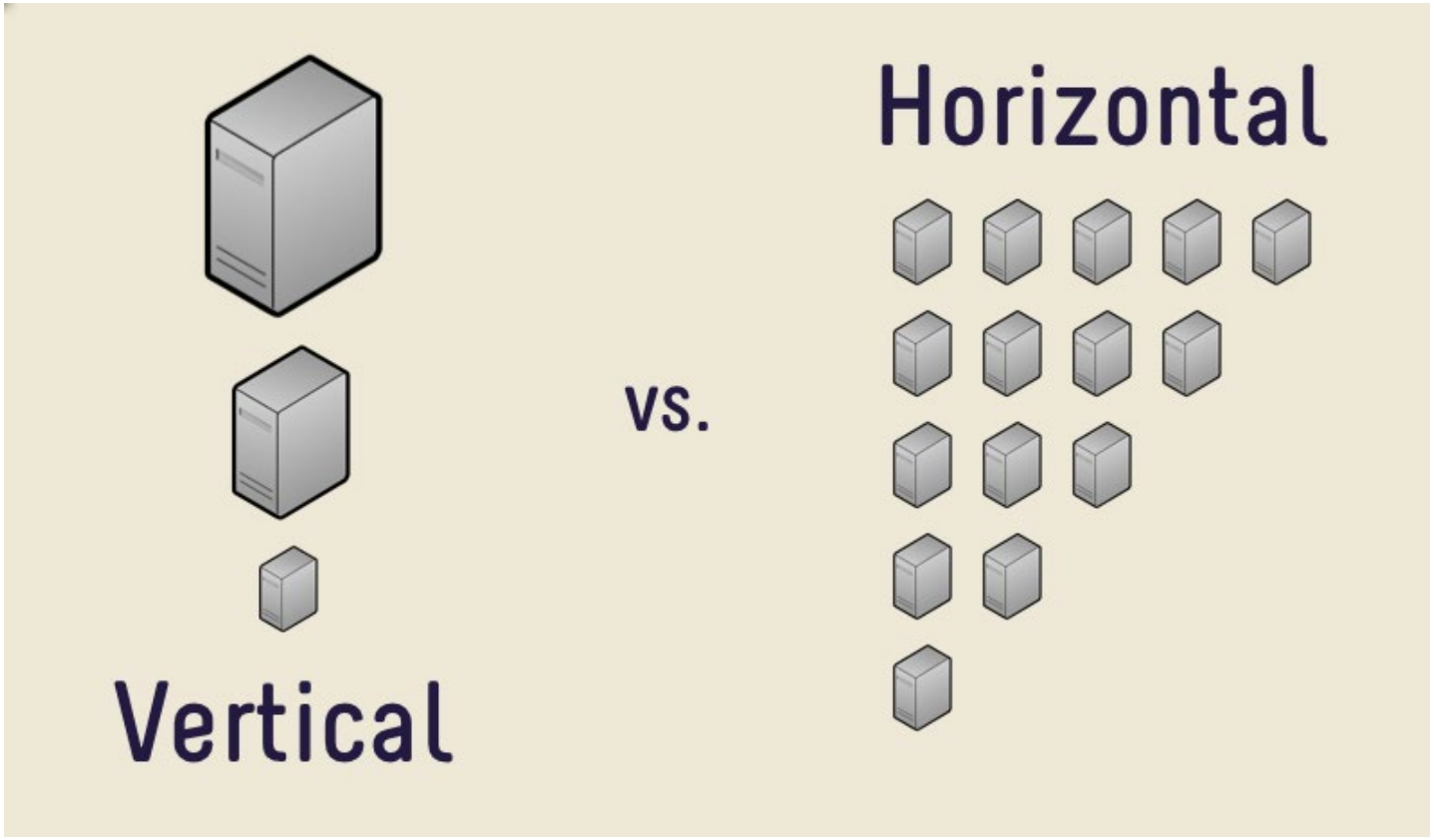
NOSQL Databases

PART 1 - Introduction

Cracks appearing in Relational Databases

- Impedance Mismatch
  - Developer must move from Object Oriented Programming Model to the Relational Model paradigm
    - Use of SQL and provided API's
- Reduce Impedance Mismatch - A Mapping Layer (ORM) e.g. Hibernate, ADO.NET, EJB, Django etc.
- Scalability and Performance can be critical for Databases
  - Typical "Scaling Up" Approach i.e. Vertical Scaling
  - Some Support for "Scaling Out" i.e. Horizontal Scaling
    - Clustering (e.g. Microsoft SQL Server, Oracle RAC) –expensive!
    - Table Partitioning usually supported– vertical or horizontal
      - Typically used to improve write\read performance by reducing\ parallelizing I/O
  - RDBMS can struggle with large volumes of read and writes which impacts latency
  - Availability – Relational Model does not do this per se
    - Vendors provide some solutions at a price e.g. Oracle Real Application Clusters (RAC)
    - In general if RDBMS server is down – data is not available

