# Introduction

# Database Management Systems

# Database Management System (DBMS)

- Collection of
  - interrelated data and
  - set of programs to access the data
- Convenient and efficient processing of data
- Database Application Software

# Evolution of Database Systems

- Early days: customized applications built on top of file systems
- Drawbacks of using file systems to store data:
  - Data redundancy and inconsistency
  - Difficulty in accessing data
  - Atomicity of updates
  - Concurrency control
  - Security
  - Data isolation — multiple files and formats
  - Integrity problems

# Abstraction

- View level: different perspectives
  - application programs hide irrelevant data
- Logical level: data models
  - Logical representation of data
  - Different approaches: relational, hierarchical, network, object oriented, semi-structured, etc.
  - **Data independence principle**
- Physical level: how data is stored

# Data Models

- A collection of tools for describing
  - Data
  - Relationships among data items
  - Semantics of stored data
  - Database constraints
- Entity-Relational Model
- UML
- Etc.

# Database Management Systems

- Smaller and smaller systems
  - Past: large and expensive DBMS
  - Present: DBMS in most personal computers
- More and more data stored
  - Past: few MB
  - Present: terabyte ($10^{12}$ bytes), petabyte ($10^{15}$ bytes)
- Functionality: from physical to view level
- Optimization

# Data Definition Language (DDL)

- Defines the database schema and constraints
- DDL compiler → data dictionary
- Metadata – data about data

# Data Manipulation Language (DML)

- Accessing and manipulating the data
- Query Languages
  - Procedural – user specifies what data is required and how to get those data
  - Nonprocedural – user specifies what data is required without specifying how to get those data

