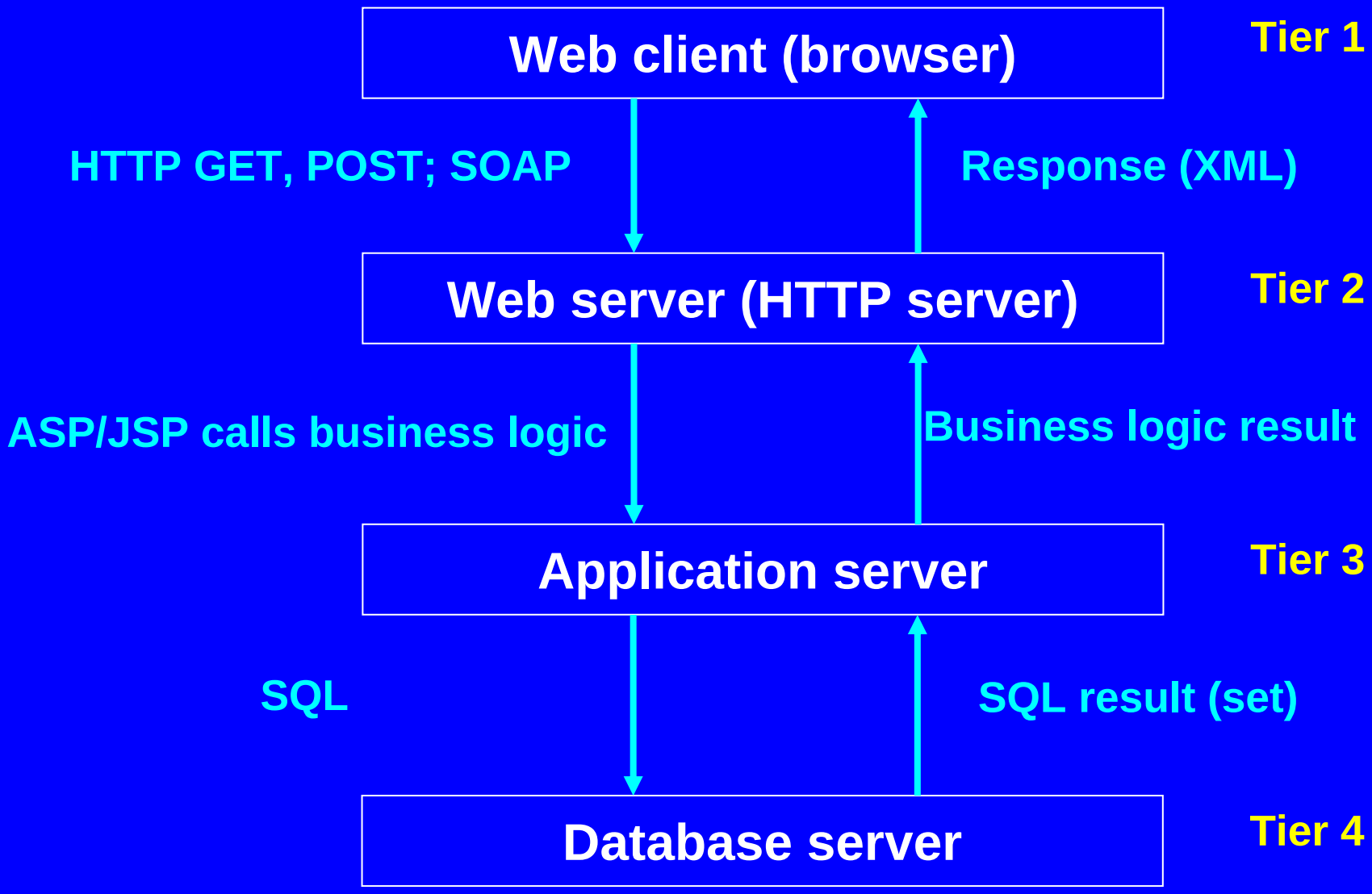
1.264 Lecture 18

System architecture

Architecture example

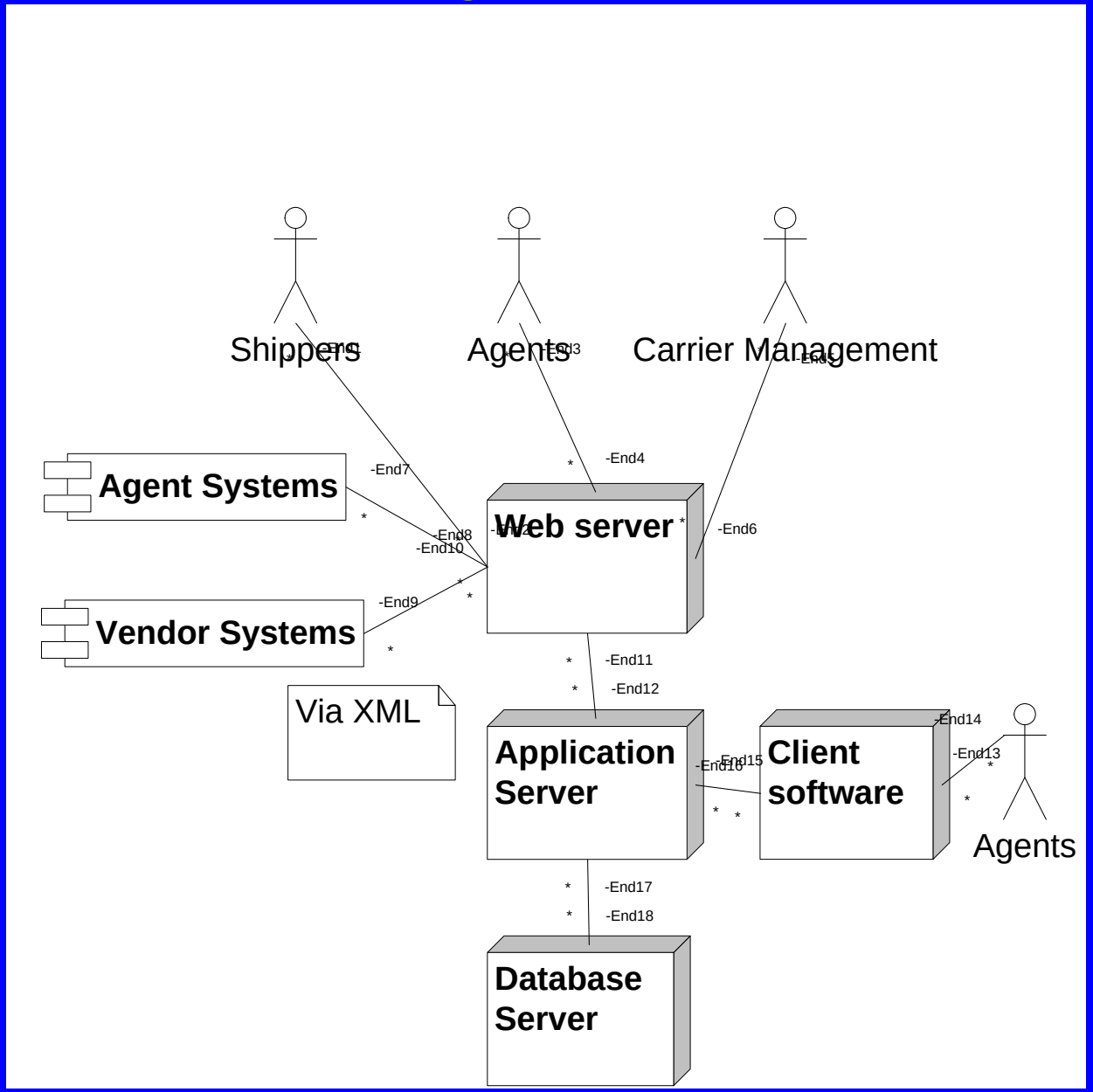


In HW7, your ASP pages include the business logic that is usually in the application server

