Data:l

Supervised: label

Un-Supervised: no label

Bubble Sort:

80 10 25 75 5 30 75

Task:

45 10 70 12 30 20 85 100 5

Sort: Bubble, insertion, selection, quick sort ,merge

Root: 12 height:1

Root:8 : 1

Root: 5 :1

Root 11: 0

Root 18:1

Root 17:0

1. Assignment:

10 40 20

In-order

Pre-order

Post-order

2. Assignment:

60 40 80 20 10 50 Binary Search Tree

3. AVL Tree

20 40 10 50

L:3

R: 2

H: l-r: 1

Table : Student: Entity

Attributes:

1)sid

2)sname

2) Age

Table: Marks:

1) sid

2)M1

3)M2

Microsoft SQL Server:

1) GUI Tool

2) Command based

Syntax:

Insert into new\_table

Select \* from old\_table

Alter : to modify the structure of the table

Update : to modify the values present inside the table

Tasks:

Create table employee(eid,ename,dob,salary)

1) display the employee salary >40000

2) display the employee salary <20000

3) display the employee salary =50000

4) display the employee salary not equal to 10000

5) update query to modify the salary of eid=2 change to 60000

Logical operator:

And : T T => T

T F =>F

F T => F

F F =>F

OR

T T => T

T F =>T

F T => T

F F =>F

Not : True : False

False : True

Between: display records based on the range

Like

Nested Quereis:

Multiple queries inside another query

1) Non-Correlated Sub-query

2) Correlated sub-query

Correlated sub-query:

The nested inner query will be having **reference to the outer query**

**It behaving like loops**

**Eg:**

select \* from Student where 5 in(select Id from marks where Student.id=id)

Non-correlated sub-query:

Inner query does **not contain any reference** to outer query

Behave similar like function

select \* from Student where Id not in (select stdid from marks where m1>=60)

--non-correlation query

Add(a,b)

Exists operator:

Exists operator is used to test for the existence of any record in the sub-query

Syntax:

Select column\_names From table\_name

(select column\_name from talbe\_name where condition )

🡺exists operator return true if the sub-uery return 1 or more records

Not Exists

Select aggregate\_function(column\_name) from table\_name

Alias:

Provide an alter name for the table , attributes

Task:

1) customer(cid,cname,address,email) 5 customers

2) Account(acc\_id,cid,type) 4 -account

3) transaction(trans\_id,acc\_id,amount) 3 transactions

Joins:

1) display the customers information who are having the account information

2) display the transaction of the first 3 customers

3) display the (cid,cname,acc\_id,acc\_type,trans\_id,amount of each customer)

4) display the (cid,cname,acc\_id,acc\_type,trans\_id,amount of each customer) : condition:amount>50000