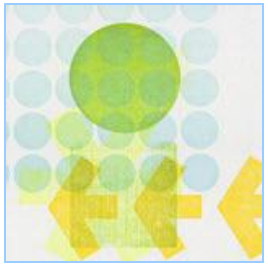


# CS & IT College – IS Dept.
# 2018/2019 Semester 2

# Database and Area of Use

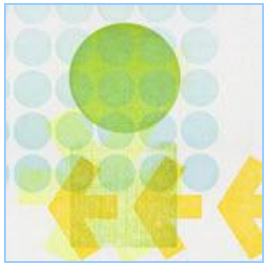
## *Lecturer : Asaad Alhijaj*

- ***CS & IT College Database Courses***
- ***Introduction to Database***
- ***Database Administration***
- ***Data Modeling and ER***
- ***Structured Query Language***

*References:*

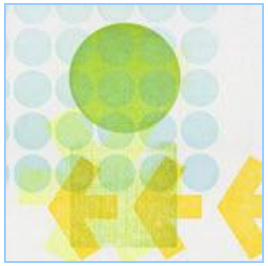

1) **"Database System Concepts Fourth Edition" by Abraham Silberschatz Henry F. Korth S. Sudarshan , McGraw-Hill ISBN 0-07-255481-9**

2) DAVID M. KROENKE'S DATABASE CONCEPTS, 2nd Edition © 2005 Pearson Prentice Hall



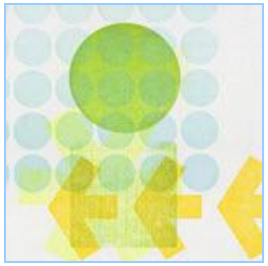
# CS & IT College Database Courses

- CS Dept.
  - **CS[206] Fundamentals of Database**
  - **CS[354] Distributed Database**
- IS Dept.
  - **IS[203] Database principles**
  - **[IS304] Database Management Systems I**
  - **IS352] Database Management Systems II**
  - **[IS402] Distributed Systems**
  - **[IS403] Data Warehouse and Data mining**



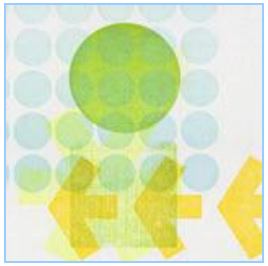
# Database principles / Fundamentals

- Introduction to Databases
- Conceptual Database Design
- Entity Relationship Diagram (ERD)
- Relational Data Model and Relational Database Constraints
- Relational Algebra
- Normalization
- Structured Query Language (SQL)
- File Structure and Indexes
- Database Performance Issues



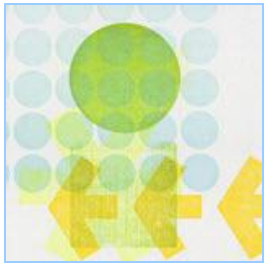
# Database Management Systems I

- A solid grounding in database management **theory** and **applications**, and its place within the **implementation** of computer based applications.
- Conceptual modeling to design the database that is implemented and queried using Structured Query Language **(SQL).**



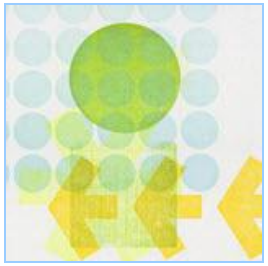
# Database Management Systems II

- Advanced concepts in Database Managements Systems:
  - practical skills in **designing**, **using**, and **optimizing** performance of databases.
  - Fundamentals of **object-oriented** and **distributed** databases and their **architectures**.
  - It aims to equip the students with the required techniques to *optimize database performance* and *troubleshoot* the *concurrency* problems of transactions.



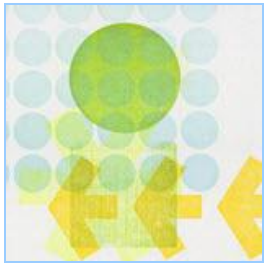
# Distributed Systems

- Characterization of distributed systems,
- Architectural models of distributed models,
- Interprocess communication,
- Distributed objects and remote invocation,
- Name services, time and global states, Coordination and agreement,
- Transactions and Concurrency control,
- Distributed transactions, Replication, and Distributed algorithms.
- Peer-to-Peer Systems, Distributed File Systems, Distributed Shared Memory, Web Services, and CORBA.



# Data Warehouse and Data mining

- The concepts of data warehousing and data mining,
- How they are used to convert data into strategic business information.
- Design, Architecture, Planning, and Project Management of a data warehouse.
- Data mining techniques (Classification, Association, Genetic algorithms, Machine learning, etc.)
- a way to discover useful relationships among data.



# Introduction to Database

**Information**

**Data**

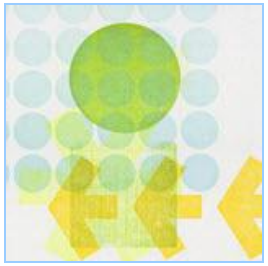
**Data Base (DB)**

**Relational Data Base (RDB)**

**Data Base Management System (DBMS)**

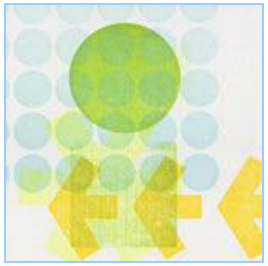
# Database Management System (DBMS)

- Collection of interrelated data
- Set of programs to access the data
- DBMS contains information about a particular enterprise
- DBMS provides an environment that is both *convenient* and *efficient* to use.
- Database Applications:
  - Banking: all transactions
  - Airlines: reservations, schedules
  - Universities: registration, grades
  - Sales: customers, products, purchases
  - Manufacturing: production, inventory, orders, supply chain
  - Human resources: employee records, salaries, tax deductions
- Databases touch all aspects of our lives



# Purpose of a Database

- The purpose of a **database** is to keep track of things
- Unlike a **list** or **spreadsheet**, a database may store information that is more complicated than a simple list



# List Modification Issues

- Redundancy and multiple themes create modification problems
  - Deletion problems
  - Update problems
  - Insertion problems

# List Modification Issues

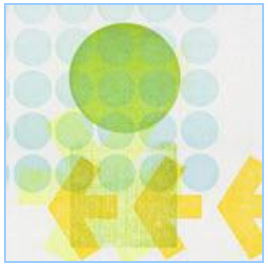
If Adviser **Baker** is changed to **Taing**, need to change *AdviserEmail* as well—if changed to **Valdez**, need to change *AdviserEmail*, *Department* and *Admin*

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | **LastName** | **FirstName** | **Email** | **AdviserName** | **AdviserEmail** | **Department** | **AdminName** |
| 2 | Andrews | Matthew | Andrews@ourcampus.edu | Baker | Baker@ourcampus.edu | Accounting | Smith |
| 3 | Brisbon | Lisa | Brisbon@ourcampus.edu | Valdez | Valdez@ourcampus.edu | Chemistry | Chaplin |
| 4 | Fischer | Douglas | Fischer@ourcampus.edu | Baker | Baker@ourcampus.edu | Accounting | Smith |
| 5 | Hwang | Terry | Hwang@ourcampus.edu | Taing | Taing@ourcampus.edu | Accounting | Smith |
| 6 | Lai | Tzu | Lai@ourcampus.edu | Valdez | Valdez@ourcampus.edu | Chemistry | Chaplin |
| 7 | ~~Marino~~ | ~~Chip~~ | ~~Marino@ourcampus.edu~~ | ~~Tran~~ | ~~Tran@ourcampus.edu~~ | ~~Info Systems~~ | ~~Rogers~~ |
| 8 | Thompson | James | Thompson@ourcampus.edu | Taing | Taing@ourcampus.edu | Accounting | Smith |
| 9 | ??? | ??? | ??? | ??? | ??? | Biology | Kelly |

Deleted row—Student, Adviser and Department data lost

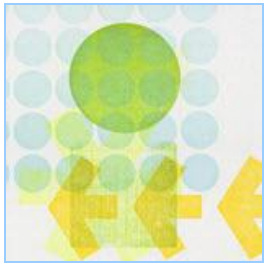
Inserted row—both Student and Adviser data missing

- Relational databases are designed to address many of the information complexity issues



# Relational Databases

- A **relational database** stores information in ***tables***. Each informational topic is stored in its own table
- In essence, a relational database will break-up a list into several parts. One part for each theme in the list
- A Project List would be divided into a CUSTOMER Table, a PROJECT Table, and a PROJECT_MANAGER Table

