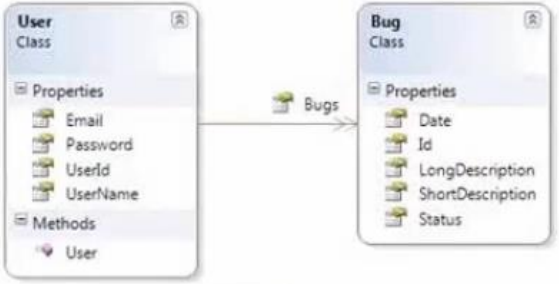
noSQL Databases

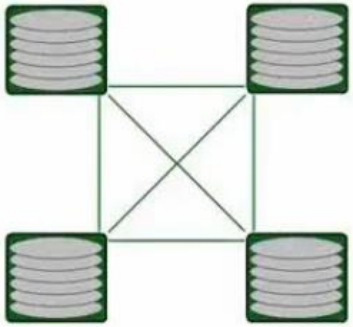
What is NoSQL?

Next Generation Databases

Not Only SQL



Non – Relational



Distributed Architecture

Open Source

Horizontally Scalable

What is NoSQL?

Schema - Free

Easy – Replication

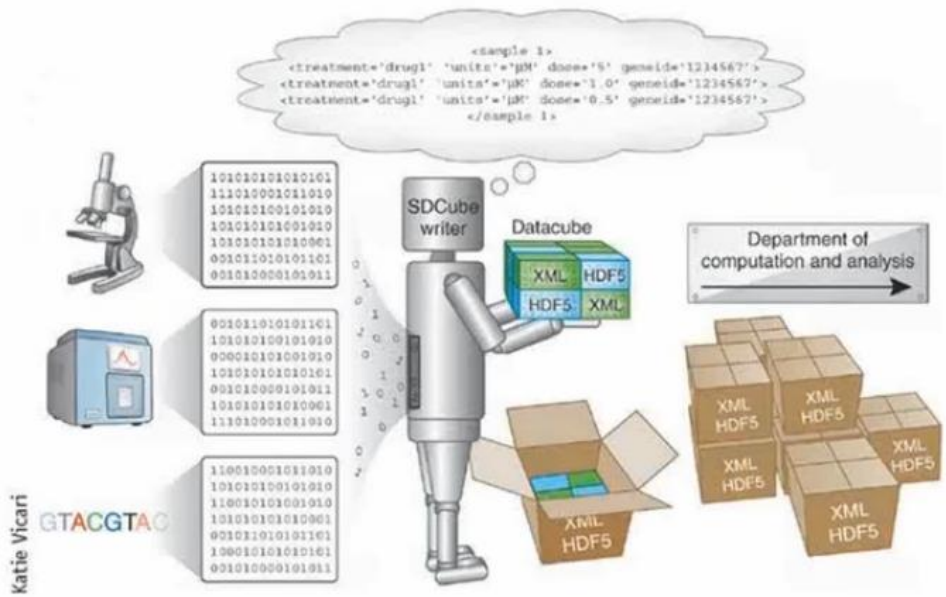
Simple API

Can Manage Huge Amount of Data

Can be implement on Commodity Hardware's

~ 150 No SQL Database are there in Market

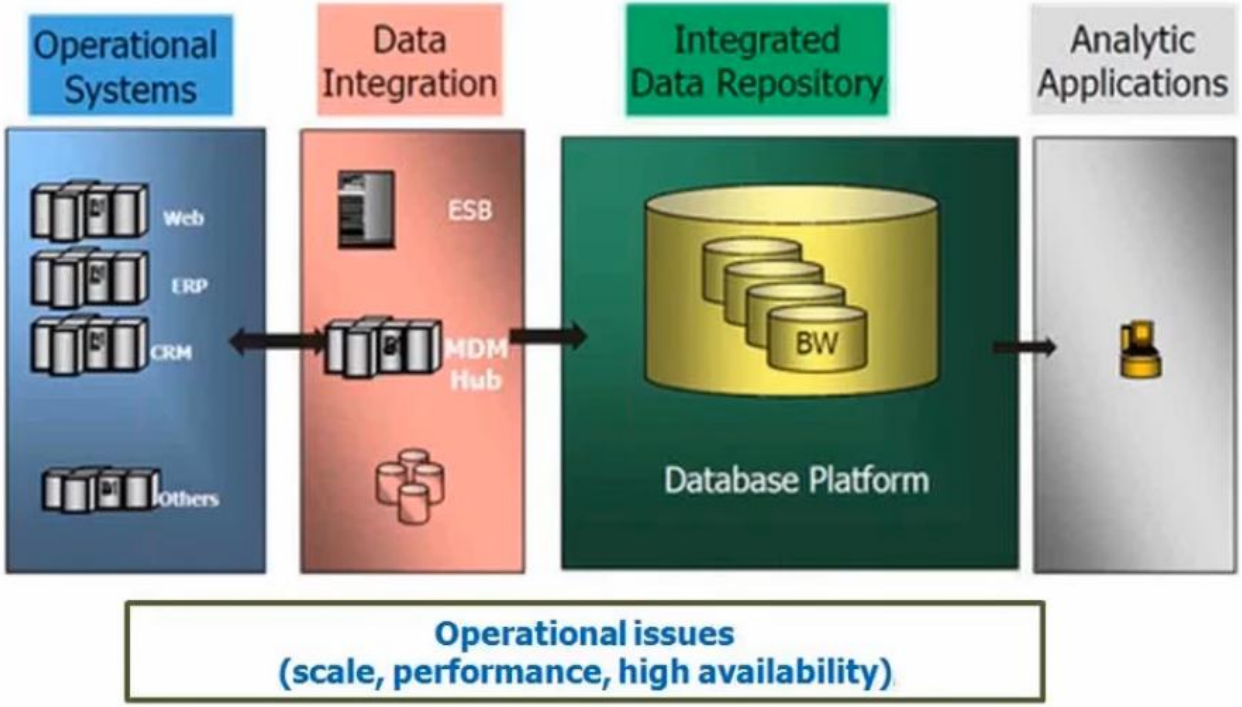
Why NoSQL?



