ST. FRANCIS INSTITUTE OF TECHNOLOGY

MT. POINSUR, BORIVALI (W), MUMBAI

LAB MANUAL

EXPERIMENT NO. 5

Aim: - Implement SQL operators, Aggregate function, Group by and having clause.

Theory:-

1. Set Operator
2. Aggregate function
3. Group by clause with having

Lab Manual:

1. SQL Operators:

1. Arithmetic Operators:

Arithmetic operators perform mathematical operations on two expressions of one or more of the data types of the numeric data type category.
SELECT <Expression> [arithmetic operator]<expression>... FROM [table_name] WHERE [expression]; -

Operator	Description	Example
/	Division (numbers and dates)	SELECT SAL / 10 FROM EMP;
*	Multiplication	SELECT SAL * 5 FROM EMP;
+	Addition (numbers and dates)	SELECT SAL + 200 FROM EMP;
-	Subtraction (numbers and dates)	SELECT SAL - 100 FROM EMP;

2. Comparison/Relational Operator:

Operator	Description	Example
=	Checks if the values of two operands are equal or not, if yes then condition becomes true.	(a = b) is not true.
!=	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(a != b) is true.
<>	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(a <> b) is true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(a > b) is not true.
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(a < b) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(a >= b) is not 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.	(a <= b) is true.
!<	Checks if the value of left operand is not less than the value of right operand, if yes then condition becomes true.	(a !< b) is false.
!>	Checks if the value of left operand is not greater than the value of right operand, if yes then condition becomes true.	(a !> b) is true.

3. **Logical/Boolean Operators: (AND, OR, NOT, Between, Exists, IN, NOT IN, Like)**

IN	"Equivalent to any member of" test. Equivalent to "= ANY".	SELECT * FROM EMP WHERE ENAME IN ('SMITH', 'WARD');
ANY/ SOME	Compares a value to each value in a list or returned by a query. Must be preceded by =, !=, >, <, <=, or >=. Evaluates to FALSE if the query returns no rows.	SELECT * FROM DEPT WHERE LOC = SOME ('NEW YORK','DALLAS');
[NOT] IN	Equivalent to "!= ANY". Evaluates to FALSE if any member of the set is NULL.	SELECT * FROM DEPT WHERE LOC NOT IN ('NEW YORK', 'DALLAS');
ALL	Compares a value with every value in a list or returned by a query. Must be preceded by =, !=, >, <, <=, or >=. Evaluates to TRUE if the query returns no rows.	SELECT * FROM emp WHERE sal >= ALL (1400, 3000);
[NOT] BETWEEN *x* and *y*	[Not] greater than or equal to *x* and less than or equal to *y*.	SELECT ENAME, JOB FROM EMP WHERE SAL BETWEEN 3000 AND 5000;
EXISTS	TRUE if a sub-query returns at least one row.	SELECT * FROM EMP WHERE EXISTS (SELECT ENAME FROM EMP WHERE MGR IS NULL);
[NOT] LIKE	TRUE if *x* does [not] match the pattern *y*. Within *y*, the character "%" matches any string of zero or more characters except null. The character "_" matches any single character.	SELECT * FROM EMP WHERE ENAME LIKE '%E%';
IS [NOT] NULL	Tests for nulls. This is the only operator that should be used to test for nulls.	SELECT * FROM EMP WHERE COMM IS NOT NULL AND SAL > 1500;

ST. FRANCIS INSTITUTE OF TECHNOLOGY

MT. POINSUR, BORIVALI (W), MUMBAI

LAB MANUAL

Exercise 1 on SQL Operator:

1. **Create table employee with the following structure : (Emp_id, Ename, Salary, Age, DOJ, DEPT)**

```
create table employee (Emp_id number(5), Ename varchar2(25), Salary number(7), Age number(3), DOJ varchar(13),  DEPT varchar2(20))
```

```
Table created.
```

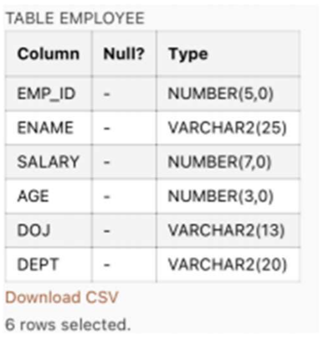


TABLE EMPLOYEE

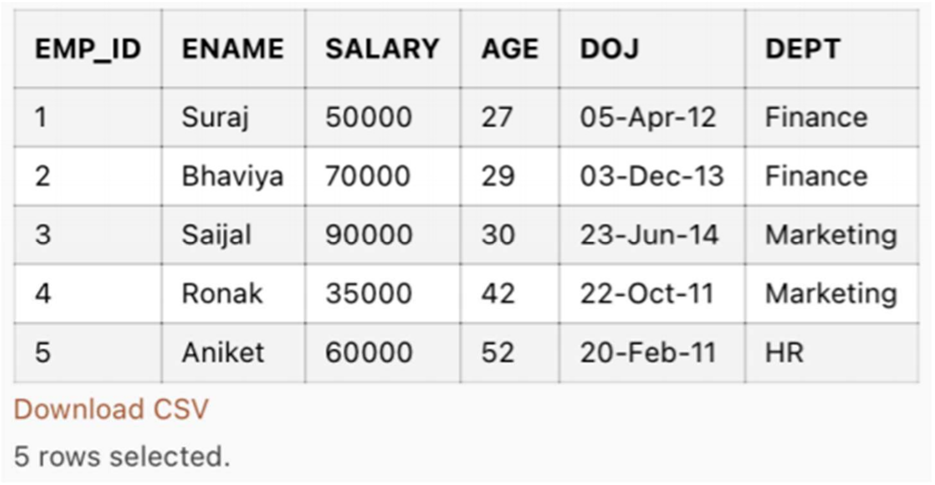
Column	Null?	Type
EMP_ID	-	NUMBER(5,0)
ENAME	-	VARCHAR2(25)
SALARY	-	NUMBER(7,0)
AGE	-	NUMBER(3,0)
DOJ	-	VARCHAR2(13)
DEPT	-	VARCHAR2(20)

Download CSV

6 rows selected.

2. **INSERT 5 RECORDS**

EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
1	Suraj	50000	27	05-April-2012	Finance
2	Bhaviya	70000	29	03-Dec-2013	Finance
3	Saijal	90000	30	23-June-2014	Marketing
4	Ronak	35000	42	22-Oct-2011	Marketing
5	Aniket	60000	52	20-Feb-2011	HR



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
1	Suraj	50000	27	05-Apr-12	Finance
2	Bhaviya	70000	29	03-Dec-13	Finance
3	Saijal	90000	30	23-Jun-14	Marketing
4	Ronak	35000	42	22-Oct-11	Marketing
5	Aniket	60000	52	20-Feb-11	HR

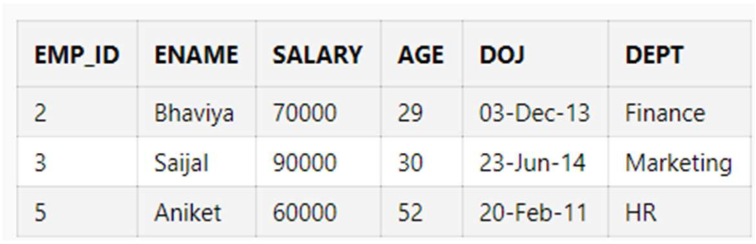
Download CSV

5 rows selected.

Queries on SQL Operator:

1. Find all employee names that have salary greater than 50000.

```
select * from employee where salary>50000
```

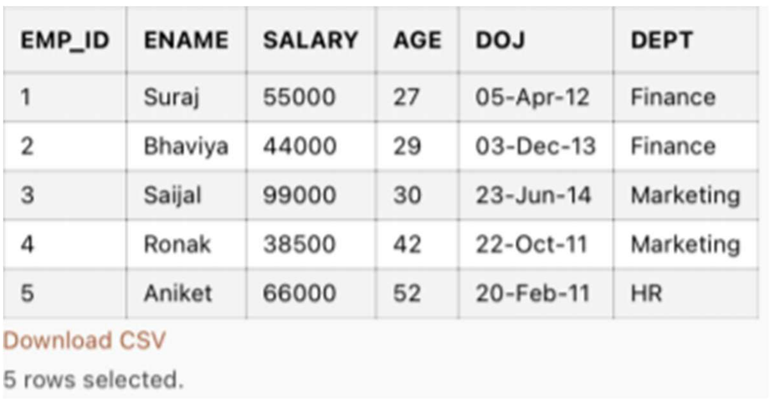


EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
2	Bhaviya	70000	29	03-Dec-13	Finance
3	Saijal	90000	30	23-Jun-14	Marketing
5	Aniket	60000	52	20-Feb-11	HR

2. Give 10% raise in salary of each employee.

```
update employee set salary = salary + salary*0.1
```

```
5 row(s) updated.
```

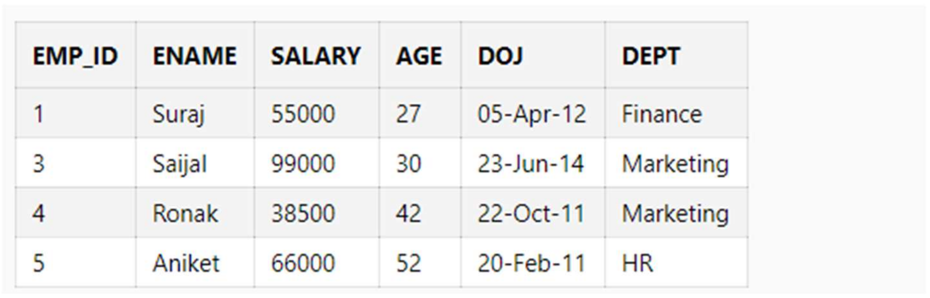


EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
1	Suraj	55000	27	05-Apr-12	Finance
2	Bhaviya	44000	29	03-Dec-13	Finance
3	Saijal	99000	30	23-Jun-14	Marketing
4	Ronak	38500	42	22-Oct-11	Marketing
5	Aniket	66000	52	20-Feb-11	HR

Download CSV
5 rows selected.

3. Give the details of employee joined from 05-april-2012 to 23-june-2014.

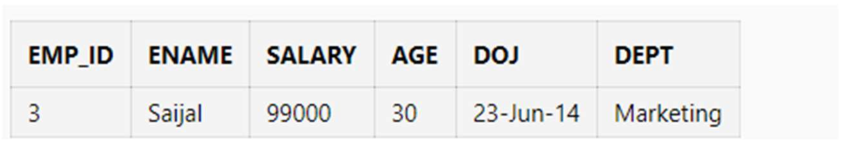
```
select * from employee where doj between '05-Apr-12' and '23-Jun-14'
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
1	Suraj	55000	27	05-Apr-12	Finance
3	Saijal	99000	30	23-Jun-14	Marketing
4	Ronak	38500	42	22-Oct-11	Marketing
5	Aniket	66000	52	20-Feb-11	HR

4. Find all employees who are having salary greater than 70000 and have joined after 3 dec 2013.

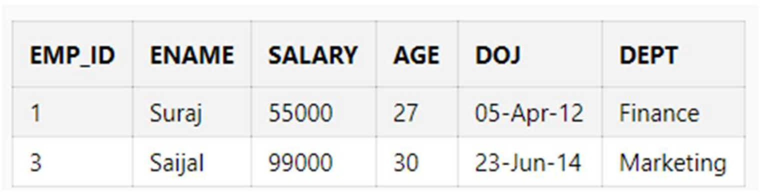
```
select * from employee where salary>70000 and doj>'03-dec-13'
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
3	Saijal	99000	30	23-Jun-14	Marketing

