Oracle Built in Functions.

There are two types of functions in Oracle.

1) Single Row Functions: Single row or Scalar functions return a value for every row that is processed in a query.

2) Group Functions: These functions group the rows of data based on the values returned by the query. This is known as SQL GROUP Functions. The group functions are used to calculate aggregate values like total or average, which return just one total or one average value after processing a group of rows.

There are four types of single row Or scalar functions.:

1) **Numeric Functions:** These are functions that accept numeric input and return numeric values.

2) **Character or Text Functions:** These are functions that accept character input and can return both character and number values.

3) **Date Functions:** These are functions that take values that are of datatype DATE as input and return values of datatype DATE, except for the MONTHS_BETWEEN function, which returns a number.

4) **Conversion Functions:** These are functions that help us to convert a value in one form to another form. For Example: a null value into an actual value, or a value from one datatype to another datatype like NVL, TO_CHAR, TO_NUMBER, TO_DATE etc.

You can combine more than one function together in an expression. This is known as nesting of functions.

1) Numeric Functions:

Numeric functions are used to perform operations on numbers. They accept numeric values as input and return numeric values as output. Few of the Numeric functions are:

The following examples explains the usage of the above numeric functions

- ABS( )

  Returns the absolute value of a number.

Syntax:-	ABS(<value>
Example:-	SELECT ABS(-100) FROM DUAL;
Output :-	100

- CEIL( )

  Smallest integer greater than or equal to a decimal value.

Syntax :-	CEIL(<value>)
Example:-	SELECT CEIL(12345.67) FROM DUAL;
Output :-	12346

- COS()

  Returns the cosine of a number (an angle expressed in radians)

Syntax :-	COS(<value>)
Example:-	SELECT COS(180*3.1415926/180) COSINE FROM Dual;
Output :-	1

- DECODE()

  The Oracle decode statement was developed to allow us to transform data values at retrieval time. For example, say we have a column named REGION, with values of N, S, W and E. When we run SQL queries, we want to transform these values into North, South, East and West. Here is how we do this with the decode function:

select decode (region,'N','North','S','South','E','East','W','West','UNKNOWN') from customer;

- EXP()

  Returns e raised to to an exponential power

```
SELECT 2.71828183 * 2.71828183 FROM DUAL;
SELECT EXP(2) FROM DUAL;
SELECT 2.71828183 * 2.71828183 * 2.71828183 FROM DUAL;
```

- FLOOR()

  Returns the largest integer less than or equal to a decimal value

```
FLOOR(<string_or_column>)
SELECT FLOOR(12345.67) FROM DUAL;
```

- GREATEST()

  Returns the largest of multiple values

```
GREATEST(<value>, <value>, ....)
SELECT GREATEST(9, 67.6, 10) FROM DUAL;
```

- LEAST()

  Returns the smallest of multiple values

```
LEAST(<value>, <value>, ....)
SELECT LEAST(9, 67.6, 10) FROM DUAL;
```

- LOG()

  Returns the logarithm, base m of n

```
SELECT LOG(10,100) FROM DUAL;
SELECT LOG(100,10) FROM DUAL;
```

- LOG10()

  LOG10 is used to calculate base-10 logarithm for the specified numeric value

```
LOG10 (float_expression)
SELECT LOG10(200) from dual;
```

- MAX()

  Returns the maximum value returned by a query

```
MAX(<column_name>)
SELECT MAX(initial_extent) FROM all_tables;
```

- MIN()

  Returns the minimum value returned by a query

```
MIN(<column_name>)
SELECT MIN(initial_extent) FROM all_tables;
```

- REMAINDER()

  Returns the modulus of a number


```
REMAINDER(<m_value>, <n_value>)
SELECT REMAINDER(2,3) FROM DUAL
```

- **ROUND()**

  Returns a value rounded to integer places

```
ROUND(<value>, <integer>)
SELECT ROUND(3.1415926, 4) FROM DUAL;
```

- **SIGN()**

  Returns the sign of a number

```
SIGN(<value>)
SELECT SIGN(15) FROM DUAL;
SELECT SIGN(0) FROM DUAL;
SELECT SIGN(-5) FROM DUAL;
```

- **SIN()**

  Returns the sine of a number

```
SIN(<value>)
SELECT SIN(2) SINE FROM DUAL;
```

- **SINH()**

  Returns the hyperbolic sine of a number

```
SINH(<value>)
SELECT SINH(2) HYPERBOLIC_SINE FROM DUAL;
```

