Database Applications and Web-Enabled Databases

University of California, Berkeley

School of Information

INFO 257: Database Management

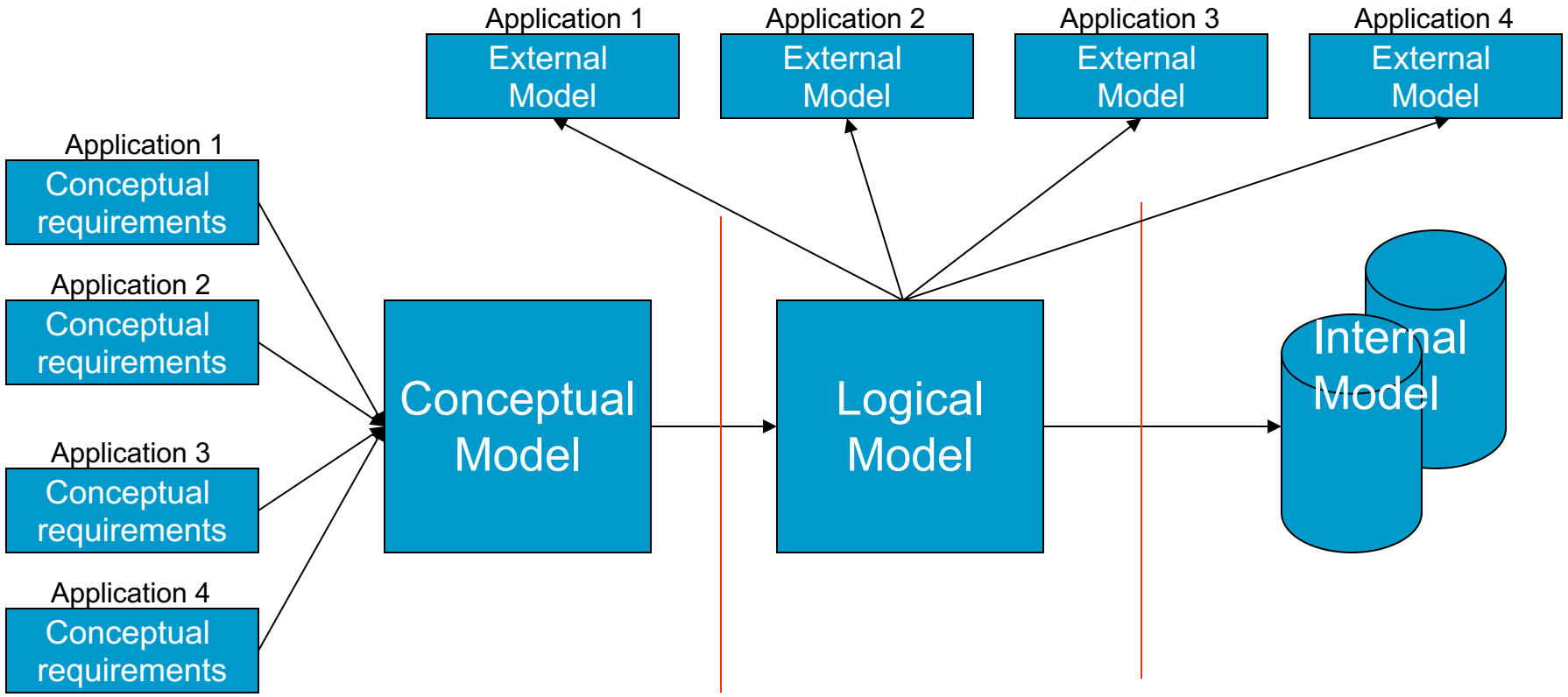
Lecture Outline & Agenda

- Review
  - Database design review
  - Introduction to SQL & MariaDB
- Databases for Web Applications – Overview
- Project 2b Examples and Final Project Outline
- Work on Workshops

Lecture Outline

- Review
  - Database design review
  - Introduction to SQL & MariaDB
- Databases for Web Applications – Overview

Database Design Process



Physical Model: SQL for Creation

- Workshop 3 looks at how an SQL "script" could be created that would create each of the relational tables, define primary keys and indexes and load data into the database

MySQL Data Types

- MySQL supports all of the standard SQL numeric data types. These types include the exact numeric data types (INTEGER, SMALLINT, DECIMAL, and NUMERIC), as well as the approximate numeric data types (FLOAT, REAL, and DOUBLE PRECISION). The keyword INT is a synonym for INTEGER, and the keyword DEC is a synonym for DECIMAL
- Numeric (can also be declared as UNSIGNED)
  - BIT(n) (variable field of n bits)
  - BOOL or BOOLEAN (internally is TINYINT with value of 0 for FALSE)
  - TINYINT (1 byte)
  - SMALLINT (2 bytes)
  - MEDIUMINT (3 bytes)
  - INTEGER (4 bytes)
  - INT (4 bytes - Synonym)
  - BIGINT (8 bytes)
  - NUMERIC or DECIMAL (Packed - up to 65 digits - DEC, FIXED synonyms)
  - FLOAT
  - DOUBLE (or DOUBLE PRECISION)
  - **SERIAL** = BIGINT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE

MySQL Data Types

- The date and time types for representing temporal values are DATETIME, DATE, TIMESTAMP, TIME, and YEAR. Each temporal type has a range of legal values, as well as a "zero" value that is used when you specify an illegal value that MySQL cannot represent
  - DATETIME '0000-00-00 00:00:00'
  - DATE '0000-00-00'
  - TIMESTAMP (4.1 and up) '0000-00-00 00:00:00'
  - TIMESTAMP (before 4.1) 00000000000000
  - TIME '00:00:00'
  - YEAR 0000

MySQL Data Types

- The string types are CHAR, VARCHAR, BINARY, VARBINARY, BLOB, TEXT, ENUM, and SET
- Maximum length for CHAR is 255 and VARCHAR is **65,535** (limited by row size)

Value	CHAR(4)	Storage	VARCHAR(4)	Storage
""	" "	4	""	1
"ab"	"ab "	4	"ab"	3
"abcd"	"abcd"	4	"abcd"	5
"abcdefg"	"abcd"	4	"abcd"	5

- **For longer things there is BLOB and TEXT**

ALTER Table

- ALTER TABLE table-name ADD COLUMN col_name col_definition;
- … DROP COLUMN col_name;
- … CHANGE col_name new_col_definition;
- Adds/removes a new column from an existing database table
- Many other options for adding constraints (like NOT NULL, or PRIMARY KEY), etc.

INSERT

- INSERT INTO table-name (attr1, attr4, attr5,…, attrK) VALUES (“val1”, val4, val5,…, “valK”);
- Adds a new row(s) to a table.
- INSERT INTO table-name (attr1, attr4, attr5,…, attrK) VALUES SELECT ...

Creating a new table data from existing tables

- Syntax:
  - INSERT INTO tablename (attr1, attr2, attr3)

    SELECT [DISTINCT] xattr1, xattr2, xattr3 FROM rel1 r1, rel2 r2,… rel3 r3 WHERE condition1 {AND | OR} condition2 ORDER BY attr1 [DESC], attr3 [DESC]

tablename has to previously exist for this to work in MySQL…

DELETE

- DELETE FROM table-name WHERE <where clause>;
- Removes rows from a table.

UPDATE

- UPDATE tablename SET attr1=newval, attr2 = newval2 WHERE <where clause>;
- changes values in existing rows in a table (those that match the WHERE clause).

