Database Management System

Additional Group Activity report

On

College Database Management System

Submitted for the partial fulfillment of Bachelor of Engineering

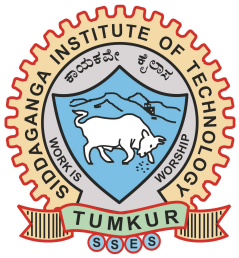
By

Bharath K S	**1SI20CS021**
Chandan Kshatriya H V	**1SI20CS025**
Chinmay Shankar S S	**1SI20CS027**
Chinmayi B	**1SI20CS029**

Under the guidance of

Dr. Srinivasa K

Assistant professor



Department of Computer Science and Engineering

(Program Accredited by NBA)

Siddaganga Institute of Technology, Tumakuru – 572103

(An autonomous institution affiliated to VTU, Belagavi, Approved by AICTE, New Delhi,
Accredited by NAAC with 'A' grade & ISO 9001:2015 Certified)

2022-2023

Problem Title

College Database Management System

Requirement Collection

Staff : The "Staff" entity type in a college database management system is a representation of an individual who is employed by the college or university in a support role, such as administrative staff, maintenance staff, or security staff. This entity type is used to store information about each staff member and is typically comprised of several attributes or fields that describe the staff member's role, responsibilities, and employment status.

- **sid (Staff ID):** A unique identifier assigned to each staff member.
- **name (Name) :** Name of the staff.
- **role(Role) :** The job title or role the staff member holds within the college or university.

Department : The "Department" entity type in a college database management system is a representation of an academic department within a college or university. This entity type is used to store information about each department and is typically comprised of several attributes or fields that describe the department's characteristics and structure.

- **dept_no(Department Number) :** The department or faculty the student is enrolled in.
- **name(Name):** The name of the department.
- **hod_id (Head of Department ID):** The identifier of the head of the department.

Teacher : The "Teacher" entity type in a college database management system is a representation of a faculty member who teaches courses and provides academic instruction within a college or university. This entity type is used to store information about each teacher and is typically comprised of several attributes or fields that describe the teacher's qualifications, experience, and teaching responsibilities.

- **t_id (Teacher ID):** A unique identifier assigned to each teacher.
- **sid (Staff ID):** A unique identifier assigned to each staff member.
- **dno(Department Number) :** The department or faculty the student is enrolled in.
- **exp(Experience) :** The number of years of teaching experience the teacher has.

Hostel : The "Hostel" entity type in a college database management system is a representation of a student housing facility within the college or university. This entity type is used to store information about each hostel and is typically comprised of several attributes or fields that describe the hostel's location, capacity, and amenities.

- **hname(Hostel Name) :** The name of the hostel.

- **gender(Gender) :** The gender for which the hostel is designated (e.g. male, female).
- **managerid(Manager ID) :** The identifier of the staff member who serves as the manager or administrator of the hostel.

Subject : The "Subject" entity type in a college database management system is a representation of a course or academic program offered by the college or university. This entity type is used to store information about each subject and is typically comprised of several attributes or fields that describe the subject's content, structure, and requirements.

- **subid(Subject ID) :** A unique identifier assigned to each subject.
- **sub_name(Subject Name) :** The name of the subject.
- **max_credit(Maximum Credits) :** The maximum number of credits a student can earn for completing the subject.

ADMISSION : The "Admission" entity type in a college database management system is a representation of the admission process for a student who seeks to enroll in the college or university. This entity type is used to store information.

- **type(Admission type) :** This provides the details of the type of admission such as KCET, COMED-K, Management.
- **fees(Fees) :** The amount to be paid each year based on the type of admission.

STUDENT: The student entity type is a representation of an individual student who is enrolled in a college or university. This entity type is used to store information about each student and is typically comprised of several attributes or fields that describe the student's personal and academic information.

- **usn(University Seat Number):** A unique identifier assigned to each student, which is specific to their university or college.
- **fn(First Name):** The student's given name.
- **ln(Last Name):** The student's family name.
- **yog(Year of graduation) :** The year in which the student graduates.
- **sem(Semester) :** The semester in which the student is studying.
- **dept_no(Department Number):** The department or faculty the student is enrolled in.
- **leader(Leader):** Indicates whether the student is a leader in any student organizations or clubs.
- **admis_type(Admission Type):** The type of admission the student was granted, such as regular admission, transfer admission, or provisional admission.

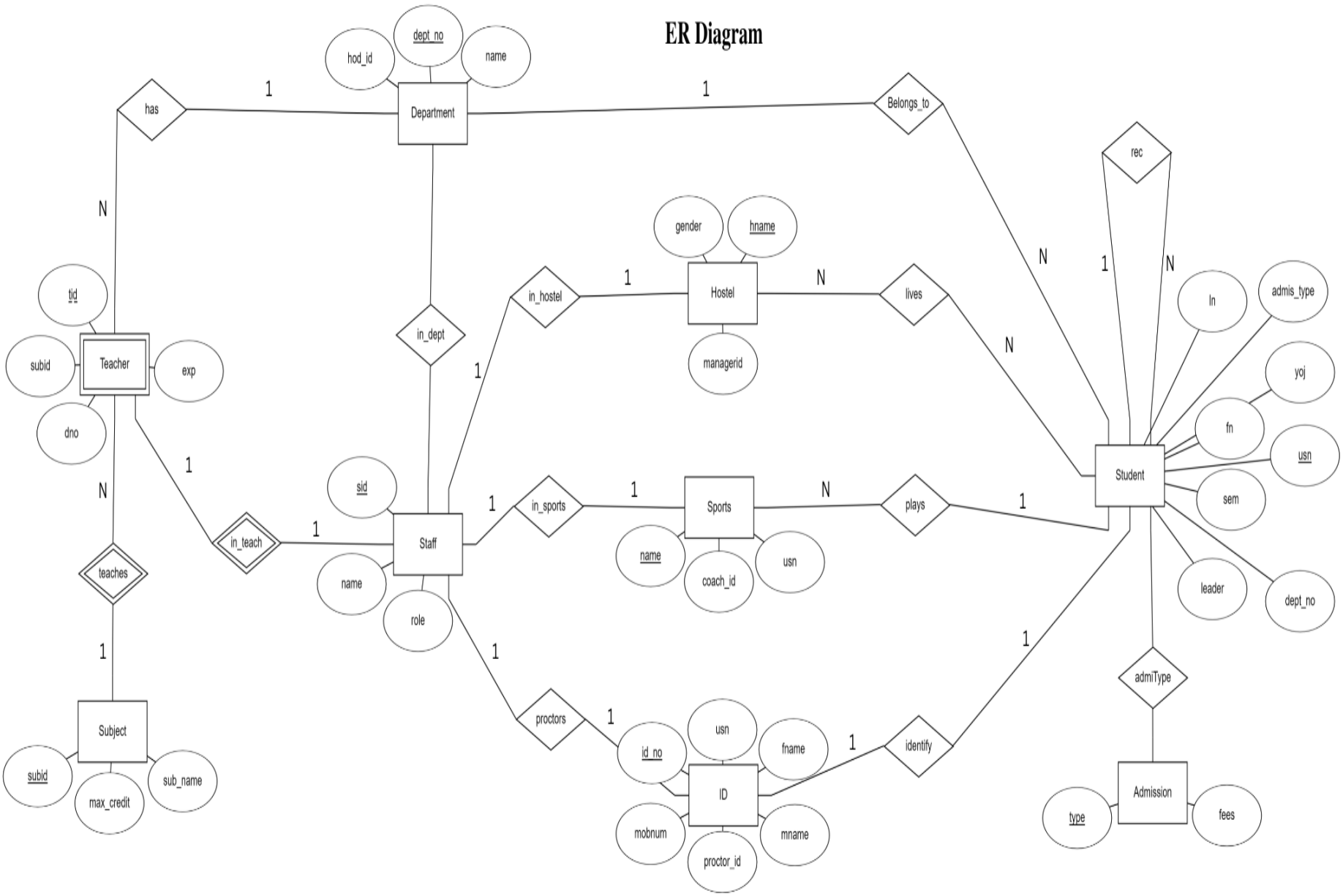
SPORTS: The "Sports" entity type in a college database management system is a representation of the athletic programs and activities offered by the college or university. This entity type is

