Mobile and Heterogeneous databases
Database System Architecture

A.R. Hurson
Computer Science
Missouri Science & Technology

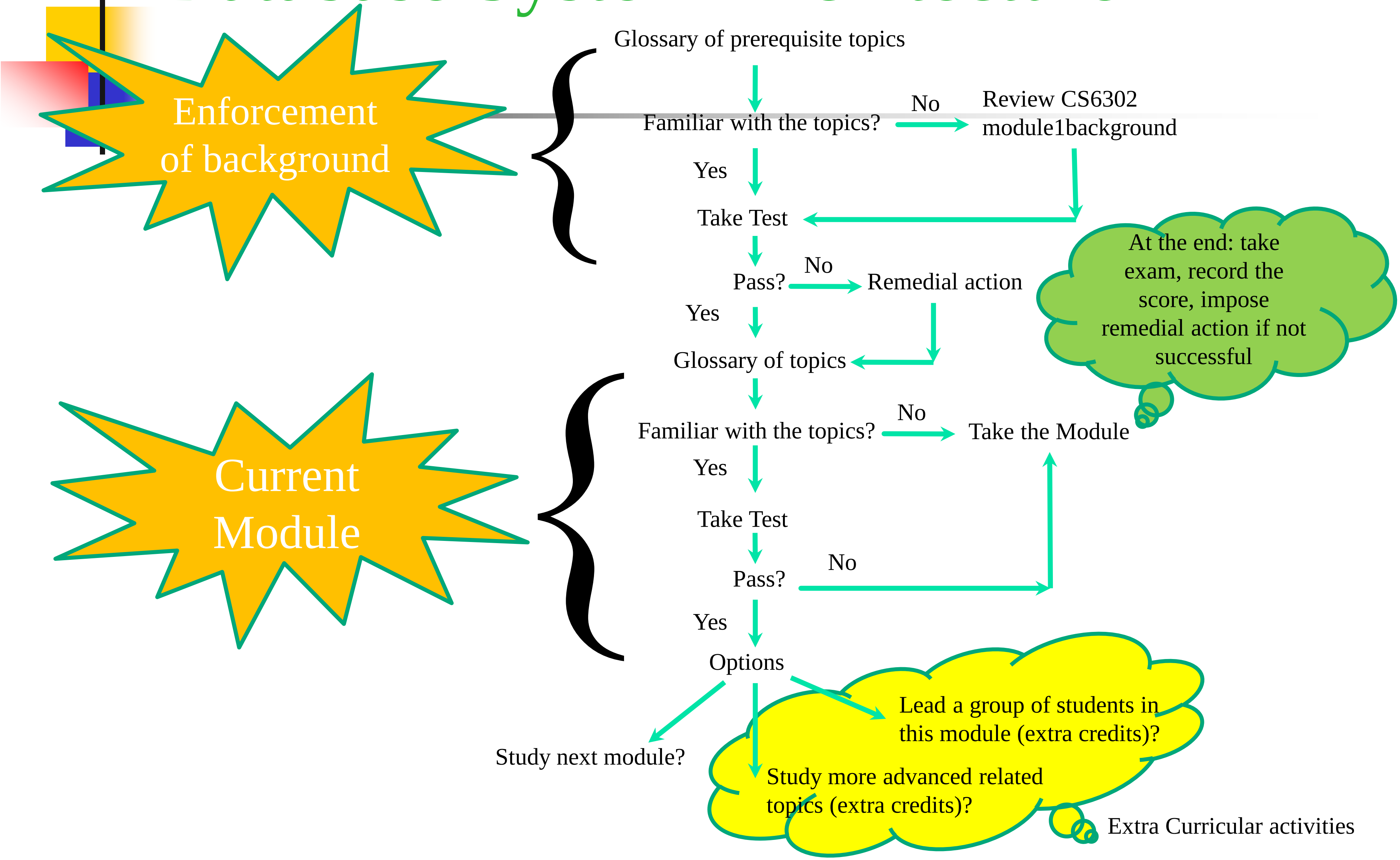
Database System Architecture

Note, this unit will be covered in four lectures. In case you finish it earlier, then you have the following options:

1) Take the early test and start CS6302.module2
2) Study the supplement module (supplement CS6302.module1)
3) Act as a helper to help other students in studying CS6302.module1

Note, options 2 and 3 have extra credits as noted in course outline.

Database System Architecture



Database System Architecture

- You are expected to be familiar with:
  - Centralized database configuration
- If not, you need to study CS6302.module1.background

Database System Architecture

- In this module, we will:
  - Define a simple database space,
  - Base on the parameters of this space, define:
    - Centralized data bases,
    - Client-server environment,
    - Peer-to-peer configuration,
    - Distributed databases, and
    - Parallel databases

Database System Architecture

- Instructor: Ali R. Hurson
  323CS Building
  hurson@mst.edu
  Office Hours by appointment

- Text: *Reading papers and class notes available at*

http://hurson.weebly.com/cs-6302-438-hetereogeneous-and-mobile-databases.html

Database System Architecture

- Grading Policy
  - In class exams & Quizzes: 35%
  - Final Exam (Comprehensive): 35%
  - Project: 20%
  - Home works: 10%
  - Individual grade will be determined based on individual effort, individual effort relative to the class effort, and proactive participation in the class.

Database System Architecture

- For on campus students, hardcopy of homeworks are collected in class,
- It is encouraged to work as a group (at most two people per group) on homeworks/project (grouping is fixed through out the semester),
- May 1st is the deadline for filing grade corrections; no requests for grade change/update will be entertained after this deadline.

Database System Architecture

- Course is composed of several modules, you will be given a test at the end of each module. There are some reading papers associated with each module.

- Modules are self paced. If you are familiar with the contents of a module or if you finish a module ahead of the class, you can contact me to test out that module.

- I am expecting you to look at the slides ahead and prior to the class period.

Database System Architecture

- Course objectives are:
  - Develop an appreciation about issues in various database spaces,
  - Develop an appreciation for solutions proposed for the aforementioned issues in various database spaces,
  - Analyze, compare, and contrast proposed solutions for the very same issue,
  - Develop an aptitude to develop potential solution for database issues.

Database System Architecture

- In a nut shell, the course is looking at the same set of concepts in various database spaces, i.e.,
  - We define a database space based on several basic parameters and study some familiar concepts in this space,
  - We expand our basic space by adding more parameters and study the same concepts in the evolve database spaces.

Database System Architecture

- Introduction
- Database System architectures
- Distributed Database systems
  - Query Processing
  - Transaction processing
  - Recovery and Concurrency control
  - Security

Database System Architecture

- Multidatabases
  - Definition
  - Issues in multidatabase systems
  - Approaches to multidatabase systems
  - Query Processing
  - Transaction Processing
  - Recovery and Concurrency Control
  - Security

