Database Management System (303105203)

Unit – 2: SQL & its Components

Dr. Rachit Adhvaryu,
Assistant Professor, Computer Science & Engineering

Structured Query Language (SQL)

- lets you access and manipulate databases
- A language which allows to control the overall database.
- Divided into 5 sub parts as follows:

Structured Query Language (SQL)

Type	*Purpose*	*Common Commands*
DDL	Define DB schema	CREATE, ALTER, DROP, TRUNCATE
DML	Manipulate data	SELECT, INSERT, UPDATE, DELETE
DCL	Control access	GRANT, REVOKE
TCL	Manage transactions	COMMIT, ROLLBACK, SAVEPOINT
DQL	Query data	SELECT

Data Definition Language (DDL)

- For definition and description.
- A language in which the storage structure and access methods used by the database system are specified.
- Specification notation for defining the database schema.

Data Definition Language (DDL)

- **CREATE:** To create the database or its objects (like table, index, function, views, store procedure and triggers).
- **DROP:** To delete objects from the database.
- **ALTER:** To alter the structure of the database.
- **TRUNCATE:** To remove all records from a table, including all spaces allocated for the records are removed.
- **RENAME:** To rename an object existing in the database.

Data Manipulation Language (DML)

- Language for accessing and manipulating the data organized by the appropriate data model.
  - DML also known as query language.
- Two classes of languages
  - Procedural – User specifies what data is required and how to get those data.
  - Declarative (nonprocedural) – User specifies what data is required without specifying how to get those data.

Data Manipulation Language (DML)

- **SELECT –** To retrieve data from the a database.
- **INSERT –** To insert data into a table.
- **UPDATE –** To update existing data within a table.
- **DELETE –** To delete records from a database table

Data Control Language (DCL)

- Deals with the rights, permissions and other controls of the database system.
- Examples:
  - **GRANT**: gives user's access privileges to database.
  - **REVOKE**: withdraw user's access privileges given by using the GRANT command.

Transaction Control Language (TCL)

- Deals with the transaction within the database
- Examples:
  - **COMMIT**: commits a Transaction.
  - **ROLLBACK**: rollbacks a transaction in case of any error occurs.
  - **SAVEPOINT**: sets a savepoint within a transaction.
  - **SET TRANSACTION**:

    -specify characteristics for the transaction.

Functions
&
Operators

Logical Functions

AND operator: allows to create sql statement based on two or more condition.

```
E.g.  select * from emp
      where   salary>=40,000   AND
salary<=55,000;
```

OR operator: allows to create sql statement where records are returned when any of the conditions are met.

```
E.g   select * from emp
      where dept='I.T' OR dept='C.E';
```

Combination of AND + OR:

E.g. Select * form student
    Where (branch='I.T' **OR** branch='C.E')
AND (per>=80)

NOT operator: returns only those records that do not satisfy the condition.

E.g. select * from student
    where **NOT** per<50;

Like Operator

LIKE: allows comparison of one string value with another string value.

1.% allows to match any string of any length.

2._allows to match on single character.

E.g.

1.Select * from Emp

where FNM LIKE 'Me%';

2.Select * from emp

where FNM LIKE '_a%' OR FNM LIKE '_s%';

3.Select FNM from student

where FNM LIKE ‘De__’;

4.Select FNM from student

where FNM LIKE ‘%V%’;

Between Operator

Between: allows the selection of rows that contains values within a specified lower and upper limit.

E.g select FNM from student

where per BETWEEN 50 AND 80;

IN Operator

<u>IN:</u>

The IN operator allows you to specify multiple values in a WHERE clause.

The IN operator is a shorthand for multiple OR conditions.

E.g selects all customers that are located in "Germany", "France" or "UK":

```
SELECT * FROM Customers
WHERE Country IN ('Germany', 'France', 'UK');
```

NOT IN Operator

NOT IN:

The NOT IN operator allows you to specify multiple values in a WHERE clause.

The NOT IN operator is a shorthand for multiple OR conditions.

