## **Database Design Prompt: Philippine Supplier and Product Management System**

**1. Objective:**

To design a relational database schema for a Philippine-based business to effectively manage information about its suppliers, including their multiple contact details (addresses, phone numbers, emails), associated contact persons, and a comprehensive catalog of products they offer. The system must accurately track product pricing, specifically "Dealer's Price" (exclusive of EVAT), and maintain a history of price changes based on entry date and supplier. The database should also manage the active/inactive status of suppliers, contact persons, and products.

**2. Core Entities & Attributes:**

2.1. Suppliers Table:

\* SupplierID (Primary Key, e.g., INT, Auto-increment)

\* TIN (VARCHAR(20), Unique, Not Null) - Taxpayer Identification Number.

\* CompanyName (VARCHAR(255), Not Null)

\* DateCreated (TIMESTAMP, Not Null, Default: CURRENT\_TIMESTAMP)

\* DateUpdated (TIMESTAMP, Not Null, Default: CURRENT\_TIMESTAMP, On Update: CURRENT\_TIMESTAMP) - Automatically updates when the supplier's core record is modified.

\* Status (ENUM('Active', 'Inactive'), Not Null, Default: 'Active')

2.2. SupplierAddresses Table: (To handle multiple addresses per supplier)

\* AddressID (Primary Key, e.g., INT, Auto-increment)

\* SupplierID (Foreign Key referencing Suppliers.SupplierID, Not Null)

\* AddressLine1 (VARCHAR(255), Not Null, e.g., Unit/Bldg No., Street Name)

\* AddressLine2 (VARCHAR(255), Optional, e.g., Subdivision)

\* Barangay (VARCHAR(100), Optional)

\* CityMunicipality (VARCHAR(100), Not Null)

\* Province (VARCHAR(100), Not Null)

\* PostalCode (VARCHAR(10), Optional)

\* AddressType (VARCHAR(50), Optional, e.g., 'Head Office', 'Branch', 'Warehouse')

\* IsPrimary (BOOLEAN, Default: FALSE) - Indicates if this is the main/default address.

2.3. SupplierContactNumbers Table: (To handle multiple contact numbers per supplier)

\* ContactNumberID (Primary Key, e.g., INT, Auto-increment)

\* SupplierID (Foreign Key referencing Suppliers.SupplierID, Not Null)

\* ContactNumber (VARCHAR(50), Not Null)

\* NumberType (VARCHAR(50), Optional, e.g., 'Mobile', 'Landline', 'Fax')

\* IsPrimary (BOOLEAN, Default: FALSE)

2.4. SupplierEmailAddresses Table: (To handle multiple email addresses per supplier)

\* EmailAddressID (Primary Key, e.g., INT, Auto-increment)

\* SupplierID (Foreign Key referencing Suppliers.SupplierID, Not Null)

\* EmailAddress (VARCHAR(255), Not Null, Validate for email format)

\* EmailType (VARCHAR(50), Optional, e.g., 'Sales', 'Support', 'General')

\* IsPrimary (BOOLEAN, Default: FALSE)

2.5. ContactPersons Table: (To handle multiple contact persons per supplier)

\* ContactPersonID (Primary Key, e.g., INT, Auto-increment)

\* SupplierID (Foreign Key referencing Suppliers.SupplierID, Not Null)

\* FirstName (VARCHAR(100), Not Null)

\* LastName (VARCHAR(100), Not Null)

\* Position (VARCHAR(100), Optional)

\* EmailAddress (VARCHAR(255), Optional, Validate for email format)

\* ContactNumber (VARCHAR(50), Optional)

\* DateCreated (TIMESTAMP, Not Null, Default: CURRENT\_TIMESTAMP)

\* DateUpdated (TIMESTAMP, Not Null, Default: CURRENT\_TIMESTAMP, On Update: CURRENT\_TIMESTAMP)

\* Status (ENUM('Active', 'Inactive'), Not Null, Default: 'Active')

2.6. Products Table: (Master list of all products)

\* ProductID (Primary Key, e.g., INT, Auto-increment or VARCHAR for SKU)

\* ProductName (VARCHAR(255), Not Null, Unique)

\* ProductDescription (TEXT, Optional)

\* Category (VARCHAR(100), Optional)

\* UnitOfMeasure (VARCHAR(50), Optional, e.g., 'PC', 'BOX', 'KG')

\* DataEntryDate (TIMESTAMP, Not Null, Default: CURRENT\_TIMESTAMP) - Date product was first added to this master list.

