Cloud & Big Data a perfect marriage?

Patrick Valduriez

Zenith

informatics mathematics

UNIVERSITÉ MONTPELLIER 2
SCIENCES ET TECHNIQUES

cnrs
advancing the frontiers

LIRMM
Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier

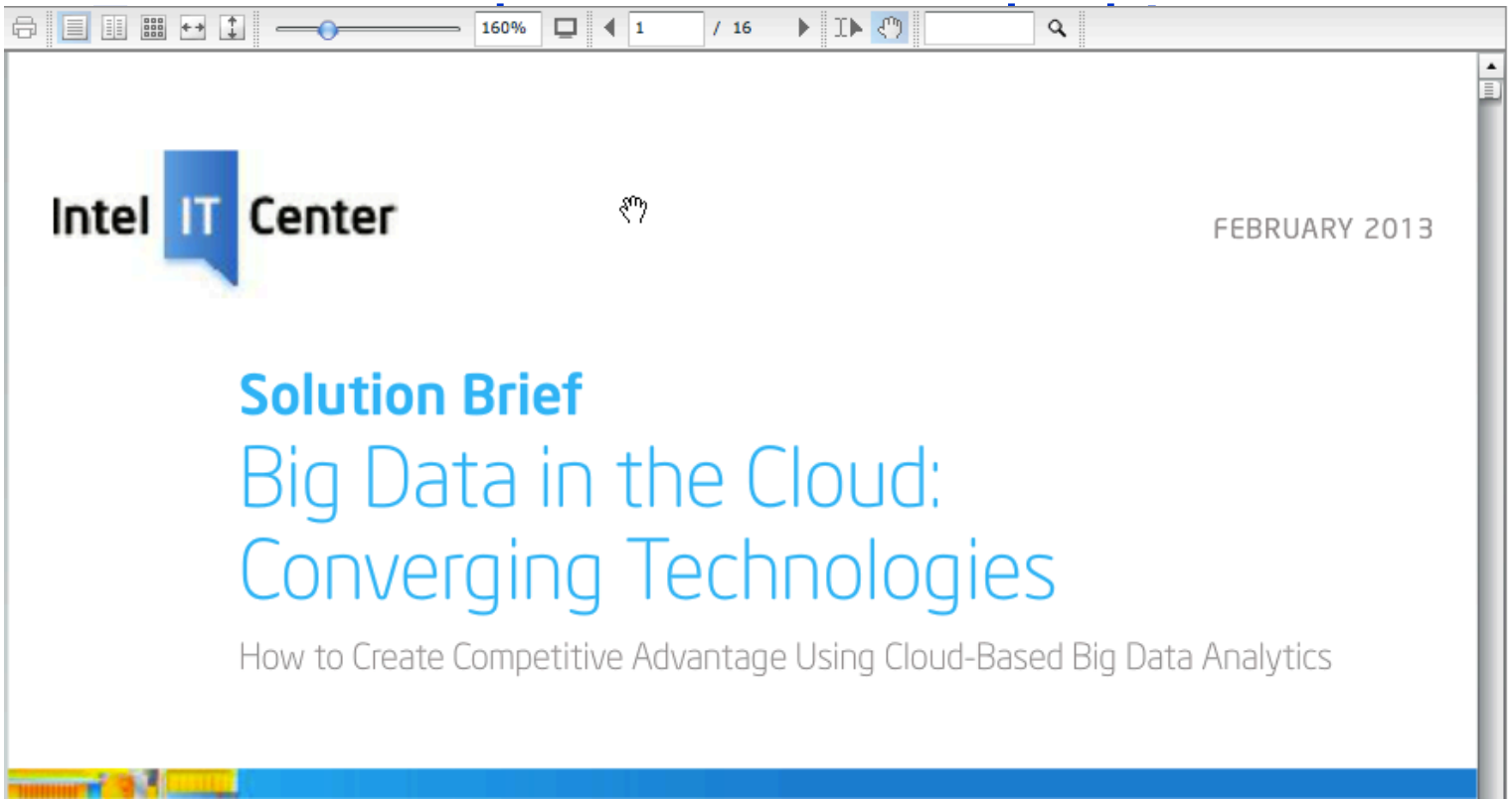
Cloud & Big Data: the hype!

Cloud & Big Data: the hype!

Behind the Hype?

- Every one who wants to make big money
  - Intel, IBM, Microsoft, Oracle, Google, Facebook, Amazon, …
- In a big market
  - $18 billion in 2013, $24 billion in 2016
  - Source: International Data Corp. (IDC)

Behind the Hype?



Behind the Hype?

IBM

developerWorks® Technical topics Evaluation software Community Events

developerWorks > Technical topics > Big data > Technical library >

Big data in the cloud

Data velocity, volume, variety, veracity

Microsoft Enlists the Cloud, Big Data for Scientific Research

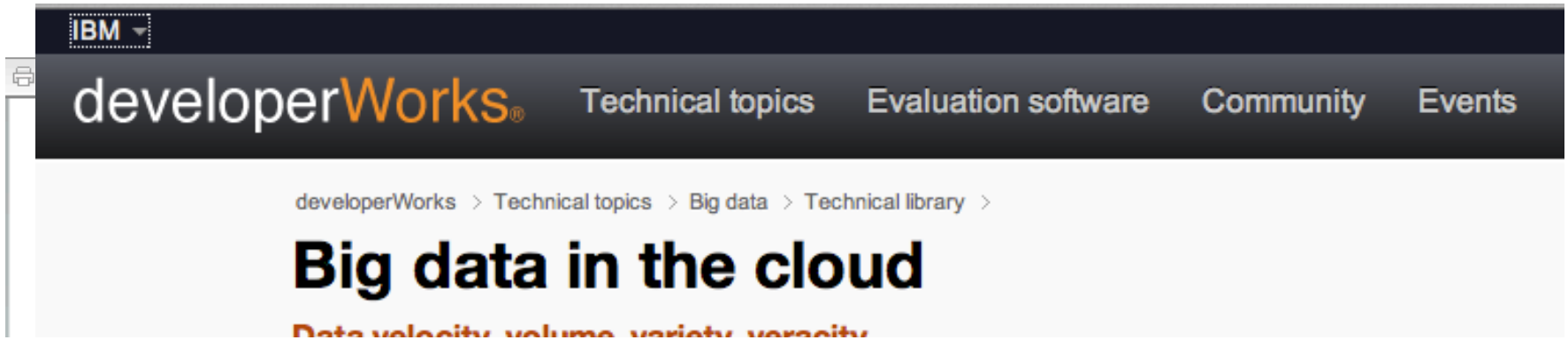
By Pedro Hernandez | Posted 2013-09-11 Email Print

Share 0 Tweet 8 Google + 0 Share 2 Like 3 Recommend 3

Stow the lab coat. Microsoft is positioning Windows Azure as a cloud platform for scientific discovery and innovation.

Microsoft isn't only hoping to attract enterprises to its Windows Azure cloud platform. The company is also courting scientists with the promise of "cloud power" to fuel new discoveries.

Behind the Hype?



