SDM College of Engineering and Technology

Dhavalagiri, Dharwad-580 002. Karnataka State. India.
Email: principal@sdmcet.ac.in, cse.sdmcet@gmail.com
1. Ph: 0836-2447465/ 2448327 Fax: 0836-2464638 Website: sdmcet.ac.in

Department

of

COMPUTER SCIENCE AND ENGINEERING

LABORATORY REPORT

[22UCSC501-DATABASE MANAGEMENT SYSTEMS]

Odd Semester: Aug-Dec-2024

Course Teacher: Dr. U.P.Kulkarni

2024-2025

Submitted by
By

Mr. Pratham Manabasannanavar

2SD22CS063

5th Semester A division

Contents

Termwork-1: Create a data model for the given business scenario and prepare a schema in 3NF. Using the appropriate SQL statement insert the data to check or validate the following Integrity constraints.

a. Row Integrity
b. Entity Integrity
c. Referential Integrity

Problem statement:

1. Creating the Employee table
2. Creating the Project table
3. Creating the Assigned_to table

Program:

1. Creating the Employee table:

```
Create table employee(
        empno integer not null,
        constraint EMP_PK_VIOL
        primary key,
        empname char(20) not null,
        sex char(1) not null
        constraint GENDER_VIOL_EMP
        check(sex in ('m', 'f')),
        phone integer null,
        dob date not null
);
```

EMPNO	EMPNAME	S	PHONE	DOB
10	Joy	m	12344	01-JAN-04
1	Gagan	m	23445	29-MAY-04
2	TARUN	m	653656	04-APR-90
3	Darshan	m	533	20-JUL-10

2. Creating the Project table

```
create table project(
        projectno integer not null,
        projectname char(20) not null,
        chiefarchitect char(20) default ‘upk’ not null,
        constraint PROJECT_PK_VIOL
        primary key(projectno)
);
```

PROJECTNO	PROJECTNAME	CHIEFARCHITECT
1	Google lens	pratham
5	Micro	upk
2	DBMS	upk
3	Portal	Joy

3. Creating the table Assigned_to

```
create table assigned_to(
    empNo integer not null,
    projectNo integer not null,
    constraint ASSIGNED_TO_PK_VIOLATION
    primary key(empno, projectno),
    constraint ASSIGNED_TO_EMP_VIOLATION
    foreign key (empno)
    references employee,
    constraint ASSIGNED_TO_FK_PRJ_VIOLATION
    foreign key(projectno)
    references project
    );
```

EMPNO	PROJECTNO
1	2
3	1
3	2
3	3
3	4
3	5

Termwork-2: Write SQL Queries for the following:

a. Write a SQL statement to obtain the employee no who is working on project 1

Query:

```
select empno
from assigned_to
where projectno=1;
```

O/P:

```
 EMPNO
--------------
         3
```

b. Write a SQL statement to get the details of all the employees working on project no 1

Query:

```
select e.*
 from employee e, assigned_to at
 where at.projectno=1 and e.empno = at.empno;
```

O/P:

```
EMPNO EMPNAME            S      PHONE       DOB
------ ------------------- - ---------- ---------   --------------------
    3    Darshan             m             533     20-JUL-10
```

c. Write a SQL statement to get the details of employees working on ‘DBMS’ Project

Query:

```
select e.* from
employee e, assigned_to at, project p
where p.projectname='DBMS' and e.empno = at.empno and p.projectno = at.projectno;
```

O/P:

```
    EMPNO EMPNAME              S      PHONE DOB
---------- -------------------- - ---------- ---------
         1 Gagan                m      23445 29-MAY-04
         3 Darshan              m        533 20-JUL-10
```

d. Write a SQL statement to get the details of the employee working on the project 1 and 2.

Query:

```
select e.*
from employee e, assigned_to at, project p
where e.empno=at.empno and at.projectno=p.projectno and at.projectno=1
intersect select e.* from employee e, assigned_to at, project p where
e.empno=at.empno and at.projectno=p.projectno and at.projectno=2;
```

O/P:

```
    EMPNO EMPNAME              S      PHONE DOB
---------- -------------------- - ---------- ---------
         3 Darshan              m        533 20-JUL-10
```

e. Write a SQL statement to get the details of employees working on projects 1 or 2.