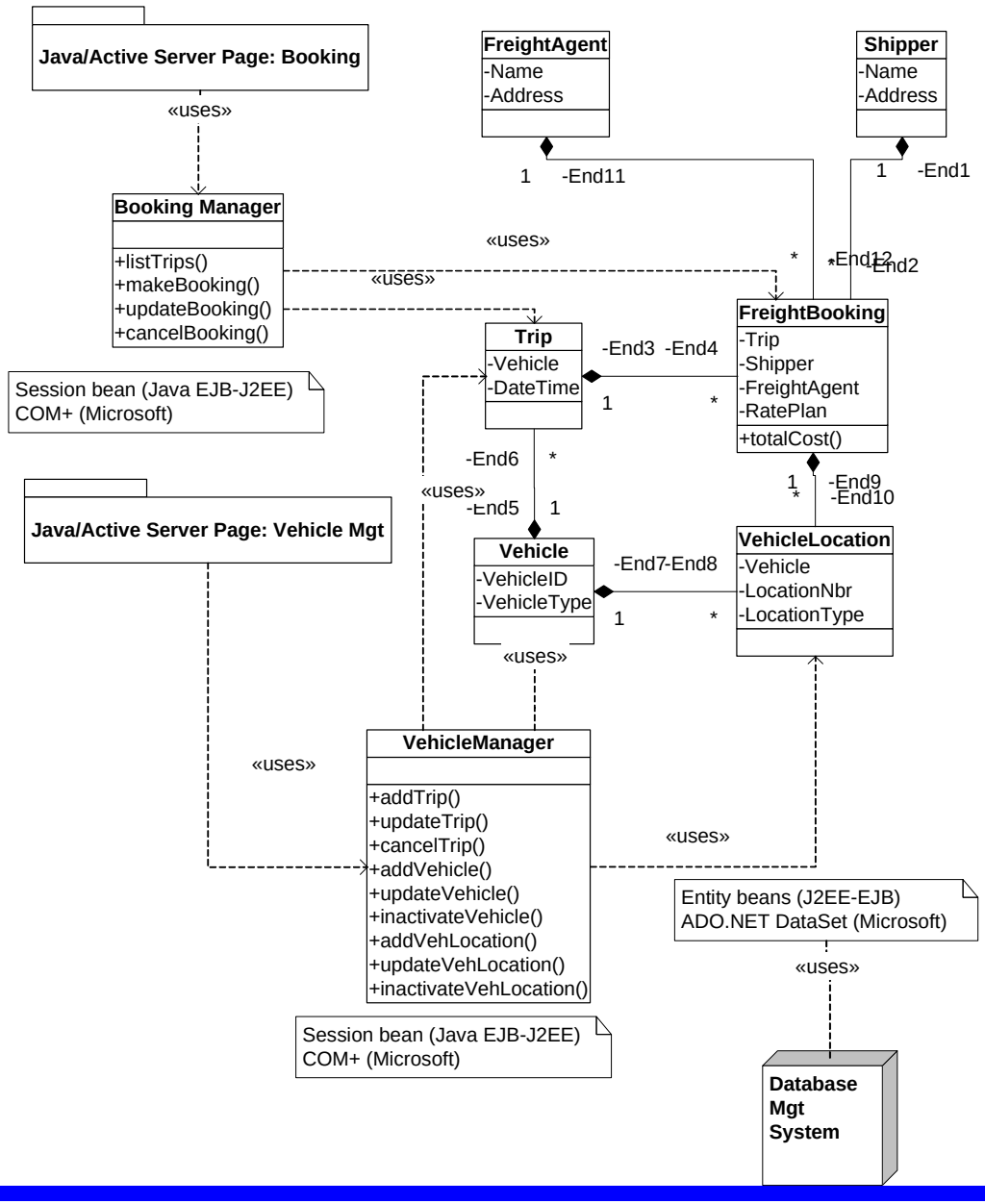
Architecture example

- Example application reserves space on intermodal carrier trips (vessel, rail, …). Used by people:
  - Agents to sell freight transportation
  - Shippers to view trip schedules and details
  - Carrier management to manage vehicle and trip data
- Application is accessed by three mechanisms:
  - Web interface (shippers, agents, carrier management)
