WEEK-2
Queries using Operators in SQL

SQL Operators

- We can define operators as symbols (represented by special characters or by keywords) that help us to perform specific mathematical and logical computations on operands.
- **unary**: A unary operator operates on only one operand.
- **binary**: A binary operator operates on two operands.

Unary operators

+ (unary) - Makes operand positive

syntax: +operand

Example: Select +3 from dual;

output: 3

- (unary)- Makes operand negative

syntax: -operand

Example: Select -4 from dual;

Binary Operators in SQL

- Arithmetic Operator
- Concatenation operator
- Logical Operator
- Comparison/Relational Operator
- Special Operators

Arithmetic Operator

*****	MULTIPLICATION
/	DIVISION
+	ADDITION
-	SUBTRACTION

Priority 1	*	/
Priority 2	+	-

Arithmetic Operators

```
SQL>  SELECT 40 + 20 FROM DUAL;
        Output: 60
SQL> SELECT 40 – 20 FROM DUAL;
        Output: 20
SQL> SELECT 40 * 20 FROM DUAL;
        Output: 800
SQL> SELECT 40 / 20 FROM DUAL;
        Output: 2
```

Arithmetic Operators

SQL> SELECT 40 / 0 FROM DUAL;

Output: divisor is equal to zero

SQL> SELECT 2 + 3 * 5 / 3 - 25 FROM DUAL;

Output: **-**18

SQL> SELECT sal*12 AS ANNUALSAL FROM EMP;

Output:

Concatenation Operator

|| - concatenation of two strings

Ex: SELECT 'oracle' || 'server' FROM DUAL;

OUTPUT: oracleserver

Relational Operators	
=	Checks if the values of two operands are equal or not, if yes then condition becomes true.
!= **<>**	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.

Relational Operators	
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.
<=	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.

Relational Operators

- SELECT * FROM EMP;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
103	randy	20	3500	01-Aug-19	clerk
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
106	jones	10	1000	01-Jan-21	clerk
107	dustin	30	3500	01-Oct-19	manager

Sql>

```
•   create table
•   emp(eno number,ename varchar(16),deptno number,sal
    number,hiredate date,job varchar(16));
•   insert all
•   into emp values(101,'john',10,2500,'01-jun-20','clerk')
•   into emp values(102,'smith',10,1500,'01-jun-22','manager')
•   into emp values(103,'randy',20,3500,'01-aug-19','clerk')
•   into emp values(104,'henry',30,2000,'01-may-18','clerk')
•   into emp values(105,'dave',20,4500,'01-jun-18','manager')
•   into emp values(106,'jones',10,1000,'01-jan-21','clerk')
•   into emp values(107,'dustin',30,3500,'01-oct-19','manager')
•   select * from dual;
•   select * from emp;
```

Relational Operators(=)

- **SQL>** SELECT * FROM EMP WHERE SAL=3500;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
103	randy	20	3500	01-Aug-19	clerk
107	dustin	30	3500	01-Oct-19	manager

Relational Operators(<)

- **SQL>** SELECT * FROM EMP WHERE SAL<2000;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
102	smith	10	1500	01-Jun-22	manager
106	jones	10	1000	01-Jan-21	clerk

Relational Operators(>)

- **SQL>** SELECT * FROM EMP WHERE SAL >2000;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
103	randy	20	3500	01-Aug-19	clerk
105	dave	20	4500	01-Jun-18	manager
107	dustin	30	3500	01-Oct-19	manager

Relational Operators(<=)

- **SQL>** SELECT * FROM EMP WHERE SAL<=2000;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
102	smith	10	1500	01-Jun-22	manager
104	henry	30	2000	01-May-18	clerk
106	jones	10	1000	01-Jan-21	clerk

Relational Operators(>=)

SQL> SELECT * FROM EMP WHERE SAL>=2000;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
103	randy	20	3500	01-Aug-19	clerk
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
107	dustin	30	3500	01-Oct-19	manager

Relational Operators(!=)

SQL> SELECT * FROM EMP WHERE SAL !=3500;
(OR) SELECT * FROM EMP WHERE SAL <> 3500;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
106	jones	10	1000	01-Jan-21	clerk

Special Operators

- IN , NOT IN
- ALL
- ANY
- LIKE, NOT LIKE
- EXISTS, NOT EXISTS
- BETWEEN - AND

IN

IN Operator

- IN operator checks a value matches with any values in the list separated by commas and retrieves the rows from the table that match.
- SQL>SELECT * FROM EMP

  WHERE SAL IN (1000,1500) FROM EMP;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
102	smith	10	1500	01-Jun-22	manager
106	jones	10	1000	01-Jan-21	clerk

Special Operators(NOT IN)

