Introduction to DBMS

Module No. 1

Introduction and motivation, Data independence, Three schema architecture, Centralized and Client/Server architectures, Database components, Database users, Entity Types, Entity Sets, Attributes, Entity Type (Strong and Weak), Relationship Types, Relationship Sets, Roles, Structural Constraints, ER diagram construction.

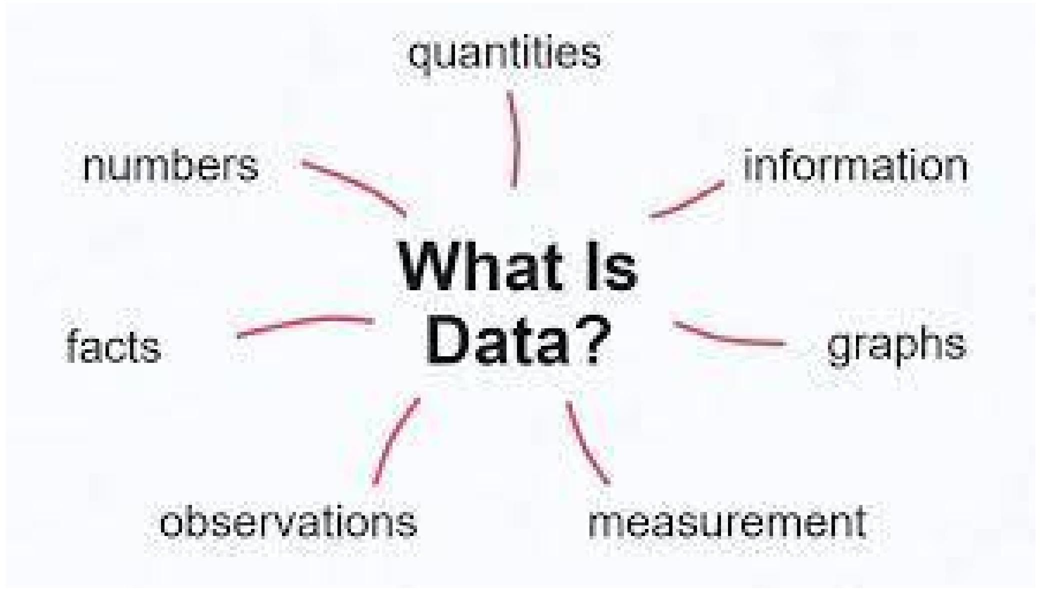
What is data, database, DBMS?

Data

- Known facts that can be recorded and have an implicit meaning; raw.

(or)

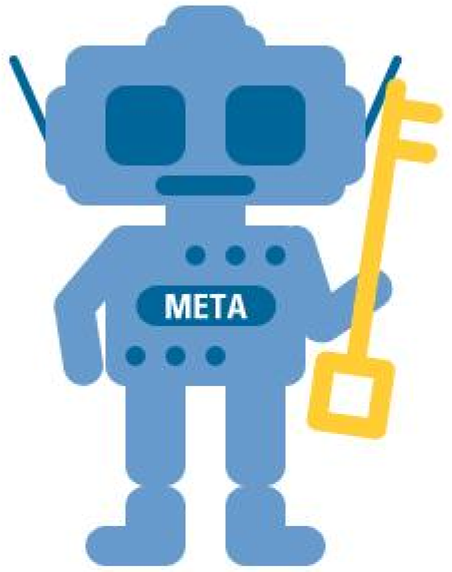
- Data is a collection of facts that is unorganized but can be made organized into useful information.



- Data that have been processed in such a way as to increase the knowledge of the person who uses the data is known as information.

What is data, database, DBMS?

Metadata



- Data that describes the properties of other data is known as metadata. Metadata keeps the information of other data i.e., it keeps the information of how and where the other data is stored.
- It describes the properties of data but does not include that data. In short, metadata is data about data.

What is data, database, DBMS?

Database

- A database is a collection of data of some given enterprise, that can be processed through one or more application programs by multiple users.

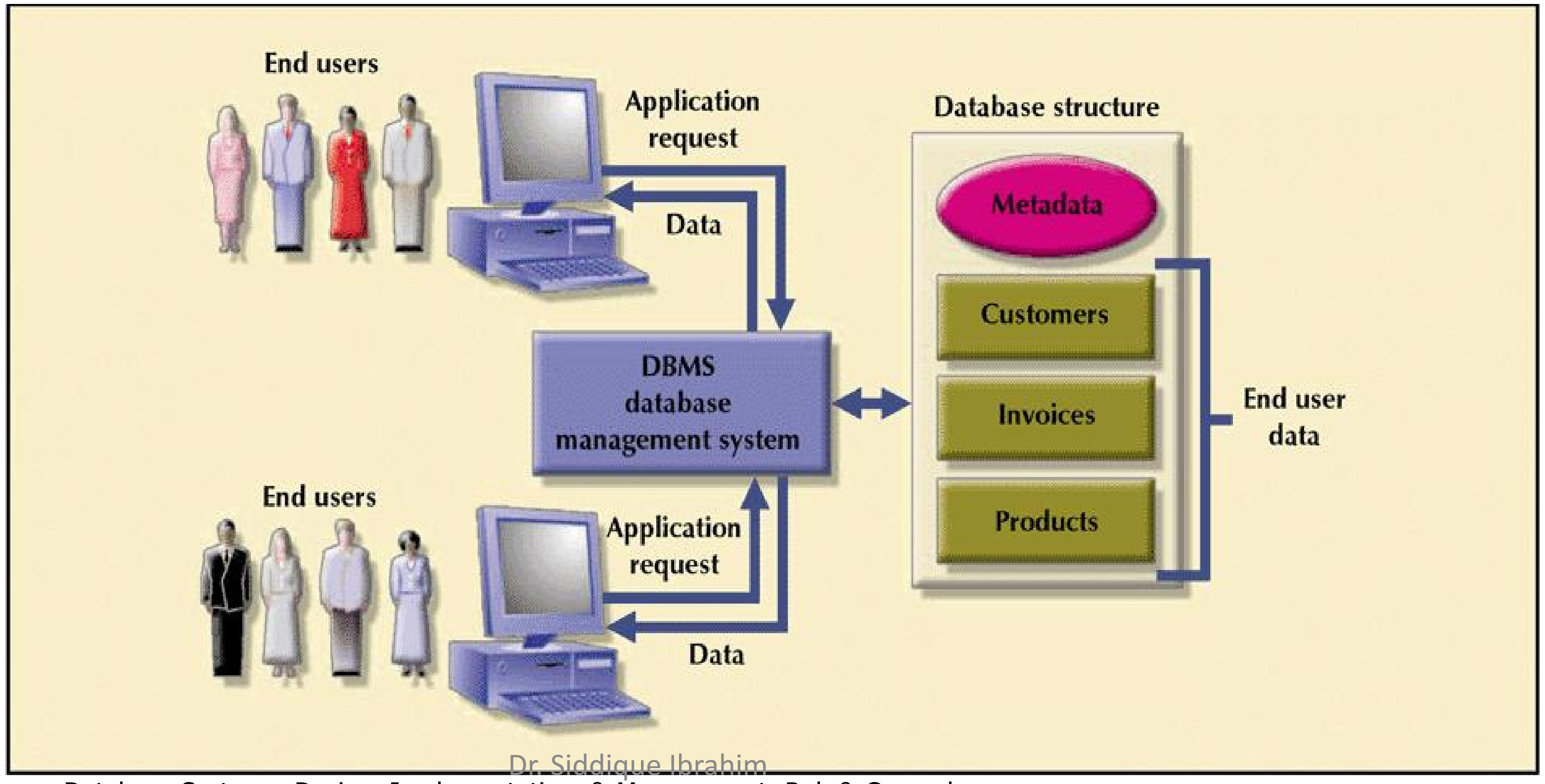
- A highly organized, interrelated, and structured set of data about a particular enterprise.

- Controlled by a Database Management System (DBMS).

What is data, database, DBMS?

