# CS1001

Lecture 26

## Overview

- Artificial Intelligence
- Database Systems

## Reading

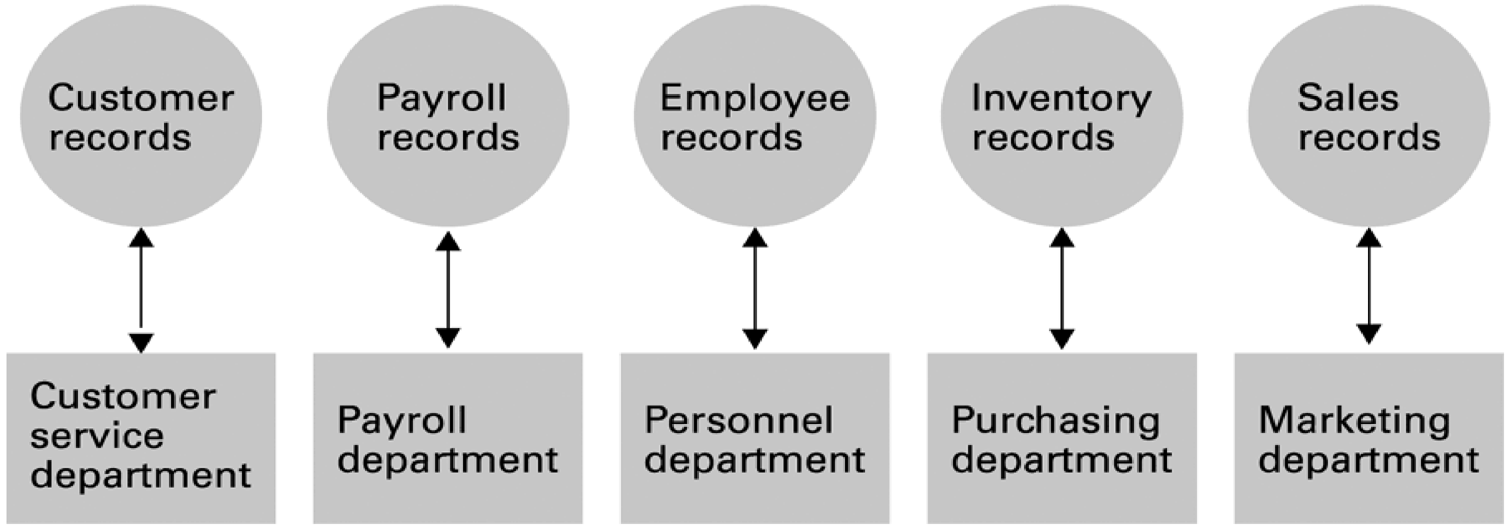
Brookshear, 9

## Databases

- Databases hold knowledge
- One fundamental limitations of all computation is having knowledge represented in an available and structured way
- (needed for AI) http://www.abenteuermedien.de/jabberwock/

