Database Design Process

- **Miniworld**
  - Requirements Collection and Analysis
    - Functional Requirements
      - Functional Analysis
        - High-level Transaction Specification
          - DBMS-independent
          - DBMS-specific
    - Database Requirements
      - Conceptual Design
        - Conceptual Schema (In a high-level data model)
      - Logical Design (Data Model Mapping)
        - Logical (Conceptual) Schema (In the data model of a specific DBMS)
      - Application Program Design
        - Internal Schema (For the same DBMS)
  - Transaction Implementation
    - Application Programs
DATA MODELS

Two data models are used commonly in the database design process:

- The Entity-Relational data model as a high-level conceptual data model, and
- The Relational data model as an implementation data model.

A database schema in the ER model
A database schema in the Relational model

EMPLOYEE
- FNAME
- MINIT
- LNAME
- SSN
- BDATE
- ADDRESS
- SEX
- SALARY
- SUPERSSN
- DNO

DEPARTMENT
- DNAME
- DNUMBER
- MGRSSN
- MGRSTARTDATE

DEPT_LOCATIONS
- DNUMBER
- DLOCATION

PROJECT
- PNAME
- PNUMBER
- PLOCATION
- DNUM

WORKS_ON
- ESSN
- PNO
- HOURS

DEPENDENT
- ESSN
- DEPENDENT NAME
- SEX
- BDATE
- RELATIONSHIP
Mapping from ER schema into Relational schema
Entity-Relationship Data Model

Case Study
Consider the following data requirements for a COMPANY database:

- The company database keeps track of a company’s employees, departments, and projects. The following data requirements are collected:
  
  - The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. The database keeps track of the start date when that employee began managing the department. A department may have several locations.
  
  - A department controls a number of projects, each of which has unique name, a unique number, and a single location.
  
  - The database stores each employee’s name, social security number, address, salary, sex, and birthdate. An employee is assigned to one department but may work on several projects, which are not necessarily controlled by the same department. The database keeps track of the number of hours per week that an employee works on each project. The database also keeps track of the direct supervisor of each employee.
  
  - The database keeps track of the dependents of each employee for insurance purposes by storing each dependent’s name, sex, birthdate, and relationship to the employee.
ER schema diagram for the company database
Entity Type Concept

An Entity:
An entity is a “thing” in the real world with an independent existence. That existence may be either:
Physical existence, such as, a particular person, car, house, or employee, or
Conceptual existence, such as a company, a job, or a university.

Each entity has particular properties, called attributes, that describe it. For example, an employee entity may be described by the employee’s name, age, address, salary, and job. A particular entity will have a value for each of its attributes.

Examples:

<table>
<thead>
<tr>
<th>e1</th>
<th>Name</th>
<th>Adel Sadek</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sex</td>
<td>Male</td>
</tr>
<tr>
<td></td>
<td>Age</td>
<td>45</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>e2</th>
<th>Name</th>
<th>Aly Sherif</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sex</td>
<td>Male</td>
</tr>
<tr>
<td></td>
<td>Age</td>
<td>55</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>d1</th>
<th>Name</th>
<th>Transport</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Manager</td>
<td>Aly Sherif</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>d2</th>
<th>Name</th>
<th>Research</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Manager</td>
<td>Adel Sadek</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>p1</th>
<th>Name</th>
<th>ProjectX</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Location</td>
<td>Toshkie</td>
</tr>
</tbody>
</table>

An Entity Type:

An entity type is the set of entities that have the same attributes.

In the company database, we may distinguish the following entity types:

Employee - Department – Project - Dependent

In another database, such as a database for a university, we may distinguish some entity types such as:

College – Department – Student – Course - Teacher
Types of Attributes

- **Simple (atomic) Attribute** is an attribute which is not divisible.
- **Composite Attribute** is an attribute which can be divided into smaller subparts, which represent more basic attributes with independent meaning of their own.
- **Single-valued Attribute** is an attribute which will have a single value for a particular entity.
- **Multivalued Attribute** is an attribute which can have several values for a particular entity.

