31269:
Business Requirements Modelling

Week 5 Lecture - Data Modelling

- ✓ References
  - ✓ BABOK Guide

Objectives

- Appreciate how modelling techniques can help understand the working of business systems

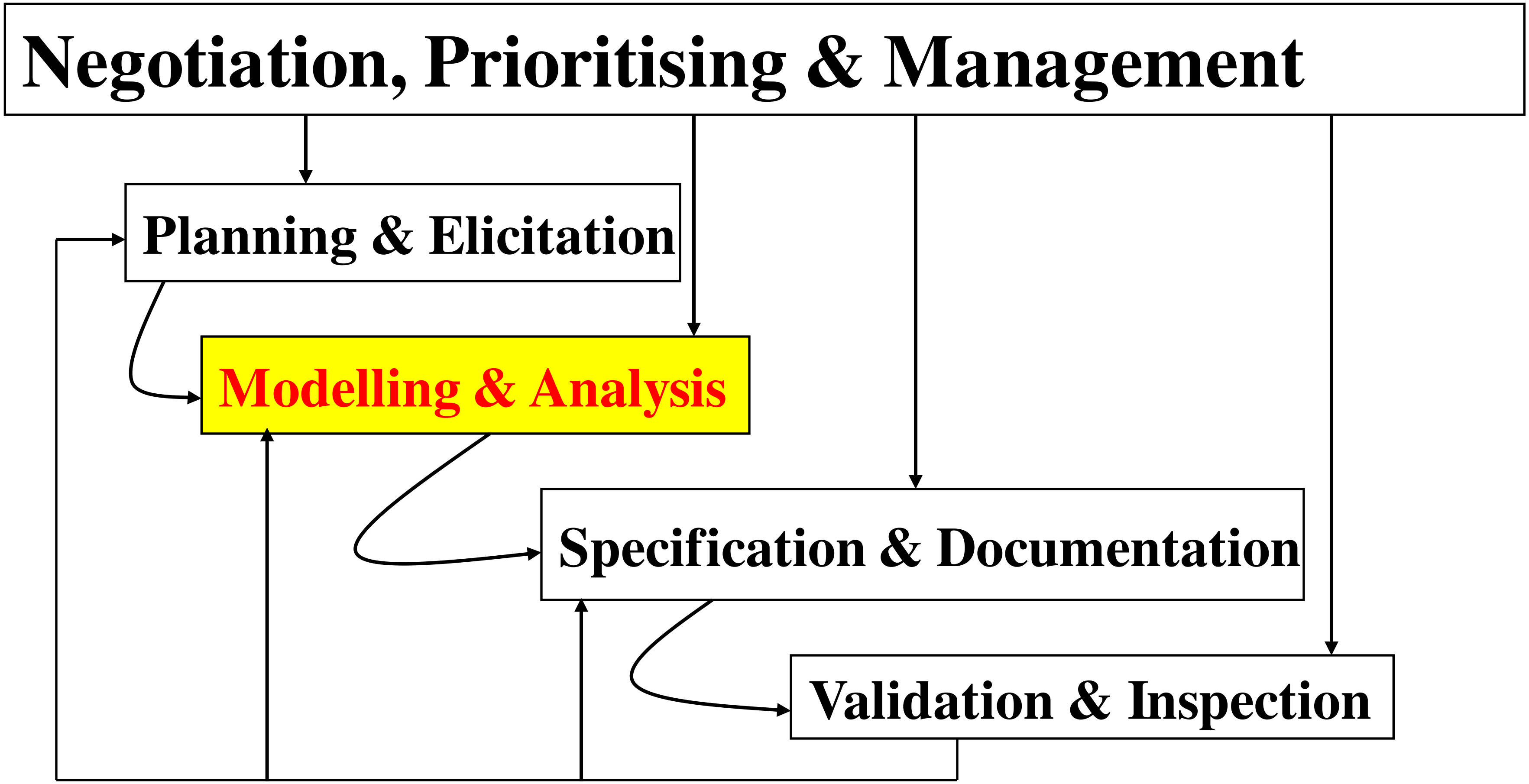
- Discover how modelling can be used to specify system and user requirements

- Discover how Entity Relationship Diagram (ERD) can be used to model, analyse and understand the **data requirements of an organisation**

Topic

- Requirements Analysis & Modelling
    - Data Modelling using ERD

Requirements Process


Negotiation, Prioritising & Management
Planning & Elicitation
Modelling & Analysis
Specification & Documentation
Validation & Inspection

Requirements Analysis

- Previously - How are the requirements collected?
  - Interviews, workshops, prototypes, surveys, observation, document review, etc.
- This Week - How do we analyse and model these requirements?
  - Data Modelling (Entity Relationship Diagram)

Data Model

- **A data model:**
  - identifies the **entities or objects that the organisation will need to hold data about.**
  - identifies the data that must be captured, stored and retrieved.
  - describes the data in a logical (conceptual) manner.
  - creates a graphical representation of the entities, and the relationships between entities.
  - Focuses on **what data is required and how it should be organised.**
  - is a **plan for building a database**.

