Database and Database Management System

Avinash Maskey

Introduction

- A ***database*** is a collection of related data that is stored and organized in a manner that enables information to be retrieved as needed.
- Database software—more formally called a ***database management system (DBMS)***—is used to create, maintain, and access a database.
- A DBMS also controls the organization of the data and protects the integrity and security of the data so it is entered accurately into the database and then protected against both intentional and accidental damage.
- While ***batch processing*** can be used with databases, most database applications today occur in ***real time***.

Introduction (Contd.)

- A key component of a DBMS is the ***database engine***—the part of the program that actually stores and retrieves data.
- In addition to a database engine, most DBMSs come bundled with a set of tools to perform a variety of necessary tasks, such as creating forms (used to input data) and reports (used to output data), and interfacing with query languages and programming languages for complex applications.
- Programming languages typically used with databases today include Visual Basic, Java, and C++, PHP etc.

Introduction (Contd.)

- A database typically consists of interrelated tables that contain **fields** and **records**.
- A ***field*** (also called a **column**) holds a single category of data (such as customer names or employee phone numbers) that will be stored in a database.
- A ***record*** (also called a **row**) is a collection of related fields.
- The technical difference between the terms ***row*** and ***record*** in database terminology is that a row is contained within a single database table, but a record is a collection of fields, which can be either a specific row from a single table or a collection of related fields from multiple tables, such as the records shown in the results of the Order Request screen in Figure.
- A ***relational database*** includes tables containing rows and columns. For example, a typical business order entry database would include a table that describes a customer with columns for name, address, phone number and so forth.

Relational Vs Non-relational

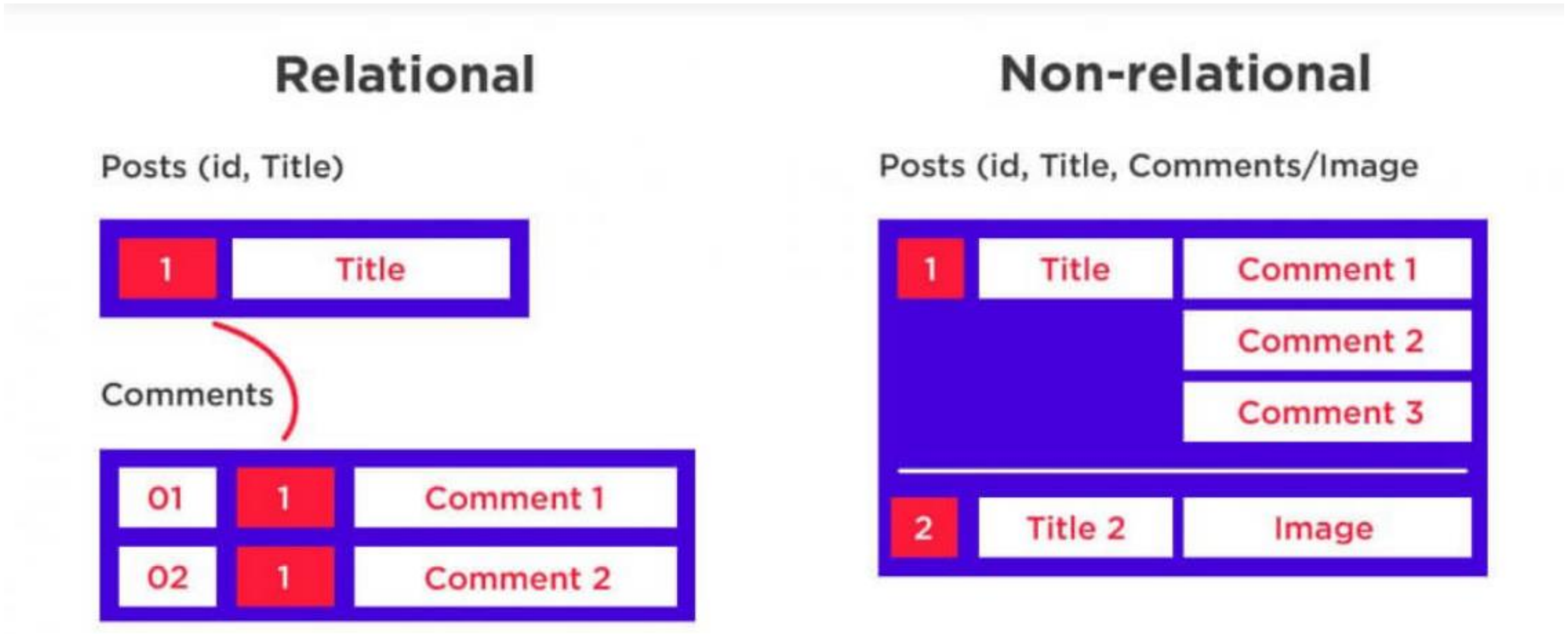
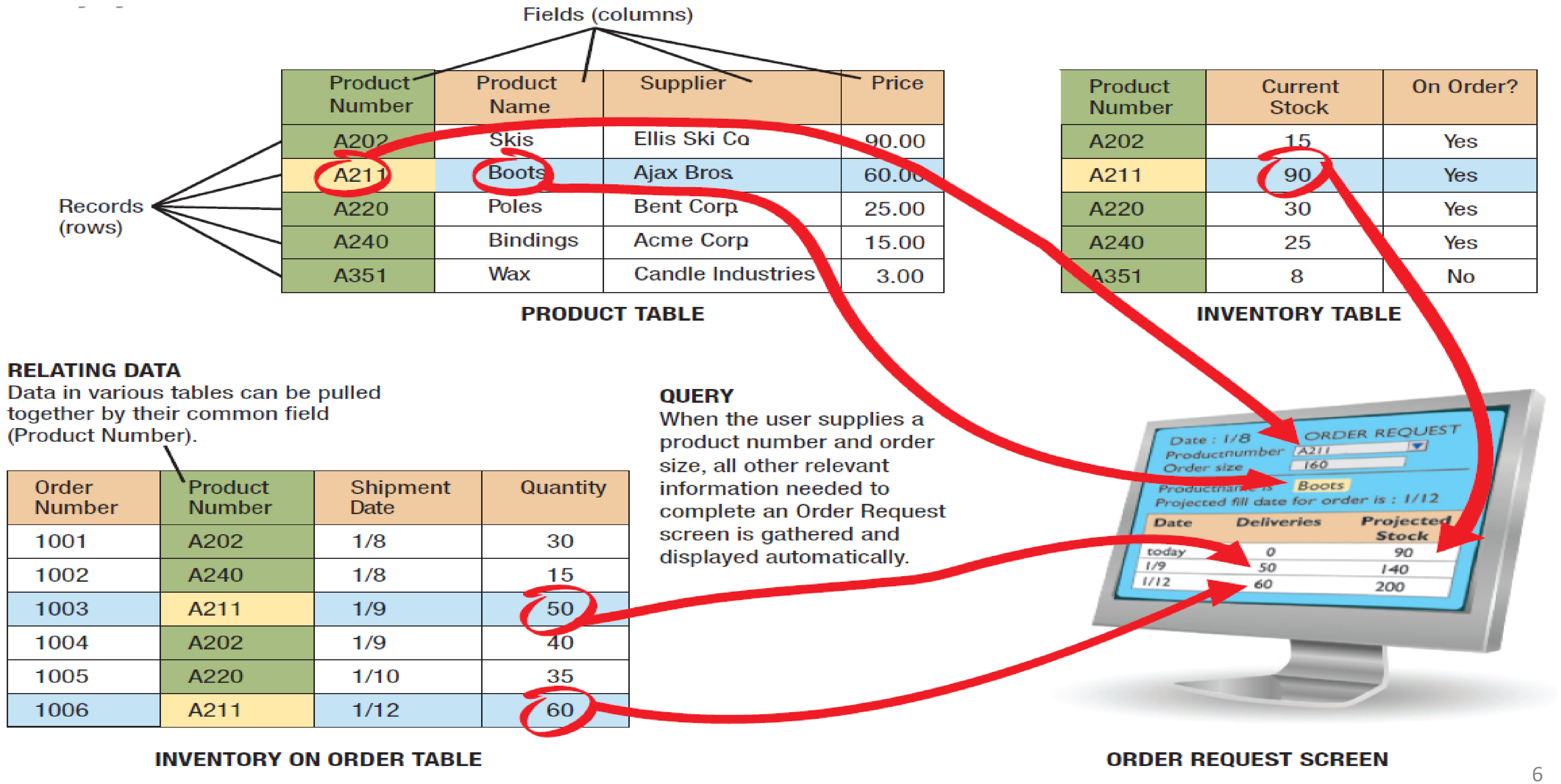


Fig: Using a RELATIONAL DATABASE in an inventory system



Product Number	Product Name	Supplier	Price
A202	Skis	Ellis Ski Co.	90.00
A211	Boots	Ajax Bros.	60.00
A220	Poles	Bent Corp.	25.00
A240	Bindings	Acme Corp	15.00
A351	Wax	Candle Industries	3.00

PRODUCT TABLE

Product Number	Current Stock	On Order?
A202	15	Yes
A211	90	Yes
A220	30	Yes
A240	25	Yes
A351	8	No

INVENTORY TABLE

RELATING DATA
Data in various tables can be pulled together by their common field (Product Number).

Order Number	Product Number	Shipment Date	Quantity
1001	A202	1/8	30
1002	A240	1/8	15
1003	A211	1/9	50
1004	A202	1/9	40
1005	A220	1/10	35
1006	A211	1/12	60

INVENTORY ON ORDER TABLE

QUERY
When the user supplies a product number and order size, all other relevant information needed to complete an Order Request screen is gathered and displayed automatically.

