Chapter 8: SQL Functions

Exercise 8.1: Using Scalar Functions

Unless the order is specified, the order of your results may differ.

Connect to the `HR` account.

1. Write a query to display the first name. last name, and salary of all employees in department 30, formatting the salary with commas and a floating dollar sign.

```
FIRST_NAME           LAST_NAME                 SALARY
-------------------- ------------------------- --------
Den                  Raphaely                   $11,000
Alexander            Khoo                        $3,100
Shelli               Baida                       $2,900
Sigal                Tobias                      $2,800
Guy                  Himuro                      $2,600
Karen                Colmenares                  $2,500
```

2. Write a query to display the first name, last name, and date hired of all employees in department 30, formatting the date to be year-month#-day.

```
FIRST_NAME           LAST_NAME                 Date Hired
-------------------- ------------------------- ----------
Den                  Raphaely                  2002-12-07
Alexander            Khoo                      2003-05-18
Shelli               Baida                     2005-12-24
Sigal                Tobias                    2005-07-24
Guy                  Himuro                    2006-11-15
Karen                Colmenares                2007-08-10
```

3. Write a query to display the salary of all employees in department 30. Also show the salary rounded and truncated to thousands.

```
FIRST_NAME           LAST_NAME                 RDSAL    TSAL  SALARY
-------------------- ------------------------- ------- ------- -------
Den                  Raphaely                  11000   11000   11000
Alexander            Khoo                       3000    3000    3100
Shelli               Baida                      3000    2000    2900
Sigal                Tobias                     3000    2000    2800
Guy                  Himuro                     3000    2000    2600
Karen                Colmenares                 3000    2000    2500
```

4. Write a query to display names of all employees in department 30. Their first name should be in lower case; their last name in upper case. Sequence the list in (ascending) first name, last name order.

```
LNAME                UNAME
-------------------- ---------------
alexander            KHOO
den                  RAPHAELY
guy                  HIMURO
karen                COLMENARES
shelli               BAIDA
sigal                TOBIAS
```

5. Write a query to display the initial of the first name followed by a period followed by the last name of all employees in department 30. Sequence the list in alphabetical order of this formatted name.

```
NAME
--------------
A. Khoo
D. Raphaely
G. Himuro
K. Colmenares
S. Baida
S. Tobias
```

6. Write a query to display the street address, followed by the street address stripped of any leading numeric digits, spaces, or dashes (Street Name) for all rows in the `locations` table. Order the list by the Street Name.

```
STREET_ADDRESS                          Street Name
--------------------------------------- ---------------------------------------
8204 Arthur St                          Arthur St
6092 Boxwood St                         Boxwood St
93091 Calle della Testa                 Calle della Testa
2004 Charade Rd                         Charade Rd
9702 Chester Road                       Chester Road
198 Clementi North                      Clementi North
2011 Interiors Blvd                     Interiors Blvd
2014 Jabberwocky Rd                     Jabberwocky Rd
9450 Kamiya-cho                         Kamiya-cho
40-5-12 Laogianggen                     Laogianggen
Magdalen Centre, The Oxford Science Park Magdalen Centre, The Oxford Science Park
Mariano Escobedo 9991                   Mariano Escobedo 9991
Murtenstrasse 921                       Murtenstrasse 921
Pieter Breughelstraat 837               Pieter Breughelstraat 837
Rua Frei Caneca 1360                    Rua Frei Caneca 1360
20 Rue des Corps-Saints                 Rue des Corps-Saints
Schwanthalerstr. 7031                   Schwanthalerstr. 7031
2017 Shinjuku-ku                        Shinjuku-ku
147 Spadina Ave                         Spadina Ave
1297 Via Cola di Rie                    Via Cola di Rie
12-98 Victoria Street                   Victoria Street
1298 Vileparle (E)                      Vileparle (E)
2007 Zagora St                          Zagora St
```

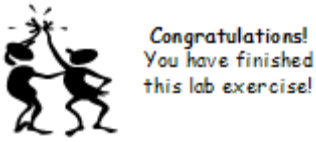
7. Write a query to display the street address, followed by length of the street address (Street Length) for all rows in the `locations` table. Sequence the list in the Street Length order.

```
STREET_ADDRESS                          Street Length
--------------------------------------- -------------
8204 Arthur St                                   14
```

```
2007 Zagora St                                          14
2004 Charade Rd                                         15
9450 Kamiya-cho                                         15
6092 Boxwood St                                         15
147 Spadina Ave                                         15
2017 Shinjuku-ku                                        16
Murtenstrasse 921                                       17
9702 Chester Road                                       17
1298 Vileparle (E)                                      18
198 Clementi North                                      18
2014 Jabberwocky Rd                                     19
40-5-12 Laogianggen                                     19
2011 Interiors Blvd                                     19
1297 Via Cola di Rie                                    20
Schwanthalerstr. 7031                                   21
12-98 Victoria Street                                   21
Rua Frei Caneca 1360                                    21
Mariano Escobedo 9991                                   21
20 Rue des Corps-Saints                                 23
93091 Calle della Testa                                 23
Pieter Breughelstraat 837                               25
Magdalen Centre, The Oxford Science Park                40
```

8. Write a query to display the location ID, the street address, city, and state province of all rows in the `locations` table that contain either the string "RUA" or "RUE" in the street address. Sequence the list in descending sequence on location ID.

```
LOCATION_ID STREET_ADDRESS            CITY          STATE_PROVINCE
----------- ------------------------- ------------- ---------------
       2900 20 Rue des Corps-Saints   Geneva        Geneve
       2800 Rua Frei Caneca 1360      Sao Paulo     Sao Paulo
```



Exercise 8.2: Additional SQL Functions

Connect to the `HR` account.