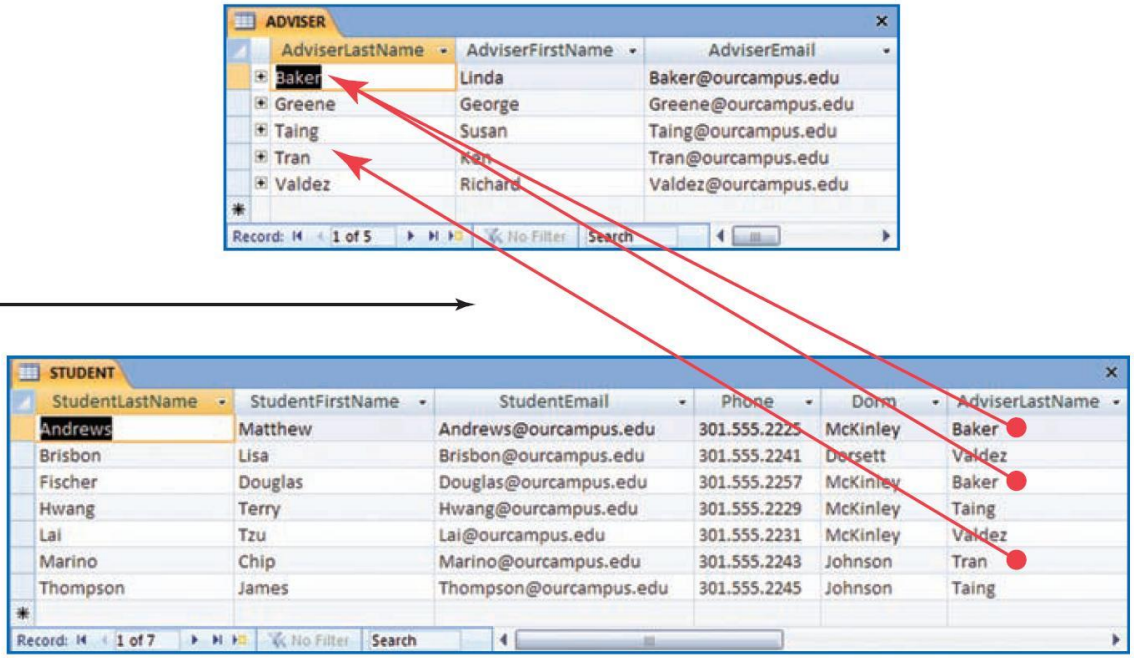


# Sounds like More Work, Not Less

- A relational database is more complicated than a list
- However, a relational database minimizes data redundancy, preserves complex relationships among topics, and allows for partial data
- Furthermore, a relational database provides a solid foundation for user forms and reports

# Relational Database Example

STUDENT data linked to ADVISER data via **AdviserLastName**



ADVISER

| AdviserLastName | AdviserFirstName | AdviserEmail |
|---|---|---|
| Baker | Linda | Baker@ourcampus.edu |
| Greene | George | Greene@ourcampus.edu |
| Taing | Susan | Taing@ourcampus.edu |
| Tran | Ken | Tran@ourcampus.edu |
| Valdez | Richard | Valdez@ourcampus.edu |

Record: 1 of 5 No Filter Search

STUDENT

| StudentLastName | StudentFirstName | StudentEmail | Phone | Dorm | AdviserLastName |
|---|---|---|---|---|---|
| Andrews | Matthew | Andrews@ourcampus.edu | 301.555.2225 | McKinley | Baker |
| Brisbon | Lisa | Brisbon@ourcampus.edu | 301.555.2241 | Dorsett | Valdez |
| Fischer | Douglas | Douglas@ourcampus.edu | 301.555.2257 | McKinley | Baker |
| Hwang | Terry | Hwang@ourcampus.edu | 301.555.2229 | McKinley | Taing |
| Lai | Tzu | Lai@ourcampus.edu | 301.555.2231 | McKinley | Valdez |
| Marino | Chip | Marino@ourcampus.edu | 301.555.2243 | Johnson | Tran |
| Thompson | James | Thompson@ourcampus.edu | 301.555.2245 | Johnson | Taing |

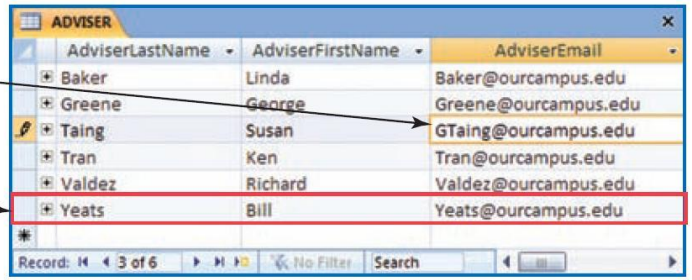
Record: 1 of 7 No Filter Search

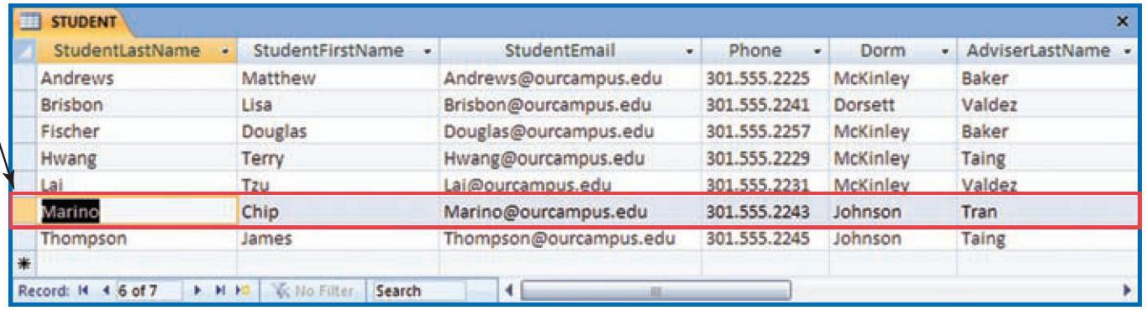
# A Relational Database Solves the Problems of Lists

Changed data—data remains consistent

Inserted data—no STUDENT data required

Deleted data—no ADVISOR data lost



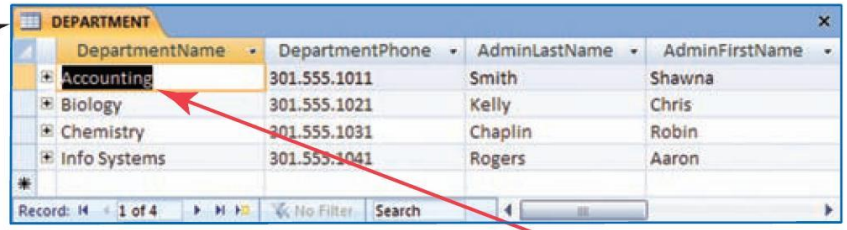



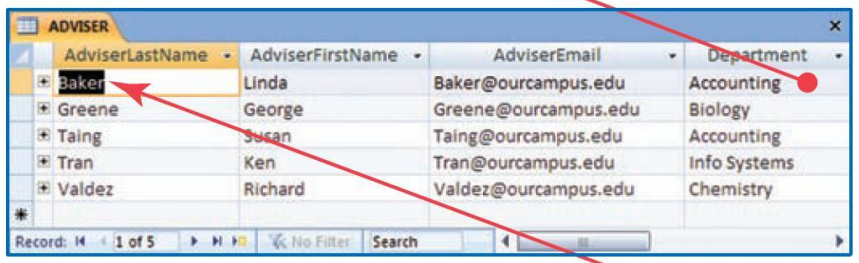
# The Department, Advisor and Student Tables

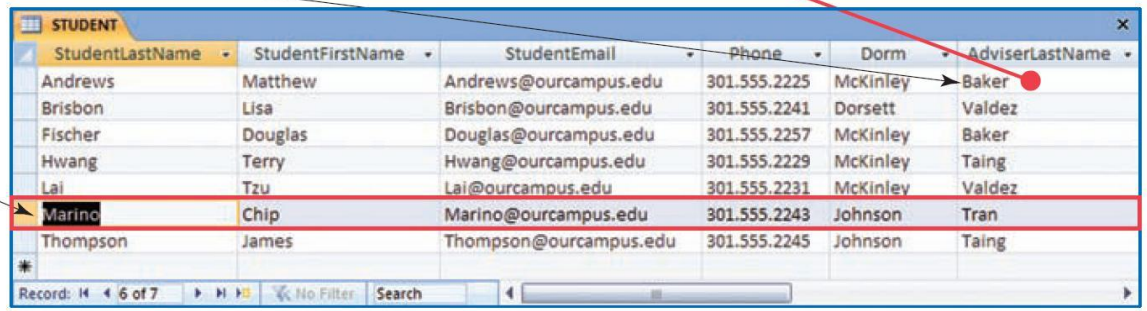
Can insert DEPARTMENT data as needed—no ADVISER or STUDENT data required