Date : 1/8 ORDER REQUEST
Productnumber A211
Order size 160
Productname is Boots
Projected fill date for order is : 1/12

Date	Deliveries	Projected Stock
today	0	90
1/9	50	140
1/12	60	200

ORDER REQUEST SCREEN

DATA CONCEPTS AND CHARACTERISTICS

- **Data** is frequently considered to be one of an organization’s most valuable assets. Without it, businesses would find it impossible to perform some of their most basic activities. Data is also the heart of a database.
- Consequently, its concepts and characteristics need to be understood in order to successfully design, create, and use a database.
- Some of the most important ***concepts and characteristics*** are discussed in the following sections:
  - Data Hierarchy
  - Entities and Entity Relationships
  - Data Definition
  - The Data Dictionary
  - Data Integrity, Security, and Privacy
  - Data Organization

Data Hierarchy

- Data in a database has a definite hierarchy. ***At the lowest level,*** **characters** are entered into database fields (columns), which hold single pieces of data in the database, such as product names or quantities (refer again to Figure). ***At the next level*** are **records** (rows)—groups of related fields (such as all the fields for a particular product). ***At the next level*** are **tables**, which are made up of related records. ***At the top*** of the hierarchy is the **database**, which consists of a group of related tables.

Entities and Entity Relationships

- An ***entity*** is something (such as a person, object, or event) of importance to the business or organization. When an entity is something that a business or an organization wants to store data about in a database system, it typically becomes a database table. The ***characteristics*** of an entity are called ***attributes***.
- For instance, if a business collects data about customers, then Customer is an entity. Possible Customer attributes are first name, last name, phone number, address, and so forth. Attributes typically become fields in the entity's database table.
- A relationship describes an association between two or more entities. The three basic entity relationships (they are also called ***mapping concept***) are:
  - **One-to-One Entity Relationships**

    One-to-one (1:1) entity relationships exist when one entity is related to only one other entity of a particular type. For example, if a business has multiple store locations and each store has a single manager, the relationship between Store and Manager is a 1:1 relationship.

Entities and Entity Relationships

- **One-to-Many Entity Relationships**

  One-to-many (O:M) entity relationships are the most common and exist when one entity can be related to more than one other entity. For example, if a supplier supplies more than one product to the company, the relationship between Supplier and Products is a O:M.

- **Many-to-Many Entity Relationships**

  Many-to-many (M:M) entity relationships exist when one entity can be related to more than one other entity, and those entities can be related to multiple entities of the same type as the original entity. For example, a many-to-many relationship exists between customers and products: customers can purchase various products, and products can be purchased by many customers.

Data Definition

- ***Data definition*** involves describing the properties of the data that go into each database table, specifically the fields that make up the database. During the data definition process, the following are supplied for each field:
  - ***Name*** (must be unique within the table).
  - ***Data type*** (such as Text, Number, Currency, Yes/No, Hyperlink, or Date/Time); indicates the type of data that will be entered into the field.
  - ***Description*** (an optional description of the field).
  - ***Properties*** (such as the field size and format of the field, any allowable range or required format for the data that will be entered into the field, whether or not the field is required, and any initial value to appear in that field when a new record is added).

The Data Dictionary

- ***The data dictionary*** contains all data definitions for a database, including ***table structures*** (containing the names, types, and properties of each field in a table), ***security information*** (such as the password needed to view or edit a table), ***relationships between the tables*** in the database, and so on. Also included in the data dictionary is basic information about each table, such as its current number of records.
- The data dictionary does not contain any of the data located in the database tables, only data (called **metadata**) about the database tables.
- The data dictionary is used by the DBMS, as data is being entered into a table to ensure that the data does not violate any of its assigned properties.
- For example, the data dictionary would not allow you to enter a seven-character product number in a Product Number field that is defined as four characters long, it would not allow you to type text-based data into a field defined as a number field, and it would not allow you to leave a required field blank.
- In addition, without the proper password, the data dictionary would not allow you to view password-protected data.

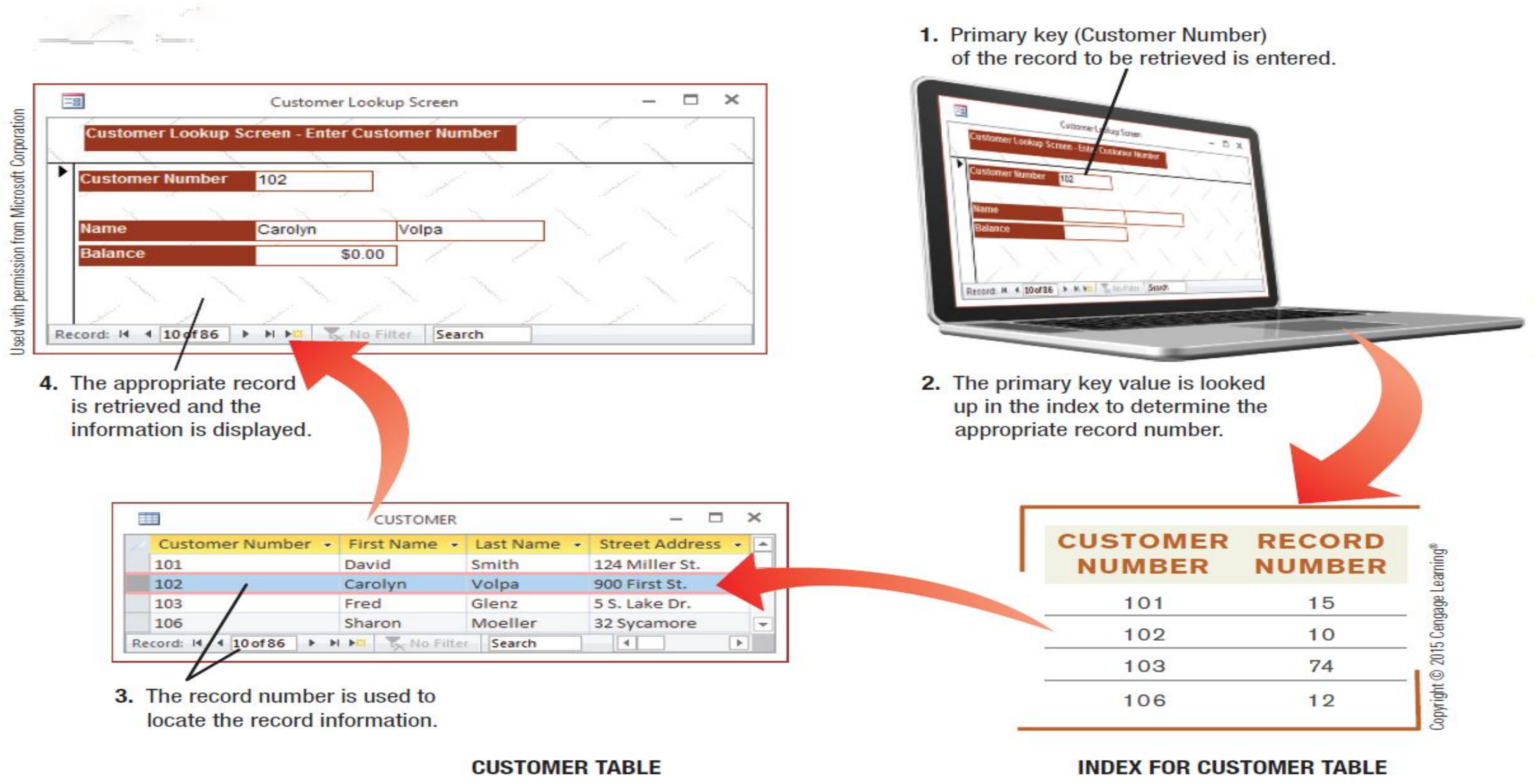
Data Integrity, Security, and Privacy

- Because data is so essential to organizations, data integrity, data security and data privacy are very important issues.
- ***Data integrity*** refers to the accuracy of data. The long-standing computer saying garbage in, garbage out is very appropriate for database systems. The quality of the information generated from a database is only as good as the accuracy of the data contained in the database.
- ***Data security*** refers to protecting data against destruction and misuse—both intentional and accidental. It involves both protecting against unauthorized access to and unauthorized use of the database, as well as preventing data loss.
- ***Data privacy*** refers to protecting the privacy of the data located in a database. It is a growing concern because of the vast amounts of personal data stored in databases today and the many data privacy breaches that are occurring.
- To protect the privacy of collected data, companies should first make sure that all the data they are collecting and storing is, in fact, necessary, and then they need to evaluate their data security measures to ensure that the data is adequately protected.

Data Organization

- Virtually all databases are organized in some manner to facilitate the retrieval of information. Arranging data for efficient retrieval is called ***data organization***.
- Most methods of data organization use a ***primary key*** to identify the locations of records so they can be retrieved when needed.
- The two most common types of data organization used today—***indexed*** and ***direct***.
  - **Indexed organization:** a method for organizing data on a storage medium or in a database that uses an index to specify the exact storage location. **Index -** a small table containing a primary key and the location of the record belonging to that key; used to locate records in a database.
  - **Direct organization:** a method of arranging data on a storage medium that uses ***hashing*** to specify the exact storage location.

