Queries to facilitate acquaintance of Built-In Functions

String Functions,

Number Functions,

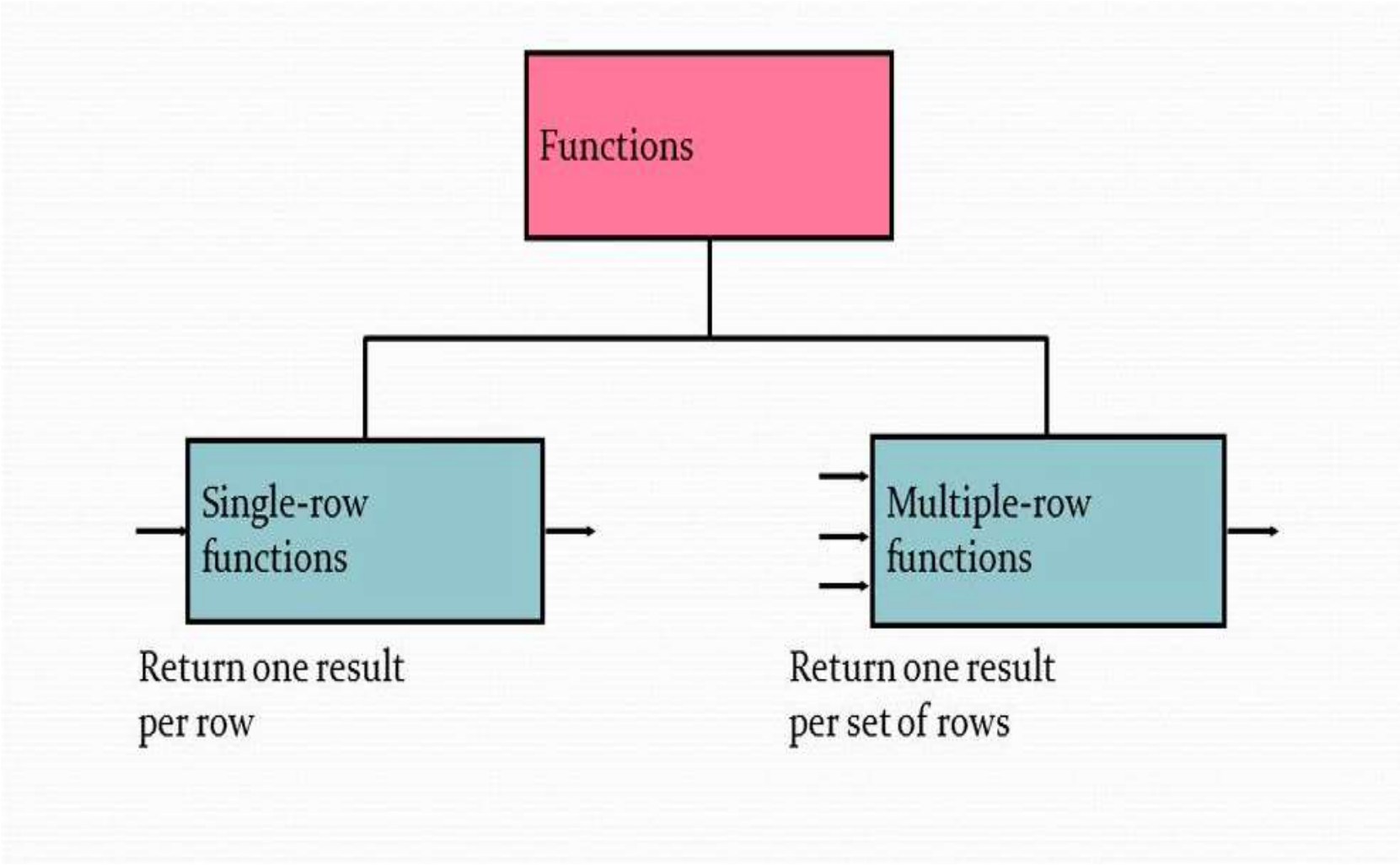
Date Functions

Conversion Functions.

