Unit 2 Graded Exercise 3

The following questions come from the Check your understanding examples of each section of Chapter 5 in your textbook.

After you are finished, please submit a Microsoft Word file that contains screenshots of the SQL queries, resulting output, and contains a comment in the queries showing your name. Your document should be named **U2\_GradedExercise3\_Lastname.docx**.

(5-2) Question 1:

1. Turn off the Autocommit option:

■ If you are using the Home Page environment, clear the Autocommit checkbox.

■ If you are using the SQL Command Line environment, issue the SQL\*Plus command:

*set autocommit off*;

2. Add a new row to a copy of the *departments* table:

insert into *sec0502\_departments*

values('IT', 'INFORMATION TECHNOLOGY');

3. Save this change by issuing a commit command:

*commit*;

4. Add another new row to the copy of the departments table:

insert into *sec0502\_departments*

values('LAW', 'LEGAL DEPARTMENT');

5. Undo this change by issuing a rollback command:

*rollback;*

6. List all the rows in the copy of the departments table:

*select \**

from *sec0502\_departments;*

7. Confirm that the table has the row for the *IT* department and that it does not have the row for the LAW department.

(5-3) Question 2:

Use the tables for checking accounts and savings accounts in this section. Use a transaction to delete the checking account for Bob Wilkins and transfer all his money to a new savings account. The ID of the new savings account is 5678.

To do this you can follow these steps:

1. Turn off the autocommit option.

2. Delete the checking account.

3. Create the savings account.

4. Verify that your changes are okay.

5. Commit your changes.

(5-3) Question 3:

Again, use the tables for this section. Use a transaction to transfer $20,000 for Fred Boyd from his savings account to his checking account.

To do this you can follow these steps:

1. Update the amount of money in his savings account.

2. Update the amount of money in his checking account.

3. Verify that your changes are okay. In particular, verify the amount of money left in his savings account. If this is a negative number, then the change is not okay.

4. Roll back the changes.

(5-5) Question 4:

Change the following SQL code to create your own example that shows you can modify the data in a table by making changes to a view based on that table.

In this example, the view you create will include all the columns and all the rows of the underlying table. This is the easiest and most straightforward case. It is also the way that this feature is used most often.

1. List all the columns and rows of the foods table. This shows what data is in the table before you make any changes to it.

*select* \*

from *sec0505\_foods*;

2. Create a view of the foods table. Include all the columns and rows in the view.

create or replace view *sec0505\_foods\_view* as

*select* \*

from *sec0505\_foods*;

3. Show that you can use an insert statement with this view.

insert into *sec0505\_foods\_view*

values('ABC', 'DEF', 51, 'BLUEBERRY PIE', 2.99, null);

4. Show that you can use an update statement with this view.

update *sec0505\_foods\_view*

set *menu\_item* = 20,

*description* = 'CARROTS'

where *description* = 'BROCCOLI';

5. Show that you can use a delete statement with this view.

delete from *sec0505\_foods\_view*

where *description* = 'DESSERT';

6. List all the columns and rows of the underlying table. Show that all the changes you made to the view actually affected the data in the underlying table.

*select \**

from *sec0505\_foods*;

(5-11) Question 5:

List the names of all the tables you own.

(5-12) Question 6:

List the names of all the views you own.

(5-14) Question 7:

Use the Data Dictionary to find the names of all the columns of the *1\_employees* table.

(5-15) Question 8:

Use the Data Dictionary to find the primary key of the *1\_employees* table.