CSCI 5020 Assignment 6

Before you start the exercises…

Before you start the exercises, you need to install Oracle Database Express Edition and Oracle SQL Developer. You also need to download and install the source files for this book, and you need to create the users and tables for this book. The procedures for doing all of these tasks are provided in appendix A and the handout posted in D2L.

Chapter 2 How to use Oracle SQL Developer and other tools

In these exercises, you'll use SQL Developer to review the tables in the AP schema that's used throughout this book. In addition, you'll use SQL developer to enter SQL statements and run them against these tables.

Make sure Oracle Database is running

1. Use the procedure in figure 2-1 to start the database service for Oracle Database. If it is already running, you'll get a message that confirms that. Otherwise, the database service will be started. Either way, you can close the DOS window that's opened.

Use SQL Developer to review the Accounts Payable database

2. Start Oracle SQL Developer. If you created a menu or desktop shortcut when you installed SQL Developer, you can use that shortcut now. Otherwise, you can use the Windows Explorer to find and double-click on the sqldeveloper.exe file.
3. Create the AP, EX, and OM connections as described in figure 2-4. When you're done, the Connections window should display three connections: AP, EX, and OM.
4. In the Connections window, click on the AP connection to expand it. When you're prompted for a password, enter "ap". That will expand the connection so you can see all of the database objects in the AP schema.
5. Use the techniques in figures 2-5 and 2-6 to navigate through the database objects and view the column definitions for at least the Vendors and Invoices tables.
6. Use the technique in figure 2-6 to view the data for the Vendors and Invoices tables.

Use SQL Developer to enter and run SQL statements

7. Use the technique in figure 2-8 to open a SQL Worksheet window for the AP connection. Then, enter and run this SQL statement:

```
SELECT vendor_name FROM vendors
```

8. Use the code completion feature described in figure 2-8 to enhance this SQL statement so it includes an ORDER BY clause and some additional columns like this:

```
SELECT vendor_name, vendor_address1, vendor_city, vendor_state,
    vendor_zip_code
FROM vendors
ORDER BY vendor_name
```

   Then, run the statement.

9. Move the cursor into the ORDER BY clause and press Ctrl+/ to comment out the line. Then, press Ctrl+/ again to uncomment the line.
10. Delete the e at the end of vendor_zip_code and run the statement again. Note that this syntax error is handled as in figure 2-10.
11. Open another Worksheet window, and use the COUNT and SUM snippets as shown in figure 2-9 as you enter this statement:

```
SELECT COUNT(*) AS number_of_invoices,
    SUM(invoice_total) AS grand_invoice_total
FROM invoices
```

    Then, run the statement.
12. Use the Tools→Preferences command to set the default path for scripts as described in figure 2-11. Then, click on the tab for the Worksheet window of exercise 8, click the Save button to save this statement, and note the directory in the Save dialog box. Next, click the Cancel button in the Save dialog box to cancel the command.

Use SQL Developer to open and run scripts

13. Use the technique in figure 2-11 to open the select_vendor_city_state script that's in the c:\murach\oracle_sql\scripts\ch02 directory. Notice that this script contains just one SQL statement. Then, run the statement. Because you didn't specify a connection for this statement, SQL Developer will ask you to select one before it runs the statement.
14. Click on the Open button. Note that the recently used directories including the ch02 directory are shown on the left side of the Open dialog box. Then, click on the ch02 directory to display the files that are stored in this directory. Next, click the Cancel button to close this dialog box.
15. Open the select_vendor_total_due script that's in the ch02 directory. Note that this opens another tab. Next, select the AP connection from the connection list and run this script.
16. Open the select_vendor_information script that's in the ch02 directory. Notice that this script contains two SQL statements that end with semicolons (scroll down if you need to). Then, move the insertion point to the first statement and press F9 to run that statement. Next, move the insertion point to the second statement and press F9 to run that statement. Last, press F5 or click the Run Script button to run both of the statements that are stored in this script. If you scroll through the Script Output window, you will see the results of the two SELECT statements that were run.