- **Stored Attribute** is an attribute whose values for different entities are stored directly in the database.
- **Derived Attribute** is an attribute whose value for a particular entity can be computed or derived from the value of one or more stored attributes for the same entity.

**Null Attribute**: In some cases a particular entity may not have any applicable value for an attribute, modified. Null can be used in two different situations: not applicable or unknown facts.

**Key Attribute of an Entity Type**
An entity type usually has an attribute whose values are distinct for each individual entity. Such an attribute - satisfying uniqueness constraint - is called a key attribute, and its values can be used to identify each entity uniquely.

**Modelling in ER model**

```
Phase 1
Initial Model

Entity Types

Phase 2
Final Model

Relationship Types

Entity Types
```
DEPARTMENT
Name, Number, {Location}, Manager, ManagerStartDate

PROJECT
Name, Number, Location, ControllingDepartment

EMPLOYEE
Name (Fname, Minit, Lname), SSN, Sex, Address, Salary, BirthDate,
Department, Supervisor,
{WorksOn(Project, Hours)}

DEPENDENT
Employee, DependentName, Sex, Relationship
ER Diagrams

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTITY TYPE</td>
<td>E1</td>
</tr>
<tr>
<td>WEAK ENTITY TYPE</td>
<td>R</td>
</tr>
<tr>
<td>RELATIONSHIP TYPE</td>
<td>TOTAL PARTICIPATION OF E2 IN R</td>
</tr>
<tr>
<td>IDENTIFYING RELATIONSHIP TYPE</td>
<td>CARDINALITY RATIO 1:N FOR E1:E2 IN R</td>
</tr>
<tr>
<td>ATTRIBUTE</td>
<td>STRUCTURAL CONSTRAINT (min, max) ON PARTICIPATION OF E IN R</td>
</tr>
<tr>
<td>KEY ATTRIBUTE</td>
<td>(min, max)</td>
</tr>
<tr>
<td>MULTIVALUED ATTRIBUTE</td>
<td></td>
</tr>
<tr>
<td>COMPOSITE ATTRIBUTE</td>
<td></td>
</tr>
<tr>
<td>DERIVED ATTRIBUTE</td>
<td></td>
</tr>
</tbody>
</table>

Initial ER diagram for the company database.
Relationship Type Concept

A relationship type $R$ among $n$ entity types $E_1, E_2, ..., E_n$ defines a set of associations among entities from these entity types. Mathematically, $R$ is a set of relationship instances $r_i$, where each $r_i$ associates $n$ entities $(e_1, e_2, ..., e_n)$, and each entity $e_j$ in $r_i$ is a member of entity type $E_j$, $1 \leq j \leq n$. Each of the entity types $E_1, E_2, ..., E_n$ is said to participate in the relationship $R$, and similarly, each of the individual entities $e_1, e_2, ..., e_n$ is said to participate in the relationship instance $r_i = (e_1, e_2, ..., e_n)$.
Some instances of the WORKS_FOR relationship

Degree of a Relationship Type

The degree of a relationship type is the number of participating entity types. Hence, the WORKS_FOR relationship type is of degree two. A relationship type of degree two is called binary, and one of degree three is called ternary.

An example of a ternary relationship type
Structural Constraints

1. **Cardinality ratio** specifies the number of relationship instances that an entity is allowed to participate in. Common cardinality ratios for binary relationship types are:

   - **1 : 1** (one to one)
   - **1 : N** (one to many)
   - **M : N** (many to many)

The **WORKS FOR** relationship type DEPARTMENT:EMPLOYEE of cardinality ratio 1:N.

The **MANAGES** relationship type DEPARTMENT:EMPLOYEE of cardinality ratio 1:1.

The **CONTROLS** relationship type DEPARTMENT:PROJECT of cardinality ratio 1:N.

