```
import csv
import sqlite3
from collections import defaultdict
from pathlib import Path

ROOT = Path(__file__).resolve().parents[1]
DATA_DIR = ROOT / 'data'
DB_PATH = ROOT / 'shipment_database.db'


def get_or_create_product(cur, name):
    cur.execute('SELECT id FROM product WHERE name = ?', (name,))
    row = cur.fetchone()
    if row:
        return row[0]
    cur.execute('INSERT INTO product (name) VALUES (?)', (name,))
    return cur.lastrowid


def process_data0(cur):
    path = DATA_DIR / 'shipping_data_0.csv'
    inserted = 0
    with path.open(newline='', encoding='utf-8') as fh:
        reader = csv.DictReader(fh)
        for r in reader:
            product = r['product'].strip()
            qty = int(r['product_quantity'])
            origin = r['origin_warehouse'].strip()
            dest = r['destination_store'].strip()

            pid = get_or_create_product(cur, product)
            cur.execute('INSERT INTO shipment (product_id, quantity, origin, destination) VALUES (?,
?,?,?)',
                        (pid, qty, origin, dest))
            inserted += 1
    return inserted


def process_data1_and_2(cur):
    # Load shipment identifier -> origin,destination from data_2
    path2 = DATA_DIR / 'shipping_data_2.csv'
    shipments_meta = {}
    with path2.open(newline='', encoding='utf-8') as fh:
        reader = csv.DictReader(fh)
        for r in reader:
            sid = r['shipment_identifier'].strip()
            shipments_meta[sid] = (r['origin_warehouse'].strip(), r['destination_store'].strip())

    # Aggregate products per shipment id from data_1
    path1 = DATA_DIR / 'shipping_data_1.csv'
    shipments_products = defaultdict(lambda: defaultdict(int))
    with path1.open(newline='', encoding='utf-8') as fh:
        reader = csv.DictReader(fh)
        for r in reader:
            sid = r['shipment_identifier'].strip()
            product = r['product'].strip()
            # count each row as one quantity of that product within the shipment
            shipments_products[sid][product] += 1

    # Insert into DB: for each shipment, each product becomes a shipment row with counted quantity
    inserted = 0
    for sid, products in shipments_products.items():
        if sid not in shipments_meta:
            # skip if we don't have origin/destination data
            continue
        origin, dest = shipments_meta[sid]
        for product, qty in products.items():
            pid = get_or_create_product(cur, product)
            cur.execute('INSERT INTO shipment (product_id, quantity, origin, destination) VALUES (?,
?,?,?)',
                        (pid, qty, origin, dest))
            inserted += 1

    return inserted


def main():
    conn = sqlite3.connect(str(DB_PATH))
    cur = conn.cursor()

    print('Processing shipping_data_0.csv...')
    inserted0 = process_data0(cur)
    print(f'Inserted {inserted0} rows from shipping_data_0.csv')

    print('Processing shipping_data_1.csv and shipping_data_2.csv...')
    inserted1 = process_data1_and_2(cur)
    print(f'Inserted {inserted1} product-rows from combined shipping_data_1/2')

    conn.commit()

    # Report final counts
    cur.execute('SELECT COUNT(*) FROM product')
    products_count = cur.fetchone()[0]
    cur.execute('SELECT COUNT(*) FROM shipment')
    shipments_count = cur.fetchone()[0]
    print(f'Total products: {products_count}, total shipments: {shipments_count}')

    conn.close()


if __name__ == '__main__':
    main()
```