Data Modeling : ER Model

Why We use Model

- We build models of complex systems because we cannot comprehend any such system in its entirety
- Need to develop a common understanding of the problem and the solution
- Cannot afford a trial-and-error approach
- to communicate the desired structure and behavior of our systems

Why We Model…

- to visualize and control system's architecture
- to understand the system we are building, often exposing opportunities for simplification and reuse
- to manage risk

How We Model

- The choice of which model we use has a profound influence on how a problem is attacked and how a solution is shaped
- No single model is sufficient; every complex system is best approached through a set of independent models
- Every model may be expressed at different levels of fidelity
- The best models are connected to reality

OUTLINE

- Data model
- Concepts of entity and relationships
- E-R diagramming
- Keys
- Weak entities
- Extended E-R model

DATA MODEL

- for representation of a part of a real world
- it is an abstraction of the reality: ignores unnecessary details
- represents operational data about real world events, entities, activities, etc.
- model may be at various levels depending of requirements :
  - logical or physical
  - external, conceptual, internal

Data Model……

- a good model
  - is easy to understand
  - has a few concepts
  - permits top-down specifications
- model offers concepts, constructs and operations
- must capture meaning of data (data semantics) which help us in interpreting and manipulating data

Data Model……

- semantics captured through data types, inter-relationships and data integrity constraints
  - uniqueness
  - existence dependence
  - restrictions on some operations such as insertions, deletions

Example : Data Model in a PL

- data structuring concepts
  - data field/variable
  - data groups and arrays
  - Record/structure : unit of file i/o
  - file : collection of records

Example : Data Model in a PL ...

- Operations
  - file level : open, close
  - record level :
    - read next/random
    - write next/random
  - field level : computations
- no inter-file and inter-record relationships
- no constraints except primary key for indexed files

ENTITY-RELATIONSHIP (ER) MODEL

- for representation of real-world
- represents overall logical structure of information
  - grouping of data elements
  - inter-relationships between groups

ER MODEL….

- a few concepts
- simple and easy-to-use
- permits top-down approach for controlling details
- useful as a tool for communication between designer and user during requirements analysis and conceptual design

ENTITY

- an object that exists
- distinguishable from other objects
- could be concrete or abstract
- Examples : a book, an item, a student, a purchase order

(a/an above indicates that we are referring to one of these)

ENTITY SET

- a set of similar entities
- need not be disjoint with other entity sets
  - e.g., supplier and customer may have common entities
- Example : set of all books in a library
  - set of all customers
- entity set also called entity type or entity class
- entity considered as an occurrence of entity type

ENTITY SET……

- we often use the words ‘entity’ to mean ‘entity-set’
- entity sets are named using singular common nouns :

  Book

  Student

  Course

ATTRIBUTE

- an entity has a set of attributes
- attribute defines property of an entity
- it is given a name
- attribute has value for each entity
- value may change over time
- same set of attributes are defined for entities in an entity set

ATTRIBUTE….

- Example : entity set BOOK has the following attributes

  TITLE ISBN

  ACC-NO AUTHOR

  PUBLISHER YEAR

  PRICE

- a particular book has value for each of the above attributes

ATTRIBUTE….

- an attribute may be multi-valued, i.e., it has more than one value for a given entity; e.g., a book may have many authors
- an attribute which uniquely identifies entities of a set is called primary key attribute of that entity set
- composite attribute : date, address, etc

DOMAIN

- gives set of permitted values for an attribute
- all values may not be present at all times in database
- may be defined by type : integer, string
- attributes are ‘roles’ played by domains
  - domain ‘personname’ can be used for attribute ‘name’ for teacher and student entities

EXAMPLE : A COLLEGE

- **STUDENT : rollno, name, hostel-no., date-of-birth**
- **COURSE : courseno, name, credits**
- **TEACHER : empno, name, rank, room-no., tel-phone**
- **DEPT : name, tel-phone**

EXAMPLE : A COLLEGE…

- this example will be refined further
- perception of reality and focus of design could have indicated more entities
  - HOSTEL SEMESTER
  - Or, teacher could only be an attribute

EXERCISE : identify entities in a hospital and give a few instances of each

RELATIONSHIP

- represents association among entities
- e.g., a particular book is a text for particular course
  - book ‘Database Systems’ by C.J. Date is text for course identified by code ‘CS644’
- e.g., student GANESH has enrolled for course CS644

RELATIONSHIP SET

- set of relationships of same type
- words ‘relationship’ and ‘relationship set’ often used interchangeably
- between certain entity sets
  - binary relationship : between two entity sets
  - ternary relationship : among three entity sets

RELATIONSHIP SET….

- e.g., binary relationship set STUDY between STUDENT and COURSE
- relationship STUDY could be ternary among STUDENT, COURSE and TEACHER
- What is the difference ?
- a relationship may have attributes
  - e.g., attribute GRADE and SEMESTER for STUDY

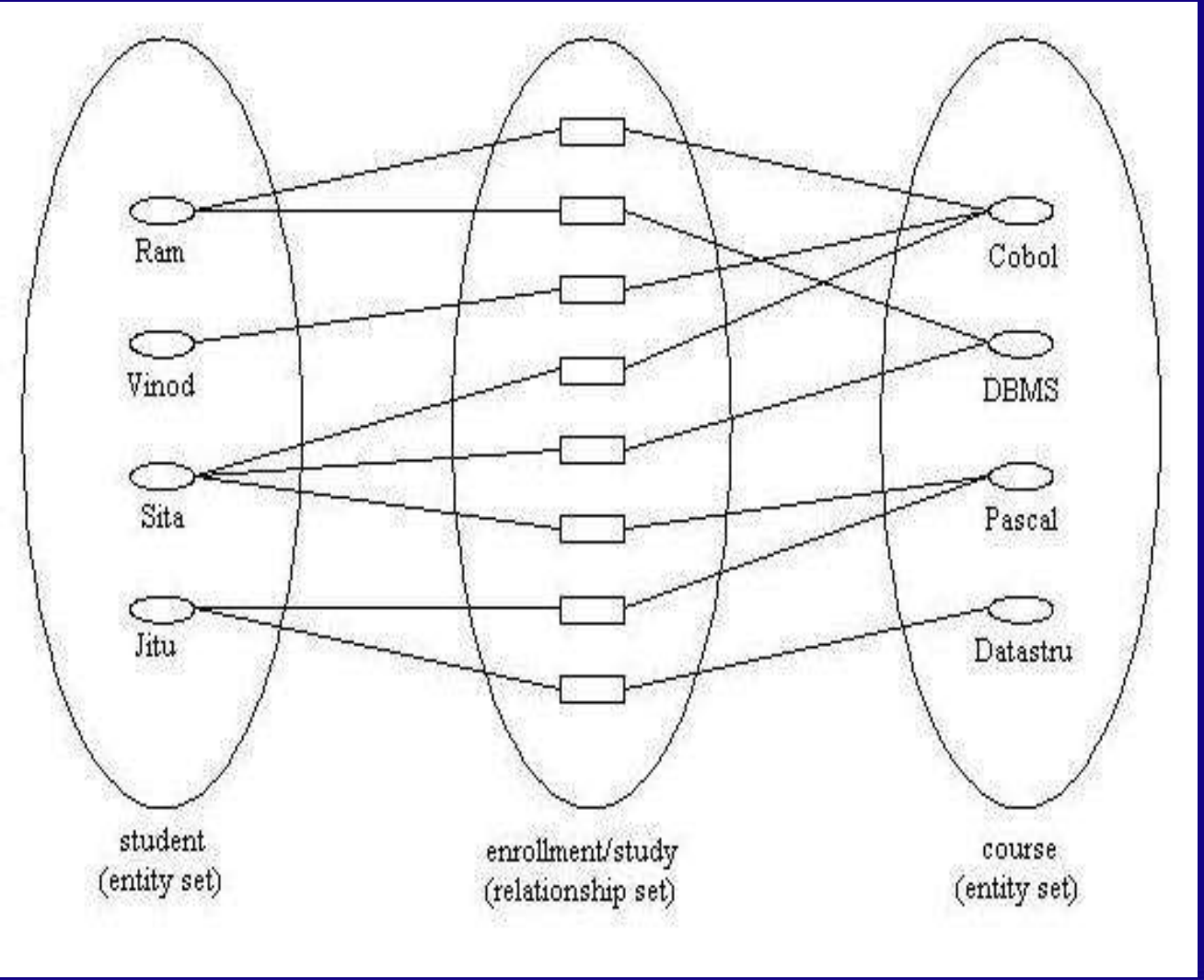
RELATIONSHIP SET….

- relationships named using verbs or nouns
  - Study
  - Enroll
  - Order

EXERCISE : identify relationships and their attributes in the hospital example and give a few instances of each

DEPICTING A RELATIONSHIP

- entity sets as a collection
- entity instances by small circles
- relationship instances by small rectangle with connections to involved entities


Ram
Vinod
Sita
Jitu
Cobol
DBMS
Pascal
Datastru
student
(entity set)
enrollment/study
(relationship set)
course
(entity set)

RELATIONSHIP CARDINALITY

- is a constraint on a relationship
- it characterizes relationships further
- given as (mapping) cardinality : how many entities of an entity set participate in a relationship
- especially useful for binary relationships

RELATIONSHIP CARDINALITY…

- a relationship set R between entity sets A and B may be one of the following
  - one-to-one : one entity in A associated with at most one entity in B
  - one-to-many : one entity in A may be associated with zero/more number of entities in B. However, one entity in B can be associated with at most one entity from A.
  - many-to-one : reverse of above definition (like a mathematical function)

RELATIONSHIP CARDINALITY…

- many-to-many : one entity in A may be associated with any number of entities in B, and vice-versa.