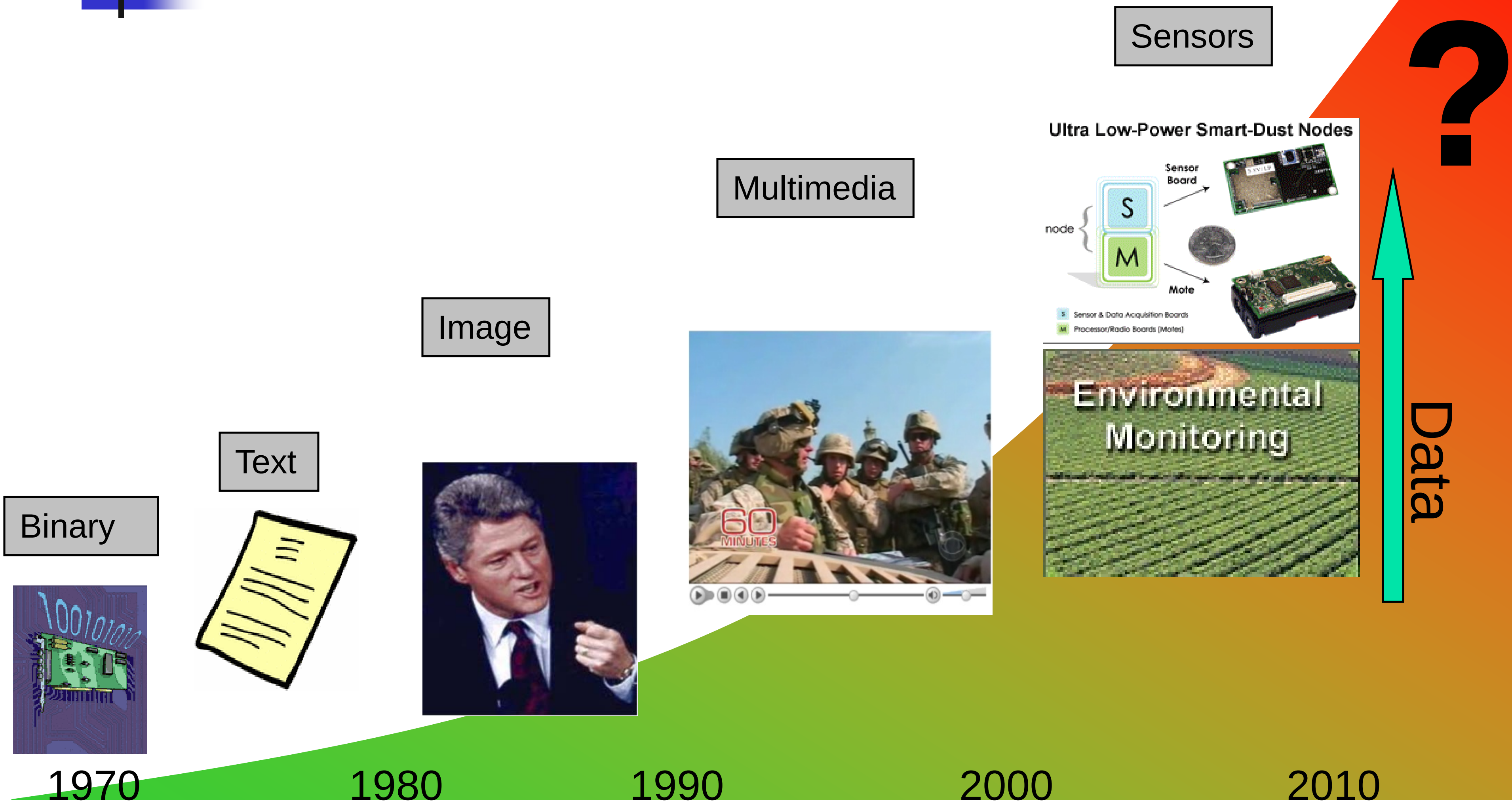
Database System Architecture

- Mobile Data Access systems
  - Mobility issues
  - On-demand services
  - Broadcast services
  - Transaction Processing
  - Security

Database System Architecture

- In this module, we will:
  - Motivate general issues,
  - Define a simple database space,
  - Base on the parameters of this space, define:
    - Centralized data bases,
    - Client-server environment,
    - Peer-to-peer configuration,
    - Distributed databases, and
    - Parallel databases

Database System Architecture



Database System Architecture

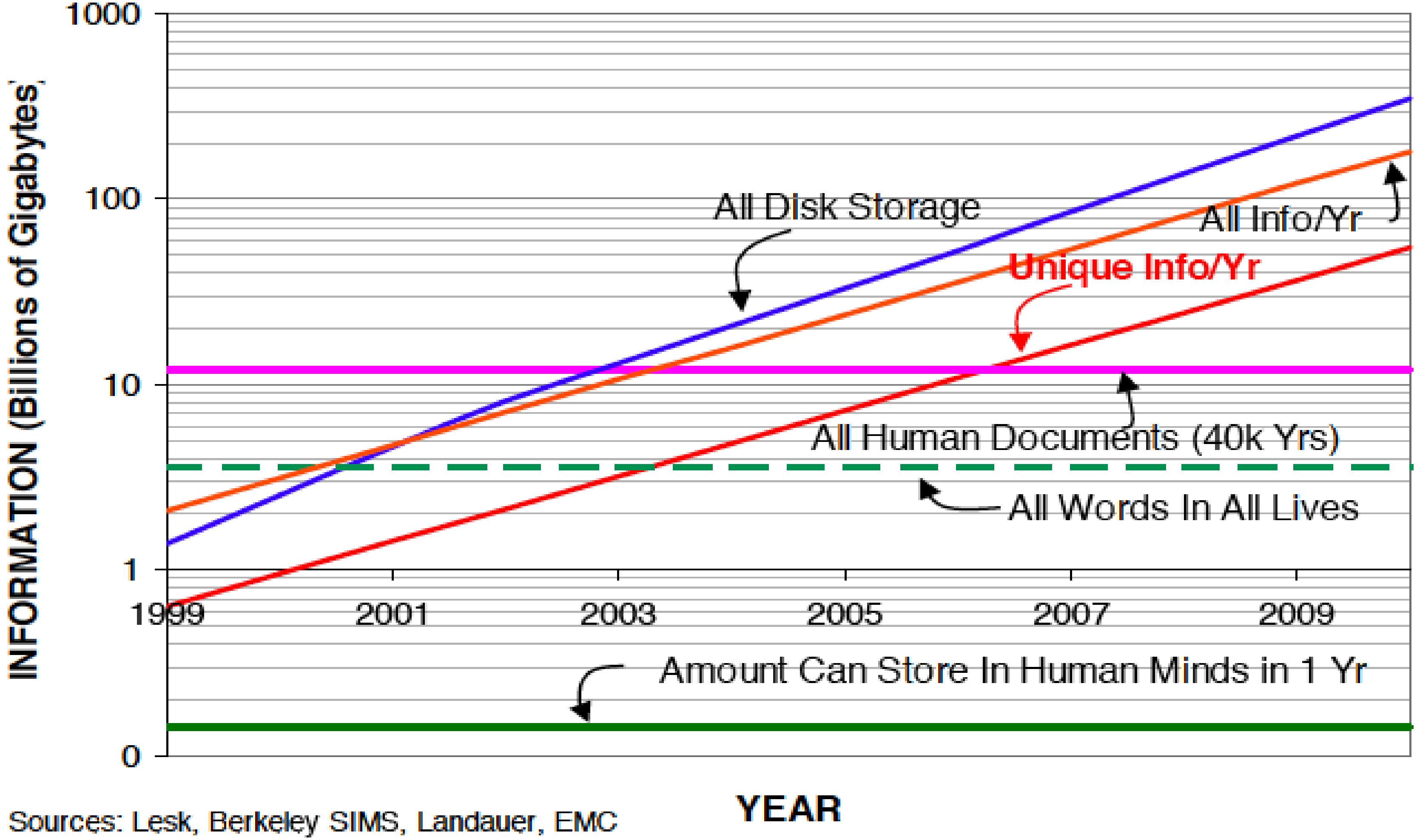
Sample Data sources

In mid 1980s, it was estimated that the U.S. Patent Office and Trademark has a database of size 25 terabytes (1 tera = 10^{12}).

In 1990s, it was estimated that the NASA's Earth Observing Project will generate more than 11,000 terabytes of data.