used to store information about each sport and is typically comprised of several attributes or fields that describe the sport's type, coach Id ,usn of the students

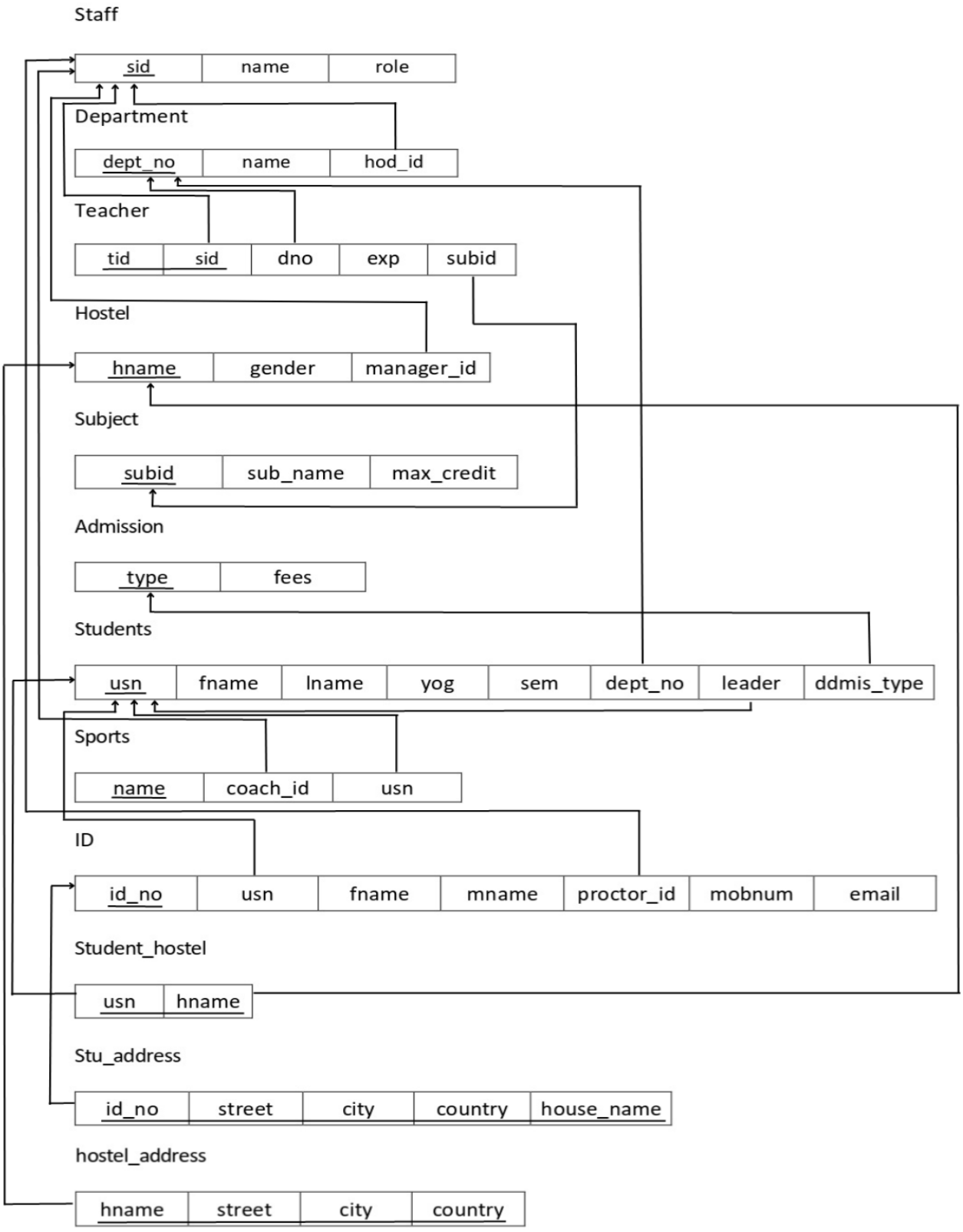
- **name(Name):** name of the sports.
- **coach_id(Coach Id):** staff ID of the coach teaching the particular sports.
- **usn(University Seat Number):** A unique identifier assigned to each student, which is specific to their university or college.

ID : The "ID" entity type in a college database management system is a unique identifier that is assigned to each entity within the database, such as a student, staff member, subject, or admission. The purpose of the ID entity type is to provide a unique identifier for each entity, which can be used to track and manage information about each entity within the database.

- **id_no(Identity number) :** Number used to uniquely identify each ID of a student.
- **fname** : Father's name.
- **mname** : Mother's name.
- **mobnum**: mobile number of the student.
- **email**: e-mail ID of the student.
- **proctor_id**: staff ID of the proctor who monitors the student.


ER Diagram
dept_no
hod_id
name
Department
has
Belongs_to
rec
tid
Teacher
subid
exp
dno
in_dept
in_hostel
Hostel
gender
hname
managerid
lives
Student
ln
admis_type
yoj
fn
usn
sem
leader
dept_no
sid
Staff
name
role
in_teach
teaches
Subject
subid
sub_name
max_credit
in_sports
Sports
name
coach_id
usn
plays
proctors
id_no
ID
usn
fname
mobnum
mname
proctor_id
identify
admiType
Admission
type
fees
1
N