Vertical Scaling Vs Horizontal Scaling



Vertical Scaling (aka Scaling up) Vs. Horizontal Scaling aka (Scaling out).

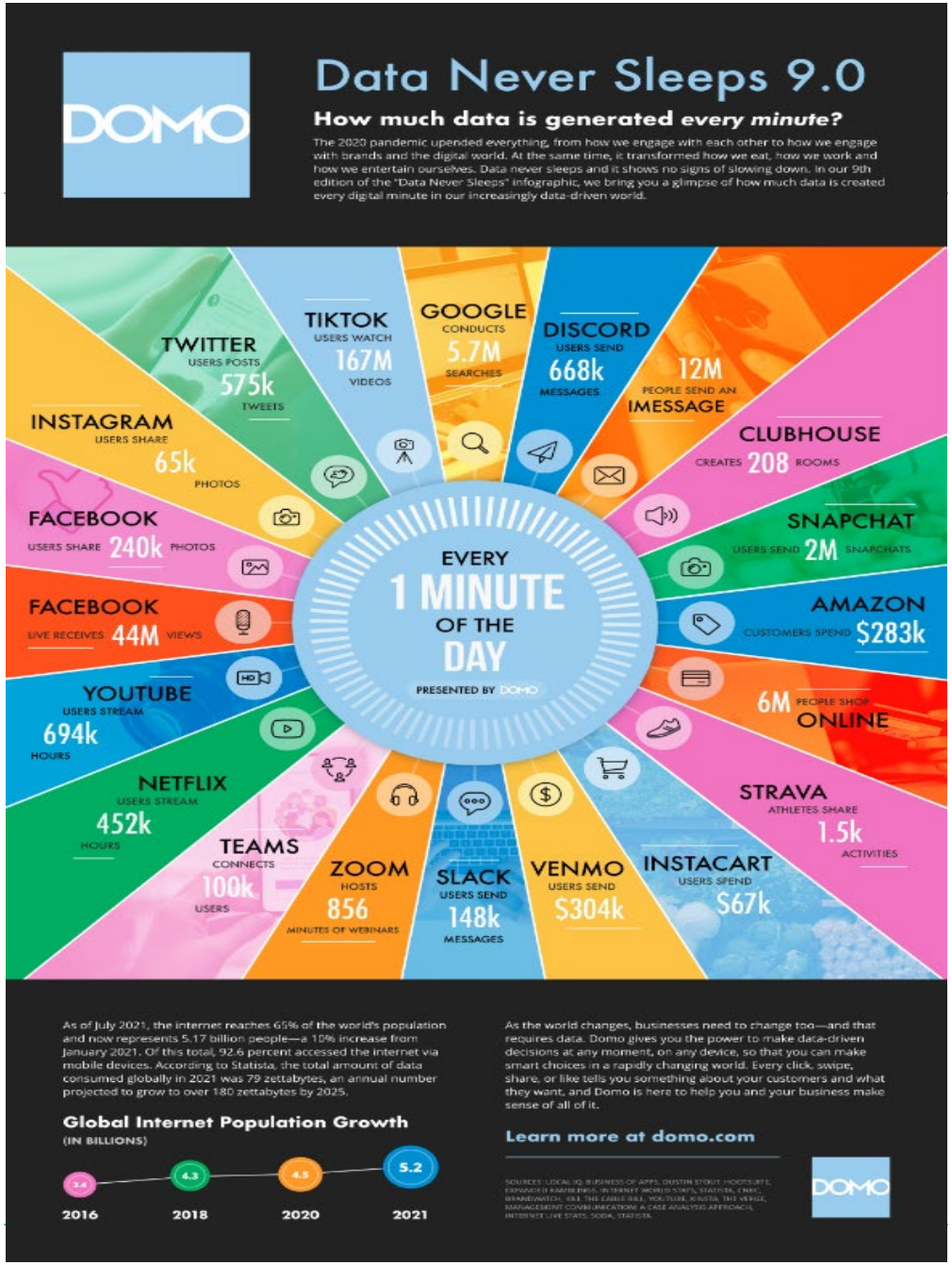
Big Data: Let's go to the Movies!

