Experiment No. 06

Advanced SQL

Name: Samiksha R. Devardekar

Div.: B

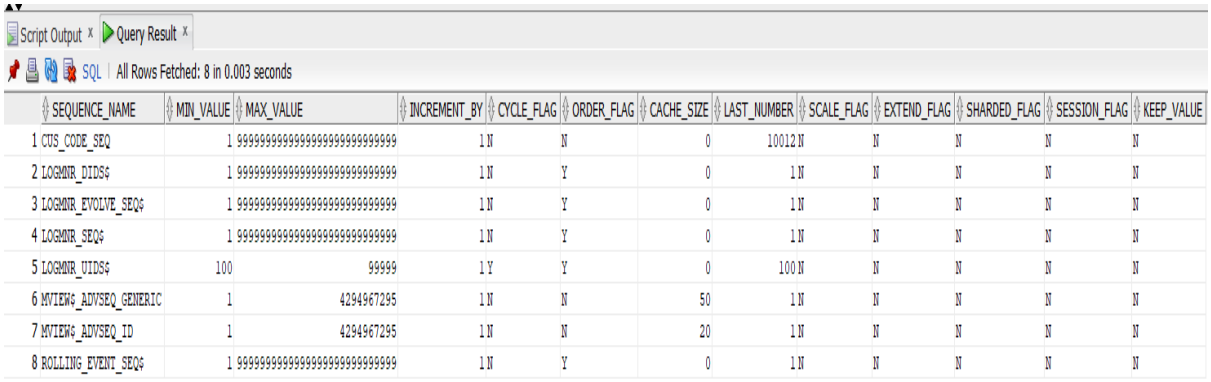
Roll No.: 05

PRN: 2122000098

Problem Statement

Oracle Sequences:

```
create table Customer(
    cus_code integer,
    cus_fname varchar(10),
    cus_lname varchar(10),
    cus_initial varchar(1),
    cus_areacode integer,
    cus_phone integer,
    cus_balance number(10,2),
    primary key(cus_code)
);

-->1. Create sequence on cus_code.
create sequence cus_code_seq start with 10007 nocache;

-->2. Display user sequences.
select * from user_sequences;
```



Script Output | Query Result

SQL | All Rows Fetched: 8 in 0.003 seconds

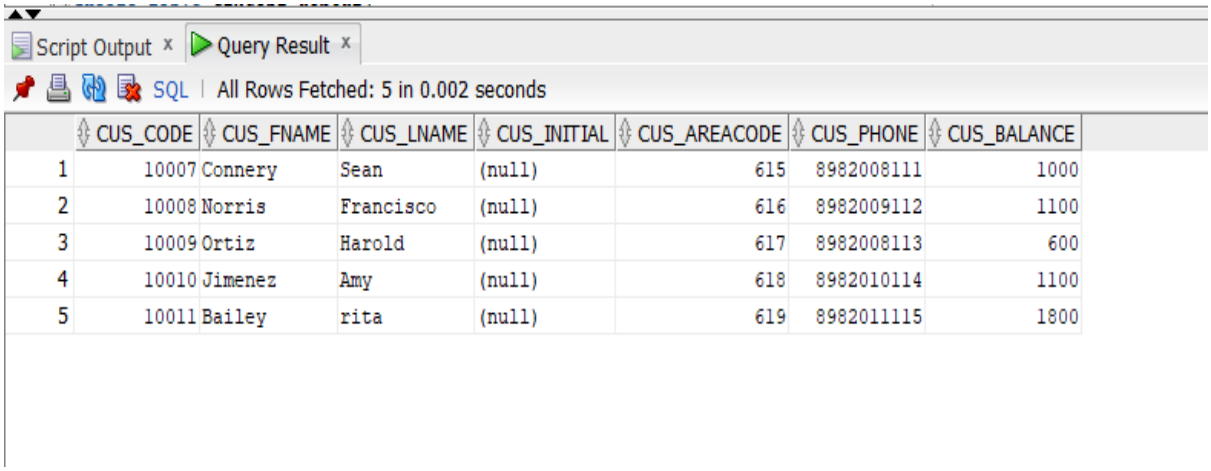
	SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	CYCLE_FLAG	ORDER_FLAG	CACHE_SIZE	LAST_NUMBER	SCALE_FLAG	EXTEND_FLAG	SHARDED_FLAG	SESSION_FLAG	KEEP_VALUE
1	CUS_CODE_SEQ	1	9999999999999999999999999999	1	N	N	0	10012	N	N	N	N	N
2	LOGMNR_DIDS$	1	9999999999999999999999999999	1	N	Y	0	1	N	N	N	N	N
3	LOGMNR_EVOLVE_SEQ$	1	9999999999999999999999999999	1	N	Y	0	1	N	N	N	N	N
4	LOGMNR_SEQ$	1	9999999999999999999999999999	1	N	Y	0	1	N	N	N	N	N
5	LOGMNR_UIDS$	100	99999	1	Y	Y	0	100	N	N	N	N	N
6	MVIEW$_ADVSEQ_GENERIC	1	4294967295	1	N	N	50	1	N	N	N	N	N
7	MVIEW$_ADVSEQ_ID	1	4294967295	1	N	N	20	1	N	N	N	N	N
8	ROLLING_EVENT_SEQ$	1	9999999999999999999999999999	1	N	Y	0	1	N	N	N	N	N

