SQL Operators

An operator is reserved word or character used to in an sql statements to perform operations such as comparison and arithmetic operations.

SQL Operators are used to specify condit^{n} in an sql statement and to serve as conjuctions (and, for, not, and) for multiple conditions in a statement.

1) Arithmetic Operators (+, -, *, /)

Precedence of Operators -
Every arithmetic operator has been given precedence.

i) Multiplication & Division take priority over addition & substraction.
ii) If there are no. of operators having the same precedence then the execution control is from left to right.
iii) Braces / Parenthesis are used to force priorities, the execut^{n} which means whatever operator whithin bracket executed just and then operator outside.

2) Comparison Operator

a) = Checks values of 2 ~~operators~~ operands are equal or not.
If yes then condition becomes true.

b) != Check values of 2 operands are equal or not.
Condition becomes true only if values are not equal.

c) > Check if the value of left operand is greater than the value of right operand.
If yes then condition becomes true.

d) < Checks if the value of left operand is less than the value of right operand.
If yes then condition becomes true.

e) >= Check if the values of left operand is greater than or equal to the values of the right operand.
If yes then condition becomes true.

f) <= Check if the values left operand is less than or equal to the value of the right operand.
If yes then condit^n becomes true.

g) !> This operator checks if the value of left operand is not greater than the value of right operand. If yes then condition becomes true.

Ques 1)

h) !< This operator checks if the value of left operand is not less than the value of right operand. If yes then condition becomes true.

Ques 2)

3) Logical Operators (and, or, not)

a) AND

The AND operator allows the existance of multiple conditions in an sql statement.

(true if all the condition seperated by AND is true.)

This operator compares both two operands / booleans as expressions & returns true when both expression are true.

b) OR

The OR is used to combine the multiple condition in an sql statement where clause.

c) NOT

The NOT operator reserves the meaning of logical operator with example.

NOT IN , NOT BETWEEN , NOT EXISTS

Page No.
Date: / /

ques 1) Write sql query to list all the employees whose salary is greater than 1500.

```
Select * from emp
where salary > 1500;
```

ques 2) Write sql query to list all the employees who are clerks or analyst and salary is above ~~1500~~. 1000

```
select * from emp
where job in ('clerk', 'analyst')
AND salary > 1000;
```

* Can you tell me rules & priorities in where clause

order evaluation	priority
1	arithmetic
2	concatenation
3	comparison
4	is [not], null, like
5	not [between]
6	NOT
7	AND
8	OR

11/4/21

dual.

select * from dual;

?
-
x

select 1+1 from dual;

1+1
2

select 40-20;
→ error

select 40-20 from dual;

40-20
20

select 10*20 from dual;

10*20
200

select 100/20 from dual;

100/20
5

select * from emp;
select 20%.10 from emp;
→ error

syntax :- mod (divident, divisor)

- Select empno, ename, salary, mod (salary, 10) from emp;
- select mod (200, 10) from dual;

mod (200, 10)
0

- select mod (200, 3) from dual;

mod (200, 3)
2

The funct^n returns dividant when the value of divisor is '0'

- select mod (200, 0) from dual;

mod (200, 0)
200

- select mod (0, 10) from dual;

mod (0, 10)
0

Comparison Operators

These operators are used to perform operations such as equal to, greater than, less than etc.

select column 1 > column 2 (comparison of columns),
column 2 < column 3 (comparison of columns),

from tablename ;

→ select * from emp where comm > 500;

→ select * from emp where comm < 500;

1) Write a query to display the list of employees whose salary is more than 3000.

select * from emp where salary > 3000;

2) to display list of employees whose comm more than salary.

select * from emp where comm > salary ;

OR

select * from emp where salary < comm ;

3) to display name, job of employees whose comm is less than salary and job is analyst.

```
select ename, job
from emp
where comm < salary
      and 'job' = 'Analyst' ;
```

4) display name, salary, comm, job of employees whose comm less than salary and job is Analyst

```
select ename, salary, comm, job
from emp
where comm < salary and job = 'Analyst';
```

→ select salary, comm, salary + comm
from emp
where comm = NULL;
error

→ select salary, comm, salary + comm
from emp
where comm IS NULL;

→ select salary, comm, salary + comm
from emp
where comm IS NOT NULL;

---

5) to display employee names whose job profile is clerk
select ename from emp where job = 'clerk';

6) to display employee name, job, whose job profile is not a clerk or other than clerk.
select ename, job from emp where job != 'clerk';

Logical Operatars.

The logical operatars are used to perform operation such as AND, OR, NOT

```
select col1, col2, col3
from tableName
where logical condition ;
```

1) to display the name from emp table whose job profile is analyst and salary greater than 2000.

```
select ename
from employee                    salary
  where job = 'Analyst AND ~~sal~~ > 2000;
```

2) to display the name, salary

```
select ename, salary
from employee
  where ~~sal~~ job = 'Analyst' AND salary > 2000;
```

In case of job profile is other than Analyst then the query will be.

```
select ename, salary
from emp
where job != 'Analyst' AND salary > 2000;
```

3) to list all the salesman in department 30

```
select * from emp
where job = 'salesman' AND deptno = 30;
```

4) to list all the salesman in dept 30 & having salary < 1500

```
select * from emp
where job = 'Salesman' AND deptno = 30
and salary < 1500;
```

5) list all the employees who are not working in dept 10 and 20.

```
select * from emp
where deptno NOT IN (10, 20);
```

6) list all the employees whose job profile is Analyst or clerk and deptno is 10 & 20.

```
select * from emp
where job = 'Analyst' OR job = 'clerk'
      AND deptno IN (10, 20);
```

output →

deptno
20
20
20
20
20
20
20
10

7) _____ deptno is not 10 & 20

```
select * from emp
where job = 'Analyst' OR job = 'clerk'
      AND deptno NOT IN (10, 20);
```

dept no
30

8) list all the employees who do not have reporting to the manager.

```
select * from emp
where mgr IS NULL;
```

o/p → 7839 KING PRESIDENT — 17-Nov-81 5000 - 10

9)

10)

note

9) list all the employees whose commision is NULL

```
select * from emp
where comm IS NULL;
```

10) list all the employees whose comm is not null

```
select * from emp
where comm IS NOT NULL;
```

SQL like operator.

