ADS Experiment 6

Name: Digvijay Patil
PRN: 2122000814
Roll No: B53

Objective

To understand and implement Oracle Sequences, Triggers, Procedures, and Cursors in a relational database system.

Part 1: Oracle Sequences

Schema for Table `customer`

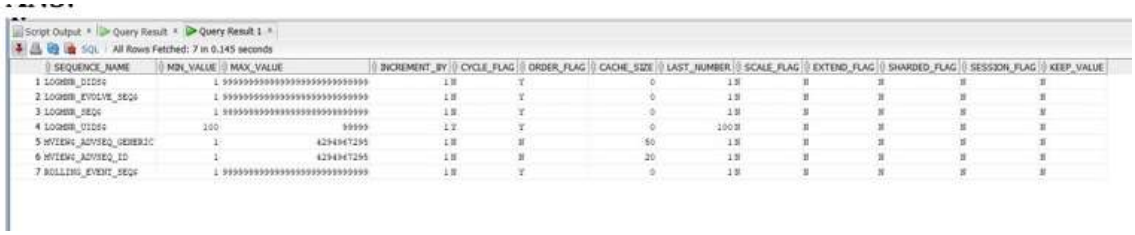
```
CREATE TABLE customer (
    cus_code INTEGER PRIMARY KEY,
    cus_lname VARCHAR2(10),
    cus_fname VARCHAR2(10),
    cus_initial VARCHAR2(1),
    cus_areacode INTEGER,
    cus_phone INTEGER,
    cus_balance NUMBER(10, 2)
);
```

i) Create a Sequence on `cus_code`

```
CREATE SEQUENCE CUS_SEQUENCES
START WITH 500
NOCACHE;
```

ii) Display User Sequences

```
SELECT * FROM USER_SEQUENCES;
```

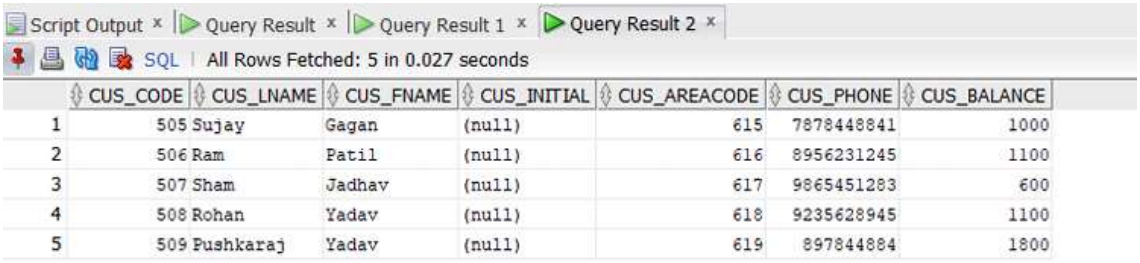


iii) Insert Values into `customer` Using the Created Sequence

```
INSERT INTO customer

VALUES (CUS_SEQUENCES.NEXTVAL, 'Sujay', 'Gagan', NULL, 615,
7878448841, 1000.00);

INSERT INTO customer

VALUES (CUS_SEQUENCES.NEXTVAL, 'Ram', 'Patil', NULL, 616,
8956231245, 1100.00);

INSERT INTO customer

VALUES (CUS_SEQUENCES.NEXTVAL, 'Sham', 'Jadhav', NULL, 617,
9865451283, 600.00);

INSERT INTO customer

VALUES (CUS_SEQUENCES.NEXTVAL, 'Rohan', 'Yadav', NULL, 618,
9235628945, 1100.00);

INSERT INTO customer

VALUES (CUS_SEQUENCES.NEXTVAL, 'Pushkaraj', 'Yadav', NULL, 619,
897844884, 1800.00);
```

iv) Display Customer Records

```
SELECT * FROM customer;
```



Script Output × | Query Result × | Query Result 1 × | Query Result 2 ×

SQL | All Rows Fetched: 5 in 0.027 seconds

	CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_BALANCE
1	505	Sujay	Gagan	(null)	615	7878448841	1000
2	506	Ram	Patil	(null)	616	8956231245	1100
3	507	Sham	Jadhav	(null)	617	9865451283	600
4	508	Rohan	Yadav	(null)	618	9235628945	1100
5	509	Pushkaraj	Yadav	(null)	619	897844884	1800