DROP Table

- DROP TABLE tablename;
- Removes a table from the database.

Lecture Outline

- Review
  - Introduction to SQL
- Databases for Web Applications – Overview

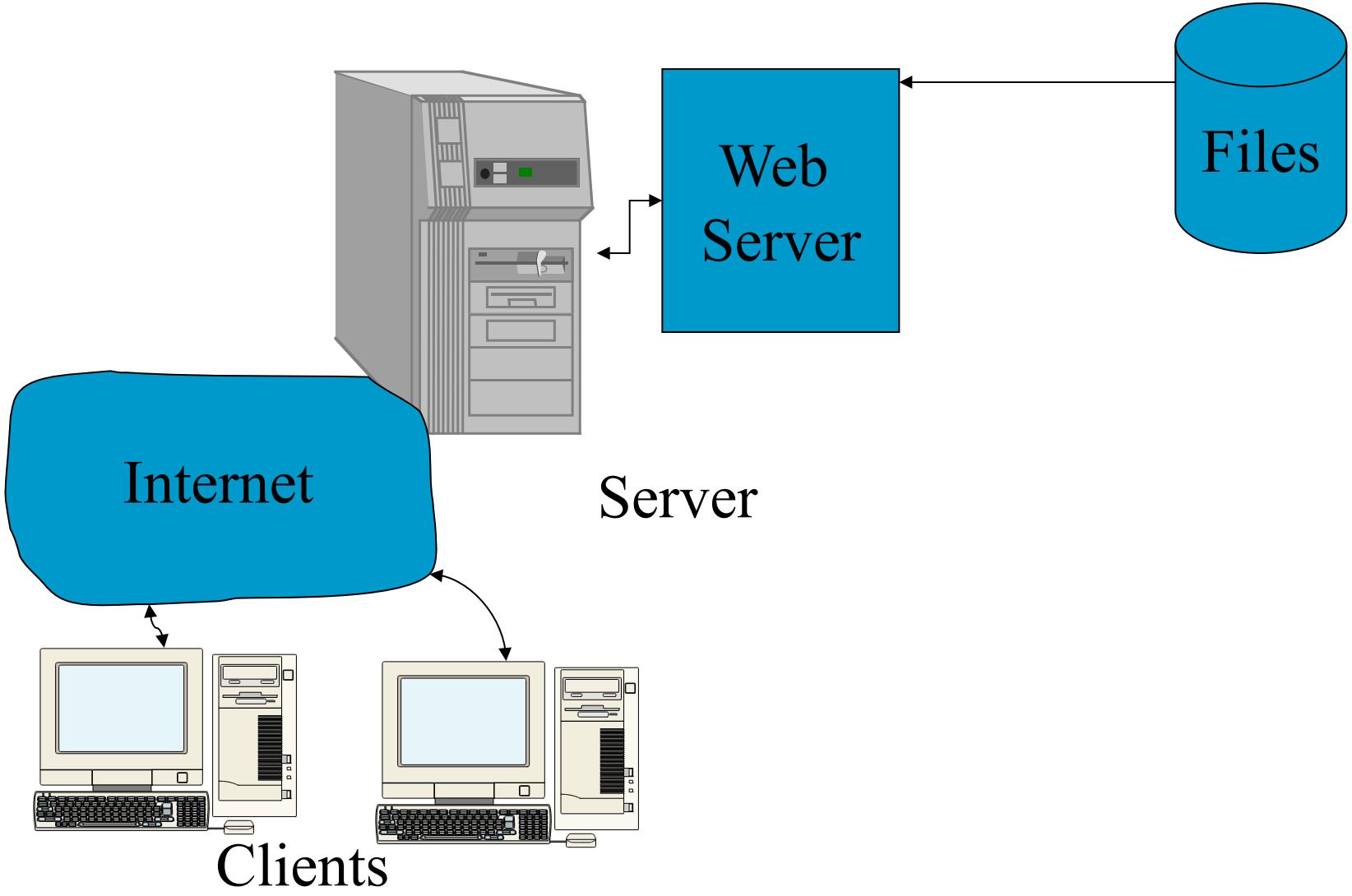
Overview

- Why use a database system for Web design and e-commerce?
- What systems are available?
- Architectures
- Pros and Cons of different web database systems?
- Search Engines for Intranet and Intrasite searching

Why Use a Database System?

- Simple Web sites with only a few pages don't need much more than static HTML files

Simple Web Applications



Adding Dynamic Content to the Site

- Small sites can often use HTML and JavaScript to access data files to create dynamic content for small sites.

Issues For Scaling Up Web Applications

- Performance
- Scalability
- Maintenance
- Data Integrity
- Transaction support

Performance Issues

- Problems arise as both the data to be managed and usage of the site grows.
  - Interpreted scripts are inherently slower than compiled native programs
  - Startup time for each connection slows down the application
  - Load on the system compounds the problem
  - Tied to other scalability issues

Scalability Issues

- Well-designed database systems will permit the applications to scale to accommodate very large databases
  - A script that works fine scanning a small data file may become unusable when the file becomes large.
  - Issues of transaction workload on the site
    - Starting a separate copy of a python program for each user is NOT a scalable solution as the workload grows

Maintenance Issues

- Dealing with multiple data files (customer list, product list, customer orders, etc.) using Python means:
  - If any data element in one of the files changes, all scripts that access that file must be rewritten
  - If files are linked, the programs must insure that data in all the files remains synchronized
  - A large part of maintenance will involve dealing with data integrity issues
  - Unanticipated requirements may require rewriting scripts

Data Integrity Constraint Issues

- These are constraints we wish to impose in order to protect the database from becoming inconsistent.
- Five basic types
  - Required data
  - attribute domain constraints
  - entity integrity
  - referential integrity
  - enterprise constraints

Transaction support

- Concurrency control (ensuring the validity of database updates in a shared multiuser environment).

No Concurrency Control: Lost updates

John	Marsha
• Read account balance (balance = $1000)	
	• Read account balance (balance = $1000)
• Withdraw $200 (balance = $800)	
	• Withdraw $300 (balance = $700)
• Write account balance (balance = $800)	
	• Write account balance (balance = $700)
	ERROR!

Concurrency Control: Locking

- Locking levels
  - Database
  - Table
  - Block or page
  - Record
  - Field
- Types
  - Shared (S locks)
  - Exclusive (X locks)

Concurrency Control: Updates with X locking

John	Marsha
• Lock account balance • Read account balance (balance = $1000) • Withdraw $200 (balance = $800) • Write account balance (balance = $800) • Unlock account balance	• Read account balance (DENIED) • Lock account balance • Read account balance (balance = $800) • etc...

Concurrency Control: Deadlocks

John	Marsha
• Place S lock	
• Read account balance (balance = $1000)	• Place S lock
	• Read account balance (balance = $1000)
• Request X lock (denied)	
• wait ...	• Request X lock (denied)
	• wait...

Deadlock!