- EXAMPLES :
  - relationship set TEACHES from TEACHER to COURSE is one-to-many

    (TAUGHT-BY from COURSE to TEACHER is many-to-one)
  - relationship STUDY between STUDENT and COURSE is many-to-many

EXISTENCE DEPENDENCE

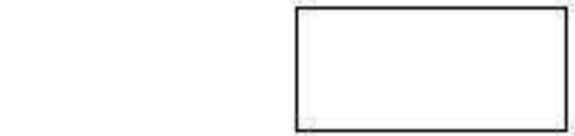
- existence dependency : another important constraint
  - existence of entity ‘a’ may depend on existence of another entity ‘b’:

    ‘b’ is called dominant entity and ‘a’ is called subordinate entity

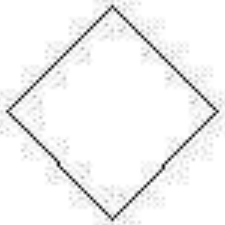
EXISTENCE DEPENDENCE…

- **there exists existence dependency of**
  - **TEACHER on DEPT as no teacher can be appointed without fixing her department**
- **Subordinate entity has its own key and may participate in more relationship**
- Whenever the existence of an entity depends on the existence of another entity, that relationship is described as an existence dependency (ED)

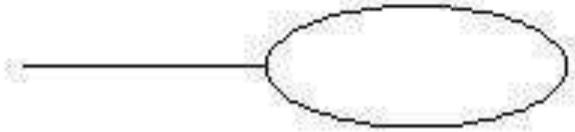
E-R NOTATION



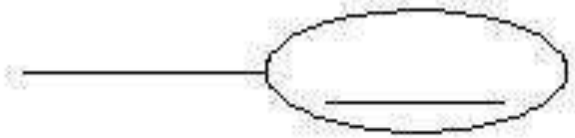
Entity (set)



