Name:Karam Zohud

The supply_db database is structured to facilitate a full-featured supply chain management system. It includes essential tables for suppliers, products, customers, orders, order details, and payments, alongside a partitioned orders table and indexed columns to enhance query efficiency. Here's an overview of the database design and functionality:

1. **Supplier Table**: This table, supplier, holds details on each supplier, with Supplier_ID as the primary key, alongside fields for Name, Contact_Info, and Rating. Each supplier can provide multiple products, creating a one-to-many relationship between suppliers and products, referenced by Supplier_ID in the Products table.

2. **Products Table**: The Products table records product information, using Product_ID as the primary key. Other fields include Name, Price, Supplier_ID, and Stock_Quantity. Supplier_ID serves as a foreign key linking each product to its supplier. A cascade setting is applied such that if a supplier is deleted, Supplier_ID is set to NULL in related products. A CHECK_PRICE constraint ensures that product prices are always positive, maintaining data integrity.

3. **Customers Table**: This table stores customer data with Customer_ID as the primary key. Additional fields include Name, Address, Email, and Credit_Score, with a constraint (CHECK_CREDIT_SCORE) enforcing valid credit scores between 300 and 850.

4. **Orders Table**: The Orders table connects customer orders to the Customers table through a foreign key

(Customer_ID), with Order_ID as the primary key. It also tracks Order_Date and Shipping_Status. When a customer is deleted, the corresponding Customer_ID is set to NULL to preserve order history.

5. **Order_Details Table**: This table records each item within an order, using Order_Detail_ID as the primary key and linking back to Orders and Products tables via Order_ID and Product_ID foreign keys. It contains fields for Quantity, Unit_Price, and Discount. Deleting an order will delete related order details (CASCADE), and Product_ID is set to NULL if a product is deleted.

6. **Payments Table**: This table stores payment records, using Payment_ID as the primary key. Fields include Customer_ID, Order_ID, Payment_Date, Payment_Amount, and Payment_Method. Foreign keys link back to Customers and Orders, ensuring payments are removed if related customers are deleted, and setting Order_ID to NULL if an order is deleted.

Additional Features:

- **Indexes**: Foreign key indexes on Orders, Products, Order_Details, and Payments tables improve query speed.
- **Trigger**: A trigger, tr_update_stock, updates Stock_Quantity in the Products table upon new entries in Order_Details, reflecting stock changes immediately.
- **Partitioning**: The Orders_partitioned table is partitioned by year, using range partitioning for better query performance and historical data management.

This database design supports data integrity, efficiency, and optimized performance for managing a supply chain system.

Constraints, triggers, and indexes are incorporated to ensure accuracy and reliability in data handling.