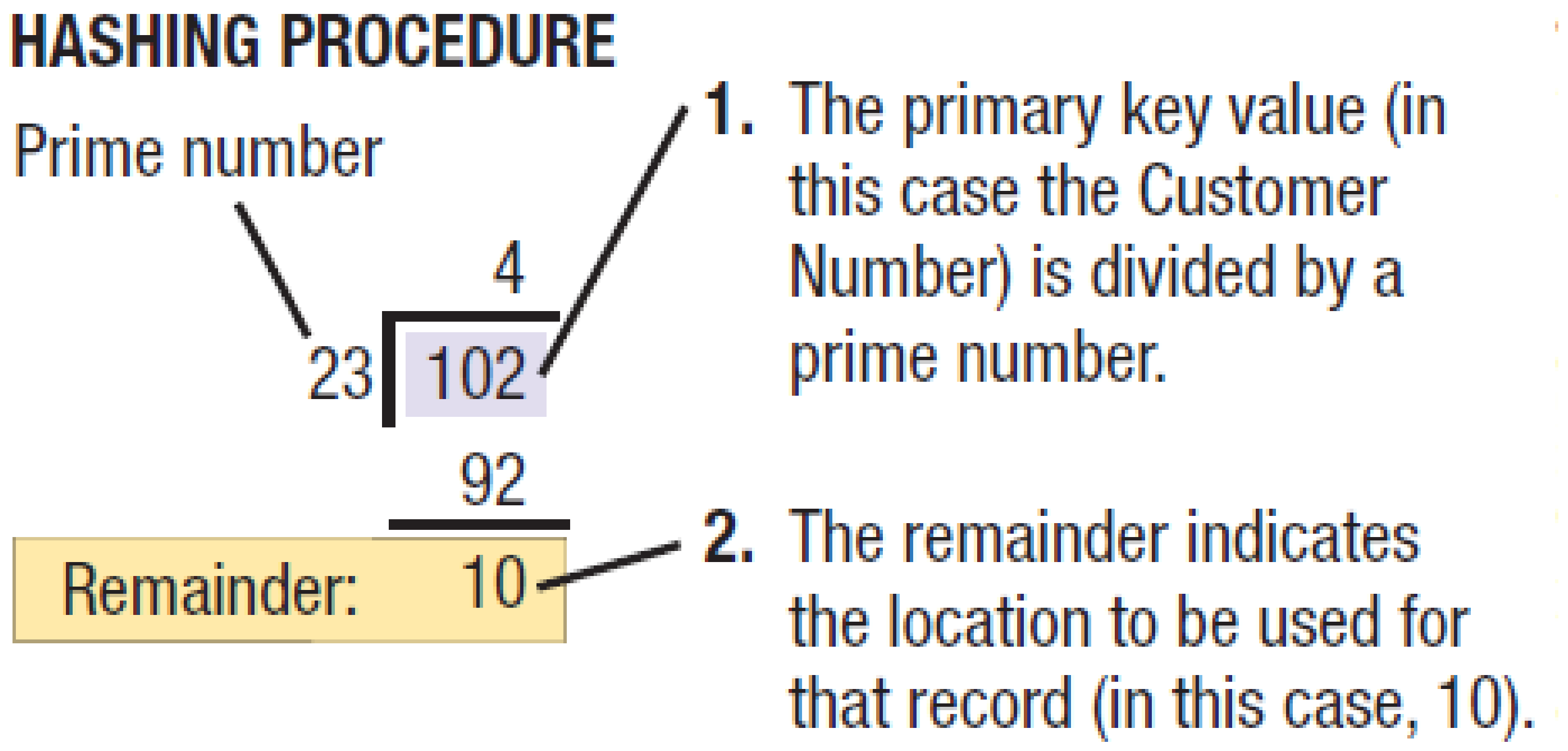
Fig: Indexed Organization


1. Primary key (Customer Number) of the record to be retrieved is entered.
2. The primary key value is looked up in the index to determine the appropriate record number.
3. The record number is used to locate the record information.
4. The appropriate record is retrieved and the information is displayed.
Customer Lookup Screen
Customer Lookup Screen - Enter Customer Number
Customer Number
102
Name
Carolyn
Volpa
Balance
$0.00
Record: 10 of 86
No Filter
Search
CUSTOMER
CUSTOMER TABLE
INDEX FOR CUSTOMER TABLE
Used with permission from Microsoft Corporation
Copyright © 2015 Cengage Learning®

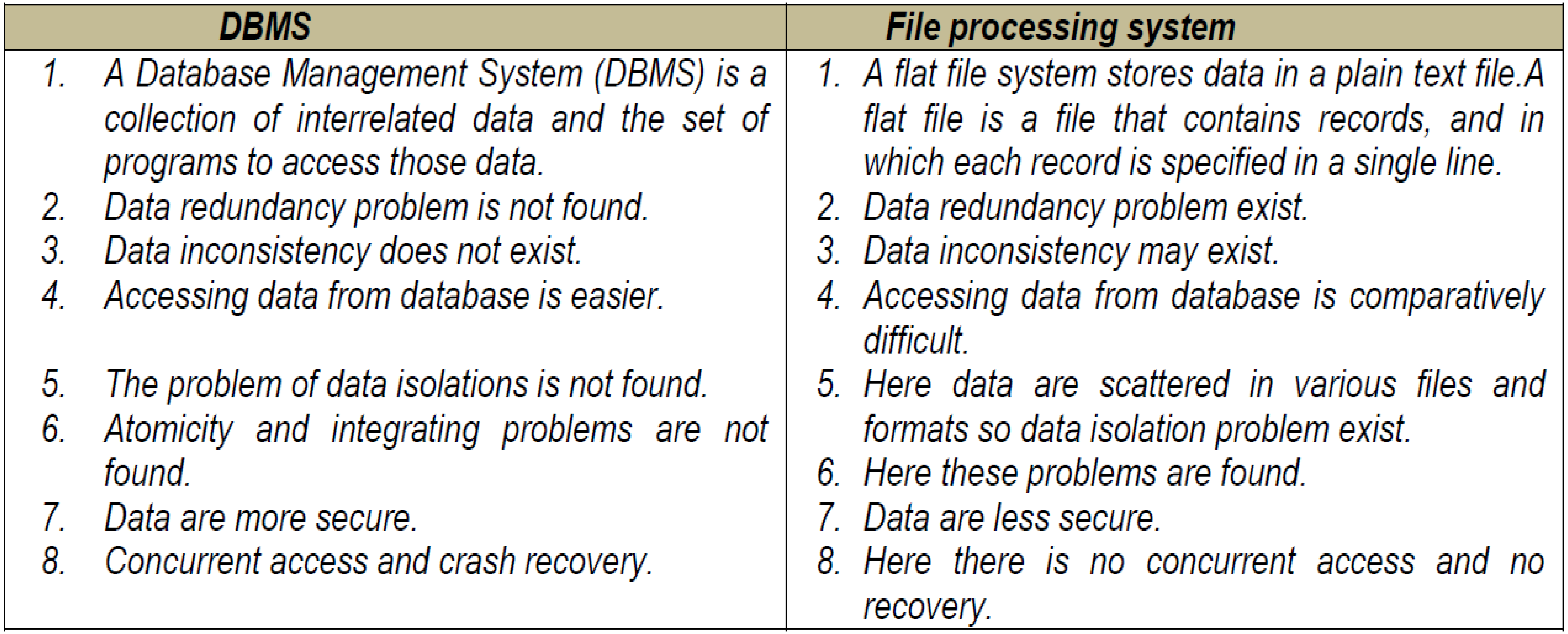
Customer Number	First Name	Last Name	Street Address
101	David	Smith	124 Miller St.
102	Carolyn	Volpa	900 First St.
103	Fred	Glenz	5 S. Lake Dr.
106	Sharon	Moeller	32 Sycamore

CUSTOMER NUMBER	RECORD NUMBER
101	15
102	10
103	74
106	12

Fig: Direct Organization


HASHING PROCEDURE
Prime number
4
23
102
92
Remainder: 10
1. The primary key value (in this case the Customer Number) is divided by a prime number.
2. The remainder indicates the location to be used for that record (in this case, 10).

Difference between DBMS and File System



DBMS	File processing system
1. A Database Management System (DBMS) is a collection of interrelated data and the set of programs to access those data.	1. A flat file system stores data in a plain text file.A flat file is a file that contains records, and in which each record is specified in a single line.
2. Data redundancy problem is not found.	2. Data redundancy problem exist.
3. Data inconsistency does not exist.	3. Data inconsistency may exist.
4. Accessing data from database is easier.	4. Accessing data from database is comparatively difficult.
5. The problem of data isolations is not found.	5. Here data are scattered in various files and formats so data isolation problem exist.
6. Atomicity and integrating problems are not found.	6. Here these problems are found.
7. Data are more secure.	7. Data are less secure.
8. Concurrent access and crash recovery.	8. Here there is no concurrent access and no recovery.

Database Management System

Introduction

- A ***database-management system*** (**DBMS**) is a collection of interrelated data and a set of programs to access those data. The collection of data, usually referred to as the database, contains information relevant to an enterprise.
- The ***primary goal of a DBMS*** is to provide a way to store and retrieve database information that is both convenient and efficient.
- Database systems are designed to manage large bodies of information. Management of data involves both defining structures for storage of information and providing mechanisms for the manipulation of information.
- In addition, the database system must ensure the safety of the information stored, despite system crashes or attempts at unauthorized access. If data are to be shared among several users, the system must avoid possible anomalous results.

Instances and Schema

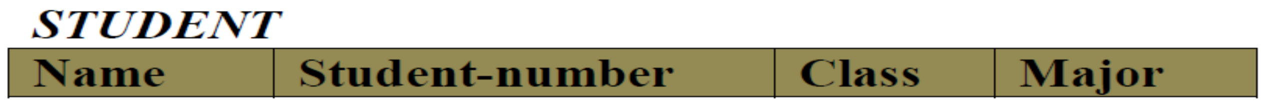
- **Instance (Database State):**
  - The collection of information stored in the database at a particular moment is called an ***instance of the database.*** It is the actual content of the database at a particular point in time.
  - The term ***instance*** is also applied to individual database components, e.g. record instance, table instance, entity instance.
- **Initial Database State**

  Refers to the database state when it is initially loaded into the system.
- **Valid State**

  A state that satisfies the structure and constraints of the database.

Instances and Schema

- **Schema:** The overall design of the database which is not expected to change frequently is called ***database schema***. Simply, the database schema is the logical structure of the database.
  - The concept of database schema and instances can be understood by analogy to a program written in a programming language.
  - A ***database schema*** corresponds to the ***variable declaration*** and the ***values of the variables in a program*** at a point in time correspond to an ***instance of a database***.
  - The database systems have several schemas and partitioned according to the level of abstraction such as physical and logical schema.



