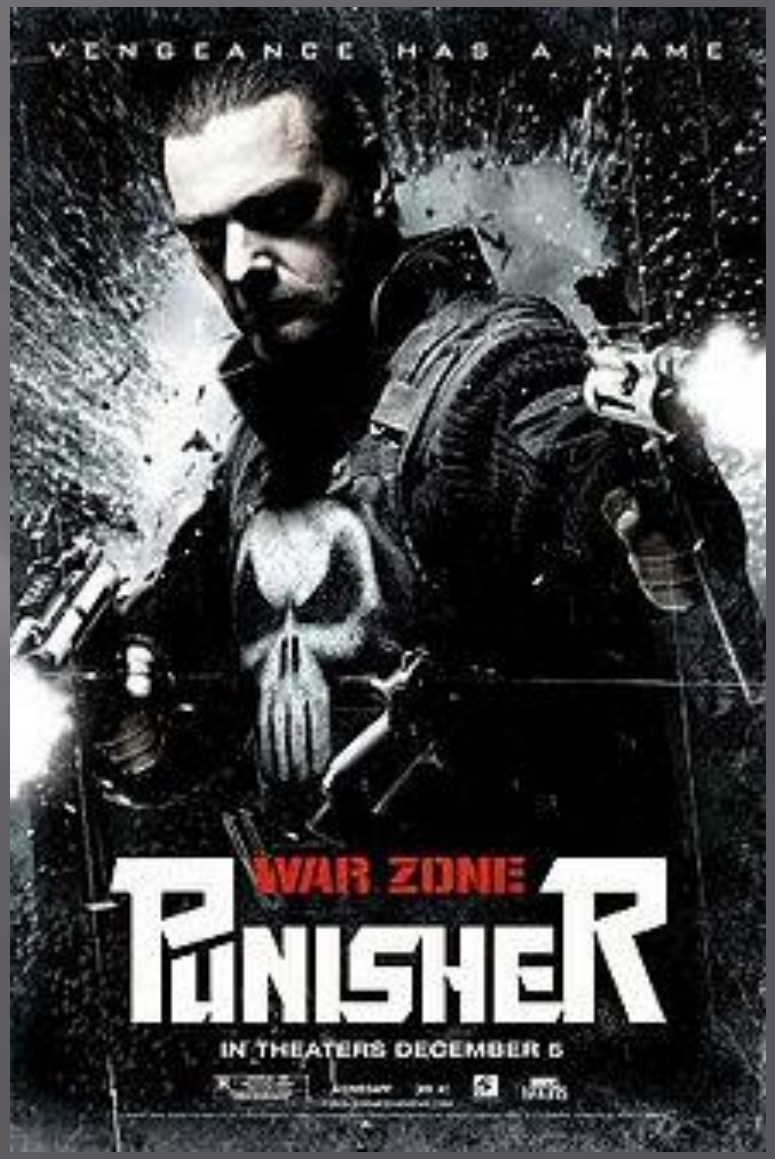
Business Analysis Tools
Part 3

ToC

1. ERD
2. DFD



ERD

ERD

- The ER model defines the conceptual view of a database.
- It works around real-world entities and the associations among them.
- At view level, the ER model is considered a good option for designing databases.

ERD - Entity

- An entity can be a real-world object, either animate or inanimate, that can be easily identifiable.
- For example, in a school database, students, teachers, classes, and courses offered can be considered as entities. All these entities have some attributes or properties that give them their identity.

- An entity set is a collection of similar types of entities.
- An entity set may contain entities with attribute sharing similar values. For example, a Students set may contain all the students of a school; likewise a Teachers set may contain all the teachers of a school from all faculties. Entity sets need not be disjoint.

ERD - Attributes

- Entities are represented by means of their properties, called attributes. All attributes have values.
- For example, a student entity may have name, class, and age as attributes.

- There exists a domain or range of values that can be assigned to attributes. For example, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.

ERD - Types of Attributes

- Simple attribute – Simple attributes are atomic values, which cannot be divided further. For example, a student's *phone number* is an atomic value of 10 digits.
- Composite attribute – Composite attributes are made of more than one simple attribute. For example, a student's complete name may have *first_name* and *last_name*.
- Derived attribute – Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database. For example, *average_salary* in a department should not be saved directly in the database, instead it can be derived. For another example, *age* can be derived from *date_of_birth*.
- Single-value attribute – Single-value attributes contain single value. For example – *Social_Security_Number*.
- Multi-value attribute – Multi-value attributes may contain more than one values. For example, a *person* can have more than one *phone number*, *email_address*, etc.

ERD - Types of Attributes

- These attribute types can come together in a way like –
- simple single-valued attributes
- simple multi-valued attributes
- composite single-valued attributes
- composite multi-valued attributes

ERD - Entity-Set and Keys

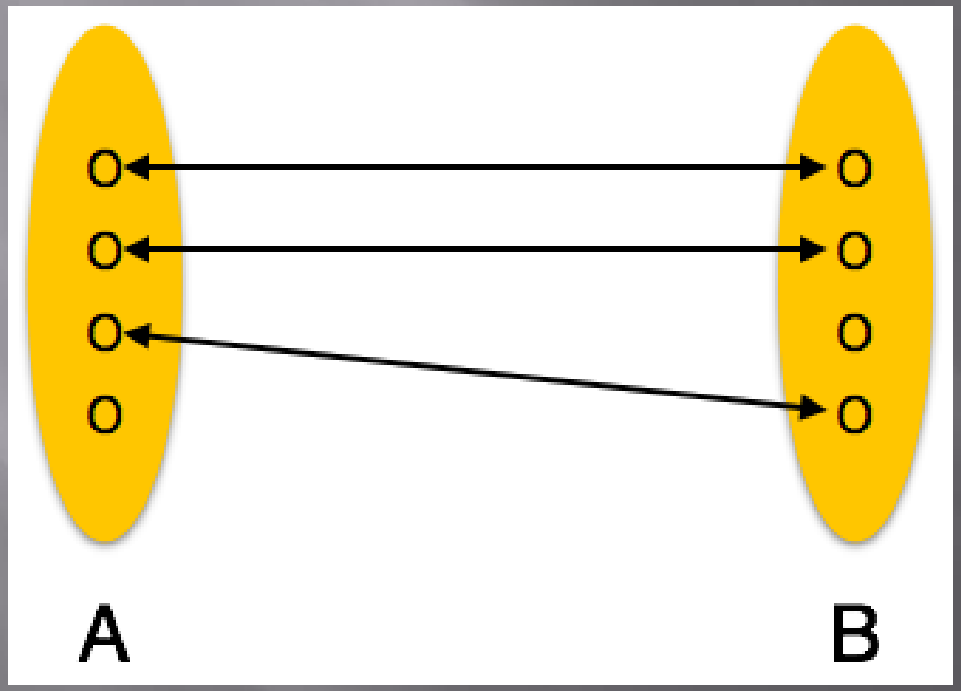
- Key is an attribute or collection of attributes that uniquely identifies an entity among entity set.
- For example, the roll_number of a student makes him/her identifiable among students.
- Super Key – A set of attributes (one or more) that collectively identifies an entity in an entity set.
- Candidate Key – A minimal super key is called a candidate key. An entity set may have more than one candidate key.
- Primary Key – A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

ERD - Relationship

- The association among entities is called a relationship. For example, an employee works_at a department, a student enrolls in a course. Here, Works_at and Enrolls are called relationships.
- Relationship Set
- A set of relationships of similar type is called a relationship set. Like entities, a relationship too can have attributes. These attributes are called descriptive attributes.
- Degree of Relationship
- The number of participating entities in a relationship defines the degree of the relationship.
- Binary = degree 2
- Ternary = degree 3
- n-ary = degree n

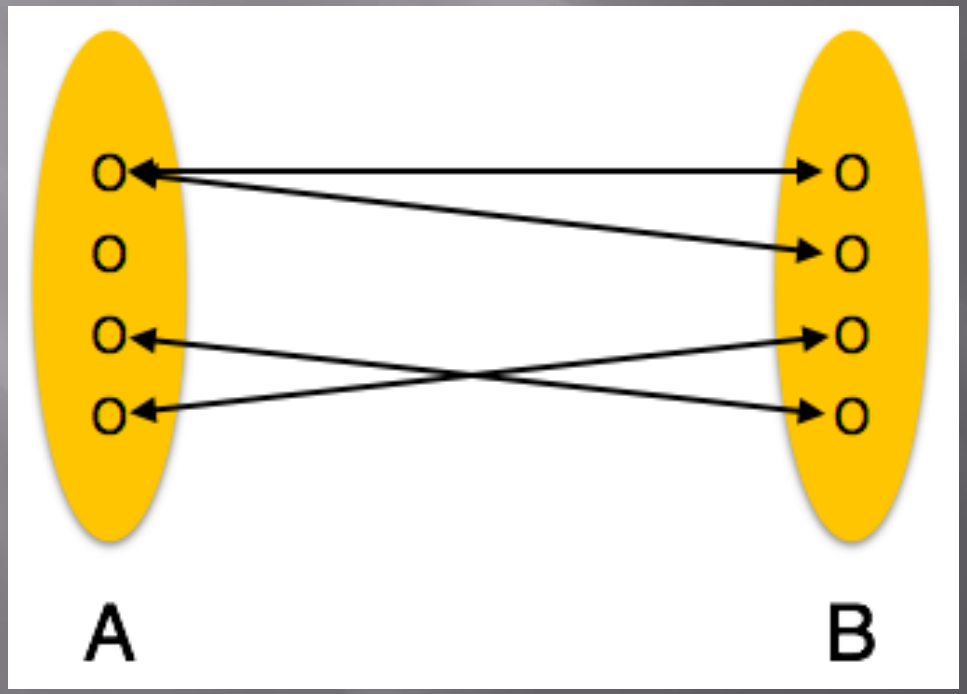
ERD - Relationship

- One-to-one – One entity from entity set A can be associated with at most one entity of entity set B and vice versa.



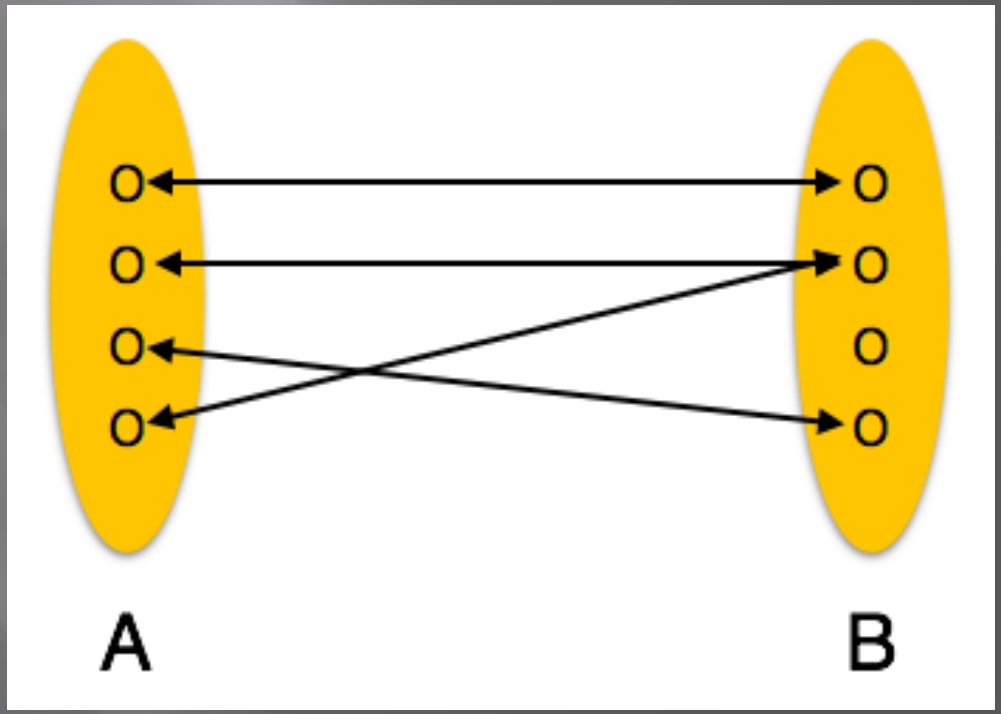
ERD - Relationship

- One-to-many – One entity from entity set A can be associated with more than one entities of entity set B however an entity from entity set B, can be associated with at most one entity.