Relationship

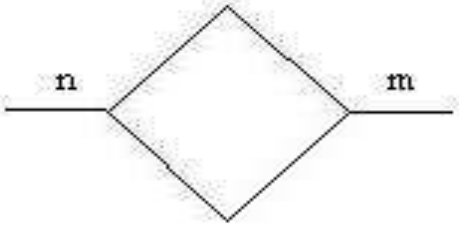


Attribute

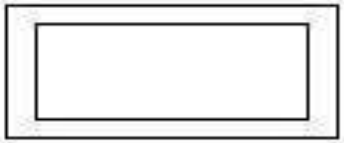


Key attribute

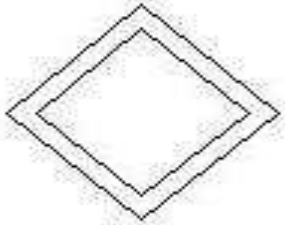
E-R NOTATION



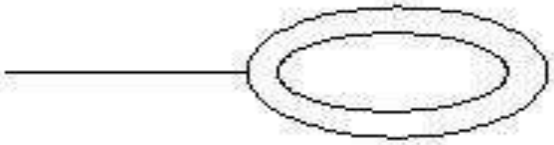
Cardinality



Weak Entity



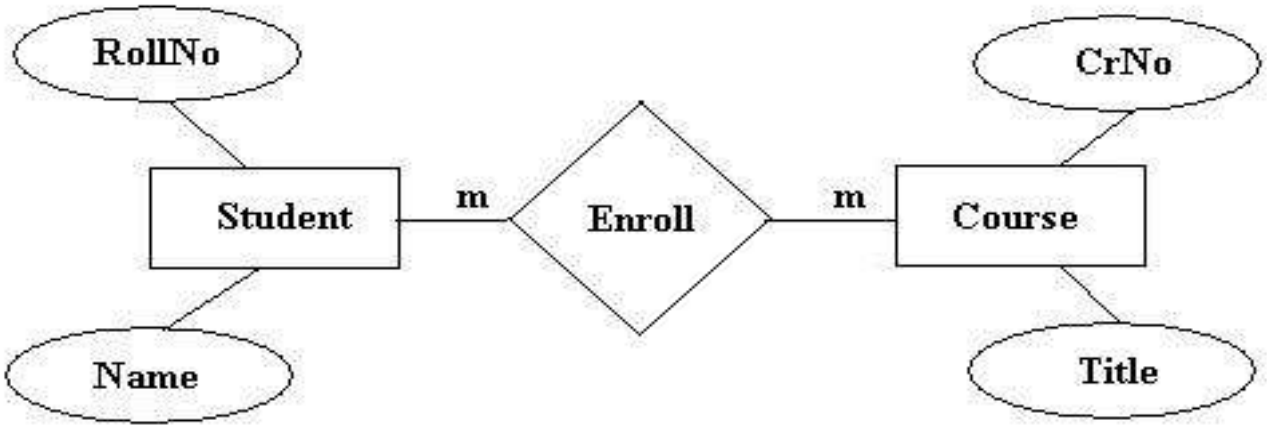
Strong-Weak Relationship

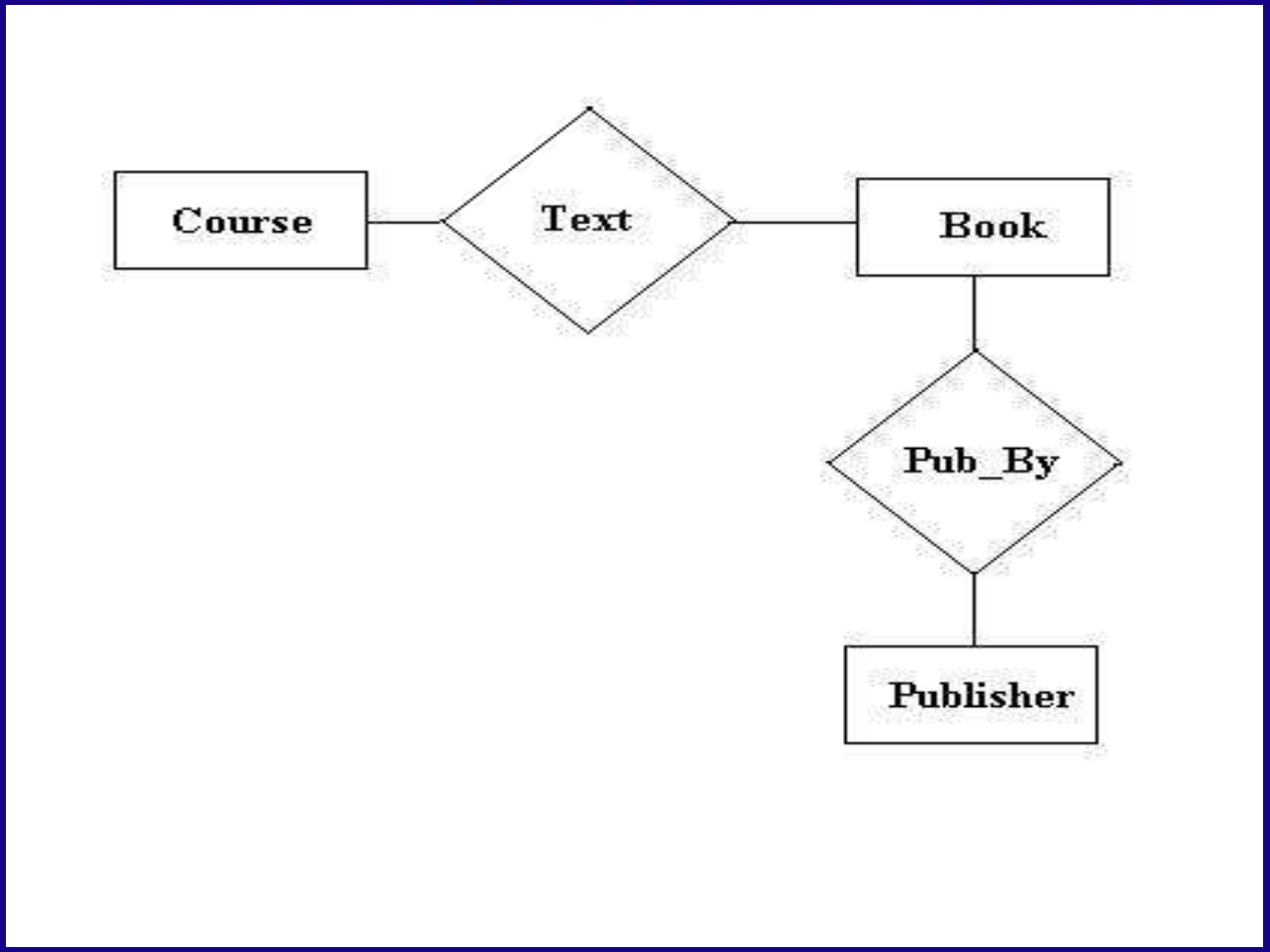


Multivalued attribute

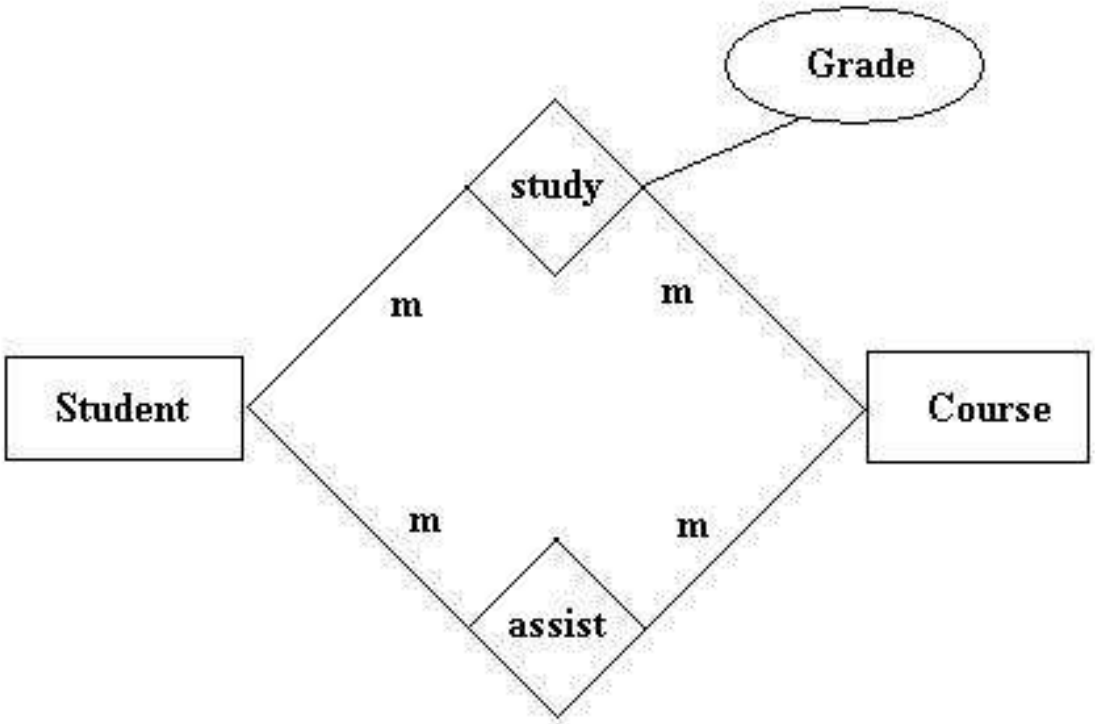
Note : some symbols explained later

E-R Diagram : Examples

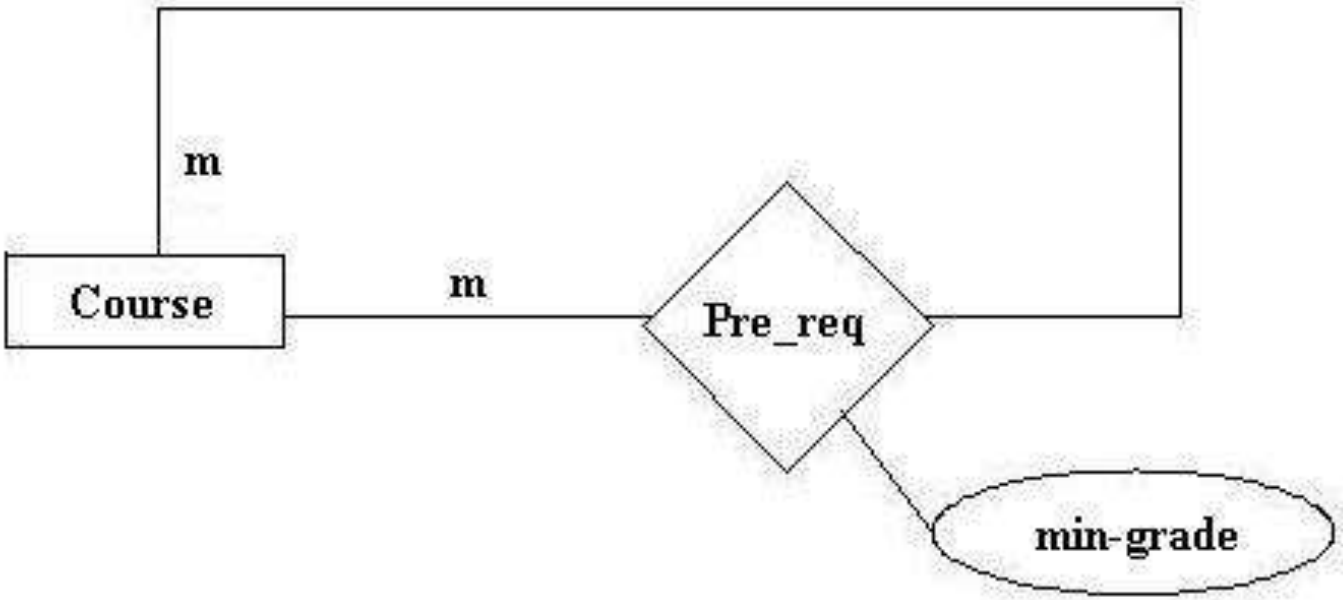




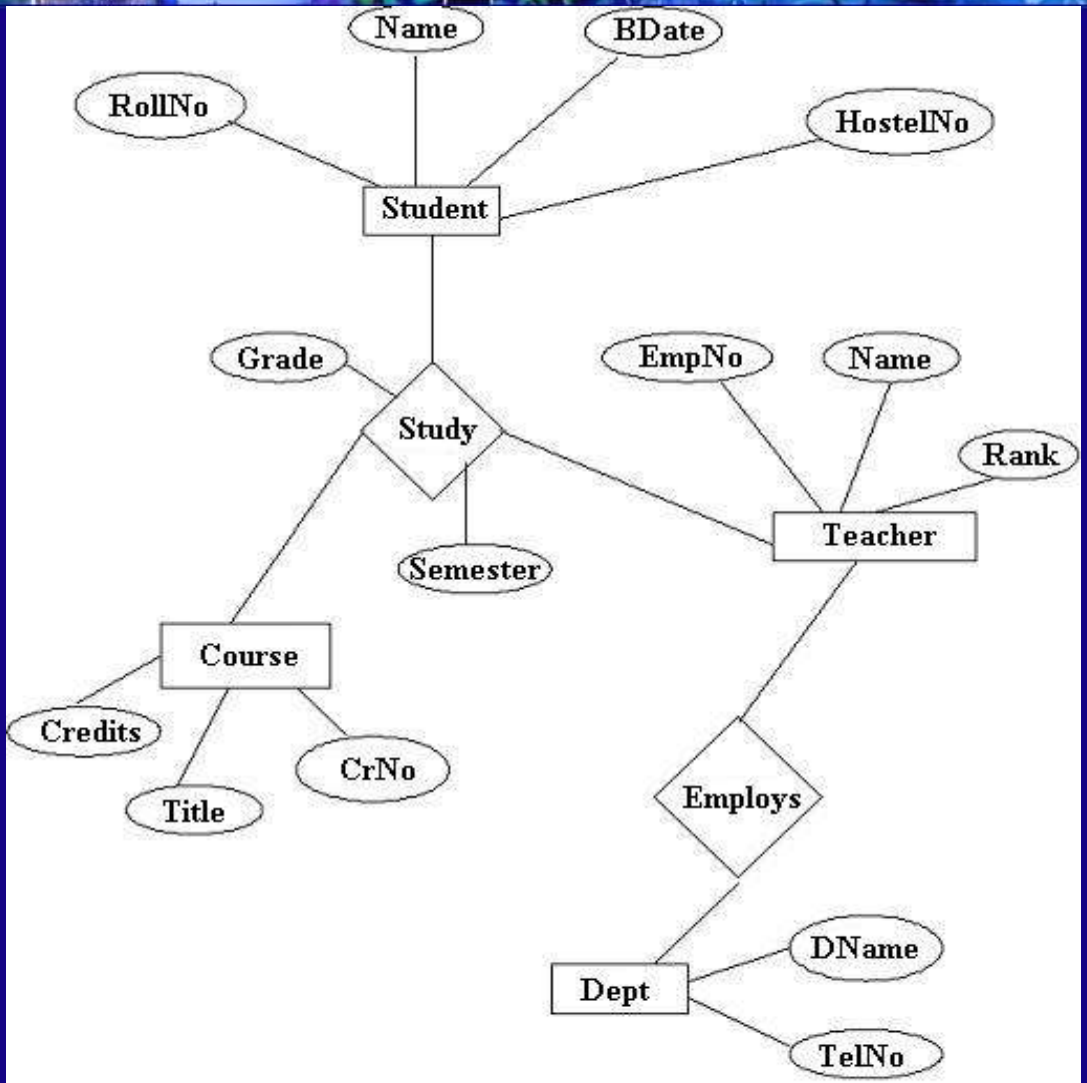
- Add some attributes to entities here
- Courses may have another course as pre-requisite



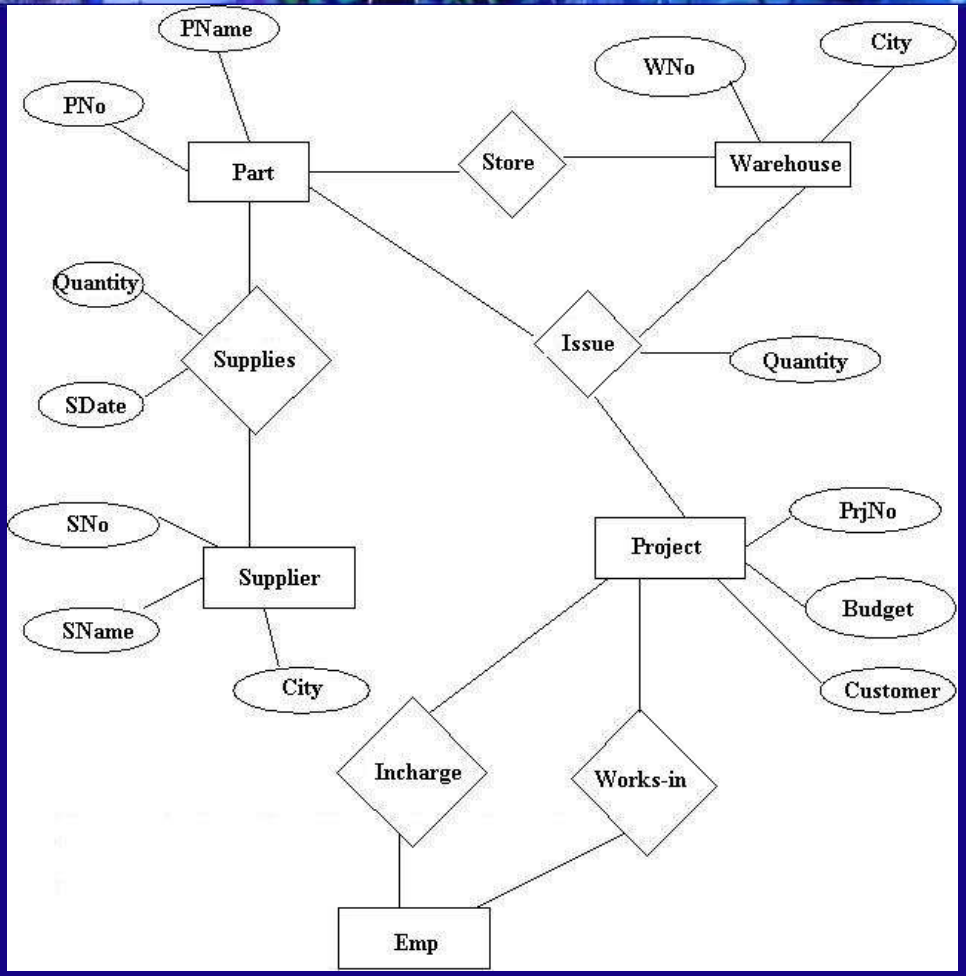
Shows multiple relationships among same entity set



Shows self-relationship: course may have other courses as pre-requisites, and min-grade defines minimum grade a student should have in a pre-requisite course



Add the fact that courses are offered by departments
Can a student repeat a course?



• Describe the real-world mapped above in words.

• Can you represents this : a supplier may supply same part many times

Note : Relationship 'supplies' could also be ternary
(by involving warehouse)

TERNARY RELATIONSHIPS

- be sure that your model reflects real-world correctly
- ternary (or, of higher order) relationships are harder to understand
- is a ternary equivalent to two binary? if not, which one is correct in a given situation?

TERNARY RELATIONSHIPS…

- consider shipments data where parts are supplied to projects by suppliers in certain quantities; given :