Can change STUDENT Adviser name as needed—new value is linked to its own data

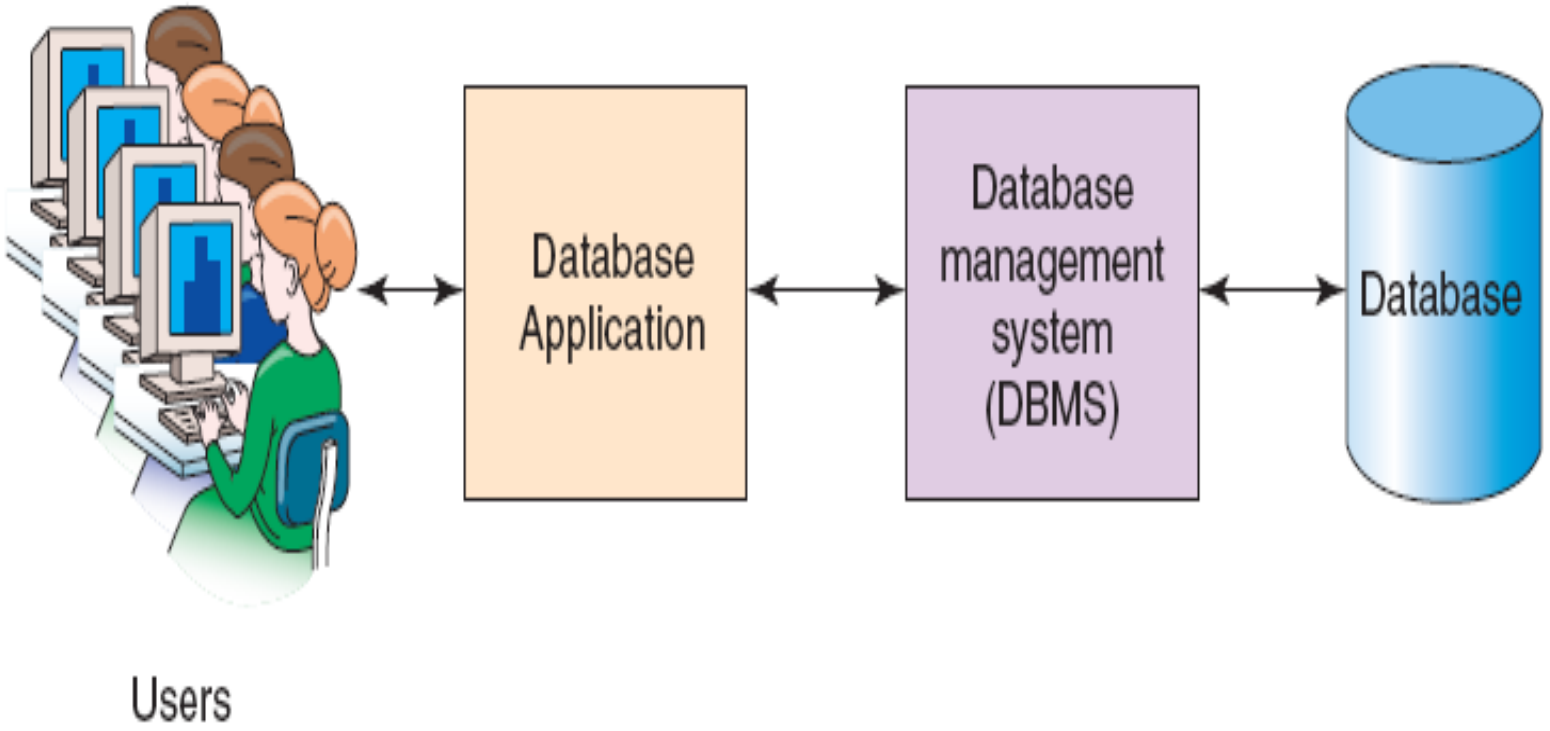
Can delete STUDENT data as needed—no DEPARTMENT or ADVISER data lost

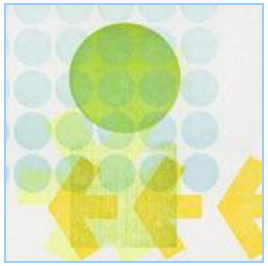


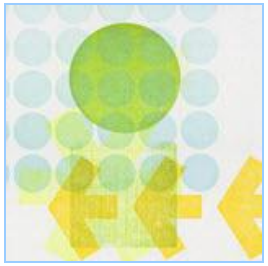
# Components of a Database System





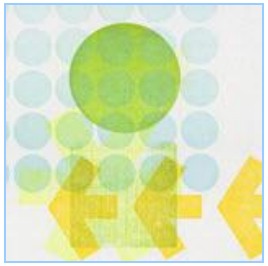
# Users

- A **user** of a database system will
  - Use a database application to track things
  - Use forms to enter, read, delete and query data
  - Produce reports



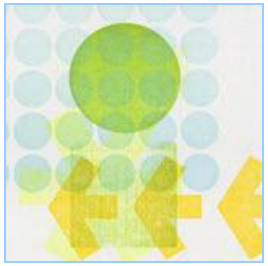
# The Database

- A database is a *self-describing* collection of *related* records
- Self-describing
  - The database itself contains the definition of its structure
  - Metadata is data describing the structure of the database data
- Tables within a relational database are related to each other



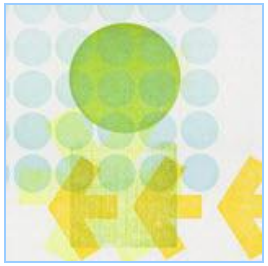
# Database Management System (DBMS)

- A **database management system (DBMS)** serves as an intermediary between database applications and the database
- The DBMS manages and controls database activities
- The DBMS creates, processes and administers the databases it controls



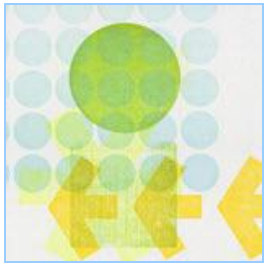
# Functions of a DBMS

- Create databases
- Create tables
- Create supporting structures
- Read database data
- Modify database data (insert, update, delete)
- Maintain database structures
- Enforce rules
- Control concurrency
- Provide security
- Perform backup and recovery



# Referential Integrity Constraints

- The DBMS will enforce many constraints
- **Referential integrity constraints** ensure that the values of a column in one table are valid based on the values in another table
  - If a 5 was entered as a CustomerID in the PROJECT table, a Customer having a CustomerID value of 5 must exist in the CUSTOMER table

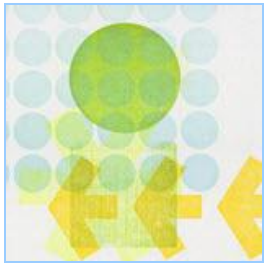


# Database Applications

- A **database application** is a set of one or more **computer programs** that serves as an intermediary between the **user** and the **DBMS**

  *Functions* :

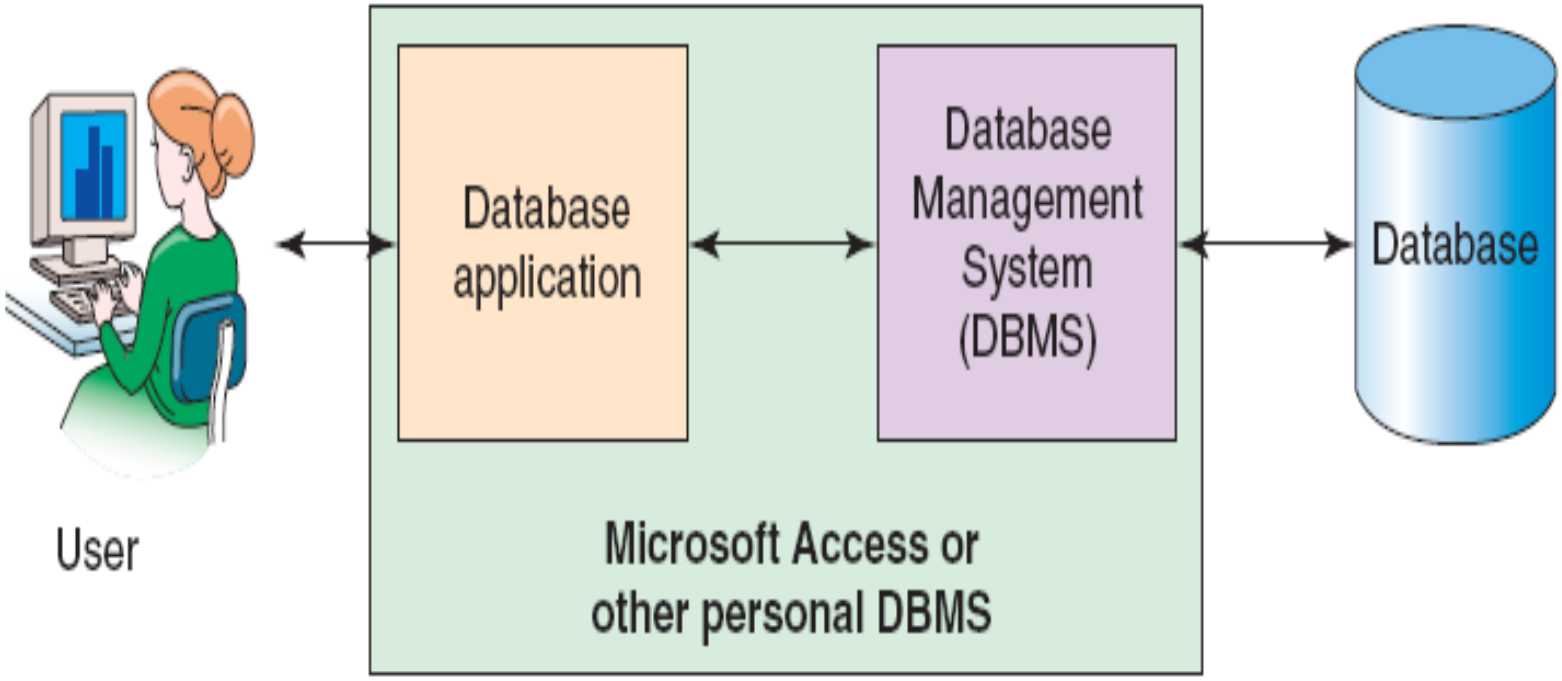
  - *Create and process forms*
  - *Process user queries*
  - *Create and process reports*
  - *Execute application logic*
  - *Control database applications*