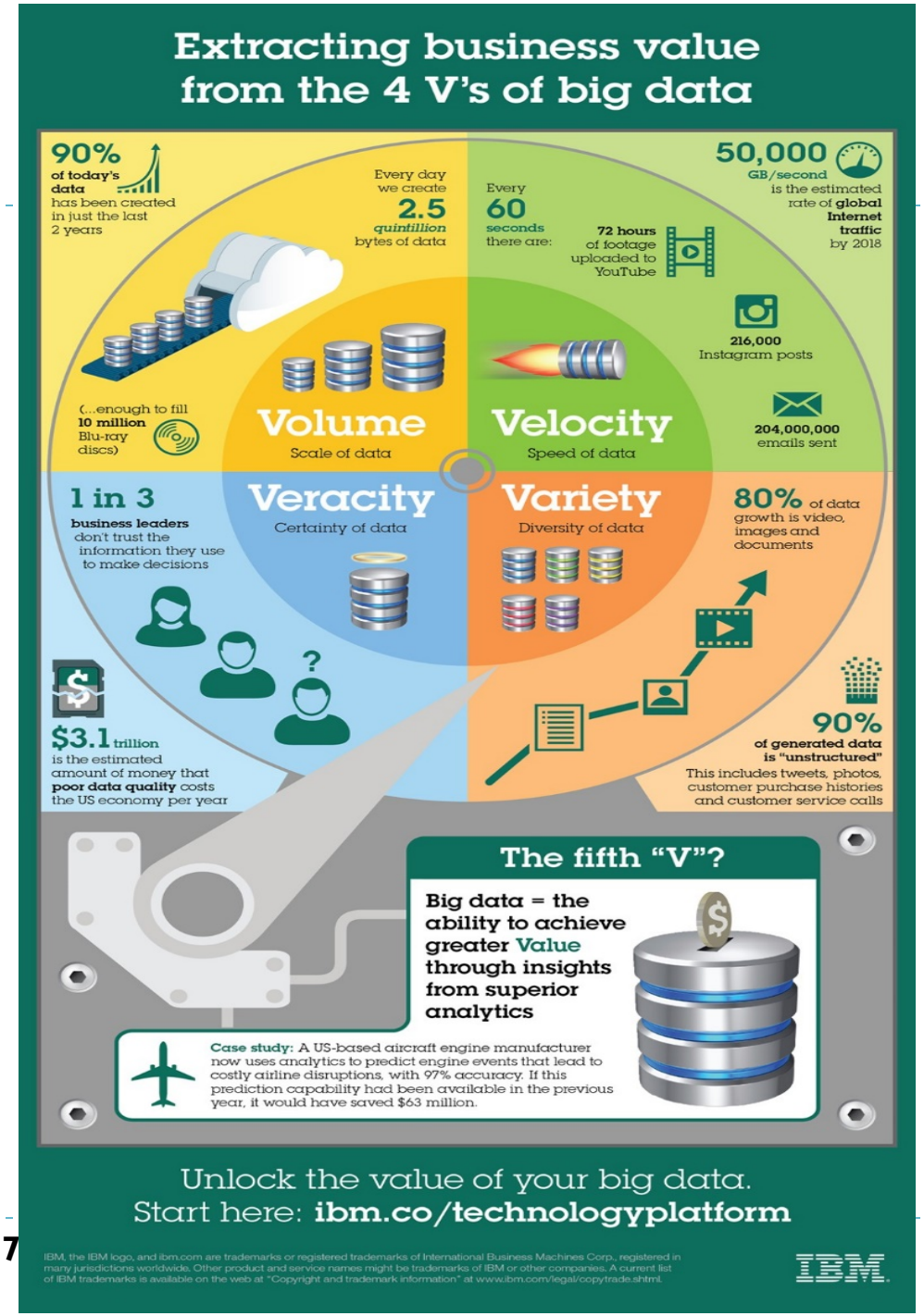
- List down the amount of data that a simple event like going to a movie can generate.
  - You start by searching for a movie on movie review sites, reading reviews about that movie, and posting queries.
  - Purchases tickets online
  - You may tweet about the movie or post photographs of going to the movie on Facebook, Snapchat or Instagram.
  - While travelling to the theatre, your GPS system tracks your course and generates data.
  - Pay for Car parking by credit card. Cinema Security Cameras capture your movements
  - You may post your review of the movie!
- Smartphones, social networking sites, and other media are creating floods of data for companies to process and store.
- When the size of data poses challenges to the ability of typical software technologies like Relational Databases to capture, process, store, and manage data, then we have BIG DATA in hand.