Data Base Management System

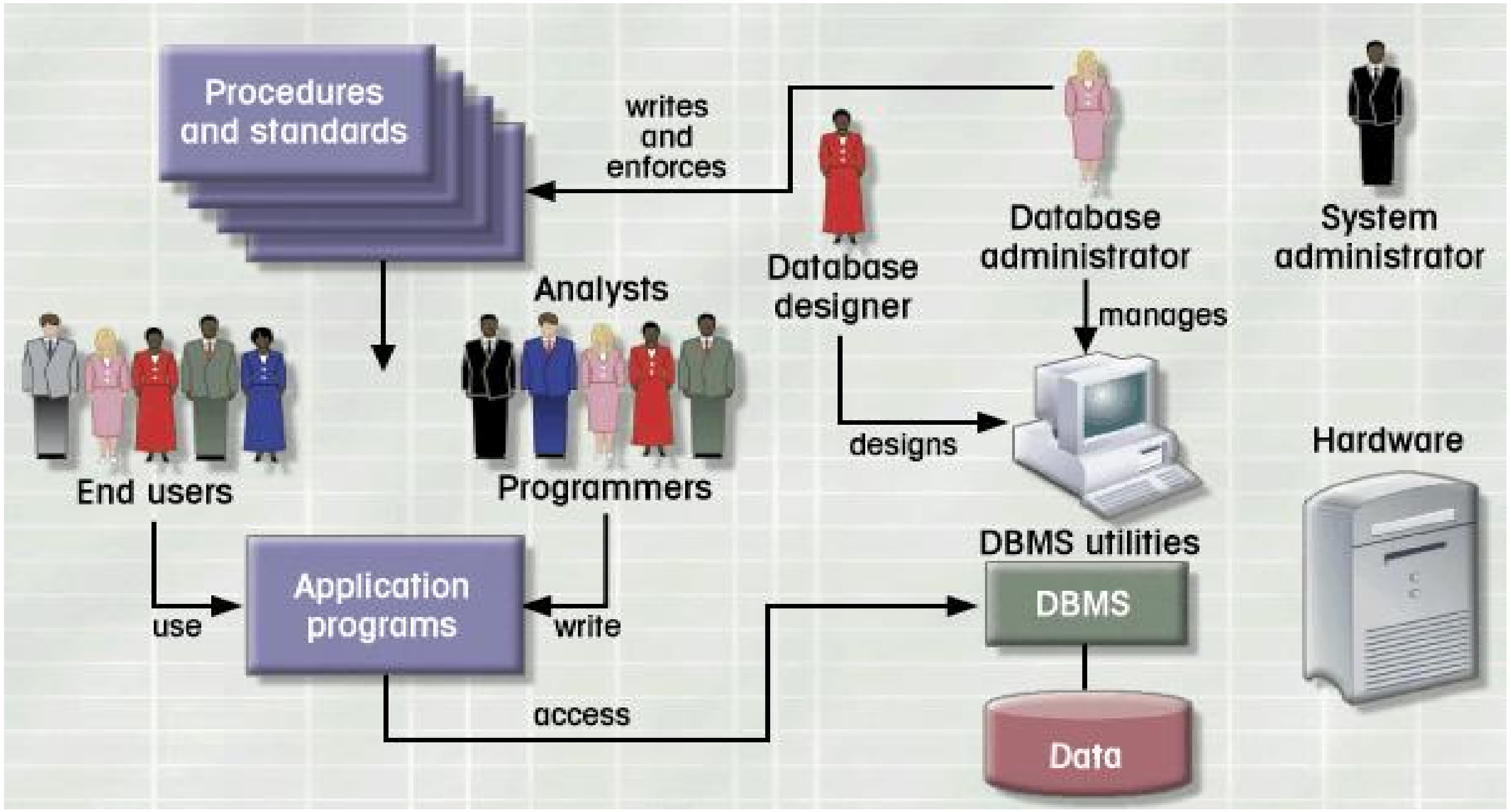
A database management system, or DBMS, is software designed to assist in maintaining and utilizing large collections of data. The need for such systems, as well as their use, is growing rapidly. The alternative to using a DBMS is to store the data in files and write application-specific code to manage it.



Database Systems: Design, Implementation, & Management: Rob & Coronel

What is data, database, DBMS?

Database System Environment



Database Systems: Design, Implementation, & Management: Rob & Coronel

- Hardware
- Software
  - OS
  - DBMS
  - Applications
- People
- Procedures
- Data

Types of Databases and Database Applications

- Numeric and Textual Databases
- Multimedia Databases
- Geographic Information Systems (GIS)
- Data Warehouses
- Real-time and Active Databases

Database system applications- Examples

- Banking: For customer information, accounts, loans and banking transactions.
- Airlines: For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner.
- Universities: For student information, course registrations and grades.
- Credit card transactions: For purchases on credit cards and generation of monthly statements.
- Telecommunication: For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards and storing information about the communication networks.
- Sales: For customer, product and purchase information.

Typical DBMS Functionality

- Defines a database : in terms of data types, structures and constraints
- Construct or Load the Database on a secondary storage medium.
- Manipulating the database: querying, generating reports, insertions, deletions and modifications to its content
- Concurrent Processing and Sharing by a set of users and programs – yet, keeping all data valid and consistent.

Typical DBMS Functionality

Other features:

- Protection or Security measures to prevent unauthorized access

- "Active" processing to take internal actions on data

- Presentation and Visualization of data.

- Maintenance of the database and associated programs over the lifetime of the database application.

Example of a Database
(with a Conceptual Data Model)

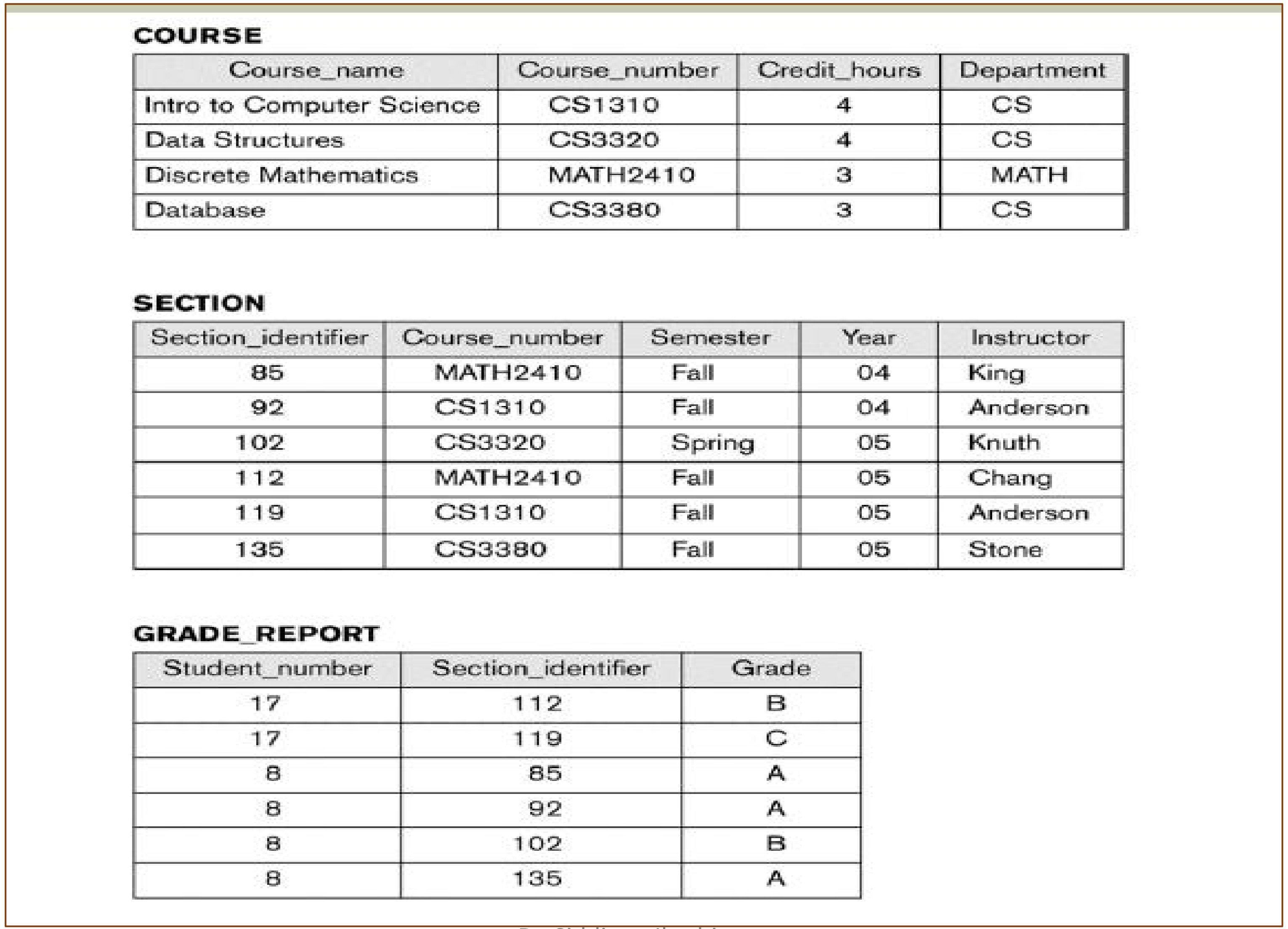
Part of a UNIVERSITY environment.

- STUDENTs
- COURSEs
- SECTIONs (of COURSEs)
- (academic) DEPARTMENTs
- INSTRUCTORs

Some mini-world relationships

- SECTIONs *are of specific* COURSEs
- STUDENTs *take* SECTIONs
- COURSEs *have prerequisite* COURSEs
- INSTRUCTORs *teach* SECTIONs
- *Courses are offered by* DEPARTMENTs
- STUDENTs *major in* DEPARTMENTs

Example of a simple database



COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

Main Characteristics of the Database Approach

- **Self-describing nature of a database system:**
  - A DBMS catalog stores the description of a particular database (e.g. data structures, types, and constraints)
  - The description is called meta-data.
  - This allows the DBMS software to work with different database applications.
- **Insulation between programs and data:**
  - Called program-data independence.
  - Allows changing data structures and storage organization without having to change the DBMS access programs.

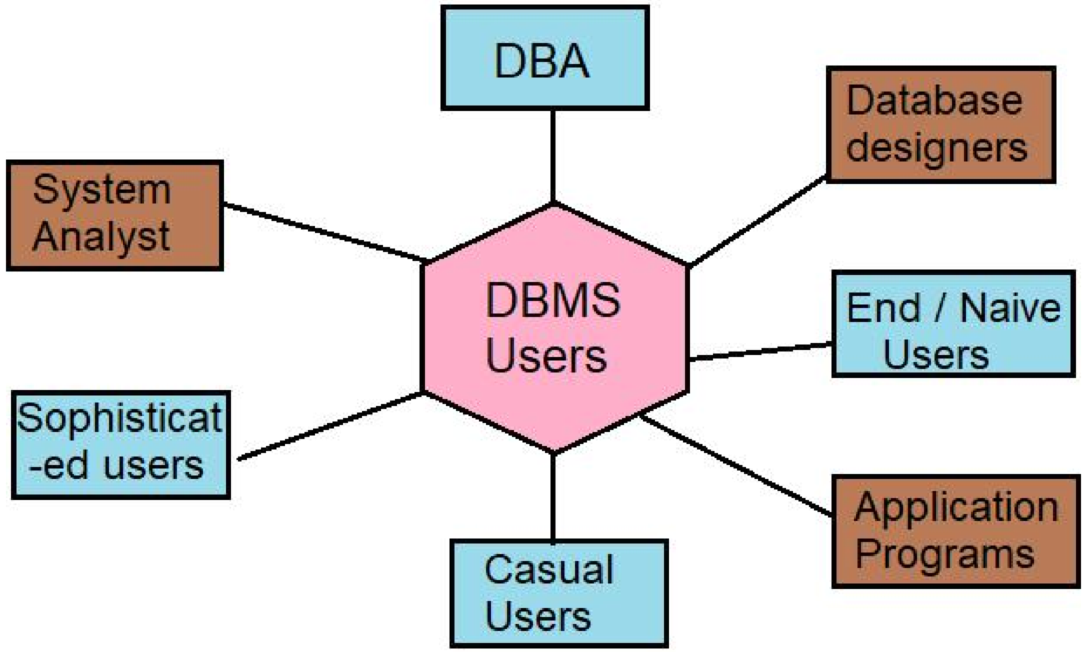
Main Characteristics of the Database Approach

- **Data Abstraction**:
  - A data model is used to hide storage details and present the users with a conceptual view of the database.
  - Programs refer to the data model constructs rather than data storage details
- **Support of multiple views of the data:**
  - Each user may see a different view of the database, which describes only the data of interest to that user.

Main Characteristics of the Database Approach

- **Sharing of data and multi-user transaction processing**:
  - Allowing a set of concurrent users to retrieve from and to update the database.
  - Concurrency control within the DBMS guarantees that each transaction is correctly executed or aborted.
  - Recovery subsystem ensures each completed transaction has its effect permanently recorded in the database
  - OLTP (Online Transaction Processing) is a major part of database applications. This allows hundreds of concurrent transactions to execute per second.

Database Users



Actors on the scene

Database Administrators:

Responsible for authorizing access to the database, for coordinating and monitoring its use, acquiring software and hardware resources, controlling its use and monitoring the efficiency of operations.

Database Designers:

Responsible for defining the content, the structure, the constraints, and functions or transactions against the database. They must communicate with the end-users and understand their needs.

Database Users

Actors on the scene (continued)

End-users: They use the data for queries, reports and some of them update the database content.

End-users can be categorized into:

Casual: access database occasionally when needed

Naïve or Parametric: they make up a large section of the end-user population.

They use previously well-defined functions in the form of "canned transactions" against the database.

Examples are bank tellers or reservation clerks who do this activity for an entire shift of operations.

Database Users

Actors on the scene (continued)

Sophisticated:

-These include business analysts, scientists, engineers, and others thoroughly familiar with the system capabilities.

- Many use tools in the form of software packages that work closely with the stored database.

Stand-alone:

- Mostly maintain personal databases using ready-to-use packaged applications.
- An example is a tax program user that creates its own internal database.
- Another example is a user that maintains an address book

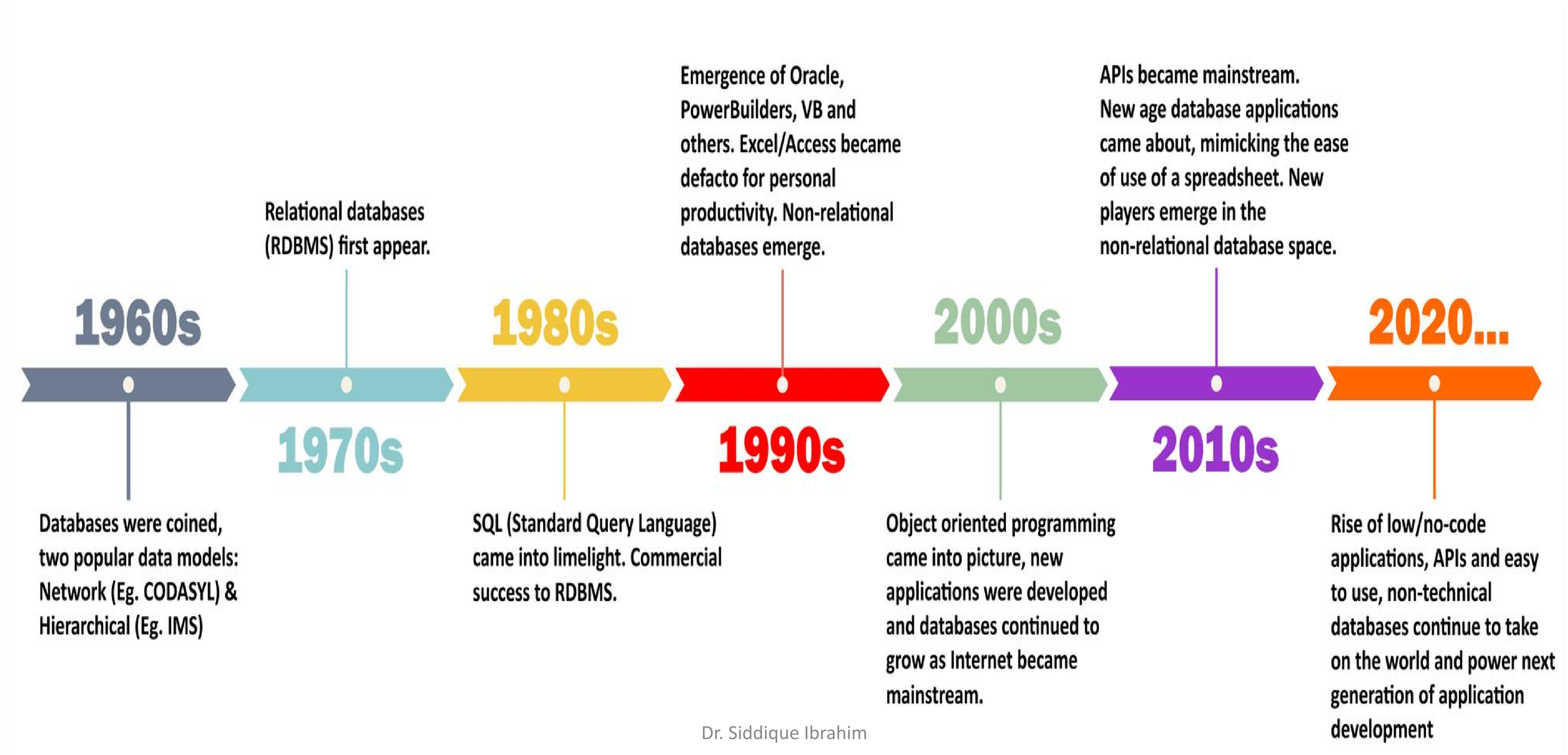
Advantages of Databases

- Controlling redundancy in data storage and in development and maintenance efforts.
    - Sharing of data among multiple users.
- Restricting unauthorized access to data.
- Providing persistent storage for program Objects
    - In Object-oriented DBMSs
- Providing Storage Structures (e.g. indexes) for efficient Query Processing
- Providing backup and recovery services.
- Providing multiple interfaces to different classes of users.
- Representing complex relationships among data.
- Enforcing integrity constraints on the database.
- Drawing inferences and actions from the stored data using deductive and active rules

When not to use a DBMS

- If the database system is not able to handle the complexity of data because of modeling limitations.
- If the database users need special operations not supported by the DBMS.
- If the database and applications are simple, well-defined, and not expected to change.
- If there are stringent real-time requirements that may not be met because of DBMS overhead.
- If access to data by multiple users is not required.

