

Indian Institute of Information Technology Vadodara

CS262: Database Management System

Lab 6

Roll No. 201951134

Name: Sameer Anand

Question 1.) Create a Table as employee and the details are

S.No Name Designation Branch

1 Ram Manager Chennai

2 Santhosh Supervisor Madurai

3 Hari Assistant Trichy

create table Employee(

sno int,

name varchar(20),

designation varchar(20),

branch varchar(20),

primary key(sno)

);

insert into Employee values(1 , 'Ram', 'Manager', 'Chennai'); insert

into Employee values(2 , 'Santosh', 'Supervisor', 'Madurai');

insert into Employee values(3 , 'Hari', 'Assistant', 'Trichy'); select

*from Employee;

	sno	name	designation	branch
▶	1	Ram	Manager	Chennai
	2	Santosh	Supervisor	Madurai
	3	Hari	Assistant	Trichy
•	NULL	NULL	NULL	NULL

Perform the following:

Alter the table by adding a column Salary

1

ALTER table Employee

Add salary int ;

Update Employee

Set salary = 100000

where sno =1;

Update Employee

Set salary = 25000

where sno =2;

Update Employee

Set salary = 35000

where sno =3;

select *from employee;

	sno	name	designation	branch	salary
▶	1	Ram	Manager	Chennai	100000
	2	Santosh	Supervisor	Madurai	25000
	3	Hari	Assistant	Trichy	35000
•	NULL	NULL	NULL	NULL	NULL

Alter the table by modifying the column Name Describe the table employee as employeeName

Alter table Employee

change name employee_name char(20);

select *from employee;

	sno	employee_name	designation	branch	salary
▶	1	Ram	Manager	Chennai	100000
	2	Santosh	Supervisor	Madurai	25000
	3	Hari	Assistant	Trichy	35000
•	NULL	NULL	NULL	NULL	NULL

Copy the table employee as emp

CREATE TABLE emp AS

SELECT *

FROM Employee;

select *from emp;

	sno	employee_name	designation	branch	salary
▶	1	Ram	Manager	Chennai	100000
	2	Santosh	Supervisor	Madurai	25000
	3	Hari	Assistant	Trichy	35000
•	NULL	NULL	NULL	NULL	NULL

Truncate the table

truncate table employee;

select *from employee;

	sno	employee_name	designation	branch	salary
•	NULL	NULL	NULL	NULL	NULL

Delete the Second row from the table

delete from employee where sno=2;

select *from employee;

	sno	employee_name	designation	branch	salary
•	NULL	NULL	NULL	NULL	NULL

Drop the table
drop table Employee;

Question 2.) Consider the following relational schema for the Office of the Controller of Examinations Application.

Student (Rollno, Name, Dob, Gender, Doa, Bcode);

Implement a check constraint for

- Gender

Branch (Bcode, Bname, Dno);

Department (Dno, Dname);

Course (Ccode, Cname, Credits, Dno);

Branch_Course (Bcode, Ccode, Semester);

Enrolls (Rollno, Ccode, Sess, Grade);

Implement a check constraint for grade Value Set ('S', 'A', 'B', 'C', 'D', 'E', 'U');

Students are admitted to Branches and they are offered by Departments. A branch is offered by only one department.

Each branch has a set of Courses (Subjects). Each student must enroll during a semester. Courses are offered by Departments. A course is offered only by one department. If a student is unsuccessful in a course he/she must enroll for the course during next session. A student has successfully completed a course if the grade obtained by is from the list (A, B, C, D, and E).

A student is unsuccessful if he/she have grade 'U' in a course.

