

**Names:****Vishal Salvi** (2019230069)**Shivam Pawar** (2019230068)**Shreyas Patel** (2018130037)**Batch: C****Class: TE comps****Experiment No 2****Aim:** To create Data Flow Diagrams (DFD) for select problem statement**Theory:**

The Data Flow Diagram (DFD) is a structured analysis and design method. It is traditional visual representation of the information flows within a system. Data Flow Diagram (DFD) is widely used for software analysis and design. A neat and clear DFD can depict a good amount of the system requirements graphically.

The Data Flow Diagram (DFD) depicts the logic models and expresses data transformation in a system. It includes a mechanism to model the data flow and supports decomposition to illustrate details of the data flows and functions. A Data Flow Diagram cannot present information on operation sequence. Therefore, it is not a process or procedure modeling method.

**DFD includes following characteristics:**

- supporting the analysis and requirement stage of system design;
- a diagramming technique with annotation;
- describing a network of activities/processes of the target system;
- allowing for behaviors of parallel and asynchronous;
- stepwise refinement through hierarchical decomposition of processes.

**Why DFD?**

DFD graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system. The visual representation makes it a good communication tool between User and System designer. Structure of DFD allows starting from a broad overview and expand it to a hierarchy of detailed diagrams.

DFD has often been used due to the following reasons:

- Logical information flow of the system
- Determination of physical system construction requirements
- Simplicity of notation
- Establishment of manual and automated systems requirements

**DFD Symbols**

There are four basic symbols that are used to represent a data-flow diagram.

**1. Process**

A process receives input data and produces output with a different content or form. Processes can be as simple as collecting input data and saving in the database, or it can be complex as producing a report containing monthly sales of all retail stores in the northwest region.

Every process has a name that identifies the function it performs.

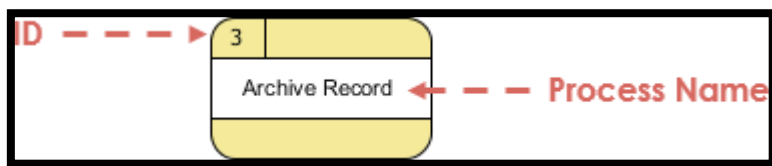
The name consists of a verb, followed by a singular noun.

**Example:**

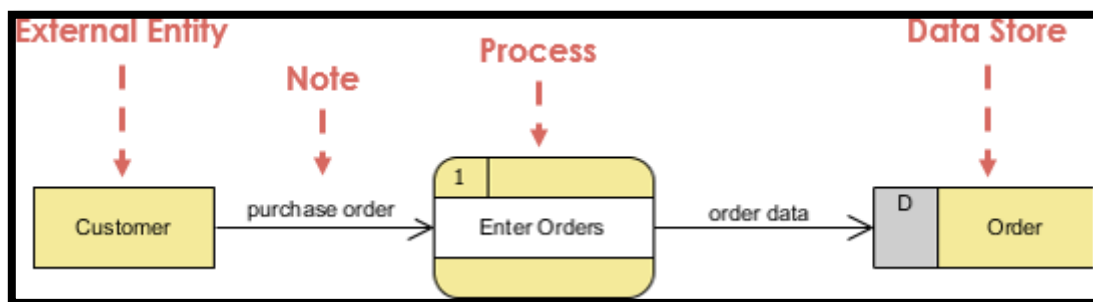
- Apply Payment
- Calculate Commission
- Verify Order

**Notation**

- A rounded rectangle represents a process
- Processes are given IDs for easy referencing



**Process Example**



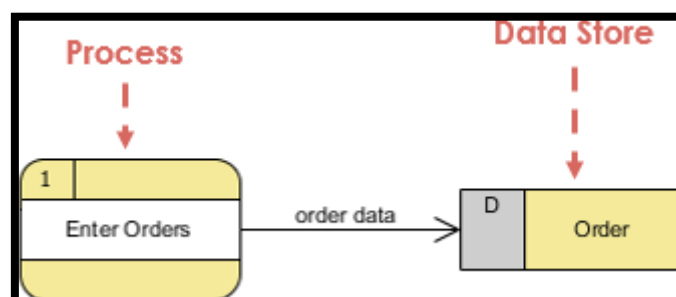
**2. Data Flow**

A data-flow is a path for data to move from one part of the information system to another. A data-flow may represent a single data element such the Customer ID or it can represent a set of data element (or a data structure).