Relational Mapping



DDL statements

```
create table staff(
sid int primary key,
name varchar(10),
role varchar(15)
);

create table department(
dept_no int primary key,
name varchar(10),
hod_id int references staff(sid)
);

create table hostel(
hname varchar(10) primary key,
gender varchar(10),
managerid int references staff(sid)
);

create table subject(
subid int primary key,
sub_name varchar(10),
max_credit int
);
```

```
create table teacher(
tid int,
sid int references staff(sid),
dno int references department(dept_no),
subid int references subject(subid),
exp int,
primary key(sid,tid)
);

create table admission(
type varchar(10) primary key,
fees int
);

create table student(
usn varchar(10) primary key,
fn varchar(10),
ln varchar(10),
yog date,
sem int,
dept_no int references department(dept_no),
leader varchar(10) references student(usn),
admis_type varchar(10) references admission(type)
);
```

```
create table sports
(
name varchar(10) ,
coach_id int references staff(sid),
usn varchar(10) references student(usn),
primary key(name,usn)
);

create table id
(
id_no int primary key,
usn varchar(10) references student(usn),
fname varchar(10),
mname varchar(10),
proctor_id int references staff(sid),
mobnum number(10),
email varchar(25)
);

create table student_hostel
(
  usn varchar(10) references student(usn),
  hname varchar(10) references hostel(hname),
 primary key(usn,hname)
);
```

```
create table stu_address
(
id_no int references id(id_no),
street varchar(15),
city varchar(15),
country varchar(15),
house_name varchar(15),
primary key(id_no,house_name,street,city,country)
);

create table hostel_address
(
hname varchar(10) references hostel(hname),
street varchar(15),
city varchar(15),
country varchar(15),
primary key(hname,street,city,country)
);

CREATE TABLE dummy_table (
id NUMBER primary key,
name VARCHAR2(50),
date_created DATE
);
ALTER TABLE dummy_table
```

```
ADD column_name VARCHAR2(50);


DROP TABLE dummy_table;
```

VIEWS

A view to show the details of the staff and their respective departments:

```
CREATE VIEW staff_dept AS

SELECT s.sid, s.name, d.name as department

FROM staff s,department d

where s.sid = d.hod_id;
```

A view to show the details of the students, their respective departments and their admission types:

```
CREATE VIEW student_dept_admis AS

SELECT s.usn, s.fn, s.ln, d.name as department, a.type as admission_type

FROM student s,department d,admission a

where s.dept_no = d.dept_no

and s.admis_type = a.type;
```

Insert statements

staff:

```
insert into staff values (1, 'John', 'Principal');
insert into staff values (2, 'Jane', 'Manager');
insert into staff values (3, 'Bob', 'Teacher');
insert into staff values (4, 'Sarah', 'FOOTBALL Coach');
insert into staff values (18, 'barah', 'Coco Coach');
insert into staff values (19, 'marah', 'Cricket Coach');
insert into staff values (5, 'avk', 'proctor');
insert into staff values (6, 'srinivas', 'proctor');
insert into staff values (7, 'ysn', 'proctor');
insert into staff values (8, 'avk', 'proctor');
insert into staff values (9, 'ram', 'Teacher');
insert into staff values (10, 'sham', 'Teacher');
insert into staff values (11, 'bhem', 'Teacher');
insert into staff values (12, 'rao', 'Teacher');
insert into staff values (13, 'poornima', 'HOD');
insert into staff values (14, 'shagun', 'HOD');
insert into staff values (15, 'shamraj', 'HOD');
insert into staff values (16, 'bhemraj', 'HOD');
insert into staff values (17, 'kulkarni', 'HOD');
```

department:

```
insert into department values (101, 'Science', 13);
```

insert into department values (102, 'Arts', 14);

insert into department values (103, 'Sports', 15);

INSERT INTO department VALUES (104, 'Commerce', 16);

INSERT INTO department VALUES (105, 'Medical', 17);

hostel:

insert into hostel values ('Basava', 'Male', 2);

insert into hostel values ('Akkamaha', 'Female', 3);

subject:

insert into subject values (1001, 'Maths', 4);

insert into subject values (1002, 'History', 3);

insert into subject values (1003, 'English', 2);

insert into subject values (1004, 'Physics', 4);

insert into subject values (1005, 'Biology', 3);

insert into subject values (1006, 'ZOology', 3);

teacher:

insert into teacher values (1, 3, 101, 1001, 10);

insert into teacher values (2, 10, 102, 1002, 5);

insert into teacher values (3, 11, 103, 1003, 3);

insert into teacher values (4, 12, 103, 1003, 13);

admission:

insert into admission values ('COMEDK', 100000);

insert into admission values ('KCET', 50000);

insert into admission values ('Management', 150000);

student:

insert into student values ('1MS19CS001', 'John', 'Doe', '2019-01-01', 3, 101, NULL, 'COMEDK');

insert into student values ('1MS18CS002', 'Jane', 'Doe', '2020-01-01', 2, 102, '1MS19CS001', 'KCET');

insert into student values ('1MS18CS003', 'Jane', 'Doe', '2020-01-01', 2, 102, '1MS19CS001', 'KCET');

insert into student values ('1MS18CS004', 'Sarah', 'Johnson', '2020-01-01', 2, 102, '1MS18CS002', 'Management');

insert into student values ('1MS17CS005', 'Joseph', 'Williams', '2020-01-01', 1, 104, '1MS19CS001', 'KCET');

insert into student values ('1MS19CS006', 'abhiram', 'Williams', '2019-01-01', 3, 105, '1MS18CS004', 'COMEDK');

insert into student values ('1MS19CS005', 'John', 'Doe', '2023-01-01', 3, 101, '1MS18CS004', 'COMEDK');

sports:

insert into sports values ('Football', 4, '1MS19CS001');

insert into sports values ('Coco',18, '1MS18CS002');

insert into sports values ('Cricket', 19, '1MS18CS003');

insert into sports values ('Football', 4, '1MS18CS004');

insert into sports values ('Coco',18, '1MS17CS005');

insert into sports values ('Cricket', 19, '1MS19CS006');

id:

insert into id values (1, '1MS19CS001', 'John', 'Doe', 5, 1234567890, 'johndoe@gmail.com');

insert into id values (2, '1MS18CS002', 'Jane', 'Doe', 5, 2345678901, 'janedoe@gmail.com');

insert into id values (3, '1MS18CS003', 'Bob', 'Smith', 6, 3456789012, 'bobsmith@gmail.com');

insert into id values (4, '1MS18CS004', 'Sarah', 'Johnson', 7, 4567890123, 'sarahjohnson@gmail');

insert into id values (5, '1MS17CS005', 'Joseph', 'williams', 8, 3456789012, 'bobsmith@gmail.com');

insert into id values (6, '1MS19CS006', 'abhiram', 'williams', 7, 4567890123, 'sarahjohnson@gmail.com');

student_hostel:

INSERT INTO student_hostel (usn, hname) VALUES ('1MS19CS001', 'Basava');

INSERT INTO student_hostel (usn, hname) VALUES ('1MS18CS002', 'Basava');

INSERT INTO student_hostel (usn, hname) VALUES ('1MS19CS006', 'Akkamaha');

student_address:

INSERT INTO stu_address VALUES (1, '123 Main St', 'Seattle', 'Washington', 'Akshay');

INSERT INTO stu_address VALUES (2, '12 Main St', 'New York', 'NY', 'Lakshmi');

INSERT INTO stu_address VALUES (3, '20 Oxford St', 'London', 'England', 'kshatriya');

INSERT INTO stu_address VALUES (4, '456 Oak Ave', 'Portland', 'Oregon', 'Mallikarjuna');

