Database design and Data modeling using Entity-Relationship Diagrams (ERDs)

Dr. Enosha Hettiarachchi

Database Design

The database design process can be broken down into four phases.

Phase 1 - Requirements Collection
and analysis phase

Phase 2 - Conceptual Design

Phase 3 - Logical Design

Phase 4 - Physical Design

Database Design...

Mini-world

Phase 1 – Requirements Collection and Analysis phase

Functional Requirements

Database Requirements

Prospective database uses are interviewed to understand and document their data requirements.

From data view (e.g. ERD) rather than functional view (e.g. DFD)

Database Design...

Database Requirements

↓

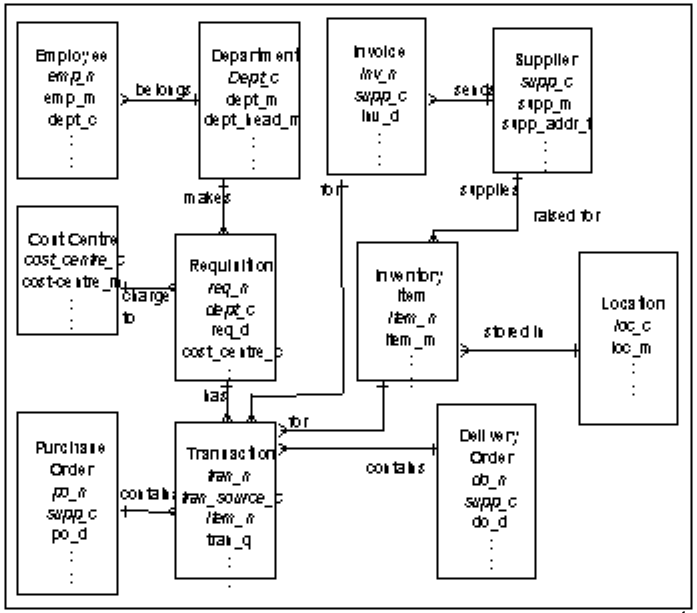
Phase 2 - Conceptual Design

↓

Conceptual Design

This is high level description of the structure of a database.
E.g. E-R diagram

Concise description of the data requirements of the users and includes detailed descriptions of the data, relationships and constraints.



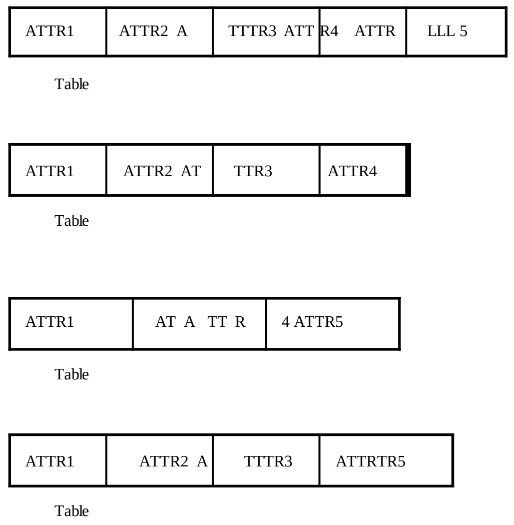
Database Design…

Conceptual Design

Phase 3 - Logical Design

This is the process of mapping the database structure developed in the previous phase to a particular database model. E.g. map E-R model to relational

Logical Design



Specific to a database model, but independent of a particular DBMS (product)

Database Design...

Logical Design

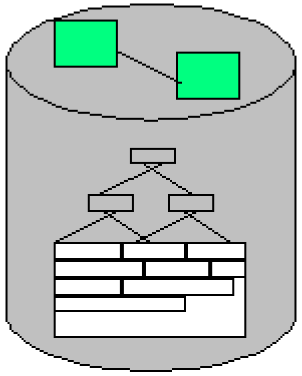
$\downarrow$

Phase 4 - Physical Design

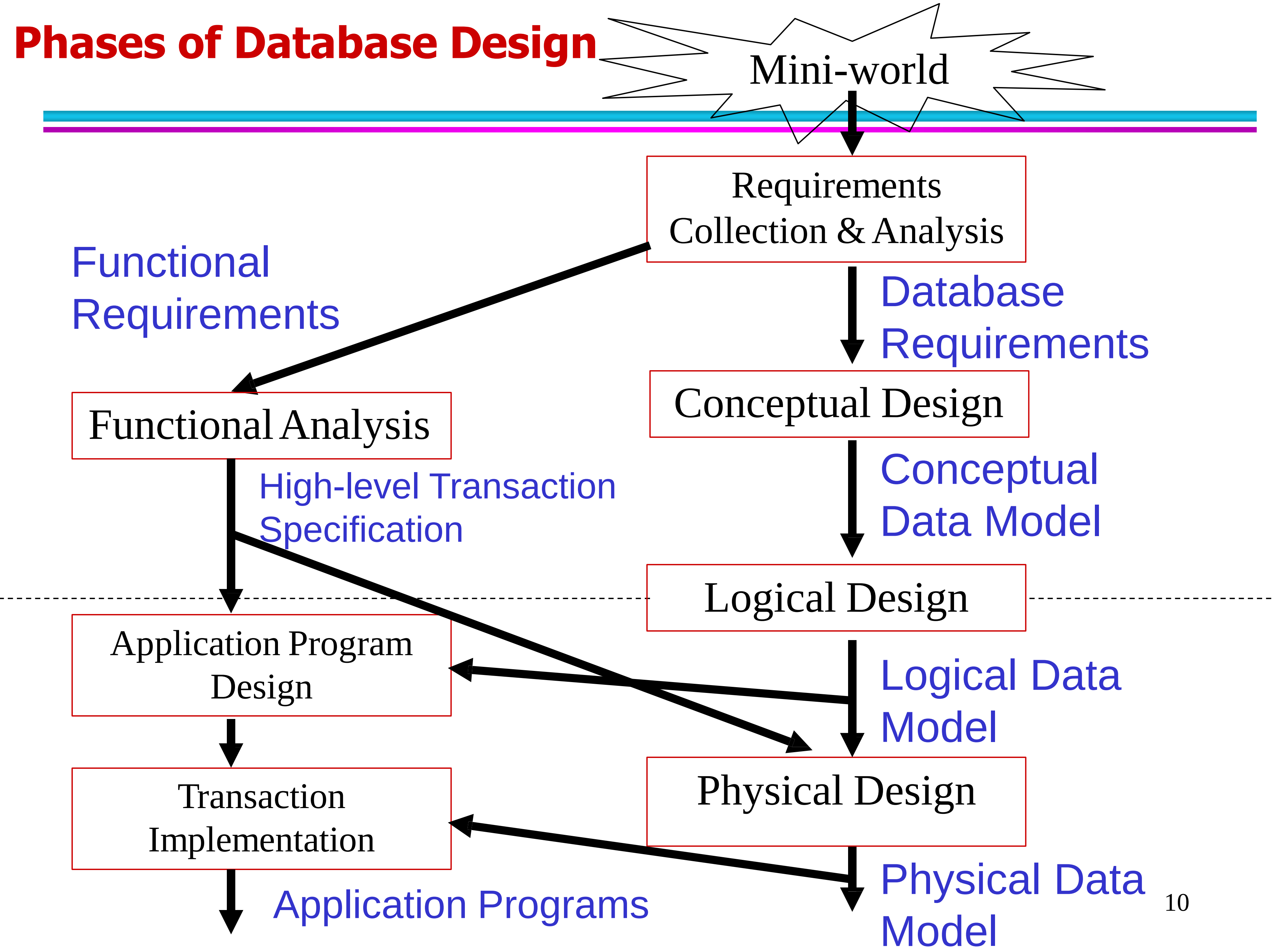
$\downarrow$

Physical Design

This is the process of defining structure that enables the database to be queried in an efficient manner.
E.g. index and hash file design, data partition

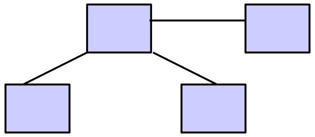


Phases of Database Design

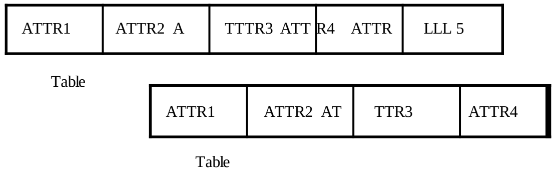


Types of Data Models

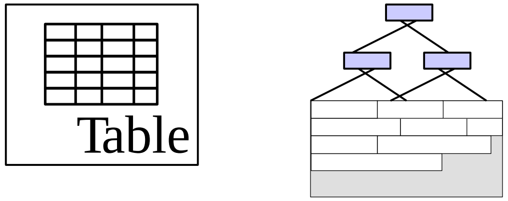
- **Conceptual Data Model**



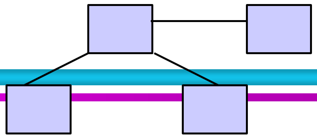
- **Logical Data Model**



- **Physical Data Model**



Conceptual Data Model



- A data model representing the objects and business rules that govern the operation of an organisation
  - *Done by a Business Analyst*
  - *Not constrained by access requirement and technology*