STUDENT

Name	Student-number	Class	Major

Fig: Schema diagram for Student

Note:

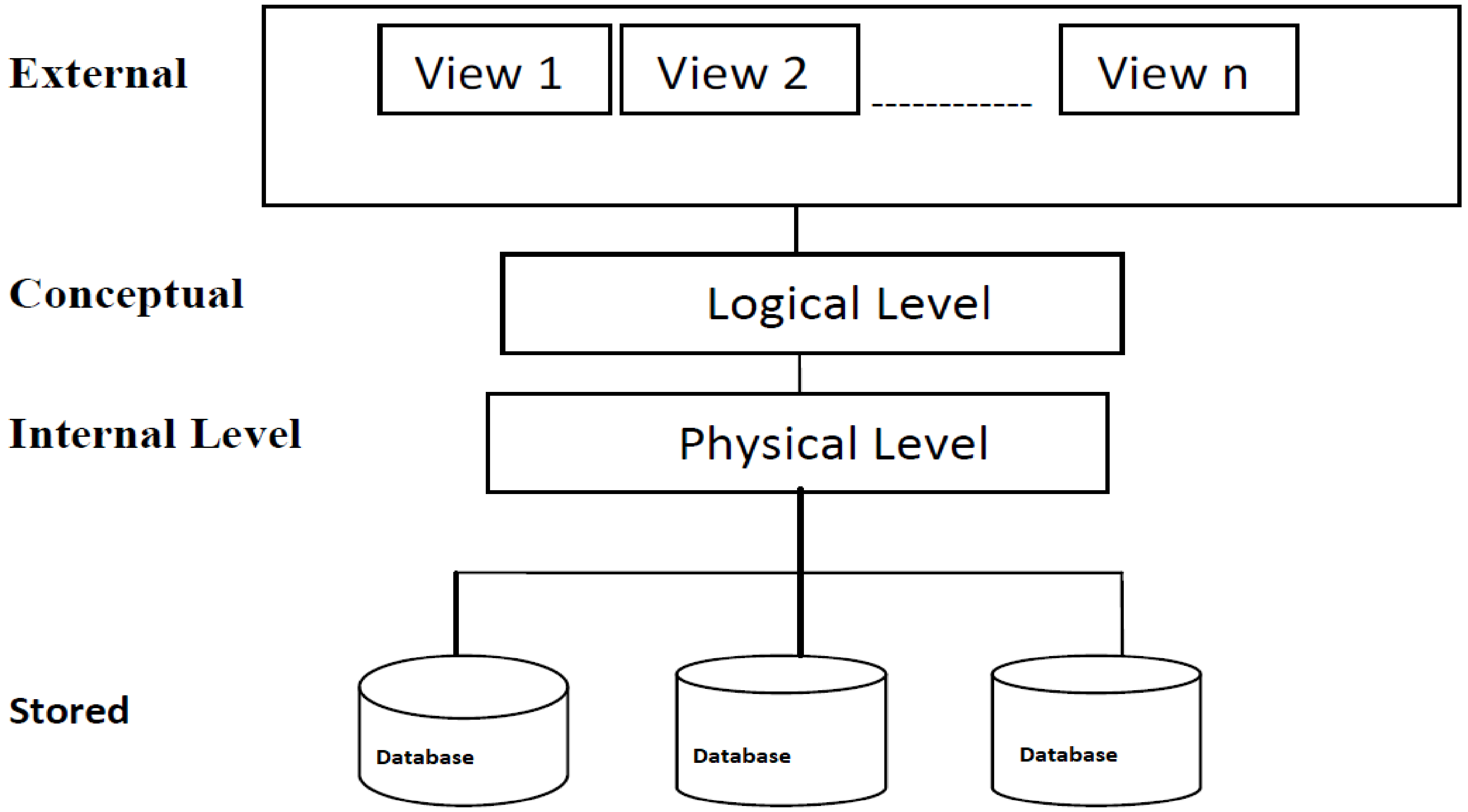
✓ The ***database schema*** changes very infrequently.

✓ The ***database state*** changes every time the database is updated.

3 Schema Architecture or Levels of Abstraction

- ***Data Abstraction:*** the system hides certain details of how the data are stored and maintained and such view is an abstract view.
- The Database System provides users with an abstract view of the data.
- The database designers use the complex data structure to represent the data in the database and developer hides the complexity from user from several level of abstraction such as ***physical level***, ***logical level***, and ***view level***. This process is called ***data abstraction***.

3 Schema Architecture or Levels of Abstraction



3 Schema Architecture or Levels of Abstraction

- **Physical level:**
  - It is the lowest level of abstractions describes how the data are actually stored.
  - The physical level describes complex low level data structure in details.
  - At this level records such as customer account can be described as a block of consecutive storage location (e.g. byte, word)
  - The database system hides many of the lowest level storage details from database programmer. Database administrator may be aware of certain details of the physical organization of the data.

3 Schema Architecture or Levels of Abstraction

- **Logical level:**
  - It is the next higher level of data abstraction which describes what data are stored in the database, and what relationships exist among those data.
  - At the logical level, each record is described by a type definition.
  - Programmers and database administrator works at this level of abstraction.

3 Schema Architecture or Levels of Abstraction

- **View level:**
  - It is the highest level of abstraction describes only a part of the database and hides some information to the user. This level describes the user interaction with database system.
  - At view level, computer users see a set of application programs that hide details of data types. Similarly at the view level several views of the database are defined and database user see only these views.
  - Views also provides the security mechanism to prevent users from accessing certain parts of the database (that is views can also hide information (such as an employee‘s salary) for security purposes.)

3 Schema Architecture or Levels of Abstraction

- **Example:** Let’s say we are storing customer information in a customer table.
  - At ***physical level*** these records can be described as blocks of storage (bytes, gigabytes, terabytes etc.) in memory. These details are often hidden from the programmers.
  - At the ***logical level*** these records can be described as fields and attributes along with their data types, their relationship among each other can be logically implemented. The programmers generally work at this level because they are aware of such things about database systems.
  - At ***view level***, user just interact with system with the help of GUI and enter the details at the screen, they are not aware of how the data is stored and what data is stored; such details are hidden from them.

Characteristics of Data in Database

- The data in a database should have the following features:
  - **Shared:** Data should be sharable among different users and applications.
  - **Persistence:** Data should exist permanently in the database. Changes in the database must not be lost because of any failure.
  - **Validity/Integrity/Correctness:** It should maintain the integrity so that there is always correct data in the database.
  - **Security:** Data should be protected from unauthorized access.
  - **Non-redundancy:** Data should not be repeated.
  - **Consistency:** A consistent state of the database satisfies all the constraints specified in the database. Data in a database is consistent if any changes in the database take the database from one consistent state to another.
  - **Independence:** The three levels in the schema should be independent of each other so that the changes in the schema in one level should not affect the other levels.

Database Models

Introduction

- ***Data model*** is a collection of tools for describing data, data relationships, data semantics and data constraints. The database model refers the way for ***organizing*** and ***structuring*** the data in the database.
- Traditionally, there are different database models which are used to design and develop the database of the organization.
  - Entity – Relationship (ER) Model
  - Object Oriented Model
  - Relational Model
  - Hierarchical model
  - Network Model

Entity - Relationship (ER) Model

- The ***E-R data models*** is based on a perception of real world that consist of a collection of basic objects called entities and relationship among these objects. In an E-R model a database can be modeled as a collection of entities, and relationship among entities.
- Overall logical structure of a database can be expressed graphically by E-R diagram. The basic components of this diagram are:
    - Rectangles (represent entity sets)
    - Ellipses (represent attributes)
    - Diamonds (represent relationship sets among entity sets)
    - Lines (link attributes to entity sets and entity sets to relationship sets)

Fig 1: Entity - Relationship Diagram

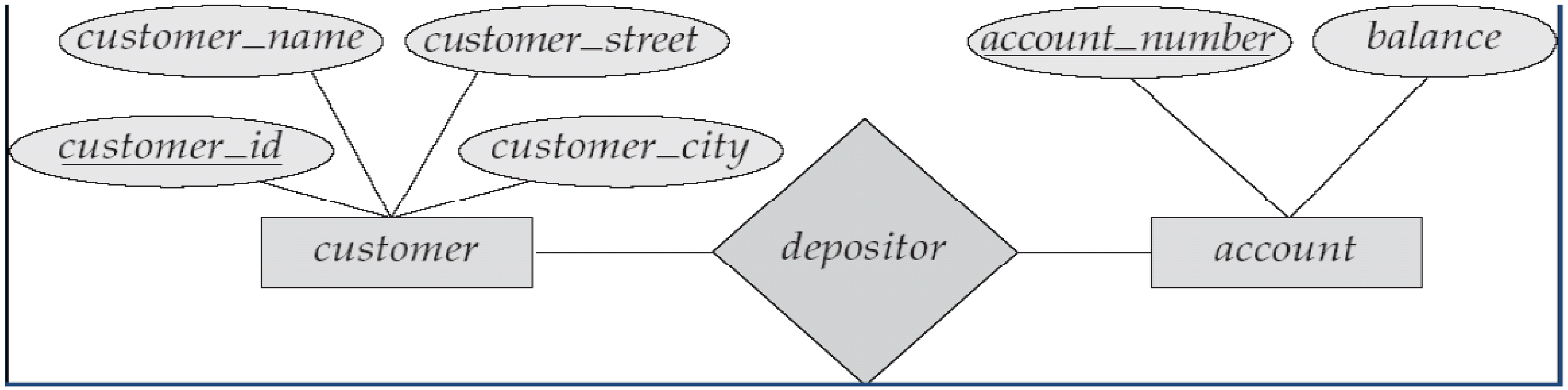
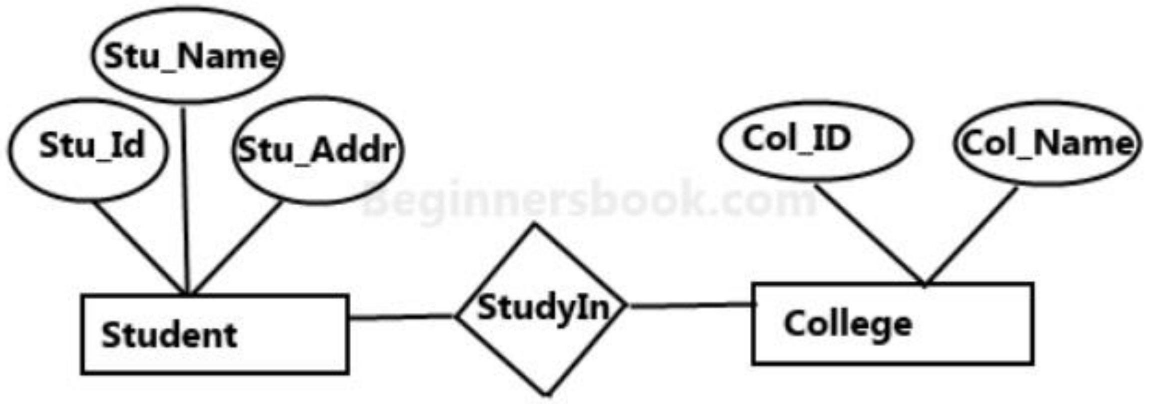