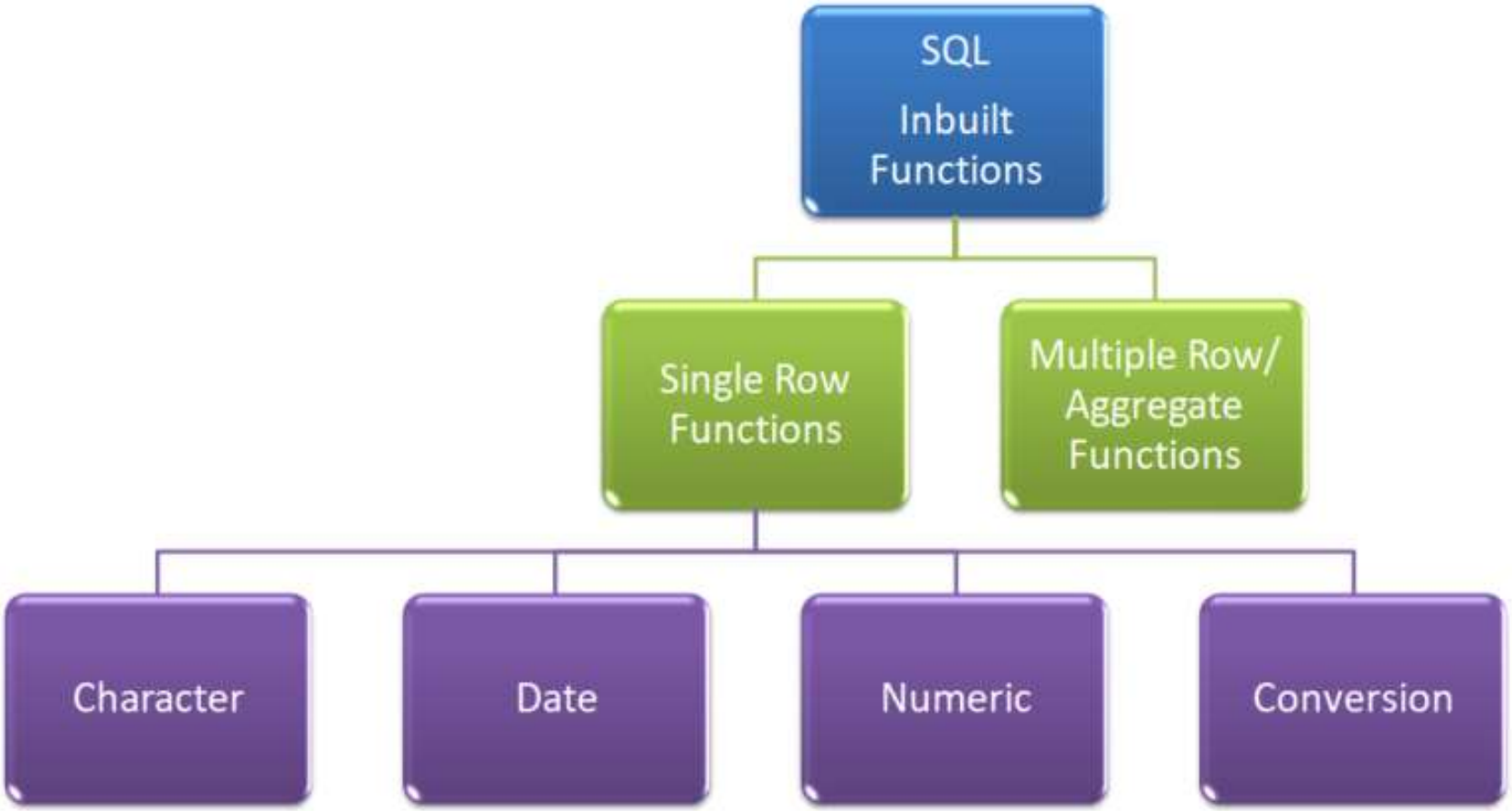
Functions

- Functions are objects in SQL which take one or more input parameter(s) and return a value.