Primary Keys are underlined.

create database Exam;

use Exam;

create table Student(

id int ,

name varchar(20),

dob date,

doa date,

```
bcode int,  
gender varchar(10),  
check (gender in('Male','Female')),  
primary key(id)  
);
```

```
create table Branch(  
bcode int primary key,  
bname varchar(20),  
dno int  
);
```

```
create table Department(  
dno int ,  
dname varchar(20),  
primary key(dno)  
);
```

```
create table Course(  
Ccode int,  
Cname varchar(20),  
credits int,  
dno int,  
primary key(Ccode)  
);
```

```
create table  
Branch_Course( bcode int,  
Ccode int ,  
Semester int,
```

```
primary key(bcode,Ccode)
);
```

```
create table Enroll(
id int ,
Ccode int,
Sess varchar(20),
Grade varchar(3),
check(Grade in('S', 'A', 'B', 'C', 'D', 'E', 'U')),
primary key(id,Ccode,Sess)
);
```

```
insert into Student values(001,'Sameer','2000-11-29','2020-07-24','100','Male');
```

```
insert into Student values(002,'Vedant','2001-12-20','2020-07-30','100','Male');
```

```
insert into Student values(003,'lishaan','2001-06-24','2020-07-23','101','Male');
```

```
insert into Student values(004,'Nishant','2001-02-27','2019-11-30','102','Male');
```

```
insert into Student values(005,'Anushka','2000-10-02','2020-07-27','103','Female');
```

```
insert into Student values(006,'Subhanjali','2000-03-21','2020-07-26','104','Female');
```

```
insert into Student values(007,'Anvya','2000-10-28','2020-07-25','105','Male');
```

```
insert into Student values(008,'Anirudh','2000-10-28','2020-07-25','105','Male');
```

```
insert into Branch values(100,'Computer Science',1001);
insert into Branch values(101,'Electronics',1002);
insert into Branch values(102,'Mechanical',1003);
insert into Branch values(103,'Biology',1004);
insert into Branch values(104,'BA',1005);
insert into Branch values(105,'Metallurgy',1006);
insert into Branch values(106,'Electrical',1007);
```

```
insert into Department values(51,'IT');
insert into Department values(52,'Medical');
insert into Department values(53,'Arts');
```

```
insert into Course values(201,'Discrete Maths',3,51);
```

```
insert into Course values(202,'Data Structure',5,51);
```

```
insert into Course values(203,'Algorithms',5,51);
```

```
insert into Course values(204,'Economics',3,52);
```

```
insert into Course values(205,'Technical Writing',2,53);
```

```
insert into Course values(206,'Networks',4,52);
```

```
insert into course values(207,'Probability',4,51);
```

```
insert into Branch_Course
values(101,201,2); insert into
Branch_Course values(102,201,2); insert
into Branch_Course values(103,202,3);
insert into Branch_Course
values(105,203,3); insert into
Branch_Course values(100,204,4); insert
into Branch_Course values(104,201,3);
```

```
insert into Branch_Course
values(106,201,4);
```

```
insert into Enroll values(1,201,'2020-19','A');
```

```
insert into Enroll values(2,202,'2019-20','B'); insert into Enroll
values(3,203,'2019-20','C'); insert into Enroll values(4,204,'2019-20','D'); insert
into Enroll values(5,204,'2020-19','S'); insert into Enroll values(5,205,'2020-19','S'); insert into Enroll
values(6,206,'2020-19','S'); insert into Enroll values(6,207,'2020-19','S'); insert
into Enroll values(5,201,'2020-19','S'); insert into Enroll values(6,202,'2019-20','S');
insert into Enroll values(7,203,'2019-20','C');
insert into Enroll values(8,204,'2019-20','D');
```

Develop a SQL query to list details of Departments that offer more than 3 branches.

```
with dno_no as (select dno,count(dno)as number from course group by
dno) select * from department where dno in(select dno from dno_no where
number>3);
```

dno	dname
51	IT

Develop a SQL query to list the details of Departments that offer more than 6 courses.

dno	dname
-----	-------

Develop a SQL query to list the details of courses that are common for more than 3 branches.

```
with Ccode_no as (select Ccode,count(Ccode) as number from
Branch_course group by Ccode) select * from course where Ccode
in(select Ccode from Ccode_no where number>3);
```

