THE BIG DATA TECHNOLOGY LANDSCAPE

L4 – L5

Objective vs. Outcomes

Learning Objectives	Learning Outcomes
The big data technology landscape 1. What is NoSQL databases? 2. Why NoSQL? 3. Key advantages of NoSQL. 4. What is NewSQL? 5. SQL vs. NoSQL. 6. Getting familiar with Hadoop.	a) Able to understand the significance of NoSQL databases. b) Able to understand the need for NewSQL. c) Able to understand the Hadoop platform and be able to appreciate the difference between Hadoop 1.0 and Hadoop 2.0.

The Big Data Technology Landscape

- The big data technology landscape can be majorly studied under two important technologies:
  - *NoSQL*
  - *Hadoop*

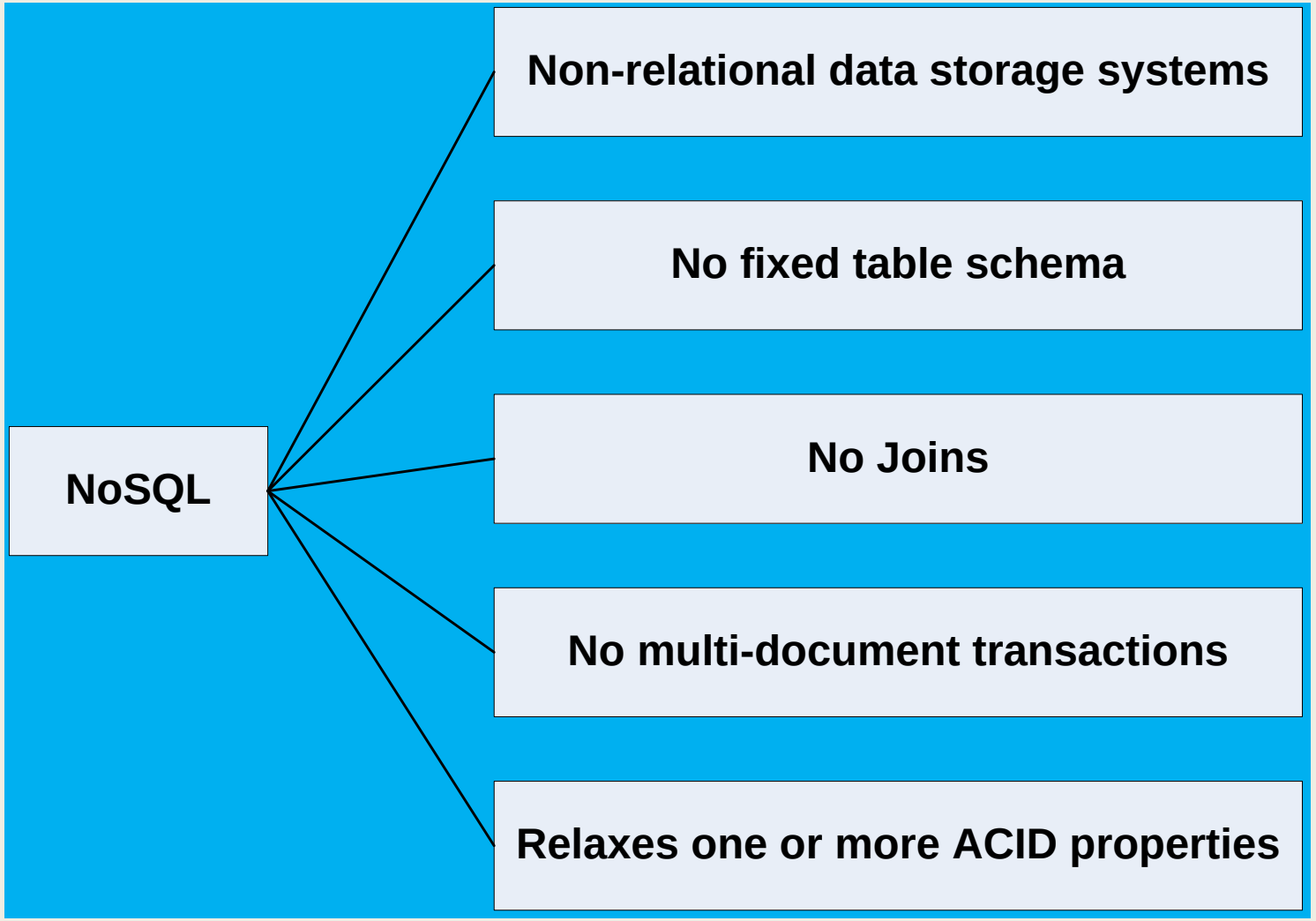
Agenda

- NoSQL
  - *What is it?*
  - *Types of NoSQL Databases*
  - *Why NoSQL?*
  - *Advantages of NoSQL*
  - *NoSQL Vendors*
  - *SQL versus NoSQL*
  - *NewSQL*
  - *Comparison of SQL, NoSQL and NewSQL*
- Hadoop
  - *Features of Hadoop*
  - *Key Advantages of Hadoop*
  - *Versions of Hadoop*

NoSQL (NOT ONLY SQL)

- NoSQL stands for Not Only SQL. These are non-relational, open source, distributed databases.
- The term NoSQL was first coined by Carlo Strozzi in 1998 to name his lightweight database that did not expose the standard SQL interface.
- NoSQL databases are widely used in big data and other real-time web applications.
- Likewise it is used to store social media data and all such data which cannot be stored and analyzed comfortably in RDBMS.