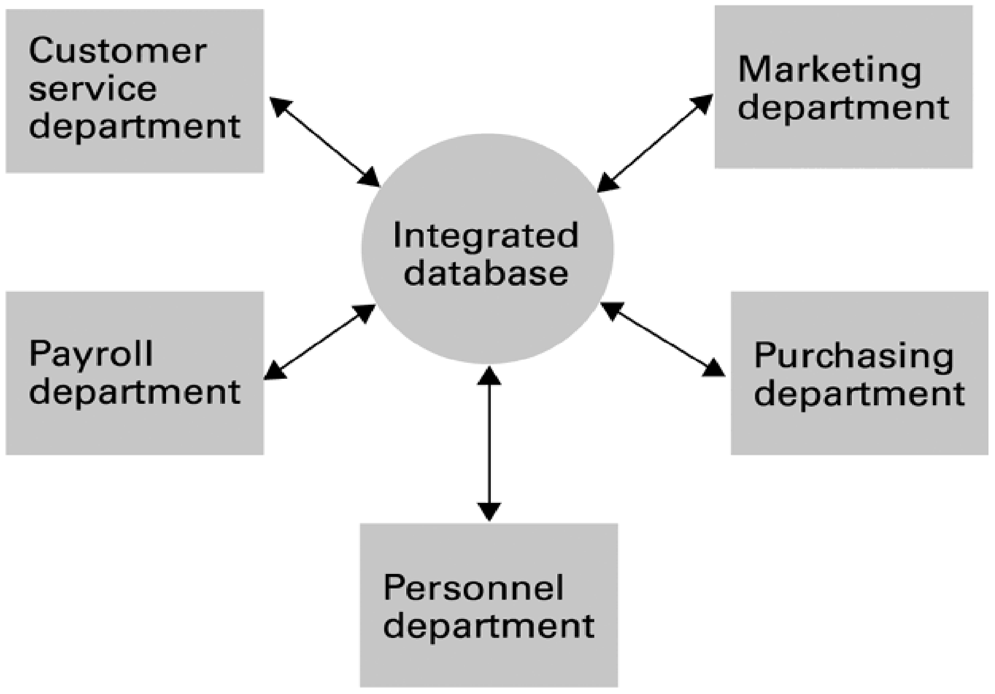
# File-Oriented System

a. File-oriented information system



## Database-Oriented System

b. Database-oriented information system



## Definition

- A database is a collection of related data
- example:
  - 1. collection of names, home address and telephone numbers
  - 2. collection of words to make paragraph in a page
- A database represents some aspect of the real world, sometimes called the miniworld or the Universe of Discourse (UoD).
- A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.

## Definition ….

- A database is designed, built, and populated with data for a specific purpose. It has an intended group of users and some preconceived applications in which these users are interested.
- A database can be of any size and of varying complexity
- A database may be generated and maintained manually or it may be computerized (set of application programs or by RDBM).

## Database Management System

- Is a collection of programs that enables users to create and maintain a database.
- The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, and manipulating databases for various applications
- **Defining:** a database involves specifying the data types, structures, and constraints for the data to be stored in the database.

## DBMS …..

- **Constructing:** the database is the process of storing the data itself on some storage medium that is controlled by the DBMS.
- **Manipulating:** a database includes such functions as querying the database to retrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data.
- We will call the database and DBMS software together a database system

## Example

- UNIVERSITY database for maintaining information concerning students, courses, and grades in a university environment
- define: file (records), data elements, data type ( for each data element)
- construct: store data in the appropriate files (note that records may be related between files)
- Manipulation: querying, updating
  - informal queries and updates must be specified precisely in the database system language before they can be processed.

## DB Vs Programming with files

- Self-Describing Nature of a Database System
  - single repository of data is maintained
  - contains not only the database itself but also a complete definition or description of the database structure and constraints (system catalogue).
  - information stored in the catalog is called meta-data
  - catalog used by the DBMS and users.
  - The DBMS software work equally well with any number of database applications.

## DB Vs Programming with files .....

- Insulation between Programs and Data, and Data Abstraction
  - program-data independence
  - The characteristic that allows program-data independence and program-operation independence is called data abstraction
  - A DBMS provides users with a conceptual representation of data that does not include many of the details of how the data is stored or how the operations are implemented. (data model )
  - data model hides storage and implementation details that are not of interest to most database users.
  - in object-oriented and object-relational databases, abstraction is carried one level further to include not only the data structure but also the operations on the data

## Database Actors

- Database Administrators
  - In a database environment, the primary resource is the database itself and the secondary resource is the DBMS and related software
  - authorizing access to the database
  - coordinating and monitoring its use
  - acquiring software and hardware resources as needed
- Database Designers
  - identifying the data to be stored in the database
  - choosing appropriate structures to represent and store this data undertaken before the database is actually implemented and populated with data

## Database Actors …..

- communicate with all prospective database users, in order to understand their requirements
- develop a view of the database that meets the data and processing requirements for each group of users
- These views are then analyzed and integrated with the views of other user groups. The final database design must be capable of supporting the requirements of all user groups

- End Users
  - access to the database for querying, updating, and generating reports
  - Casual end users:
  - occasionally access the database
  - need different information each time
  - learn only a few facilities that they may use repeatedly.