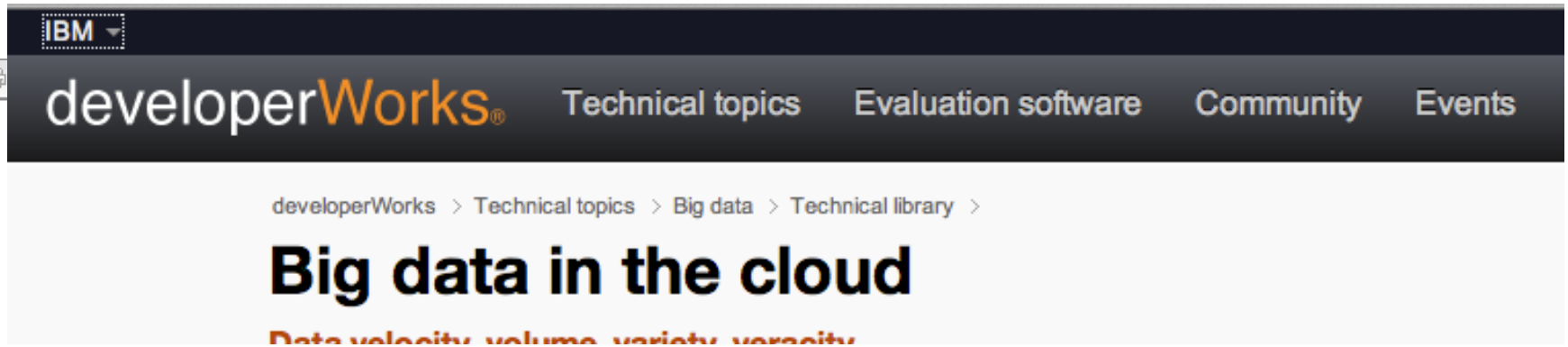
Cloudera and Oracle

Oracle Big Data Appliance brings big data solutions to mainstream enterprises. Built using industry-standard hardware and CDH, the Big Data Appliance is designed and optimized for big data workloads. By integrating the key components of a big data platform into a single product, Oracle Big Data Appliance delivers an affordable, scalable and fully supported big data infrastructure without the risks of a custom built solution.

Stow the lab coat. Microsoft is positioning Windows Azure as a cloud platform for scientific discovery and innovation.

Microsoft isn't only hoping to attract enterprises to its Windows Azure cloud platform. The company is also courting scientists with the promise of "cloud power" to fuel new discoveries.

Behind the Hype?



Cloudera a

Oracle Big Data A
hardware and CD
the key componer
affordable, scalab

Cloud et Big Data : Bull s'offre fastConnect

Business : *FastConnect est une société française leader dans les architectures applicatives distribuées. Le montant de la transaction n' pas été dévoilé.*

Par La rédaction de ZDNet.fr | Lundi 30 Septembre 2013

Suivre @zdnetfr

courting scientists with the promise of "cloud power" to fuel new discoveries.

Cloud & Big Data: how?

- *But do they have the same goals ?*

Outline of the Talk

- Big data
- Cloud
- Cloud & big data
- Principles of data management
- Big data in the cloud
- Problems and research directions
- The CoherentPaaS project

Big Data

Big Data: what is it?

- A buzz word!
  - With different meanings depending on your perspective
    - E.g. 10 terabytes is big for an OLTP system, but small for a web search engine
- A definition (Wikipedia)
  - Consists of data sets that grow so *large* that they become awkward to work with using on-hand data management tools
  - *But size is only one dimension of the problem*
- How *big* is big?
  - Moving target: terabyte (10^{12} bytes), petabyte (10^{15} bytes), exabyte (10^{18}), zetabyte (10^{21})
  - Landmarks in DBMS products
    - 1980: Teradata database machine
    - 2010: Oracle Exadata database machine

Why Big Data Today?

- Overwhelming amounts of data
  - Exponential growth, generated by all kinds of networks, programs and devices
    - E.g. Web 2.0 (social networks, etc.), mobile devices, computer simulations, satellites, radiotelescopes, sensors, etc.
- Increasing storage capacity
  - Storage capacity has doubled every 3 years since 1980 with prices steadily going down
    - 1 Gigabyte for: $1M in 1982, $1K in 1995, $0.12 in 2011
- Very useful in a digital world!
  - Massive data => high-value information and knowledge
  - Critical for data analysis, decision support, forecasting, intelligence, research, (data-intensive) science, etc.

Some Estimates*

- 1,8 zetaoctets: an estimation for the data stored by humankind in 2011
- 40 zetaoctets in 2020
- But
  - Less than 1% of big data is analyzed
  - Less than 20% of big data is protected

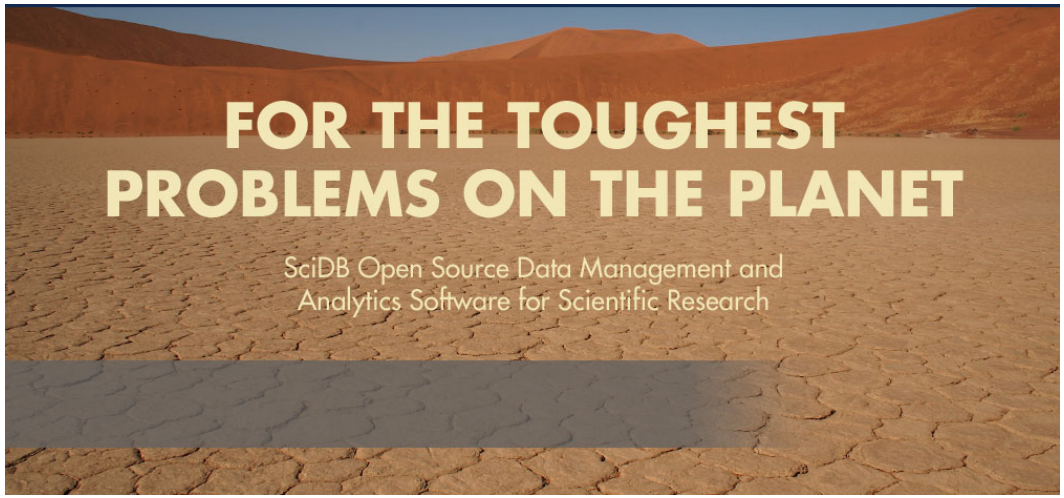
* Source: Digital Universe study of IDC, december 2012

Big Data Dimensions: the three V's

- Volume
  - Refers to massive amounts of data
  - Makes it hard to store, manage, and analyze (big analytics)
- Velocity
  - Continuous data streams are being produced
  - Makes it hard to perform online processing
- Variety
  - Different data formats, different semantics, uncertain data, multiscale data, etc.
  - Makes it hard to integrate and analyze
- Other V's
  - Veracity: are the results meaningful?
  - Validity: is the data correct and accurate?
  - Volatility: how long do you need to store this data?