What is NoSQL?



Types of NoSQL

Key value data store

- Riak
- Redis
- Membase

Column-oriented data store

- Cassandra
- HBase
- HyperTable

Document data store

- MongoDB
- CouchDB
- RavenDB

Graph data store

- Infinite Graph
- Neo4
- Allegro Graph

Types of NoSQL

1. **Key-value:** It maintains a big hash table of keys and values. For example, Dynamo, Redis, Riak, etc.

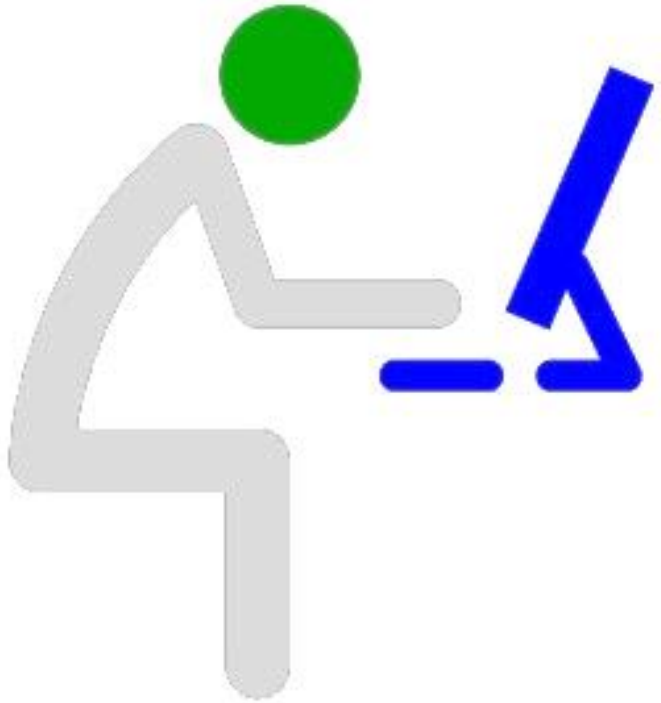
   Sample Key- Value Pair in Key- Value Database

Key	Value
First Name	Akhilesh
Last Name	Singh

key-value pairs

A `key` is a field name, an attribute, an identifier. The content of that field is its `value`, the data that is being identified and stored.

key-value pairs

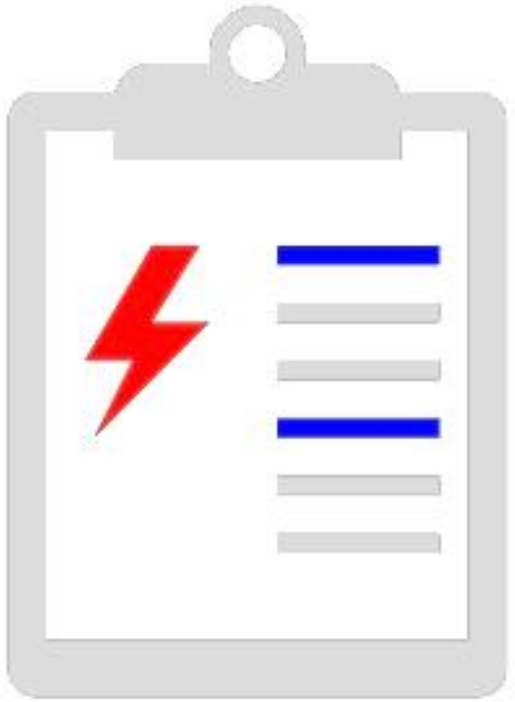


A field name, together with the data entered into that field, is a key-value pair.

a unique identifier

A `key` is a unique identifier, and a `value` is the actual data that is being identified.

show me an example



For instance, in a Contacts database (*i.e.*, an address book), each record will have a field called "city".

`city:` “Buffalo”

Every record contains the `key` "city", and in one particular record its `value` might be "Buffalo". It's pretty straightforward.

`city:` “Toronto”

In another `record` in the same Contacts database, the city might be "Toronto”.
Same key, different `value`.

`age 19:` “yes”

Now the pair above is not well designed. The `key` itself holds some data, which is a no-no.

Data should be only be stored in the `value`.

`age 19:` “yes”

And what if:

`age 19` = "no” … ?

How old is the person then?