Entity Relationship Diagram (ERD) as a Data Model

- ERD as a “Data Model” is
  - an abstraction of the data the organisation works with
  - a way to organise the data of interest into a standardised structure
- ERD is a **graphical representation of the data requirements** for a database.
- The Entity-Relationship Model is a conceptual data model that views the real world as consisting of entities and relationships.
- ERD is a graphical representation of the entities relevant to a chosen problem domain, the relationships between them, and their attributes.
- The basic constructs/parts of an ERD are:
  - **Entities** are anything about whom you want to **record or store data;** concepts, real or abstract, about which information is collected.
  - **Relationships** are associations between the entities.
  - **Attributes** are properties which describe the entities.
  - **Cardinality** is a numerical relationship between two entities.

Why ERD?

This slide to be read at home by students

- Describe data requirements of the business
- Represent the business rules that apply to an organizations data
- Confirm business and data requirements and provide direction to the architecture and design team as they move forward with database design

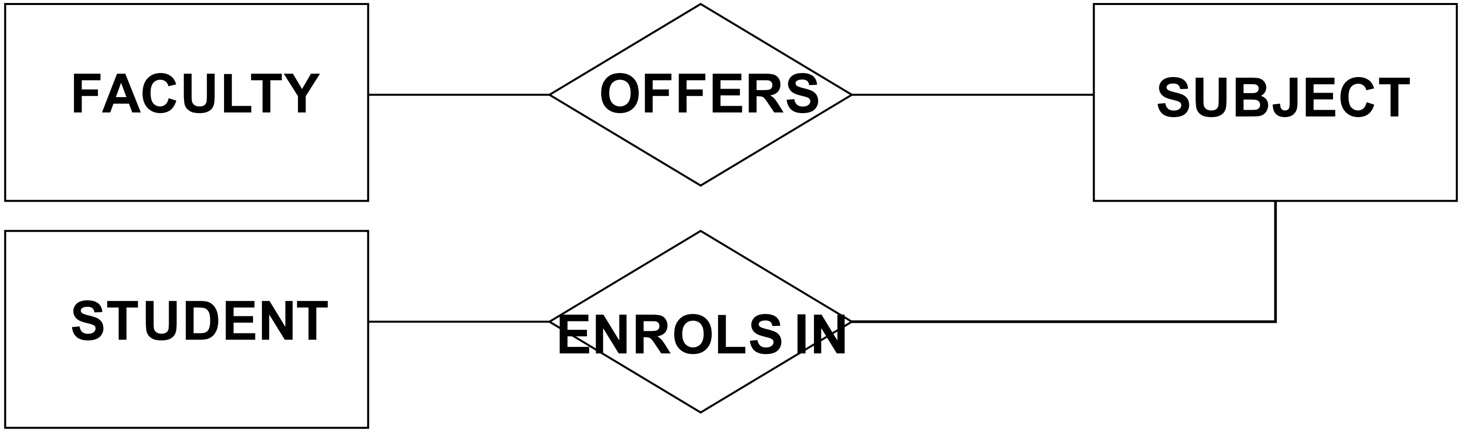
ERD Vs Process Model

- Focus of ERDs is to capture the important data in the organisation, whereas the focus of Process Models is to capture the business processes in an organisation