Example: Scientific Data

- **Big data**



- With additional characteristics
  - Manipulated through complex, distributed *workflows*
  - Important *metadata* about experiments and their provenance
  - Mostly append-only (with rare updates)
    - Good news as transactions are hard to deal with

Cloud

Cloud Computing

- The vision
  - On demand, reliable services provided over the Internet (the "cloud") with easy access to virtually infinite computing, storage and networking resources
- Simple and effective!
  - Through simple Web interfaces, users can outsource complex tasks
    - Data storage, system administration, application deployment
  - The complexity of managing the infrastructure gets shifted from the users' organization to the cloud provider
- Capitalizes on previous computing models
  - Web services, utility computing, cluster computing, virtualization, grid computing
- Major players
  - Amazon, Microsoft, Google, IBM, Intel, etc.

Cloud Benefits

- Reduced cost
  - Customer side: the IT infrastructure needs not be owned and managed, and is billed only based on resource consumption
  - Cloud provider side: by sharing costs for multiple customers, reduces its cost of ownership and operation to the minimum
- Ease of access and use
  - Customers can have access to IT services anytime, from anywhere with an Internet connection
- Quality of Service (QoS)
  - The operation of the IT infrastructure by a specialized, experienced provider (including with its own infrastructure) increases QoS
- Elasticity
  - Easy for customers to deal with sudden increases in loads by simply creating more virtual machines (VMs)

Safety and Security?

TECH

More: Cloud Computing Enterprise Amazon

Amazon's Cloud Crash Disaster Permanently Destroyed Many Customers' Data

HENRY BLODGET | APR. 28, 2011, 7:10 AM | 92,357 | 75

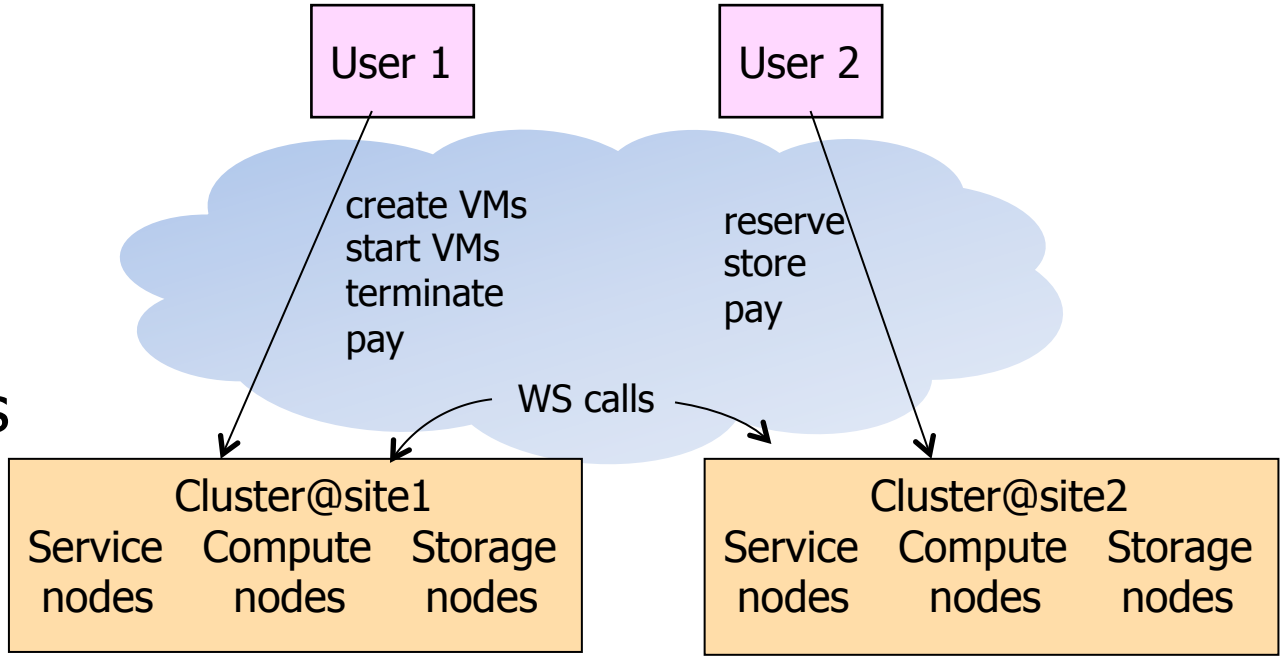
Security Solutions

- Internal (or private) cloud (vs public cloud)
  - The use of cloud technologies but in a private network: much tighter security
  - But reduced cost advantage: the infrastructure is not shared with other customers
  - Compromise: **hybrid cloud** (internal cloud + public cloud)
- Virtual private cloud: VPN within a public cloud with security services
  - Promise of a similar level of security as an internal cloud and tighter integration with internal cloud security
  - *But such security integration is complex*

Much room for innovation

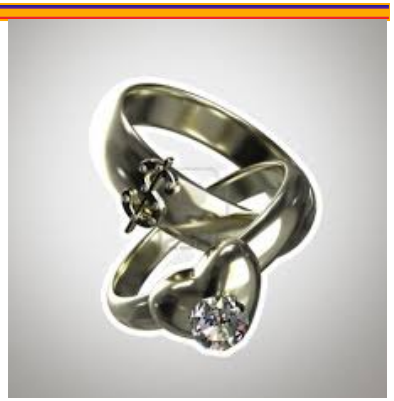
Cloud Architecture

- Like grid, but less distribution, more homogeneity and transparency
- Very different customers
- Replication across sites for high availability
- Scalability, SLA, accounting and pricing essential



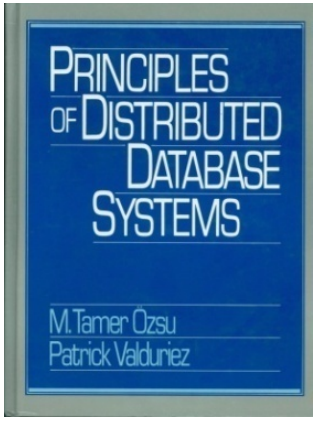
Cloud & big data

A Marriage of Convenience?

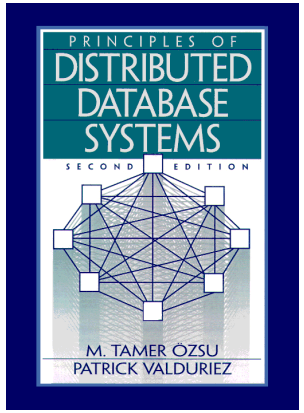


- Cloud and big data have different goals
  - Big data aims at added value and operational performance
  - Cloud targets flexibility and reduced cost
- But they can help each other by
  1. Encouraging organizations to outsource more and more strategic internal data in the cloud
  2. Get value out of it, e.g. by integrating their data with external data, through big data analytics at affordable cost