**Example:**

- Customer\_info (LastName, FirstName, SS#, Tel #, etc.)
- Order\_info (OrderId, Item#, OrderDate, CustomerID, etc.).

**Data flow Example:**



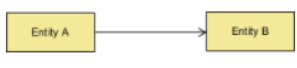






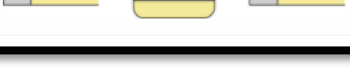
## Notation

- Straight lines with incoming arrows are input data flow
- Straight lines with outgoing arrows are output data flows

## Rule of Data Flow

One of the rule for developing DFD is that all flow must begin with and end at a processing step.

This is quite logical, because data can't transform on its own with being process. By using the thumb rule, it is quite easily to identify the illegal data flows and correct them in a DFD.

Wrong	Right	Description
		An entity cannot provide data to another entity without some processing occurred.
		Data cannot move directly from an entity to a data store without being processed.
		Data cannot move directly from a data store to an entity without being processed.
		Data cannot move directly from one data store to another without being processed.

## 3. Data Store

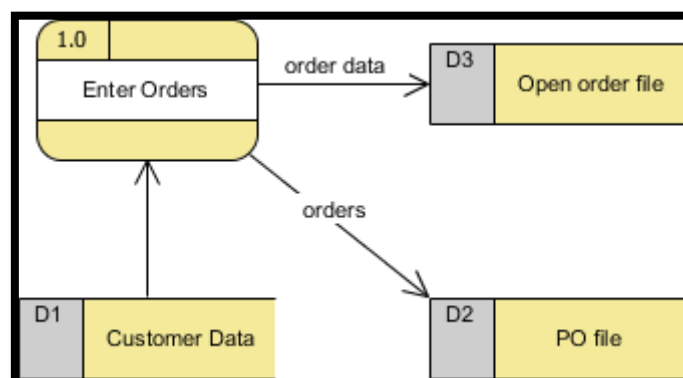
A data store or data repository is used in a data-flow diagram to represent a situation when the system must retain data because one or more processes need to use the stored data in a later time.

## Notation

- Data can be written into the data store, which is depicted by an outgoing arrow
- Data can be read from a data store, which is depicted by an incoming arrow.
- Examples are: inventory, Accounts receivables, Orders, and Daily Payments.



## Data Store Example



**Note that:**

- A data store must be connected to a process with a data-flow.
- Each data store must have at least one input data-flow and at least one output data-flow (even if the output data-flow is a control or confirmation message).

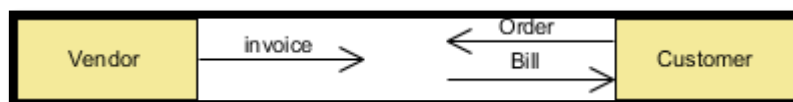
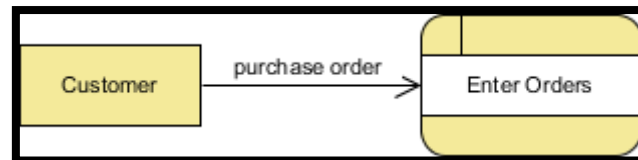
**4. External Entity**

An external entity is a person, department, outside organization, or other information system that provides data to the system or receives outputs from the system. External entities are components outside of the boundaries of the information systems. They represent how the information system interacts with the outside world.

- A rectangle represents an external entity
- They either supply data or receive data
- They do not process data

**Notation**

- A customer submitting an order and then receive a bill from the system
- A vendor issues an invoice

**External Entity Example****Note that:**

- External entities also are called terminators because they are data origins or final destinations.
- An external entity must be connected to a process through a data-flow.

**Levels in Data Flow Diagrams (DFD)**

The DFD may be used to perform a system or software at any level of abstraction. Infact, DFDs may be partitioned into levels that represent increasing information flow and functional detail. Levels in DFD are numbered 0, 1, 2 or beyond. Here, we will see primarily three levels in the data flow diagram, which are: 0-level DFD, 1-level DFD, and 2-level DFD.

**0-level DFDM**

It is also known as fundamental system model, or context diagram represents the entire software requirement as a single bubble with input and output data denoted by incoming and outgoing arrows. Then the system is decomposed and described as a DFD with multiple bubbles. Parts of the system represented by each of these bubbles are then decomposed and documented as more and more detailed DFDs. This process may be repeated at as many levels as necessary until the program at hand is well understood. It is essential to preserve the number of inputs and outputs between levels.