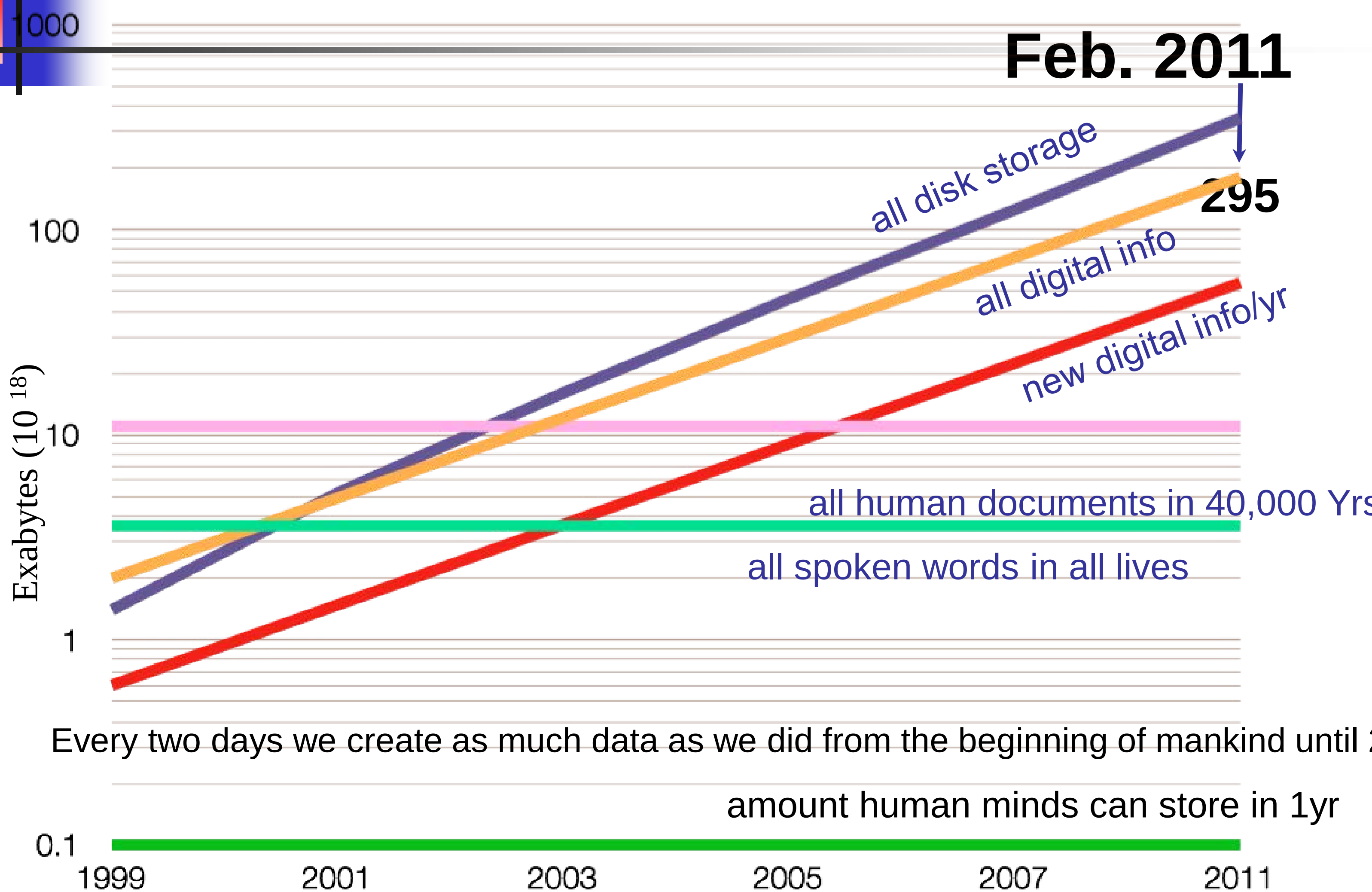
An estimate puts the amount of new information generated in 2002 to 5 exabytes (1 exa = 10^{18}).

Database System Architecture



Source Chris Johnson, University of Utah, IPDPS2012

Database System Architecture



Sources: Lesk, Berkeley SIMS, Landauer, EMC, TechCrunch, Smart Planet

Database System Architecture

How much is an Exabyte?

- 1 Exabyte = 1000 Petabytes = could hold approximately 500,000,000,000,000 pages of standard printed text.
- It takes one tree to produce 94,200 pages of a book.
- It will take 530,785,562,327 trees to store an Exabyte of data.
- In 2005, there were 400,246,300,201 trees on Earth.
- We can store .75 Exabytes of data using all the trees on the entire planet.

- Sources: http://www.whatsabyte.com/ and http://wiki.answers.com

Database System Architecture

Information Everywhere

Heterogeneous Legacy and unreliable (partially) information in some form and shape already exists.

- **Q_1**: How to locate information intelligently, efficiently, and transparently?
- **Q_2**: How to extract, process, and integrate relevant information efficiently?
- **Q_3**: How to interpret information intelligently?
- **Q_4**: How to provide uniform global access methods?
- **Q_5**: How to support user and data source mobility?

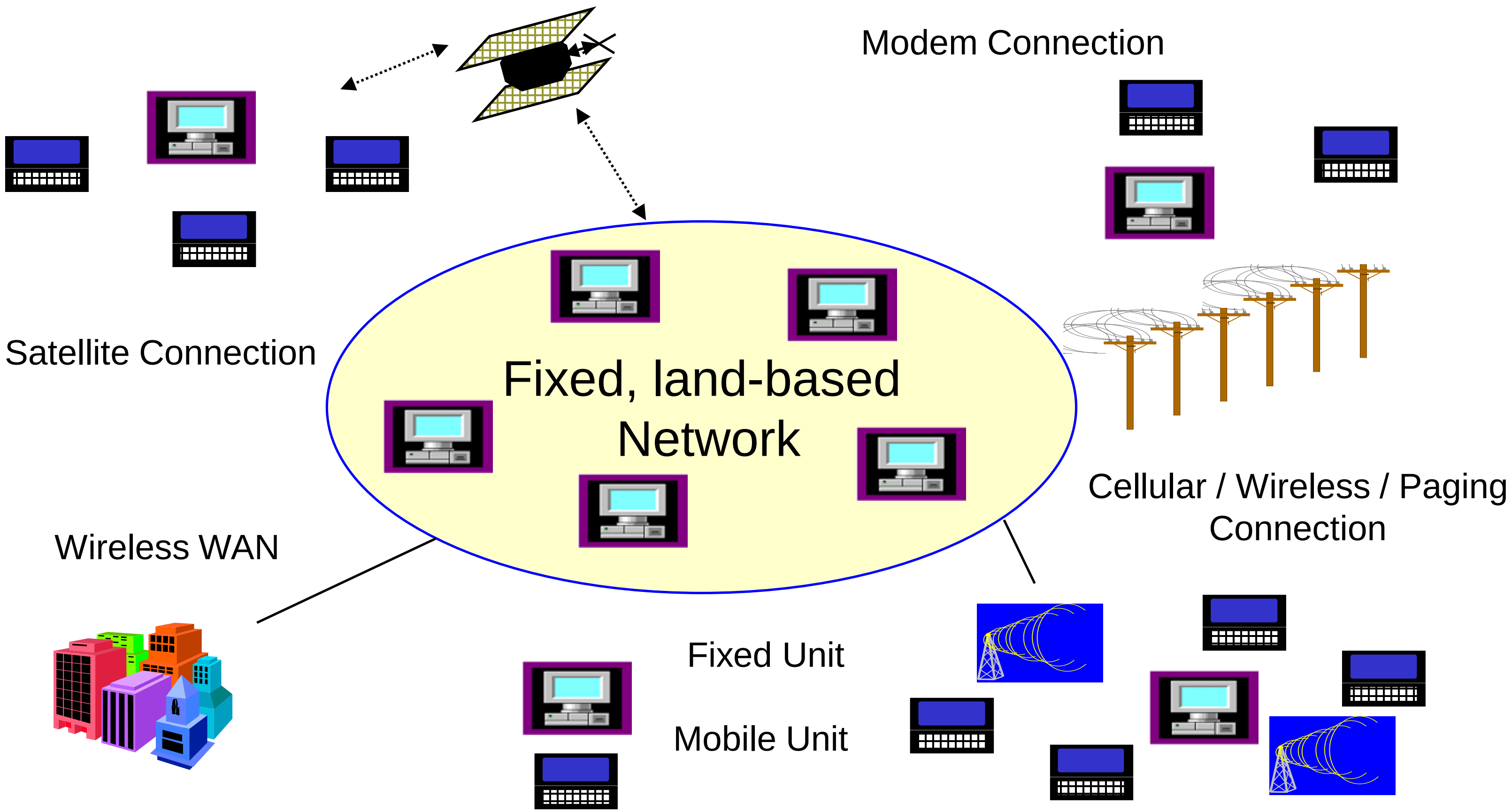
Data Distribution, Data Storage and recording, Data extraction and cleaning,
Data integration, aggregation, and representation
Data interpretation (resolving semantic heterogeneity),
Data incompleteness, data interpretation and mining
Query processing, data modeling, and data analysis
Scalability, Velocity

Anytime, anywhere, transparent, intelligent, secure, timely, reliable, and cost effective access to Global Information regardless of:

- Heterogeneity of access devices,
- Heterogeneity of communication medium,
- Heterogeneity and autonomy of data sources.