Types of NoSQL

2. Schema-less
   A. ***Document:*** *It maintains data in collections constituted of documents. For example, MongoDB, Apache CouchDB, Couchbase, MarkLogic, etc.*

Sample Document in Document Database

{

“Book Name”:“Fundamentals of Business Analytics”,

“Publisher”:“Wiley India”,

“Year of Publication”: “2011”

}

Types of NoSQL

B. ***Column:*** *Each storage block has data from only one column. For example: Cassandra, HBase, etc.*

Column Store NoSQL Database

- **Data Model**
  - ***ColumnFamily:*** *ColumnFamily is a single structure that can group Columns and SuperColumns with ease.*
  - ***Key:*** *the permanent name of the record. Keys have different numbers of columns, so the database can scale in an irregular way.*
  - ***Keyspace:*** *This defines the outermost level of an organization, typically the name of the application. For example, '3PillarDataBase' (database name).*
  - ***Column:*** *It has an ordered list of elements aka tuple with a name and a value defined.*
- The best known examples are Google's BigTable and HBase & Cassandra that were inspired from BigTable.

Column Store NoSQL Database

- BigTable, for instance is a high performance, compressed and proprietary data storage system owned by Google. It has the following attributes:
  - ***Sparse*** *– some cells can be empty*
  - ***Distributed*** *– data is partitioned across many hosts*
  - ***Persistent*** *– stored to disk*
  - ***Multidimensional*** *– more than 1 dimension*
  - ***Map*** *– key and value*
  - ***Sorted*** *– maps are generally not sorted but this one is*
- The best known examples are Google's BigTable and HBase & Cassandra that were inspired from BigTable.

Column Store NoSQL Database

- A 2-dimensional table comprising of rows and columns is part of the relational database system.

City	Pincode	Strength	Project
Noida	201301	250	20
Cluj	400606	200	15
Timisoara	300011	150	10
Fairfax	VA 22033	100	5

Column Store NoSQL Database

```
{
3PillarNoida: {
city: Noida
pincode: 201301
},
details: {
strength: 250
projects: 20
}
}
{
3PillarCluj: {
address: {
city: Cluj
pincode: 400606
},
details: {
strength: 200
projects: 15
}
},
{
3PillarTimisoara: {
address: {
city: Timisoara
pincode: 300011
},
details: {
strength: 150
projects: 10
}
}
{
3PillarFairfax : {
address: {
city: Fairfax
pincode: VA 22033
},
details: {
strength: 100
projects: 5
}
}
```

- For the given RDBMS table a BigTable map can be visualized as shown below.

```
{
3PillarNoida: {
city: Noida
pincode: 201301
},
details: {
strength: 250
projects: 20
}
}
```

```
{
3PillarCluj: {
address: {
city: Cluj
pincode: 400606
},
details: {
strength: 200
projects: 15
}
},
```