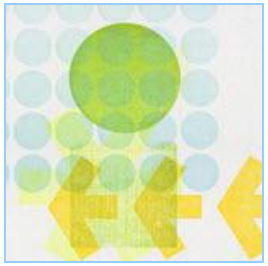


# Personal Database Systems

- Personal database systems typically:
  - Have one application
  - Have only a few tables
  - Are simple in design
  - Involve only one computer
  - Support one user at a time

# Personal Database Systems

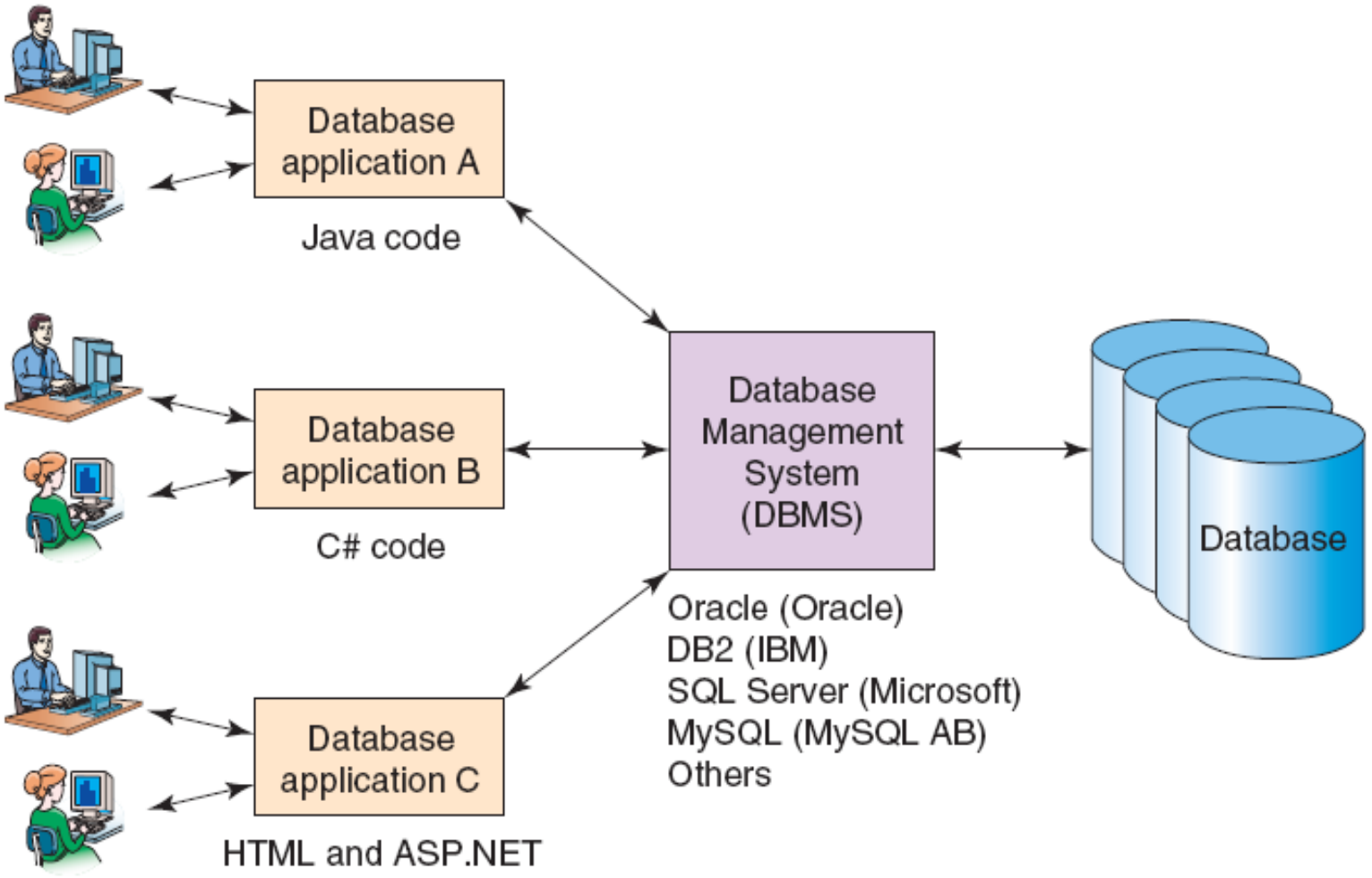


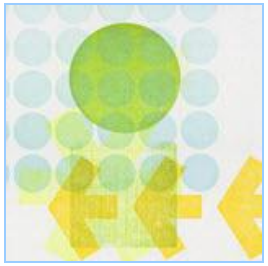


# Enterprise-class Database Systems

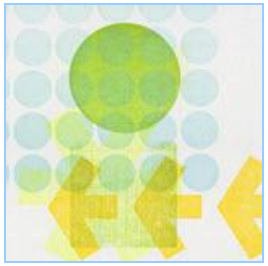
- Enterprise-Class database systems typically:
  - Support several users simultaneously
  - Include more than one application
  - Involve multiple computers
  - Are complex in design
  - Have many tables
  - Have many databases

# Organizational Database Systems



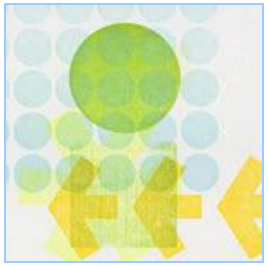


# Database Administration



# Database Processing Environment

- A database processing environment is complicated and multi-faceted
  - Multiple users
  - Multiple queries
  - Multiple forms
  - Multiple reports
  - Multiple application programs

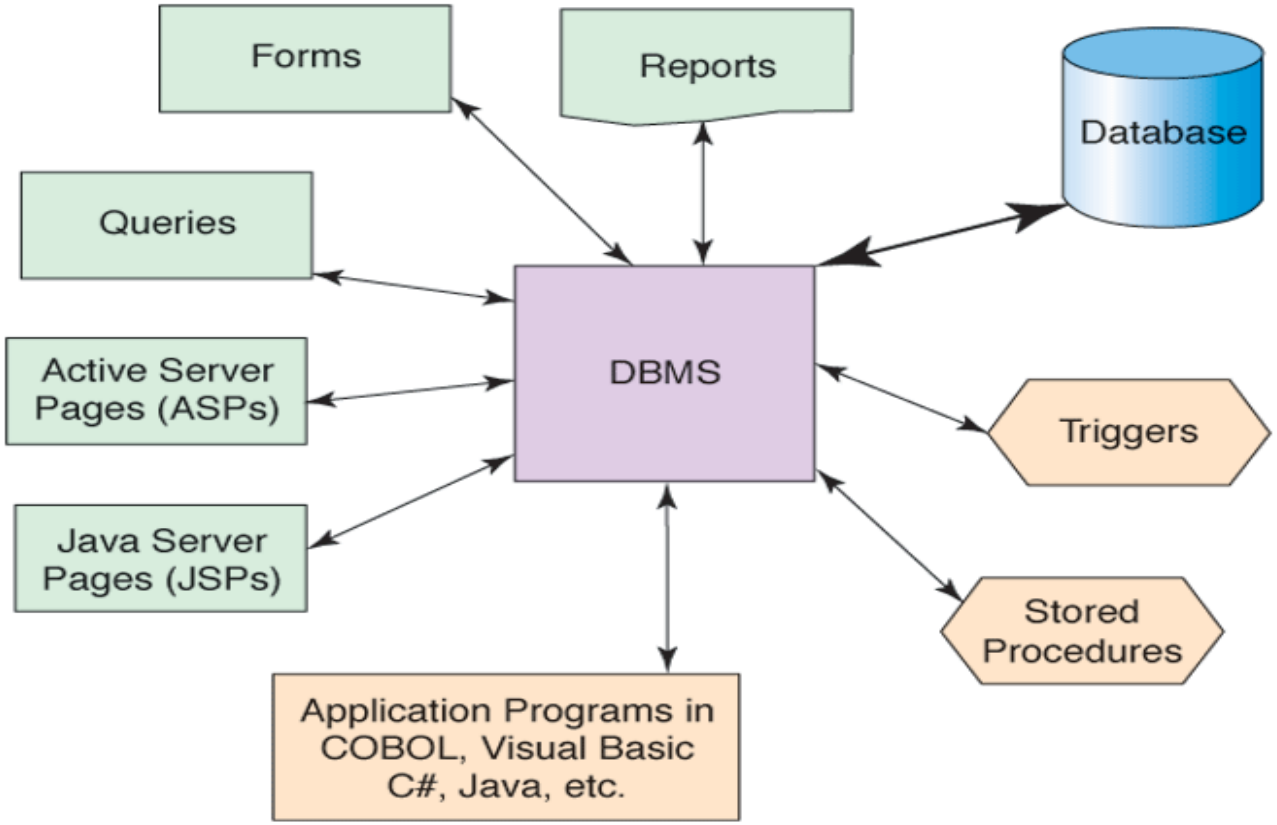


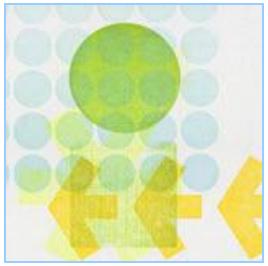
# Internet Application Processing

- Internet Application Processing is more complicated than traditional application processing
- Specifically, with Internet Application Processing …
  - The **network** becomes an integral part of the application

