CS 509 Python Assignment
Fall 2023

OVERVIEW
You are implementing the start of a sales system. You will access one data table with product information, one data table with customer information, and one data table with sales records. I will provide the tables, you will import them into your current system, determine the columns and data types, and build your program to be compatible with them.

The program must feature a menu system like what I demonstrated in class. It will look like:

1. First option
2. Second option
3. Third option
Q. Quit

However, replace the 'options' with more specific actions. If you need more menu items, you can make your menu bigger. But please follow this format.

The program allows a sales user to configure an order that he just received after speaking to a customer on the phone. The program begins by printing the menu. Then the sales user should be able to select inventory, which accesses the product data table and displays all entries. The sales user can also select the customer table to reference customer information. Finally, the sales user should be able to access the sales records, input a customer ID, a product ID, and a quantity, and the table will update with the new sales record. However, the last step is more complicated than it seems because it involves querying the product table to get pricing, and querying the customer table to get the encrypted account information.

That means we'll have to store the encryption key within the program which is not very secure, but remember this is just a beginning design. More advanced features will keep the encryption key elsewhere and avoid network communications.

TECHNICAL DETAILS
Use the tutorial and recommended libraries to establish a connection to your database. The program will access two data tables for reading, and a third data table for writing. The user will navigate program options through a simple menu system that offers choices of what to do that should include accessing the different data tables or writing to a table. Connections should use try/except blocks to test for errors and print any error conditions. Finally, you will document each significant component of your program to show that you understand what is happening and why that segment of code exists.

COMMENTS
When documenting your software, you are adding comments and descriptions to the code itself. Python uses # to start a line of comments and you can add short descriptions to one line, or full paragraphs to the beginning of a function.

If you write short and minimal statements like "write to table" or "print to screen", that isn't worth much. Instead, comments should be specific about what is happening and WHY it is happening. Imagine that someone is taking over your project and all they see is 'get input'. How will that person know what the input is for? You have to explain why you are doing what you are doing.

Comments should be more like:

```
# Establish connection with server to access sales table
# Test for error during connection
# Receive user input for product info  (notice, it isn't just 'get input', but it explains what the input is)
# Copy user input to query string for Product Table (again, specify the reason for transferring data)
# Display current records showing order history
# Create dictionary to store sales records  (notice I didn't just say 'records', I said 'sales records')
```

REQUIREMENTS
Use the basic structure provided at the end of the tutorial to start your program.
Assume that the database is on 'localhost', which is IP address 127.0.0.1. Save this address in a constant:

```
HOST = 127.0.0.1
```

Please also include:

```
HOST2 = 10.0.0.87
```

because that is what I will use for testing. Now when you connect to the database, you can just use HOST instead of an IP address.

Use the data tables that I will post on Blackboard.

The user menu should allow a user to see the full product table, the full customer info table (as proof that the customer info is encrypted, not because it's a good idea), and all sales records. Viewing sales is necessary to confirm that an order was placed correctly. It should also have the option to place an order, and a quit function. You shouldn't need more features than that, but you can add more if it helps you make a simple design.

The customer account information will be encrypted. The key for the encrypted column is "secretKey".

TIPS:
Use the tutorial for quick code design

Ask me questions if you get stuck

Updating the sales table may sound intimidating, but what you are really doing is:

1. get user input and store it in local variables
2. make specific queries to the product and customer tables using the user input values
3. store the price and customer account info in two more local variables
4. make calculations to get the total price
5. write the customer info, customer account, product info, quantity, and total cost to the sales table.
6. The customer account should always be encrypted in tables with aes_encrypt() and 'secretKey', and decrypted with aes_decrypt() when storing a local copy of the string in your program.

You will have to figure out that query results will return as a list of records.
Use list and array indexing to access the data you need.

POINTS
5 points for the menu
15 points for querying all 3 tables and handling connection errors.
15 points for correctly writing an order to the sales table
15 points for clear and descriptive comments throughout your code

-9,999,999 points if you copy from someone else (except the tutorials, those are acceptable and expected)