EXPERIMENT 13

LAB INTERNAL 2 - QUESTIONS

AIM : Using the below tables we need to retrieve data from tables for the given Queries.

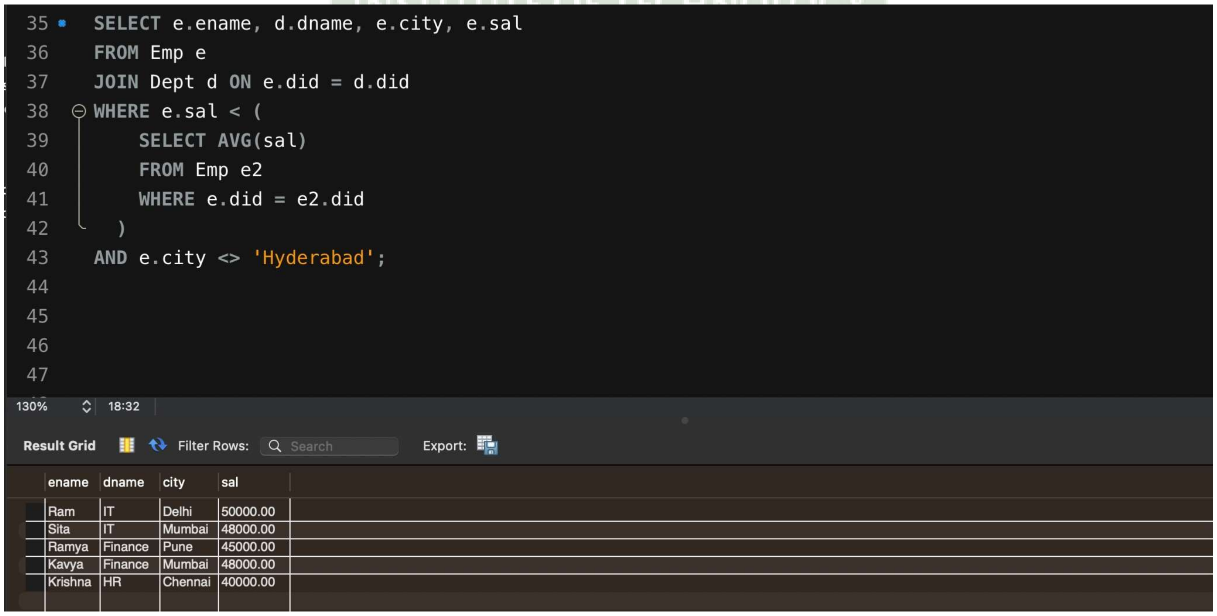
GIVEN TABLES :

1. Consider tables Emp and Dept
Emp (eid, ename, sal, city, did)
Dept (did, dname)
Find enames , concern dname , city, sal of those who are earning less than average salary of their own department and lives in other than Hyderabad

CODE :

```
SELECT e.ename, d.dname, e.city, e.sal

FROM Emp e JOIN Dept d ON e.did = d.did

WHERE e.sal < ( SELECT AVG(sal)

FROM Emp e2   WHERE e.did = e2.did)AND e.city <> 'Hyderabad';
```

OUTPUT:



```
SELECT e.ename, d.dname, e.city, e.sal
FROM Emp e
JOIN Dept d ON e.did = d.did
WHERE e.sal < (
    SELECT AVG(sal)
    FROM Emp e2
    WHERE e.did = e2.did
  )
AND e.city <> 'Hyderabad';
```

