**Explain the relationship between the "Product" and "Product\_Category" entities from the above diagram.**

The relationship between the "product" table and the "product\_category" table is likely a one-to-many relationship, where one product category can have multiple products associated with it. This relationship is established through the "category\_id" column in the "product" table, which likely serves as a foreign key referencing the primary key of the "product\_category" table.

In this scenario, the "category\_id" column in the "product" table would store the identifier of the corresponding product category from the "product\_category" table. This allows each product to be assigned to a specific category.

Additionally, the "product\_category" table may have a primary key column (perhaps named "id") that serves as the primary key for identifying each unique category. This primary key would likely be referenced by the "category\_id" foreign key column in the "product" table.

It's important to note that without knowing the exact schema and constraints of the database, such as primary keys, foreign keys, and any additional constraints, it's difficult to provide a precise answer. However, based on the information provided, this is the likely relationship between the two tables.

### How could you ensure that each product in the "Product" table has a valid category assigned to it?

1. Define a Foreign Key Constraint: You need to define a foreign key constraint on the "category\_id" column in the "Product" table, referencing the primary key of the "Product\_Category" table.

2. Enforce NOT NULL Constraint: Ensure that the "category\_id" column in the "Product" table does not allow NULL values. This ensures that every product must have a valid category assigned to it.

3. Maintain Consistency: Ensure that the category IDs referenced in the "Product" table exist in the "Product\_Category" table. This can be achieved by defining the foreign key constraint with the appropriate reference options (e.g., ON DELETE CASCADE, ON UPDATE CASCADE) to maintain referential integrity.

EXAMPLE:

ALTER TABLE Product

ADD CONSTRAINT FK\_Product\_Category

FOREIGN KEY (category\_id)

REFERENCES Product\_Category(id)

ON DELETE CASCADE; -- Optional: Specify the desired reference options

### Create schema in any Database script or any ORM (Object Relational Mapping).

CREATE TABLE Product (

id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(255) NOT NULL,

description TEXT,

sku VARCHAR(50),

category\_id INT,

inventory\_id INT,

price DECIMAL(10, 2) NOT NULL,

discount DECIMAL(5, 2),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

modified\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

deleted\_at TIMESTAMP,

FOREIGN KEY (category\_id) REFERENCES Product\_Category(id) ON DELETE CASCADE

);

This SQL script creates two tables: "Product. The "Product" table has columns for the product ID, name, description, SKU, category ID (which references the ID of a category . The foreign key constraint ensures that each product must have a valid category assigned to it.

This script can be executed in a SQL client or any database management tool to create the schema in a database.