INSERT INTO stu_address VALUES (5, '789 Maple Blvd', 'San Francisco', 'California', 'shivashankar');

INSERT INTO stu_address VALUES (6, '111 Pine St', 'Los Angeles', 'California', 'shivakrupa');

hostel_address:

INSERT INTO hostel_address (hname, street, city, country) VALUES ('Basava','456 Park Ave', 'London', 'United Kingdom');

INSERT INTO hostel_address (hname, street, city, country) VALUES ('Akkamaha', '789 Ocean Dr', 'Miami', 'United States');

Queries

select * from student;

select * from id where usn = '1MS18CS003';

SELECT id_no, fname, mname, email FROM id WHERE proctor_id = 7;

To retrieve all rows from the student table where the yog is before '2022-01-01':

SELECT * FROM student WHERE yog < '2022-01-01';

aggregate

SELECT AVG(exp) avg_of_exp FROM teacher;

SELECT AVG(max_credit) FROM subject;

SELECT MIN(fees) FROM admission;

SELECT MAX(fees) FROM admission;

SELECT MAX(max_credit) FROM subject;

SELECT SUM(max_credit) FROM subject;

SELECT COUNT(DISTINCT proctor_id) FROM id;

alias

SELECT COUNT(*) total_Staff FROM staff;

distinct

Find the unique subject names in the "subject" table:

SELECT DISTINCT sub_name

FROM subject;

pattern matching

To find all the student first names which contain the pattern "A%":

SELECT *

FROM student

WHERE fn like 'a%';

Arithmetic operation

Find the number of students enrolled in each department, along with the department name:

SELECT department.name, COUNT(student.usn)

FROM student

JOIN department ON student.dept_no = department.dept_no

GROUP BY department.name;

order by

To order the records in the student table in descending order of their yog (year of graduation), use the following query:

```
SELECT * FROM student

ORDER BY yog DESC;
```

To order the records in the teacher table by the exp field, and break ties by ordering by the tid field in ascending order, use the following query:

```
SELECT * FROM teacher

ORDER BY exp DESC, tid ASC;
```

Nesting

To find the names of all students who are staying in hostel 'Boys Hostel':

```
SELECT fn, ln

FROM student

WHERE usn IN (SELECT usn FROM student_hostel WHERE hname = 'Akkamaha');
```

Retrieve the name and the head of department of all departments.

```
SELECT d.name as department_name, s.name as hod_name

FROM department d ,staff s

where d.hod_id = s.sid;
```

EXISTS

Find all departments where any of the teachers have more than 10 years of experience:

```
SELECT * FROM department
```

```
WHERE EXISTS (SELECT * FROM teacher
        WHERE dno = department.dept_no
         AND exp > 10);
```

UPDATE

UPDATE query to increase the fees for admission type 'BTech' by 1000:

```
UPDATE admission
SET fees = fees + 1000
WHERE type = 'Management';
```

VIEWS

A view to show the details of the staff and their respective departments:

```
CREATE VIEW staff_dept AS
SELECT s.sid, s.name, d.name as department
FROM staff s,department d
where s.sid = d.hod_id;
```

A view to show the details of the students, their respective departments and their admission types:

```
CREATE VIEW student_dept_admis AS
SELECT s.usn, s.fn, s.ln, d.name as department, a.type as admission_type
FROM student s,department d,admission a
where s.dept_no = d.dept_no
and s.admis_type = a.type;
```

Stored procedures

1) **Procedure to retrieve all students in a specific hostel:**

```
CREATE OR REPLACE PROCEDURE display_students_in_hostel (hostel_name varchar)
IS
student_rec student%rowtype;
cursor c is select s.* from student s, student_hostel sh
where sh.hname = hostel_name and s.usn = sh.usn;
begin
for student_rec in c loop
sys.dbms_output.put_line(student_rec.usn||' '||student_rec.fn||' '||student_rec.ln);
end loop;
end;
/
```

2) **Display all departments along with the name of their Head of Department:**

```
CREATE OR REPLACE PROCEDURE prc_display_hod_and_dept IS
CURSOR c_dept IS
SELECT d.dept_no, d.name as dept_name, s.name as hod_name
FROM department d
INNER JOIN staff s ON d.hod_id = s.sid;
X c_dept%ROWTYPE;
BEGIN
sys.dbms_output.put_line('Department Number' || ' ' || 'Department Name' || ' ' || 'HOD Name');
FOR X IN c_dept LOOP
sys.dbms_output.put_line(X.dept_no || ' ' || X.dept_name || ' ' || X.hod_name);
```

```
END LOOP;

END;

/
```

3) Procedure to retrieve the student and ID details for a specific student:

```
CREATE OR REPLACE PROCEDURE prc_student_and_id_by_usn(p_usn IN VARCHAR) IS

X student%ROWTYPE;

X1 id%ROWTYPE;

CURSOR c_student IS

SELECT s.* FROM student s

WHERE s.usn = p_usn;

CURSOR c_id IS

SELECT i.* FROM id i

WHERE i.usn = p_usn;

BEGIN

FOR X IN c_student LOOP

DBMS_OUTPUT.PUT_LINE('Student details:');

DBMS_OUTPUT.PUT_LINE(X.usn || ' ' || X.fn || ' ' || X.ln || ' ' || X.yog || ' ' || X.sem || '
'||X.dept_no || ' ' || X.leader || ' ' || X.admis_type);

END LOOP;

FOR X1 IN c_id LOOP

DBMS_OUTPUT.PUT_LINE('ID details:');

DBMS_OUTPUT.PUT_LINE(X1.id_no ||' ' || X1.usn || ' ' || X1.fname|| ' ' || X1.mname||'
'||X1.mobnum||' '||X1.proctor_id||' '||X1.email);

END LOOP;

END;
```

/

4) A procedure to display the details of all sports staff and teachers:

```
create or replace procedure prc_sports_staff_and_teachers as
s_rec staff%rowtype;
t_rec teacher%rowtype;
cursor c_sports_staff is
select s.*
from staff s
where s.role = 'sports staff';
cursor c_teacher is
select t.*
from teacher t;
begin
sys.dbms_output.put_line('Sports staff details:');
for s_rec in c_sports_staff loop
sys.dbms_output.put_line(s_rec.sid || ' ' || s_rec.name || ' ' || s_rec.role);
end loop;
sys.dbms_output.put_line('Teacher details:');
for t_rec in c_teacher loop
sys.dbms_output.put_line(t_rec.tid || ' ' || t_rec.sid || ' ' || t_rec.dno || ' ' || t_rec.subid || ' '
||t_rec.exp);
end loop;
end;
/
```

Triggers

1) **Check if the student's year of joining is in the future**

```
CREATE OR REPLACE TRIGGER validate_student_insert
BEFORE INSERT ON student
FOR EACH ROW
DECLARE
cnt NUMBER;
BEGIN
IF :NEW.yog > sysdate THEN
RAISE_APPLICATION_ERROR(-20001, 'Year of joining must not be in the future');
END IF;
END;
/
```