# The Database Processing Environment

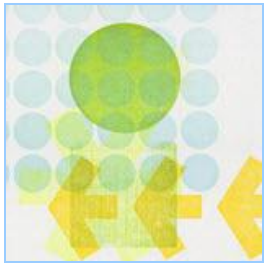
**Figure 6-1** Database Processing Environment





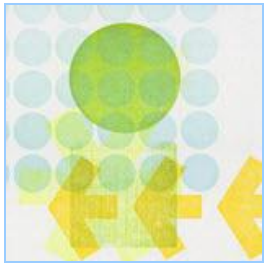
# Stored Procedures and Triggers

- A *stored procedure* is a module similar to subroutine or function that performs database actions
  - Stored in the database itself
- A *trigger* is a stored procedure that is automatically invoked by the DBMS when a specified activity occurs
  - BEFORE, AFTER and INSTEAD OF



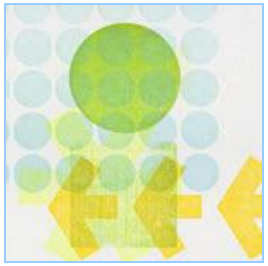
# Control, Security and Reliability

- Three necessary database administration functions
  - Concurrency control
  - Security
  - Backup and Recovery



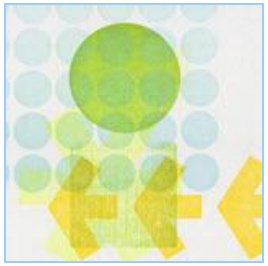
# Database Security

- Database Security strives to ensure:
  - Only authorized users
  - Perform authorized activities
  - At authorized times



# Database Security Guidelines

- Run the DBMS behind a firewall
- Apply the latest operating system and DBMS service packs and patches
- Limit DBMS functionality to needed features
- Protect the computer that runs the DBMS
- Manage accounts and passwords

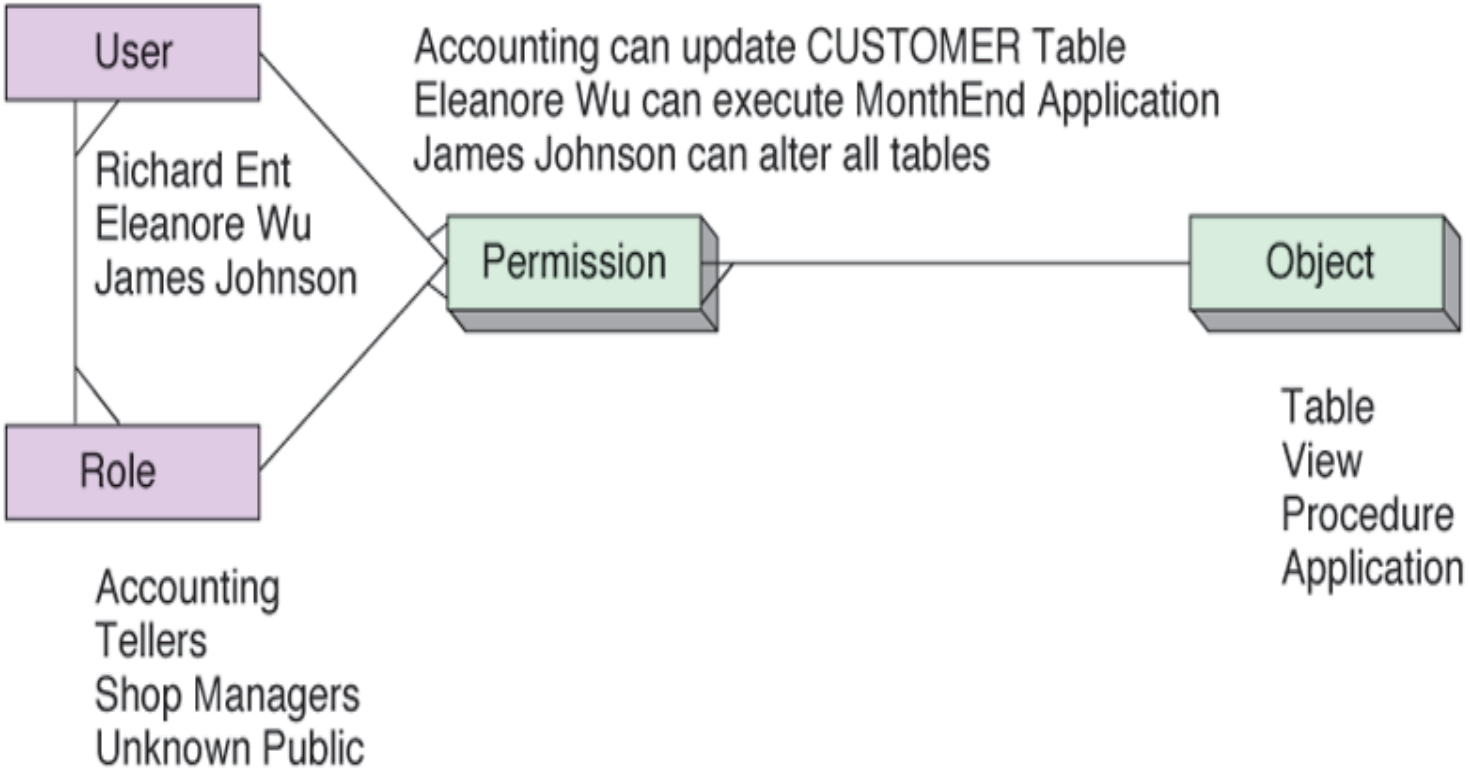


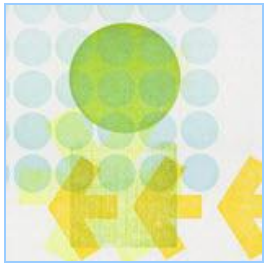
# Application Security

- Beyond providing generic access limitations to users, an application may introduce specific access rights for particular users.

# A Model of DBMS Security

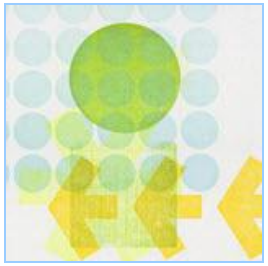
**Figure 6-13** A Model of DBMS Security





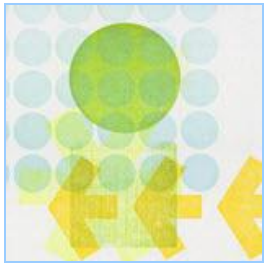
# Database Backup and Recovery

- Common causes of database failures…
  - Hardware failures
  - Programming bugs
  - Human errors/mistakes
  - Malicious actions
- Since these issues are impossible to completely avoid, recovery procedures are essential



# Data Models

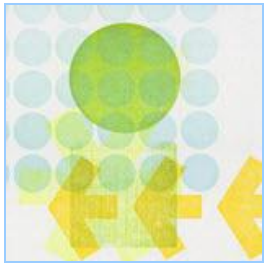
## *Data Modeling and ER*



# Data Models

- A collection of tools for describing
  - data
  - data relationships
  - data semantics
  - data constraints

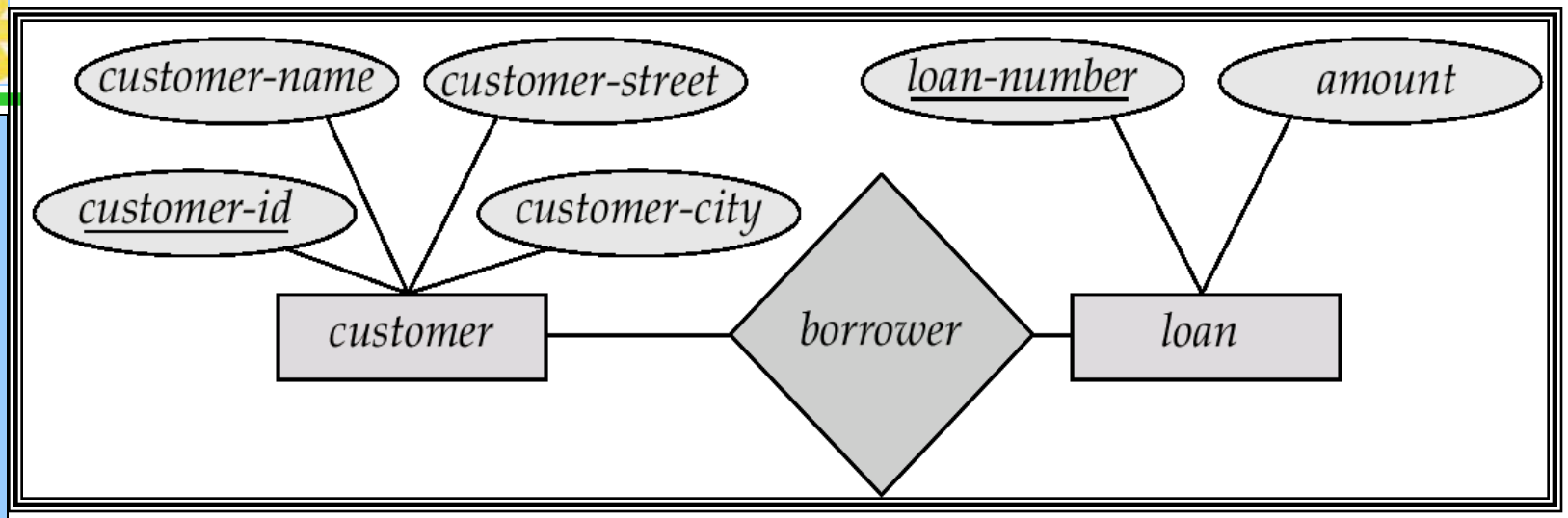
1. Entity-Relationship model (ER)
2. Relational model
3. Other models:
   A. object-oriented model
   B. semi-structured data models
   C. Older models: network model and hierarchical model