  - Standalone application (agents)
  - Other systems:
    - External agent systems
    - Vehicle maintenance companies that need physical data (vehicle type, fuel capacity, fuel type, …)

UML Diagram: Use Case



UML Diagram: Class Diagram



Web server

App server (session or business logic beans)

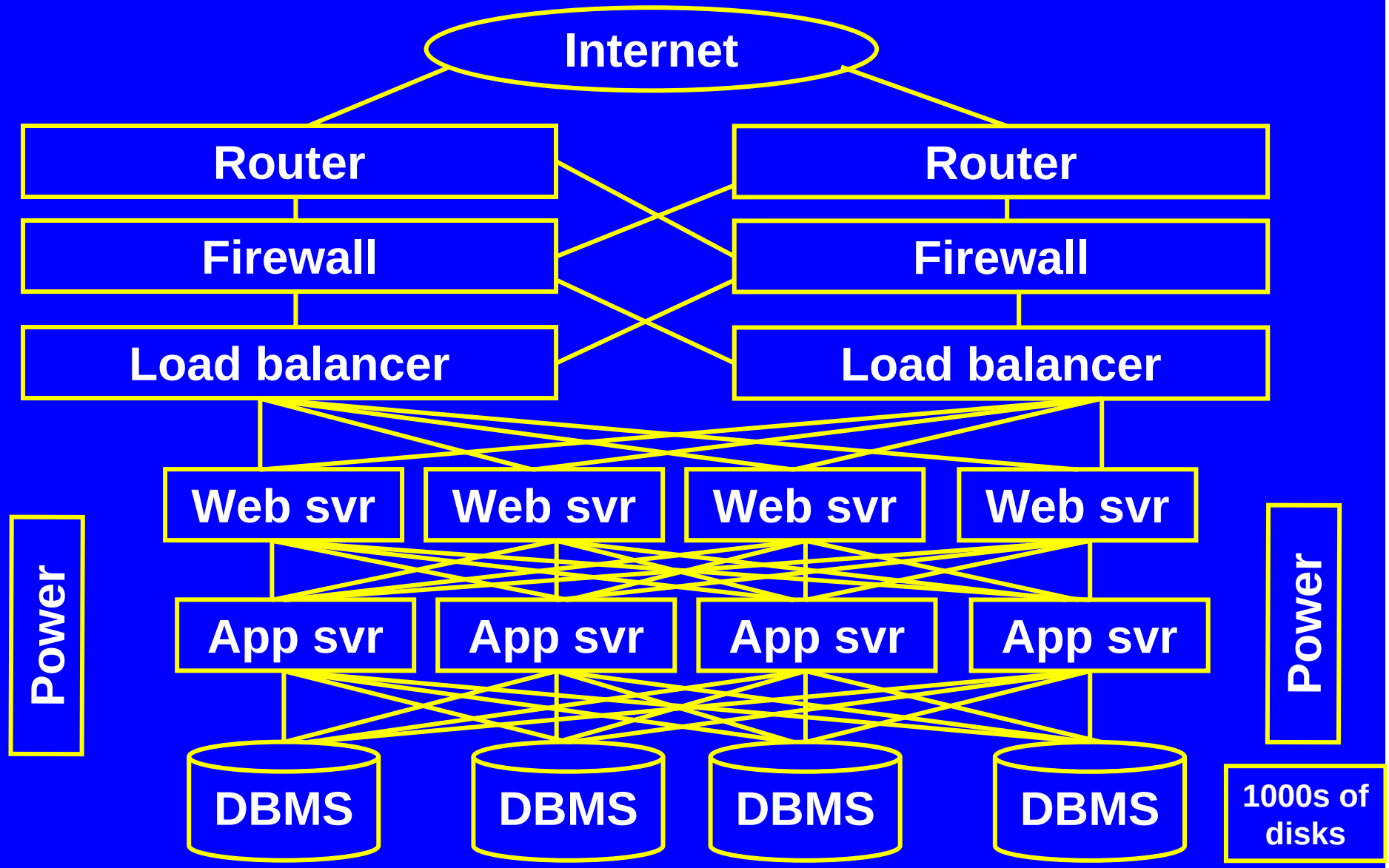
App server (entity or db beans)