- **SQRT()**

  Returns the square root of a number

```
SQRT(<value>)
SELECT SQRT(2) FROM DUAL;
```

- **TAN()**

  Tangent in radians

```
TAN(<value>)
SELECT TAN(135 * 3.14159265359/180) FROM DUAL;
```

- **TRUNC()**

  Truncates a Number or a Datetime to the Specified Number of Decimal Places.
  Do not confuse this with the TRUNCATE function

```
RUNC(<value>, <decimal_places>) RETURN NUMBER;
SELECT TRUNC(15.79, 1) FROM DUAL;

SELECT TRUNC(15.79, -1) FROM DUAL;
```

  These functions can be used on database columns.

2) Character or Text Functions:

Character or text functions are used to manipulate text strings. They accept strings or characters as input and can return both character and number values as output.
Few of the character or text functions are as given below:

- ***Chr()***

The Oracle CHR() function returns the ASCII character that corresponds to the value passed to it. If using NCHAR_CS is specified, it returns the character that corresponds to the national character set.

```
SELECT CHR(195)||CHR(193)||CHR(227) "output" FROM DUAL;
```

- ***Concat()***

  The Oracle/PLSQL CONCAT function allows you to concatenate two strings together.
  Sy.

```
  CONCAT( string1, string2 )
Select concat ('I','Love','Country') from dual;
```

- ***Initcap()***

INITCAP returns char, with the first letter of each word in uppercase, all other letters in lowercase. Words are delimited by white space or characters that are not alphanumeric.

```
SELECT INITCAP('the soap') "Capitals" FROM DUAL;
```

- ***Upper()***

SQL UPPER function return uppercase character in every word.

```
SELECT UPPER('me and mine') "UPPER" FROM DUAL;
```

- ***Lower()***

SQL LOWER function return lowercase character in every word.

```
SELECT LOWER('ME AND MINE') "LOWER" FROM DUAL;
```

- ***Lpad()***

The Oracle LPAD() function is used to padding the left side of a string with a specific set of characters. The function is useful for formatting the output of a query.

```
SELECT LPAD('Oracle',10,'+') FROM DUAL;
```

- ***Ltrim()***

The LTRIM function removes characters from the left of a text expression, with all the leftmost characters that appear in another text expression removed.

```
SELECT LTRIM('Z' FROM 'Zebra') FROM dual;
```

- ***Replace()***

Replace function replaces the string with another string if it matches. It is a character function in Oracle. Replace function is case-sensitive.

```
REPLACE('input string','find_string','replace_string');
SELECT REPLACE('JACK and JUE','J','BL') "Changes" FROM DUAL;
```

```
  SELECT REPLACE('MAN and MAT','M','F') "New String"  FROM DUAL;
```

- ***Rpad()***

The RPAD function returns an expression, right-padded to a specified length with the specified characters

```
SELECT RPAD('Honey', 8, '-') FROM dual;
```

- ***Rtrim()***

The RTRIM or Right Trim function returns char, with all the rightmost characters that appear in set removed; set defaults to a single blank. If char is a character literal, you must enclose it in single quotes. RTRIM works similarly to LTRIM.

```
SELECT RTRIM('days', 's') FROM dual;
select ename, rtrim(ename,'N') from emp;
```

- ***soundex()***

SOUNDEX function help you to find words matching phonetically (By pronunciation)
SOUNDEX is very useful for finding similar pronouncing names. For example say I need to find all employee sound "Daniel".

```
Syntax :- SOUNDEX(column_name)
select empno,empname from emp where soundex(empname) = soundex('daniyal');
```

- ***substr()***

The Oracle SUBSTR (SUBSTRING) function lets you extract a portion of a string (a 'substring') from a string of characters.

1. If the *starting_position* is 0, then SUBSTR treats start_position as 1 (ie: the first position in the string).
2. If the *starting_position* is a positive number, then SUBSTR starts from the beginning of the string.
3. If the *starting_position* is a negative number, then SUBSTR starts from the *end* of the source_string and counts backwards.


```
Syntax:-SUBSTR(source_string, starting_position, [ length ]);
SUBSTR('Dinner starts in one hour.', 8, 6)    - will return 'starts'
```

SUBSTR('Dinner starts in one hour.', 8) - will return 'starts in one hour.'

