DEPARTMENT OF COMPUTER SYSTEMS ENGINEERING
MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
Database Management Systems (4th Semester) 18CS
Lab Experiment 09

Roll No:	**Date of Conduct:**
Submission Date:	**Grade Obtained:**

Problem Recognition (0.3)	Completeness & accuracy (0.4)	Timeliness (0.3)	Score (1.0)

Objective: To understand Indexes, Views & Sequences.

Tools: MySQL/Oracle.

Introduction:

INDEXES: A database index is a data structure that improves the speed of operations in a table. Indexes can be created using one or more columns. The users cannot see the indexes, they are just used to speed up queries and will be used by the Database Search Engine to locate records very fast.

- The INSERT and UPDATE statements take more time on tables having indexes, whereas the SELECT statements become fast on those tables.

- The reason is that while doing insert or update, a database needs to insert or update the index values as well.

VIEWS: In SQL, a view is a virtual table based on the result-set of an SQL statement. A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.

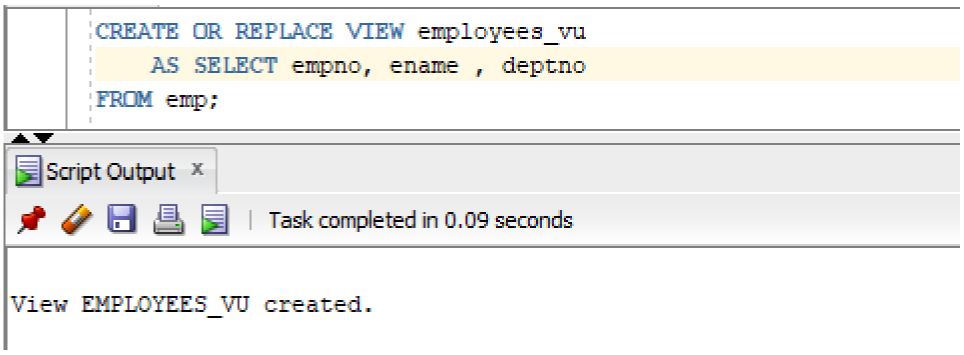
- You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.
- Note that a view does not physically store the data.

SEQUENCE: A sequence is a list of integers generated in the ascending order**.** Many applications need sequences to generate unique numbers mainly for identification

Lab Task

1. Create a view called emp_vu based on the employee number, employee name, and department number from the EMP table. Change the heading for the employee name to EMPLOYEE.

Task

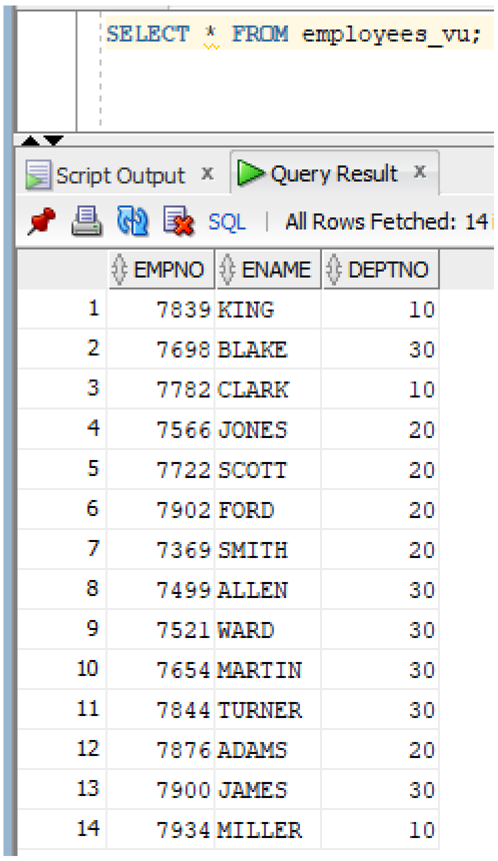


```
CREATE OR REPLACE VIEW employees_vu
    AS SELECT empno, ename , deptno
FROM emp;
```

```
View EMPLOYEES_VU created.
```