History of Database System



1960s

Databases were coined, two popular data models: Network (Eg. CODASYL) & Hierarchical (Eg. IMS)

1970s

Relational databases (RDBMS) first appear.

1980s

SQL (Standard Query Language) came into limelight. Commercial success to RDBMS.

1990s

Emergence of Oracle, PowerBuilders, VB and others. Excel/Access became defacto for personal productivity. Non-relational databases emerge.

2000s

Object oriented programming came into picture, new applications were developed and databases continued to grow as Internet became mainstream.

2010s

APIs became mainstream. New age database applications came about, mimicking the ease of use of a spreadsheet. New players emerge in the non-relational database space.

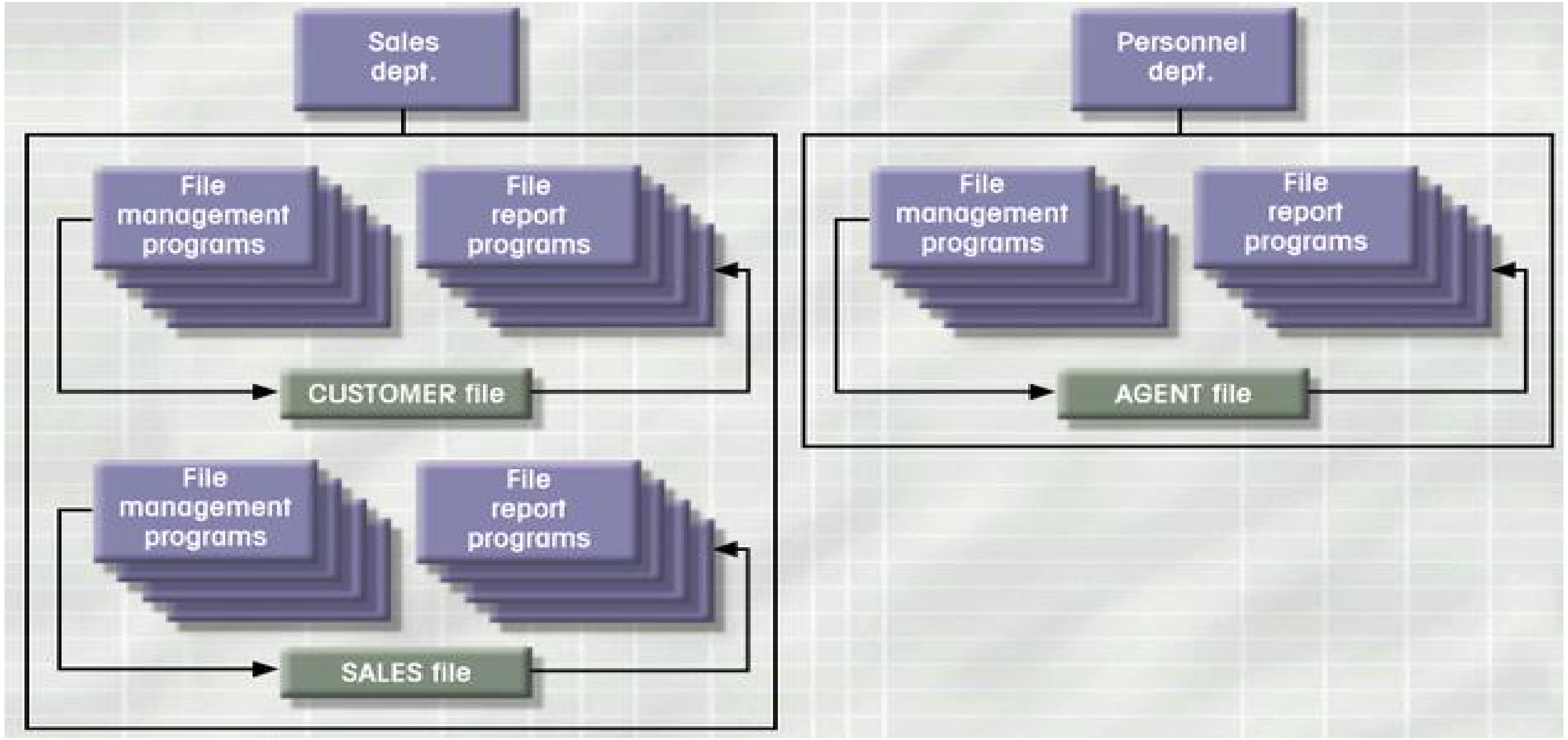
2020...

Rise of low/no-code applications, APIs and easy to use, non-technical databases continue to take on the world and power next generation of application development

Database: Historical Roots

- Manual File System
  - to keep track of data
  - used tagged file folders in a filing cabinet
  - organized according to expected use
    - e.g. file per customer
  - easy to create, but hard to
    - locate data
    - aggregate/summarize
- Computerized File System
  - to accommodate the data growth and information need
  - Manual file system structures were duplicated in the computer
  - Data Processing (DP) specialists wrote customized programs to
    - write, delete, update data (i.e. management)
    - extract and present data in various formats (i.e. report)