  S1 supplies 40 number of P1 to J1

  - we lose context if we replace it by

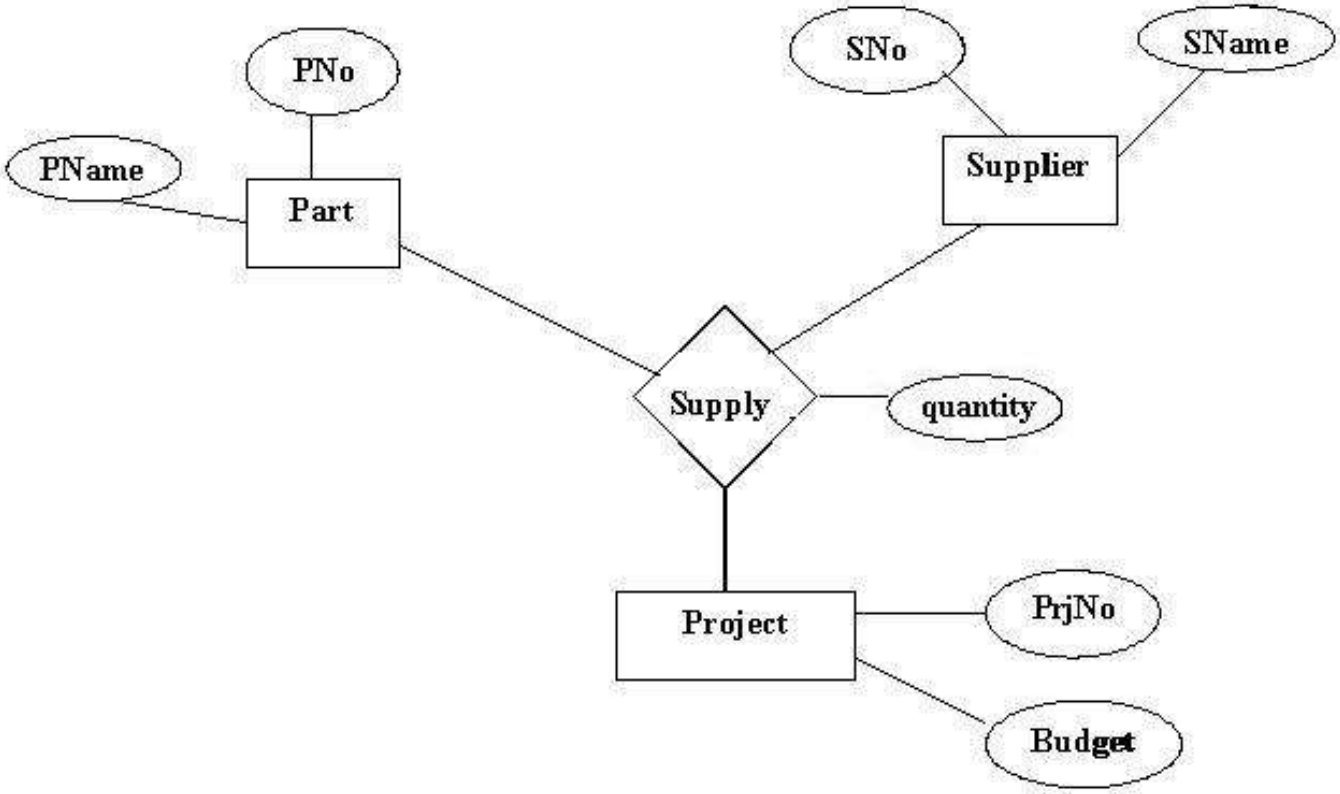
    S1 supplies 40 of P1

    S1 supplies to J1

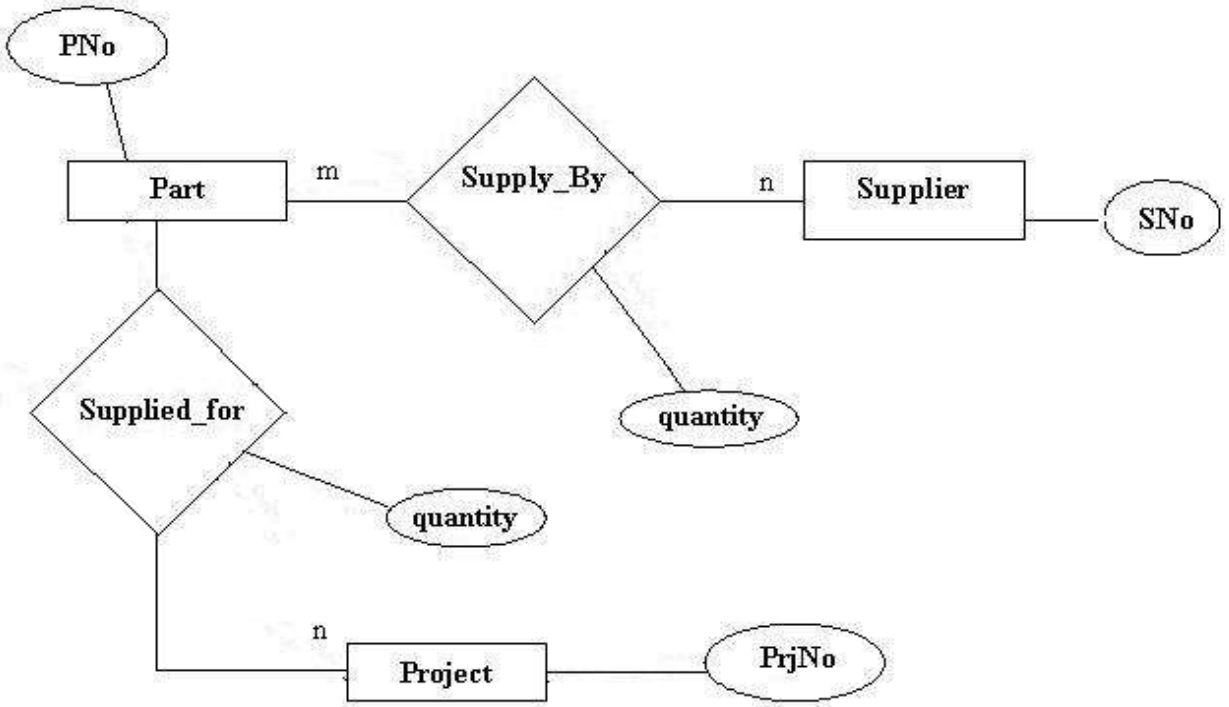
  - thus, ternary relationship is not same as two binary relationships

Exercise : Compare the following E-R Diagram with the one on next page using sample data

A: Ternary



B: Two Binaries



Can you point out "loss of data" in going from A to B above?
Will adding a relationship between supplier and project help?

PRIMARY KEYS

- to distinguish occurrences of entities and relationships
- distinction made using values of some attributes
- superkey : set of one/more attributes which, taken collectively, uniquely identify an entity in an entity set
- superkey may contain extraneous attributes

PRIMARY KEYS.....

- e.g., rollno is sufficient to identify students
  - it is a primary key
  - combination (rollno, name) is a superkey
  - name itself may not be sufficient as key
- candidate key is minimal superkey. No subset of it is a superkey
- an entity may have multiple candidate keys
- primary key is a candidate key chosen by designer as the principal means of identification

PRIMARY KEY FOR REPATIONSHIPS

- made of primary keys of all participating entities e.g., primary key of STUDY is

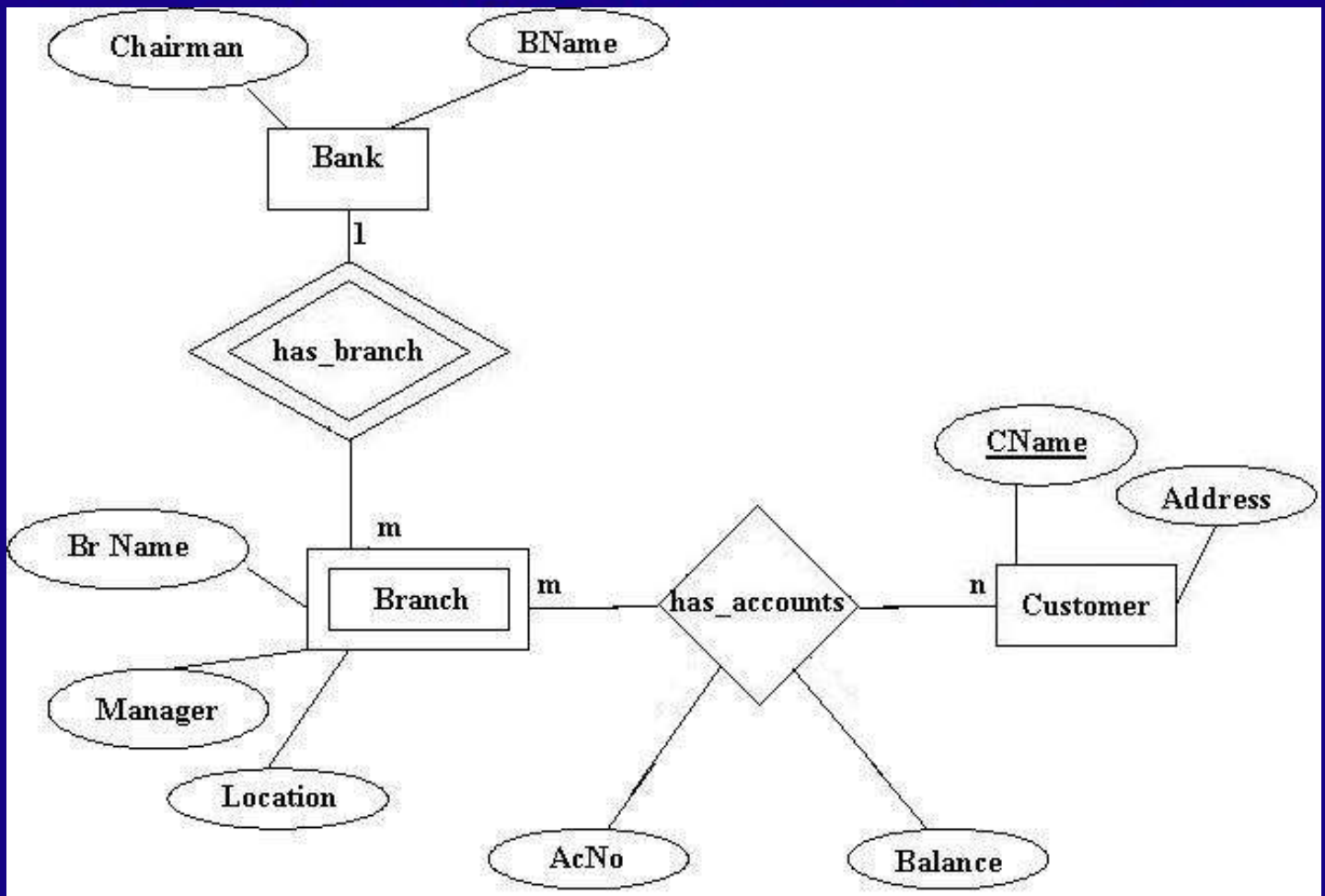
  (rollno, courseno)

WEAK ENTITY

- does not have a primary key on its own
- they are related to one/more ‘strong’ entities
- they often can be visualized as multivalued attribute or group of attributes
- they either have a ‘partial’ key or we add one to distinguish between those which are related to same strong entity

WEAK ENTITY…

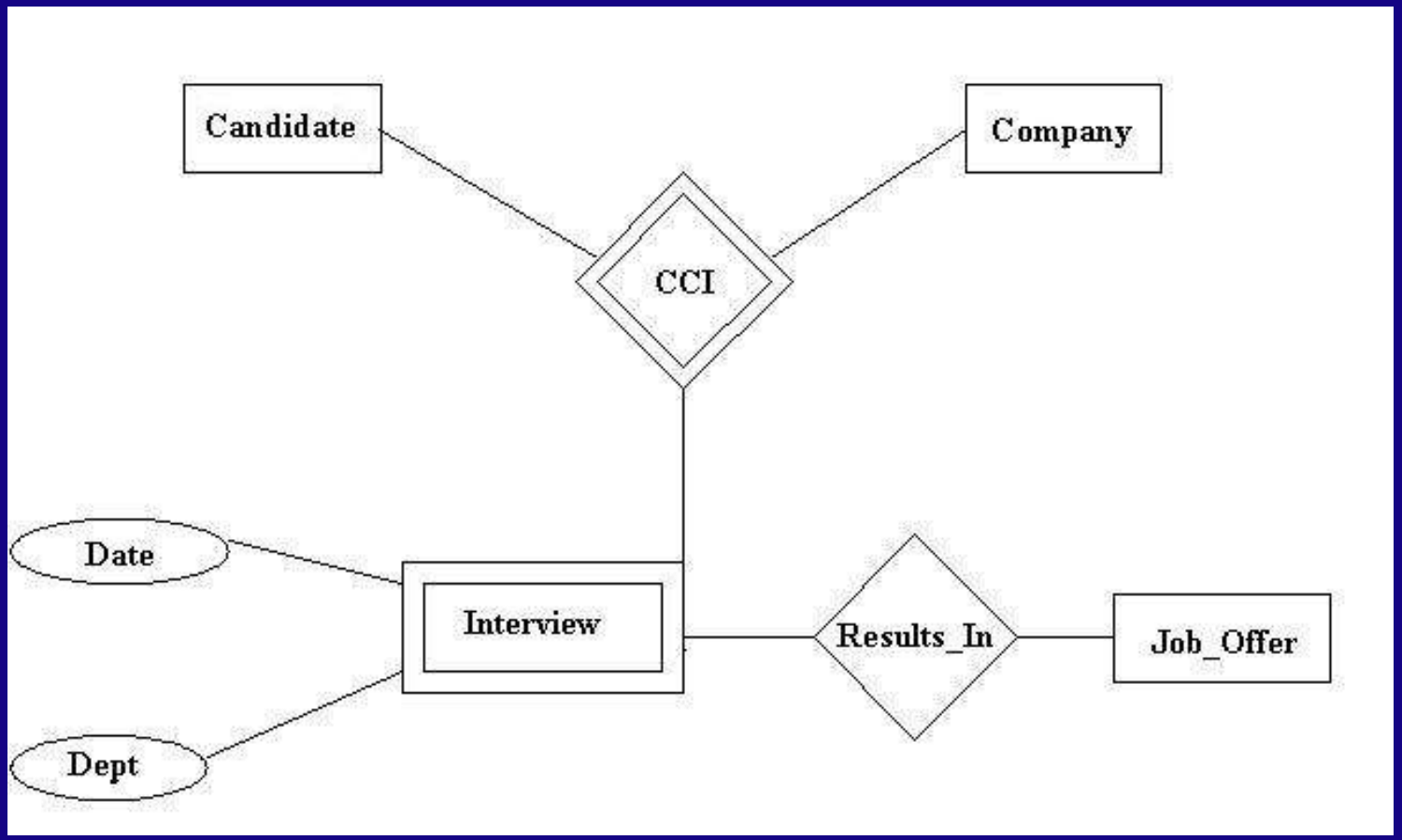
- examples:
  - branches of a bank
  - interviews between candidates and companies viewed as entities (not relationships) so that they can participate further in relationships
  - E-R diagrams follow



