

EXERCISE-6

Single Row Functions

Objective

- After the completion of will be able to do the following:
- Describe various in SQL.
 - Use character, in SELECT statement.
 - Describe the use

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.

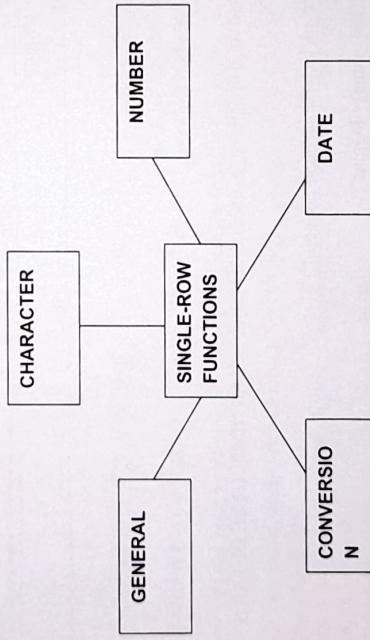
Accept arguments which can be a column or an expression

Syntax

Function_name(arg1,...,argn)

An argument can be one of the following

- ✓ User-supplied constant
- ✓ Variable value
- ✓ Column name
- ✓ Expression



- Character Functions: Accept character input and can return both character and number values.

- Number Functions: Accept numeric input and return numeric values.
- Date Functions: Operate on values of the DATE data type.
- Conversion Functions: Convert a value from one type to another.

Character Functions

Character Functions

Character-manipulation functions

1. Lower
2. Upper
3. Initcap

Character-manipulation functions

1. Concat
2. Substr
3. Length
4. Instr
5. Lpad Rpad
6. Trim
7. Replace

Function	Purpose
Lower(<i>alpha_expr</i>)	Converts alpha character values to lowercase
Upper(<i>alpha_expr</i>)	Converts alpha character values to uppercase
Initcap(<i>alpha_expr</i>)	Converts alpha character values the to uppercase for the first letter of each word, all other letters in lowercase
Concat(<i>char1_expr</i> , <i>char2_expr</i>)	Concatenates the first character to the second character
Substr(<i>string_expr</i> , <i>m</i> , <i>n</i>)	Returns specified characters from character value starting at character position m, n characters long
Length(<i>string_expr</i>)	Returns the number of characters in the expression
Instr(<i>string1_expr</i> , <i>string2</i>)	Returns the numeric position of a named string
Rpad(<i>char_expr</i> , <i>n</i> , <i>string</i>)	Pads the character value right-justified to a total width of n character positions
Lpad(<i>char_expr</i> , <i>n</i> , <i>string</i>)	Pads the character value left-justified to a total width of n character positions
Trim(<i>leading_trail_trail</i> , <i>char</i>) FROM (<i>table_name</i>)	Enables you to trim heading or string, trailing or both from a character
Replace(<i>string1</i> , <i>search_string</i> , <i>replacement_string</i>)	

Example:

```
lower('SQL Course') AS course
upper('SQL Course') AS SQL COURSE
initcap('SQL Course') AS Sql Course
```

```
SELECT 'The job is for' || upper(last_name) || 'is' || lower(job_id) AS "EMPLOYEE DETAILS"
FROM employees;
```

```
SELECT employee_id, last_name, department_id
FROM employees
WHERE LOWER(last_name)='Jiggins';
```

Function	Result
CONCAT('Hello', 'world')	HelloWorld
Substr('HelloWorld', 1, 5)	Hello
Length('HelloWorld')	10

Init('helloworld', 'w')	6
Lpad(salary, 10, '**')	*****24000
Rpad(salary, 10, '**')	24000****
Trim('h' FROM 'helloworld')	elloworld

Command	Query	Output
initcap(char);	select initcap('hello') from dual;	Hello
lower (char);	select lower ('HELLO') from dual;	Hello
upper (char);	select upper ('hello') from dual;	HELLO
ltrim (char,[set]);	select ltrim ('cseit', 'cse') from dual;	IT
rtrim (char,[set]);	select rtrim ('cseit', 'it') from dual;	CSE
replace (char,search string, replace string);	select replace ('jack and jue', 'j', 'bl') from dual;	black and blue
substr (char,m,n);	select substr ('information', 3, 4) from dual;	orm

Example:

```
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)= 'ERP';
```

NUMBER FUNCTIONS

Function	Purpose
round(column/expr, n)	Rounds the value to specified decimal
trunc(column/expr,n)	Truncates value to specified decimal
mod(m,n)	Returns remainder of division

Example

Function	Result
round(45.926,2)	45.93
trunc(45.926,2)	45.92
mod(1600,300)	100

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

NOTE: Dual is a dummy table you can use to view results from functions and calculations.

```
SELECT TRUNC(45.923,2), TRUNC(45.923), TRUNC(45.923,-2) FROM dual;
```

```
SELECT last_name, salary, MOD(salary, 5000) FROM employees WHERE job_id = 'sa_rep';
```

Working with Dates

The Oracle database stores dates in an internal numeric format: century, year, month, day, hours, minutes, and seconds.