Managing Deadlock

- Deadlock prevention:
  - Lock all records required at the beginning of a transaction
  - Two-phase locking protocol
    - Growing phase
    - Shrinking phase
  - May be difficult to determine all needed resources in advance
- Deadlock Resolution:
  - Allow deadlocks to occur
  - Mechanisms for detecting and breaking deadlock
    - Resource usage matrix
    - Back out one deadlock at a time
    - Rerun transaction

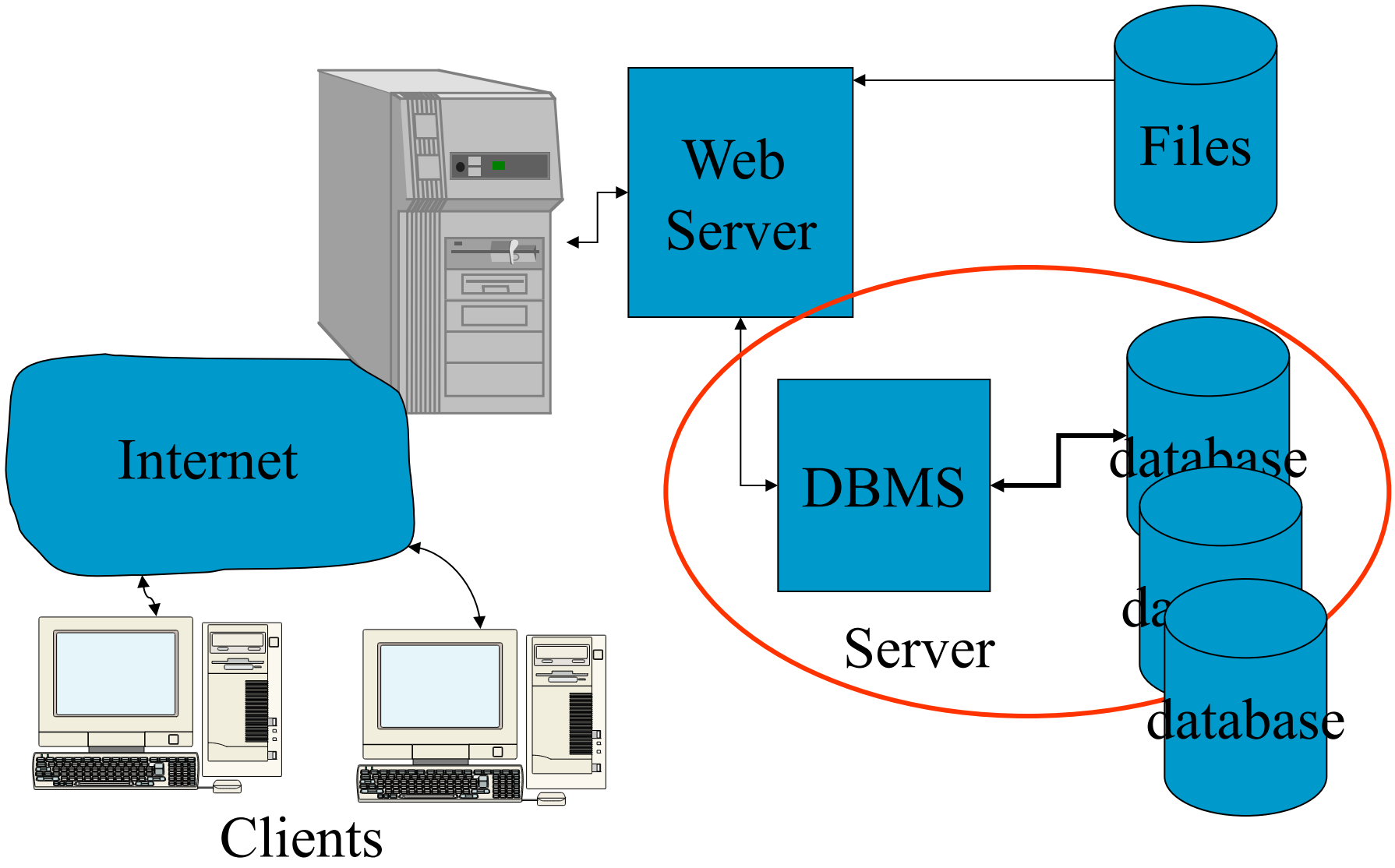
Transaction Processing

- Transactions should be ACID:
  - Atomic – Results of transaction are either all committed or all rolled back
  - Consistent – Data is transformed from one consistent state to another
  - Isolated – The results of a transaction are invisible to other transactions
  - Durable – Once committed the results of a transaction are permanent and survive system or media failures

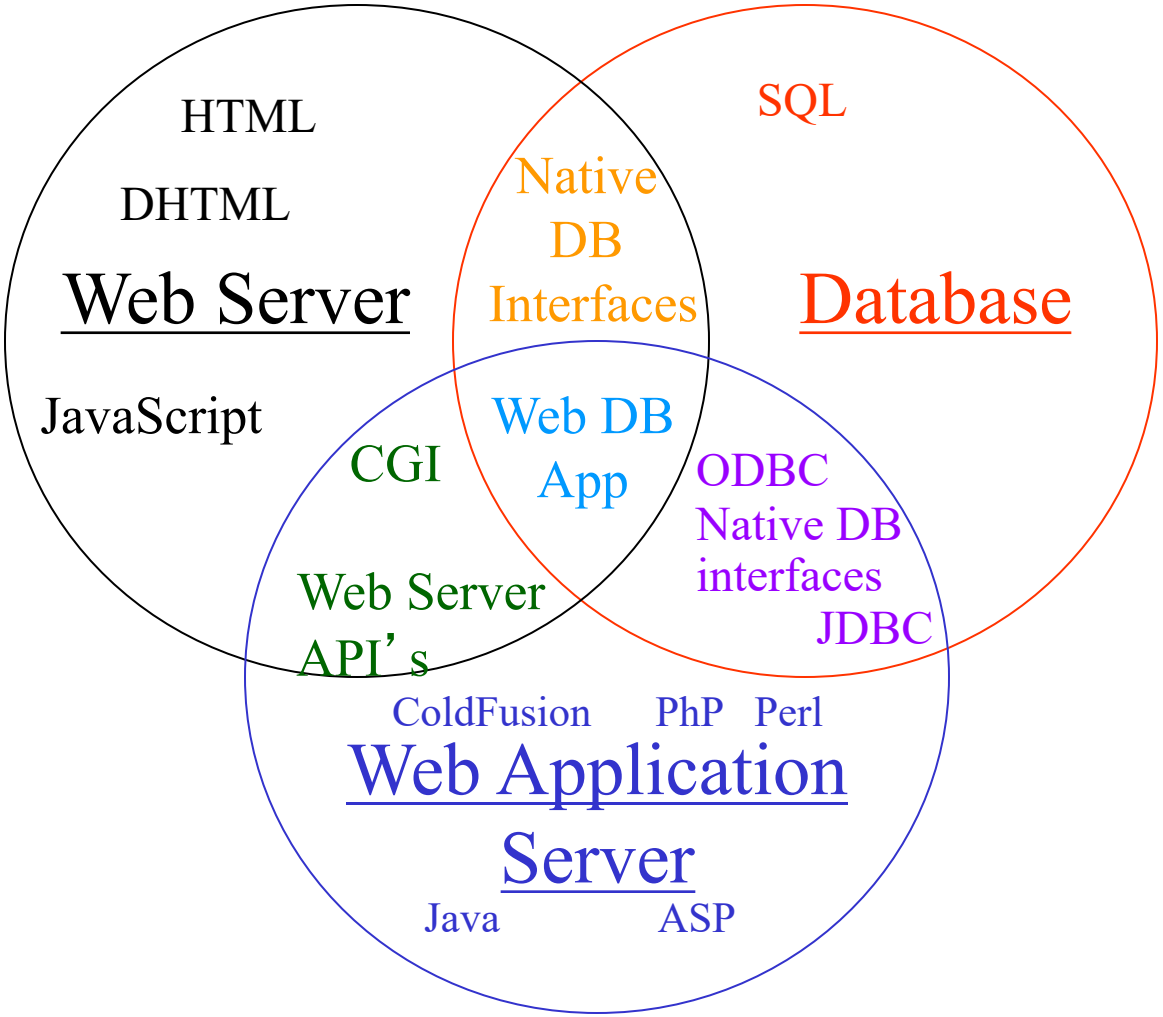
Why Use a Database System?