Make account an entity, possible held jointly by many customers

WEAK ENTITY…

- partial key (BrName in example) also called discriminatory attribute
- a weak entity can participate further in relationships with other entities
- a weak entity can also have weak entities dependent on in
- primary key of weak entity = primary key of its strong entity + discriminating attribute of weak entity within the context of strong entity



- shows weak entity depending on two strong entities.
- Taken from Elmasri/Navathe's book

EXERCISE (Post-Graduate studies)

Students join a particular specialization offered by a department. A specialization with same title (e.g., MICROCOMPUTER) may be offered by one/more depts independently. Teachers are appointed to a specific dept, and given a room and telephone. Depts have some teacher as its head. Courses are offered under various specializations. A teacher may teach many courses and a course may be taught by many. A student studies a course under a teacher during some semester (e.g., semester 1 of 1989), and is awarded a grade. A teacher's research interest may lie in one/more specializations. Courses have one/more/zero prerequisites

EXTENDED E-R MODEL

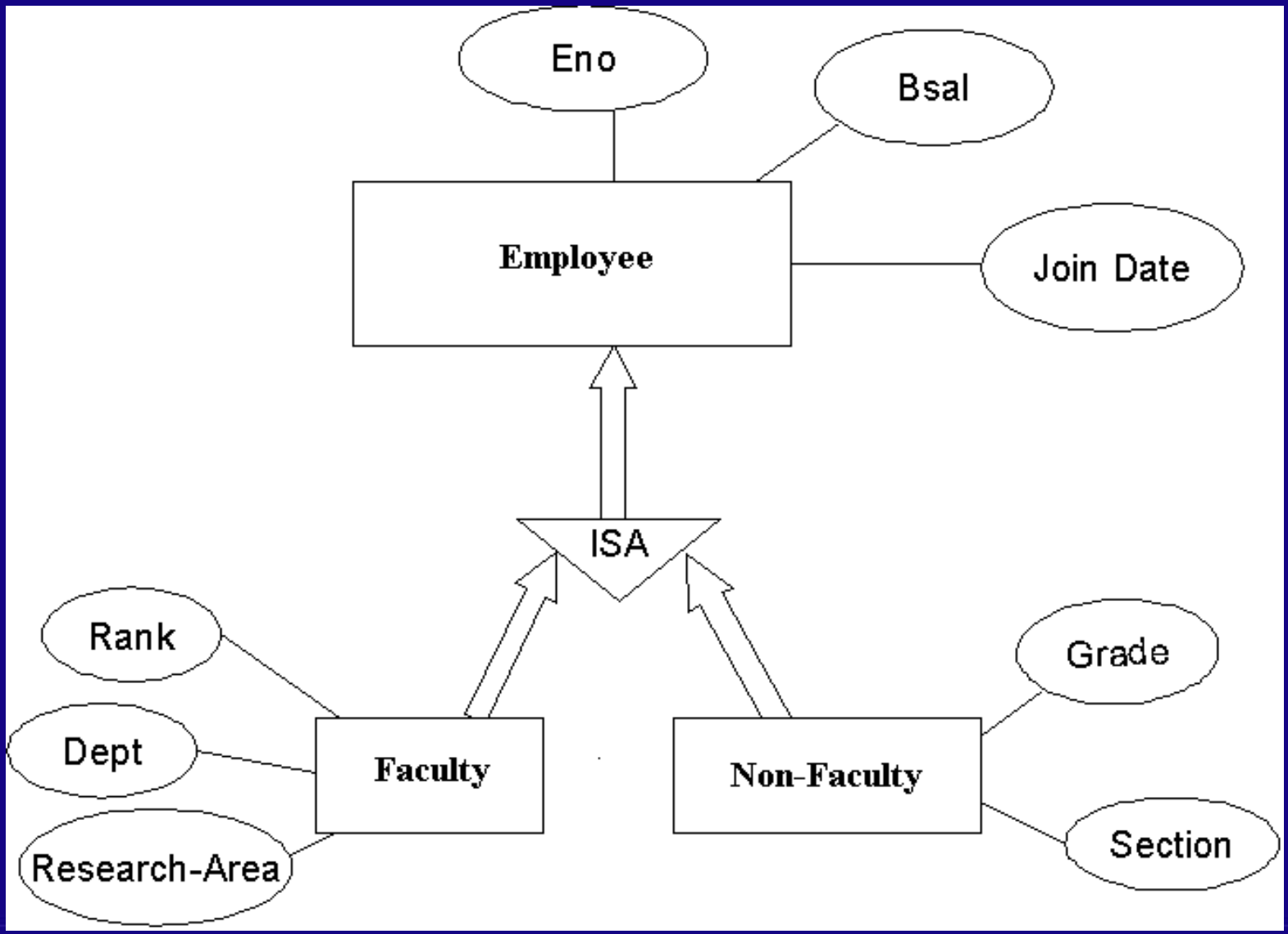
- extensions to capture more meaning
- concepts of generalization, aggregation and sub-set hierarchies added
  - Similar to OO concepts : inheritance, composite objects

Generalization

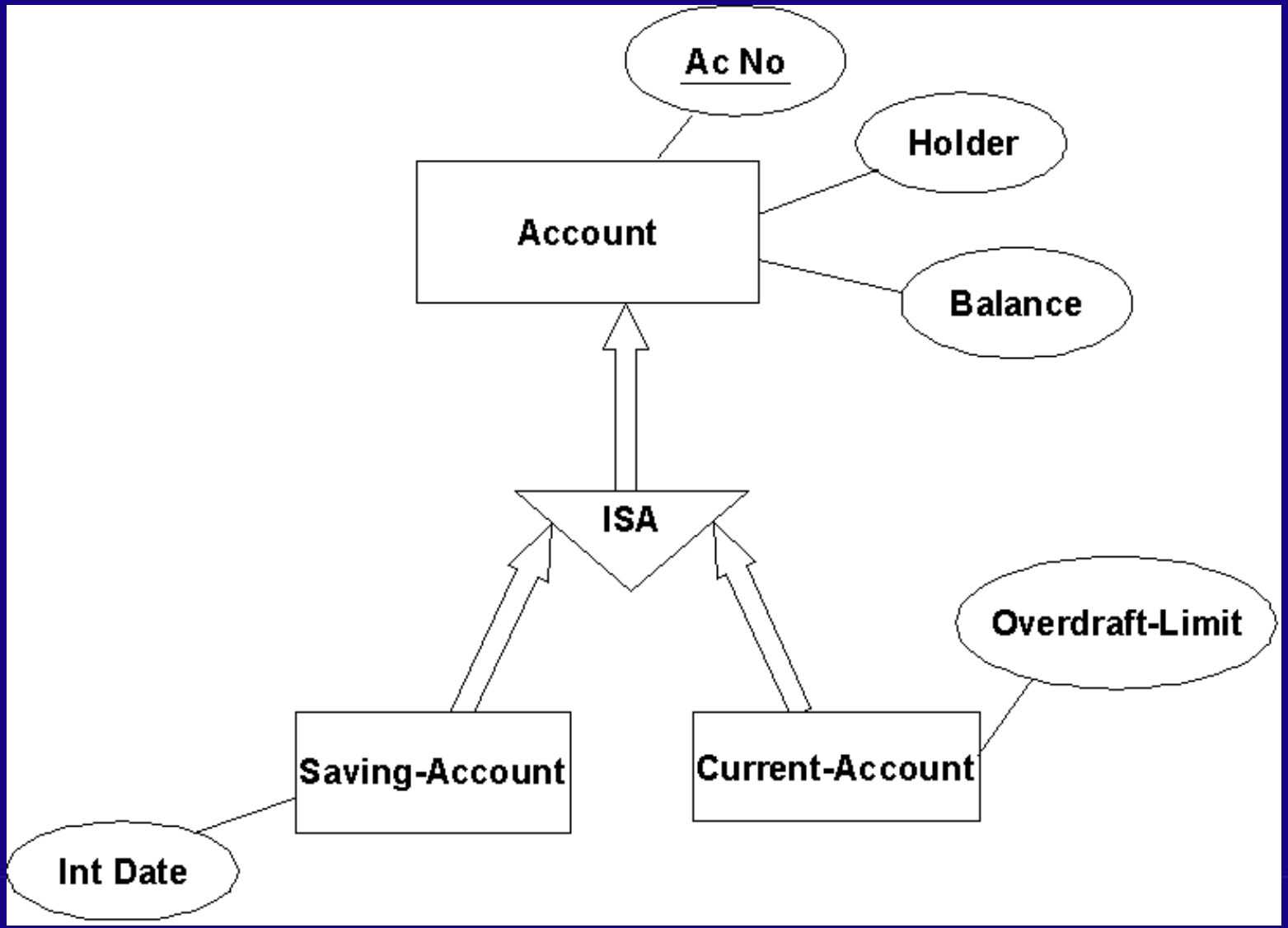
- to generalize from two or more entity sets and factor out commonality
- entity E is generalization of entities E1, E2, E3 … if each instance of E is also an instance of one and only one of E1, E2, etc.; E called superclass of E1, E2, …
- represented by IS-A relationship

Generalization…

- Example : given two entities Faculty and Non-faculty,we can define a ‘general’ entity called Employee
- Common attributes are factored out to define ‘Employee’ entity; specific (non-common) attributes incorporated in ‘Faculty’ and ‘Non-faculty’ entities


Eno
Bsal
Employee
Join Date
ISA
Rank
Dept
Research-Area
Faculty
Non-Faculty
Grade
Section