Functions


Functions
Single-row functions
Multiple-row functions
Return one result per row
Return one result per set of rows

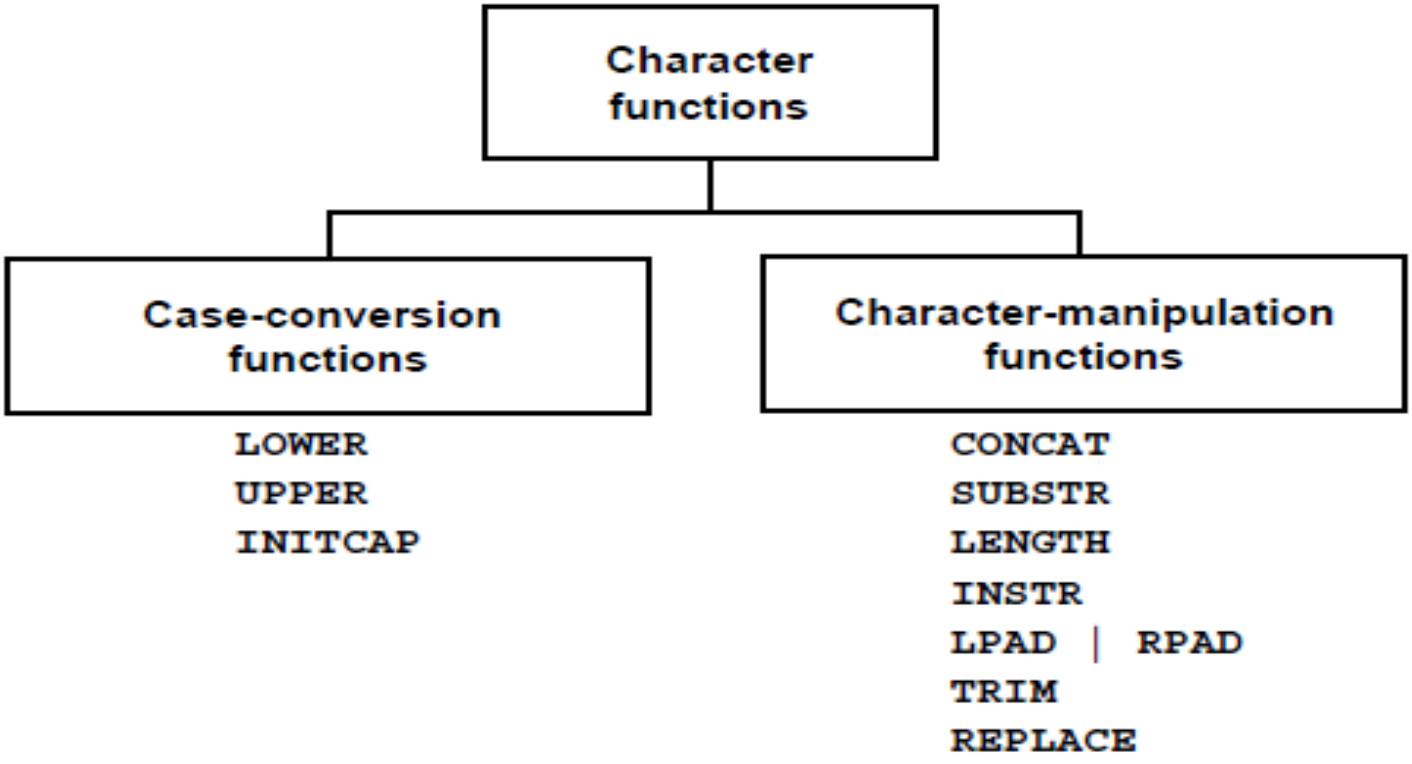
Functions


SQL
Inbuilt
Functions
Single Row
Functions
Multiple Row/
Aggregate
Functions
Character
Date
Numeric
Conversion

Character Functions(string Functions)

- Character functions accept character input values and can return either characters or number values as output.

