**Oracle Sequences:**

Consider table customer with primary key(cus\_code))

| Field Type | Data Type |
| --- | --- |
| cus\_code cus\_lname cus\_fname cus\_initial cus\_areacode  cus\_phone cus\_balance) | Integer varchar2(10) varchar2(10) varchar2(1) INTEGER INTEGER  number(10,2 |

# Create sequence on cus\_code

**ANS:**CREATE SEQUENCE CUS\_SEQUENCES START WITH 500 noCache;

drop sequence CUS\_SEQUENCES; select \* from USER\_SEQUENCES;

1. Display user sequences
2. Insert values into customer using created sequence

# ANS:

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

);

select \*from customer;

1. Display customer records

# ANS:s

**Trigger:**

Consider Student Report table, in which student marks assessment is recorded. In such schema, create a trigger so that the total and percentage of specified marks is automatically inserted whenever a record is inserting. Initial insert 0 for total and per attributes. Maximum marks should be 20 for each subject

Field | Type | Null | Key |

+ + + + +

| tid | int(4) | NO | PRI |

| name | varchar(30) | YES |

| subj1 | int(2) | YES |

| subj2 | int(2) | YES |

| subj3 | int(2) | YES |

| total | int(3) | YES |

| per | int(3) | YES |

## -- Creating the trigger to calculate total and percentage before insert

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

);

create or replace trigger calc\_total\_perc before insert or update on student\_reportfor each row

:new.total := nvl(:new.subj1, 0) + nvl(:new.subj2, 0) + nvl(:new.subj3, 0);

:new.per := (:new.total \* 100) / 60;end;

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); **ANS:**

--Checking constraint

insert into student\_report (tid, name, subj1, subj2, subj3)values (2, 'Bob', 0, 10, 12);

--Checking triggered or not select \* from student\_report;

# Procedure and Cursor:

Consider Course Table with course\_num as primary key.

| Field Type | Data Type |
| --- | --- |
| course\_num course\_name dept\_name  credits | Integer varchar2(20) varchar2(15)  Integer |

# 1.)Write a procedure which includes cursors: Find course\_name and credits where course name starts with ‘C’

create table Course( course\_num integer primary key, course\_name varchar2(20), dept\_name varchar2(15),

credits integer

)

drop table 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);

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

-- Opening and fetching cursor data 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;

SET SERVEROUTPUT ON;

BEGIN

find\_courses\_starting\_with\_C; END;

# 2.) Write a procedure which includes cursors: Find course names 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';

1

v\_course\_name Course.course\_name%TYPE; BEGIN

OPEN c\_courses\_cse;

LOOP

-- Fetch data from the cursor into variable

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;

SET SERVEROUTPUT ON;

BEGIN

find\_courses\_from\_CSE; END;

## ANS: