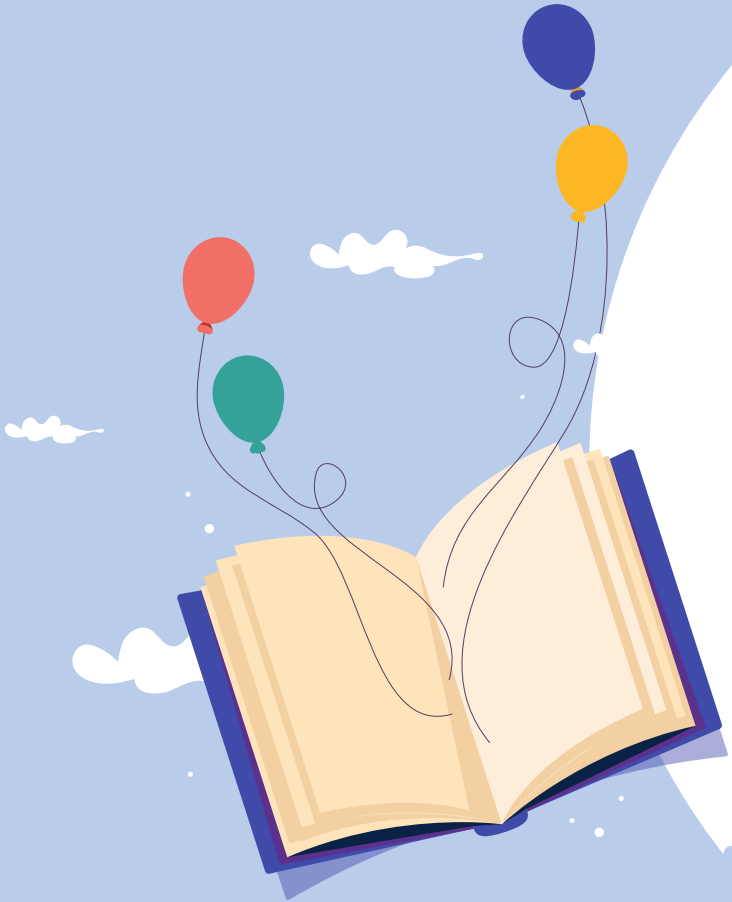
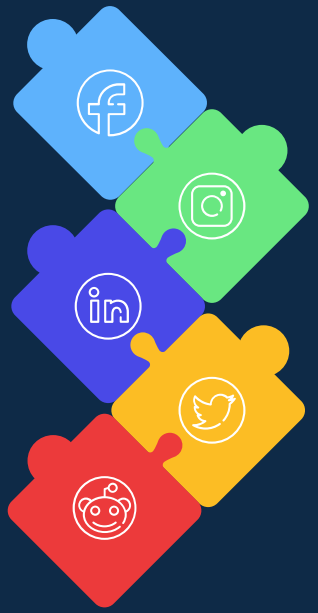


OracleDatabase

For coding starter

INDEX

- Database concept
- **Practice SQL**
- DML
- PL/SQL



FUNCTION

Here are the key characteristics of an Oracle Database function:

1. **Name**: A function is given a unique name within the database schema.
2. **Input Parameters**: Functions can take input parameters, which are values passed to the function for processing.
3. **Return Value**: Unlike procedures that don't have a return value, functions return a single value. This value can be of any data type such as number, string, date, or even a complex user-defined type.
4. **Logic**: Inside the function's body, you can write PL/SQL code to perform calculations, queries, or any other operations required to compute the return value.
5. **Deterministic vs. Non-deterministic**: Functions can be classified as deterministic or non-deterministic. A deterministic function, when given the same input, will always produce the same output. This is crucial for optimizations within the database.
6. **Usage**: Functions can be used in SQL queries, PL/SQL blocks, or any other context where an expression can be used. For example, you could use a function to calculate the total sales of a product or determine the age of a person based on their birthdate.
7. **Security**: Functions can have access to the database's data, but their permissions can be controlled to restrict what data they can access and modify.

CHARACTER FUNCTION

- UPPER returns *char*, with all letters uppercase. *char* can be any of the data types CHAR, VARCHAR2, NCHAR, NVARCHAR2, CLO B, or NCLOB. The return value is the same data type as *char*. The database sets the case of the characters based on the binary mapping defined for the underlying character set. For linguistic-sensitive uppercase, refer to NLS_UPPER.

