```
from DBHelper import DBHelper
from helper_functions import *
from Product import *
from Customer import *

class Receipt:
    def __init__(self):
        self.db = DBHelper()

    def __updateReceiptTotal (self, receiptNo):
        sql = ("UPDATE receipt SET "
               "total_receipt = new_total_receipt"
               " FROM (SELECT rli.receipt_no , SUM(rli.amount_paid_here) As new_total_receipt From receipt_line_item rli GROUP BY rli.receipt_no) rli "
               " Where receipt.receipt_no = rli.receipt_no "
               "AND receipt.receipt_no = '{}' ".format(receiptNo))
        self.db.execute (sql)

    def __updateReceiptAmountUnpaid (self, receiptNo):
        sql = ("UPDATE receipt_line_item SET "
               " amount_unpaid = new_amount_unpaid"
               " FROM (SELECT rli.receipt_no , SUM(rli.amount_paid_here) As new_total_receipt From receipt_line_item rli GROUP BY rli.receipt_no) rli "
               " Where receipt.receipt_no = rli.receipt_no "
               "AND receipt.receipt_no = '{}' ".format(receiptNo))
        self.db.execute (sql)

    def __updateLineItem (self, receiptNo, receiptLineItemList):
        self.db.execute ("DELETE FROM receipt_line_item WHERE receipt_no = '{}' ".format(receiptNo))
        for lineItem in receiptLineItemList:
            self.db.execute ("INSERT INTO receipt_line_item (receipt_no, invoice_no, amount_paid_here) VALUES ('{}' ,'{}','{}')".format(receiptNo,lineItem["Invoice No"],lineItem["Amount Paid Here"]))
        self.__updateReceiptTotal(receiptNo)



def create(self, receiptNo, receiptDate, customerCode, paymenMethod, paymenReference, remark, receiptLineItemList):
    data, columns = self.db.fetch ("SELECT * FROM receipt WHERE receipt_no = '{}' ".format(receiptNo))
    if len(data) > 0:
        return {'Is Error': True, 'Error Message': "Receipt No '{}' already exists. Cannot Create. ".format(receiptNo)}
    else:
        self.db.execute ("INSERT INTO receipt (receipt_no, receipt_date, customer_code, payment_method, payment_reference, remark) VALUES ('{}' ,{} ,'{}' ,'{}' ,'{}' ,'{}')".format(receiptNo,receiptDate,customerCode,paymenMethod,paymenReference,remark))
        self.__updateLineItem(receiptNo, receiptLineItemList)

    return {'Is Error': False, 'Error Message': ""}

def read(self, receiptNo):
    data, columns = self.db.fetch ("SELECT receipt_no, receipt_date, customer_code, payment_method, payment_reference, remark FROM receipt WHERE receipt_no = '{}' ".format(receiptNo))
    if len(data) > 0:
        retReceipt = row_as_dict(data, columns)
    else:
        return ({'Is Error': True, 'Error Message': "Receipt No '{}' not found. Cannot Read.".format(receiptNo)},{})

    return ({'Is Error': False, 'Error Message': ""},retReceipt)
```

```
  def update(self, receiptNo, newReceiptDate, newCustomerCode, newPaymenMethod, newPaymenReference, newRemark ,newReceiptLineItemList):
    # Finds the invoice number in invoices object and then changes the values to the new ones.
    # Returns dictionary {‘Is Error’: ___, ‘Error Message’: _____}.
    data, columns = self.db.fetch ("SELECT * FROM receipt WHERE receipt_no = '{}' ".format(receiptNo))
    if len(data) > 0:
      self.db.execute ("UPDATE receipt SET receipt_date = {}, customer_code = '{}', payment_method = '{}', payment_reference = '{}', remark= '{}' WHERE receipt_no = '{}' ".format(newReceiptDate, newCustomerCode, newPaymenMethod, newPaymenReference, newRemark,receiptNo))
      self.__updateLineItem(receiptNo, newReceiptLineItemList)
    else:
      return {'Is Error': True, 'Error Message': "Receipt No '{}' not found. Cannot Update.".format(receiptNo)}

    return {'Is Error': False, 'Error Message': ""}

  def delete(self, receiptNo):
    # Finds the invoice number invoices object and removes it from the dictionary.
    # Returns dictionary {‘Is Error’: ___, ‘Error Message’: _____}.
    data, columns = self.db.fetch ("SELECT * FROM receipt WHERE receipt_no = '{}' ".format(receiptNo))
    if len(data) > 0:
      self.db.execute ("DELETE FROM receipt WHERE receipt_no = '{}' ".format(receiptNo))
      self.db.execute ("DELETE FROM receipt_line_item WHERE receipt_no = '{}' ".format(receiptNo))
    else:
      return {'Is Error': True, 'Error Message': "Receipt No '{}' not found. Cannot Delete".format(receiptNo)}
    return {'Is Error': False, 'Error Message': ""}

  def dump(self):
    # Will dump all invoice data by returning 1 dictionary as output.
    data, columns = db.fetch ('SELECT r.receipt_no as "Receipt No", r.receipt_date as "Receipt Date", r.customer_code as "Customer Code", r.payment_method as "Payment Method", r.payment_reference as "Payment Reference", r.remark as "Remark" FROM receipt r JOIN customer c ON r.customer_code = c.customer_code')

    return row_as_dict(data, columns)

  def update_receipt_line(self, receiptNo, invoiceNo, newAmountPaid):
    data, columns = self.db.fetch ("SELECT * FROM receipt_line_item WHERE receipt_no = '{}' AND invoice_no = '{}' ".format(receiptNo, invoiceNo))
    if len(data) > 0:
      self.db.execute ("UPDATE receipt_line_item SET amount_paid_here = {} WHERE receipt_no = '{}' AND invoice_no = '{}' ".format(newAmountPaid, receiptNo, invoiceNo))
      self.__updateReceiptTotal(receiptNo)
    else:
      return {'Is Error': True, 'Error Message': "Invoice Code '{}' not found in Receipt No '{}'. Cannot Update.".format(invoiceNo, receiptNo)}

    return {'Is Error': False, 'Error Message': ""}

  def delete_receipt_line(self, receiptNo, invoiceNo):
    data, columns = self.db.fetch ("SELECT * FROM receipt_line_item WHERE receipt_no = '{}' AND invoice_no = '{}' ".format(receiptNo, invoiceNo))
    if len(data) > 0:
      self.db.execute ("DELETE FROM receipt_line_item WHERE receipt_no = '{}' AND invoice_no = '{}' ".format(receiptNo, invoiceNo))
      self.__updateReceiptTotal(receiptNo)

    else:
      return {'Is Error': True, 'Error Message': "Invoice Code '{}' not found in Receipt No '{}'. Cannot Delete.".format(invoiceNo, receiptNo)}

    return {'Is Error': False, 'Error Message': ""}
```