File System: Example

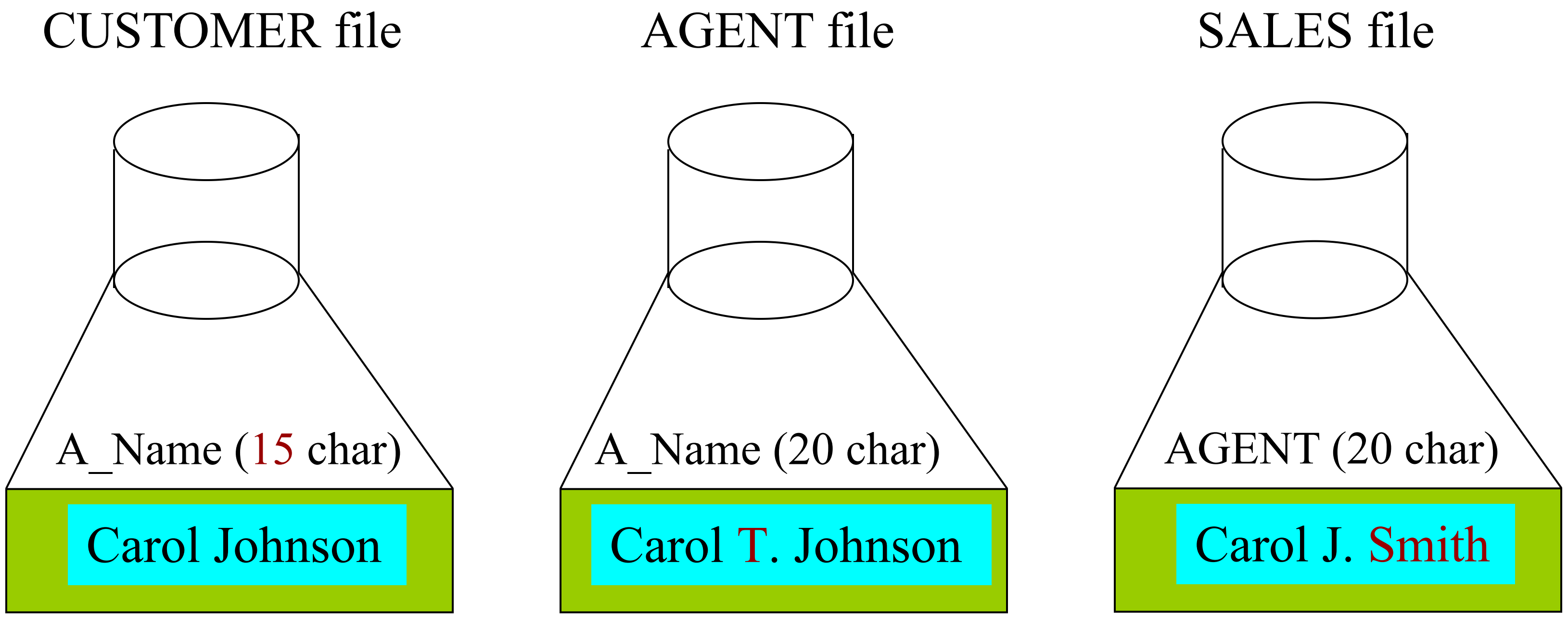


Database Systems: Design, Implementation, & Management: Rob & Coronel

File System: Weakness

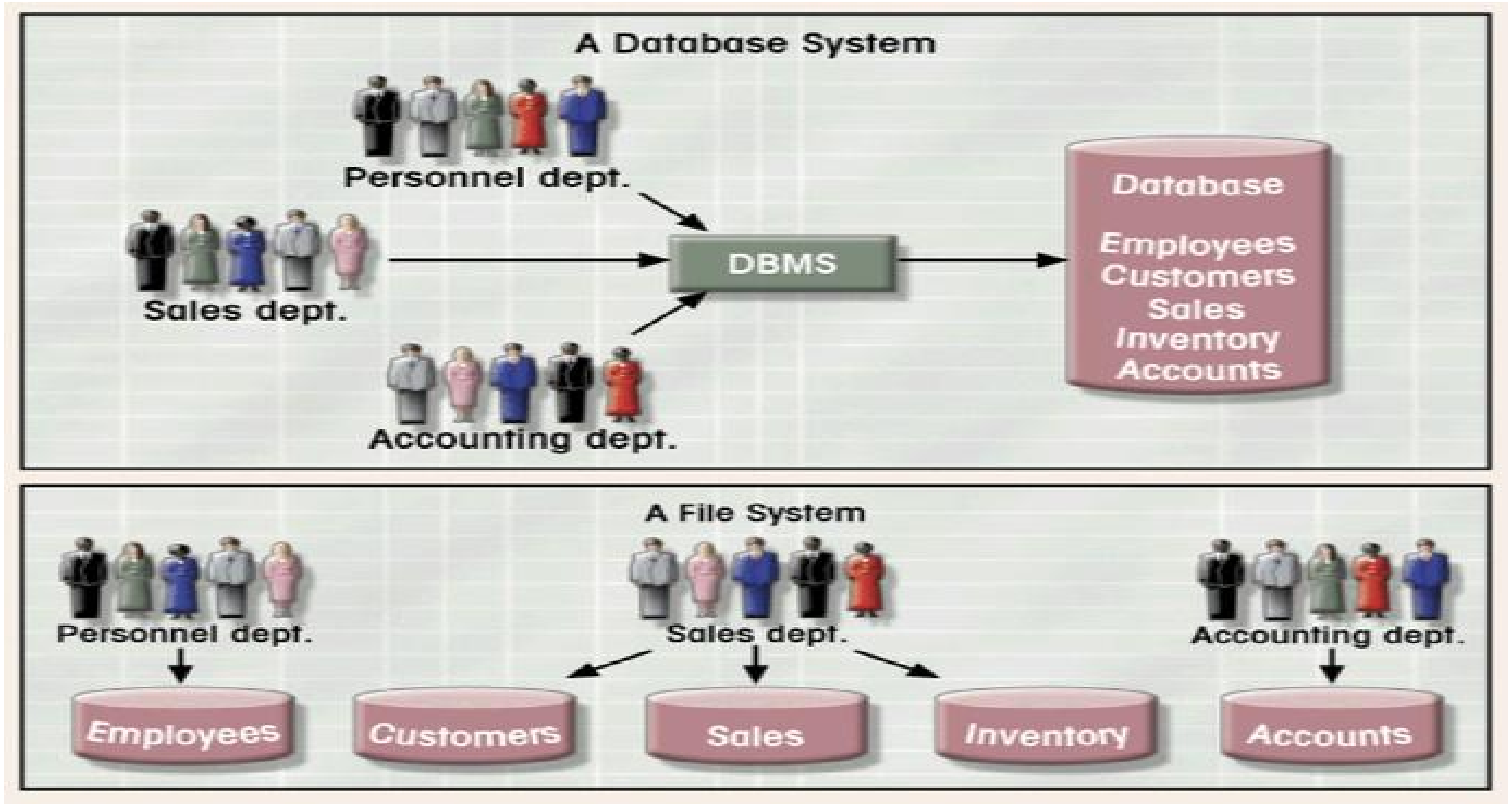
- Weakness
  - “Islands of data” in scattered file systems.
- Problems
  - Duplication
    - same data may be stored in multiple files
  - Inconsistency
    - The same data may be stored by different names in different format
  - Rigidity
    - requires customized programming to implement any changes
    - cannot do ad-hoc queries
- Implications
  - Waste of space
  - Data inaccuracies
  - High overhead of data manipulation and maintenance

File System: Problem Case



- inconsistent field name, field size
- inconsistent data values
- data duplication

Database System vs. File System



Database Systems: Design, Implementation, & Management: Rob & Coronel

Database System vs. File System

A DBMS coordinates both physical and logical access to the data,

- file-processing system coordinates only the physical access.

DBMS reduces data duplication.

- whereas data written by one program may not be readable by another program in the file system.

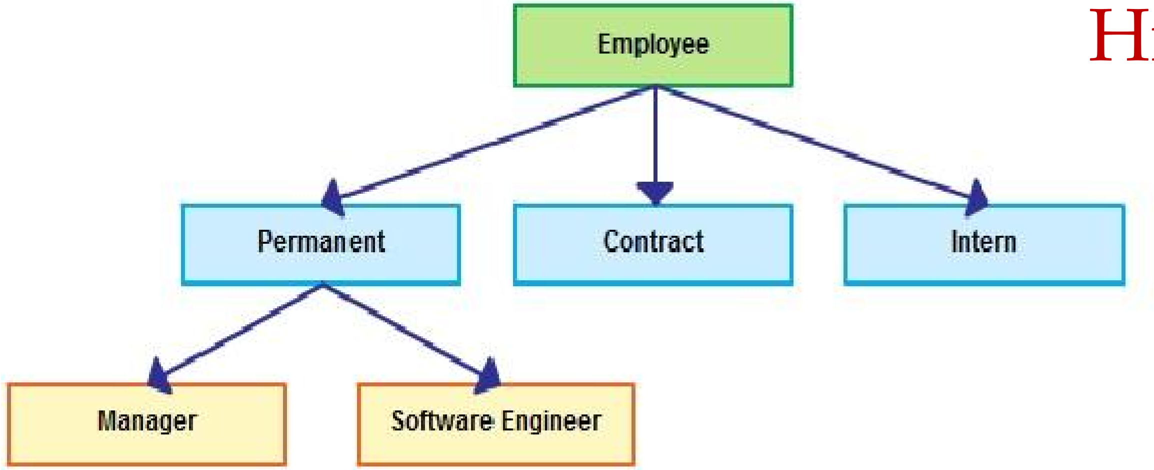
A DBMS is designed to allow flexible access to data by using queries.

- In file, the system is designed to allow predetermined access to data i.e. compiled program.
- Redundancy control in DBMS.
- Unauthorized access restricted in DBMS.
- DBMS provides backup and recovery.

Historical Development of Database Technology

Early Database Applications:

- The Hierarchical and Network Models were introduced in the mid 1960s and dominated during the seventies.
- A bulk of the worldwide database processing still occurs using these models, particularly, the hierarchical model using IBM's IMS system.



Hierarchical Database example

Hierarchical Database Model

- Assumes data relationships are hierarchical
  - One-to-Many (1:M) relationships
    - Each parent can have many children
    - Each child has only one parent
- Logically represented by an upside down tree

Historical Development of Database Technology

Hierarchical Database Model

Advantages

- Conceptual simplicity
  - groups of data could be related to each other
  - related data could be viewed together
- Centralization of data
  - reduced redundancy and promoted consistency

Disadvantages

- Limited representation of data relationships
  - did not allow Many-to-Many (M:N) relations
- Complex implementation
  - required in-depth knowledge of physical data storage
- Structural Dependence
  - data access requires physical storage path
- Lack of Standards
  - limited portability

Historical Development of Database Technology

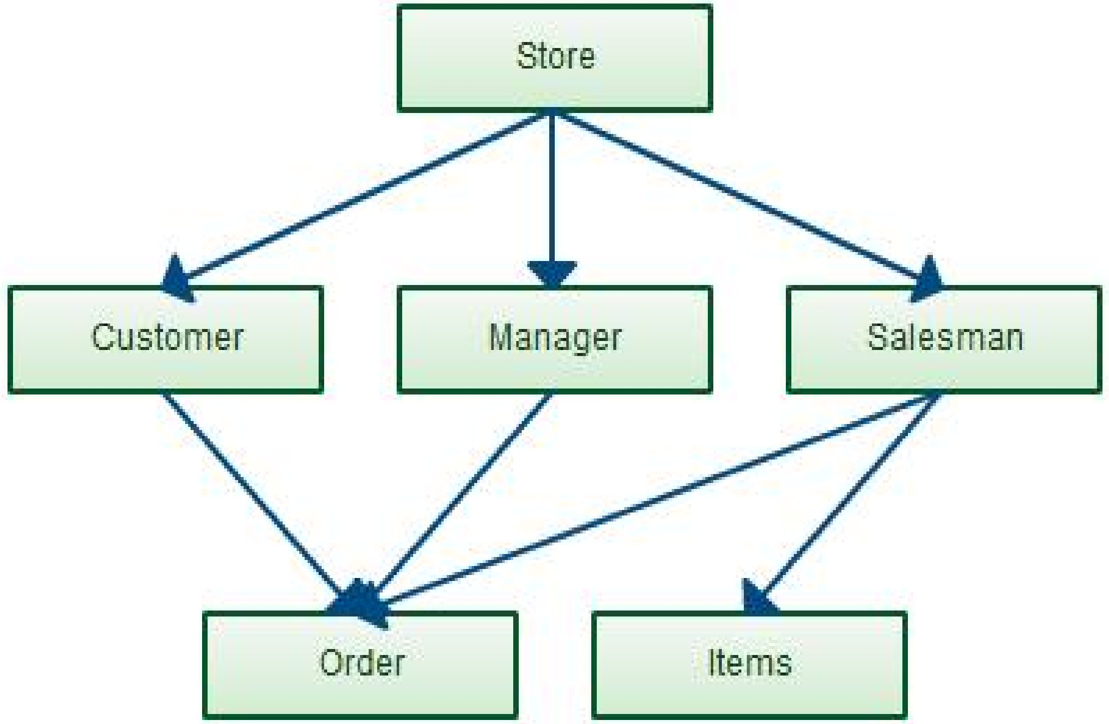
Network Database

Objectives

- Represent more complex data relationships
- Improve database performance
- Impose a database standard

