Experiment-1

Date: 30/9/23

1. Student should decide on a case study, analyse and then formulate the problem statement by populating object (entities) and their role.

ROADWAY TRAVELS

"Roadway Travels" is in business since 1977 with several buses connecting different places in India. Its main office is located in Hyderabad.

EXAMPLE

Entities

1. BUS
2. Ticket
3. Passenger

Relationships

1. Reservation
2. Cancellation

PRIMARY KEY ATTRIBUTES

1. Ticket ID (Ticket Entity)
2. Pnr_No (Passenger Entity)
3. Bus_No (Bus Entity)

EXAMPLE

Entities

1. BUS
2. Ticket
3. Passenger

The Following are the entities and its attributes

BUS

Name	Type
BUSNO	NUMBER(10)
SOURCE	CHAR(20)
DESTINATION	CHAR(20)

PASSENGER

Name	Type
PNR NO	NUMBER(10) (Primary Key)
NAME	CHAR(20)
AGE	NUMBER(10)
GENDER	CHAR(30)

RESERVATION

Name	Type
TICKET NO	NUMBER(20) (Primary Key)
PNR NO	NUMBER(10)
DATES	DATE
TIME	VARCHAR2(20)

CANCELLATION

Name	Type
PNR NO	NUMBER(10) (Primary Key)
SOURCE	CHAR(20)
DESTINATION	CHAR(20)
DATES	DATE

TICKET

Name	Type
TICKET NO	NUMBER(10) (Primary Key)
SOURCE	CHAR(20)
DESTINATION	CHAR(20)
DATES	DATE
TIME	VARCHAR2(20)

- When graduate students work on a project, a professor must supervise their work on the project. Graduate students can work on multiple projects, in which case they will have a (potentially different) supervisor for each one.
- Departments have a department number, a department name, and a main office.
- Departments have a professor (known as the chairman) who runs the department.
- Professors work in one or more departments, and for each department that they work in, a time percentage is associated with their job.
- Graduate students have one major department in which they are working on their degree.
- Each graduate student has another, more senior graduate student (known as a student advisor) who advises him or her on what courses to take.

Design and draw an ER diagram that captures the information about the university.Use only the basic ER model here; that is, entities, relationships, and attributes. Be sure to indicate any key and participation constraints.

Entities

1) Student – Id, Name, address, age, gender, ph.no, e-mail.

2) Faculty - Id, Name, dept, address, email, age, qualification

3) Dept - Id, name, location.

4) Courses - Course Id, course name, course dept

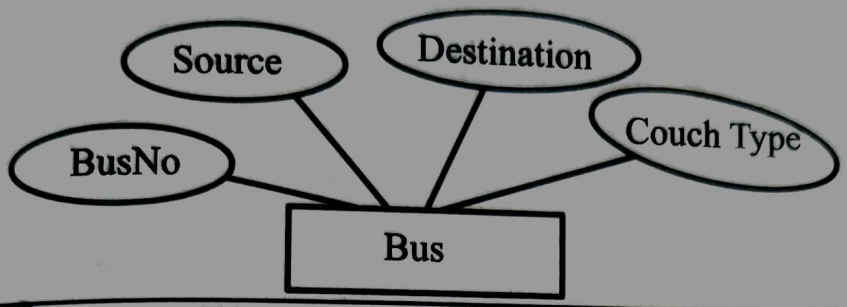
5) R&D - project Id, project name, Supervisor, budgit.

