**7BDIN006W.1 BIG DATA THEORY AND PRACTICE**

**Week 4 Seminar Tasks**

**Exercises on SQL**

Log in to Oracle Apex.

Consider the Oracle Employees / Departments database. Write SQL expressions to generate queries, ensuring they align with the below specifications.

Refer to the *Lecture 3 Notes* as a foundational guide for constructing your expressions.

*Note:* For these tasks, the corresponding solutions will not be provided immediately. They will be made available at the end of the week to give you ample opportunity to explore, attempt, and learn from this exercise. Utilise this chance to delve deeply into the task, experiment with different approaches, and solidify your understanding of SQL expressions and querying.

**a.** Display details of employees who work either as managers earning more than £1500 or as salesmen earning more than £1200.

SELECT \*

FROM emp

WHERE (job = 'MANAGER' AND sal > 1500)

OR (job = 'SALESMAN' AND sal > 1200);

**b.** Display details of employees who work either as clerks for dept. 10 or as managers.

SELECT \*

FROM emp

WHERE job = 'CLERK' AND deptno = 10

OR job = 'MANAGER';

**c.** Display details of employees who do not work as managers or for dept. 20.

SELECT \*

FROM emp

WHERE job != 'MANAGER' AND deptno <> 20;

--An alternative SQL statement can be the following

SELECT \*

FROM emp

WHERE NOT (job = 'MANAGER'

OR deptno = 20)

**d.** Display details of employees whose employee numbers are between 7500 and 7799

inclusive, and who work either for dept. 30 or for dept. 10.

SELECT \*

FROM emp

WHERE empno BETWEEN 7500 AND 7799

AND (deptno = 10

OR deptno = 30) -- you can use instead “=ANY” or “=SOME”

--An alternative SQL statement can be the following

SELECT \*

FROM emp

WHERE empno BETWEEN 7500 AND 7799

AND deptno IN (30, 10) -- you can use instead “=ANY” or “=SOME”

**e.** Display employees who earn more than 1200 in commission or salary.

SELECT \*

FROM emp

WHERE sal > 1200

OR comm > 1200

AND comm IS NOT NULL

--Considering that the IS NOT NULL predicate is not really required

(WHY?), an alternative SQL expression can be the following

SELECT \*

FROM emp

WHERE sal > 1200

OR comm > 1200

*NB:* The “comm IS NOT NULL” condition is not necessary because even if the value of comm IS NULL (i.e. the “comm > 1200 AND comm IS NOT NULL” condition evaluates to NULL, because of the “comm >1200”), the outcome of the WHERE clause will depend on the outcome of the “sal > 1200” condition.

**f.** Display the employees whose combined earnings is more than 1500.

SELECT \*

FROM emp

WHERE (comm IS NULL

AND sal+comm > 1500)

OR (comm IS NOT NULL

AND sal > 1500);

--An alternative SQL statement can be the following

SELECT \*

FROM emp

WHERE sal+comm > 1500

OR sal > 1500;

*NB:* The NULL predicates are not really required – see note in question

(f) above).

**g.** Display details of employees whose empno value is less than that of their manager's.

SELECT empno, mgr

FROM emp

WHERE mgr < empno;

**h.** Display details of employees whose names are four letters long.

SELECT \*

FROM emp

WHERE ename LIKE '\_\_\_\_'; -- the string is made up of four "\_"

**i.** Display details of employees whose names do not contain the letter ‘R’.

SELECT \*

FROM emp

WHERE ename NOT LIKE '%R%';

--An alternative SQL statement can be the following

SELECT \*

FROM emp

WHERE NOT ename LIKE '%R%';

--If the query was asking to produce employees who names contains

both the letter “R” and the letter “N” then the following could

be a possible answer:

SELECT \*

FROM emp

WHERE ename LIKE '%R%'

and ename LIKE '%N%';

--An equally correct answer can be the following:

SELECT \*

FROM emp

WHERE ename LIKE '%R%N%'

and ename LIKE '%N%R%';

**j.** Display names of employees, whose name the penultimate character is ‘E’.

SELECT \*

FROM emp

WHERE ename LIKE '%E\_';

**k.** Display employees who work for the Sales department.

SELECT \*

FROM emp

WHERE deptno =

(SELECT deptnoFROM dept

WHERE dname = 'SALES');

--Alternative expressions can be the following

SELECT \*

FROM emp, dept

WHERE emp.deptno = dept.deptno

AND dname = 'SALES';

SELECT \*

FROM emp NATURAL JOIN dept

WHERE dname = 'SALES';

SELECT \*

FROM emp JOIN dept

USING (deptno)

WHERE dname = 'SALES';

**l.** Display details of employees who either have the same job description as that of JONES or whose salary is greater or equal to that of FORD.

If there is only one employee called FORD and only one employee called JONES in the

emp table, then the following expression will return the correct results.

SELECT \*

FROM emp

WHERE job =

(SELECT job

FROM emp

WHERE ename = 'JONES')

OR sal >=

(SELECT sal

FROM emp

WHERE ename = 'FORD');

A better solution (that does not require unique employee names) is the following. You may want to consider how you can use "ALL" instead of "ANY" in relation to the job predicate.

SELECT \*

FROM emp

WHERE job = ANY(SELECT job

FROM emp

WHERE ename = 'JONES')

OR sal >= ANY

(SELECT sal

FROM emp

WHERE ename = 'FORD');

An alternative SQL statement can be the following

SELECT e.\*

FROM emp e CROSS JOIN emp o

WHERE (e. job = o.job

AND o.ename = 'JONES')

OR (e.sal >= o.sal

AND o.ename = 'FORD');

**m.** Display details of employees whose job description is among the job descriptions of employees working for a department located in CHICAGO.

