W05 Scenario:

Karl, the owner of the bowling alley, has decided that he wants to start emailing "lunch & bowl" coupons to the league members between seasons. As you go to look at the bowler table to add this new column, you have a feeling that not everyone should be able to view the personally identifiable information in the bowling table. You decide to take some steps to keep lower-level employees from seeing the contact information of each bowler.

W05 Scenario Submission Guidelines:

Be sure you submit all elements labeled by the bolded word, **SHOW**.

1. Create a new email address column for bowlers:

    a. Verify that you have a recent backup of your bowling database or take a new one.
    b. Write an SQL ALTER TABLE statement to add an e-mail address column **alter** statement to add an email address column of length 50 to the "bowlers" table.
    c. Write an **UPDATE statement** (not INSERT) to simulate email addresses for each customer using their first name plus last name @gmail.com (for example, row 1 should be barbarafournier@gmail.com). You will use concatenation (this tutorial shows how) to do this.
    NOTE: Save your commands from steps 1-3, so you can run them again on the class server in step 4!

    SHOW 1: The contents of the bowlers table with the new email addresses you added. (*If you already encrypted the value in a future step*, you can simply show the commands you used in this step along with the encrypted contents.)

2. Under the chapter section, "Data Security and Views," we learn how we can keep users from seeing certain columns that they should not view:

    a. Create a new login/user for a junior employee. Name it bob_the_scorekeeper and use a password of your choice.
    b. Create a view that displays all other information in the "bowlers" table **except** for street address, phone number and email address for each bowler. The view must be created using the same schema name as the bowling table. Grant access to this view to the new user created in step 1.
    NOTE: Pay close attention to what access the new user should have based on the instructions. Also. generate and save your scripts from steps 1-3 so you can run them on the class server in step 4!

SHOW 2: That the view works by logging in to the database as the new user and selecting from the view, and show what tables and views they are able to see. Provide a screenshot that shows that the personal information (other than first and last name) is not displayed in the results.

3. Your new programmer needs a bit more access than Bob the scorekeeper. Create her a login/user called "carol_the_programmer" with a password of your choice. Carol needs to be able to see all the columns, but not the actual data. To solve this, you should:

    a. Implement column level encryption (as shown at that link or in the stepping stones) on street address, phone number and email address columns.
    b. Grant Carol the ability to select and insert into the "bowlers" table.
    NOTE: Pay close attention to what access the new user should have based on the instructions. Generate and save your scripts from steps 1-3 so you can run them on the class server in step 4!

    SHOW 3: A connection logged in as Carol and prove that she only sees encrypted values in the address, phone number, email columns, and show what tables and views they are able to see.

4. Let's test your level of security with a classmate's help!

    a. Repeat your saved scripts from steps 1-3 in your bowling database on the cloud class server. You should have copied this database last week. **OR**, you can once again backup your bowling database and restore it to the class server (using the stepping stone instructions from last week).
    b. Create the same logins from steps 1-3 on the class cloud server, but this time add your last name after Bob and Carol (such as carol_jones_the_programmer).
    c. Give the names of these logins and their passwords to a running buddy. You, in turn, should receive logins from another student. **HINT:** If you are having trouble arranging this, make or respond to a post in the weekly Teams channel. It is alright if you need to login to someone's database who already has had someone else look as well.
    d. Login to your running buddy's (or other classmate's) database and scout out what you can see as Bob and as Carol. The sensitive data should be encrypted and/or protected with views. Are you able to see more than you should with the account? Are you able to do anything you are not intended to do, such as delete a row or make a new table?
    e. Report your findings to your classmate.

    SHOW 4:
        I. Your connections to your classmate's database (one as Bob, one as Carol).

II. What you were able to see and do when logged in as each account.
NOTE: You are not required to submit anything regarding what they found in your database (only what you saw in theirs).