Part 2: Triggers

Schema for Table `student_report`

```
CREATE TABLE student_report (
    tid NUMBER(4) PRIMARY KEY,
    name VARCHAR2(30),
    subj1 NUMBER(2) CHECK (subj1 > 0 AND subj1 <= 20),
    subj2 NUMBER(2) CHECK (subj2 > 0 AND subj2 <= 20),
    subj3 NUMBER(2) CHECK (subj3 > 0 AND subj3 <= 20),
    total NUMBER(3) DEFAULT 0,
    per NUMBER(3) DEFAULT 0
);
```

Creating a Trigger for Total and Percentage Calculation

```
CREATE OR REPLACE TRIGGER calc_total_perc
BEFORE INSERT OR UPDATE ON student_report
FOR EACH ROW
BEGIN
```

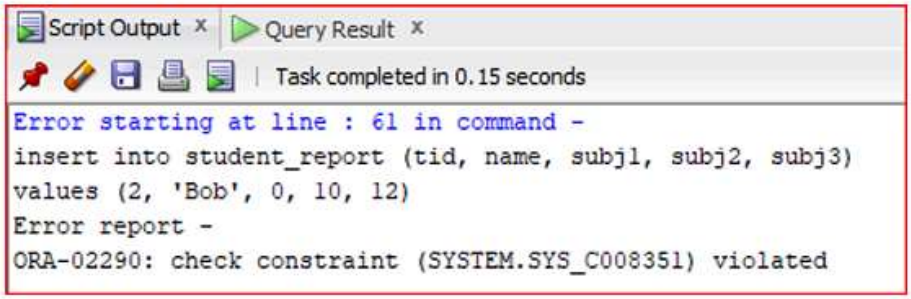
```
    :NEW.total := NVL(:NEW.subj1, 0) + NVL(:NEW.subj2, 0) +
NVL(:NEW.subj3, 0);

    :NEW.per := (:NEW.total * 100) / 60;

END;
```

Inserting Data into `student_report`

```
INSERT INTO student_report (tid, name, subj1, subj2, subj3, total,
per)

VALUES (1, 'Alice', 18, 15, 17, 0, 0);

INSERT INTO student_report (tid, name, subj1, subj2, subj3, total,
per)

VALUES (2, 'Bob', 18, 15, 17, 0, 0);
```

Verify Trigger Execution

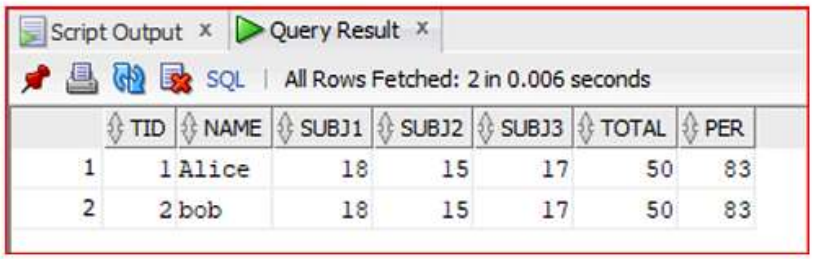
```
SELECT * FROM student_report;
```



--Checking triggered or not

select * from

student_report;



	TID	NAME	SUBJ1	SUBJ2	SUBJ3	TOTAL	PER
1	1	Alice	18	15	17	50	83
2	2	bob	18	15	17	50	83

Part 3: Procedure and Cursor

Schema for Table `Course`

```
CREATE TABLE Course (
    course_num INTEGER PRIMARY KEY,
    course_name VARCHAR2(20),
    dept_name VARCHAR2(15),
    credits INTEGER
);
```