Column Store NoSQL Database

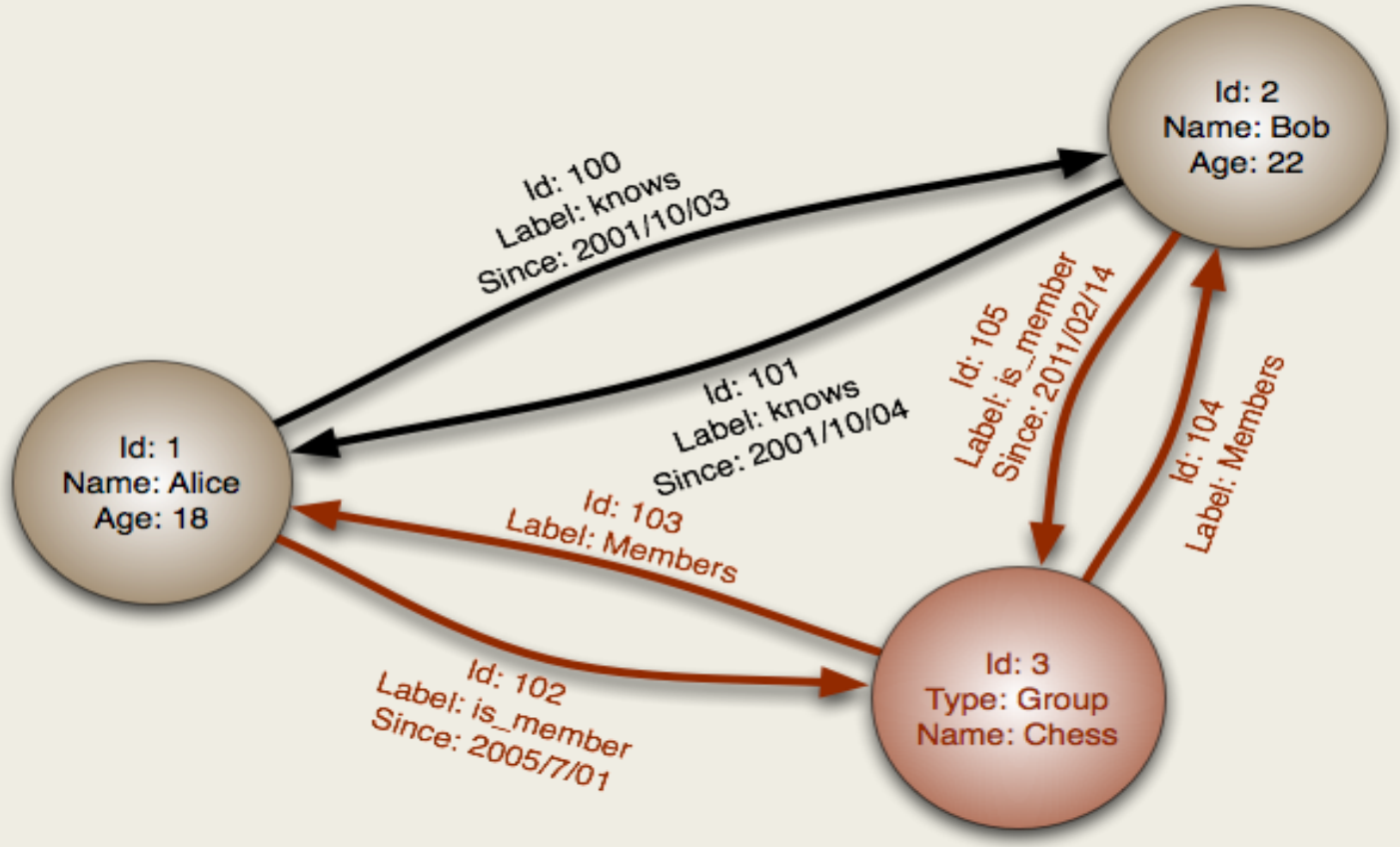
```
{
3PillarNoida: {
city: Noida
pincode: 201301
},
details: {
strength: 250
projects: 20
}
}
{
3PillarCluj: {
address: {
city: Cluj
pincode: 400606
},
details: {
strength: 200
projects: 15
}
},
{
3PillarTimisoara: {
address: {
city: Timisoara
pincode: 300011
},
details: {
strength: 150
projects: 10
}
}
{
3PillarFairfax : {
address: {
city: Fairfax
pincode: VA 22033
},
details: {
strength: 100
projects: 5
}
}
```

- The outermost keys 3PillarNoida, 3PillarCluj, 3PillarTimisoara and 3PillarFairfax are analogues to rows.
- 'address' and 'details' are called **column families**.
- The column-family 'address' has **columns** 'city' and 'pincode'.
- The column-family details' has **columns** 'strength' and 'projects'.
- Columns can be referenced using CloumnFamily.

```
{
3PillarCluj: {
address: {
city: Cluj
pincode: 400606
},
details: {
strength: 200
projects: 15
}
},
```

Types of NoSQL

C. **Graph:** They are also called network database. A graph stores data in nodes. For example: Neo4j, HyperGraphDB, etc.