Network Database Model

- Similar to Hierarchical Model
  - Records linked by pointers
- Composed of sets
  - Each set consists of owner (parent) and member (child)
- Many-to-Many (M:N) relationships representation
  - Each owner can have multiple members (1:M)
  - A member may have several owners



Network Database example

Historical Development of Database Technology

Network Database

Advantages

- More data relationship types
- More efficient and flexible data access
  - "network" vs. "tree" path traversal
- Conformance to standards
  - enhanced database administration and portability

Disadvantages

- System complexity
  - require familiarity with the internal structure for data access
- Lack of structural independence
  - small structural changes require significant program changes

Historical Development of Database Technology

Relational Database

- The dominant database model of today
- Eliminated pointers and used tables to represent data

Tables

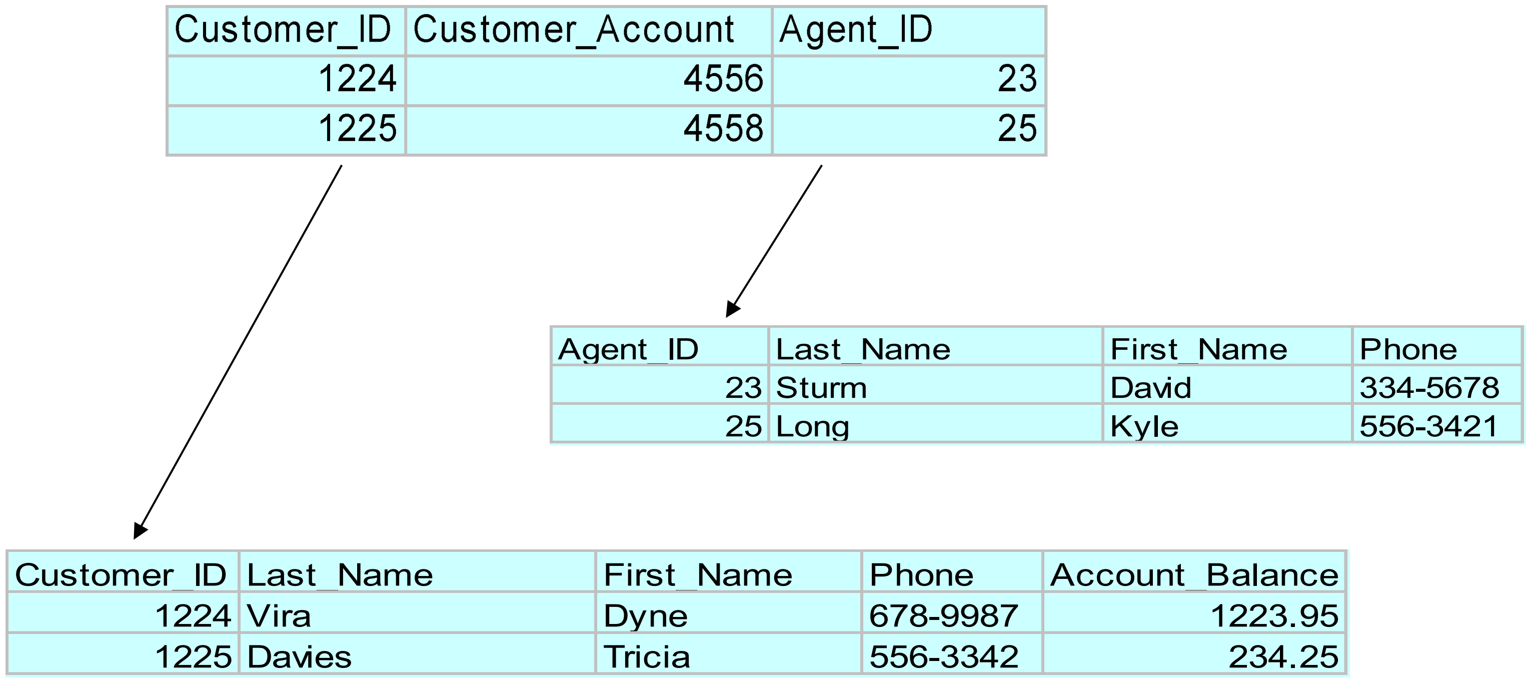
- flexible logical structure for data representation
- a series of row/column intersections
- related by sharing common entity characteristic(s)

- Relational model was originally introduced in 1970, and was heavily researched and experimented with within IBM Research and several universities.
- Relational DBMS Products emerged in the early 1980s.

Historical Development of Database Technology

Relational Database

- Provides a logical "human-level" view of the data and associations among groups of data (i.e., tables)



Customer_ID	Customer_Account	Agent_ID
1224	4556	23
1225	4558	25

Agent_ID	Last_Name	First_Name	Phone
23	Sturm	David	334-5678
25	Long	Kyle	556-3421

Customer_ID	Last_Name	First_Name	Phone	Account_Balance
1224	Vira	Dyne	678-9987	1223.95
1225	Davies	Tricia	556-3342	234.25

Historical Development of Database Technology

Relational Database

Advantages

- Structural independence
  - Separation of database design and physical data storage/access
  - Easier database design, implementation, management, and use
- Ad hoc query capability with Structured Query Language (SQL)
  - SQL translates user queries to codes

Disadvantages

- Substantial hardware and system software overhead
  - more complex system
- Poor design and implementation is made easy
  - ease-of-use allows careless use of RDBMS

Historical Development of Database Technology

Relational Database

Advantages

- Structural independence
  - Separation of database design and physical data storage/access
  - Easier database design, implementation, management, and use
- Ad hoc query capability with Structured Query Language (SQL)
  - SQL translates user queries to codes

Disadvantages

- Substantial hardware and system software overhead
  - more complex system
- Poor design and implementation is made easy
  - ease-of-use allows careless use of RDBMS

Historical Development of Database Technology

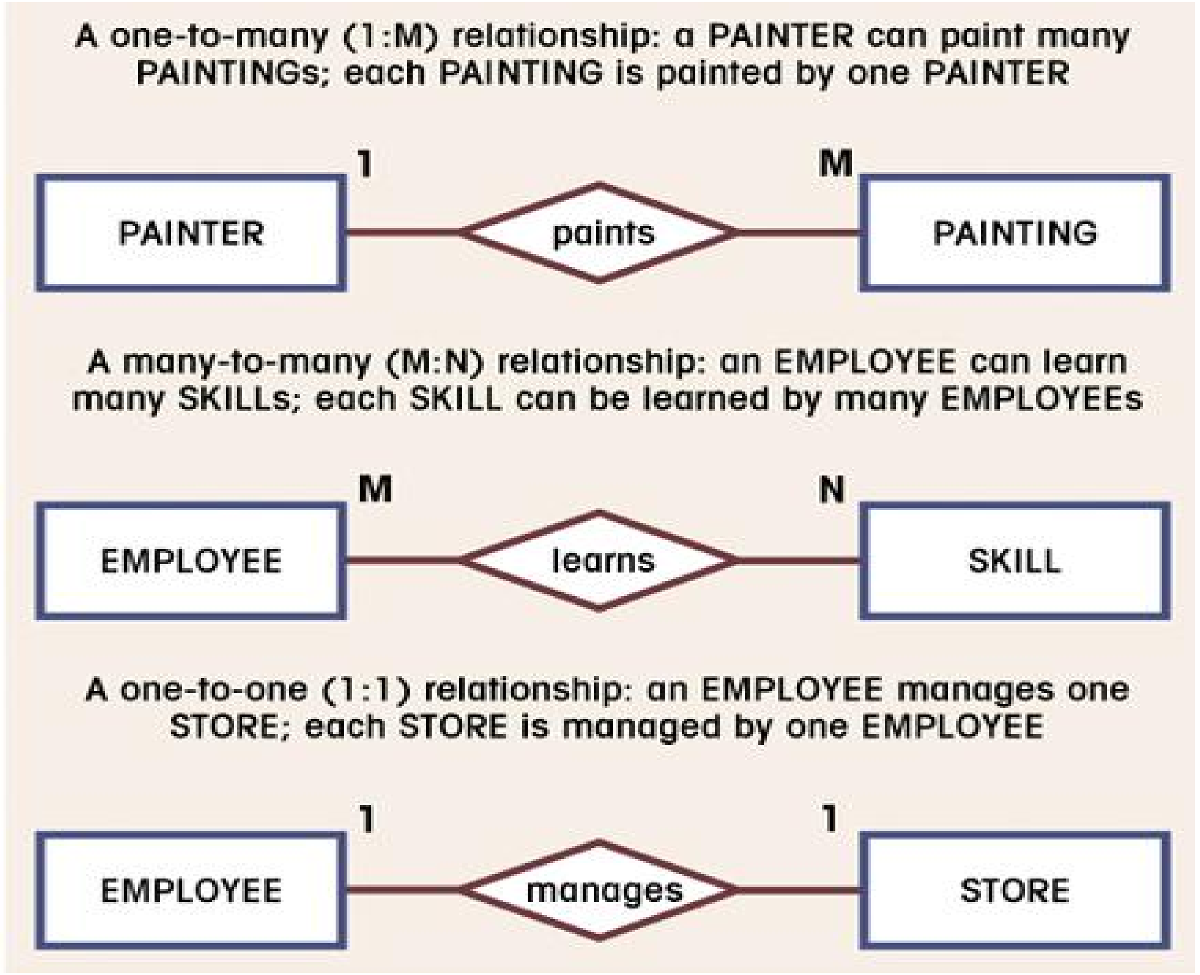
Entity Relationship Model

- Peter Chen's Landmark Paper in 1976
  - "The Relationship Model: Toward a Unified View of Data"
  - Graphical representation of entities and their relationships