# Entity Relationship Model

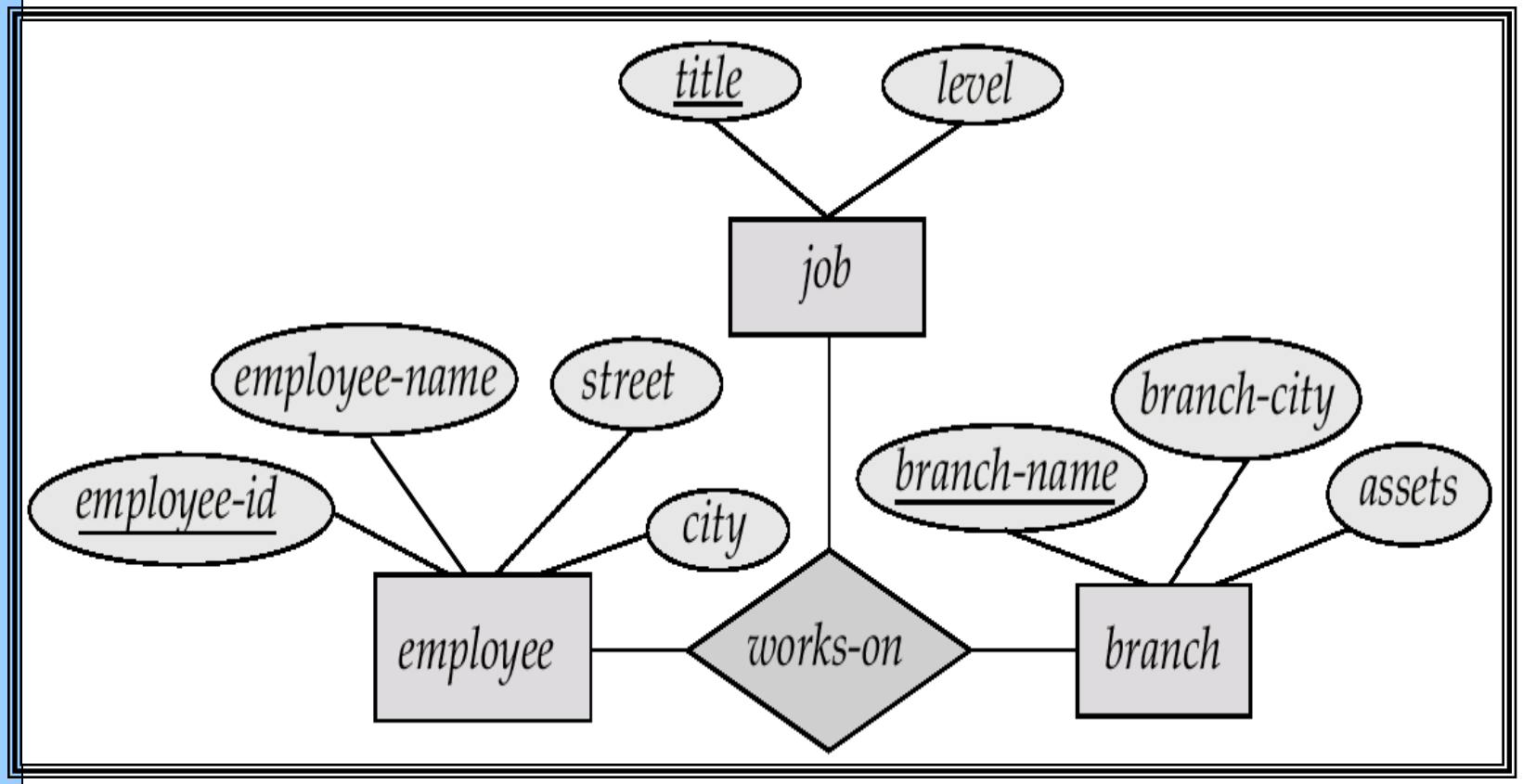
- E-R model of real world
  - **Entities** (objects)
    - E.g. customers, accounts, bank branch
  - **Relationships** between entities
    - E.g. Account A-101 is *held by* customer Johnson
    - Relationship set *depositor* associates customers with accounts
- Widely used for database design
  - Database design in **E-R model** usually converted to design in the **relational model** (coming up next) which is used for storage and processing

# E-R Diagrams

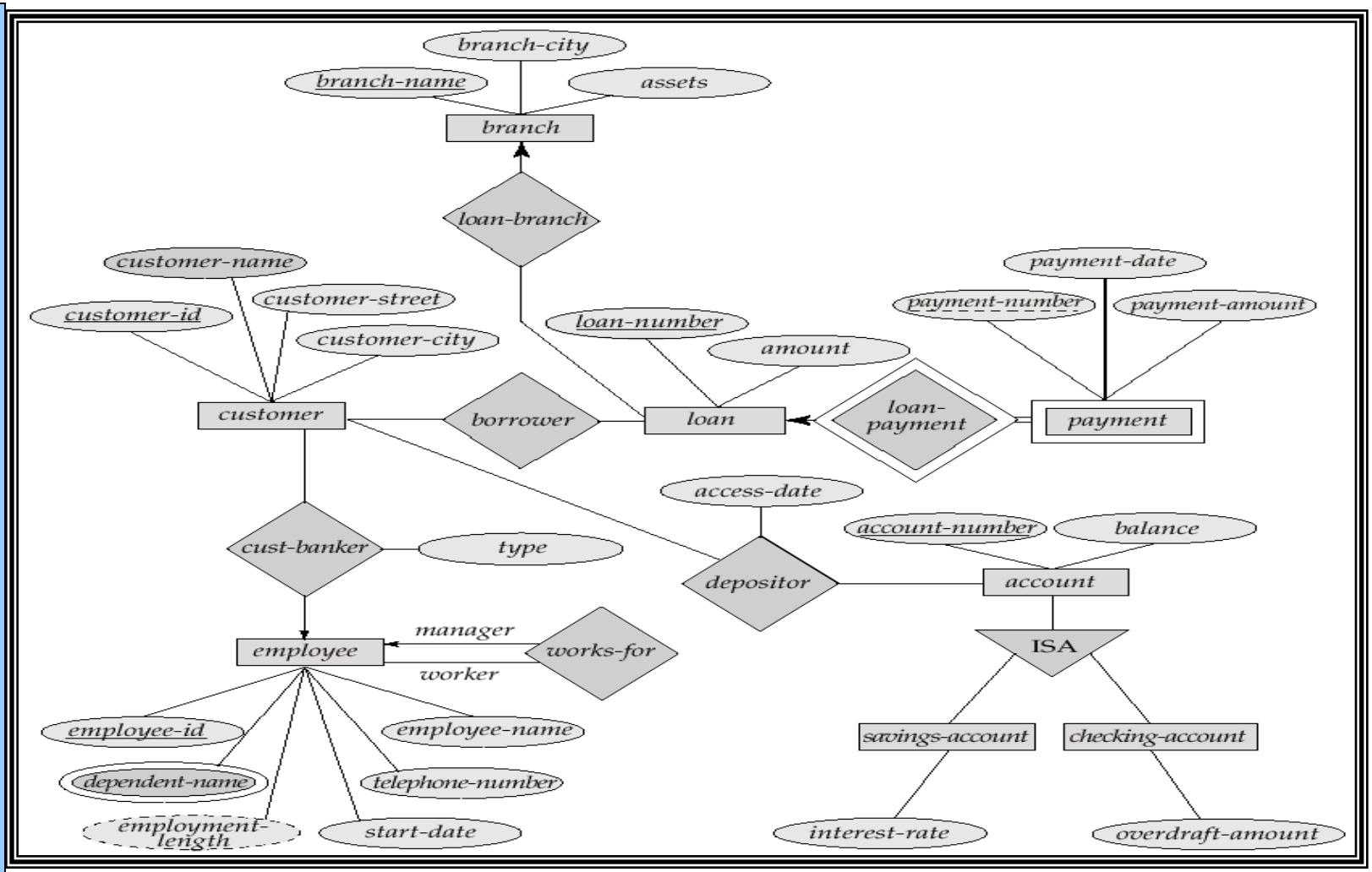



- **Rectangles** represent entity sets.
- **Diamonds** represent relationship sets.
- **Lines** link attributes to entity sets and entity sets to relationship sets.
- **Ellipses** represent attributes
  - **Double ellipses** represent multivalued attributes.
  - **Dashed ellipses** denote derived attributes.
- **Underline** indicates primary key attributes (will study later)