2. Display the contents of the EMP_VU view.

Task

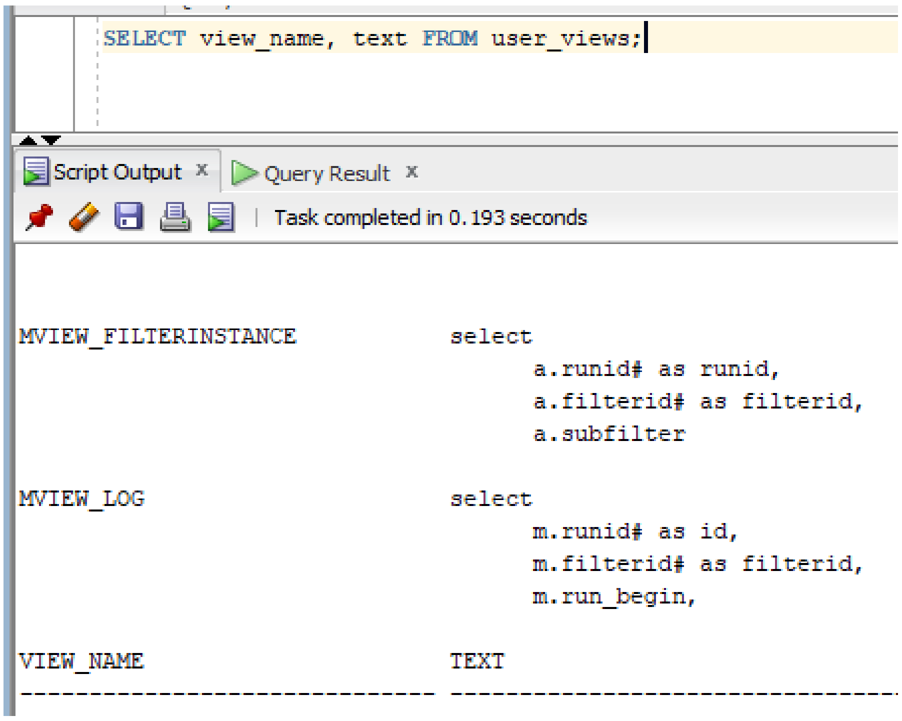


```
SELECT * FROM employees_vu;
```

	EMPNO	ENAME	DEPTNO
1	7839	KING	10
2	7698	BLAKE	30
3	7782	CLARK	10
4	7566	JONES	20
5	7722	SCOTT	20
6	7902	FORD	20
7	7369	SMITH	20
8	7499	ALLEN	30
9	7521	WARD	30
10	7654	MARTIN	30
11	7844	TURNER	30
12	7876	ADAMS	20
13	7900	JAMES	30
14	7934	MILLER	10

3. Select the view name and text from the data dictionary USER_views.

Task



```
SELECT view_name, text FROM user_views;
```

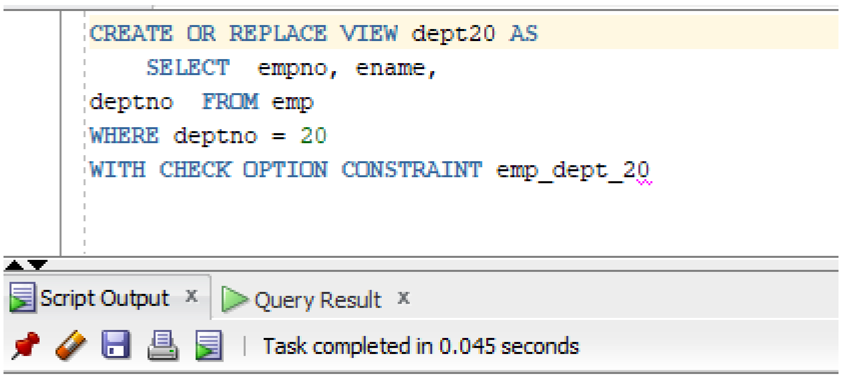
```
MVIEW_FILTERINSTANCE           select
                                      a.runid# as runid,
                                      a.filterid# as filterid,
                                      a.subfilter

MVIEW_LOG                      select
                                      m.runid# as id,
                                      m.filterid# as filterid,
                                      m.run_begin,

VIEW_NAME                      TEXT
------------------------------ --------------------------------
```

4. Create a view named DEPT_VU_20 that contains the employee number, employee name and department number for all employees in department 20. Label the view column Employee_Id, Employee, and Department_Id. Do not allow an employee to be reassigned to another department through the view.