- Database systems have concentrated on providing solutions for all of these issues for scaling up Web applications
  - Performance
  - Scalability
  - Maintenance
  - Data Integrity
  - Transaction support
- While systems differ in their support, most offer some support for all of these.

Dynamic Web Applications 2



Server Interfaces



Adapted from
John P Ashenfelter,
Choosing a Database for Your Web Site

Client/Server Architectures

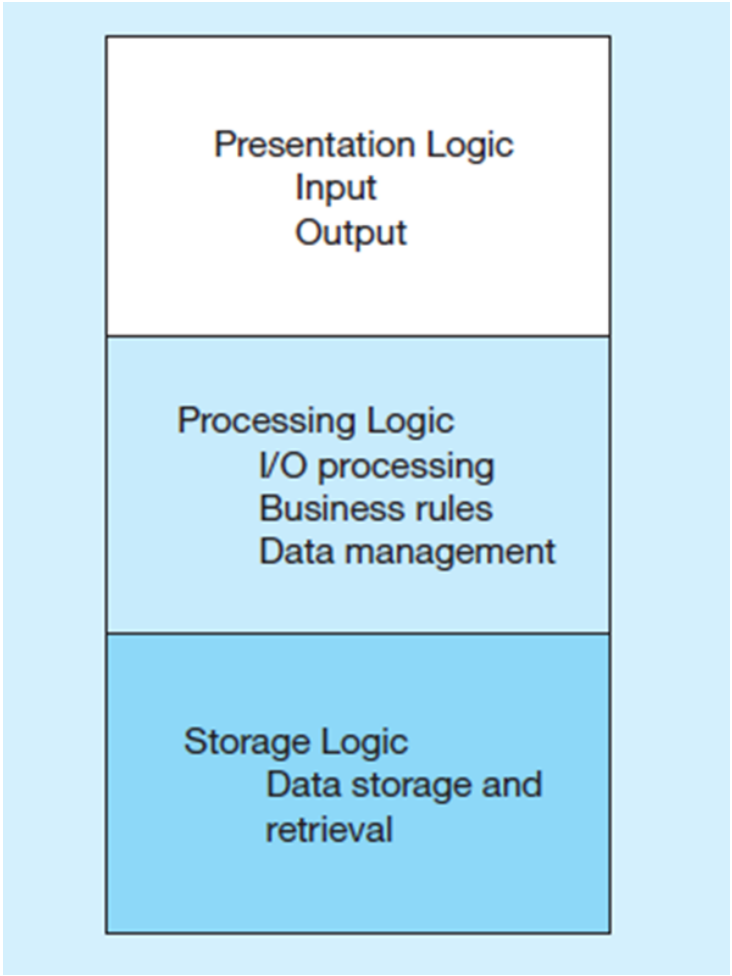
- Networked computing model
- Processes distributed between clients and servers
- Client – Workstation (PC, smartphone, tablet) that requests and uses a service
- Server – Powerful computer (PC/mini/mainframe) that provides a service
- For DBMS, server is a database server
- For the Internet, server is a Web server

Application Logic Components

GUI interface (e.g. through a browser)

Procedures, functions, programs

DBMS activities



Application Partitioning

- Placing portions of the application code in different locations (client vs. server) after it is written
- Advantages
  - Improved performance
  - Improved interoperability
  - Balanced workloads

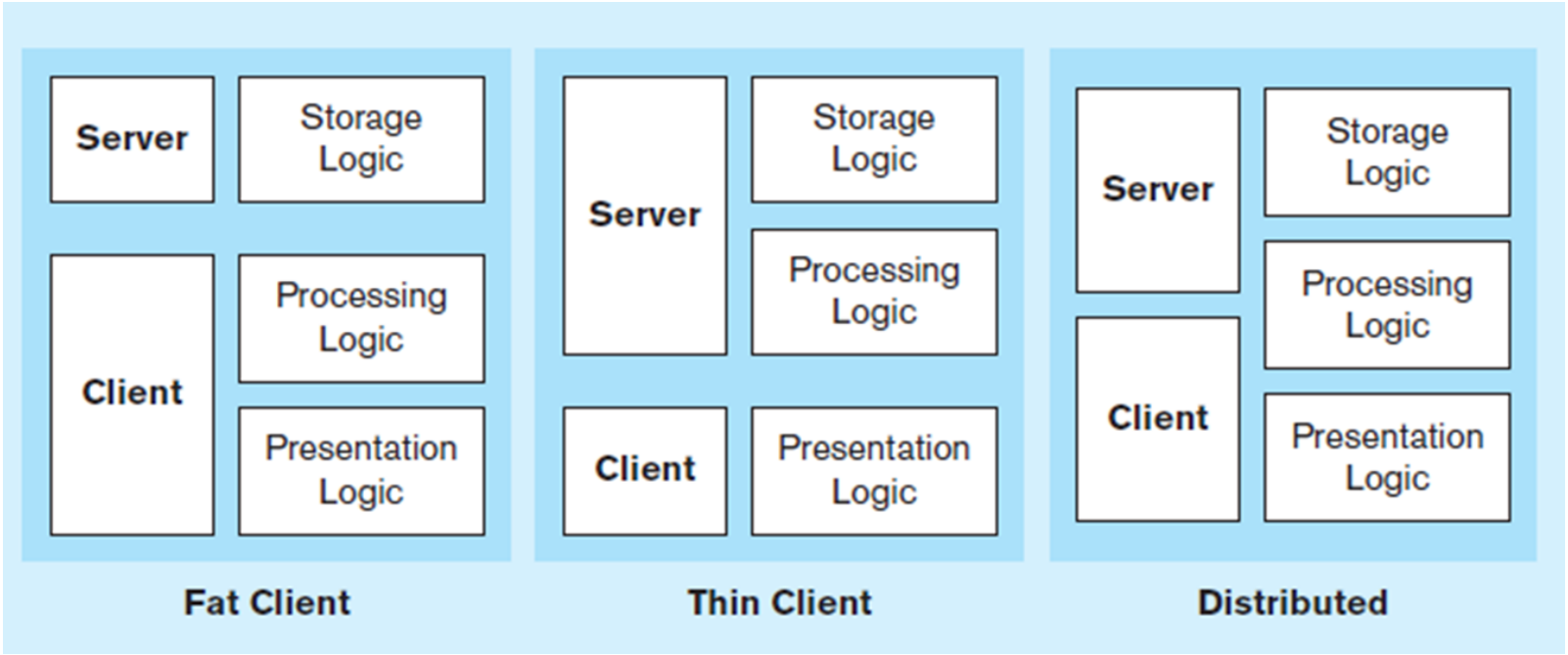
Fat vs. Thin Clients

- Fat client – a client PC that is responsible for processing presentation logic, extensive application and business rules logic, and many DBMS functions
- Thin client – an application where the client (PC) accessing the application primarily provides the user interfaces and some application processing, usually with no or limited local data storage