Another example :

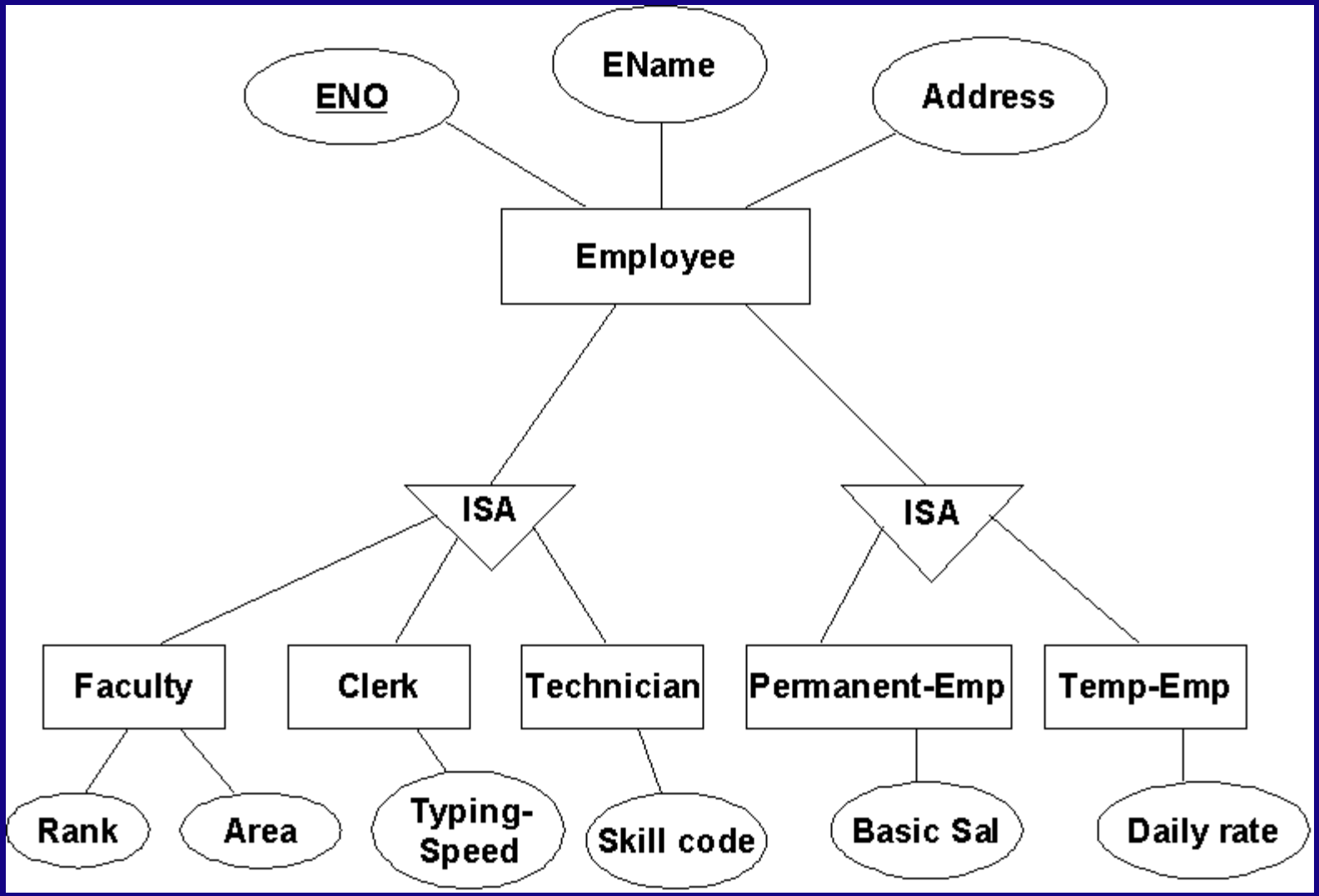


Specialization

- also called subset hierarchy
- entity E1 is subset of E if every instance of E1 is also an instance of E; this is also IS-A relationship
- E called superset and E1 as subset (or sub-class); E may have multiple and possibly over-lapping subsets
- every instance in E need not be present in subsets of E

Specialization.....

- specialization allows classification of an entity in subsets based on some distinguishing attribute/property
- we may have several specialization of same entity
- the subsets may have additional attributes


ENO
EName
Address
Employee
ISA
ISA
Faculty
Clerk
Technician
Permanent-Emp
Temp-Emp
Rank
Area
Typing-Speed
Skill code
Basic Sal
Daily rate

Inheritance

- there is inheritance of attributes from superclass or superset
- the subclass/subset automatically inherits attributes defined at superclass/superset level
- thus, inheritance present in both Generalization and specialization
  - Direction important : bottom-up in generalization, top-down in Specialization
  - Important to distinguish the two cases

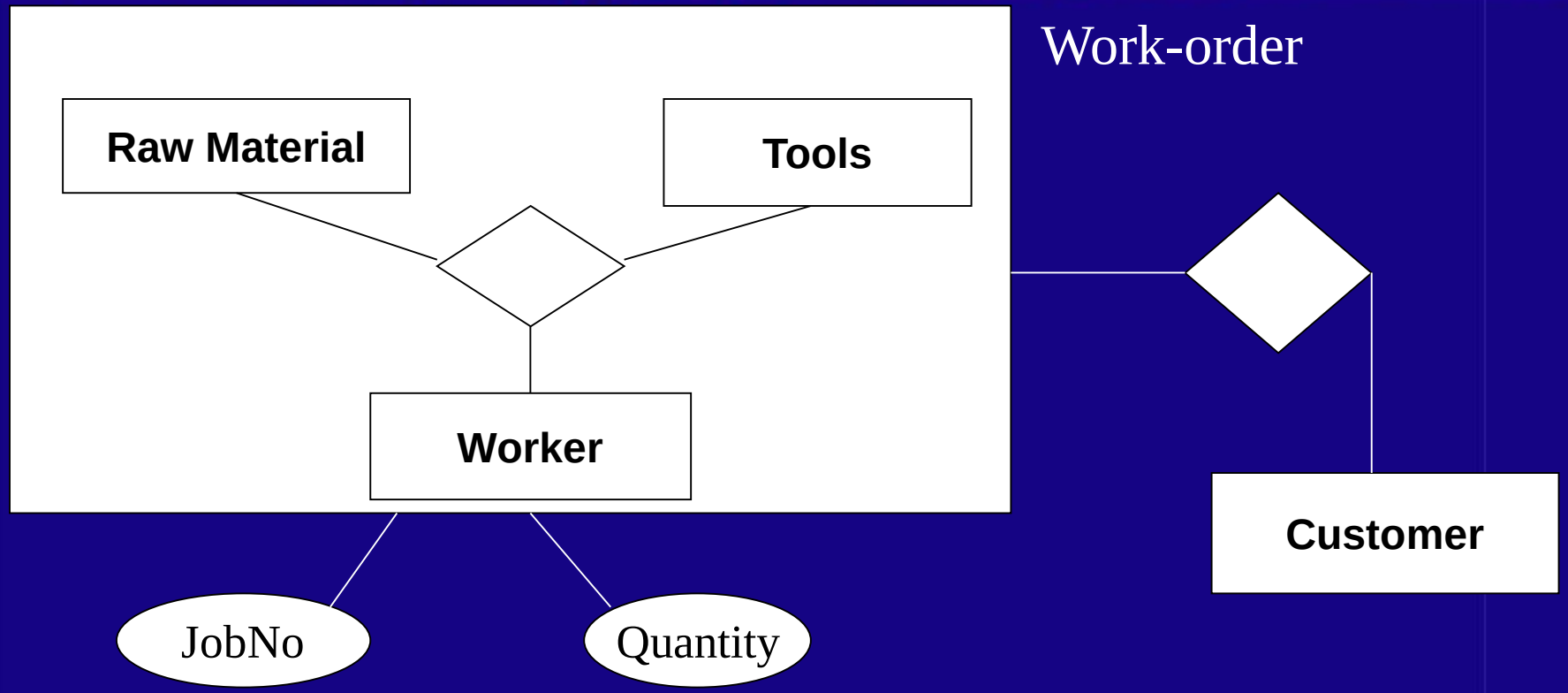
Aggregation

- for building complex entity from existing entities (or existing entities and relationships)
- two ways of defining complex entities :
  - create an attribute whose value is another entity
  - define an entity as containing a group of related entities

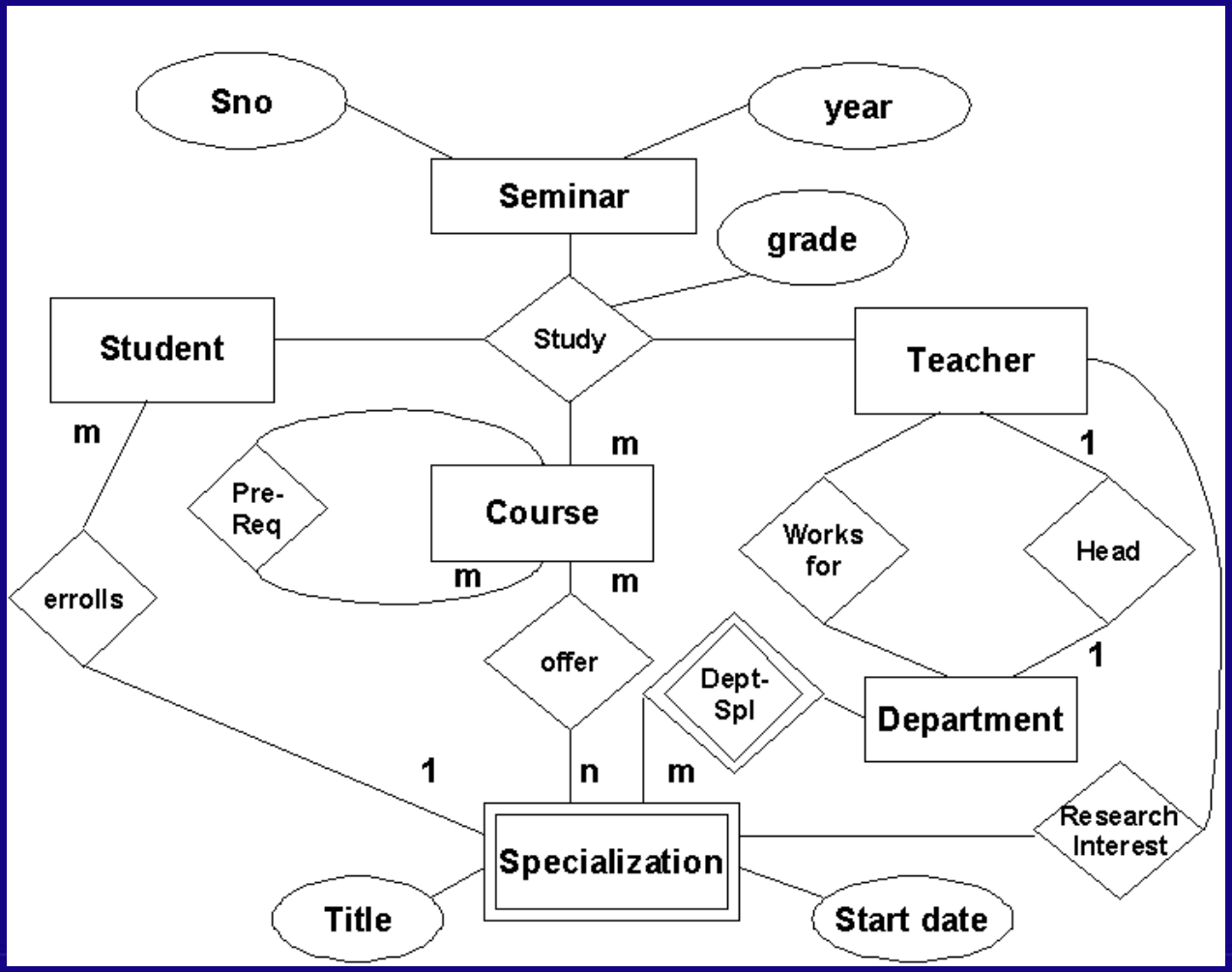
Examples :