SUBSTR('Dinner starts in one hour.', 1, 6) - will return 'Dinner'

SUBSTR('Dinner starts in one hour.', 0, 6) - will return 'Dinner'

SUBSTR('Dinner starts in one hour.', -4, 3) - will return 'our'

SUBSTR('Dinner starts in one hour.', -9, 3) - will return 'one'

SUBSTR('Dinner starts in one hour.', -9, 2) - will return 'on'
Assume a double-byte database character set.
SHOW SUBSTRB('abcdefg',5,4.2)

- *Trim()*
  The Oracle/PLSQL TRIM function removes all specified characters either from the beginning or the end of a string.

```
SELECT TRIM('   removing leading and trailing white spaces    ') FROM DUAL
SELECT TRIM('   removing leading and trailing white spaces    ') FROM DUAL
```

Syntax

TRIM([[LEADING | TRAILING | BOTH] trim_character FROM] string1)

3) Date Functions:

These are functions that take values that are of datatype DATE as input and return values of datatypes DATE, except for the MONTHS_BETWEEN function, which returns a number as output.
Few date functions are as given below.
The below table provides the examples for the above functions.

- **add_months ()**
  The add_months Oracle date function gives you the same day, n number of months away. The n can be positive or negative.

```
Syntax:-Add months(d,n)
Example:-       Select sysdate,add_months(sysdate,1) from dual;
                Select sysdate,add_months('24-aug-17',2) from dual;
```

- **last_day()**
  The last_day Oracle date function returns the last day of the month of the date d. If you want to find the first day of the next month, simply add one to the last_day results.

```
Syntax :- last_day(date)
Example :-Select sysdate,last_day(sysdate) EOM,last_day (sysdate) + 1 FOM from dual;
```

- **months_between (l,e)**
  The MONTHS_BETWEEN function calculates the number of months between two dates. When the two dates have the same day component or are both the last day of the month, then the return value is a whole number.

```
                Syntax :-
MONTHS_BETWEEN(datetime_expression1, datetime_expression2)
        Example :-  select MONTHS_BETWEEN ('31-JAN-2014', '28-FEB-2014')from dual
    select MONTHS_BETWEEN ('31-MAR-2013', '28-FEB-2013') from dual
```

- **LAST_DAY(date)**
  Returns the last day in the month of the specified date .

```
Syntax :-   LAST_DAY(date)
Example:- select sysdate, last_day(sysdate) as last_day_curr_month ,last_day (sysdate) + 1 as first_day_next_month  from dual
```

- **NEXT_DAY(date, day_of_week): –**
  Returns the date of the first weekday specified that is later than the date.
  Example:-Select next_day(sysdate, 'monday') as next_Monday from dual;

- **Round (date)**
  The ROUND() function is used to get the date rounded to the unit specified by the format model.

```
SELECT ROUND(TO_DATE ('16-SEP-2015'),'MONTH')   "New Month",
ROUND(TO_DATE ('16-SEP-2015'),'YEAR')   "New Year"   FROM DUAL;
```

- **Sysdate()**
  Returns the current date and time set for the operating system.

```
Example:- SELECT SYSDATE FROM DUAL;
```

- **SYSTIMESTAMP**
  The Oracle/PLSQL SYSTIMESTAMP function returns the current system date and time (including fractional seconds and time zone) on your local database.

```
Example:- SELECT TO_CHAR(SYSTIMESTAMP, 'SSSS.FF') FROM dual;
```

- **Trunc(date)**
  The Oracle/PLSQL TRUNC function returns a date truncated to a specific unit of measure

```
SELECT TRUNC(TO_DATE('27-OCT-92','DD-MON-YY'), 'YEAR')"New Year" FROM DUAL;
```

- **TO_DATE()**

  The Oracle TO_DATE function will convert either a character string or an expression into a date value.

```
select to_date('2012/07/09', 'yyyy/mm/dd') from dual;
select to_date('070912', 'MMDDYY') from dual;
```

4) Conversion Functions:
These are functions that help us to convert a value in one form to another form. For Ex: a null value into an actual value, or a value from one datatype to another datatype like NVL, TO_CHAR, TO_NUMBER, TO_DATE.
Few of the conversion functions available in oracle are:
The below table provides the examples for the above functions