The **WORKS ON** relationship type EMPLOYEE:PROJECT of cardinality ratio M:N.

The 1:1 relationship **MANAGES**
The M:N relationship WORKS_ON

2. Participation Constraint

   Specifies whether the existence of an entity depends on its being related to another entity via relationship type. There are two types of participation constraints:

   • Partial
   • Total

The participation of EMPLOYEE in WORKS_FOR is total.
The participation of DEPARTMENT in MANAGES is total.
All other participations are partial.
Attributes of Relationship Types

Relationship types can also have attributes, similar to those of entity types.

Attributes of 1:1 or 1:N relationship types can be migrated to one of the participating entity types.

An attribute of 1:N relationship type
(a) equivalent to (b)
Recursive Relationship Type

A recursive relationship type is a relationship type associating entities of the same entity type.

Role Names
Each entity type that participates in a relationship type plays a particular role in the relationship. The role name signifies the role that a participating entity from the entity type plays in each relationship instance.

The recursive relationship type SUPERVISION. The EMPLOYEE plays two roles:
Supervisor (1), and
Supervisee (2).
Weak Entity Types

Some entity types may not have any key attributes of their own; these are called weak entity types. Entities of a weak entity type are identified by being related to specific entities from another entity type in combination with some of their attribute values. That other entity type is called the identifying owner and the relationship type that relates a weak entity type to its owner is called identifying relationship of the weak entity type. A weak entity type has a total participation constraint with respect to its identifying relationship, because a weak entity cannot be identified without an owner entity.

**Diagram:**

**DEPENDENT** as a weak entity type

**Diagram:**

The weak entity type **DEPENDENT** as a composite multivalued attribute
Multi-level weak entity types
Refining the ER diagram

This is the second and last stage in the ER data modelling process. In that stage, relationships are specified explicitly in the ER diagram. Structural constraints (cardinality ratio and participation) are added to each defined relationship type. Refining is done on the initial ER model, where several implicit relationships among the various entity types can be identified.

In fact, whenever an attribute on entity type refers to another entity type, then this is representing an explicit relationship type among the two entity types.

Applying these rules, the following relationship types are specified in the ER model of the company database:

1- **MANAGES**, a 1:1 relationship type between EMPLOYEE and DEPARTMENT. EMPLOYEE participation is partial. DEPARTMENT participation is not clear from the data requirements. We question the users, who say that a department must have a manager at all times, which implies total participation. The attribute StartDate is assigned to this relationship type.

2- **WORKS_FOR**, a 1:N relationship type between DEPARTMENT and EMPLOYEE. Both participations are partial.

3- **CONTROLS**, a 1:N relationship type between DEPARTMENT and PROJECT. Both participations are partial.

4- **SUPERVISION**, a 1:N relationship type between EMPLOYEE (in the supervisor role) and EMPLOYEE (in the supervisee role). Both participations are determined to be partial.

5- **WORKS_ON**, an M:N relationship type between EMPLOYEE and PROJECT. The attribute Hours is assigned to this relationship type. Both participations are determined to be partial.

6- **DEPENDENTS_OF**, a 1:N relationship type between EMPLOYEE and DEPENDENT, which is also the identifying relationship for the weak entity type DEPENDENT. The participation of EMPLOYEE is partial, whereas that of DEPENDENT has to be total.

Alternative ER Notation

An alternative ER notation for specifying structural constraints involves associating a pair of integer numbers (min, max) with each participation of an entity type E in a relationship type R, where

\[0 \leq \text{min} \leq \text{max} \text{ and } \text{max} \geq 1\]

The numbers mean that, for each entity e in E, e must participate in at least min and at most max relationship instances in R at all times.

In this method, min = 0 implies partial participation, whereas, min > 0 implies total participation. This method is more precise, and we can use it easily to specify structural constraints for relationship types of any degree.
ER diagram for the COMPANY schema, with all role names included and with structural constraints on relationships specified using the alternate notation (min, max).