

|  |
| --- |
| **Database:** A database is a collection of related data which represents some aspect of the real world. A database system is designed to be built and populated with data for a certain task. |

|  |
| --- |
| An entity set is a set of the same type of entities.   * **Strong Entity Set:** * A strong entity set is an entity set that contains sufficient attributes to **uniquely identify** all its entities. * In other words, a **primary key exists** for a strong entity set. * **Primary key** of a strong entity set is **represented** by **underlining** it. * **Weak Entity Set:** * A weak entity set is an entity set that **does not contain sufficient attributes** to uniquely identify its entities. * In other words, a **primary key does not exist** for a weak entity set. * However, it contains a **partial key** called a **discriminator**. * **Discriminator** can identify a group of **entities** from the **entity set**. * **Discriminator** is represented by **underlining** with a **dashed line**. |

|  |
| --- |
| A **relationship** is defined as an **association** among several entities.   1. **Unary Relationship Set** - Unary relationship set is a relationship set where only one entity set participates in a relationship set. 2. **Binary Relationship Set** - Binary relationship set is a relationship set where two entity sets participate in a relationship set. 3. **Ternary Relationship Set** - Ternary relationship set is a relationship set where three entity sets participate in a relationship set. 4. **N-ary Relationship Set** - N-ary relationship set is a relationship set where ‘n’ entity sets participate in a relationship set. |

|  |
| --- |
| **Attributes** are the **descriptive properties** which are **owned by each entity** of an Entity Set. **Types** of **Attributes**:   * **Simple Attributes** - **Simple attributes** are those attributes which **cannot be divided** further. Ex. Age * **Composite Attributes** - **Composite attributes** are those attributes which are **composed of** many **other simple attributes**. Ex. Name, Address * **Multi Valued Attributes** - Multi valued attributes are those attributes which can take more than one value for a given entity from an entity set. Ex. Mobile No, Email ID * **Derived Attributes** - Derived attributes are those attributes which can be derived from other attribute(s). Ex. Age can be derived from DOB. * **Key Attributes** - Key attributes are those attributes which can identify an entity uniquely in an entity set. Ex. Roll No. |

|  |
| --- |
| A key is a set of attributes that can identify each tuple uniquely in the given relation.  Types of Keys:   * **Super Key** - A super key is a set of attributes that can identify **each tuple uniquely** in the given relation. A **super key** may **consist** of **any number of attributes**. * **Candidate Key** - A set of **minimal attribute**(s) that can identify each tuple **uniquely** in the given relation is called a **candidate key**. * **Primary Key** - A primary key is a candidate key that the database designer selects while designing the database. **Primary** Keys are **unique** and **NOT NULL**. * **Alternate Key** - Candidate keys that are left **unimplemented** or **unused** after implementing the primary key are called as alternate keys. * **Foreign Key** - An attribute ‘X’ is called as a foreign key to some other attribute ‘Y’ when its values are dependent on the values of attribute ‘Y’. The relation in which attribute ‘Y’ is present is called as the referenced relation. The relation in which attribute ‘X’ is present is called as the referencing relation. * **Composite Key** - A primary key composed of multiple attributes and not just a single attribute is called a composite key. * **Unique Key** - It is unique for all the records of the table. Once assigned, its value cannot be changed i.e. it is non-updatable. It may have a NULL value. |

|  |
| --- |
| Properties of Decomposition:   1. **Lossless Decomposition** - Lossless decomposition ensures    1. No information is lost from the original relation during decomposition.    2. When the sub relations are joined back, the same relation is obtained that was decomposed. 2. **Dependency Preservation** - Dependency preservation ensures    1. None of the functional dependencies that hold on the original relation are lost.    2. The sub relations still hold or satisfy the functional dependencies of the original relation. |

|  |
| --- |
| **ACID** properties are a set of principles that ensure the **reliability** and **consistency** of data in a database system. ACID stands for **Atomicity**, **Consistency**, **Isolation**, and **Durability**. Here are some real-life examples of ACID properties in database transactions: |
| **Atomicity**: Suppose you want to book a flight and a hotel for your vacation. You use an online travel agency that offers a package deal for both. The transaction involves two operations: reserving a seat on the flight and booking a room in the hotel. Atomicity means that either both operations are completed successfully, or none of them are. If the flight reservation fails, for example, due to unavailability of seats, then the hotel booking is also canceled. This way, you don’t end up paying for a hotel without a flight, or vice versa. |
| **Consistency**: Suppose you have a bank account with a balance of $1000. You withdraw $200 from an ATM and then check your balance from another ATM. Consistency means that the balance shown on both ATMs is $800, reflecting the effect of your withdrawal. The database maintains the **integrity** of the data by enforcing rules and constraints, such as the balance cannot be negative, or the sum of all debits and credits must be zero. |
| **Isolation**: Suppose you and your friend share a Netflix account and want to watch different shows at the same time. You log in from your laptop and your friend logs in from his phone. Isolation means that each of you can access and modify the data on your own device without affecting the other’s view or experience. The database ensures that concurrent transactions do not interfere with each other by using locking mechanisms or other techniques. |
| **Durability**: Suppose you order a pizza online and pay with your credit card. The transaction involves updating the inventory of the pizza store, charging your credit card, and sending you a confirmation email. Durability means that once the transaction is completed successfully, the changes made to the database are permanent and will not be lost or undone even if there is a power outage, a system crash, or a network failure. The database achieves durability by writing the changes to a log file or a backup device. |