SendNodesMarkII Bookwork Phase 1

CHAPTER 1:

1.1

- Data – facts and statistics collected together for reference or analysis.

- Database – a structured set of data held in a computer, especially one that is accessible in various ways.

- DBMS – system software for creating and managing databases.

- Database system – a software system that facilitates the creation and maintenance and use of an electronic database

- Data-base catalog – consists of metadata in which definitions of database objects such as base tables, views (virtual tables), synonyms, value ranges, indexes, users, and user groups are stored.

- Program-data independence – a type of data transparency that refers to the immunity of changes made to users applications and their definition and organization of data.

- User view – A view of part or all of the contents of a database specified to facilitate a particular purpose or user activity.

- DBA – Database administrator

- End user – the person who actually uses a particular product.

- Canned transaction – standard types of queries and updates which are frequently used by Naive end users to constantly querying and updating database.

- Deductive database system – a database system that can make deductions (i.e., conclude additional facts) based on rules and facts stored in the (deductive) database.

- Persistent object – In object technology, a persistent object is one that continues to exist after the program that created it has been unloaded.

· Meta-data – a set of data that describes and gives information about other data.

· Transaction-processing application – information processing in computer science that is divided into individual, indivisible operations called transactions.

1.2

Four main types of actions:

· Define database – Define database with database structures and types i.e the metadata of the data.
· Construct database – Process of data being stores and maintained by DBMS.
· Manipulate database – Retrieve the database by using query; insert, update, and delete the database.
· Share database – Share the information so the data can be used amongst multiple users.

1.3

A database management system exists on top of a traditional file system. The database provides table abstraction, using rows and columns. Data is strongly typed. A database management system provides a query language in which can be used to query and maintain tables.

A traditional file system provides files to store data. Privileges can be updated to determine who can control and access the files. Often, file systems include features such as keyed access and partitioning and encryption. Some file systems also include integration with a transaction manager.

CHAPTER 2:

2.1

· Data Model: A collection of concepts that can be used to describe the structure of a database.

· Database Schema: The description of a database.

· Database State: The data in a database at a particular moment in time.

· Internal Schema: describes the physical storage structure of the database; Uses a physical data model and describes the complete details of data storage and access paths for the database.

· Conceptual Schema: describes the structure of the whole database for a community of users; Hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints.

· External Schema: describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

· Data Independence: The capacity to change the schema at one level of a database system without having to change the schema at the next higher level.

· DDL (Data Definition Language): Used by the DBA and by database designers to define both the conceptual and internal schemas.

· DML (Data Manipulation Language): Used to manipulate the database including: retrieval, insertion, deletion, and modification of data.

· SDL (Storage Definition Language): Used to specify the internal schema, when there is a clear separation maintained between the conceptual and internal levels.

· VDL (View Definition Language): Specifies user views and mappings to the conceptual schema in relational DBMSs, where SQL is used in the role of VDL.

· Query Language: A high-level DML used in a standalone interactive manner.

· Host Language: The language, whenever DML commands, whether high-level or low-level, are embedded in a general-purpose programming language.

· Data Sublanguage: The DML, whenever DML commands, whether high-level or low-level, are embedded in a general-purpose programming language.

· Database Utility: Help the DBA manage the database system including: Loading, backup, database storage reorganization, and performance monitoring.

· Catalog: Stores information such as the structure of each file, the type and storage format of each data item, and various constraints on the data.

· Client/Server Architecture: Developed to deal with computing environments in which software and equipment are connected via a network.

· Three-tier Architecture: Adds an intermediate/middle layer, called the application/web server, between the client and the databases server.

· N-tier Architecture: Created by dividing the layers between the user and the stored data further into finer components.

2.3

The database schema is a description of the database, whereas, the database state, or snapshot, is data in the database at a particular moment in time.

CHAPTER 3:

3.3

· Entity: A representation of a real-world object or concept, such as an employee or a project from the miniworld described in the database.

· Attribute: Some relevant property of the entity that further describes it, such as an employee's name or salary.

· Attribute Value: The actual data that is stored and represents an attribute.

· Relationship Instance**:** A relationship instance r_i among the relationship set R, where r_i associates exactly one entity from each participating entity type.

· Composite Attribute: Attributes which can be divided into smaller subparts, which represent more basic attributes with independent meanings.

· Multivalued Attribute: An attribute that can have different numbers of values for each entity, and may have lower and upper bounds to constrain the numbers allowed for each individual entity.

· Derived Attribute: An attribute which can be determined from either another attribute known as a stored attribute, or from related entities; for example, number of employees of a department can be determined from counting the number of employees related to the department.

· Complex Attribute: The nesting of composite and multivalued attributes by grouping components of a composite attribute between parentheses, separating the components with commas, and displaying multivalued attributes between braces.

· Key Attribute: An attribute whose values are distinct for each individual entity in the entity set, and which can be used to identify the individual entity.

· Value Set (Domain): The set of values associated to each simple attribute of an entity type, which defines the set of values that may be associated to that attribute.

3.4

An entity type is a collection of entities that have the same attributes. They share the same attributes, but with different values. They are described by name and attributes. An entity set is the collection of all entities of a particular entity type at any time in the database.
An entity is a real-world object or concept. Entities of similar types and information make up an entity type, with names and attributes. The collection of entities at a given time within a database makes up an entity set, which can differ between other times.

3.6

A relationship type is a set of relationships among entities from a set of entity types.
A relationship instance is one single relationship between each participating entity type.
A relationship type is a name for the set of relationships between each entity type (for example, the entity set employee has the relationship works_for with entity set department). The relationship set is the set of all relationships between each individual entities of the participating entity types.

CHAPTER 4:

4.2

· Superclass of a subclass: In each entity type, there are smaller groupings on basis of one or other attribute. All groups like these can be represented as separate classes or entity types. These form subclass of bigger entity type. The bigger entity is called superclass.

· Superclass/subclass relationship: The relationship between the subclass and the super class is known as Superclass/subclass relationship.

· IS-A relationship: A Superclass/subclass relationship is often called a is-a relationship because of the way in which concept is referred

· Specialization: Specialization is a process of defining a set of subclass of an entity type. The set of subclass that forms a specialization is defined on basis of some distinguishing characteristic of the entities in the superclass.

· Generalization: The opposite of specialization, a bottom-up approach. Instead of finding a new attribute, class relationship is made by finding similar attributes.

· Category: used when need arises for modeling a single superclass/subclass relationship with more than one superclass, where the super classes represent different entity types.

· Specific (local) attributes: single-valued attribute where the value can be mono or zero. It is tied to a specific type and its subtypes. Stored in columns directly on the underlying object class.

· Specific relationship: Relationships that are only true for a subclass of superclass and not for all subclasses or for superclass.

4.8

The process of extracting common characteristics from two or more classes and combining them into a generalized superclass, is called Generalization. The common characteristics can be attributes or methods. Generalization is represented by a triangle followed by a line.
Specialization is the reverse process of Generalization means creating new sub classes from an existing class.

Generalization is a bottom-up approach in which two lower level entities combine to form a higher level entity. In generalization, the higher level entity can also combine with other lower level entity to make further higher level entity. **Specialization** is opposite to Generalization. It is a top-down approach in which one higher level entity can be broken down into two lower level entity. In specialization, some higher level entities may not have lower-level entity sets at all.
so, the schema diagrams are same.