3-Single-Row Functions

Objectives

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 statements**
- **Describe the use of conversion functions**

SQL Functions

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

Functions

Single-row functions

Single-Row Functions

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,...)]

Single-Row Functions

Single-Row Functions

Character

General

Number

Single-row functions

Conversion

Date

Character Functions

Character functions

Case-manipulation functions

Character-manipulation functions

LOWER
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

Using Case Manipulation Functions

Display the employee number, name, and department number for employee Higgins:

```
SELECT employee_id, last_name, department_id
FROM   employees
WHERE  last_name = 'higgins';
no rows selected
SELECT employee_id, last_name, department_id
FROM   employees
WHERE  LOWER(last_name) = 'higgins';
```

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(salary,10,'*')	*****24000
RPAD(salary, 10, '*')	24000*****
TRIM('H' FROM 'HelloWorld')	elloWorld

EMPLOYEE_ID	NAME	JOB_ID	LENGTH(LAST_NAME)	Contains 'a'?
174	EllenAbel	SA_REP	4	0
176	JonathonTaylor	SA_REP	6	2
178	KimberelyGrant	SA_REP	5	3
202	PatFay	MK_REP	3	2

EMPLOYEE_ID	NAME	LENGTH(LAST_NAME)	Contains 'a'?
102	LexDe Haan	7	5
200	JenniferWhalen	6	3
201	MichaelHartstein	9	2

Using the Character-Manipulation Functions

SELECT employee_id, CONCAT(first_name, last_name) NAME, job_id,
LENGTH (last_name), INSTR(last_name, 'a') "Contains 'a'?"
FROM employees
WHERE SUBSTR(job_id, 4) = 'REP';

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

ROUND(45.923,2)	ROUND(45.923,0)	ROUND(45.923,-1)
45.92	46	50

Using the ROUND Function

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.

TRUNC(45.923,2)	TRUNC(45.923)	TRUNC(45.923, 2)
45.92	45	0

Using the TRUNC Function

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

LAST_NAME	SALARY	MOD(SALARY,5000)
Abel	11000	1000
Taylor	8600	3600
Grant	7000	2000

Using the MOD Function

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

SELECT last_name, salary, MOD(salary, 5000)
FROM employees

WHERE job_id = 'SA_REP';

LAST_NAME	HIRE_DATE
Gietz	07-JUN-94
Grant	24-MAY-99

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-MON-RR.
– Allows you to store 21st century dates in the 20th century by specifying only the last two digits of the year.
– Allowa you to store 20th century dates in the 21st century in the same way.

SELECT last_name, hire_date FROM employees
WHERE last_name like 'G%';

SYSDATE
08-MAR-01

Working with Dates

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.
•Subtract two dates to find the number of days between those dates.
•Add hours to a date by dividing the number of hours by 24.

Operation Result Description
date + number Date Adds a number of days to a date
date - number Date Subtracts a number of days from a date
date - date Number of days Subtracts one date from another
date + number/24 Date Adds a number of hours to a date

LAST_NAME	WEEKS
King	716.227963
Kochhar	588.084706
De Haan	425.227963

Using Arithmetic Operators with Dates

SELECT last_name, (SYSDATE-hire_date)/7 AS WEEKS

```
FROM   employees
WHERE  department_id = 90;
```

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.

EMPLOYEE_ID	HIRE_DATE	TENURE	REVIEW	NEXT_DAY(	LAST_DAY(
107	07-FEB-99	25.0548529	07-AUG-99	12-FEB-99	28-FEB-99
124	16-NOV-99	15.7645303	16-MAY-00	19-NOV-99	30-NOV-99
143	15-MAR-98	35.7967884	15-SEP-98	20-MAR-98	31-MAR-98
144	09-JUL-98	31.9903368	09-JAN-99	10-JUL-98	31-JUL-98
149	29-JAN-00	13.3451755	29-JUL-00	04-FEB-00	31-JAN-00
176	24-MAR-98	35.5064658	24-SEP-98	27-MAR-98	31-MAR-98
178	24-MAY-99	21.5064658	24-NOV-99	28-MAY-99	31-MAY-99

7 rows selected.

Date Functions

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'

EMPLOYEE_ID	HIRE_DATE	ROUND(HIR	TRUNC(HIR
142	29-JAN-97	01-FEB-97	01-JAN-97
202	17-AUG-97	01-SEP-97	01-AUG-97

Using Date Functions

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 1997. Display the employee number, hire date, and month started using the ROUND and TRUNC functions.
SELECT employee_id, hire_date, ROUND(hire_date, 'MONTH'), TRUNC(hire_date, 'MONTH') FROM employees WHERE hire_date LIKE '%97';
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
VARCHAR2 or CHAR	DATE

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

Explicit Data-Type Conversion
TO_NUMBER
TO_DATE
NUMBER
DATE
CHARACTER
TO_CHAR
TO_CHAR

Function	Purpose
TO_CHAR(*number* | *date* ,[*fmt*], VARCHAR2 *[nlsparams*])	Converts a number or date value to a character string with format model *fmt*.
TO_NUMBER(*char,[fmt],* *[nlsparams]*)	Converts a character string containing digits to a 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(*char* ,[*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.