5. Find all employees with name starting with s.

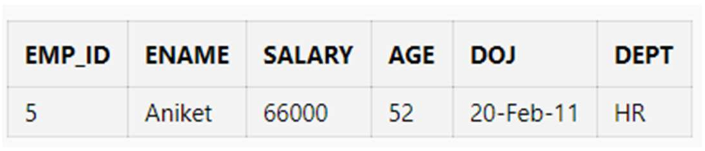
```
select * from employee where ename like 'S%'
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
1	Suraj	55000	27	05-Apr-12	Finance
3	Saijal	99000	30	23-Jun-14	Marketing

6. Find all employees who have at least one 'e' in their names.

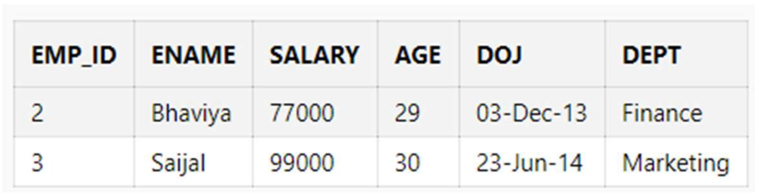
```
select * from employee where ename like '%e%'
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
5	Aniket	66000	52	20-Feb-11	HR

7. Find all employees with age either 29 or 30.

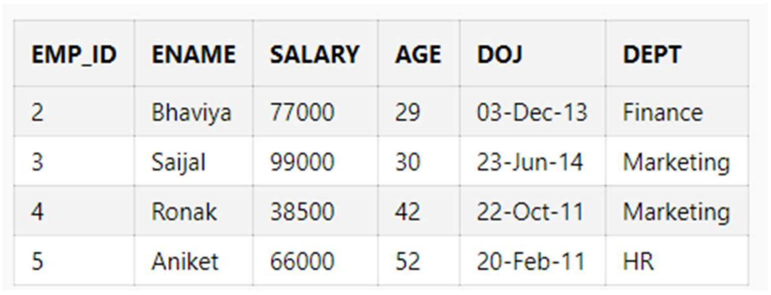
```
select * from employee where age =29 or age = 30
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
2	Bhaviya	77000	29	03-Dec-13	Finance
3	Saijal	99000	30	23-Jun-14	Marketing

8. Find all employees who have not joined on 05-april-2012.

```
select * from employee where doj!='05-Apr-12'
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT
2	Bhaviya	77000	29	03-Dec-13	Finance
3	Saijal	99000	30	23-Jun-14	Marketing
4	Ronak	38500	42	22-Oct-11	Marketing
5	Aniket	66000	52	20-Feb-11	HR

9. Alter the table by adding new column as amount deducted from salary towards tax. Update the value of tax in the table as 20% of salary.

```
alter table employee add tax number(7)
```

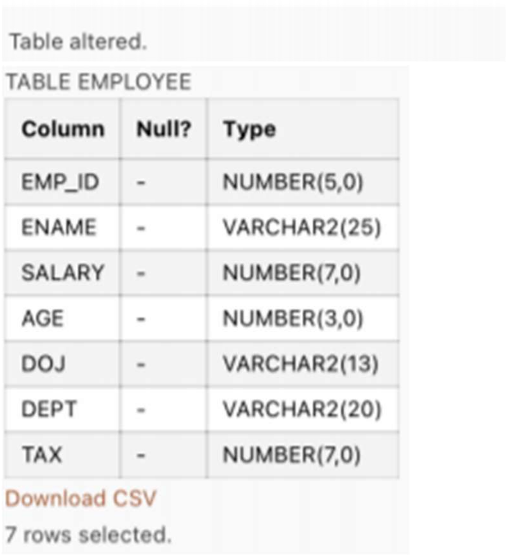


Table altered.

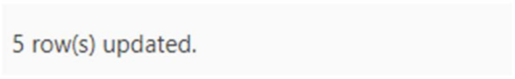
TABLE EMPLOYEE

Column	Null?	Type
EMP_ID	-	NUMBER(5,0)
ENAME	-	VARCHAR2(25)
SALARY	-	NUMBER(7,0)
AGE	-	NUMBER(3,0)
DOJ	-	VARCHAR2(13)
DEPT	-	VARCHAR2(20)
TAX	-	NUMBER(7,0)

Download CSV

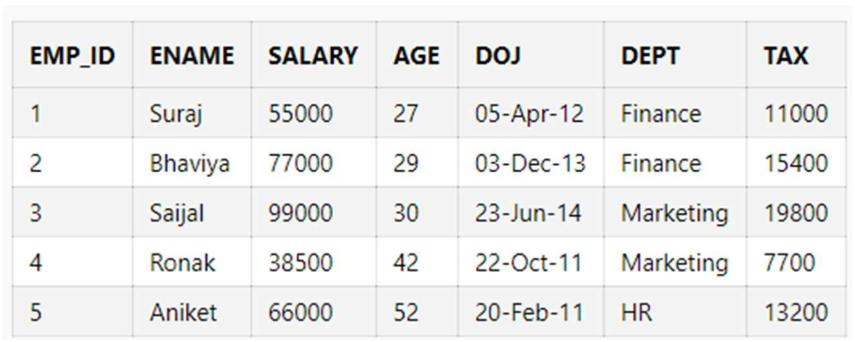
7 rows selected.

```
Update employee set tax = salary*0.2
```



5 row(s) updated.