-->3. Insert values into customer using created sequence.

```
insert into Customer
values(cus_code_seq.nextval,'Connery','Sean',NULL,'615','8982008111',1000.0
0);

insert into Customer
values(cus_code_seq.nextval,'Norris','Francisco',NULL,'616','8982009112',1100
.00);

insert into Customer
values(cus_code_seq.nextval,'Ortiz','Harold',NULL,'617','8982008113',600.00);

insert into Customer
values(cus_code_seq.nextval,'Jimenez','Amy',NULL,'618','8982010114',1100.00
);

insert into Customer
values(cus_code_seq.nextval,'Bailey','rita',NULL,'619','8982011115',1800.00);
```

-->4. Display customer records.

```
select * from customer;
```



	CUS_CODE	CUS_FNAME	CUS_LNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_BALANCE
1	10007	Connery	Sean	(null)	615	8982008111	1000
2	10008	Norris	Francisco	(null)	616	8982009112	1100
3	10009	Ortiz	Harold	(null)	617	8982008113	600
4	10010	Jimenez	Amy	(null)	618	8982010114	1100
5	10011	Bailey	rita	(null)	619	8982011115	1800

Trigger:

```
create table student_report(
    tid number(4),
    name varchar(30),
    subj1 number(2),
    subj2 number(2),
    subj3 number(2),
    total number(3),
    per number(3),
    primary key(tid),
    check(subj1>=0 and subj1<=20),
    check(subj2>=0 and subj2<=20),
    check(subj3>=0 and subj3<=20)
);

create or replace procedure student_report_check_procedure
as
begin
    update student_report set total=subj1+subj2+subj3;
```

```
    update student_report set per=((subj1+subj2+subj3)/60)*100;
end;
```

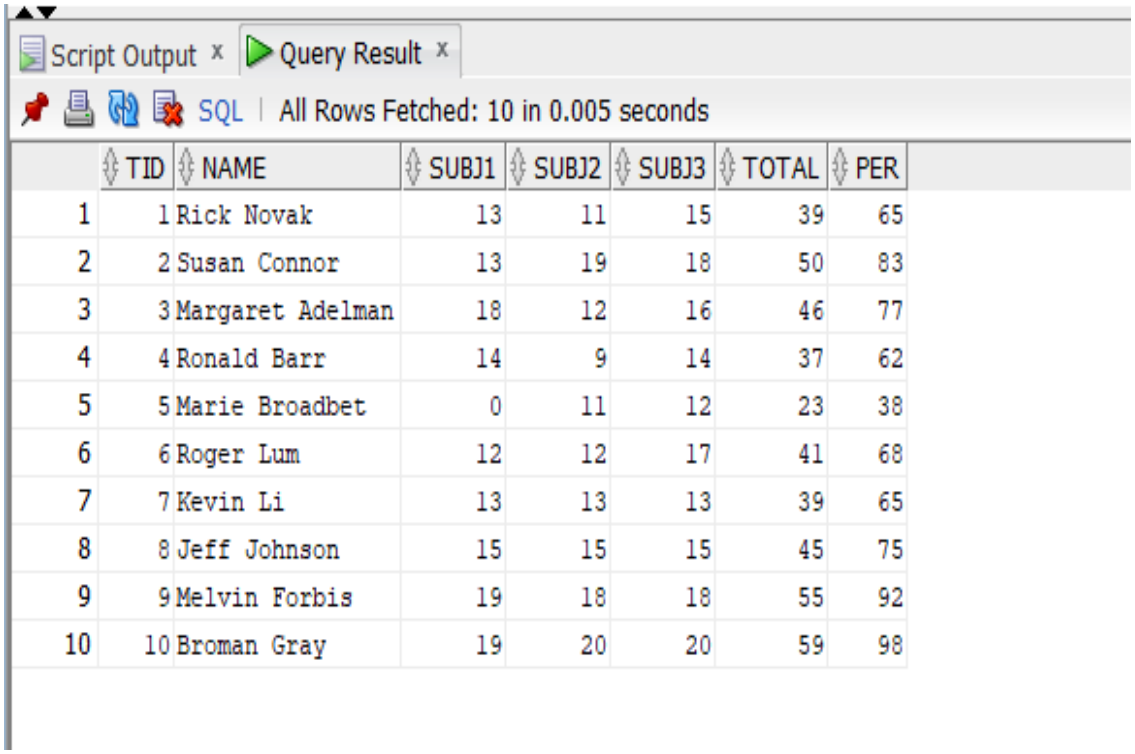
```
Procedure STUDENT_REPORT_CHECK_PROCEDURE compiled
```

```
create or replace trigger trg_check_report
before insert or update on student_report
for each row
declare
begin
    :new.total := :new.subj1 + :new.subj2 + :new.subj3;
    :new.per := ((:new.subj1 + :new.subj2 + :new.subj3)/60)*100;
end;
```

```
Trigger TRG_CHECK_REPORT compiled
```

```
insert into student_report values(1,'Rick Novak',13,11,15,0,0);
insert into student_report values(2,'Susan Connor',13,19,18,0,0);
insert into student_report values(3,'Margaret Adelman',18,12,16,0,0);
insert into student_report values(4,'Ronald Barr',14,9,14,0,0);
insert into student_report values(5,'Marie Broadbet',0,11,12,0,0);
insert into student_report values(6,'Roger Lum',12,12,17,0,0);
insert into student_report values(7,'Kevin Li',13,13,13,0,0);
insert into student_report values(8,'Jeff Johnson',15,15,15,0,0);
insert into student_report values(9,'Melvin Forbis',19,18,18,0,0);
insert into student_report values(10,'Broman Gray',19,20,20,0,0);
```