# MODELING DATA SEMANTICS

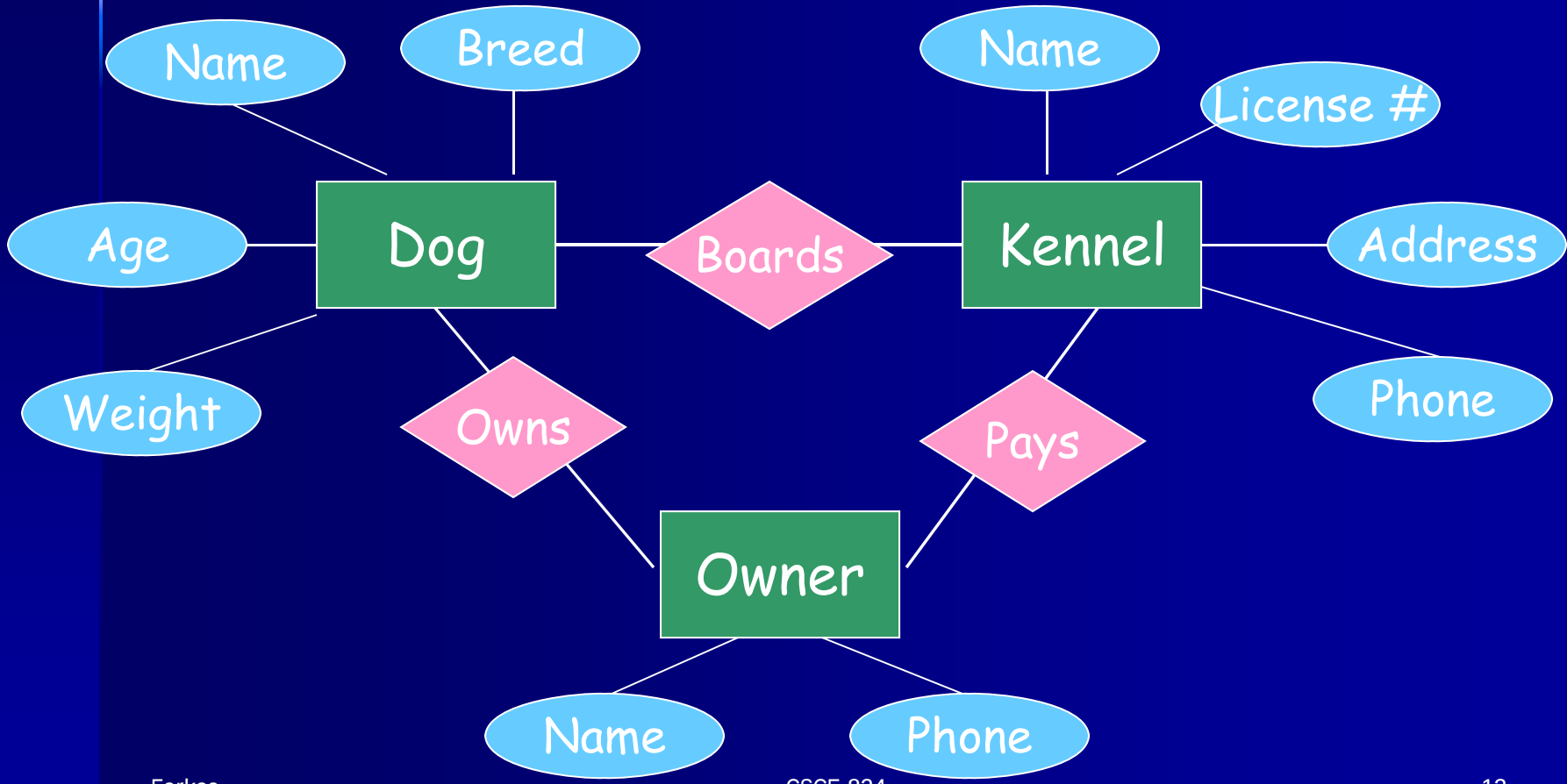
# Entity Sets and Relationship Sets

- Database: collection of entities and relationship among entities
- Entity: object that exists and distinguishable from other objects
- Entity set: collection of similar objects
- Attribute: property of an entity and relationship sets

# Attributes

- Domain: set of permitted values for each attributes
- Attribute types:
  - Simple vs. composite
  - Single-valued vs. multi-valued
  - Derived