Model of Generating and Consuming Data has Changed

Old model: Few companies are generating data and all others are consuming data

New model: All of us are generating data, and all of us are consuming data


DOMO
Data Never Sleeps 9.0
How much data is generated every minute?
The 2020 pandemic upended everything, from how we engage with each other to how we engage with brands and the digital world. At the same time, it transformed how we eat, how we work and how we entertain ourselves. Data never sleeps and it shows no signs of slowing down. In our 9th edition of the "Data Never Sleeps" infographic, we bring you a glimpse of how much data is created every digital minute in our increasingly data-driven world.
EVERY 1 MINUTE OF THE DAY
PRESENTED BY DOMO
TWITTER USERS POSTS 575k TWEETS
TIKTOK USERS WATCH 167M VIDEOS
GOOGLE CONDUCTS 5.7M SEARCHES
DISCORD USERS SEND 668k MESSAGES
12M PEOPLE SEND AN IMESSAGE
INSTAGRAM USERS SHARE 65k PHOTOS
CLUBHOUSE CREATES 208 ROOMS
FACEBOOK USERS SHARE 240k PHOTOS
SNAPCHAT USERS SEND 2M SNAPCHATS
FACEBOOK LIVE RECEIVES 44M VIEWS
AMAZON CUSTOMERS SPEND $283k
YOUTUBE USERS STREAM 694k HOURS
6M PEOPLE SHOP ONLINE
NETFLIX USERS STREAM 452k HOURS
STRAVA ATHLETES SHARE 1.5k ACTIVITIES
TEAMS CONNECTS 100k USERS
ZOOM HOSTS 856 MINUTES OF WEBINARS
SLACK USERS SEND 148k MESSAGES
VENMO USERS SEND $304k
INSTACART USERS SPEND $67k
As of July 2021, the internet reaches 65% of the world's population and now represents 5.17 billion people—a 10% increase from January 2021. Of this total, 92.6 percent accessed the internet via mobile devices. According to Statista, the total amount of data consumed globally in 2021 was 79 zettabytes, an annual number projected to grow to over 180 zettabytes by 2025.
Global Internet Population Growth
(IN BILLIONS)
3.4
4.3
4.5
5.2
2016
2018
2020
2021
As the world changes, businesses need to change too—and that requires data. Domo gives you the power to make data-driven decisions at any moment, on any device, so that you can make smart choices in a rapidly changing world. Every click, swipe, share, or like tells you something about your customers and what they want, and Domo is here to help you and your business make sense of all of it.
Learn more at domo.com
DOMO



+ 1 more V – Value

Now the "5 Vs"

The Emergence of NoSQL

- No Prescriptive Definition
- Common Characteristics of NoSQL Databases
  - Not using the relational model – some support a SQL dialect though!
  - Running well on clusters often geographically spread
  - Open-Source
  - Use of Commodity Hardware (thus component failure is a standard mode of operation!)
  - Built for the 21st Century Web estates
  - Schema-less
- Come in a variety of shapes and functionality
  - E.g. Key Value Stores, Graph Databases, Document Databases

NoSQL Database Types

- Key Value Stores
  - E.g. Riak(Open Source based on DynamoDB) , Redis, MemcacheD DB, Amazon DynamoDB(not open source), Project Voldemort (Open Source based on DynamoDB)
- Document Databases
  - E.g. MongoDB, CouchDB, Terrastore, OrientDB, RavenDB, Lotus Notes storage facility
- Column Family Databases
  - E.g. Google Big Table(not open source).Cassandra, HBase(Open Source based on Big Table), Hypertable, Graph Databases
- Graph Databases
  - neo4J, Inifinite Graph, OrientDB, FlockDB
- Hadoop Distributed File System (HDFS) – Hadoop
  - Does not fit the NoSQL categories as it is a distributed file system but important in the Big Data World!
- http://nosql-database.org/ Over 200 NoSQL database listed