select * from student_report;



Script Output | Query Result

SQL | All Rows Fetched: 10 in 0.005 seconds

	TID	NAME	SUBJ1	SUBJ2	SUBJ3	TOTAL	PER
1	1	Rick Novak	13	11	15	39	65
2	2	Susan Connor	13	19	18	50	83
3	3	Margaret Adelman	18	12	16	46	77
4	4	Ronald Barr	14	9	14	37	62
5	5	Marie Broadbet	0	11	12	23	38
6	6	Roger Lum	12	12	17	41	68
7	7	Kevin Li	13	13	13	39	65
8	8	Jeff Johnson	15	15	15	45	75
9	9	Melvin Forbis	19	18	18	55	92
10	10	Broman Gray	19	20	20	59	98

exec student_report_check_procedure;

```
PL/SQL procedure successfully completed.
```

Procedure and Cursor:

```
create table course(
    course_num integer,
    course_name varchar(50),
    dept_name varchar(15),
    credits integer,
    primary key(course_num)
);
insert into course values(1001,'Math 1','BSH',3);
```

```
insert into course values(1002,'Math 2','BSH',3);
insert into course values(1061,'Compiler Construction Theory','CSE',3);
insert into course values(1071,'Advanced Database System Theory','CSE',3);
insert into course values(1072,'Distributed System Theory','CSE',3);
insert into course values(1073,'Unix Operating System Theory','CSE',3);
insert into course values(1161,'Compiler Construction Theory','CSE',3);
insert into course values(1171,'Advanced Database System Lab','CSE',1);
insert into course values(1172,'Distributed System Lab','CSE',1);
insert into course values(1173,'Unix Operating System Lab','CSE',1);
```

-->1. Write a procedure which includes cursors: Find course_name and credits where course_name starts with 'C'.

```
set serveroutput on;
create or replace procedure lab5_q2_proc1
is
c_name varchar(50);
c_credit integer;
cursor cur is
   select course_name, credits
   from course
   where course_name like 'C%';
begin
   dbms_output.put_line('Course Name                    Credit');

dbms_output.put_line('=========================================
=======');
   open cur;
   loop
```

```
        fetch cur into c_name,c_credit;
        exit when cur%notfound;
        dbms_output.put_line(' '||c_name||'                    '||c_credit);
    end loop;

dbms_output.put_line('=========================================
=======');
    dbms_output.put_line('Total Courses -> '||cur%rowcount);
    close cur;
end;
exec lab5_q2_proc1;
```

```
Course Name                                Credit
=============================================
 Compiler Construction Theory                    3
 Compiler Construction Theory                    3
=============================================
Total Courses -> 2


PL/SQL procedure successfully completed.
```

-->2. Write a procedure which includes cursors: Find course names from 'CSE' department.

```
create or replace procedure lab5_q2_proc2
is
c_name varchar(50);
cursor cur is
    select course_name
    from course
    where dept_name='CSE';
begin
```

```
    dbms_output.put_line('CSE Course Name');

dbms_output.put_line('=========================================
=======');
    open cur;
    loop
        fetch cur into c_name;
        exit when cur%notFound;
        dbms_output.put_line(' '||c_name);
    end loop;

dbms_output.put_line('=========================================
=======');
        dbms_output.put_line('Total Courses -> '||cur%rowcount);
        close cur;
end;

exec lab5_q2_proc2;
```

```
CSE Course Name
=============================================
 Compiler Construction Theory
 Advanced Database System Theory
 Distributed System Theory
 Unix Operating System Theory
 Compiler Construction Theory
 Advanced Database System Lab
 Distributed System Lab
 Unix Operating System Lab
=============================================
Total Courses -> 8


PL/SQL procedure successfully completed.
```