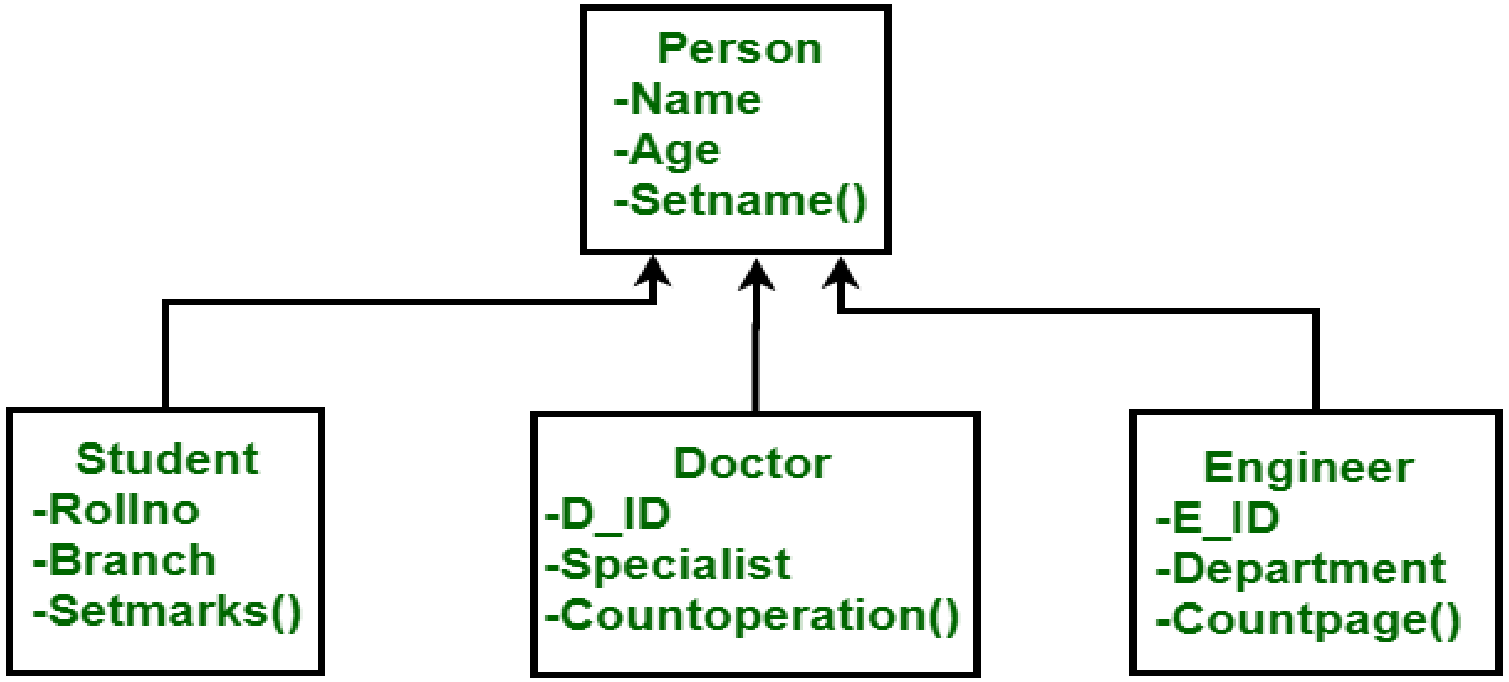
Fig 2: Entity - Relationship Diagram



Sample E-R Diagram

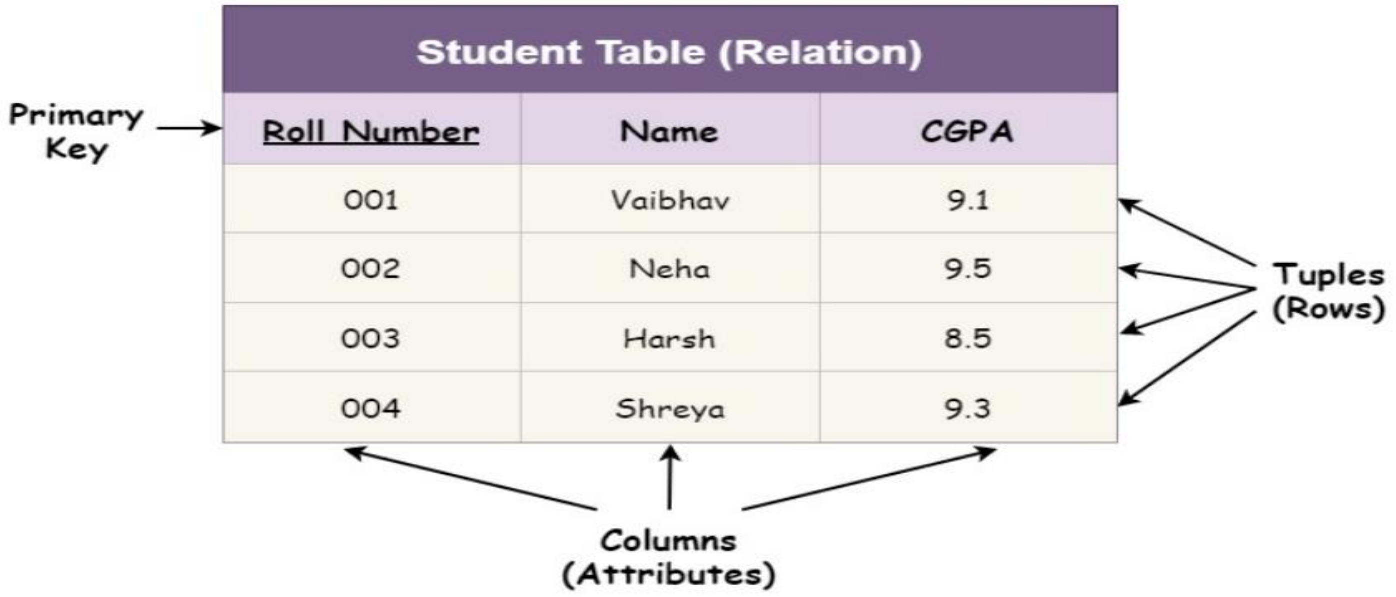
Object Oriented Model

- The ***object oriented data model*** is based on object-oriented programming paradigm. It is based on the concept of encapsulating the data and the functions that operate on those data in a single unit called an object.
- The internal parts of objects are not visible externally. Here one object communicates with other objects by sending message.



Relational Model

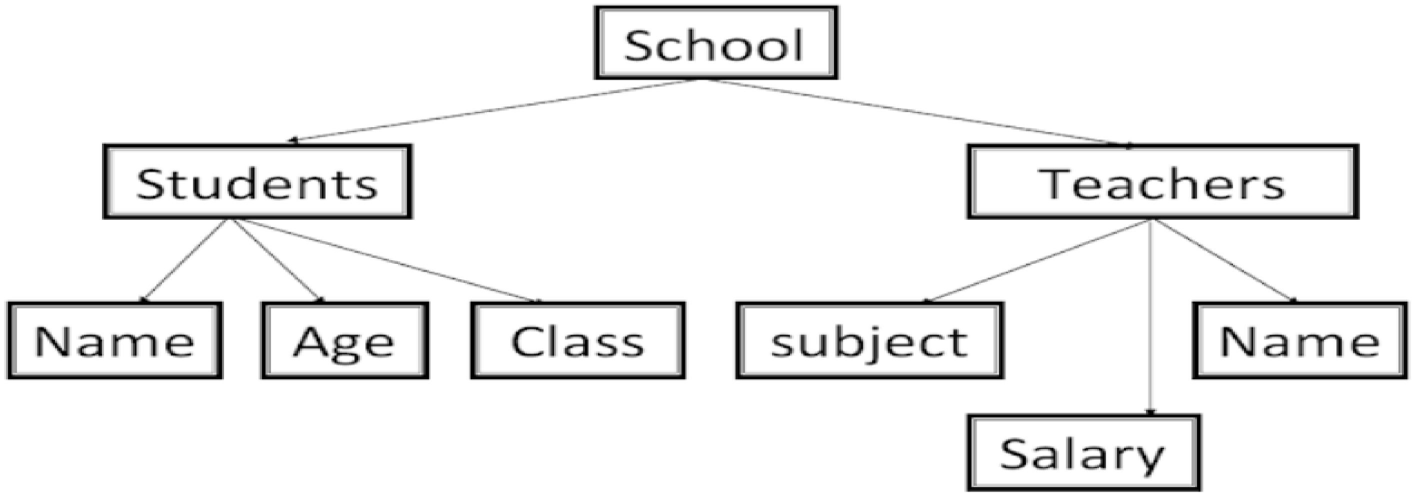
- It is the current favorite model. The ***relational model*** is a lower level model that uses a collection of tables to represent both data and relationships among those data.
- Each table has multiple columns, and each column has a unique name. Each table corresponds to an entity set or relationship set, and each row represents an instance of that entity set or relationship set. ***Structured Query Language*** (**SQL**) is used to manipulate data stored in tables.