The Ranking of traditional SQL and NoSQL Database Types

380 systems in ranking, October 2021

Rank Oct 2021	Rank Sep 2021	Rank Oct 2020	DBMS	Database Model	Score Oct 2021	Score Sep 2021	Score Oct 2020
1.	1.	1.	Oracle	Relational, Multi-model	1270.35	-1.19	-98.42
2.	2.	2.	MySQL	Relational, Multi-model	1219.77	+7.24	-36.61
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	970.61	-0.24	-72.51
4.	4.	4.	PostgreSQL	Relational, Multi-model	586.97	+9.47	+44.57
5.	5.	5.	MongoDB	Document, Multi-model	493.55	-2.95	+45.53
6.	6.	↑ 8.	Redis	Key-value, Multi-model	171.35	-0.59	+18.07
7.	7.	↓ 6.	IBM Db2	Relational, Multi-model	165.96	-0.60	+4.06
8.	8.	↓ 7.	Elasticsearch	Search engine, Multi-model	158.25	-1.98	+4.41
9.	9.	9.	SQLite	Relational	129.37	+0.72	+3.95
10.	10.	10.	Cassandra	Wide column	119.28	+0.29	+0.18
11.	11.	11.	Microsoft Access	Relational	116.38	-0.56	-1.87
12.	12.	12.	MariaDB	Relational, Multi-model	102.59	+1.90	+10.82
13.	13.	13.	Splunk	Search engine	90.61	-0.99	+1.21
14.	14.	↑ 15.	Hive	Relational	84.74	-0.83	+15.19
15.	15.	↑ 17.	Microsoft Azure SQL Database	Relational, Multi-model	79.72	+1.46	+15.32
16.	16.	16.	Amazon DynamoDB	Multi-model	76.55	-0.38	+8.14
17.	17.	↓ 14.	Teradata	Relational, Multi-model	69.83	+0.15	-5.96
18.	↑ 21.	↑ 64.	Snowflake	Relational	58.26	+6.19	+52.32
19.	↓ 18.	↑ 21.	Neo4j	Graph	57.87	+0.24	+6.53
20.	↓ 19.	↓ 19.	SAP HANA	Relational, Multi-model	55.28	-0.96	+1.04
21.	↓ 20.	↑ 23.	FileMaker	Relational	52.84	+0.52	+5.46
22.	22.	↓ 20.	Solr	Search engine, Multi-model	51.17	+1.36	-1.31
23.	23.	↓ 18.	SAP Adaptive Server	Relational, Multi-model	48.59	+1.57	-6.58
24.	24.	↓ 22.	HBase	Wide column	45.20	+0.14	-3.16
25.	25.	↓ 24.	Google BigQuery	Relational	43.79	-0.13	+9.38