Task

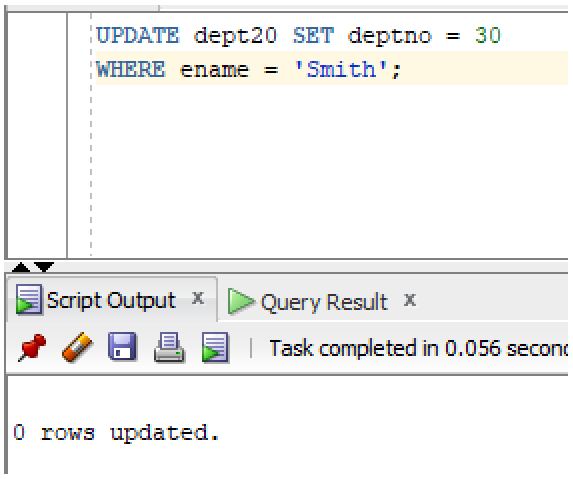


```
CREATE OR REPLACE VIEW dept20 AS
    SELECT  empno, ename,
deptno  FROM emp
WHERE deptno = 20
WITH CHECK OPTION CONSTRAINT emp_dept_20
```

```
View DEPT20 created.
```

5. Attempt to reassign Smith to department 30.

Task

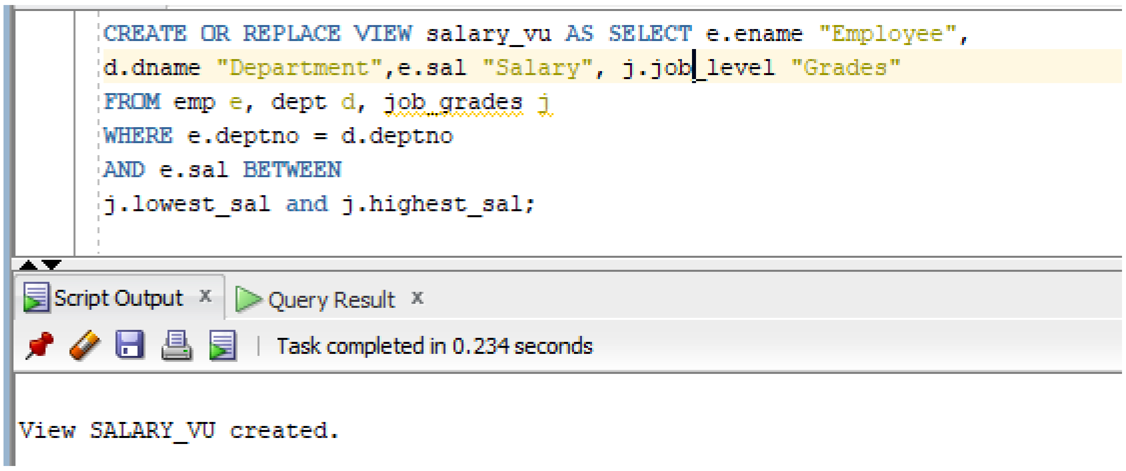


```
UPDATE dept20 SET deptno = 30
WHERE ename = 'Smith';
```

```
0 rows updated.
```

6. Create a view Salary_vu based on the employee name, department name, salary and salary grade for all employees. Label the columns Employee, Department, Salary and Grade.