EMPLOYEE_ID	MONTH
205	06/94

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 employee_id, TO_CHAR(hire_date, 'MM/YY') Month_Hired
FROM   employees
WHERE  last_name = 'Higgins';
```

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 last_name, TO_CHAR(hire_date, 'fmDD Month YYYY') HIREDATE
FROM   employees;
```

SELECT last_name, TO_CHAR(hire_date, 'fmDdspth "of" Month YYYY fmHH:MI:SS AM') HIREDATE FROM employees;
Notice that the month follows the format model specified: in oth er 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

SALARY
$5,000.00

values as blank, not 0 B9999.99 1234.00

Using the TO_CHAR Function with Numbers

SELECT TO_CHAR(salary, '$99,999.00') SALARY
FROM employees
WHERE last_name = 'Ernst';

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 hire dates of all the employees who joined on May 24, 1999. Because the fx modifier is used, an exact match is required and the spaces afte r the word "May" are not recognized.

```
SELECT last_name, hire_date FROM   employees
WHERE  hire_date = TO_DATE('May       24, 1999', 'fxMonth DD, YYYY')
```

Example of RR Date Format

To find employees hired prior to 1990, use the RR format, which produces the same results whether the command is run in 1999 or now:

```
SELECT last_name, TO_CHAR(hire_date, 'DD-Mon-YYYY')
FROM   employees
WHERE  hire_date < TO_DATE('01-Jan-90', 'DD-Mon-RR');
SELECT last_name, TO_CHAR(hire_date, 'DD -Mon-yyyy')
FROM   employees
WHERE  TO_DATE(hire_date, 'DD-Mon-yy') < '01-Jan-1990';
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.

LAST_NAME	NVL(TO_CHAR(MANAGER_ID),'NOMANAGER')
King	No Manager

Nesting Functions

```
SELECT last_name,
NVL(TO_CHAR(manager_id), 'No Manager')
FROM   employees
WHERE  manager_id IS NULL;
```

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      employees
ORDER BY  hire_date;
```

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_pct,0)
– NVL(hire_date,'01-JAN-97')
– NVL(job_id,'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')

Vargas	2500		
Zlotkey	10500	.2	151200
Abel	11000	.3	171600
Taylor	8600	.2	123840

CHAR or VARCHAR2 NVL(*character_column* , 'Unavailable')

LAST_NAME	SALARY	NVL(COMMISSION_PCT,0)	AN_SAL
King	24000	0	288000
Kochhar	17000	0	204000
De Haan	17000	0	204000
Hunold	9000	0	108000
Ernst	6000	0	72000
Lorentz	4200	0	50400
Mourgos	5800	0	69600
Rajs	3500	0	42000
Davies	3100	0	37200
Matos	2600	0	31200
Vargas	2500	0	30000
Zlotkey	10500	.2	151200
Abel	11000	.3	171600

20 rows selected.

LAST_NAME	SALARY	COMMISSION_PCT	AN_SAL

20 rows selected.

Using the NVL Function

SELECT last_name, salary, NVL(commission_pct, 0),
(salary*12) + (salary*12*NVL(commission_pct, 0)) AN_SAL
FROM employees;

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 last_name, salary, commission_pct, (salary*12) + (salary*12*commission_pct) AN_SAL FROM employees;

LAST_NAME	SALARY	COMMISSION_PCT	INCOME
Zlotkey	10500	.2	SAL+COMM
Abel	11000	.3	SAL+COMM
Taylor	8600	.2	SAL+COMM
Mourgos	5800		SAL
Rajs	3500		SAL
Davies	3100		SAL
Matos	2600		SAL
Vargas	2500		SAL

8 rows selected.

Using the NVL2 Function

SELECT last_name, salary, commission_pct,
NVL2(commission_pct,'SAL+COMM', 'SAL') income
FROM employees WHERE department_id IN (50, 80);

FIRST_NAME	expr1	LAST_NAME	expr2	RESULT
William	7	Gietz	5	7
Shelley	7	Higgins	7	
Pat	3	Fay	3	
Michael	7	Hartstein	9	7
Jennifer	8	Whalen	6	8
Kimberely	9	Grant	5	9
Jonathon	8	Taylor	6	8
Ellen	5	Abel	4	5
Eleni	5	Zlotkey	7	5
Peter	5	Vargas	6	5
Randall	7	Matos	5	7
Curtis	6	Davies	6	
Trenna	6	Rajs	4	6
Kevin	5	Mourgos	7	5

20 rows selected.

Using the NULLIF Function

SELECT first_name, LENGTH(first_name) "expr1",
last_name, LENGTH(last_name) "expr2",
NULLIF(LENGTH(first_name), LENGTH(last_name)) result
FROM employees;

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)

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 names, 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.

Practice 2, Part 2

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.

Gietz earns $8,300.00 monthly but wants $24,900.00.

20 rows selected.

Gietz	$$$$$$$$$$8300

20 rows selected.

Practice 3, Part 2 (continued)

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 . 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 .

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 similar to "Monday, the Thirty-First of July, 2000."

EMPLOYEE_AND_THEIR_SALARIES
King **********
Kochhar **********
De Haan **********
Hartstei **********
Higgins **********
Abel **********
Zlotkey **********
Hunold **********
Taylor **********
Gietz **********
Grant **********

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.

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.

12. Create a query that displays the employees' last names and indic ates 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.

13. Using the DECODE function, write a query that displays the grade of all employee s 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

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