1.

−Produce a report that returns the last name, salary, department number, and average salary of all the departments where salary is greater than the average salary

SELECT e.LAST\_NAME,

e.SALARY,

e.DEPARTMENT\_ID,

d.SALAVG

FROM employees e

JOIN (

SELECT DEPARTMENT\_ID,

AVG(SALARY) AS SALAVG

FROM employees

GROUP BY DEPARTMENT\_ID

) d ON e.DEPARTMENT\_ID = d.DEPARTMENT\_ID

WHERE e.SALARY > d.SALAVG;

2.

−Create a view named V2 that returns the highest salary, lowest salary, average salary, and department name

CREATE VIEW V2 AS

SELECT d.DEPARTMENT\_NAME,

MIN(e.SALARY) AS LOWEST\_SALARY,

MAX(e.SALARY) AS HIGHEST\_SALARY,

AVG(e.SALARY) AS AVERAGE\_SALARY

FROM emp e

JOIN dept d ON e.DEPARTMENT\_ID = d.DEPARTMENT\_ID

GROUP BY d.DEPARTMENT\_NAME;

3.

−Create a view named Dept\_Managers\_view that returns a listing of department names along with the manager initial and surname for that department

−Test the view by returning all the rows from it

−Make sure no rows can be updated through the view

−Try to run an UPDATE statement against the view

**Create view**

CREATE VIEW Dept\_Managers\_view AS

SELECT d.DEPARTMENT\_NAME AS DEPT\_NAME,

SUBSTR(e.FIRST\_NAME, 1, 1) || '.' || e.LAST\_NAME AS MGR\_NAME

FROM departments d

JOIN employees e ON d.MANAGER\_ID = e.EMPLOYEE\_WITH READ ONLY;

**Test view**

SELECT \* FROM Dept\_Managers\_view;

**Try to Run an UPDATE Statement Against the View**

UPDATE Dept\_Managers\_view

SET MGR\_NAME = 'D.Doe'

WHERE DEPT\_NAME = 'IT';

4.

−The following statement contains errors −Fix them and run the code to get the displayed result

DROP VIEW V3;

5.

−Create a sequence named ct\_seq with all the default values Run the statements and fix the error −Correct the statement to return the subsequent number

SELECT ct\_seq.NEXTVAL

FROM dual;

SELECT ct\_seq.CURRVAL

FROM dual;

6.

−Look at the insert statement and fix the error

• Code: INSERT emp (employee\_id, first\_name, last\_name, email, phone\_number, hire\_date, job\_id, salary, commission\_pct, manager\_id, department\_id) VALUES (currval.ct\_seq, 'Kaare', 'Hansen', 'KHANSEN', '4496583212', sysdate, 'Manager', 6500, null, 100, 10)

DECLARE

new\_id NUMBER;

BEGIN

SELECT ct\_seq.NEXTVAL INTO new\_id FROM dual;

sequence value

INSERT INTO emp

(employee\_id, first\_name, last\_name, email, phone\_number, hire\_date, job\_id, salary, commission\_pct, manager\_id, department\_id)

VALUES

(new\_id, 'Kaare', 'Hansen', 'KHANSEN', '4496583212', SYSDATE, 'Manager', 6500, NULL, 100, 10);

END;

7.

−Fix the error in the SQL statement to create the index as shown in the screenshot

• Code: TABLE\_ NAME INDEX\_NAME INDEX\_TYPE COLUMN\_EXPRESSION COLUMN\_ POSITION EMP EMP\_INDX FUNCTIONBASED NORMAL "EMPLOYEE\_ID" 1 EMP EMP\_INDX FUNCTIONBASED NORMAL UPPER(SUBSTR("FIRST\_NAME",1,1)||' '||"LAST\_NAME") 2

CREATE INDEX EMP\_INDX ON EMP (

EMPLOYEE\_ID,

UPPER(SUBSTR(FIRST\_NAME, 1, 1) || ' ' || LAST\_NAME)

);

8.

−Write the SQL statement to list all the user tables which contain the name PRIV

• Tables Used: −dictionary

SELECT TABLE\_NAME

FROM USER\_TABLES

WHERE TABLE\_NAME LIKE '%PRIV%';

9.

−Give select access to public on the EMP table, and verify the grant by running this query. The query contains errors that you must fix before you can run the select statement

• Code: GRANTEE OWNER TABLE\_NAME GRANTOR PRIVILEGE GRANTABLE HIERARCHY PUBLIC US\_A009EMEA815\_PLSQL\_T01 EMP US\_A009EMEA815\_PLSQL\_T01 SELECT NO NO

GRANT SELECT ON EMP TO PUBLIC;

SELECT GRANTEE, OWNER, TABLE\_NAME, PRIVILEGE, GRANTOR, GRANTABLE

FROM ALL\_TAB\_PRIVS

WHERE TABLE\_NAME = 'EMP'

AND GRANTEE = 'PUBLIC';

10.

−Using Oracle proprietary joins, construct a statement that returns all the employee\_id's joined to all the department\_names

• Tables Used: −Employees, departments

SELECT e.employee\_id, d.department\_name

FROM employees e, departments d;

11.

− Still using Oracle Joins, correct the previous statement so that it returns only the name of the department that the employee actually works in

• Tables Used: − Employees, departments

SELECT e.employee\_id, d.department\_name

FROM employees e

JOIN departments d

ON e.department\_id = d.department\_id;

12.

− Still using Oracle Joins, construct a query that lists the employees last name, the department name, the salary and the country name of all employees

• Tables Used: − Employees, departments, locations and countries

SELECT e.last\_name,

d.department\_name,

e.salary,

c.country\_name

FROM employees e

JOIN departments d

ON e.department\_id = d.department\_id

JOIN locations l

ON d.location\_id = l.location\_id

JOIN countries c

ON l.country\_id = c.country\_id;

13.

− Still using Oracle join syntax, alter the previous query so that it also includes the employee record of the employee with no department\_id, 'Grant'

• Tables Used: − Employees, departments, locations and countries

SELECT e.last\_name,

d.department\_name,

e.salary,

c.country\_name

FROM employees e

LEFT JOIN departments d

ON e.department\_id = d.department\_id

LEFT JOIN locations l

ON d.location\_id = l.location\_id

LEFT JOIN countries c

ON l.country\_id = c.country\_id

WHERE e.last\_name = 'Grant'

OR e.department\_id IS NULL;