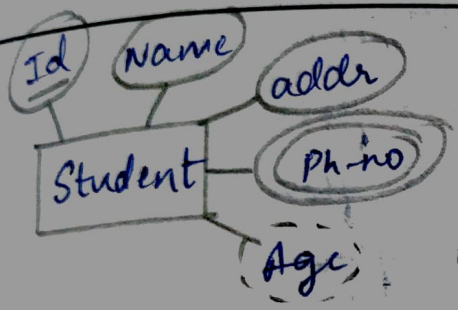

age
SSN
rank
speciality
Professor
work in
Manages
P-id
start date
sponsor
end-date
Project
budget
work-dept
Runs
Supervises
Work proj
SSN
Dept
Major
d-no
office
d-name
Graduate
senior
Advisor
grad
name
SSN
age
deg
proj

Relational model For Employees 1

Empid	Empname	Designation

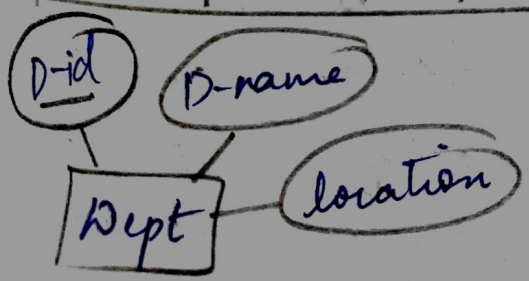
Create Relation model for Bus





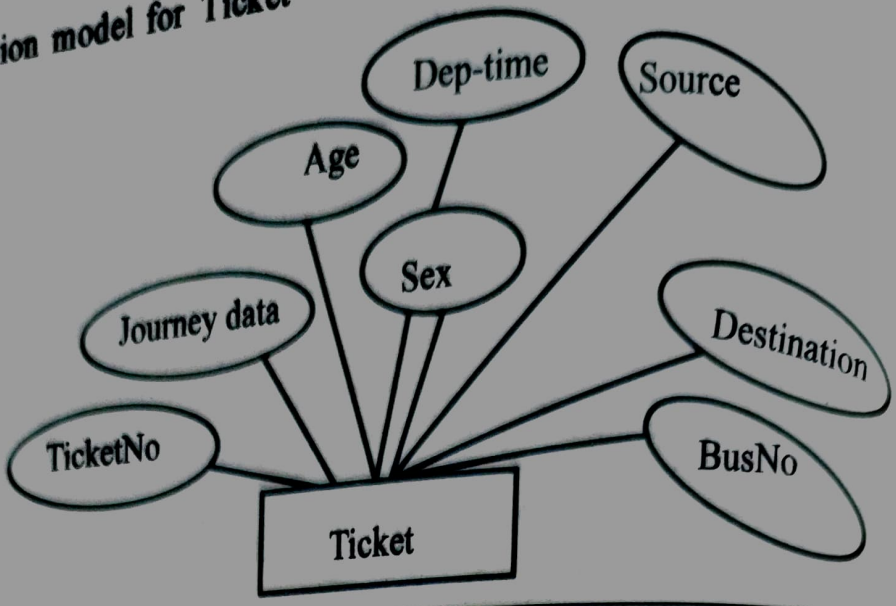

```
Create table Stdt (Id number(10) primary key
name varchar(30), addr varchar(30),
ph-no number(20), Age number(20));
insert into stdt values ('001', 'Roy', 'Hyd',
'9876543210', '38');
select * from stdt;
```

ID	NAME	ADDR	PHNO	AGE
1	Roy	Hyd	9876543210	38
2	Lily	London	0123456789	23
3	Jay	bombay	8765123412	18
4	Sam	Delhi	1234598765	20
5	Rose	Bengal	1235671079	19

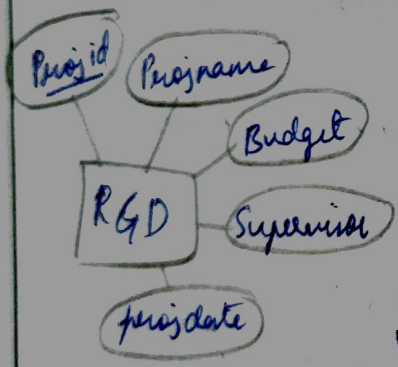



```
Create table department (id number(10)
primary key, name varchar(20),
location varchar(20));
insert into department values ('101',
'CSE', 'Second floor');
```

