**Master data vs Transaction data:**

Transaction data deals with information related to a business activity, for example Sales, Sales Returns, Purchases, Purchase Returns, Payments, Material Movement etc.

Master data deals with the information related to the entities that are involved in a transaction/business activity, for example Material, Plant, Customer, Vendor, Bank etc.

The Sales activity involves Customer and Material. There cannot be any sale activity without knowing to which customer the sale is being made, and what material (or materials) is being sold. In this example Customer and Material are master data, Sales Order is transaction data.

In a situation where a customer is returning some of the materials delivered as they were found to be defective, such a business activity is called Sales Returns. Sales Returns is an example of transaction data. This transaction also involves Customer master and Material master data.

Extending the above example further, if sales of a business are being made in different parts of the world and in different currencies, then the Sales Order should also relate to which Country, which Currency and the Banks that are involved. Data related Country, Currency and Bank are also master data.

Salary payment is a business activity hence this is transaction data. But, to pay salaries we need to know to which employee, and through which bank account the payment is made. The entities Employee and Bank Account are master data which is necessary to execute the salary payment transaction.

* Master data can exist independent of transaction data, but vice versa is not true.
* Master data changes less frequently while transaction data is always subject to frequent changes.
* Database tables storing master data will have a lesser number of records compared to the number of records in the tables that store transaction data.

Examples of Master Data:

Customer data

Vendor data

Material data

Employee data

Bank data

Examples of Transaction Data:

Sales Order data / Sales Returns data

Purchase Order data / Purchase Returns data

Delivery data

Salary payment data

Material Movement data

Explain the below **cardinal-relationships** in database with appropriate examples:

1. One-to-One
2. One-to-Many / Many-to-one
3. Many-to-Many

Let’s say there are two tables named A and B:

**One-to-One:**

For one row in table-A if there can be zero or a maximum of one related row in table-B, then the relationship between the tables A and B is said to be 1-to-1.

Examples:

1. One Employee can have only one user of the company’s intranet application.

Two common ways to implement one-to-one relationships in databases:

1. Using Shared Primary Keys: The same primary key is used for both tables.

This ensures that each record in one table is associated with exactly one record in the other table.

1. Using Separate Primary Keys and a Foreign Key:

Each table has its own primary key.

One table has a foreign key that references the primary key of the other table.

EMP

**EMP\_NO**

EMP\_NAME

EMAIL

USER

**EMP\_NO** *(FK)*

ACTIVE (Boolean)

VALID\_UPTO (Date)

EMP

**EMP\_NO**

EMP\_NAME

EMAIL

***USER\_ID*** *(Unique & FK)*

USER

**USER\_ID**

ACTIVE (Boolean)

VALID\_UPTO (Date)

**One-to-Many:**

For one row in table-A if there can be zero or **more** related rows in table-B, then the relationship between the tables A and B is said to be 1-to-M.

Examples:

Customer - Orders:

* Each customer can place multiple orders.
* Each order belongs to only one customer.

To implement one-to-many relationship in database:

Have a foreign key in the "many" table that references the primary key of the "one" table.

In the above example Customer-Orders the database table ORDERS must have a CUST\_ID field with foreign key referencing the primary key field of CUSTOMER table.

**Many-to-Many:**

For multiple rows in table-A if there can be multiple related rows in table-B, then the relationship between the tables A and B is said to be M-to-M.

Examples:

Employee – Qualification:

* Each employee can have multiple qualifications
* Each qualification can be held by multiple employees

Student – Courses:

* Each student can study multiple courses
* Each course can be studied by multiple students

To implement many-to-many relationship in database:

1. Create two tables with their respective primary keys.
2. Create a third table which is referred to as Junction Table or Join Table. In this table create a composite primary key comprising the primary key columns of the first two tables.
3. The individual columns of the composite primary key of the third table should have a foreign key relation with the primary key columns of the respective tables.