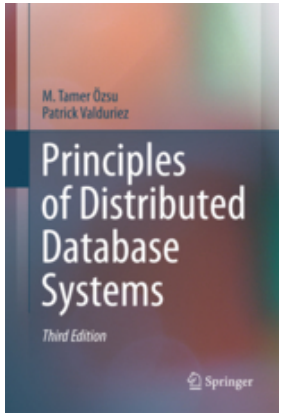
Principles of Data Management



Prentice Hall 1991
560p
distributed relational DBMS



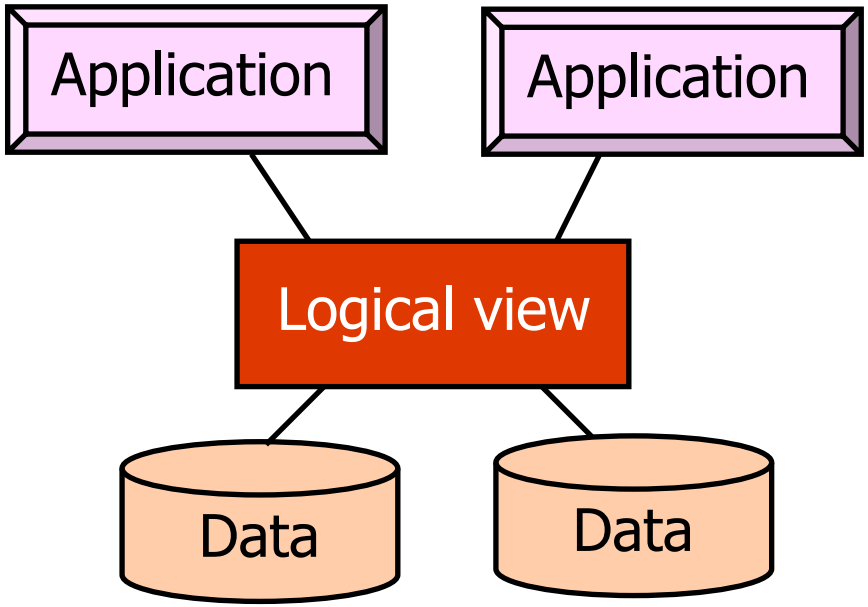
Prentice Hall 1999
660p
+ parallel DBMS



Springer 2011
850p
+ P2P & Cloud

Fundamental Principle

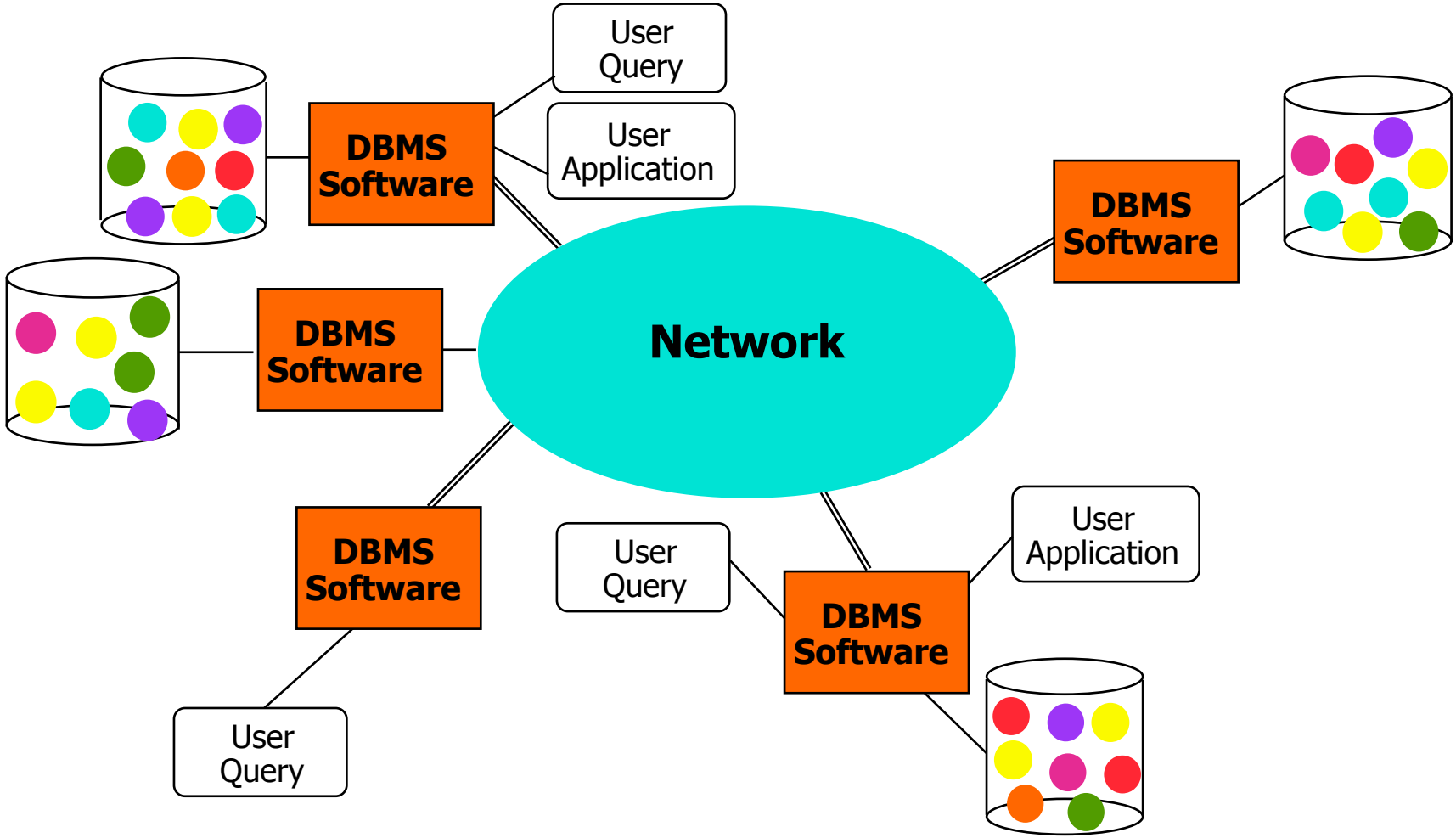
- *Data Independence:* enables hiding complexity
  - Distribution, implementation, etc.