# Database Actors …..

- use a sophisticated database query language to specify their requests
- typically middle- or high-level managers or other occasional browsers

- Naive or parametric end users
  - constantly querying and updating the database, using standard types of queries and updates called canned transactions that have been carefully programmed and tested
  - need to learn very little about the facilities provided by the DBMS
  - Bank tellers check account balances and post withdrawals and deposits
  - Reservation clerks for airlines, hotels, and car rental companies check availability for a given request and make reservations
  - Clerks at receiving stations for courier mail enter package identifications via bar codes and descriptive information through buttons to update a central database of received and in-transit packages

## Database Actors …..

- Sophisticated end users
  - Engineers, scientists, business analysts, and others who thoroughly familiarize themselves with the facilities of the DBMS so as to implement their applications to meet their complex requirements
  - Try to learn most of the DBMS facilities in order to achieve their complex requirements
- Stand-alone users
  - Maintain personal databases by using ready-made program packages that provide easy-to-use menu- or graphics-based interfaces. An example is the user of a tax package that stores a variety of personal financial data for tax purposes
  - Typically become very proficient in using a specific software package

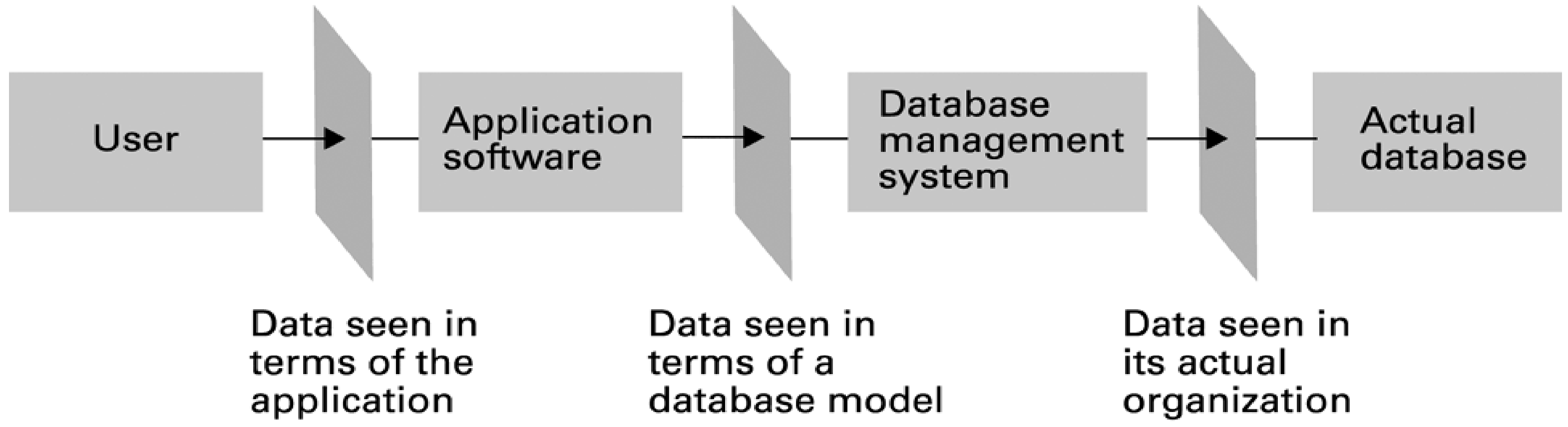
## Database Actors …..

- System Analysts and Application Programmers
  - Determine the requirements of end users, especially naive and parametric end users, and develop specifications for canned transactions that meet these requirements
  - Application programmers implement these specifications as programs; then they test, debug, document, and maintain these canned transactions
- Workers behind the Scene
  - Typically do not use the database for their own purposes
  - DBMS system designers and implementers
  - design and implement the DBMS modules (for implementing the catalog, query language, interface processors, data access, concurrency control, recovery, and security. ) and interfaces as a software package

## Database Actors …..

- Tool developers
  - Tools are optional packages that are often purchased separately
  - include packages for database design, performance monitoring, natural language or graphical interfaces, prototyping, simulation, and test data generation.
- Operators and maintenance personnel
  - system administration personnel who are responsible for the actual running and maintenance of the hardware and software environment for the database system

## The conceptual layers of a database


User
Application software
Database management system
Actual database
Data seen in terms of the application
Data seen in terms of a database model
Data seen in its actual organization

