WEEK 4: READING MATERIAL 3

Functions in SQL*PLUS

Functions are programs that take zero or more arguments and return a single value. Oracle has built a number of functions into SQL, and these functions can be called from SQL or PL/SQL statements (detail in next chapters).

Need of Functions

Functions can be used for the following purposes:

- To Perform Data calculations.
- To make modification of data.
- To manipulate data for desired output.
- To converting data values from one type to another.

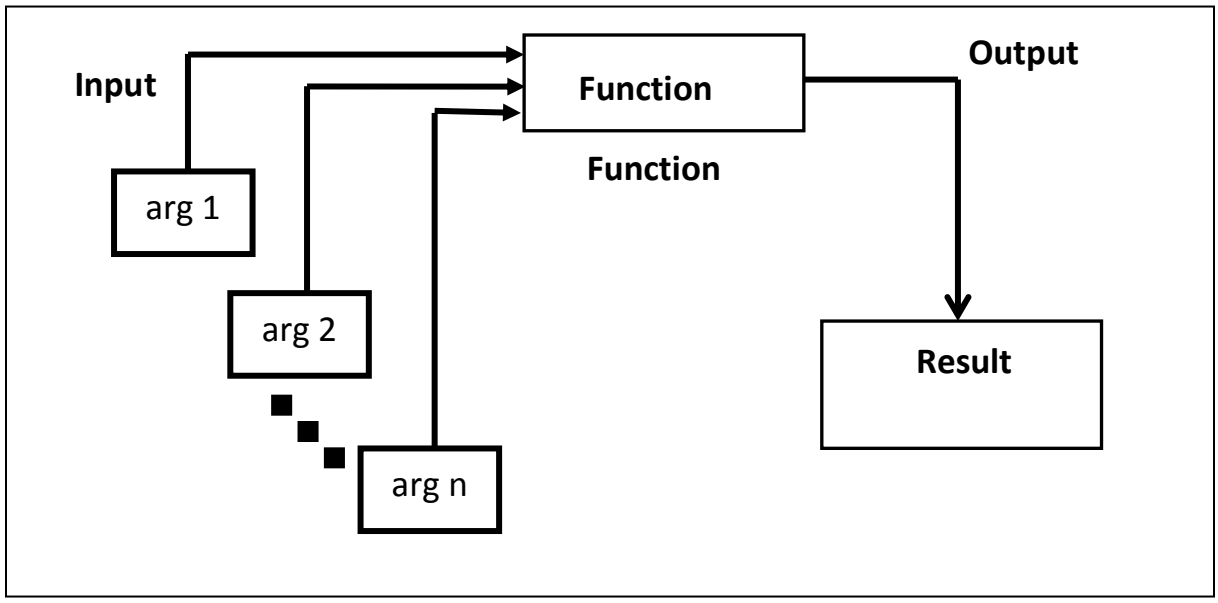

Figure 5.1: Basic functionality of function

Classes of Functions

There are two significant classes of functions as shown in figure 5.2.

- Single-row functions
- Group functions (also known as aggregate functions)