Query:

```
select e.*, p.projectname
from employee e, assigned_to at, project p
where (at.projectno = 1 or at.projectno = 2) and e.empno = at.empno and
p.projectno = at.projectno;
```

O/P:

```
EMPNO EMPNAME            S      PHONE DOB
------- -------------------- - ---------- ---------
      1 Gagan                m      23445 29-MAY-04
      3 Darshan              m        533 20-JUL-10
      3 Darshan              m        533 20-JUL-10
```

Termwork - 3: Prepare a modified schema to store the information about fine to be paid by each employee.

Query:

```
create table EMPFINE(
empno int not null,
fine int not null,
constraint EMPNO_FK_ERR
foreign key(empno) references employee(empno)
);
```

O/P:

```
    EMPNO       FINE
---------- ----------
         2        300
         1       3000
         1        100
         3        600
         3        600
```

Termwork-4: Modify the schema to store the information about the dependents of each employee if exists.

Query:

```
create table EMPDEP(
empno int not null,
```

```
dName char(30) not null,
dpRelation char(1) not null,
constraint DEP_EMP_RELATION_CHECK
check (dpRelation in ('m', 'f')),
constraint EMP_FK_VIOLATION
foreign key(empno) references employee(empno)
);
```

O/P:

CUSTID	DEPNAME
1	Harish
1	Manoj
3	Tarun

Termwork-5: Study the following:

Creating the temporary table Student:

NAME	ROLLNO
Prateek	62
Prajwal	58
Pratham	63
Prasanna	61

1. Updating the rows:

Query:

```
SQL> update student set name='Prateek Pandarikar' where rollno=62;
```

O/P:

NAME	ROLLNO
Prateek Pandarikar	62
Prajwal	58
Pratham	63
Prasanna	61

1 row updated.

2. Deleting the rows of the table:

Query

```
delete from student

where rollno=63;
```

O/P:

```
NAME                 ROLLNO
-------------------- ----------
Prateek Pandarikar         62
Prajwal                    58
Prasanna                   61
```

3. Dropping the table:

Query:

```
drop table student;
```

O/P:

```
TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
ASSIGNED_TO                    TABLE
BIN$Llq6YoFoSPuJLYY            TABLE
EMPDEP                         TABLE
EMPFINE                        TABLE
EMPLOYEE                       TABLE
PROJECT                        TABLE

6 rows selected.
```

Termwork-6: Study the impact of deleting the rows and dropping the table on Referential integrity.

```
SQL> insert into student values('Jayanth', 3);

1 row created.

SQL> select * from employee;
```

```
    EMPNO EMPNAME            S      PHONE DOB
--------- -------------------- - ---------- ---------
        1 Gagan                m      23445 29-MAY-04
        2 TARUN                  m     653656 04-APR-90
        3 Darshan              m        533 20-JUL-10

SQL> insert into employee values(10, 'Joy', 'm', 12344, '1-Jan-2004');

1 row created.

SQL> insert into student values('Joy', 10);

1 row created.

SQL> delete from employee where empno=10;
delete from employee where empno=10
*
ERROR at line 1:
ORA-02292: integrity constraint (22CS063.SYS_C00120091) violated - child record
found
```

Termwork-7: Display names of all the employees who are on bench.

Query:

```
select empname, dob
from employee
where empno not in (
select distinct empno from assigned_to
);
```

O/P:

```
EMPNAME              DOB
-------------------- ---------
TARUN                04-APR-90
Joy                  01-JAN-04
```

Termwork-8: Display names of all the employees working on the DBMS project.

Query:

```
select empname, dob

from employee

where empno not in (select distinct empno from assigned_to);
```

O/P:

```
EMPNAME              DOB
-------------------- ---------
TARUN                04-APR-90
Joy                  01-JAN-04
```

Termwork-9: Display name of all the employees working on atleast on all the project that employee 1 is working.

Query:

```
select empno

from assigned_to group by empno

having count(distinct projectno) = (select count(*) from project);
```

O/P:

```
    EMPNO
----------
       3
```

Termwork-10: Display the details of the top 3 senior employees

Query:

```
Select * from (

Select * from employee

Order by dob

) where rownum <= 3;
```

O/P:

name	empid	dob
Dinesh	1	2013-05-12
Rahul	6	2014-09-09
Naveen	3	2015-02-02
Jagadesh	4	2019-04-28
harish	2	2023-01-02

Termwork-11: Find for each employee the penalty incurred.