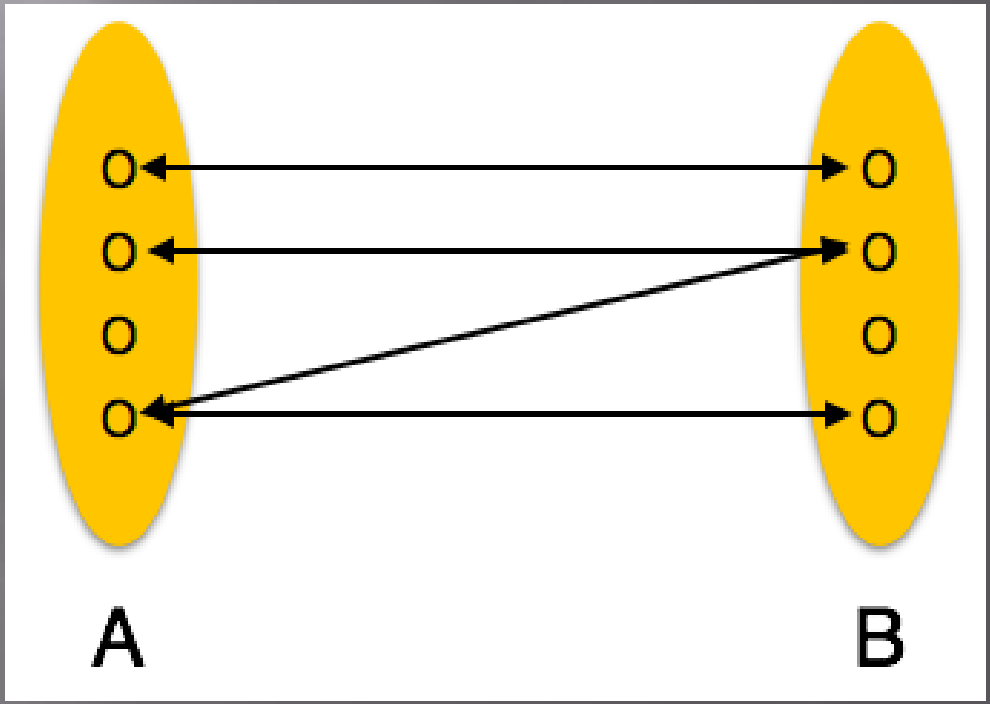
ERD - Relationship

- Many-to-one – More than one entities from entity set A can be associated with at most one entity of entity set B, however an entity from entity set B can be associated with more than one entity from entity set A.



ERD - Relationship

- Many-to-many – One entity from A can be associated with more than one entity from B and vice versa.

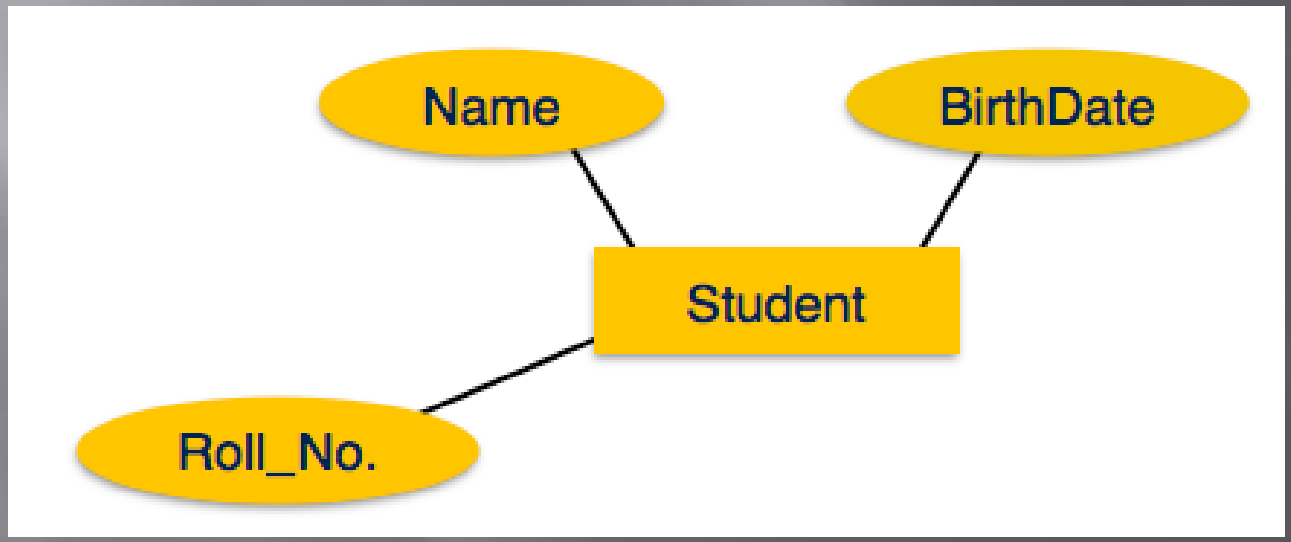


ERD - Entity

- Entity
- Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.

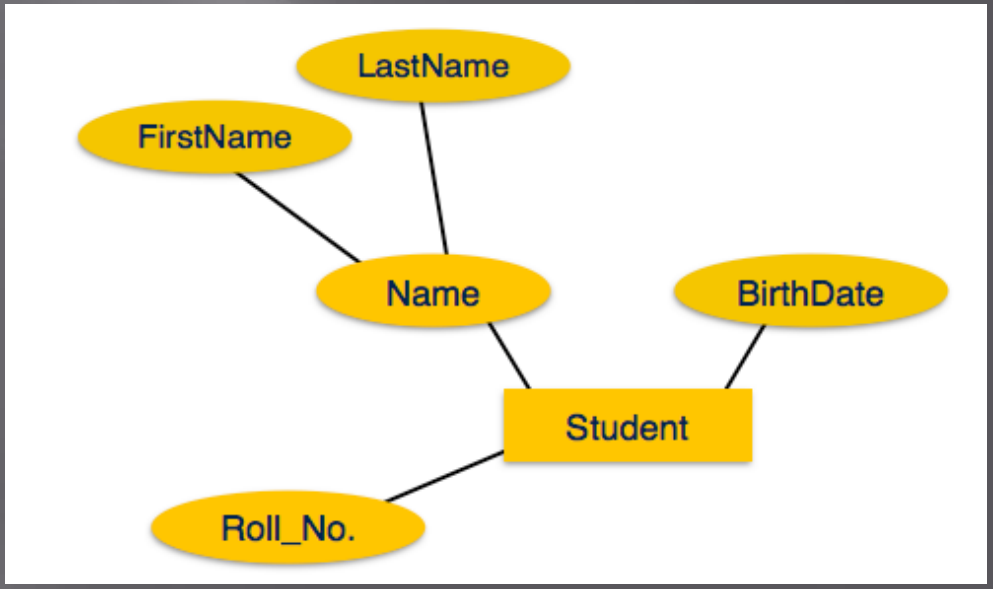
ERD - Attributes

- Attributes
- Attributes are the properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity (rectangle).



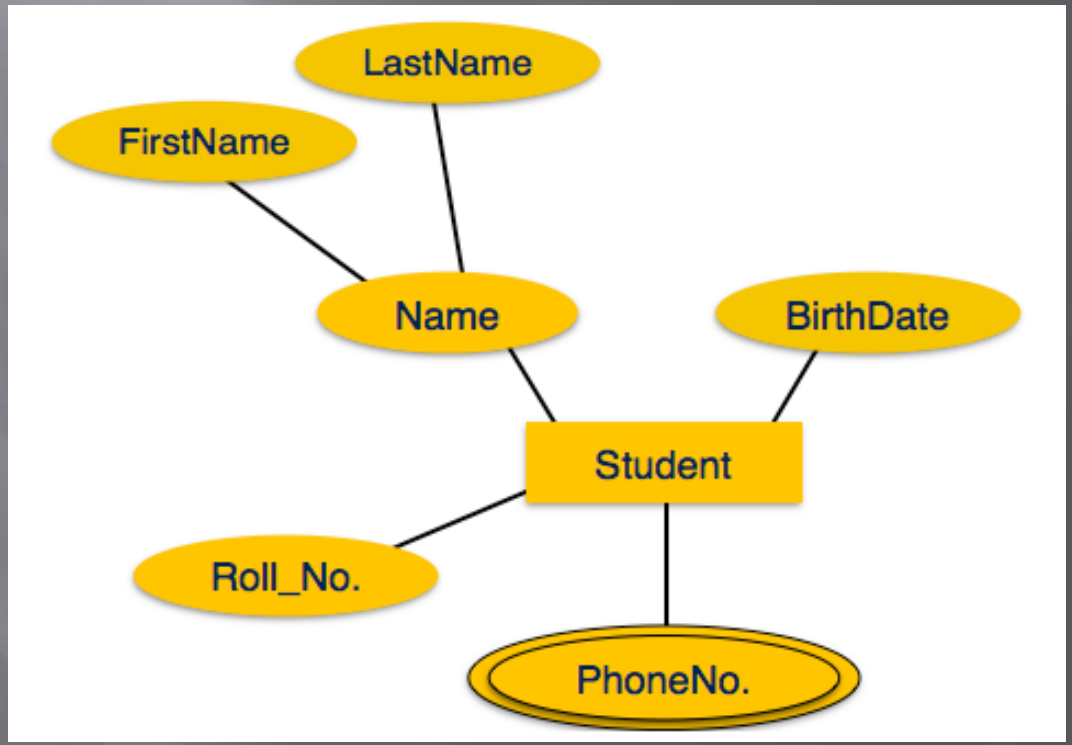
ERD - Attributes

- Attributes
- If the attributes are composite, they are further divided in a tree like structure. Every node is then connected to its attribute. That is, composite attributes are represented by ellipses that are connected with an ellipse.