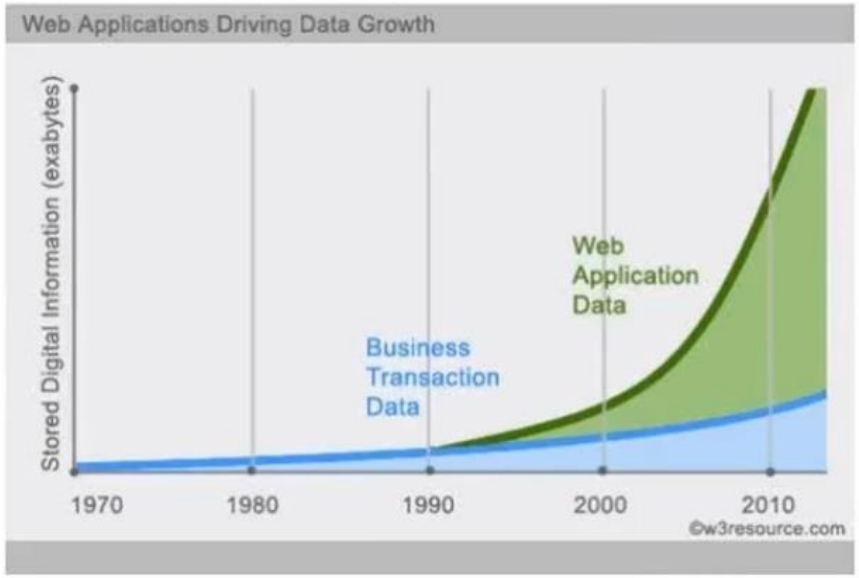
Nature of data

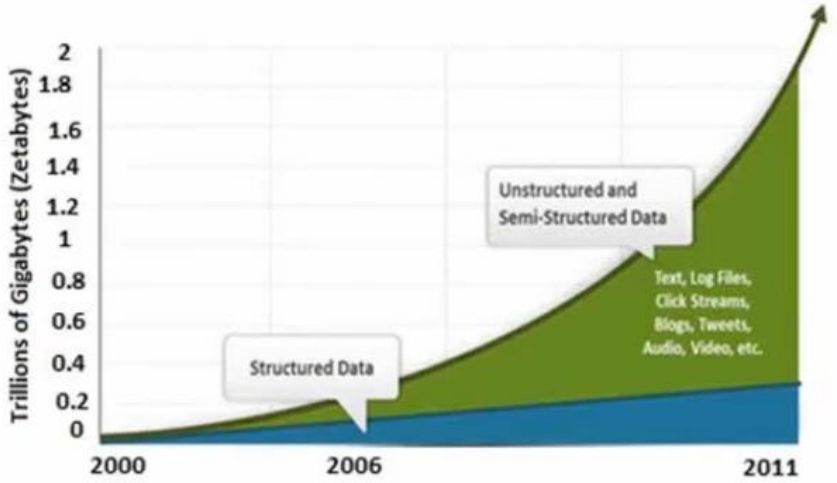
Application development (high coding velocity & agility)

Why NoSQL?


Operational Systems
Data Integration
Integrated Data Repository
Analytic Applications
Web
ERP
CRM
Others
ESB
MDM Hub
BW
Database Platform
Operational issues
(scale, performance, high availability)

Why NoSQL?