# E-R Diagram with a Ternary Relationship

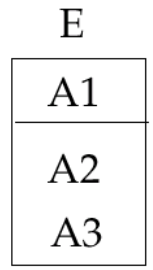


# E-R Diagram for a Banking Enterprise

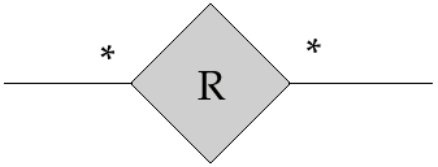


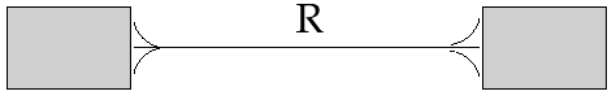
# Alternative E-R Notations

Entity set E with attributes A1, A2, A3 and primary key A1

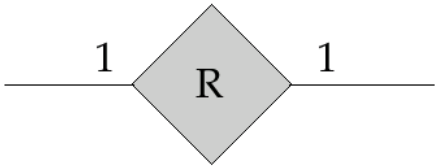


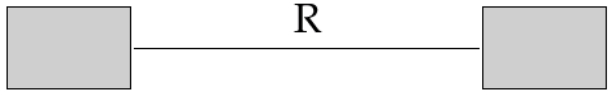

Many to Many Relationship



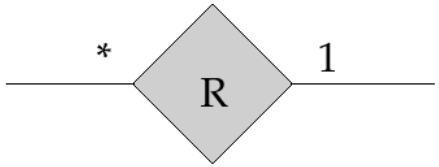



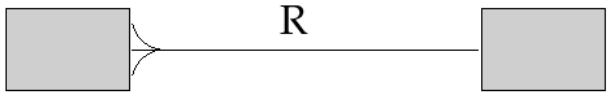

One to One Relationship

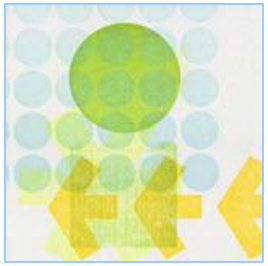



Many to One Relationship



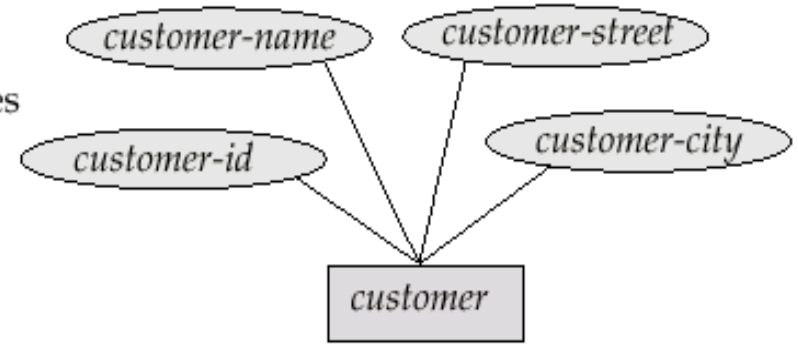


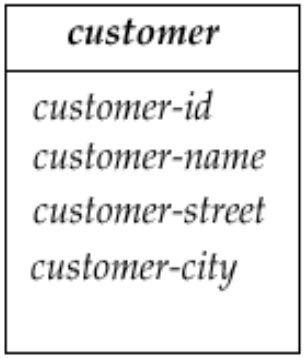
# UML

- UML: Unified Modeling Language
- UML has many components to graphically model different aspects of an entire software system
- UML Class Diagrams correspond to E-R Diagram, but several differences.