Student Table (Relation)		
Roll Number	Name	CGPA
001	Vaibhav	9.1
002	Neha	9.5
003	Harsh	8.5
004	Shreya	9.3

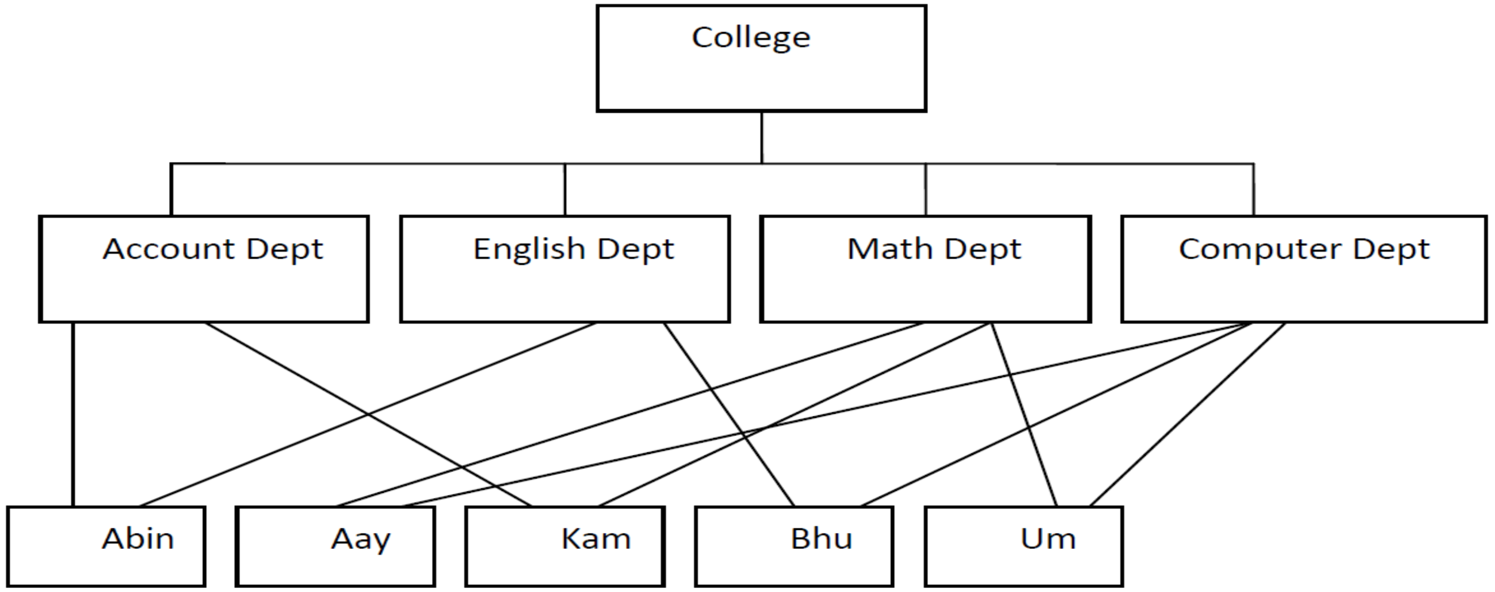
Hierarchical Model

- In a ***hierarchical data model***, the data elements are linked in the form of an inverted tree structure with the root at the top and the branches formed below.
- Below the single root data element are subordinate elements, each of which, in turn, has one or more other elements. There is a parent child relationship among the data elements of a hierarchical database. There may be many child elements under each parent element, but there can be only one parent element for any child element. ***The branches in the tree are not connected.***



Network Model

- A ***network data model*** is an extension of the hierarchical database structure. In this model also, the data elements of a database are organized in the form of parent-child relationships and all types of relationships among the data elements must be determined when the database is first designed.
- In a network database, a child data element ***can have more than one parent element*** or ***no parent at all***. Moreover, in this type of database, the database management system permits the extraction of the needed information from any data element in the database structure, instead of starting from the root data element.



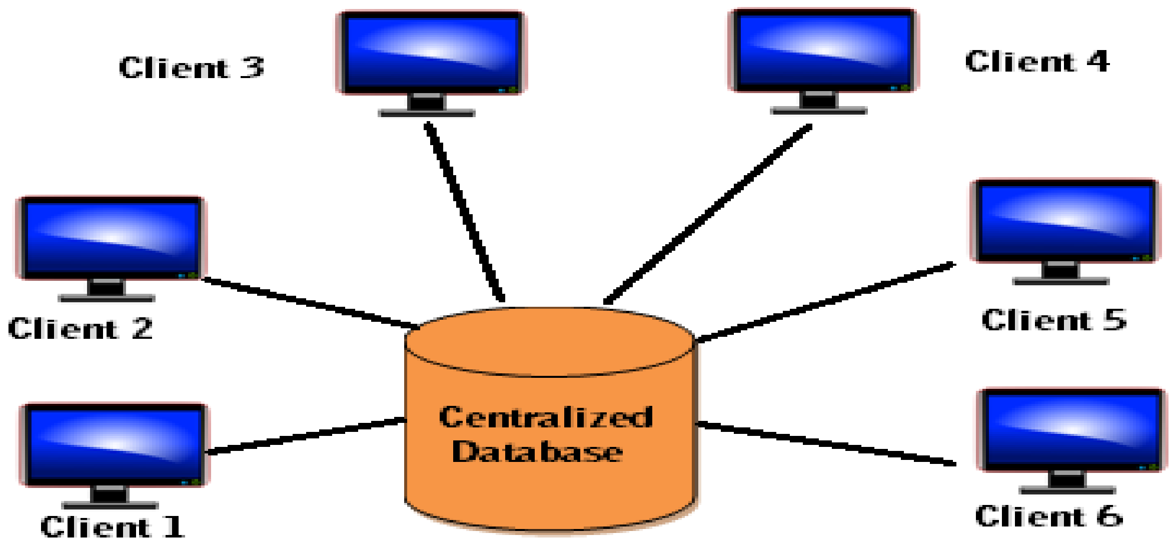
Database System Architectures

Introduction

- The computers have evolved from big mainframe computers to small desktop personal computers. The advances in computer and its architectures have also resulted in the advances in database and its architectures.
- A ***Database Architecture*** is a representation of ***DBMS design***. It helps to ***design***, ***develop***, ***implement***, and ***maintain*** the database management system.
- A ***DBMS architecture*** allows dividing the database system into individual components that can be independently modified, changed, replaced, and altered. It also helps to understand the components of a database.
- A Database stores critical information and helps access data quickly and securely. Therefore, selecting the correct Architecture of DBMS helps in easy and efficient data management.
- Here, we discuss the architecture of databases of three kinds:
    - Centralized Database Architecture
    - Client/Server Database Architecture
    - Distributed Database Architecture

Centralized Database Architecture

- A ***centralized database*** is a database that is located, stored, and maintained in a ***single location.*** This location is most often a central computer or database system, for example a desktop or server CPU, or a mainframe computer.
- It is maintained and modified from that location only and usually accessed using an internet connection such as a LAN or WAN. The centralized database is used by organizations such as colleges, companies, banks etc.



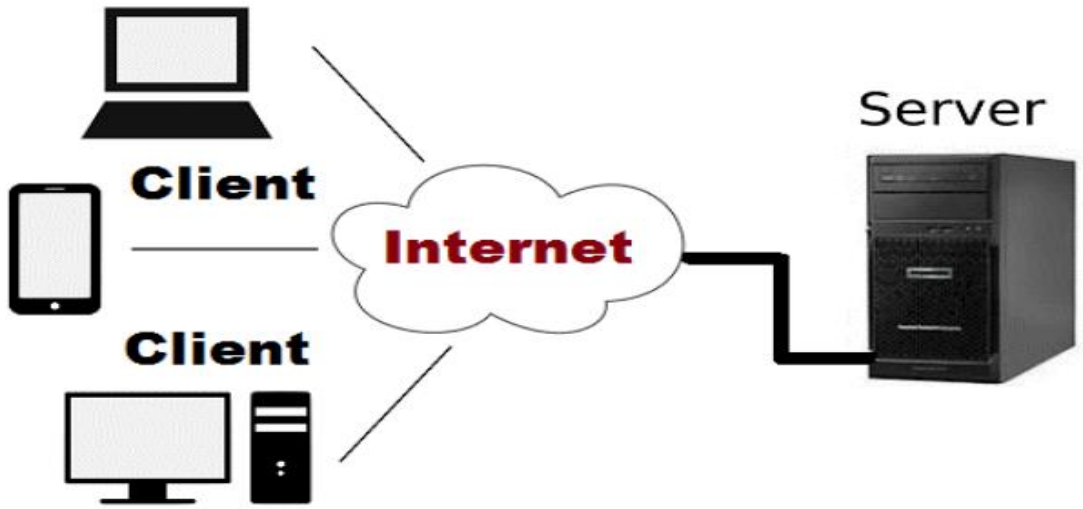
Centralized Database Architecture (Contd.)

- **Advantages:**
  - Since all data is stored at a single location only thus it is easier to access and co-ordinate data.
  - The centralized database has very minimal data redundancy since all data is stored at a single place.
  - It is cheaper in comparison to all other databases available.

- **Disadvantages:**
  - The data traffic in case of centralized database is more.
  - If any kind of system failure occurs at centralized system then entire data will be destroyed.

Client/Server Database Architecture

- The ***client/server architecture*** is based on the hardware and software components that interact to form a system.
- The system includes three main components:
  - **Client:** The client is any computer process that requests service from the server.
  - **Server:** The server is any computer process providing services to the clients.
  - **Communication Middleware:** The communication middleware is any computer process through which clients and servers communicate and is also known as ***communication layer***.



Client/Server Database Architecture (Contd.)

- **Advantages:**
  - All the data and resources are controlled by server in this way all data and resources are very consistent.
  - You can easily increase the number of client in this architecture at any time. This all increases the scalability of the network.
  - This is very easy to maintain you can easily repair, replace or add clients in this network. The independence of the changes also known as encapsulation.
  - This network is very easy to use and it is not complicated.
- **Disadvantages:**
  - Traffic is a big problem in this network.
  - When you add large numbers of the client with server this network will be more complicated.
  - When the server goes down all the clients are not able to send their request. The whole work will be stopped.
  - The hardware and software are very expensive.
  - The client does not have resources for each resource they need to request the server because of all resources exit on server.

Client/Server Database Architecture (Contd.)

- There are basically three-types of client/server architectures:
  - **One-tier architecture:**

    In this architecture, it keeps all of the elements of an application, including the interface, middleware and back-end data, in one place.
  - **Two-tier Architecture:**

    In this architecture, the ***user interface*** and ***application programs*** are placed on the ***client side*** and ***database system*** on the ***server side***. The application programs that resides at the client side invoke the DBMS at the server side.
  - **Three-tier Architecture:**

    This architecture adds ***application server*** between the ***client*** and ***database server***. The client communicates with the application server, which in turn communicates with the database server. The application server stores the business rule (procedures and constraints) used for accessing data from database server.