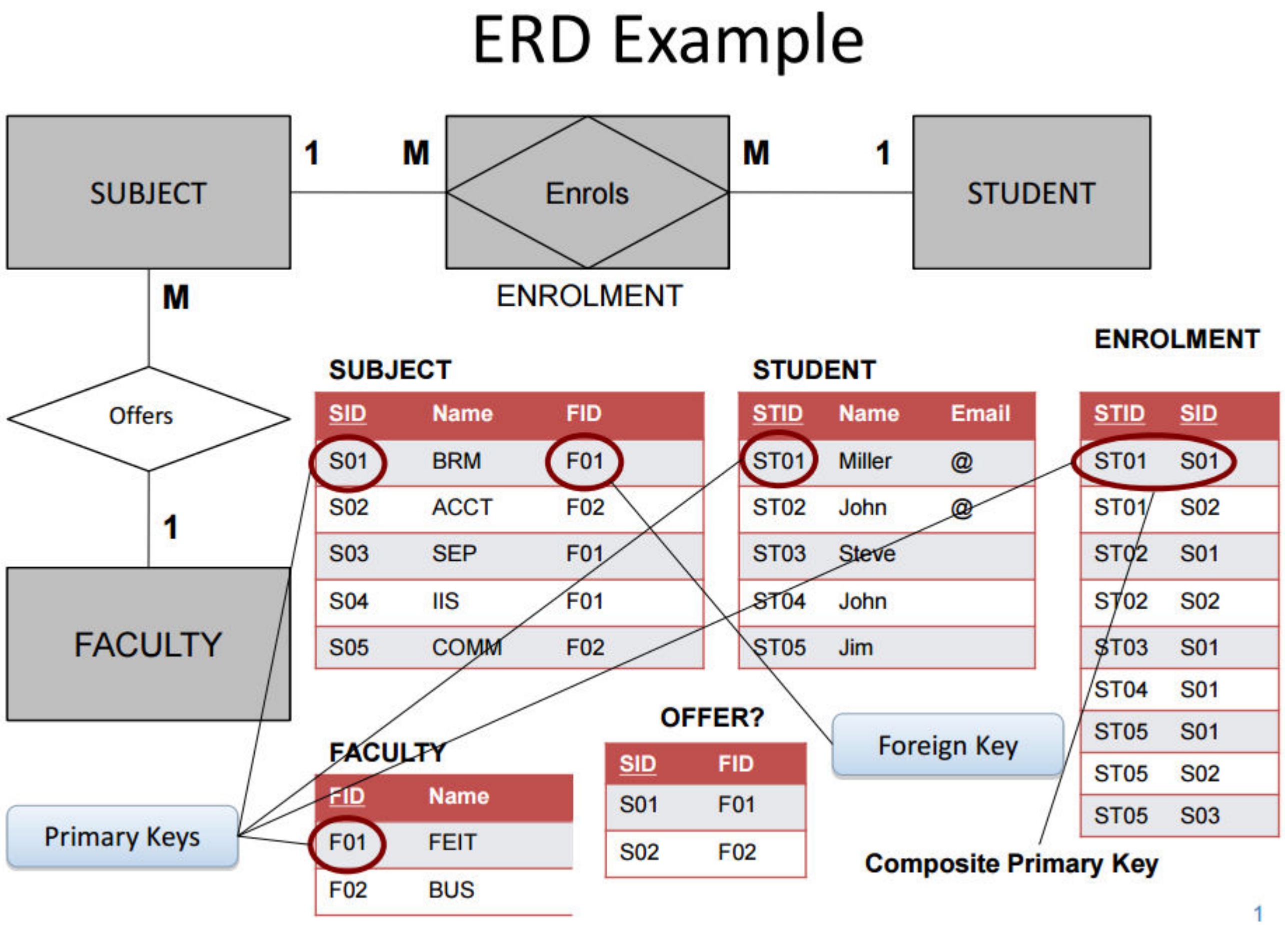
- ERDs focus on the data to be stored and retrieved in an efficient manner. It shows the relationship between the data. Whereas a process model shows the business processes so that the processes can be analysed and improved.

Example of an ERD

- Network structure
- Entities are located at the nodes of the network
- Connections between the nodes define the relationships and cardinality



Example of an ERD



ERD Example

SUBJECT 1 — M Enrols M — 1 STUDENT

ENROLMENT

SUBJECT M — Offers — 1 FACULTY

SUBJECT

SID	Name	FID
S01	BRM	F01
S02	ACCT	F02
S03	SEP	F01
S04	IIS	F01
S05	COMM	F02

STUDENT

STID	Name	Email
ST01	Miller	@
ST02	John	@
ST03	Steve	
ST04	John	
ST05	Jim	

ENROLMENT

STID	SID
ST01	S01
ST01	S02
ST02	S01
ST02	S02
ST03	S01
ST04	S01
ST05	S01
ST05	S02
ST05	S03

FACULTY

FID	Name
F01	FEIT
F02	BUS

OFFER?

SID	FID
S01	F01
S02	F02

Primary Keys

Foreign Key

Composite Primary Key

Entity Types

- An entity is a place, person, thing or an event or a concept about which we wish to record information.
- Entities will become a table in a database, one table for each entity.
- Entities can be classified into different types.
- Entity Types classify things about which we want to store data: e.g.
  - Places – e.g. country, park, shopping centre
  - People – e.g. customer, student, employee
  - Things – e.g. product, vehicle, building
  - Events – e.g. a purchase, accident, treatment
  - Concepts – e.g. law, crime, tax, fault

Hint: Look for **NOUNS** to identify Entities

Identify the entities/nouns in specification, requirements documents, interview transcripts, etc. that are **important to the business**.