Character Functions



Character Functions

Case-Conversion Functions

1. **LOWER**

   Returns a given string in lower case.

   SYNTAX: LOWER(COLNAME | STRING)

   Example: select LOWER('ORACLE')
   AS LOWER_OUTPUT from dual;

   LOWER-OUTPUT
   ---
   oracle

Character Functions

Case-Conversion Functions

2. UPPER

Returns a given string in UPPER case.

SYNTAX: UPPER (COLNAME | STRING)

Example: select UPPER('oracle') AS UPPER_OUTPUT
from dual;

```
UPPER_OUTPUT
------------
ORACLE
```

Character Functions

Case-Conversion Functions

3. INITCAP

- Returns a given string with Initial letter in capital.
- **SYNTAX:** INITCAP (COLNAME | STRING)
- **Example**: select INITCAP('oracle') from dual;

```
INITCAP
----------------
Oracle
```

String manipulative Functions

- CONCAT
- SUBSTR
- LENGTH
- INSTR
- LPAD
- RPAD
- REPLACE
- TRIM | LTRIM | RTRIM

String manipulative Functions

- **CONCAT** function joins two strings and returns a single string.
- **Syntax:** CONCAT(column1|string1 , column2|String2)
- This function always appends string2 to the end of string1.
- **Ex1:** SELECT CONCAT(‘oracle', ’server ') FROM DUAL;

o/p: Concat

---

oracleserver

String manipulative Functions

Example2

- Select **concat**(concat(ename,' is a '),job) from emp;

output:

Concat

Smith is a clerk
John is a Manager
Sunny is a GenManager

String manipulative Functions

- **Example3**
- Select **concat**(concat(‘oracle’, ‘server'),’2021’) from dual;

- **Output:** oracleserver2021

String manipulative Functions

LENGTH

- Returns the length of a given string.
- **Syntax:** LENGTH(Column|String)
- **Example1:** select length('oracle') from dual;

  LENGTH

  6
- **Example2:** select length(ename) from emp;

String manipulative Functions

SUBSTR

- Returns a substring from a given string starting from position '**pos'** upto '**n'** characters
- **Syntax:** SUBSTR(Column|String, pos, [N])
- **Ex1:** select substr('ABCDEFGH', 3, 4) from dual;

Substr

CDEF

String manipulative Functions

SUBSTR

- **Ex2:** select substr('ABCDEFGH', 2) from dual;

Substr
BCDEFGH

- **Ex3:** select substr('ABCDEFGH', -3, 2) from dual;

Substr
FG

String manipulative Functions

INSTR

- Checks whether a given character or pattern occurs in the given string or not.
- If the pattern occurs in the string then returns the first position of its occurrence otherwise returns 0.
- **Syntax:**
- INSTR(Column|String, ‘pattern', [startposition-m] , [occurrence-n])
- Optionally, you can provide a position *m* to start searching, and the occurrence *n* of pattern.
- Also, if the starting position is not given, then it starts search from index 1, by default.

String manipulative Functions

INSTR

Example 1:

SELECT INSTR('ABCDABCDABCD', 'AB', 1 , 1)
FROM DUAL;

Output: 1

Example 2:

SELECT INSTR('ABCDABCDABCD', 'AB', 1 , 3)
FROM DUAL;

Output: 9

String manipulative Functions

INSTR

Example 3:

SELECT INSTR('ABCDABCDABCD', 'EF', 1 , 1)

FROM DUAL;

Output: 0

Example 4:

SELECT INSTR('ABCDABCDABCD', 'AB', 4 , 1)

FROM DUAL;

Output: 5

String manipulative Functions

REVERSE

- Reverses the given string.
- **Syntax**: REVERSE (Columnname | String)

- **Example:** SELECT REVERSE(‘ABCD’) FROM DUAL;