Common Logic Distributions (1 of 2)

a) Two-tier client-server environments

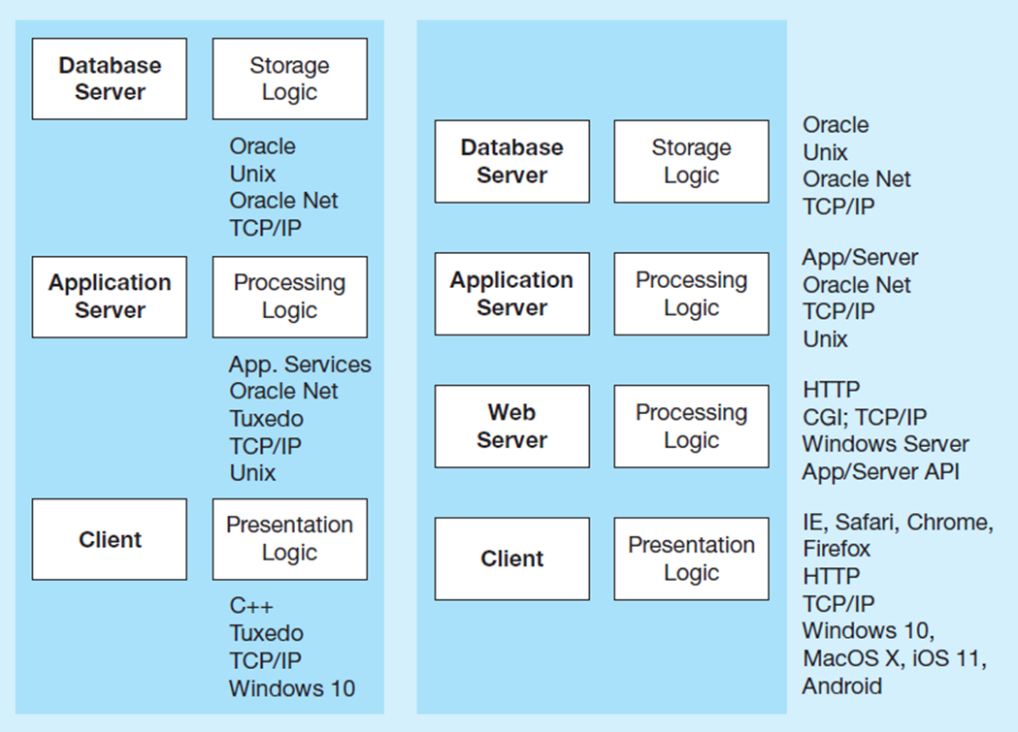
Processing logic could be at client (fat client), server (thin client), or both (distributed environment).



Common Logic Distributions (2 of 2)

b) Three-tier and *n*-tier client-server environments

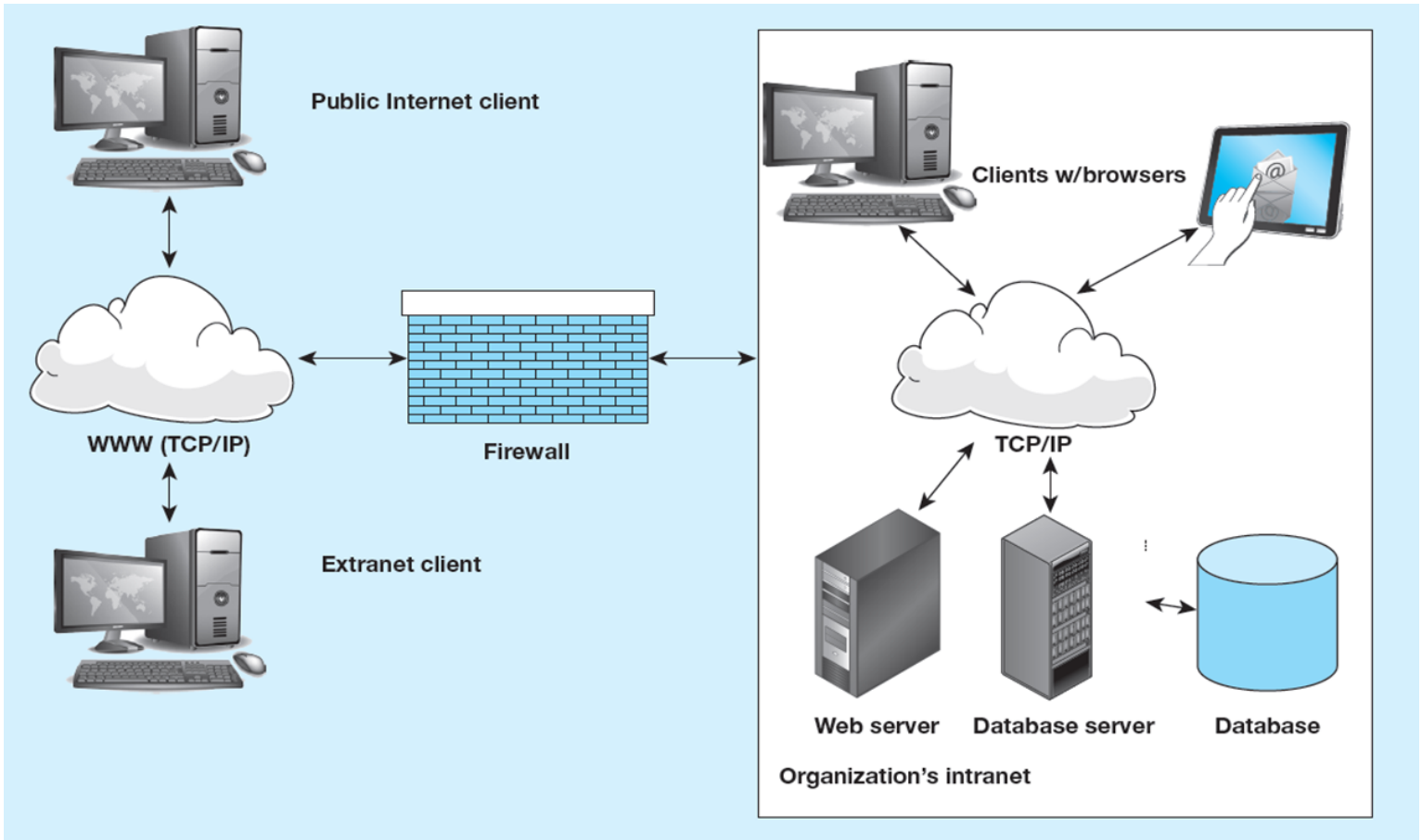
Processing logic will be at application server or Web server.