- Entity Relationship (ER) Model
  - Based on Entity, Attributes & Relationships
    - Entity is a thing about which data are to be collected and stored
      - e.g. EMPLOYEE
    - Attributes are characteristics of the entity
      - e.g. SSN, last name, first name
    - Relationships describe associations between entities
      - i.e. 1:M, M:N, 1:1
  - Complements the relational data model concepts
    - Helps to visualize the structure and content of data groups
      - entity is mapped to a relational table
    - Tool for conceptual data modeling (higher level representation)
  - Represented in an Entity Relationship Diagram (ERD)
    - Formalizes a way to describe relationships between groups of data

Historical Development of Database Technology

E-R Diagram - Chen Model



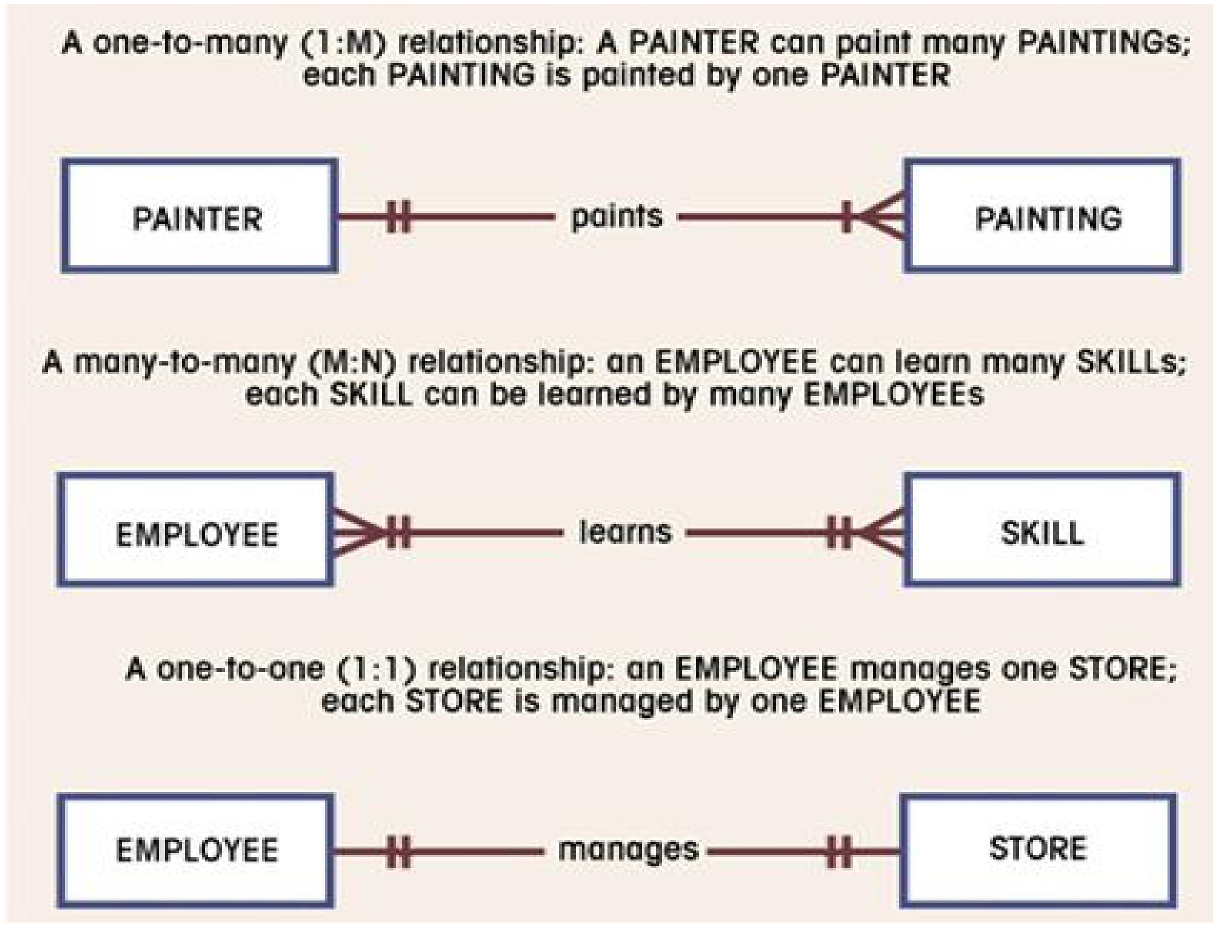
A one-to-many (1:M) relationship: a PAINTER can paint many PAINTINGs; each PAINTING is painted by one PAINTER
1
M
PAINTER
paints
PAINTING
A many-to-many (M:N) relationship: an EMPLOYEE can learn many SKILLs; each SKILL can be learned by many EMPLOYEEs
M
N
EMPLOYEE
learns
SKILL
A one-to-one (1:1) relationship: an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE
1
1
EMPLOYEE
manages
STORE

Database Systems: Design, Implementation, & Management: Rob & Coronel

- Entity
  - Represented by a rectangle with its name in capital letters.
- Relationships
  - Represented by an active or passive verb inside the diamond that connects the related entities.
- Connectivities
  - i.e., types of relationship
  - written next to each entity box.

Historical Development of Database Technology

E-R Diagram: Crow's Foot Model



A one-to-many (1:M) relationship: A PAINTER can paint many PAINTINGs; each PAINTING is painted by one PAINTER
PAINTER
paints
PAINTING
A many-to-many (M:N) relationship: an EMPLOYEE can learn many SKILLs; each SKILL can be learned by many EMPLOYEEs
EMPLOYEE
learns
SKILL
A one-to-one (1:1) relationship: an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE
EMPLOYEE
manages
STORE

Database Systems: Design, Implementation, & Management: Rob & Coronel

- Entity
  - represented by a rectangle with its name in capital letters.
- Relationships
  - represented by an active or passive verb that connects the related entities.
- Connectivities
  - indicated by symbols next to entities.
    - 2 vertical lines for 1
    - "crow's foot" for M

Historical Development of Database Technology

E-R Model:

- Advantages
    - Exceptional conceptual simplicity
        - easily viewed and understood representation of database
        - facilitates database design and management
    - Integration with the relational database model
        - enables better database design via conceptual modeling
- Disadvantages
    - Incomplete model on its own
        - Limited representational power
            - cannot model data constraints not tied to entity relationships
                - e.g. attribute constraints
            - cannot represent relationships between attributes within entities
        - No data manipulation language (e.g. SQL)
    - Loss of information content
        - Hard to include attributes in ERD

Historical Development of Database Technology

Object-Oriented Database

- Semantic Data Model (SDM)
  - Modeled both data and their relationships in a single structure (object)
    - Developed by Hammer & McLeod in 1981
- Object-oriented concepts became popular in 1990s
  - Modularity facilitated program reuse and construction of complex structures
  - Ability to handle complex data types (e.g. multimedia data)

Historical Development of Database Technology

- Object-Oriented Database Model (OODBM)
  - Maintains the advantages of the ER model but adds more features
  - Object = entity + relationships (between & within entity)
    - consists of attributes & methods
      - attributes describe properties of an object
      - methods are all relevant operations that can be performed on an object
    - self-contained abstraction of real-world entity
  - Class = collection of similar objects with shared attributes and methods
    - e.g. EMPLOYEE class = (employ1 object, employ2 object, …)
    - organized in a class hierarchy
      - e.g. PERSON > EMPLOYEE, CUSTOMER
  - Incorporates the notion of inheritance
    - attributes and methods of a class are inherited by its descendent classes