Database server

Server-side architecture components

- Network interface architecture components
  - Router
  - Firewall
  - Load balancer
- Application components
  - Web server
  - Application server
  - Database server

Architecture components at ‘server’



Router, Firewall, Load Balancer

- Router routes IP (TCP/IP and UDP/IP) packets
  - Routing table: either send packet to local machine or to another router
- Firewall examines all packets
  - Checks IP address, port
  - Can extract and store contents from all Internet protocols if suspicious packets are seen, especially HTTP/SOAP these days
  - Has security language to set traffic rules
- Load balancer sends HTTP packets to Web servers
  - Balances based on HTTP header: client IP, cookie, …
  - Many inexpensive servers function as one large single virtual server
  - Use with multiple heterogeneous hardware platforms
  - Add new servers, and sites without interruption to service
  - Google runs thousands of Linux servers to index Web

Web server, App server, Database server

- **Web server:**
  - Handles HTTP requests and responses
  - Reads and writes XML and HTML
- **Application server:**
  - Runs programs that implement business logic. These are processes, while the database stores entities.
  - Processes tend to be simple, modular.
  - Designed from UML use cases, sequence diagrams… E.g.,
    - Order entry; receiving, shipping, payment, engineering quotes…
- **Database server:**
  - Persistent storage of data
  - Sharing of data across applications and organizations
  - Repository of business rules, within database and to guide XML document exchange

System architecture and configuration

- Done early in system development or configuration project
  - Assess users, applications, system software, networks, hardware
  - Configurations are complex and changing
  - Information is almost always wrong on which estimates are based
- Successful configurations are usually overbuilt
- Organization of all system components (hardware, software, network) is the system architecture
  - Wong text, though old, is still the best practical treatment
- Don’t forget about development, staging, QA servers
  - Spend your entire budget, always!
- We will focus on configuring a database server for online transactions
  - Web servers, application servers are generally similar
  - Each computer system is configured to do just one thing:
    - HTTP (Web server), session logic (app server), SQL (database svr)
    - We can understand and characterize its task this way. If a box handles many functions, sizing and managing it is impossible

Virtual Storage Hierarchy

Level	Speed	Size
Registers	1 ns	1KB per CPU
Cache (L1, L2)	1-3ns	64KB-2MB per CPU
Main memory	5-70ns	30GB per system
Disk storage	10,000,000ns (10-15 millisec)	10TB per system
Network virtual memory	100,000,000ns (100 millisec)	10TB per network (LAN)