Creating the EMPFINE table:

```
create table empfine(
   empid int,
   fine decimal(10, 2),
   foreign key(empid) references employee(empid)
);
```

EMPID	FINE
1	200
1	200
2	580
5	30
4	500
4	500
2	20

Query:

```
select e.empid, name, NVL(fine, 0)
from employee e left join (select empid, sum(fine) fine from empfine f
group by f.empid) f
on f.empid = e.empid;
```

O/P:

EMPID	NAME	NVL(FINE,0)
1	Raju	400
2	Naveen	600
5	Jay	30
4	Akshay	1000
3	Manas	0

Termwork-12: Display the count of the employees working on each project having count >= 3.

Query:

```
select empno from assigned_to
group by empno
having count(projectno) > 3;
```

O/P:

EMPNO
3

Termwork-13: Study of Order by and Alter clause.

1. Order by Clause:

Printing the Junior most employees from top to bottom.

Query:

```
Select *

from employee1

order by dob DESC;
```

O/P:

```
EMPID  FINE

------------

1      200

1      200

2      580

5      30

4      500

4      500

2      20
```

2. Alter Clause:

Adding the dob column in the employee table

Query1:

```
Alter table employee add dob date not null;
```

O/P1:

EMPID	NAME	SALARY	EMAIL
1	Raju	52.33	-
2	Naveen	2.33	-
4	Akshay	7.33	-
3	Manas	43.3	-
5	Jay	5	-

Query2:

Alter table employee drop column dob;

O/P2:

EMPID	NAME	SALARY
1	Raju	52.33
2	Naveen	2.33
4	Akshay	7.33
3	Manas	43.3
5	Jay	5

Termwork-14: Study of statistical functions:

1. min ():

Query:

```
select min(fine)
from empfine;
```

O/P:

MIN(FINE)
20

2. max ():

Query:

```
select max(fine)
from empfine;
```

O/P:

```
MAX(FINE)
---------
      580
```

3. sum ():

Query:

```
select sum(fine)
from empfine;
```

O/P:

```
SUM(FINE)
---------
     2030
```

4. avg ():

Query:

```
select avg(fine)
from empfine;
```

O/P:

```
AVG(FINE)
---------
      290
```

5. STDDEV ():

Query:

```
select stddev(fine)
from empfine;
```

O/P:

STDDEV(FINE)

234.301893

6. Variance ():

Query:

```
select variance(fine)
from empfine;
```

O/P:

VARIANCE(FINE)

54833. 3333

Termwork-15: Study of:

1. Between

Query:

```
select *
from employee1
where dob between to_date('1-Jan-2013', 'DD-Mon-YYYY') and to_date('1-
Jan-2018', 'DD-Mon-YYYY');
```

O/P:

```
NAME      EMPID DOB
--------- ------- ----------
Dinesh        1 12-MAY-13
Naveen        3 02-FEB-15
Rahul         6 09-SEP-14
```

2. Like:

Query:

```
select * from employee1
where empname like 'J%';
```

O/P:

EMPID	NAME	SALARY
5	Jay	5

3. All:

Query:

Printing the employees except the employee with max salary.

```
select *
from employee1
where salary >= all (select salary from employee);
```

O/P:

EMPID	NAME	SALARY
1	Raju	52.33

4. Any:

Query:

Printing the employees except the employee with least salary.

```
select *
from employee1
where salary > any (select salary from employee);
```

O/P:

EMPID	**NAME**	**SALARY**
1	Raju	52.33
3	Manas	43.3
4	Akshay	7.33

5	Jay	5

5. In:

Query:

Printing the employees working on the project 2 or 3.

```
select e.*, at.pid
from employee e, assigned_to at
where e.empid = at.empid and pid in(2,3);
```

O/P:

EMPID	NAME	SALARY	PID
1	Raju	52.33	2
2	Naveen	2.33	2
2	Naveen	2.33	3

6. Exists:

Query:

```
select distinct(e.empid), e.*
from employee e, assigned_to at
where not exists(
   (select pid from assigned_to where empid=1)
   Minus
   (select pid from assigned_to at where e.empid = at.empid)
);
```

O/P:

EMPID	**EMPID**	**NAME**	**SALARY**
1	1	Raju	52.33
2	2	Naveen	2.33

7. Rownum:

Query:

```
select * from
(select * from employee order by dob)
where rownum<=2;
```

O/P:

```
EMPID NAME  SALARY
---------- ---------- ----------
1          Raju      52.33
3          Manas     43.3
```

8. Count:

Query:

```
select count(*) from employee;
```

O/P:

```
COUNT(*)
       5
```

Termwork-16: Study of Date related functions:

Query1:

```
select to_char(to_date('12-Feb-2015', 'dd-Mon-YYYY'), 'DY')
from dual;
```

O/P1:

```
TO_CHAR(TO_DATE('12-FEB-2015','DD-MON-YYYY'),'DY')
--------------------------------------------------------------------------------
THU
```

Query2:

```
select to_char(to_date('12-Feb-2015', 'dd-Mon-YYYY'), 'SYEAR')
from dual;
```

O/P2:

```
TO_CHAR(TO_DATE('12-FEB-2015','DD-MON-YYYY'),'SYEAR')
--------------------------------------------------------------------------------
TWENTY FIFTEEN
```

Query3:

```
select ADD_MONTHS (to_date('12-Feb-2015', 'dd-Mon-YYYY'), 1)
from dual;
```

O/P3:

```
ADD_MONTHS (TO_DATE('12-FEB-2015','DD-MON-YYYY'), 1)
--------------------------------------------------------------------------------
12-MAR-2015
```

Query4:

```
select months_between(to_date('12-Feb-2015', 'dd-Mon-YYYY'), to_date('19-
Oct -2015', 'dd-Mon-YYYY')) Months_diff from dual;
```

O/P4:

```
MONTHS_DIFF
-------------------
12-MAR-2015
```

Termwork-17: Study of Views

1. With check option:

Query1: Creating the view

```
create view empview1 as (
select * from employee where empid<3
);
```

Query2:

```
Select *
from empview1;
```

O/P2:

EMPID	NAME	SALARY
1	Raju	52.33
2	Naveen	2.33

Query3:

```
insert into empview1 values(9, 'Rohan', 25.2);
```

O/P3:

EMPID	NAME	SALARY
1	Raju	52.33
2	Naveen	2.33
4	Akshay	7.33
3	Manas	43.3
5	Jay	5
9	Rohan	25.2

2. Without Check Option:

Query:

```
Create view empview2 as (select * from employee);
```

O/P:

EMPID	NAME	SALARY
1	Raju	52.33
2	Naveen	2.33
4	Akshay	7.33
3	Manas	43.3
5	Jay	5
9	Rohan	25.2

Termwork-18: Study of Copying table and synonym

Copying table:

Query:

```
Create table employee2 as Select * from employee;
```

O/P:

EMPID	**NAME**	**SALARY**
1	Raju	52.33
2	Naveen	2.33
4	Akshay	7.33
3	Manas	43.3
5	Jay	5
9	Rohan	25.2

Synonym:

Query1:

```
Create synonym emp2 for employee;
```

Query2:

```
select * from emp2;
```

O/P:

EMPID	NAME	SALARY
1	Raju	52.33
2	Naveen	2.33
4	Akshay	7.33
3	Manas	43.3
5	Jay	5
9	Rohan	25.2

Termwork-19: Study of PL SQL

Query1:

```
declare
sal decimal(10,2);
empid int;
begin
select salary, empid into sal, empid
from employee e
where empid=3;
dbms_output.put_line('employee salary is ' || sal);
dbms_output.put_line('employee id is ' || empid);
end;
/
```

O/P1:

```
employee salary is 43.3
employee id is 3
```

Query2:

```
begin
```

```
for i in 1..10 loop
if(mod(i, 2) = 0) then
dbms_output.put_line(i);
end if;
end loop;
end;
/
```

O/P2:

```
2
4
6
8
10
```

Termwork-20: Study of Triggers:

Query1:

Creating a Deleted_EMP table for storing the employees who are deleted from employee table.

```
create or replace trigger employeeTrig
after delete on employee1 for each row

begin
    insert into deletedEmp values(:old.empno, :old.empname);
    dbms_output.put_line('done');
end;
/
```

O/P1:

EMPID	NAME	SALARY
1	Raju	52.33

Query2:

Adding the constraint to the assigned_to such that 1 employee can work on only 3 projects.

```
create or replace trigger assigned_toTrig
before insert on assigned_to for each row
declare
    projcount int;
begin
    select count(distinct pid) into projcount from assigned_to at
    where :new.empid = at.empid;
    dbms_output.put_line(projcount);
    if(projCount >= 3) then
        RAISE_APPLICATION_ERROR(-20001, 'EMPLOYEE CANT
        WORK ON > 3 PROJECTS');
    end if;
end;
/
```

O/P2:

Employee can't work on > 3 projects.