E.g selects all customers that are located in other than "Germany", "France" or "UK":

```
SELECT * FROM Customers
WHERE Country NOT IN ('Germany', 'France', 'UK');
```

There are different types of single row function:

- String/Character functions
- Aggregate functions
- Arithmetic functions
- Conversion functions
- Date function

String or Character functions

1.LOWER:- returns char, with all letters in lowercase

Syntax:-

lower(char)

e.g.

select lower('IVAN BAYROSS')"Lower" from dual;

Output

```
Lower
--------------
ivan  bayross
```

2.INITCAP:- returns a string with the first letter of each word in upper case.

Syntax:-

initcap(char)

e.g.

```
select initcap('IVAN BAYROSS')"Title case" from dual;
```

Output

```
Title case
------------------
Ivan Bayross
```

3.UPPER:- returns char, with all letters in uppercase.

syntax:-

upper(char)

e.g.

select upper(‘ivan bayross’)”capitalized” from dual;

Output

```
Capitalized
----------------
IVAN BAYROSS
```

4.SUBSTR:-returns a portion of characters beginning at character m, and going up to character n.

- if n is omitted the result returned is up to the last character in the string. The first position of char is 1.

Syntax:-
substr(<string>,<start_position>,[<length>])

e.g.

```
select substr('SECURE',3,4) "Substring"

from dual;
```

output:

```
Substring
--------------
CURE
```

5.ASCII:-returns the number code that represents the specified character.

- If more than one character is entered, the function will return the value for the first character and ignore all the characters after the first.

syntax:-

ascii(character)

e.g.

select ascii(‘a’) “Ascii 1”, ascii(‘A’) “Ascii 2”, ascii(‘cure’) “Ascii3” from dual;

output:

```
Ascii1   Ascii2    Ascii3
------------------------------
  97       65        99
```

6. LENGTH:- returns a length of a word.

Syntax:-

length(word)

e.g.

```
select length('sharanam') "length of string"     from dual;
```

Output

```
length of string
---------------
 8
```

- **7.LTRIM**:- returns characters from the left of char with initial characters removed upto the first character not in set.

Syntax:-

ltrim(char[,set])

e.g.

select ltrim('nisha','n') "LTRIM" from dual;

Output

```
LTRIM
---------
isha
```

8.RTRIM:- returns char, with final characters removed after the last character not in set. ‘set’ is optional, it defaults to spaces.

Syntax:-

```
rtrim(char[,set])
```

e.g.

```
select rtrim(‘sunila’,’a’) “RTRIM” from dual;
```

Output

```
RTRIM
-------------
sunil
```

9. TRIM:- remove all specified character either from beginning or the ending of a string.

Syntax:-

trim([leading|trailing|both[<trim_character>from]]<string>)

e.g.

select trim(‘ hansel ’) “trim both side” from dual;

Output

```
trim both side
------------------
hansel
```

e.g.

select trim(leading ‘x’ from ‘xxxhanselxxx’) “remove prefixes” from dual;

Output:

```
 remove prefixes
--------------------
    hanselxxx
```

e.g.

select trim(both ‘x’ from ‘xxxhanselxxx’) “remove both” from dual;

Output:

```
remove both
-------------------
hansel
```

10.LPAD:- returns char1, left-papped to length n with the sequence of character specified in char2.

Syntax:-

lpad('char1,n[,char2])

E.g.

select lpad('page1',10, '*') "lpad" from dual;

```
Output
lapad
---------------
*****page1
```

11. RPAD:- returns char1, right papped to length n with the character specified in char2.

Syntax:-

rpad(char1,n[,char2])

e.g.

select rpad(ivan,10,'x') "RPAD" from dual;

Output

```
RPAD
----------------
ivanxxxxxx
```

Arithmetic functions

1. ABS:- returns the absolute value of ‘n’.

syntax:- ABS(-15)

e.g. Select ABS(-15) “absolute” from dual;

output:

```
absolute
------------
15
```

2. POWER:- returns m raised to the nth power. n must be an integer else an error is returned.

syntax:-power(m,n)

e.g. Select power(3,2)”raised” from dual;

output:

```
raised
---------
9
```

3.**Round**:-returns n, rounded to m places to the right of the decimal point. If m is omitted, n is rounded to 0 places.

syntax:-round(n,[m])

e.g.: select round(15.91,1) “round” from dual;

```
output   round
         -----------
          15.9
```

4.**SQRT**:- returns square root of n.

syntax:-sqrt(n)

e.g. select sqrt(25) “square root” from dual;

```
output    square root
          --------------
          5
```

5. <u>GREATEST</u> :- returns a greatest value in a list of expressions.