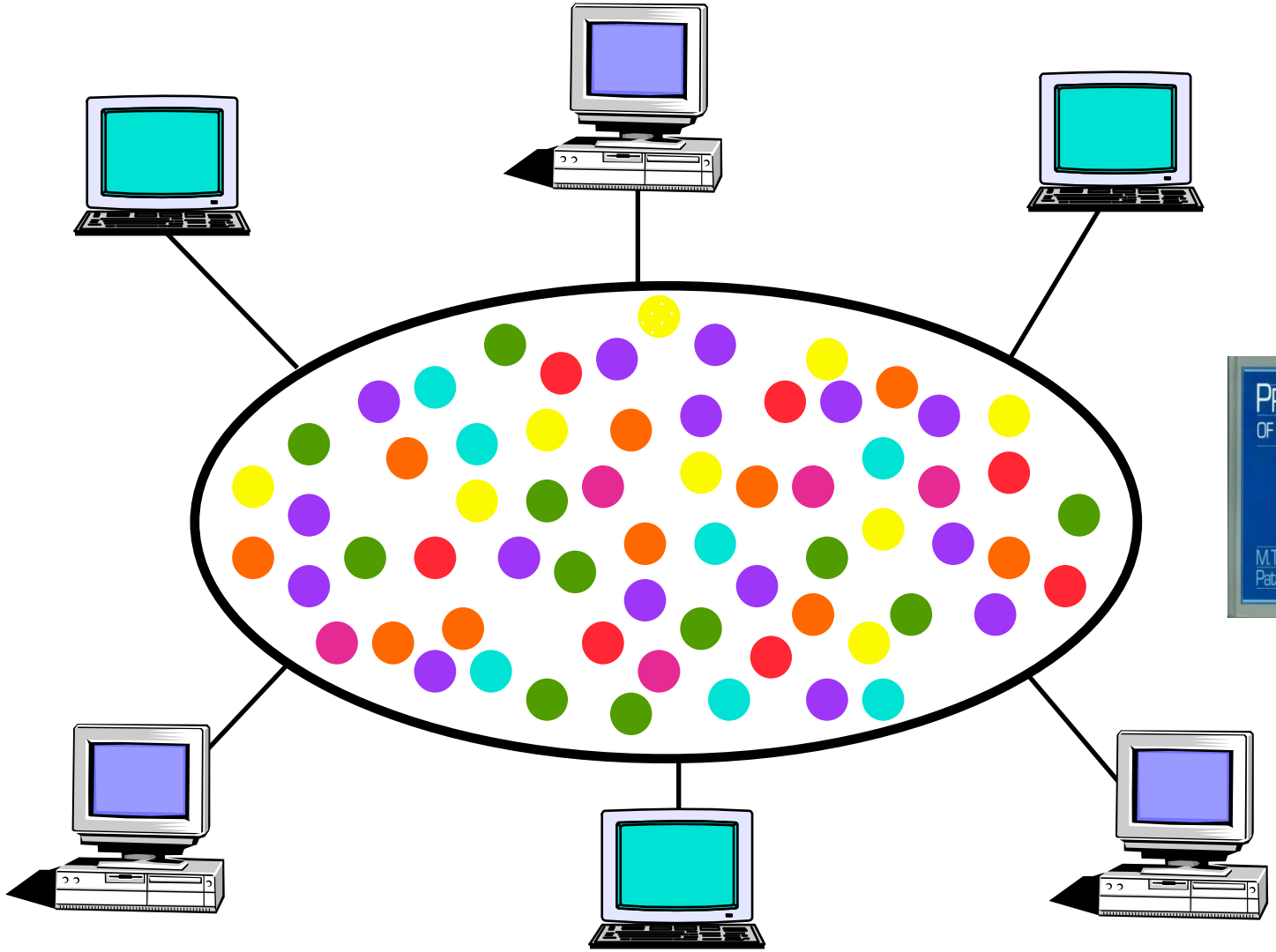
Provision for high-level services

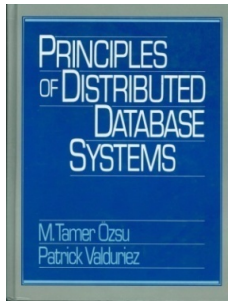
- Queries (SQL, Xquery)
- Automatic optimization & parallelization
- Strong consistency with ACID (Atomicity, Consistency, Isolation, Durability) transactions
- Privacy, security
- And many others

Distributed Database – System View

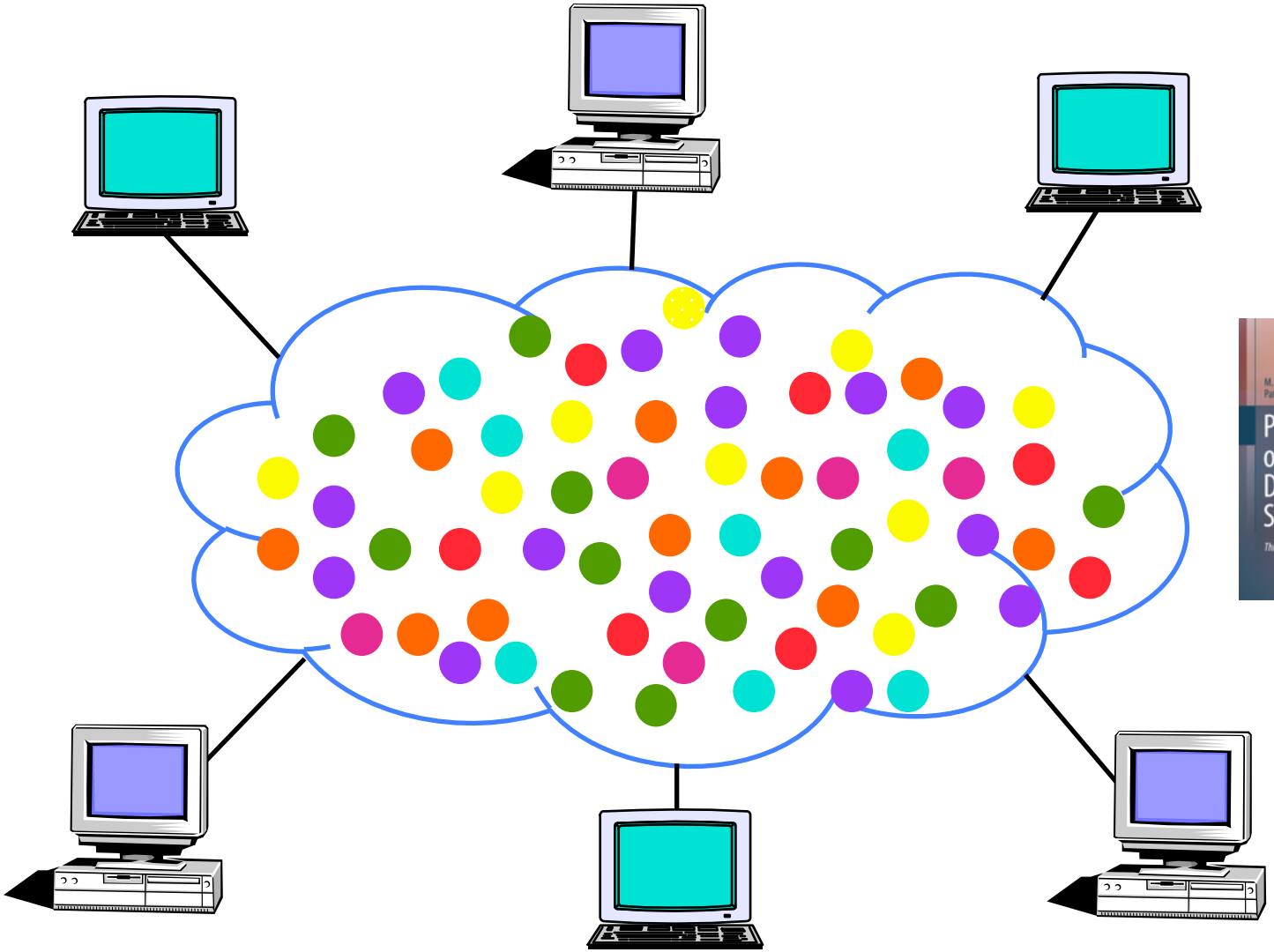


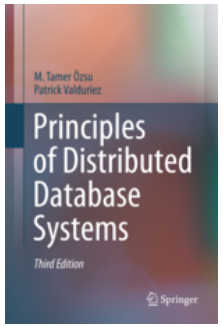
Distributed Database – User View (1991)