Source:DB-Engines systems https://db-engines.com/en/ranking

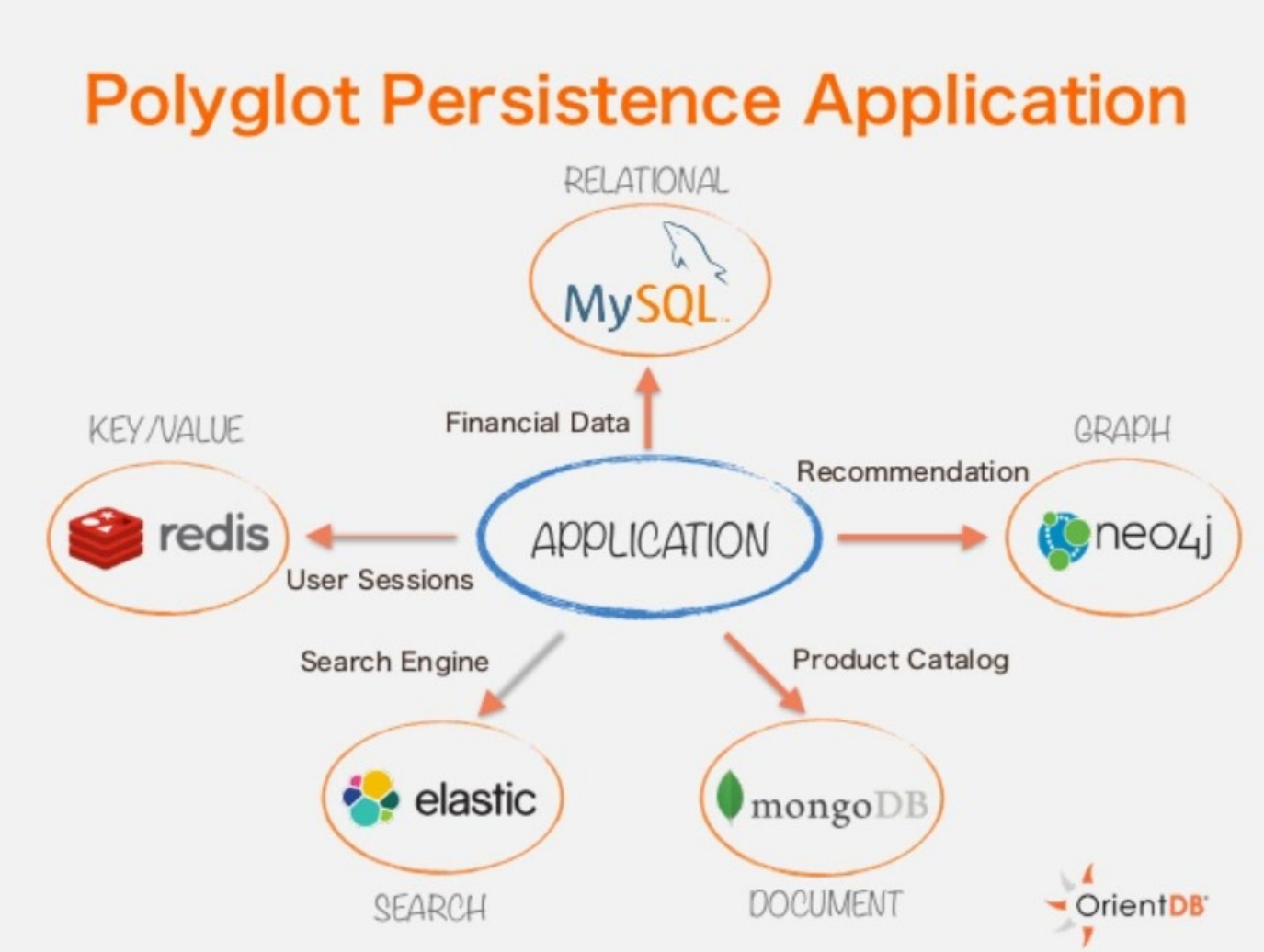
NoSQL Main Motivations and Drivers

1. Avoidance of Unneeded Complexity
2. High Throughput
3. Horizontal Scalability and running on Commodity Hardware
4. Avoidance of Expensive Object- to-Relational Database Mapping
5. The current "One size fit's all" Database Thinking **WAS** and **IS** wrong
6. The myth of effortless Distribution and Partitioning of Centralized Data Models

nosql

- The word now is **Polyglot Persistence**!

Polyglot Persistence



NoSQL Pros and Cons

- Pros
  - High Scalability
    - To expand horizontally using low-end commodity servers
  - Manageability and administration
    - NoSQL databases are designed to mostly work with automated repairs, distributed data, and simpler data models
  - Relative Low Cost
    - Designed to work with a cluster of cheap commodity servers, enabling the users to store and process more data at a low cost
  - Flexible Data Models
    - Can evolve data that is stored
    - No rigid data models like we have seen in RDBMS!

NoSQL Pros and Cons

- Cons
  - Maturity
    - Some of the popular NoSQL databases are evolving and beginning to mature while others have not reached that point
  - Support
    - Varied depending on the NoSQL database
  - Limited Query Capabilities
    - In some case you can only carry out POST and GET and search on the key while in other cases you can search what is stored
  - Administration
    - Installation and maintenance can still be hindrance
  - Expertise
    - Has improved through their use but expertise for the less popular NoSQL Databases can be a problem.

NoSQL and ACID

- ACID properties are generally not supported especially around atomicity and consistency
- An operation that touches one "record" is generally atomic
- Operations that read or write more that one "record" are often
  - Non-atomic, non-consistent, non-isolated i.e. ACID properties not preserved
  - Varying degrees of implementations
    - Must examine the NoSQL technology to determine it's support
  - **It is important to get your unit of storage correct in NoSQL environment**
    - **e.g. the structure of the document in MongoDB**
    - **Unit of Storage we will call the "Aggregate"**