Database System Architecture

Underlying Environment



Database System Architecture

In this infrastructure we distinguish:

Three classes of data – Private data, Public data, and Shared data, and

Three classes of services – on-demand based services, broadcast-based services, and pervasive-based services.

Database System Architecture

On-demand service – Based on the user request information is processed and result will be available to the user.

Broadcast based service – Based on some intelligent knowledge, potential information is broadcast and users pull information from the broadcast channels.

Pervasive based Service – Computers work in the background and based on some intelligent knowledge, potential information is pervasively accessed and made available to the users.

Database System Architecture

Autonomy and heterogeneity,
Transparency,
Query Processing and Query Optimization,
Data Integration,
Data Replication, data Duplication, data Caching, and Synchronization,
Transaction Processing and Concurrency Control,
Resource Management - Power Management,
Security.

Database System Architecture

- In this module, we will try to establish some understandings about the various database organizations and to motivate the overall scope of this course.

Database System Architecture

- Different parameters can be used to classify the architecture of data base systems.
- We classify data base systems along the following three parameters:
  - Physical infrastructure
  - Services
  - Distribution

Database System Architecture

- Physical infrastructure
  - This dimension refers to the underlying platform composed of access devices (homogeneous/heterogeneous) interconnected through different communication medium:
    - Processing devices
      - Powerful Machines
      - Portable Devices
    - Network Architecture
      - Land-based Connection
      - Wireless Connection

Database System Architecture

- Services
  - Along this dimension we can distinguish two approaches:
    - There is no distinction between services,
    - Distinction between User processes and Data Processes.
  - For example, in a client-server model, some tasks are executed on the server system and some tasks are executed on client systems.

Database System Architecture

- Distribution
  - Along this dimension we can distinguish :
    - Distribution of data
    - Distribution of control, and
    - Distribution of processes

Database System Architecture

- Physical infrastructure
  - As noted earlier, two elements constitute this dimension;
    - Underlying computing platform, and
    - The communication medium that allows computers to communicate with each other.

Database System Architecture

- Physical Infrastructure: Networking
  - Networking represents the interconnectivity among the elements of the system. It also shows the division of work.

Database System Architecture

- Physical Infrastructure: Parallelism
  - Parallelism within a system allows activities to be sped up — faster response time, higher through put. Note, here we used term “parallelism” as a generic term that refers to the ability of executing more than one tasks at a time, also note that, we did not define the granularity of the task at this point.
  - Requests can be processed in a way that exploits the parallelism offered by the underlying system.

Database System Architecture

- Increased use of parallelism and data/processing distribution are the important trends in database design and implementation. There are several motivations for this:
  - Performance,
  - Distributed access to data,
    - Increased availability,
    - Increased reliability.

Database System Architecture

- Performance metrics
  - Response Time (Execution time, Latency) – The time elapse between the start and the completion of an event.
  - Throughput – The number of tasks that can be completed during a given time interval.

Database System Architecture

- Performance metrics
  - Scaleup – Handling large task by increasing the degree of parallelism. It is the ability to process larger tasks in the same amount of time by providing more resources.
  - Speedup – Running a task in less time by increasing the degree of parallelism.

$$\mathbf{S} = \frac{\text{Execution time on Original Machine}}{\text{Execution time on Larger Machine}}$$

Database System Architecture

- Performance metrics
  - Power Consumption – Becomes an important performance metric when we use mobile wireless access devices.
  - Network Connectivity – Becomes of interest when connectivity is through wireless medium.
  - Data reliability and integrity – Becomes even more of concern at the presence of mobility and wireless communication.

Database System Architecture

- Distribution
  - Along this dimension we can distinguish distribution of:
    - Data,
    - Processing, and
    - Control

    spanning over multiple geographically separated sources.
  - Distribution of control is also referred to as the autonomy.
  - Data resides where it is generated or needed most:
    - Distributed data should be accessible by other sites.
    - Data distribution also implies data duplication/data replication.

Database System Architecture

- Centralized database systems
  - Centralized database systems are those that run on a single computer platform and do not interact with other computer systems.
  - The underlying platform could range from a single-user database system running on personal computer to high-performance database system running on high-end server systems.

Database System Architecture

- Centralized database systems
  - Within the scope of a centralized database systems we distinguish two configurations:
    - Single-user configuration
    - Multi-user configuration
  - Database systems designed for single-user configuration do not provide many facilities needed for a multi-user system — e.g., concurrency control, security, privacy.

Database System Architecture

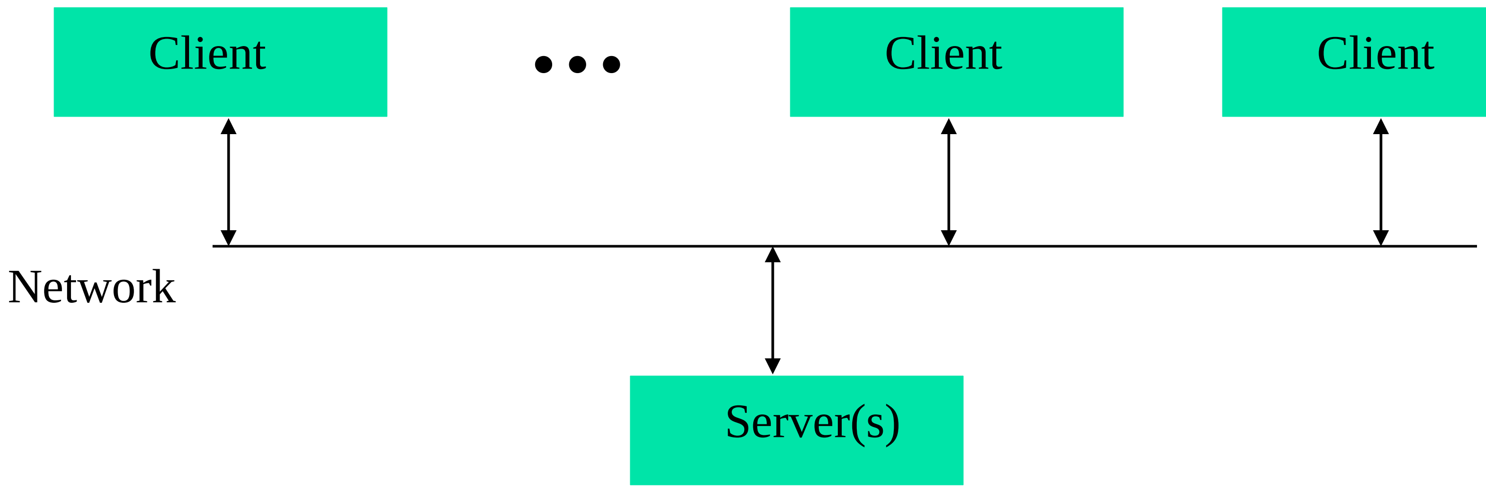
- Client-Server topology
  - Client-Server topology is the direct result of the advances in technology and a step towards distributed topology.
  - It is a two-level topology based on a simple general idea: distinguish the needed functionalities and divide them into two coarse groups:
    - Server functions (The back end functions) — Query processing, Query optimization, concurrency control, recovery.
    - Client functions (The front end functions) — Report writer, Graphical User Interface facilities.

Database System Architecture

- Client-Server topology
  - In comparison to the centralized configuration:
    - The personal computer (Client) assumes the user-interface functionality.
    - Centralized system (Sever) satisfies requests generated by the clients.

Database System Architecture

- Client-Server topology
  - Client-Server topology has functionality split between a server and multiple clients.