## 1-level DFD

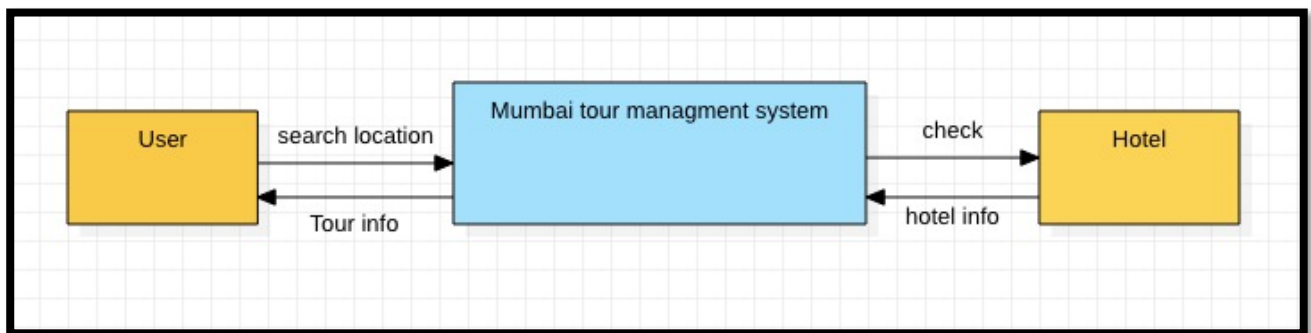
In 1-level DFD, a context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main objectives of the system and breakdown the high-level process of 0-level DFD into subprocesses.

## 2-Level DFD

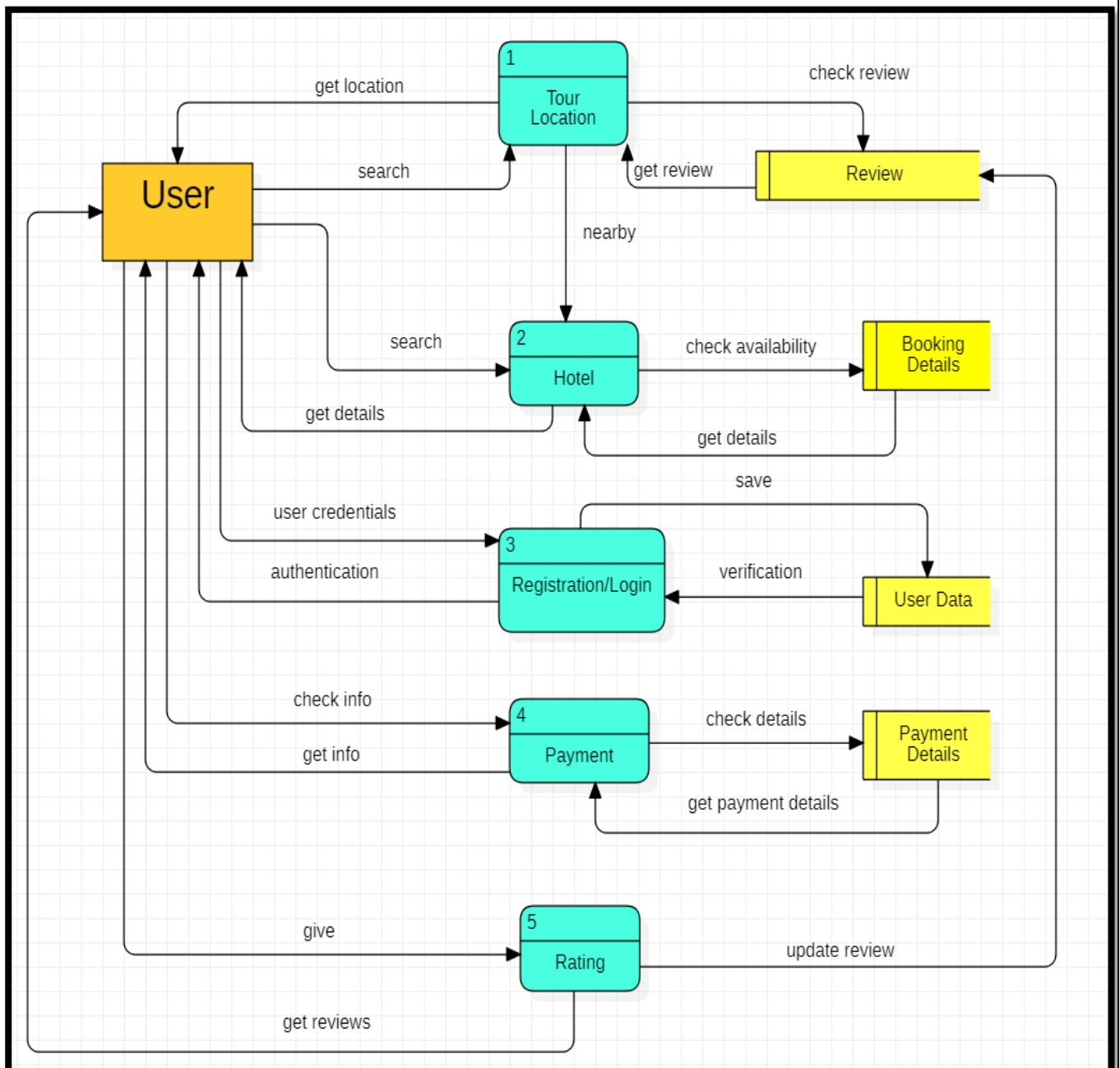
2-level DFD goes one process deeper into parts of 1-level DFD. It can be used to project or record the specific/necessary detail about the system's functioning.