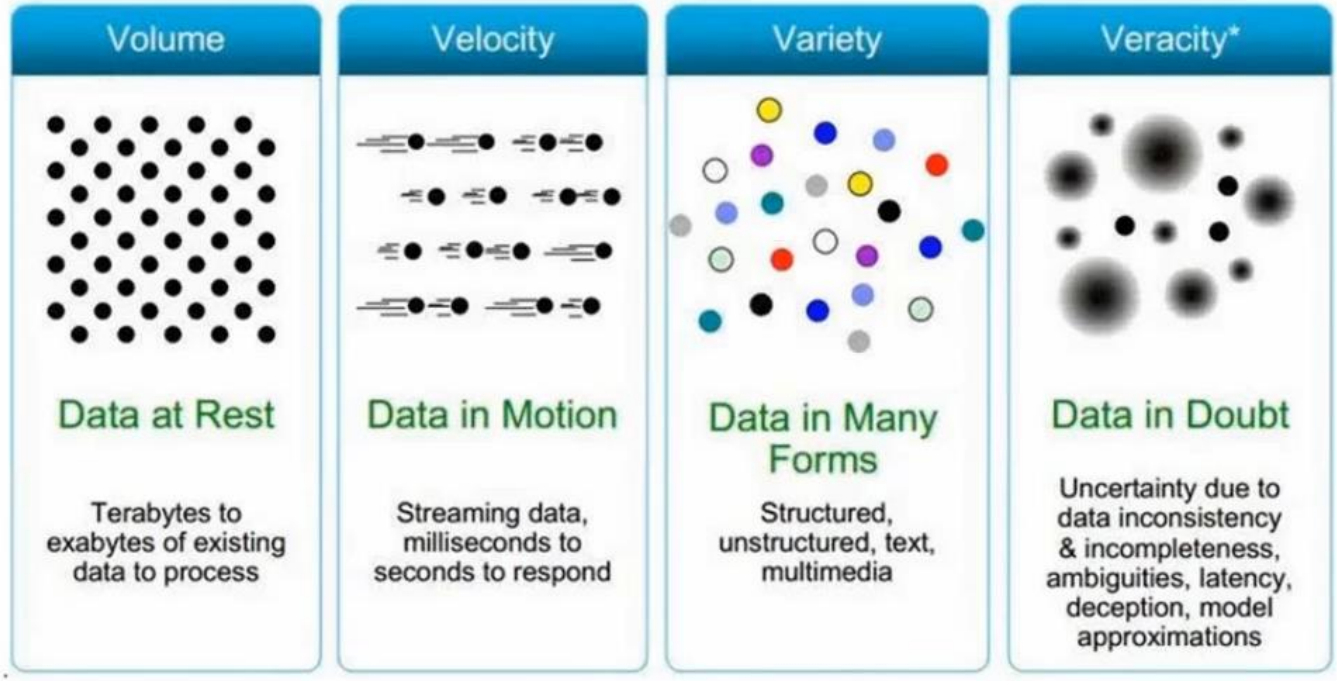




Data warehousing and analytics

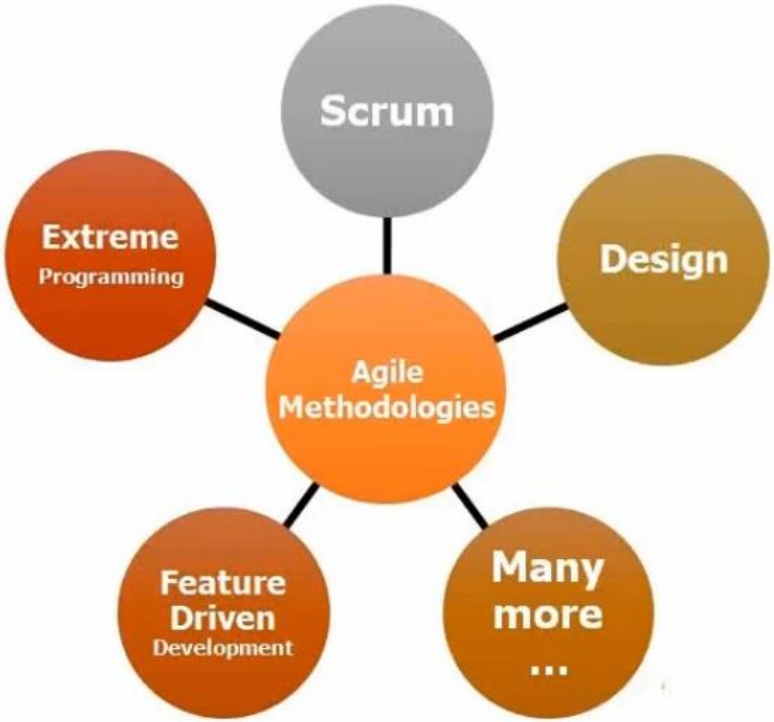
Benefits of NoSQL

- ✓ Large volumes of structured, semi-structured, and unstructured data



Benefits of NoSQL

- ✓ Agile development, quick changes, and frequent code pushes

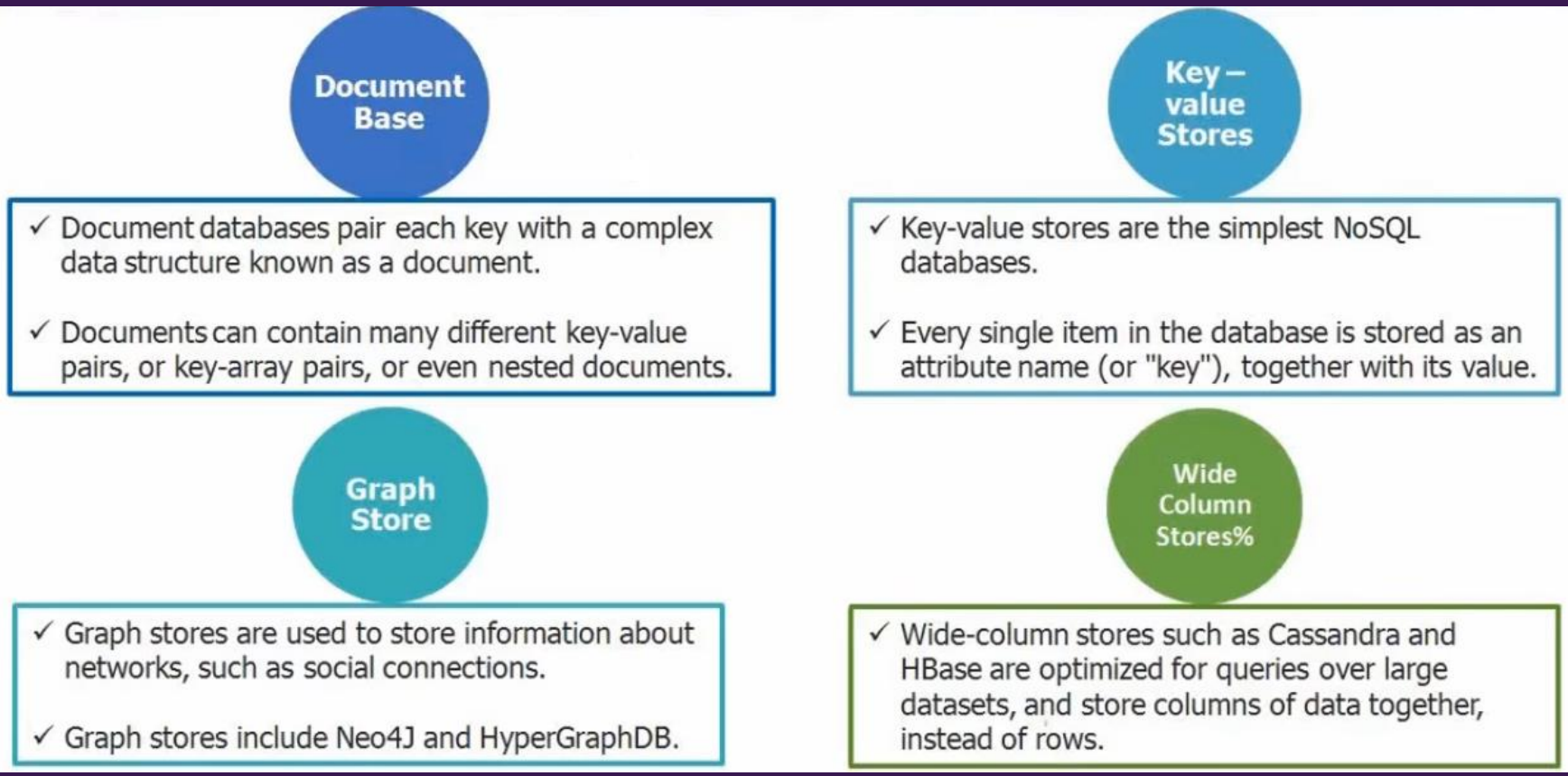


Benefits of NoSQL

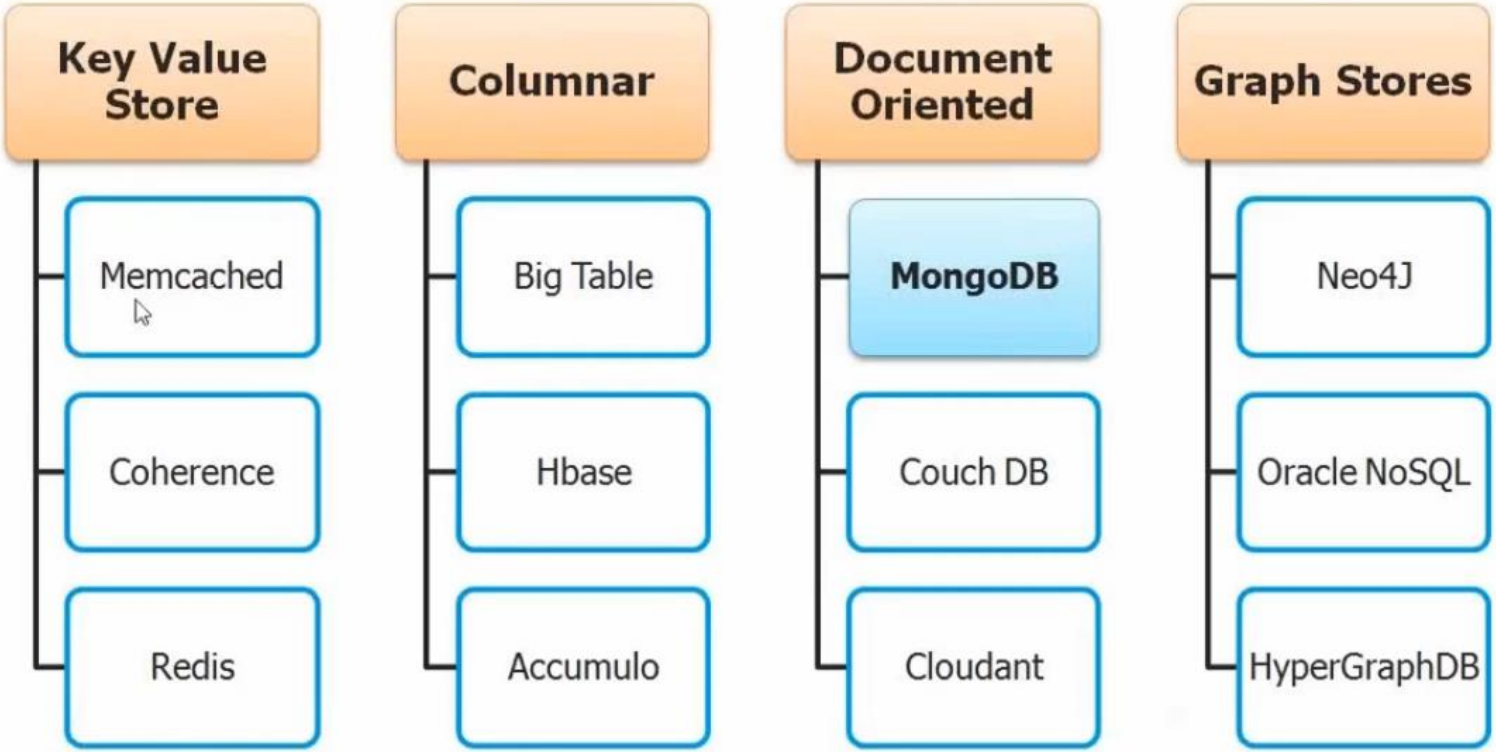
- ✓ Object-oriented programming that is easy to use and flexible

- ✓ Horizontal scaling instead of expensive hardware's