Task

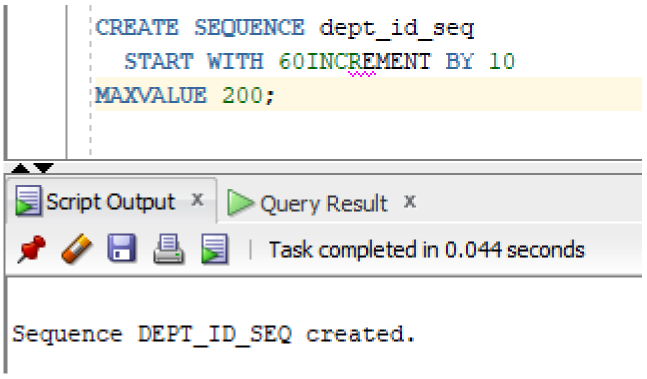


```
CREATE OR REPLACE VIEW salary_vu AS SELECT e.ename "Employee",
d.dname "Department",e.sal "Salary", j.job_level "Grades"
FROM emp e, dept d, job_grades j
WHERE e.deptno = d.deptno
AND e.sal BETWEEN
j.lowest_sal and j.highest_sal;
```

```
View SALARY_VU created.
```

7. Create a sequence to be used with primary key column of the department table. The sequence should start at 60 and have a maximum value of 200. Have your sequence increment by ten numbers. Name the sequence Dept_Id_Seq.

Task

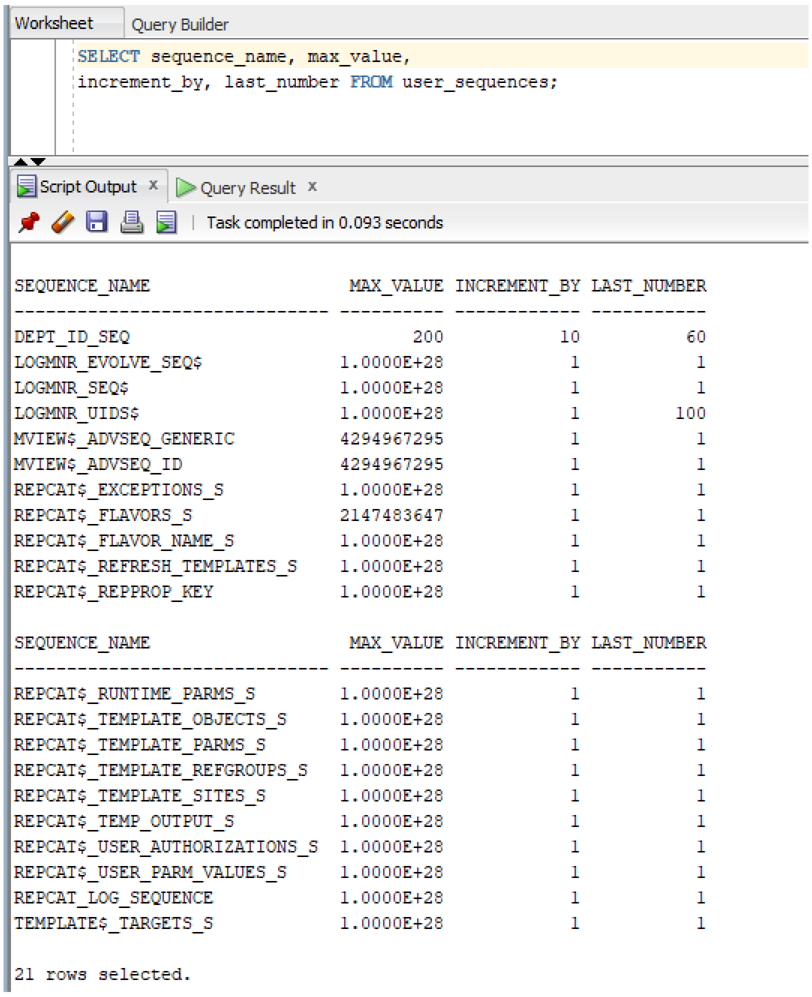


```
CREATE SEQUENCE dept_id_seq
  START WITH 60INCREMENT BY 10
MAXVALUE 200;
```

```
Sequence DEPT_ID_SEQ created.
```

8. Display the following information about your sequences: sequence name, max value, increment size, and last number.

Task



```
SELECT sequence_name, max_value,
increment_by, last_number FROM user_sequences;
```