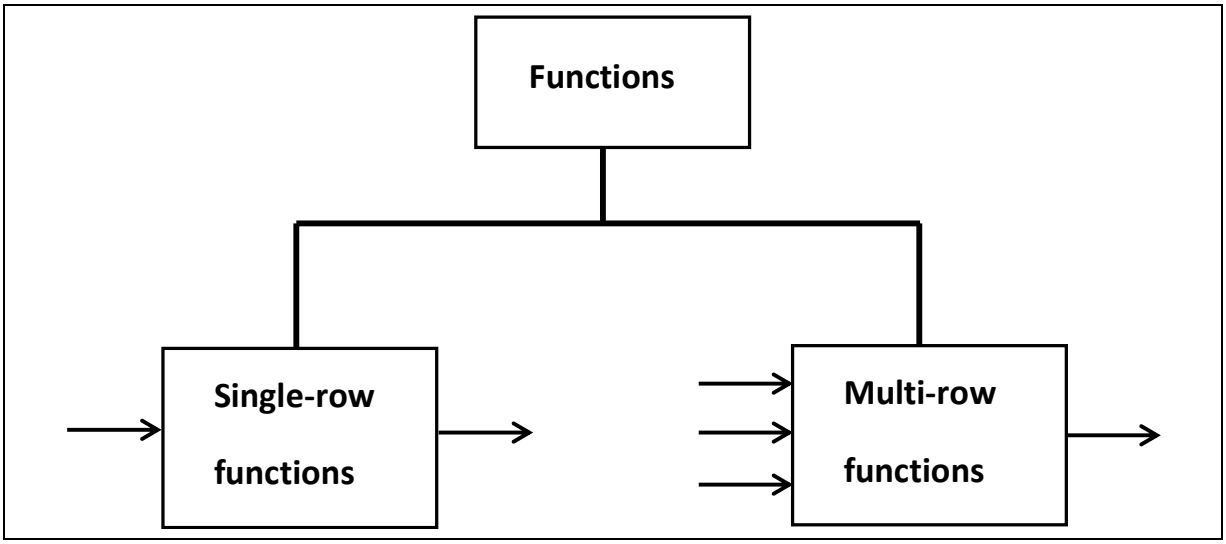

Figure 5.2: Classes of functions

Single-row functions know how many arguments they will have to process before data is fetched from the tables. Group functions don't know how many arguments they will have to process until all the data is extracted and grouped into categories.

Single Row Functions

Single Row Functions act on each row returned by the query. It result one result per row**.**

Classification of Single Row Functions

Single Row Functions can be classified into the following categories as shown in figure 5.3.

(i) Character

(ii) Number

(iii) Date

(iv) Conversion

(v) General

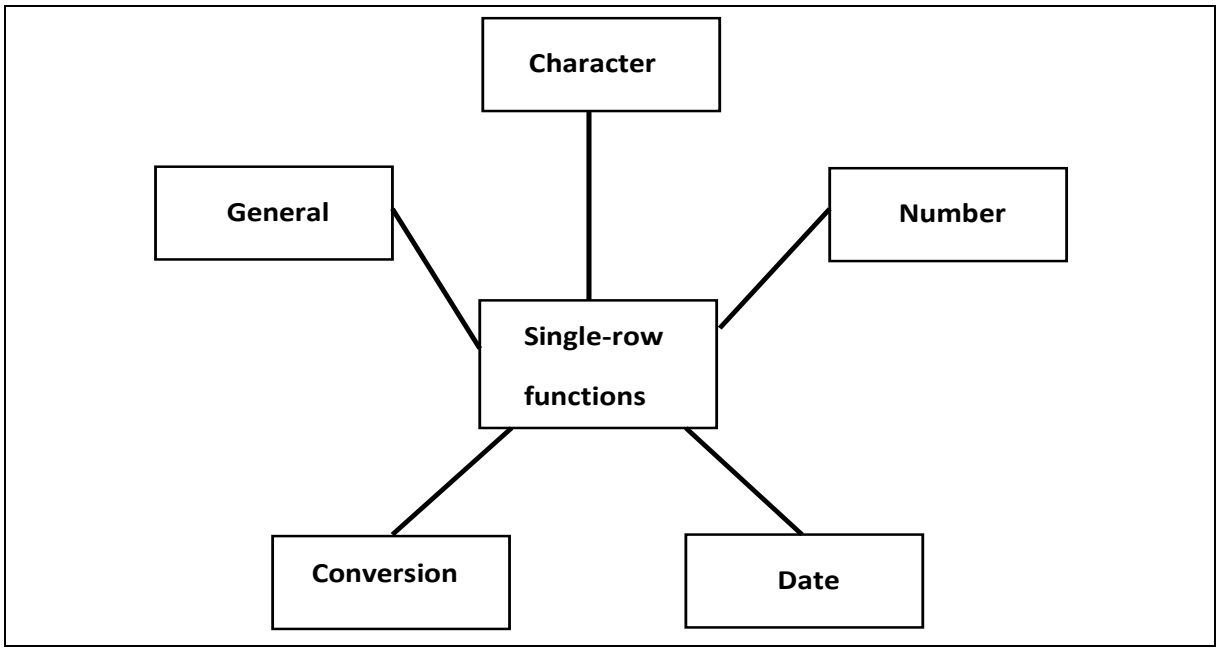

Figure 5.3: Classification of single row functions

Single-Row Character Functions

Single-row character functions operate on character data. Most have one or more character arguments, and most return character values. The classification of single row character is shown in figure 5.4.