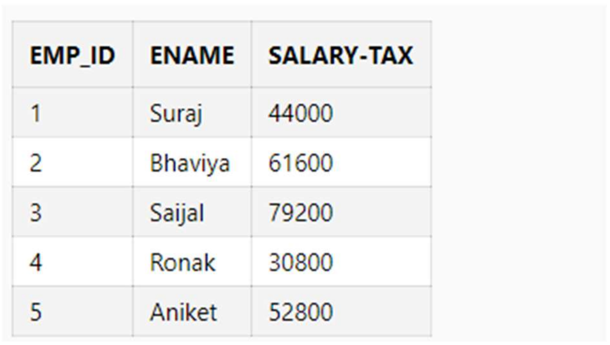
```
select * from employee
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
1	Suraj	55000	27	05-Apr-12	Finance	11000
2	Bhaviya	77000	29	03-Dec-13	Finance	15400
3	Saijal	99000	30	23-Jun-14	Marketing	19800
4	Ronak	38500	42	22-Oct-11	Marketing	7700
5	Aniket	66000	52	20-Feb-11	HR	13200

10. Calculate the net salary for each employee.

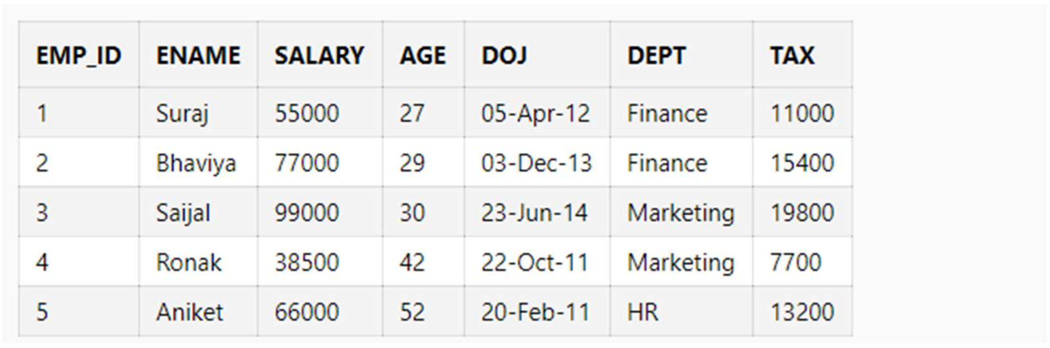
```
SELECT emp_id,ename,salary-tax  FROM employee
```



EMP_ID	ENAME	SALARY-TAX
1	Suraj	44000
2	Bhaviya	61600
3	Saijal	79200
4	Ronak	30800
5	Aniket	52800

11. Find all employees whose age is greater than 25 and earns salary. (use exists clause).

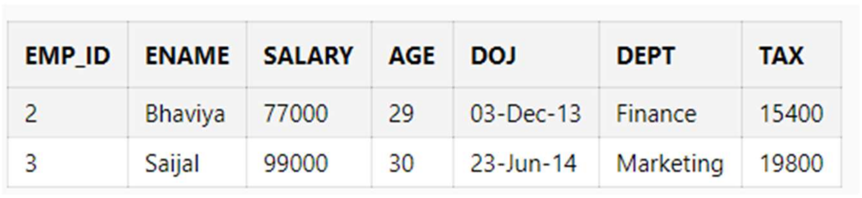
```
SELECT * FROM Employee WHERE EXISTS (SELECT ENAME FROM Employee WHERE age>25)
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
1	Suraj	55000	27	05-Apr-12	Finance	11000
2	Bhaviya	77000	29	03-Dec-13	Finance	15400
3	Saijal	99000	30	23-Jun-14	Marketing	19800
4	Ronak	38500	42	22-Oct-11	Marketing	7700
5	Aniket	66000	52	20-Feb-11	HR	13200

12. Find all employee names whose age is from the list given ‘25, 30, 24, 29’.

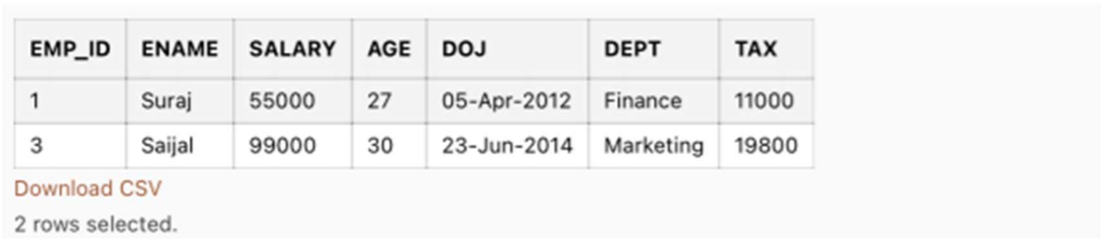
```
SELECT * FROM employee WHERE age IN (24,25,29,30)
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
2	Bhaviya	77000	29	03-Dec-13	Finance	15400
3	Saijal	99000	30	23-Jun-14	Marketing	19800

13. Find all employee names who has not joined on these dates {22-oct-201, 20-feb-2011, 03-dec-2013}.