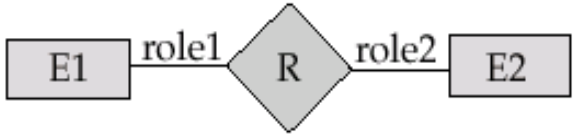
# Summary of UML Class Diagram Notation

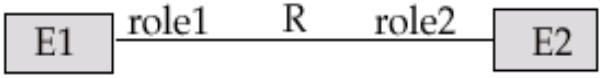
1. Entity sets and attributes

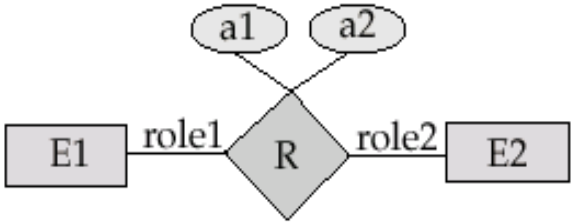


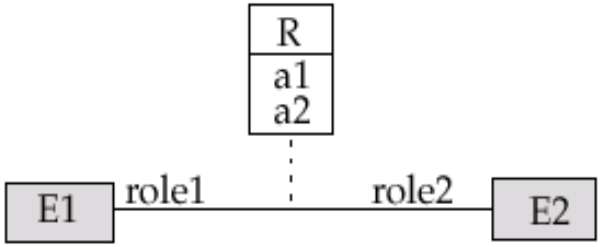




2. Relationships

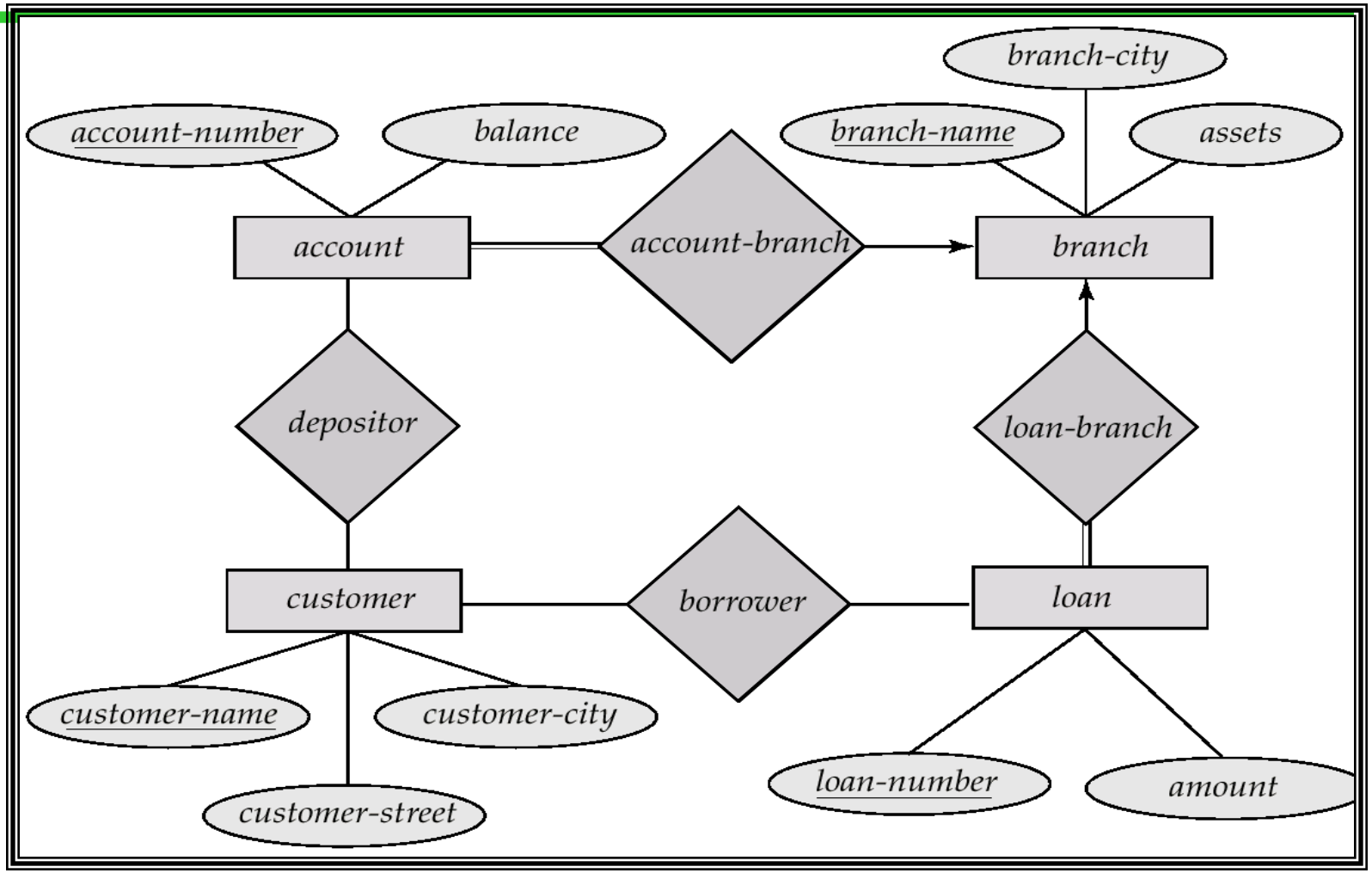


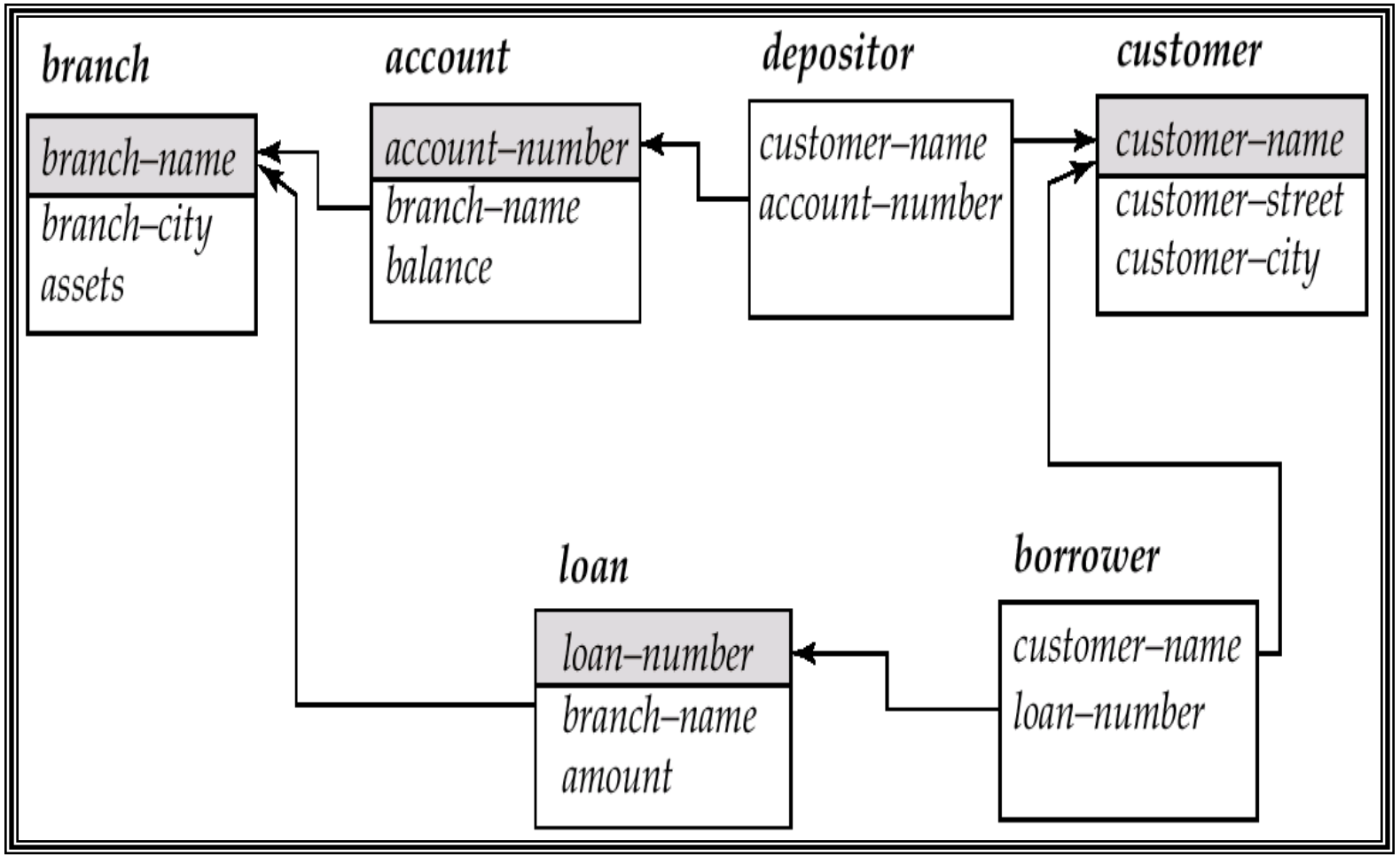



# E-R Diagram for the Banking Enterprise



# Schema Diagram for the Banking Enterprise

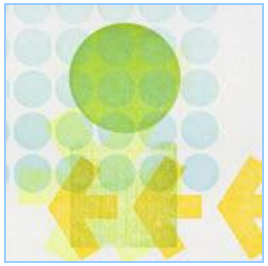


# Relational Model

- Example of tabular data in the relational

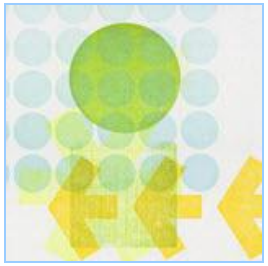
Attributes

| *Customer-id* | *customer-name* | *customer-street* | *customer-city* | *account-number* |
|---|---|---|---|---|
| 192-83-7465 | Johnson | Alma | Palo Alto | A-101 |
| 019-28-3746 | Smith | North | Rye | A-215 |
| 192-83-7465 | Johnson | Alma | Palo Alto | A-201 |
| 321-12-3123 | Jones | Main | Harrison | A-217 |
| 019-28-3746 | Smith | North | Rye | A-201 |



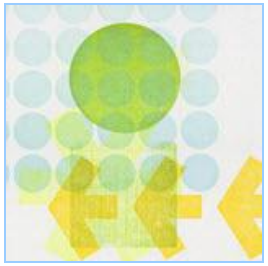
# Structured Query Language

**SQL**



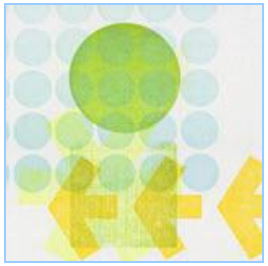
# Structured Query Language

- **Structured Query Language**
  - Acronym: SQL
  - Pronounced as “S-Q-L” [“Ess-Que-El”]
  - Originally developed by IBM as the SEQUEL language in the 1970s
  - SQL-92 is an ANSI national standard adopted in 1992
  - SQL:2008 is current standard



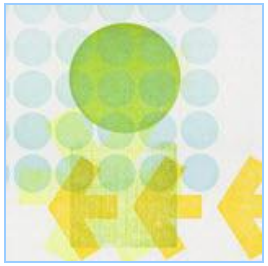
# SQL Defined

- SQL is not a programming language, but rather a data sub-language
- SQL is comprised of
  - A data definition language (DDL)
    - Used to define database structures
  - A data manipulation language (DML)
    - Data definition and updating
    - Data retrieval (Queries)
  - There are other SQL functions not covered in this chapter
    - Concurrency control [See Chapter 6]
    - Transaction control [See Chapter 6]



# SQL for Data Definition

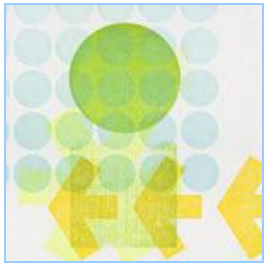
- The SQL data definition statements include
  - CREATE
    - To create database objects
  - ALTER
    - To modify the structure and/or characteristics of database objects
  - DROP
    - To delete database objects



# SQL for Data Definition: CREATE

- Creating database tables
  - The SQL CREATE TABLE statement

```
CREATE TABLE EMPLOYEE(
   EmpID        Integer        PRIMARY KEY,
   EmpName      Char(25)       NOT NULL
   );
```

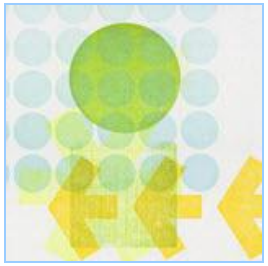


# Adding Data: INSERT

- To add a row to an existing table, use the INSERT statement
- Non-numeric data must be enclosed in straight ( ' ) quotes

```
INSERT INTO EMPLOYEE VALUES(91, 'Smither', 12);

INSERT INTO EMPLOYEE (EmpID, SalaryCode)
       VALUES (62, 11);
```



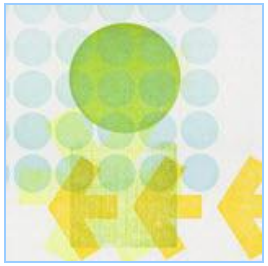
# SQL for Data Retrieval: Queries

- SELECT is the best known SQL statement
- SELECT will retrieve information from the database that matches the specified criteria using the SELECT/FROM/WHERE framework

```
SELECT EmpName
FROM   EMPLOYEE
WHERE  EmpID = 2010001;
```

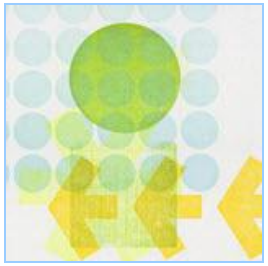
# SQL for Data Retrieval: The Results of a Query is a Relation

- A query pulls information from one or more relations and creates (temporarily) a new relation
- This allows a query to:
  - Create a new relation
  - Feed information to another query (as a "sub-query")



# Modifying Data using SQL

- Insert
  - Will add a new row in a table (already discussed above)
- Update
  - Will update the data in a table that matches the specified criteria
- Delete
  - Will delete the data in a table that matches the specified criteria



# Thank You