Attributes

- All Entities have attributes (properties) that **describe the characteristics or are a description** of an entity. For example, a Student entity has Student Id, Name, Address, etc., as its attributes.
- All **instances** of an Entity type/set have the same attributes. For example, all Students will have student id, name, address , etc attributes.
- However the values of the Attributes will vary for each instance. For e.g., the student ids and student names will be different for each student instance.
- Attributes will become columns in a table in database.

Attributes: examples

- Examples:

  Student (Entity): Student Id, Name, DOB, etc

  Order (Entity): Order number, Quantity, Amount, etc

  Staff (Entity): Staff Id, Name, Department, etc

- Do not make an exhaustive list of attributes for each entity. Instead store only those attributes that are important to the organisation and the ones that are in scope of that particular project.
- Each instance of an entity (for example, each individual student instance) will become a record or row in a table in a database.

Entity Keys

- KEYS
  - are the basis of the relationship between entities
  - is a logical (and physical) pointer
  - is used to identify another entity that the current one wishes to be associated with

- Finding and identifying keys is an important step in understanding requirements for an information system.

Entity Keys – Primary Key

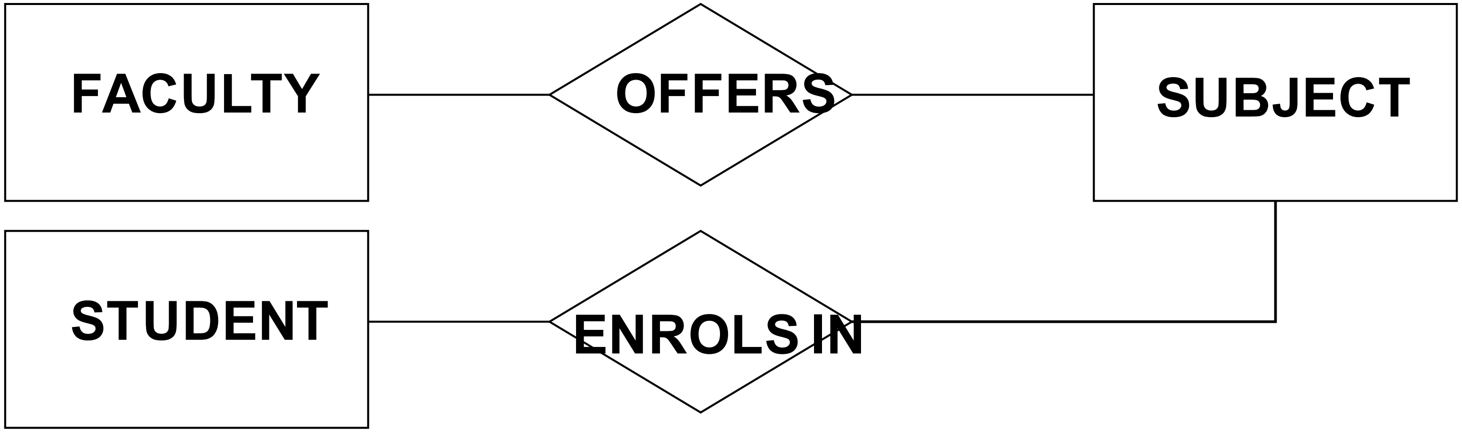
- **Primary Key:** an attribute or group of attributes that **uniquely identifies an instance of an entity**.
  - For example, **Student Id** is a primary key since it identifies an individual Student instance.
  - Another example; an **Employee Id** is a primary key for the Employee entity as it uniquely identifies each employee.
- The Primary key must be unique
  - absolutely no duplicates
  - Very often are numeric
    - have to be created to be unique

Entity Keys – Foreign Key

- **Foreign Key:** are attributes that are used to identify an instance of another Entity Type.
- When a Primary Key of one entity is used in another Record/Table, it is referred to as a 'Foreign Key'. For eg the primary key Department Id being used in Employee table
- Employee record/table
  - Employee Id (Primary key)
  - Name
  - Address
  - Drivers licence
  - Tax File Number
  - **Department Id (Foreign key)**
- Department record/table
  - Department Id (Primary key)
  - Department Name
  - Location

Relationship

- A **Relationship** connects entities to each other. It is an association among two or more entities. A relationship is indicated by **a verb** connecting two or more entities.
- Relationships are significant business associations between entities.
- Examples:
  - Each faculty **offers** many subjects
  - Each student **enrols in** many subjects