NOT IN Operator

- SQL>SELECT * FROM EMP

  WHERE SAL NOT IN (2500,3500) FROM EMP;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
102	smith	10	1500	01-Jun-22	manager
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
106	jones	10	1000	01-Jan-21	clerk

NOT IN Operator

NOT IN Operator

- SQL>SELECT * FROM EMP
  WHERE SAL NOT IN (2500,3500) FROM EMP;

(OR)

- SQL>SELECT * FROM EMP
  WHERE SAL <> 2500 AND SAL <> 3500;

Special Operators(ALL)

ALL

- ALL operator is used to compare a value with a list of values.
- ALL operator must be preceded by an comparison operator such as =, !=, >,>=, <, <= and followed by a list.
- >ALL <ALL !=ALL >=ALL <=ALL

Special Operators(ALL)

< ALL Operator - Less than minimum

- SQL>SELECT * FROM EMP

  WHERE SAL < ALL(2500,2000, 3000);

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
102	smith	10	1500	01-Jun-22	manager
106	jones	10	1000	01-Jan-21	clerk

Special Operators(ALL)

> ALL Operator - Greater than maximum

- **SQL>**SELECT * FROM EMP
  WHERE SAL > ALL(1500,2000, 2500);

Special Operators(NOT IN)

<>ALL Operator

SQL>SELECT * FROM EMP

WHERE SAL <> ALL(2500,3500);

NOTE: **<>ALL** Operator - same as NOT IN

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
102	smith	10	1500	01-Jun-22	manager
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
106	jones	10	1000	01-Jan-21	clerk

Special Operators(ANY)

- **ANY** operator is used to compare a value with a list of values.
- **Syntax:** *operator* **ANY** (v1, v2, v3)
- ANY operator must be preceded by a comparison operator such as =, !=, >, >=,<, <=
- <ANY >ANY <=ANY >=ANY !=ANY

Special Operators(ANY)

- **SQL>**SELECT * FROM EMP

  WHERE SAL **< ANY**(1500,2000, 2500);

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
102	smith	10	1500	01-Jun-22	manager
104	henry	30	2000	01-May-18	clerk
106	jones	10	1000	01-Jan-21	clerk

Special Operators(ANY)

- **SQL>**SELECT * FROM EMP

  WHERE SAL **= ANY**(1500,2000, 2500);

NOTE: = ANY is same as IN operator

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
104	henry	30	2000	01-May-18	clerk

Special Operators(LIKE)

- LIKE operator tests whether values in a column match a specific **pattern**.
- **Syntax:** Expression LIKE Pattern
- Expression is column name
- pattern is a string to search for in the expression.
- The pattern includes wildcard characters:
- % (percent) matches any string of zero or more characters.
- _ (underscore) matches any single character.

Special Operators(LIKE)

Find names start with j

- **SQL>**SELECT * FROM EMP

  WHERE ENAME LIKE 'j%' ;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
106	jones	10	1000	01-Jan-21	clerk

Special Operators(LIKE)

Find names end with y

```
SQL> SELECT  *  FROM  EMP
       WHERE  ENAME  LIKE  '%y' ;
```

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
103	randy	20	3500	01-Aug-19	clerk
104	henry	30	2000	01-May-18	clerk

Special Operators(LIKE)

Find names containing letter “a”

- SELECT * FROM EMP
  WHERE ENAME LIKE '%a%';

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
103	randy	20	3500	01-Aug-19	clerk
105	dave	20	4500	01-Jun-18	manager

Special Operators(LIKE)

Find Names having ‘m’ as second character

- SELECT * FROM EMP

  WHERE ENAME LIKE '_m%';

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
102	smith	10	1500	01-Jun-22	manager

Special Operators(EXISTS)

- EXISTS operator is used in combination with a subquery.
- EXISTS operator is a Boolean operator that returns either true or false.
- EXISTS operator returns **TRUE** if the subquery returns one or more records. It returns **FALSE** if the sub query returns NO records.
- It is used to test for the existence of any record in a sub query.

syntax

- SELECT *column_name(s)*
  FROM *table_name*
  WHERE EXISTS
  (SELECT *column_name* FROM *table_name* WH
  ERE *condition*);

Special Operators(EXISTS)

- SELECT * FROM EMP

  WHERE **EXISTS**

  (SELECT SAL FROM EMP WHERE SAL=2500);

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
103	randy	20	3500	01-Aug-19	clerk
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
106	jones	10	1000	01-Jan-21	clerk
107	dustin	30	3500	01-Oct-19	manager

Special Operators(EXISTS)

- SELECT * FROM EMP
  WHERE **EXISTS**
  (SELECT SAL FROM EMP WHERE SAL=5000);

Output: no data found

Special Operators(NOT EXISTS)

- NOT EXISTS operator works the opposite of the EXISTS operator.
- NOT EXISTS operator returns **TRUE** if the sub query returns no row. Otherwise, it returns false.