Database System Architecture

- The client parses a query and decomposes it into a number of independent site queries. Each site query is sent to the appropriate server site.
- Each server processes the local query and sends the result to the client site.
- The client site combines the results of the sub-queries to generate the final result.

Database System Architecture

- Client-Server topology
  - Popularity of this topology is due to many factors, including:
    - Simplicity of implementation – distinct separation of functionality,
    - Higher degree of hardware utilization at the server side, and
    - Offering a user friendly environment.

Database System Architecture

- Summary (last lecture)
  - Database space
  - Performance metrics
  - Centralized Databases
  - Client-server platform
- Read Database System Architecture — A Walk through Time: From Centralized Platform to Mobile Computing (Keynote Address - ISSADS05)

Database System Architecture

- Client-Server topology
  - It can be further grouped into:
    - Multiple client-single server, and
    - Multiple client-multiple server
      - A client is communicating with a unique “home server”, or
      - A client manages its communication to the appropriate server(s).

Database System Architecture

- Client-Server topology
  - Server system can be categorized as:
    - Transaction Server – thin client, and
    - Data Server – fat client

Database System Architecture

- Choice between thick client *and* thin client
  - The very same application may run at many client sites,
  - Large amount of trust is required between the server and the client,
  - Scalability:
    - number of clients,
    - number of databases.
  - Trend of technology (discussion and justification).

Database System Architecture

- Client-server topology – Transaction servers
  - A typical transaction server consists of multiple processes accessing data in shared memory:
    - Server processes — these are processes that receive user queries, execute them, and send the result back.
    - Lock manager process — this process implements lock manager functionality, lock grant, lock release, and deadlock detection.
    - Database writer process — These are processes that output modified buffer blocks back to disk.

Database System Architecture

- Client-server topology – Transaction servers
  - Log writer process – this process outputs log records from the log record buffer to a stable storage.
  - Checkpoint process – this process performs periodic checkpoints.
  - Process monitor process – this process monitors other processes and if any of them fails, takes recovery actions for the process.

Database System Architecture

- Client-server topology – Transaction servers
  - The shared memory contains all shared data:
    - Buffer pool
    - Lock Table
    - Log buffer
    - Cached query plans
  - Since multiple server processes may access shared memory, mutual exclusion must be ensured on the lock table.
  - If a lock cannot be obtained, the lock request code keeps monitoring the lock table to check when the lock has been granted.

Database System Architecture

- Client-server topology – Data Servers
  - This configuration is used, where:
    - There is a high-speed connection between clients and server (why?) – Local area network,
    - The client machines are powerful, and
    - Task being executed are computation intensive.
  - Note, this configuration requires full back end functionality at the client side.

Database System Architecture

- Client-server topology – Data Servers
  - In this configuration communication cost between clients and server is not much higher than memory references in transaction server configuration. This brings out several interesting issues:

Database System Architecture

- Client-server topology – Data Servers
  - Data granularity – coarse vs. fine granularity
    - For communication
    - For locking
  - Data Caching – Cache coherency
  - Lock Caching

Database System Architecture

- Client-server topology – Data Servers
  - Data Granularity – Size of data communicated between clients and server
    - Communication overhead of message passing justifies coarse data granularity. Specially, in applications with high data locality.
    - Coarse data granularity may have adverse effect on throughput – Lock on a coarse data block may block other clients, unnecessarily.

Database System Architecture

- Client-server topology – Data Servers
  - Data Caching – data that are transferred to a client can be cached for future use. As a result, successive transactions at the same client may be able to make use of the cached data.
  - Data Caching brings out the issue of cache coherency. As a result, it should be guaranteed that all copies of the same data items are synchronized.

Database System Architecture

- Summary
  - Parameters influencing architecture of a database.
  - Performance metrics
  - Centralized databases.
  - Client/server models
    - Thin client
    - Fat client
    - Comparative analysis

Database System Architecture

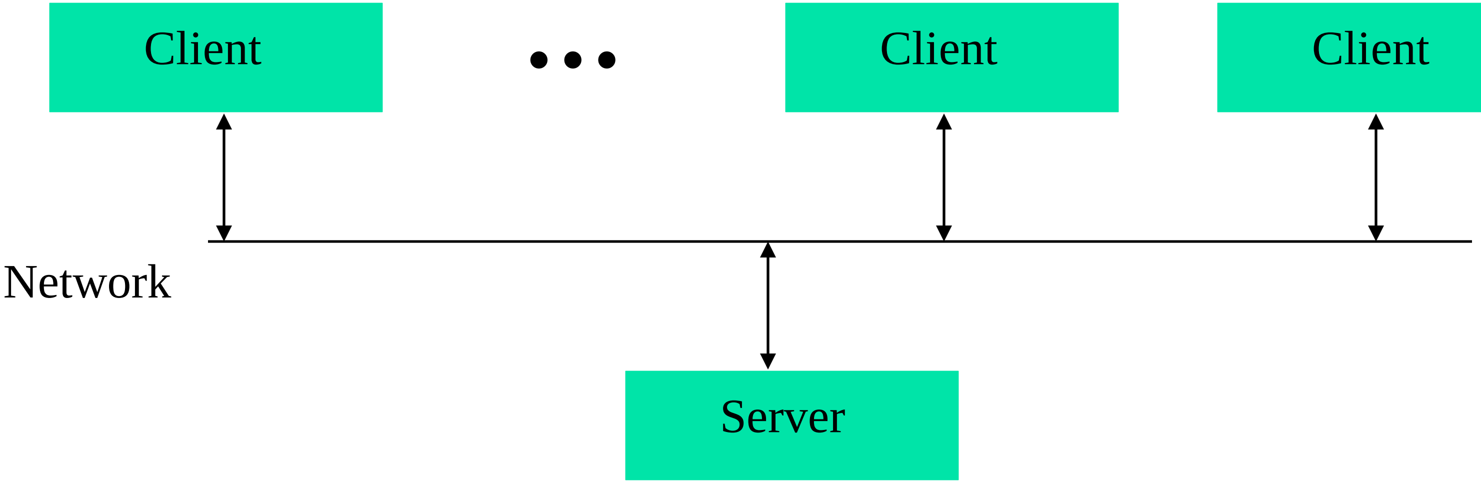
- Single-tier
- Two-tier client-server architecture
- Three-tier client-server architecture

Database System Architecture

- Single-tier
  - Application typically runs on a main frame and users access it through “dumb terminals”.
  - It is easy to maintain by a central administrator, however, it is not scalable.

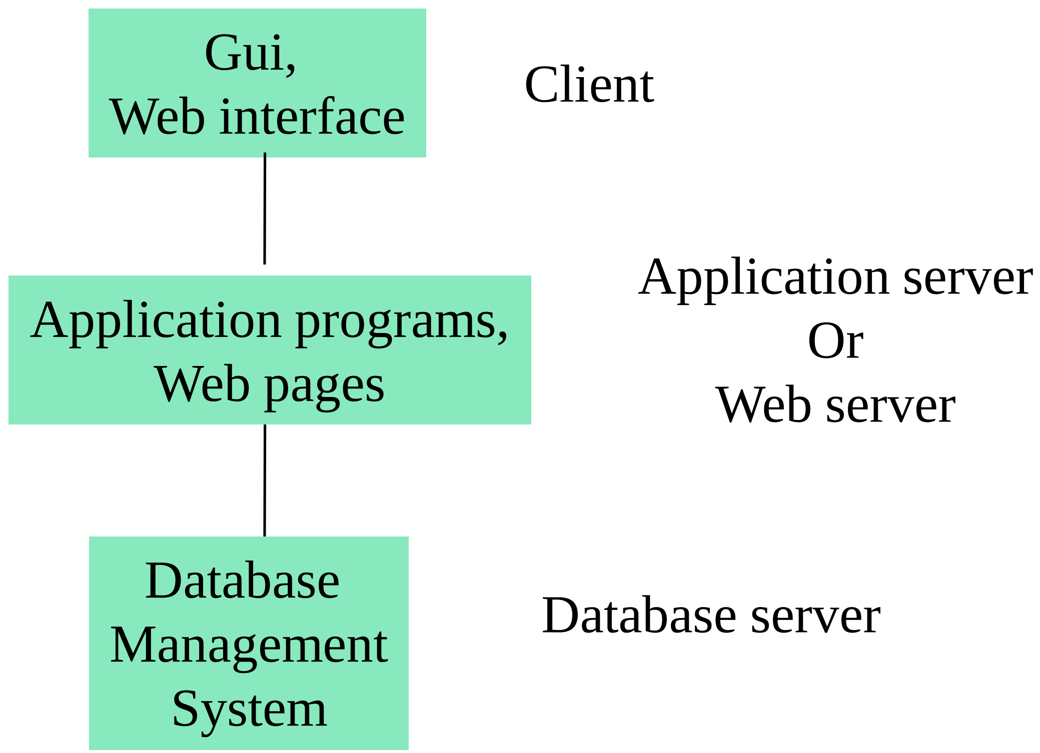
Database System Architecture

- Two-tier client-server architecture
  - At the client site then there is no “dumb terminal”.