## Data Flow Diagram:

### 0-level Data Flow

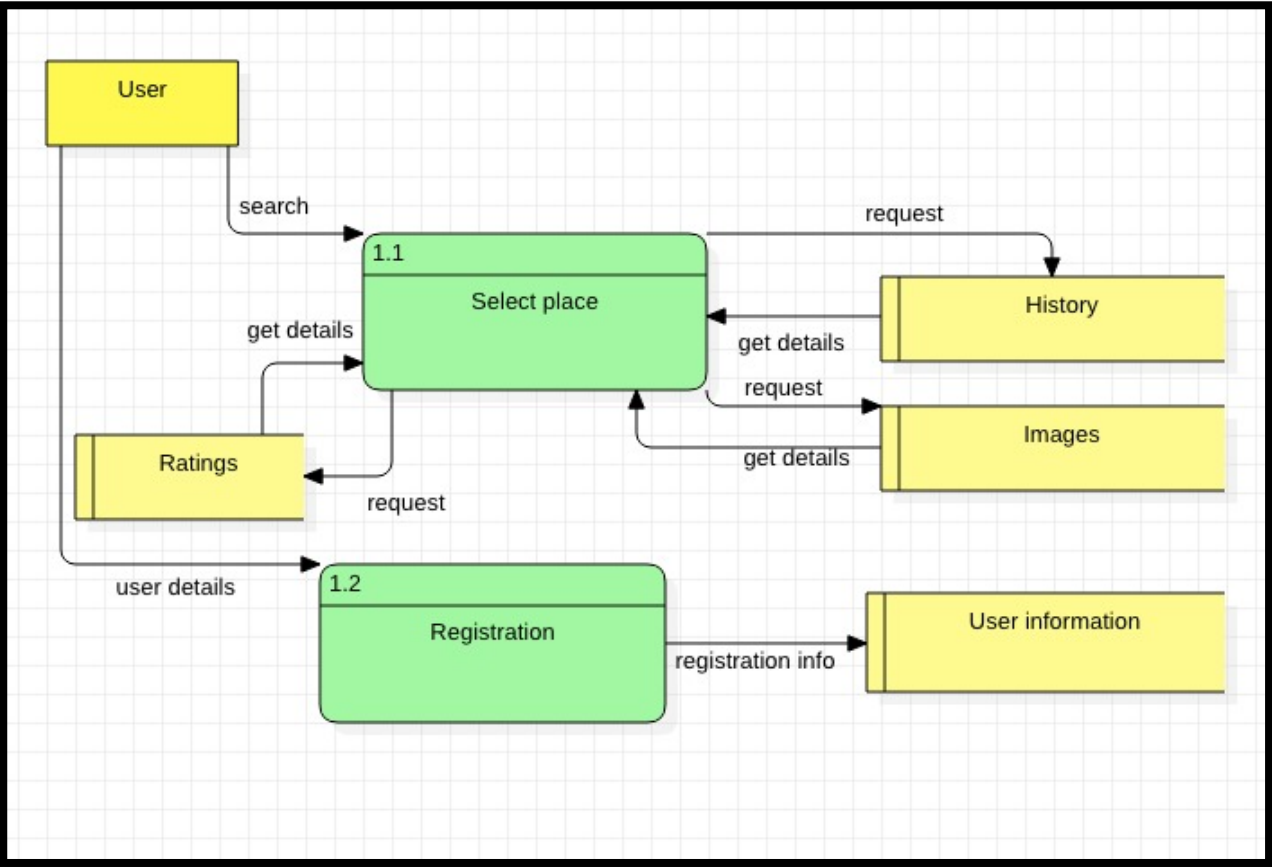


## 1-level Data Flow

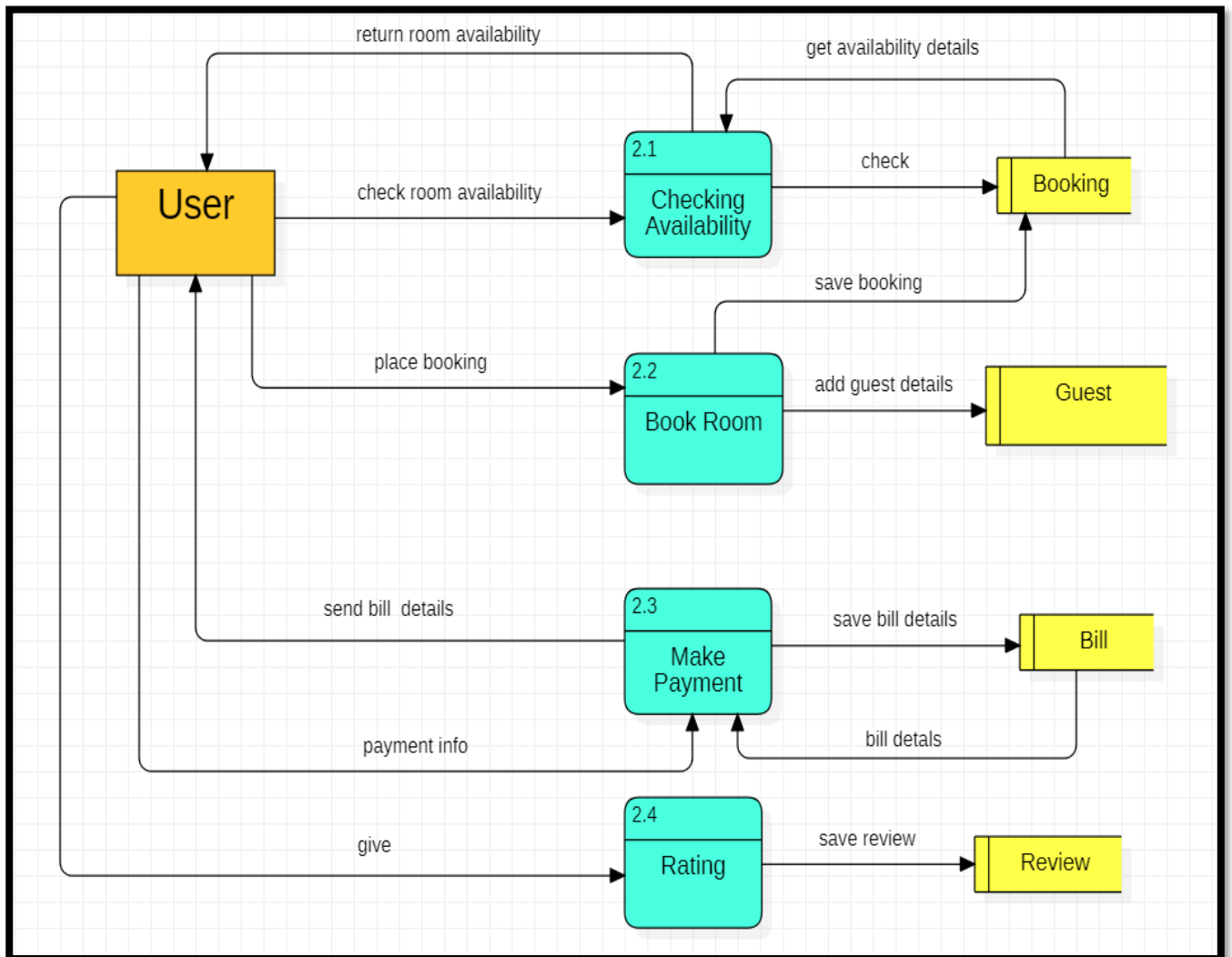


2-level Data Flow

Tour Location:



## Hotel:



**Conclusion:** We understood how to construct different levels of data flow diagram and its use.