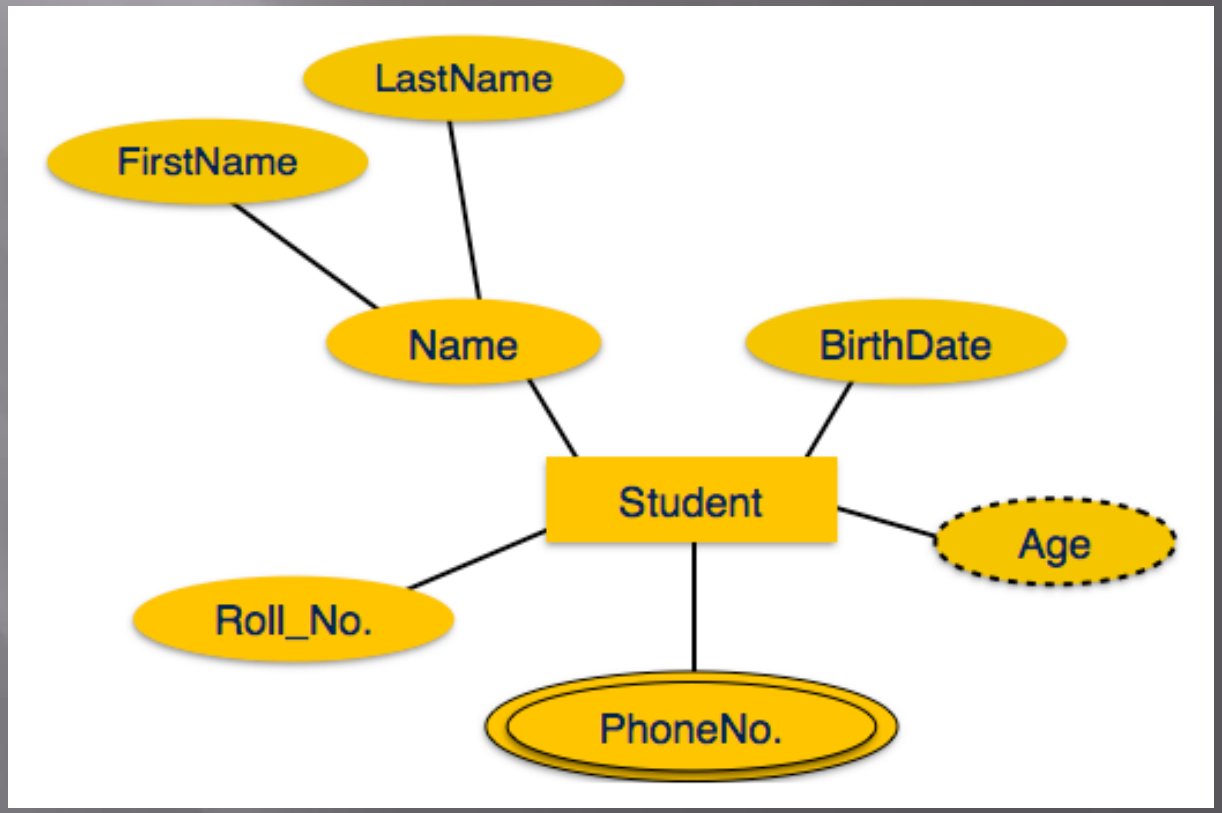
ERD - Attributes

- Attributes
- Multivalued attributes are depicted by double ellipse.



ERD - Attributes

- Attributes
- Derived attributes are depicted by dashed ellipse.



ERD - Relationship

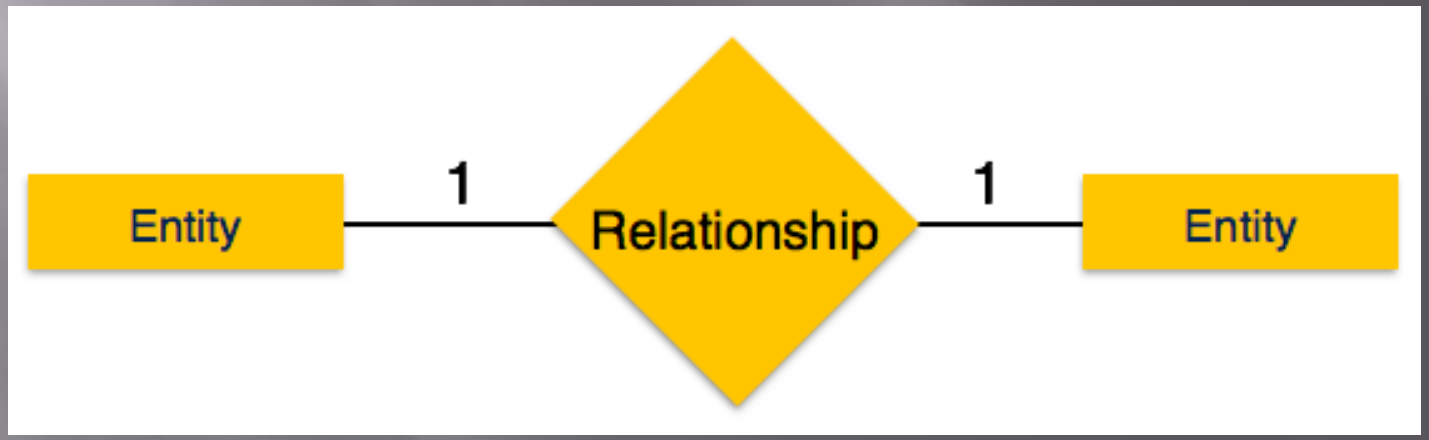
- Relationship
- Relationships are represented by diamond-shaped box. Name of the relationship is written inside the diamond-box. All the entities (rectangles) participating in a relationship, are connected to it by a line.

ERD - Relationship

- Binary Relationship and Cardinality
- A relationship where two entities are participating is called a binary relationship. Cardinality is the number of instance of an entity from a relation that can be associated with the relation.

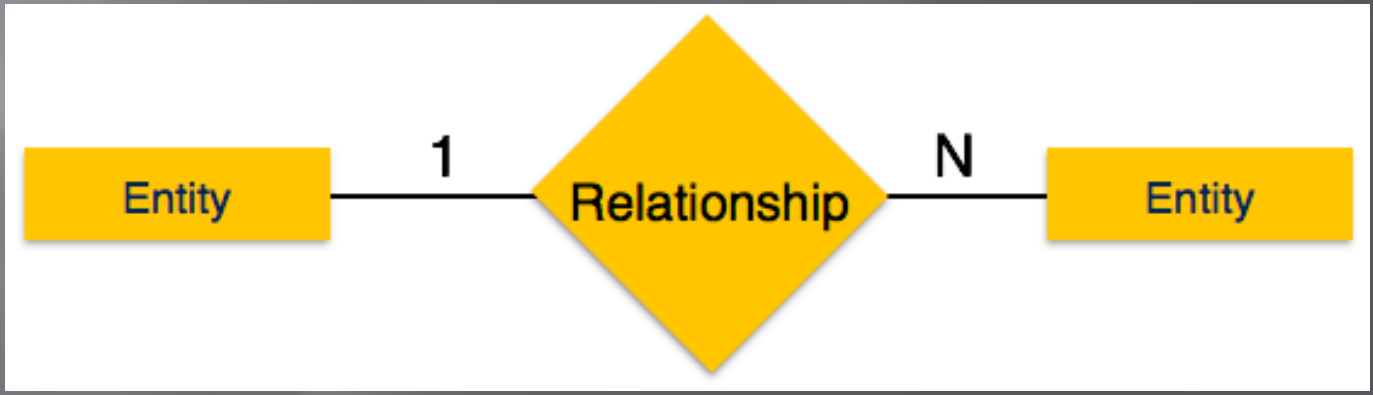
ERD - Relationship

- One-to-one – When only one instance of an entity is associated with the relationship, it is marked as '1:1'. The following image reflects that only one instance of each entity should be associated with the relationship. It depicts one-to-one relationship.



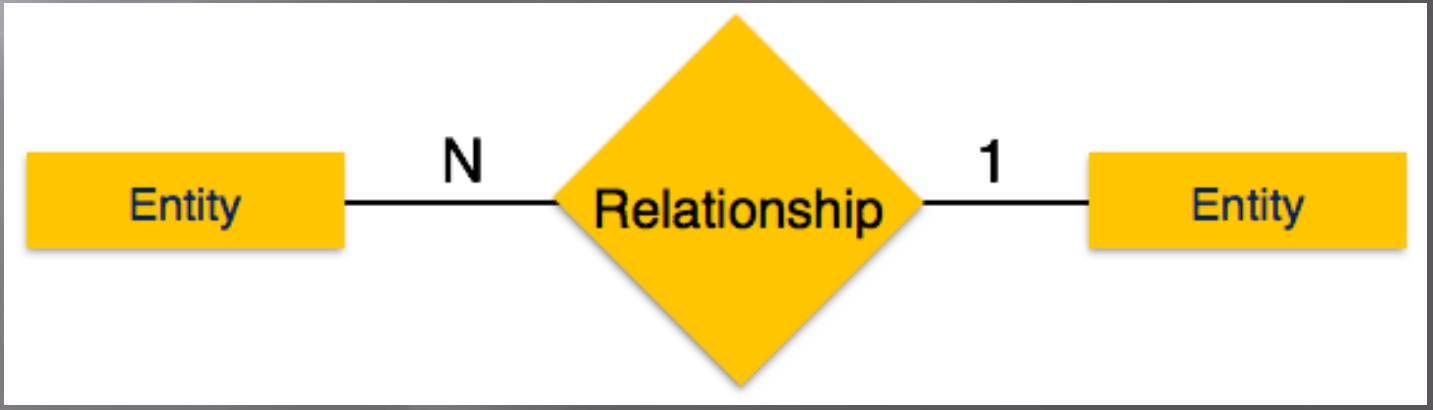
ERD - Relationship

- One-to-many – When more than one instance of an entity is associated with a relationship, it is marked as '1:N'. The following image reflects that only one instance of entity on the left and more than one instance of an entity on the right can be associated with the relationship. It depicts one-to-many relationship.



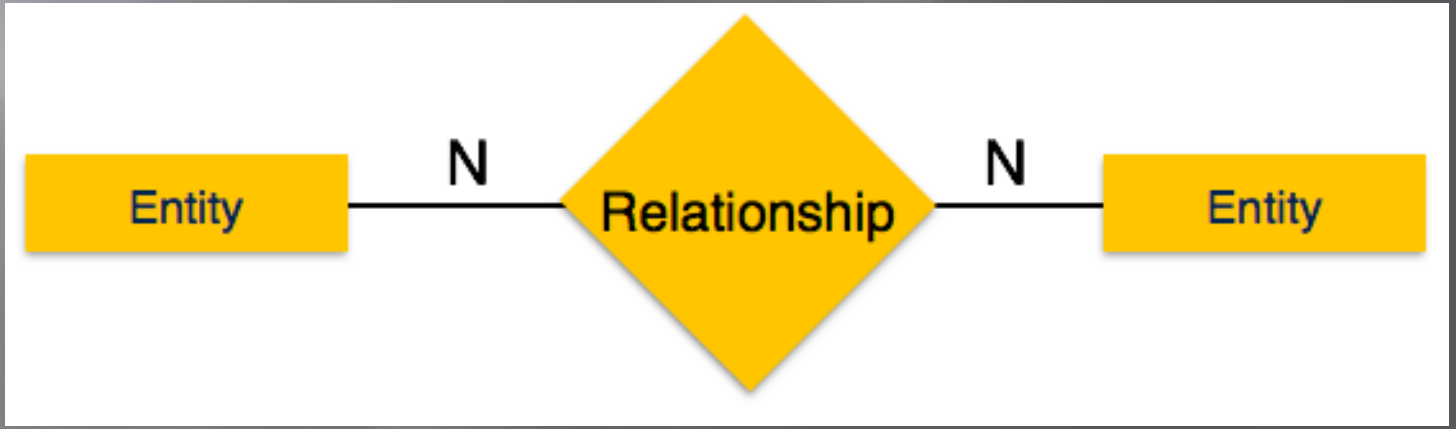
ERD - Relationship

- Many-to-one – When more than one instance of entity is associated with the relationship, it is marked as 'N:1'. The following image reflects that more than one instance of an entity on the left and only one instance of an entity on the right can be associated with the relationship. It depicts many-to-one relationship.