```
SELECT * FROM employee WHERE doj NOT IN ('22-Oct-2011', '20-Feb-2011','03-Dec-2013')
```



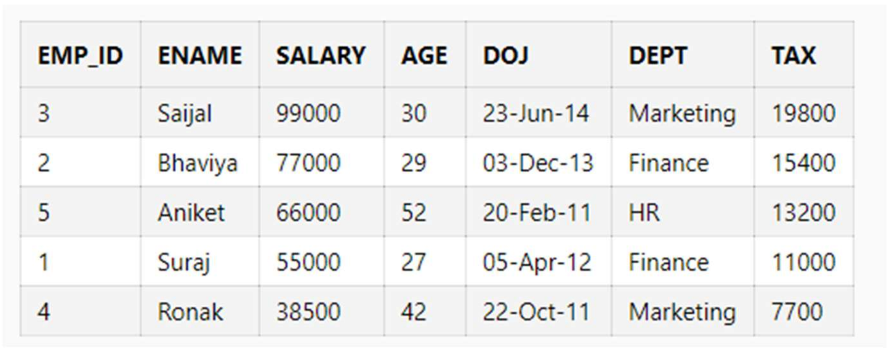
EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
1	Suraj	55000	27	05-Apr-2012	Finance	11000
3	Saijal	99000	30	23-Jun-2014	Marketing	19800

Download CSV

2 rows selected.

14. List all employees in descending order of their salary.

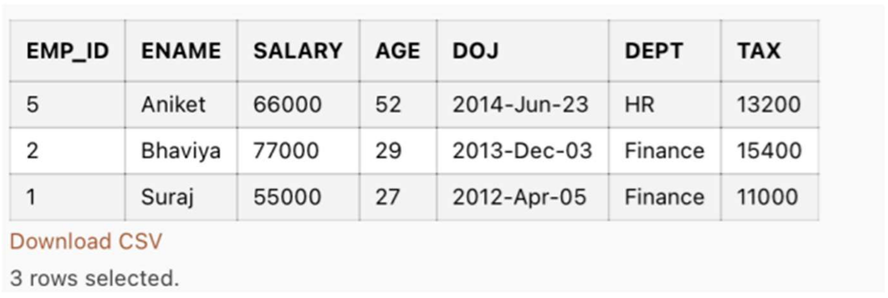
```
select * from employee order by salary desc
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
3	Saijal	99000	30	23-Jun-14	Marketing	19800
2	Bhaviya	77000	29	03-Dec-13	Finance	15400
5	Aniket	66000	52	20-Feb-11	HR	13200
1	Suraj	55000	27	05-Apr-12	Finance	11000
4	Ronak	38500	42	22-Oct-11	Marketing	7700

15. List all employees name in ascending order with joining date '05-april-2012' or after this date.

```
select * from employee where doj>= '2012-Apr-05' order by ename
```



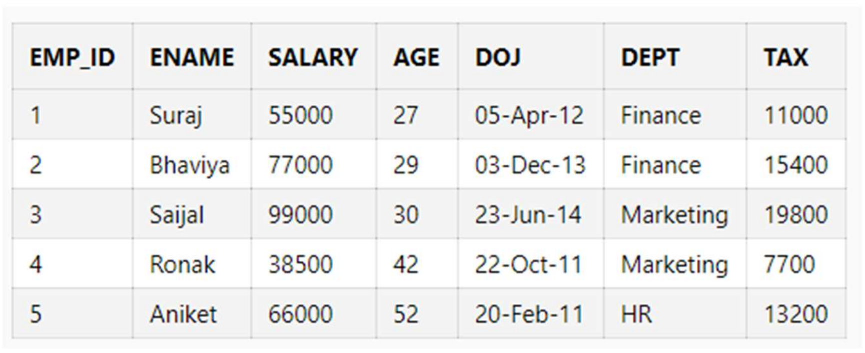
EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
5	Aniket	66000	52	2014-Jun-23	HR	13200
2	Bhaviya	77000	29	2013-Dec-03	Finance	15400
1	Suraj	55000	27	2012-Apr-05	Finance	11000

Download CSV

3 rows selected.

16. List the employees whose age is not null.

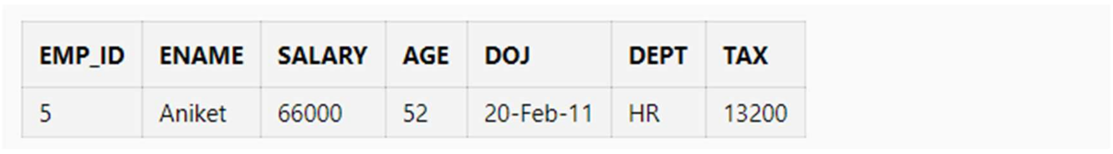
```
SELECT * FROM EMPLOYEE WHERE age IS NOT NULL
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
1	Suraj	55000	27	05-Apr-12	Finance	11000
2	Bhaviya	77000	29	03-Dec-13	Finance	15400
3	Saijal	99000	30	23-Jun-14	Marketing	19800
4	Ronak	38500	42	22-Oct-11	Marketing	7700
5	Aniket	66000	52	20-Feb-11	HR	13200

17. List the employees whose age is greater than the age of all the employees having salary greater than 5000.

```
SELECT * FROM EMPLOYEE WHERE age=(SELECT MAX(age)From EMPLOYEE WHERE salary>50000)
```



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
5	Aniket	66000	52	20-Feb-11	HR	13200

ST. FRANCIS INSTITUTE OF TECHNOLOGY

MT. POINSUR, BORIVALI (W), MUMBAI

LAB MANUAL

2. Aggregate function, group by and having clause:

Aggregate functions take a collection of values i.e. a set or multiple set of values as input and return a single value as output. These are often referred to as Group functions as they operate on groups of rows and return one value for the entire group.

❖ **Different Aggregate Functions are:**

1. **avg:** average value- Avg(value)
2. **min:** minimum value- Min(value)
3. **max:** maximum value- Max(value)
4. **Sum:** sum of values- Sum(value)
5. **Count:** number of values- Count(value)

❖ **SQL GROUP BY Syntax:**

```
SELECT column_name, aggregate_function(column_name)
FROM table_name
WHERE column_name operator value
GROUP BY column_name;
```

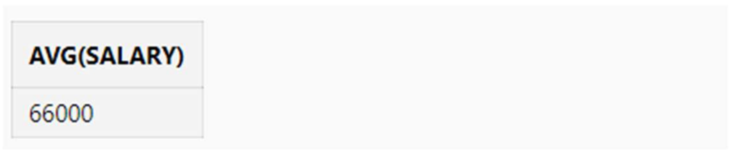
❖ **SQL HAVING Syntax:**

```
SELECT column_name, aggregate_function(column_name)
FROM table_name
WHERE column_name operator value
GROUP BY column_name
HAVING [conditions]
      ORDER BY column;
```

Exercise 2 on aggregate function, group by and having clause:

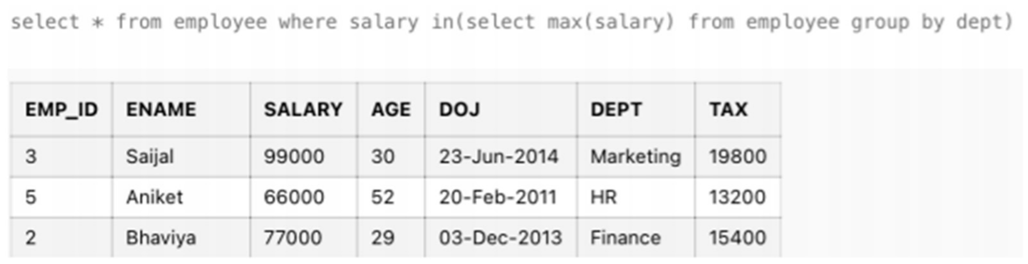
1. Find average salary of all employees for a particular department.