Historical Development of Database Technology

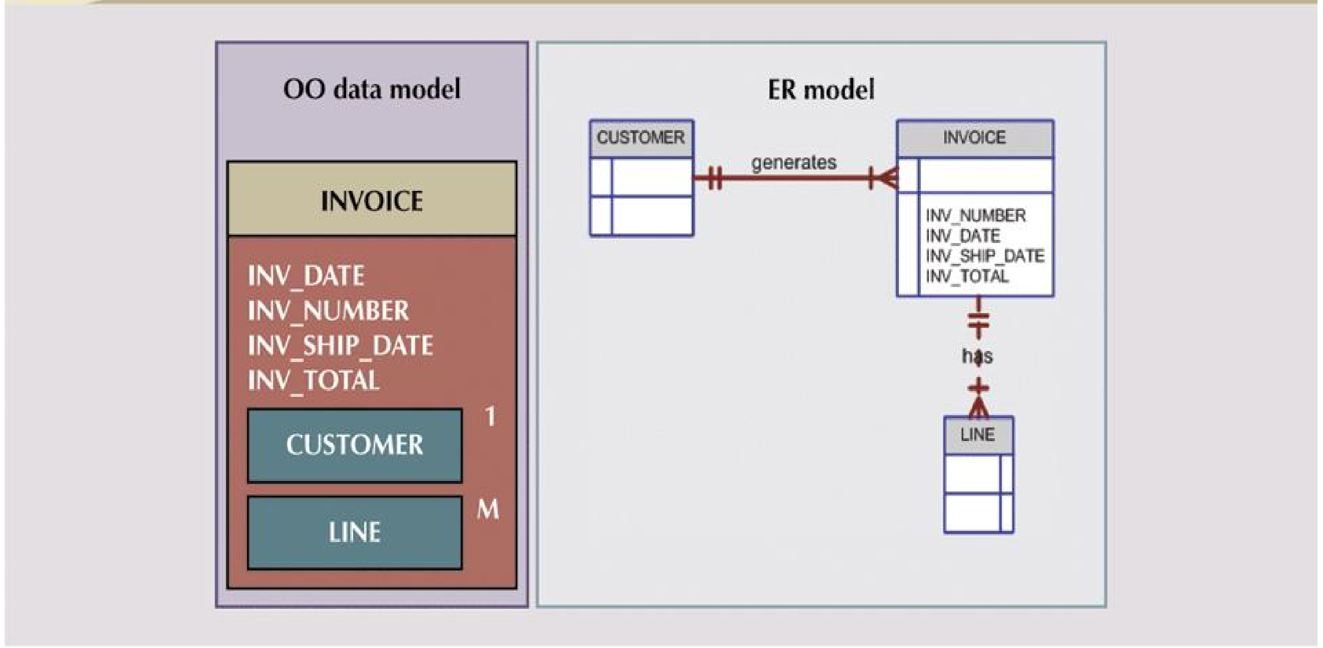
OO Database Model vs. E-R Model

OODBM:

- can accommodate relationships within a object

- objects to be used as building blocks for autonomous structures

FIGURE 2.7 A comparison of the OO model and the ER model



Database Systems: Design, Implementation, & Management: Rob & Coronel

Object-Oriented Database:

- Advantages
  - Semantic representation of data
    - fuller and more meaningful description of data via object
  - Modularity, reusability, inheritance
  - Ability to handle
    - complex data
    - sophisticated information requirements

- Disadvantages
  - Lack of standards
    - no standard data access method
  - Complex navigational data access
    - class hierarchy traversal
  - Steep learning curve
    - difficult to design and implement properly
  - More system-oriented than user-centered
  - High system overhead
    - slow transactions

Historical Development of Database Technology

Object-Relational Model

- Most Recent Trend. Started with Informix Universal Server.
- Relational systems incorporate concepts from object databases leading to object-relational.
- Exemplified in the latest versions of Oracle-10i, DB2, and SQL Server and other DBMSs.
- Standards included in SQL-99 and expected to be enhanced in future SQL standards.

Historical Development of Database Technology

Web Database

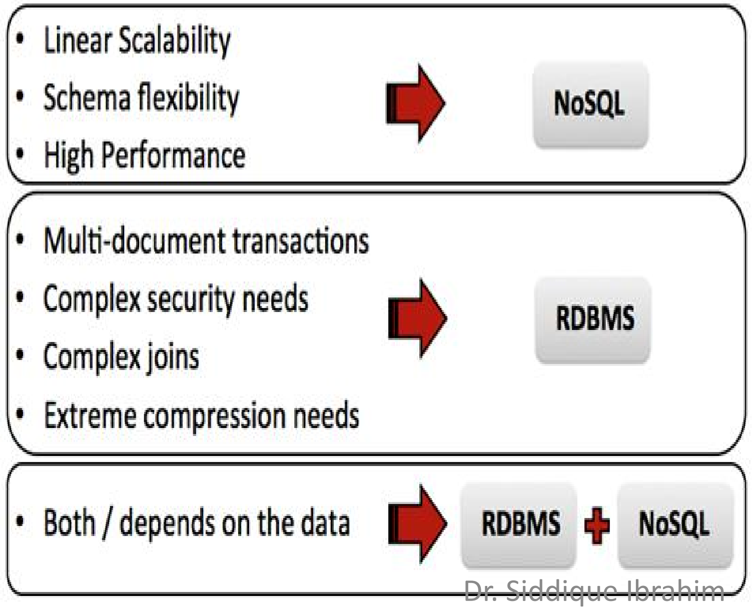
- Internet is emerging as a prime business tool
  - Shift away from models (e.g. relational vs. O-O)
  - Emphasis on interfacing with the Internet

- Characteristics of "Internet age" databases
  - Flexible, efficient, and secure Internet access
  - Support for complex data types and relationships
  - Seamless interfaces with multiple data sources and structures
  - Ease of use for end-user, database architect, and database administrator
    - Simplicity of conceptual database model
    - Many database design, implementation, and application development tools
    - Powerful DBMS GUI

Historical Development of Database Technology

NoSQL

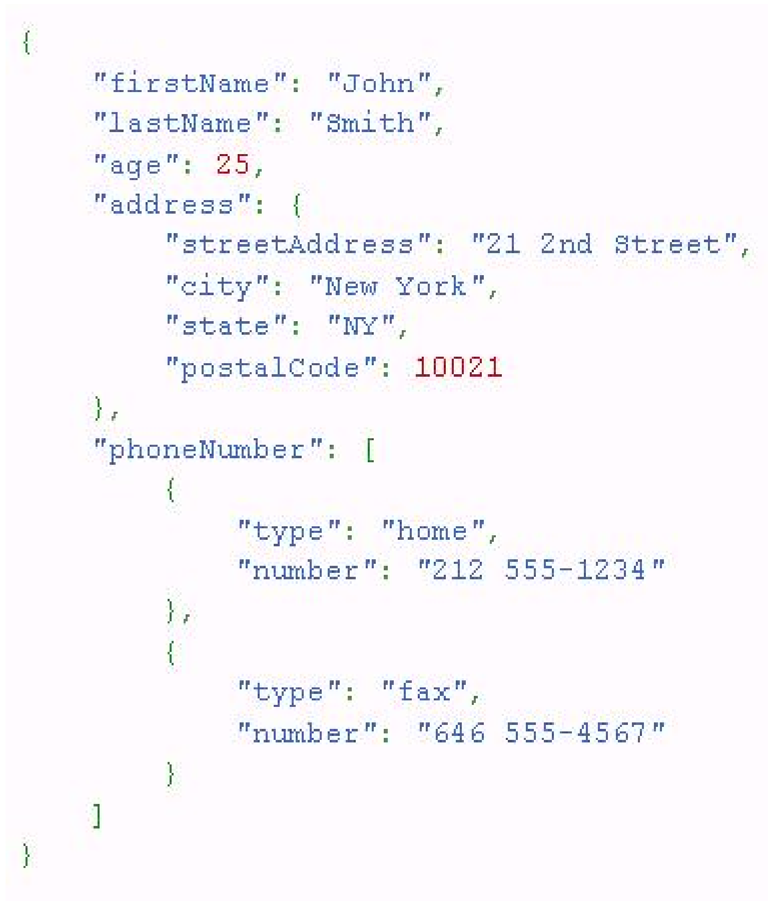
- NoSql is not literally "no sql". They are non relational data stores.
- Next Generation Databases being non-relational, distributed, open-source and horizontally scalable have become a favorite back end storage for cloud community . High performance is the driving force.



Historical Development of Database Technology

NoSQL

- Pros
  - open source (Cassandra, CouchDB, Hbase, MongoDB, Redis)
  - Elastic scaling
  - Key-value pairs, easy to use
  - Useful for statistical and real-time analysis of growing lists of elements (tweets, posts, comments)
- Cons
  - Security (No ACID: ACID (Atomicity, Consistency, Isolation, Durability)
  - No indexing support
  - Immature
  - Absence of standardization



```
{
    "firstName": "John",
    "lastName": "Smith",
    "age": 25,
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": 10021
    },
    "phoneNumber": [
        {
            "type": "home",
            "number": "212 555-1234"
        },
        {
            "type": "fax",
            "number": "646 555-4567"
        }
    ]
}
```