Logical Data Model

ATTR1	ATTR2 A	TTTR3 ATT	R4 ATTR	LLL 5

Table

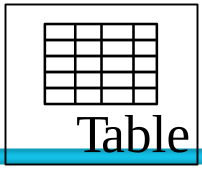
employee(empno, …)

•A set of data structures assembled following rules that describe the processing requirements (access paths) of the data in terms of a logical database model

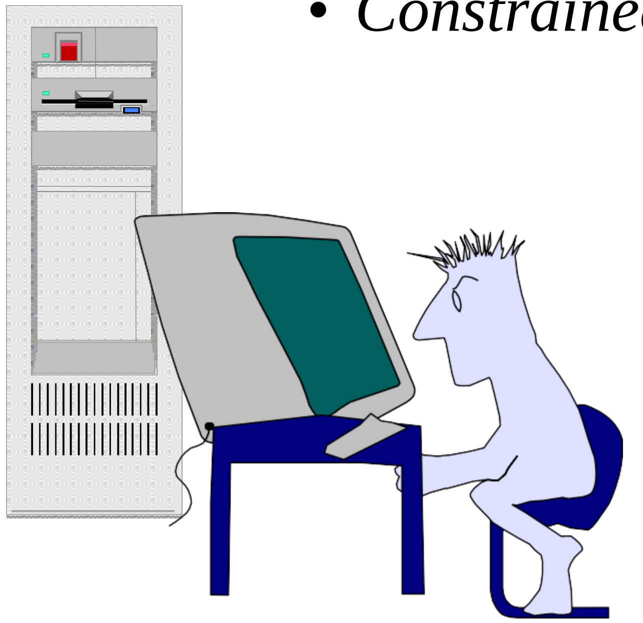
- *Done by a Data Analyst*
- *Not constrained by technology*

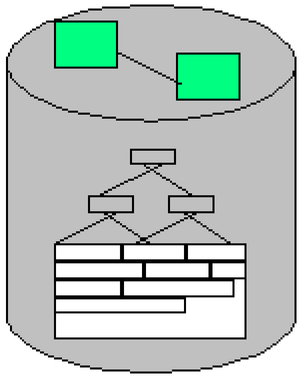
Physical Data Model

create table employee(empno …)

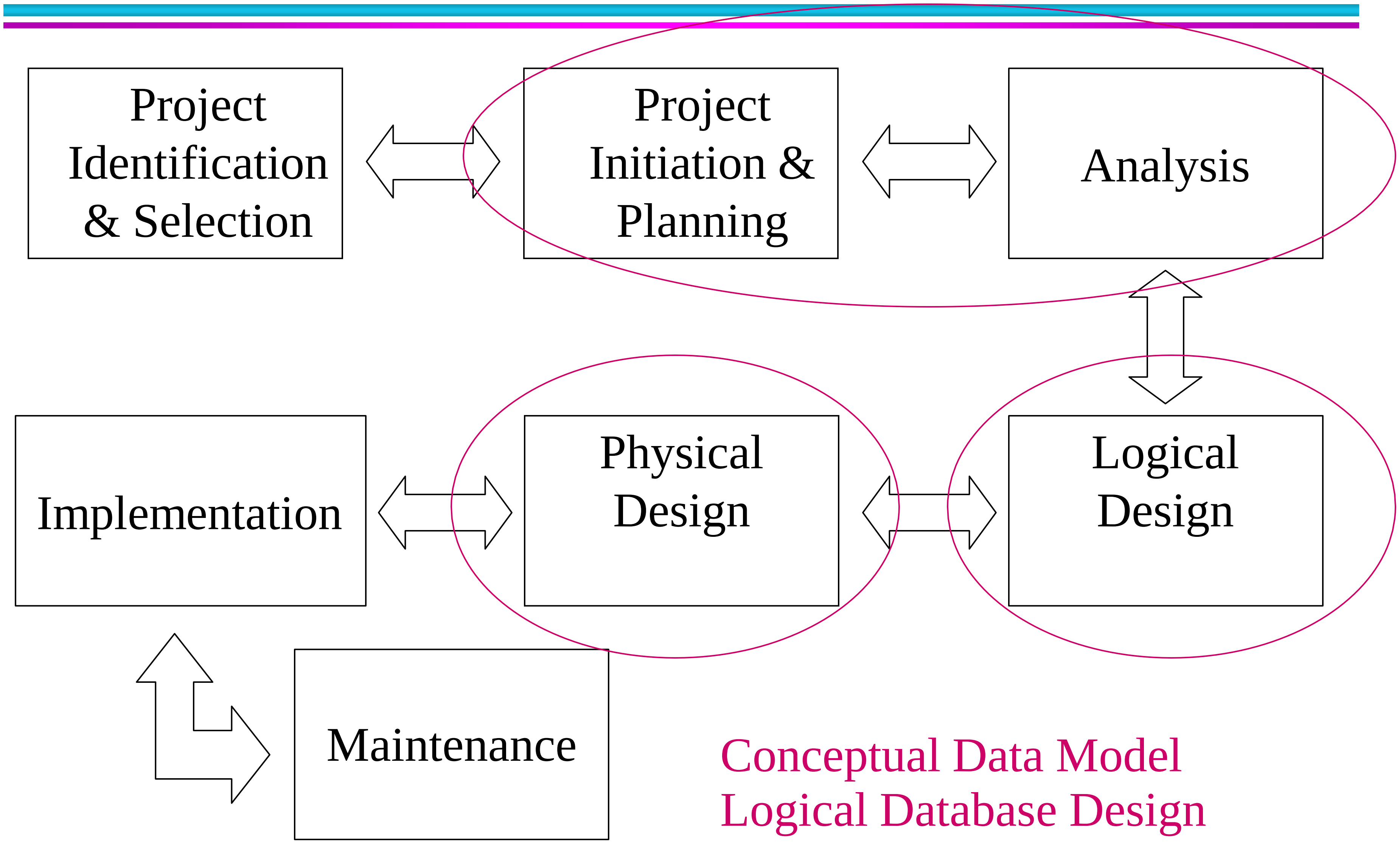


- A model prepared for the purpose of implementing a database that runs under the control of a particular DBMS (product)
  - *Done by a DBA*
  - *Constrained by Technology*





Systems Development Life Cycle (SDLC)


Project Identification & Selection
Project Initiation & Planning
Analysis
Logical Design
Physical Design
Implementation
Maintenance

Conceptual Data Model
Logical Database Design
Physical Database Design

Database Development Activities

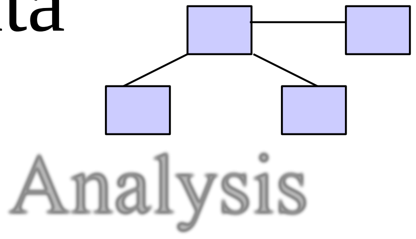
- Enterprise Modelling
- Conceptual Data Modelling
- Logical Database Design
- Physical Database Design and Creation
- Database Implementation
- Database Maintenance

Enterprise Modelling

- Analyse current data processing
- Analyse the general business functions and their database needs
- Justify need for new data and databases in support of business

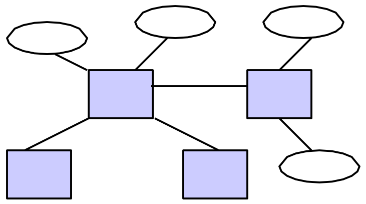
Conceptual Data Modelling

- Identify **scope of database requirements** for the proposed information system
- **Analyse overall data requirements** for business function(s) supported by database
- **Develop preliminary conceptual data model** including entities and relationships
- **Compare** preliminary conceptual data model with enterprise data model



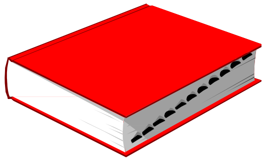
Conceptual Data Modelling…

- **Develop detailed conceptual data model**, including all entities, relationships, attributes and business rules



- Make conceptual data model **consistent** with other models of information system

- Populate repository with all conceptual database specifications



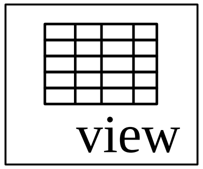
Analysis

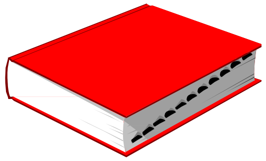
Logical Database Design

- Analyse in detail the transactions, forms, displays and inquires (data views) required by the business functions supported by the database