  Output: DCBA

String manipulative Functions

REPLACE

- Replace a sequence of characters in a string with another set of characters.
- The REPLACE function accepts three parameter
- **Syntax:**

  REPLACE(input_str, str_to_replace, replacement_str)

String manipulative Functions

REPLACE

Ex: SELECT REPLACE(‘oracle server’, ’oracle', ’sql')
FROM DUAL;

output: sql server

String manipulative Functions

LPAD

- returns the string padded leftside with a specified pad string and to a specified length.
- If the pad string is not specified, then the given string is padded on the left with spaces.
- **Syntax:** LPAD(Column|String, n, ‘padstring')

  n is specified length of string.

  Padstring is a character or string.

String manipulative Functions

LPAD

Ex1: SELECT LPAD(‘oracle',10,‘#') FROM DUAL;

Output: ####oracle

Example2: SELECT LPAD(‘oracle',10) FROM DUAL;

Output: oracle

String manipulative Functions

RPAD

- returns the string padded rightside with a specified pad string and to a specified length.
- If the pad string is not specified, then the given string is padded on the right with spaces.
- **Syntax:** RPAD(Column|String, n, ‘padstring')

  n is specified length of string.

  Padstring is a character or string.

String manipulative Functions

RPAD

Ex1: SELECT RPAD(‘oracle',10,‘#') FROM DUAL;

Output: oracle####

Example2: SELECT RPAD(‘oracle',10) FROM DUAL;

Output:

O	**r**	**a**	**c**	**l**	**e**				

String manipulative Functions

Nesting LPAD and RPAD

Example: SELECT RPAD(LPAD(‘oracle',10,‘#') ,14, ‘#’)
FROM DUAL;

Output: ####oracle####

String manipulative Functions

LTRIM

- Trims(removes) unwanted character(s) from leftside of given string.
- **Syntax:**

  LTRIM(column|string, [‘trim_characters’])
- Ex1: SELECT LTRIM(‘oracle‘, ‘o’) from dual;

  Output: racle

String manipulative Functions

- SELECT LTRIM(‘oracle‘, ‘x’) from dual;

  Output: oracle
- SELECT LTRIM(‘oracle‘, ‘oa’) from dual;

  Output: racle
- SELECT LTRIM(‘oracle‘, ‘ora’) from dual;

  Output: cle
- SELECT LTRIM(‘oracle‘, ‘roa’) from dual;

  Output: cle

String manipulative Functions

RTRIM

- Trims(removes) unwanted character(s) from rightside of given string.
- **Syntax:**

  RTRIM(column|string, [‘trim_characters’])
- Ex1: SELECT RTRIM(‘oracle‘, ‘cle’) from dual;

  Output: ora

String manipulative Functions

- SELECT RTRIM(‘oracle‘, ‘o’) from dual;

  Output: oracle

- SELECT RTRIM(‘oracle‘, ‘@lce’) from dual;

  Output: ora

String manipulative Functions

- **TRIM :** This function trims the specified character(s) from the left or right or bothsides of the given string.
- If no character is specified to be trimmed from the string and there exists some extra space at start or end of the string, then those extra spaces are trimmed off.
- **Syntax:**
- TRIM([LEADING|TRAILING|BOTH]

  ['trim_character' FROM] 'SOURCE')

String manipulative Functions

- SELECT TRIM(‘o' FROM ‘ooracleoo') FROM DUAL;

  output: racle

- SELECT TRIM(LEADING ‘o' FROM ‘ooracleoo') FROM DUAL;

  output: racleoo

String manipulative Functions

- SELECT TRIM(TRAILING ‘o' FROM ‘ooracleoo') FROM DUAL;

  output: ooracle

- SELECT TRIM(BOTH ‘o' FROM ‘ooracleoo') FROM DUAL;

  output: racle

NUMBER FUNCTIONS

- Number functions accept numeric input and return numeric value.