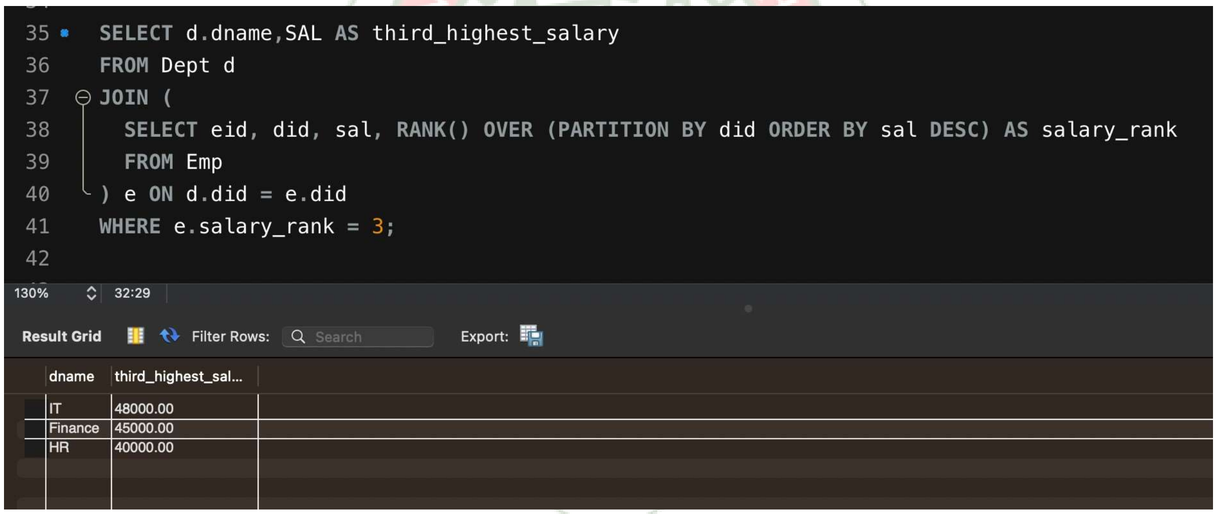
ename	dname	city	sal
Ram	IT	Delhi	50000.00
Sita	IT	Mumbai	48000.00
Ramya	Finance	Pune	45000.00
Kavya	Finance	Mumbai	48000.00
Krishna	HR	Chennai	40000.00

2. Consider tables Emp and Dept.
Emp (eid, ename, sal, did)
Dept (did, dname)
Find 3rd highest salary of each department. Output table should have two columns, one is dname and theother is 3rd highest salary of each department.

CODE :

```
SELECT d.dname,SAL AS third_highest_salary

FROM Dept d JOIN (SELECT eid, did, sal, RANK() OVER

(PARTITION BY did ORDER BY sal DESC) ASsalary_rank FROM Emp) as e

ON d.did =e.didWHERE e.salary_rank = 3;
```

OUTPUT:



```
SELECT d.dname,SAL AS third_highest_salary
FROM Dept d
JOIN (
  SELECT eid, did, sal, RANK() OVER (PARTITION BY did ORDER BY sal DESC) AS salary_rank
  FROM Emp
) e ON d.did = e.did
WHERE e.salary_rank = 3;
```

dname	third_highest_sal...
IT	48000.00
Finance	45000.00
HR	40000.00

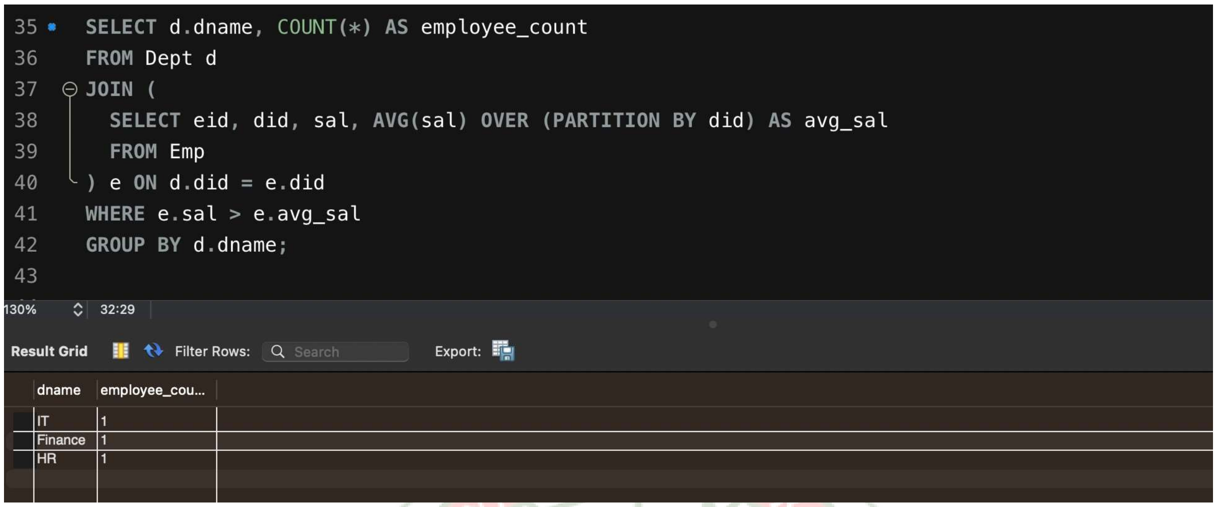
3. Consider tables Emp and Dept.
Emp (eid, ename, sal, did)
Dept (did, dname)
Find the number of employees in each department whose salary is greater than average salary of their own department. Output table should have two column one is dname and other is count of employees as said above.

