use super\_store;

CREATE TABLE customers (

customer\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(255) NOT NULL,

email VARCHAR(255) UNIQUE,

phone VARCHAR(20),

address TEXT NOT NULL, -- Added Address Column

registration\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE vendors (

vendor\_id INT PRIMARY KEY AUTO\_INCREMENT,

vendor\_name VARCHAR(255) NOT NULL,

contact\_email VARCHAR(255) UNIQUE,

contact\_phone VARCHAR(20),

registration\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE products (

product\_id INT PRIMARY KEY AUTO\_INCREMENT,

product\_name VARCHAR(255) NOT NULL,

vendor\_id INT,

category VARCHAR(100),

price DECIMAL(10,2) NOT NULL CHECK (price >= 0),

stock\_quantity INT DEFAULT 0 CHECK (stock\_quantity >= 0),

FOREIGN KEY (vendor\_id) REFERENCES vendors(vendor\_id) ON DELETE CASCADE

);

CREATE TABLE orders (

order\_id INT PRIMARY KEY AUTO\_INCREMENT,

customer\_id INT,

order\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

status ENUM('Pending', 'Completed', 'Cancelled', 'Returned') DEFAULT 'Pending',

total\_amount DECIMAL(10,2) NOT NULL CHECK (total\_amount >= 0),

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id) ON DELETE SET NULL

);

CREATE TABLE order\_details (

order\_detail\_id INT PRIMARY KEY AUTO\_INCREMENT,

order\_id INT,

product\_id INT,

quantity INT NOT NULL CHECK (quantity > 0),

price DECIMAL(10,2) NOT NULL CHECK (price >= 0),

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE CASCADE,

FOREIGN KEY (product\_id) REFERENCES products(product\_id) ON DELETE CASCADE

);

CREATE TABLE payments (

payment\_id INT PRIMARY KEY AUTO\_INCREMENT,

order\_id INT UNIQUE,

payment\_method ENUM('Credit Card', 'PayPal', 'Bank Transfer', 'COD'),

payment\_status ENUM('Success', 'Failed', 'Refunded'),

payment\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE CASCADE

);

CREATE TABLE reviews (

review\_id INT PRIMARY KEY AUTO\_INCREMENT,

customer\_id INT,

product\_id INT,

rating INT CHECK (rating BETWEEN 1 AND 5),

review\_text TEXT,

review\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id) ON DELETE CASCADE,

FOREIGN KEY (product\_id) REFERENCES products(product\_id) ON DELETE CASCADE

);

CREATE TABLE session\_logs (

session\_id INT PRIMARY KEY AUTO\_INCREMENT,

customer\_id INT,

session\_start TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

session\_end TIMESTAMP NULL,

time\_spent INT CHECK (time\_spent >= 0),

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id) ON DELETE CASCADE

);

CREATE TABLE recommendations (

recommendation\_id INT PRIMARY KEY AUTO\_INCREMENT,

customer\_id INT,

product\_id INT,

view\_count INT DEFAULT 0 CHECK (view\_count >= 0),

last\_viewed\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id) ON DELETE CASCADE,

FOREIGN KEY (product\_id) REFERENCES products(product\_id) ON DELETE CASCADE

);

CREATE INDEX idx\_orders\_customer ON orders(customer\_id);

CREATE INDEX idx\_products\_vendor ON products(vendor\_id);

CREATE INDEX idx\_order\_details\_order ON order\_details(order\_id);

CREATE INDEX idx\_order\_details\_product ON order\_details(product\_id);

CREATE INDEX idx\_payments\_order ON payments(order\_id);

CREATE INDEX idx\_reviews\_product ON reviews(product\_id);

CREATE INDEX idx\_session\_logs\_customer ON session\_logs(customer\_id);

CREATE INDEX idx\_recommendations\_customer ON recommendations(customer\_id);

DELIMITER //

CREATE TRIGGER validate\_review\_rating

BEFORE INSERT ON reviews

FOR EACH ROW

BEGIN

IF NEW.rating < 1 OR NEW.rating > 5 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Rating must be between 1 and 5';

END IF;

END;

//

DELIMITER ;