- Work-order object (entity) defined as consisting of entities Raw-material, Tools and Workers;
- Work-order itself related with Customer entity
- Aggregation notation not explicitly provided in Extended E-R model

Work-order



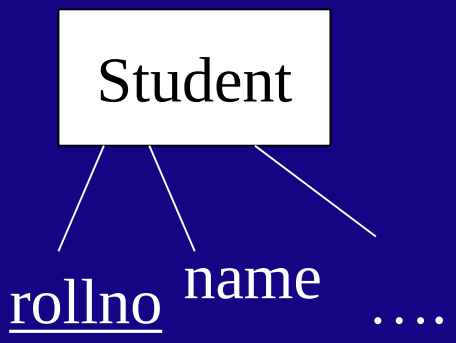
(ANSWER TO EXERCISE)



Going from E-R to Relational Data Model

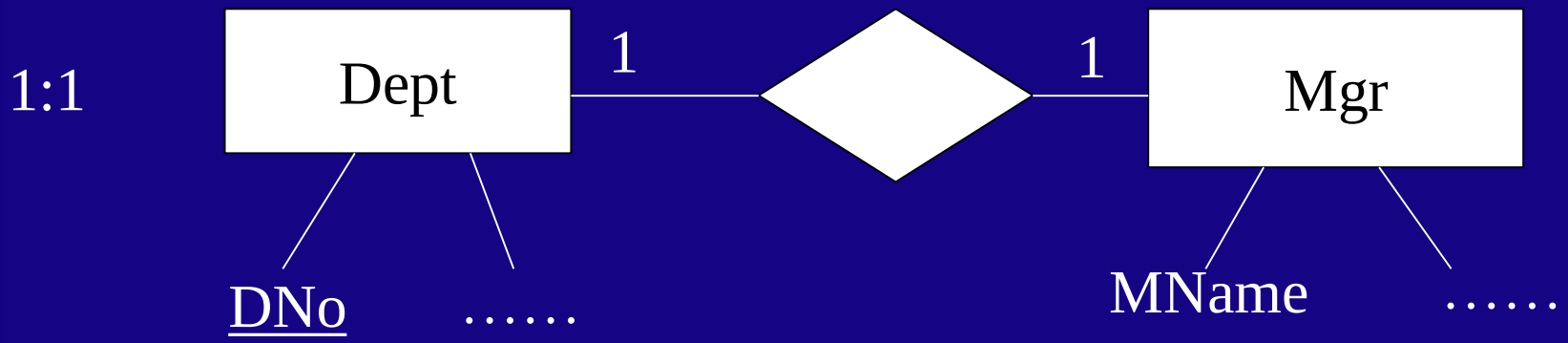
Need to match
ER model concepts : entity, relationship, attribute
with
Relational model concepts : relation, attribute

entity



Student (rollno, name, ….)

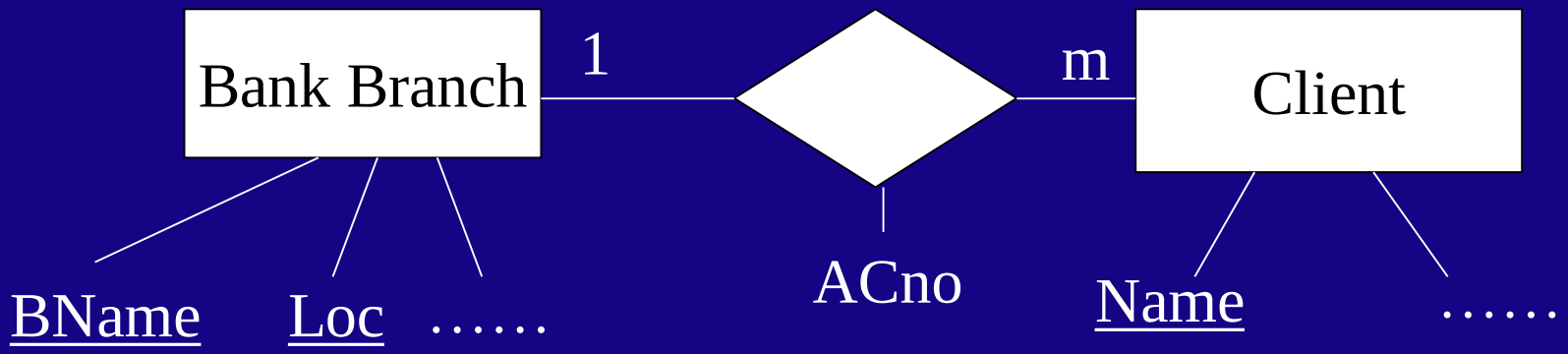
E-R to Relational …



DEPT (DNo , ……. , MName)
MGR (MName , ……, DNo)

also keys; in one or both

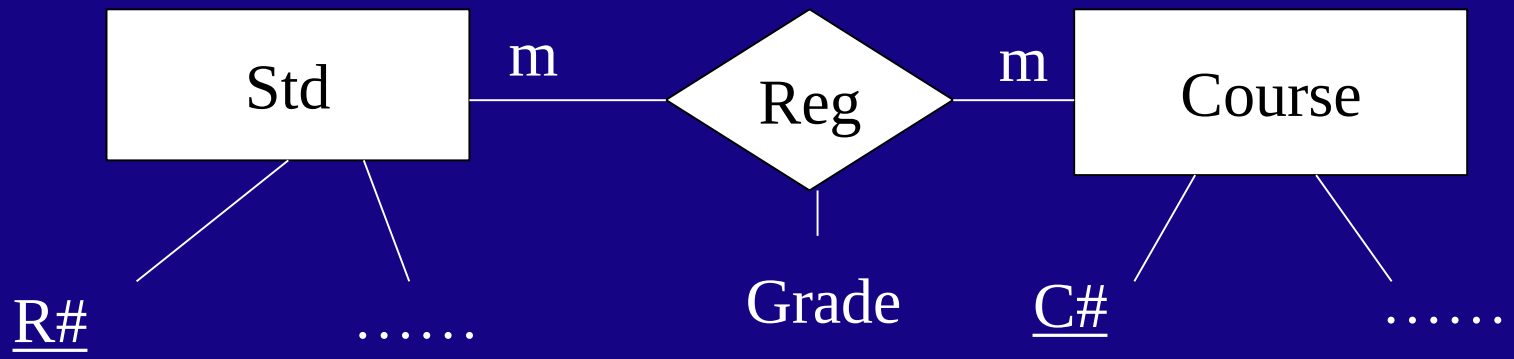
E-R to Relational …



BKBranch (BName , Loc , …….)
Client (Name , ……, BName , Loc , ACno)

The relationship is included in the entity on ‘many’ side;
It includes primary key of entity on ‘one’ side and
Relationship attributes, if any

E-R to Relational …



STD (R# , …….)
COURSE (C# , ……)
REG (R# , C# , Grade)

Exercise: Airport database

- keeps track of airplanes, their owners, airport employees and pilots
- Each airplane has a registration number, is of a particular plane type and is stored in a particular hanger. Each plane type has a model number, capacity and weight. Each hanger has a number, capacity and location. The database also keeps track of who owns which plane. Persons have name, address and phones. A person buys a plane on a particular date and cost.

Airport database….

- Each plane undergoes service many times. A service information contains date of work, nature, hours spent, cost, etc. Pilots and employees are persons. Pilots have a license number with validity and salary. Employees have a number, rank and salary. Each pilot is authorized to fly certain types of planes. Employees are involved in servicing of planes.

- prepare E-R model
- convert to the relational model

Exercises

Prepare E-R models and convert to relational schema

- **Railway Reservation**
  - **30 days in advance**
  - **trains, stations, quotas, coaches**
  - **passengers, tickets, wait-list, etc.**

Exercises…

- Old Car Mart
  - buying and selling of old cars
  - cars, purchases, sales – direct or installment-wise
  - service to sold cars, pre-sale repairs
  - agents
- Cricket Database
  - countries, players, teams
  - matches, results, scores
  - Prepare sample data

Exercise : Portfolio Management

- for individual investors
- investments are made in shares, debentures, bonds, National saving certificates, various schemes like PPF, ELSS, mutual funds, etc.
- these may be acquired at public issue time, purchased from market, obtained as bonus (free), on ‘rights-basis’, etc.

Portfolio …

- some investments have regular returns; e.g., yearly, 6-monthly, etc. at fixed or announced rates
- these will be sold in market or re-deemed, converted, etc.
- how, when and how much invested, what returns already obtained are important for the investor to know how good are his investments. At year end, he may wants to know the market value of his investments

Portfolio …

- Exercise :
  - Draw E-R diagram
  - convert to relational scheme
  - check if all relations are in 3NF

QUALITY AND COMPLETENESS CHECKS

- **ENTITIES**
  - **are they really entities? i.e., things of real significance about which information needs to be held**
- **Checklist**
  - **singular meaningful name**
  - **mutual exclusivity**
  - **at least 2 attributes (and < 8)**
  - **synonyms/homonyms**
  - **full definition**

QUALITY AND COMPLETENESS CHECKS…

- ENTITIES (Checklist…)
  - volumetric information
  - a unique identifier
  - at least one relationship
  - at least one business function to create, update, delete, archieve and use the entity
  - changes over time
  - it (functionally) determines its attributes
  - is it too generic ?
  - is it sufficiently generic ?