2) **Before insert trigger to find maximum number of subject is limited to 5**

```
create or replace trigger max_subject
before insert on subject
for each row
declare
cnt number;
begin
select count(*) into cnt from subject;
if(cnt>4) then
raise_application_error(-20010,'MAX SUBJECT LIMIT REACHED');
end if;
end;
```

/

3) **Deletion trigger to not allow deleting students whose year of joining is less than 4 years ago**

```
CREATE OR REPLACE TRIGGER trg_prevent_delete_current_yog

BEFORE DELETE ON student

FOR EACH ROW

DECLARE

current_date DATE := SYSDATE;

graduation_year NUMBER;

BEGIN

graduation_year := TO_NUMBER(TO_CHAR(current_date, 'YYYY')) -
TO_NUMBER(TO_CHAR(:OLD.yog, 'YYYY'));

IF graduation_year < 4 THEN

   RAISE_APPLICATION_ERROR(-20999, 'Cannot delete current graduates');

 END IF;

END;

/
```

4) **Trigger to allow a student to participate in only one sport.**

```
CREATE OR REPLACE TRIGGER sport_participation

 BEFORE INSERT OR UPDATE ON sports

 FOR EACH ROW

DECLARE

 cnt NUMBER;

BEGIN
```

```
  SELECT COUNT(*)
  INTO cnt
  FROM sports
  WHERE usn = :new.usn;
  IF cnt >= 1 THEN
    RAISE_APPLICATION_ERROR(-20001, 'The student is already enrolled in a sport');
  END IF;
END;
/
```

Snapshots

DDL

```
SQL> create table staff(
  2  sid int primary key,
  3  name varchar(10),
  4  role varchar(15)
  5  );

Table created.

SQL> create table department(
  2  dept_no int primary key,
  3  name varchar(10),
  4  hod_id int references staff(sid)
  5  );

Table created.

SQL> create table hostel(
  2  hname varchar(10) primary key,
  3  gender varchar(10),
  4  managerid int references staff(sid)
  5  );

Table created.

SQL> create table subject(
  2  subid int primary key,
  3  sub_name varchar(10),
  4  max_credit int
  5  );
```

```
SQL> create table sports(
  2  name varchar(10) ,
  3  coach_id int references staff(sid),
  4  usn varchar(10) references student(usn),
  5  primary key(name,usn)
  6  );

Table created.

SQL> create table id(
  2  id_no int primary key,
  3  usn varchar(10) references student(usn),
  4  fname varchar(10),
  5  mname varchar(10),
  6  proctor_id int references staff(sid),
  7  mobnum number(10),
  8  email varchar(25)
  9  );

Table created.

SQL> create table student_hostel
  2  (
  3    usn varchar(10) references student(usn),
  4    hname varchar(10) references hostel(hname),
  5   primary key(usn,hname)
  6  );

Table created.
```

```
SQL> create table stu_address
  2  (
  3  id_no int references id(id_no),
  4  street varchar(15),
  5  city varchar(15),
  6  country varchar(15),
  7  house_name varchar(15),
  8  primary key(id_no,house_name,street,city,country)
  9  );

Table created.

SQL> create table hostel_address
  2  (
  3  hname varchar(10) references hostel(hname),
  4  street varchar(15),
  5  city varchar(15),
  6  country varchar(15),
  7  hostel_name varchar(15),
  8  primary key(hname,hostel_name,street,city,country)
  9  );

Table created.

SQL> CREATE TABLE dummy_table (
  2  id NUMBER primary key,
  3  name VARCHAR2(50),
  4  date_created DATE
  5  );

Table created.

SQL>
SQL> ALTER TABLE dummy_table
  2  ADD column_name VARCHAR2(50);

Table altered.
```

```
SQL> DROP TABLE dummy_table;

Table dropped.

SQL> CREATE VIEW staff_dept AS
  2  SELECT s.sid, s.name, d.name as department
  3  FROM staff s,department d
  4  where s.sid = d.hod_id;

View created.

SQL> CREATE VIEW student_dept_admis AS
  2  SELECT s.usn, s.fn, s.ln, d.name as department, a.type as admission_type
  3  FROM student s,department d,admission a
  4  where s.dept_no = d.dept_no
  5  and s.admis_type = a.type;

View created.
```

```
SQL> create table hostel_address
  2  (
  3  hname varchar(10) references hostel(hname),
  4  street varchar(15),
  5  city varchar(15),
  6  country varchar(15),
  7  primary key(hname,street,city,country)
  8  );

Table created.
```