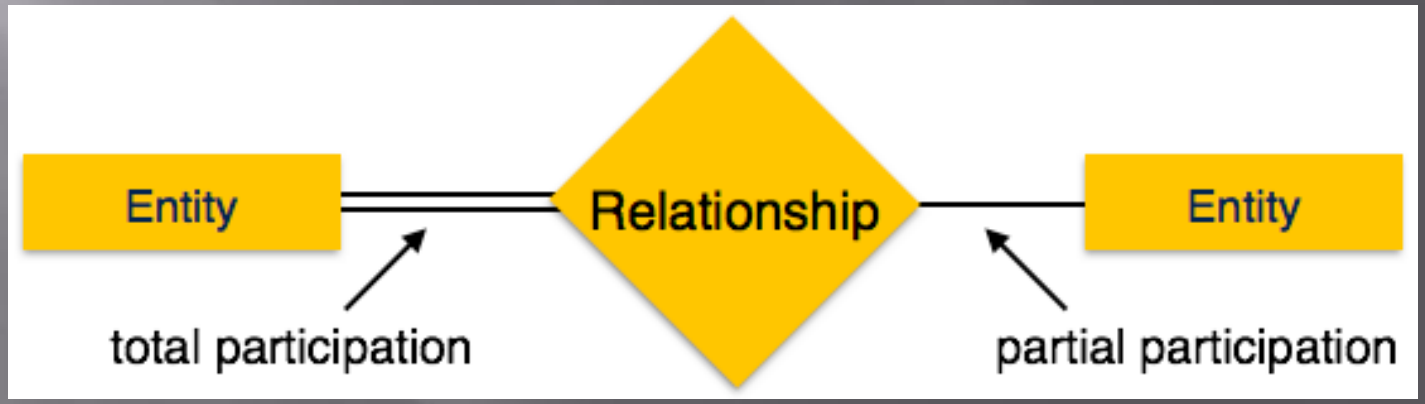
ERD - Relationship

- Many-to-many – The following image reflects that more than one instance of an entity on the left and more than one instance of an entity on the right can be associated with the relationship. It depicts many-to-many relationship.



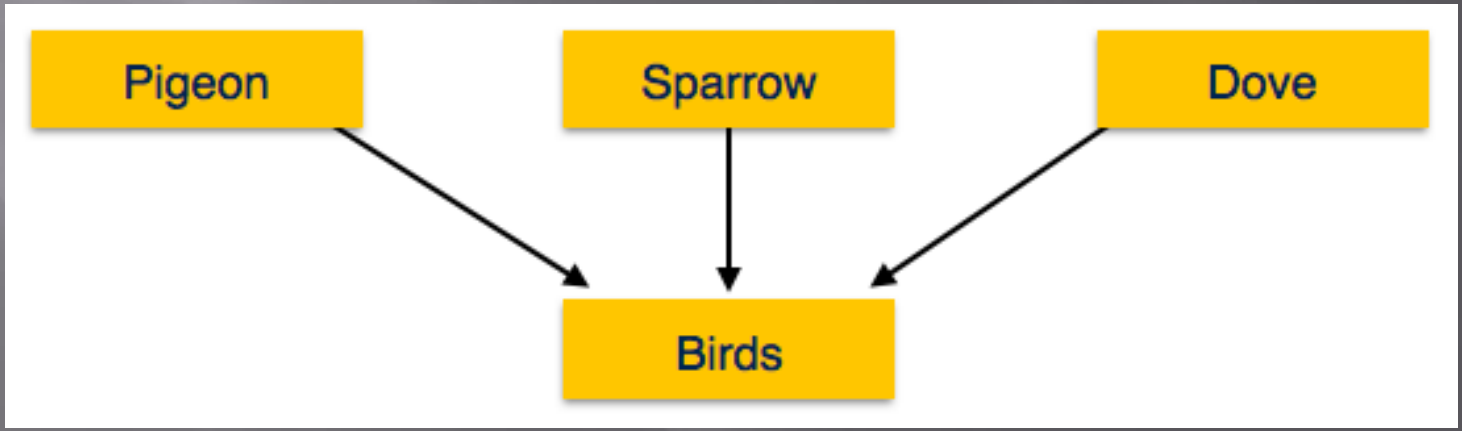
ERD - Relationship

- Participation Constraints
- Total Participation – Each entity is involved in the relationship. Total participation is represented by double lines.
- Partial participation – Not all entities are involved in the relationship. Partial participation is represented by single lines.



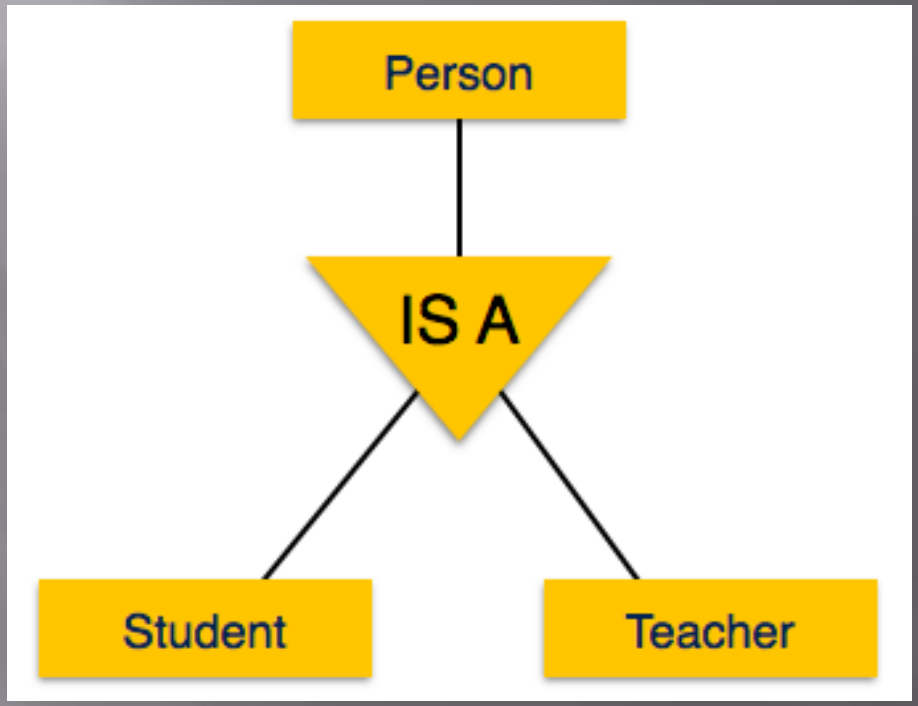
ERD - Relationship

- Generalization
- As mentioned above, the process of generalizing entities, where the generalized entities contain the properties of all the generalized entities, is called generalization. In generalization, a number of entities are brought together into one generalized entity based on their similar characteristics. For example, pigeon, house sparrow, crow and dove can



ERD - Relationship

- Specialization
- Specialization is the opposite of generalization. In specialization, a group of entities is divided into sub-groups based on their characteristics. Take a group ‘Person’ for example. A person has name, date of birth, gender, etc. These properties are common in all persons, human beings. But in a company, persons can be identified as employee, employer, customer, or vendor, based on what role they play in the company.

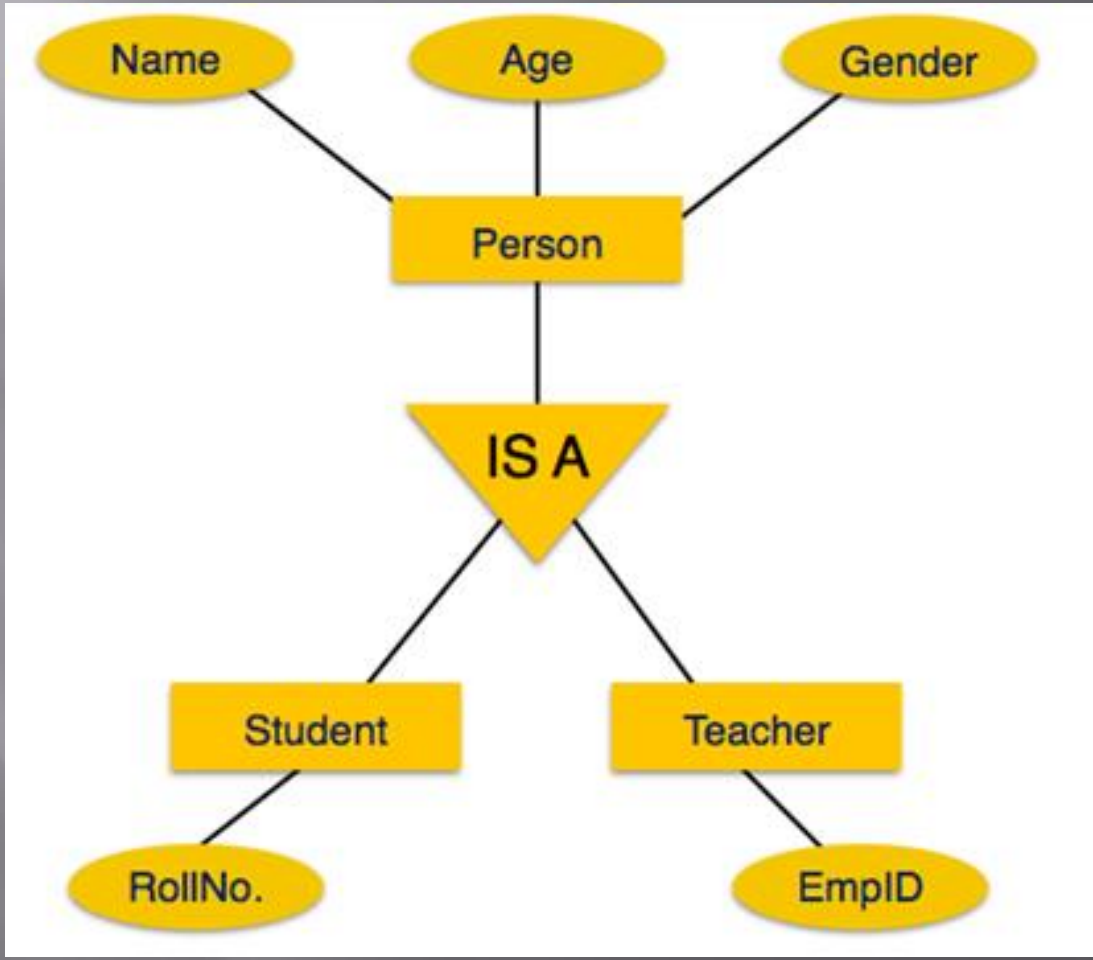

Person
IS A
Student
Teacher

ERD - Relationship

- Inheritance
- We use all the above features of ER-Model in order to create classes of objects in object-oriented programming. The details of entities are generally hidden from the user; this process known as abstraction.
- Inheritance is an important feature of Generalization and Specialization. It allows lower-level entities to inherit the attributes of higher-level entities.
- For example, the attributes of a Person class such as name, age, and gender can be inherited by lower-level entities such as Student or Teacher.