Fig: One Tier Client Server Architecture

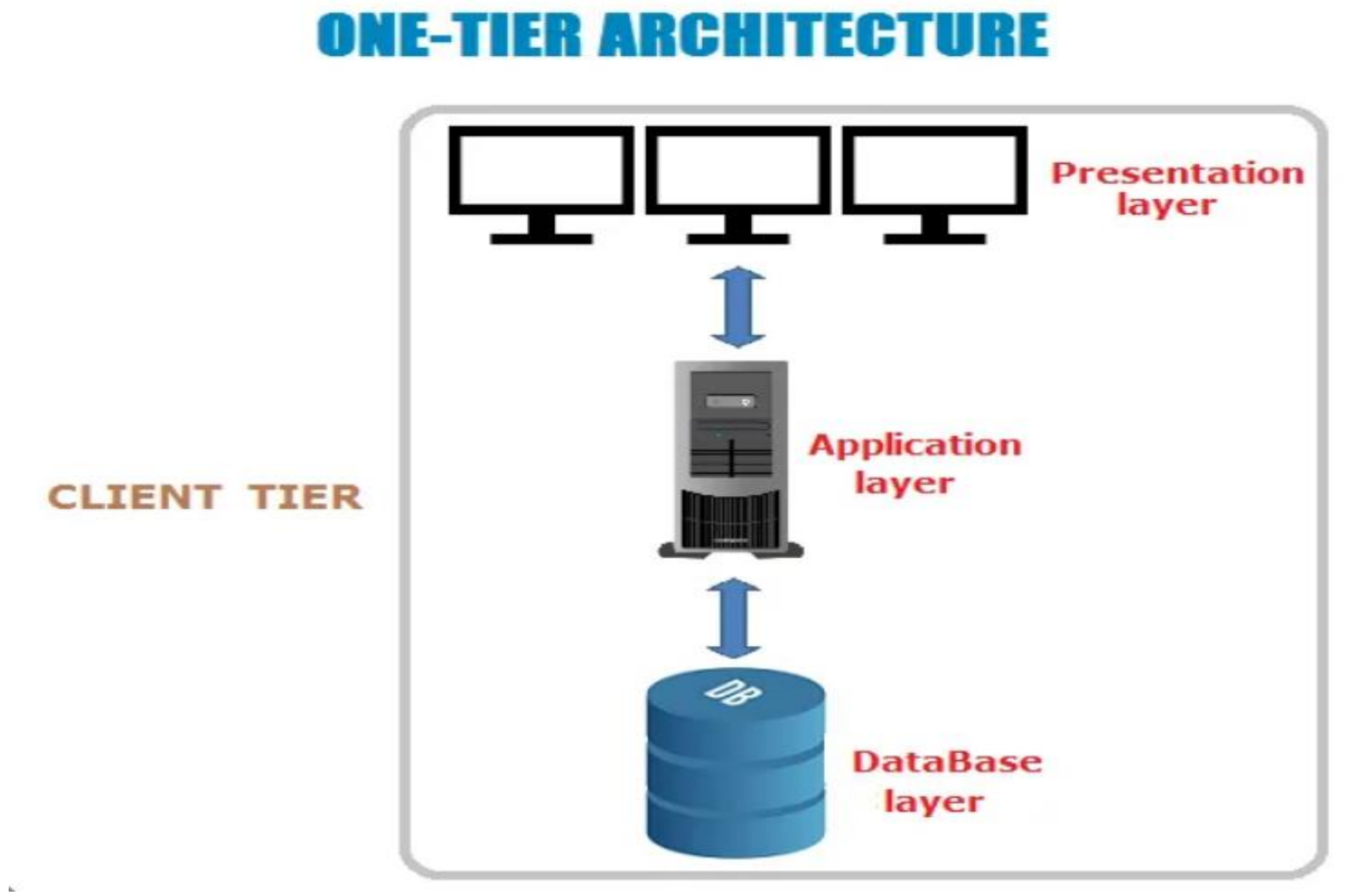


Fig: Two Tier Client Server Architecture

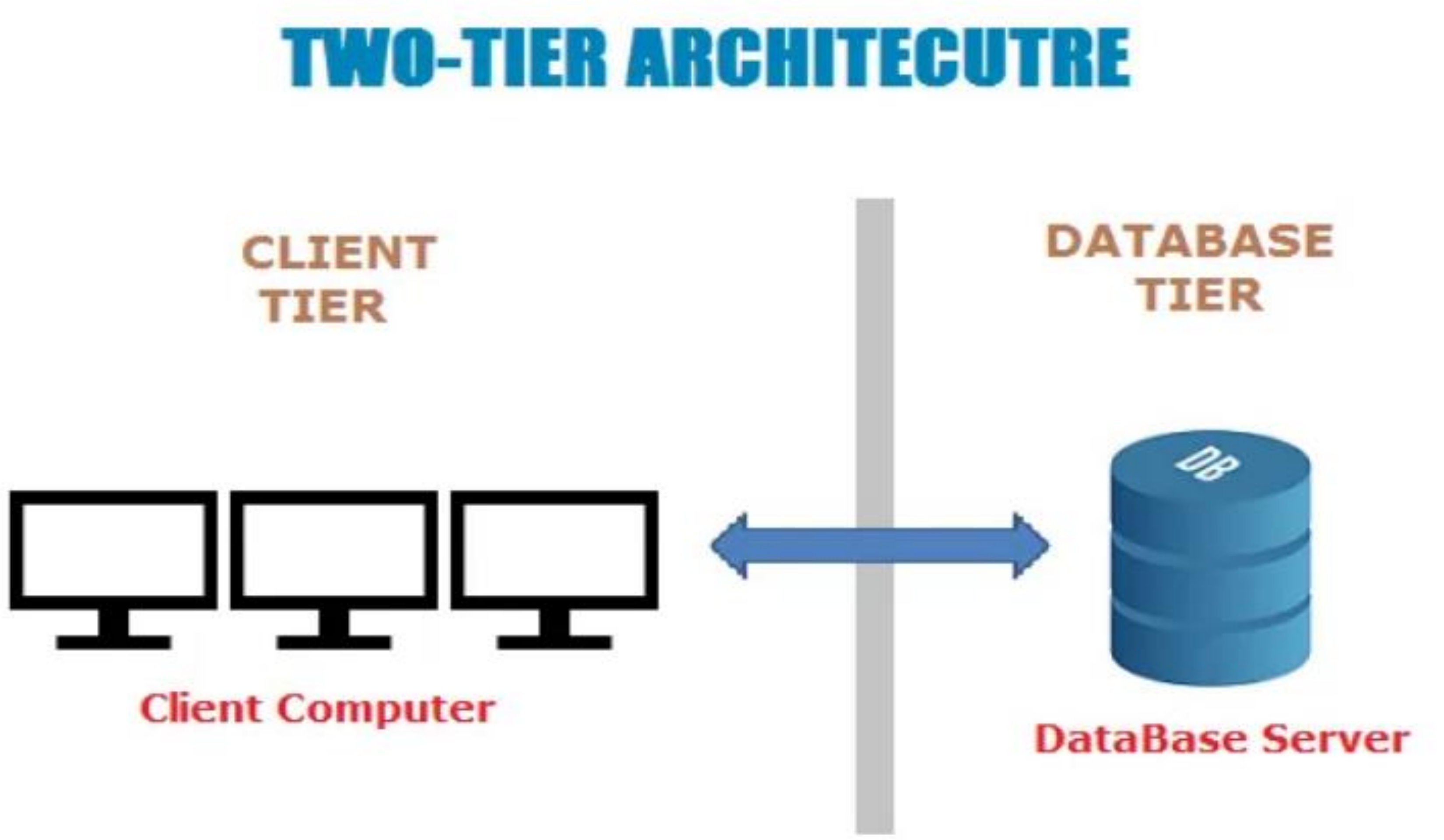
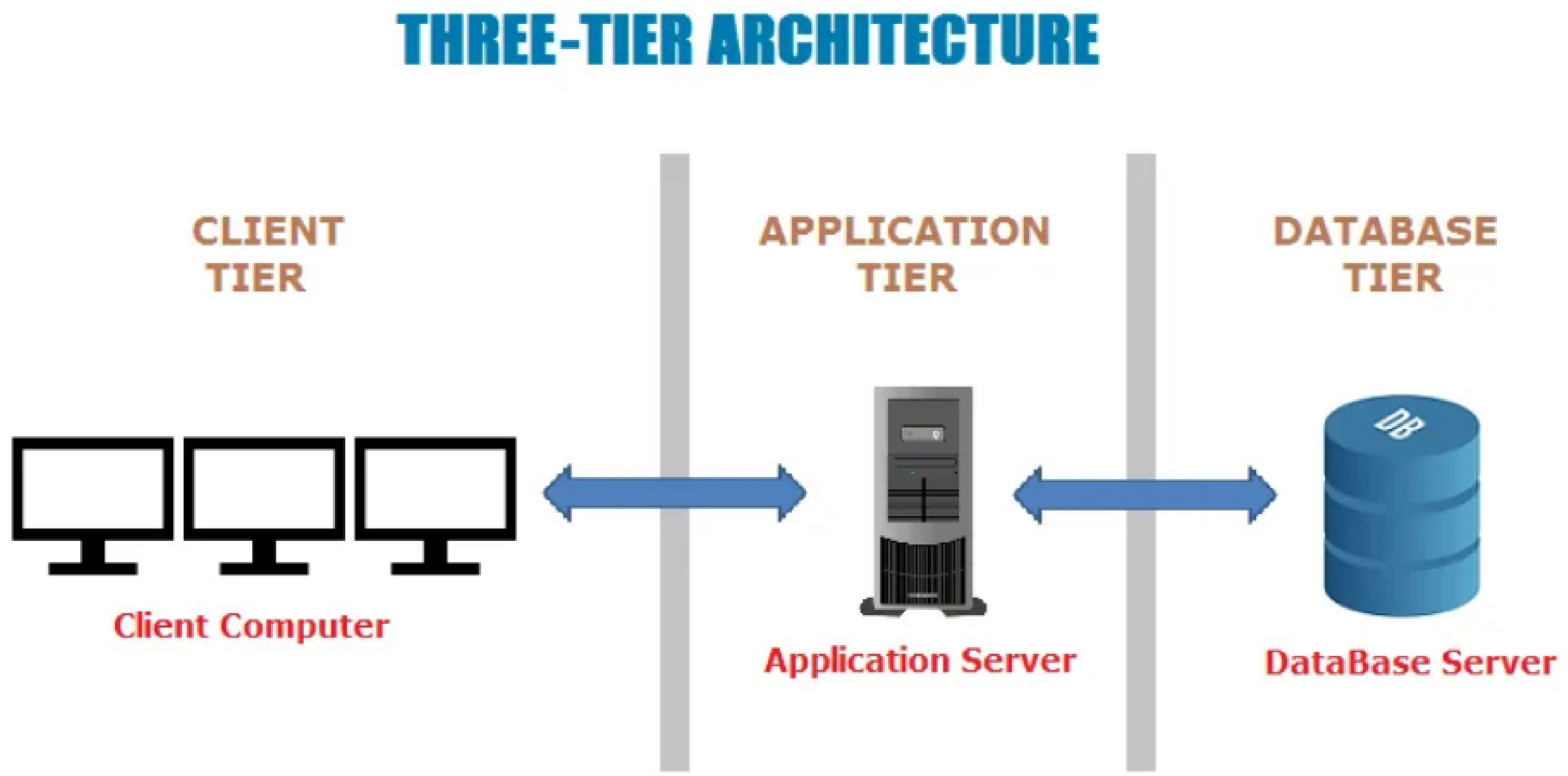
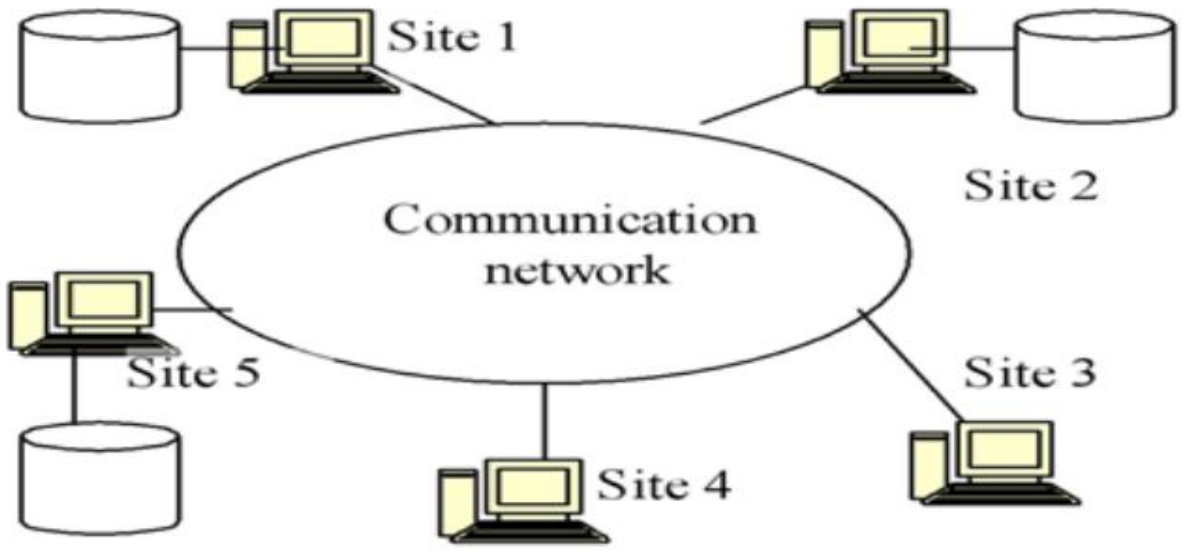