Function	Explain
UPPER	Convert character data in parameters to all uppercase letters
LOWER	Convert all character data in parameters to lowercase
INITCAP	Convert the first letter to uppercase and the rest to lowercase

```
SELECT JOB, UPPER(JOB), LOWER(JOB), INITCAP(JOB)
    FROM EMP;
```

JOB	UPPER(JOB)	LOWER(JOB)	INITCAP(JOB)
CLERK	CLERK	clerk	Clerk
SALESMAN	SALESMAN	salesman	Salesman
SALESMAN	SALESMAN	salesman	Salesman
MANAGER	MANAGER	manager	Manager
SALESMAN	SALESMAN	salesman	Salesman
MANAGER	MANAGER	manager	Manager
MANAGER	MANAGER	manager	Manager
PRESIDENT	PRESIDENT	president	President
SALESMAN	SALESMAN	salesman	Salesman
CLERK	CLERK	clerk	Clerk
ANALYST	ANALYST	analyst	Analyst
CLERK	CLERK	clerk	Clerk

CHARACTER FUNCTION

- The LENGTH functions return the length of *char*. LENGTH calculates length using characters as defined by the input character set. LENGTHB uses bytes instead of characters. LENGTHC uses Unicode complete characters. LENGTH2 uses UCS2 code points. LENGTH4 uses UCS4 code points.

```
SELECT ENAME, LENGTH(ENAME)
    FROM EMP;
```

ENAME	LENGTH(ENAME)
SMITH	5
ALLEN	5
WARD	4
JONES	5
MARTIN	6
BLAKE	5
CLARK	5

```
SELECT LENGTH('오라클'), LENGTHB('오라클')
    FROM DUAL;
```

LENGTH('오라클')	LENGTHB('오라클')
3	9

CHARACTER FUNCTION

- The INSTR functions search *string* for *substring*. The search operation is defined as comparing the *substring* argument with substrings of *string* of the same length for equality until a match is found or there are no more substrings left.

```
SELECT INSTR('DATABASE IS GOOD', 'A') AS INSTR_1,
    INSTR('DATABASE IS GOOD', 'A', 5) AS INSTR_2,
    INSTR('DATABASE IS GOOD', 'A', 3, 2) AS INSTR_3
    FROM DUAL;
```

INSTR_1	INSTR_2	INSTR_3
2	6	6

CHARACTER FUNCTION

- REPLACE returns *char* with every occurrence of *search_string* replaced with *replacement_string*. If *replacement_string* is omitted or null, then all occurrences of *search_string* are removed. If *search_string* is null, then *char* is returned.

```
SELECT JOB,
    REPLACE(JOB, 'A', 'O') AS REPLACE_1,
    REPLACE(JOB, 'CLERK', 'Employee') AS REPLACE_2
    FROM EMP;
```

JOB	REPLACE_1	REPLACE_2
CLERK	CLERK	Employee
SALESMAN	SOLESMON	SALESMAN
SALESMAN	SOLESMON	SALESMAN
MANAGER	MONOGER	MANAGER
SALESMAN	SOLESMON	SALESMAN
MANAGER	MONOGER	MANAGER
MANAGER	MONOGER	MANAGER
PRESIDENT	PRESIDENT	PRESIDENT
SALESMAN	SOLESMON	SALESMAN
CLERK	CLERK	Employee
ANALYST	ONOLYST	ANALYST
CLERK	CLERK	Employee

CHARACTER FUNCTION

- LPAD and RPAD stand for Left Padding and Right Padding, respectively.

```
SELECT 'Oracle',
       LPAD('Oracle', 10, '#') AS LPAD_1,
       RPAD('Oracle', 10, '*') AS RPAD_1,
       LPAD('Oracle', 10) AS LPAD_2,
       RPAD('Oracle', 10) AS RPAD_2
  FROM DUAL;
```