9. Write a query to display the department ID, first and last names, hire date, and commission percentage of all employees with manager 100 (Steven King) whose hire date is within 2 years of Jan 1, 2007. Sequence the list in department, hire date order. Use a non-standard function to display null commissions as 0.

```
DEPARTMENT_ID FIRST_NAME  LAST_NAME  HIRE_DATE   COMMISSION
------------- ----------- ---------- ----------- ----------
           50 Adam        Fripp      10-APR-2005          0
           50 Shanta      Vollman    10-OCT-2005          0
           50 Kevin       Mourgos    16-NOV-2007          0
```

```
        80 Karen       Partners   05-JAN-2005         .3
        80 Alberto     Errazuriz  10-MAR-2005         .3
        80 Gerald      Cambrault  15-OCT-2007         .3
        80 Eleni       Zlotkey    29-JAN-2008         .2
        90 Neena       Kochhar    21-SEP-2005          0
```

10. Change the previous query to use a standard function to display the commission percentage. This time, order the list so that those hired closest to Jan 1, 2007 are listed first.

```
DEPARTMENT_ID FIRST_NAME  LAST_NAME  HIRE_DATE   COMMISSION
------------- ----------- ---------- ----------- ----------
           80 Gerald      Cambrault  15-OCT-2007         .3
           50 Kevin       Mourgos    16-NOV-2007          0
           80 Eleni       Zlotkey    29-JAN-2008         .2
           50 Shanta      Vollman    10-OCT-2005          0
           90 Neena       Kochhar    21-SEP-2005          0
           50 Adam        Fripp      10-APR-2005          0
           80 Alberto     Errazuriz  10-MAR-2005         .3
           80 Karen       Partners   05-JAN-2005         .3
```

Congratulations!
You have finished
this lab exercise!

Chapter 9: Programming with PL/SQL

Exercise 9.1: Building Anonymous Blocks

Connect to the `HR` account.

1. In the declaration section, declare a record `emp_rec` that uses the `employees` table as a basis.

   Use `%ROWTYPE`.

2. In the executable section, retrieve information for employee with a last name of `'Austin'` into `emp_rec`.

3. In the executable section, write a conditional structure that sets the new salary based on the employee's commission.

```
if commission_pct is undefined or zero then
  increase salary by a flat $500
if commission_pct is less than .2 then
  increase salary by $300
otherwise
  increase salary by $100
```

4. Write an `UPDATE` statement that sets the employee's salary to the value derived in the `IF` statement.

5. Select from the employees table for employee Austin to view the salary prior to running the block.

6. Execute the block. Again, query the employees table to check your results.

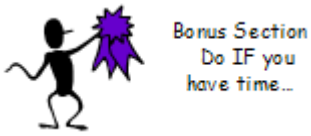
```
LAST_NAME       SALARY
--------------- ----------
Austin          5300
```

7. Test your code with other employees. Use Lee and then King. Validate your results by viewing the before and after values.

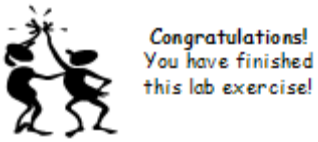
8. Roll back your changes.

9. Enhance your PL/SQL block by using a `CASE` statement. Execute the block and check your results. Test with other employees as in Step 6.

10. Roll back your changes.



11. Enhance the block by adding exception handling. Use the block which uses the `CASE` statement. The select statement may return no rows, one row, or many rows. Add the `EXCEPTION` section and add a handler for each of the possible errors.
12. Good practice recommends that you also add a handler for any other error that may occur. If unexpected conditions occur, raise an error, display “Contact support” and append the Oracle error message.
13. Test the program by running it for different last names. Use Austin, Smith, and Howard.
14. Roll back your changes.



Exercise 9.2: Using Cursors

Connect to the HR account.

In this exercise, you will declare and use a cursor to process the records in the employees table. Increase employee salary by $5,000 for employees who were hired before Jan. 1, 2003. The cursor will accept one parameter, which limits the query to return only the required employees.

1. In the declaration section, declare a cursor that selects an employee record from the employees table. The cursor should accept one parameter called in_date_hired. Compare in_date_hired with the date_hired column in the WHERE clause of the SELECT statement. The cursor should include a FOR UPDATE clause to lock the selected rows.

2. Declare a record to hold the cursor results. Also declare a numeric variable, raise, and initialize it to 5000. Finally, declare a date variable, v_date, and initialize it to Jan. 1, 2003.