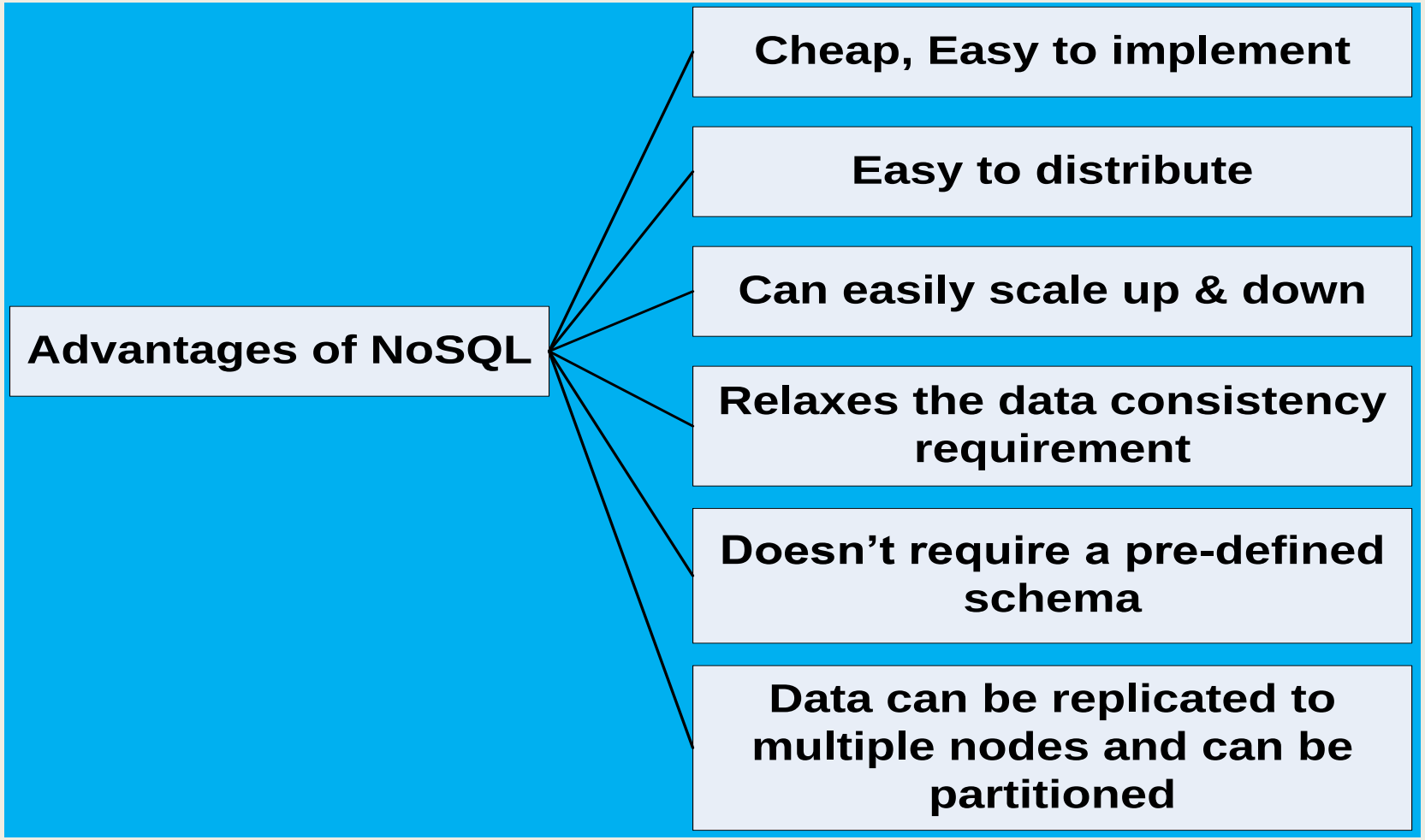
Why NoSQL

- It has scale out architecture instead of the monolithic architecture of relational databases. It can house large volumes of structured, semi-structured, and unstructured data.
- **Dynamic schema:** NoSQL database allows insertion of data without a pre-defined schema. It facilitates application changes in real time, which thus supports faster development, easy code integration, and requires less database administration.

Why NoSQL

- **Auto-sharding:** It automatically spreads data across an arbitrary number of servers. It balances the load of data and query on the available servers; and if and when a server goes down, it is quickly replaced without any major activity disruptions.
- **Replication:** It offers good support for replication which in turn guarantees high availability, fault tolerance, and disaster recovery.

Advantages of NoSQL



Advantages of NoSQL

1. **Can easily scale up and down:** NoSQL database supports scaling rapidly and elastically and even allows to scale to the cloud.
   - *Cluster scale*
   - *Performance scale*
   - *Data scale*

Advantages of NoSQL

2. Doesn’t require a pre-defined schema
3. Cheap, easy to implement
4. Relaxes the data consistency requirement
5. Data can be replicated to multiple nodes and can be partitioned

NoSQL Vendors

Company	Product	Most widely used by
Amazon	DynamoDB	LinkedIn, Mozilla
Facebook	Cassandra	Netflix, Twitter, eBay
Google	BigTable	Adobe Photoshop

SQL Vs. NoSQL

SQL	NoSQL
Relational database	Non-relational, distributed database
Relational model	Model-less approach
Pre-defined schema	Dynamic schema for unstructured data
Table based databases	Document-based or graph-based or wide column store or key-value pairs databases
Vertically scalable (by increasing system resources)	Horizontally scalable (by creating a cluster of commodity machines)

SQL Vs. NoSQL

SQL	NoSQL
Uses SQL	Uses UnQL (Unstructured Query Language)
Not preferred for large datasets	Largely preferred for large datasets
Not a best fit for hierarchical data	Best fit for hierarchical storage as it follows the key-value pair of storing data similar to JSON (Java Script Object Notation)
Emphasis on ACID properties	Follows Brewer's CAP theorem
Excellent support from vendors	Relies heavily on community support