ABS	MOD
CEIL	LOG
FLOOR	LN
POWER	ROUND
EXP	TRUNCATE
SQRT	

NUMBER FUNCTIONS

ABS(N): It returns the absolute value of N.

EXAMPLES: SELECT ABS(-10) FROM DUAL;

OUTPUT: 10

SELECT ABS(1) , ABS(-2) FROM DUAL;

OUTPUT: 1 2

NUMBER FUNCTIONS

CEIL(N): It returns the smallest integer that is greater than or equal to N.

Example: SELECT CEIL(25.75) FROM DUAL;

Output: 26

NUMBER FUNCTIONS

FLOOR(N): It returns the largest integer value that is less than or equal to N.

Example: SELECT FLOOR(25.75) FROM DUAL;

Output: 25

NUMBER FUNCTIONS

POWER(m, n): It returns **m** raised to the **nth** power.

Example: SELECT POWER(4, 2) FROM DUAL;

Output: 16

NUMBER FUNCTIONS

EXP(N): It returns ‘e’ raised to the power of N where e = 2.718.

Example: SELECT EXP(1) FROM DUAL;

Output: 2.718281828459045

NUMBER FUNCTIONS

SQRT(N): It returns the square root of N.

Example: SELECT SQRT(25) FROM DUAL;

Output: 5

NUMBER FUNCTIONS

MOD(M,N): Returns the remainder after dividing M with N.

ExampleS: SELECT MOD(10,2) FROM DUAL;

Output: 0

SELECT MOD(18, 4) FROM DUAL;

Output: 2

NUMBER FUNCTIONS

LOG(M,N) Returns the logarithm of N with base M.

Example: SELECT LOG(10,100) FROM DUAL;

output: 2

SELECT LOG(10,10), LOG(20,20) FROM DUAL;

output: 1 1

NUMBER FUNCTIONS

LN(N)- Returns natural logarithm of n (base ‘e’).

Example: SELECT LN(3) FROM DUAL;

output: 1.098

NUMBER FUNCTIONS

SIGN(N): SIGN function returns -1 if N < 0

0 if N = 0

1 if N > 0

Example: SELECT SIGN(-3), SIGN(0), SIGN(3)
FROM DUAL;

output: -1 0 1

NUMBER FUNCTIONS

ROUND(N, M)

- It returns a number **N** rounded off to **M** decimal places.
- If you omit **M**(optional), then **N** is rounded to 0 places.

Example:

- SELECT ROUND(3.4573,2) FROM DUAL;

```
Round
-----------
      3.46
```

NUMBER FUNCTIONS

ROUND(N, M)

- **Example**: SELECT ROUND(3.45) FROM DUAL;

  output: 3

- **Example**: SELECT ROUND(3.67) FROM DUAL;

  output: 4

NUMBER FUNCTIONS

- **TRUNC(N, M) :** Returns a decimal number N truncated to **M** decimal positions.
- If **M** is ignored, then decimal part is trunacted.

Example1: SELECT TRUNC(3.457,2) FROM DUAL;
output : 3.45

Example2: SELECT TRUNC(3.457) FROM DUAL;
output : 3

DATE FUNCTIONS

- SYSDATE
- ADD_MONTHS()
- LAST_DAY()
- NEXT_DAY()
- MONTHS_BETWEEN
- NEW_TIME()
- TRUNC()
- ROUND()
- EXTRACT()

DATE FUNCTIONS

SYSDATE

- Function used to return the current date and time.
- **Examples:**

  1. SELECT **SYSDATE** AS **CURRENTDATE** FROM DUAL;

  OUTPUT: 06-OCT-23

  2. SELECT **SYSDATE**+1 FROM DUAL

  OUTPUT: 07-OCT-23

DATE FUNCTIONS

ADD_MONTHS()

- This function adds a number(**n**) of months to a given date and returns the same day of month n .
- **Syntax:**

  ADD_MONTHS(date, integer)

DATE FUNCTIONS

ADD_MONTHS()