•Integrate database views into conceptual data model

•Identify data integrity and security requirements, and populate repository

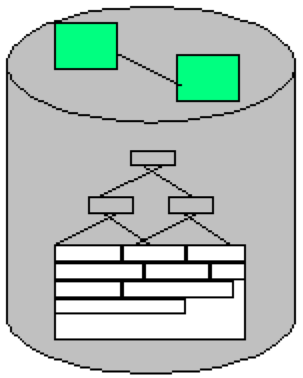




Logical Design

Physical Database Design

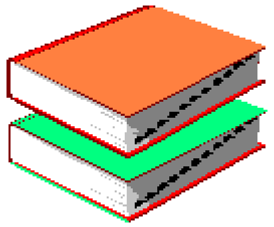
- Define database to DBMS (often generated from repository)
- Decide on physical organisation of data
- Design database processing programs



Physical Design

Database Implementation

- Code and test database processing programs
- Complete database documentation and training materials



- Install database and convert data from prior systems

Implementation

Database Maintenance

- Analyse database and database applications to ensure that evolving information requirements are met
- Tune database for improved performance
- Fix errors in database and database applications and recover database when it is contaminated

Maintenance

Conceptual Modeling

- Very important phase in designing a successful database application.
- Entity-Relationship (ER) model – popular high-level conceptual data model. Concepts of the ER Model
    - **Entity types**
    - **Relationship types**
    - **Attributes**

Conceptual Modeling

- UML (Universal Modeling Language) – Object modeling methodology
- Increasingly popular in s/w design and engineering

Conceptual Modeling

- Class diagrams – similar in many ways to the ER diagram

- Functional requirements

  Specify the operations on objects/entities during database design

Conceptual Design

- All the requirements collected at Requirements Collection and Analysis phase are analysed to create a Conceptual Schema.

- This process is called the Conceptual Design.

- We identify the entities, their attributes, relationships and constraints (business rules).

Conceptual Design

- The conceptual schema is used as a reference to ensure that all user's data requirements are met, and the requirements do not include any conflicts.

Entity Type

- Entity – A ‘thing’ in the real world with an independent existence.
- An entity may be **an object with a physical existence** or may be **an object with conceptual existence**.
- Entity type – defines a collection (a set) of entities that have the same attributes.

A mini-world example

- A Company is organised into departments. Each department has a number and an employee who manages the department. It is necessary to keep track of the start date when that employee started managing the department. A department may have several locations.

- A department controls a number of projects. Each of which has a name, a number and a single location.

A mini-world example

- Data such as the number of hours per week that an employee works on each project and the direct supervisor of each employee are required to be maintained.

- The dependants of each employee are maintained for insurance purposes. We keep each dependant’s name, gender, birth date and relationship to the employee.

A mini-world example

- It is necessary to store each employee’s name, national Id number, address, salary, birth date and gender. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled, by the same department.

- **Entities**

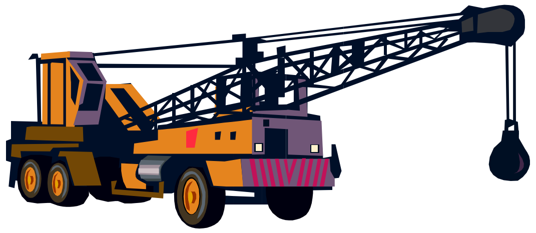
Conceptual Design

Entities

- Department

- Employee

- Project



- Dependent

Attributes

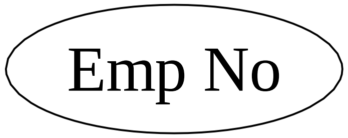
- **Attribute**

  A property or characteristic of an entity type that is of interest to the organisation.

- **Simple Attribute**

  An attribute that cannot be broken down into smaller components.

  – e.g. Emp No

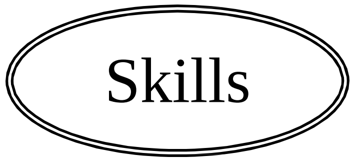


Attributes

Multi-valued Attribute {}

An attribute that may take on more than one value for a given entity instance

e.g. Employee Skills, Qualifications



Composite Attribute

An attribute that can be broken down into component parts

e.g. Address (Street, City, State, Postal Code)

Name (First Name, Middle Initials, Last Name)

Attributes

Stored Attribute

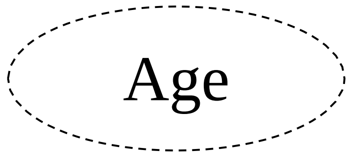
An attribute whose value is stored in the database

Derived Attribute

An attribute whose values can be calculated from related attribute values

e.g. Years Employed (using Employed Date)

Age (using Date of Birth)

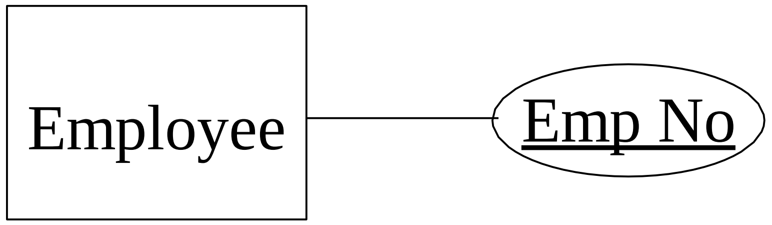


Key Attribute(Identifier)

Identifier

An attribute (or combination of attributes) that uniquely identifiers individual instances of an entity type

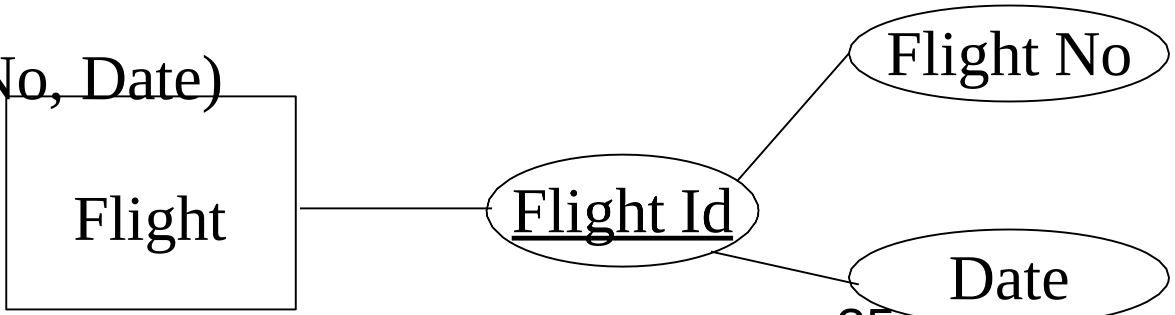
e.g. Emp No



Composite Identifier

An identifier that consists of a composite attribute

e.g. Flight Id (Flight No, Date)



Key Attribute (Identifier)

- Choose an identifier that will not change its value over the life of each instance of the entity type.
- Choose an identifier such that each instance of the entity type, the attribute is guaranteed to have valid values and not be null (or unknown).
- Consider substituting single-attribute identifiers for large composite identifiers.

Initial Conceptual Design of the Company Database

- **Department**

  Name, Number,{Locations}

- **Project**

  Name, Number, Location

- **Employee**

  Name(Fname, Lname), NID, Gender, Address, Salary, BirthDate

- **Dependent**

  dependentName, Gender, BirthDate, Relationship

Initial Conceptual Design of the Company Database

- **Department**

  Name, Number,{Locations}, Manager, ManagerStartDate
- **Project**

  Name, Number, Location, ControllingDepartment
- **Employee**

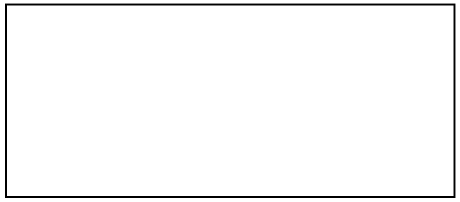
  Name(Fname, Lname), NID, Gender, Address, Salary, BirthDate, Department, Supervisor{WorksOn (Project,Hours)}
- **Dependent**