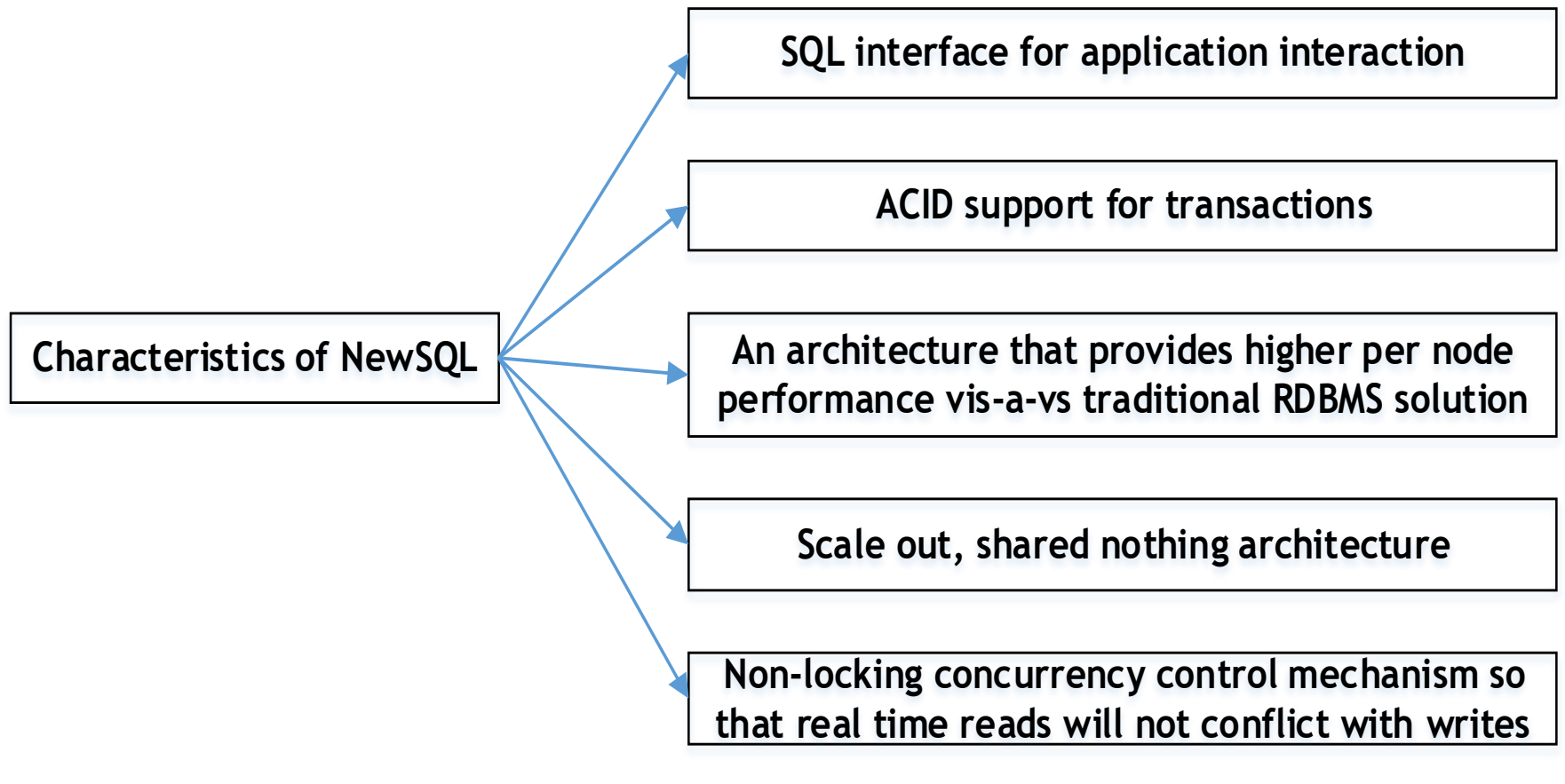
SQL Vs. NoSQL

SQL	NoSQL
Supports complex querying and data keeping needs	Does not have good support for complex querying
Can be configured for strong consistency	Few support strong consistency (e.g., MongoDB), few others can be configured for eventual consistency (e.g., Cassandra)
Examples: Oracle, DB2, MySQL, MS SQL, PostgreSQL, etc.	MongoDB, HBase, Cassandra, Redis, Neo4j, CouchDB, Couchbase, Riak, etc.

NewSQL

- We need a database that has the same scalable performance of NoSQL systems for On Line Transaction Processing (OLTP) while still maintaining the ACID guarantees of a traditional database.
- This new modern RDBMS is called NewSQL. It supports relational data model and uses SQL as their primary interface.
- NewSQL is based on the shared nothing architecture with a SQL interface for application interaction.

NewSQL


Characteristics of NewSQL
SQL interface for application interaction
ACID support for transactions
An architecture that provides higher per node performance vis-a-vs traditional RDBMS solution
Scale out, shared nothing architecture
Non-locking concurrency control mechanism so that real time reads will not conflict with writes