Insert

```
SQL> insert into staff values (1, 'John', 'Principal');

1 row created.

SQL> insert into staff values (2, 'Jane', 'Manager');

1 row created.

SQL> insert into staff values (3, 'Bob', 'Teacher');

1 row created.

SQL> insert into staff values (4, 'Sarah', 'FOOTBALL Coach');

1 row created.

SQL> insert into staff values (18, 'barah', 'Coco Coach');

1 row created.

SQL> insert into staff values (19, 'marah', 'Cricket Coach');

1 row created.

SQL> insert into staff values (5, 'avk', 'proctor');

1 row created.

SQL> insert into staff values (6, 'srinivas', 'proctor');

1 row created.

SQL> insert into staff values (7, 'ysn', 'proctor');

1 row created.

SQL> insert into staff values (8, 'avk', 'proctor');

1 row created.
```

```
SQL> insert into staff values (6, 'srinivas', 'proctor');

1 row created.

SQL> insert into staff values (7, 'ysn', 'proctor');

1 row created.

SQL> insert into staff values (8, 'avk', 'proctor');

1 row created.

SQL> insert into staff values (9, 'ram', 'Teacher');

1 row created.

SQL> insert into staff values (10, 'sham', 'Teacher');

1 row created.

SQL> insert into staff values (11, 'bhem', 'Teacher');

1 row created.

SQL> insert into staff values (12, 'rao', 'Teacher');

1 row created.

SQL> insert into staff values (13, 'poornima', 'HOD');

1 row created.
```

```
SQL> insert into staff values (14, 'shagun', 'HOD');

1 row created.

SQL> insert into staff values (15, 'shamraj', 'HOD');

1 row created.

SQL> insert into staff values (16, 'bhemraj', 'HOD');

1 row created.

SQL> insert into staff values (17, 'kulkarni', 'HOD');

1 row created.
```

```
SQL> insert into department values (101, 'Science', 13);

1 row created.

SQL> insert into department values (102, 'Arts', 14);

1 row created.

SQL> insert into department values (103, 'Sports', 15);

1 row created.

SQL> INSERT INTO department VALUES (104, 'Commerce', 16);

1 row created.

SQL> INSERT INTO department VALUES (105, 'Medical', 17);

1 row created.
```

```
SQL> insert into hostel values ('Basava', 'Male', 2);

1 row created.

SQL> insert into hostel values ('Akkamaha', 'Female', 3);

1 row created.
```

```
SQL> insert into subject values (1002, 'History', 3);

1 row created.

SQL> insert into subject values (1003, 'English', 2);

1 row created.

SQL> insert into subject values (1004, 'Physics', 4);

1 row created.

SQL> insert into subject values (1005, 'Biology', 3);

1 row created.

SQL> insert into subject values (1006, 'ZOology', 3);

1 row created.
```

```
SQL> insert into teacher values (2, 10, 102, 1002, 5);

1 row created.

SQL> insert into teacher values (3, 11, 103, 1003, 3);

1 row created.

SQL> insert into teacher values (4, 12, 103, 1003, 13);

1 row created.

SQL> insert into teacher values (1, 3, 101, 1004, 10);

1 row created.
```

```
SQL> insert into admission values ('COMEDK', 100000);

1 row created.

SQL> insert into admission values ('KCET', 50000);

1 row created.

SQL> insert into admission values ('Management', 150000);

1 row created.
```

```
SQL> insert into student values ('1MS19CS001', 'John', 'Doe', '2019-01-01', 3, 101, NULL, 'COMEDK');

1 row created.

SQL> insert into student values ('1MS18CS002', 'Jane', 'Doe', '2020-01-01', 2, 102, '1MS19CS001', 'KCET');

1 row created.

SQL> insert into student values ('1MS18CS003', 'Jane', 'Doe', '2020-01-01', 2, 102, '1MS19CS001', 'KCET');

1 row created.

SQL> insert into student values ('1MS18CS004', 'Sarah', 'Johnson', '2020-01-01', 2, 102, '1MS18CS002', 'Management');

1 row created.

SQL> insert into student values ('1MS17CS005', 'Joseph', 'Williams', '2020-01-01', 1, 104, '1MS19CS001', 'KCET');

1 row created.

SQL> insert into student values ('1MS19CS006', 'abhiram', 'Williams', '2019-01-01', 3, 105, '1MS18CS004', 'COMEDK');

1 row created.

SQL> insert into student values ('1MS19CS005', 'John', 'Doe', '2023-01-01', 3, 101, '1MS18CS004', 'COMEDK');

1 row created.
```

```
SQL> insert into sports values ('Football', 4, '1MS19CS001');

1 row created.

SQL> insert into sports values ('Coco',18, '1MS18CS002');

1 row created.

SQL> insert into sports values ('Cricket', 19, '1MS18CS003');

1 row created.

SQL> insert into sports values ('Football', 4, '1MS18CS004');

1 row created.

SQL> insert into sports values ('Coco',18, '1MS17CS005');

1 row created.

SQL> insert into sports values ('Cricket', 19, '1MS19CS006');

1 row created.
```

```
SQL> insert into id values (1, '1MS19CS001', 'John', 'Doe', 5, 1234567890, 'johndoe@gmail.com');

1 row created.

SQL> insert into id values (2, '1MS18CS002', 'Jane', 'Doe', 5, 2345678901, 'janedoe@gmail.com');

1 row created.

SQL> insert into id values (3, '1MS18CS003', 'Bob', 'Smith', 6, 3456789012, 'bobsmith@gmail.com');

1 row created.

SQL> insert into id values (4, '1MS18CS004', 'Sarah', 'Johnson', 7, 4567890123, 'sarahjohnson@gmail');

1 row created.

SQL> insert into id values (5, '1MS17CS005', 'Joseph', 'williams', 8, 3456789012, 'bobsmith@gmail.com');

1 row created.

SQL> insert into id values (6, '1MS19CS006', 'abhiram', 'williams', 7, 4567890123, 'sarahjohnson@gmail.com');

1 row created.
```

```
SQL> INSERT INTO student_hostel (usn, hname)
  2  VALUES ('1MS19CS001', 'Basava');

1 row created.

SQL>
SQL> INSERT INTO student_hostel (usn, hname)
  2  VALUES ('1MS18CS002', 'Basava');

1 row created.

SQL>
SQL> INSERT INTO student_hostel (usn, hname)
  2  VALUES ('1MS19CS006', 'Akkamaha');

1 row created.
```

```
SQL> INSERT INTO stu_address
  2  VALUES (1, '123 Main St', 'Seattle', 'Washington', 'Akshay');

1 row created.

SQL>
SQL> INSERT INTO stu_address
  2  VALUES (2, '12 Main St', 'New York', 'NY', 'Lakshmi');

1 row created.

SQL>
SQL> INSERT INTO stu_address
  2  VALUES (3, '20 Oxford St', 'London', 'England', 'kshatriya');

1 row created.

SQL>
SQL> INSERT INTO stu_address
  2  VALUES (4, '456 Oak Ave', 'Portland', 'Oregon', 'Mallikarjuna');

1 row created.

SQL>
SQL> INSERT INTO stu_address
  2  VALUES (5, '789 Maple Blvd', 'San Francisco', 'California', 'shivashankar');

1 row created.

SQL>
SQL> INSERT INTO stu_address
  2  VALUES (6, '111 Pine St', 'Los Angeles', 'California', 'shivakrupa');

1 row created.
```

```
SQL> INSERT INTO hostel_address (hname, street, city, country) VALUES ('Basava','456 Park Ave', 'London', 'United Kingdom');

1 row created.

SQL> INSERT INTO hostel_address (hname, street, city, country) VALUES ('Akkamaha', '789 Ocean Dr', 'Miami', 'United States');

1 row created.
```