Create Relation model for Ticket



ID	NAME	LOCATION
100	CSE	second floor
200	CSM	second floor
300	CSD	first floor
400	CSC	third floor
500	AIML	third floor



Create table resdev(projid number(10) primary key, projname varchar(20), budget number(20), supervr varchar(20), projdate date);

insert into resdev values('1', 'ar/vr', '2000', 'vijay', '14-oct-2023');

PROJ ID	PROJ NAME	BUDGET	SUPERVR	PROJDATE
1	ar/vr	2000	vijay	14-OCT-23
2	laptop dev	1000	james	20-Oct-23
3	webdev	3000	betty	18-OCT-23
4	appdev	2500	taylor	19-Oct-23
5	research cor	1500	joe	21-Oct-23

4. Different tables should not have same name.
5. We should specify a unique column name.
6. We should specify proper data type along with width.
7. We can include "not null" condition when needed. By default it is 'null'.

1. Create a table called EMP with the following structure.

Name	Type
EMPNO	NUMBER(6)
ENAME	VARCHAR2(20)
JOB	VARCHAR2(10)
DEPTNO	NUMBER(3)
SAL	NUMBER(7,2)

Allow NULL for all columns except e name and job.

```
Create table EMP (empno number(6), ename varchar(20),
    job varchar2(10), deptno number(3), sal number
                                        (7,2));

desc EMP;
```

2. Create a table called DEPT with the following structure.

Name	Type
DEPTNO	NUMBER(2)
DNAME	VARCHAR2(10)
LOC	VARCHAR2(10)

Creat DEPTNO as the primary key

```
Create table DEPT (deptno number(2), dname varchar(10),
                                    loc varchar(10));
```

ALTER TABLE

Alter command is used to:

1. Add a new column.
2. Modify the existing column definition.
3. To include or drop integrity constraint.

Syntax: alter table tablename add/modify (attribute datatype(size));

3. Add a column experience to the emp table.

```
alter table EMP add (experience varchar(20));
desc EMP;
```

Name	Type
EMPNO	NUMBER(6)
ENAME	VARCHAR2(20)
JOB	VARCHAR2(20)
DEPTNO	NUMBER(3)
SAL	NUMBER(7,2)
EXPERIENCE	VARCHAR2(20)

4. Modify the column width of the job field of emp table.

```
alter table EMP modify (job varchar(15));
```

Name	Type
EMPNO	NUMBER(6)
ENAME	VARCHAR2(20)
JOB	VARCHAR2(15)
DEPTNO	NUMBER(3)
SAL	NUMBER(7,2)
EXPERIENCE	VARCHAR2(20)

5. Create the emp1 table with ename and empno, add constraints to check the empno value while entering (i.e) empno > 100.

```
create table emp1 (ename varchar(15), empno
number(5), check(empno > 100));
insert into emp1 values ('rani', '50');
ERROR: check constraint (CHAITANYA.SYS_C0007275) violated.
insert into emp1 values ('rani', '101');
select * from emp1;
```

ENAME	EMPNO
ravi	101

DROP TABLE

It will delete the table structure provided the table should be empty.

Syntax: drop table tablename;

6. Drop any column in the emp table.

```
alter table emp1 drop(empno);
desc emp1;
```

Name	Type
ENAME	VARCHAR2(15)

TRUNCATE TABLE

If there is no further use of records stored in a table and the structure has to be retained the records alone can be deleted.

Syntax: TRUNCATE TABLE <TABLE NAME>;

7. Truncate the emp table

```
truncate table emp1;
→ Table Truncated
select * from emp1;
→ no rows selected
desc emp1;
```

Name	Type
ENAME	VARCHAR2(15)

DESC

This is used to view the structure of the table.

Syntax: desc tablename;

Example

desc emp;

Name	Type
EMP NO	NUMBER(6)
ENAME	VARCHAR2(20)
JOB	VARCHAR2(15)
DEPTNO	NUMBER(3)
SAL	NUMBER(7,2)
EXPERIENCE	VARCHAR2(20)

Skipping the fields while inserting:

Syntax:Insert into <tablename(coln names to which datas to b inserted)> values (list of values).

1. Insert a single record into dept table.

```
insert into emp values ('101','ravi','5','10','10000');

empid   empname   experience   deptno   sal     role
101     ravi      5            30       10000   clerk
```

2. Insert more than a record into emp table using a single insert command.

```
insert into emp values (&empid, &empname, &experience,
&deptno, &sal, &role);
Enter value for empid : '102'
Enter value for empname : 'taylor'
Enter value for experience : '7'
Enter value for deptno : '13'
Enter value for sal : '50000'
Enter value for role : 'manager'
old 1 : insert into emp values (&empid, &empname, &experience,
                                &deptno, &sal, &role);
new 1 : insert into emp values ('102', 'taylor', '7', '13', '50000',
'manager');
```

Enter

Select command

It is used to retrieve information from the table. it is generally referred to as querying the table.we can either display all columns in the table or only specify the column in the table.

Selects all rows from the table

Syntax: Select * from tablename;

The retrieval of specific columns from a table

It retrieves the specified columns from the table

Syntax: Select column_name1, …..,column_namen from table name;

Elimination of duplicates from the select clause:

It prevents retriving the duplicated values .Distinct keyword is to be used.

Syntax: Select DISTINCT col1, col2 from table name;

Select command with where clause:

To select specific rows from a table we include 'where' clause in the select command. It can appear only after the 'from' clause.

Syntax: Select column_name1, …..,column _name from table name where condition;

Select command with order by clause:

Syntax: Select column_name1, …..,column _name
from table name where condition order by colmn_name;

Select command to create a table:

Syntax: create table table_name as select * from existing_tablename;

Select command to insert records:

Syntax: insert into table name (select columns from existing_tablename);

1. Select employee name, job from the emp table

```
> select empname, role from emp;

empname        role
rani           clerk
taylor         manager
joe            assistant
aks            manager
olivia         clerk
josh           clerk
```

2. Create a pseudo table employee with the same structure as the table emp and insert rows into the table using select clauses.

```
>Create table employee as select * from emp;
 Table created
> select * from employee;

empid   empname   experience   deptno   sal     role
101     rani      5            30       10000   clerk
102     taylor    7            13       50000   manager
103     joe       6            22       30000   assistant
104     aks       5            25       45000   manager
105     olivia    3            15       13000   clerk
106     josh      4            30       12000   clerk
```

3. Display only those employees whose deptno is 30.

```
select * from emp where deptno = '30';
```

empid	empname	experience	deptno	sal	role
101	rani	5	30	10000	clerk
106	josh	4	30	12000	clerk

4. Display deptno from the table employee avoiding the duplicated values.

```
select distinct deptno from emp;
```

deptno
30
13
22
25
18

5. List the records in the emp table order by salary in descending order.

```
select * from emp order by sal desc;

empid   empname   experience   deptno   sal     designation
102     taylor    7            13       50000   manager
104     aks       5            25       45000   manager
103     joe       6            22       30000   analyst
105     olivia    3            18       13000   clerk
106     josh      4            30       12000   clerk
101     rani      5            30       10000   clerk
```

6. List the records in the emp table order by salary in ascending order.

```
select * from emp order by sal;

empid   empname   experience   deptno   sal     designation
101     rani      5            30       10000   clerk
106     josh      4            30       12000   clerk
105     olivia    3            18       13000   clerk
103     joe       6            22       30000   analyst
104     aks       5            25       45000   manager
102     taylor    7            13       50000   manager
```

UPDATE COMMAND

It is used to alter the column values in a table. A single column may be updated or more than one columns can be updated.

Syntax: update table_name set field=values where condition;

UPDATE table_name SET column_name1= value1, column_name2= value2;

9. Update the emp table to set the salary of all employees to Rs15000/- who are working as manager.

```
update emp set sal = '15000' where role = 'manager'
select * from emp;
```

empid	empname	empmgrno	deptno	sal	role
102	taylor	7	13	15000	manager
104	aks	5	25	15000	manager

DELETE COMMAND

After inserting a row in the table we can also delete row in the table if required. The delete command consists of a from clause followed by an optional where clause.

Syntax: Delete from table where conditions;

10. Delete only those who are working as lecturer

```
delete from emp role = 'clerk';
select * from emp;
```

empid	empname	empmgrno	deptno	sal	role
102	taylor	7	13	15000	manager
103	joe	6	22	10000	assistant
104	aks	5	23	15000	manager

1) Find sailors whose rating is better than some sailor called Horatio. (Any)

```
> Select s.sid from sailors s where s.rating > any
  (select s2.rating from sailors s2 where s2.name = 'horatio');

 SID
 ---
  58
  71
  31
  32
```

2) Find the sailors with the highest rating. (All)

```
> Select s.sid from sailor s where s.rating >= all
  (select s2.rating from sailors s2);

  SID
 ---
  71
  58
```

3) Find the names of sailors who have reserved both a red and a green boat. (In)

```
> select s.sname from sailors s, reserves r, boats b
where s.sid = name and r.bid = b.bid and b.color = 'red'
and s.sname in (select s2.sname from sailors s2,
boats b2, reserves r2 where s2.sid = r2.sid and
r2.bid = b2.bid and b2.color = 'green');

 SNAME
 dustin
 dustin
```

4) Find the names of sailors who have reserved boat 103.(Exists)

```
select s.sname from sailors s where exists
(select * from reserves r where r.bid = 103
and r.bid = s.sid);

SNAME
----
dustin
lubber
```

5) Find the names of sailors who have reserved all boats. (Not exists)

```
select s.sname from sailors s where not exists
(select b.bid from boats b where not exists (select
r.bid from reserves r where r.bid = b.bid and
r.sid = s.sid));

SNAME
----
dustin
```

6) Find the names of sailors who have reserved a red and a green boat. (Union)

```
Select s.sname from sailors s, reserves r, boats b
where s.sid = r.sid and r.bid = b.bid and b.color =
'red'. Union select s2.sname from sailors s2,
reserves r2, boats b2 where s2.sid = r2.sid and
r2.bid = b2.bid and b2.color = 'green';

SNAME
----
dustin
horatio
```

7) Find the names of sailors who have reserved both a red and a green boat. (Intersect)

```
>select s.sname from sailors s, reserves r, boats
b where s.sid = r.sid and r.bid = b.bid and
b.color = red intersect select s2.sname
from sailors s2, reserves r2, boats b2
where s2.sid = r2.sid and r2.bid = b2.bid
and b2.color = 'green'
```

SNAME
dustin
lubber

8) Find the names of sailors who have reserved a red and a green boat. (Union All)

```
>Select s.sname from sailors s, reserves r, boats b
where s.sid = r.sid and r.bid and b.color =
'red' union all select s2.name from sailors
s2, boats b2, reserves r2, where s2.sid and
r2.bid = b2.bid and b2.color = 'green';
```

SNAME
dustin
dustin
lubber
lubber
horatio
dustin
lubber

Create the above tables using DDL and DML commands and perform the following Queries

1) Find the names of sailors who have reserved red boat (Nested query)

```
> select s.sname from sailors s where s.sid
in (select r.sid from reserves r where
r.bid in (select b.bid from boats b
where b.color = 'red'));
```

OUTPUT

```
SNAME
-----
dustin
lubber
horatio
```

2) Find the names of sailors who have reserved boat number 103(Correlated Nest
query)

```
> select s.sname from sailors s where exists
(select * from reserves r where r.bid = 103
and r.sid = s.sid);
```

OUTPUT

```
SNAME
-----
dustin
lubber
```

SQL JOIN

The SQL Joins clause is used to combine records from two or more tables in a database. A JOIN is a means for combining fields from two tables by using values common to each.

