and easy to use



- **Free and open-source**, published under the GNU Affero General Public License
- The document model **maps to the objects in your application code**, making data easy to work with
- **Ad hoc queries, indexing, and real time aggregation** provide powerful ways to access and analyze your data



- Here we are **connecting** to a locally hosted MongoDB database called test with a collection named restaurants.

```
# 1. Connect to MongoDB instance running on localhost  
client = pymongo.MongoClient()
```

```
# Access the 'restaurants' collection in the 'test' database  
collection = client.test.restaurants
```



- 5 example documents are being **inserted** into the restaurants collection. Each document represents a restaurant with a name, star rating, and categories (stored as an array).

```
# 2. Insert
new_documents = [
    {
        "name": "Sun Bakery Trattoria",
        "stars": 4,
        "categories": ["Pizza", "Pasta", "Italian", "Coffee", "Sandwiches"]
    }, {
        "name": "Blue Bagels Grill",
        "stars": 3,
        "categories": ["Bagels", "Cookies", "Sandwiches"]
    }, {
        "name": "Hot Bakery Cafe",
        "stars": 4,
        "categories": ["Bakery", "Cafe", "Coffee", "Dessert"]
    }, {
        "name": "XYZ Coffee Bar",
        "stars": 5,
        "categories": ["Coffee", "Cafe", "Bakery", "Chocolates"]
    }, {
        "name": "456 Cookies Shop",
        "stars": 4,
        "categories": ["Bakery", "Cookies", "Cake", "Coffee"]
    }
]
collection.insert_many(new_documents)
```



- In this example, we run a simple query to get all of the documents in the restaurants collection and store them as an array.

```
# 3. Query
for restaurant in collection.find():
    pprint.pprint(restaurant)
```

- Indexes in MongoDB are similar to indexes in other database systems. MongoDB supports indexes on any field or sub-field of a document in a collection.
- Here, we are building an index on the name field with sort order ascending.

```
# 4. Create Index
collection.create_index([('name', pymongo.ASCENDING)])
```



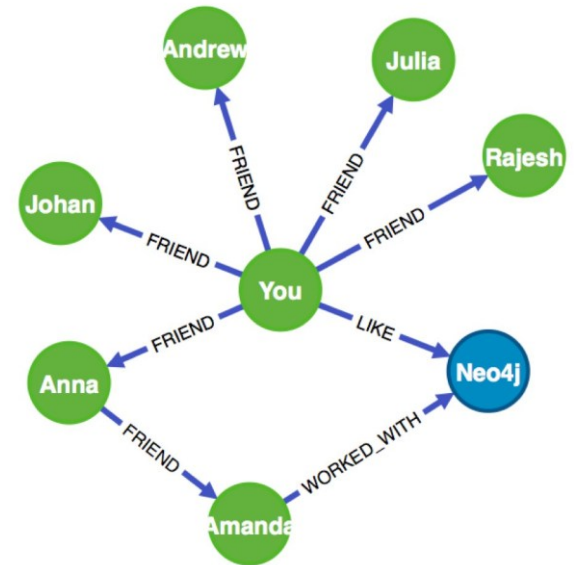
- Using MongoDB's aggregation pipeline, you can filter and analyse data based on a given set of criteria.
- In this example, we pull all the documents in the restaurants collection that have a category of Bakery using the \$match operator and then group them by their star rating using the \$group operator. Using the accumulator operator, \$sum, we can see how many bakeries in our collection have each star rating.

```
# 5. Perform aggregation
pipeline = [
    {"$match": {"categories": "Bakery"}},
    {"$group": {"_id": "$stars", "count": {"$sum": 1}}}
]
```

```
pprint.pprint(list(collection.aggregate(pipeline)))
```



Find Someone in your Network Who Can Help You Learn Neo4j



```
MATCH (you {name:"You"})
```

```
MATCH (expert)-[:WORKED_WITH]->(db:Database  
{name:"Neo4j"})
```

```
MATCH path = shortestPath( (you)-[:FRIEND*..5]-(expert) )
```

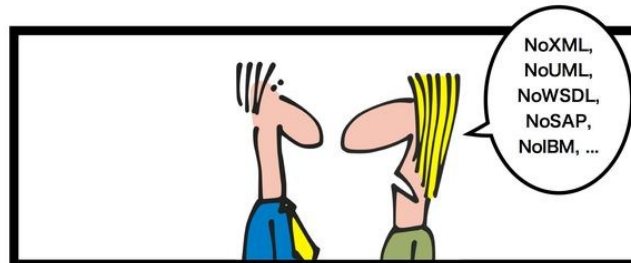
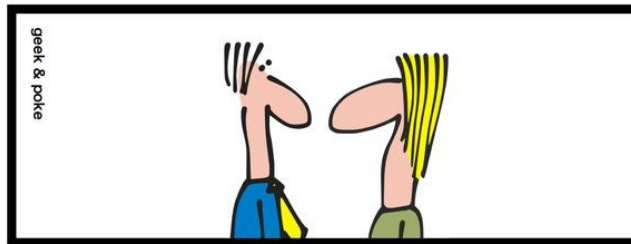
```
RETURN db,expert,path
```

HOW TO WRITE A CV



Leverage the NoSQL boom

RECENTLY DURING THE JOB INTERVIEW



References

Node.js MongoDB Get Started. (n.d.). Retrieved November 26, 2017, from https://www.w3schools.com/nodejs/nodejs_mongodb.asp

What Is MongoDB? (n.d.). Retrieved November 26, 2017, from <https://www.mongodb.com/what-is-mongodb>

What is a Graph Database? A Property Graph Model Intro. (n.d.). Retrieved November 26, 2017, from <https://neo4j.com/developer/graph-database/>

NOSQL Databases. (n.d.). Retrieved November 26, 2017, from <http://nosql-database.org/>