Categories of NoSQL Database



Types of NoSQL Database



WHEN TO USE

NOSQL

AND WHEN TO USE

SQL

SQL

ATOMICITY

CONSISTENCY

ISOLATION

DURABILITY

noSQL

BASICALLY

AVAILABLE

SOFT STATE

EVENTUALLY CONSISTENT

SQL

THE GOOD

High performance for transactions. Follow's ACID

Highly structured, very portable

Small amounts of data

*SMALL IS LESS THAN 500GB

Supports many tables with different types of data

Can fetch ordered data

Compatible with lots of tools

SQL

THE BAD

Complex queries take a long time

The relational model takes a long time to learn

Not really scalable

Not suited for rapid development

noSQL

THE GOOD

Fits well for volatile data

High read and write throughput

In general it’s faster than SQL

Scales really well (Horizontally)

Rapid development is possible

noSQL

THE BAD

Key/Value pairs need to be packed/unpacked all the time

Still working on getting security for these working as well as SQL

Lack of relations from one key to another

tl;dr

SQL

works great, can’t scale for large data

noSQL

works great, doesn't fit all situations

so use both, but think about when you want to use them!

NOSQL USE CASES

LARGE DATA VOLUMES

MASSIVELY DISTRIBUTED ARCHITECTURE
REQUIRED TO STORE THE DATA
GOOGLE, AMAZON, FACEBOOK, 100K SERVERS

EXTREME QUERY WORKLOAD

IMPOSSIBLE TO EFFICIENTLY DO JOINS AT THAT
SCALE WITH AN RDBMS

SCHEMA EVOLUTION

SCEMA FLEXIBILITY IS NOT TRIVIAL AT A LARGE SCALE
BUT IT CAN BE WITH NO SQL