  Employee, dependentName, Gender, BirthDate, Relationship

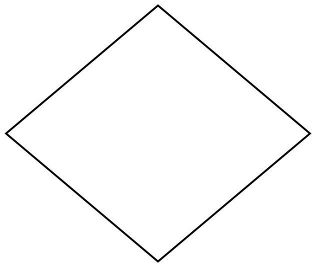
Conceptual Design

Notations

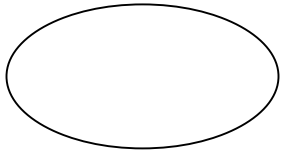
Entity



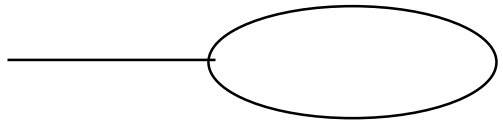
Relationship



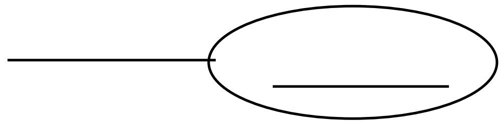
Attribute



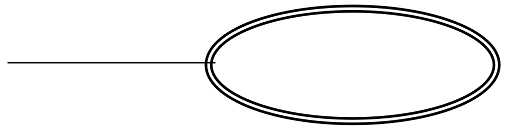
Conceptual Design



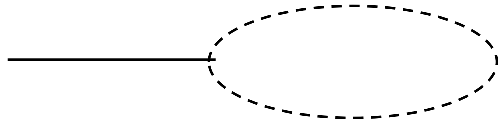
Attribute



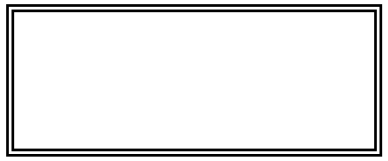
Key Attribute



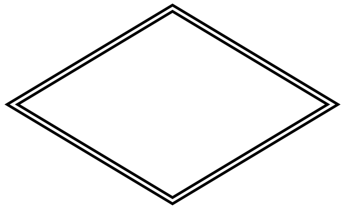
Multivalued attributes



Derived Attribute



Weak Entity

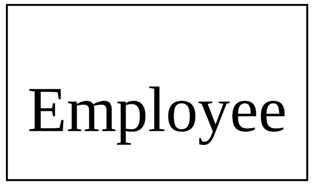


Identifying Relationship

Entity Types

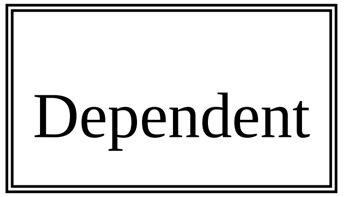
Strong (Regular) Entity

An entity that exists independently of other entity types



Weak Entity

An entity types whose existence depends on some other entity

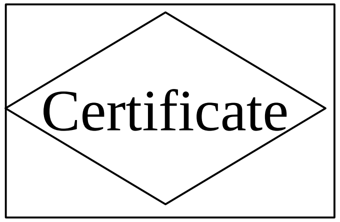


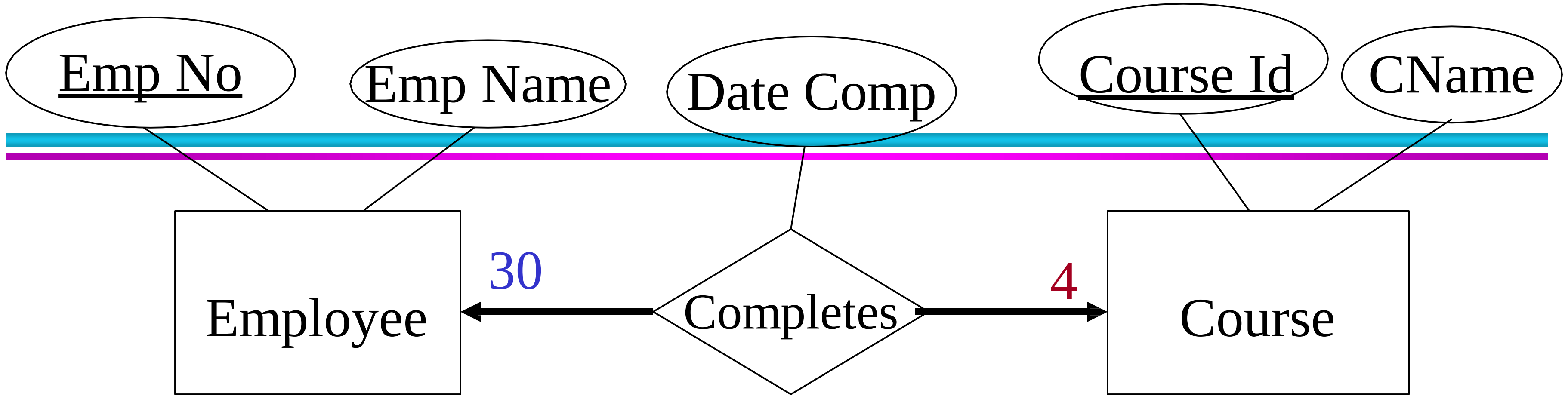
Entity Types

- **Identifying Owner**
  - The entity type on which the weak entity type depends
    e.g. Employee is the Owner of Dependent

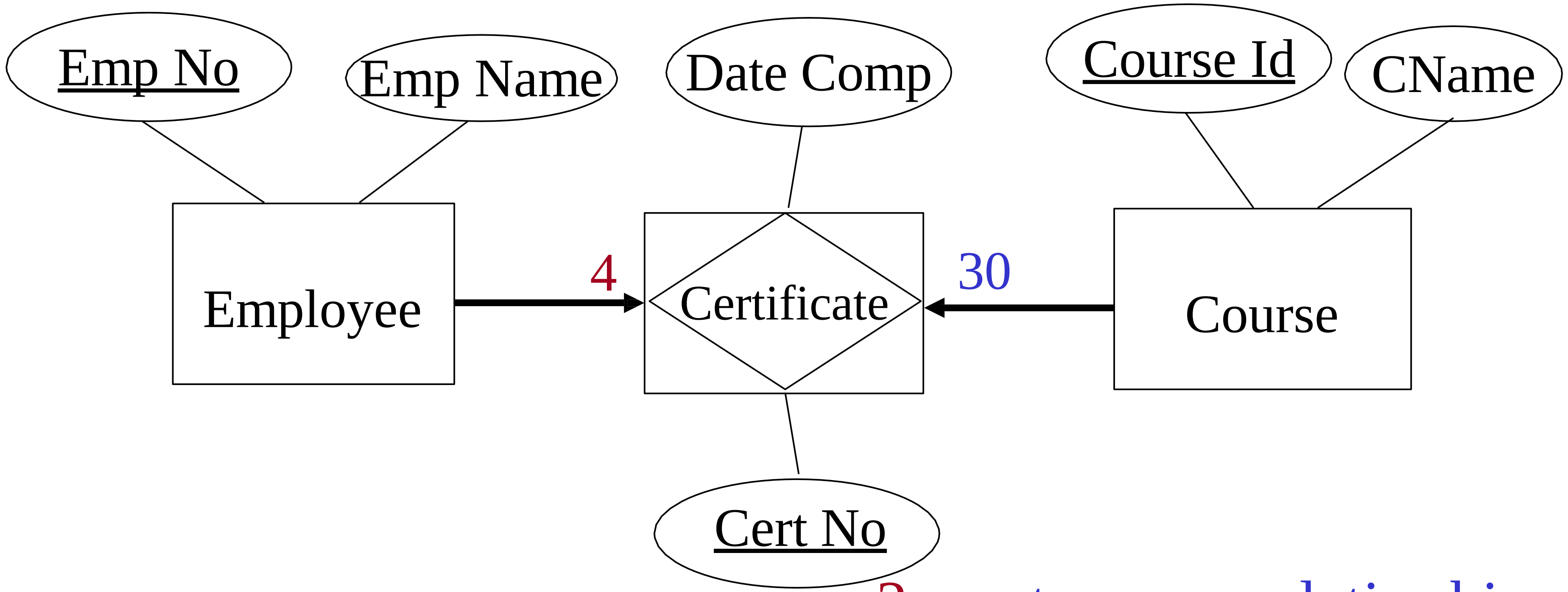
Associative Entity

- An entity type that associates the instances of one or more entity types and contains attributes that are specific to the relationship between those entity instances





1 many to many relationship



2 one to many relationship

Associative Entity

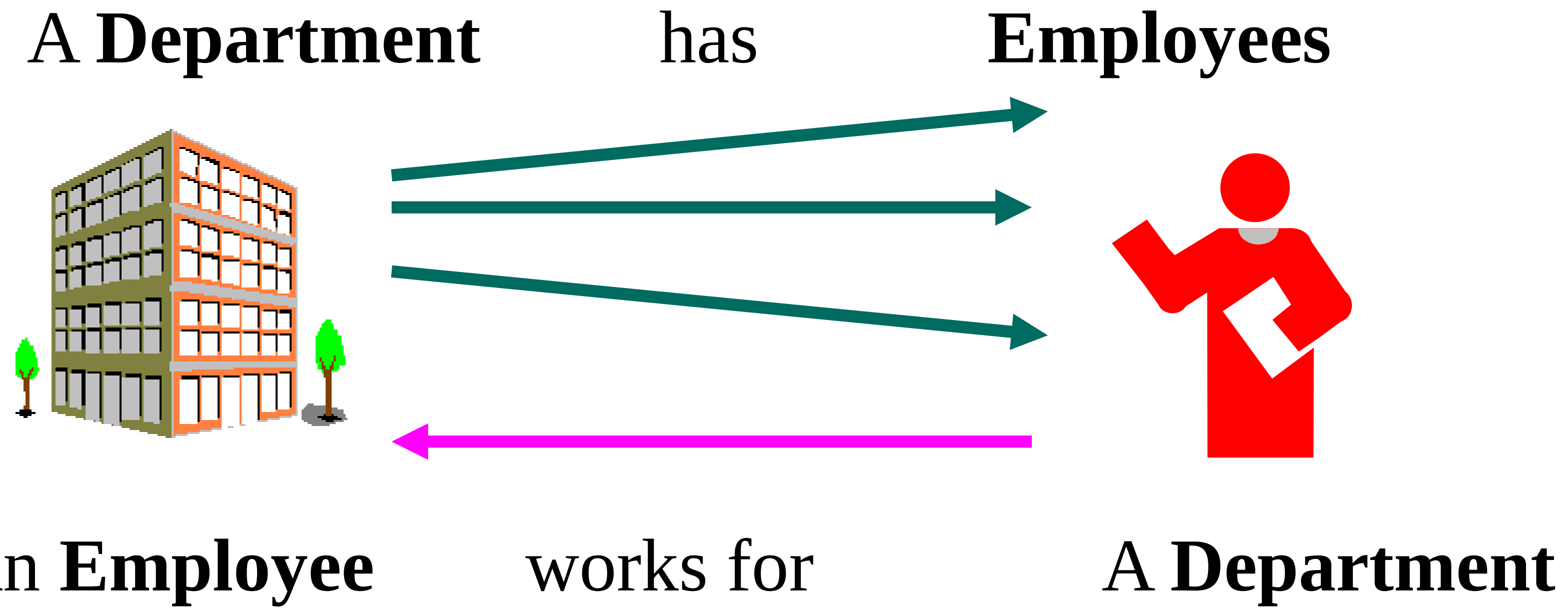
- All of the relationships for the participating entity types are "many" relationships
- The resulting associative entity type has independent meaning to end users, and preferably can be identified with a single-attribute identifier
- The associative entity has one or more attributes, in addition to the identifier
- The associative entity participates in one or more relationships independent of the entities related in the associated relationships