Each level caches for the one above it, with 95%+ hit rate

Even 1-2% misses degrade performance: lost cycles

Levels go from fast, expensive, small -> slow, cheap, large

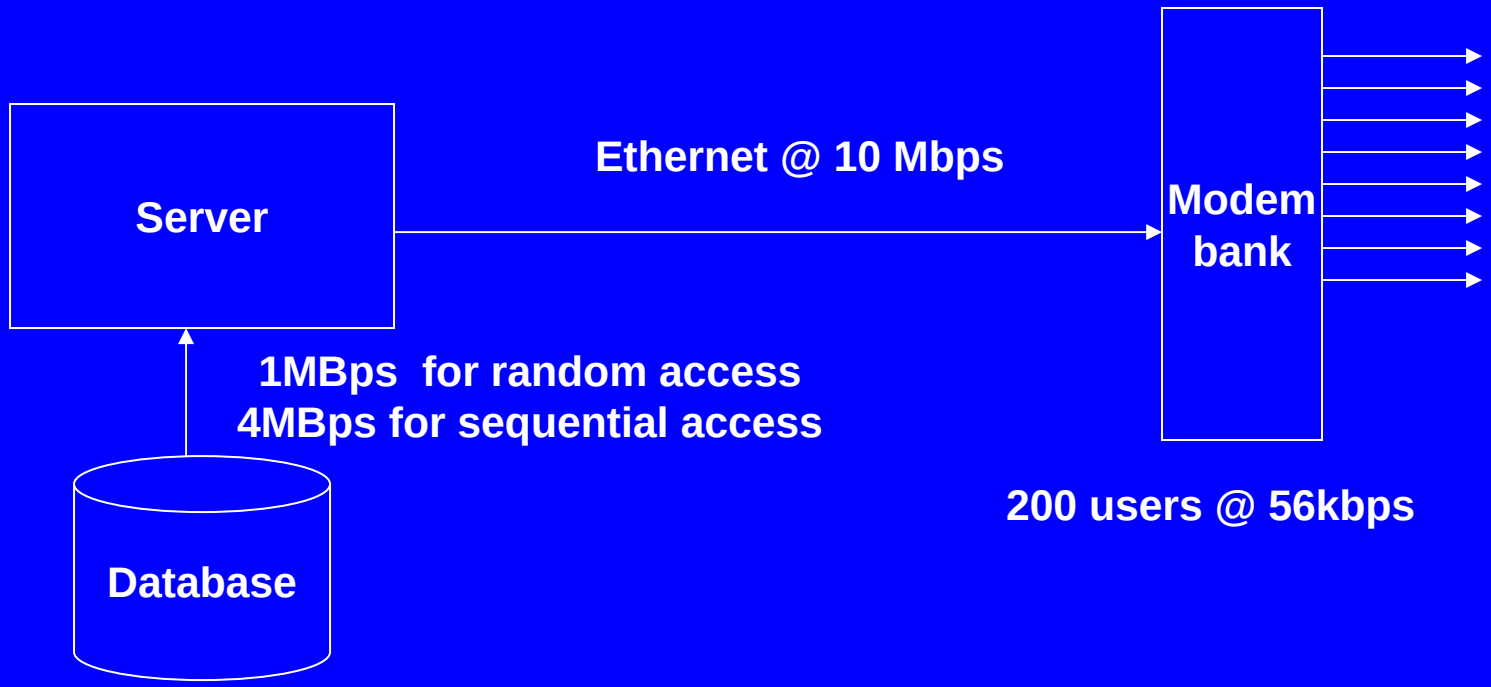
Virtual Storage Hierarchy

- Servers are not just big PCs; they are organized differently
- Memory is used to stage data to the processor in servers
  - Memory use for program execution is secondary
- Cache is used for program execution. Many programs have 1MB footprint, so 2MB cache allows OS and database to run, for example.
- Disk is used only for persistence
  - Disks are 1,000,000+ times slower than memory and are the bottleneck
  - Servers use many small disks (rather than single large disks as on a PC) for performance: RAID (next lecture)
- Why does all this matter?
  - Applications grow big quickly if successful. You need to know basics.
  - Many companies have had big crashes (Ebay, others)
  - Someone in a company needs an overall perspective and knowledge: it might as well be you.