- The default date display format is DD-MON-RR.
- Enables you to store 21st-century dates in the 20th century by specifying only the last two digits of the year
- Enables you to store 20th-century dates in the 21st century in the same way

Example

```
SELECT last_name, hire_date FROM employees WHERE hire_date < '01-FEB-88';
```

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.

Arithmetic with Dates

Because the database stores dates as numbers, you can perform calculations using arithmetic Operators such as addition and subtraction. You can add and subtract number constants as well as dates.

You can perform the following operations:

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

Example

```
SELECT last_name, (SYSDATE-hire_date)/7 AS WEEKS  
FROM employees  
WHERE department_id = 90;
```

Date Functions

Function	Result
MONTHS_BETWEEN	Number of months between two dates
ADD_MONTHS	Add calendar months to date
NEXT_DAY	Next day of the date specified
LAST_DAY	Last day of the month
ROUND	Round date
TRUNC	Truncate date

Date Functions

Date functions operate on Oracle dates. All date functions return a value of DATE data type except MONTHS_BETWEEN, which returns a numeric value.

- MONTHS_BETWEEN(date1, date2)::: Finds the number of months between date1 and date2. The result can be positive or negative. If date1 is later than date2, the result is positive; if date1 is earlier than date2, the result is negative. The noninteger part of the result represents a portion of the month.

- **ADD_MONTHS(date, n)**: Adds a number of calendar months to date. The value of n must be an integer and can be negative.
- **NEXT_DAY(date, 'char')**: Finds the date of the next specified day of the week ('char') following date. The value of char may be a number representing a day or a character string.
- **LAST_DAY(date)**: Finds the date of the last day of the month that contains date.
- **ROUND(date[,Tmt])**: Returns date rounded to the unit that is specified by the format model tmt. If the format model tmt is omitted, date is rounded to the nearest day.
- **TRUNC(date[,Tmt])**: Returns date with the time portion of the day truncated to the unit that is specified by the format model tmt. If the format model tmt is omitted, date is truncated to the nearest day.

Using Date Functions

Function	Result
HIRE_DATE_BETWEEN ('01-JAN-91', '31-JAN-94')	'31-07-93-14'
ADD_MONTHS ('31-JAN-94', 6)	'31-JUL-94'
NEXT_DAY ('01-JAN-97', 'FRIDAY')	'08-JAN-97'
LAST_DAY ('01-JUN-97')	'30-JUN-97'

Example

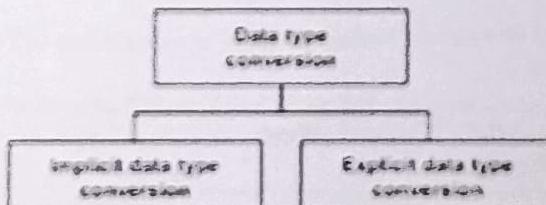
Display the employee number, hire date, number of months employed, maximum tenure date, last Friday after hire date, and last day of the hire month for all employees who have been employed for fewer than 70 months.

```
SELECT employee_id, hire_date, MONTHS_BETWEEN(SYSDATE, hire_date)
TENURE, ADD_MONTHS(hire_date, 6) REVIEW, NEXT_DAY(hire_date, FRIDAY),
LAST_DAY(hire_date)
FROM employees
WHERE MONTHS_BETWEEN(SYSDATE, hire_date) < 70;
```

Conversion Functions

This covers the following topics:

- Writing a query that displays the current date
- Creating queries that require the use of numeric, character, and date functions
- Performing calculations of years and months of service for an employee



Find the Solution for the following:

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

Select current_date as date from dual;

2. The HR department needs a report to display the employee number, last name, salary, and increased by 15.5% (expressed as a whole number) for each employee. Label the column New Salary.

Select employee_id, last_name, salary, round(salary * 1.155)
as "newSalary" from employees;

3. Modify your query lab_03_02.sql to add a column that subtracts the old salary from the new salary. Label the column Increase

Select employee_id, last_name, salary, round(salary * 1.155)
as "newSalary", round(salary * 1.155) - salary as increase
from employees;

4. Write a query that displays the last name (with the first letter uppercase and all other letters lowercase) and the length of the last name for all employees whose name starts with the letters J, A, or M. Give each column an appropriate label. Sort the results by the employees' last names.

Select initcap(last_name) as "Last Name", length(last_name)
as "Length" from employees where upper(substr(last_name, 1, 1))
in ('J', 'A', 'M') order by last_name;

5. Rewrite the query so that the user is prompted to enter a letter that starts the last name. For example, if the user enters H when prompted for a letter, then the output should show all employees whose last name starts with the letter H.

Select employee_id, last_name, salary from employees
where upper(substr(last_name, 1, 1)) = upper('& letter');

6. The HR department wants to find the length of employment for each employee. For each employee, display the last name and calculate the number of months between today and the date on which 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.