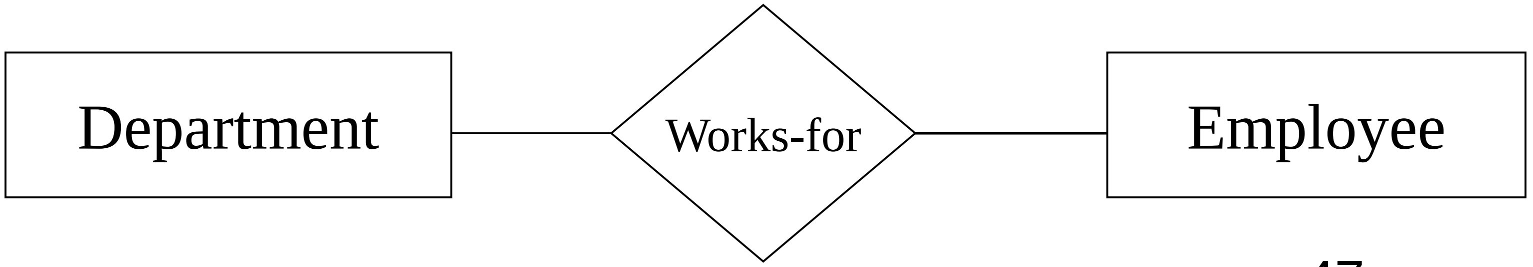
Relationships

- Relationship exists, whenever an attribute of one entity type refers to another entity type.

- In the ER model these references **should not be** represented as attributes but as relationships.

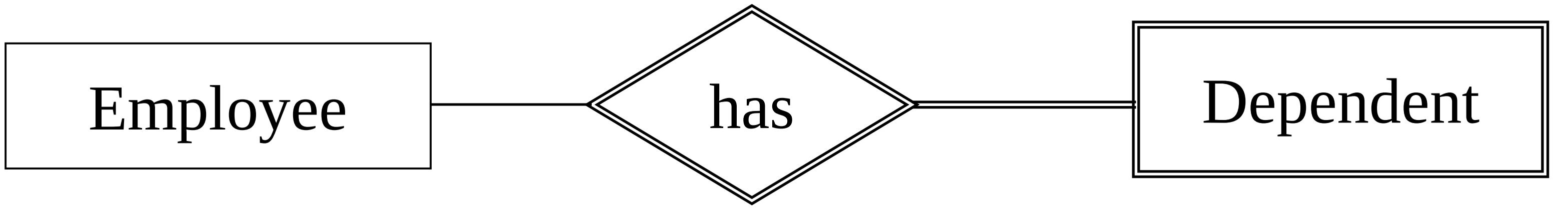
Relationships


A **Department** has **Employees**
An **Employee** works for A **Department**



Relationships

- A relationship between a weak entity type and its owner is represented as given below.



Relationships

- **Relationship Type**

  A meaningful association between (or among) entity types

- **Relationship Instances**

  An association between (or among) entity instances, where each relationship instance includes exactly one entity from each participating entity type

e.g. De Silva works for Personnel Department

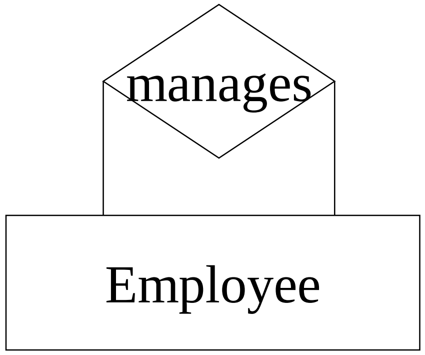
Degree of Relationship Type

Number of participating entity types.

- **Recursive (Unary) Relationships**
  - A relationship between the instances of a single entity type

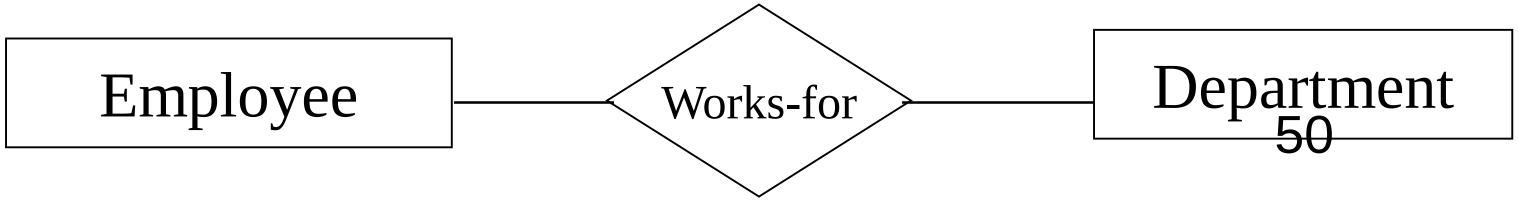
  e.g. Person is married to a Person

  Employee manages Employees



- **Binary Relationship**
  - A relationship between the instances of two entity types