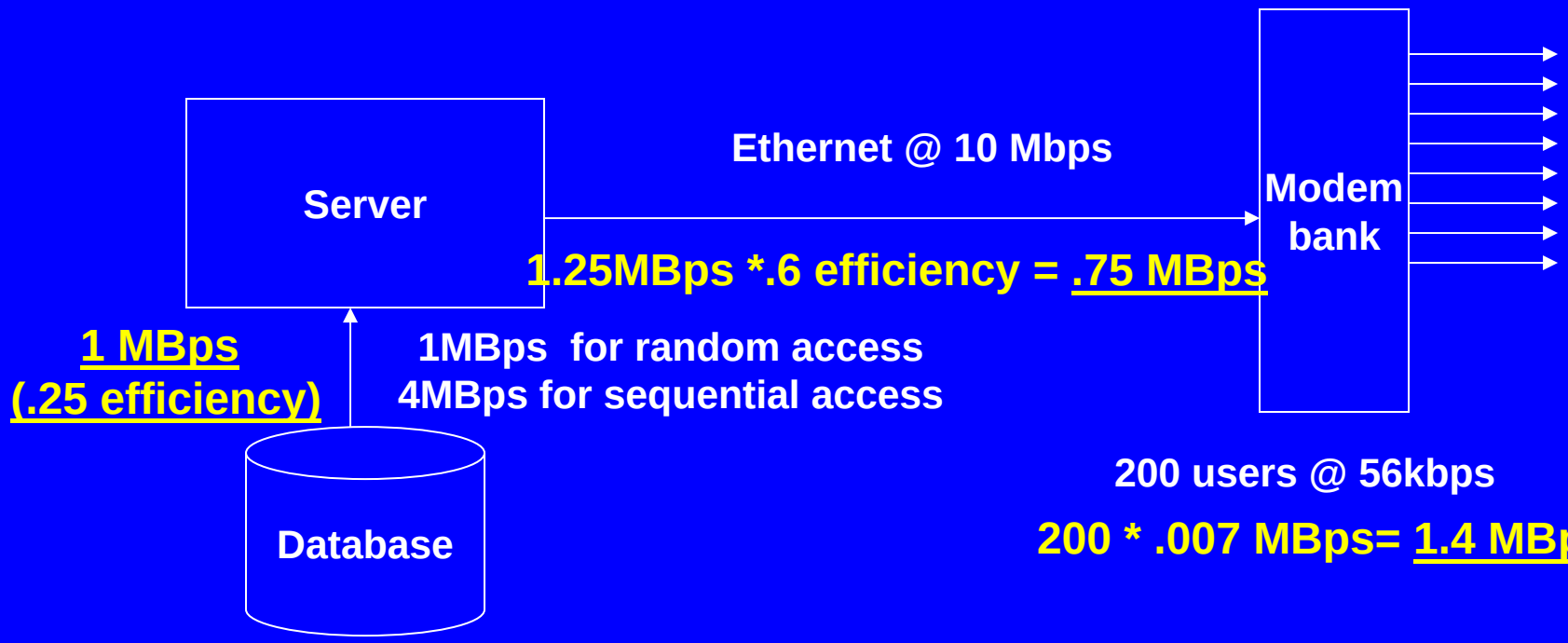
Performance metrics

- Metrics (database server as an example)
  - Throughput: I/O operations/second, data transfer rate/second
    - This is system view
    - If throughput is low, must buy more servers than really needed
  - Latency: seek time (seconds), response time (seconds)
    - This is user view
    - If latency is high, users leave your site (8 seconds max)
  - Utilization: percentage of data transfer rate, disk capacity used
    - This is future view
    - If avg utilization > 60%, your systems will crawl on busy days
  - Efficiency: can system actually meet its theoretical limits?
    - Is the software written well enough?
    - Will the 20 user database really support 20 users on your box?
- Units
  - Ethernet data transfer: 10 Mbit/sec = 1.250 MB/sec
  - Disk data transfer: 10.000 MB/sec
  - Modem data transfer: 56,000 bits/sec= 0.007 MB/sec
- Use comparable units when configuring systems!

Architecture:
Where is the bottleneck?



Architecture:
Where is the bottleneck?



DBMS server configuration

- Three major categories of use:
  - Online transaction processing (OLTP)
    - Standardized queries, used by many users (like your Web app)
    - Queries are keyed (indexed), so little disk access is needed
    - OLTP apps spend as much time formatting and presenting as retrieving data
    - Most queries return a small amount of data
    - Database administrator can optimize database for well-known queries
  - Batch processing
    - Long sequential table scans, and sometimes many updates
    - Little or no user interface, little time on formatting
    - Most resources spent finding and manipulating data
    - Exactly the opposite of OLTP

DBMS server configuration, cont

- Decision support systems (DSS) or online analytic processing (OLAP)
  - Ad hoc queries by power users
  - Typical DSS query is 1,000,000 times more resource intensive as OLTP query
  - Most queries have aggregation, sorts, multiway joins, …
  - DSS users make unexpected queries and are not DBMS experts…
  - Typical DSS application is 1,000 times as resource intensive as the application that produces its data
  - DSS offloaded to data warehouse in many cases

Application	DBMS	Application	GUI	Disk Access
OLTP	Light	Light	Medium	Random
Batch	Medium	Heavy	None	Sequential
DSS	Heavy	Medium	Light	Sequential

Database server configuration steps

- So, how do we configure a database (or Web or app) server?
- Key elements to configure:
  - CPU
  - Cache (configure as much as available, always)
  - Memory
  - Disk
  - Network
  - Backup
- Review (learn) principles of configuration
- Then use configuration software (demo later in lecture)