Syntax:-greatest(expr1,expr2,expr3…expr n)

e.g.:-

```
select greatest(4,5,17) “num”,


greatest(‘4’, ‘5’, ‘17’) “text” from dual;
```

output

```
num    text
----------------
17       5
```

6.LEAST:- returns the least value in a list of expressions.

Syntax:- least(expr1,expr2,…..,exprn);

e.g. select least(4,5,17)"num",

least('4','5','17')"text" from dual;

Output

```
num  text
---------------
  4    17
```

8. FLOOR:- return a largest integer value that is equal to less than a number.

Syntax:-floor(n)

e.g. select floor(24.8) “flr1”,
floor(13.15)”flr2” from dual;

Output=24   13

9.CEIL:-return the smallest integer value that is greater than or equal to a number.

Syntax:-ceil(n)

e.g. select ceil(24.8)”ceil”,
ceil(13.15)”ceil2” from dual;

Output= 25   14

Aggregate Functions

1.AVG :- returns the average value

e.g.:- Select avg(sal) from emp;

output: 25000

2.MIN :- return the minimum value of expr.

e.g. :-select min(sal) from emp;

output:20000

3.COUNT :- returns the no. of rows where expr. Is not null

e.g.:-select count(acct_no) “no.of accounts” from acct_mstr;

output: No.of accounts

```
--------------------
100
```

4.MAX:- Returns the maximum value of expr.

e.g.:-select max(curbal) “max” from acct_mstr;

output:

```
max
----------
 120000
```

5.SUM:-Returns the sum of the value of ‘n’

e.g.:-select sum(curbal) “total” from acct_mstr;

output:

```
Total
----------
 1350000
```

CONVERSION FUNCTIONS

- TO_CHAR(n [,fmt])

Converts a value of number datatype to character datatype and date datatype.

Example:

SELECT SYSDATE, TO_CHAR(‘Oct-05-2020’,’mon’) FROM DUAL;

OUTPUT:

```
SYSDATE                    TO_CHAR(S
   ------------- ------------       --------------------------------------
Oct-05-2020                      Oct
```

Date Functions

- **SYSDATE**

SYSDATE is a pseudo-column that returns the system's current date and time of type DATE.

The SYSDATE can be used just as any other column name. it takes no arguments.

Example:

```
SELECT  SYSDATE FROM DUAL;
```

output: Oct-03-2020

- **ADD_MONTH(d,n)**

This function adds months to date, it returns a date as result.

Example:

```
SELECT SYSDATE, ADD_MONTHS(SYSDATE,4) FROM DUAL;
```

OUTPUT:

```
 SYSDATE       ADD_MONTHS
------------     ------------------
Oct-03-2020    Feb-03-2021
```

• MONTHS_BETWEEN(d1,d2)

This function returns the number of months between two dates, d1and d2. if d1 is later than d2, then the result is positive. If d1 is earlier than d2, then the result is negative. The output will be a number.

Example

SELECT MONTHS_BETWEEN(“25-DEC-81’,25-DEC-79’) AS DATE1, MONTHS_BETWEEN(‘25-DEC-79’,’25-DEC-81’) AS DATE2 FROM DUAL;

OUTPUT:

```
DATE1      DATE2
---------  ----------
24         -24
```

• NEXT_DAY(DATE,DAY)

THIS FUNCTION RETURNS THE DATE OF NEXT SPECIFIED DAY OF THE WEEK AFTER THE ‘DATE’.

EXAMPLE

SELECT SYSDATE, NEXT_DAY(SYSDATE,’FRIDAY’) FROM DUAL;

OUTPUT:

```
SYSDATE       NEXT_Day(
-------------  ---------------
03-OCT-20      09-oct-20
```

- **LAST_DAY(d)**

This function returns the date of the last day of the month specified. The result will be a date.

Example:

```
SELECT SYSDATE, LAST_DAY(SYSDATE) FROM DUAL;
```

OUTPUT:

```
SYSDATE      LAST_DAY(
------------ ----    -------------------
03-SEP-13        30-SEP-13
```

Thanks