Cardinality

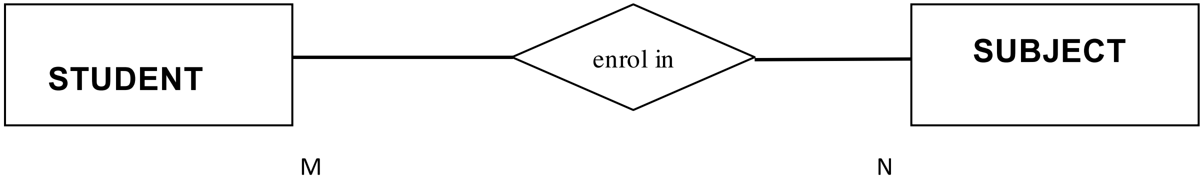
- Cardinality is a
  - business rule to indicate how many entities may participate in the given relationship between entities.
  - way to specify the number of occurrences of one entity in a data model that are linked to a second entity.
  - way to specify
    - how many times one instance of entity type can participate in **relationships** with the instances of another entity type.
  - **Numerical relation**
    - One department has many employees (1:M)
    - One subject has one lecturer (1:1)
    - Many tutors teach many subjects (M:N)

Cardinality

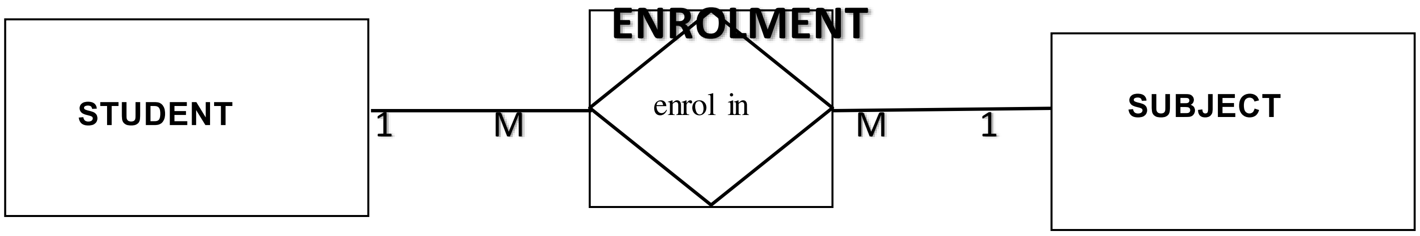
- Types :
  - **1:0** (Employee : Department) An Employee may or may not be assigned to a Department
  - **1:1** (must be one – Manager : Department) Each Manager heads one Department and vice versa
  - **1:N** (Company : Addresses) Each company has many addresses
  - **M:N** (Tutor : Subject) One Tutor teaches many Subjects, one Subject has many Tutors
  - **M:N** (Student : Subject) One Student enrolls in many Subjects, one Subject has many Students

M:N Relationships (Associative Entity)

- If you have a M:N (many to many) relationship you will need to create a new entity **called Associative Entity (ENROLMENT)** between the two entities that have M:N relationship.



- The new Associative entity (bridge entity) in most cases will have a **composite primary key**