CODE :-

```
SELECT d.dname, COUNT(*) AS employee_count

FROM Dept d JOIN ( SELECT eid, did, sal, AVG(sal) OVER (PARTITION BY did) AS avg_sal
```

FROM Emp) e ON d.did = e.did WHERE e.sal > e.avg_sal GROUP BY d.dname;

OUTPUT :



```
SELECT d.dname, COUNT(*) AS employee_count
FROM Dept d
JOIN (
  SELECT eid, did, sal, AVG(sal) OVER (PARTITION BY did) AS avg_sal
  FROM Emp
) e ON d.did = e.did
WHERE e.sal > e.avg_sal
GROUP BY d.dname;
```

dname	employee_cou...
IT	1
Finance	1
HR	1

4.A. Delete the Unique records (which are seen only once) from table in single shot.

4.B. Delete duplicate records (which are seen more than once) from table in single shot.

Eg :- Before execution table is:

COL1	COL2
X	Y
A	B
X	Y
K	L

AFTER execution of query A:

COL1	COL2
X	Y
X	Y

AFTER execution of query B:

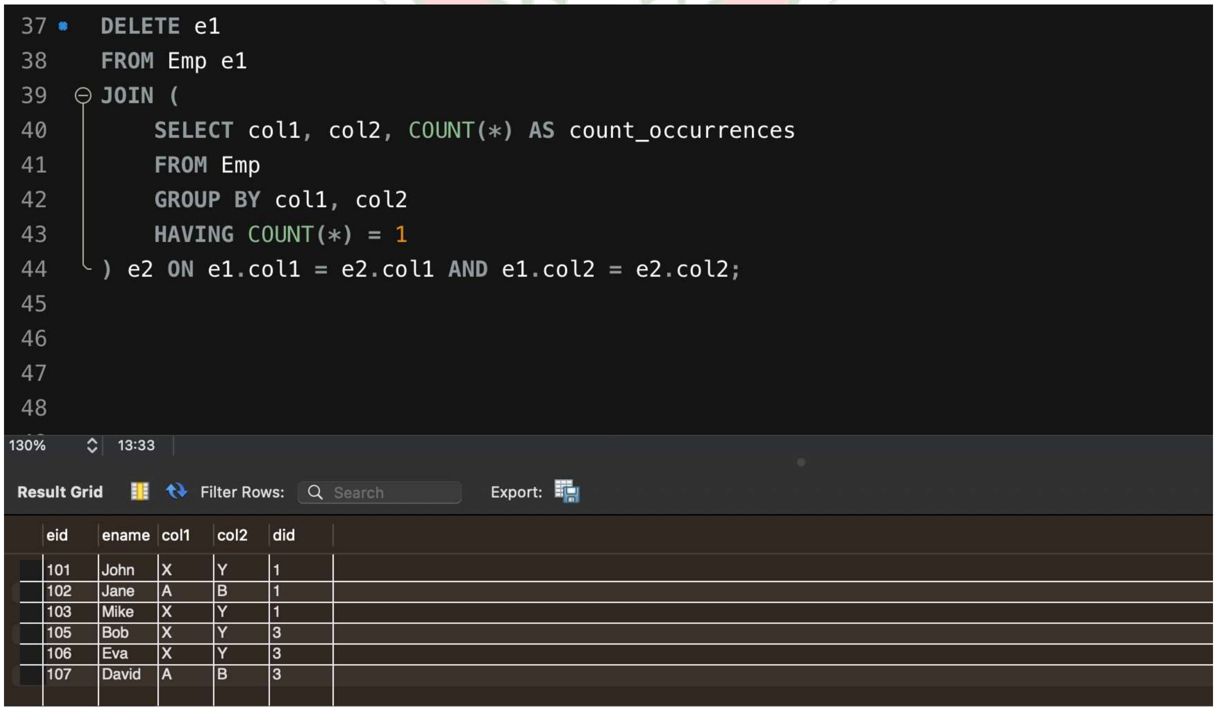
COL1	COL2
A	B
K	L

CODE : 4A

In this table we have take the unique record as 104 and deleted it:

```
DELETE e1FROM Emp e1

JOIN (SELECT col1, col2, COUNT(*) AS count_occurrences  FROM Emp

GROUP BY col1, col2 HAVING COUNT(*) = 1) e2 ON e1.col1 = e2.col1