\* Status (ENUM('Active', 'Inactive', 'Discontinued'), Not Null, Default: 'Active')

2.7. SupplierProductCatalog Table: (Junction table linking Suppliers to Products and managing prices)

\* SupplierProductCatalogID (Primary Key, e.g., INT, Auto-increment)

\* SupplierID (Foreign Key referencing Suppliers.SupplierID, Not Null)

\* ProductID (Foreign Key referencing Products.ProductID, Not Null)

\* DealersPrice (DECIMAL(12, 2), Not Null) - Price exclusive of EVAT.

\* PriceEntryDate (DATE, Not Null) - The date this specific price from this supplier for this product was recorded or became effective.

\* SupplierProductCode (VARCHAR(100), Optional) - Supplier's own code for the product, if different.

\* Notes (TEXT, Optional, e.g., 'Promo price until YYYY-MM-DD')

\* Composite Unique Key: (SupplierID, ProductID, PriceEntryDate) - To ensure a product from a supplier doesn't have duplicate prices for the same effective date.

**3. Relationships:**

* **Suppliers to SupplierAddresses**: One-to-Many (A supplier can have multiple addresses).
* **Suppliers to SupplierContactNumbers**: One-to-Many (A supplier can have multiple contact numbers).
* **Suppliers to SupplierEmailAddresses**: One-to-Many (A supplier can have multiple email addresses).
* **Suppliers to ContactPersons**: One-to-Many (A supplier can have multiple contact persons).
* **Suppliers to Products**: Many-to-Many, implemented through the SupplierProductCatalog table.
  + A supplier can offer 0 to many products.
  + A product can be offered by 0 to many suppliers.
  + A product can have 0 to many prices from the same supplier, differentiated by PriceEntryDate in the SupplierProductCatalog.
  + A product in the Products table represents its existence in the overall database/catalog. Its linkage to a supplier (and its price from that supplier) is via the SupplierProductCatalog.

**4. Key Considerations & Constraints:**

* **Philippine Context:**
  + **TIN:** Ensure validation for the Philippine TIN format if possible, or at least store it accurately.
  + **EVAT:** The DealersPrice explicitly *excludes* EVAT. The application layer will be responsible for calculating and adding EVAT as needed.
  + **Addresses:** Accommodate typical Philippine address components (Barangay, City/Municipality, Province).
* **Data Integrity:**
  + Implement Foreign Key constraints with appropriate ON DELETE and ON UPDATE actions (e.g., RESTRICT, SET NULL, or CASCADE depending on business rules). For instance, deleting a Supplier might cascade to delete their addresses, contact numbers, etc., or be restricted if they have product price entries.
* **Indexing:**
  + Create indexes on all Foreign Keys.
  + Index TIN in Suppliers.
  + Index CompanyName in Suppliers.
  + Index ProductName in Products.
  + Consider composite indexes for frequently queried combinations, e.g., (SupplierID, ProductID) in SupplierProductCatalog.
* **Uniqueness:**
  + TIN in Suppliers must be unique.
  + ProductName in Products must be unique.
  + The combination of (SupplierID, ProductID, PriceEntryDate) in SupplierProductCatalog should be unique.
* **Status Fields:** The Status columns are crucial for soft-deleting records and managing the lifecycle of suppliers, contacts, and products without losing historical data.
* **Date Tracking:**
  + DateUpdated fields help in auditing changes.
  + PriceEntryDate in SupplierProductCatalog is critical for historical price tracking.
  + DataEntryDate in Products tracks when a product was first cataloged.
* **Default Values:** Use default values appropriately (e.g., for Status, DateCreated).
* **Data Types:** Select the most appropriate data types for each field to ensure data accuracy and storage efficiency (e.g., DECIMAL for currency, VARCHAR with suitable lengths, TIMESTAMP vs. DATE).

This detailed prompt should provide a solid foundation for a database developer to create the required schema.