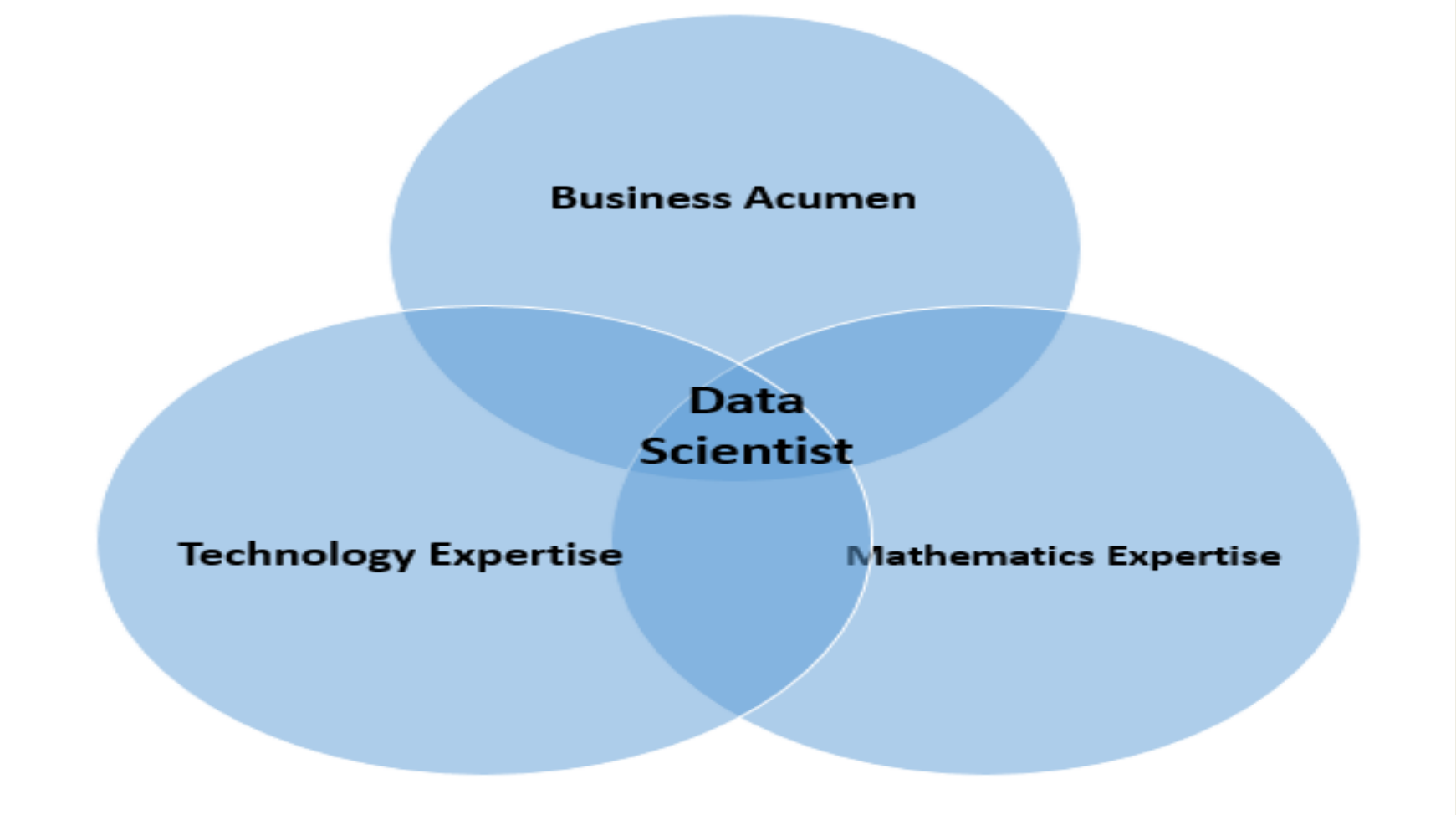
SQL Vs. NoSQL Vs. NewSQL

	SQL	NoSQL	NewSQL
Adherence to ACID properties	Yes	No	Yes
OLTP/OLAP	Yes	No	Yes
Schema rigidity Adherence to data model	Yes Adherence to relational model	No	Maybe
Data Format Flexibility	No	Yes	Maybe
Scalability	Scale up Vertical Scaling	Scale out Horizontal Scaling	Scale out
Distributed Computing	Yes	Yes	Yes
Community Support	Huge	Growing	Slowly growing

Data Science

- Data science is the science of extracting knowledge from data. Data science is multi-disciplinary.
- Data Science is a blend of various tools, algorithms, and machine learning principles with the goal to discover hidden patterns from the raw data.
- Data Science is primarily used to make decisions and predictions making use of predictive causal analytics, prescriptive analytics (predictive plus decision science) and machine learning.

Data Scientist



Business Acumen Skills

- A data scientist should have business acumen skills to counter the pressure of business:
  - *Understanding of domain*
  - *Business strategy*
  - *Problem solving*
  - *Communication*
  - *Presentation*
  - *Inquisitiveness*

Technology Expertise Skills

- A data scientist should be technology expert to convert the business into business logic:
  - *Good database knowledge such as RDBMS.*
  - *Good NoSQL database knowledge such as MongoDB, Cassandra, HBase, etc.*
  - *Programming languages such as Java, Python, etc.*
  - *Open-source tools such as Hadoop, R.*
  - *Datawarehousing, Datamining.*
  - *Visualization such as Tableau, Flare, Google visualization APIs, etc.*

Mathematics Expertise Skills

- A data scientist should be mathematics expert to formulize and analyze data:
  - *Mathematics.*
  - *Statistics.*
  - *Artificial Intelligence (AI).*
  - *Machine learning.*
  - *Pattern recognition.*
  - *Natural Language Processing.*

Data Science Process

- Collecting raw data from multiple data sources.
- Processing the data.
- Integrating the data and preparing clean datasets.
- Engaging in explorative data analysis using model and algorithms.
- Preparing presentations using data visualizations (commonly called Infographics, or BizAnalytics, etc.)
- Communicating the findings to all stakeholders.
- Making faster and better decisions.