Fig: Three Tier Client Server Architecture



Distributed Database Architecture

- A ***distributed database*** is a set of logically interrelated database that are stored on computers at ***several geographically different sites*** and are linked by means of a computer network. These interrelated database work together to perform certain specific tasks.
- Distributed computer system work by splitting a large task into a number of smaller ones that can then be solved in a coordinated fashion. Each processing element can be managed independently and can be developed its own applications.



Distributed Database Architecture (Contd.)

- **Advantages:**
  - This database can be easily expanded as data is already spread across different physical locations.
  - The distributed database can easily be accessed from different networks.
  - This database is more secure in comparison to centralized database.

- **Disadvantages:**
  - This database is very costly and it is difficult to maintain because of its complexity.
  - Designing a distributed database is more complex compared to a centralized database.
  - In a vast Distributed database system, maintaining data consistency is important. All changes made to data at one site must be reflected on all the sites.

Database Applications

- Database system are widely used in different areas because of their numerous advantages. Some of the most common database applications are:
  - **Banking:**

    To store information about customers, their account number, balance etc.
  - **Airlines:**

    For reservations and schedule information.
  - **Telecommunication:**

    To keep records of customers, call made, balance left, generating monthly bills etc.
  - **Universities:**

    To keep records of students, courses, marks of students etc.
  - **Sales:**

    To keep information of customers, products list, purchase information etc.
  - **Manufacturing:**

    To store orders, tracking production of items etc.
  - **Human Resources:**

    To keep records of employee, their salary, bonus etc.

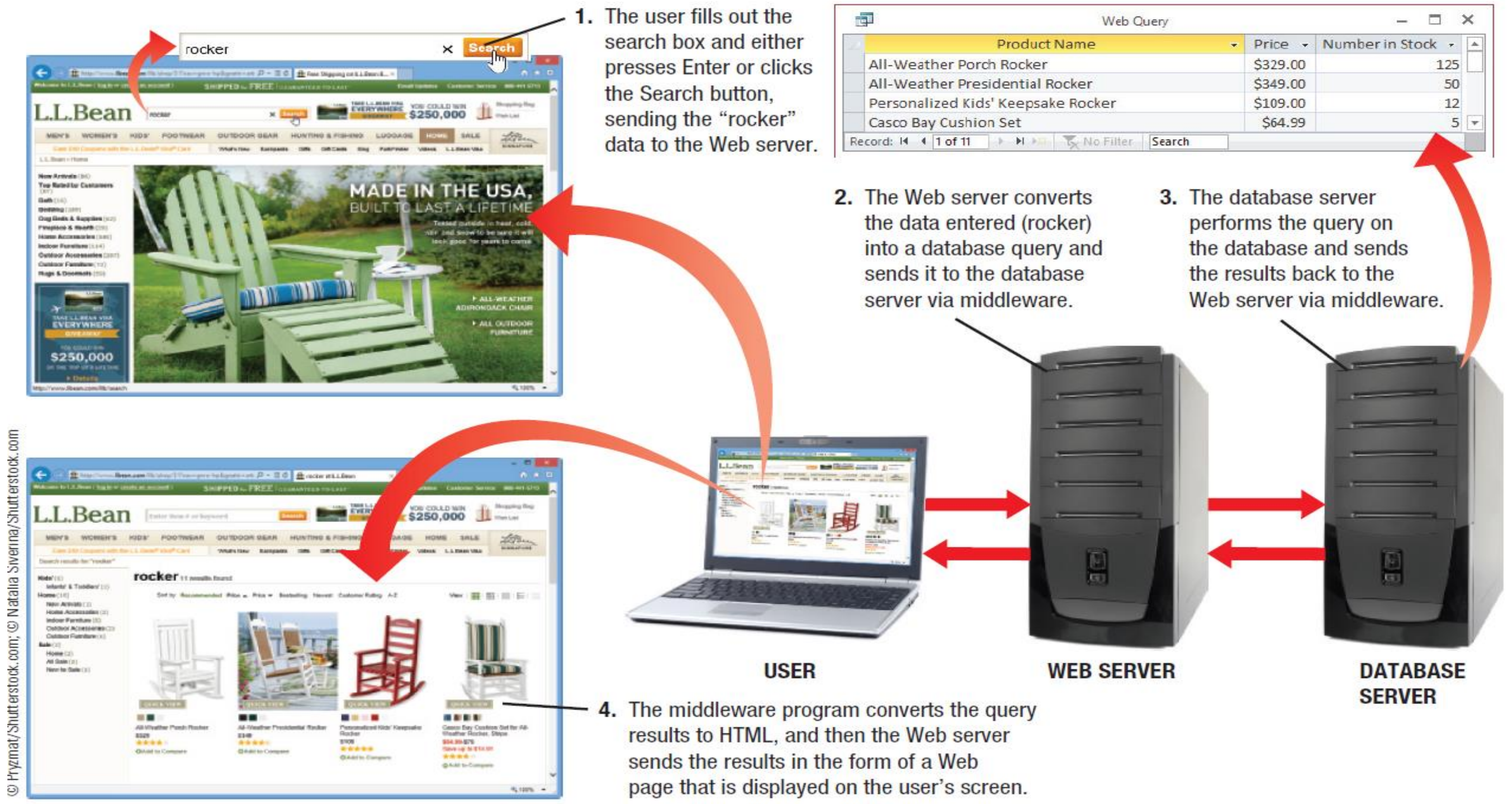
CLOUD DATABASES

- Databases are extremely common on the Web. Virtually all companies that offer product information, online ordering, research resources, or similar activities via a Web site use a database.
- For instance, one of the largest cloud databases belongs to Amazon, which stores data about its customers and their orders, products for sale, and customer reviews of products.
- In addition, it stores the actual text of many books so they can be searched and viewed online. Cloud databases are also increasingly used to store user generated content (that individuals upload to be shared with others), such as content uploaded to Flickr, YouTube, Facebook, Pinterest, and other social networking sites.

CLOUD DATABASES (Contd.)

- Cloud databases are typically built using the infrastructure of a cloud provider (such as Windows Azure, Amazon SimpleDB, or Google Cloud SQL). Also known as Database-as-a-Service (DBaaS), cloud databases enable businesses to create an easily scalable database with less in-house hardware and maintenance requirements, and to only pay for the storage and traffic they use.
- Cloud databases can also be set up for mobile access when appropriate. However, businesses utilizing databases hosted in the cloud need to ensure that those databases are adequately protected against security and privacy breaches, as well as have a recovery plan in case of a disaster or the bankruptcy of the cloud provider.

How Cloud Databases Work – Cloud in Action



1. The user fills out the search box and either presses Enter or clicks the Search button, sending the "rocker" data to the Web server.

Web Query

Product Name	Price	Number in Stock
All-Weather Porch Rocker	$329.00	125
All-Weather Presidential Rocker	$349.00	50
Personalized Kids' Keepsake Rocker	$109.00	12
Casco Bay Cushion Set	$64.99	5

Record: 1 of 11 No Filter Search

2. The Web server converts the data entered (rocker) into a database query and sends it to the database server via middleware.

3. The database server performs the query on the database and sends the results back to the Web server via middleware.

4. The middleware program converts the query results to HTML, and then the Web server sends the results in the form of a Web page that is displayed on the user's screen.

© Pryzmat/Shutterstock.com; © Natalia Siverina/Shutterstock.com

THANK YOU!