QUALITY AND COMPLETENESS CHECKS…

- ATTRIBUTES
  - do they really describe the particular entity ?
- Checklist
  - singular meaningful name
  - name not to include entity name
  - only one value : no repeating / group value
  - complete metadata (format, allowed values, etc.)
  - is it really an entity
  - value depends only on the entity (not on part of identifier or some other attribute)

QUALITY AND COMPLETENESS CHECKS…

- Relationships
  - are they really significant associations ?
- Checklist
  - each end named and capable of being read accurately and sensibly
  - each end has a degree and optionality
  - is it redundant
  - does it cater for time
  - check arity
- Try populating the E-R model

Entity/Process Matrix

- after building data model and defining elementary processes, create a matrix with entities along columns, processes along rows
- fill entries indicating which processes create entities, and which read, update and delete the entities
- Also called CRUD (Create, Read, Update and Delete) matrix

Relationships between diagrams

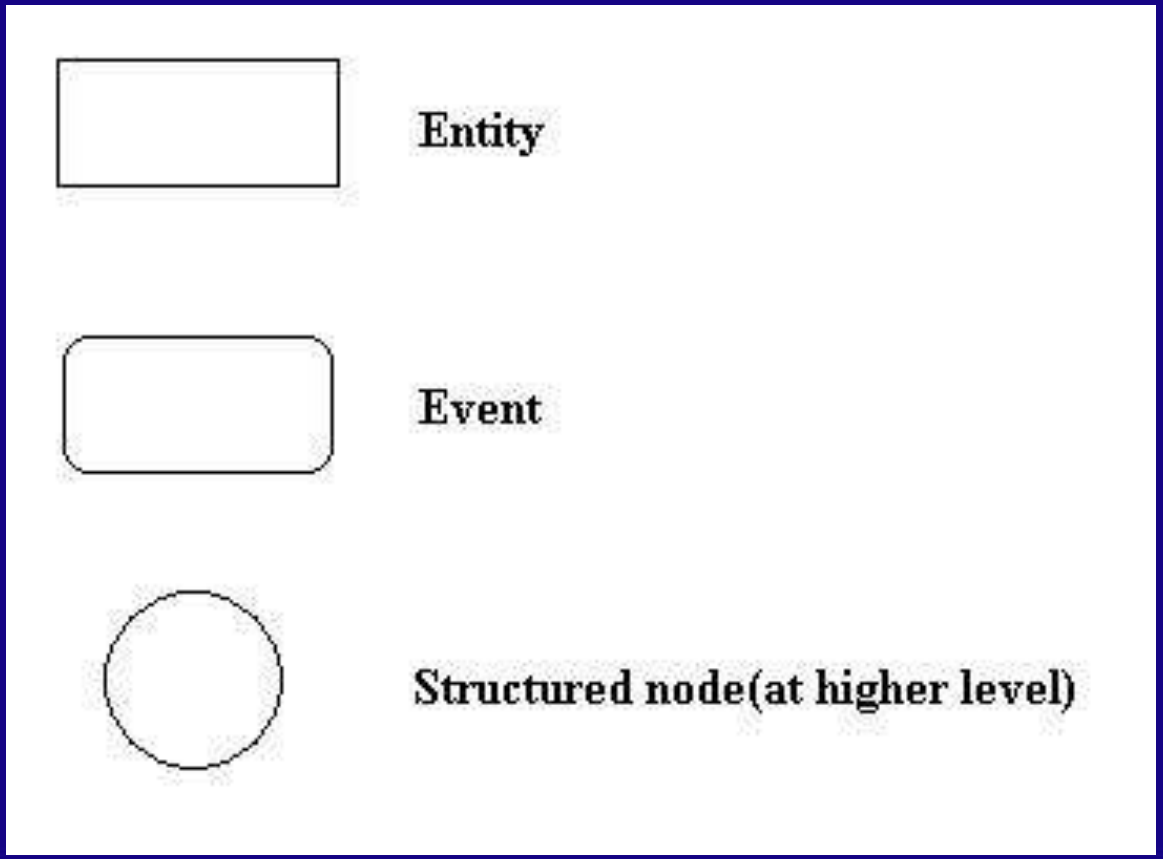
- FDD identifies processes
- These processes shown in DFDs
  - DFDs give process dependencies with data interactions added
- Data stores in DFDs are basis for E-R diagram
- These three diagrams should be consistent

ENTITY LIFE HISTORY (ELH)

- depicts pictorially the events affecting life of an entity from creation to deletion
- dynamic history : state changes over time are depicted
- events may be triggered by input or time
- possible effects of events : creation, deletion, modify attributes of entities, relationships or both
- ELH Notation

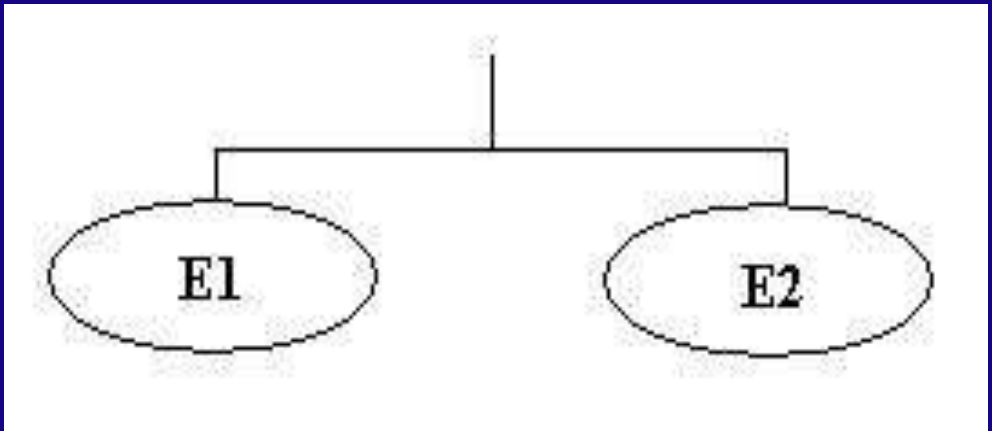
ENTITY LIFE HISTORY (ELH)..

- **ELH Notation**



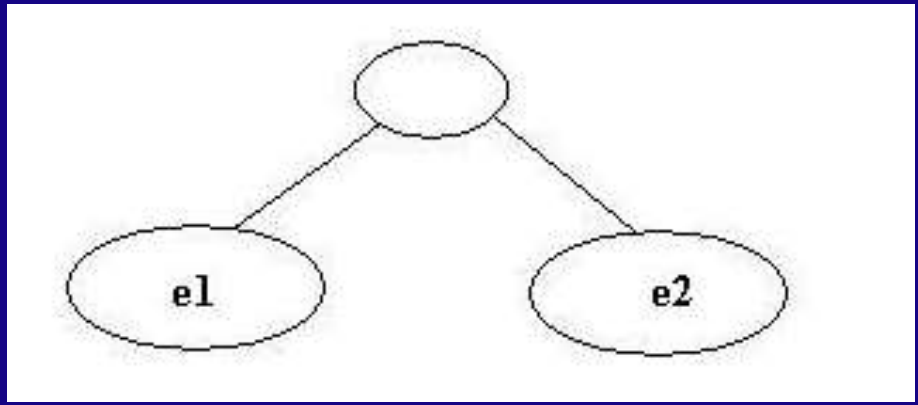
ENTITY LIFE HISTORY (ELH)..

- ELH is a tree with an entity type as its root
- Event compositions
  - sequence of E1 and E2

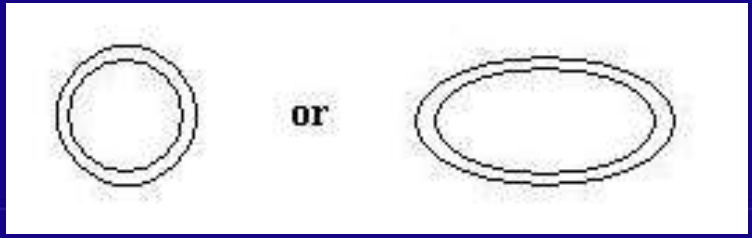


ENTITY LIFE HISTORY (ELH)..

- **Selection : e1 or e2 (one-of)**

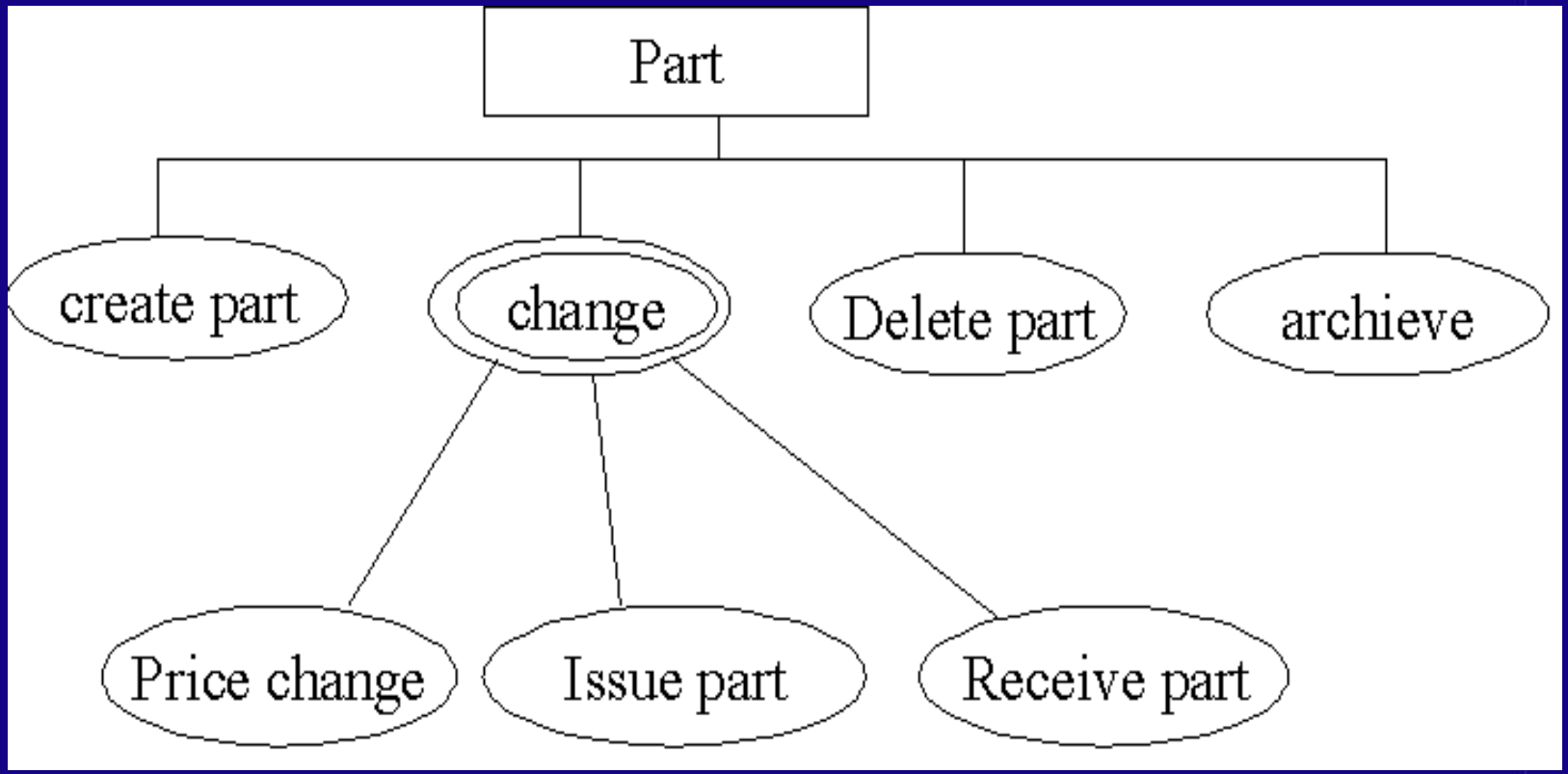


- **iteration :**



ENTITY LIFE HISTORY (ELH)..

- **example**



ENTITY LIFE HISTORY (ELH)..

- **Parallelism between events may be shown :**