AND   e1.col2 = e2.col2;
```

OUTPUT :



```
DELETE e1
FROM Emp e1
JOIN (
    SELECT col1, col2, COUNT(*) AS count_occurrences
    FROM Emp
    GROUP BY col1, col2
    HAVING COUNT(*) = 1
) e2 ON e1.col1 = e2.col1 AND e1.col2 = e2.col2;
```

130% 13:33

Result Grid Filter Rows: Search Export:

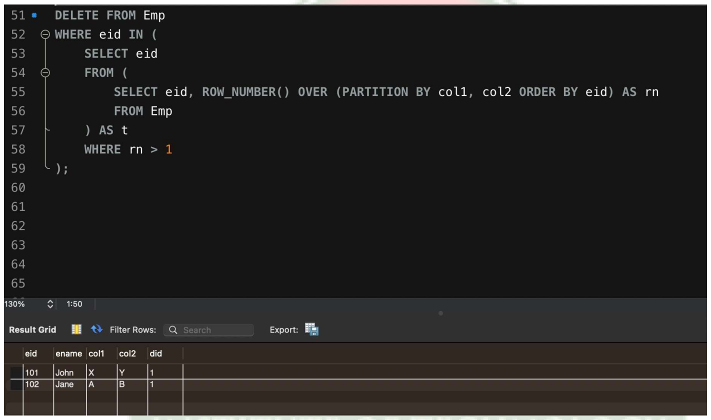
eid	ename	col1	col2	did
101	John	X	Y	1
102	Jane	A	B	1
103	Mike	X	Y	1
105	Bob	X	Y	3
106	Eva	X	Y	3
107	David	A	B	3

CODE : 4B

In this we have deleted all the remaining duplicate records.

```
DELETE FROM Emp

WHERE eid IN (SELECT eid FROM

(SELECT eid, ROW_NUMBER() OVER

(PARTITION BY col1, col2 ORDER BY eid) AS rn FROM Emp)

AS t WHERE rn > 1);
```

OUTPUT :



```
DELETE FROM Emp
WHERE eid IN (
    SELECT eid
    FROM (
        SELECT eid, ROW_NUMBER() OVER (PARTITION BY col1, col2 ORDER BY eid) AS rn
        FROM Emp
    ) AS t
    WHERE rn > 1
);
```

eid	ename	col1	col2	did
101	John	X	Y	1
102	Jane	A	B	1

RESULT : The queries have been successfully executed by using various concepts such as sub-queries, correlation sub-queries, and join operations.

****