Distributed Database – user view in 2011



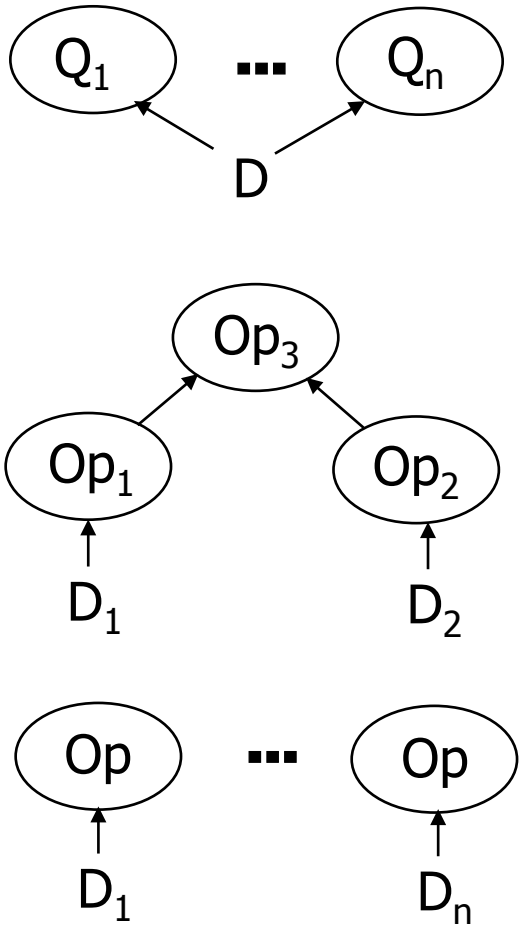


Big data processing

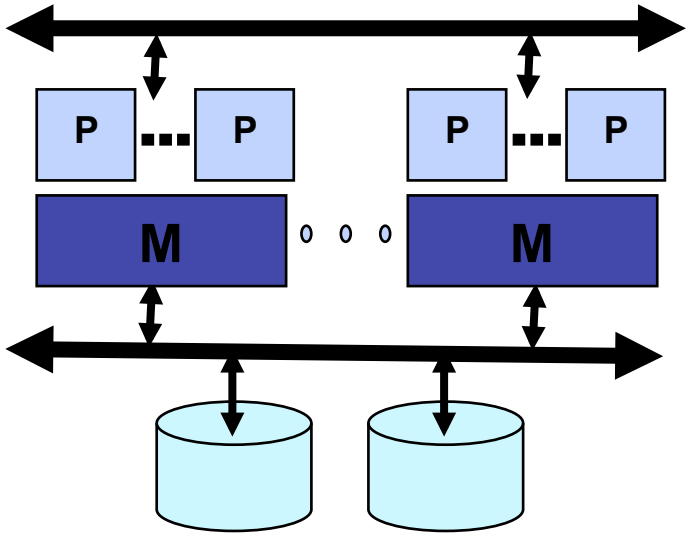
- Exploit massive parallel processing
  - Computers with lots of processors (CPUs, GPUs), main memory (RAM, flash) and disk (HDD, SSD)
- To obtain
  - *High performance* through data-based parallelism
    - High throughput for OLTP loads
    - Low response time for OLAP queries
  - *High availability* and reliability through data replication
  - *Extensibility* of the architecture
    - With speed-up, scale-up, scale-out

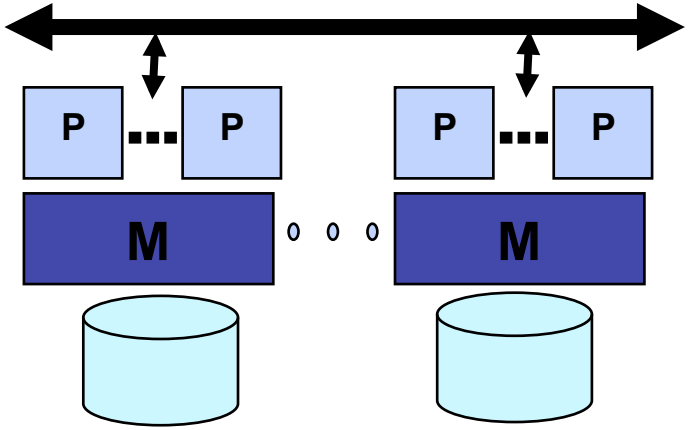
Data-based Parallelism

- Inter-query
  - Different queries on the same data
  - For concurrent queries
- Inter-operation
  - Different operations of the same query on different data
  - For complex queries
- Intra-operation
  - The same operation on different data
  - For large queries



Shared-disk vs Shared-nothing





- Disk interconnect
- Cache coherency
+ Simple for admins
+ Scales to VLDB

- Data partitioning
- Distributed transactions
+ Cost/performance
+ Scales to XLDB

OLAP vs OLTP Workloads

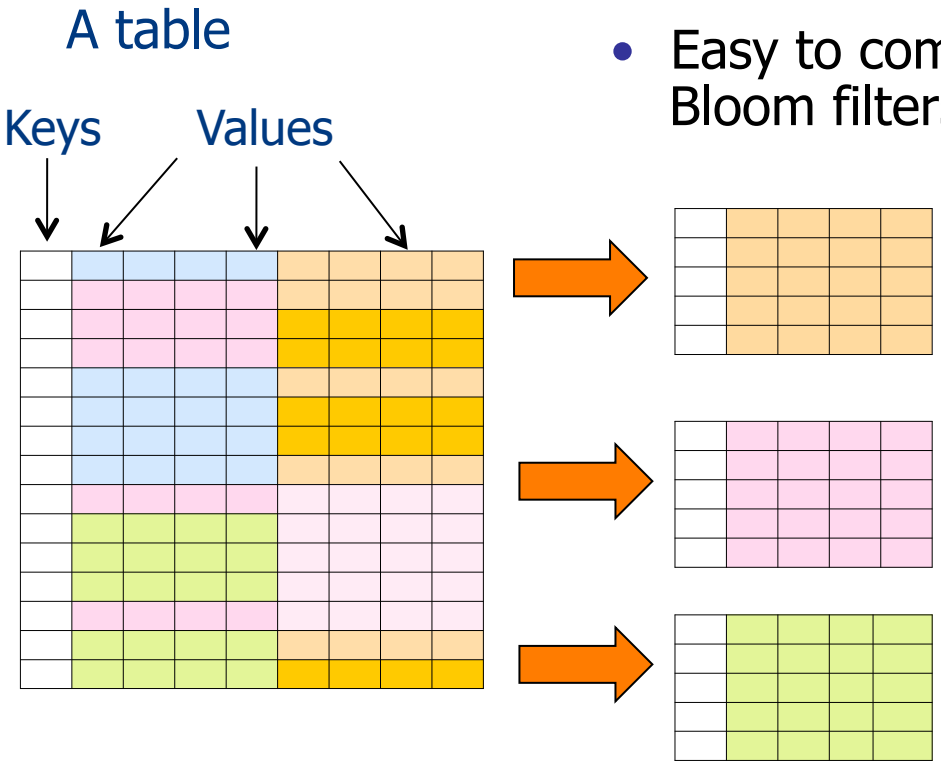
- Online Transaction Processing (OLTP)
  - Operational databases of average sizes (TB), write-intensive
  - ACID transactions, strong data protection, response time guarantees
  - Corporate data can't get lost or stolen
  - Shared-disk preferred
- Online Analytical Processing (OLAP)
  - Historical databases of very large sizes (PB), read-intensive
  - Relaxed ACID properties
  - Sensitive data can be anonymized
  - Shared-nothing cost-effective