```
select avg(salary) from employee where dept='Finance'
```



AVG(SALARY)
66000

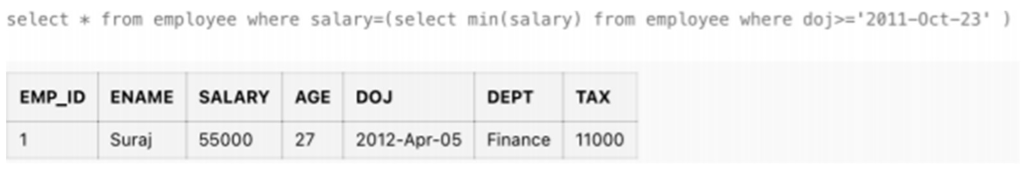
2. Find the details of that employee who has maximum salary in all departments.



```
select * from employee where salary in(select max(salary) from employee group by dept)
```

EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
3	Saijal	99000	30	23-Jun-2014	Marketing	19800
5	Aniket	66000	52	20-Feb-2011	HR	13200
2	Bhaviya	77000	29	03-Dec-2013	Finance	15400

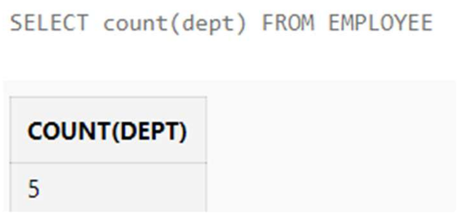
3. Find the details of employee who has minimum salary and who has joined after 23-oct-2011.



```
select * from employee where salary=(select min(salary) from employee where doj>='2011-Oct-23' )
```

EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
1	Suraj	55000	27	2012-Apr-05	Finance	11000

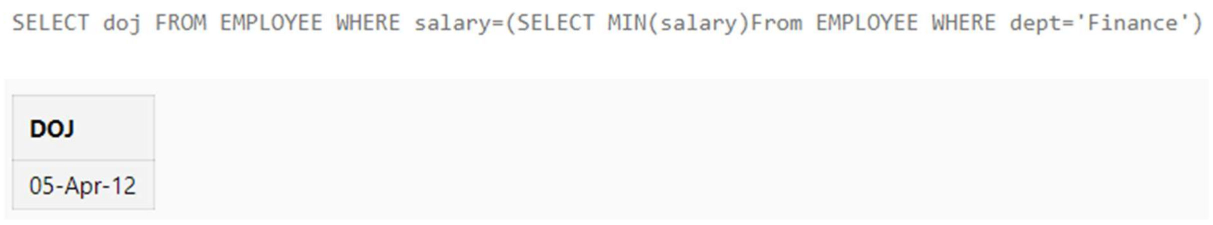
4. What are the total numbers of rows in the Employee table?



```
SELECT count(dept) FROM EMPLOYEE
```

COUNT(DEPT)
5

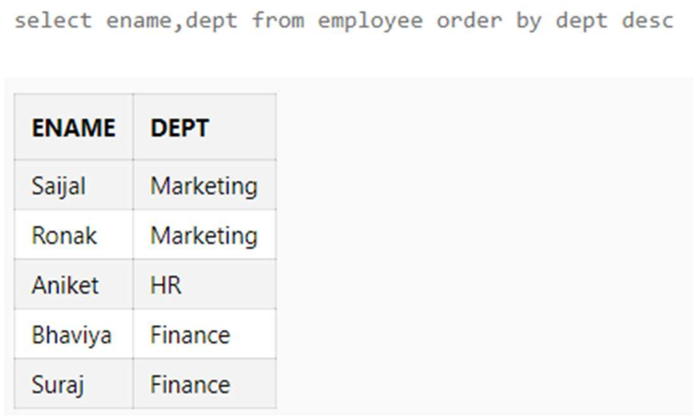
5. For all of the employees in computer department, what is the joining date of an employee with lowest salary in that department?



```
SELECT doj FROM EMPLOYEE WHERE salary=(SELECT MIN(salary)From EMPLOYEE WHERE dept='Finance')
```

DOJ
05-Apr-12

6. Display the name of each employee department wise and order it in descending order.

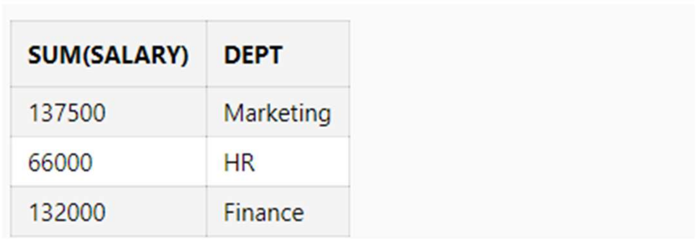