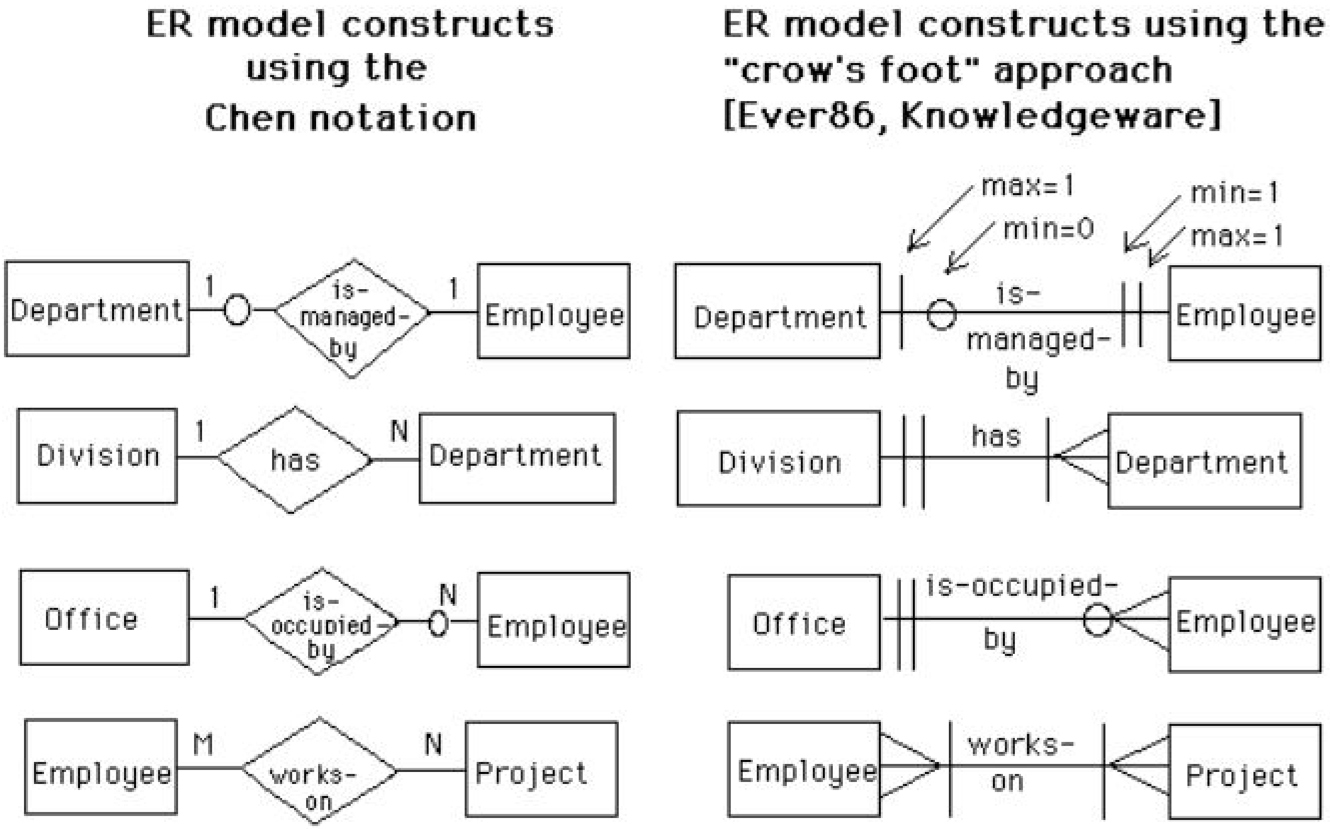


Composite Primary Keys

- Mainly used for M:N (many to many) relations
- A composite primary key uniquely identifies Student Subject attributes

- Student Subject Relationship (**is called “Enrolment” – Associative Entity)**
  - Student ID (part of Composite Primary Key)
  - Subject ID (part of Composite Primary key)
  - Semester
  - Mark
  - Grade