Customer table

Id	Name	Age	Address	Salary
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	Kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

Order table

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08	3	3000
100	2009-10-08	3	1500
101	2009-11-20	2	1560
103	2008-05-20	4	2060

There are different types of joins available in SQL "

- INNER JOIN " returns rows when there is a match in both tables.
- LEFT JOIN " returns all rows from the left table, even if there are no matches in the right table.
- RIGHT JOIN " returns all rows from the right table, even if there are no matches in the left table.
- FULL JOIN " returns rows when there is a match in one of the tables.
- SELF JOIN " is used to join a table to itself as if the table were two tables, temporarily renaming at least one table in the SQL statement.
- CARTESIAN JOIN " returns the Cartesian product of the sets of records from the two or more joined tables.

INNER JOIN

```
SELECT columns FROM table1  INNER JOIN table2 ON table1.column = table2.column;
```

Example: Write a query to perform **INNER JOIN** between any two tables

```
> select id, name, amount, day from customer
inner join orders on customer.id = orders.
customer_id;
```

OUTPUT

ID	NAME	AMOUNT	DAY
2	Khilan	1560	20-NOV-09
3	Kaushik	1500	08-OCT-09
3	Kaushik	3000	08-OCT-09
4	Chaitali	2060	20-MAY-08

LEFT OUTER JOIN

SELECT columns FROM table1 LEFT [OUTER] JOIN table 2 ON table1.column = table2.column;

Example: Write a query to perform **LEFT OUTER** join between any two tables

```
> select id, name, amount, day from
customer left outer join orders on customer.id
= orders.customer_id;
```

OUTPUT

ID	NAME	AMOUNT	DAY
3	Kaushik	3000	08-Oct-09
3	Kaushik	1500	08-Oct-09
2	Khilan	1560	20-Nov-09
4	Chaitali	2060	20-May-08
5	haldik		
1	ramesh		
6	Komal		
7	muffy		

RIGHT OUTER JOIN

SELECT columns FROM table1 RIGHT [OUTER] JOIN table2 ON table1.column = table2.column;

Example: Write a query to perform RIGHT OUTER join between any two tables

```
> select id, name, amount, day from customer
right outer join orders on customer.id =
orders.customer_id;
```

ID	NAME	AMOUNT	DAY
2	Khilan	1560	20-Nov-09
3	Kaushik	1500	08-Oct-09
3	Kaushik	3000	08-Oct-09
4	Chaitali	2060	20-May-08

OUTPUT

ID	NAME	AMOUNT	DAY
2	Khilan	1560	20-nov-09
3	Kaushik	1000	08-oct-09
3	Kaushik	3000	08-oct-09
4	Chaitali	2060	20-may-08

EQUI JOIN

```
SELECT column_list FROM table1, table2.... WHERE table1.column_name =
table2.column_name;
```

Example: Write a query to perform **EQUI JOIN** between any two tables

1) Find the average age of sailors with a rating of 10?

```
SQL> select avg(s.age) from sailors s
     where s.rating = 10;
```

Output:

Avg (S.AGE)
25.5

2) Find the name and age of the oldest sailor?

```
SQL> select s.sname, s.age from sailors s
     where s.age = (select max(s.age)
     from sailors s);
```

Output:

SNAME	AGE
bob	63

3) Write the query to Count the number of different sailor names?

```
SQL> select count (distinct s.sname)
from sailors 's;
```

Output:

```
Count (distincts.name)
----------------------
                    9
```

4) Find the age of the youngest sailor for each rating level?

```
SQL> select s.rating, min (s.age) from
sailors s. group by s.rating;
```

Output:

RATING	MIN(S.AGE)
1	33
8	25
7	35
3	25
10	16

) **Find the sum of ages of sailors whose rating is above 9?**

```
SQL> select sum(age) from sailors where
rating >10;
```

Output:

```
SUM(AGE)
--------
```

6) Find the average age of sailors for each rating level that has at least two sailors?(group by and Having)

```
SQL> select s.rating, avg(s.age) as
average from sailors s. group by
s.rating having count(*)>1;
```

Output:

RATING	AVERAGE
8	25
7	40
3	44
10	25.5

VIEW

A VIEW in SQL is a logical subset of data from one or more tables. View is used to restrict data access.

To create a view the syntax is:

CREATE or REPLACE VIEW view_name AS SELECT column_name(s) FROM table_name WHERE condition

Example: Write a query to create a view on a table

```
SQL> create view v11 as select sid, age
from sailors where rating >7;

View Created;

SQL> select * from v11;

O/P:    SID    AGE
        31     25
        32     25
        58     35
        71     16
```

To drop a view Syntax is:

Drop View View_ name

Example: Write a query to drop the existing view

```
SQL> drop view v12;

View dropped.
```

END IF;

END

Write a PL/SQL code for creation of Trigger to insert data into a table

```
SQL> Create or replace trigger t8
     before
     insert on sailors
     for each row
     begin
     :new.sname := upper (:new.sname);
     end;
     /
Trigger created.
SQL> insert into sailors values (04, 'dustin', 7, 75);
```

OUTPUT

SID	SNAME	RATING	AGE
22	dustin	7	45
29	brutus	1	33
31	lubber	8	56
32	andy	8	26
58	rusty	10	35
64	horatio	7	35
71	zorba	10	16
85	art	3	26
95	bob	3	64
4	DUSTIN	7	75

Write a PL/SQL code for creation of trigger to UPDATE data into a table:

```
SQL> Create trigger t13
     after update of sid on sailors
     for each row
     begin
     if (:new.sid<80) then
     raise_application_error(-20017,'Cant update');
     end if;
     end;
     /
Trigger created.
```

OUTPUT

```
SQL> update sailors set sid=79 where rating=10;
update sailors set sid=79 where rating=10
       *
ERROR at line 1:
ORA-20017: Cant update
ORA-06512: at "CHAITANYA.T13", line 5
ORA-04088: error during execution of trigger
'CHAITANYA.T13'

SQL> update sailors set sid=95 where
     rating=10;
```

```
SQL> Create or replace trigger t29
    after
    delete on sailors
    for each row
     begin
    if :old.sid=22 then
    raise_application_error
    (-20019,'you cant delete this row');
    end if;
    end;
    /
Trigger created.
```

OUTPUT

```
SQL> delete from sailors where sid=22;
delete from sailors where sid=22
           *
ERROR at line 1:
ORA-20019: you cant delete this row
ORA-06512: at "CHAITANYA.T29", line
ORA-04088: error during execution of
                trigger 'CHAITANYA.T29'
```

Write a PL/SQL code for creation of procedure to view some specified columns from a table:

```
> set serveroutput on
> create procedure p_sail (sid1 in number)
  is
  v_sname char(20);
  v_age number(20);
  begin
  select sname, age into v_sname, v_age from
    sailors where sid = sid1;
  dbms_output.putline ('sname: ' || v_sname);
  dbms_output.putline ('age: ' || v_age);
  end;
  /
  procedure created.
```

OUTPUT

```
> execute p_sail(22);
   Sname : dustin
     age : 45
PL/SQL procedure successfully completed.
```

rite a PL/SQL code for modification of procedure to view some specified columns om a table

```
> set serveroutput on
> create peroedure p_sail (v_sid1 in sailors.
sid%. type, v_sname in sailors.sname%. t
 v_age in sailors.age%. type) is

begin
update sailors set sname = v_sname, ag
      v_age where sid = v_sid1);

commit;
end;
/
peroedure created.
```

