

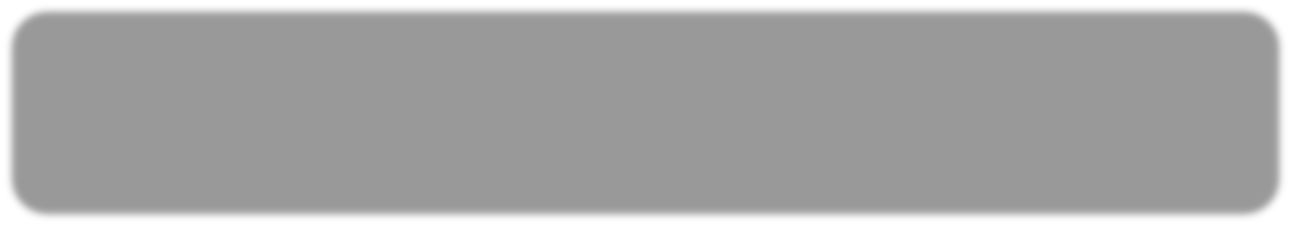
**Database Management System**

**(DBMS – 204)**

**Experiment # 03**

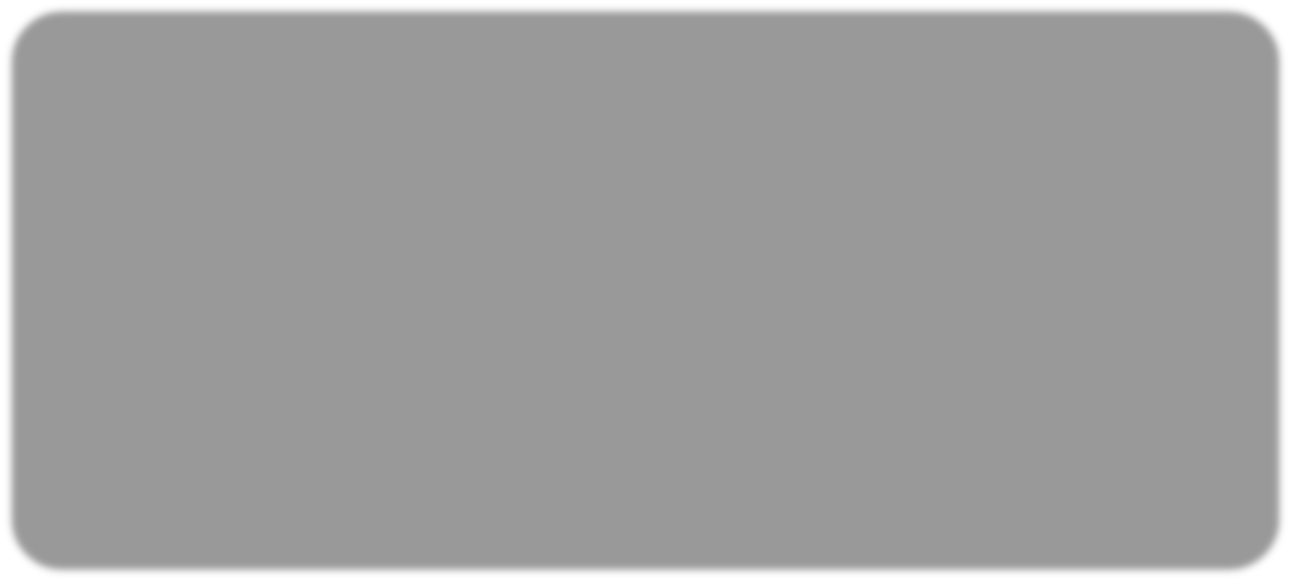
# Single Row Functions

Se-061



Student Name: Muhammad Faraz Ansar

Roll Number:



**Maximum Marks**

**Performance = 05**

**Viva = 05**

**Total = 10**

**Marks Obtained**

**Remarks (if any)**

**Experiment evaluated by**

Instructor Name

:

Engr. Adiba Jafar

Signature

and Date

:

***OUTCOMES***

After completing this lesson, you should be able to do the following:

* Describe various types of functions available in SQL.
* Use character, number, and date functions in SELECT statement.
* Describe the use of conversion functions.

## SQL Functions

**1. Input Output Function**

**Function performs action arg 1 arg 2 Result value arg n Note:**

Most of the functions described in this lesson are specific to Oracle Corporation’s version of SQL.