Queries

```
SQL> select * from student;

USN        FN         LN         YOG              SEM    DEPT_NO LEADER
---------- ---------- ---------- ---------- ---------- ---------- ----------ADMIS_TYPE
----------
1MS19CS001 John       Doe        2019-01-01          3        101 1MS18CS004COMEDK

1MS18CS002 Jane       Doe        2020-01-01          2        102 1MS19CS001KCET

1MS18CS003 Jane       Doe        2020-01-01          2        102 1MS19CS001KCET


USN        FN         LN         YOG              SEM    DEPT_NO LEADER
---------- ---------- ---------- ---------- ---------- ---------- ----------ADMIS_TYPE
----------
1MS18CS004 Sarah      Johnson    2020-01-01          2        102 1MS18CS002Management

1MS17CS005 Joseph     Williams   2020-01-01          1        104 1MS19CS001KCET

1MS19CS006 abhiram    Williams   2019-01-01          3        105 1MS18CS004COMEDK


USN        FN         LN         YOG              SEM    DEPT_NO LEADER
---------- ---------- ---------- ---------- ---------- ---------- ----------ADMIS_TYPE
----------
1MS19CS005 John       Doe        2023-01-01          3        101 1MS18CS004COMEDK


7 rows selected.
```

```
SQL> select * from id where usn = '1MS18CS003';

     ID_NO USN        FNAME      MNAME      PROCTOR_ID     MOBNUM
---------- ---------- ---------- ---------- ---------- ----------
EMAIL
-------------------------
         3 1MS18CS003 Bob        Smith               6 3456789012
bobsmith@gmail.com


SQL>
SQL> SELECT id_no, fname, mname, email FROM id WHERE proctor_id = 7;

     ID_NO FNAME      MNAME      EMAIL
---------- ---------- ---------- -------------------------
         4 Sarah      Johnson    sarahjohnson@gmail
         6 abhiram    williams   sarahjohnson@gmail.com
```

```
SQL> SELECT * FROM student WHERE yog < '2022-01-01';

USN        FN         LN         YOG                SEM    DEPT_NO LEADER
---------- ---------- ---------- ---------- ---------- ---------- ----------ADMIS_TYPE
----------
1MS19CS001 John       Doe        2019-01-01          3        101 1MS18CS004COMEDK

1MS18CS002 Jane       Doe        2020-01-01          2        102 1MS19CS001KCET

1MS18CS003 Jane       Doe        2020-01-01          2        102 1MS19CS001KCET


USN        FN         LN         YOG                SEM    DEPT_NO LEADER
---------- ---------- ---------- ---------- ---------- ---------- ----------ADMIS_TYPE
----------
1MS18CS004 Sarah      Johnson    2020-01-01          2        102 1MS18CS002Management

1MS17CS005 Joseph     Williams   2020-01-01          1        104 1MS19CS001KCET

1MS19CS006 abhiram    Williams   2019-01-01          3        105 1MS18CS004COMEDK
```

```
SQL> SELECT AVG(exp) avg_of_exp FROM teacher;

AVG_OF_EXP
----------
      7.75
```

```
SQL> SELECT AVG(max_credit) FROM subject;

AVG(MAX_CREDIT)
---------------
            3.2

SQL>
SQL> SELECT MIN(fees) FROM admission;

 MIN(FEES)
----------
     50000

SQL>
SQL> SELECT MAX(fees) FROM admission;

 MAX(FEES)
----------
    150000

SQL>
SQL> SELECT MAX(max_credit) FROM subject;

MAX(MAX_CREDIT)
---------------
              4

SQL>
SQL> SELECT SUM(max_credit) FROM subject;

SUM(MAX_CREDIT)
---------------
             16
```

```
SQL> SELECT COUNT(DISTINCT proctor_id)  FROM id;

COUNT(DISTINCTPROCTOR_ID)
-------------------------
                        4

SQL> SELECT COUNT(*) total_Staff FROM staff;

TOTAL_STAFF
-----------
         19
```

```
SQL> SELECT DISTINCT sub_name
  2  FROM subject;

SUB_NAME
----------
Physics
Maths
English
Biology
History
```

```
SQL> SELECT *
  2  FROM student
  3  WHERE fn like 'a%';

USN        FN         LN         YOG                SEM    DEPT_NO LEADER
---------- ---------- ---------- ---------- ---------- ---------- ----------
ADMIS_TYPE
----------
1MS19CS006 abhiram    Williams   2019-01-01          3        105 1MS18CS004
COMEDK


SQL> SELECT department.name, COUNT(student.usn)
  2  FROM student
  3  JOIN department ON student.dept_no = department.dept_no
  4  GROUP BY department.name;

NAME       COUNT(STUDENT.USN)
---------- ------------------
Commerce                    1
Science                     2
Medical                     1
Arts                        3
```

```
SQL> SELECT * FROM student
  2  ORDER BY yog DESC;

USN        FN         LN         YOG                SEM    DEPT_NO LEADER
---------- ---------- ---------- ---------- ---------- ---------- ----------
ADMIS_TYPE
----------
1MS19CS005 John       Doe        2023-01-01          3        101 1MS18CS004
COMEDK

1MS18CS003 Jane       Doe        2020-01-01          2        102 1MS19CS001
KCET

1MS18CS002 Jane       Doe        2020-01-01          2        102 1MS19CS001
KCET


USN        FN         LN         YOG                SEM    DEPT_NO LEADER
---------- ---------- ---------- ---------- ---------- ---------- ----------
ADMIS_TYPE
----------
1MS18CS004 Sarah      Johnson    2020-01-01          2        102 1MS18CS002
Management

1MS17CS005 Joseph     Williams   2020-01-01          1        104 1MS19CS001
KCET

1MS19CS006 abhiram    Williams   2019-01-01          3        105 1MS18CS004
COMEDK


USN        FN         LN         YOG                SEM    DEPT_NO LEADER
---------- ---------- ---------- ---------- ---------- ---------- ----------
ADMIS_TYPE
----------
1MS19CS001 John       Doe        2019-01-01          3        101 1MS18CS004
COMEDK


7 rows selected.
```

```
SQL> SELECT * FROM teacher
  2  ORDER BY exp DESC, tid ASC;

       TID        SID        DNO      SUBID        EXP
---------- ---------- ---------- ---------- ----------
         4         12        103       1003         13
         1          3        101       1001         10
         2         10        102       1002          5
         3         11        103       1003          3

SQL> SELECT fn, ln
  2  FROM student
  3  WHERE usn IN (SELECT usn FROM student_hostel WHERE hname = 'Akkamaha');

FN         LN
---------- ----------
abhiram    Williams

SQL> SELECT d.name as department_name, s.name as hod_name
  2  FROM department d ,staff s
  3  where d.hod_id = s.sid;

DEPARTMENT HOD_NAME
---------- ----------
Science    poornima
Arts       shagun
Sports     shamraj
Commerce   bhemraj
Medical    kulkarni
```

```
SQL> SELECT * FROM department
  2  WHERE EXISTS (SELECT * FROM teacher
  3                WHERE dno = department.dept_no
  4                  AND exp > 10);

   DEPT_NO NAME           HOD_ID
---------- ---------- ----------
       103 Sports             15
```

```
SQL> UPDATE admission
  2  SET fees = fees + 1000
  3  WHERE type = 'Management';

1 row updated.
```