Web Application Components

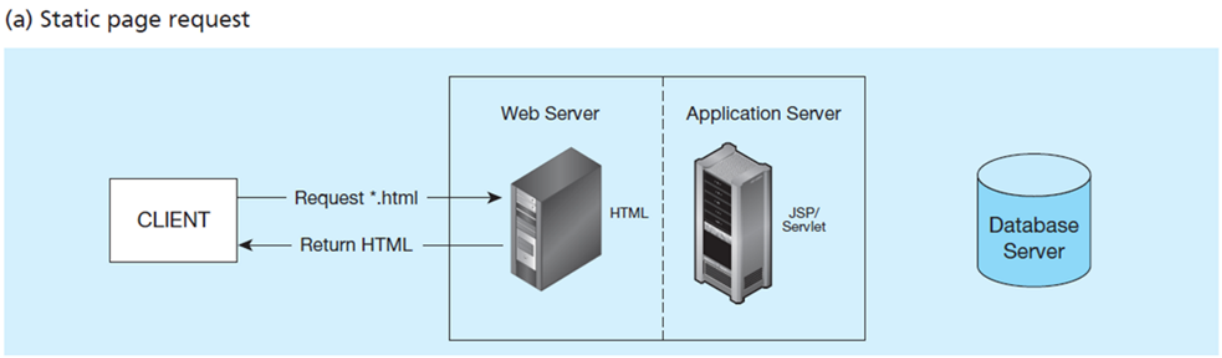
- Database server – hosts the DBMS
  - e.g., Oracle, SQL Server, Informix, MS Access, MySql
- Web server – receives and responds to browser requests using HTT P protocol
  - e.g., Apache, Internet Information Services (IIS)
- Application server – software building blocks for creating dynamic Web sites
  - e.g., MS ASP.NET framework, Java EE, PHP
- Web browser – client program that sends Web requests and receives Web pages
  - e.g., Internet Explorer, Mozilla Firefox, Apple Safari, Google Chrome

A Database-Enabled Intranet/Internet Environment

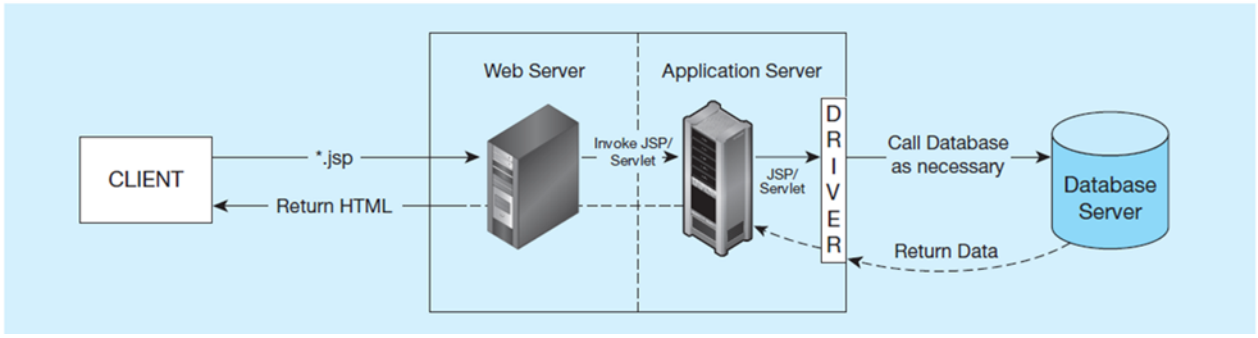


Information Flow in a Three-Tier Architecture

(a) Has no server side processing, just a page return



(b) Has server side processing, including database access



Middleware and APIs

- Middleware – software that allows an application to interoperate with other software without requiring user to understand and code low-level operations
- Application Program Interface (API) – routines that an application uses to direct the performance of procedures by the computer's operating system
- Common database APIs – ODBC, ADO .NET, JDBC

Steps for Using Databases via Middleware APIs

1. Identify and register a database driver.
2. Open a connection to a database.
3. Execute a query against the database.
4. Process the results of the query.
5. Repeat steps 3–4 as necessary.
6. Close the connection to the database.

Database Access from a Java Program

Java implementation of database access to an Oracle database using JDBC

```
import java.sql.*;
public class TestJDBC {
  public static void main(String[ ] args) {
  try {
  Driverd =
  (Driver)Class.forName("oracle.jdbc.driver.OracleDriver").newInstance( );
  System.out.println(d);
  DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver( ));
  Connection conn =
  DriverManager.getConnection ("jdbc:oracle:thin:@durga.uits.indiana.edu:1
  521:OED1", args[0], args[1]);

  Statement st = conn.createStatement( );
  ResultSet rec = st.executeQuery("SELECT * FROM Student");
  while(rec.next( )) {
    System.out.println(rec.getString("name"));}
  conn.close( );
    }
  catch (Exception e) {
System.out.println("Error – " + e);
  }
  }
}
```

Register the driver to be used.

Identify the type of driver to be used.

Open a connection to a database.

Create a Statement variable that can be used to issue queries against the database

Issue a query and get a result.

Process the result, one row at a time.

Close the connection.

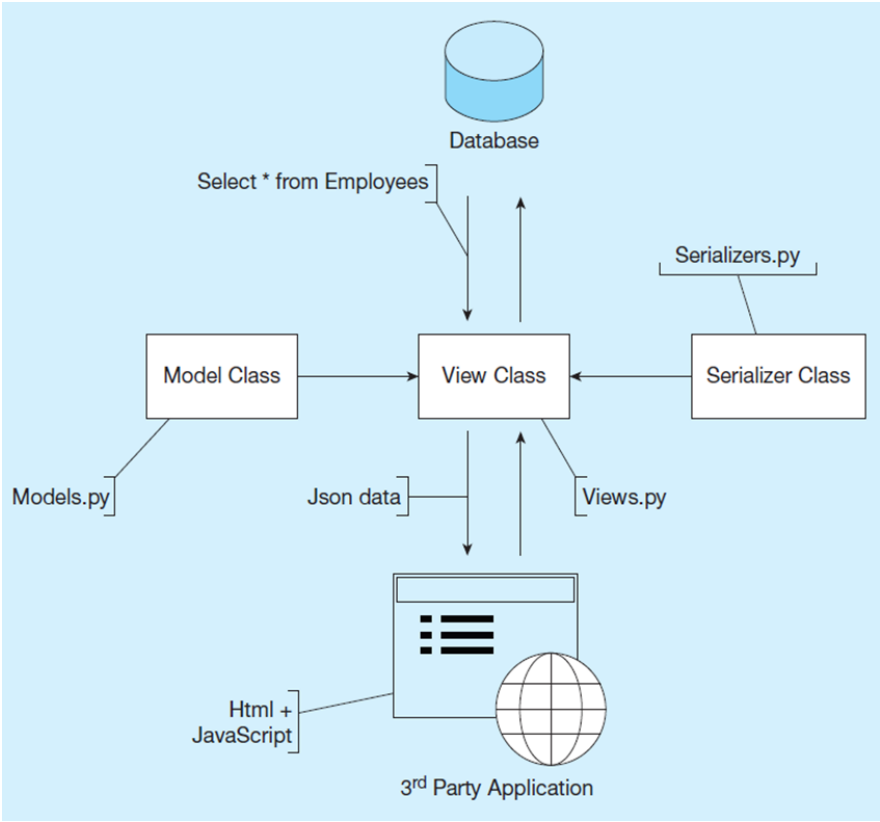
System Architecture of Sample Application

Python application for database processing.

Model class creates instances of data from table queries.

Serializer class transforms database data into a form that can be used by the client (JSON).

View classes display in browsers for view or edit.



Considerations in 3-Tier Applications

- Stored procedures
  - Code logic embedded in DBMS
  - Improve performance, but proprietary
- Transactions
  - Involve many database updates
  - Either all must succeed, or none should occur
- Database connections
  - Maintaining an open connection is resource-intensive
  - Use of connection pooling

Advantages and Disadvantages of Stored Procedures

- Advantages
  - Performance improves for compiled SQL statements
  - Reduced network traffic
  - Improved security
  - Improved data integrity
  - Thinner clients
- Disadvantages
  - Programming takes more time
  - Proprietary, so algorithms are not portable

Benefits of Three-Tier Architectures

- Scalability
- Technological flexibility
- Long-term cost reduction
- Better match of systems to business needs
- Improved customer service
- Competitive advantage
- Reduced risk

What Database systems are available?

