```
!pip install faker
```

```
Requirement already satisfied: faker in /usr/local/lib/python3.12/dist-packages (38.0.0)
Requirement already satisfied: tzdata in /usr/local/lib/python3.12/dist-packages (from faker)
```

```
import sqlite3
import random
from faker import Faker
import json
```

```
# Initialize Faker
fake = Faker()
```

```
# ---------------------------
# 1. CONNECT TO SQLITE DATABASE
# ---------------------------
conn = sqlite3.connect("online_store.db")
cursor = conn.cursor()
```

```
# ---------------------------
# 2. CREATE TABLES
# ---------------------------

# Customers table
cursor.execute("""
CREATE TABLE IF NOT EXISTS Customers (
    customer_id INTEGER PRIMARY KEY,
    name TEXT NOT NULL,
    age INTEGER CHECK(age >= 0),
    gender TEXT CHECK(gender IN ('Male','Female','Other')),
    loyalty_level TEXT CHECK(loyalty_level IN ('Bronze','Silver','Gold','Platinum'))
);
""")
```

```
<sqlite3.Cursor at 0x7d978b3090c0>
```

```
# Products table
cursor.execute("""
CREATE TABLE IF NOT EXISTS Products (
    product_id INTEGER PRIMARY KEY,
    name TEXT NOT NULL,
    category TEXT CHECK(category IN ('Electronics','Clothing','Home','Sports','Books','Beauty
    price REAL CHECK(price >= 0),
    stock_quantity INTEGER CHECK(stock_quantity >= 0)
);
""")
```

```
<sqlite3.Cursor at 0x7d978b3090c0>
```

```
# Orders table
cursor.execute("""
CREATE TABLE IF NOT EXISTS Orders (
    order_id INTEGER PRIMARY KEY,
    customer_id INTEGER,
    order_date TEXT NOT NULL,
    products_ordered TEXT,
    total_amount REAL CHECK(total_amount >= 0),
    payment_method TEXT CHECK(payment_method IN ('Credit Card','PayPal','Bank Transfer','Cas
```

```
    status TEXT CHECK(status IN ('Pending','Shipped','Delivered','Cancelled')),
    FOREIGN KEY(customer_id) REFERENCES Customers(customer_id)
);
""")
```

```
<sqlite3.Cursor at 0x7d978b3090c0>
```

```
# ---------------------------
# 3. GENERATE SYNTHETIC DATA
# ---------------------------
NUM_CUSTOMERS = 1000
NUM_PRODUCTS = 100
NUM_ORDERS = 1500

# Fixed options
genders = ["Male", "Female", "Other"]
loyalty_levels = ["Bronze", "Silver", "Gold", "Platinum"]
categories = ["Electronics", "Clothing", "Home", "Sports", "Books", "Beauty"]
payment_methods = ["Credit Card", "PayPal", "Bank Transfer", "Cash"]
order_status = ["Pending", "Shipped", "Delivered", "Cancelled"]
```

```
# --- Customers ---
customers = []
for i in range(NUM_CUSTOMERS):
    customers.append((
        i + 1,
        fake.name(),
        random.randint(18, 70),
        random.choice(genders),
        random.choice(loyalty_levels)
    ))

cursor.executemany("""
INSERT INTO Customers (customer_id, name, age, gender, loyalty_level)
VALUES (?, ?, ?, ?, ?)
""", customers)
conn.commit()
```

```
# --- Products ---
products = []
for i in range(NUM_PRODUCTS):
    products.append((
        i + 1,
        fake.word().capitalize(),
        random.choice(categories),
        round(random.uniform(5, 500), 2),  # price
        random.randint(10, 500)            # stock quantity
    ))

cursor.executemany("""
INSERT INTO Products (product_id, name, category, price, stock_quantity)
VALUES (?, ?, ?, ?, ?)
""", products)
conn.commit()
```

```
# --- Orders ---
orders = []
for i in range(NUM_ORDERS):
    customer_id = random.randint(1, NUM_CUSTOMERS)
    num_items = random.randint(1, 5)
    order_products = []
    total_amount = 0
```

```
        for _ in range(num_items):
            product_id = random.randint(1, NUM_PRODUCTS)
            quantity = random.randint(1, 3)
            price = next(p[3] for p in products if p[0] == product_id)
            total_amount += price * quantity
            order_products.append({"product_id": product_id, "quantity": quantity})

        orders.append((
            i + 1,
            customer_id,
            str(fake.date_between(start_date="-1y", end_date="today")),
            json.dumps(order_products),  # store products and quantity as JSON
            round(total_amount, 2),
            random.choice(payment_methods),
            random.choice(order_status)
        ))
```

```
cursor.executemany("""
INSERT INTO Orders (order_id, customer_id, order_date, products_ordered, total_amount, payme
VALUES (?, ?, ?, ?, ?, ?, ?)
""", orders)
conn.commit()
```

```
conn.close()
print("\n 'online_store.db' created successfully")
```

```
 'online_store.db' created successfully
```