It is used for pattern matching.
like operator has number of pattern matching characters.

eg :- (%) percent it matches 0 or n no. of characters

(_) underscore underscore matches only or exactly one character.

syntax :-

```
select col1, col2, ----
from table name
where column like pattern;
```

note → you can also combine any no. of cond^n using AND or OR operators

1) list all the employees whose name starts with S

```
select * from emp where ename like 'S%';
```

2) list all the employees whose name is having letter A as a second ~~letter~~ character.

```
select * from emp where ename like '_A%';
```

3) list all the employees whose name having 2 l's ie 2 consecutive 'l'
select * from emp where ename like '%LL%';

4) display all employees whose name ends with 'N';
select * from emp where ename like '%N';

5) display all the employees whose name ~~starts~~ ends with second last char as T
Smith
Scott
select * from emp where ename like '%T_';

6) display all employees whose name ends with third last char as O
John
Scott
Ford
select * from emp where ename like '%O--';

7) display all the employee whose job has 'man' in it.
select * from emp where job like '%MAN%';

Assignment

1) Write sql query to list the employees whose name contains letter 'A'.
select * from ~~employee~~
where ename like '%A%';

2) to list all the employees whose names containing the character set 'AR' together.
select * from emp
where ename like '%AR%';

3) to find the distinct salaries of the employees.
select distinct salary from emp;

4) to list all the employees whose name does not starts with 'S';
select * from emp
where ename # NOT like 'S%';

5) display all the employees who are from deptno 10 and reporting to someone.
select * from emp
where deptno = 10 AND mgr IS NOT NULL;

Ques. Write sql query to display having employee whose names are having underscore (_).

Underscore is used to match only 1 character, if we use it in a usual way in the like ~~operator~~ operatn it will only match 1 character.

So, to change this wherein (_) should not behave like pattern matching.

We use escape character like :, / which will define the original meaning of underscore.

6) display all the ~~managers~~ employees whose manager are either 7902, 7566, 7839
select * from emp
where mgr IN (7902, 7566, 7839);

Distinct should be used after select immediately.
Distinct keyword used to remove duplicate rows.
Distinct is used as a keyword & can only be used in the select statement.

1) Write sql query to display unique dept where the employees are working.

```
select distinct deptno from emp;
```

Orderby clause is used to sort the rows with respect to particular order
Sorting can be ascending order (ASC) or descending order (DESC).

Syntax :-
```
select column list
from table name
where condition
order by col1, col2, ----
```

Orderby clause comes last in select statement.
You can sort particular table by column which is not mention in the select list.

ASC - default sort

2) display all the employees from employee table & sort it in ascending order by salary.

```
select * from emp
order by salary;
```

2) display all the employees from emp table & sort it in desc order by their name.

```
select * from emp
order by ename desc ;
```

3) display all the employees & sort acc. to deptno and salary

```
select * from emp
order by deptno, salary ;
```

only deptno in asc.

4) display all the employees acc. to deptno but salary in desc order.

```
select * from emp
order by deptno , salary desc ;
```

5) display all the employees acc. to deptno and salary both in the desc order

```
select * from emp
order by deptno desc , salary desc ;
```

6) display all the employees whose salary is not in the range of 2500 and 5000 and belongs to dept 10 & 20

```
select * from emp
where salary NOT BETWEEN 2500 AND 5000
    AND deptno in (10, 20);
```

error
```
select * from emp
where salary between (2500 - 5000) ;
```

error
```
select * from emp
where salary between (2500 To 500);
```

Page No.
Date: / /

In case if deptno other 10, 20 then

```
select * from emp
where salary NOT BETWEEN 2500 AND 5000
      AND deptno not in (10, 20);
```

display all the employees who are salesman & having 'E' as second last character in ename

```
select * from emp
where job = 'salesman' AND ename like '%e_';
```

list display the company owneris informat (he should not report to anyone.

```
select * from emp
where mgr is null ;
```

Functions

Functions can be categorized in 2 types :-
↑ function is sub program which is used to perform some operations.

1) System defined functions
2) User defined functions

Single row funct^n. (System defined)
In single row funct^n it accept 1 row & generate 1 output
In single row funct^n returns a value for every row that is processed in a query.
In multiple row funct^n it take multiple row as input & generate 1 o/p (1 result)

a) Character funct^n
b) generic / special funct^n
c) Conversion funct^n
d) Date funct^n
e) Number funct^n

Ques. What is the benefits of funct^n ~~whi~~ within Sql
funct^n are very powerful features in Sql & are used for performing calculations (arithmetic, date, --)

funct^n are used to modify individual data
—"— manipulate the o/p, group of rows
—"— convert column data type
—"— format date & numbers for display purpose

a) Character funct^n.

Character functions :- LOWER it is used to convert the
~~UPPER~~ uppercase char. into lowercase char.