1. What do you mean by Entity- Relationship Diagram? Explain.

ANS: An entity relationship model, also called an entity-relationship (ER) diagram, is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of [data](http://www.webopedia.com/TERM/D/data.html) within [databases](http://www.webopedia.com/TERM/D/database.html) or information systems. An entity is a piece of data-an [object](http://www.webopedia.com/TERM/O/object.html)or concept about which data is stored.

1. Define entity and give an example.

ANS: An entity is something that [exists](https://en.wikipedia.org/wiki/Existence) in itself, actually or potentially, concretely or abstractly, physically or not. It need not be of material existence. In particular, [abstractions](https://en.wikipedia.org/wiki/Abstraction) and [legal fictions](https://en.wikipedia.org/wiki/Legal_fiction) are usually regarded as entities. In general, there is also no presumption that an entity is [animate](https://en.wikipedia.org/wiki/Life).

An entity is a grouping of things with rules or data in common. An entity often represents a group of people (eg children, applicants, stakeholders) but it can also represent a group of objects (eg textbooks), activities (eg assignments) or concepts (eg school terms).

By defining entities, the same set of rules can be used for multiple instances of the same type, and rules can be written which relate to all of those instances.

A member of the entity group is called an entity instance. For example, if a family had 2 children, Sarah and Peter, Sarah would be one instance of "the child" entity and Peter would be another instance of "the child" entity. By creating an entity to represent "the child", information such as the child's age can be collected for each child.

We can also infer attributes from this data just as we can in a normal rule. Therefore, the attribute "the child is a young child" can hold a different value for each child, depending on the child's age. For instance, if we had a rule:

the child is a young child if the child's age < 8

we could infer the following:

| **Child Instance 1 (Sarah)** | **Child Instance 2 (Peter)** |
| --- | --- |
| the child's age = 6 | the child's age = 12 |
| the child is a young child = true | the child is a young child = false |

1. Explain the different between an entity class and an entity instance.

ANS:

**4. Define attribute and its types.**

In general, an attribute is a characteristic property of an entity. In a database management system (DBMS), an attribute refers to a database component, such a table. It also may refer to a database field. Attributes describe the instances in the row of a database.

The types of attributes are:

**Required or Optional Attributes:**

A required attribute is an attribute that must have a value in it, while an optional attribute may not have a value in it and can be left blank. The reasoning for making an attribute required is to put emphasis on what is important in that entity and what makes it stand out from other entities.

Example**:** Consider the entity Student above stud\_LastName and studFirstName would be required attributes as it uniquely defines that table and we assume all students have a first and last name. Optional attributes in the table Student could be stu\_MiddleName, stu\_Email, and stu\_Phone since some students may not have a middle name, a phone number, or an email address .

**Keys and non-keys Attributes:**

In every entity an attribute or grouped attributes uniquely identify that entity. These attributes are the key attributes and range from Primary key (single attribute identifier) to a Composite Key (Multi attribute Identifier). The rest of the attributes after the identifier are considered the non-key attributes or descriptors, which just describe the entity.

Example: Above in the table Student there is only one unique identifier, stu\_LastName, which is the primary key of the table. The rest of the attributes are descriptors.

**Single and Composite Attributes:**

Attributes can be classified as having many parts to them or just a single unbreakable attribute. The composite attribute is an attribute that can be subdivided into other single attributes with meanings of their own. A single attribute is just an attribute that cannot be subdivided into parts.

Example: Imagine from the entity Student that instead of having the three attributes: stu\_LastName, stu\_MiddleName, stu\_FirstName it had one attribute called stu\_Name. The attribute stu\_Name would be considered a composite attribute since it can be subdivided into the other three attributes: stu\_LastName, stu\_MiddleName, stu\_FirstName. The rest of attributes would be consider single attributes since they can't be subdivided into parts.

**Single-valued and multi-valued Attributes:**

Attributes can be classified as single or multi-value. The single-value attribute can only have one value, while the multi-valued attributes usually can store multiple data in them.

Example: In the entity Student, stu\_Address could be considered a multi-value attribute since a student could have multiple addresses where he lives at. An example of a single-value attribute would be stu\_LastName since a student usually has one last name that uniquely identifies him/her.

**Derived Attributes:**

Derived attribute, where one attribute is calculated from another attribute. The derived attribute may not be stored in the database but rather calculated using algorithm.

Example: In the entity Student, stu\_Age would be considered a derived attribute since it could be calculated using the student's date of birth with the current date to find their age.﻿﻿

**5. What is derived attributes?**

The attribute from which another attribute value is derived is called derived or stored attribute.

Derived attributes are usually created by a formula or by a summary operation on other attributes.

There may be a case when two or more attributes values are related. Take the example of age. Age of a person can be calculated from person’s date of birth and present date. Difference between the two gives the value of age. In this case, age is the derived attribute.

**6. Define relationship and give an example.**

A relationship, in the context of databases, is a situation that exists between two relational database tables when one table has a foreign key that references the primary key of the other table. Relationships allow relational databases to split and store data in different tables, while linking disparate data items.

For example, in a bank database a CUSTOMER\_MASTER table stores customer data with a primary key column named CUSTOMER\_ID; it also stores customer data in an ACCOUNTS\_MASTER table, which holds information about various bank accounts and associated customers. To link these two tables and determine customer and bank account information, a corresponding CUSTOMER\_ID column must be inserted in the ACCOUNTS\_MASTER table, referencing existing customer IDs from the CUSTOMER\_MASTER table. In this case, the ACCOUNTS\_MASTER table’s CUSTOMER\_ID column is a foreign key that references a column with the same name in the CUSTOMER\_MASTER table. This is an example of a relationship between the two tables.