Database System Architecture

- Three-tier client-server architecture



Database System Architecture

- Three-tier client-server architecture
  - Client tier (presentation tier) – natural interface with user (thin client),
  - Middle tier – application logic executes here,
  - Database server tier (Data management tier) – data base management system resides.

Database System Architecture

- In an internet shopping scenario:
  - The customer should be able to browse the catalog and make a purchase,
  - Before the sale, the customer has to go through several steps:
    - Add an item into shopping cart,
    - Provide shipping address and credit card number,
    - Confirm the sale.
    - Controlling the flow among these steps and remembering already executed steps is done at the middle tier level.

Database System Architecture

- Advantages of three-tier architecture:
  - Accommodates heterogeneous systems,
  - Supports thin clients (technology),
  - Allows integrated data access,
  - It is scalable with the number of clients.

Database System Architecture

- Peer-to-Peer topology
  - This is the direct evolution of the client-server topology. Note that in a Client-server topology functionality is split into user processes and data processes.
  - User processes handle interaction with the user and data processes handle interaction with data.
  - In a Peer-to-Peer topology, one should expect to find both class of processes placed on every machines.

Database System Architecture

- Peer-to-Peer topology
  - From a data logical perspective, Client-server topology and Peer-to-Peer topology provide the same view of data — data distribution transparency. The distinction lies in the architectural paradigm that is used to realize this level of transparency.

Database System Architecture

- Parallel Systems
  - Parallel configurations are aimed at improving the processing and I/O speeds by using multiple processing units and I/O devices in parallel.
  - Distribute task among several processors at a finer granularity, say relative to client-server paradigm.
  - Within the scope of parallelism, we can talk about:
    - Coarse grain parallelism
    - Fine grain parallelism

Database System Architecture

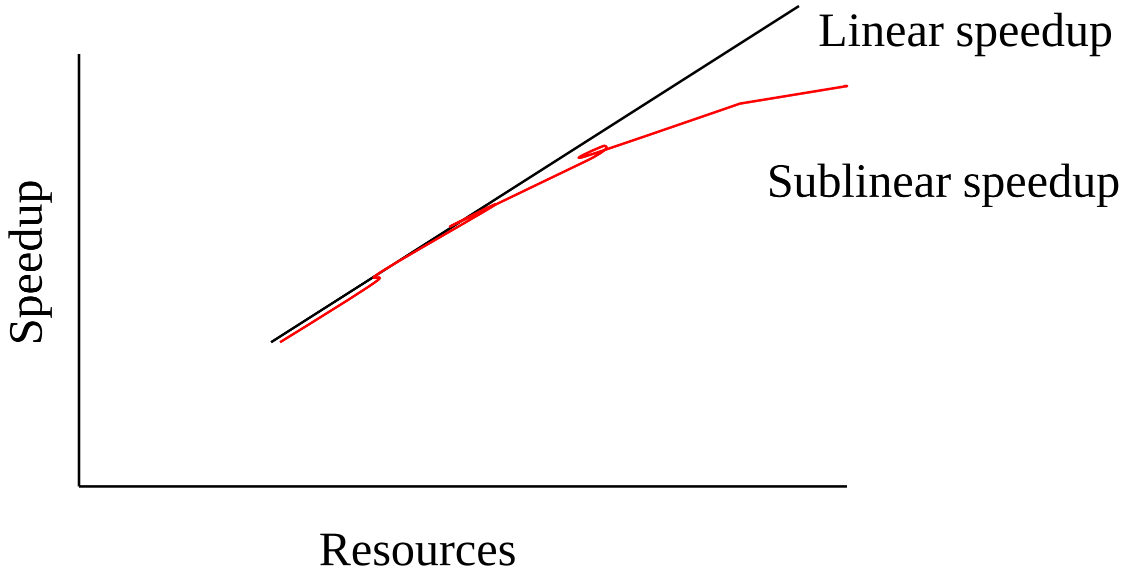
- Parallel Systems
  - Increase throughput by processing many small tasks in parallel,
  - Decrease response time by breaking out a task into subtasks and parallel execution of subtasks.

Database System Architecture

- Questions
  - Define linear speedup.
  - Define sublinear speedup.
  - Define linear scaleup.
  - Define sublinear scaleup.
  - Is there any relationship between speedup and scaleup?
  - For a database application, which one is of more important speedup or scaleup.

