

Product Attributes

**Database Design**

**ERD (Entity Relationship Diagram)**

**Relationships:**

* **One-to-Many between Categories and Products:** A Category can have many Products, but each Product belongs to only one Category. This is enforced by the category\_id Foreign Key in the Products table.
* **One-to-Many between Products and Product Attributes:** A Product can have many Product Attributes (one for each custom attribute it possesses), but each Product Attribute belongs to only one Product. This is enforced by the product\_id Foreign Key in the Product Attributes table
* **Justification of Design Decisions**

1. **Normalization (Third Normal Form - 3NF):**
   * The design separates categories, products, and product attributes into distinct tables. This avoids data redundancy. For example, the attribute name (like "OS" or "RAM") is not repeated within the Products table for every product. Instead, it's stored once as a distinct entry for each product's specific attribute value in Product Attributes.
   * This also prevents update and deletion anomalies, as changes to category details or attribute names don't require modifying numerous product records.
2. **Flexibility for Dynamic Attributes:**
   * The Product Attributes table is key to handling **custom attributes per category**. Instead of adding new columns to the Products table every time a new attribute type is needed (which would require a schema migration and make the Products table very wide and sparse), the attribute name and its value are stored as rows.
   * This allows for categories to have completely different sets of attributes without altering the core database schema. If "Laptops" need "CPU" and "Storage," you just add new Product Attribute entries for each laptop, referencing the same attribute\_name if consistent across products.

Category\_Id

Name (VARCHAR)

Description (Text)

Categories

Attribute\_Id

Product\_id (FK)

Attribute\_name (VARCHAR)

Attribute \_value (Text)

Products

Product\_Id

Product\_id (FK)

Name (VARCHAR)

Price (Decimal)

Image.url (VARCHAR)

Description (Text)