Ex: Fetch Emp into <variable>;

Close Cursor: Open cursor is closed

Ex:Close Emp;

Attributes used in the cursor programs:

C1%ROWCOUNT: The number of tuples in C1(C1 is cursor name)

C1%FOUND: True if the last fetch was successful

C1%ROWCOUNT: True if the last fetch was not successful

C1%ROWCOUNT: True if C1 is open

Example of cursor program

```
Declare c is select * from emp_information where emp_no<=2;
tmp emp_information%rowtype;
Begin
Open c;
for tmp in c loop
```

```
Fetch c into tmp;
Exit when c %. notfound;
Dbms_output.put_line('EMP_no:    '||tmp.emp_no);
Dbms_output.put_line('EMP_name:    '||tmp.emp_name);
Dbms_output.put_line('EMP_dept:    '||tmp.emp_dept);
Dbms_output.put_line('EMP_salary:    '||tmp.emp_salary);
End loop;
Close c;
End;
/
```

OUTPUT

```
emp_no : 0
emp_name : joy
emp_dept : cse
emp_salary : 30000
emp_no : 1
emp_name : roy
```

AIM

Write a PL/SQL program that uses cursor operation on any data base .

```
>select * from sailors;
SQL> set serveroutput on
declare
v_sname sailors.sname%type;
v_age sailors.age%type;
cursor c2 is select sname, age from sailors;
begin
open c2;
loop
fetch c2 into v_sname, v_age;
exit
when c2%rowcount>3;
dbms_output.put_line (v_sname||' '||v_age);
end loop;
close c2;
end;
/
```

A relation schema is in 1NF:

- If and only if all the attributes of the relation R are atomic in nature.
- **Atomic:** The smallest level to which data may be broken down and remain meaningful

GENERATE THE TABLE IN TO FIRST NORMAL FORM

STUDENT TABLE

Students	StudentName	Date/Birth
101	Danis	04-Nov-1986
102	Daniel	06-Nov-1987
103	Sandra	02-Oct-1988
104	Evelyn	22-Feb-1986
105	Susan	31-Aug-1985
106	Mike	04-Feb-1987
107	Juliet	09-Nov-1981
108	Tom	07-Oct-1986
109	Catherine	06-Jun-1984

COURSE TABLE

Course	Coursename	Pre-requisitis	course duration
M1	Basic mathematics		7
M4	Applied mathematics	M1	4
H6	American history		5
C1	Basic Chemistry		11
C3	Bio chemistry	C1	8
B3	Botony		8
P1	Basic physics		13
P3	Nuclear physics	P1	5
B4	Zoology		

Second normal form:2NF

- A Relation is said to be in second Normal Form if and only if:
- It is in the First normal form, and
- No Partial dependency exists between non-key attributes and key attributes.

Generate the First Normal form table in to Second Normal Form

Students	Course#	Marks	Grade
101	MH	82	A
102	MH	62	C
101	H6	79	B
103	C3	65	B
104	B3	77	B
102	P3	68	B
105	B4	89	A
103	H6	54	D
105	MH	87	A
104		65	B

Exam date table

Course#	Date of Exam
MH	11-Nov-04
H6	22-Nov-04
C3	16-Nov-04
B3	26-Nov-04
P3	12-Nov-04
B4	27-Nov-04

Third normal form: 3NF

A relation R is said to be in the third normal form(3NF) if and only if

- It is in 2NF and
- No transitive dependency exists between non –key attributes and key attributes.

Generate the Second Normal form table in to Third Normal Form

Student	Course#	Marks
101	M4	82
102	M4	62
101	H6	79
103	C3	65
104	B3	77
102	P3	68
105	P3	89
103	B4	54
105	H6	87
104	M4	65

UpperBound	LowerBound	Grade
100	95	A+
94	85	A
84	70	B
69	65	B-
64	55	C
54	45	D
44	0	E