Name : Prachi Mahavir Patil.

Roll No. : B27

PRN : 2122000442

Experiment No. 6

Advanced SQL

Oracle Sequences:

i) Create sequence on cus_code

```
CREATE TABLE customer_new (
cus_code INTEGER PRIMARY KEY,
cus_lname VARCHAR2(10),
cus_fname VARCHAR2(10),
cus_initial VARCHAR2(1),
cus_areacode INTEGER,     cus_phone
INTEGER,     cus_balance
NUMBER(10,2)
);


SELECT table_name
FROM user_tables
WHERE table_name = 'CUSTOMER';  -- Note: Oracle stores table names in uppercase by default


CREATE SEQUENCE customer_seq_new
```

START WITH 1

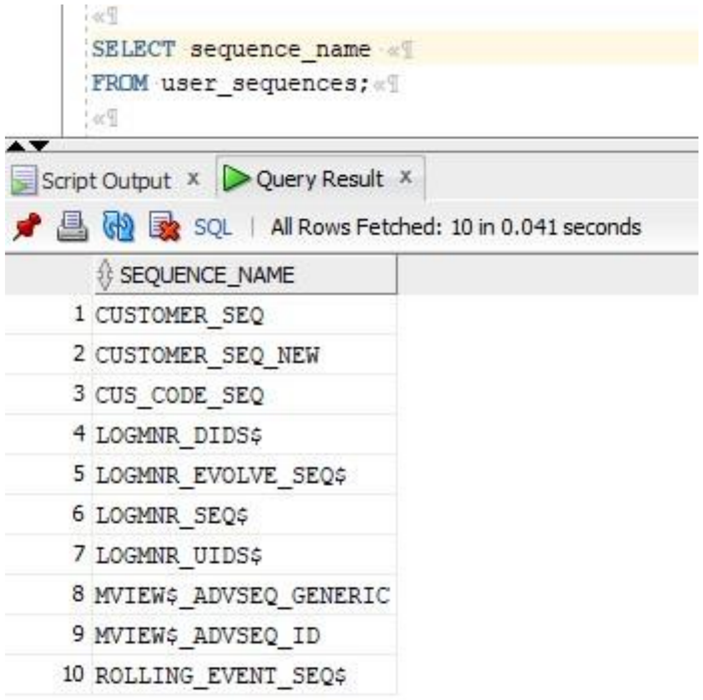
INCREMENT BY 1

NOCACHE;

ii) Display user sequences

SELECT sequence_name

FROM user_sequences;



```
SELECT sequence_name
FROM user_sequences;
```

Script Output × Query Result ×

SQL | All Rows Fetched: 10 in 0.041 seconds

```
   SEQUENCE_NAME
 1 CUSTOMER_SEQ
 2 CUSTOMER_SEQ_NEW
 3 CUS_CODE_SEQ
 4 LOGMNR_DIDS$
 5 LOGMNR_EVOLVE_SEQ$
 6 LOGMNR_SEQ$
 7 LOGMNR_UIDS$
 8 MVIEW$_ADVSEQ_GENERIC
 9 MVIEW$_ADVSEQ_ID
10 ROLLING_EVENT_SEQ$
```

iii) Insert values into customer using created sequence

INSERT INTO customer_new (cus_code, cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone, cus_balance)

VALUES (customer_seq_new.NEXTVAL, 'Doe', 'John', 'J', 123, 4567890, 1000.00);

INSERT INTO customer_new (cus_code, cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone, cus_balance)

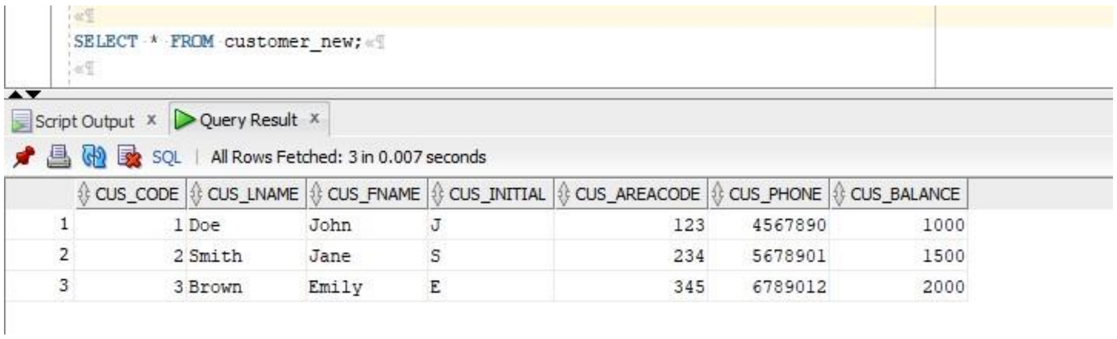
VALUES (customer_seq_new.NEXTVAL, 'Smith', 'Jane', 'S', 234, 5678901, 1500.00);

INSERT INTO customer_new (cus_code, cus_lname, cus_fname, cus_initial, cus_areacode, cus_phone, cus_balance)