Close and restart SQL Developer

17. Continue to experiment on your own. Make sure to leave at least one saved script open. When you're ready to end this session, use the File→Exit command or click on the Close button in the upper right corner of the SQL Developer window.
18. Restart SQL Developer. When it starts, notice that all of the saved scripts that you left open are automatically opened. However, any unsaved scripts that you entered are lost.
19. Run one of the open scripts. Note that you have to select a connection and provide a password for the connection before the script will run.
20. Exit from SQL Developer.

Chapter 3 How to retrieve data from a single table

7. Write a SELECT statement that returns four columns from the Invoices table named Due Date, Invoice Total, 10%, and Plus 10%. These columns should contain this data:

Due Date	The invoice_due_date column
Invoice Total	The invoice_total column
10%	10% of the value of invoice_total
Plus 10%	The value of invoice_total plus 10%

   (For example, if invoice_total is 100, 10% is 10, and Plus 10% is 110.) Next, filter the result set so it returns only those rows with an invoice total that's greater than or equal to 500 and less than or equal to 1000. Then, sort the result set in descending sequence by invoice_due_date.

Chapter 4 How to retrieve data from two or more tables

4. Write a SELECT statement that returns five columns from three tables:

vendor_name	vendor_name from the Vendors table
invoice_date	invoice_date from the Invoices table
invoice_number	invoice_number from the Invoices table
li_sequence	invoice_sequence from the Invoice_Line_Items table
li_amount	line_item_amt from the Invoice_Line_Items table

   Use these aliases for the tables: Ven for the Vendors table, Inv for the Invoices table, and LI for the Invoice_Line_Items table. Also, sort the final result set by vendor_name, invoice_date, invoice_number, and invoice_sequence.

7. Use the UNION operator to generate a result set consisting of two columns from the Vendors table: vendor_name and vendor_state. If the vendor is in California, the vendor_state value should be "CA"; otherwise, the VendorState value should be "Outside CA." Sort the final result set by vendor_name.

Chapter 5 How to code summary queries

3. Write a SELECT statement that returns one row for each vendor that contains three columns:

   The vendor_name column from the Vendors table

   The count of the invoices for each vendor in the Invoices table

   The sum of the invoice_total column for each vendor in the Invoices table

   Sort the result set so the vendor with the most invoices appears first.

7. Write a SELECT statement that answers this question: Which vendors are being paid from more than one account? Return two columns: the vendor name and the total number of accounts that apply to that vendor's invoices. *Hint: Use the DISTINCT keyword to count the account_number column in the Invoice_Line_Items table.*

Chapter 6 How to code subqueries

1. Write a SELECT statement that returns the same result set as this SELECT statement but don't use a join. Instead, use a subquery in a WHERE clause that uses the IN keyword.

```
SELECT DISTINCT vendor_name
FROM vendors JOIN invoices
    ON vendors.vendor_id = invoices.vendor_id
ORDER BY vendor_name
```

5. Write a SELECT statement that returns a single value that represents the sum of the largest unpaid invoices for each vendor (just one for each vendor). Use an inline view that returns MAX(invoice_total) grouped by vendor_id, filtering for invoices with a balance due.

6. Rewrite exercise 6 so it uses subquery factoring.

Chapter 7 How to insert, update, and delete data

To test whether a table has been modified correctly as you do these exercises, you can write and run an appropriate SELECT statement. Or, when you're using Oracle SQL Developer, you can click on a table name in the Connections window and then on the Data tab to display the data for all of the columns in the table. To refresh the data on this tab, click the Refresh button.

1. Write an INSERT statement that adds this row to the Invoices table:

invoice_id	The next id in sequence (find out what this should be)
vendor_id:	32
invoice_number:	AX-014-027
invoice_date:	8/1/2008
invoice_total:	$434.58
payment_total:	$0.00
credit_total:	$0.00
terms_id:	2
invoice_due_date:	8/31/2008
payment_date:	null

2. Write an UPDATE statement that modifies the Vendors table. Change the default account number to 403 for each vendor that has a default account number of 400.

4. Write a DELETE statement that deletes the row that you added to the Invoices table in exercise 1.

5. After you have verified that all of the modifications for the first four exercises have been successful, rollback the changes. Then, verify that they have been rolled back.

Turn in screenshot for Chapter 2 Exercise 8, 11, and 16. Write each statement in separate script files (*.sql) for exercises of other chapters. Zip all the files into a single file and submit it to the Dropbox.