```python
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()

    def populate(self, spreadsheet_folder):
        """
        Populate the database with data imported from each spreadsheet.
        """
    # open the spreadsheets
        with open(f"{spreadsheet_folder}/shipping_data_0.csv", "r") as spreadsheet_file_0:
            with open(f"{spreadsheet_folder}/shipping_data_1.csv", "r") as spreadsheet_file_1:
                with open(f"{spreadsheet_folder}/shipping_data_2.csv", "r") as spreadsheet_file_2:
    # prepare the csv readers
                    csv_reader_0 = csv.reader(spreadsheet_file_0)
                    csv_reader_1 = csv.reader(spreadsheet_file_1)
                    csv_reader_2 = csv.reader(spreadsheet_file_2)
    # populate first spreadsheet
                    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:
                product_name = row[2]
                product_quantity = row[4]
                origin = row[0]
                destination = row[1]
        # insert the data into the database
                self.insert_product_if_it_does_not_already_exist(product_name)
                self.insert_shipment(product_name, product_quantity, origin, destination)
        # give an indication of progress
                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.
        """
    # collect shipment info
        shipment_info = {}
        for row_index, row in enumerate(csv_reader_2):
        # ignore the header row
            if row_index > 0:
        # extract each required field
                shipment_identifier = row[0]
                origin = row[1]
                destination = row[2]
        # store them for later use
                shipment_info[shipment_identifier] = {
                "origin": origin, "destination": destination, "products": {}
                }
        # read in product information
        for row_index, row in enumerate(csv_reader_1):
        # ignore the header row
            if row_index > 0:
        # extract each required field
                shipment_identifier = row[0]
                product_name = row[1]
        # populate intermediary data structure
                products = shipment_info[shipment_identifier]["products"]
                if products.get(product_name, None) is None:
                    products[product_name] = 1
                else:
                    products[product_name] += 1
        # insert the data into the database
        count = 0
        for shipment_identifier, shipment in shipment_info.items():
        # collect origin and destination
            origin = shipment_info[shipment_identifier]["origin"]
            destination = shipment_info[shipment_identifier]["destination"]
            for product_name, product_quantity in shipment["products"].items():
        # iterate through products and insert into database
                self.insert_product_if_it_does_not_already_exist(product_name)
                self.insert_shipment(product_name, product_quantity, origin, destination)
        # give an indication of progress
                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 (?);
```

```python
        """
        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.
        """
        # collect the product id
        query = """
        SELECT id
        FROM product
        WHERE product.name = ?;
        """
        self.cursor.execute(query, (product_name,))
        product_id = self.cursor.fetchone()[0]
        # insert the shipment
        query = """
        INSERT OR IGNORE 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()
```