'ORACLE'	LPAD_1	RPAD_1	LPAD_2	RPAD_2
Oracle	####Oracle	Oracle****	Oracle	Oracle

CHARACTER FUNCTION

- CONCAT returns *char1* concatenated with *char2*. Both *char1* and *char2* can be any of the data types CHAR, VARCHAR2, NCHAR, NVARCHAR2, CLOB, or NCLOB. The string returned is in the same character set as *char1*. Its data type depends on the data types of the arguments.

```
SELECT CONCAT(EMPNO, ENAME) AS CONCAT_1,
       CONCAT(EMPNO, CONCAT(' : ', ENAME)) AS CONCAT_2
  FROM EMP;
```

CONCAT_1	CONCAT_2
7369SMITH	7369 : SMITH
7499ALLEN	7499 : ALLEN
7521WARD	7521 : WARD
7566JONES	7566 : JONES
7654MARTIN	7654 : MARTIN
7698BLAKE	7698 : BLAKE
7782CLARK	7782 : CLARK
7839KING	7839 : KING
7844TURNER	7844 : TURNER
7900JAMES	7900 : JAMES

CHARACTER FUNCTION

- The TRIM, LTRIM, and RTRIM functions are used to delete specific characters within string data.

```
SELECT 'S' || TRIM('  _DATA_  ') || 'E' AS TRIM,
       'S' || TRIM(LEADING FROM '  _DATA_  ') || 'E' AS TRIM_LEADING,
       'S' || TRIM(TRAILING FROM '  _DATA_  ') || 'E' AS TRIM_TRAILING,
       'S' || TRIM(BOTH FROM '  _DATA_  ') || 'E' AS TRIM_BOTH
  FROM DUAL;
```

TRIM	TRIM_LEADING	TRIM_TRAILING	TRIM_BOTH
S_DATA_E	S_DATA_ E	S _DATA_E	S_DATA_E

CHARACTER FUNCTION

- ROUND returns *n* rounded to *integer* places to the right of the decimal point. If you omit *integer*, then *n* is rounded to zero places. If *integer* is negative, then *n* is rounded off to the left of the decimal point.

```
SELECT ROUND(1234.5678) AS ROUND,
       ROUND(1234.5678, 0) AS ROUND_0,
       ROUND(1234.5678, 1) AS ROUND_1,
       ROUND(1234.5678, 2) AS ROUND_2,
       ROUND(1234.5678, -1) AS ROUND_MINUS1,
       ROUND(1234.5678, -2) AS ROUND_MINUS2
  FROM DUAL;
```

ROUND	ROUND_0	ROUND_1	ROUND_2	ROUND_MINUS1	ROUND_MINUS2
1235	1235	1234.6	1234.57	1230	1200

CHARACTER FUNCTION

- The TRUNC (number) function returns *n1* truncated to *n2* decimal places. If *n2* is omitted, then *n1* is truncated to 0 places. *n2* can be negative to truncate (make zero) *n2* digits left of the decimal point.

```
SELECT TRUNC(1234.5678) AS TRUNC,
       TRUNC(1234.5678, 0) AS TRUNC_0,
       TRUNC(1234.5678, 1) AS TRUNC_1,
       TRUNC(1234.5678, 2) AS TRUNC_2,
       TRUNC(1234.5678, -1) AS TRUNC_MINUS1,
       TRUNC(1234.5678, -2) AS TRUNC_MINUS2
  FROM DUAL;
```

TRUNC	TRUNC_0	TRUNC_1	TRUNC_2	TRUNC_MINUS1	TRUNC_MINUS2
1234	1234	1234.5	1234.56	1230	1200

CHARACTER FUNCTION

- CEIL returns the smallest integer that is greater than or equal to *n*.
- FLOOR returns the largest integer equal to or less than *n*.

```
SELECT CEIL(3.14),
       FLOOR(3.14),
       CEIL(-3.14),
       FLOOR(-3.14)
  FROM DUAL;
```

CEIL(3.14)	FLOOR(3.14)	CEIL(-3.14)	FLOOR(-3.14)
4	3	-3	-4

CHARACTER FUNCTION

- This function takes as arguments any numeric data type or any nonnumeric data type that can be implicitly converted to a numeric data type. Oracle determines the argument with the highest numeric precedence, implicitly converts the remaining arguments to that data type, and returns that data type.

```
SELECT SAL, MOD(SAL, 3)
  FROM EMP;
```