```
select ename,dept from employee order by dept desc
```

ENAME	DEPT
Saijal	Marketing
Ronak	Marketing
Aniket	HR
Bhaviya	Finance
Suraj	Finance

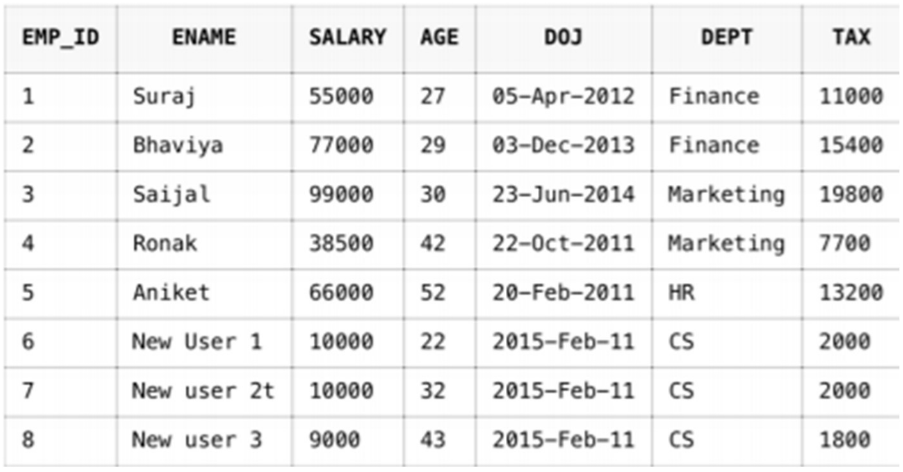
7. Find the total salary of all employees for each department.

```
SELECT sum(salary),dept FROM EMPLOYEE GROUP BY dept
```



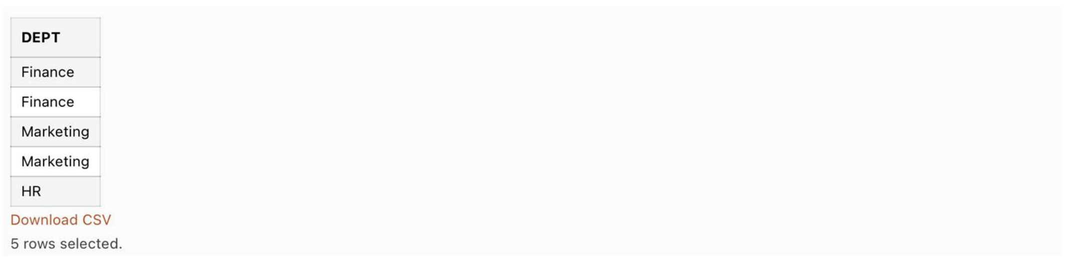
SUM(SALARY)	DEPT
137500	Marketing
66000	HR
132000	Finance

8. Find the names of all departments where the average salary of employee is more than 30,000.



EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
1	Suraj	55000	27	05-Apr-2012	Finance	11000
2	Bhaviya	77000	29	03-Dec-2013	Finance	15400
3	Saijal	99000	30	23-Jun-2014	Marketing	19800
4	Ronak	38500	42	22-Oct-2011	Marketing	7700
5	Aniket	66000	52	20-Feb-2011	HR	13200
6	New User 1	10000	22	2015-Feb-11	CS	2000
7	New user 2t	10000	32	2015-Feb-11	CS	2000
8	New user 3	9000	43	2015-Feb-11	CS	1800

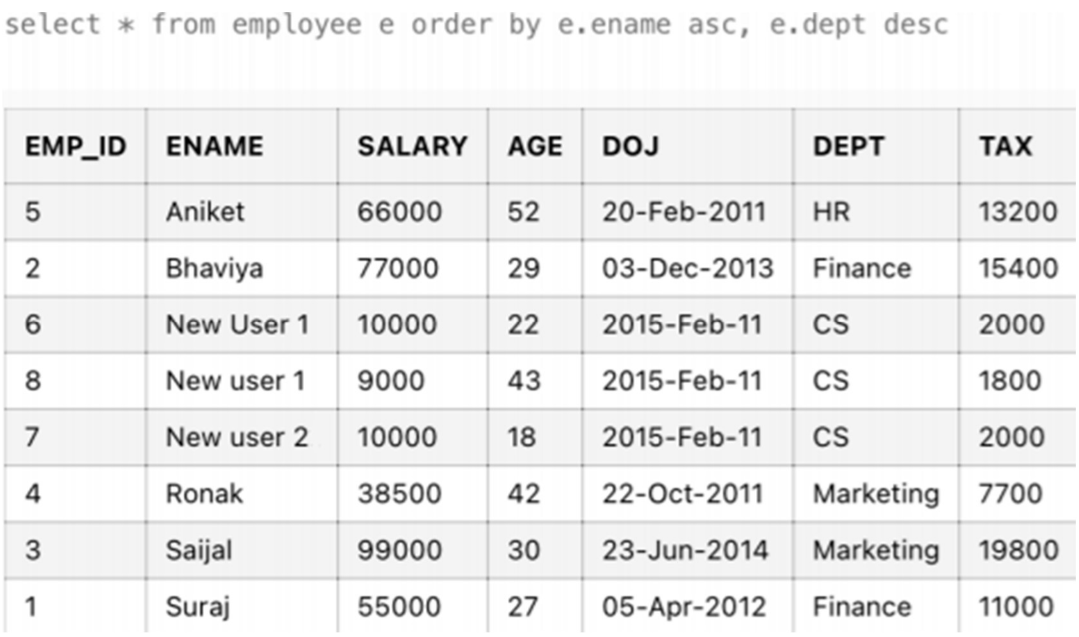
```
select dept from employee where salary in (select salary from employee group by salary having avg(salary)>30000)
```



DEPT
Finance
Finance
Marketing
Marketing
HR

Download CSV
5 rows selected.

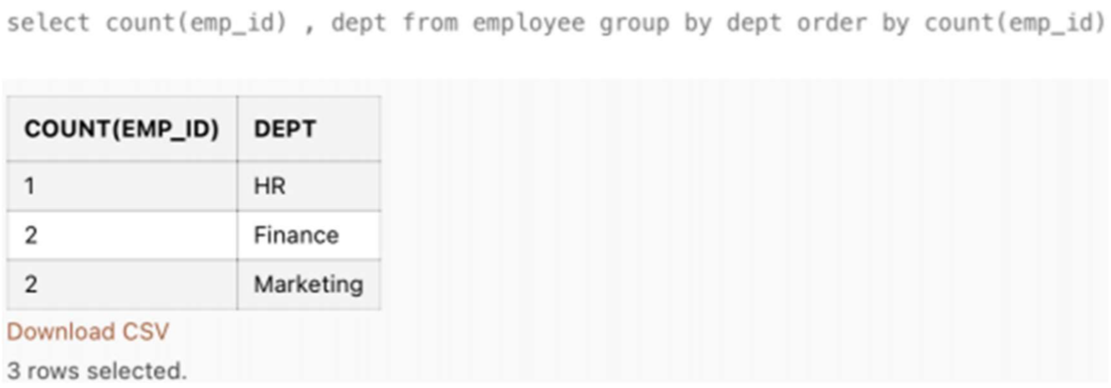
9. Sort Employee name in ascending order, and if Employee name is same, then it is sorted Department wise in descending order.