  e.g. An employee works for a department

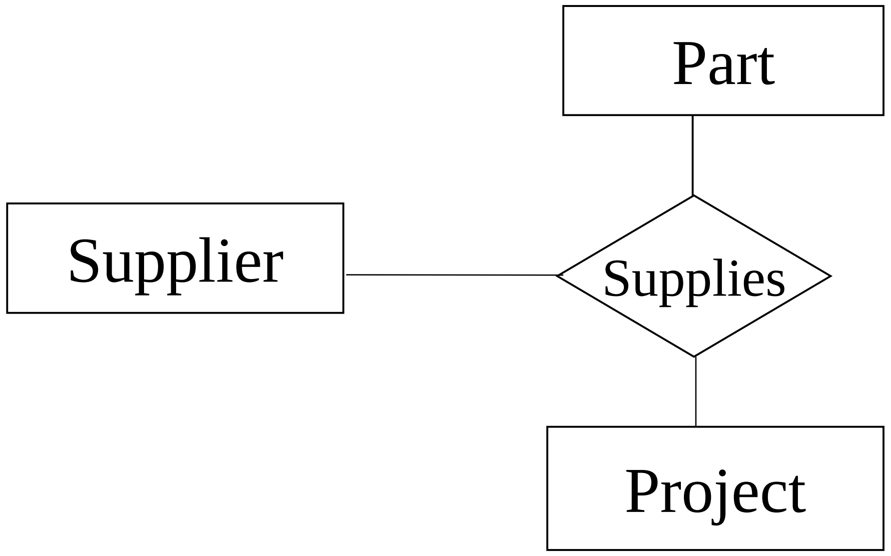


Degree of Relationship Type

Ternary Relationship

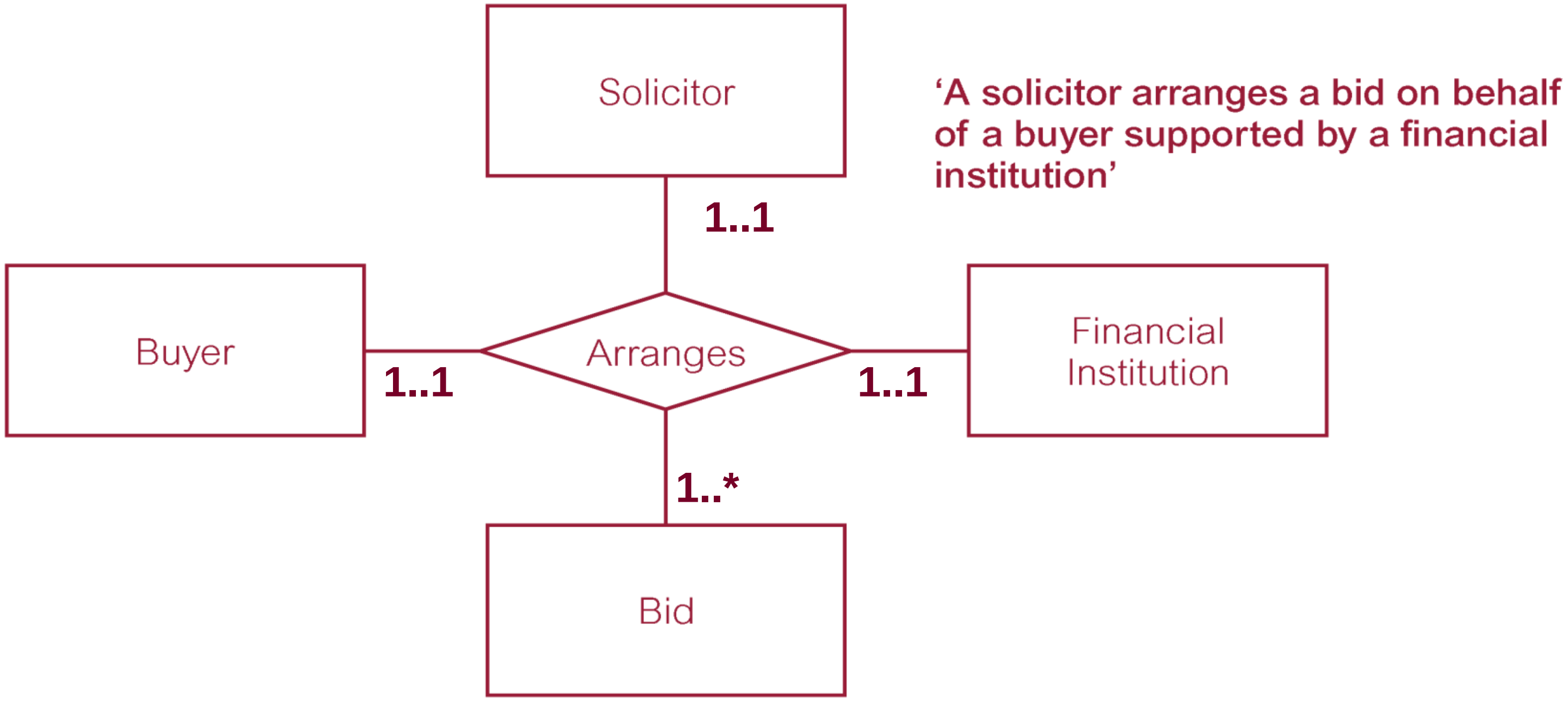
A simultaneous relationship among the instances of three entity types

Supplier s supplies part p to project j



Degree of Relationship Type

- **Quaternary relationship**


Solicitor
‘A solicitor arranges a bid on behalf of a buyer supported by a financial institution’
1..1
Buyer
Arranges
Financial Institution
1..1
1..1
1..*
Bid

Constraints on Relationship Type

Cardinality Ratios for Binary Relationships

Specify the number of instances of one entity that can (or must) be associated with each instance of another entity.

The possible cardinality ratios for binary relationship types are

1:1

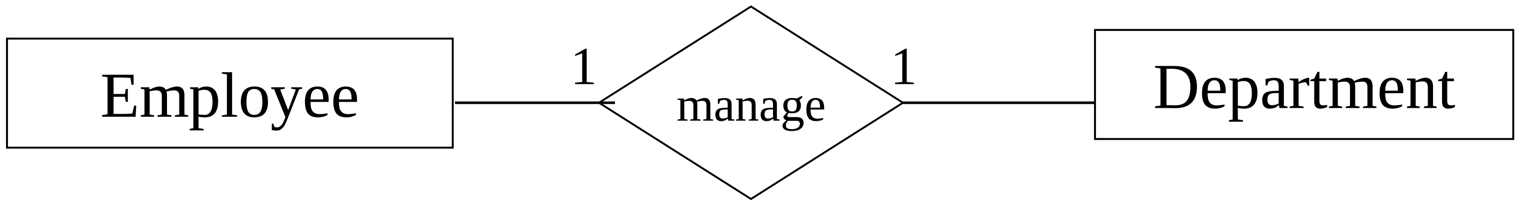
1:N

M:N

1:1

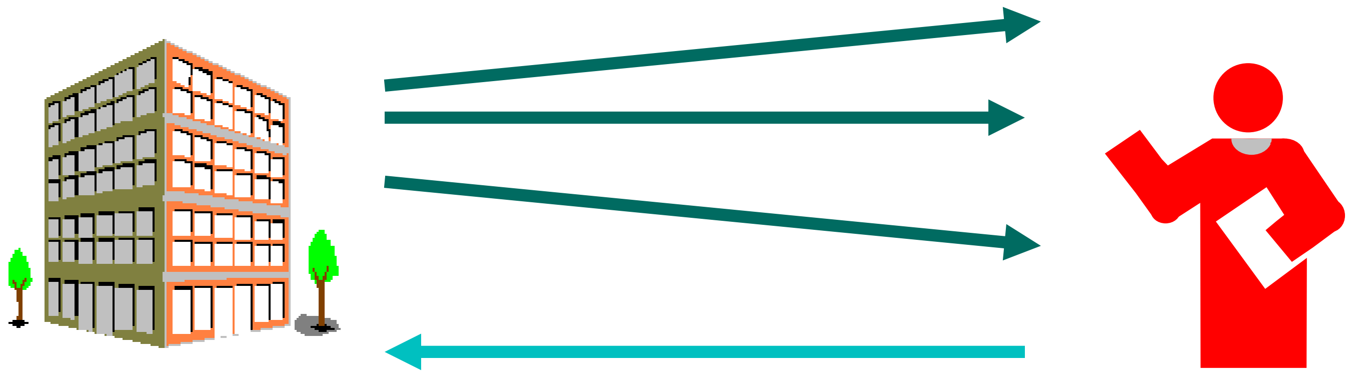
A **Department** has A Manager (**Employee**)

An **Employee** manage A **Department**

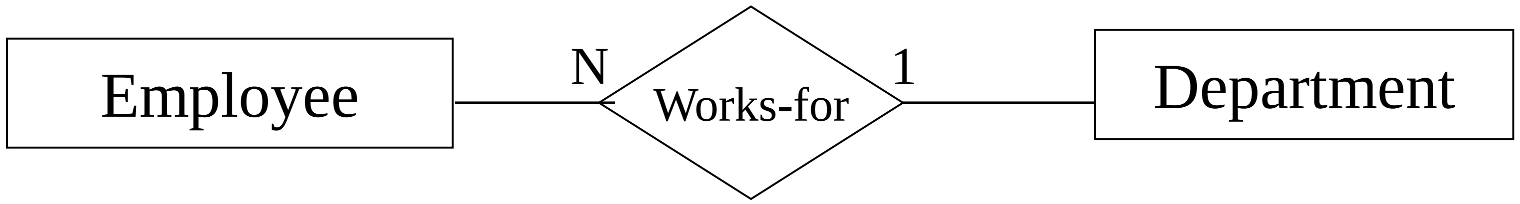


1:N

A **Department** has Many **Employees**



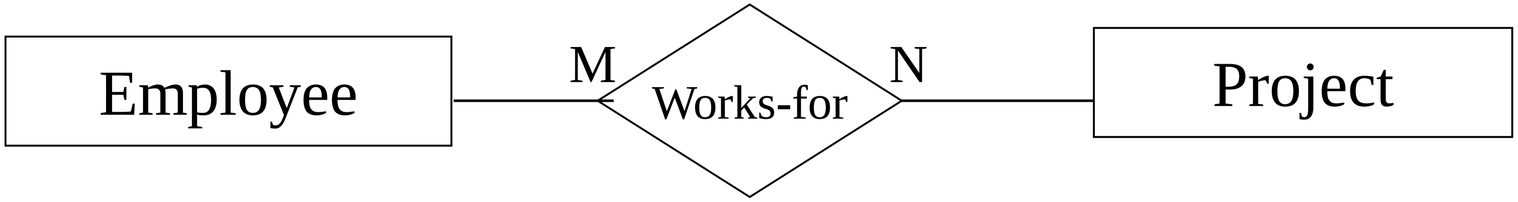
An **Employee** works for A **Department**



M:N

An **Employee** works on Many **Projects**

A **Project** has Many **Employees**

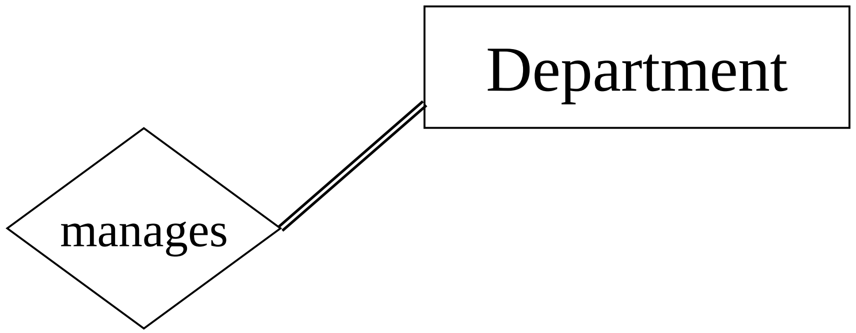


Constraints on Relationship Type

Participation constraints - There are two types

- Total Participation (existence dependency):
  e.g. every employee must work for a department.