## Two Types of SQL Functions

### 1. Single-row functions

* Manipulate data items
* Accept arguments and return one value
* Act on each row returned
* Return one result per row
* May modify the data type
* Can be nested
* Accept arguments which can be a column or an expression function\_name[(arg1, arg2,...)]

### 2. Character Functions

* Case-manipulation functions

Character-manipulation functions

* CONCAT
* UPPER
* SUBSTR
* INITCAP
* LENGTH
* INSTR
* LPAD | RPAD
* TRIM
* REPLACE

**Note:** The functions discussed in this lesson are only some of the available functions.

### Case Manipulation Functions

These functions convert case for character strings.

Function Result

LOWER('SQL Course') sql course

UPPER('SQL Course') SQL COURSE

INITCAP('SQL Course') Sql Course

**QUERY**

SELECT empno, ename, deptno FROM emp WHERE ename = 'smith'; no rows selected

SELECT empno, ename, deptno FROM emp WHERE LOWER(ename) = 'smith`';

### Character-Manipulation Functions

These functions manipulate character strings:

|  |  |
| --- | --- |
| **Function** | **RESULT** |
| CONCAT('Hello', 'World') | HelloWorld |
| SUBSTR('HelloWorld',1,5) | Hello |
| LENGTH('HelloWorld') | 10 |
| INSTR('HelloWorld', 'W') | 6 |
| LPAD(sal,10,'\*') | \*\*\*\*\*24000 |
| RPAD(sal, 10, '\*') | 24000\*\*\*\*\* |
| TRIM('H' FROM 'HelloWorld') | elloWorld |

### QUERY

SELECT empno, CONCAT(ename, job) NAME, sal,

LENGTH (ename), INSTR(ename, 'A') "Contains 'a'?"

FROM emp WHERE SUBSTR(job, 1,3) = 'MAN';

### Number Functions

* ROUND : Rounds value to specified decimal ROUND(45.926, 2) 45.93
* TRUNC : Truncates value to specified decimal TRUNC(45.926, 2) 45.92
* MOD : Returns remainder of division MOD(1600, 300) 100

### QUERY (ROUND)

SELECT ROUND(45.923,2), ROUND(45.923,0),ROUND(45.923,-1) FROM DUAL;

DUAL is a dummy table you can use to view results from functions and calculations.



### QUERY (TRUNC)

**SELECT TRUNC(45.923,2), TRUNC(45.923), TRUNC(45.923,-2) FROM DUAL;**



### QUERY (MOD FUNCTION)

Calculate the remainder of a salary after it is divided by 5000 for all employees whose job title is sales representative.

SELECT ename, sal, MOD(sal, 5000) FROM emp WHERE job = 'MANAGER';



### Working with Dates

* Oracle database stores dates in an internal numeric format: century, year, month, day, hours,minutes, seconds.
* The default date display format is DD-MM-YY.
  + Allows you to store 21st century dates in the 20th century by specifying only the last two digits of the year.
  + Allows you to store 20th century dates in the 21st century in the same way.

### QUERY

SYSDATE is a function that returns:

•Date

•Time

Example Display the current date using the DUAL table.

SELECT SYSDATE FROM DUAL;



### Arithmetic with Dates

* Add or subtract a number to or from a date for a resultant date value. ❖ date + number Date Adds a number of days to a date
  + date - number Date Subtracts a number of days from a date

* Subtract two dates to find the number of days between those dates.
  + date - date Number of days Subtracts one date from another

* Add hours to a date by dividing the number of hours by 24.
  + date + number/24 Date Adds a number of hours to a date

### Using Arithmetic Operators with Dates

SELECT ename AS last\_name , (SYSDATE-hiredate)/7 AS WEEKS FROM emp

WHERE deptno= 30;



**Note:**SYSDATE is a SQL function that returns the current date and time. Your results may differ from the example. If a more current date is subtracted from an older date, the difference is a negative number.

### Using Date Functions

* MONTHS\_BETWEEN ('01-SEP-95','11-JAN-94')

19.6774194

* ADD\_MONTHS ('11-JAN-94',6) '11-JUL-94'
* NEXT\_DAY ('01-SEP-95','FRIDAY') '08-SEP-95'
* LAST\_DAY('01-FEB-95') '28-FEB-95'

**Assume SYSDATE = '25-JUL-95':**

* ROUND(SYSDATE,'MONTH') 01-AUG-95
* ROUND(SYSDATE ,'YEAR') 01-JAN-96
* TRUNC(SYSDATE ,'MONTH') 01-JUL-95
* TRUNC(SYSDATE ,'YEAR') 01-JAN-95

### Example

Compare the hire dates for all employees who started in 1981. Display the employee number, hire date, and month started using the ROUND and TRUNC functions.