- Choices depend on:
  - Size (current and projected) of the application
  - Hardware and OS Platforms to be used in the application
  - Features required
    - E.g.: SQL? Upgrade path? Full-text indexing? Attribute size limitations? Locking protocols? Direct Web Server access? Security?
  - Staff support for DBA, etc.
  - Programming support (or lack thereof)
  - Cost/complexity of administration
  - Budget

Desktop Database Systems

System (producer)	Platform	SQL	ODBC	Scaling	Price
Access (Microsoft)	Windows	Yes	Yes	SQL Server	~$200
FoxPro (Microsoft)	Windows, Mac	Yes	Yes	SQL Server	~$200
FileMaker (FileMaker)	Windows, Mac	No	No	FileMaker Server	~$200
Excel (Microsoft)	Windows, Mac	No	Yes	Convert to Acces	~$200
Files (owner)	Windows, Mac	No	No	Import into DB	?

- Individuals or very small enterprises can create DBMS-enabled Web applications relatively inexpensively
- Some systems will require an application server (such as Python/Flask) to provide the access path between the Web server and the DBMS

Pros and Cons of Database Options

- Desktop databases
  - usually simple to set up and administer
  - inexpensive
  - often will not scale to a very large number of users or very large database size
  - May lack locking management appropriate for multiuser access
  - Poor handling for full-text search
  - Well supported by application software (Python, PHP, etc.)

Enterprise Database Systems

System	Platform	SQL	ODBC	JDBC	Web?
SQL-Server (Microsoft)	WIndowsNT -2000	Yes	Yes	?	Yes (IIS)
Oracle Internet Platform	Unix, Linux, NT	Yes	Yes	Yes	Yes
Informix Internet Foundation.2000	Unix, Linux, NT	Yes	Yes	Yes	Yes
Sybase Adaptive Server	Unix, Linux, NT	Yes	Yes	Yes	Yes
DB2 (IBM)	IBM Unix, Linux, NT	Yes	Yes	Yes	Yes?

- Enterprise servers are powerful and available in many different configurations
- They also tend to be VERY expensive
- Pricing is usually based on users, or CPU's

Pros and Cons of Database Options

- Enterprise databases
  - Can be very complex to set up and administer
    - Oracle, for example recommends RAID-1 with 7x2 disk configuration as a bare minimum, more recommended
  - Expensive
  - Will scale to a very large number of users
  - Will scale to very large databases
  - Incorporate good transaction control and lock management
  - Native handling of Text search is poor, but most DBMS have add-on text search options
  - Support for applications software (Python, PHP, etc.)

Free Database Servers

System	Platform	SQL	ODBC	JDBC	
mSQL	Unix, Linux	Yes	Yes	No(?)	
MariaDB	Unix, Linux, NT	Yes	Yes	Yes	
PostgreSQL	Unix, Linux, NT	Yes	Yes	Yes	

- System is free, but there is also no help line.
- Include many of the features of Enterprise systems, but tend to be lighter weight
- Versions may vary in support for different systems
- Open Source -- So programmers can add features

Pros and Cons of Database Options

- Free databases
  - Can be complex to set up and administer
  - Inexpensive (FREE!)
  - usually will scale to a large number of users
  - Incorporate good transaction control and lock management
  - Native handling of Text search has improved, and there are IR-like capabilities in MySQL and PostgreSQL
  - Support for applications software (ColdFusion, Python, etc.)

Embedded Database Servers

System	Platform	SQL	ODBC	JDBC	Web?
Oracle Berkeley DB	Unix, Linux, Win	No	No	Java API	No?
Solid	Unix, Linux, Win	Yes	Yes	Yes	Yes
SQLite	Unix,Linux,Win	Yes	No	No	Yes

- May require programming experience to install
- Tend to be fast and economical in space requirements
- Includes many NOSQL databases

Pros and Cons of Database Options

- Embedded databases
  - Must be embedded in a program
  - Can be incorporated in a scripting language
  - inexpensive (for non-commercial application)
  - May not scale to a very large number of users (depends on how it is used)
  - (May) Incorporate good transaction control and lock management
  - Text search support is minimal
  - May not support SQL

NOSQL Databases

System	Platform	SQL	ODBC	JDBC	Web?
MongoDB	Unix, Linux, Win	No	No	No	?
REDIS	Unix, Linux, Win	NO	NO	No	?

Evaluation Criteria	Tokyo Cabinet + Tokyo Tyrant	Berkeley DB + MemcacheDB	Voldemort + BDB JE	Redis	MongoDB
Insertion (small data set)	4	2	2	4	4
Insertion (large data set)	1	1	1	1	3
Random Read (small data set)	4	2	2	4	4
Random Read (large data set)	2	2	1	3	3
Speed Consistence	1	3	2	4	4
Storage Efficiency	4	2	1	2	3
Horizontal Scalability	4	3	2	3	4
Manageability	4	4	2	2	4
Stability	4	4	2	3	4
Feature Set	1	1	1	1	2
Project Activeness and Community Support	2	1	1	2	4

PerfectMarket™ Content Matters

Database Security

- Different systems vary in security support:
  - Views or restricted subschemas
  - Authorization rules to identify users and the actions they can perform
  - User-defined procedures (and rule systems) to define additional constraints or limitations in using the database
  - Encryption to encode sensitive data
  - Authentication schemes to positively identify a person attempting to gain access to the database

Views

- A subset of the database presented to some set of users.
  - SQL: CREATE VIEW viewname AS SELECT field1, field2, field3,…, FROM table1, table2 WHERE <where clause>;
  - Note: “queries” in Access function as views.

Authorization Rules

- Most current DBMS permit the DBA to define “access permissions” on a table by table basis (at least) using the GRANT and REVOKE SQL commands.
- Some systems permit finer grained authorization (most use GRANT and REVOKE on variant views.
- Some desktop systems have poor authorization support.

