Name : Samiksha Bhashte

Roll No: C05

Experiment No. 5 : Advanced SQL Part 1

----Oracle Sequences:

```
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)
);
--q1 Create a sequence on cus_code starting from 1000
CREATE SEQUENCE cus_code_seq
START WITH 1000
INCREMENT BY 1;

--q2 Display user sequences
SELECT * FROM USER_SEQUENCES;

--q3 Insert values into customer using the created sequence
INSERT INTO customer_new (cus_code, cus_lname, cus_fname, cus_initial,
cus_areacode, cus_phone, cus_balance)
VALUES (cus_code_seq.NEXTVAL, 'Smith', 'John', 'J', 123, 4567890, 100.00);
```

--q4 Display customer records

SELECT * FROM customer_new;

Script Output × Query Result ×

SQL | All Rows Fetched: 1 in 0.008 seconds

	CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_BALANCE
1	1000	Smith	John	J	123	4567890	100

--q5 drop sequence created

DROP SEQUENCE cus_code_seq;

--QUESTION 2

```
CREATE TABLE employee (
  emp_ID INTEGER,
  first_name VARCHAR2(20),
  last_name VARCHAR2(20)
);
```

--q1

```
CREATE SEQUENCE emp_seq
START WITH 100
INCREMENT BY 1
MAXVALUE 9999
CACHE 10
NOCYCLE;
```

--q2

```
SELECT sequence_name
FROM user_sequences;
```

INSERT INTO employee (emp_ID, first_name, last_name)

VALUES (emp_seq.NEXTVAL, 'John', 'Dev');

	SEQUENCE_NAME
1	DEC_SEQ
2	EMP_SEQ
3	LOGMNR_DIDS$
4	LOGMNR_EVOLVE_SEQ$
5	LOGMNR_SEQ$
6	LOGMNR_UIDS$
7	MVIEW$_ADVSEQ_GENERIC
8	MVIEW$_ADVSEQ_ID
9	ROLLING_EVENT_SEQ$

--q3

ALTER SEQUENCE emp_seq

INCREMENT BY 5;

--verify

INSERT INTO employee (emp_ID, first_name, last_name)

VALUES (emp_seq.NEXTVAL, 'Sakshi', 'Mulik');

INSERT INTO employee (emp_ID, first_name, last_name)

VALUES (emp_seq.NEXTVAL, 'Sakshi', 'Mulik');

INSERT INTO employee (emp_ID, first_name, last_name)

VALUES (emp_seq.NEXTVAL, 'Pooja', 'Khot');

--dec_seq

CREATE SEQUENCE dec_seq

START WITH 100

INCREMENT BY -5

MINVALUE 0

MAXVALUE 100

CYCLE;

```
INSERT INTO employee (emp_ID, first_name, last_name)
VALUES (dec_seq.NEXTVAL, 'Apeksha', 'Nikam');

INSERT INTO employee (emp_ID, first_name, last_name)
VALUES (dec_seq.NEXTVAL, 'Rasika', 'Sawant');
```

---TRIGER

```
CREATE TABLE Student_Report (
  tid NUMBER(4) PRIMARY KEY,
  name VARCHAR2(30),
  subj1 NUMBER(2),
  subj2 NUMBER(2),
  subj3 NUMBER(2),
  total NUMBER(3) DEFAULT 0,
  percentage NUMBER(3) DEFAULT 0
);

CREATE TRIGGER calculate_total_percentage
BEFORE INSERT ON Student_Report
FOR EACH ROW
BEGIN
  :NEW.total := :NEW.subj1 + :NEW.subj2 + :NEW.subj3;
  :NEW.percentage := (:NEW.total / 60) * 100;
END;
```

```
----QUESTION 2
CREATE TABLE Instructor (
  ID INT PRIMARY KEY,
  name VARCHAR(20),
  age INT,
  salary INT
);



--
INSERT ALL
  INTO Instructor (ID, name, age, salary) VALUES (1, 'John Smith', 35, 50000)
  INTO Instructor (ID, name, age, salary) VALUES (2, 'Jane Doe', 30, 60000)
  INTO Instructor (ID, name, age, salary) VALUES (3, 'Bob Johnson', 40, 70000)
SELECT 1 FROM DUAL;



CREATE OR REPLACE TRIGGER salary_diff_trigger
BEFORE INSERT OR UPDATE OR DELETE ON Instructor
FOR EACH ROW
BEGIN
  IF INSERTING THEN
    DBMS_OUTPUT.PUT_LINE('New salary: ' || :NEW.salary);
  ELSIF UPDATING THEN
    DBMS_OUTPUT.PUT_LINE('Old salary: ' || :OLD.salary || ', New salary: ' ||
:NEW.salary);
    DBMS_OUTPUT.PUT_LINE('Salary difference: ' || (:NEW.salary - :OLD.salary));
  ELSIF DELETING THEN
    DBMS_OUTPUT.PUT_LINE('Deleted salary: ' || :OLD.salary);
  END IF;
END;
```

```
-- Update salary of instructor with ID = 1
UPDATE Instructor
SET salary = 55000
WHERE ID = 1;
```

```
-- Check the result
SELECT * FROM Instructor WHERE ID = 1;
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 1 in 0.003 seconds

	ID	NAME	AGE	SALARY
1	1	John Smith	35	55000

```
-- Delete instructor with ID = 2
DELETE FROM Instructor
WHERE ID = 2;

-- Check the result
SELECT * FROM Instructor WHERE ID = 2;
```