3. In the executable section, open the cursor and pass v_date as a parameter.

4. Use a simple LOOP…END LOOP construct that loops through each record that the cursor returns.

5. Within the LOOP, use a FETCH statement to retrieve the row into the cursor record.

6. Add an EXIT WHEN statement to exit the loop when no more rows are returned by the cursor.

   Warning! An exit statement must be included to avoid an endless loop.

7. Add an UPDATE statement, which uses the current cursor row.

8. Execute the block and check your results. Make sure to check records that should not have been updated as well!

```
EMPLOYEE_ID FIRST_NAME      LAST_NAME        HIRE_DATE     SALARY
----------- --------------- ---------------- ----------- --------
        203 Susan           Mavris           07-JUN-2002    11500
        204 Hermann         Baer             07-JUN-2002    15000
        205 Shelley         Higgins          07-JUN-2002    17008
        206 William         Gietz            07-JUN-2002    13300
        102 Lex             De Haan          13-JAN-2001    22000
        108 Nancy           Greenberg        17-AUG-2002    17008
        109 Daniel          Faviet           16-AUG-2002    14000
        114 Den             Raphaely         07-DEC-2002    16000

8 rows selected.
```

9. Roll back your changes.

10. Modify the previous example to use a `FOR-LOOP` cursor instead of a regular cursor with a `LOOP`. ***Set*** the salary to $11000; i.e., not a raise.

```
EMPLOYEE_ID FIRST_NAME      LAST_NAME        HIRE_DATE     SALARY
----------- --------------- ---------------- ----------- --------
        203 Susan           Mavris           07-JUN-2002    11000
        204 Hermann         Baer             07-JUN-2002    11000
        205 Shelley         Higgins          07-JUN-2002    11000
        206 William         Gietz            07-JUN-2002    11000
        102 Lex             De Haan          13-JAN-2001    11000
        108 Nancy           Greenberg        17-AUG-2002    11000
        109 Daniel          Faviet           16-AUG-2002    11000
        114 Den             Raphaely         07-DEC-2002    11000

8 rows selected.
```

11. Roll back your changes.

Chapter 10:
Creating Stored Procedures, Functions, and Packages

Exercise 10.1: Stored Procedures, Functions, and Packages

Connect to the `HR` account.

In this exercise, you will create and execute a procedure that updates an employee if one exists. Otherwise, assume this is a new employee and insert a new employee into the table.

1. Use the `CREATE PROCEDURE` statement to define a procedure called `update_emp`. It requires six input parameters: `parm_employee_id`, `parm_last_name`, `parm_email`, `parm_hire_date`, `parm_job_id` and `parm_salary`.

2. In the executable section, update the salary in the employees table for the specified employee_id. Verify that the last name, hire date, and job_id match the input parameters prior to making the change.

3. If no rows were updated because the input employee id does not exist in the `employees` table, then insert the input data into the table as a new row.

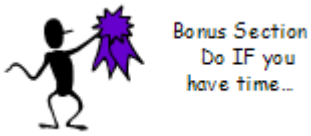
4. Complete the procedure with an END statement.

5. Store the procedure in the database by executing the CREATE PROCEDURE statement.

   If you get a warning that the procedure was created with compilation errors, use the `SHOW ERRORS` command.

6. Test the procedure by creating a simple PL/SQL block that calls the procedure and passes parameters to it. Use employees Chen and Johnston for your tests. Check your results.

7. Roll back your changes.



Change the procedure from the previous exercise to perform an `INSERT` only if the department the employee is assigned to presently has a manager.

We will use a separate function to perform the test. The function is used only by the procedure. Therefore, it can be hidden by defining it as a private function in a package.

8. Start by dropping the independent procedure update_emp since we now want to include it into a package.
9. Write a package specification, pack_employee that contains a declaration of the update_emp procedure. Use the CREATE PACKAGE statement. You will need to add one more parameter to pass in a department id.
10. Use the procedure from previous section as the basis for this procedure specification. Remember to add the additional parameter for department id. Remove the procedure body, which starts with the keyword `IS` and finishes with an `END` statement.
11. Submit the package specification to the database for compilation and storage.
12. Write the package body for pack_employee using the `CREATE PACKAGE BODY` statement.
13. Write a function definition in the package body. The function should return the manager id for the assigned department or a null value if a manager is not assigned. The function will require a single parameter for the `department_id`.
14. The function should be defined before the procedure.
15. Use a `FUNCTION` definition statement. The `FUNCTION` consists of a `SELECT` statement and a `RETURN` statement.
16. Copy the procedure from the previous section and make the following changes:
    a. Add an additional parameter to pass in the `department_id`.
    b. Add an `IF-THEN` construct around the `INSERT` statement so that it inserts only if the department being assigned to has a manager assigned.
    c. Use a `PROCEDURE` definition statement instead of the `CREATE PROCEDURE` statement within the package body.
17. The package body must be completed with an `END` statement.
18. Submit the package body to the database for compilation and storage.

19. Test the package by executing the packaged procedure from an anonymous PL/SQL block. Check your results.