SELECT empno, hiredate, ROUND(hiredate, 'MONTH'), TRUNC(hiredate, 'MONTH') FROM emp WHERE hiredate LIKE '%81';

Practice 3, Part 1

This practice is designed to give you a variety of exercises using different functions available for character,number, and date data types.

Complete questions 1 through 5 of Practice 3, found at the end of this lesson.

## Conversion Functions

**Data-type conversion**

**Implicit data-type conversion**

### Explicit data-type conversion

**Note:** Although implicit data-type conversion is available, it is recommended that you do explicit data type conversion to ensure the reliability of your SQL statements.

### Implicit Data-Type Conversion

For assignments, the Oracle server can automatically convert the following:

|  |  |
| --- | --- |
| From | To |
| VARCHAR2 or CHAR | NUMBER |
| VARCHAR2 or CHAR | DATE |
| NUMBER | VARCHAR2 |
| DATE | VARCHAR2 |

Note: CHAR to NUMBER conversions succeed only if the character string represents a valid number.

### Explicit Data-Type Conversion

1. TO\_NUMBER
2. TO\_DATE
3. NUMBER
4. DATE
5. CHARACTER
6. TO\_CHAR

|  |  |
| --- | --- |
| **Function** | **Purpose** |
| TO\_CHAR( *number* | *date* ,[ *fmt* ], | Converts a number or date value to a |
| VARCHAR2 *[nlsparams* ]) | character string with format model *fmt*. |
| TO\_NUMBER( *char,[fmt],* | Converts a character string containing digits to a *[nlsparams]* ) number in the format specified by the optional format model *Fmt* . The nlsparams parameter has the same purpose in this function as in the TO\_CHAR function for number conversion. |
| TO\_DATE(c*har* ,[ *fmt* ],[ *nlsparams* ]) | Converts a character string representing a date to |

a date value according to the *fmt* specified. If *fmt* is omitted, the format is DD-MON-YY.The nlsparams parameter has the same purpose in this function as in the TO\_CHAR function for date conversion.

Note: The list of functions mentioned in this lesson includes only some of the available conversion functions.



## Using the TO\_CHAR Function with Dates

TO\_CHAR(date, ' format\_model ') The format model:

* Must be enclosed in single quotation marks and is case Sensitive.
* Can include any valid date format element.
* Has an fm element to remove padded blanks or suppress leading zeros.
* Is Separated from the date value by a comma.

SELECT empno, TO\_CHAR(hiredate, 'MM/YYYY') Hiredate

FROM emp

WHERE ename = 'SMITH';

### Elements of the Date Format Model

|  |  |
| --- | --- |
| YYYY | Full year in numbers |
| YEAR | Year spelled out |
| MM | Two-digit value for month |
| MONTH | Full name of the month |
| MON | Three-letter abbreviation of the |
| Month | Three-letter abbreviation of the |
| DY | day of the week |
| DAY | Full name of the day of the week |
| DD | Numeric day of the month |

### Elements of the Date Format Model

* Time elements format the time portion of the date. HH24:MI:SS AM 15:45:32 PM
* Add character strings by enclosing them in double quotation marks. DD "of" MONTH 12 of OCTOBER
* Number suffixes spell out numbers. ddspth fourteenth

### Using the TO\_CHAR Function with Dates

SELECT ename, TO\_CHAR(hiredate, 'fmDD Month YYYY') HIREDATE FROM emp;

SELECT ename, TO\_CHAR(hiredate, 'fmDdspth "of" Month YYYY fmHH:MI:SS AM') HIREDATE FROM emp;

Notice that the month follows the format model specified: in other words, the first letter is capitalized and the rest are lowercase.