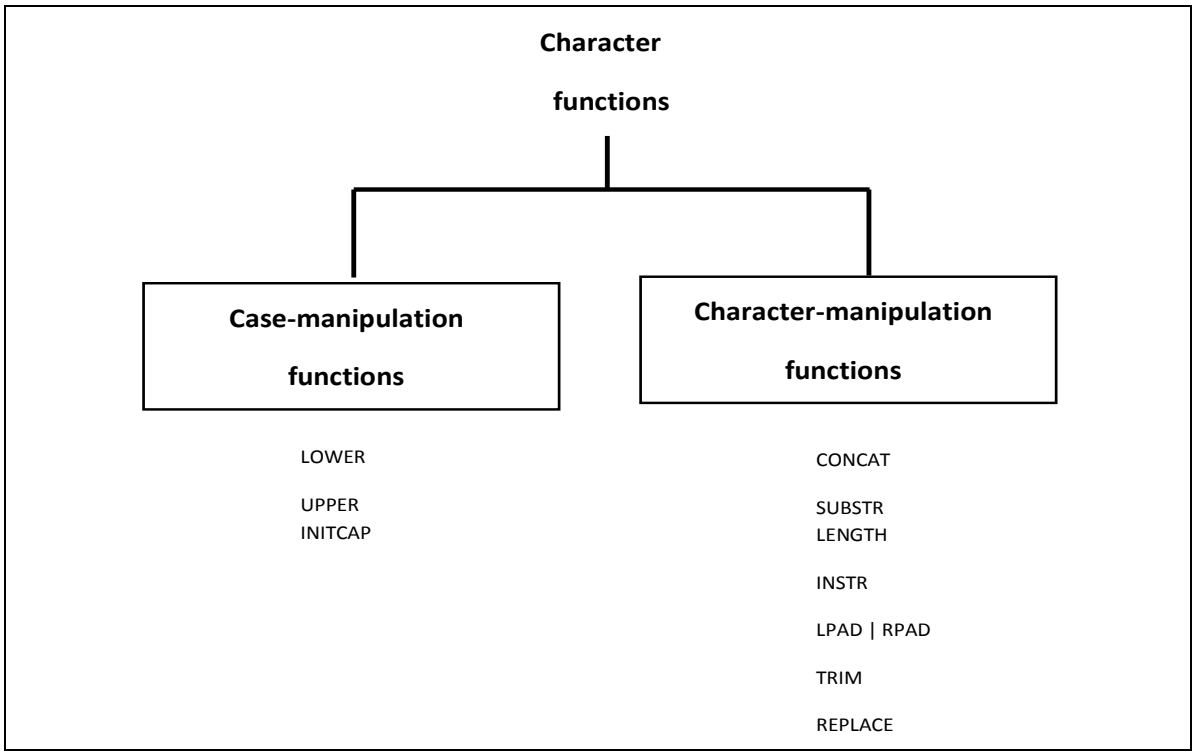

Figure 5.4: Classification of character fonctions

CONCAT(<*c1*>, <*c2*>)

Where *c1* and *c2* are character strings. This function returns *c2* appended to *c1*. If *c1* is NULL, then *c2* is returned. If *c2* is NULL, then *c1* is returned. If both *c1* and *c2* are NULL, then NULL is returned. CONCAT returns the same results as using the concatenation operator: *c1*||*c2*.

SQL>*SELECT CONCAT ('dav ','college') CollgeName FROM dual ;*

OUTPUT:

COLLEGENAME

---

dav college

ASCII(<*c1*>)

Where *c1* is a character string. This function returns the ASCII decimal equivalent of the first character in *c1*. See also CHR() for the inverse operation.

SQL>*SELECT ASCII ('A') Big_A, ASCII('z') Little_Z FROM dual;*

OUTPUT:

BIG_A	LITTLE_Z
65	122

CHR(x)

This function gives the result as a character corresponding to the value x in the character set.

SQL>*SELECT CHR(97) first , chr(99) second from dual ;*

OUTPUT:

FIRST	SECOND
a	c

INITCAP (<*c1*>)

Where *c1* is a character string. This function returns *c1* with the first character of each word in uppercase and all others in lowercase.

SQL>*SELECT INITCAP ('amit', 'raju', 'rajesh') name FROM dual;*

OUTPUT:

NAME

Amit, Raju, Rajesh

LENGTH (<*c*>)

Where *c* is a character string. This function returns the numeric length in characters of *c*. If *c* is NULL, a NULL is returned.

SQL> *SELECT LENGTH ('dav college') name FROM dual;*

OUTPUT:

NAME

11

LOWER (<*c*>)

Where *c* is a character string. This function returns the character string *c* with all characters in lowercase. It frequently appears in WHERE clauses. See also UPPER.

SQL>*select lower(dname) from dept ;*

OUTPUT:

lower(dname)
accounting
Research
Sales
operations

LPAD(<*c1*>, <*i*> [,<*c2*>])

Where *c1* and *c2* are character strings and *i* is an integer. This function returns the character string *c1* expanded in length to *i* characters using *c2* to fill in space as needed on the left-hand side of *c1*. If *c1* is over *i* characters, it is truncated to *i* characters. *c2* defaults to a single space. See also RPAD.