SELECT \*

FROM emp

WHERE job =

(SELECT job

FROM emp

WHERE deptno IN

(SELECT deptno

FROM dept

WHERE loc = 'CHICAGO');

**n.** Display details of employees who work in N.Y. or CHICAGO.

SELECT \*

FROM emp

WHERE deptno IN

(SELECT deptno

FROM dept

WHERE loc = ANY

('NEW YORK', 'CHICAGO'));

SELECT emp.\*

FROM emp, dept –- an alternative is to use “emp CROSS JOIN dept”

WHERE emp.deptno = dept.deptnoAND (loc = 'NEW YORK'

OR loc = 'CHICAGO');

The following SQL expression are also equivalent to the above

SELECT emp.\*

FROM emp CROSS JOIN dept

WHERE emp.deptno = dept.deptno

AND loc = ANY -- instead of "=ANY" you can use "IN"

('NEW YORK', 'CHICAGO');

The following expression that involves the use of the EXISTS predicate is a better answer as it overcomes the problem of subqueries that return no rows (You may want to read about queries returning no results)

SELECT \*

FROM emp

WHERE EXISTS

(SELECT \*

FROM dept

WHERE emp.deptno = deptno

AND loc = ANY

('NEW YORK', 'CHICAGO'));

The above SQL statement also demonstrates the scope of SQL identifiers; i.e. the column names of the emp table mentioned in the outer query are accessible from any subqueries specified in the WHERE clause. Thus, in the WHERE clause of the subquery query to disambiguate to which deptno column we refer to we need to qualify it with the name of the table it comes from, e.g. “WHERE emp.deptno = dept.deptno”; however, as the dept.deptno refers to a column of the dept table, a table that is specified in the subquery, there is no need to qualify the column as the DBMS will assume by default that we are referring to the dept.deptno, hence “WHERE emp.deptno = deptno“.

Equivalent to the above answers that involve a “JOIN” expression can be the following (a “\*” is used in the SELECT clause a single copy of all common columns and all other columns of the two tables will be returned):

SELECT \*

FROM emp NATURAL JOIN dept –- or INNER NATURAL JOIN

WHERE loc IN ('NEW YORK', 'CHICAGO');

SELECT \*

FROM emp INNER JOIN dept -- the keyword INNER can be omitted

USING (deptno)

WHERE loc IN ('NEW YORK', 'CHICAGO');

*NB:* the JOIN with a USING is a form of NATURAL JOIN where only the listed common columns participate in the “natural” join.

SELECT \*

FROM emp INNER JOIN dept -- the keyword INNER can be omitted

ON (emp.deptno = dept.deptno)

WHERE loc IN ('NEW YORK', 'CHICAGO');

**o.** Display details of employees in grade 3 who work in CHICAGO.

SELECT \*

FROM emp, dept, salgrade

WHERE emp.deptno = dept.deptno

AND sal between losal and hisal

AND loc = 'CHICAGO'

AND grade = 3;

An equivalent (but computationally more efficient) SQL statement (as selections are done first reducing the size of the tables involved) can be the following

SELECT \*

FROM emp,

(SELECT \*

FROM dept

WHERE loc = 'CHICAGO') d,

(SELECT \*

FROM salgrade

WHERE grade = 3) salg,

WHERE emp.deptno = dept.deptno

AND sal between losal and hisal

An equivalent SQL statement using JOINs can be the following

SELECT \*

FROM (emp NATURAL JOIN dept)

JOIN salgrade

ON sal BETWEEN losal AND hisal

WHERE loc = 'CHICAGO'

AND grade = 3;

or the following

SELECT \*

FROM emp

NATURAL JOIN

(SELECT \*FROM dept

WHERE loc = 'CHICAGO')

JOIN

(SELECT \*

FROM salgrade

WHERE grade = 3)

ON sal BETWEEN losal AND hisal

**p.** Display details of employees in grade 3 or 4, who work in N.Y. or CHICAGO, whose job description is SALESMAN and whose name contains the letter “N”.

Some equivalent SQL expressions that answer the above query can be

SELECT \*

FROM emp, dept, salgrade

WHERE emp.deptno = dept.deptno

AND sal between losal and hisal

AND (loc = 'NEW YORK'

OR loc = 'CHICAGO') –- or use the IN, =ANY, =SOME

AND grade IN (3, 4)

AND job = 'SALESMAN'

AND ename LIKE '%N%';

SELECT \*

FROM emp CROSS JOIN dept CROSS JOIN salgrade

WHERE emp.deptno = dept.deptno

AND sal between losal and hisal

AND (loc = 'NEW YORK'

OR loc = 'CHICAGO') –- or use the IN, =ANY, =SOME

AND grade IN (3, 4)

AND job = 'SALESMAN'

AND ename LIKE '%N%';

SELECT \*

FROM (emp NATURAL JOIN dept)

JOIN salgrade

ON sal BETWEEN losal AND hisal

WHERE loc =ANY ('NEW YORK', 'CHICAGO')

AND grade IN (3, 4)

AND job = 'SALESMAN'

AND ename LIKE '%N%'

SELECT \*

FROM (SELECT \*

FROM emp

WHERE job = 'SALESMAN'

AND ename LIKE '%N%')

NATURAL JOIN

(SELECT \*

FROM dept

WHERE loc =ANY ('NEW YORK', 'CHICAGO'))

JOIN

(SELECT \*

FROM salgrade

WHERE grade IN (3, 4))

ON sal BETWEEN losal AND hisal