### Using the TO\_CHAR Function with Numbers

TO\_CHAR( number, ' format\_model ')

These are some of the format elements you can use with the TO\_CHAR function to display a number value as a character:

9 Represents a number 0

Forces a zero to be displayed $ Places a floating dollar sign

L Uses the floating local currency symbol. Prints a decimal point, Prints a thousand indicator

Element Description Example Result

9 Numeric position (number of 9s determine display 999999 1234 width)

0 Display leading zeros 099999 001234

$ Floating dollar sign $999999 $1234

L Floating local currency symbol L999999 FF1234

. Decimal point in position specified 999999.99 1234.00

, Comma in position specified 999,999 1,234

MI Minus signs to right (negative values) 999999MI 1234- PR Parenthesize negative numbers 999999PR <1234>

EEEE Scientific notation (format must specify four Es) 99.999EEEE 1.234E+03

V Multiply by 10 *n* times ( *n* = number of 9s after V) 9999V99 123400 B Display zero values as blank, not 0 B9999.99 1234.00



### Using the TO\_CHAR Function with Numbers

SELECT TO\_CHAR(sal, '$99,999.00') SAL FROM emp;

#### Using the TO\_NUMBER And TO\_DATE Functions

* Convert a character string to a number format using the TO\_NUMBER function: TO\_NUMBER( char [,'format\_model'])
* Convert a character string to a date format using the TO\_DATE function:

TO\_DATE( char [, 'format\_model'])

* These functions have an fx modifier. This modifier specifies the exact matching for the character argument and date format model of a TO\_DATE function.

Example

Display the names and hiredates of all the employees who joined on May 01, 1981. Because the fx modifier is used, an exact match is required and the spaces after the word

“May” are not recognized.

SELECT ename, hiredate FROM emp

WHERE hiredate = TO\_DATE('May 01, 1981', 'fxMonth DD, YYYY')

### Example of RR Date Format

To find employees hired prior to 1982, use the RR format, which produces the same results whether the command is run in 1982 or now:

SELECT ename, TO\_CHAR(hiredate, 'DD-Mon-YYYY')

FROM emp

WHERE hiredate < TO\_DATE('01-Jan-82', 'DD-Mon-RR');

SELECT ename, TO\_CHAR(hiredate, 'DD -Mon-YYYY')

FROM emp

WHERE TO\_DATE(hiredate, 'DD-Mon-YY') < '01-Jan-1982'; no rows selected

### Nesting Functions

* Single-row functions can be nested to any level.
* Nested functions are evaluated from deepest level to the least deep level.

F3(F2(F1(col,arg1),arg2),arg3)

Step 1 = Result 1

Step 2 = Result 2

Step 3 = Result 3

Nesting Functions

Single-row functions can be nested to any depth. Nested functions are evaluated from the innermost level to the outermost level. Some examples follow to show you the flexibility of these functions.

#### Example

Display the date of the next Friday that is six months from the hire date. The resulting date should appear as Friday, August 13th, 1999. Order the results by hire date.

SELECT TO\_CHAR(NEXT\_DAY(ADD\_MONTHS(hire\_date, 6), 'FRIDAY'), 'fmDay, Month DDth, YYYY')

"Next 6 Month Review"

FROM emp

ORDER BY hiredate;

### General Functions

These functions work with any data type and pertain to using null value.

* NVL (expr1, expr2)
* NVL2 (expr1, expr2, expr3)
* NULLIF (expr1, expr2)

•COALESCE (expr1, expr2, ..., exprn)

|  |  |  |
| --- | --- | --- |
| **Function** |  | **Description** |
| NVL |  | Converts a null value to an actual value |
| NVL2 |  | If expr1 is not null, NVL2 returns expr2. |
| NULLIF |  | If expr1 is null, NVL2 returns. The argument can have any data type. expr3 expr1 |
| COALESCE |  | Compares two expressions and returns null if they are equal, or the first expression if they are not equal Returns the first non-null expression in the expression list |

### NVL Function

* Converts a null to an actual value
* Data types that can be used are date, character,and number.
* Data types must match:
  + NVL(commission,0)
  + NVL(hiredate,'01-JAN-97')
  + NVL(job,'No Job Yet')

NVL Conversions for Various Data Types

Data Type Conversion Example

NUMBER NVL( *number\_column* ,9)

DATE NVL( *date\_column,* '01-JAN-95')

CHAR or VARCHAR2 NVL( *character\_column* , 'Unavailable')

**Using the NVL Function** SELECT ename, sal, NVL(comm, 0),

(sal\*12) + (sal\*12\*NVL(comm, 0)) AN\_SAL FROM emp;

The NVL Function

To calculate the annual compensation of all employees, you need to multiply the monthly salary by 12 and then add the commission percentage to it.