Termwork-21: Study of Stored procedures.

Query:

```
create or replace procedure findemployee(empid in number) as
  emprow employee%rowtype;
```

```
begin
  select * into emprow
  from employee e
  where e.empid = empid
  and rownum = 1;

  dbms_output.put_line('empname: ' || emprow.name);
  dbms_output.put_line('empno: ' || emprow.empid);

exception
  when no_data_found then
    dbms_output.put_line('no employee found with empid: ' || empid);

  when too_many_rows then
    dbms_output.put_line('more than one employee found with empid: ' ||
empid);

  when others then
    dbms_output.put_line('an error occurred: ' || sqlerrm);
end;
/
```

O/P:

```
exec findEmployee(2);
statement processed.
Empname: Raju
Empno: 1
```

Termwork-22: Study for Stored functions

Query:

```
create or replace function countEmployee
        return int is
        countemp int;
        begin
        select count(*) into countemp from employee;
        return countemp;
        end;
/
```

O/P:

```
begin
        dbms_output.put_line('employee count = ' || countEmployee());
end;
Statement processed.
employee count = 6
```

Termwork-23: Study of cursors:

Query:

```
begin
open empCursor;
dbms_output.put_line('The employee names are:');
loop
        fetch empCursor into empdetails;
        exit when empCursor%NOTFOUND;
        dbms_output.put_line(empdetails.name);
end loop;
close empCursor;
end;
```

O/P:

Statement processed.
Raju
Naveen
Akshay
Manas
Jay
Rohan

Termwork-24: Study of Transactions

Query and O/P 1:

```
Set autocommit off;

insert into employee1 values('Gagan',5,'3-April-2000');

select * from employee1;
```

NAME	EMPID	DOB
Dinesh	1	2013-05-12
harish	2	2023-01-02
Naveen	3	2015-02-02
Jagadesh	4	2019-04-28
Gagan	5	0000-00-00
Rahul	6	2014-09-09

6 rows selected.

```
Rollback;
Select * from employee1;
```

NAME	EMPID	DOB

Dinesh 1 2013-05-12

harish 2 2023-01-02

Naveen 3 2015-02-02

Jagadesh 4 2019-04-28

Rahul 6 2014-09-09

5 rows selected.

Query and O/P 2:

delete from employee1 where empid=6;

Query OK, 1 row affected (0.00 sec)

savepoint s1;

Query OK, 0 rows affected (0.00 sec)

select * from employee1;

name id dob

--- --- ---

Dinesh 1 2013-05-12

harish 2 2023-01-02

Naveen 3 2015-02-02

Jagadesh 4 2019-04-28

4 rows selected.

update employee1 set name = 'Praveen' where name='Naveen';

Query OK, 1 row affected (0.00 sec)

Rows matched: 1 Changed: 1 Warnings: 0

select * from employee1;

NAME EMPID DOB

---------- ----- ------------

Dinesh 1 2013-05-12

harish 2 2023-01-02

Praveen 3 2015-02-02

Jagadesh 4 2019-04-28

4 rows selected.

rollback s1;

rollback to s1;

Query OK, 0 rows affected (0.00 sec)

select * from employee1;

NAME EMPID DOB

---------- ----- ------------

Dinesh 1 2013-05-12

harish 2 2023-01-02

Naveen 3 2015-02-02

Jagadesh 4 2019-04-28

4 rows selected.

rollback;

Query OK, 0 rows affected (0.03 sec)

select * from employee1;

NAME	EMPID	DOB
Dinesh	1	2013-05-12
harish	2	2023-01-02
Naveen	3	2015-02-02
Jagadesh	4	2019-04-28
Rahul	6	2014-09-09

4 rows selected.

References:

1. https://www.w3schools.com/SQL/deFault.asp
2. https://www.geeksforgeeks.org/sql-tutorial/