Example:

SQL> *select lpad(dname,12,' '),lpad(dname,12,'*') from dept ;*

OUTPUT:

lpad(dname,12,' ')	lpad(dname,12,'* ')
ACCOUNTING	**ACCOUNTING
RESEARCH	****RESEARCH
SALES	*******SALES
OPERATIONS	**OPERATIONS

LTRIM (<*c1*>, <*c2*>)

Where *c1* and *c2* are character strings. This function returns *c1* without any leading characters that appear in *c2*. If no *c2* characters are leading characters in *c1*, then *c1* is returned unchanged. *c2* defaults to a single space. See also RTRIM.

SQL> *SELECT LTRIM ('dav','d') FROM dual;*

OUTPUT:

```
LT
--
av
```

RPAD (<*c1*>, <*i*>[, <*c2*>])

Where *c1* and *c2* are character strings and *i* is an integer. This function returns the character string *c1* expanded in length to *i* characters using *c2* to fill in space as needed on the right-hand side of *c1*. If *c1* is over *i* characters, it is truncated to *i* characters. *c2* defaults to a single space. See also LPAD.

SQL> *SELECT RPAD (table_name, 38, '.'), num_rows FROM user_tables;*

OUTPUT:

RPAD(TABLE_NAME,38, '.')	NUM_ROWS
TEMP_ERRORS.....	9
CUSTOMERS.....	367,296

RTRIM(<*c1*>,<*c2*>)

Where *c1* and *c2* are character strings. This function returns *c1* without any trailing characters that appear in *c1*. If no *c2* characters are trailing characters in *c1*, then *c1* is returned unchanged. *c2* defaults to a single space. See also LTRIM.

SQL>*SELECT RTRIM ('Mississippi', 'ip') FROM dual;*

OUTPUT:

RTRIM('
Mississ

REPLACE (<*c1*>, <*c2*>[, <*c3*>])

Where *c1, c2* and *c3* are all characters string. This function returns *c1* with all occurrences of *c2* replaced with *c3*. *c3* defaults to NULL. If *c3* is NULL, all occurrences of *c2* are removed. If *c2* is NULL, then *c1* is returned unchanged. If *c1* is NULL, then NULL is returned.

SQL>*SELECT REPLACE ('uptown', 'up', 'down') FROM dual;*

OUTPUT:

REPLACE (
downtown

SUBSTR (<*c1*>, <*i*>[,<*j*>])

Where *c1*is a character string and both *i* and *j* are integers. This function returns the portion of *c1* that is *j* characters long, beginning at position *i*. If *j* is negative, the position is counted backwards (that is, right to left). This function returns NULL if *i* is 0 or negative. *j* defaults to 1.

SQL>*SELECT SUBSTR ('Message', 1,4) from dual;*

OUTPUT:

SUBS

Mess

TRANSLATE(<*c1*>, <*c2*>, <*c3*>)

Where *c1, c2,* and *c3* are all character string. This function returns *c1* with all occurrences of character in *c2* replaced with the position ally corresponding character in *c3*. A NULL is returned if any of *c1, c2,* or *c3* is NULL. If *c3* has fewer characters than *c2*, then the unmatched characters in *c2* are removed from *c1*. If *c2* has fewer characters than *c3*, then the unmatched characters in *c3* are ignored.

SQL>*SELECT TRANSLATE ('fumble', 'uf', 'aR') test FROM dual;*

OUTPUT:

TEST

Ramble

TRIM (string[,char(s)])

It removes all the blank spaces from the left as well as right side of the string if no char is specified. If we give an char, then it removes the leading and trailing occurrences of that character from the string. This function is new to 8i.

SQL>*SELECT TRIM(' space padded ') trimmed FROM dual;*

OUTPUT:

TRIMMED

---

space padded

UPPER (<*c*>)

Where *c* is a character string. This function returns the character string *c* with all characters in upper case. UPPER frequently appears in WHERE clauses.

Example:

SQL>*select upper(dname) from dept ;*

OUTPUT:

UPPER(DNAME)
ACCOUNTING
RESEARCH
SALES
OPERATIONS

Character Function Summary

Function	**Description**
ASCII	Returns the ASCII decimal equivalent of a character
CHR	Returns the character given the decimal equivalent
CONCAT	Concatenates two string; same as the operator ||
INITCAP	Returns the string with the first letter of each word in uppercase
LENGTH	Returns the length of a string in characters
LOWER	Converts string to all lowercase
LPAD	Left-fills a string to a set length using a specified character
LTRIM	Strips leading characters from a string
RPAD	Right-fills a string to a set length using a specified character
RTRIM	Strips trailing characters from a string
REPLACE	Performs substring search and replace
SUBSTR	Returns a section of the specified string, specified by numeric character positions
TRANSLATE	Performs character search and replace
TRIM	Strings leading, trailing, or both leading and trailing characters from a string
UPPER	Converts string to all uppercase