Ccode	Cname	credits	dno
201	Discrete Maths	3	51

Develop a SQL query to list students who got 'S' in more than 2 courses during single enrollment.

```
Select * FROM Student as S WHERE S.Id IN (SELECT E.Id FROM Enroll as E WHERE E.grade = 'S' GROUP BY E.Id HAVING count(E.grade) > 2);
```

Create a view that will keep track of the roll number, name and number of courses, a student has completed successfully.

```
Create view Student_Data as SELECT E.Id, S.name, count(E.Ccode) FROM Student as S, Enroll as E WHERE E.Id = S.Id AND E.grade <>'U' GROUP BY E.Id, S.name;
```

Question 3.) Consider the following relations for an Order Processing Database application in a Company.

Customer (Customerno varchar2 (5), Cname varchar2 (50));

Implement check constraints to check Customerno starts with 'C'.

Cust_Order (Orderno varchar2(5), Odate Date, Customerno references Customer, Ord_amt number(8));

Implement check constraints to check Orderno starts with 'O'.

Ord_amt is derived attribute (default value is 0);

Item (Itemno varchar2 (5), Item_name varchar2 (30), unit_price number (5));

Implement check constraint to check Itemno starts with 'I'.

Order_item (Orderno references Cust_order, Itemno references item, qty number (3));

Develop DDL to implement above schema enforcing primary key, check constraints and foreign key constraints.

```
create database order_process ;
```

```
CREATE TABLE Customer(
```

```
    Customerno varchar(5) PRIMARY KEY,
```

```
    Cname varchar(25),
```

```
    CHECK (Customerno like 'C%')
```

```
);
```

```
CREATE TABLE Cust_Order(
```

```
    Orderno varchar(5) PRIMARY KEY,
```

```
    Odate date,
```

```
    Customerno varchar(5),
```

```
    Ord_amt int,
```



```
FOREIGN KEY (Customerno) REFERENCES Customer(Customerno),  
CHECK(Orderno LIKE 'O%')  
);
```

```
CREATE TABLE Item(  
    Itemno varchar(5) PRIMARY KEY,  
    Item_name varchar(30),  
    unit_price int,  
CHECK (Itemno LIKE 'I%')  
);
```

```
CREATE TABLE Order_item(  
    Orderno varchar(5) PRIMARY KEY,  
    Itemno varchar(5),  
    qty int,  
    FOREIGN KEY (Orderno) REFERENCES  
    Cust_Order(Orderno), FOREIGN KEY (Itemno)  
    REFERENCES Item(Itemno)  
);
```

```
INSERT INTO Customer VALUES('C0001','Sudhanshu  
pandey'); INSERT INTO Customer VALUES('C0003','ARUN  
SHARMA'); INSERT INTO Customer  
VALUES('C0005','suman YADAV'); INSERT INTO Customer  
VALUES('C0007','MANAS KUMAR'); INSERT INTO  
Customer VALUES('C0009','kishan TAILOR');
```

```
INSERT INTO Cust_Order values('O01', '2019-05-12',  
'C0003',1180); INSERT INTO Cust_Order values('O02',  
'2020-01-11', 'C0005',200); INSERT INTO Cust_Order  
values('O03', '2019-12-29', 'C0001',900);  
INSERT INTO Cust_Order values('O04', '2020-02-17',  
'C0007',1200); INSERT INTO Cust_Order values('O05', '2020-  
02-17', 'C0001',1200);
```

```
INSERT INTO Cust_Order values('O06', '2020-02-17',  
'C0001',1200); INSERT INTO Cust_Order values('O07', '2020-  
02-17', 'C0001',1200);
```

```
INSERT INTO Item values('I002',"FAN",100);  
INSERT INTO Item values('I001',"Biscuit",2000);  
INSERT INTO Item values('I003',"TABLE",100);  
INSERT INTO Item values('I004',"COLE",50);  
INSERT INTO Item values('I005',"CHAIR",200);
```

```
INSERT INTO Order_item VALUES('O01', 'I002', 4);  
INSERT INTO Order_item VALUES('O02', 'I001', 3);  
INSERT INTO Order_item VALUES('O03', 'I005', 2);  
INSERT INTO Order_item VALUES('O04', 'I001', 5);  
INSERT INTO Order_item VALUES('O05', 'I004', 5);  
INSERT INTO Order_item VALUES('O06', 'I001', 6);  
INSERT INTO Order_item VALUES('O07', 'I005', 5);
```