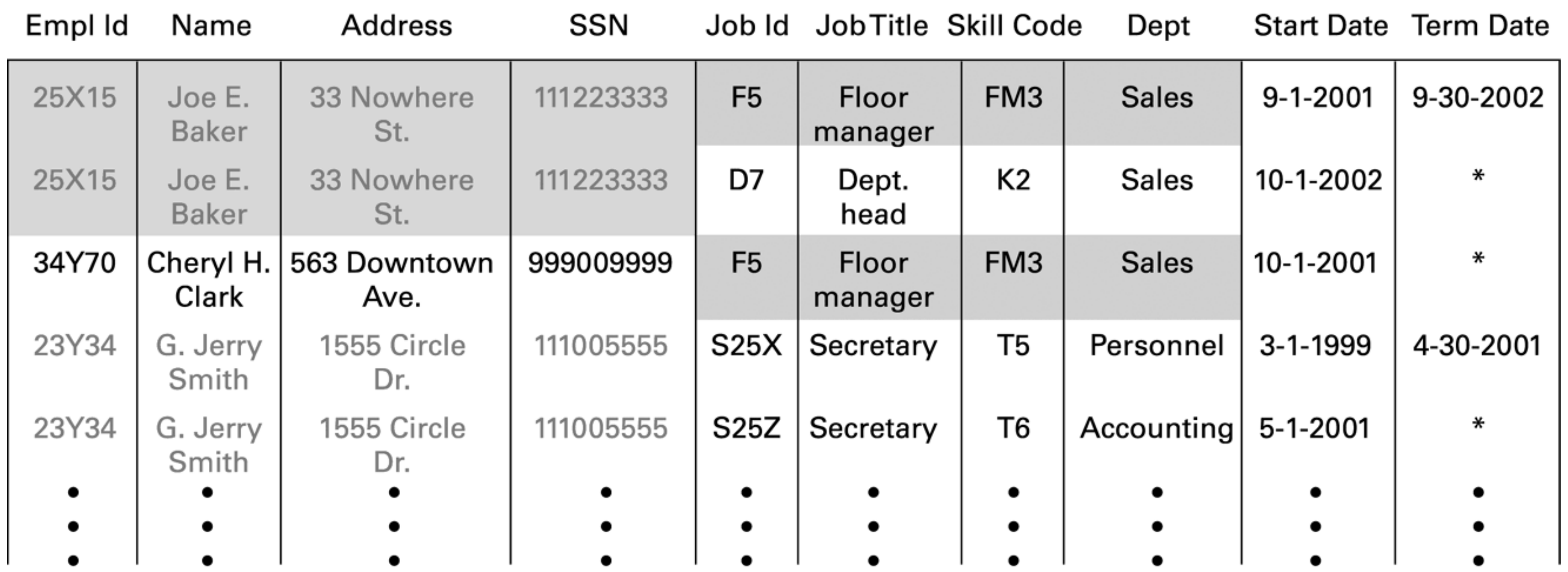
## Asking Questions

- The Database is an oracle – you need to be able to pose it questions and receive answers
- For relational databases, the language of choice is SQL (Structured Query Language)

## Questions

- The goal is to allow unambiguous questions to be formulated and for the system to encode information in such a way as to provide unambiguous answers
- You must *model* your problem to properly represent its information
- Example: Employee Records