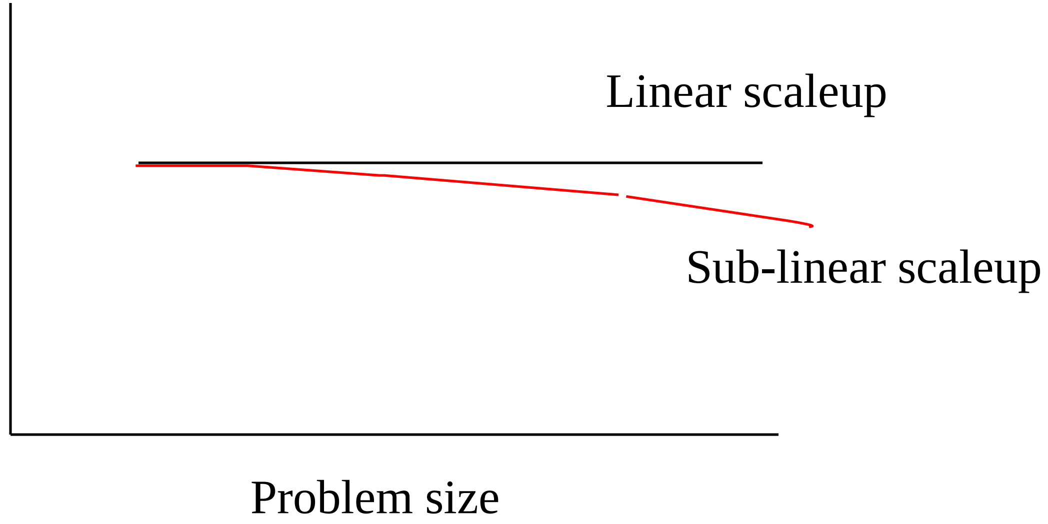
Database System Architecture

- Parallel Systems



Database System Architecture

- Parallel Systems

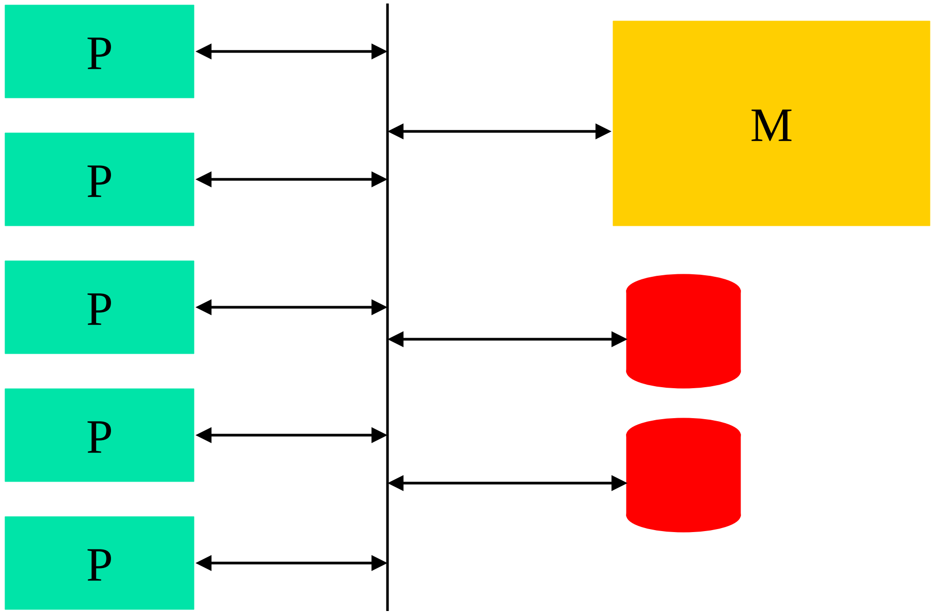


Database System Architecture

- Parallel Systems
  - There are several architectural models for parallel systems:
    - Shared Memory (tightly coupled)
    - Shared Disk (loosely coupled)
    - Shared nothing
    - Hierarchical

Database System Architecture

- Parallel Systems – Shared Memory
  - All processors share a common global memory

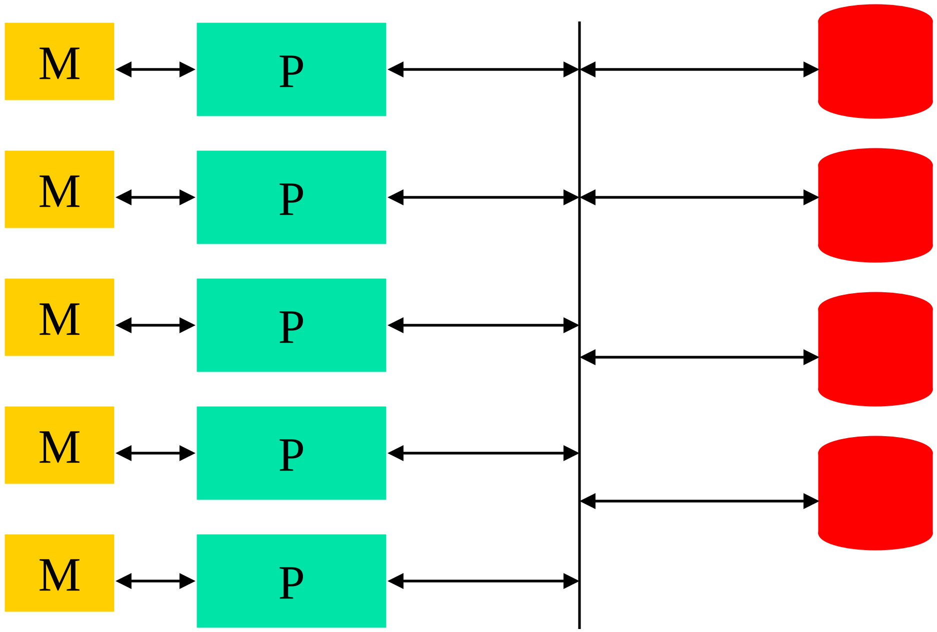


Database System Architecture

- Parallel Systems – Shared Memory
  - Processors and disks have access to a common memory, via a bus or an interconnection network.
  - In general, each processor has a private cache.
  - Efficient communication between processors, via common memory address space.
  - Not scalable, communication network becoming the bottleneck.

Database System Architecture

- Parallel Systems – Shared Disk
  - All processors share a common set of disks



Database System Architecture

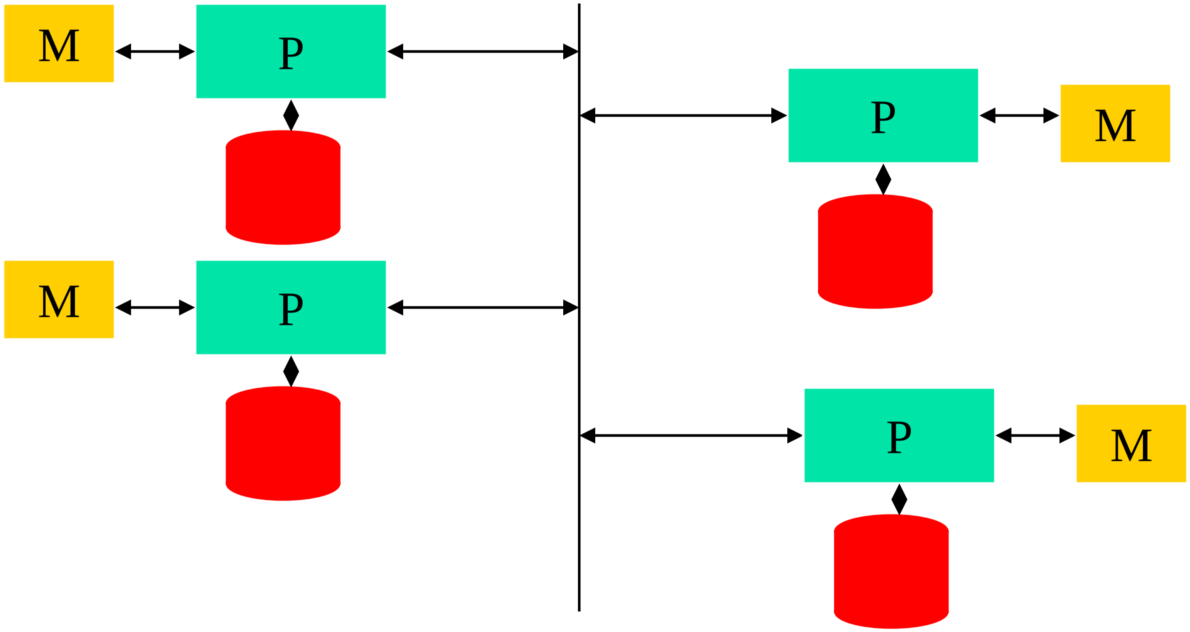
- Parallel Systems – Shared Disk
  - Processors have direct access to all disks via an interconnection network.
  - Each processor has its own private memory.
  - Relative to shared memory organization, this configuration offers:
    - More fault tolerance and
    - Higher memory bandwidth
  - Disk subsystem can become more fault tolerance and faster by application of RAID architecture.

Database System Architecture

- Parallel Systems – Shared Disk
  - System is more scalable than the shared memory configuration.
  - Degree of scalability is limited due to the bottleneck at the interconnection network between disk subsystem and processor.
  - Communication among processors is slow – message passing.

Database System Architecture

- Parallel Systems – Shared Nothing
  - The processors share neither a common memory nor common disks

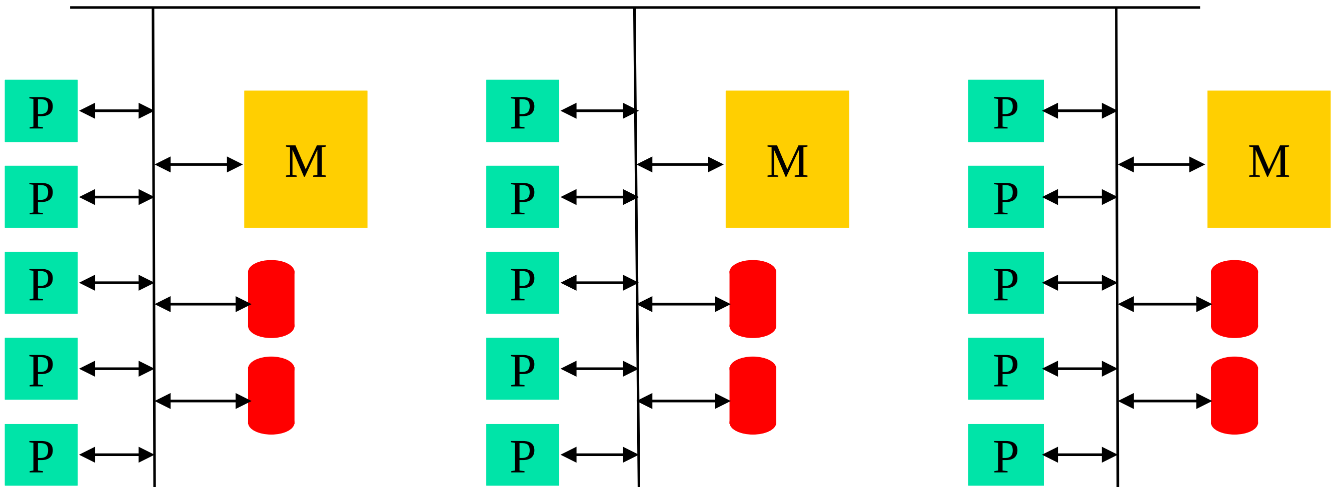


Database System Architecture

- Parallel Systems – Shared Nothing
  - This configuration is scalable.
  - Communication among processors and non-local disk accesses are expensive.

