Unit 1: Entity–Relationship Model and its Relational Implementation
Purpose of the ER Model and Basic Concepts

The purpose of the Entity–Relationship model, referred further to, as *ER model* is to provide a common, informal, and convenient model for communication between users and the DBA for the purpose of modeling the data’s structure.

The ER model, frequently employs *ER diagrams* which visually present the relations between various data items. Many tools exist that will translate ER diagrams into create statements including referential integrity constraints.
Entity

• This is a “thing” that is distinguished from others in our application
  – Example: John

• All entities of the same “type” form an entity set
  – Example: PERSON (actually a set of persons)
  – We frequently say “entity” while actually referring to entity sets

• In an ER diagram, an entity set is depicted by a rectangle with its type written inside
**Relationship**

- A *relationship* between entity-sets $A$ and $B$ is an association between the members of $A$ and the members of $B$
  - Example: If we are given entities: PERSON, CITY, then consider the relationship LIKES, where a person and a city are related just if the person likes the city
- Visualize as a graph with edges going from $A$ to $B$
- In ER diagram, depicted by a diamond with its type written inside
Attribute

• This is a property of an entity

• Example: If we are given an entity such as a person, it could have attributes: name, phone, age, ssn (social security number), etc.

• In ER diagram, depicted as an oval attached to the associated rectangle
**Keys**

- A *key* for an entity-set is a subset of its attributes such that:
  - The values for the attributes in the key identify a *unique* entity from the entity set
  - The set of attributes does not contain unnecessary attributes – it is minimal

- Example: `{ssn}` is a key for PERSON

- Example: `{name, phone, birthtime}` is a key for PERSON
ER diagrams

- To show which entities participate in which relationships, and which attributes belong to which entities, we draw line segments between:
  - Entities and relationships they participate in
  - Attributes and entities they belong to

- We also underline the elements of some chosen key of each entity-set
Can Relationships have Attributes?

- Consider the relationship
  \[\text{BUYS(PERSON,CAR,PRICE)}\]
- PERSON and CAR are entity sets
- PRICE can either be considered as:
  - An entity set that consists of numbers
  - An attribute attached to the relationship PRICE
The Degree of a Relationship

- A relationship is *binary* if it connects two entity-sets
  - Example: LIKES(PERSON,CITY)
  - Example: LIKES(PERSON,PERSON)

- Example of a trinary relationship:
  SELL(COMPANY,PRODUCT,CUSTOMER)
**Types of Binary Relationships**

- The relationship $R(A,B)$ is said to be *many-to-one* from $A$ to $B$ if each element of $A$ is related to at most one element of $B$
  - $\text{FATHER-OF}(\text{PERSON},\text{MAN})$ is many-to-one from $\text{PERSON}$ to $\text{FATHER}$

- $R(A,B)$ is *one-to-one* if each element of $A$ is related to at most one element of $B$, and vice-versa
  - $\text{SPOUSE}(\text{PERSON},\text{PERSON})$ is one-to-one

- The general case of $R(A,B)$ is called *many-to-many*
  - $\text{LIKES}(\text{PERSON},\text{CITY})$ is many-to-many
Effect of the Type of Binary Relationship on the Table Design

- We designate many-to-one, and one-to-one, relationships with arrows

- Compare the following ER diagrams:
  - LIKES(PERSON, CITY)
  - FAVORITE-CITY(PERSON, CITY)

- Since LIKES is many-to-many, a separate table is needed for LIKES

- Since FAVORITE-CITY is many-to-one, it can be combined with the PERSON table
Many-To-One Relationships are a Special Kind of Functional Dependency

- Suppose \( R(A,B) \) is many-to-one from \( A \) to \( B \)
- Then \( R \) is a (partial) function from \( A \) to \( B \)
- This functional dependency can be written \( A \rightarrow B \)
- Example of a functional dependency for a non-binary relationship: in \( \text{MAJOR(\text{STUDENT}, \text{UNIVERSITY}, \text{SUBJECT})} \), we have \( \text{STUDENT,UNIVERSITY} \rightarrow \text{SUBJECT} \)
The ISA relationship

• Sometimes it is convenient to divide an entity set into several entity sets (into subsets). This can be done with a relationship ISA, which stands for “is a.”

• ISA is a one-to-one relationship between an entity-set and a containing superset

• Examples:
  – ISA(PROGRAMMER, EMPLOYEE)
  – ISA(SECRETARY, EMPLOYEE)
FROM ER diagram to Relational Design

• Entities correspond directly to Relations with primary keys being transformed into key constraints

• Entity E(A1,A2,A3) really stands for a relation E with three attributes A1,A2,A3

• Relationship R(E1,E2) stands for a relation whose schema contains all of the attributes in the keys of E1 and E2

• If R is many-to-one from E1 to E2, then R can be eliminated by augmenting the relation for E1 with the key attributes of E2
APPLICATION

• Consider the entity set of EMPLOYEE with attributes, EID, SALARY

• An employee is either a secretary or a programmer

• There are differences about the type of information we need about these two types of employees:
  – Secretaries have an attribute: TYPING_SPEED
  – Programmers have a relationship:
    KNOWS(PROGRAMMER, LANGUAGE)

• All employees are assigned to a LOCATION
ER Diagram for the Application
Relational Design for the Application

Here is a schema that implements the ER diagram:

- EMPLOYEE(EID, SALARY, LOCATION)
- PROGRAMMER(EID)
- KNOWS(EID, LANGUAGE)
- SECRETARY(EID, TYPING_SPEED)
Conclusions

• The ER model gives a good first cut at design

• Intuitive

• Can help design by suppressing unnecessary detail (i.e. attributes)

• Easily convertible into actual database models
Topics covered in this Unit:

1. ER diagrams
2. Entities
3. Relationships
4. Attributes