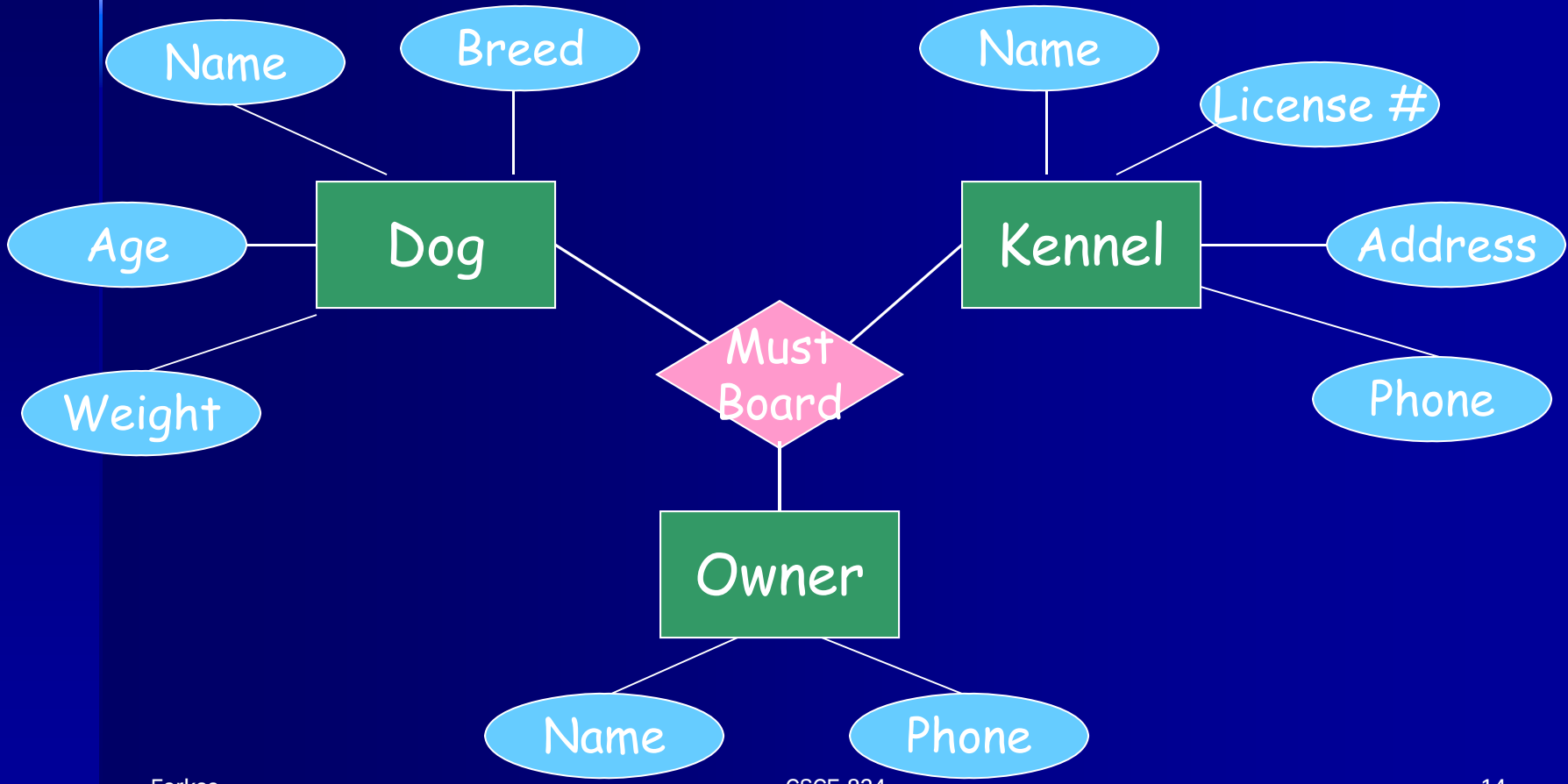
# Example E/R Diagram



# Degree of Relationship Sets

- Number of entity sets participating in a relationship set
- Binary relationship set: two entity sets (most common)
- Multiway relationship set: connects more than two entity sets
- E.g., An owner frequents certain kennels for certain dogs
  - Binary relationship can't represent these requirements
  - Need 3-way relationship

# Example 3-Way Diagram



# Mapping Cardinality

- Number of entities to which another entity can be associated via a relationship set
  - One-one
  - Many-one (One-many)
  - Many-many

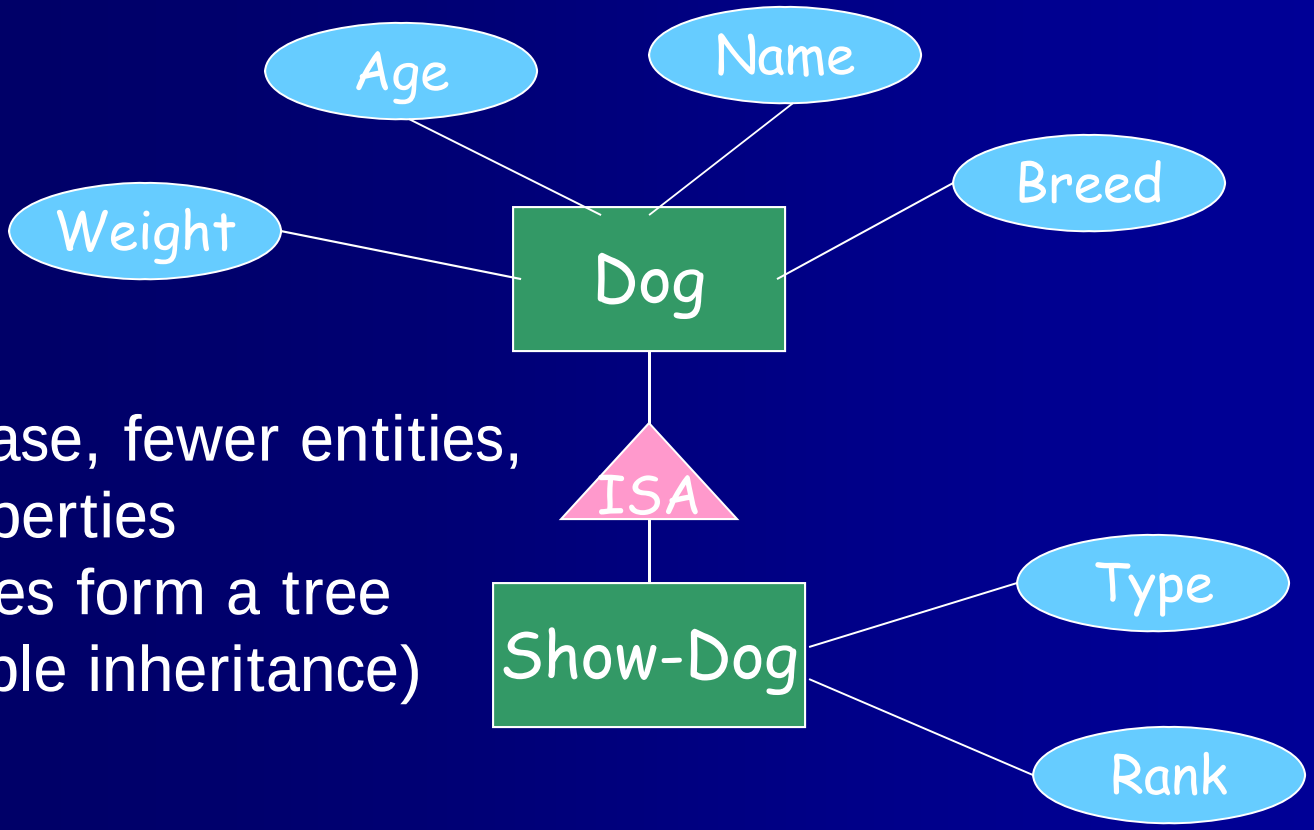
# Roles

- Entity set may appear more then once in a relationship
- Label the edges between the relationships and the entity set with names called *roles*.
- E.g., relationships among the dogs:

# Subclasses in E/R

- Special case, fewer entities, more properties
- E.g., show dog is a dog, but not all dogs are show dogs. It also have properties, type of competition, rank, etc.
- Assume subclasses form a tree (no multiple inheritance)
- ISA relationship

# Example Subclass




- Special case, fewer entities, more properties
- Subclasses form a tree (no multiple inheritance)

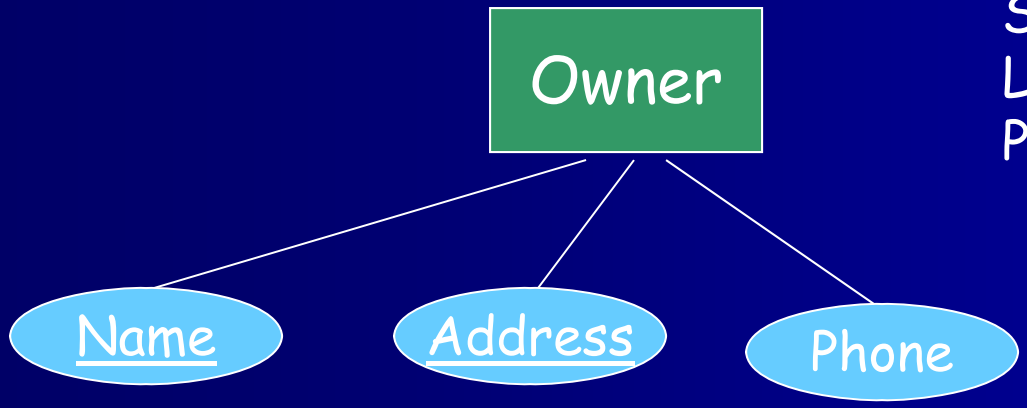
# Keys

- Set of attributes for one entity set such that no two entities in the set has the same value for **all the attributes** of the key
- Each entity set must have a key

# Keys

- Super key: set of one or more attributes whose value iniquely determine each entity
- Candidate key: minimal super key
- Primary key: a selected candidate key

# Example Multi-Attribute Key for Owner




There are more than One person with the Same name, but they Live at different Places.

# Converting ER Model into Relations

# Example Relation

Dog

| Name | Age | Weight (lb) | Breed |
|---|---|---|---|
| Pepper | 10 | 75 | German Shepherd |
| Joker | 4 | 83 | Mix |
| Bruno | null | 51 | Boxer |

# Relational Data Model

- Set of relation names: $\mathbb{R}$
- Set of attribute names: $\mathbb{A}$
- Relation schema: $S=(r,\{a_1, \ldots,a_n\})$
  - r relation name in $\mathbb{R}$
  - $\{a_1, \ldots,a_n\}$ subset of $\mathbb{A}$

  e.g., (*Dog*,{Name, Age, Weight, Breed})

# Relational Data Model

- Tuple over a relation scheme S is a mapping

  t: $\{a_1, \ldots, a_n\} \rightarrow dom(a_1 \cup \ldots \cup a_n)$

  e.g., t(Dog-Name)=Pepper
  t(Dog-Age)=10
  t(Dog-Weight)= 75
  t(Dog-Breed)=German Shepherd

# Relational Data Model

- Relation over schema S is a set of tuples over the scheme
- Database: set of relations

# Query Languages

- Relational Algebra
  - Set operations
- SQL
  - Bag operations

# Relational Algebra

- **Select** ($\sigma$)
- **Project** ($\Pi$)
- **Set difference** (-)
- **Union** ($\cup$)
- **Rename** ($\rho x(r)$)
- **Set intersection** ($\cap$)
- **Natural join** ($\otimes$)

# Structured Query Language SQL

- Typical SQL query form:

  **SELECT** $A_1, A_2, \ldots, A_n$
  **FROM** $r_1, r_2, \ldots, r_m$
  **WHERE** $C$

  – $A_i$s represent attributes to be returned
  – $r_i$s represent relations
  – $C$ is a condition

# Next Class
# Overview of Information Security
# (from CSCE 522 slides)