Database Backup and Recovery

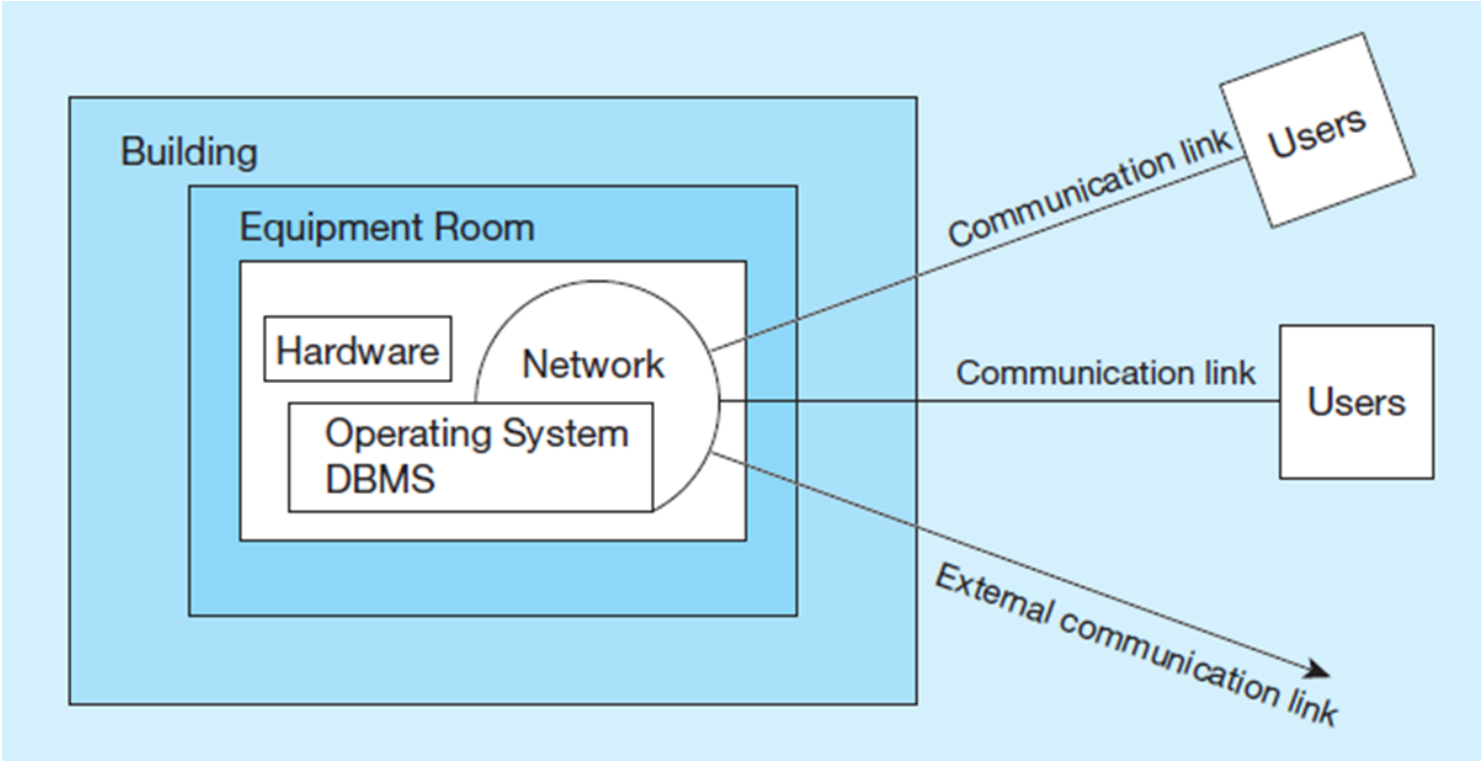
- Backup
- Journaling (audit trail)
- Checkpoint facility
- Recovery manager

Threats to Data Security

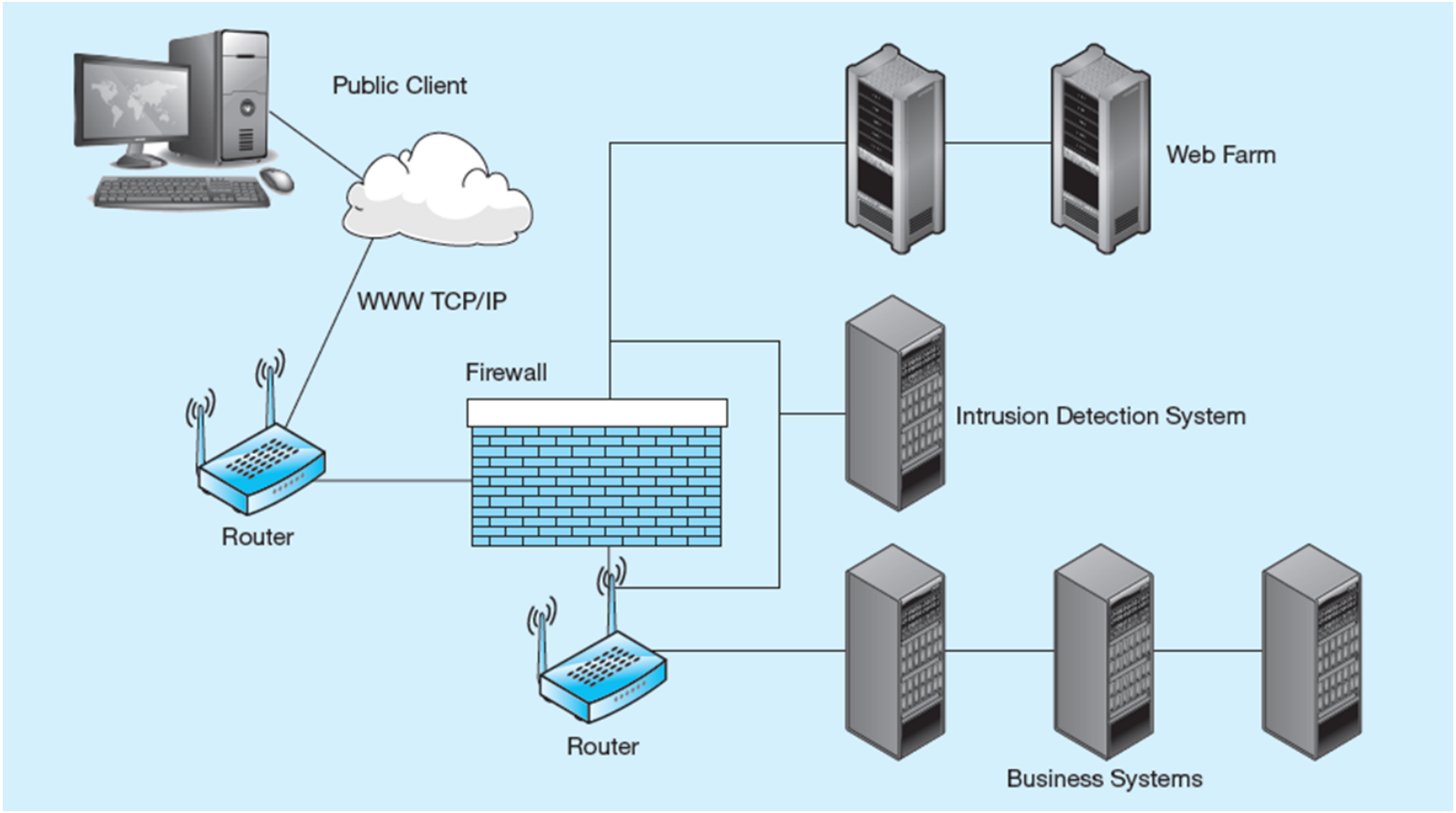
- Accidental losses attributable to:
  - Human error
  - Software failure
  - Hardware failure
- Theft and fraud
- Loss of privacy or confidentiality
  - Loss of privacy (personal data)
  - Loss of confidentiality (corporate data)
- Loss of data integrity
- Loss of availability (e.g., through sabotage)

Possible Locations of Data Security Threats

Threats come from many sources and vulnerabilities exist in multiple places within an information system.



Establishing Internet Security



Client–Server Application Security

- Static HTML files are easy to secure
  - Standard database access controls
  - Place Web files in protected directories on server
- Dynamic pages are harder
  - User authentication
  - Session security
  - SSL for encryption
  - Restrict number of users and open ports
  - Remove unnecessary programs

Data Privacy

- W3C Web Privacy Standard
  - Platform for Privacy Protection (P3P)
- Addresses the following:
  - Who collects data
  - What data is collected and for what purpose
  - Who is data shared with
  - Can users control access to their data
  - How are disputes resolved
  - Policies for retaining data
  - Where are policies kept and how can they be accessed

Web Application Server Software

- **Python + Flask**
- Java/Spring Struts
- Ruby on Rails
- Django
- PHP
- ASP
- JSP

Next Week(s)

- Docker Containers
- Python + Flask
- Twitter Bootstrap?

Now

- Example Project
- Work on Workshop