ERD - Relationship

- Inheritance
- For example, the attributes of a Person class such as name, age, and gender can be inherited by lower-level entities such as Student or Teacher.


Name
Age
Gender
Person
IS A
Student
Teacher
RollNo.
EmpID

Definitions:

entity something about which data is collected, stored, and maintained

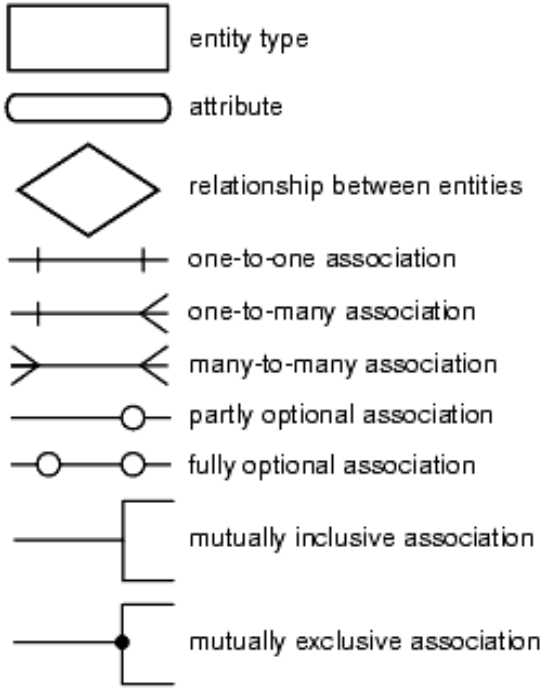
attribute a characteristic of an entity

relationship an association between entities

entity type a class of entities that have the same set of attributes

record an ordered set of attribute values that describe an instance of an entity type

Symbols:



Examples:

One A is associated with one B:

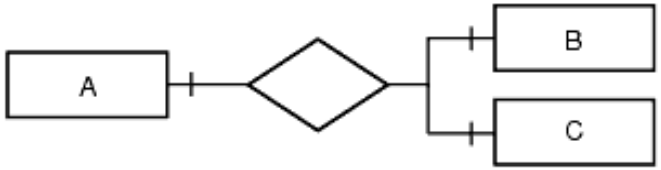
One A is associated with one or more B's:

One or more A's are associated with one or more B's:

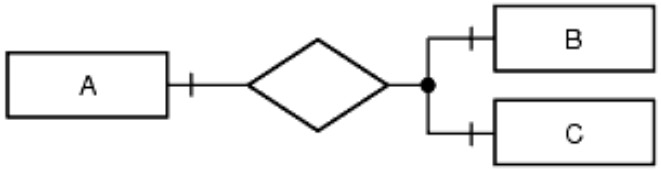
One A is associated with zero or one B:

One A is associated with zero or more B's:

One A is associated with one B and one C:



One A is associated with one B or one C (but not both):



DFD

DFD

- A Data Flow Diagram (DFD) is a diagrammatic representation of the information flows within a system, showing:
- 1. How information enters and leaves the system,
- 2. What changes the information,
- 3. Where information is stored.

DFD - Advantages

1. Simple but powerful graphic technique which is easily understood.
2. Represents an information system from the viewpoint of data movements, which includes the inputs and outputs to which people can readily relate.
3. The ability to represent the system at different levels of details gives added advantage (you can include the advantages of decomposition listed earlier).
4. Helps to define the boundaries of the system.
5. A useful tool to use during interviews.
6. Serve to identify the information services the users require, on the basis of which the future information system will be constructed.

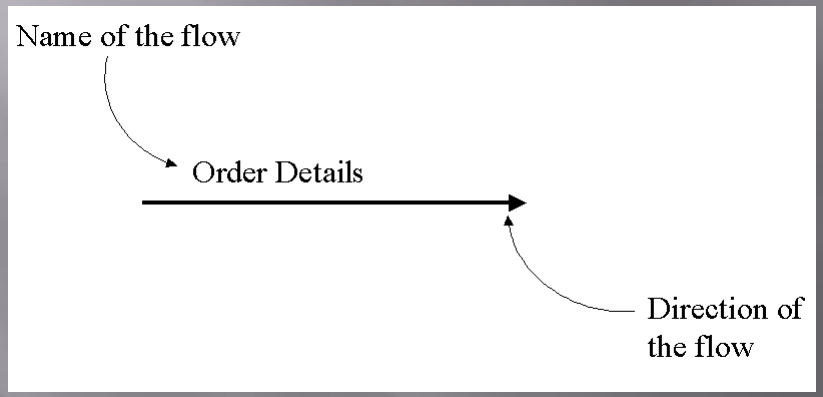
DFD - Notation

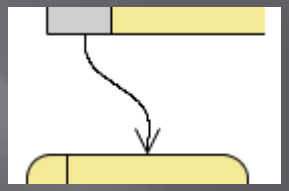
- There are 4 basic constructs:

1. Data flows,
2. Processes,
3. Data Stores,
4. External Entities,

DFD – Data Flow

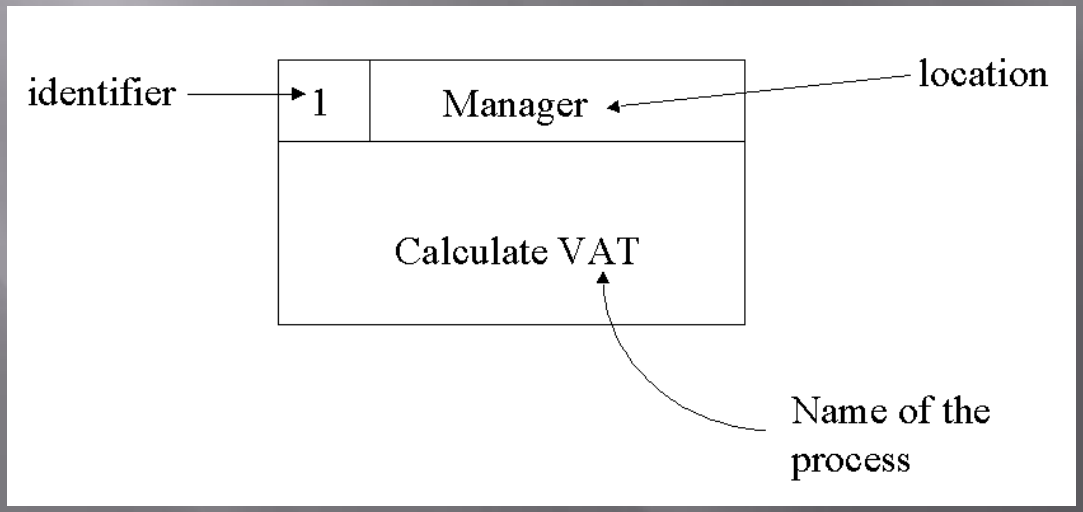
- Data Flow
- A data flow represents the flow of information, with its direction represented by an arrow head that shows at the end(s) of flow connector.

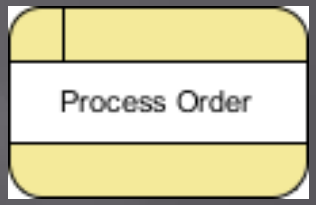




DFD – Processes

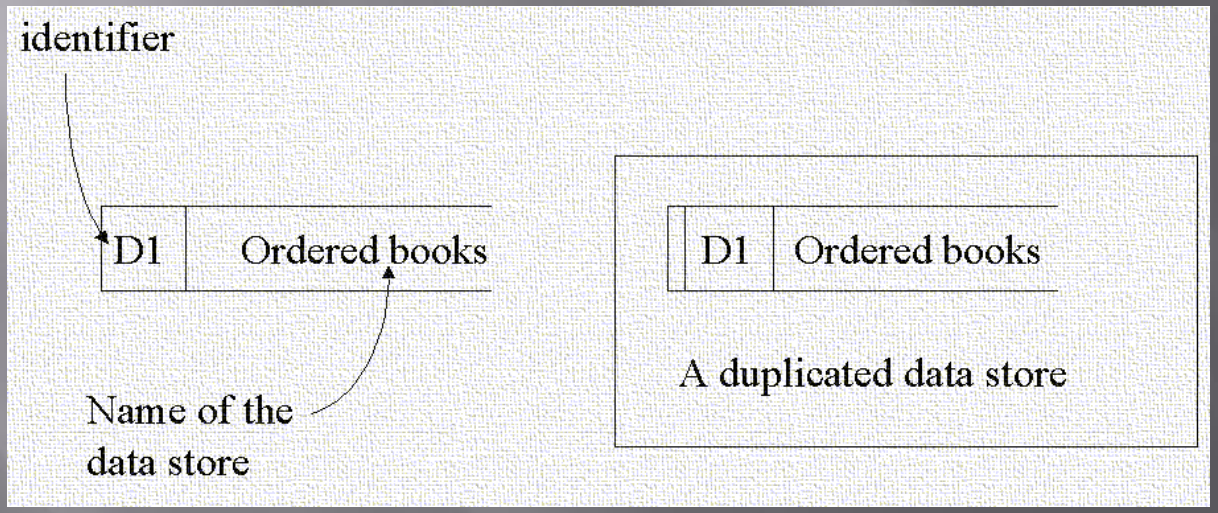
- Processes
- A process is a business activity or function where the manipulation and transformation of data takes place. A process can be decomposed to finer level of details, for representing how data is being processed within the process.





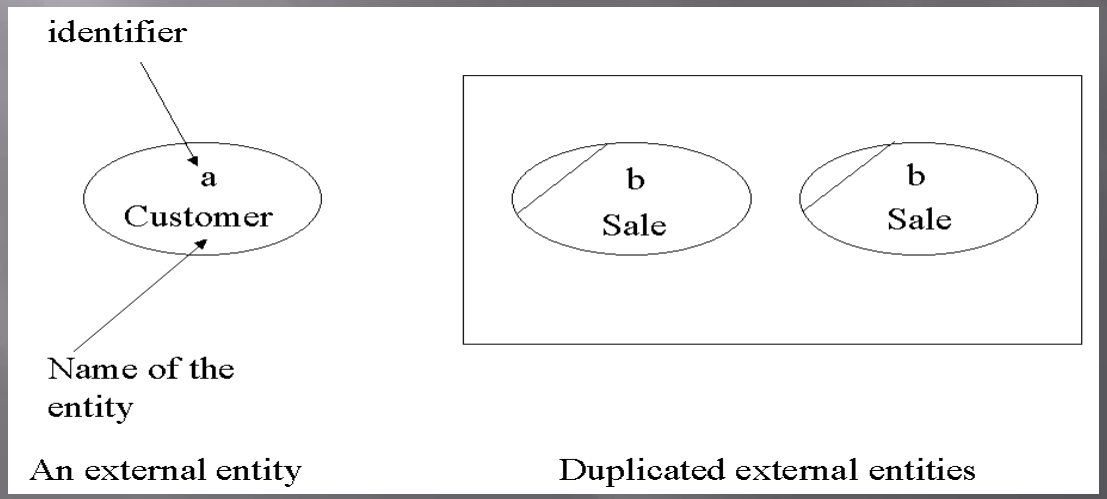
DFD – Data Stores

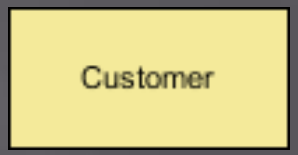
- Data Stores
- A data store represents the storage of persistent data required and/or produced by the process. Here are some examples of data stores: membership forms, database table, etc.