Select last_name, Round (months_between (current_date) - hire_date) as months_worked from employees order by months_worked;

Note: Your results will differ.

7. Create a report that produces the following for each employee:

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

Select concat (last_name, ' earns ', salary, ' monthly but wants ', salary * 3) as "Dream Salaries" from employees;

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 the \$ symbol. Label the column SALARY.

Select last_name, LPAD (concat ('\$', salary, ' ', '')) as 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."

Select last_name, hiredate, to_char (NEXT_DAY (ADD_MONTHS (hiredate, 6), 'Monday'), 'the DD th of month, YYYY') as 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.

Select last_name, hire_date, to_char(hire_date, 'Day') as Day from employees order by to_char(hire_date, 'D');

Evaluation Procedure	Marks awarded
Query(5)	5
Execution (5)	5
Viva(5)	5
Total (15)	15
Faculty Signature	D 19/25

Practice Questions

Introduction to Functions

1. For each task, choose whether a single-row or multiple row function would be most appropriate:
 - a. Showing all of the email addresses in upper case letters

Single row

- b. Determining the average salary for the employees in the sales department

multiple row

- c. Showing hire dates with the month spelled out (*September 1, 2004*)

single row

- d. Finding out the employees in each department that had the most seniority (the earliest hire date)

multiple row

- e. Displaying the employees' salaries rounded to the hundreds place

single row

- f. Substituting zeros for null values when displaying employee commissions.

single row:

2. The most common multiple-row functions are: AVG, COUNT, MAX, MIN, and SUM. Give your own definition for each of these functions.

AVG - calculates the average for a column

COUNT - counts the number of non-null values in a column

MAX - returns the highest value in a column across multiple rows

MIN - returns the lowest value in a column across multiple rows

SUM - Adds up all the values in a column across multiple rows.

3. Test your definitions by substituting each of the multiple-row functions in this query.

SELECT FUNCTION(salary)

FROM employees

Write out each query and its results.

Select Avg(salary) as average_salary from employees;

Select count(salary) as ~~total~~ total_employees from employees;

Select max(salary) as highest_salary from employees;

Select min(salary) as lowest_salary from employees;

Select sum(salary) as total_salary from employees;

Case and Character Manipulation

1. Using the three separate words "Oracle," "Internet," and "Academy," use one command to produce the following output:
The Best Class Oracle Internet Academy

Select upper ('oracle Internet Academy') from dual;

2. Use the string "Oracle Internet Academy" to produce the following output:
The Net net

Select initcap('Oracle internet academy') from dual;

3. What is the length of the string "Oracle Internet Academy"?

Select length ('oracle internet academy') From dual;

4. What's the position of "I" in "Oracle Internet Academy"?

Select instr('oracle internet academy', 'I') from dual;

5. Starting with the string "Oracle Internet Academy", pad the string to create
****Oracle****Internet****Academy****

Select Substr('oracle internet academy', 8) from dual;

Number Functions

1. Display Oracle database employee last_name and salary for employee_ids between 100 and 102. Include a third column that divides each salary by 1.55 and rounds the result to two decimal places.

Select last_name, salary round (salary / 1.55, 2) as adjusted salary from employees where employee_id between 100 and 102;

2. Display employee last_name and salary for those employees who work in department 80. Give each of them a raise of 5.333% and truncate the result to two decimal places.

Select last_name, salary, trunc (salary * 1.053333, 2) as raised_salary from employees where department_id = 80;

3. Use a MOD number function to determine whether 38873 is an even number or an odd number.

Select mod (38873, 2) as result from dual;

4. Use the DUAL table to process the following numbers:

845.553 - round to one decimal place

Select round (845.553, 1) as rounded_1 from dual;

30695.348 - round to two decimal places

Select round (30695.348, 2) as round_2 from dual;

30695.348 - round to -2 decimal Places

Select round (30695.348, -2) as round_imprecise from dual;

2.3454 - truncate the .454 from the decimal place

Select trunc (2.3454, 2) as truncated_2 from dual;

5. Divide each employee's salary by 3. Display only those employees' last names and salaries who earn a salary that is a multiple of 3.

Select last_name, salary from employees where mod (salary, 3) = 0;

6. Divide 34 by 8. Show only the remainder of the division. Name the output as EXAMPLE.

Select mod (34, 8) as example from dual;

7. How would you like your paycheck – rounded or truncated? What if your paycheck was calculated to be \$565.784 for the week, but you noticed that it was issued for \$565.78. The loss of .004 cent would probably make very little difference to you. However, what if this was done to a thousand people, a 100,000 people, or a million people! Would it make a difference then? How much difference?

$$0.004 \times 1000 = 4 \text{ dollars}$$

$$0.004 \times 100,000 = 400 \text{ dollars}$$

$$0.004 \times 1,100,000 = 4,000 \text{ dollars}$$

Evaluation Procedure	Marks awarded
Practice Evaluation (5)	5
Viva(5)	5
Total (10)	10
Faculty Signature	RPL 8/10