VALUES (customer_seq_new.NEXTVAL, 'Brown', 'Emily', 'E', 345, 6789012, 2000.00);

iv) Display customer records

SELECT * FROM customer_new;



```
SELECT * FROM customer_new;
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 3 in 0.007 seconds

	CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_BALANCE
1	1	Doe	John	J	123	4567890	1000
2	2	Smith	Jane	S	234	5678901	1500
3	3	Brown	Emily	E	345	6789012	2000

Trigger:

CREATE TABLE student_report_new (tid INT

PRIMARY KEY,   name VARCHAR2(30),   subj1

INT CHECK (subj1 BETWEEN 0 AND 20),   subj2

INT CHECK (subj2 BETWEEN 0 AND 20),   subj3

INT CHECK (subj3 BETWEEN 0 AND 20),   total

INT,   per INT

);

CREATE OR REPLACE TRIGGER trg_student_report

BEFORE INSERT ON student_report_new

FOR EACH ROW

BEGIN

-- Calculate the total

:NEW.total := NVL(:NEW.subj1, 0) + NVL(:NEW.subj2, 0) + NVL(:NEW.subj3, 0);

-- Calculate the percentage

:NEW.per := (:NEW.total / 60) * 100; -- 60 is the total max marks (20*3) END;

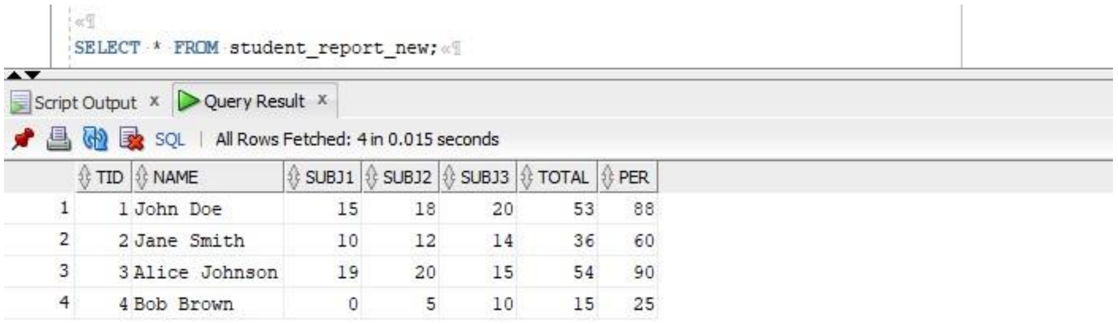
INSERT INTO student_report_new (tid, name, subj1, subj2, subj3) VALUES (1, 'John Doe', 15, 18, 20);

INSERT INTO student_report_new (tid, name, subj1, subj2, subj3) VALUES (2, 'Jane Smith', 10, 12, 14);

INSERT INTO student_report_new (tid, name, subj1, subj2, subj3) VALUES (3, 'Alice Johnson', 19, 20, 15);

INSERT INTO student_report_new (tid, name, subj1, subj2, subj3) VALUES (4, 'Bob Brown', 0, 5, 10);

SELECT * FROM student_report_new;



```
SELECT * FROM student_report_new;
```

All Rows Fetched: 4 in 0.015 seconds

	TID	NAME	SUBJ1	SUBJ2	SUBJ3	TOTAL	PER
1	1	John Doe	15	18	20	53	88
2	2	Jane Smith	10	12	14	36	60
3	3	Alice Johnson	19	20	15	54	90
4	4	Bob Brown	0	5	10	15	25

Procedure and Cursor:

i) Write a procedure which includes cursors: Find course_name and credits where course name starts with 'C' CREATE TABLE courseTable (course_num INTEGER PRIMARY KEY, course_name VARCHAR2(20),

dept_name VARCHAR2(15), credits

INTEGER

);

```
INSERT INTO courseTable (course_num, course_name, dept_name, credits) VALUES (101, 'Computer
Science', 'CSE', 4);

INSERT INTO courseTable (course_num, course_name, dept_name, credits) VALUES (102, 'Data
Structures', 'CSE', 3);

INSERT INTO courseTable (course_num, course_name, dept_name, credits) VALUES (103, 'Database
Systems', 'CSE', 3);

INSERT INTO courseTable (course_num, course_name, dept_name, credits) VALUES (104, 'Digital Logic',
'ECE', 3);

INSERT INTO courseTable (course_num, course_name, dept_name, credits) VALUES (105, 'Operating
Systems', 'CSE', 4);

INSERT INTO courseTable (course_num, course_name, dept_name, credits) VALUES (106, 'Computer
Networks', 'CSE', 4);

INSERT INTO courseTable (course_num, course_name, dept_name, credits) VALUES (107, 'Algorithms',
'CSE', 3);

INSERT INTO courseTable (course_num, course_name, dept_name, credits) VALUES (108, 'Embedded
Systems', 'ECE', 3);

INSERT INTO courseTable (course_num, course_name, dept_name, credits) VALUES (110, 'Software
Engineering', 'IT', 4);


CREATE OR REPLACE PROCEDURE find_courses_starting_with_C IS

  CURSOR c_courses IS

    SELECT course_name, credits

    FROM courseTable

    WHERE course_name LIKE 'C%';


     v_course_name courseTable.course_name%TYPE; v_credits

     courseTable.credits%TYPE;

BEGIN

  OPEN c_courses;
```

```
   LOOP
      FETCH c_courses INTO v_course_name, v_credits;
      EXIT WHEN c_courses%NOTFOUND;
      DBMS_OUTPUT.PUT_LINE('Course Name: ' || v_course_name || ', Credits: ' || v_credits);
   END LOOP;
   CLOSE c_courses;
END;
```

-- To find courses starting with 'C' EXEC find_courses_starting_with_C; **ii)**

Write a procedure which includes cursors: Find course names from 'CSE' department

```
CREATE OR REPLACE PROCEDURE find_courses_in_CSE IS
   CURSOR c_courses IS
      SELECT course_name
      FROM courseTable
      WHERE dept_name = 'CSE';
       v_course_name courseTable.course_name%TYPE;
BEGIN
   OPEN c_courses;
       LOOP
      FETCH c_courses INTO v_course_name;
      EXIT WHEN c_courses%NOTFOUND;
      DBMS_OUTPUT.PUT_LINE('Course Name: ' || v_course_name);
   END LOOP;
   CLOSE c_courses;
```

```
END;

-- To find courses in the 'CSE' department

EXEC find_courses_in_CSE;
```