Database System Architecture

- Parallel Systems – Hierarchical
  - A hybrid of the other models



Database System Architecture

- Parallel Systems – Hierarchical
  - At higher level system acts as a shared nothing organization.
  - Each node of the system at the lower level can be a shared memory and/or shared disk system.

Database System Architecture

- Parallel Database Management Systems
  - Database management systems developed on parallel systems are called parallel database management system.

Database System Architecture

- Parallel database system seeks to improve performance through parallelization of operations. In another words, parallel database system is motivated by performance improvement.

Database System Architecture

- Summary (last lecture)
  - Different classes of client-server topology
  - Peer-to-Peer topology
  - Parallel systems
  - Different classes of parallel systems
  - Homework #1
  - New reading paper

Database System Architecture

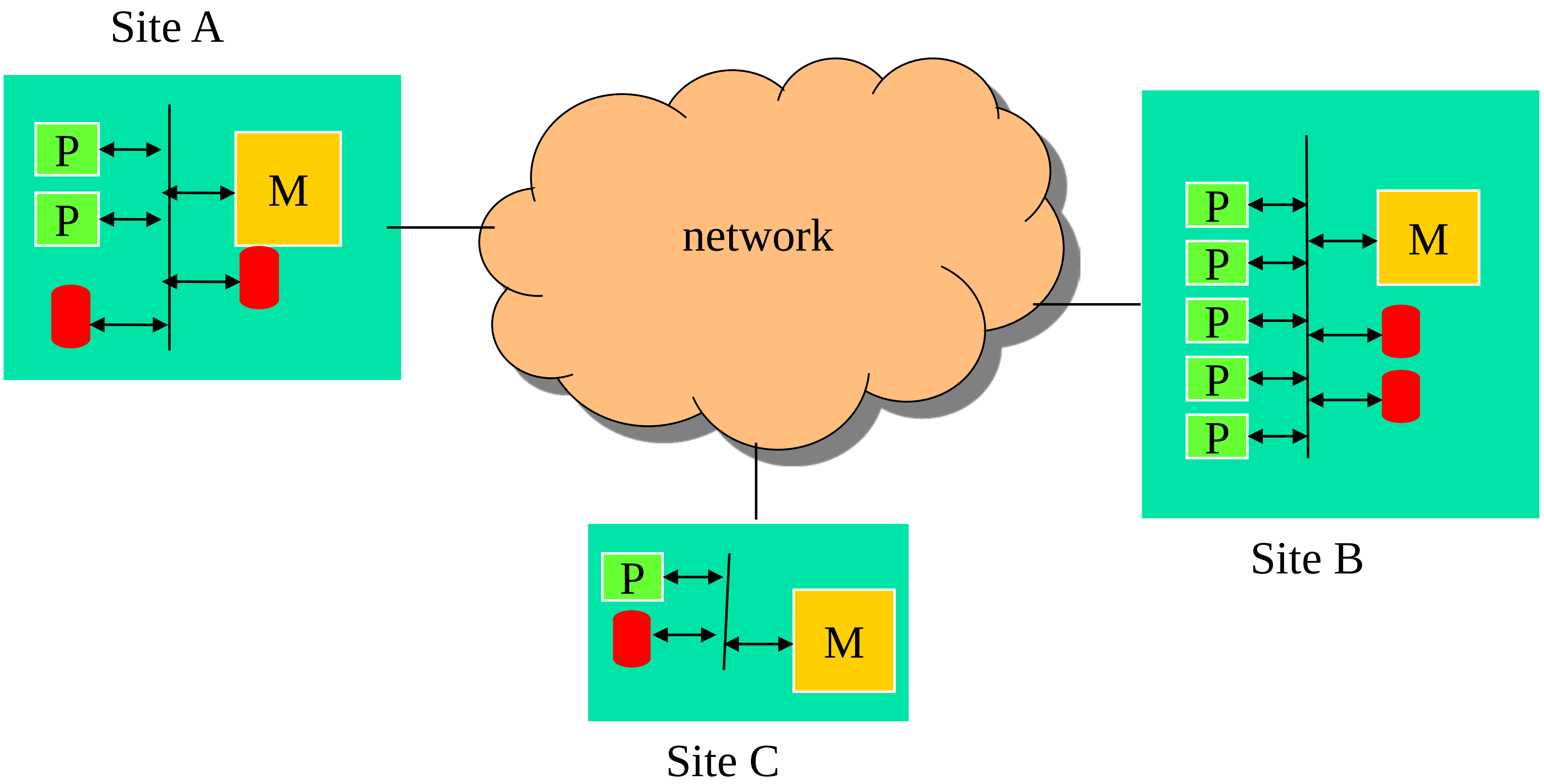
- Distributed Systems
  - Distributed databases bring the advantages of distributed computing to the database management domain.
  - A distributed system is a collection of processors, not necessarily homogeneous, interconnected by a computer network.

Database System Architecture

- Distributed Database Systems
  - Database is stored on several computers and computers communicate with each other through various communication media.
  - Computers do not share resources ─ disks, memory, processor, …

Database System Architecture

- Distributed Systems



Database System Architecture

- ## Distributed Database Systems
  - A distributed database is a collection of multiple logically interrelated databases distributed over a computer network — related data sources:
    - Are closer to the application domain (s) that uses it.
    - Might be replicated — to improve performance.
    - Are split (Fragmented), horizontally and/or vertically, and distributed — to balance the load and improve performance.
  - A distributed database management system is a software system that manages a distributed database while making the distribution transparent to the user.

Database System Architecture

- Questions
  - What is the RAID?
  - What are the major differences between shared nothing and distributed configurations.
    - Distributed systems are typically geographically separated, are separately administrated, have slower interconnection, and there is a distinction between local processes and global processes.
    - With respect to the databases, in distributed databases, data is distributed while in shared nothing configuration, data is not distributed.

Database System Architecture

- Distributed Database Systems
  - In comparison to parallel systems in which processors are tightly coupled and constitute a single data base system, a distributed data base system is a collection of loosely coupled systems that share no physical components.
  - In general distributed databases can be classified as:
    - Homogeneous databases
    - Heterogeneous databases

Database System Architecture

- Distributed Database Systems
  - There are several reasons for building distributed database systems:
    - Sharing data
    - Autonomy
    - Increased reliability and availability
    - Improved performance
    - Ease of expansion

Database System Architecture

- Distributed Database Systems
  - Data distribution is an effort to improve performance:
    - To reduce communication costs and hence to reduce response time,
    - To maintain more control and enforce better security,
    - To improve quality of service in case of network failure.

Database System Architecture

- In a distributed database system, data is physically stored across several sites and each site is typically managed by a database management system capable of running independent of the other sites.
- Data distribution is motivated by:
  - Increased availability, and
  - Distributed access to data – locality in access patterns

Database System Architecture

- Distributed Database System :
  - Increased reliability and availability – reliability means probability that a system is running at a certain time point, availability means probability that a system is continuously available during a time interval.
  - Improved performance, and
  - Ease of expansion.