In the cloud: OLAP easier, OLTP more difficult but doable

- e.g. UCSB ElasTraS, MS SQL Azure, MIT Relational Cloud

Data Partitioning

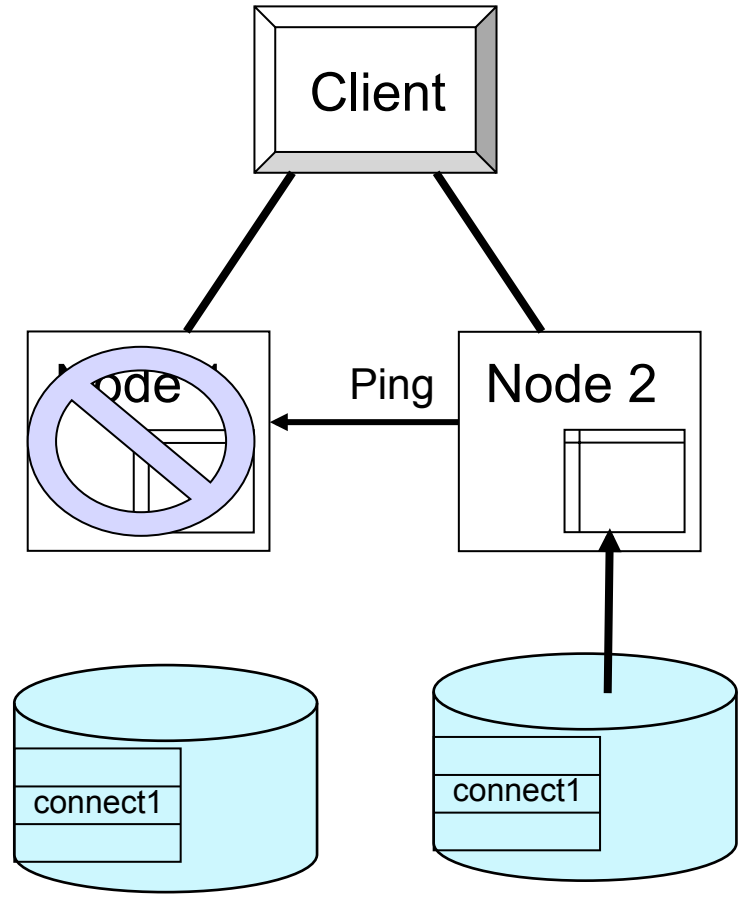
- Vertical partitioning
  - Basis for column-oriented DBMS (e.g. MonetDB, Vertica): efficient for OLAP queries
  - Easy to compress, e.g. using Bloom filters



- Horizontal partitioning (sharding)
  - Basis for parallel DBMS and MapReduce
  - Shards can be stored and replicated at different nodes

Replication and Failover

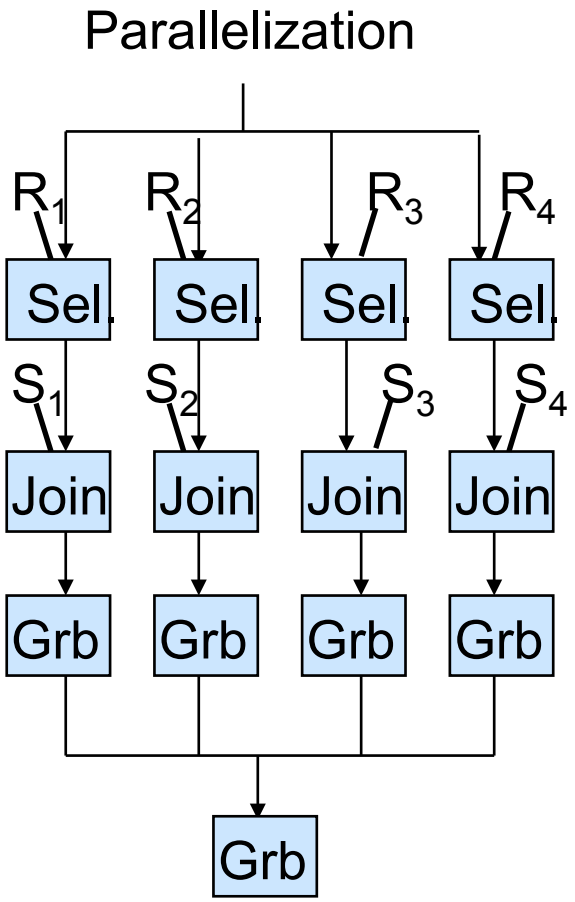
- Replication
  - The basis for fault-tolerance and availability
  - Have several copies of each shard
- Failover
  - On a node failure, another node detects and recovers the node's tasks



Parallel Query Processing

1. Query parallelization
   - Produces an optimized parallel execution plan
   - Based on partitioning
2. Parallel execution
   - Relies on parallel main memory algorithms for operators
   - Use of hashed-based algorithms

Select … from R,S
where …group by…



Big Data in the Cloud

Problem and solution

- Why not relational DBMS?
  - "One size fits all" has reached the limits
  - Not designed for loosely structured data
- Cloud users and application developers
  - In very high numbers, with very diverse expertise but very little DBMS expertise

"New" data management solutions

- Distributed file systems: GFS, HDFS, …
- NOSQL systems: Amazon SimpleDB, Google Base, Bigtable, Hbase, MongoDB, etc.
- Parallel programming frameworks: MapReduce, Dryad

And new architectures

- BigTable/GFS, Hbase/HDFS, MapReduce/GFS, MapReduce/HDFS

NOSQL (Not Only SQL): definition

- Specific DBMS, for web-based data
  - Specialized data models
    - Principle: No one size fits all
    - Key-value, table, document, graph
  - Trade relational DBMS properties
    - Full SQL, ACID transactions, data independence
  - For
    - Simplicity (schemaless, basic API)
    - Scalability and performance
    - Flexibility for the programmer (integration with programming language)

- NB: SQL is just a language and has nothing to do with the story

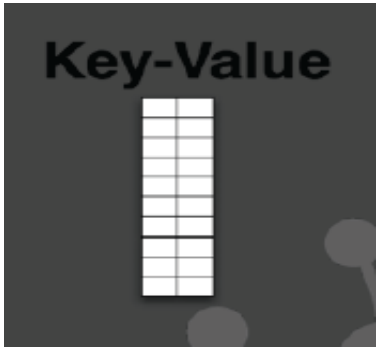
NoSQL Approaches

- Characterized by the data model, in increasing order of complexity:
  1. key-value: DynamoDB, Cassandra, Voldemort
  2. big table: Bigtable, Haddop Hbase, Accumulo
  3. document: 10gen MongoDB, Expresso
  4. graph: Neo4J, Pregel, DEX