DBMS server configuration: benchmarks

- Transaction Processing Council (www.tpc.org) is a clearinghouse
- Four benchmarks:
  - TPC-C: Orders to a warehouse with inventory, replenishment
    - Your chemical manufacturer matches TPC-C well
  - TPC-H: Ad-hoc, decision support (DSS, OLAP)
    - No pre-knowledge of the queries can be built into the DBMS system and the query execution times can be very long.
  - TPC-R: Standard reports and queries
    - Pre-knowledge of queries built into DBMS, which can be optimized to run the standard queries efficiently.
  - TPC-W: Simulated Web ecommerce environment.
    - Exercises broad set of system components.
- Select your transaction type; estimate the transaction rate
- Select a server from benchmarks that can support it (see Web). Be very conservative; the benchmarks are tuned!
- If you find an exact match in the benchmarks, great!
  - If not, read on…

TPC-C

- Simulates retail warehouse activity. Warehouse has:
  - Stock (30MB)
  - Item descriptions (8MB)
  - Customers (20MB)
  - Districts (1KB)
  - Orders (23MB)
  - Total database is 180MB, after adding indexes
- Five transactions
  - New order, order status, payment, delivery, stock level
- Users have a ‘think time’ between transactions
  - Each user generates an order once every 18 minutes, etc.
  - Multiply TPC transactions per minute by 2/3 to get rough estimate of users supported
- Note that only 43% (new order) of TPC-C transactions are reported in the benchmark

TPC-C transaction characteristics

Transaction	Reads	Writes	% of Trans
New order Check all items valid Update inventory level	23	23	43%
Payment Pay invoice for order	4	4	44%
Order status Retrieve by customer ID Retrieve by name, dups	14	0	4%
Delivery Update order status to show delivery	130	130	4%
Stock level Query items in recent orders to restock	410	0	4%

Browse www.tpc.org to see benchmark results for C, H, R and W

(There are 46 order-related I/O operations per TPC-C order transaction and about 120 total I/O ops per TPC-C order)

Database servers: memory

- Memory used as cache to avoid disk input/output (I/O)
  - Underconfiguring memory has severe performance impacts (factor of 100,000+)
- Memory size for OLTP:
  - User rule: 500KB per user
  - 5 minute rule: Cache all data accessed every 5 min or less
  - 1-5% rule: Cache at least 1% of DBMS raw data size, perhaps 5%
- Example: 5GB database (1 million customers, 5000 bytes) with 200 simultaneous Web users. Memory sizes:
  - Minimum: 200*500KB= 100MB
  - 5 minute rule: Needs analysis, typically 100MB-1GB for 5GB database
  - 1-5% rule: 50-250MB
- Additional memory:
  - 128MB for operating system (Win2K or Linux/UNIX)
  - 64MB for DBMS executable
  - Memory for any other applications running on server (monitoring, security, etc.)

Database servers: processors

- Maximum of 300 simultaneous users per Pentium processor in database server, even in very lightweight application
- For Oracle Financials, 50 users per Pentium processor max
  - Typical of logistics, manufacturing applications
- For CAD/engineering applications, 1 user per processor!

Database servers: disks

- Margin of error is about 1000:1, depending on query set!
  - Standard sizing assumes all queries are indexed/keyed
  - If they are not, sequential scans are much more resource intensive
- Disk access capacity is key issue
  - Disks have the same performance, regardless of size
    - Data transfer rates, I/O operations/second are about the same
  - A single large disk will have much lower data transfer rate than a set of small disks. An example for a 180 GB database:

Disk	# Disks	Ops/disk	Cost	$/GB	$/Op	IO Ops	MB/sec
4.5GB	40	140	$70	$16	$0.50	5,600	400
9GB	20	140	$90	$10	$0.64	2,800	200
18GB	10	140	$150	$8	$1.07	1,400	100