SAL	MOD(SAL,3)
800	2
1600	1
1250	2
2975	2
1250	2
2850	0
2450	2
5000	2
1500	0

CHARACTER FUNCTION

- SYSDATE returns the current date and time set for the operating system on which the database server resides.

```
SELECT SYSDATE AS NOW,
       SYSDATE-1 AS YESTERDAY,
       SYSDATE+1 AS TOMORROW
  FROM DUAL;
```

NOW	YESTERDAY	TOMORROW
8/30/2023 4:51:03 오후	8/29/2023 4:51:03 오후	8/31/2023 4:51:03 오후

CHARACTER FUNCTION

- ADD_MONTHS returns the date *date* plus *integer* months.

```
SELECT SYSDATE,
       ADD_MONTHS(SYSDATE, 3)
  FROM DUAL;
```

SYSDATE	ADD_MONTHS(SYSDATE,3)
8/30/2023 4:52:31 오후	11/30/2023 4:52:31 오후

CHARACTER FUNCTION

- MONTHS_BETWEEN returns number of months between dates *date1* and *date2*.

```
SELECT EMPNO, ENAME, HIREDATE, SYSDATE,
       MONTHS_BETWEEN(HIREDATE, SYSDATE) AS MONTHS1,
       MONTHS_BETWEEN(SYSDATE, HIREDATE) AS MONTHS2,
       TRUNC(MONTHS_BETWEEN(SYSDATE, HIREDATE)) AS MONTHS3
FROM EMP;
```

EMPNO	ENAME	HIREDATE	SYSDATE	MONTHS1	MONTHS2	MONTHS3
7369	SMITH	12/17/1980	8/30/2023 4:53:42 오후	-512.442063...	512.44206...	512
7499	ALLEN	2/20/1981	8/30/2023 4:53:42 오후	-510.345288...	510.34528...	510
7521	WARD	2/22/1981	8/30/2023 4:53:42 오후	-510.280772...	510.28077...	510
7566	JONES	4/2/1981	8/30/2023 4:53:42 오후	-508.925934...	508.92593...	508
7654	MARTIN	9/28/1981	8/30/2023 4:53:42 오후	-503.087224...	503.08722...	503
7698	BLAKE	5/1/1981	8/30/2023 4:53:42 오후	-507.958192...	507.95819...	507
7782	CLARK	6/9/1981	8/30/2023 4:53:42 오후	-506.700127...	506.70012...	506

CHARACTER FUNCTION

- NEXT_DAY returns the date of the first weekday named by *char* that is later than the date *date*.
- LAST_DAY returns the date of the last day of the month that contains *date*.

```
SELECT SYSDATE,
       NEXT_DAY(SYSDATE, '월요일'),
       LAST_DAY(SYSDATE)
  FROM DUAL;
```

SYSDATE	NEXT_DAY(SYSDATE,'월요일')	LAST_DAY(SYSDATE)
8/30/2023 4:54:47 오후	9/4/2023 4:54:47 오후	8/31/2023 4:54:47 오후

CHARACTER FUNCTION

- TO_CHAR (datetime) converts a datetime or interval value of DATE, TIMESTAMP, TIMESTAMP WITH TIME ZONE, TIMESTAMP WITH LOCAL TIME ZONE, INTERVAL DAY TO SECOND, or INTERVAL YEAR TO MONTH data type to a value of VARCHAR2 data type in the format specified by the date format *fmt*. If you omit *fmt*, then *date* is converted to a VARCHAR2 value as follows:

```
SELECT TO_CHAR(SYSDATE, 'YYYY/MM/DD HH24:MI:SS') AS 현재날짜시간
  FROM DUAL;
```

현재날짜시간
2023/08/30 17:03:56

CHARACTER FUNCTION

- TO_NUMBER converts *expr* to a value of NUMBER data type. The *expr* can be a number value of CHAR, VARCHAR2, NCHAR, NVARCHAR2, BINARY_FLOAT, or BINARY_DOUBLE data type.

```
SELECT TO_NUMBER('1,300', '999,999') - TO_NUMBER('1,500', '999,999')
  FROM DUAL;
```