- What about object DBMS or XML DBMS?
  - Were there much before NoSQL
  - Sometimes presented as NoSQL
  - But not designed for scaling

Key-value store: DynamoDB

- The basis for many systems
  - Cassandra, Voldemort
- Simple (key, value) data model
  - Key = unique id
  - Value = a small object (< 1 Mbyte)
- Simple queries
  - Put (key, value)
  - Get (key)
- Replication and eventual consistency
  - If no more updates, the replicas get mutually consistent
- No security
  - Assumes the environment is secured (cloud)
- High availability, performance and scalability using P2P techniques in a SN cluster
- Integration with MapReduce

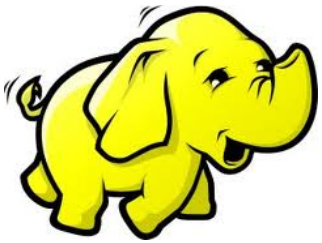


Google Bigtable

- Database storage system for a SN cluster
  - Uses GFS to store structured data, with fault-tolerance and availability
- Used by popular Google applications
  - Google Earth, Google Analytics, Google+, etc.

K_1	A_{11}	A_{12}	...	A_{1n}
K_2	A_{21}	A_{22}	...	A_{2n}
⋮			*M* and *n* very big	
K_m	A_{m1}	A_{m2}	...	A_{mn}

- The basis for popular Open Source implementations
  - Hadoop Hbase on top of HDFS (Apache & Yahoo)
- Specific data model that combines aspects of row-store and column-store DBMS
  - Rows with multi-valued, timestamped attributes
- Dynamic partitioning of tables for scalability



Bigtable Language

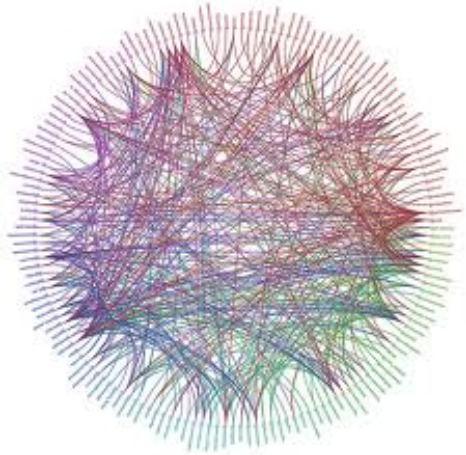
- No such thing as SQL
- Basic API for defining and manipulating tables, within a programming language such as C++
  - No impedance mismatch
  - Various operators to write and update values, and to iterate over subsets of data, produced by a scan operator
  - Various ways to restrict the rows, columns and timestamps produced by a scan, as in relational select, but no complex operator such as join or union
  - Transactional atomicity for single row updates only

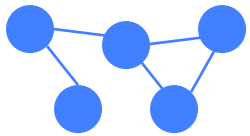
Document DBMS: MongoDB

- Objectives: performance and scalability as in (key, value) stores, but with typed values
  - A document is a collection of (key, typed value) with a unique key (generated by MongoDB)
- Data model and query language
  - Based on the Binary JSON format
- No schema, no join, no complex transaction
- Sharding, replication and failover
- Secondary indices
- Integration with MapReduce

Graph DBMS: Neo4J

- Applications with very big graphs
  - Billions of nodes and links
  - Social networks, hypertext documents, linked open data, etc.
- Direct support of graphs
  - Data model, API, query language
  - Implemented by linked lists on disk
  - Optimized for graph processing
  - Transactions
- Implemented on SN cluster
  - Asymmetric replication
  - Graph partitioning





NoSQL versus Relational

- The techniques are not new
  - Database machines, SN cluster
  - But very large scale
- Pros NoSQL
  - Scalability, performance
  - APIs suitable for programming
- Pros Relational
  - Strong consistency, transactions
  - Standard SQL, many tools (OLAP cubes, BI, etc.)
- Towards NoSQL/Relational hybrids?
  - Google F1: "combines the scalability, fault tolerance, transparent sharding, and cost benefits so far available only in NoSQL systems with the usability, familiarity, and transactional guarantees expected from an RDBMS"

MapReduce

- Parallel programming framework
  - Invented by Google, proprietary (and protected by software patents)
- For data analysis of very large data sets
  - Highly dynamic, irregular, schemaless, etc.
  - SQL or Xquery too heavy
- New, simple parallel programming model
  - Data structured as (key, value) pairs
    - E.g. (doc-id, content), (word, count), etc.
  - Functional programming style with two functions to be given by the programmer:
    - Map(key, value) -> ikey, ivalue
    - Reduce(ikey, list (ivalue)) –> list(fvalue)
- Implemented on GFS on very large clusters
- The basis for popular implementations
  - Hadoop, Hadoop++, Amazon MapReduce, etc.

Problems and Research Directions

Problems

- ACID properties abandoned within and across data stores
- Wide diversification of APIs and divergence of the programming paradigms used for different types of data
- This makes it very hard to develop new cloud applications and services with correct semantics, requiring tremendous programming effort and expertise

Research Directions

- Basic techniques are not new
  - Data partitioning, replication, indexing, parallel hash algorithms, etc.
  - But need to scale up
- Much room for research and innovation
  - Big data integration
  - Big data analytics
  - Data consistency and transaction support
  - Data privacy and security
  - Data-oriented scientific workflows
  - Uncertain data management
  - Data semantics, recommendation, etc.

CoherentPaaS

The CoherentPaaS Project

- European FP7 IP 2013-2016 (€5 million)
  - **U. Madrid**, INRIA Zenith, FORTH, ICCS, INESC
  - MonetDB, QuartetFS, Sparsity, Neurocom, Portugal Telecom
- Goal: a cloud PaaS with
  - A rich set of cloud data management technologies
  - A common query language to unify the programming models of all systems
  - Holistic coherence across data stores using a scalable, transactional management system

Common Query Language: objectives

- Design an SQL-like query language to query multiple databases (SQL, NoSQL) in a cloud
  - Common data model
  - Common query language
- Design a query engine for that language
  - Compiler/optimizer
    - To produce an execution plan
  - Query runtime
    - To run the query, by calling the databases and integrating the results
- Validate with a prototype
  - With multiple DBs: MonetDB, Dex, MongoDB, etc.

Our Design Choices

- Data model: schemaless, table-based
  - With rich data types
    - To allow computing on typed values
  - No global schema and schema mappings to define
- Query language: functional-style SQL*
  - Can represent all query building blocks as functions
    - A function can be expressed in one of the DB languages
  - Function results can be used as input to subsequent functions
    - Elegant way to pass data between different DBs

- P. Valduriez, S. Danforth. Functional SQL, an SQL Upward Compatible Database Programming Language. Information Sciences, 1992.
- C. Binnig et al. FunSQL: it is time to make SQL functional. EDBT/ICDT Conference, 2012.

Cloud & Bigdata: what's next ?