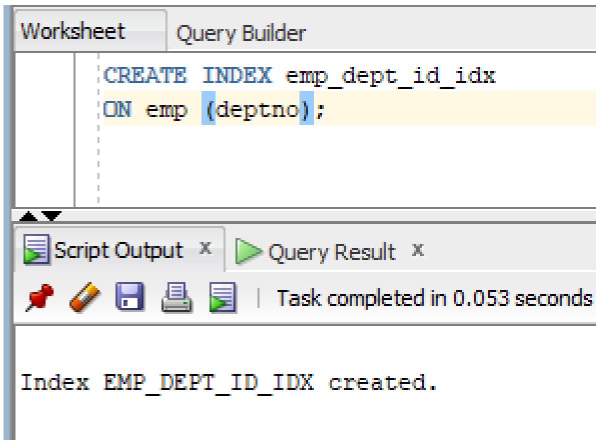
```
SEQUENCE_NAME                   MAX_VALUE INCREMENT_BY LAST_NUMBER
------------------------------ ---------- ------------ -----------
DEPT_ID_SEQ                           200           10          60
LOGMNR_EVOLVE_SEQ$             1.0000E+28            1           1
LOGMNR_SEQ$                    1.0000E+28            1           1
LOGMNR_UIDS$                   1.0000E+28            1         100
MVIEW$_ADVSEQ_GENERIC          4294967295            1           1
MVIEW$_ADVSEQ_ID               4294967295            1           1
REPCAT$_EXCEPTIONS_S           1.0000E+28            1           1
REPCAT$_FLAVORS_S              2147483647            1           1
REPCAT$_FLAVOR_NAME_S          1.0000E+28            1           1
REPCAT$_REFRESH_TEMPLATES_S    1.0000E+28            1           1
REPCAT$_REPPROP_KEY            1.0000E+28            1           1

SEQUENCE_NAME                   MAX_VALUE INCREMENT_BY LAST_NUMBER
------------------------------ ---------- ------------ -----------
REPCAT$_RUNTIME_PARMS_S        1.0000E+28            1           1
REPCAT$_TEMPLATE_OBJECTS_S     1.0000E+28            1           1
REPCAT$_TEMPLATE_PARMS_S       1.0000E+28            1           1
REPCAT$_TEMPLATE_REFGROUPS_S   1.0000E+28            1           1
REPCAT$_TEMPLATE_SITES_S       1.0000E+28            1           1
REPCAT$_TEMP_OUTPUT_S          1.0000E+28            1           1
REPCAT$_USER_AUTHORIZATIONS_S  1.0000E+28            1           1
REPCAT$_USER_PARM_VALUES_S     1.0000E+28            1           1
REPCAT_LOG_SEQUENCE            1.0000E+28            1           1
TEMPLATE$_TARGETS_S            1.0000E+28            1           1

21 rows selected.
```

9. Create a non-unique index on the foreign key column in the employee table.

Task

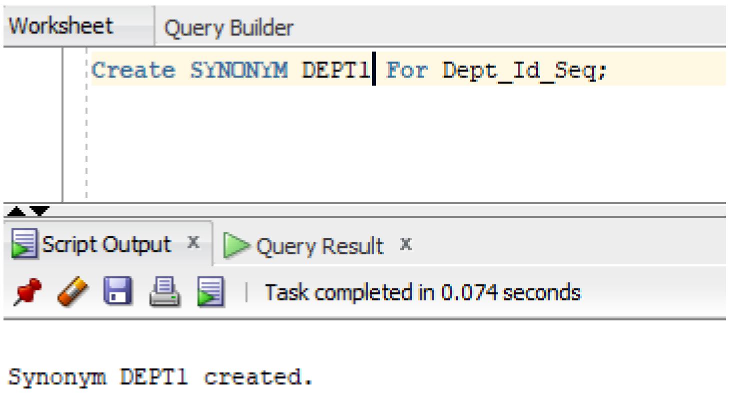


```
CREATE INDEX emp_dept_id_idx
ON emp (deptno);
```

```
Index EMP_DEPT_ID_IDX created.
```

10. Create a synonym for Dept_Id_Seq.

Task



```
Create SYNONYM DEPT1 For Dept_Id_Seq;
```

Task completed in 0.074 seconds

```
Synonym DEPT1 created.
```