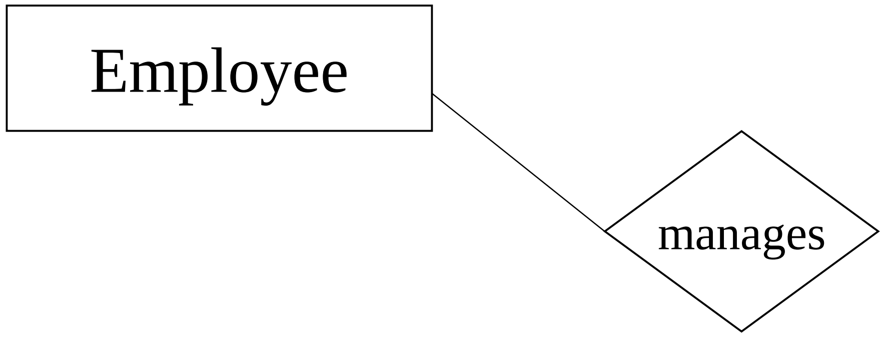
Therefore, participation of Employee in works-for relationship type is total.



Constraints on Relationship Type

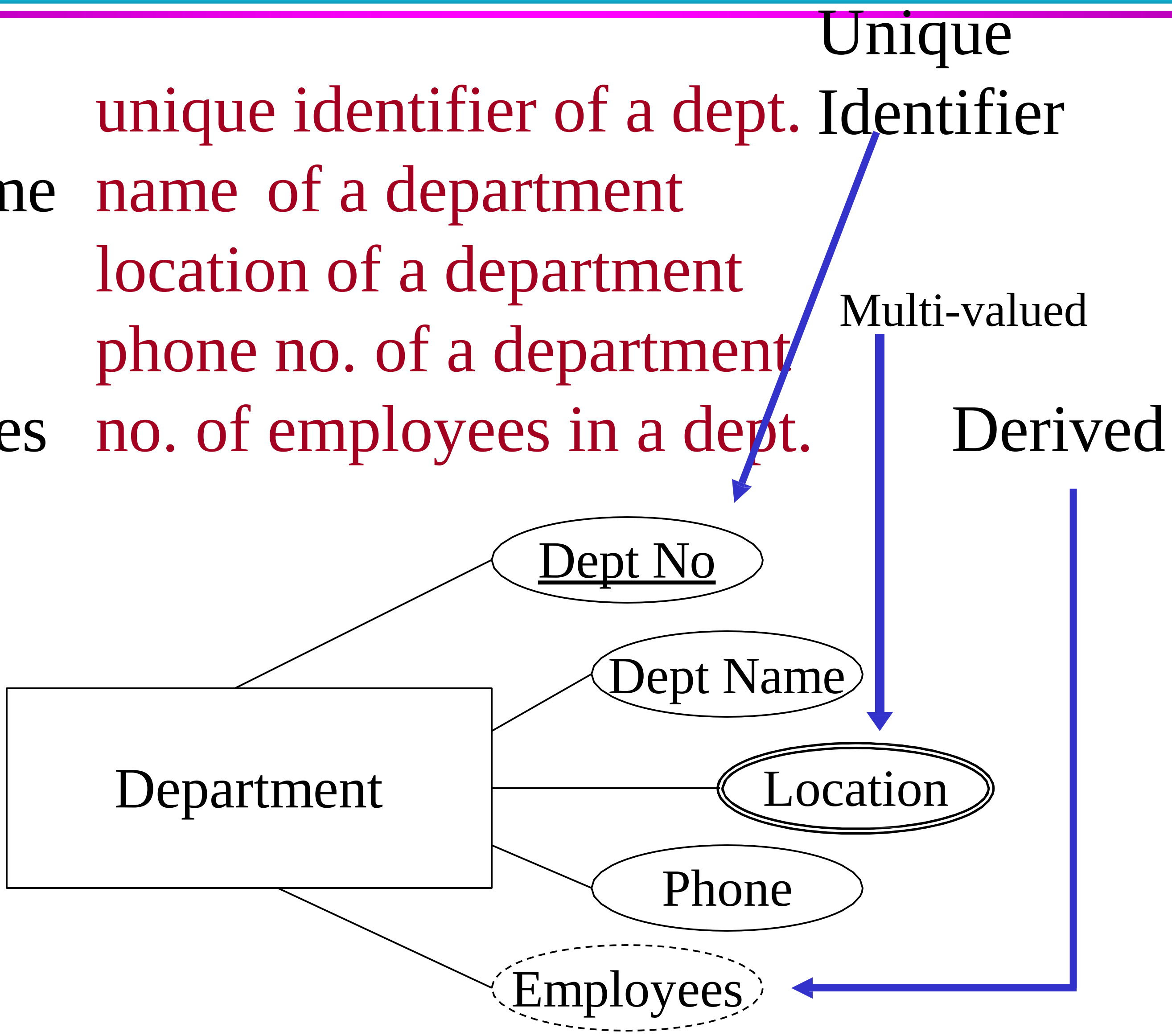
- **Partial Participation**

  The participation of Employee in manages relationship type is partial.