Develop SQL query to list the details of customers who have placed more than 3 orders.

```
SELECT * FROM Customer as C WHERE C.Customerno IN  
(SELECT O.Customerno FROM  
Cust_Order as O GROUP BY O.Customerno HAVING  
count(O.Orderno) > 3);
```

Develop a SQL query to list details of items whose price is less than the average price of all items in each order.

```
SELECT * FROM Item as I WHERE I.unit_price < (SELECT  
avg(unit_price) FROM Item);
```

	Itemno	Item_name	unit_price
▶	I002	FAN	100
	I003	TABLE	100
	I004	COLE	50
	I005	CHAIR	200
•	NULL	NULL	NULL

Develop a SQL query to list the orderno and number of items in each order.

```
SELECT Orderno, SUM(qty) FROM Order_item GROUP BY
```

	Orderno	SUM(qty)
▶	O01	4
	O02	3
	O03	2
	O04	5
	O05	5
	O06	6
	O07	5

Orderno;

Develop a SQL query to list the details of items that are present in 25% of the orders.

```
SELECT * FROM Item WHERE Itemno IN (SELECT Itemno FROM
Order_item GROUP BY
```

```
Itemno HAVING count(Itemno) >= (SELECT (count(*)/4)
FROM Cust_Order));
```

	Itemno	Item_name	unit_price
▶	I002	FAN	100
	I003	TABLE	100
	I004	COLE	50
	I005	CHAIR	200
•	NULL	NULL	NULL

Develop an update statement to update the value of Ord_amt.

```
UPDATE Cust_Order SET Ord_amt = (SELECT sum(O.qty *
I.unit_price)
FROM Order_item as O, Item as I WHERE Cust_Order.Orderno =
O.Orderno AND O.Itemno= I.Itemno);
```

	Itemno	Item_name	unit_price
▶	I001	Biscuit	2000
	I005	CHAIR	200
•	NULL	NULL	NULL

Create a view that keeps track of detail of each customer and number of Order placed.

```
CREATE VIEW CUSTOMER_DETAILS AS SELECT
```

```
    C.Customerno,C.Cname,count(Orderno) as Order_C from Customer as
    C,Cust_Order as O
```

```
        where C.Customerno=O.Customerno group by
```

```
C.Customerno,C.Cname;
```

```
select * from CUSTOMER_DETAILS;
```

Question 4.) Create the relation employee(eid integer, managerid integer) where managerid is the foreign key referencing to the employee. Develop DDL to implement above schema enforcing primary key as eid.

```
create database office;
```

```
create table employee(
```

```
    eid integer primary key ,
```

```
    managerid integer ,
```

```
    foreign key employee(managerid) references employee (eid) on delete
    cascade
```

```
);
```

Insert some data to the employee and run a delete query.

```
insert into employee values (1,1);
```

```
insert into employee values (2,1);
```

```
insert into employee values (3,2);
```

```
insert into employee values (4,3);
```

```
insert into employee values (5,5);
```

```
insert into employee values (6,5);
```

```
insert into employee values (7,5);
```

```
insert into employee values (8,5);
```

```
insert into employee values (9,6);
```

```
insert into employee values (10,7);
```

```
select * from employee;
```

Now use on delete cascade and again run delete query.

```
delete from employee where eid = 1;
```

```
select * from employee;
```