Chen and Crows Foot Notations for ERD

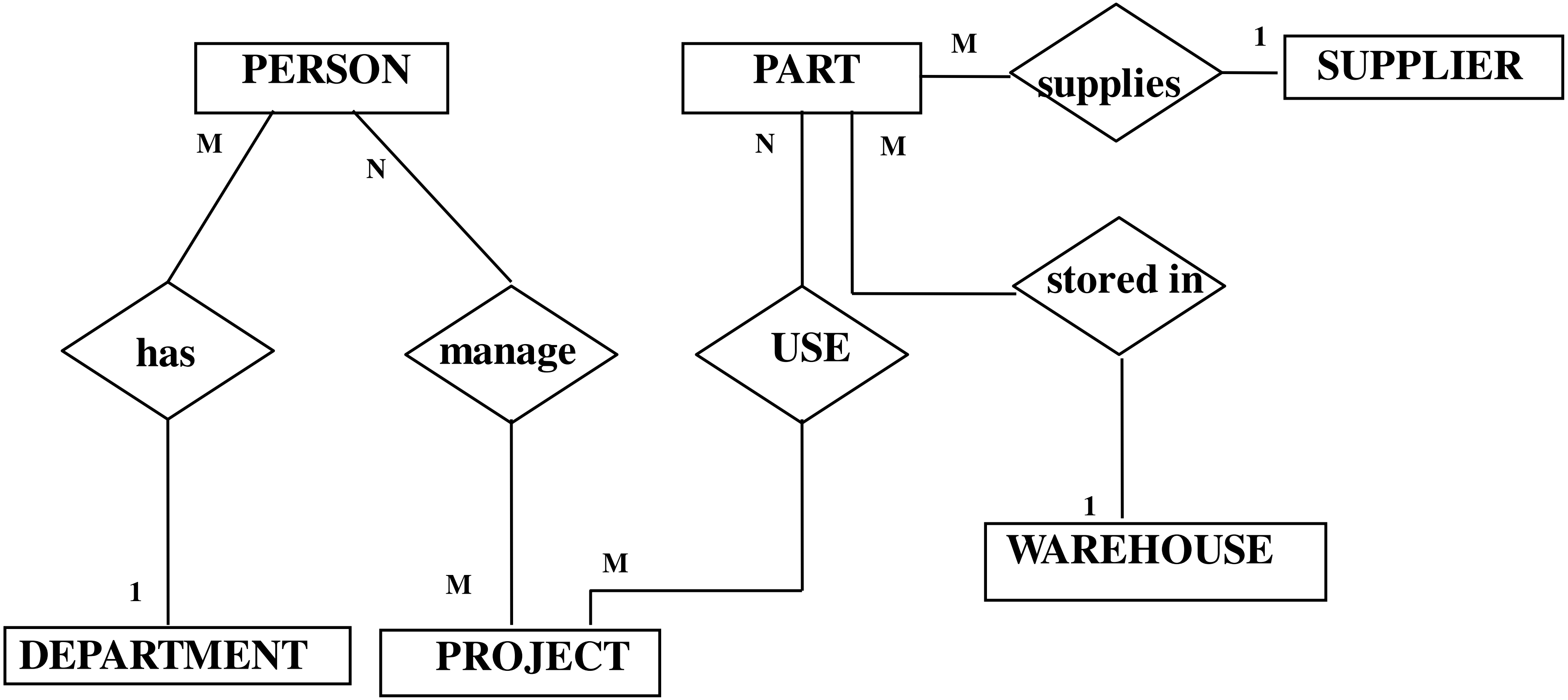


ER model constructs using the Chen notation

ER model constructs using the "crow's foot" approach [Ever86, Knowledgeware]

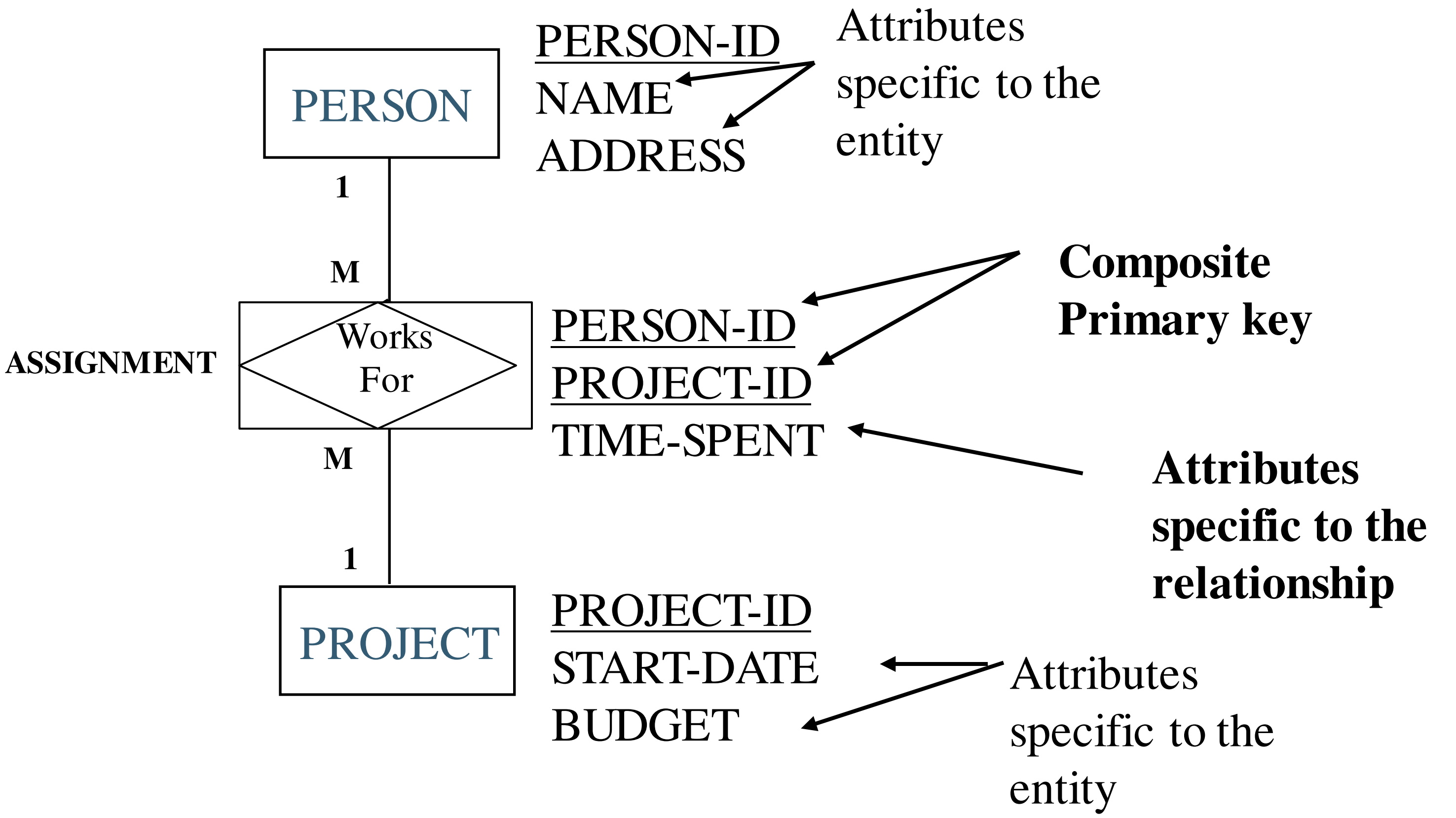
You can use either notation, but be consistent and do not use both notation styles in the same diagram.