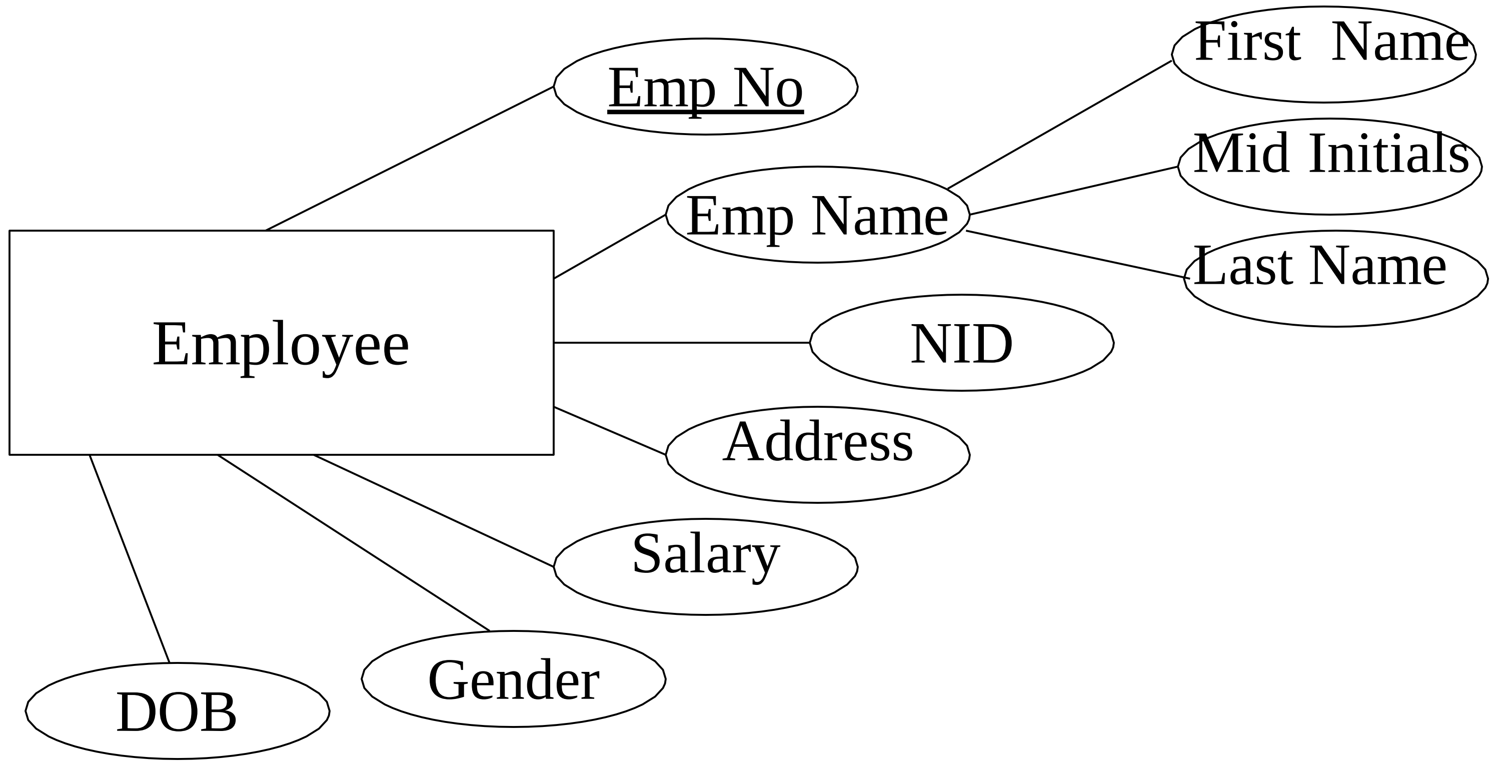


Detailed Conceptual Design

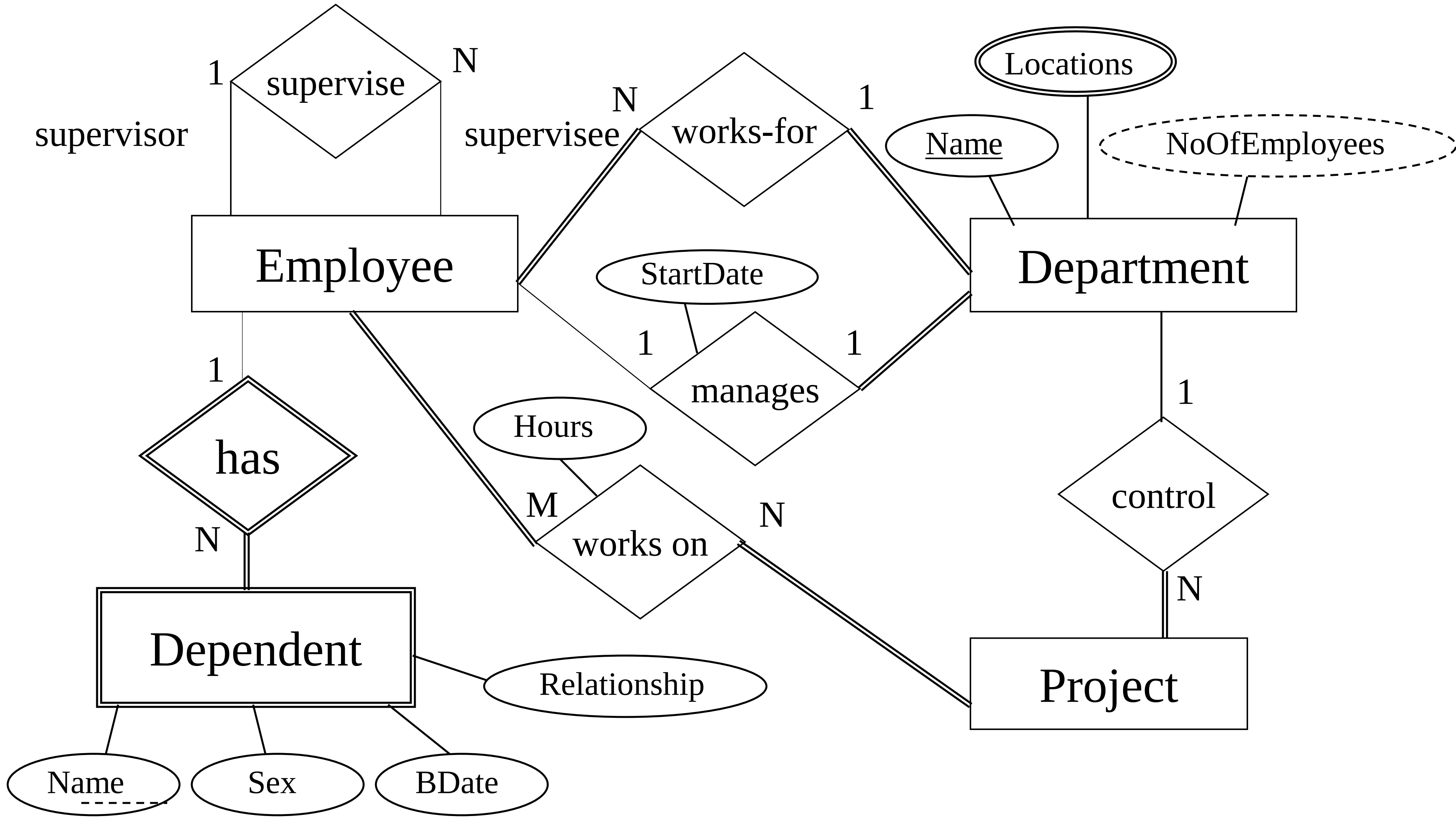
Dept No	unique identifier of a dept.
Dept Name	name of a department
Location	location of a department
Phone	phone no. of a department
Employees	no. of employees in a dept.



Detailed Conceptual Design



ER Schema Diagram

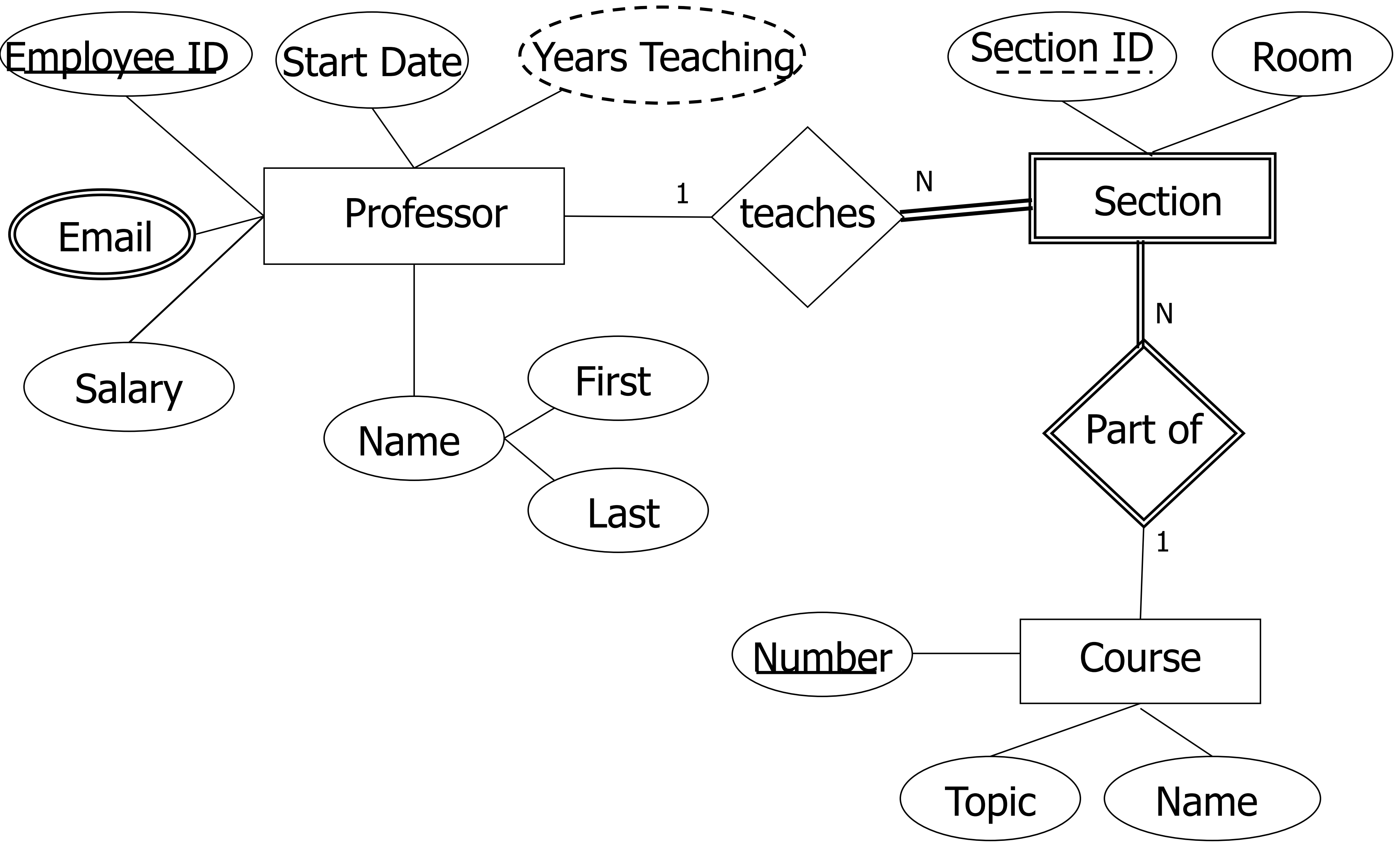


Teaching Database

Design an E-R schema for a database to store info about professors, courses and course sections indicating the following:

- The name and employee ID number, salary and email address(es) of each professor
- How long each professor has been at the university
- The course sections each professor teaches
- The name, number and topic for each course offered
- The section and room number for each course section
- Each course section must have only one professor
- Each course can have multiple sections

ER Schema Diagram


Employee ID
Start Date
Years Teaching
Section ID
Room
Email
Professor
1
teaches
N
Section
Salary
N
First
Name
Part of
Last
1
Number
Course
Topic
Name