Procedures

```
SQL> CREATE OR REPLACE PROCEDURE display_students_in_hostel (hostel_name varchar)
  2  IS
  3  student_rec student%rowtype;
  4  cursor c is select s.* from student s, student_hostel sh
  5  where sh.hname = hostel_name and s.usn = sh.usn;
  6  begin
  7  for student_rec in c loop
  8  sys.dbms_output.put_line(student_rec.usn||' '||student_rec.fn||' '||student_rec.ln);
  9  end loop;
 10  end;
 11  /

Procedure created.
```

```
SQL> set serveroutput on;
SQL> exec display_students_in_hostel('Basava');
1MS19CS001 John Doe
1MS18CS002 Jane Doe

PL/SQL procedure successfully completed.
```

```
SQL> CREATE OR REPLACE PROCEDURE prc_display_hod_and_dept IS
  2      CURSOR c_dept IS
  3          SELECT d.dept_no, d.name as dept_name, s.name as hod_name
  4          FROM department d
  5          INNER JOIN staff s ON d.hod_id = s.sid;
  6      X c_dept%ROWTYPE;
  7  BEGIN
  8      sys.dbms_output.put_line('Department Number' || ' ' || 'Department Name' || ' ' || 'HOD Name');
  9      FOR X IN c_dept LOOP
 10          sys.dbms_output.put_line(X.dept_no || ' ' || X.dept_name || ' ' || X.hod_name);
 11      END LOOP;
 12  END;
 13  /

Procedure created.

SQL> exec  prc_display_hod_and_dept;
Department Number Department Name HOD Name
101 Science poornima
102 Arts shagun
103 Sports shamraj
104 Commerce bhemraj
105 Medical kulkarni

PL/SQL procedure successfully completed.
```

```
SQL> create or replace procedure prc_sports_staff_and_teachers as
  2    s_rec staff%rowtype;
  3    t_rec teacher%rowtype;
  4    cursor c_sports_staff is
  5      select s.*
  6      from staff s
  7      where s.role = 'sports staff';
  8    cursor c_teacher is
  9      select t.*
 10      from teacher t;
 11  begin
 12    sys.dbms_output.put_line('Sports staff details:');
 13    for s_rec in c_sports_staff loop
 14      sys.dbms_output.put_line(s_rec.sid || ' ' || s_rec.name || ' ' || s_rec.role);
 15    end loop;
 16
 17    sys.dbms_output.put_line('Teacher details:');
 18    for t_rec in c_teacher loop
 19      sys.dbms_output.put_line(t_rec.tid || ' ' || t_rec.sid || ' ' || t_rec.dno || ' ' || t_rec.subid || ' ' || t_rec.exp);
 20    end loop;
 21  end;
 22  /

Procedure created.

SQL>
SQL>
SQL>
SQL> exec  prc_sports_staff_and_teachers;
Sports staff details:
Teacher details:
2 10 102 1002 5
3 11 103 1003 3
4 12 103 1003 13
1 3 101 1004 10

PL/SQL procedure successfully completed.
```

```
SQL> CREATE OR REPLACE PROCEDURE prc_student_and_id_by_usn(p_usn IN VARCHAR) IS
  2      X student%ROWTYPE;
  3      X1 id%ROWTYPE;
  4      CURSOR c_student IS
  5          SELECT s.* FROM student s
  6          WHERE s.usn = p_usn;
  7      CURSOR c_id IS
  8          SELECT i.* FROM id i
  9          WHERE i.usn = p_usn;
 10  BEGIN
 11      FOR X IN c_student LOOP
 12          DBMS_OUTPUT.PUT_LINE('Student details:');
 13          DBMS_OUTPUT.PUT_LINE(X.usn || ' ' || X.fn || ' ' || X.ln || ' ' || X.yog || ' ' || X.sem || ' ' || X.dept_no || ' ' || X.leader || ' ' || X.adm
is_type);
 14      END LOOP;
 15      FOR X1 IN c_id LOOP
 16          DBMS_OUTPUT.PUT_LINE('ID details:');
 17          DBMS_OUTPUT.PUT_LINE(X1.id_no ||' ' || X1.usn || ' ' || X1.fname|| ' ' || X1.mname||' '||X1.mobnum||' '||X1.proctor_id||' '||X1.email);
 18      END LOOP;
 19  END;
 20  /

Procedure created.

SQL> set serveroutput on;
SQL> exec prc_student_and_id_by_usn('1MS19CS001');
Student details:
1MS19CS001 John Doe 2019-01-01 3 101  COMEDK
ID details:
1 1MS19CS001 John Doe 1234567890 5 johndoe@gmail.com

PL/SQL procedure successfully completed.
```

Triggers

```
SQL> CREATE OR REPLACE TRIGGER validate_student_insert
  2  BEFORE INSERT ON student
  3  FOR EACH ROW
  4  DECLARE
  5    cnt NUMBER;
  6  BEGIN
  7    -- check if the student's year of joining is in the future
  8    IF :NEW.yog > sysdate THEN
  9      RAISE_APPLICATION_ERROR(-20001, 'Year of joining must not be in the future');
 10    END IF;
 11
 12  END;
 13  /

Trigger created.

SQL> create or replace trigger max_subject
  2  before insert on subject
  3  for each row
  4  declare
  5  cnt number;
  6  begin
  7  select count(*) into cnt from subject;
  8  if(cnt>4) then
  9  raise_application_error(-20010,'MAX SUBJECT LIMIT REACHED');
 10  end if;
 11  end;
 12  /

Trigger created.
```

```
SQL> CREATE OR REPLACE TRIGGER trg_prevent_delete_current_yog
  2  BEFORE DELETE ON student
  3  FOR EACH ROW
  4  DECLARE
  5    current_date DATE := SYSDATE;
  6    graduation_year NUMBER;
  7  BEGIN
  8    graduation_year := TO_NUMBER(TO_CHAR(current_date, 'YYYY')) - TO_NUMBER(TO_CHAR(:OLD.yog, 'YYYY'));
  9    IF graduation_year < 4 THEN
 10      RAISE_APPLICATION_ERROR(-20999, 'Cannot delete current graduates');
 11    END IF;
 12  END;
 13  /

Trigger created.

SQL>
SQL> CREATE OR REPLACE TRIGGER sport_participation
  2    BEFORE INSERT OR UPDATE ON sports
  3    FOR EACH ROW
  4  DECLARE
  5    cnt NUMBER;
  6  BEGIN
  7    SELECT COUNT(*)
  8    INTO cnt
  9    FROM sports
 10    WHERE usn = :new.usn;
 11
 12    IF cnt >= 1 THEN
 13      RAISE_APPLICATION_ERROR(-20001, 'The student is already enrolled in a sport');
 14    END IF;
 15  END;
 16  /

Trigger created.
```

```
SQL> insert into student values ('1MS19CS005', 'John', 'Doe', '2025-01-01', 3, 101, '1MS18CS004', 'COMEDK');
insert into student values ('1MS19CS005', 'John', 'Doe', '2025-01-01', 3, 101, '1MS18CS004', 'COMEDK')
            *
ERROR at line 1:
ORA-20001: Year of joining must not be in the future
ORA-06512: at "BHARATHKS.VALIDATE_STUDENT_INSERT", line 6
ORA-04088: error during execution of trigger
'BHARATHKS.VALIDATE_STUDENT_INSERT'
```

```
SQL> insert into subject values (1001, 'Maths', 4);
insert into subject values (1001, 'Maths', 4)
               *
ERROR at line 1:
ORA-20010: MAX SUBJECT LIMIT REACHED
ORA-06512: at "BHARATHKS.MAX_SUBJECT", line 6
ORA-04088: error during execution of trigger 'BHARATHKS.MAX_SUBJECT'
```

```
SQL> insert into student values ('1MS19CS025', 'John', 'Doe', '2015-01-01', 3, 101, '1MS18CS004', 'COMEDK');

1 row created.

SQL> delete from student where usn='1MS19CS005';
delete from student where usn='1MS19CS005'
            *
ERROR at line 1:
ORA-20999: Cannot delete current graduates
ORA-06512: at "BHARATHKS.TRG_PREVENT_DELETE_CURRENT_YOG", line 7
ORA-04088: error during execution of trigger
'BHARATHKS.TRG_PREVENT_DELETE_CURRENT_YOG'
```

```
SQL> insert into sports values ('Football', 4, '1MS19CS006');
insert into sports values ('Football', 4, '1MS19CS006')
            *
ERROR at line 1:
ORA-20001: The student is already enrolled in a sport
ORA-06512: at "BHARATHKS.SPORT_PARTICIPATION", line 10
ORA-04088: error during execution of trigger 'BHARATHKS.SPORT_PARTICIPATION'
```