1: SELECT ADD_MONTHS(‘06-OCT-2023', 1)

FROM dual;

OUTPUT: 06-NOV-2023

2: SELECT ADD_MONTHS(‘06-OCT-2023', -1) FROM dual;

OUTPUT: 06-SEP-2023

3: SELECT ADD_MONTHS(‘SYSDATE', 1) FROM dual;

DATE FUNCTIONS

LAST_DAY()

- It takes a date argument and returns the last day of the month of that date.
- **Syntax:** LAST_DAY(date)
- **Example:**
- SELECT LAST_DAY('06/OCT/2023') AS LAST FROM DUAL;
- **OUTPUT:** 31/0CT/2023

DATE FUNCTIONS

- **NEXT_DAY()** function returns the date of the first weekday that is later than a date.
- Weekday is the day of the week that you wish to return.
- The weekday can be full name(**eg: Tuesday)** or abbreviation (**e.g., Tue)**
- **Syntax**
- NEXT_DAY(date, weekday)

DATE FUNCTIONS

NEXT_DAY()

- SELECT NEXT_DAY('07-OCT-2023', ‘MONDAY') FROM dual;

  OUTPUT: 09-OCT-23

- SELECT NEXT_DAY('07-OCT-2023', ‘MON') FROM dual;

  OUTPUT: 09-OCT-23

DATE FUNCTIONS

MONTHS_BETWEEN

- This function returns the number of months between two dates.
- **Syntax:** MONTHS_BETWEEN(date1, date2)
- If date1 is later than date2, then the result is positive.
- If date1 is earlier than date2, then the result is negative.
- If date1 and date2 are either the same days of the month or both last days of months, then the result is always an integer.

DATE FUNCTIONS

MONTHS_BETWEEN

1. SELECT MONTHS_BETWEEN('06/oct/23', '06/sep/23') FROM DUAL;

   Output: 1

2. SELECT MONTHS_BETWEEN('06/sep/23', '06/oct/23') FROM DUAL;

   Output: -1

3. SELECT MONTHS_BETWEEN('06/oct/23', '06/oct/23') FROM DUAL;

   Output: 0

DATE FUNCTIONS

NEW_TIME() function is used to convert a date from timezone1 to a date in timezone2.

- **Syntax:**

  NEW_TIME(date, timezone1, timezone2)

- date : A date value
- timezone1 : A time zone of the date
- timezone2: A time zone to which the date should be converted

DATE FUNCTIONS

- AST Atlantic Standard Time
- BST Bering Standard Time
- CST Central Standard Time
- EST Eastern Standard Time
- GMT Greenwich Mean Time
- HST Alaska-Hawaii Standard Time
- MST Mountain Standard Time
- PST Pacific Standard Time

DATE FUNCTIONS

NEW_TIME()

- SELECT NEW_TIME('07-oct-2023', 'AST', 'MST') from dual;

- **Output**: 06-oct-2023

DATE FUNCTIONS

- **TRUNC()** function returns a DATE value truncated to a specified unit of measure.
- **Syntax :** TRUNC(date, [format])
- format argument determines the unit to which the date will be truncated.

DATE FUNCTIONS

format values

- YYYY, YEAR, YY, Y **- Year**
- Q - **Quarter**
- MONTH, MON, MM, - **Month of the year**
- W - Day no of the **week of month.**
- ---- ww – week of the year
- dd, ddd - begin of the **day**
- day, dy ,d - start day of the week

DATE FUNCTIONS

TRUNC() Examples

```
SELECT TRUNC(TO_DATE('22-AUG-23'), 'YEAR')
  FROM DUAL;
```

OUTPUT: '01-JAN-23'

```
SELECT TRUNC(TO_DATE('22-AUG-23'), 'Q')
  FROM DUAL;
```

OUTPUT: '01-JUL-23‘ '

DATE FUNCTIONS

TRUNC() Examples

SELECT TRUNC(TO_DATE('22-AUG-23'), 'MONTH') FROM DUAL;