Disk geometry

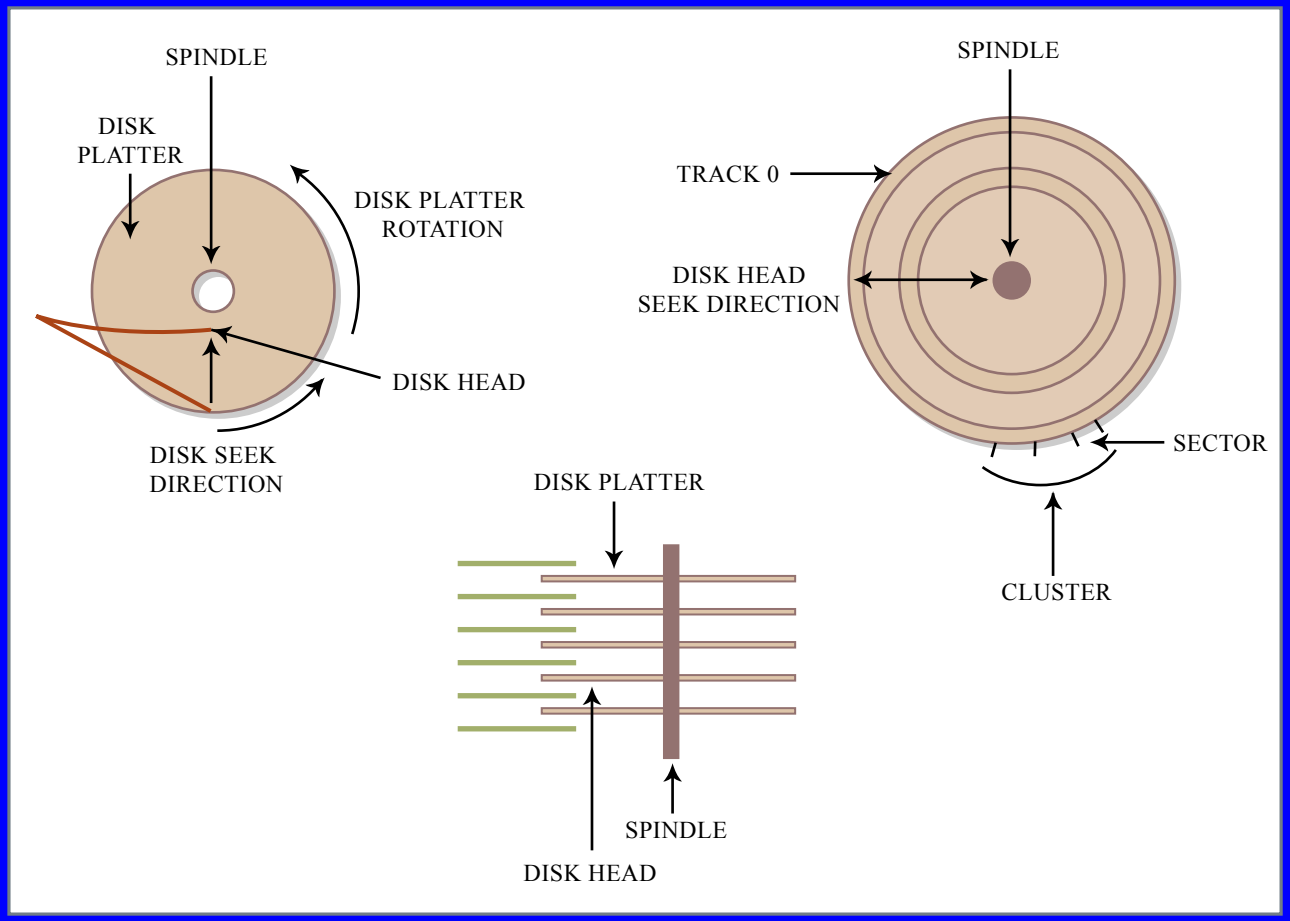


Figure by MIT OCW.

Database server: disk configuration guidelines

- Use smallest feasible disk size, as many disks as possible
  - Use RAID rather than manual configuration (next lecture)
  - Disks can transfer at 10MB/sec max, but 60% utilization is max
  - Keep I/O operations < 60% of maximum (typically 80 I/O ops/second)
- Use maximum number of buses: connection from disk to memory.
  - Keep bus utilization < 60%
  - Prefer SCSI to IDE except in fairly small implementations
- Configure at least twice the space as the raw data occupies
  - Temp tables, indexes, logs, etc. all take space
  - Allow room for future growth
  - Keep storage use < 60% of disk capacity
- Backup necessary:
  - FibreChannel to disks 2km away, with mirroring (next lecture)
  - Tape backup. Compute capacity on max time to restore, which is 5 times as long as it takes to back up.

Database server: data distribution

- Six storage functions in a DBMS
  - Data tables: hold actual data
  - Indexes (incl keys): increase performance of tables
  - Temporary tables: used in joins, sorts, etc.
  - Transaction logs: used for failure recovery
  - Rollback buffers: keep transactions before committed
  - System tables: data dictionary, DBMS control info
- Even minimal database server should have 4 disks
  - One for operating system (Win or UNIX) and system (swap) space
  - One for data (tables)
  - One for indexes
  - One for log:
  - Log writes are synchronous and slow. Log write must be on physical disk before transaction (update or insert) can be committed. Bottleneck.

Database server: data distribution, cont

- Example:
  - 1.5GB database, 400MB indexes, 40MB log fit on 2GB disk
  - Update writes to tables, indexes and log:
    - Shuttles disk arm across disk each transaction
    - Makes index reads, log writes look like random access vs sequential
    - Seeks tend to be longer than average
      - Adjacent cylinder seeks are 2ms
      - Maximum seeks are 20ms, which is what we get with 1 disk for DBMS: almost every seek will be a maximum seek!
  - Even ignoring space considerations, always place log on separate disk
    - It's the recovery mechanism if the data disk crashes!
    - If it's on the same disk, it will be lost also.