TO_NUMBER('1,300','999,999')-TO_NUMBER('1,500','999,999')
-200

CHARACTER FUNCTION

- TO_DATE converts *char* of CHAR, VARCHAR2, NCHAR, or NVARCHAR2 data type to a value of DATE data type.

```
SELECT TO_DATE('2018-07-14', 'YYYY-MM-DD') AS TODATE1,
       TO_DATE('20180714', 'YYYY-MM-DD') AS TODATE2
  FROM DUAL;
```

TODATE1	TODATE2
7/14/2018	7/14/2018

CHARACTER FUNCTION

- NVL lets you replace null (returned as a blank) with a string in the results of a query. If *expr1* is null, then NVL returns *expr2*. If *expr1* is not null, then NVL returns *expr1*.

```
SELECT EMPNO, ENAME, SAL, COMM, SAL+COMM,
       NVL(COMM, 0),
       SAL+NVL(COMM, 0)
  FROM EMP;
```

EMPNO	ENAME	SAL	COMM	SAL+COMM	NVL(COMM,0)	SAL+NVL(COMM,0)
7369	SMITH	800			0	800
7499	ALLEN	1600	300	1900	300	1900
7521	WARD	1250	500	1750	500	1750
7566	JONES	2975			0	2975
7654	MARTIN	1250	1400	2650	1400	2650
7698	BLAKE	2850			0	2850
7782	CLARK	2450			0	2450

CHARACTER FUNCTION

- NVL2 lets you determine the value returned by a query based on whether a specified expression is null or not null. If *expr1* is not null, then NVL2 returns *expr2*. If *expr1* is null, then NVL2 returns *expr3*.

```
SELECT EMPNO, ENAME, COMM,
       NVL2(COMM, 'O', 'X'),
       NVL2(COMM, SAL*12+COMM, SAL*12) AS ANNSAL
  FROM EMP;
```

EMPNO	ENAME	COMM	NVL2(COMM,'O','X')	ANNSAL
7369	SMITH		X	9600
7499	ALLEN	300	O	19500
7521	WARD	500	O	15500
7566	JONES		X	35700
7654	MARTIN	1400	O	16400
7698	BLAKE		X	34200

CHARACTER FUNCTION

- DECODE compares *expr* to each *search* value one by one. If *expr* is equal to a *search*, then Oracle Database returns the corresponding *result*. If no match is found, then Oracle returns *default*. If *default* is omitted, then Oracle returns null.

```
SELECT EMPNO, ENAME, JOB, SAL,
       DECODE(JOB,
              'MANAGER' , SAL*1.1,
              'SALESMAN', SAL*1.05,
              'ANALYST' , SAL,
              SAL*1.03) AS UPSAL
  FROM EMP;
```

EMPNO	ENAME	JOB	SAL	UPSAL
7369	SMITH	CLERK	800	824
7499	ALLEN	SALESMAN	1600	1680
7521	WARD	SALESMAN	1250	1312.5
7566	JONES	MANAGER	2975	3272.5
7654	MARTIN	SALESMAN	1250	1312.5
7698	BLAKE	MANAGER	2850	3135
7782	CLARK	MANAGER	2450	2695
7839	KING	PRESIDENT	5000	5150

CHARACTER FUNCTION

- CASE expressions let you use IF ... THEN ... ELSE logic in SQL statements without having to invoke procedures. The syntax is:

```
SELECT EMPNO, ENAME, JOB, SAL,
       CASE JOB
          WHEN 'MANAGER' THEN SAL*1.1
          WHEN 'SALESMAN' THEN SAL*1.05
          WHEN 'ANALYST' THEN SAL
          ELSE SAL*1.03
       END AS UPSAL
  FROM EMP;
```

EMPNO	ENAME	JOB	SAL	UPSAL
7369	SMITH	CLERK	800	824
7499	ALLEN	SALESMAN	1600	1680
7521	WARD	SALESMAN	1250	1312.5
7566	JONES	MANAGER	2975	3272.5
7654	MARTIN	SALESMAN	1250	1312.5
7698	BLAKE	MANAGER	2850	3135
7782	CLARK	MANAGER	2450	2695
7839	KING	PRESIDENT	5000	5150
7844	TURNER	SALESMAN	1500	1575
7900	JAMES	CLERK	950	978.5