Insert Data into `Course`

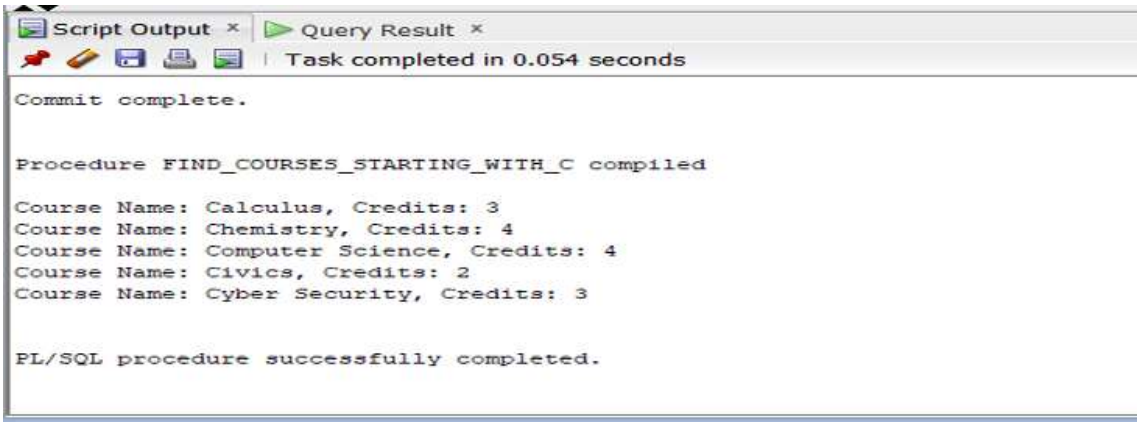
```
INSERT INTO Course (course_num, course_name, dept_name, credits)
VALUES (101, 'Calculus', 'MATH', 3);

INSERT INTO Course (course_num, course_name, dept_name, credits)
VALUES (102, 'Chemistry', 'SCIENCE', 4);

INSERT INTO Course (course_num, course_name, dept_name, credits)
VALUES (103, 'Computer Science', 'CSE', 4);

INSERT INTO Course (course_num, course_name, dept_name, credits)
VALUES (104, 'Biology', 'SCIENCE', 3);

INSERT INTO Course (course_num, course_name, dept_name, credits)
VALUES (105, 'Civics', 'ARTS', 2);

INSERT INTO Course (course_num, course_name, dept_name, credits)
VALUES (106, 'Physics', 'SCIENCE', 4);

INSERT INTO Course (course_num, course_name, dept_name, credits)
VALUES (107, 'Cyber Security', 'CSE', 3);
```

Procedure 1: Find Course Names Starting with ‘C’

```
CREATE OR REPLACE PROCEDURE find_courses_start_with_C IS
    CURSOR c_courses IS
        SELECT course_name, credits
        FROM Course
        WHERE course_name LIKE 'C%';
    v_course_name Course.course_name%TYPE;
    v_credits Course.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;
```

Execution:

```
SET SERVEROUTPUT ON;
BEGIN
    find_courses_start_with_C;
END;
```

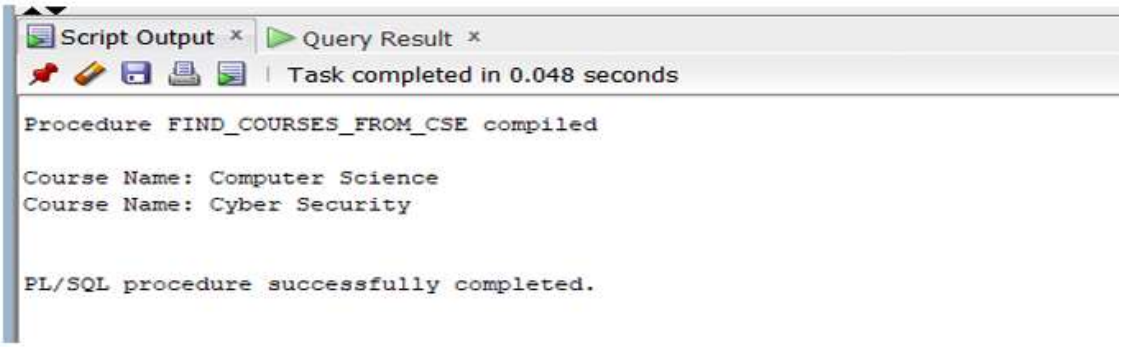


Procedure 2: Find Courses from CSE Department

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

Execution:

```
SET SERVEROUTPUT ON;

BEGIN

    find_courses_from_CSE;

END;
```



Conclusion

This experiment demonstrated the creation and usage of Oracle sequences, triggers, and procedures. It also illustrated the application of cursors for data retrieval and manipulation in a database.