SELECT ename, sal, comm, (sal\*12) + (sal\*12\*comm) AN\_SAL FROM emp;

### Using the NVL2 Function

SELECT ename, sal, comm,

NVL2(comm,'SAL+COMM', 'SAL') income

FROM emp WHERE deptno IN (10, 30);

**Using the NULLIF Function** SELECT ename, LENGTH(ename) "expr1", ename, LENGTH(ename) "expr2",

NULLIF(LENGTH(ename), LENGTH(ename)) result

FROM emp; Note:

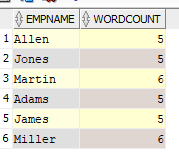
The NULLIF function is logically equivalent to the following CASE expression. The CASE expression is discussed in a subsequent page:

CASE WHEN expr1 = expr 2 THEN NULL ELSE expr1 END

### Practice 2, Part 1 (continued)

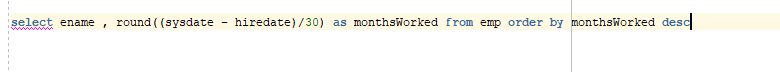
Write a query that displays the employee’s names with the first letter capitalized and all other letters lowercase and the length of the names, for all employees whose name starts with *J* , *A* , or *M* . Give each column an appropriate label. Sort the results by the employees’ names.

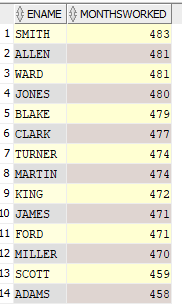




#### Practice 2, Part 2

For each employee, display the employee’s 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.



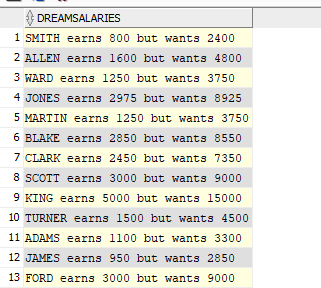


### Practice 3, Part 2 (continued)

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

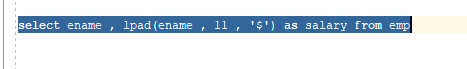
< employee name> earns <salary> monthly but wants <3 times salary >. Label the column Dream Salaries .

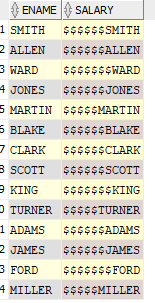




If you have time, complete the following exercises:

1. Create a query to display the name and salary for all employees. Format the salary to be 15 characters long, left-padded with $. Label the column SALARY .

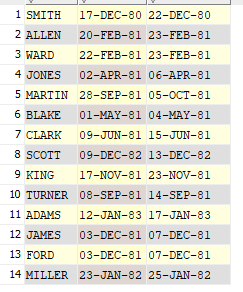


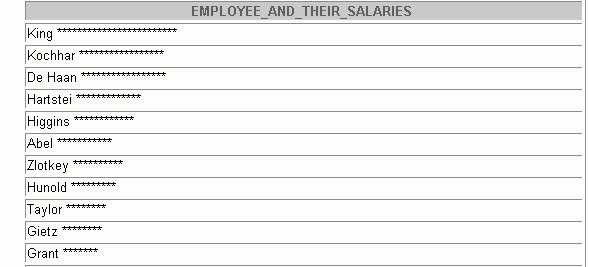


1. Display each employee’s name, hiredate, and salary review date, which is the first Monday after six months of service. Label the column

REVIEW. Format the dates to appear similar to “Monday, the Thirty-First of July, 2000.”

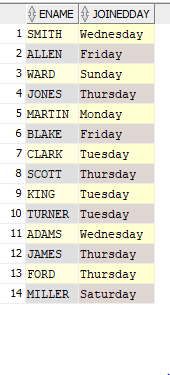






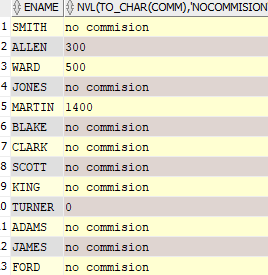
1. Display the name, hiredate, and day of the week on which the employee started. Label the column DAY . Order the results by the day of the week starting with Monday.





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





1. Create a query that displays the employees’ names and indicates the amounts of their annual salaries with asterisks. Each asterisk signifies a thousand dollars. Sort the data in descending order of salary. Label the column

EMPLOYEES\_AND\_THEIR\_SALARIES.



