OPERATORS IN SQL*PLUS

ARITHMETIC OPERATORS:+,-,*, /

COMPARISON OPERATORS: Used to compare one expression to another. They are =,!=,>=,<=,<,>,

BETWEEN ..AND (to check between any two values),

IN (to match with any of the values in the list),

LIKE (to match a character pattern) and **IS NULL**(to check whether a value in a tuple is null),

NOT BETWEEN,NOT LIKE and so on.

LOGICAL OPERATORS: Used to combine the results of two or more conditions to produce a single result. The logical operators are: OR, AND, NOT.

Operator Precedence

- Arithmetic operators-Highest precedence
- Comparison operators
- NOT operator
- AND operator
- OR operator----Lowest precedence
- The order of precedence can be altered using parenthesis.

CHARACTER FUNCTIONS: Character functions accept a character input and return either character or number values. Some of them supported by Oracle are listed below

FUNCTION	INPUT	OUTPUT
Initcap(char)	SQL>select initcap('hello') from dual;	Hello
Lower(char)	SQL>select lower('FUN') from dual;	fun
Upper(char)	SQL>select upper('sun') from dual;	SUN
Ltrim(char, set)	SQL>select ltrim('xyzhello','xyz')from dual;	hello
Rtrim(char, set)	SQL>select rtrim('xyzhello','llo')from dual;	xyzhe
translate(char,from,to)	SQL>select translate('jack','j','b') from dual;	back
Replace(char,	SQL>select replace('jack and jue',' j', 'bl')	black and blue

- **Lpad** is a function that takes three arguments. The first argument is the character string which has to be displayed with the left padding. The second is the number which indicates the total length of the return value, the third is the string with which the left padding has to be done when required.

Ex: **SQL>**select lpad (‘function’, 15,’*’) Lpd from dual;

Lpd

*******function

Ex: **SQL>**select rpad (‘function’, 15,’*’) Rpd from dual;

Rpd

function*******

Length: returns the length of a string

Ex: **SQL>**select length (‘VIT’) from dual;

(Displays 3).

Concatenation || operator: it is used to merge or more strings.

SQL>select Ename|| 'works as'|| job from employee;

NUMERIC FUNCTIONS: Numeric functions accept numeric input and returns numeric values as output.

FUNCTION	INPUT	OUTPUT
Abs(n)	SQL>select abs(-15) from dual	15
Ceil(n)	SQL>select ceil(48.778) from dual;	49
Cos(n)	SQL>select cons(180) from dual;	-0.59884601
Cost(n)	SQL>select cushy(0) from dual;	1
Exp(n)	SQL>select exp(4) from dual;	54.59815
Floor(n)	SQL>select floor(4.678) from dual;	4
Power(m ,n)	SQL>select power(5,2) from dual;	25
Mod(m ,n)	SQL>select mod(11,2) from dual;	1
Round(m ,n)	SQL>select round(112.257,2) from dual;	112.26

SQL> select ln (2) from dual; (returns natural logarithm value of 2)

SQL>select sign (-35) from dual; (output is -1)

CONVERSION FUNCTIONS: Convert a value from one data type to another.

To__char ( ): To_ char (d [,fmt]) where d is the date fmt is the format model which specifies the format of the date. This function converts date to a value of varchar2datatype in a form specified by date format fmt.if fmt is neglected then it converts date to varchar2 in the default date format.

Ex: SQL>select to_ char (sys_ date, 'ddth "of" fmmonth yyyy') from dual;

Output is 18th of December 2007.

To_ date ( ): The format is to_date (char [, fmt]). This converts char or varchar data type to date data type. Format model, fmt specifies the form of character.

Ex: **SQL>**select to_date ('December 18 2007','month-dd-yyyy') from dual;

18-DEC-07 is the output.

SQL>select round (to_date ('27-dec-98'),'year') from dual;

To_ Number( ): allows the conversion of string containing numbers into the number data type on which arithmetic operations can be performed.

Ex: **SQL>**select to_number ('100') from dual;

DATE FUNCTIONS

Function Name	Return Value
ADD_MONTHS (date, n)	Returns a date value after adding *'n'* months to the date *'x'*.
MONTHS_BETWEEN (x1, x2)	Returns the number of months between dates x1 and x2.
ROUND (x, date_format)	Returns the date *'x'* rounded off to the nearest century, year, month, date, hour, minute, or second as specified by the *'date_format'*.
TRUNC (x, date_format)	Returns the date *'x'* lesser than or equal to the nearest century, year, month, date, hour, minute, or second as specified by the 'date_format'.
NEXT_DAY (x, week_day)	Returns the next date of the *'week_day'* on or after the date *'x'* occurs.
LAST_DAY (x)	It is used to determine the number of days remaining in a month from the date *'x'* specified.
SYSDATE	Returns the systems current date and time.

Examples of Date Functions

Function Name	Examples	Return Value
ADD_MONTHS ( )	ADD_MONTHS ('16-Sep-81', 3)	16-Dec-81
MONTHS_BETWEEN( )	MONTHS_BETWEEN ('16-Sep-81', '16-Dec-81')	3
NEXT_DAY( )	NEXT_DAY ('01-Jun-08', 'Wednesday')	04-JUN-08
LAST_DAY( )	LAST_DAY ('01-Jun-08')	30-Jun-08