DFD – External Entity

- External Entity
- An external entity can represent a human, system or subsystem. It is where certain data comes from or goes to. It is external to the system we study, in terms of the business process. For this reason, people used to draw external entities on the edge of a diagram.

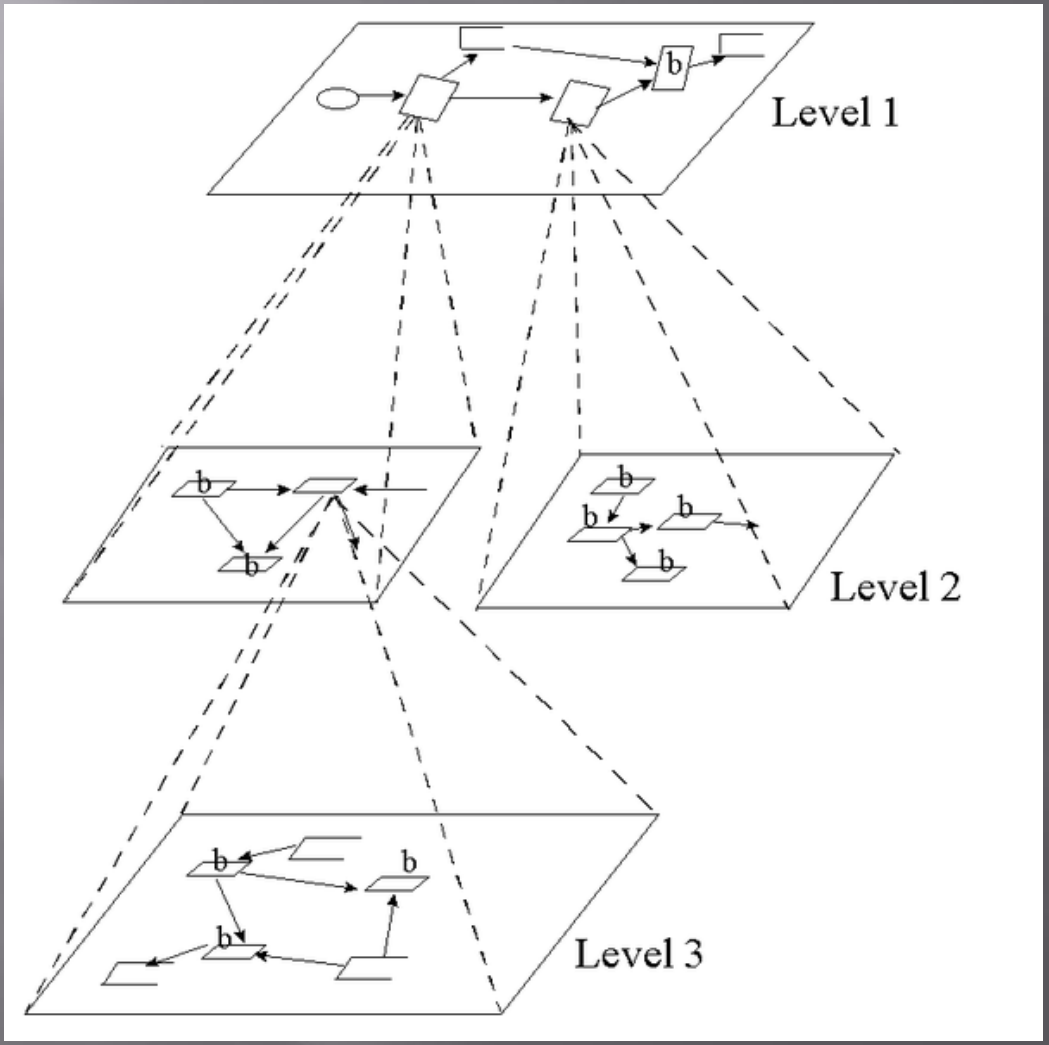


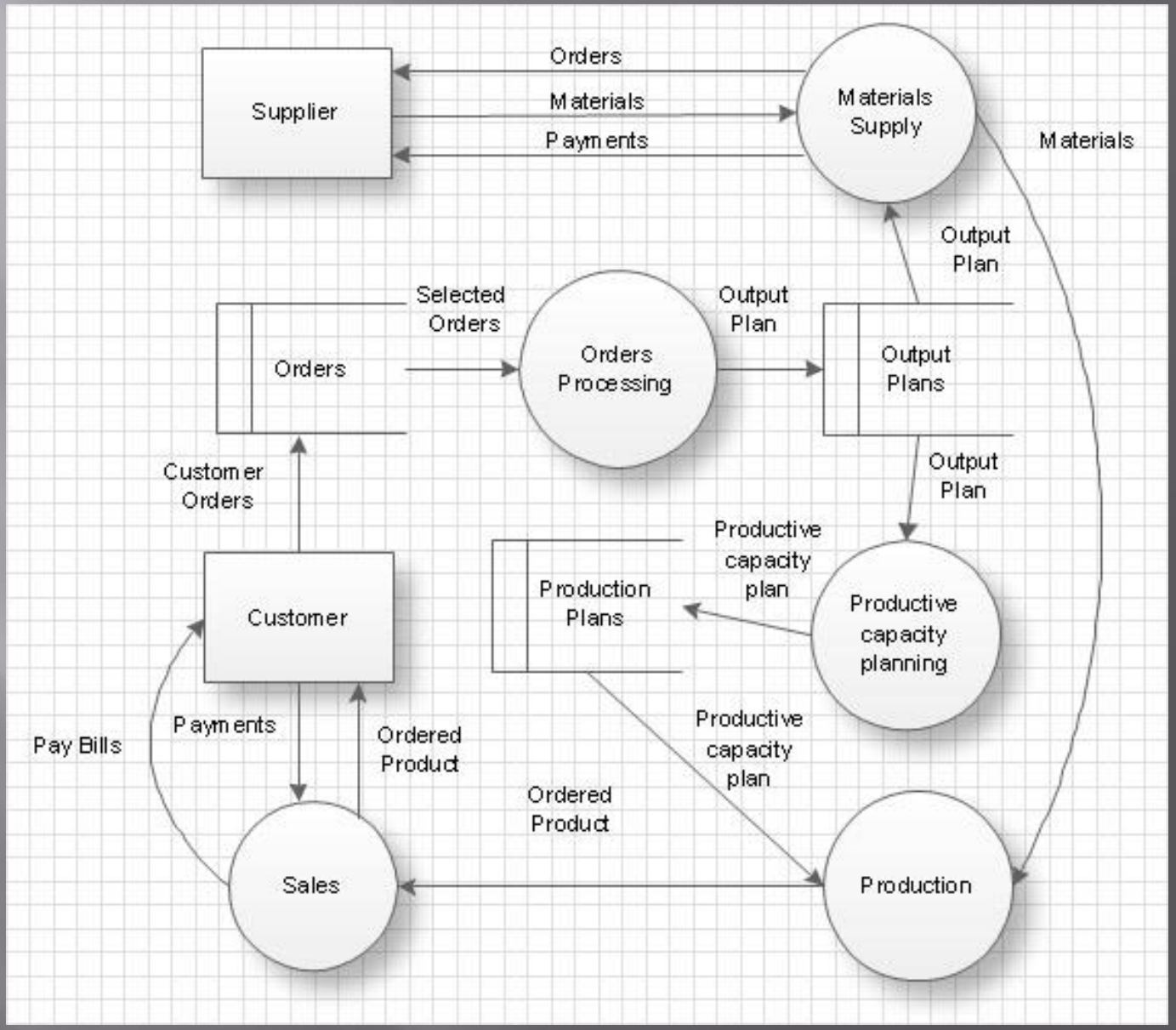


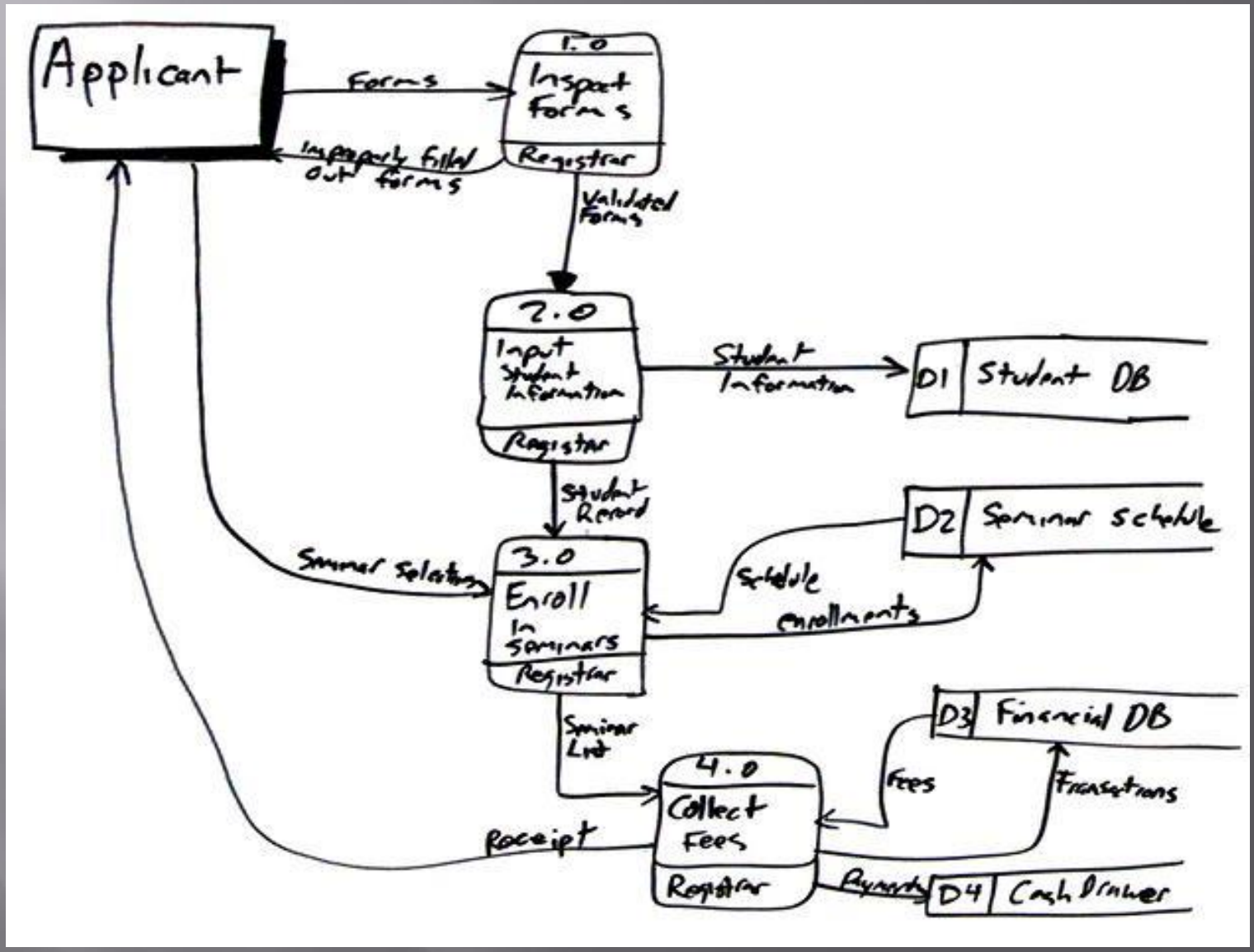
DFD – Hints on names on DFDs

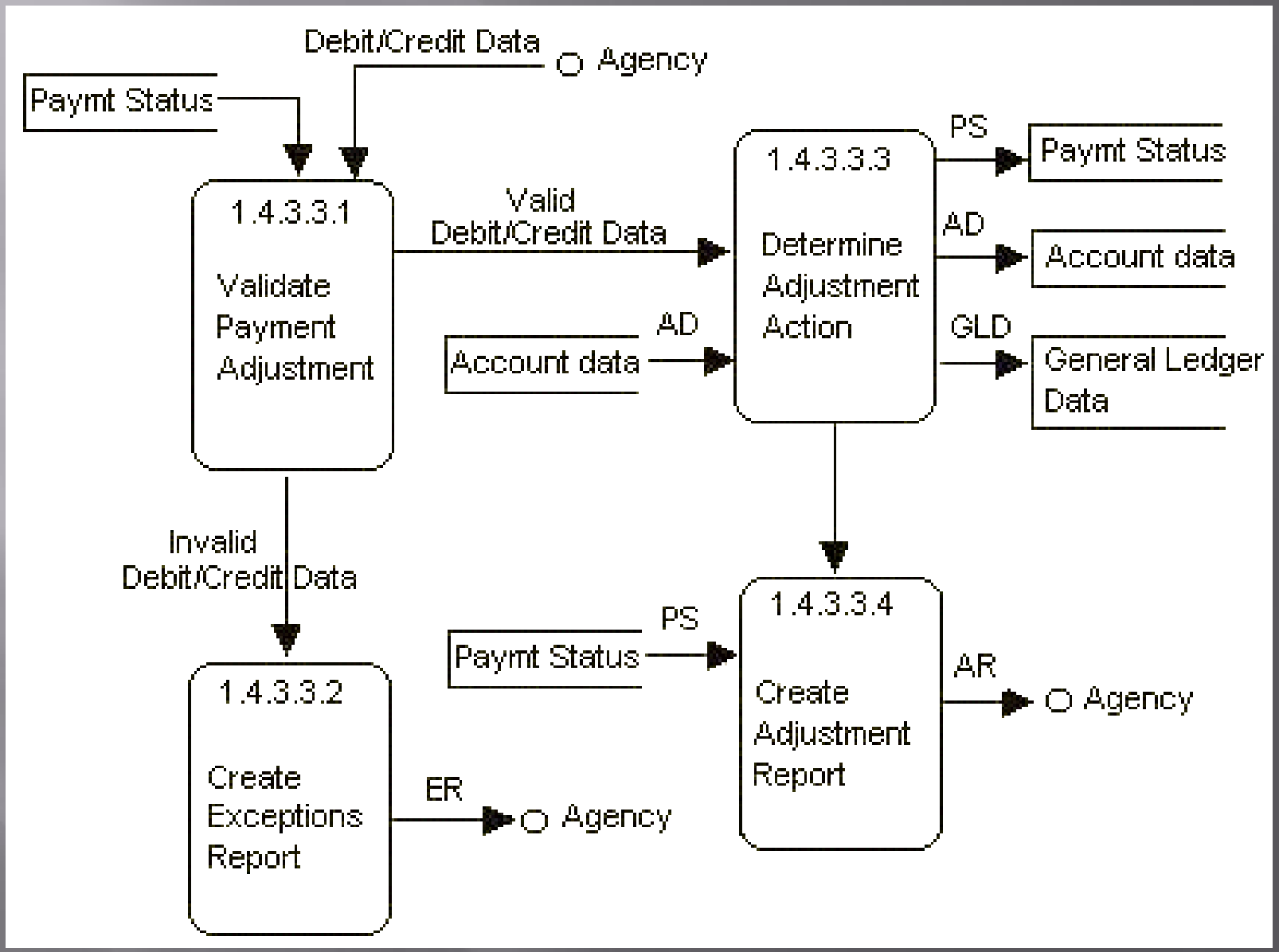
- *Data Flows*
- Name the data + adjective(s)
- Say what is known about it (e.g. valid account number)
-
- *Processes*
- The name is a command: verb + nouns (e.g. validate account number) or