Special Operators(NOT EXISTS)

- SELECT * FROM EMP

  WHERE NOT EXISTS

  (SELECT SAL FROM EMP WHERE SAL<1000);

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
103	randy	20	3500	01-Aug-19	clerk
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
106	jones	10	1000	01-Jan-21	clerk
107	dustin	30	3500	01-Oct-19	manager

BETWEEN –AND

- **BETWEEN** operator allows to specify a range to test. SELECT statement return rows whose values are in the specified range.

- **SQL>**SELECT * FROM EMP

  WHERE **SAL** BETWEEN **1000** AND **2500**;

LOGICAL OPERATORS

- Logical AND operator returns **TRUE** if both expressions are true. Otherwise FALSE.
- Logical OR operator returns **TRUE** if any one of the two expressions are true.
- Logical NOT operator is used to **negate** the given condition. If a condition is **TRUE** then will make it **FALSE**.

Logical Operators

- SELECT * FROM EMP;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
103	randy	20	3500	01-Aug-19	clerk
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
106	jones	10	1000	01-Jan-21	clerk
107	dustin	30	3500	01-Oct-19	manager

LOGICAL OPERATORS(AND)

```
SQL> SELECT * FROM EMP
     WHERE SAL=2500 AND DEPTNO=10;
```

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk

LOGICAL OPERATORS(OR)

SQL> SELECT * FROM EMP
WHERE SAL=2500 OR DEPTNO=10;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
106	jones	10	1000	01-Jan-21	clerk

LOGICAL OPERATORS(NOT)

SQL> SELECT * FROM EMP

WHERE DNO IS NOT NULL;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
103	randy	20	3500	01-Aug-19	clerk
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
106	jones	10	1000	01-Jan-21	clerk
107	dustin	30	3500	01-Oct-19	manager

SET Operators

UNION

- Combines rows of two queries with out duplication.

UNION ALL

- Combines rows of two queries with duplication.

I**NTERSECT**

- Common rows of 2 queries with out duplication.

MINUS

- Resultant rows in the first query after eliminating common rows of second.

SET Operators

Rules for set operators:

- No of columns should match in two queries.
- Data types of the corresponding columns of two queries should match.
- Sorts data in ascending order based on first column.(except union all)

SET Operators(UNION)

- **UNION** operator is used to combine the result sets of two Queries (SELECT statements).
- It removes duplicate rows between them.

SET Operators(UNION)

```
SELECT * FROM EMP WHERE SAL>2000
 UNION
SELECT * FROM EMP WHERE SAL<3000;
```

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
103	randy	20	3500	01-Aug-19	clerk
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
106	jones	10	1000	01-Jan-21	clerk
107	dustin	30	3500	01-Oct-19	manager

SET Operators(UNION ALL)

- **UNION ALL** operator is used to combine the result sets of 2 or more SELECT statements.
- It is different from UNION operator in a way that it does not remove duplicate rows.

SET Operators(UNION ALL)

SELECT * FROM EMP WHERE SAL>1500

UNION ALL

SELECT * FROM EMP WHERE SAL<3000;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
103	randy	20	3500	01-Aug-19	clerk
104	henry	30	2000	01-May-18	clerk
105	dave	20	4500	01-Jun-18	manager
107	dustin	30	3500	01-Oct-19	manager
101	john	10	2500	01-Jun-20	clerk
102	smith	10	1500	01-Jun-22	manager
104	henry	30	2000	01-May-18	clerk
106	jones	10	1000	01-Jan-21	clerk

SET Operators(INTERSECT)

- Compares   the rows of two or more SELECT statements.
- After the comparing process, the INTERSECT operator returns the common records with out duplication.

SET Operators(INTERSECT)

```
SELECT * FROM EMP WHERE SAL>1500
 INTERSECT
SELECT * FROM EMP WHERE SAL<3000;
```

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
101	john	10	2500	01-Jun-20	clerk
104	henry	30	2000	01-May-18	clerk

SET Operators(MINUS)

SELECT * FROM EMP WHERE SAL>1500

MINUS

SELECT * FROM EMP WHERE SAL<3000;

ENO	ENAME	DEPTNO	SAL	HIREDATE	JOB
103	randy	20	3500	01-Aug-19	clerk
105	dave	20	4500	01-Jun-18	manager
107	dustin	30	3500	01-Oct-19	manager

SET Operators

SELECT **ENO** FROM EMP WHERE SAL>2000

UNION

SELECT **ENAME** FROM EMP WHERE SAL<3000;

OUTPUT: Expression must have same data type

SET Operators

SELECT **ENO,ENAME** FROM EMP WHERE SAL>2000

UNION

SELECT **ENAME** FROM EMP WHERE SAL<3000;

OUTPUT:

query block has incorrect number of result columns