Responsibilities of Data Scientist

- Data Management
- Analytical Techniques
- Business Analysts

Terminologies Used in Big data Environments

- In-Memory Analytics
- In-Database Processing
- Massively Parallel Processing
- Parallel System
- Distributed System
- Shared Nothing Architecture

In-Memory Analytics

- In-memory analytics is an approach to querying data when it resides in a computer's random access memory (RAM), as opposed to querying data that is stored on physical disks.
- This results in vastly shortened query response times, allowing business intelligence (BI) and analytic applications to support faster business decisions.
- As the cost of RAM declines, in-memory analytics is becoming feasible for many businesses.
- In addition to providing incredibly fast query response times, in-memory analytics can reduce or eliminate the need for data indexing and storing pre-aggregated data in OLAP cubes or aggregate tables.

In-Database Processing

- In-database processing is also called as in-database analytics. It works by fusing data warehouses with analytical systems.
- Typically the data from various enterprise OLTP systems after cleaning up (de-duplication, scrubbing, etc.) through the process of ETL is stored in the Enterprise Data Warehouse (EDW) or data marts.
- With in-database processing, the database program itself can run the computations eliminating the need for export and thereby saving on time.

Symmetric Multiprocessor System

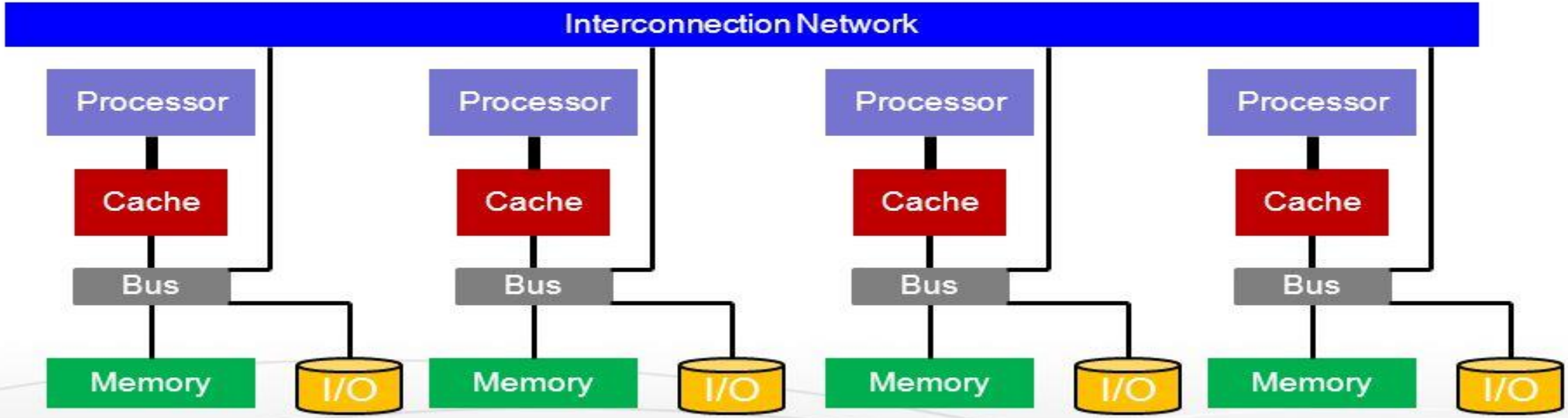
- In SMP, there is a single common main memory that is shared by two or more identical processors. The processors have full access to all I/O devices and are controlled by a single operating system instance.
- SMP are tightly coupled multiprocessor systems. Each processor has its own high-speed memory, called cache memory and are connected using a system bus.

Massively Parallel Processing

- MPP refers to the coordinated processing of programs by a number of processors working parallel.
- The processors, each have their own operating systems and dedicated memory.
- They work on different parts of the same program. The MPP processors communicate using some sort of messaging interface.

Massively Parallel Processors

- Massively Parallel Processors (MPP) architecture consists of nodes with each having its own processor, memory and I/O subsystem



- An independent OS runs at each node

Parallel and Distributed System

- A parallel database system is a tightly coupled system. The processor, co-operate for query processing. The user is unaware of the parallelism.
- Distributed database systems are known to be loosely coupled and are composed by individual machines.
- Each of the machines can run their individual application and serve their own respective users. The data is usually distributed across several machines.

Shared Nothing Architecture

- In shared nothing architecture, neither memory nor disk is shared among multiple processors.
- Advantages:
  - *Fault Isolation*
  - *Scalability*

SHARE NOTHING ARCHITECTURE