# Example Data



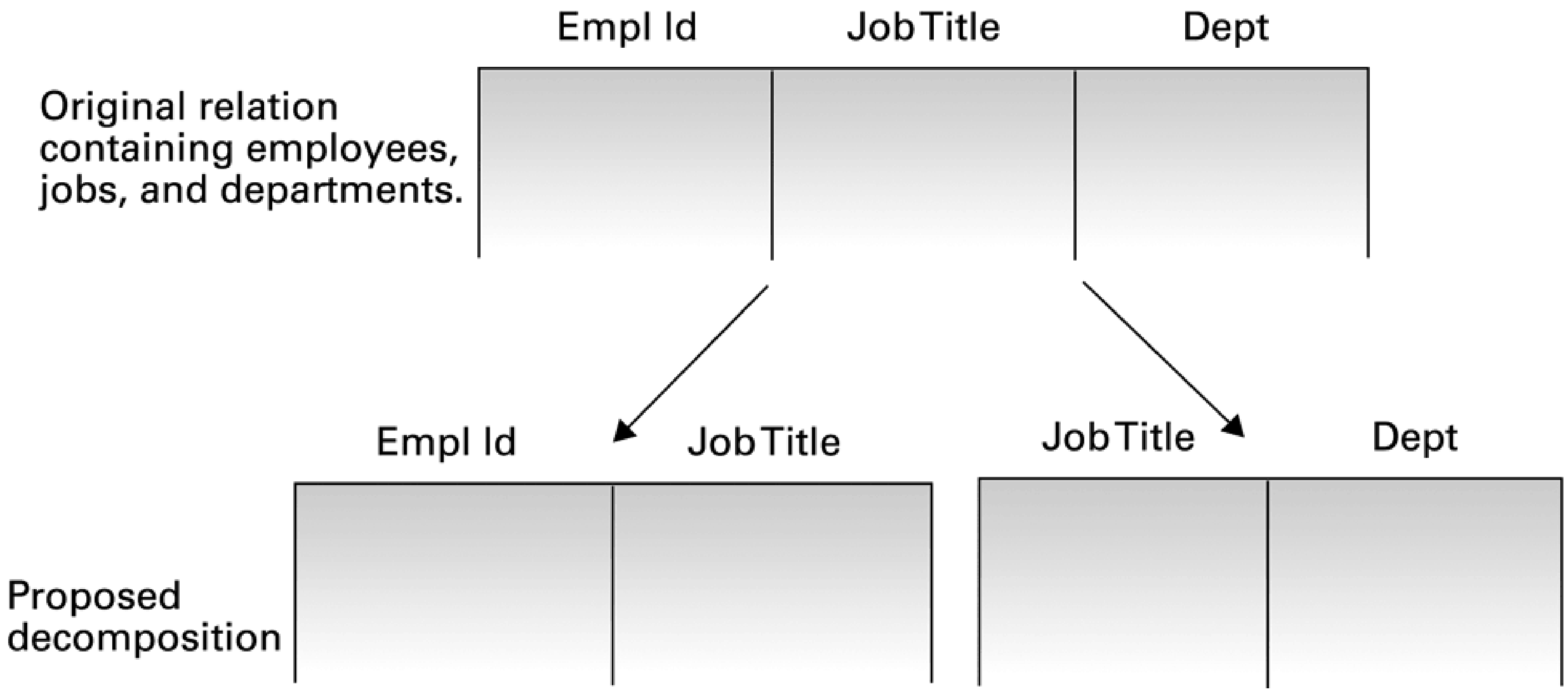
| Empl Id | Name | Address | SSN | Job Id | JobTitle | Skill Code | Dept | Start Date | Term Date |
|---|---|---|---|---|---|---|---|---|---|
| 25X15 | Joe E. Baker | 33 Nowhere St. | 111223333 | F5 | Floor manager | FM3 | Sales | 9-1-2001 | 9-30-2002 |
| 25X15 | Joe E. Baker | 33 Nowhere St. | 111223333 | D7 | Dept. head | K2 | Sales | 10-1-2002 | * |
| 34Y70 | Cheryl H. Clark | 563 Downtown Ave. | 999009999 | F5 | Floor manager | FM3 | Sales | 10-1-2001 | * |
| 23Y34 | G. Jerry Smith | 1555 Circle Dr. | 111005555 | S25X | Secretary | T5 | Personnel | 3-1-1999 | 4-30-2001 |
| 23Y34 | G. Jerry Smith | 1555 Circle Dr. | 111005555 | S25Z | Secretary | T6 | Accounting | 5-1-2001 | * |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

Key Point is how to structure this as to reduce redundancy
Eliminating Redundancy (replacing it with *relations*) allows for unambiguous question answering

Note that this is a *table* (fundamental structure of a database)

## Solution

- The solution is to decompose a single database *table* into multiple tables and set up relationships between the tables



## Files

- Note that each table is similar in structure to a *flat file*
- The added value of a database makes it easy to search among relationships between such flat files