Lab on Schema objects: ABIODUN OKE 117180166

Solve the following:

1. Create the EMPLOYEES2 table based on the structure of the EMPLOYEES table. Include only the EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, SALARY, and DEPARTMENT\_ID columns. Name the columns in your new table ID, FIRST\_NAME, LAST\_NAME, SALARY, and DEPT\_ID, respectively.

**ANSWER**:

CREATE TABLE EMPLOYEES2 AS

SELECT EMPLOYEE\_ID ID, FIRST\_NAME, LAST\_NAME, SALARY, DEPARTMENT\_ID DEPT\_ID

FROM EMPLOYEES;

1. Use the employees2 table that you created in the previous lab on DDL and Schema Objects. Alter the EMPLOYEES2 table status to read-only.

**ANSWER**:

ALTER TABLE EMPLOYEES2 READ ONLY;

1. Try to insert the following row in the table:

34, ‘Grant’, ‘Marcie’, 5678, 10

Show the result of executing the insert statement.

**OUTPUT**:

**\*Cause: An attempt was made to update a read-only materialized view.**

**\*Action: No action required. Only Oracle is allowed to update a**

**read-only materialized view.**

1. Revert the EMPLOYEES2 table to read/write status. Now try to insert the same row again.

Show the result.

**ALTER TABLE EMPLOYEES2 READ WRITE;**

**OUTPUT:**

**SELECT \* FROM EMPLOYEES2**

**ID FIRST\_NAME LAST\_NAME SALARY DEPT\_ID**

**---------- -------------------- ------------------------- ---------- ----------**

**34 Grant Marice 5678 10**

1. Drop the EMPLOYEES2 table.

**ANSWER: DROP TABLE EMPLOYEES2**;

1. The staff in the HR department wants to hide some of the data in the EMPLOYEES table. They want a view called EMPLOYEES\_VU based on the employee numbers, employee last names, and department numbers from the EMPLOYEES table. They want the heading for the employee name to be EMPLOYEE.

**ANSWER:**

**CREATE VIEW EMPLOYEES\_VU**

**AS (SELECT EMPLOYEE\_ID, LAST\_NAME AS "EMPLOYEE", DEPARTMENT\_ID**

**FROM EMPLOYEES);**

1. Confirm that the view works. Display the contents of the EMPLOYEES\_VU view.

**OUTPUT:**

EMPLOYEE\_ID EMPLOYEE DEPARTMENT\_ID

----------- ------------------------- -------------

100 King 90

101 Kochhar 90

102 De Haan 90

103 Hunold 60

104 Ernst 60

107 Lorentz 60

124 Mourgos 50

141 Rajs 50

142 Davies 50

143 Matos 50

144 Vargas 50

149 Zlotkey 80

174 Abel 80

176 Taylor 80

178 Grant

200 Whalen 10

201 Hartstein 20

202 Fay 20

205 Higgins 110

206 Gietz 110

20 rows selected

1. Using your EMPLOYEES\_VU view, write a query for the HR department to display all employee names and department numbers.

**OUTPUT:**

**EMPLOYEE DEPARTMENT\_ID**

**------------------------- -------------**

**King 90**

**Kochhar 90**

**De Haan 90**

**Hunold 60**

**Ernst 60**

**Lorentz 60**

**Mourgos 50**

**Rajs 50**

**Davies 50**

**Matos 50**

**Vargas 50**

**Zlotkey 80**

**Abel 80**

**Taylor 80**

**Grant**

**Whalen 10**

**Hartstein 20**

**Fay 20**

**Higgins 110**

**Gietz 110**

**20 rows selected**

1. Department 50 needs access to its employee data. Create a view named DEPT 50 that contains the **employee numbers, employee last names, and department numbers for all employees in department 50**. They have requested that you label the view columns EMPNO, EMPLOYEE, and DEPTNO. For security purposes, do not allow an employee to be reassigned to another department through the view.

**ANSWER:**

**CREATE VIEW DEPT50**

**AS**

**SELECT EMPLOYEE\_ID AS "EMPNO", LAST\_NAME AS "EMPLOYEE", DEPARTMENT\_ID AS "DEPTNO"**

**FROM EMPLOYEES**

**WHERE DEPARTMENT\_ID = 50**

**WITH CHECK OPTION CONSTRAINT EMP\_DEPT\_50;**

1. Display the structure and contents of the DEPT50 view.

**DESCRIBE DEPT50;**

**Name Null Type**

**-------- -------- ------------**

**EMPNO NOT NULL NUMBER(6)**

**EMPLOYEE NOT NULL VARCHAR2(25)**

**DEPTNO NUMBER(4)**

**OUTPUT:**

**EMPNO EMPLOYEE DEPTNO**

**---------- ------------------------- ----------**

**124 Mourgos 50**

**141 Rajs 50**

**142 Davies 50**

**143 Matos 50**

**144 Vargas 50**

1. Test your view. Attempt to reassign Matos to department 80.

**UPDATE DEPT50**

**SET DEPTNO = 80**

**WHERE EMPLOYEE = 'Matos';**

**OUTPUT:** view WITH CHECK OPTION where-clause violation

1. Create the DEPT table based on departments table. You need a sequence that can be used with the primary key column of the DEPT table. The sequence should start at 200 and have a maximum value of 1000. Have your sequence increment by 10. Name the sequence DEPT\_ID\_SEQ.

**ANSWER:**

**CREATE SEQUENCE DEPT\_ID\_SEQ**

**START WITH 200**

**INCREMENT BY 10**

**MAXVALUE 1000;**

1. To test your sequence, insert two rows in the DEPT table. Be sure to use the sequence that you created for the ID column. Add two departments: Education and Administration. Confirm your additions.

**INSERT INTO DEPT VALUES (DEPT\_ID\_SEQ.NEXTVAL, 'Education');**

**INSERT INTO DEPT VALUES (DEPT\_ID\_SEQ.NEXTVAL, 'Administration');**

1. Create a nonunique index on the NAME column in the DEPT table.

**CREATE INDEX DEPT\_NAME\_IDX ON DEPT(NAME);**

1. Create a synonym for your EMPLOYEES table. Call it EMP.

**CREATE SYNONYM EMP FOR EMPLOYEES;**