An ERD Example (first draft)



- **ERD often contains many entities and relationship sets**

Associative Entity and Composite Keys



Steps for Drawing Entity Relationship Diagram

- Identify the **entities**/nouns in specification, requirements documents, interview transcripts, etc
- Identify the **attributes** for each entity
- Identify the **relationship** between entities
  - Name with a verb such that the diagram reads in simple sentences
- Identify **cardinality** for each relationship between entities as 1:1, 1:M, M:N, 0:1, 0:N
- Identify **keys** (primary and foreign keys) for each entity
  - Identify the attributes -The attributes(s) that uniquely identifies the entity
  - Identify the composite primary keys, if any associative entities exist

Data Dictionary

- Is a centralized repository of information about data such as meaning, relationships to other data, origin, usage, and format.
- A file that defines the basic organization of a database.
- A data dictionary contains a list of all files in the database, the number of records in each file, and the names and types of each field.
- Contains a **read-only** set of tables that provides information about the database.

Data Dictionary Example

Can hold letters and numbers (alpha numeric)

Ref.	Data Element	Format	Length	Description
01. Department				
DE-1001	Department ID	Number	3	Primary Key, Mandatory data field, must be numeric and must not be less than 3 digits i.e. 101
DE-1002	Department Name	Varchar	20	Name of the department. Sales, Marketing, Procurement Management
02. Staff				
DE-2001	Staff ID	Numeric	6	Primary Key, Mandatory data field, must be numeric and must not be less than 6 digits i.e. 100001
DE-2002	Staff Name	Varchar	50	Name of the Staff as Last Name, First Name i.e. Smith, John
DE-2003	Department ID	Numeric	3	Foreign Key, Refer to the Department ID attribute in the Department entity/table
03. Item				
DE-3001	Item ID	Varchar	10	Primary Key, Mandatory data field, must be alpha-numeric and must not be less than 10 digits i.e. COM-000001, LAT-000002
DE-3002	Item Name	Varchar	30	Name of the Item as Computer, Laptop etc.
04. Purchase Request (PR)				
DE-4001	PR ID	Varchar	10	Primary Key, Mandatory data field, must be alpha-numeric and must not be less than 10 alpha-digits i.e. PR-0000001
DE-4002	PR Date	Date	8	Mandatory field, must be in the following format dd/mm/yy
DE-4003	Staff ID	Numeric	6	Foreign Key, Refer to the Staff ID attribute in the Staff entity/table
05. PR Detail				
DE-5001	PR ID	Varchar	10	Composite Key (part of a composite key), Refer to PR ID in the PR entity/table
DE-5002	Item ID	Varchar	10	Composite Key (part of a composite key), Refer to Item ID in the Item entity/table
De-5003	Quantity	Numeric	**3**	Mandatory data field, must be numeric and must not be less than 0

Database Design

This slide to be read at home by students

- Direct database design
  - Entity are implemented as relational tables. For example, an Employee table.
  - Properties of Entities are implemented as columns (attributes) in the relational tables. For example, Employee Id, Name, Designation, etc.
  - Composite primary key of an associative entity is a primary key of a new relational table, primary key is also a foreign key here
  - If one to many cardinality, foreign key will be placed at the table of many cardinality (for example if a department has many employees, a foreign key will be a departmentID in an Employee table)
  - If one to one cardinality, foreign key can be placed at either table

Summary

- ERDs are the basis for the database design
- ERDs capture the data requirements of the system
- ERDs are structured around unique keys

Conclusion

- This Week's Workshop
  - **Quiz 3 - Process Modelling (3 marks)**
  - Workshop Tasks – Data Modelling

- Next Week's Lecture
  - Software Requirements Specification
- Next Week's Workshop
  - **Quiz 4 – Data Modelling (3 marks)**
  - Tasks – Software Requirement Specification

Note: By the end of this week you should be able to finish your assignment 1.