***Single Row Functions***

**Practice 3 Solutions**

1. Write a query to display the current date. Label the column Date.

**Ans:**

**SELECT sysdate "Date"**

**FROM dual;**

2. For each employee, display the employee number, last\_name, salary, and salary increased by 15%

and expressed as a whole number. Label the column New Salary. Place your SQL statement in a

text file named lab3\_2.sql.

**Ans:**

**SELECT employee\_id, last\_name, salary,**

**ROUND(salary \* 1.15, 0) "New Salary"**

**FROM employees;**

1. Run your query in the file lab3\_2.sql.

**Ans:**

**SELECT employee\_id, last\_name, salary,**

**ROUND(salary \* 1.15, 0) "New Salary"**

**FROM employees;**

4. Modify your query lab3\_2.sql to add a column that subtracts the old salary from

the new salary. Label the column Increase. Save the contents of the file as lab3\_4.sql. Run

the revised query.

**Ans:**

**SELECT employee\_id, last\_name,**

**salary,ROUND(salary \* 1.15, 0) "New Salary",**

**ROUND(salary \* 1.15, 0) - salary "Increase"**

**FROM employees;**

5. Write a query that displays the employee’s last names with the first letter capitalized and all other

letters lowercase and the length of the name for all employees whose name starts with *J*, *A*, or *M*.

Give each column an appropriate label. Sort the results by the employees’ last names.

**Ans:**

**SELECT INITCAP(last\_name) "Name",LENGTH(last\_name) "Length"**

**FROM employees**

**WHERE last\_name LIKE 'J%'**

**OR last\_name LIKE 'M%'**

**OR last\_name LIKE 'A%'**

**ORDER BY last\_name;**

6. For each employee, display the employee’s last name, and calculate the number of months between

today and the date the employee was hired. Label the column MONTHS\_WORKED. Order your results

by the number of months employed. Round the number of months up to the closest whole number.

**Note:** Your results will differ.

**Ans:**

**SELECT last\_name, ROUND(MONTHS\_BETWEEN**

**(SYSDATE, hire\_date)) MONTHS\_WORKED**

**FROM employees**

**ORDER BY MONTHS\_BETWEEN(SYSDATE, hire\_date);**

7. Write a query that produces the following for each employee:

<employee last name> earns <salary> monthly but wants <3 times

salary>. Label the column Dream Salaries.

**Ans:**

**SELECT last\_name || ' earns '|| TO\_CHAR(salary, 'fm$99,999.00')**

**|| ' monthly but wants '|| TO\_CHAR(salary \* 3, 'fm$99,999.00')**

**|| '.' "Dream Salaries"**

**FROM employees;**

If you have time, complete the following exercises:

8. Create a query to display the last name and salary for all employees. Format the salary to be 15

characters long, left-padded with $. Label the column SALARY.

**Ans:**

**SELECT last\_name,LPAD(salary, 15, '$') SALARY**

**FROM employees;**

9. Display each employee’s last name, hire date, and salary review date, which is the first Monday after six months of service. Label the column REVIEW. Format the dates to appear in the format similar to “Monday, the Thirty-First of July, 2000.”

**Ans:**

**SELECT last\_name, hire\_date,**

**TO\_CHAR(NEXT\_DAY(ADD\_MONTHS(hire\_date, 6),'MONDAY'),**

**'fmDay, "the" Ddspth "of" Month, YYYY') REVIEW**

**FROM employees;**

10. Display the last name, hire date, and day of the week on which the employee started. Label the column DAY. Order the results by the day of the week starting with Monday.

**Ans:**

**SELECT last\_name, hire\_date,TO\_CHAR(hire\_date, 'DAY') DAY**

**FROM employees**

**ORDER BY TO**\_**CHAR(hire\_date - 1, 'd');**

If you want an extra challenge, complete the following exercises:

11. Create a query that displays the employees’ last names and commission amounts. If an employee does not earn commission, put “No Commission.” Label the column COMM.

**Ans:**

**SELECT last\_name,NVL(TO\_CHAR(commission\_pct), 'No Commission') COMM**

**FROM employees;**

12. Create a query that displays the employees’ last names and indicates the amounts of their annual

salaries with asterisks. Each asterisk signifies a thousand doll ars. Sort the data in descending order

of salary. Label the column EMPLOYEES\_AND\_THEIR\_SALARIES.

**Ans:**

**SELECT rpad(last\_name, 8)||' '|| rpad(' ', salary/1000+1, '\*')**

**EMPLOYEES\_AND\_THEIR\_SALARIES**

**FROM employees**

**ORDER BY salary DESC;**

13. Using the DECODE function, write a query that displays the grade of all employees based on the value of the column JOB\_ID, as per the following data:

*JOB GRADE*

AD\_PRES A

ST\_MAN B

IT\_PROG C

SA\_REP D

ST\_CLERK E

None of the above 0

**Ans:**

**SELECT job\_id, decode (job\_id,**

**'ST\_CLERK', 'E',**

**'SA\_REP', 'D',**

**'IT\_PROG', 'C',**

**'ST\_MAN', 'B',**

**'AD\_PRES', 'A',**

**'0')GRADE**

**FROM employees;**

14. Rewrite the statement in the preceding question using the CASE syntax.

**Ans:**

**SELECT job\_id, CASE job\_id**

**WHEN 'ST\_CLERK' THEN 'E'**

**WHEN 'SA\_REP' THEN 'D'**

**WHEN 'IT\_PROG' THEN 'C'**

**WHEN 'ST\_MAN' THEN 'B'**

**WHEN 'AD\_PRES' THEN 'A'**

**ELSE '0' END GRADE**

**FROM employees;**