```
import csv
import sqlite3

class DatabaseConnector:
    """
    Manages a connection to a SQLite database.
    """
    def __init__(self, database_file):
        self.connection = sqlite3.connect(database_file)
        self.cursor = self.connection.cursor()
        self.create_tables()

    def create_tables(self):
        """
        Create the necessary tables if they don't already exist.
        """
        self.cursor.execute('''
        CREATE TABLE IF NOT EXISTS product (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            name TEXT UNIQUE
        )
        ''')
        self.cursor.execute('''
        CREATE TABLE IF NOT EXISTS shipment (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            product_id INTEGER,
            quantity INTEGER,
            origin TEXT,
            destination TEXT,
            FOREIGN KEY(product_id) REFERENCES product(id)
        )
        ''')

    def populate(self, spreadsheet_folder):
        """
        Populate the database with data imported from each spreadsheet.
        """
        with open(f"{spreadsheet_folder}/shipping_data_0.csv", "r") as spreadsheet_file_0, \
             open(f"{spreadsheet_folder}/shipping_data_1.csv", "r") as spreadsheet_file_1, \
             open(f"{spreadsheet_folder}/shipping_data_2.csv", "r") as spreadsheet_file_2:

            csv_reader_0 = csv.reader(spreadsheet_file_0)
            csv_reader_1 = csv.reader(spreadsheet_file_1)
            csv_reader_2 = csv.reader(spreadsheet_file_2)

            self.populate_first_shipping_data(csv_reader_0)
            self.populate_second_shipping_data(csv_reader_1, csv_reader_2)

    def populate_first_shipping_data(self, csv_reader_0):
        """
        Populate the database with data imported from the first spreadsheet.
        """
        for row_index, row in enumerate(csv_reader_0):
            if row_index == 0:
                continue
        origin = row[0]
        destination = row[1]
        product_name = row[2]
        product_quantity = int(row[4])

        self.insert_product_if_it_does_not_already_exist(product_name)
        self.insert_shipment(product_name, product_quantity, origin, destination)

        print(f"Inserted product {row_index} from shipping_data_0")

def populate_second_shipping_data(self, csv_reader_1, csv_reader_2):
    """
    Populate the database with data imported from the second and third spreadsheets.
    """
    shipment_info = {}
    for row_index, row in enumerate(csv_reader_2):
        if row_index == 0:
            continue

        shipment_identifier = row[0]
        origin = row[1]
        destination = row[2]

        shipment_info[shipment_identifier] = {
            "origin": origin,
            "destination": destination,
            "products": {}
        }

    for row_index, row in enumerate(csv_reader_1):
        if row_index == 0:
            continue

        shipment_identifier = row[0]
        product_name = row[1]

        products = shipment_info[shipment_identifier]["products"]
        if products.get(product_name) is None:
            products[product_name] = 1
        else:
            products[product_name] += 1

    count = 0
    for shipment_identifier, shipment in shipment_info.items():
        origin = shipment["origin"]
        destination = shipment["destination"]
        for product_name, product_quantity in shipment["products"].items():
            self.insert_product_if_it_does_not_already_exist(product_name)
            self.insert_shipment(product_name, product_quantity, origin, destination)

            print(f"Inserted product {count} from shipping_data_1")
            count += 1

def insert_product_if_it_does_not_already_exist(self, product_name):
    """
    Insert a new product into the database.
```

```
        If a product already exists in the database with the given name, ignore it.
        """
        query = "INSERT OR IGNORE INTO product (name) VALUES (?);"
        self.cursor.execute(query, (product_name,))
        self.connection.commit()

    def insert_shipment(self, product_name, product_quantity, origin, destination):
        """
        Insert a new shipment into the database.
        """
        query = "SELECT id FROM product WHERE name = ?;"
        self.cursor.execute(query, (product_name,))
        product_id = self.cursor.fetchone()[0]

        query = "INSERT INTO shipment (product_id, quantity, origin, destination) VALUES (?, ?, ?, ?);"
        self.cursor.execute(query, (product_id, product_quantity, origin, destination))
        self.connection.commit()

    def close(self):
        self.connection.close()

if __name__ == '__main__':
    database_connector = DatabaseConnector("shipment_database.db")
    database_connector.populate("./data")
    database_connector.close()
```