- verb + object/object phrase
- Do not use ‘and’, ‘or’, ‘then’, ‘if’, ‘repeat’, or any other words that specify control flow.
-
- *Data Stores*
- Show only net flow in/out/both (i.e. indicate whether it is read-only, write-only, or updated).

DFD - Modelling Hierarchy

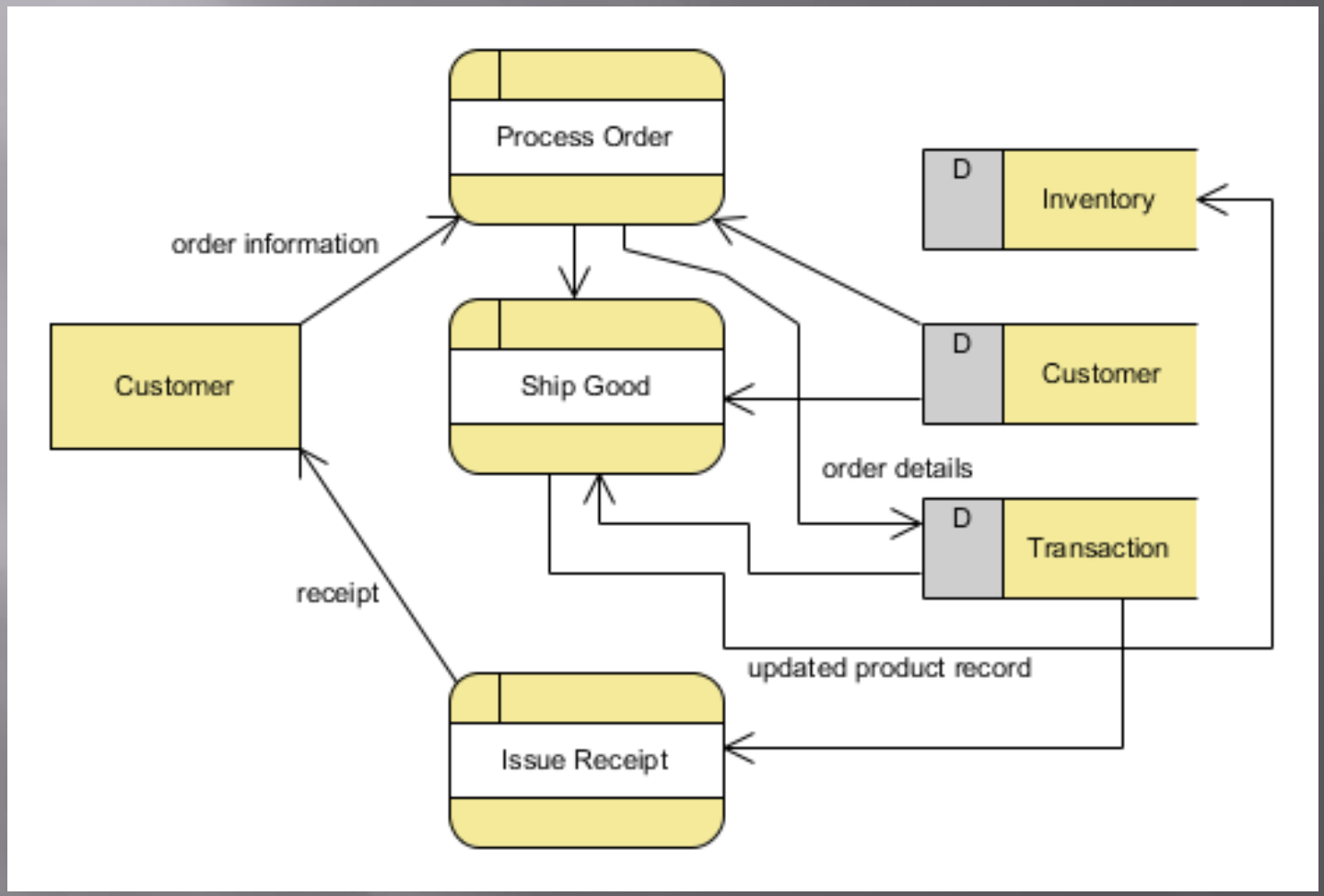



Supplier
Orders
Materials
Payments
Materials Supply
Materials
Output Plan
Selected Orders
Orders
Orders Processing
Output Plan
Output Plans
Output Plan
Customer Orders
Customer
Production Plans
Productive capacity plan
Productive capacity planning
Pay Bills
Payments
Ordered Product
Productive capacity plan
Ordered Product
Sales
Production