```
select * from employee e order by e.ename asc, e.dept desc
```

EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
5	Aniket	66000	52	20-Feb-2011	HR	13200
2	Bhaviya	77000	29	03-Dec-2013	Finance	15400
6	New User 1	10000	22	2015-Feb-11	CS	2000
8	New user 1	9000	43	2015-Feb-11	CS	1800
7	New user 2	10000	18	2015-Feb-11	CS	2000
4	Ronak	38500	42	22-Oct-2011	Marketing	7700
3	Saijal	99000	30	23-Jun-2014	Marketing	19800
1	Suraj	55000	27	05-Apr-2012	Finance	11000

10. Count the number of Employees in each department.



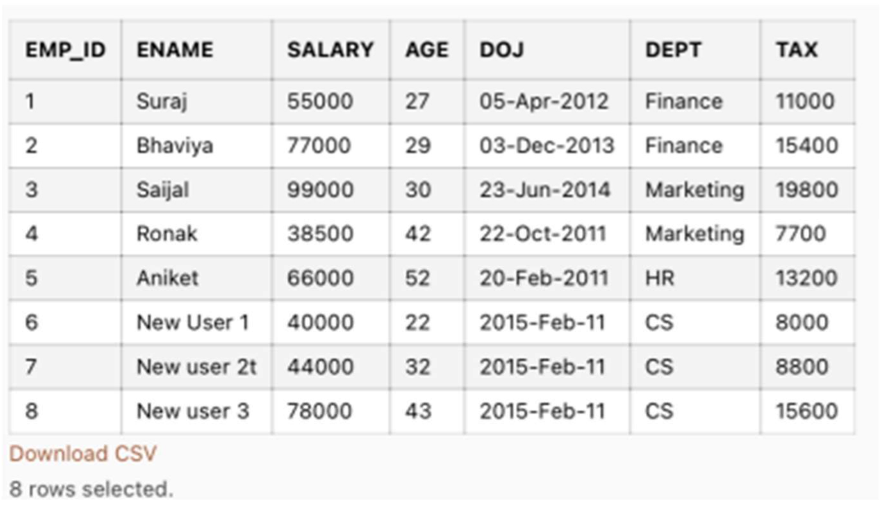
```
select count(emp_id) , dept from employee group by dept order by count(emp_id)
```

COUNT(EMP_ID)	DEPT
1	HR
2	Finance
2	Marketing

Download CSV
3 rows selected.

11. Display the name of Employees who have joined on the same date.

```
select * from employee
```

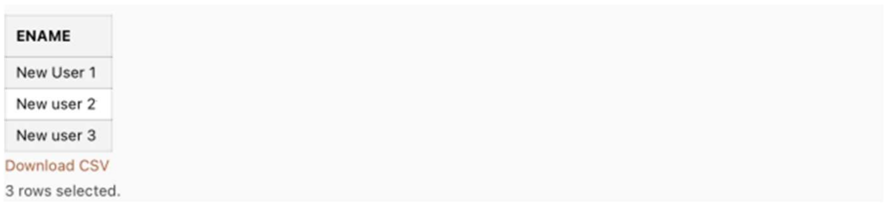


EMP_ID	ENAME	SALARY	AGE	DOJ	DEPT	TAX
1	Suraj	55000	27	05-Apr-2012	Finance	11000
2	Bhaviya	77000	29	03-Dec-2013	Finance	15400
3	Saijal	99000	30	23-Jun-2014	Marketing	19800
4	Ronak	38500	42	22-Oct-2011	Marketing	7700
5	Aniket	66000	52	20-Feb-2011	HR	13200
6	New User 1	40000	22	2015-Feb-11	CS	8000
7	New user 2t	44000	32	2015-Feb-11	CS	8800
8	New user 3	78000	43	2015-Feb-11	CS	15600

Download CSV

8 rows selected.

```
select ename from employee where doj in (select doj from employee group by doj having count(*)>1)
```



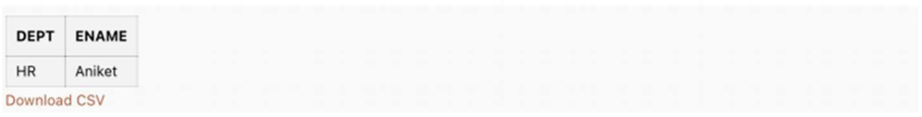
ENAME
New User 1
New user 2
New user 3

Download CSV

3 rows selected.

12. Display the name of employees who have less than 2 employees in a particular department.

```
select dept,ename from employee where dept in (select dept from employee group by dept having count(*)<2)
```



DEPT	ENAME
HR	Aniket

Download CSV

Conclusion: From the above experiment we are able to learn about different aggregate functions like HAVING clause and GROUP By clause. We are easily able to learn to find average, minimum, maximum, sum and count functions in SQL. The different set operators can also be learnt easily like arithmetic operators, Comparison/Relational operators, Logical/Boolean operators.