OUTPUT: '01-AUG-23'

SELECT TRUNC(TO_DATE('22-AUG-23'), 'DDD') FROM DUAL;

OUTPUT: '22-AUG-23'

DATE FUNCTIONS

TRUNC() Examples

SELECT TRUNC(TO_DATE('22-AUG-23'), 'DAY')
FROM DUAL;

OUTPUT: 21-AUG-23

SELECT TRUNC(TO_DATE('28-SEP-23'), 'W')
FROM DUAL;

OUTPUT: 22-SEP-23

DATE FUNCTIONS

- **ROUND()** function is used to get the date rounded to the unit specified by the format.
- **SYNTAX**: ROUND(date [,format])

- SELECT ROUND(TO_DATE ('16-SEP-2023'), ‘YEAR‘) FROM DUAL;
- **OUTPUT**: 01-JAN-2024

DATE FUNCTIONS

- **EXTRACT()** function extracts a specific component (year, month, day) from a date value.
- **Syntax :** EXTRACT(field FROM date)
- field argument for date allows YEAR, MONTH, DAY.
- **Example:** SELECT SYSDATE, EXTRACT(YEAR FROM SYSDATE) AS YEAR from dual;

OUTPUT: 07-OCT-23 2023

Conversion Functions

- TO_CHAR
- TO_DATE
- TO_NUMBER()

Conversion Functions

- **TO_CHAR()** function converts a number or date to a string.
- **Syntax**

  TO_CHAR(value , [format_mask])
- **Value** - A number or date to be converted to a string.
- **format_mask**(Optional). This determines the format of the result string.

Conversion Functions

TO_CHAR() with date values

1. SELECT TO_CHAR(sysdate, 'YYYY-MM-DD') FROM dual;

   Output: 2023-10-07

2. SELECT TO_CHAR(sysdate) FROM dual;

   Output: 07-OCT-23

3. SELECT ename, TO_CHAR(joindate, 'DD-MON -YYYY') as joineddate FROM emp;

Conversion Functions

TO_CHAR() with numbers

1. SELECT TO_CHAR(15000) AS RESULT FROM DUAL;

   OUTPUT: 15000

2. SELECT TO_CHAR(15000, '$99,999') AS RESULT FROM DUAL;

   OUTPUT: $15,000

Conversion Functions

- **TO_DATE** function converts a string to a date.
- ***Syntax:***

  TO_DATE (string, [format_mask])
- **String:** It is used to specify the string to be converted.
- **Format_mask:** It is an optional parameter which is used to specify the format to be used for conversion.

Conversion Functions

TO_DATE

- SELECT TO_DATE ('07102023', 'DDMMYYYY') FROM DUAL;

  OUTPUT: 07-OCT-23

- SELECT TO_DATE('2023/OCT/07', ‘YYYY/MON/DD') FROM DUAL;

  OUTPUT: 07-OCT-23

Conversion Functions

- **TO_NUMBER()** function converts a string to a number.
- **Syntax**

  TO_NUMBER(string, [format_mask])

Ex: SELECT TO_NUMBER('1234.56', '9999.99') AS RESULT FROM DUAL;

1234.56

Ex: SELECT TO_NUMBER('123') AS RESULT FROM DUAL;

123

Aggregate Functions

- Aggregate functions calculate on a group of rows and return a single value for each group.
- AVG() – returns the average of a set.
- COUNT() – returns the number of items in a set.
- MAX() – returns the maximum value in a set.
- MIN() – returns the minimum value in a set
- SUM() – returns the sum of all values in a set

Aggregate Functions

- Select **AVG** (salary) FROM emp;
- Select **MAX**(salary) FROM emp;
- Select **MIN**(salary) FROM emp;
- Select **SUM**(salary) FROM emp;

Aggregate Functions

- Select **COUNT**(salary) FROM emp;
- Select **COUNT** (**DISTINCT** salary) FROM emp;
- Select **COUNT** (*) FROM emp;