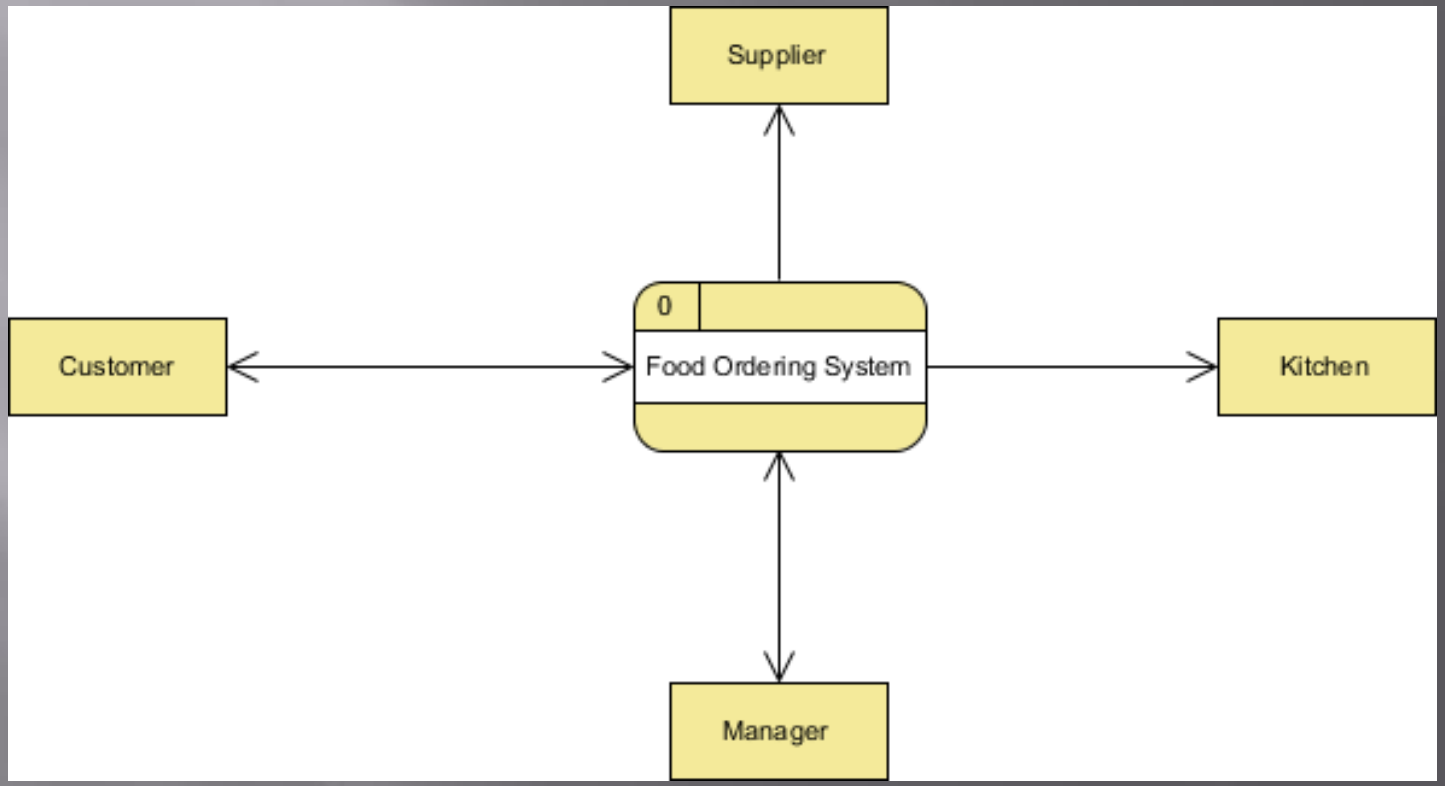

Applicant
Forms
1.0
Input forms
Registrar
Improperly filled out forms
Validated forms
2.0
Input student information
Registrar
Student Information
D1
Student DB
Student Record
D2
Seminar schedule
Seminar Selection
3.0
Enroll in Seminars
Registrar
Schedule
enrollments
Seminar List
D3
Financial DB
4.0
Collect Fees
Registrar
Fees
Transactions
Receipt
Payment
D4
Cash Drawer


Debit/Credit Data
Agency
Paymt Status
1.4.3.3.1
Validate Payment Adjustment
Valid Debit/Credit Data
1.4.3.3.3
Determine Adjustment Action
PS
Paymt Status
AD
Account data
GLD
General Ledger Data
AD
Account data
Invalid Debit/Credit Data
1.4.3.3.2
Create Exceptions Report
ER
Agency
PS
Paymt Status
1.4.3.3.4
Create Adjustment Report
AR
Agency

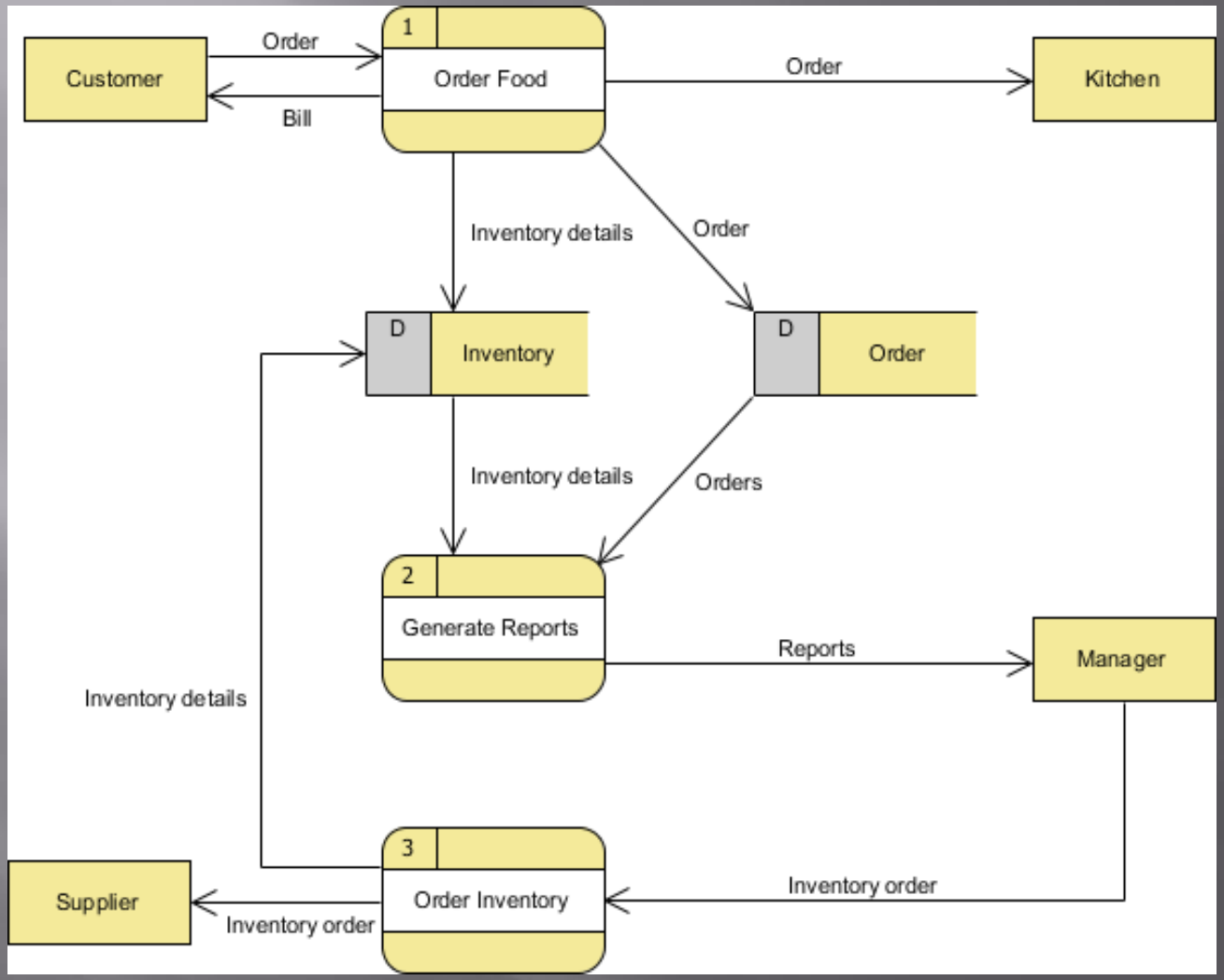
Example: Order Processing



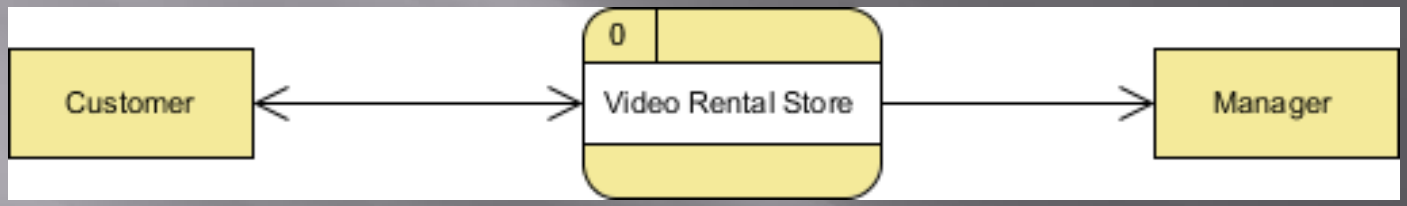
Example: Food Order Processing – Level 0



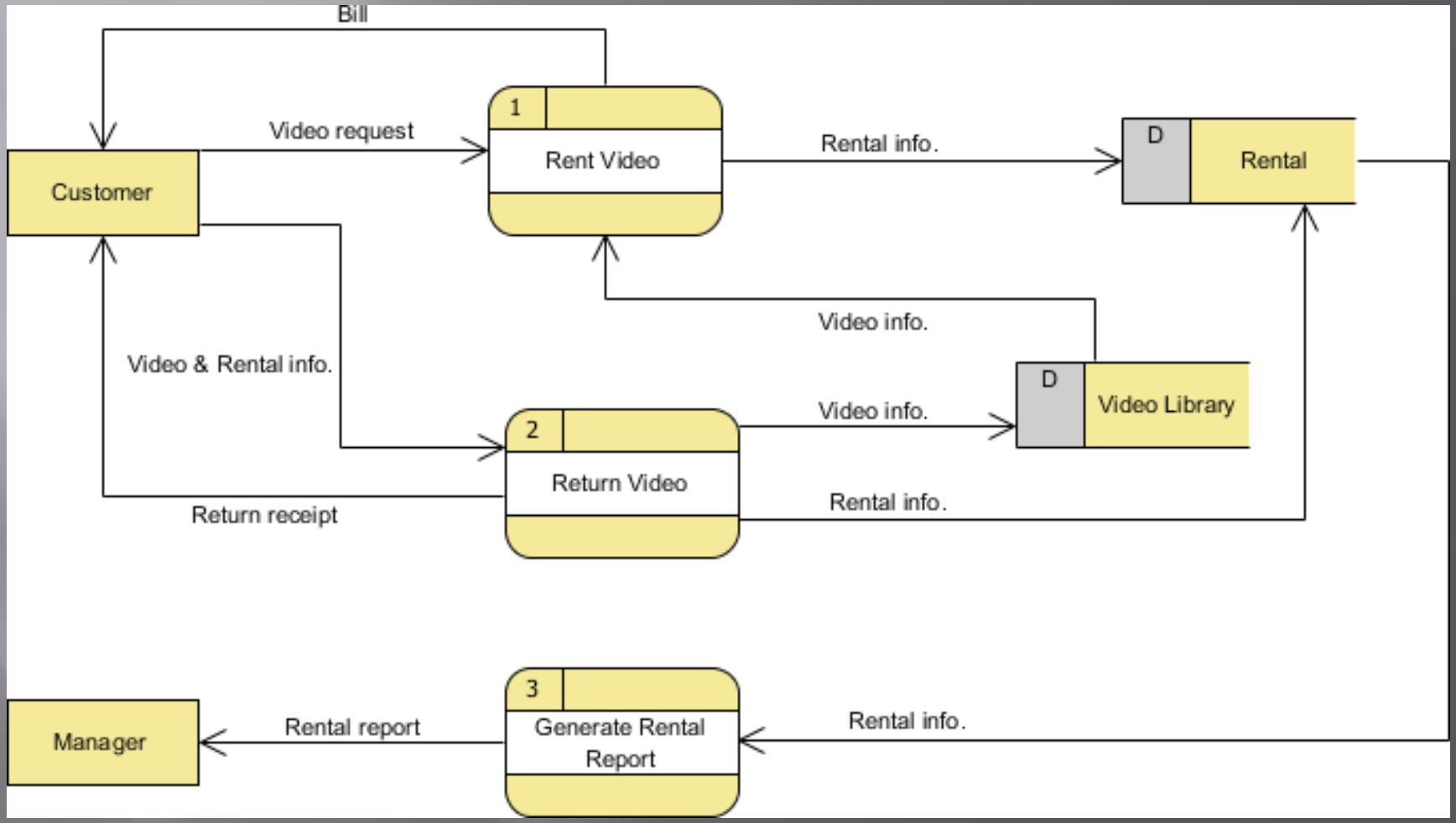
Example: Food Order Processing – Level 1



Example: Video Rental – Level 0



Example: Video Rental – Level 1



See you in Part 4…